# Gooey Documentation

## Version 1.0.2

*Generated on 2025-11-01 19:00:24*

# Table of Contents

API: gooey_switch.h

API: gooey_tabs.h

API: gooey_textbox.h

API: freetype.h

API: ftbdf.h

API: ftcache.h

API: ftcolor.h

API: ftdriver.h

API: ftglyph.h

API: ftimage.h

API: ftincrem.h

API: ftlcdfil.h

API: ftmm.h

API: ftmodapi.h

API: ftoutln.h

API: ftrender.h

API: ftsnames.h

API: ftstroke.h

API: ftsystem.h

API: fttrigon.h

API: fttypes.h

API: ftwinfnt.h

API: otsvg.h

API: t1tables.h

API: tttables.h

API: integer-types.h

API: autohint.h

API: cffotypes.h

API: cfftypes.h

API: ftdebug.h

API: ftdrv.h

API: ftgloadr.h

API: fthash.h

API: ftmmtypes.h

API: ftobjs.h

API: ftrfork.h

API: ftserv.h

API: ftstream.h

API: ftvalid.h

API: psaux.h

API: pshints.h

API: sfnt.h

API: t1types.h

API: tttypes.h

API: wofftypes.h

API: svttcmap.h

API: afblue.h

API: afcjk.h

API: afglobal.h

API: afhints.h

API: aflatin.h

API: afloader.h

API: afmodule.h

API: aftypes.h

API: ft-hb.h

API: bdf.h

API: ftccache.h

API: ftcglyph.h

API: ftcimage.h

API: ftcmanag.h

API: ftcmru.h

API: ftcsbits.h

API: cffcmap.h

API: cffparse.h

API: cidload.h

API: cidobjs.h

API: cidparse.h

API: gxvcommn.h

API: gxvfeat.h

API: gxvmort.h

API: ftzopen.h

API: otvcommn.h

API: pfrcmap.h

API: pfrobjs.h

API: pfrtypes.h

API: afmparse.h

API: psarrst.h

API: psfixed.h

API: psfont.h

API: psglue.h

API: psread.h

API: psstack.h

API: pstypes.h

API: t1cmap.h

API: pshalgo.h

API: pshglob.h

API: pshrec.h

API: ftmisc.h

API: ftsdf.h

API: ftsdfcommon.h

API: ftsdfrend.h

API: ttcmap.h

API: svgtypes.h

API: ttgxvar.h

API: ttinterp.h

API: ttobjs.h

API: t1load.h

API: t1objs.h

API: t1parse.h

API: t42objs.h

API: t42parse.h

API: t42types.h

API: winfnt.h

API: glps_audio_stream.h

API: glps_window_manager.h

API: glps_common.h

API: glps_wayland.h

API: pico_logger.h

API: nfd.h

# API Reference

## gooey.h

### *Functions*

### *Gooey_Init*

**Description:** Initializes the Gooey system with the selected backend. Currently, only the GLPS (OpenGL) backend is supported.
**Signature:**
int Gooey_Init()

# gooey_common.h

## *Data Types*

### *typedef WIDGET_TYPE*

**Description:** Internal definitions for the Gooey GUI library.
**Definition:**
typedef enum { WIDGET_LABEL, WIDGET_SLIDER, WIDGET_RADIOBUTTON, WIDGET_CHECKBOX, WIDGET_BUTTON, WIDGET_TEXTBOX, WIDGET_DROPDOWN, WIDGET_CANVAS, WIDGET_LAYOUT, WIDGET_PLOT, WIDGET_DROP_SURFACE, WIDGET_IMAGE, WIDGET_LIST, WIDGET_PROGRESSBAR, WIDGET_METER, WIDGET_CONTAINER, WIDGET_SWITCH, WIDGET_WEBVIEW, WIDGET_CTXMENU, WIDGET_NODE_EDITOR, WIDGET_TABS } WIDGET_TYPE;

# gooey_timers.h

## *Functions*

### **GooeyTimer_Create*

**Description:** Creates a new timer object.
**Signature:**
GooeyTimer *GooeyTimer_Create()

### *GooeyTimer_Destroy*

**Description:** Destroys the timer and frees its resources.
**Signature:**
void GooeyTimer_Destroy(GooeyTimer * timer)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| timer | GooeyTimer * | Pointer to the timer to destroy. |

### *GooeyTimer_Stop*

**Description:** Stops the timer if it is running.
**Signature:**
void GooeyTimer_Stop(GooeyTimer * timer)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| timer | GooeyTimer * | Pointer to the timer to stop. |

# gooey_widget.h

## *Functions*

### *GooeyWidget_MakeVisible*

**Description:** Sets the visibility state of a widget.
**Signature:**
void GooeyWidget_MakeVisible(void* widget, bool state)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| widget | void* | Pointer to the widget. |
| state | bool | True to make visible, false to hide. |

### *GooeyWidget_MoveTo*

**Description:** Moves the widget to a new position.
**Signature:**
void GooeyWidget_MoveTo(void* widget, int x, int y)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| widget | void* | Pointer to the widget. |
| x | int | New x-coordinate. |
| y | int | New y-coordinate. |

### *GooeyWidget_Resize*

**Description:** Resizes the widget.
**Signature:**
void GooeyWidget_Resize(void* widget, int w, int h)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| widget | void* | Pointer to the widget. |
| w | int | New width. |
| h | int | New height. |

# gooey_window.h

## *Functions*

### **GooeyTheme_LoadFromFile*

**Description:** Loads a theme from the specified file path. This function reads a theme file and returns a `GooeyTheme` object that can be applied to Gooey windows.
**Signature:**
GooeyTheme *GooeyTheme_LoadFromFile(const char * theme_path)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| theme_path | const char * | The path to the theme file. |

### **GooeyWindow_Create*

**Description:** Creates a new Gooey window with the specified title, width, and height. This function initializes and returns a new window. The window can be customized with various properties such as its visibility and theme.
**Signature:**
GooeyWindow *GooeyWindow_Create(const char * title, int width, int height, bool visibility)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| title | const char * | The title of the window. |
| width | int | The width of the window. |
| height | int | The height of the window. |
| visibility | bool | The initial visibility of the window (visible or hidden). |

### *GooeyWindow_Cleanup*

**Description:** Cleans up the resources associated with Gooey windows. This function deallocates memory and resources for the specified windows and their associated widgets.
**Signature:**
void GooeyWindow_Cleanup(int num_windows, GooeyWindow * first_win)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| num_windows | int | The number of windows to clean up. |
| first_win | GooeyWindow * | The first window to clean up. |

### *GooeyWindow_MakeResizable*

**Description:** Sets the resizable property of a window. This function determines whether the user can resize the window. If the window is made resizable, the user can adjust its size at runtime.
**Signature:**

void GooeyWindow_MakeResizable(GooeyWindow * msgBoxWindow, bool is_resizable)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| msgBoxWindow | GooeyWindow * | The window to modify. |
| is_resizable | bool | `true` to make the window resizable, `false` to make it fixed-size. |

### *GooeyWindow_MakeVisible*

**Description:** Sets the visibility of a Gooey window. This function controls whether the window is visible or hidden.
**Signature:**
void GooeyWindow_MakeVisible(GooeyWindow * win, bool visibility)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| win | GooeyWindow * | The window whose visibility is to be set. |
| visibility | bool | `true` to make the window visible, `false` to hide it. |

### *GooeyWindow_RegisterWidget*

**Description:** Registers a widget with the specified window. This function adds a widget (such as a button, label, or slider) to a window, so it can be rendered and interact with user input.
**Signature:**
void GooeyWindow_RegisterWidget(GooeyWindow * win, void * widget)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| win | GooeyWindow * | The window to register the widget with. |
| widget | void * | The widget to register. |

### *GooeyWindow_RequestRedraw*

**Description:** Requests a redraw of the window. This function triggers a redraw of the specified window, typically in response to changes that affect its appearance (e.g., updates to the window's content or layout).
**Signature:**
void GooeyWindow_RequestRedraw(GooeyWindow * win)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| win | GooeyWindow * | The window that requires a redraw. |

### *GooeyWindow_Run*

**Description:** Runs the Gooey window's event loop. This function starts the main event loop for the specified window, where user input and window events are processed until the window is closed.
**Signature:**

void GooeyWindow_Run(int num_windows, GooeyWindow * first_win)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| num_windows | int | The number of windows to handle in the event loop. |
| first_win | GooeyWindow * | The first window to run in the event loop. |

## *GooeyWindow_SetTheme*

**Description:** Sets the theme for the specified Gooey window. This function applies the given theme to the window, updating its appearance.
**Signature:**
void GooeyWindow_SetTheme(GooeyWindow * win, GooeyTheme * theme)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| win | GooeyWindow * | The Gooey window to set the theme for. |
| theme | GooeyTheme * | The theme to apply to the window. |

# gooey_signals.h

## *Functions*

### *GooeySignal_Create*

**Signature:**
GooeySignal GooeySignal_Create()

### *GooeySignal_Emit*

**Description:** Emits a signal. This function triggers the signal, invoking all linked callbacks in the order they were added. Each callback receives the specified data.
**Signature:**
void GooeySignal_Emit(GooeySignal * signal, void * data)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| signal | GooeySignal * | A pointer to the signal to emit. |
| data | void * | A void pointer to the data passed to all linked callbacks. |

### *GooeySignal_Link*

**Description:** Links a callback function to a signal. This function connects a signal to a specific callback function. When the signal is emitted, the callback function will be executed. The callback is linked to the signal and can be triggered when the signal is emitted.
**Signature:**
void GooeySignal_Link(GooeySignal * signal, GooeySignal_CallbackFunction callback, void * context)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| signal | GooeySignal * | A pointer to the signal to which the callback is linked. |
| callback | GooeySignal_CallbackFunction | The callback function to execute when the signal is emitted. |
| context | void * | A user-defined context pointer passed to the callback. |

### *GooeySignal_UnLinkAll*

**Description:** Unlinks all callbacks from a signal. Removes all callback functions linked to the specified signal, effectively clearing its event listeners.
**Signature:**
void GooeySignal_UnLinkAll(GooeySignal * signal)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| signal | GooeySignal * | A pointer to the signal from which callbacks should be unlinked. |

## *Data Types*

## *typedef GooeySignal*

**Definition:**
typedef struct { GooeySignal_Slot *slots; /**< List of slots (callbacks) associated with the signal */ } GooeySignal;

## *struct GooeySignal_Slot*

**Description:** A slot representing a linked callback in the signal system. A slot contains a callback function, its associated context, and a link to the next slot in a potentially linked list of callbacks.
**Members:**

| Type | Name | Description |
|---|---|---|
| GooeySignal_CallbackFunction | callback | |
| void | *context | |
| struct GooeySignal_Slot | *next | |

**Definition:**
typedef struct GooeySignal_Slot { GooeySignal_CallbackFunction callback; /**< The callback function to be executed */ void *context; /**< The user-defined context to be passed to the callback */ struct GooeySignal_Slot *next; /**< Pointer to the next slot in the list */ } GooeySignal_Slot;

## *struct GooeySignal_Slot*

**Members:**

| Type | Name | Description |
|---|---|---|
| GooeySignal_Slot | *slots | |

**Definition:**
struct GooeySignal_Slot *next; /**< Pointer to the next slot in the list */

# gooey_theme.h

## *Functions*

### *parser_load_theme_from_file*

**Description:** Structure representing a theme for the Gooey UI. The colors in the theme are stored as unsigned long values representing the color codes (typically in hexadecimal format).
**Signature:**
GooeyTheme parser_load_theme_from_file(const char * filePath, bool * is_theme_loaded)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| filePath | const char * | |
| is_theme_loaded | bool * | |

## *Data Types*

### *typedef base*

**Description:** Structure representing a theme for the Gooey UI. The colors in the theme are stored as unsigned long values representing the color codes (typically in hexadecimal format).
**Definition:**
typedef struct { unsigned long base; /**< Base window background color */ unsigned long neutral; /**< Neutral color for text */ unsigned long widget_base; /**< Base widget color */ unsigned long primary; /**< Primary color */ unsigned long danger; /**< Danger color */ unsigned long info; /**< Info color */ unsigned long success; /**< Success color */ } GooeyTheme;

# gooey_button.h

## *Functions*

### *GooeyButton_SetHighlight*

**Description:** Highlights or unhighlights a button. This function visually indicates whether a button is highlighted, which can be used to show focus or selection.
**Signature:**
void GooeyButton_SetHighlight(GooeyButton * button, bool is_highlighted)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| button | GooeyButton * | A pointer to the button to modify. |
| is_highlighted | bool | `true` to highlight the button; `false` to remove the highlight. |

### *GooeyButton_SetText*

**Description:** Sets the text of the button. This function updates the label text displayed on the button.
**Signature:**
void GooeyButton_SetText(GooeyButton * button, const char * text)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| button | GooeyButton * | The button to set the text for. |
| text | const char * | The new text to display on the button. |

# gooey_canvas.h

## *Functions*

### *GooeyCanvas_DrawLine*

**Description:** Draws a rectangle onto the user-defined canvas. The rectangle can be either filled with a solid color or outlined.
**Signature:**
void GooeyCanvas_DrawLine(GooeyCanvas * canvas, int x1, int y1, int x2, int y2, unsigned long color_hex)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| canvas | GooeyCanvas * | The user-defined canvas. |
| x1 | int | |
| y1 | int | |
| x2 | int | |
| y2 | int | |
| color_hex | unsigned long | The color of the rectangle in hexadecimal format. |

### *GooeyCanvas_SetForeground*

**Description:** Draws an arc onto the user-defined canvas. The arc is drawn within the specified bounding box and between the given angles.
**Signature:**
void GooeyCanvas_SetForeground(GooeyCanvas * canvas, unsigned long color_hex)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| canvas | GooeyCanvas * | The user-defined canvas. |
| color_hex | unsigned long | |

# gooey_container.h

## *Functions*

### *GooeyContainer_AddWidget*

**Description:** Adds a widget to a specific container within the window.
**Signature:**
void GooeyContainer_AddWidget(GooeyWindow* window, GooeyContainers* container, size_t container_id, void * widget)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| window | GooeyWindow* | The target GooeyWindow. |
| container | GooeyContainers* | The container to add the widget into. |
| container_id | size_t | The ID of the target sub-container. |
| widget | void * | Pointer to the widget to add. |

### *GooeyContainer_Create*

**Description:** Creates a new container with the given position and dimensions.
**Signature:**
GooeyContainers* GooeyContainer_Create(int x, int y, int width, int height)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| x | int | X-coordinate of the container. |
| y | int | Y-coordinate of the container. |
| width | int | Width of the container. |
| height | int | Height of the container. |

### *GooeyContainer_InsertContainer*

**Description:** Inserts a sub-container into the given container.
**Signature:**
void GooeyContainer_InsertContainer(GooeyContainers * container)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| container | GooeyContainers * | Pointer to the parent GooeyContainers. |

### *GooeyContainer_SetActiveContainer*

**Description:** Sets which container is currently active/visible.
**Signature:**
void GooeyContainer_SetActiveContainer(GooeyContainers * container, size_t container_id)

**Parameters:**

| Name | Type | Description |
| --- | --- | --- |
| container | GooeyContainers * | Pointer to the container set. |
| container_id | size_t | ID of the container to activate. |

# gooey_dropdown.h

## *Functions*

### *GooeyDropdown_Update*

**Description:** Header file for the GooeyDropdown module. Provides functions to create, handle, and render dropdown menus within a GooeyWindow.
**Signature:**
void GooeyDropdown_Update(GooeyDropdown * dropdown, const char ** new_options, int new_num_options)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| dropdown | GooeyDropdown * | |
| new_options | const char ** | |
| new_num_options | int | |

# gooey_drop_surface.h

## *Functions*

### *GooeyDropSurface_Clear*

**Description:** Header file for the GooeyDropSurface module.
**Signature:**
void GooeyDropSurface_Clear(GooeyDropSurface * drop_surface)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| drop_surface | GooeyDropSurface * | |

# gooey_image.h

## *Functions*

### *GooeyImage_Damage*

**Description:** Marks the image as needing redrawing (damaged).
**Signature:**
void GooeyImage_Damage(GooeyImage * image)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| image | GooeyImage * | The image widget to damage. |

### *GooeyImage_SetImage*

**Description:** Image handling functions for the Gooey GUI library. This file contains functions for adding and drawing images in a Gooey window.
**Signature:**
void GooeyImage_SetImage(GooeyImage * image, const char * image_path)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| image | GooeyImage * | |
| image_path | const char * | |

# gooey_label.h

## *Functions*

### **GooeyLabel_Create*

**Description:** Header file for the GooeyLabel module. Provides functions for creating, modifying, and rendering text labels within a GooeyWindow.
**Signature:**
GooeyLabel *GooeyLabel_Create(const char * text, float font_size, int x, int y)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| text | const char * | |
| font_size | float | |
| x | int | |
| y | int | |

### *GooeyLabel_SetColor*

**Description:** Sets the text color of a label. Changes the text color of a label to the specified color.
**Signature:**
void GooeyLabel_SetColor(GooeyLabel * label, unsigned long color)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| label | GooeyLabel * | A pointer to the label whose color is to be changed. |
| color | unsigned long | An unsigned long representing the color (e.g., 0xFF0000 for red). |

### *GooeyLabel_SetText*

**Description:** Sets the text of an existing label. Updates the text displayed by a given GooeyLabel.
**Signature:**
void GooeyLabel_SetText(GooeyLabel * label, const char * text)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| label | GooeyLabel * | The label to update. |
| text | const char * | The new text to display on the label. |

# gooey_layout.h

## *Functions*

### *GooeyLayout_AddChild*

**Description:** Header file for the GooeyLayout module. Provides functionality for managing and arranging widgets within a structured layout inside a GooeyWindow.
**Signature:**
void GooeyLayout_AddChild(GooeyWindow * window, GooeyLayout * layout, void * widget)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| window | GooeyWindow * | |
| layout | GooeyLayout * | |
| widget | void * | |

### *GooeyLayout_Build*

**Description:** Finalizes and builds the layout. This function applies the layout rules and arranges all child widgets accordingly.
**Signature:**
void GooeyLayout_Build(GooeyLayout * layout)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| layout | GooeyLayout * | The layout to build. |

# gooey_list.h

## *Functions*

### *GooeyList_AddItem*

**Description:** Adds an item to the specified list widget.
**Signature:**
void GooeyList_AddItem(GooeyList * list, const char * title, const char * description)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| list | GooeyList * | The list widget to which the item will be added. |
| title | const char * | The title of the list item. |
| description | const char * | The description of the list item. |

### *GooeyList_ClearItems*

**Description:** Clears all items from the specified list widget. Removes all list items from the provided list widget.
**Signature:**
void GooeyList_ClearItems(GooeyList * list)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| list | GooeyList * | The list widget to be cleared. |

### *GooeyList_ShowSeparator*

**Description:** Toggles the visibility of the separator in a list widget. Enables or disables the visual separator between list items.
**Signature:**
void GooeyList_ShowSeparator(GooeyList * list, bool state)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| list | GooeyList * | The list widget. |
| state | bool | `true` to show the separator, `false` to hide it. |

### *GooeyList_UpdateItem*

**Description:** Updates the title and description of a specific item in the list.
**Signature:**
void GooeyList_UpdateItem(GooeyList * list, size_t item_index, const char * title, const char * description)
**Parameters:**

| Name | Type | Description |
|---|---|---|

| | | |
|---|---|---|
| list | GooeyList * | The list widget. |
| item_index | size_t | The index of the item to update. |
| title | const char * | The new title for the list item. |
| description | const char * | The new description for the list item. |

# gooey_menu.h

## *Functions*

### **GooeyMenu_AddChild*

**Description:** Adds a child menu item to the window's menu. Creates a submenu or a category inside the main menu.
**Signature:**
GooeyMenuChild *GooeyMenu_AddChild(GooeyWindow * win, char * title, void* user_data)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| win | GooeyWindow * | The window to which the menu child will be added. |
| title | char * | The title of the menu child. |
| user_data | void* | |

### **GooeyMenu_Set*

**Description:** Header file for the Gooey menu system. Provides functions to create and manage menus and menu items within a Gooey window.
**Signature:**
GooeyMenu *GooeyMenu_Set(GooeyWindow * win)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| win | GooeyWindow * | |

# gooey_messagebox.h

## *Functions*

### *GooeyMessageBox_Show*

**Description:** Displays the specified message box window. This function makes the message box visible on the screen. The message box must have been created using `GooeyMessageBox_Create`.
**Signature:**
void GooeyMessageBox_Show(GooeyWindow * msgBoxWindow)
**Parameters:**

| Name | Type | Description |
| --- | --- | --- |
| msgBoxWindow | GooeyWindow * | A pointer to the message box window to display. |

# gooey_meter.h

## *Functions*

### **GooeyMeter_Create*

**Description:** Creates a meter widget. Initializes a new GooeyMeter widget at the specified position and size, with an initial value, a label, and an optional icon.
**Signature:**
GooeyMeter *GooeyMeter_Create(int x, int y, int width, int height, long initial_value, const char * label, const char * icon_path)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| x | int | The x-coordinate of the meter's position. |
| y | int | The y-coordinate of the meter's position. |
| width | int | The width of the meter. |
| height | int | The height of the meter. |
| initial_value | long | The initial value displayed by the meter. |
| label | const char * | A text label to display with the meter. |
| icon_path | const char * | Path to an icon image to display with the meter (can be NULL). |

### *GooeyMeter_Update*

**Description:** Updates the value displayed by the meter.
**Signature:**
void GooeyMeter_Update(GooeyMeter * meter, long new_value)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| meter | GooeyMeter * | The GooeyMeter instance to update. |
| new_value | long | The new value to set. |

# gooey_plot.h

## *Functions*

### **GooeyPlot_Create*

**Description:** Creates a plot widget. Adds a plot widget of the specified type to the given position and size, using the provided plot data.
**Signature:**
GooeyPlot *GooeyPlot_Create(GOOEY_PLOT_TYPE plot_type, GooeyPlotData * data, int x, int y, int width, int height)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| plot_type | GOOEY_PLOT_TYPE | The type of plot to be created (e.g., LINE, BAR). |
| data | GooeyPlotData * | Pointer to the plot data structure. |
| x | int | The x-coordinate of the plot's position. |
| y | int | The y-coordinate of the plot's position. |
| width | int | The width of the plot widget. |
| height | int | The height of the plot widget. |

### *GooeyPlot_Update*

**Description:** Updates an existing plot with new data. Updates the content of the given plot widget while maintaining its configuration and type.
**Signature:**
void GooeyPlot_Update(GooeyPlot * plot, GooeyPlotData * new_data)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| plot | GooeyPlot * | Pointer to the plot widget to update. |
| new_data | GooeyPlotData * | Pointer to the new data to update the plot with. |

# gooey_progressbar.h

## *Functions*

### **GooeyProgressBar_Create*

**Description:** Creates a progress bar widget. Initializes a new progress bar at the specified position and size, with an initial value.
**Signature:**
GooeyProgressBar *GooeyProgressBar_Create(int x, int y, int width, int height, long initial_value)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| x | int | The x-coordinate of the progress bar. |
| y | int | The y-coordinate of the progress bar. |
| width | int | The width of the progress bar. |
| height | int | The height of the progress bar. |
| initial_value | long | The initial progress value. |

### *GooeyProgressBar_Update*

**Description:** Updates the value of an existing progress bar. Changes the displayed progress to the new value.
**Signature:**
void GooeyProgressBar_Update(GooeyProgressBar * progressbar, long new_value)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| progressbar | GooeyProgressBar * | Pointer to the progress bar to update. |
| new_value | long | The new progress value. |

# gooey_radiobutton.h

## *Functions*

### **GooeyRadioButtonGroup_Create*

**Description:** Creates a new radio button group. Allows grouping of multiple radio buttons, ensuring single selection.
**Signature:**
GooeyRadioButtonGroup *GooeyRadioButtonGroup_Create()

### *GooeyRadioButtonGroup_Draw*

**Description:** Draws all radio buttons in the group on the window. Renders the visual state of each radio button, reflecting selection.
**Signature:**
void GooeyRadioButtonGroup_Draw(GooeyWindow * win)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| win | GooeyWindow * | The window on which to draw the group. |

### *GooeyRadioButtonGroup_HandleClick*

**Description:** Handles click events for a group of radio buttons. Ensures that only one radio button in the group is selected at a time.
**Signature:**
bool GooeyRadioButtonGroup_HandleClick(GooeyWindow * win, int x, int y)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| win | GooeyWindow * | The window containing the radio button group. |
| x | int | The x-coordinate of the click event. |
| y | int | The y-coordinate of the click event. |

### *GooeyRadioButton_HandleClick*

**Description:** Handles a click event on radio buttons within a window. Checks if a radio button was clicked and triggers the appropriate callback.
**Signature:**
bool GooeyRadioButton_HandleClick(GooeyWindow * win, int x, int y)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| win | GooeyWindow * | The window containing the radio buttons. |
| x | int | The x-coordinate of the click event. |
| y | int | The y-coordinate of the click event. |

# gooey_slider.h

## *Functions*

### *GooeySlider_GetValue*

**Description:** Retrieves the current value of the slider. Returns the current slider value, guaranteed to be within the configured range.
**Signature:**
long GooeySlider_GetValue(GooeySlider * slider)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| slider | GooeySlider * | The slider instance. |

### *GooeySlider_SetValue*

**Description:** Sets the slider's value. Updates the slider's position and value to the specified value within its range.
**Signature:**
void GooeySlider_SetValue(GooeySlider * slider, long value)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| slider | GooeySlider * | The slider instance. |
| value | long | The new value to set. |

# gooey_switch.h

## *Functions*

### *GooeySwitch_GetState*

**Description:** Retrieves the current value of the slider. Returns the current slider value, guaranteed to be within the configured range.
**Signature:**
bool GooeySwitch_GetState(GooeySwitch * gswitch)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| gswitch | GooeySwitch * | |

### *GooeySwitch_Toggle*

**Description:** Sets the slider's value. Updates the slider's position and value to the specified value within its range.
**Signature:**
void GooeySwitch_Toggle(GooeySwitch * gswitch)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| gswitch | GooeySwitch * | |

# gooey_tabs.h

## *Functions*

### **GooeyTabs_Create*

**Description:** Creates a tab widget. Creates a GooeyTabs widget at the specified position with given dimensions. If `is_sidebar` is true, the tabs behave as a sidebar.
**Signature:**
GooeyTabs *GooeyTabs_Create(int x, int y, int width, int height, bool is_sidebar)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| x | int | The x-coordinate of the tabs widget. |
| y | int | The y-coordinate of the tabs widget. |
| width | int | The width of the tabs widget. |
| height | int | The height of the tabs widget. |
| is_sidebar | bool | Whether the tabs should behave as a sidebar. |

### *GooeyTabs_AddWidget*

**Description:** Adds a widget to a specified tab. Associates a widget with a tab identified by tab_id.
**Signature:**
void GooeyTabs_AddWidget(GooeyWindow * window, GooeyTabs * tabs, size_t tab_id, void * widget)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| window | GooeyWindow * | The parent window containing the tabs. |
| tabs | GooeyTabs * | The tabs widget. |
| tab_id | size_t | The index of the tab to add the widget to. |
| widget | void * | Pointer to the widget to add. |

### *GooeyTabs_InsertTab*

**Description:** Inserts a new tab with the specified name. Adds a new tab to the tab widget.
**Signature:**
void GooeyTabs_InsertTab(GooeyTabs * tab_widget, char * tab_name)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| tab_widget | GooeyTabs * | The tab widget to add a tab to. |
| tab_name | char * | The name/title of the new tab. |

### *GooeyTabs_SetActiveTab*

**Description:** Sets the active tab. Switches the active tab to the one specified by tab_id.
**Signature:**
void GooeyTabs_SetActiveTab(GooeyTabs * tabs, size_t tab_id)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| tabs | GooeyTabs * | The tabs widget. |
| tab_id | size_t | The index of the tab to activate. |

### *GooeyTabs_Sidebar_Close*

**Description:** Closes the sidebar tabs (if tabs are in sidebar mode).
**Signature:**
void GooeyTabs_Sidebar_Close(GooeyTabs * tabs_widget)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| tabs_widget | GooeyTabs * | The tabs widget. |

### *GooeyTabs_Sidebar_Open*

**Description:** Opens the sidebar tabs (if tabs are in sidebar mode).
**Signature:**
void GooeyTabs_Sidebar_Open(GooeyTabs * tabs_widget)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| tabs_widget | GooeyTabs * | The tabs widget. |

# gooey_textbox.h

## *Functions*

### **GooeyTextbox_GetText*

**Description:** Gets the current text from the textbox.
**Signature:**
const char *GooeyTextbox_GetText(GooeyTextbox * textbox)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| textbox | GooeyTextbox * | The textbox to retrieve text from. |

### *GooeyTextbox_Draw*

**Description:** Draws the textbox on the specified window. Renders the textbox UI, reflecting its current state and contents.
**Signature:**
void GooeyTextbox_Draw(GooeyWindow * win)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| win | GooeyWindow * | The window to draw the textbox on. |

### *GooeyTextbox_HandleClick*

**Description:** Handles click events on the textbox. Detects clicks on the textbox area and activates the textbox for input if clicked.
**Signature:**
bool GooeyTextbox_HandleClick(GooeyWindow * win, int x, int y)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| win | GooeyWindow * | The window containing the textbox. |
| x | int | The x-coordinate of the click. |
| y | int | The y-coordinate of the click. |

### *GooeyTextbox_HandleKeyPress*

**Description:** Processes key press events for the textbox. Updates the textbox content based on keyboard input.
**Signature:**
void GooeyTextbox_HandleKeyPress(GooeyWindow * win, void * event)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| win | GooeyWindow * | The window containing the textbox. |

| | | |
|---|---|---|
| event | void * | Pointer to the key event to handle. |

### *GooeyTextbox_SetText*

**Description:** Sets the text content of the textbox.
**Signature:**
void GooeyTextbox_SetText(GooeyTextbox * textbox, const char * text)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| textbox | GooeyTextbox * | The textbox to update. |
| text | const char * | The new text to set. |

# freetype.h

## *Data Types*

### *struct FT_CharMapRec_*

**Definition:**
typedef struct FT_CharMapRec_* FT_CharMap;

### *struct FT_CharMapRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Face | face | |
| FT_Encoding | encoding | |
| FT_UShort | platform_id | |
| FT_UShort | encoding_id | |

**Definition:**
typedef struct FT_CharMapRec_ { FT_Face face; FT_Encoding encoding; FT_UShort platform_id; FT_UShort encoding_id; } FT_CharMapRec;

### *struct FT_DriverRec_*

**Definition:**
typedef struct FT_DriverRec_* FT_Driver;

### *enum FT_Encoding_*

**Definition:**
typedef enum FT_Encoding_ { FT_ENC_TAG( FT_ENCODING_NONE, 0, 0, 0, 0 ), FT_ENC_TAG( FT_ENCODING_MS_SYMBOL, 's', 'y', 'm', 'b' ), FT_ENC_TAG( FT_ENCODING_UNICODE, 'u', 'n', 'i', 'c' ), FT_ENC_TAG( FT_ENCODING_SJIS, 's', 'j', 'i', 's' ), FT_ENC_TAG( FT_ENCODING_PRC, 'g', 'b', ' ', ' ' ), FT_ENC_TAG( FT_ENCODING_BIG5, 'b', 'i', 'g', '5' ), FT_ENC_TAG( FT_ENCODING_WANSUNG, 'w', 'a', 'n', 's' ), FT_ENC_TAG( FT_ENCODING_JOHAB, 'j', 'o', 'h', 'a' ), /* for backward compatibility */ FT_ENCODING_GB2312 = FT_ENCODING_PRC, FT_ENCODING_MS_SJIS = FT_ENCODING_SJIS, FT_ENCODING_MS_GB2312 = FT_ENCODING_PRC, FT_ENCODING_MS_BIG5 = FT_ENCODING_BIG5, FT_ENCODING_MS_WANSUNG = FT_ENCODING_WANSUNG, FT_ENCODING_MS_JOHAB = FT_ENCODING_JOHAB, FT_ENC_TAG( FT_ENCODING_ADOBE_STANDARD, 'A', 'D', 'O', 'B' ), FT_ENC_TAG( FT_ENCODING_ADOBE_EXPERT, 'A', 'D', 'B', 'E' ), FT_ENC_TAG( FT_ENCODING_ADOBE_CUSTOM, 'A', 'D', 'B', 'C' ), FT_ENC_TAG( FT_ENCODING_ADOBE_LATIN_1, 'l', 'a', 't', '1' ), FT_ENC_TAG( FT_ENCODING_OLD_LATIN_2, 'l', 'a', 't', '2' ), FT_ENC_TAG( FT_ENCODING_APPLE_ROMAN, 'a', 'r', 'm', 'n' ) } FT_Encoding;

### *typedef FT_Face*

**Definition:**
* Similarly, creation and destruction of `FT_Face` with the same * @FT_Library object can only be done from one thread at a time. On the * other hand, functions like @FT_Load_Glyph and its siblings are * thread-safe and do not need the lock to be held as long as the same * `FT_Face` object is not used from multiple threads at the same time. * * @also: * See @FT_FaceRec for the publicly accessible fields of a given face * object. */ typedef struct FT_FaceRec_* FT_Face;

## *struct FT_FaceRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Long | num_faces | |
| FT_Long | face_index | |
| FT_Long | face_flags | |
| FT_Long | style_flags | |
| FT_Long | num_glyphs | |
| FT_String* | family_name | |
| FT_String* | style_name | |
| FT_Int | num_fixed_sizes | |
| FT_Bitmap_Size* | available_sizes | |
| FT_Int | num_charmaps | |
| FT_CharMap* | charmaps | |
| FT_Generic | generic | |
| FT_BBox | bbox | |
| FT_UShort | units_per_EM | |
| FT_Short | ascender | |
| FT_Short | descender | |
| FT_Short | height | |
| FT_Short | max_advance_width | |
| FT_Short | max_advance_height | |
| FT_Short | underline_position | |
| FT_Short | underline_thickness | |
| FT_GlyphSlot | glyph | |
| FT_Size | size | |
| FT_CharMap | charmap | |
| FT_Driver | driver | |
| FT_Memory | memory | |
| FT_Stream | stream | |

| | | |
|---|---|---|
| FT_ListRec | sizes_list | |
| FT_Generic | autohint | |
| void* | extensions | |
| FT_Face_Internal | internal | |

**Definition:**
typedef struct FT_FaceRec_ { FT_Long num_faces; FT_Long face_index; FT_Long face_flags; FT_Long style_flags; FT_Long num_glyphs; FT_String* family_name; FT_String* style_name; FT_Int num_fixed_sizes; FT_Bitmap_Size* available_sizes; FT_Int num_charmaps; FT_CharMap* charmaps; FT_Generic generic; /* The following member variables (down to `underline_thickness`) */ /* are only relevant to scalable outlines; cf. @FT_Bitmap_Size */ /* for bitmap fonts. */ FT_BBox bbox; FT_UShort units_per_EM; FT_Short ascender; FT_Short descender; FT_Short height; FT_Short max_advance_width; FT_Short max_advance_height; FT_Short underline_position; FT_Short underline_thickness; FT_GlyphSlot glyph; FT_Size size; FT_CharMap charmap; /* private fields, internal to FreeType */ FT_Driver driver; FT_Memory memory; FT_Stream stream; FT_ListRec sizes_list; FT_Generic autohint; /* face-specific auto-hinter data */ void* extensions; /* unused */ FT_Face_Internal internal; } FT_FaceRec;

## *typedef FT_Face_Internal*

**Definition:**
* An opaque handle to an `FT_Face_InternalRec` structure that models the * private data of a given @FT_Face object. * * This structure might change between releases of FreeType~2 and is not * generally available to client applications. */ typedef struct FT_Face_InternalRec_* FT_Face_Internal;

## *struct FT_GlyphSlotRec_*

**Definition:**
typedef struct FT_GlyphSlotRec_* FT_GlyphSlot;

## *struct FT_GlyphSlotRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Library | library | |
| FT_Face | face | |
| FT_GlyphSlot | next | |
| FT_UInt | glyph_index | |
| FT_Generic | generic | |
| FT_Glyph_Metrics | metrics | |
| FT_Fixed | linearHoriAdvance | |
| FT_Fixed | linearVertAdvance | |
| FT_Vector | advance | |
| FT_Glyph_Format | format | |

| | | |
|---|---|---|
| FT_Bitmap | bitmap | |
| FT_Int | bitmap_left | |
| FT_Int | bitmap_top | |
| FT_Outline | outline | |
| FT_UInt | num_subglyphs | |
| FT_SubGlyph | subglyphs | |
| void* | control_data | |
| long | control_len | |
| FT_Pos | lsb_delta | |
| FT_Pos | rsb_delta | |
| void* | other | |
| FT_Slot_Internal | internal | |

**Definition:**
typedef struct FT_GlyphSlotRec_ { FT_Library library; FT_Face face; FT_GlyphSlot next; FT_UInt glyph_index; /* new in 2.10; was reserved previously */ FT_Generic generic; FT_Glyph_Metrics metrics; FT_Fixed linearHoriAdvance; FT_Fixed linearVertAdvance; FT_Vector advance; FT_Glyph_Format format; FT_Bitmap bitmap; FT_Int bitmap_left; FT_Int bitmap_top; FT_Outline outline; FT_UInt num_subglyphs; FT_SubGlyph subglyphs; void* control_data; long control_len; FT_Pos lsb_delta; FT_Pos rsb_delta; void* other; FT_Slot_Internal internal; } FT_GlyphSlotRec;

## *struct FT_Glyph_Metrics_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Pos | width | |
| FT_Pos | height | |
| FT_Pos | horiBearingX | |
| FT_Pos | horiBearingY | |
| FT_Pos | horiAdvance | |
| FT_Pos | vertBearingX | |
| FT_Pos | vertBearingY | |
| FT_Pos | vertAdvance | |

**Definition:**
typedef struct FT_Glyph_Metrics_ { FT_Pos width; FT_Pos height; FT_Pos horiBearingX; FT_Pos horiBearingY; FT_Pos horiAdvance; FT_Pos vertBearingX; FT_Pos vertBearingY; FT_Pos vertAdvance; } FT_Glyph_Metrics;

## *enum FT_Kerning_Mode_*

**Definition:**

typedef enum FT_Kerning_Mode_ { FT_KERNING_DEFAULT = 0, FT_KERNING_UNFITTED, FT_KERNING_UNSCALED } FT_Kerning_Mode;

### *struct FT_LibraryRec_*

**Definition:**
typedef struct FT_LibraryRec_ *FT_Library;

### *struct FT_ModuleRec_*

**Definition:**
typedef struct FT_ModuleRec_* FT_Module;

### *struct FT_Open_Args_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UInt | flags | |
| const FT_Byte* | memory_base | |
| FT_Long | memory_size | |
| FT_String* | pathname | |
| FT_Stream | stream | |
| FT_Module | driver | |
| FT_Int | num_params | |
| FT_Parameter* | params | |

**Definition:**
typedef struct FT_Open_Args_ { FT_UInt flags; const FT_Byte* memory_base; FT_Long memory_size; FT_String* pathname; FT_Stream stream; FT_Module driver; FT_Int num_params; FT_Parameter* params; } FT_Open_Args;

### *enum FT_Render_Mode_*

**Definition:**
typedef enum FT_Render_Mode_ { FT_RENDER_MODE_NORMAL = 0, FT_RENDER_MODE_LIGHT, FT_RENDER_MODE_MONO, FT_RENDER_MODE_LCD, FT_RENDER_MODE_LCD_V, FT_RENDER_MODE_SDF, FT_RENDER_MODE_MAX } FT_Render_Mode;

### *struct FT_RendererRec_*

**Definition:**
typedef struct FT_RendererRec_* FT_Renderer;

### *struct FT_SizeRec_*

**Definition:**
typedef struct FT_SizeRec_* FT_Size;

## *struct FT_SizeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Face | face | |
| FT_Generic | generic | |
| FT_Size_Metrics | metrics | |
| FT_Size_Internal | internal | |

**Definition:**
typedef struct FT_SizeRec_ { FT_Face face; /* parent face object */ FT_Generic generic; /* generic pointer for client uses */ FT_Size_Metrics metrics; /* size metrics */ FT_Size_Internal internal; } FT_SizeRec;

## *typedef FT_Size_Internal*

**Definition:**
* OpenType table with bit~1 in its `flags` field set, instructing the * application to overlay the bitmap strike with the corresponding * outline glyph. See @FT_HAS_SBIX for pseudo code how to use it. * * @since: * 2.12 */ #define FT_HAS_SBIX_OVERLAY( face ) \ ( !!( (face)->face_flags & FT_FACE_FLAG_SBIX_OVERLAY ) ) /********** * * @section: * face_creation * */ /********** * * @enum: * FT_STYLE_FLAG_XXX * * @description: * A list of bit flags to indicate the style of a given face. These are * used in the `style_flags` field of @FT_FaceRec. * * @values: * FT_STYLE_FLAG_ITALIC :: * The face style is italic or oblique. * * FT_STYLE_FLAG_BOLD :: * The face is bold. * * @note: * The style information as provided by FreeType is very basic. More * details are beyond the scope and should be done on a higher level (for * example, by analyzing various fields of the 'OS/2' table in SFNT based * fonts). */ #define FT_STYLE_FLAG_ITALIC ( 1 << 0 ) #define FT_STYLE_FLAG_BOLD ( 1 << 1 ) /********** * * @section: * other_api_data * */ /********** * * @type: * FT_Size_Internal * * @description: * An opaque handle to an `FT_Size_InternalRec` structure, used to model * private data of a given @FT_Size object. */ typedef struct FT_Size_InternalRec_* FT_Size_Internal;

## *typedef FT_Size_Internal*

**Definition:**
* @enum: * FT_STYLE_FLAG_XXX * * @description: * A list of bit flags to indicate the style of a given face. These are * used in the `style_flags` field of @FT_FaceRec. * * @values: * FT_STYLE_FLAG_ITALIC :: * The face style is italic or oblique. * * FT_STYLE_FLAG_BOLD :: * The face is bold. * * @note: * The style information as provided by FreeType is very basic. More * details are beyond the scope and should be done on a higher level (for * example, by analyzing various fields of the 'OS/2' table in SFNT based * fonts). */ #define FT_STYLE_FLAG_ITALIC ( 1 << 0 ) #define FT_STYLE_FLAG_BOLD ( 1 << 1 ) /********** * * @section: * other_api_data * */ /********** * * @type: * FT_Size_Internal * * @description: * An opaque handle to an `FT_Size_InternalRec`

structure, used to model * private data of a given @FT_Size object. */ typedef struct FT_Size_InternalRec_* FT_Size_Internal;

## *typedef FT_Size_Internal*

**Definition:**
* An opaque handle to an `FT_Size_InternalRec` structure, used to model * private data of a given @FT_Size object. */ typedef struct FT_Size_InternalRec_* FT_Size_Internal;

## *struct FT_Size_Metrics_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | x_ppem | |
| FT_UShort | y_ppem | |
| FT_Fixed | x_scale | |
| FT_Fixed | y_scale | |
| FT_Pos | ascender | |
| FT_Pos | descender | |
| FT_Pos | height | |
| FT_Pos | max_advance | |

**Definition:**
typedef struct FT_Size_Metrics_ { FT_UShort x_ppem; /* horizontal pixels per EM */ FT_UShort y_ppem; /* vertical pixels per EM */ FT_Fixed x_scale; /* scaling values used to convert font */ FT_Fixed y_scale; /* units to 26.6 fractional pixels */ FT_Pos ascender; /* ascender in 26.6 frac. pixels */ FT_Pos descender; /* descender in 26.6 frac. pixels */ FT_Pos height; /* text height in 26.6 frac. pixels */ FT_Pos max_advance; /* max horizontal advance, in 26.6 pixels */ } FT_Size_Metrics;

## *struct FT_Size_RequestRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Size_Request_Type | type | |
| FT_Long | width | |
| FT_Long | height | |
| FT_UInt | horiResolution | |
| FT_UInt | vertResolution | |

**Definition:**
typedef struct FT_Size_RequestRec_ { FT_Size_Request_Type type; FT_Long width; FT_Long height; FT_UInt horiResolution; FT_UInt vertResolution; } FT_Size_RequestRec;

## *struct FT_Size_RequestRec_*

**Definition:**
typedef struct FT_Size_RequestRec_ *FT_Size_Request;

## *enum FT_Size_Request_Type_*

**Definition:**
typedef enum FT_Size_Request_Type_ { FT_SIZE_REQUEST_TYPE_NOMINAL, FT_SIZE_REQUEST_TYPE_REAL_DIM, FT_SIZE_REQUEST_TYPE_BBOX, FT_SIZE_REQUEST_TYPE_CELL, FT_SIZE_REQUEST_TYPE_SCALES, FT_SIZE_REQUEST_TYPE_MAX } FT_Size_Request_Type;

## *struct FT_Slot_InternalRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Library | library | |
| FT_Face | face | |
| FT_GlyphSlot | next | |
| FT_UInt | glyph_index | |
| FT_Generic | generic | |
| FT_Glyph_Metrics | metrics | |
| FT_Fixed | linearHoriAdvance | |
| FT_Fixed | linearVertAdvance | |
| FT_Vector | advance | |
| FT_Glyph_Format | format | |
| FT_Bitmap | bitmap | |
| FT_Int | bitmap_left | |
| FT_Int | bitmap_top | |
| FT_Outline | outline | |
| FT_UInt | num_subglyphs | |
| FT_SubGlyph | subglyphs | |
| void* | control_data | |
| long | control_len | |
| FT_Pos | lsb_delta | |
| FT_Pos | rsb_delta | |
| void* | other | |
| FT_Slot_Internal | internal | |

**Definition:**
typedef struct FT_Slot_InternalRec_* FT_Slot_Internal;

## *typedef FT_SubGlyph*

**Definition:**
* @struct: * FT_SubGlyph * * @description: * The subglyph structure is an internal object used to describe * subglyphs (for example, in the case of composites). * * @note: * The subglyph implementation is not part of the high-level API, hence * the forward structure declaration. * * You can however retrieve subglyph information with * @FT_Get_SubGlyph_Info. */ typedef struct FT_SubGlyphRec_* FT_SubGlyph;

## *typedef height*

**Definition:**
* @struct: * FT_Bitmap_Size * * @description: * This structure models the metrics of a bitmap strike (i.e., a set of * glyphs for a given point size and resolution) in a bitmap font. It is * used for the `available_sizes` field of @FT_Face. * * @fields: * height :: * The vertical distance, in pixels, between two consecutive baselines. * It is always positive. * * width :: * The average width, in pixels, of all glyphs in the strike. * * size :: * The nominal size of the strike in 26.6 fractional points. This * field is not very useful. * * x_ppem :: * The horizontal ppem (nominal width) in 26.6 fractional pixels. * * y_ppem :: * The vertical ppem (nominal height) in 26.6 fractional pixels. * * @note: * Windows FNT: * The nominal size given in a FNT font is not reliable. If the driver * finds it incorrect, it sets `size` to some calculated values, and * `x_ppem` and `y_ppem` to the pixel width and height given in the * font, respectively. * * TrueType embedded bitmaps: * `size`, `width`, and `height` values are not contained in the bitmap * strike itself. They are computed from the global font parameters. */ typedef struct FT_Bitmap_Size_ { FT_Short height; FT_Short width; FT_Pos size; FT_Pos x_ppem; FT_Pos y_ppem; } FT_Bitmap_Size;

## *typedef tag*

**Definition:**
* @struct: * FT_Parameter * * @description: * A simple structure to pass more or less generic parameters to * @FT_Open_Face and @FT_Face_Properties. * * @fields: * tag :: * A four-byte identification tag. * * data :: * A pointer to the parameter data. * * @note: * The ID and function of parameters are driver-specific. See section * @parameter_tags for more information. */ typedef struct FT_Parameter_ { FT_ULong tag; FT_Pointer data; } FT_Parameter;

# ftbdf.h

## *Data Types*

### *struct BDF_PropertyRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| BDF_PropertyType | type | |
| const char* | atom | |
| FT_Int32 | integer | |
| FT_UInt32 | cardinal | |

**Definition:**
typedef struct BDF_PropertyRec_* BDF_Property;

### *struct BDF_PropertyRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| BDF_PropertyType | type | |
| const char* | atom | |
| FT_Int32 | integer | |
| FT_UInt32 | cardinal | |

**Definition:**
typedef struct BDF_PropertyRec_ { BDF_PropertyType type; union { const char* atom; FT_Int32 integer; FT_UInt32 cardinal; } u; } BDF_PropertyRec;

### *typedef BDF_PropertyType*

**Definition:**
* @enum: * BDF_PropertyType * * @description: * A list of BDF property types. * * @values: * BDF_PROPERTY_TYPE_NONE :: * Value~0 is used to indicate a missing property. * * BDF_PROPERTY_TYPE_ATOM :: * Property is a string atom. * * BDF_PROPERTY_TYPE_INTEGER :: * Property is a 32-bit signed integer. * * BDF_PROPERTY_TYPE_CARDINAL :: * Property is a 32-bit unsigned integer. */ typedef enum BDF_PropertyType_ { BDF_PROPERTY_TYPE_NONE = 0, BDF_PROPERTY_TYPE_ATOM = 1, BDF_PROPERTY_TYPE_INTEGER = 2, BDF_PROPERTY_TYPE_CARDINAL = 3 } BDF_PropertyType;

# ftcache.h

## *Data Types*

### *struct FTC_CMapCacheRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FTC_FaceID | face_id | |
| FT_UInt | width | |
| FT_UInt | height | |
| FT_Int32 | flags | |

**Definition:**
typedef struct FTC_CMapCacheRec_* FTC_CMapCache;

### *typedef FTC_FaceID*

**Definition:**
* These pointers are typically used to point to a user-defined structure * containing a font file path, and face index. * * @note: * Never use `NULL` as a valid @FTC_FaceID. * * Face IDs are passed by the client to the cache manager that calls, * when needed, the @FTC_Face_Requester to translate them into new * @FT_Face objects. * * If the content of a given face ID changes at runtime, or if the value * becomes invalid (e.g., when uninstalling a font), you should * immediately call @FTC_Manager_RemoveFaceID before any other cache * function. * * Failure to do so will result in incorrect behaviour or even memory * leaks and crashes. */ typedef FT_Pointer FTC_FaceID;

### *struct FTC_ImageCacheRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte | width | |
| FT_Byte | height | |
| FT_Char | left | |
| FT_Char | top | |
| FT_Byte | format | |
| FT_Byte | max_grays | |
| FT_Short | pitch | |
| FT_Char | xadvance | |
| FT_Char | yadvance | |
| FT_Byte* | buffer | |

**Definition:**
typedef struct FTC_ImageCacheRec_* FTC_ImageCache;

### *struct FTC_ImageTypeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte | width | |
| FT_Byte | height | |
| FT_Char | left | |
| FT_Char | top | |
| FT_Byte | format | |
| FT_Byte | max_grays | |
| FT_Short | pitch | |
| FT_Char | xadvance | |
| FT_Char | yadvance | |
| FT_Byte* | buffer | |

**Definition:**

typedef struct FTC_ImageTypeRec_* FTC_ImageType;

### *struct FTC_ManagerRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FTC_FaceID | face_id | |
| FT_UInt | width | |
| FT_UInt | height | |
| FT_Int | pixel | |
| FT_UInt | x_res | |
| FT_UInt | y_res | |

**Definition:**

typedef struct FTC_ManagerRec_* FTC_Manager;

### *struct FTC_NodeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FTC_FaceID | face_id | |
| FT_UInt | width | |
| FT_UInt | height | |
| FT_Int | pixel | |

| | | |
|---|---|---|
| FT_UInt | x_res | |
| FT_UInt | y_res | |

**Definition:**
typedef struct FTC_NodeRec_* FTC_Node;

## *typedef FTC_SBit*

**Definition:**
* A handle to a small bitmap descriptor. See the @FTC_SBitRec structure * for details. */ typedef struct FTC_SBitRec_* FTC_SBit;

## *struct FTC_SBitCacheRec_*

**Definition:**
typedef struct FTC_SBitCacheRec_* FTC_SBitCache;

## *struct FTC_SBitRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte | width | |
| FT_Byte | height | |
| FT_Char | left | |
| FT_Char | top | |
| FT_Byte | format | |
| FT_Byte | max_grays | |
| FT_Short | pitch | |
| FT_Char | xadvance | |
| FT_Char | yadvance | |
| FT_Byte* | buffer | |

**Definition:**
typedef struct FTC_SBitRec_ { FT_Byte width; FT_Byte height; FT_Char left; FT_Char top; FT_Byte format; FT_Byte max_grays; FT_Short pitch; FT_Char xadvance; FT_Char yadvance; FT_Byte* buffer; } FTC_SBitRec;

## *struct FTC_ScalerRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FTC_FaceID | face_id | |
| FT_UInt | width | |

| | | |
|---|---|---|
| FT_UInt | height | |
| FT_Int32 | flags | |

**Definition:**
typedef struct FTC_ScalerRec_* FTC_Scaler;

## *typedef face_id*

**Definition:**
* @struct: * FTC_ScalerRec * * @description: * A structure used to describe a given character size in either pixels * or points to the cache manager. See @FTC_Manager_LookupSize. * * @fields: * face_id :: * The source face ID. * * width :: * The character width. * * height :: * The character height. * * pixel :: * A Boolean. If 1, the `width` and `height` fields are interpreted as * integer pixel character sizes. Otherwise, they are expressed as * 1/64 of points. * * x_res :: * Only used when `pixel` is value~0 to indicate the horizontal * resolution in dpi. * * y_res :: * Only used when `pixel` is value~0 to indicate the vertical * resolution in dpi. * * @note: * This type is mainly used to retrieve @FT_Size objects through the * cache manager. */ typedef struct FTC_ScalerRec_ { FTC_FaceID face_id; FT_UInt width; FT_UInt height; FT_Int pixel; FT_UInt x_res; FT_UInt y_res; } FTC_ScalerRec;

## *typedef face_id*

**Definition:**
* @struct: * FTC_ImageTypeRec * * @description: * A structure used to model the type of images in a glyph cache. * * @fields: * face_id :: * The face ID. * * width :: * The width in pixels. * * height :: * The height in pixels. * * flags :: * The load flags, as in @FT_Load_Glyph. * */ typedef struct FTC_ImageTypeRec_ { FTC_FaceID face_id; FT_UInt width; FT_UInt height; FT_Int32 flags; } FTC_ImageTypeRec;

# ftcolor.h

## *Data Types*

### *struct FT_Affine_23_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Fixed  xx, xy, | dx | |
| FT_Fixed  yx, yy, | dy | |

**Definition:**
typedef struct FT_Affine_23_ { FT_Fixed xx, xy, dx; FT_Fixed yx, yy, dy; } FT_Affine23;

### *struct FT_COLR_Paint_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_PaintFormat | format | |
| FT_PaintColrLayers | colr_layers | |
| FT_PaintGlyph | glyph | |
| FT_PaintSolid | solid | |
| FT_PaintLinearGradient | linear_gradient | |
| FT_PaintRadialGradient | radial_gradient | |
| FT_PaintSweepGradient | sweep_gradient | |
| FT_PaintTransform | transform | |
| FT_PaintTranslate | translate | |
| FT_PaintScale | scale | |
| FT_PaintRotate | rotate | |
| FT_PaintSkew | skew | |
| FT_PaintComposite | composite | |
| FT_PaintColrGlyph | colr_glyph | |

**Definition:**
typedef struct FT_COLR_Paint_ { FT_PaintFormat format; union { FT_PaintColrLayers colr_layers; FT_PaintGlyph glyph; FT_PaintSolid solid; FT_PaintLinearGradient linear_gradient; FT_PaintRadialGradient radial_gradient; FT_PaintSweepGradient sweep_gradient; FT_PaintTransform transform; FT_PaintTranslate translate; FT_PaintScale scale; FT_PaintRotate rotate; FT_PaintSkew skew; FT_PaintComposite composite; FT_PaintColrGlyph colr_glyph; } u; } FT_COLR_Paint;

### *struct FT_ClipBox_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Vector | bottom_left | |
| FT_Vector | top_left | |
| FT_Vector | top_right | |
| FT_Vector | bottom_right | |

**Definition:**
typedef struct FT_ClipBox_ { FT_Vector bottom_left; FT_Vector top_left; FT_Vector top_right; FT_Vector bottom_right; } FT_ClipBox;

## *struct FT_ColorIndex_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UInt16 | palette_index | |
| FT_F2Dot14 | alpha | |

**Definition:**
typedef struct FT_ColorIndex_ { FT_UInt16 palette_index; FT_F2Dot14 alpha; } FT_ColorIndex;

## *struct FT_ColorLine_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_PaintExtend | extend | |
| FT_ColorStopIterator | color_stop_iterator | |

**Definition:**
typedef struct FT_ColorLine_ { FT_PaintExtend extend; FT_ColorStopIterator color_stop_iterator; } FT_ColorLine;

## *struct FT_ColorStopIterator_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UInt | num_color_stops | |
| FT_UInt | current_color_stop | |
| FT_Byte* | p | |
| FT_Bool | read_variable | |

**Definition:**
typedef struct FT_ColorStopIterator_ { FT_UInt num_color_stops; FT_UInt current_color_stop; FT_Byte* p; FT_Bool read_variable; } FT_ColorStopIterator;

## *struct FT_ColorStop_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Fixed | stop_offset | |
| FT_ColorIndex | color | |

**Definition:**
typedef struct FT_ColorStop_ { FT_Fixed stop_offset; FT_ColorIndex color; } FT_ColorStop;

## *struct FT_Color_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte | blue | |
| FT_Byte | green | |
| FT_Byte | red | |
| FT_Byte | alpha | |

**Definition:**
typedef struct FT_Color_ { FT_Byte blue; FT_Byte green; FT_Byte red; FT_Byte alpha; } FT_Color;

## *enum FT_Color_Root_Transform_*

**Definition:**
typedef enum FT_Color_Root_Transform_ { FT_COLOR_INCLUDE_ROOT_TRANSFORM, FT_COLOR_NO_ROOT_TRANSFORM, FT_COLOR_ROOT_TRANSFORM_MAX } FT_Color_Root_Transform;

## *enum FT_Composite_Mode_*

**Definition:**
typedef enum FT_Composite_Mode_ { FT_COLR_COMPOSITE_CLEAR = 0, FT_COLR_COMPOSITE_SRC = 1, FT_COLR_COMPOSITE_DEST = 2, FT_COLR_COMPOSITE_SRC_OVER = 3, FT_COLR_COMPOSITE_DEST_OVER = 4, FT_COLR_COMPOSITE_SRC_IN = 5, FT_COLR_COMPOSITE_DEST_IN = 6, FT_COLR_COMPOSITE_SRC_OUT = 7, FT_COLR_COMPOSITE_DEST_OUT = 8, FT_COLR_COMPOSITE_SRC_ATOP = 9, FT_COLR_COMPOSITE_DEST_ATOP = 10, FT_COLR_COMPOSITE_XOR = 11, FT_COLR_COMPOSITE_PLUS = 12, FT_COLR_COMPOSITE_SCREEN = 13, FT_COLR_COMPOSITE_OVERLAY = 14, FT_COLR_COMPOSITE_DARKEN = 15, FT_COLR_COMPOSITE_LIGHTEN = 16, FT_COLR_COMPOSITE_COLOR_DODGE = 17, FT_COLR_COMPOSITE_COLOR_BURN = 18, FT_COLR_COMPOSITE_HARD_LIGHT = 19, FT_COLR_COMPOSITE_SOFT_LIGHT = 20, FT_COLR_COMPOSITE_DIFFERENCE = 21, FT_COLR_COMPOSITE_EXCLUSION = 22, FT_COLR_COMPOSITE_MULTIPLY = 23, FT_COLR_COMPOSITE_HSL_HUE = 24, FT_COLR_COMPOSITE_HSL_SATURATION = 25, FT_COLR_COMPOSITE_HSL_COLOR = 26, FT_COLR_COMPOSITE_HSL_LUMINOSITY = 27, FT_COLR_COMPOSITE_MAX = 28 } FT_Composite_Mode;

### *struct FT_Opaque_Paint_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte* | p | |
| FT_Bool | insert_root_transform | |

**Definition:**
typedef struct FT_Opaque_Paint_ { FT_Byte* p; FT_Bool insert_root_transform; } FT_OpaquePaint;

### *struct FT_PaintColrGlyph_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UInt | glyphID | |

**Definition:**
typedef struct FT_PaintColrGlyph_ { FT_UInt glyphID; } FT_PaintColrGlyph;

### *struct FT_PaintColrLayers_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_LayerIterator | layer_iterator | |

**Definition:**
typedef struct FT_PaintColrLayers_ { FT_LayerIterator layer_iterator; } FT_PaintColrLayers;

### *struct FT_PaintComposite_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_OpaquePaint | source_paint | |
| FT_Composite_Mode | composite_mode | |
| FT_OpaquePaint | backdrop_paint | |

**Definition:**
typedef struct FT_PaintComposite_ { FT_OpaquePaint source_paint; FT_Composite_Mode composite_mode; FT_OpaquePaint backdrop_paint; } FT_PaintComposite;

### *enum FT_PaintExtend_*

**Definition:**
typedef enum FT_PaintExtend_ { FT_COLR_PAINT_EXTEND_PAD = 0, FT_COLR_PAINT_EXTEND_REPEAT = 1, FT_COLR_PAINT_EXTEND_REFLECT = 2 } FT_PaintExtend;

### *enum FT_PaintFormat_*

**Definition:**
typedef enum FT_PaintFormat_ { FT_COLR_PAINTFORMAT_COLR_LAYERS = 1, FT_COLR_PAINTFORMAT_SOLID = 2, FT_COLR_PAINTFORMAT_LINEAR_GRADIENT = 4, FT_COLR_PAINTFORMAT_RADIAL_GRADIENT = 6, FT_COLR_PAINTFORMAT_SWEEP_GRADIENT = 8, FT_COLR_PAINTFORMAT_GLYPH = 10, FT_COLR_PAINTFORMAT_COLR_GLYPH = 11, FT_COLR_PAINTFORMAT_TRANSFORM = 12, FT_COLR_PAINTFORMAT_TRANSLATE = 14, FT_COLR_PAINTFORMAT_SCALE = 16, FT_COLR_PAINTFORMAT_ROTATE = 24, FT_COLR_PAINTFORMAT_SKEW = 28, FT_COLR_PAINTFORMAT_COMPOSITE = 32, FT_COLR_PAINT_FORMAT_MAX = 33, FT_COLR_PAINTFORMAT_UNSUPPORTED = 255 } FT_PaintFormat;

## *struct FT_PaintGlyph_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_OpaquePaint | paint | |
| FT_UInt | glyphID | |

**Definition:**
typedef struct FT_PaintGlyph_ { FT_OpaquePaint paint; FT_UInt glyphID; } FT_PaintGlyph;

## *struct FT_PaintLinearGradient_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ColorLine | colorline | |
| FT_Vector | p0 | |
| FT_Vector | p1 | |
| FT_Vector | p2 | |

**Definition:**
typedef struct FT_PaintLinearGradient_ { FT_ColorLine colorline; /* TODO: Potentially expose those as x0, y0 etc. */ FT_Vector p0; FT_Vector p1; FT_Vector p2; } FT_PaintLinearGradient;

## *struct FT_PaintRadialGradient_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ColorLine | colorline | |
| FT_Vector | c0 | |
| FT_Pos | r0 | |
| FT_Vector | c1 | |
| FT_Pos | r1 | |

**Definition:**
typedef struct FT_PaintRadialGradient_ { FT_ColorLine colorline; FT_Vector c0; FT_Pos r0; FT_Vector c1; FT_Pos r1; } FT_PaintRadialGradient;

### *struct FT_PaintRotate_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_OpaquePaint | paint | |
| FT_Fixed | angle | |
| FT_Fixed | center_x | |
| FT_Fixed | center_y | |

**Definition:**
typedef struct FT_PaintRotate_ { FT_OpaquePaint paint; FT_Fixed angle; FT_Fixed center_x; FT_Fixed center_y; } FT_PaintRotate;

### *struct FT_PaintScale_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_OpaquePaint | paint | |
| FT_Fixed | scale_x | |
| FT_Fixed | scale_y | |
| FT_Fixed | center_x | |
| FT_Fixed | center_y | |

**Definition:**
typedef struct FT_PaintScale_ { FT_OpaquePaint paint; FT_Fixed scale_x; FT_Fixed scale_y; FT_Fixed center_x; FT_Fixed center_y; } FT_PaintScale;

### *struct FT_PaintSkew_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_OpaquePaint | paint | |
| FT_Fixed | x_skew_angle | |
| FT_Fixed | y_skew_angle | |
| FT_Fixed | center_x | |
| FT_Fixed | center_y | |

**Definition:**
typedef struct FT_PaintSkew_ { FT_OpaquePaint paint; FT_Fixed x_skew_angle; FT_Fixed y_skew_angle; FT_Fixed center_x; FT_Fixed center_y; } FT_PaintSkew;

### *struct FT_PaintSolid_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ColorIndex | color | |

**Definition:**
typedef struct FT_PaintSolid_ { FT_ColorIndex color; } FT_PaintSolid;

## *struct FT_PaintSweepGradient_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ColorLine | colorline | |
| FT_Vector | center | |
| FT_Fixed | start_angle | |
| FT_Fixed | end_angle | |

**Definition:**
typedef struct FT_PaintSweepGradient_ { FT_ColorLine colorline; FT_Vector center; FT_Fixed start_angle; FT_Fixed end_angle; } FT_PaintSweepGradient;

## *struct FT_PaintTransform_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_OpaquePaint | paint | |
| FT_Affine23 | affine | |

**Definition:**
typedef struct FT_PaintTransform_ { FT_OpaquePaint paint; FT_Affine23 affine; } FT_PaintTransform;

## *struct FT_PaintTranslate_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_OpaquePaint | paint | |
| FT_Fixed | dx | |
| FT_Fixed | dy | |

**Definition:**
typedef struct FT_PaintTranslate_ { FT_OpaquePaint paint; FT_Fixed dx; FT_Fixed dy; } FT_PaintTranslate;

## *typedef num_layers*

**Definition:**
* @struct: * FT_LayerIterator * * @description: * This iterator object is needed for @FT_Get_Color_Glyph_Layer. * * @fields: * num_layers :: * The number of glyph layers for the

requested glyph index. Will be * set by @FT_Get_Color_Glyph_Layer. * * layer :: * The current layer. Will be set by @FT_Get_Color_Glyph_Layer. * * p :: * An opaque pointer into 'COLR' table data. The caller must set this * to `NULL` before the first call of @FT_Get_Color_Glyph_Layer. */ typedef struct FT_LayerIterator_ { FT_UInt num_layers; FT_UInt layer; FT_Byte* p; } FT_LayerIterator;

## *typedef num_palettes*

**Definition:**
* @struct: * FT_Palette_Data * * @description: * This structure holds the data of the 'CPAL' table. * * @fields: * num_palettes :: * The number of palettes. * * palette_name_ids :: * An optional read-only array of palette name IDs with `num_palettes` * elements, corresponding to entries like 'dark' or 'light' in the * font's 'name' table. * * An empty name ID in the 'CPAL' table gets represented as value * 0xFFFF. * * `NULL` if the font's 'CPAL' table doesn't contain appropriate data. * * palette_flags :: * An optional read-only array of palette flags with `num_palettes` * elements. Possible values are an ORed combination of * @FT_PALETTE_FOR_LIGHT_BACKGROUND and * @FT_PALETTE_FOR_DARK_BACKGROUND. * * `NULL` if the font's 'CPAL' table doesn't contain appropriate data. * * num_palette_entries :: * The number of entries in a single palette. All palettes have the * same size. * * palette_entry_name_ids :: * An optional read-only array of palette entry name IDs with * `num_palette_entries`. In each palette, entries with the same index * have the same function. For example, index~0 might correspond to * string 'outline' in the font's 'name' table to indicate that this * palette entry is used for outlines, index~1 might correspond to * 'fill' to indicate the filling color palette entry, etc. * * An empty entry name ID in the 'CPAL' table gets represented as value * 0xFFFF. * * `NULL` if the font's 'CPAL' table doesn't contain appropriate data. * * @note: * Use function @FT_Get_Sfnt_Name to map name IDs and entry name IDs to * name strings. * * Use function @FT_Palette_Select to get the colors associated with a * palette entry. * * @since: * 2.10 */ typedef struct FT_Palette_Data_ { FT_UShort num_palettes; const FT_UShort* palette_name_ids; const FT_UShort* palette_flags; FT_UShort num_palette_entries; const FT_UShort* palette_entry_name_ids; } FT_Palette_Data;

# ftdriver.h

## *Data Types*

### *struct FT_Prop_GlyphToScriptMap_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Face | face | |
| FT_UShort* | map | |

**Definition:**
typedef struct FT_Prop_GlyphToScriptMap_ { FT_Face face; FT_UShort* map; } FT_Prop_GlyphToScriptMap;

### *typedef face*

**Definition:**
* @enum: * FT_AUTOHINTER_SCRIPT_XXX * * @description: * **Experimental only** * * A list of constants used for the @glyph-to-script-map property to * specify the script submodule the auto-hinter should use for hinting a * particular glyph. * * @values: * FT_AUTOHINTER_SCRIPT_NONE :: * Don't auto-hint this glyph. * * FT_AUTOHINTER_SCRIPT_LATIN :: * Apply the latin auto-hinter. For the auto-hinter, 'latin' is a very * broad term, including Cyrillic and Greek also since characters from * those scripts share the same design constraints. * * By default, characters from the following Unicode ranges are * assigned to this submodule. * * ``` * U+0020 - U+007F // Basic Latin (no control characters) * U+00A0 - U+00FF // Latin-1 Supplement (no control characters) * U+0100 - U+017F // Latin Extended-A * U+0180 - U+024F // Latin Extended-B * U+0250 - U+02AF // IPA Extensions * U+02B0 - U+02FF // Spacing Modifier Letters * U+0300 - U+036F // Combining Diacritical Marks * U+0370 - U+03FF // Greek and Coptic * U+0400 - U+04FF // Cyrillic * U+0500 - U+052F // Cyrillic Supplement * U+1D00 - U+1D7F // Phonetic Extensions * U+1D80 - U+1DBF // Phonetic Extensions Supplement * U+1DC0 - U+1DFF // Combining Diacritical Marks Supplement * U+1E00 - U+1EFF // Latin Extended Additional * U+1F00 - U+1FFF // Greek Extended * U+2000 - U+206F // General Punctuation * U+2070 - U+209F // Superscripts and Subscripts * U+20A0 - U+20CF // Currency Symbols * U+2150 - U+218F // Number Forms * U+2460 - U+24FF // Enclosed Alphanumerics * U+2C60 - U+2C7F // Latin Extended-C * U+2DE0 - U+2DFF // Cyrillic Extended-A * U+2E00 - U+2E7F // Supplemental Punctuation * U+A640 - U+A69F // Cyrillic Extended-B * U+A720 - U+A7FF // Latin Extended-D * U+FB00 - U+FB06 // Alphab. Present. Forms (Latin Ligatures) * U+1D400 - U+1D7FF // Mathematical Alphanumeric Symbols * U+1F100 - U+1F1FF // Enclosed Alphanumeric Supplement * ``` * * FT_AUTOHINTER_SCRIPT_CJK :: * Apply the CJK auto-hinter, covering Chinese, Japanese, Korean, old * Vietnamese, and some other scripts. * * By default, characters from the following Unicode ranges are * assigned to this submodule. * * ``` * U+1100 - U+11FF // Hangul Jamo * U+2E80 - U+2EFF // CJK Radicals Supplement * U+2F00 - U+2FDF // Kangxi Radicals * U+2FF0 - U+2FFF // Ideographic Description Characters * U+3000 - U+303F // CJK Symbols and Punctuation * U+3040 - U+309F // Hiragana * U+30A0 - U+30FF // Katakana * U+3100 - U+312F // Bopomofo * U+3130 - U+318F // Hangul Compatibility Jamo * U+3190 - U+319F // Kanbun * U+31A0 - U+31BF // Bopomofo Extended * U+31C0 - U+31EF // CJK Strokes * U+31F0 - U+31FF // Katakana Phonetic Extensions * U+3200 - U+32FF // Enclosed CJK Letters and Months * U+3300 - U+33FF // CJK Compatibility * U+3400 - U+4DBF // CJK Unified Ideographs Extension A * U+4DC0 - U+4DFF // Yijing Hexagram Symbols * U+4E00 - U+9FFF // CJK Unified Ideographs * U+A960 - U+A97F // Hangul Jamo Extended-A * U+AC00 - U+D7AF // Hangul Syllables * U+D7B0 - U+D7FF // Hangul Jamo Extended-B * U+F900 -

U+FAFF // CJK Compatibility Ideographs * U+FE10 - U+FE1F // Vertical forms * U+FE30 - U+FE4F // CJK Compatibility Forms * U+FF00 - U+FFEF // Halfwidth and Fullwidth Forms * U+1B000 - U+1B0FF // Kana Supplement * U+1D300 - U+1D35F // Tai Xuan Hing Symbols * U+1F200 - U+1F2FF // Enclosed Ideographic Supplement * U+20000 - U+2A6DF // CJK Unified Ideographs Extension B * U+2A700 - U+2B73F // CJK Unified Ideographs Extension C * U+2B740 - U+2B81F // CJK Unified Ideographs Extension D * U+2F800 - U+2FA1F // CJK Compatibility Ideographs Supplement * ``` * * FT_AUTOHINTER_SCRIPT_INDIC :: * Apply the indic auto-hinter, covering all major scripts from the * Indian sub-continent and some other related scripts like Thai, Lao, * or Tibetan. * * By default, characters from the following Unicode ranges are * assigned to this submodule. * * ``` * U+0900 - U+0DFF // Indic Range * U+0F00 - U+0FFF // Tibetan * U+1900 - U+194F // Limbu * U+1B80 - U+1BBF // Sundanese * U+A800 - U+A82F // Syloti Nagri * U+ABC0 - U+ABFF // Meetei Mayek * U+11800 - U+118DF // Sharada * ``` * * Note that currently Indic support is rudimentary only, missing blue * zone support. * * @since: * 2.4.11 * */ #define FT_AUTOHINTER_SCRIPT_NONE 0 #define FT_AUTOHINTER_SCRIPT_LATIN 1 #define FT_AUTOHINTER_SCRIPT_CJK 2 #define FT_AUTOHINTER_SCRIPT_INDIC 3
/********************************************************************** * * @struct: * FT_Prop_GlyphToScriptMap * * @description: * **Experimental only** * * The data exchange structure for the @glyph-to-script-map property. * * @since: * 2.4.11 * */ typedef struct FT_Prop_GlyphToScriptMap_ { FT_Face face; FT_UShort* map; } FT_Prop_GlyphToScriptMap;

## *typedef face*

**Definition:**
* @struct: * FT_Prop_IncreaseXHeight * * @description: * The data exchange structure for the @increase-x-height property. * */ typedef struct FT_Prop_IncreaseXHeight_ { FT_Face face; FT_UInt limit; } FT_Prop_IncreaseXHeight;

# ftglyph.h

## *Data Types*

### *struct FT_BitmapGlyphRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_GlyphRec | root | |
| FT_Int | left | |
| FT_Int | top | |
| FT_Bitmap | bitmap | |

**Definition:**
typedef struct FT_BitmapGlyphRec_* FT_BitmapGlyph;

### *struct FT_BitmapGlyphRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_GlyphRec | root | |
| FT_Int | left | |
| FT_Int | top | |
| FT_Bitmap | bitmap | |

**Definition:**
typedef struct FT_BitmapGlyphRec_ { FT_GlyphRec root; FT_Int left; FT_Int top; FT_Bitmap bitmap; } FT_BitmapGlyphRec;

### *struct FT_GlyphRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Library | library | |
| const FT_Glyph_Class* | clazz | |
| FT_Glyph_Format | format | |
| FT_Vector | advance | |

**Definition:**
typedef struct FT_GlyphRec_* FT_Glyph;

### *struct FT_GlyphRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Library | library | |
| const FT_Glyph_Class* | clazz | |
| FT_Glyph_Format | format | |
| FT_Vector | advance | |

**Definition:**
typedef struct FT_GlyphRec_ { FT_Library library; const FT_Glyph_Class* clazz; FT_Glyph_Format format; FT_Vector advance; } FT_GlyphRec;

## *typedef FT_Glyph_BBox_Mode*

**Definition:**
* @enum: * FT_Glyph_BBox_Mode * * @description: * The mode how the values of @FT_Glyph_Get_CBox are returned. * * @values: * FT_GLYPH_BBOX_UNSCALED :: * Return unscaled font units. * * FT_GLYPH_BBOX_SUBPIXELS :: * Return unfitted 26.6 coordinates. * * FT_GLYPH_BBOX_GRIDFIT :: * Return grid-fitted 26.6 coordinates. * * FT_GLYPH_BBOX_TRUNCATE :: * Return coordinates in integer pixels. * * FT_GLYPH_BBOX_PIXELS :: * Return grid-fitted pixel coordinates. */ typedef enum FT_Glyph_BBox_Mode_ { FT_GLYPH_BBOX_UNSCALED = 0, FT_GLYPH_BBOX_SUBPIXELS = 0, FT_GLYPH_BBOX_GRIDFIT = 1, FT_GLYPH_BBOX_TRUNCATE = 2, FT_GLYPH_BBOX_PIXELS = 3 } FT_Glyph_BBox_Mode;

## *struct FT_Glyph_Class_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Library | library | |
| const FT_Glyph_Class* | clazz | |
| FT_Glyph_Format | format | |
| FT_Vector | advance | |

**Definition:**
typedef struct FT_Glyph_Class_ FT_Glyph_Class;

## *struct FT_OutlineGlyphRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_GlyphRec | root | |
| FT_Outline | outline | |

**Definition:**
typedef struct FT_OutlineGlyphRec_* FT_OutlineGlyph;

## *struct FT_OutlineGlyphRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_GlyphRec | root | |
| FT_Outline | outline | |

**Definition:**
typedef struct FT_OutlineGlyphRec_ { FT_GlyphRec root; FT_Outline outline; } FT_OutlineGlyphRec;

## *struct FT_SvgGlyphRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_GlyphRec | root | |
| FT_Byte* | svg_document | |
| FT_ULong | svg_document_length | |
| FT_UInt | glyph_index | |
| FT_Size_Metrics | metrics | |
| FT_UShort | units_per_EM | |
| FT_UShort | start_glyph_id | |
| FT_UShort | end_glyph_id | |
| FT_Matrix | transform | |
| FT_Vector | delta | |

**Definition:**
typedef struct FT_SvgGlyphRec_* FT_SvgGlyph;

## *struct FT_SvgGlyphRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_GlyphRec | root | |
| FT_Byte* | svg_document | |
| FT_ULong | svg_document_length | |
| FT_UInt | glyph_index | |
| FT_Size_Metrics | metrics | |
| FT_UShort | units_per_EM | |
| FT_UShort | start_glyph_id | |
| FT_UShort | end_glyph_id | |
| FT_Matrix | transform | |
| FT_Vector | delta | |

**Definition:**

typedef struct FT_SvgGlyphRec_ { FT_GlyphRec root; FT_Byte* svg_document; FT_ULong svg_document_length; FT_UInt glyph_index; FT_Size_Metrics metrics; FT_UShort units_per_EM; FT_UShort start_glyph_id; FT_UShort end_glyph_id; FT_Matrix transform; FT_Vector delta; } FT_SvgGlyphRec;

# ftimage.h

## *Data Types*

### *struct FT_BBox_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Pos  xMin, | yMin | |
| FT_Pos  xMax, | yMax | |

**Definition:**
typedef struct FT_BBox_ { FT_Pos xMin, yMin; FT_Pos xMax, yMax; } FT_BBox;

### *struct FT_Bitmap_*

**Members:**

| Type | Name | Description |
|---|---|---|
| unsigned int | rows | |
| unsigned int | width | |
| int | pitch | |
| unsigned char* | buffer | |
| unsigned short | num_grays | |
| unsigned char | pixel_mode | |
| unsigned char | palette_mode | |
| void* | palette | |

**Definition:**
typedef struct FT_Bitmap_ { unsigned int rows; unsigned int width; int pitch; unsigned char* buffer; unsigned short num_grays; unsigned char pixel_mode; unsigned char palette_mode; void* palette; } FT_Bitmap;

### *enum FT_Glyph_Format_*

**Definition:**
typedef enum FT_Glyph_Format_ { FT_IMAGE_TAG( FT_GLYPH_FORMAT_NONE, 0, 0, 0, 0 ), FT_IMAGE_TAG( FT_GLYPH_FORMAT_COMPOSITE, 'c', 'o', 'm', 'p' ), FT_IMAGE_TAG( FT_GLYPH_FORMAT_BITMAP, 'b', 'i', 't', 's' ), FT_IMAGE_TAG( FT_GLYPH_FORMAT_OUTLINE, 'o', 'u', 't', 'l' ), FT_IMAGE_TAG( FT_GLYPH_FORMAT_PLOTTER, 'p', 'l', 'o', 't' ), FT_IMAGE_TAG( FT_GLYPH_FORMAT_SVG, 'S', 'V', 'G', ' ' ) } FT_Glyph_Format;

### *struct FT_Outline_*

**Members:**

| Type | Name | Description |
|---|---|---|

| | | |
|---|---|---|
| unsigned short | n_contours | |
| unsigned short | n_points | |
| FT_Vector* | points | |
| unsigned char* | tags | |
| unsigned short* | contours | |
| int | flags | |

**Definition:**
typedef struct FT_Outline_ { unsigned short n_contours; /* number of contours in glyph */ unsigned short n_points; /* number of points in the glyph */ FT_Vector* points; /* the outline's points */ unsigned char* tags; /* the points flags */ unsigned short* contours; /* the contour end points */ int flags; /* outline masks */ } FT_Outline;

## *struct FT_Outline_Funcs_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Outline_MoveToFunc | move_to | |
| FT_Outline_LineToFunc | line_to | |
| FT_Outline_ConicToFunc | conic_to | |
| FT_Outline_CubicToFunc | cubic_to | |
| int | shift | |
| FT_Pos | delta | |

**Definition:**
typedef struct FT_Outline_Funcs_ { FT_Outline_MoveToFunc move_to; FT_Outline_LineToFunc line_to; FT_Outline_ConicToFunc conic_to; FT_Outline_CubicToFunc cubic_to; int shift; FT_Pos delta; } FT_Outline_Funcs;

## *enum FT_Pixel_Mode_*

**Definition:**
typedef enum FT_Pixel_Mode_ { FT_PIXEL_MODE_NONE = 0, FT_PIXEL_MODE_MONO, FT_PIXEL_MODE_GRAY, FT_PIXEL_MODE_GRAY2, FT_PIXEL_MODE_GRAY4, FT_PIXEL_MODE_LCD, FT_PIXEL_MODE_LCD_V, FT_PIXEL_MODE_BGRA, FT_PIXEL_MODE_MAX /* do not remove */ } FT_Pixel_Mode;

## *typedef FT_Pos*

**Definition:**
typedef signed long FT_Pos;

## *struct FT_RasterRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Glyph_Format | glyph_format | |
| FT_Raster_NewFunc | raster_new | |
| FT_Raster_ResetFunc | raster_reset | |
| FT_Raster_SetModeFunc | raster_set_mode | |
| FT_Raster_RenderFunc | raster_render | |
| FT_Raster_DoneFunc | raster_done | |

**Definition:**
typedef struct FT_RasterRec_* FT_Raster;

## *struct FT_Raster_Funcs_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Glyph_Format | glyph_format | |
| FT_Raster_NewFunc | raster_new | |
| FT_Raster_ResetFunc | raster_reset | |
| FT_Raster_SetModeFunc | raster_set_mode | |
| FT_Raster_RenderFunc | raster_render | |
| FT_Raster_DoneFunc | raster_done | |

**Definition:**
typedef struct FT_Raster_Funcs_ { FT_Glyph_Format glyph_format; FT_Raster_NewFunc raster_new; FT_Raster_ResetFunc raster_reset; FT_Raster_SetModeFunc raster_set_mode; FT_Raster_RenderFunc raster_render; FT_Raster_DoneFunc raster_done; } FT_Raster_Funcs;

## *struct FT_Vector_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Pos | x | |
| FT_Pos | y | |

**Definition:**
typedef struct FT_Vector_ { FT_Pos x; FT_Pos y; } FT_Vector;

## *typedef target*

**Definition:**
* @struct: * FT_Raster_Params * * @description: * A structure to hold the parameters used by a raster's render function, * passed as an argument to @FT_Outline_Render. * * @fields: * target :: * The target bitmap. * * source :: * A pointer to the source glyph image (e.g., an @FT_Outline). * * flags :: * The rendering flags. * * gray_spans :: * The gray span drawing callback. * * black_spans :: * Unused. * * bit_test :: * Unused. * * bit_set :: * Unused. * * user :: * User-supplied data that is passed to each

drawing callback. * * clip_box :: * An optional span clipping box expressed in _integer_ pixels * (not in 26.6 fixed-point units). * * @note: * The @FT_RASTER_FLAG_AA bit flag must be set in the `flags` to * generate an anti-aliased glyph bitmap, otherwise a monochrome bitmap * is generated. The `target` should have appropriate pixel mode and its * dimensions define the clipping region. * * If both @FT_RASTER_FLAG_AA and @FT_RASTER_FLAG_DIRECT bit flags * are set in `flags`, the raster calls an @FT_SpanFunc callback * `gray_spans` with `user` data as an argument ignoring `target`. This * allows direct composition over a pre-existing user surface to perform * the span drawing and composition. To optionally clip the spans, set * the @FT_RASTER_FLAG_CLIP flag and `clip_box`. The monochrome raster * does not support the direct mode. * * The gray-level rasterizer always uses 256 gray levels. If you want * fewer gray levels, you have to use @FT_RASTER_FLAG_DIRECT and reduce * the levels in the callback function. */ typedef struct FT_Raster_Params_ { const FT_Bitmap* target; const void* source; int flags; FT_SpanFunc gray_spans; FT_SpanFunc black_spans; /* unused */ FT_Raster_BitTest_Func bit_test; /* unused */ FT_Raster_BitSet_Func bit_set; /* unused */ void* user; FT_BBox clip_box; } FT_Raster_Params;

## *typedef x*

**Definition:**
* @struct: * FT_Span * * @description: * A structure to model a single span of consecutive pixels when * rendering an anti-aliased bitmap. * * @fields: * x :: * The span's horizontal start position. * * len :: * The span's length in pixels. * * coverage :: * The span color/coverage, ranging from 0 (background) to 255 * (foreground). * * @note: * This structure is used by the span drawing callback type named * @FT_SpanFunc that takes the y~coordinate of the span as a parameter. * * The anti-aliased rasterizer produces coverage values from 0 to 255, * that is, from completely transparent to completely opaque. */ typedef struct FT_Span_ { short x; unsigned short len; unsigned char coverage; } FT_Span;

# ftincrem.h

## *Data Types*

### *struct FT_IncrementalRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Long | bearing_x | |
| FT_Long | bearing_y | |
| FT_Long | advance | |
| FT_Long | advance_v | |

**Definition:**
typedef struct FT_IncrementalRec_* FT_Incremental;

### *struct FT_Incremental_FuncsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Incremental_GetGlyphDataFunc | get_glyph_data | |
| FT_Incremental_FreeGlyphDataFunc | free_glyph_data | |
| FT_Incremental_GetGlyphMetricsFunc | get_glyph_metrics | |

**Definition:**
typedef struct FT_Incremental_FuncsRec_ { FT_Incremental_GetGlyphDataFunc get_glyph_data; FT_Incremental_FreeGlyphDataFunc free_glyph_data; FT_Incremental_GetGlyphMetricsFunc get_glyph_metrics; } FT_Incremental_FuncsRec;

### *typedef FT_Incremental_Interface*

**Definition:**
typedef FT_Incremental_InterfaceRec* FT_Incremental_Interface;

### *struct FT_Incremental_InterfaceRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| const FT_Incremental_FuncsRec* | funcs | |
| FT_Incremental | object | |

**Definition:**
typedef struct FT_Incremental_InterfaceRec_ { const FT_Incremental_FuncsRec* funcs; FT_Incremental object; } FT_Incremental_InterfaceRec;

## *struct FT_Incremental_MetricsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Long | bearing_x | |
| FT_Long | bearing_y | |
| FT_Long | advance | |
| FT_Long | advance_v | |

**Definition:**

typedef struct FT_Incremental_MetricsRec_ { FT_Long bearing_x; FT_Long bearing_y; FT_Long advance; FT_Long advance_v; /* since 2.3.12 */ } FT_Incremental_MetricsRec;

## *struct FT_Incremental_MetricsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Incremental_GetGlyphDataFunc | get_glyph_data | |
| FT_Incremental_FreeGlyphDataFunc | free_glyph_data | |
| FT_Incremental_GetGlyphMetricsFunc | get_glyph_metrics | |

**Definition:**

typedef struct FT_Incremental_MetricsRec_* FT_Incremental_Metrics;

# ftlcdfil.h

## *Data Types*

### *typedef FT_LcdFilter*

**Definition:**
* ClearType-style LCD rendering exploits the color-striped structure of * LCD pixels, increasing the available resolution in the direction of * the stripe (usually horizontal RGB) by a factor of~3. Using the * subpixel coverages unfiltered can create severe color fringes * especially when rendering thin features. Indeed, to produce * black-on-white text, the nearby color subpixels must be dimmed * evenly. Therefore, an equalizing 5-tap FIR filter should be applied * to subpixel coverages regardless of pixel boundaries and should have * these properties: * * 1. It should be symmetrical, like {~a, b, c, b, a~}, to avoid * any shifts in appearance. * * 2. It should be color-balanced, meaning a~+ b~=~c, to reduce color * fringes by distributing the computed coverage for one subpixel to * all subpixels equally. * * 3. It should be normalized, meaning 2a~+ 2b~+ c~=~1.0 to maintain * overall brightness. * * Boxy 3-tap filter {0, 1/3, 1/3, 1/3, 0} is sharper but is less * forgiving of non-ideal gamma curves of a screen (and viewing angles), * beveled filters are fuzzier but more tolerant. * * Use the @FT_Library_SetLcdFilter or @FT_Library_SetLcdFilterWeights * API to specify a low-pass filter, which is then applied to * subpixel-rendered bitmaps generated through @FT_Render_Glyph. * * Harmony LCD rendering is suitable to panels with any regular subpixel * structure, not just monitors with 3 color striped subpixels, as long * as the color subpixels have fixed positions relative to the pixel * center. In this case, each color channel can be rendered separately * after shifting the outline opposite to the subpixel shift so that the * coverage maps are aligned. This method is immune to color fringes * because the shifts do not change integral coverage. * * The subpixel geometry must be specified by xy-coordinates for each * subpixel. By convention they may come in the RGB order: {{-1/3, 0}, * {0, 0}, {1/3, 0}} for standard RGB striped panel or {{-1/6, 1/4}, * {-1/6, -1/4}, {1/3, 0}} for a certain PenTile panel. * * Use the @FT_Library_SetLcdGeometry API to specify subpixel positions. * If one follows the RGB order convention, the same order applies to the * resulting @FT_PIXEL_MODE_LCD and @FT_PIXEL_MODE_LCD_V bitmaps. Note, * however, that the coordinate frame for the latter must be rotated * clockwise. Harmony with default LCD geometry is equivalent to * ClearType with light filter. * * As a result of ClearType filtering or Harmony shifts, the resulting * dimensions of LCD bitmaps can be slightly wider or taller than the * dimensions the original outline with regard to the pixel grid. * For example, for @FT_RENDER_MODE_LCD, the filter adds 2~subpixels to * the left, and 2~subpixels to the right. The bitmap offset values are * adjusted accordingly, so clients shouldn't need to modify their layout * and glyph positioning code when enabling the filter. * * The ClearType and Harmony rendering is applicable to glyph bitmaps * rendered through @FT_Render_Glyph, @FT_Load_Glyph, @FT_Load_Char, and * @FT_Glyph_To_Bitmap, when @FT_RENDER_MODE_LCD or @FT_RENDER_MODE_LCD_V * is specified. This API does not control @FT_Outline_Render and * @FT_Outline_Get_Bitmap. * * The described algorithms can completely remove color artefacts when * combined with gamma-corrected alpha blending in linear space. Each of * the 3~alpha values (subpixels) must by independently used to blend one * color channel. That is, red alpha blends the red channel of the text * color with the red channel of the background pixel. */
/************************************************************************** * * @enum: * FT_LcdFilter * * @description: * A list of values to identify various types of LCD filters. * * @values: * FT_LCD_FILTER_NONE :: * Do not perform filtering. When used with subpixel rendering, this * results in sometimes severe color fringes. * * FT_LCD_FILTER_DEFAULT :: * This is a beveled, normalized, and color-balanced five-tap filter * with weights of [0x08 0x4D 0x56 0x4D 0x08] in 1/256 units. * * FT_LCD_FILTER_LIGHT :: * this is a boxy, normalized, and color-balanced three-tap filter with * weights of [0x00 0x55 0x56 0x55 0x00] in 1/256 units. * * FT_LCD_FILTER_LEGACY :: * FT_LCD_FILTER_LEGACY1 :: * This filter corresponds to the original libXft color filter. It * provides

high contrast output but can exhibit really bad color * fringes if glyphs are not extremely well hinted to the pixel grid. * This filter is only provided for comparison purposes, and might be * disabled or stay unsupported in the future. The second value is * provided for compatibility with FontConfig, which historically used * different enumeration, sometimes incorrectly forwarded to FreeType. * * @since: * 2.3.0 (`FT_LCD_FILTER_LEGACY1` since 2.6.2) */ typedef enum FT_LcdFilter_ { FT_LCD_FILTER_NONE = 0, FT_LCD_FILTER_DEFAULT = 1, FT_LCD_FILTER_LIGHT = 2, FT_LCD_FILTER_LEGACY1 = 3, FT_LCD_FILTER_LEGACY = 16, FT_LCD_FILTER_MAX /* do not remove */ } FT_LcdFilter;

## *enum FT_LcdFilter_*

**Definition:**
typedef enum FT_LcdFilter_ { FT_LCD_FILTER_NONE = 0, FT_LCD_FILTER_DEFAULT = 1, FT_LCD_FILTER_LIGHT = 2, FT_LCD_FILTER_LEGACY1 = 3, FT_LCD_FILTER_LEGACY = 16, FT_LCD_FILTER_MAX /* do not remove */ } FT_LcdFilter;

# ftmm.h

## *Data Types*

### *struct FT_MM_Axis_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_String* | name | |
| FT_Long | minimum | |
| FT_Long | maximum | |

**Definition:**
typedef struct FT_MM_Axis_ { FT_String* name; FT_Long minimum; FT_Long maximum; } FT_MM_Axis;

### *struct FT_MM_Var_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UInt | num_axis | |
| FT_UInt | num_designs | |
| FT_UInt | num_namedstyles | |
| FT_Var_Axis* | axis | |
| FT_Var_Named_Style* | namedstyle | |

**Definition:**
typedef struct FT_MM_Var_ { FT_UInt num_axis; FT_UInt num_designs; FT_UInt num_namedstyles; FT_Var_Axis* axis; FT_Var_Named_Style* namedstyle; } FT_MM_Var;

### *struct FT_Multi_Master_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UInt | num_axis | |
| FT_UInt | num_designs | |
| FT_MM_Axis | axis | |

**Definition:**
typedef struct FT_Multi_Master_ { FT_UInt num_axis; FT_UInt num_designs; FT_MM_Axis axis[T1_MAX_MM_AXIS]; } FT_Multi_Master;

### *struct FT_Var_Axis_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_String* | name | |
| FT_Fixed | minimum | |
| FT_Fixed | def | |
| FT_Fixed | maximum | |
| FT_ULong | tag | |
| FT_UInt | strid | |

**Definition:**
typedef struct FT_Var_Axis_ { FT_String* name; FT_Fixed minimum; FT_Fixed def; FT_Fixed maximum; FT_ULong tag; FT_UInt strid; } FT_Var_Axis;

## *struct FT_Var_Named_Style_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Fixed* | coords | |
| FT_UInt | strid | |
| FT_UInt | psid | |

**Definition:**
typedef struct FT_Var_Named_Style_ { FT_Fixed* coords; FT_UInt strid; FT_UInt psid; /* since 2.7.1 */ } FT_Var_Named_Style;

## *typedef name*

**Definition:**
* @enum: * T1_MAX_MM_XXX * * @description: * Adobe MM font limits as defined in their specifications. * * @values: * T1_MAX_MM_AXIS :: * The maximum number of Adobe MM font axes. * * T1_MAX_MM_DESIGNS :: * The maximum number of Adobe MM font designs. * * T1_MAX_MM_MAP_POINTS :: * The maximum number of elements in a design map. * */ #define T1_MAX_MM_AXIS 4 #define T1_MAX_MM_DESIGNS 16 #define T1_MAX_MM_MAP_POINTS 20 /********** * * @struct: * FT_MM_Axis * * @description: * A structure to model a given axis in design space for Adobe MM fonts. * * This structure can't be used with TrueType GX or OpenType Font * Variations. * * @fields: * name :: * The axis's name. * * minimum :: * The axis's minimum design coordinate. * * maximum :: * The axis's maximum design coordinate. */ typedef struct FT_MM_Axis_ { FT_String* name; FT_Long minimum; FT_Long maximum; } FT_MM_Axis;

# ftmodapi.h

## *Data Types*

### *struct FT_Module_Class_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ULong | module_flags | |
| FT_Long | module_size | |
| const FT_String* | module_name | |
| FT_Fixed | module_version | |
| FT_Fixed | module_requires | |
| const void* | module_interface | |
| FT_Module_Constructor | module_init | |
| FT_Module_Destructor | module_done | |
| FT_Module_Requester | get_interface | |

**Definition:**

typedef struct FT_Module_Class_ { FT_ULong module_flags; FT_Long module_size; const FT_String* module_name; FT_Fixed module_version; FT_Fixed module_requires; const void* module_interface; FT_Module_Constructor module_init; FT_Module_Destructor module_done; FT_Module_Requester get_interface; } FT_Module_Class;

### *typedef FT_Module_Interface*

**Definition:**

* @FT_Module_Class structure. * * ``` * autofitter * bdf * cff * gxvalid * otvalid * pcf * pfr * psaux * pshinter * psnames * raster1 * sfnt * smooth * truetype * type1 * type42 * t1cid * winfonts * ``` * * Note that the FreeType Cache sub-system is not a FreeType module. * * @order: * FT_Module * FT_Module_Constructor * FT_Module_Destructor * FT_Module_Requester * FT_Module_Class * * FT_Add_Module * FT_Get_Module * FT_Remove_Module * FT_Add_Default_Modules * * FT_FACE_DRIVER_NAME * FT_Property_Set * FT_Property_Get * FT_Set_Default_Properties * * FT_New_Library * FT_Done_Library * FT_Reference_Library * * FT_Renderer * FT_Renderer_Class * * FT_Get_Renderer * FT_Set_Renderer * * FT_Set_Debug_Hook * */ /* module bit flags */ #define FT_MODULE_FONT_DRIVER 1 /* this module is a font driver */ #define FT_MODULE_RENDERER 2 /* this module is a renderer */ #define FT_MODULE_HINTER 4 /* this module is a glyph hinter */ #define FT_MODULE_STYLER 8 /* this module is a styler */ #define FT_MODULE_DRIVER_SCALABLE 0x100 /* the driver supports */ /* scalable fonts */ #define FT_MODULE_DRIVER_NO_OUTLINES 0x200 /* the driver does not */ /* support vector outlines */ #define FT_MODULE_DRIVER_HAS_HINTER 0x400 /* the driver provides its */ /* own hinter */ #define FT_MODULE_DRIVER_HINTS_LIGHTLY 0x800 /* the driver's hinter */ /* produces LIGHT hints */ /* deprecated values */ #define ft_module_font_driver FT_MODULE_FONT_DRIVER #define ft_module_renderer FT_MODULE_RENDERER #define ft_module_hinter FT_MODULE_HINTER #define ft_module_styler FT_MODULE_STYLER #define ft_module_driver_scalable FT_MODULE_DRIVER_SCALABLE #define ft_module_driver_no_outlines FT_MODULE_DRIVER_NO_OUTLINES #define ft_module_driver_has_hinter

FT_MODULE_DRIVER_HAS_HINTER #define ft_module_driver_hints_lightly FT_MODULE_DRIVER_HINTS_LIGHTLY typedef FT_Pointer FT_Module_Interface;

## *typedef FT_TrueTypeEngineType*

**Definition:**
* @enum: * FT_TrueTypeEngineType * * @description: * A list of values describing which kind of TrueType bytecode engine is * implemented in a given FT_Library instance. It is used by the * @FT_Get_TrueType_Engine_Type function. * * @values: * FT_TRUETYPE_ENGINE_TYPE_NONE :: * The library doesn't implement any kind of bytecode interpreter. * * FT_TRUETYPE_ENGINE_TYPE_UNPATENTED :: * Deprecated and removed. * * FT_TRUETYPE_ENGINE_TYPE_PATENTED :: * The library implements a bytecode interpreter that covers the full * instruction set of the TrueType virtual machine (this was governed * by patents until May 2010, hence the name). * * @since: * 2.2 * */ typedef enum FT_TrueTypeEngineType_ { FT_TRUETYPE_ENGINE_TYPE_NONE = 0, FT_TRUETYPE_ENGINE_TYPE_UNPATENTED, FT_TRUETYPE_ENGINE_TYPE_PATENTED } FT_TrueTypeEngineType;

# ftoutln.h

## *Data Types*

### *typedef FT_Orientation*

**Definition:**
* @enum: * FT_Orientation * * @description: * A list of values used to describe an outline's contour orientation. * * The TrueType and PostScript specifications use different conventions * to determine whether outline contours should be filled or unfilled. * * @values: * FT_ORIENTATION_TRUETYPE :: * According to the TrueType specification, clockwise contours must be * filled, and counter-clockwise ones must be unfilled. * * FT_ORIENTATION_POSTSCRIPT :: * According to the PostScript specification, counter-clockwise * contours must be filled, and clockwise ones must be unfilled. * * FT_ORIENTATION_FILL_RIGHT :: * This is identical to @FT_ORIENTATION_TRUETYPE, but is used to * remember that in TrueType, everything that is to the right of the * drawing direction of a contour must be filled. * * FT_ORIENTATION_FILL_LEFT :: * This is identical to @FT_ORIENTATION_POSTSCRIPT, but is used to * remember that in PostScript, everything that is to the left of the * drawing direction of a contour must be filled. * * FT_ORIENTATION_NONE :: * The orientation cannot be determined. That is, different parts of * the glyph have different orientation. * */ typedef enum FT_Orientation_ { FT_ORIENTATION_TRUETYPE = 0, FT_ORIENTATION_POSTSCRIPT = 1, FT_ORIENTATION_FILL_RIGHT = FT_ORIENTATION_TRUETYPE, FT_ORIENTATION_FILL_LEFT = FT_ORIENTATION_POSTSCRIPT, FT_ORIENTATION_NONE } FT_Orientation;

# ftrender.h

## *Data Types*

### *struct FT_Renderer_Class_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Module_Class | root | |
| FT_Glyph_Format | glyph_format | |
| FT_Renderer_RenderFunc | render_glyph | |
| FT_Renderer_TransformFunc | transform_glyph | |
| FT_Renderer_GetCBoxFunc | get_glyph_cbox | |
| FT_Renderer_SetModeFunc | set_mode | |
| const FT_Raster_Funcs* | raster_class | |

**Definition:**

typedef struct FT_Renderer_Class_ { FT_Module_Class root; FT_Glyph_Format glyph_format; FT_Renderer_RenderFunc render_glyph; FT_Renderer_TransformFunc transform_glyph; FT_Renderer_GetCBoxFunc get_glyph_cbox; FT_Renderer_SetModeFunc set_mode; const FT_Raster_Funcs* raster_class; } FT_Renderer_Class;

# ftsnames.h

## *Data Types*

### *struct FT_SfntName_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | platform_id | |
| FT_UShort | encoding_id | |
| FT_UShort | language_id | |
| FT_UShort | name_id | |
| FT_Byte* | string | |
| FT_UInt | string_len | |

**Definition:**
typedef struct FT_SfntName_ { FT_UShort platform_id; FT_UShort encoding_id; FT_UShort language_id; FT_UShort name_id; FT_Byte* string; /* this string is *not* null-terminated! */ FT_UInt string_len; /* in bytes */ } FT_SfntName;

### *typedef string*

**Definition:**
* @struct: * FT_SfntLangTag * * @description: * A structure to model a language tag entry from an SFNT 'name' table. * * @fields: * string :: * The language tag string, encoded in UTF-16BE (without trailing * `NULL` bytes). * * string_len :: * The length of `string` in **bytes**. * * @note: * Please refer to the TrueType or OpenType specification for more * details. * * @since: * 2.8 */ typedef struct FT_SfntLangTag_ { FT_Byte* string; /* this string is *not* null-terminated! */ FT_UInt string_len; /* in bytes */ } FT_SfntLangTag;

# ftstroke.h

## *Data Types*

### *enum FT_StrokerBorder_*

**Definition:**
typedef enum FT_StrokerBorder_ { FT_STROKER_BORDER_LEFT = 0,
FT_STROKER_BORDER_RIGHT } FT_StrokerBorder;

### *struct FT_StrokerRec_*

**Definition:**
typedef struct FT_StrokerRec_* FT_Stroker;

### *enum FT_Stroker_LineCap_*

**Definition:**
typedef enum FT_Stroker_LineCap_ { FT_STROKER_LINECAP_BUTT = 0,
FT_STROKER_LINECAP_ROUND, FT_STROKER_LINECAP_SQUARE } FT_Stroker_LineCap;

### *enum FT_Stroker_LineJoin_*

**Definition:**
typedef enum FT_Stroker_LineJoin_ { FT_STROKER_LINEJOIN_ROUND = 0,
FT_STROKER_LINEJOIN_BEVEL = 1, FT_STROKER_LINEJOIN_MITER_VARIABLE = 2,
FT_STROKER_LINEJOIN_MITER = FT_STROKER_LINEJOIN_MITER_VARIABLE,
FT_STROKER_LINEJOIN_MITER_FIXED = 3 } FT_Stroker_LineJoin;

# ftsystem.h

## *Data Types*

### *typedef FT_Memory*

**Definition:**
* @FT_MemoryRec structure. * */ typedef struct FT_MemoryRec_* FT_Memory;

### *struct FT_MemoryRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| void* | user | |
| FT_Alloc_Func | alloc | |
| FT_Free_Func | free | |
| FT_Realloc_Func | realloc | |

**Definition:**
struct FT_MemoryRec_ { void* user; FT_Alloc_Func alloc; FT_Free_Func free; FT_Realloc_Func realloc; };

### *union FT_StreamDesc_*

**Definition:**
typedef union FT_StreamDesc_ { long value; void* pointer; } FT_StreamDesc;

### *struct FT_StreamRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| long | value | |
| void* | pointer | |

**Definition:**
typedef struct FT_StreamRec_* FT_Stream;

### *struct FT_StreamRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| unsigned char* | base | |
| unsigned long | size | |
| unsigned long | pos | |

| | | |
|---|---|---|
| FT_StreamDesc | descriptor | |
| FT_StreamDesc | pathname | |
| FT_Stream_IoFunc | read | |
| FT_Stream_CloseFunc | close | |
| FT_Memory | memory | |
| unsigned char* | cursor | |
| unsigned char* | limit | |

**Definition:**

typedef struct FT_StreamRec_ { unsigned char* base; unsigned long size; unsigned long pos; FT_StreamDesc descriptor; FT_StreamDesc pathname; FT_Stream_IoFunc read; FT_Stream_CloseFunc close; FT_Memory memory; unsigned char* cursor; unsigned char* limit; } FT_StreamRec;

# fttrigon.h

## *Data Types*

### *typedef FT_Angle*

**Definition:**
typedef FT_Fixed FT_Angle;

# fttypes.h

## *Data Types*

### *typedef FT_Bool*

**Definition:**
* font-specific structures are defined in a different section. Note * that FreeType does not use floating-point data types. Fractional * values are represented by fixed-point integers, with lower bits * storing the fractional part. * * @order: * FT_Byte * FT_Bytes * FT_Char * FT_Int * FT_UInt * FT_Int16 * FT_UInt16 * FT_Int32 * FT_UInt32 * FT_Int64 * FT_UInt64 * FT_Short * FT_UShort * FT_Long * FT_ULong * FT_Bool * FT_Offset * FT_PtrDist * FT_String * FT_Tag * FT_Error * FT_Fixed * FT_Pointer * FT_Pos * FT_Vector * FT_BBox * FT_Matrix * FT_FWord * FT_UFWord * FT_F2Dot14 * FT_UnitVector * FT_F26Dot6 * FT_Data * * FT_MAKE_TAG * * FT_Generic * FT_Generic_Finalizer * * FT_Bitmap * FT_Pixel_Mode * FT_Palette_Mode * FT_Glyph_Format * FT_IMAGE_TAG * */ /************************************************************************** * * @type: * FT_Bool * * @description: * A typedef of unsigned char, used for simple booleans. As usual, * values 1 and~0 represent true and false, respectively. */ typedef unsigned char FT_Bool;

### *typedef FT_Bool*

**Definition:**
typedef unsigned char FT_Bool;

### *typedef FT_Byte*

**Definition:**
typedef unsigned char FT_Byte;

### *typedef FT_Bytes*

**Definition:**
typedef const FT_Byte* FT_Bytes;

### *typedef FT_Char*

**Definition:**
typedef signed char FT_Char;

### *struct FT_Data_*

**Members:**

| Type | Name | Description |
|---|---|---|
| const FT_Byte* | pointer | |
| FT_UInt | length | |

**Definition:**

typedef struct FT_Data_ { const FT_Byte* pointer; FT_UInt length; } FT_Data;

### *typedef FT_Error*

**Definition:**
typedef int FT_Error;

### *typedef FT_F26Dot6*

**Definition:**
typedef signed long FT_F26Dot6;

### *typedef FT_F2Dot14*

**Definition:**
typedef signed short FT_F2Dot14;

### *typedef FT_FWord*

**Definition:**
typedef signed short FT_FWord; /* distance in FUnits */

### *typedef FT_Fixed*

**Definition:**
typedef signed long FT_Fixed;

### *struct FT_Generic_*

**Members:**

| Type | Name | Description |
|---|---|---|
| void* | data | |
| FT_Generic_Finalizer | finalizer | |

**Definition:**
typedef struct FT_Generic_ { void* data; FT_Generic_Finalizer finalizer; } FT_Generic;

### *typedef FT_Int*

**Definition:**
typedef signed int FT_Int;

### *struct FT_ListNodeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|

| | | |
|---|---|---|
| FT_ListNode | prev | |
| FT_ListNode | next | |
| void* | data | |

**Definition:**
typedef struct FT_ListNodeRec_* FT_ListNode;

## *struct FT_ListNodeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ListNode | prev | |
| FT_ListNode | next | |
| void* | data | |

**Definition:**
typedef struct FT_ListNodeRec_ { FT_ListNode prev; FT_ListNode next; void* data; } FT_ListNodeRec;

## *struct FT_ListRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ListNode | prev | |
| FT_ListNode | next | |
| void* | data | |

**Definition:**
typedef struct FT_ListRec_* FT_List;

## *struct FT_ListRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ListNode | head | |
| FT_ListNode | tail | |

**Definition:**
typedef struct FT_ListRec_ { FT_ListNode head; FT_ListNode tail; } FT_ListRec;

## *typedef FT_Long*

**Definition:**
typedef signed long FT_Long;

## *struct FT_Matrix_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Fixed xx, | xy | |
| FT_Fixed yx, | yy | |

**Definition:**
typedef struct FT_Matrix_ { FT_Fixed xx, xy; FT_Fixed yx, yy; } FT_Matrix;

## *typedef FT_Offset*

**Definition:**
typedef size_t FT_Offset;

## *typedef FT_Pointer*

**Definition:**
typedef void* FT_Pointer;

## *typedef FT_PtrDist*

**Definition:**
typedef ft_ptrdiff_t FT_PtrDist;

## *typedef FT_Short*

**Definition:**
typedef signed short FT_Short;

## *typedef FT_String*

**Definition:**
typedef char FT_String;

## *typedef FT_Tag*

**Definition:**
typedef FT_UInt32 FT_Tag;

## *typedef FT_UFWord*

**Definition:**
typedef unsigned short FT_UFWord; /* unsigned distance */

## *typedef FT_UInt*

**Definition:**
typedef unsigned int FT_UInt;

### *typedef FT_ULong*

**Definition:**
typedef unsigned long FT_ULong;

### *typedef FT_UShort*

**Definition:**
typedef unsigned short FT_UShort;

### *struct FT_UnitVector_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_F2Dot14 | x | |
| FT_F2Dot14 | y | |

**Definition:**
typedef struct FT_UnitVector_ { FT_F2Dot14 x; FT_F2Dot14 y; } FT_UnitVector;

# ftwinfnt.h

## *Data Types*

### *struct FT_WinFNT_HeaderRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | version | |
| FT_ULong | file_size | |
| FT_Byte | copyright | |
| FT_UShort | file_type | |
| FT_UShort | nominal_point_size | |
| FT_UShort | vertical_resolution | |
| FT_UShort | horizontal_resolution | |
| FT_UShort | ascent | |
| FT_UShort | internal_leading | |
| FT_UShort | external_leading | |
| FT_Byte | italic | |
| FT_Byte | underline | |
| FT_Byte | strike_out | |
| FT_UShort | weight | |
| FT_Byte | charset | |
| FT_UShort | pixel_width | |
| FT_UShort | pixel_height | |
| FT_Byte | pitch_and_family | |
| FT_UShort | avg_width | |
| FT_UShort | max_width | |
| FT_Byte | first_char | |
| FT_Byte | last_char | |
| FT_Byte | default_char | |
| FT_Byte | break_char | |
| FT_UShort | bytes_per_row | |
| FT_ULong | device_offset | |
| FT_ULong | face_name_offset | |
| FT_ULong | bits_pointer | |

| | | |
|---|---|---|
| FT_ULong | bits_offset | |
| FT_Byte | reserved | |
| FT_ULong | flags | |
| FT_UShort | A_space | |
| FT_UShort | B_space | |
| FT_UShort | C_space | |
| FT_UShort | color_table_offset | |
| FT_ULong | reserved1 | |

**Definition:**
typedef struct FT_WinFNT_HeaderRec_ { FT_UShort version; FT_ULong file_size; FT_Byte copyright[60]; FT_UShort file_type; FT_UShort nominal_point_size; FT_UShort vertical_resolution; FT_UShort horizontal_resolution; FT_UShort ascent; FT_UShort internal_leading; FT_UShort external_leading; FT_Byte italic; FT_Byte underline; FT_Byte strike_out; FT_UShort weight; FT_Byte charset; FT_UShort pixel_width; FT_UShort pixel_height; FT_Byte pitch_and_family; FT_UShort avg_width; FT_UShort max_width; FT_Byte first_char; FT_Byte last_char; FT_Byte default_char; FT_Byte break_char; FT_UShort bytes_per_row; FT_ULong device_offset; FT_ULong face_name_offset; FT_ULong bits_pointer; FT_ULong bits_offset; FT_Byte reserved; FT_ULong flags; FT_UShort A_space; FT_UShort B_space; FT_UShort C_space; FT_UShort color_table_offset; FT_ULong reserved1[4]; } FT_WinFNT_HeaderRec;

## *struct FT_WinFNT_HeaderRec_*

**Definition:**
typedef struct FT_WinFNT_HeaderRec_* FT_WinFNT_Header;

## *typedef version*

**Definition:**
* @enum: * FT_WinFNT_ID_XXX * * @description: * A list of valid values for the `charset` byte in @FT_WinFNT_HeaderRec. * Exact mapping tables for the various 'cpXXXX' encodings (except for * 'cp1361') can be found at 'ftp://ftp.unicode.org/Public/' in the * `MAPPINGS/VENDORS/MICSFT/WINDOWS` subdirectory. 'cp1361' is roughly a * superset of `MAPPINGS/OBSOLETE/EASTASIA/KSC/JOHAB.TXT`. * * @values: * FT_WinFNT_ID_DEFAULT :: * This is used for font enumeration and font creation as a 'don't * care' value. Valid font files don't contain this value. When * querying for information about the character set of the font that is * currently selected into a specified device context, this return * value (of the related Windows API) simply denotes failure. * * FT_WinFNT_ID_SYMBOL :: * There is no known mapping table available. * * FT_WinFNT_ID_MAC :: * Mac Roman encoding. * * FT_WinFNT_ID_OEM :: * From Michael Poettgen : * * The 'Windows Font Mapping' article says that `FT_WinFNT_ID_OEM` is * used for the charset of vector fonts, like `modern.fon`, * `roman.fon`, and `script.fon` on Windows. * * The 'CreateFont' documentation says: The `FT_WinFNT_ID_OEM` value * specifies a character set that is operating-system dependent. * * The 'IFIMETRICS' documentation from the 'Windows Driver Development * Kit' says: This font supports an OEM-specific character set. The * OEM character set is system dependent. * * In general OEM, as opposed to ANSI (i.e., 'cp1252'), denotes the * second default codepage that most international versions of Windows * have. It is one of the OEM codepages from * * https://learn.microsoft.com/windows/win32/intl/code-page-identifiers , * * and is used for the 'DOS boxes', to support legacy applications. A * German Windows version for example usually uses ANSI

codepage 1252 * and OEM codepage 850. * * FT_WinFNT_ID_CP874 :: * A superset of Thai TIS 620 and ISO 8859-11. * * FT_WinFNT_ID_CP932 :: * A superset of Japanese Shift-JIS (with minor deviations). * * FT_WinFNT_ID_CP936 :: * A superset of simplified Chinese GB 2312-1980 (with different * ordering and minor deviations). * * FT_WinFNT_ID_CP949 :: * A superset of Korean Hangul KS~C 5601-1987 (with different ordering * and minor deviations). * * FT_WinFNT_ID_CP950 :: * A superset of traditional Chinese Big~5 ETen (with different * ordering and minor deviations). * * FT_WinFNT_ID_CP1250 :: * A superset of East European ISO 8859-2 (with slightly different * ordering). * * FT_WinFNT_ID_CP1251 :: * A superset of Russian ISO 8859-5 (with different ordering). * * FT_WinFNT_ID_CP1252 :: * ANSI encoding. A superset of ISO 8859-1. * * FT_WinFNT_ID_CP1253 :: * A superset of Greek ISO 8859-7 (with minor modifications). * * FT_WinFNT_ID_CP1254 :: * A superset of Turkish ISO 8859-9. * * FT_WinFNT_ID_CP1255 :: * A superset of Hebrew ISO 8859-8 (with some modifications). * * FT_WinFNT_ID_CP1256 :: * A superset of Arabic ISO 8859-6 (with different ordering). * * FT_WinFNT_ID_CP1257 :: * A superset of Baltic ISO 8859-13 (with some deviations). * * FT_WinFNT_ID_CP1258 :: * For Vietnamese. This encoding doesn't cover all necessary * characters. * * FT_WinFNT_ID_CP1361 :: * Korean (Johab). */ #define FT_WinFNT_ID_CP1252 0 #define FT_WinFNT_ID_DEFAULT 1 #define FT_WinFNT_ID_SYMBOL 2 #define FT_WinFNT_ID_MAC 77 #define FT_WinFNT_ID_CP932 128 #define FT_WinFNT_ID_CP949 129 #define FT_WinFNT_ID_CP1361 130 #define FT_WinFNT_ID_CP936 134 #define FT_WinFNT_ID_CP950 136 #define FT_WinFNT_ID_CP1253 161 #define FT_WinFNT_ID_CP1254 162 #define FT_WinFNT_ID_CP1258 163 #define FT_WinFNT_ID_CP1255 177 #define FT_WinFNT_ID_CP1256 178 #define FT_WinFNT_ID_CP1257 186 #define FT_WinFNT_ID_CP1251 204 #define FT_WinFNT_ID_CP874 222 #define FT_WinFNT_ID_CP1250 238 #define FT_WinFNT_ID_OEM 255 /********************************************************************** * * @struct: * FT_WinFNT_HeaderRec * * @description: * Windows FNT Header info. */ typedef struct FT_WinFNT_HeaderRec_ { FT_UShort version; FT_ULong file_size; FT_Byte copyright[60]; FT_UShort file_type; FT_UShort nominal_point_size; FT_UShort vertical_resolution; FT_UShort horizontal_resolution; FT_UShort ascent; FT_UShort internal_leading; FT_UShort external_leading; FT_Byte italic; FT_Byte underline; FT_Byte strike_out; FT_UShort weight; FT_Byte charset; FT_UShort pixel_width; FT_UShort pixel_height; FT_Byte pitch_and_family; FT_UShort avg_width; FT_UShort max_width; FT_Byte first_char; FT_Byte last_char; FT_Byte default_char; FT_Byte break_char; FT_UShort bytes_per_row; FT_ULong device_offset; FT_ULong face_name_offset; FT_ULong bits_pointer; FT_ULong bits_offset; FT_Byte reserved; FT_ULong flags; FT_UShort A_space; FT_UShort B_space; FT_UShort C_space; FT_UShort color_table_offset; FT_ULong reserved1[4]; } FT_WinFNT_HeaderRec;

# otsvg.h

## *Data Types*

### *struct FT_SVG_DocumentRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte* | svg_document | |
| FT_ULong | svg_document_length | |
| FT_Size_Metrics | metrics | |
| FT_UShort | units_per_EM | |
| FT_UShort | start_glyph_id | |
| FT_UShort | end_glyph_id | |
| FT_Matrix | transform | |
| FT_Vector | delta | |

**Definition:**
typedef struct FT_SVG_DocumentRec_ { FT_Byte* svg_document; FT_ULong svg_document_length; FT_Size_Metrics metrics; FT_UShort units_per_EM; FT_UShort start_glyph_id; FT_UShort end_glyph_id; FT_Matrix transform; FT_Vector delta; } FT_SVG_DocumentRec;

### *struct FT_SVG_DocumentRec_*

**Definition:**
typedef struct FT_SVG_DocumentRec_* FT_SVG_Document;

### *struct SVG_RendererHooks_*

**Members:**

| Type | Name | Description |
|---|---|---|
| SVG_Lib_Init_Func | init_svg | |
| SVG_Lib_Free_Func | free_svg | |
| SVG_Lib_Render_Func | render_svg | |
| SVG_Lib_Preset_Slot_Func | preset_slot | |

**Definition:**
typedef struct SVG_RendererHooks_ { SVG_Lib_Init_Func init_svg; SVG_Lib_Free_Func free_svg; SVG_Lib_Render_Func render_svg; SVG_Lib_Preset_Slot_Func preset_slot; } SVG_RendererHooks;

# t1tables.h

## *Data Types*

### *struct CID_FaceDictRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| PS_PrivateRec | private_dict | |
| FT_UInt | len_buildchar | |
| FT_Fixed | forcebold_threshold | |
| FT_Pos | stroke_width | |
| FT_Fixed | expansion_factor | |
| FT_Byte | paint_type | |
| FT_Byte | font_type | |
| FT_Matrix | font_matrix | |
| FT_Vector | font_offset | |
| FT_UInt | num_subrs | |
| FT_ULong | subrmap_offset | |
| FT_UInt | sd_bytes | |

**Definition:**

typedef struct CID_FaceDictRec_ { PS_PrivateRec private_dict; FT_UInt len_buildchar; FT_Fixed forcebold_threshold; FT_Pos stroke_width; FT_Fixed expansion_factor; /* this is a duplicate of */ /* `private_dict->expansion_factor' */ FT_Byte paint_type; FT_Byte font_type; FT_Matrix font_matrix; FT_Vector font_offset; FT_UInt num_subrs; FT_ULong subrmap_offset; FT_UInt sd_bytes; } CID_FaceDictRec;

### *struct CID_FaceDictRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_String* | cid_font_name | |
| FT_Fixed | cid_version | |
| FT_Int | cid_font_type | |
| FT_String* | registry | |
| FT_String* | ordering | |
| FT_Int | supplement | |
| PS_FontInfoRec | font_info | |
| FT_BBox | font_bbox | |

| | | |
|---|---|---|
| FT_ULong | uid_base | |
| FT_Int | num_xuid | |
| FT_ULong | xuid | |
| FT_ULong | cidmap_offset | |
| FT_UInt | fd_bytes | |
| FT_UInt | gd_bytes | |
| FT_ULong | cid_count | |
| FT_UInt | num_dicts | |
| CID_FaceDict | font_dicts | |
| FT_ULong | data_offset | |

**Definition:**
typedef struct CID_FaceDictRec_* CID_FaceDict;

## *struct CID_FaceInfoRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_String* | cid_font_name | |
| FT_Fixed | cid_version | |
| FT_Int | cid_font_type | |
| FT_String* | registry | |
| FT_String* | ordering | |
| FT_Int | supplement | |
| PS_FontInfoRec | font_info | |
| FT_BBox | font_bbox | |
| FT_ULong | uid_base | |
| FT_Int | num_xuid | |
| FT_ULong | xuid | |
| FT_ULong | cidmap_offset | |
| FT_UInt | fd_bytes | |
| FT_UInt | gd_bytes | |
| FT_ULong | cid_count | |
| FT_UInt | num_dicts | |
| CID_FaceDict | font_dicts | |
| FT_ULong | data_offset | |

**Definition:**

typedef struct CID_FaceInfoRec_ { FT_String* cid_font_name; FT_Fixed cid_version; FT_Int cid_font_type; FT_String* registry; FT_String* ordering; FT_Int supplement; PS_FontInfoRec font_info; FT_BBox font_bbox; FT_ULong uid_base; FT_Int num_xuid; FT_ULong xuid[16]; FT_ULong cidmap_offset; FT_UInt fd_bytes; FT_UInt gd_bytes; FT_ULong cid_count; FT_UInt num_dicts; CID_FaceDict font_dicts; FT_ULong data_offset; } CID_FaceInfoRec;

### *struct CID_FaceInfoRec_*

**Definition:**
typedef struct CID_FaceInfoRec_* CID_FaceInfo;

### *typedef CID_FontDict*

**Definition:**
typedef CID_FaceDictRec CID_FontDict;

### *typedef CID_Info*

**Definition:**
typedef CID_FaceInfoRec CID_Info;

### *enum PS_Dict_Keys_*

**Definition:**
typedef enum PS_Dict_Keys_ { /* conventionally in the font dictionary */ PS_DICT_FONT_TYPE, /* FT_Byte */ PS_DICT_FONT_MATRIX, /* FT_Fixed */ PS_DICT_FONT_BBOX, /* FT_Fixed */ PS_DICT_PAINT_TYPE, /* FT_Byte */ PS_DICT_FONT_NAME, /* FT_String* */ PS_DICT_UNIQUE_ID, /* FT_Int */ PS_DICT_NUM_CHAR_STRINGS, /* FT_Int */ PS_DICT_CHAR_STRING_KEY, /* FT_String* */ PS_DICT_CHAR_STRING, /* FT_String* */ PS_DICT_ENCODING_TYPE, /* T1_EncodingType */ PS_DICT_ENCODING_ENTRY, /* FT_String* */ /* conventionally in the font Private dictionary */ PS_DICT_NUM_SUBRS, /* FT_Int */ PS_DICT_SUBR, /* FT_String* */ PS_DICT_STD_HW, /* FT_UShort */ PS_DICT_STD_VW, /* FT_UShort */ PS_DICT_NUM_BLUE_VALUES, /* FT_Byte */ PS_DICT_BLUE_VALUE, /* FT_Short */ PS_DICT_BLUE_FUZZ, /* FT_Int */ PS_DICT_NUM_OTHER_BLUES, /* FT_Byte */ PS_DICT_OTHER_BLUE, /* FT_Short */ PS_DICT_NUM_FAMILY_BLUES, /* FT_Byte */ PS_DICT_FAMILY_BLUE, /* FT_Short */ PS_DICT_NUM_FAMILY_OTHER_BLUES, /* FT_Byte */ PS_DICT_FAMILY_OTHER_BLUE, /* FT_Short */ PS_DICT_BLUE_SCALE, /* FT_Fixed */ PS_DICT_BLUE_SHIFT, /* FT_Int */ PS_DICT_NUM_STEM_SNAP_H, /* FT_Byte */ PS_DICT_STEM_SNAP_H, /* FT_Short */ PS_DICT_NUM_STEM_SNAP_V, /* FT_Byte */ PS_DICT_STEM_SNAP_V, /* FT_Short */ PS_DICT_FORCE_BOLD, /* FT_Bool */ PS_DICT_RND_STEM_UP, /* FT_Bool */ PS_DICT_MIN_FEATURE, /* FT_Short */ PS_DICT_LEN_IV, /* FT_Int */ PS_DICT_PASSWORD, /* FT_Long */ PS_DICT_LANGUAGE_GROUP, /* FT_Long */ /* conventionally in the font FontInfo dictionary */ PS_DICT_VERSION, /* FT_String* */ PS_DICT_NOTICE, /* FT_String* */ PS_DICT_FULL_NAME, /* FT_String* */ PS_DICT_FAMILY_NAME, /* FT_String* */ PS_DICT_WEIGHT, /* FT_String* */ PS_DICT_IS_FIXED_PITCH, /* FT_Bool */ PS_DICT_UNDERLINE_POSITION, /* FT_Short */ PS_DICT_UNDERLINE_THICKNESS, /* FT_UShort */ PS_DICT_FS_TYPE, /* FT_UShort */ PS_DICT_ITALIC_ANGLE, /* FT_Long */ PS_DICT_MAX = PS_DICT_ITALIC_ANGLE } PS_Dict_Keys;

## *struct PS_FontInfoRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_String* | version | |
| FT_String* | notice | |
| FT_String* | full_name | |
| FT_String* | family_name | |
| FT_String* | weight | |
| FT_Long | italic_angle | |
| FT_Bool | is_fixed_pitch | |
| FT_Short | underline_position | |
| FT_UShort | underline_thickness | |

**Definition:**

typedef struct PS_FontInfoRec_ { FT_String* version; FT_String* notice; FT_String* full_name; FT_String* family_name; FT_String* weight; FT_Long italic_angle; FT_Bool is_fixed_pitch; FT_Short underline_position; FT_UShort underline_thickness; } PS_FontInfoRec;

## *struct PS_FontInfoRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Int | unique_id | |
| FT_Int | lenIV | |
| FT_Byte | num_blue_values | |
| FT_Byte | num_other_blues | |
| FT_Byte | num_family_blues | |
| FT_Byte | num_family_other_blues | |
| FT_Short | blue_values | |
| FT_Short | other_blues | |
| FT_Short family_blues | | |
| FT_Short | family_other_blues | |
| FT_Fixed | blue_scale | |
| FT_Int | blue_shift | |
| FT_Int | blue_fuzz | |
| FT_UShort | standard_width | |
| FT_UShort | standard_height | |
| FT_Byte | num_snap_widths | |

| | | |
|---|---|---|
| FT_Byte | num_snap_heights | |
| FT_Bool | force_bold | |
| FT_Bool | round_stem_up | |
| FT_Short snap_widths | | |
| FT_Short | snap_heights | |
| FT_Fixed | expansion_factor | |
| FT_Long | language_group | |
| FT_Long | password | |
| FT_Short | min_feature | |

**Definition:**

typedef struct PS_FontInfoRec_* PS_FontInfo;

## *struct PS_PrivateRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Int | unique_id | |
| FT_Int | lenIV | |
| FT_Byte | num_blue_values | |
| FT_Byte | num_other_blues | |
| FT_Byte | num_family_blues | |
| FT_Byte | num_family_other_blues | |
| FT_Short | blue_values | |
| FT_Short | other_blues | |
| FT_Short family_blues | | |
| FT_Short | family_other_blues | |
| FT_Fixed | blue_scale | |
| FT_Int | blue_shift | |
| FT_Int | blue_fuzz | |
| FT_UShort | standard_width | |
| FT_UShort | standard_height | |
| FT_Byte | num_snap_widths | |
| FT_Byte | num_snap_heights | |
| FT_Bool | force_bold | |
| FT_Bool | round_stem_up | |
| FT_Short snap_widths | | |

| | | |
|---|---|---|
| FT_Short | snap_heights | |
| FT_Fixed | expansion_factor | |
| FT_Long | language_group | |
| FT_Long | password | |
| FT_Short | min_feature | |

**Definition:**
typedef struct PS_PrivateRec_ { FT_Int unique_id; FT_Int lenIV; FT_Byte num_blue_values; FT_Byte num_other_blues; FT_Byte num_family_blues; FT_Byte num_family_other_blues; FT_Short blue_values[14]; FT_Short other_blues[10]; FT_Short family_blues [14]; FT_Short family_other_blues[10]; FT_Fixed blue_scale; FT_Int blue_shift; FT_Int blue_fuzz; FT_UShort standard_width[1]; FT_UShort standard_height[1]; FT_Byte num_snap_widths; FT_Byte num_snap_heights; FT_Bool force_bold; FT_Bool round_stem_up; FT_Short snap_widths [13]; /* including std width */ FT_Short snap_heights[13]; /* including std height */ FT_Fixed expansion_factor; FT_Long language_group; FT_Long password; FT_Short min_feature[2]; } PS_PrivateRec;

## *struct PS_PrivateRec_*

**Definition:**
typedef struct PS_PrivateRec_* PS_Private;

## *enum T1_Blend_Flags_*

**Definition:**
typedef enum T1_Blend_Flags_ { /* required fields in a FontInfo blend dictionary */ T1_BLEND_UNDERLINE_POSITION = 0, T1_BLEND_UNDERLINE_THICKNESS, T1_BLEND_ITALIC_ANGLE, /* required fields in a Private blend dictionary */ T1_BLEND_BLUE_VALUES, T1_BLEND_OTHER_BLUES, T1_BLEND_STANDARD_WIDTH, T1_BLEND_STANDARD_HEIGHT, T1_BLEND_STEM_SNAP_WIDTHS, T1_BLEND_STEM_SNAP_HEIGHTS, T1_BLEND_BLUE_SCALE, T1_BLEND_BLUE_SHIFT, T1_BLEND_FAMILY_BLUES, T1_BLEND_FAMILY_OTHER_BLUES, T1_BLEND_FORCE_BOLD, T1_BLEND_MAX /* do not remove */ } T1_Blend_Flags;

## *typedef T1_EncodingType*

**Definition:**
* @enum: * T1_EncodingType * * @description: * An enumeration describing the 'Encoding' entry in a Type 1 dictionary. * * @values: * T1_ENCODING_TYPE_NONE :: * T1_ENCODING_TYPE_ARRAY :: * T1_ENCODING_TYPE_STANDARD :: * T1_ENCODING_TYPE_ISOLATIN1 :: * T1_ENCODING_TYPE_EXPERT :: * * @since: * 2.4.8 */ typedef enum T1_EncodingType_ { T1_ENCODING_TYPE_NONE = 0, T1_ENCODING_TYPE_ARRAY, T1_ENCODING_TYPE_STANDARD, T1_ENCODING_TYPE_ISOLATIN1, T1_ENCODING_TYPE_EXPERT } T1_EncodingType;

## *typedef T1_FontInfo*

**Definition:**
typedef PS_FontInfoRec T1_FontInfo;

### *typedef T1_Private*

**Definition:**
typedef PS_PrivateRec T1_Private;

### *typedef version*

**Definition:**
* including structures related to other PostScript font formats. * * @order: * PS_FontInfoRec * PS_FontInfo * PS_PrivateRec * PS_Private * * CID_FaceDictRec * CID_FaceDict * CID_FaceInfoRec * CID_FaceInfo * * FT_Has_PS_Glyph_Names * FT_Get_PS_Font_Info * FT_Get_PS_Font_Private * FT_Get_PS_Font_Value * * T1_Blend_Flags * T1_EncodingType * PS_Dict_Keys * */ /* Note that we separate font data in PS_FontInfoRec and PS_PrivateRec */ /* structures in order to support Multiple Master fonts. */ /************************************************************************** * * @struct: * PS_FontInfoRec * * @description: * A structure used to model a Type~1 or Type~2 FontInfo dictionary. * Note that for Multiple Master fonts, each instance has its own * FontInfo dictionary. */ typedef struct PS_FontInfoRec_ { FT_String* version; FT_String* notice; FT_String* full_name; FT_String* family_name; FT_String* weight; FT_Long italic_angle; FT_Bool is_fixed_pitch; FT_Short underline_position; FT_UShort underline_thickness; } PS_FontInfoRec;

# tttables.h

## *Data Types*

### *enum FT_Sfnt_Tag_*

**Definition:**
typedef enum FT_Sfnt_Tag_ { FT_SFNT_HEAD, FT_SFNT_MAXP, FT_SFNT_OS2, FT_SFNT_HHEA, FT_SFNT_VHEA, FT_SFNT_POST, FT_SFNT_PCLT, FT_SFNT_MAX } FT_Sfnt_Tag;

### *struct TT_HoriHeader_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Fixed | Version | |
| FT_Short | Ascender | |
| FT_Short | Descender | |
| FT_Short | Line_Gap | |
| FT_UShort | advance_Width_Max | |
| FT_Short | min_Left_Side_Bearing | |
| FT_Short | min_Right_Side_Bearing | |
| FT_Short | xMax_Extent | |
| FT_Short | caret_Slope_Rise | |
| FT_Short | caret_Slope_Run | |
| FT_Short | caret_Offset | |
| FT_Short | Reserved | |
| FT_Short | metric_Data_Format | |
| FT_UShort | number_Of_HMetrics | |
| void* | long_metrics | |
| void* | short_metrics | |

**Definition:**
typedef struct TT_HoriHeader_ { FT_Fixed Version; FT_Short Ascender; FT_Short Descender; FT_Short Line_Gap; FT_UShort advance_Width_Max; /* advance width maximum */ FT_Short min_Left_Side_Bearing; /* minimum left-sb */ FT_Short min_Right_Side_Bearing; /* minimum right-sb */ FT_Short xMax_Extent; /* xmax extents */ FT_Short caret_Slope_Rise; FT_Short caret_Slope_Run; FT_Short caret_Offset; FT_Short Reserved[4]; FT_Short metric_Data_Format; FT_UShort number_Of_HMetrics; /* The following fields are not defined by the OpenType specification */ /* but they are used to connect the metrics header to the relevant */ /* 'hmtx' table. */ void* long_metrics; void* short_metrics; } TT_HoriHeader;

## *struct TT_MaxProfile_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Fixed | version | |
| FT_UShort | numGlyphs | |
| FT_UShort | maxPoints | |
| FT_UShort | maxContours | |
| FT_UShort | maxCompositePoints | |
| FT_UShort | maxCompositeContours | |
| FT_UShort | maxZones | |
| FT_UShort | maxTwilightPoints | |
| FT_UShort | maxStorage | |
| FT_UShort | maxFunctionDefs | |
| FT_UShort | maxInstructionDefs | |
| FT_UShort | maxStackElements | |
| FT_UShort | maxSizeOfInstructions | |
| FT_UShort | maxComponentElements | |
| FT_UShort | maxComponentDepth | |

**Definition:**

typedef struct TT_MaxProfile_ { FT_Fixed version; FT_UShort numGlyphs; FT_UShort maxPoints; FT_UShort maxContours; FT_UShort maxCompositePoints; FT_UShort maxCompositeContours; FT_UShort maxZones; FT_UShort maxTwilightPoints; FT_UShort maxStorage; FT_UShort maxFunctionDefs; FT_UShort maxInstructionDefs; FT_UShort maxStackElements; FT_UShort maxSizeOfInstructions; FT_UShort maxComponentElements; FT_UShort maxComponentDepth; } TT_MaxProfile;

## *struct TT_OS2_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | version | |
| FT_Short | xAvgCharWidth | |
| FT_UShort | usWeightClass | |
| FT_UShort | usWidthClass | |
| FT_UShort | fsType | |
| FT_Short | ySubscriptXSize | |
| FT_Short | ySubscriptYSize | |
| FT_Short | ySubscriptXOffset | |

| | | |
|---|---|---|
| FT_Short | ySubscriptYOffset | |
| FT_Short | ySuperscriptXSize | |
| FT_Short | ySuperscriptYSize | |
| FT_Short | ySuperscriptXOffset | |
| FT_Short | ySuperscriptYOffset | |
| FT_Short | yStrikeoutSize | |
| FT_Short | yStrikeoutPosition | |
| FT_Short | sFamilyClass | |
| FT_Byte | panose | |
| FT_ULong | ulUnicodeRange1 | |
| FT_ULong | ulUnicodeRange2 | |
| FT_ULong | ulUnicodeRange3 | |
| FT_ULong | ulUnicodeRange4 | |
| FT_Char | achVendID | |
| FT_UShort | fsSelection | |
| FT_UShort | usFirstCharIndex | |
| FT_UShort | usLastCharIndex | |
| FT_Short | sTypoAscender | |
| FT_Short | sTypoDescender | |
| FT_Short | sTypoLineGap | |
| FT_UShort | usWinAscent | |
| FT_UShort | usWinDescent | |
| FT_ULong | ulCodePageRange1 | |
| FT_ULong | ulCodePageRange2 | |
| FT_Short | sxHeight | |
| FT_Short | sCapHeight | |
| FT_UShort | usDefaultChar | |
| FT_UShort | usBreakChar | |
| FT_UShort | usMaxContext | |
| FT_UShort | usLowerOpticalPointSize | |
| FT_UShort | usUpperOpticalPointSize | |

**Definition:**

typedef struct TT_OS2_ { FT_UShort version; /* 0x0001 - more or 0xFFFF */ FT_Short xAvgCharWidth; FT_UShort usWeightClass; FT_UShort usWidthClass; FT_UShort fsType; FT_Short ySubscriptXSize; FT_Short ySubscriptYSize; FT_Short ySubscriptXOffset; FT_Short ySubscriptYOffset; FT_Short ySuperscriptXSize; FT_Short ySuperscriptYSize; FT_Short ySuperscriptXOffset; FT_Short ySuperscriptYOffset; FT_Short yStrikeoutSize; FT_Short yStrikeoutPosition; FT_Short sFamilyClass;

FT_Byte panose[10]; FT_ULong ulUnicodeRange1; /* Bits 0-31 */ FT_ULong ulUnicodeRange2; /* Bits 32-63 */ FT_ULong ulUnicodeRange3; /* Bits 64-95 */ FT_ULong ulUnicodeRange4; /* Bits 96-127 */ FT_Char achVendID[4]; FT_UShort fsSelection; FT_UShort usFirstCharIndex; FT_UShort usLastCharIndex; FT_Short sTypoAscender; FT_Short sTypoDescender; FT_Short sTypoLineGap; FT_UShort usWinAscent; FT_UShort usWinDescent; /* only version 1 and higher: */ FT_ULong ulCodePageRange1; /* Bits 0-31 */ FT_ULong ulCodePageRange2; /* Bits 32-63 */ /* only version 2 and higher: */ FT_Short sxHeight; FT_Short sCapHeight; FT_UShort usDefaultChar; FT_UShort usBreakChar; FT_UShort usMaxContext; /* only version 5 and higher: */ FT_UShort usLowerOpticalPointSize; /* in twips (1/20 points) */ FT_UShort usUpperOpticalPointSize; /* in twips (1/20 points) */ } TT_OS2;

## *struct TT_PCLT_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Fixed | Version | |
| FT_ULong | FontNumber | |
| FT_UShort | Pitch | |
| FT_UShort | xHeight | |
| FT_UShort | Style | |
| FT_UShort | TypeFamily | |
| FT_UShort | CapHeight | |
| FT_UShort | SymbolSet | |
| FT_Char | TypeFace | |
| FT_Char | CharacterComplement | |
| FT_Char | FileName | |
| FT_Char | StrokeWeight | |
| FT_Char | WidthType | |
| FT_Byte | SerifStyle | |
| FT_Byte | Reserved | |

**Definition:**

typedef struct TT_PCLT_ { FT_Fixed Version; FT_ULong FontNumber; FT_UShort Pitch; FT_UShort xHeight; FT_UShort Style; FT_UShort TypeFamily; FT_UShort CapHeight; FT_UShort SymbolSet; FT_Char TypeFace[16]; FT_Char CharacterComplement[8]; FT_Char FileName[6]; FT_Char StrokeWeight; FT_Char WidthType; FT_Byte SerifStyle; FT_Byte Reserved; } TT_PCLT;

## *struct TT_Postscript_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Fixed | FormatType | |
| FT_Fixed | italicAngle | |

| | | |
|---|---|---|
| FT_Short | underlinePosition | |
| FT_Short | underlineThickness | |
| FT_ULong | isFixedPitch | |
| FT_ULong | minMemType42 | |
| FT_ULong | maxMemType42 | |
| FT_ULong | minMemType1 | |
| FT_ULong | maxMemType1 | |

**Definition:**
typedef struct TT_Postscript_ { FT_Fixed FormatType; FT_Fixed italicAngle; FT_Short underlinePosition; FT_Short underlineThickness; FT_ULong isFixedPitch; FT_ULong minMemType42; FT_ULong maxMemType42; FT_ULong minMemType1; FT_ULong maxMemType1; /* Glyph names follow in the 'post' table, but we don't */ /* load them by default. */ } TT_Postscript;

## *struct TT_VertHeader_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Fixed | Version | |
| FT_Short | Ascender | |
| FT_Short | Descender | |
| FT_Short | Line_Gap | |
| FT_UShort | advance_Height_Max | |
| FT_Short | min_Top_Side_Bearing | |
| FT_Short | min_Bottom_Side_Bearing | |
| FT_Short | yMax_Extent | |
| FT_Short | caret_Slope_Rise | |
| FT_Short | caret_Slope_Run | |
| FT_Short | caret_Offset | |
| FT_Short | Reserved | |
| FT_Short | metric_Data_Format | |
| FT_UShort | number_Of_VMetrics | |
| void* | long_metrics | |
| void* | short_metrics | |

**Definition:**
typedef struct TT_VertHeader_ { FT_Fixed Version; FT_Short Ascender; FT_Short Descender; FT_Short Line_Gap; FT_UShort advance_Height_Max; /* advance height maximum */ FT_Short min_Top_Side_Bearing; /* minimum top-sb */ FT_Short min_Bottom_Side_Bearing; /* minimum bottom-sb */ FT_Short yMax_Extent; /* ymax extents */ FT_Short caret_Slope_Rise; FT_Short caret_Slope_Run; FT_Short caret_Offset; FT_Short Reserved[4]; FT_Short metric_Data_Format; FT_UShort number_Of_VMetrics; /* The following fields are not defined by the OpenType specification

*/ /* but they are used to connect the metrics header to the relevant */ /* 'vmtx' table. */ void* long_metrics; void* short_metrics; } TT_VertHeader;

## *typedef Table_Version*

**Definition:**
* @struct: * TT_Header * * @description: * A structure to model a TrueType font header table. All fields follow * the OpenType specification. The 64-bit timestamps are stored in * two-element arrays `Created` and `Modified`, first the upper then * the lower 32~bits. */ typedef struct TT_Header_ { FT_Fixed Table_Version; FT_Fixed Font_Revision; FT_Long CheckSum_Adjust; FT_Long Magic_Number; FT_UShort Flags; FT_UShort Units_Per_EM; FT_ULong Created [2]; FT_ULong Modified[2]; FT_Short xMin; FT_Short yMin; FT_Short xMax; FT_Short yMax; FT_UShort Mac_Style; FT_UShort Lowest_Rec_PPEM; FT_Short Font_Direction; FT_Short Index_To_Loc_Format; FT_Short Glyph_Data_Format; } TT_Header;

# integer-types.h

## *Data Types*

### *typedef FT_Int16*

**Definition:**
* A typedef for a 16bit signed integer type. */ typedef signed short FT_Int16;

### *typedef FT_Int32*

**Definition:**
typedef signed XXX FT_Int32;

### *typedef FT_Int64*

**Definition:**
typedef signed XXX FT_Int64;

### *typedef FT_Int64*

**Definition:**
typedef FT_INT64 FT_Int64;

### *typedef FT_UInt16*

**Definition:**
typedef unsigned short FT_UInt16;

### *typedef FT_UInt32*

**Definition:**
typedef unsigned XXX FT_UInt32;

### *typedef FT_UInt64*

**Definition:**
typedef unsigned XXX FT_UInt64;

# autohint.h

## *Data Types*

### *struct FT_AutoHinterRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_AutoHinter_GlobalResetFunc | reset_face | |
| FT_AutoHinter_GlobalGetFunc | get_global_hints | |
| FT_AutoHinter_GlobalDoneFunc | done_global_hints | |
| FT_AutoHinter_GlyphLoadFunc | load_glyph | |

**Definition:**

typedef struct FT_AutoHinterRec_ *FT_AutoHinter;

### *struct FT_AutoHinter_InterfaceRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_AutoHinter_GlobalResetFunc | reset_face | |
| FT_AutoHinter_GlobalGetFunc | get_global_hints | |
| FT_AutoHinter_GlobalDoneFunc | done_global_hints | |
| FT_AutoHinter_GlyphLoadFunc | load_glyph | |

**Definition:**

typedef struct FT_AutoHinter_InterfaceRec_ { FT_AutoHinter_GlobalResetFunc reset_face; FT_AutoHinter_GlobalGetFunc get_global_hints; FT_AutoHinter_GlobalDoneFunc done_global_hints; FT_AutoHinter_GlyphLoadFunc load_glyph; } FT_AutoHinter_InterfaceRec, *FT_AutoHinter_Interface;

# cffotypes.h

## *Data Types*

### *typedef CFF_Face*

**Definition:**
typedef TT_Face CFF_Face;

### *struct CFF_GlyphSlotRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_GlyphSlotRec | root | |
| FT_Bool | hint | |
| FT_Bool | scaled | |
| FT_Fixed | x_scale | |
| FT_Fixed | y_scale | |

**Definition:**
typedef struct CFF_GlyphSlotRec_ { FT_GlyphSlotRec root; FT_Bool hint; FT_Bool scaled; FT_Fixed x_scale; FT_Fixed y_scale; } CFF_GlyphSlotRec, *CFF_GlyphSlot;

### *struct CFF_InternalRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| PSH_Globals | topfont | |
| PSH_Globals | subfonts | |

**Definition:**
typedef struct CFF_InternalRec_ { PSH_Globals topfont; PSH_Globals subfonts[CFF_MAX_CID_FONTS]; } CFF_InternalRec, *CFF_Internal;

### *struct CFF_SizeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |
| FT_ULong | strike_index | |

**Definition:**
typedef struct CFF_SizeRec_ { FT_SizeRec root; FT_ULong strike_index; /* 0xFFFFFFFF to indicate invalid */ } CFF_SizeRec, *CFF_Size;

## *struct CFF_Transform_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Fixed xx, | xy | |
| FT_Fixed yx, | yy | |
| FT_F26Dot6 ox, | oy | |

**Definition:**

typedef struct CFF_Transform_ { FT_Fixed xx, xy; /* transformation matrix coefficients */ FT_Fixed yx, yy; FT_F26Dot6 ox, oy; /* offsets */ } CFF_Transform;

# cfftypes.h

## *Data Types*

### *typedef stream*

**Definition:**
* @struct: * CFF_IndexRec * * @description: * A structure used to model a CFF Index table. * * @fields: * stream :: * The source input stream. * * start :: * The position of the first index byte in the input stream. * * count :: * The number of elements in the index. * * off_size :: * The size in bytes of object offsets in index. * * data_offset :: * The position of first data byte in the index's bytes. * * data_size :: * The size of the data table in this index. * * offsets :: * A table of element offsets in the index. Must be loaded explicitly. * * bytes :: * If the index is loaded in memory, its bytes. */ typedef struct CFF_IndexRec_ { FT_Stream stream; FT_ULong start; FT_UInt hdr_size; FT_UInt count; FT_Byte off_size; FT_ULong data_offset; FT_ULong data_size; FT_ULong* offsets; FT_Byte* bytes; } CFF_IndexRec, *CFF_Index;

# ftdebug.h

## *Data Types*

### *typedef FT_Trace*

**Definition:**
* Define the trace enums as well as the trace levels array when they are * needed. * */ #ifdef FT_DEBUG_LEVEL_TRACE #define FT_TRACE_DEF( x ) trace_ ## x , /* defining the enumeration */ typedef enum FT_Trace_ { #include trace_count } FT_Trace;

### *struct dlg_origin*

**Definition:**
ft_log_handler( const struct dlg_origin* origin, const char* string, void* data );

# ftdrv.h

## *Data Types*

### *struct FT_Driver_ClassRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Module_Class | root | |
| FT_Long | face_object_size | |
| FT_Long | size_object_size | |
| FT_Long | slot_object_size | |
| FT_Face_InitFunc | init_face | |
| FT_Face_DoneFunc | done_face | |
| FT_Size_InitFunc | init_size | |
| FT_Size_DoneFunc | done_size | |
| FT_Slot_InitFunc | init_slot | |
| FT_Slot_DoneFunc | done_slot | |
| FT_Slot_LoadFunc | load_glyph | |
| FT_Face_GetKerningFunc | get_kerning | |
| FT_Face_AttachFunc | attach_file | |
| FT_Face_GetAdvancesFunc | get_advances | |
| FT_Size_RequestFunc | request_size | |
| FT_Size_SelectFunc | select_size | |

**Definition:**

typedef struct FT_Driver_ClassRec_ { FT_Module_Class root; FT_Long face_object_size; FT_Long size_object_size; FT_Long slot_object_size; FT_Face_InitFunc init_face; FT_Face_DoneFunc done_face; FT_Size_InitFunc init_size; FT_Size_DoneFunc done_size; FT_Slot_InitFunc init_slot; FT_Slot_DoneFunc done_slot; FT_Slot_LoadFunc load_glyph; FT_Face_GetKerningFunc get_kerning; FT_Face_AttachFunc attach_file; FT_Face_GetAdvancesFunc get_advances; /* since version 2.2 */ FT_Size_RequestFunc request_size; FT_Size_SelectFunc select_size; } FT_Driver_ClassRec, *FT_Driver_Class;

# ftgloadr.h

## *Data Types*

### *typedef index*

**Definition:**
* @struct: * FT_GlyphLoader * * @description: * The glyph loader is an internal object used to load several glyphs * together (for example, in the case of composites). */ typedef struct FT_SubGlyphRec_ { FT_Int index; FT_UShort flags; FT_Int arg1; FT_Int arg2; FT_Matrix transform; } FT_SubGlyphRec;

# fthash.h

## *Data Types*

### *union FT_Hashkey_*

**Definition:**
typedef union FT_Hashkey_ { FT_Int num; const char* str; } FT_Hashkey;

# ftmmtypes.h

## *Data Types*

### *typedef FT_ItemVarDelta*

**Definition:**
typedef FT_Int32 FT_ItemVarDelta;

# ftobjs.h

## *Data Types*

### *struct FT_CMapRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_CharMapRec | charmap | |
| FT_CMap_Class | clazz | |

**Definition:**
typedef struct FT_CMapRec_* FT_CMap;

### *struct FT_Face_InternalRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Matrix | transform_matrix | |
| FT_Vector | transform_delta | |
| FT_Int | transform_flags | |
| FT_ServiceCacheRec | services | |
| FT_Incremental_InterfaceRec* | incremental_interface | |
| FT_Char | no_stem_darkening | |
| FT_Int32 | random_seed | |
| FT_LcdFiveTapFilter | lcd_weights | |
| FT_Bitmap_LcdFilterFunc | lcd_filter_func | |
| FT_Int | refcount | |

**Definition:**
typedef struct FT_Face_InternalRec_ { FT_Matrix transform_matrix; FT_Vector transform_delta; FT_Int transform_flags; FT_ServiceCacheRec services; #ifdef FT_CONFIG_OPTION_INCREMENTAL FT_Incremental_InterfaceRec* incremental_interface; #endif FT_Char no_stem_darkening; FT_Int32 random_seed; #ifdef FT_CONFIG_OPTION_SUBPIXEL_RENDERING FT_LcdFiveTapFilter lcd_weights; /* filter weights, if any */ FT_Bitmap_LcdFilterFunc lcd_filter_func; /* filtering callback */ #endif FT_Int refcount; } FT_Face_InternalRec;

### *struct FT_RendererRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ModuleRec | root | |
| FT_Renderer_Class* | clazz | |

| | | |
|---|---|---|
| FT_Glyph_Format | glyph_format | |
| FT_Glyph_Class | glyph_class | |
| FT_Raster | raster | |
| FT_Raster_Render_Func | raster_render | |
| FT_Renderer_RenderFunc | render | |

**Definition:**
typedef struct FT_RendererRec_ { FT_ModuleRec root; FT_Renderer_Class* clazz; FT_Glyph_Format glyph_format; FT_Glyph_Class glyph_class; FT_Raster raster; FT_Raster_Render_Func raster_render; FT_Renderer_RenderFunc render; } FT_RendererRec;

## *struct FT_Size_InternalRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| void* | module_data | |
| FT_Render_Mode | autohint_mode | |
| FT_Size_Metrics | autohint_metrics | |

**Definition:**
typedef struct FT_Size_InternalRec_ { void* module_data; FT_Render_Mode autohint_mode; FT_Size_Metrics autohint_metrics; } FT_Size_InternalRec;

## *struct FT_Slot_InternalRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_GlyphLoader | loader | |
| FT_UInt | flags | |
| FT_Bool | glyph_transformed | |
| FT_Matrix | glyph_matrix | |
| FT_Vector | glyph_delta | |
| void* | glyph_hints | |
| FT_Int32 | load_flags | |

**Definition:**
typedef struct FT_Slot_InternalRec_ { FT_GlyphLoader loader; FT_UInt flags; FT_Bool glyph_transformed; FT_Matrix glyph_matrix; FT_Vector glyph_delta; void* glyph_hints; FT_Int32 load_flags; } FT_GlyphSlot_InternalRec;

## *typedef clazz*

**Definition:**
* @struct: * FT_ModuleRec * * @description: * A module object instance. * * @fields: * clazz :: * A pointer to the module's class. * * library :: * A handle to the parent library object. * * memory :: * A

handle to the memory manager. */ typedef struct FT_ModuleRec_ { const FT_Module_Class* clazz; FT_Library library; FT_Memory memory; } FT_ModuleRec;

## *typedef memory*

**Definition:**
* @struct: * FT_LibraryRec * * @description: * The FreeType library class. This is the root of all FreeType data. * Use FT_New_Library() to create a library object, and FT_Done_Library() * to discard it and all child objects. * * @fields: * memory :: * The library's memory object. Manages memory allocation. * * version_major :: * The major version number of the library. * * version_minor :: * The minor version number of the library. * * version_patch :: * The current patch level of the library. * * num_modules :: * The number of modules currently registered within this library. * This is set to 0 for new libraries. New modules are added through * the FT_Add_Module() API function. * * modules :: * A table used to store handles to the currently registered * modules. Note that each font driver contains a list of its opened * faces. * * renderers :: * The list of renderers currently registered within the library. * * cur_renderer :: * The current outline renderer. This is a shortcut used to avoid * parsing the list on each call to FT_Outline_Render(). It is a * handle to the current renderer for the FT_GLYPH_FORMAT_OUTLINE * format. * * auto_hinter :: * The auto-hinter module interface. * * debug_hooks :: * An array of four function pointers that allow debuggers to hook into * a font format's interpreter. Currently, only the TrueType bytecode * debugger uses this. * * lcd_weights :: * The LCD filter weights for ClearType-style subpixel rendering. * * lcd_filter_func :: * The LCD filtering callback function for for ClearType-style subpixel * rendering. * * lcd_geometry :: * This array specifies LCD subpixel geometry and controls Harmony LCD * rendering technique, alternative to ClearType. * * pic_container :: * Contains global structs and tables, instead of defining them * globally. * * refcount :: * A counter initialized to~1 at the time an @FT_Library structure is * created. @FT_Reference_Library increments this counter, and * @FT_Done_Library only destroys a library if the counter is~1, * otherwise it simply decrements it. */ typedef struct FT_LibraryRec_ { FT_Memory memory; /* library's memory manager */ FT_Int version_major; FT_Int version_minor; FT_Int version_patch; FT_UInt num_modules; FT_Module modules[FT_MAX_MODULES]; /* module objects */ FT_ListRec renderers; /* list of renderers */ FT_Renderer cur_renderer; /* current outline renderer */ FT_Module auto_hinter; FT_DebugHook_Func debug_hooks[4]; #ifdef FT_CONFIG_OPTION_SUBPIXEL_RENDERING FT_LcdFiveTapFilter lcd_weights; /* filter weights, if any */ FT_Bitmap_LcdFilterFunc lcd_filter_func; /* filtering callback */ #else FT_Vector lcd_geometry[3]; /* RGB subpixel positions */ #endif FT_Int refcount; } FT_LibraryRec;

## *typedef root*

**Definition:**
* @struct: * FT_DriverRec * * @description: * The root font driver class. A font driver is responsible for managing * and loading font files of a given format. * * @fields: * root :: * Contains the fields of the root module class. * * clazz :: * A pointer to the font driver's class. Note that this is NOT * root.clazz. 'class' wasn't used as it is a reserved word in C++. * * faces_list :: * The list of faces currently opened by this driver. * * glyph_loader :: * Unused. Used to be glyph loader for all faces managed by this * driver. */ typedef struct FT_DriverRec_ { FT_ModuleRec root; FT_Driver_Class clazz; FT_ListRec faces_list; FT_GlyphLoader glyph_loader; } FT_DriverRec;

## *struct will*

**Definition:**
* The struct will be allocated in the global scope (or the scope where * the macro is used). */ #define FT_DECLARE_GLYPH( class_ ) \ FT_CALLBACK_TABLE const FT_Glyph_Class class_;

# ftrfork.h

## *Data Types*

### *struct FT_RFork_Ref_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Short | res_id | |
| FT_Long | offset | |

**Definition:**

typedef struct FT_RFork_Ref_ { FT_Short res_id; FT_Long offset; } FT_RFork_Ref;

# ftserv.h

## *Data Types*

### *typedef Rec*

**Definition:**
* A macro used to define new service structure types. */ #define FT_DEFINE_SERVICE( name ) \ typedef struct FT_Service_ ## name ## Rec_ \ FT_Service_ ## name ## Rec ; \

### *typedef serv_id*

**Definition:**
* The following structure is used to _describe_ a given service to the * library. This is useful to build simple static service lists. */ typedef struct FT_ServiceDescRec_ { const char* serv_id; /* service name */ const void* serv_data; /* service pointer/data */ } FT_ServiceDescRec;

### *typedef service_POSTSCRIPT_FONT_NAME*

**Definition:**
* This structure is used to store a cache for several frequently used * services. It is the type of `face->internal->services'. You should * only use FT_FACE_LOOKUP_SERVICE to access it. * * All fields should have the type FT_Pointer to relax compilation * dependencies. We assume the developer isn't completely stupid. * * Each field must be named `service_XXXX' where `XXX' corresponds to the * correct FT_SERVICE_ID_XXXX macro. See the definition of * FT_FACE_LOOKUP_SERVICE below how this is implemented. * */ typedef struct FT_ServiceCacheRec_ { FT_Pointer service_POSTSCRIPT_FONT_NAME; FT_Pointer service_MULTI_MASTERS; FT_Pointer service_METRICS_VARIATIONS; FT_Pointer service_GLYPH_DICT; FT_Pointer service_PFR_METRICS; FT_Pointer service_WINFNT; } FT_ServiceCacheRec, *FT_ServiceCache;

# ftstream.h

## *Data Types*

### *enum FT_Frame_Op_*

**Definition:**
typedef enum FT_Frame_Op_ { ft_frame_end = 0, ft_frame_start = FT_MAKE_FRAME_OP( FT_FRAME_OP_START, 0, 0 ), ft_frame_byte = FT_MAKE_FRAME_OP( FT_FRAME_OP_BYTE, 0, 0 ), ft_frame_schar = FT_MAKE_FRAME_OP( FT_FRAME_OP_BYTE, 0, 1 ), ft_frame_ushort_be = FT_MAKE_FRAME_OP( FT_FRAME_OP_SHORT, 0, 0 ), ft_frame_short_be = FT_MAKE_FRAME_OP( FT_FRAME_OP_SHORT, 0, 1 ), ft_frame_ushort_le = FT_MAKE_FRAME_OP( FT_FRAME_OP_SHORT, 1, 0 ), ft_frame_short_le = FT_MAKE_FRAME_OP( FT_FRAME_OP_SHORT, 1, 1 ), ft_frame_ulong_be = FT_MAKE_FRAME_OP( FT_FRAME_OP_LONG, 0, 0 ), ft_frame_long_be = FT_MAKE_FRAME_OP( FT_FRAME_OP_LONG, 0, 1 ), ft_frame_ulong_le = FT_MAKE_FRAME_OP( FT_FRAME_OP_LONG, 1, 0 ), ft_frame_long_le = FT_MAKE_FRAME_OP( FT_FRAME_OP_LONG, 1, 1 ), ft_frame_uoff3_be = FT_MAKE_FRAME_OP( FT_FRAME_OP_OFF3, 0, 0 ), ft_frame_off3_be = FT_MAKE_FRAME_OP( FT_FRAME_OP_OFF3, 0, 1 ), ft_frame_uoff3_le = FT_MAKE_FRAME_OP( FT_FRAME_OP_OFF3, 1, 0 ), ft_frame_off3_le = FT_MAKE_FRAME_OP( FT_FRAME_OP_OFF3, 1, 1 ), ft_frame_bytes = FT_MAKE_FRAME_OP( FT_FRAME_OP_BYTES, 0, 0 ), ft_frame_skip = FT_MAKE_FRAME_OP( FT_FRAME_OP_BYTES, 0, 1 ) } FT_Frame_Op;

# ftvalid.h

## *Data Types*

### *enum FT_ValidationLevel_*

**Definition:**
typedef enum FT_ValidationLevel_ { FT_VALIDATE_DEFAULT = 0, FT_VALIDATE_TIGHT, FT_VALIDATE_PARANOID } FT_ValidationLevel;

### *struct FT_ValidatorRec_*

**Definition:**
typedef struct FT_ValidatorRec_ volatile* FT_Validator;

# psaux.h

## *Data Types*

### *struct AFM_ParserRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte* limit | ) | |
| (*done)( AFM_Parser parser | ) | |
| (*parse)( AFM_Parser parser | ) | |

**Definition:**
typedef struct AFM_ParserRec_* AFM_Parser;

### *struct AFM_ParserRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Memory | memory | |
| AFM_Stream | stream | |
| AFM_FontInfo | FontInfo | |
| void* user_data | ) | |
| void* | user_data | |

**Definition:**
typedef struct AFM_ParserRec_ { FT_Memory memory; AFM_Stream stream; AFM_FontInfo FontInfo; FT_Int (*get_index)( const char* name, FT_Offset len, void* user_data ); void* user_data; } AFM_ParserRec;

### *struct CFF_Builder_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Bool hinting | ) | |
| (*done)( CFF_Builder* builder | ) | |
| CFF_Builder_Check_Points_Func | check_points | |
| CFF_Builder_Add_Point_Func | add_point | |
| CFF_Builder_Add_Point1_Func | add_point1 | |
| CFF_Builder_Add_Contour_Func | add_contour | |
| CFF_Builder_Start_Point_Func | start_point | |
| CFF_Builder_Close_Contour_Func | close_contour | |

**Definition:**
typedef struct CFF_Builder_ CFF_Builder;

## *struct CFF_Builder_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Memory | memory | |
| TT_Face | face | |
| CFF_GlyphSlot | glyph | |
| FT_GlyphLoader | loader | |
| FT_Outline* | base | |
| FT_Outline* | current | |
| FT_Pos | pos_x | |
| FT_Pos | pos_y | |
| FT_Vector | left_bearing | |
| FT_Vector | advance | |
| FT_BBox | bbox | |
| FT_Bool | path_begun | |
| FT_Bool | load_points | |
| FT_Bool | no_recurse | |
| FT_Bool | metrics_only | |
| void* | hints_funcs | |
| void* | hints_globals | |
| CFF_Builder_FuncsRec | funcs | |

**Definition:**
struct CFF_Builder_ { FT_Memory memory; TT_Face face; CFF_GlyphSlot glyph; FT_GlyphLoader loader; FT_Outline* base; FT_Outline* current; FT_Pos pos_x; FT_Pos pos_y; FT_Vector left_bearing; FT_Vector advance; FT_BBox bbox; /* bounding box */ FT_Bool path_begun; FT_Bool load_points; FT_Bool no_recurse; FT_Bool metrics_only; void* hints_funcs; /* hinter-specific */ void* hints_globals; /* hinter-specific */ CFF_Builder_FuncsRec funcs; };

## *struct CFF_Decoder_Zone_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte* | base | |
| FT_Byte* | limit | |
| FT_Byte* | cursor | |

**Definition:**

typedef struct CFF_Decoder_Zone_ { FT_Byte* base; FT_Byte* limit; FT_Byte* cursor; } CFF_Decoder_Zone;

## *struct PSAux_ServiceRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| const PS_Table_FuncsRec* | ps_table_funcs | |
| const PS_Parser_FuncsRec* | ps_parser_funcs | |
| const T1_Builder_FuncsRec* | t1_builder_funcs | |
| const T1_Decoder_FuncsRec* | t1_decoder_funcs | |
| FT_UShort  seed | ) | |
| (*cff_random)( FT_UInt32  r | ) | |
| FT_Bool  is_t1 | ) | |
| CFF_SubFont  subfont | ) | |
| T1_CMap_Classes | t1_cmap_classes | |
| const AFM_Parser_FuncsRec* | afm_parser_funcs | |
| const CFF_Decoder_FuncsRec* | cff_decoder_funcs | |

**Definition:**

typedef struct PSAux_ServiceRec_ { /* don't use `PS_Table_Funcs' and friends to avoid compiler warnings */ const PS_Table_FuncsRec* ps_table_funcs; const PS_Parser_FuncsRec* ps_parser_funcs; const T1_Builder_FuncsRec* t1_builder_funcs; const T1_Decoder_FuncsRec* t1_decoder_funcs; void (*t1_decrypt)( FT_Byte* buffer, FT_Offset length, FT_UShort seed ); FT_UInt32 (*cff_random)( FT_UInt32 r ); void (*ps_decoder_init)( PS_Decoder* ps_decoder, void* decoder, FT_Bool is_t1 ); void (*t1_make_subfont)( FT_Face face, PS_Private priv, CFF_SubFont subfont ); T1_CMap_Classes t1_cmap_classes; /* fields after this comment line were added after version 2.1.10 */ const AFM_Parser_FuncsRec* afm_parser_funcs; const CFF_Decoder_FuncsRec* cff_decoder_funcs; } PSAux_ServiceRec, *PSAux_Service;

## *struct PS_Builder_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Bool  is_t1 | ) | |
| (*done)( PS_Builder*  builder | ) | |

**Definition:**

typedef struct PS_Builder_ PS_Builder;

## *struct PS_Builder_*

**Members:**

| Type | Name | Description |
|---|---|---|

| | | |
|---|---|---|
| FT_Memory | memory | |
| FT_Face | face | |
| CFF_GlyphSlot | glyph | |
| FT_GlyphLoader | loader | |
| FT_Outline* | base | |
| FT_Outline* | current | |
| FT_Pos* | pos_x | |
| FT_Pos* | pos_y | |
| FT_Vector* | left_bearing | |
| FT_Vector* | advance | |
| FT_BBox* | bbox | |
| FT_Bool | path_begun | |
| FT_Bool | load_points | |
| FT_Bool | no_recurse | |
| FT_Bool | metrics_only | |
| FT_Bool | is_t1 | |
| PS_Builder_FuncsRec | funcs | |

**Definition:**
struct PS_Builder_ { FT_Memory memory; FT_Face face; CFF_GlyphSlot glyph; FT_GlyphLoader loader; FT_Outline* base; FT_Outline* current; FT_Pos* pos_x; FT_Pos* pos_y; FT_Vector* left_bearing; FT_Vector* advance; FT_BBox* bbox; /* bounding box */ FT_Bool path_begun; FT_Bool load_points; FT_Bool no_recurse; FT_Bool metrics_only; FT_Bool is_t1; PS_Builder_FuncsRec funcs; };

## *struct PS_Decoder_Zone_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte* | base | |
| FT_Byte* | limit | |
| FT_Byte* | cursor | |

**Definition:**
typedef struct PS_Decoder_Zone_ { FT_Byte* base; FT_Byte* limit; FT_Byte* cursor; } PS_Decoder_Zone;

## *struct PS_DriverRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_DriverRec | root | |

| | | |
|---|---|---|
| FT_UInt | hinting_engine | |
| FT_Bool | no_stem_darkening | |
| FT_Int | darken_params | |
| FT_Int32 | random_seed | |

**Definition:**
typedef struct PS_DriverRec_ { FT_DriverRec root; FT_UInt hinting_engine; FT_Bool no_stem_darkening; FT_Int darken_params[8]; FT_Int32 random_seed; } PS_DriverRec, *PS_Driver;

## *struct PS_ParserRec_*

**Definition:**
typedef struct PS_ParserRec_* PS_Parser;

## *struct PS_ParserRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte* | cursor | |
| FT_Byte* | base | |
| FT_Byte* | limit | |
| FT_Error | error | |
| FT_Memory | memory | |
| PS_Parser_FuncsRec | funcs | |

**Definition:**
typedef struct PS_ParserRec_ { FT_Byte* cursor; FT_Byte* base; FT_Byte* limit; FT_Error error; FT_Memory memory; PS_Parser_FuncsRec funcs; } PS_ParserRec;

## *struct PS_Parser_FuncsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Memory  memory | ) | |
| (*done)( PS_Parser  parser | ) | |
| (*skip_spaces)( PS_Parser  parser | ) | |
| (*skip_PS_token)( PS_Parser  parser | ) | |
| (*to_int)( PS_Parser  parser | ) | |
| FT_Int  power_ten | ) | |
| FT_Bool  delimiters | ) | |
| FT_Short*  coords | ) | |
| FT_Int  power_ten | ) | |

| | | |
|---|---|---|
| T1_Token token | ) | |
| FT_Int* pnum_tokens | ) | |
| FT_ULong* pflags | ) | |
| FT_ULong* pflags | ) | |

**Definition:**
typedef const struct PS_Parser_FuncsRec_* PS_Parser_Funcs;

## *struct PS_TableRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Memory memory | ) | |
| (*done)( PS_Table table | ) | |
| FT_UInt length | ) | |
| (*release)( PS_Table table | ) | |

**Definition:**
typedef struct PS_TableRec_* PS_Table;

## *struct PS_TableRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte* | block | |
| FT_Offset | cursor | |
| FT_Offset | capacity | |
| FT_ULong | init | |
| FT_Int | max_elems | |
| FT_Byte** | elements | |
| FT_UInt* | lengths | |
| FT_Memory | memory | |
| PS_Table_FuncsRec | funcs | |

**Definition:**
typedef struct PS_TableRec_ { FT_Byte* block; /* current memory block */ FT_Offset cursor; /* current cursor in memory block */ FT_Offset capacity; /* current size of memory block */ FT_ULong init; FT_Int max_elems; FT_Byte** elements; /* addresses of table elements */ FT_UInt* lengths; /* lengths of table elements */ FT_Memory memory; PS_Table_FuncsRec funcs; } PS_TableRec;

## *struct PS_Table_FuncsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|

| | | |
|---|---|---|
| FT_Memory memory | ) | |
| (*done)( PS_Table table | ) | |
| FT_UInt length | ) | |
| (*release)( PS_Table table | ) | |

**Definition:**
typedef struct PS_Table_FuncsRec_ { FT_Error (*init)( PS_Table table, FT_Int count, FT_Memory memory ); void (*done)( PS_Table table ); FT_Error (*add)( PS_Table table, FT_Int idx, const void* object, FT_UInt length ); void (*release)( PS_Table table ); } PS_Table_FuncsRec;

## *struct T1_BuilderRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Bool hinting | ) | |
| (*done)( T1_Builder builder | ) | |
| T1_Builder_Check_Points_Func | check_points | |
| T1_Builder_Add_Point_Func | add_point | |
| T1_Builder_Add_Point1_Func | add_point1 | |
| T1_Builder_Add_Contour_Func | add_contour | |
| T1_Builder_Start_Point_Func | start_point | |
| T1_Builder_Close_Contour_Func | close_contour | |

**Definition:**
typedef struct T1_BuilderRec_* T1_Builder;

## *struct T1_BuilderRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Memory | memory | |
| FT_Face | face | |
| FT_GlyphSlot | glyph | |
| FT_GlyphLoader | loader | |
| FT_Outline* | base | |
| FT_Outline* | current | |
| FT_Pos | pos_x | |
| FT_Pos | pos_y | |
| FT_Vector | left_bearing | |
| FT_Vector | advance | |
| FT_BBox | bbox | |

| | | |
|---|---|---|
| T1_ParseState | parse_state | |
| FT_Bool | load_points | |
| FT_Bool | no_recurse | |
| FT_Bool | metrics_only | |
| void* | hints_funcs | |
| void* | hints_globals | |
| T1_Builder_FuncsRec | funcs | |

**Definition:**
typedef struct T1_BuilderRec_ { FT_Memory memory; FT_Face face; FT_GlyphSlot glyph; FT_GlyphLoader loader; FT_Outline* base; FT_Outline* current; FT_Pos pos_x; FT_Pos pos_y; FT_Vector left_bearing; FT_Vector advance; FT_BBox bbox; /* bounding box */ T1_ParseState parse_state; FT_Bool load_points; FT_Bool no_recurse; FT_Bool metrics_only; void* hints_funcs; /* hinter-specific */ void* hints_globals; /* hinter-specific */ T1_Builder_FuncsRec funcs; } T1_BuilderRec;

## *struct T1_CMap_ClassesRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_CMap_Class | standard | |
| FT_CMap_Class | expert | |
| FT_CMap_Class | custom | |
| FT_CMap_Class | unicode | |

**Definition:**
typedef const struct T1_CMap_ClassesRec_* T1_CMap_Classes;

## *struct T1_Decoder_ZoneRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte* | cursor | |
| FT_Byte* | base | |
| FT_Byte* | limit | |

**Definition:**
typedef struct T1_Decoder_ZoneRec_ { FT_Byte* cursor; FT_Byte* base; FT_Byte* limit; } T1_Decoder_ZoneRec, *T1_Decoder_Zone;

# pshints.h

## *Data Types*

### *struct PSH_GlobalsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| PSH_Globals_NewFunc | create | |
| PSH_Globals_SetScaleFunc | set_scale | |
| PSH_Globals_DestroyFunc | destroy | |

**Definition:**
typedef struct PSH_GlobalsRec_* PSH_Globals;

### *typedef T1_Hints*

**Definition:**
* This is a handle to an opaque structure used to record glyph hints * from a Type 1 character glyph character string. * * The methods used to operate on this object are defined by the * @T1_Hints_FuncsRec structure. Recording glyph hints is normally * achieved through the following scheme: * * - Open a new hint recording session by calling the 'open' method. * This rewinds the recorder and prepare it for new input. * * - For each hint found in the glyph charstring, call the corresponding * method ('stem', 'stem3', or 'reset'). Note that these functions do * not return an error code. * * - Close the recording session by calling the 'close' method. It * returns an error code if the hints were invalid or something strange * happened (e.g., memory shortage). * * The hints accumulated in the object can later be used by the * PostScript hinter. * */ typedef struct T1_HintsRec_* T1_Hints;

### *struct T1_Hints_FuncsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| T1_Hints | hints | |
| T1_Hints_OpenFunc | open | |
| T1_Hints_CloseFunc | close | |
| T1_Hints_SetStemFunc | stem | |
| T1_Hints_SetStem3Func | stem3 | |
| T1_Hints_ResetFunc | reset | |
| T1_Hints_ApplyFunc | apply | |

**Definition:**
typedef const struct T1_Hints_FuncsRec_* T1_Hints_Funcs;

### *struct T1_Hints_FuncsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| T1_Hints | hints | |
| T1_Hints_OpenFunc | open | |
| T1_Hints_CloseFunc | close | |
| T1_Hints_SetStemFunc | stem | |
| T1_Hints_SetStem3Func | stem3 | |
| T1_Hints_ResetFunc | reset | |
| T1_Hints_ApplyFunc | apply | |

**Definition:**
typedef struct T1_Hints_FuncsRec_ { T1_Hints hints; T1_Hints_OpenFunc open; T1_Hints_CloseFunc close; T1_Hints_SetStemFunc stem; T1_Hints_SetStem3Func stem3; T1_Hints_ResetFunc reset; T1_Hints_ApplyFunc apply; } T1_Hints_FuncsRec;

## *typedef T2_Hints*

**Definition:**
* This is a handle to an opaque structure used to record glyph hints * from a Type 2 character glyph character string. * * The methods used to operate on this object are defined by the * @T2_Hints_FuncsRec structure. Recording glyph hints is normally * achieved through the following scheme: * * - Open a new hint recording session by calling the 'open' method. * This rewinds the recorder and prepare it for new input. * * - For each hint found in the glyph charstring, call the corresponding * method ('stems', 'hintmask', 'counters'). Note that these functions * do not return an error code. * * - Close the recording session by calling the 'close' method. It * returns an error code if the hints were invalid or something strange * happened (e.g., memory shortage). * * The hints accumulated in the object can later be used by the * Postscript hinter. * */ typedef struct T2_HintsRec_* T2_Hints;

## *struct T2_Hints_FuncsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| T2_Hints | hints | |
| T2_Hints_OpenFunc | open | |
| T2_Hints_CloseFunc | close | |
| T2_Hints_StemsFunc | stems | |
| T2_Hints_MaskFunc | hintmask | |
| T2_Hints_CounterFunc | counter | |
| T2_Hints_ApplyFunc | apply | |

**Definition:**
typedef const struct T2_Hints_FuncsRec_* T2_Hints_Funcs;

## *struct T2_Hints_FuncsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| T2_Hints | hints | |
| T2_Hints_OpenFunc | open | |
| T2_Hints_CloseFunc | close | |
| T2_Hints_StemsFunc | stems | |
| T2_Hints_MaskFunc | hintmask | |
| T2_Hints_CounterFunc | counter | |
| T2_Hints_ApplyFunc | apply | |

**Definition:**

typedef struct T2_Hints_FuncsRec_ { T2_Hints hints; T2_Hints_OpenFunc open; T2_Hints_CloseFunc close; T2_Hints_StemsFunc stems; T2_Hints_MaskFunc hintmask; T2_Hints_CounterFunc counter; T2_Hints_ApplyFunc apply; } T2_Hints_FuncsRec;

# sfnt.h

## *Data Types*

### *struct SFNT_Interface_*

**Members:**

| Type | Name | Description |
|---|---|---|
| TT_Loader_GotoTableFunc | goto_table | |
| TT_Init_Face_Func | init_face | |
| TT_Load_Face_Func | load_face | |
| TT_Done_Face_Func | done_face | |
| FT_Module_Requester | get_interface | |
| TT_Load_Any_Func | load_any | |
| TT_Load_Table_Func | load_head | |
| TT_Load_Metrics_Func | load_hhea | |
| TT_Load_Table_Func | load_cmap | |
| TT_Load_Table_Func | load_maxp | |
| TT_Load_Table_Func | load_os2 | |
| TT_Load_Table_Func | load_post | |
| TT_Load_Table_Func | load_name | |
| TT_Free_Table_Func | free_name | |
| TT_Load_Table_Func | load_kern | |
| TT_Load_Table_Func | load_gpos | |
| TT_Load_Table_Func | load_gasp | |
| TT_Load_Table_Func | load_pclt | |
| TT_Load_Table_Func | load_bhed | |
| TT_Load_SBit_Image_Func | load_sbit_image | |
| TT_Get_PS_Name_Func | get_psname | |
| TT_Free_Table_Func | free_psnames | |
| TT_Face_GetKerningFunc | get_kerning | |
| TT_Face_GetKerningFunc | get_gpos_kerning | |
| TT_Load_Table_Func | load_font_dir | |
| TT_Load_Metrics_Func | load_hmtx | |
| TT_Load_Table_Func | load_eblc | |
| TT_Free_Table_Func | free_eblc | |

| | | |
|---|---|---|
| TT_Set_SBit_Strike_Func | set_sbit_strike | |
| TT_Load_Strike_Metrics_Func | load_strike_metrics | |
| TT_Load_Table_Func | load_cpal | |
| TT_Load_Table_Func | load_colr | |
| TT_Free_Table_Func | free_cpal | |
| TT_Free_Table_Func | free_colr | |
| TT_Set_Palette_Func | set_palette | |
| TT_Get_Colr_Layer_Func | get_colr_layer | |
| TT_Get_Color_Glyph_Paint_Func | get_colr_glyph_paint | |
| TT_Get_Color_Glyph_ClipBox_Func | get_color_glyph_clipbox | |
| TT_Get_Paint_Layers_Func | get_paint_layers | |
| TT_Get_Colorline_Stops_Func | get_colorline_stops | |
| TT_Get_Paint_Func | get_paint | |
| TT_Blend_Colr_Func | colr_blend | |
| TT_Get_Metrics_Func | get_metrics | |
| TT_Get_Name_Func | get_name | |
| TT_Get_Name_ID_Func | get_name_id | |
| TT_Load_Table_Func | load_svg | |
| TT_Free_Table_Func | free_svg | |
| TT_Load_Svg_Doc_Func | load_svg_doc | |

**Definition:**

typedef struct SFNT_Interface_ { TT_Loader_GotoTableFunc goto_table; TT_Init_Face_Func init_face; TT_Load_Face_Func load_face; TT_Done_Face_Func done_face; FT_Module_Requester get_interface; TT_Load_Any_Func load_any; /* these functions are called by `load_face' but they can also */ /* be called from external modules, if there is a need to do so */ TT_Load_Table_Func load_head; TT_Load_Metrics_Func load_hhea; TT_Load_Table_Func load_cmap; TT_Load_Table_Func load_maxp; TT_Load_Table_Func load_os2; TT_Load_Table_Func load_post; TT_Load_Table_Func load_name; TT_Free_Table_Func free_name; /* this field was called `load_kerning' up to version 2.1.10 */ TT_Load_Table_Func load_kern; TT_Load_Table_Func load_gpos; TT_Load_Table_Func load_gasp; TT_Load_Table_Func load_pclt; /* see `ttload.h'; this field was called `load_bitmap_header' up to */ /* version 2.1.10 */ TT_Load_Table_Func load_bhed; TT_Load_SBit_Image_Func load_sbit_image; /* see `ttpost.h' */ TT_Get_PS_Name_Func get_psname; TT_Free_Table_Func free_psnames; /* starting here, the structure differs from version 2.1.7 */ /* this field was introduced in version 2.1.8, named `get_psname' */ TT_Face_GetKerningFunc get_kerning; /* new elements introduced after version 2.1.10 */ TT_Face_GetKerningFunc get_gpos_kerning; /* load the font directory, i.e., the offset table and */ /* the table directory */ TT_Load_Table_Func load_font_dir; TT_Load_Metrics_Func load_hmtx; TT_Load_Table_Func load_eblc; TT_Free_Table_Func free_eblc; TT_Set_SBit_Strike_Func set_sbit_strike; TT_Load_Strike_Metrics_Func load_strike_metrics; TT_Load_Table_Func load_cpal; TT_Load_Table_Func load_colr; TT_Free_Table_Func free_cpal; TT_Free_Table_Func free_colr; TT_Set_Palette_Func set_palette; TT_Get_Colr_Layer_Func get_colr_layer; TT_Get_Color_Glyph_Paint_Func get_colr_glyph_paint; TT_Get_Color_Glyph_ClipBox_Func get_color_glyph_clipbox; TT_Get_Paint_Layers_Func get_paint_layers;

TT_Get_Colorline_Stops_Func get_colorline_stops; TT_Get_Paint_Func get_paint;
TT_Blend_Colr_Func colr_blend; TT_Get_Metrics_Func get_metrics; TT_Get_Name_Func get_name;
TT_Get_Name_ID_Func get_name_id; /* OpenType SVG Support */ TT_Load_Table_Func load_svg;
TT_Free_Table_Func free_svg; TT_Load_Svg_Doc_Func load_svg_doc; } SFNT_Interface;

# t1types.h

## *Data Types*

### *struct AFM_TrackKernRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Int | degree | |
| FT_Fixed | min_ptsize | |
| FT_Fixed | min_kern | |
| FT_Fixed | max_ptsize | |
| FT_Fixed | max_kern | |

**Definition:**
typedef struct AFM_TrackKernRec_ { FT_Int degree; FT_Fixed min_ptsize; FT_Fixed min_kern; FT_Fixed max_ptsize; FT_Fixed max_kern; } AFM_TrackKernRec, *AFM_TrackKern;

### *struct T1_FaceRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_FaceRec | root | |
| T1_FontRec | type1 | |
| const void* | psnames | |
| const void* | psaux | |
| const void* | afm_data | |
| FT_CharMapRec | charmaprecs | |
| FT_CharMap | charmaps | |
| PS_Blend | blend | |
| FT_Int | ndv_idx | |
| FT_Int | cdv_idx | |
| FT_UInt | len_buildchar | |
| FT_Long* | buildchar | |
| const void* | pshinter | |

**Definition:**
typedef struct T1_FaceRec_* T1_Face;

### *typedef num_chars*

**Definition:**

* @struct: * T1_EncodingRec * * @description: * A structure modeling a custom encoding. * * @fields: * num_chars :: * The number of character codes in the encoding. Usually 256. * * code_first :: * The lowest valid character code in the encoding. * * code_last :: * The highest valid character code in the encoding + 1. When equal to * code_first there are no valid character codes. * * char_index :: * An array of corresponding glyph indices. * * char_name :: * An array of corresponding glyph names. */ typedef struct T1_EncodingRecRec_ { FT_Int num_chars; FT_Int code_first; FT_Int code_last; FT_UShort* char_index; const FT_String** char_name; } T1_EncodingRec, *T1_Encoding;

# tttypes.h

## *Data Types*

### *struct GX_BlendRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte* | table | |
| FT_Byte* | table_end | |
| FT_Byte* | strings | |
| FT_ULong | strings_size | |
| FT_UInt | num_strikes | |
| FT_Bool | loaded | |

**Definition:**
typedef struct GX_BlendRec_ *GX_Blend;

### *struct SFNT_HeaderRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ULong | format_tag | |
| FT_UShort | num_tables | |
| FT_UShort | search_range | |
| FT_UShort | entry_selector | |
| FT_UShort | range_shift | |
| FT_ULong | offset | |

**Definition:**
typedef struct SFNT_HeaderRec_ { FT_ULong format_tag; FT_UShort num_tables; FT_UShort search_range; FT_UShort entry_selector; FT_UShort range_shift; FT_ULong offset; /* not in file */ } SFNT_HeaderRec, *SFNT_Header;

### *struct TT_BDFRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte* | table | |
| FT_Byte* | table_end | |
| FT_Byte* | strings | |
| FT_ULong | strings_size | |

| | | |
|---|---|---|
| FT_UInt | num_strikes | |
| FT_Bool | loaded | |

**Definition:**
typedef struct TT_BDFRec_ { FT_Byte* table; FT_Byte* table_end; FT_Byte* strings; FT_ULong strings_size; FT_UInt num_strikes; FT_Bool loaded; } TT_BDFRec, *TT_BDF;

## *struct TT_FaceRec_*

**Definition:**
typedef struct TT_FaceRec_* TT_Face;

## *struct TT_FaceRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_FaceRec | root | |
| TTC_HeaderRec | ttc_header | |
| FT_ULong | format_tag | |
| FT_UShort | num_tables | |
| TT_Table | dir_tables | |
| TT_Header | header | |
| TT_HoriHeader | horizontal | |
| TT_MaxProfile | max_profile | |
| FT_Bool | vertical_info | |
| TT_VertHeader | vertical | |
| FT_UShort | num_names | |
| TT_NameTableRec | name_table | |
| TT_OS2 | os2 | |
| TT_Postscript | postscript | |
| FT_Byte* | cmap_table | |
| FT_ULong | cmap_size | |
| TT_Loader_GotoTableFunc | goto_table | |
| TT_Loader_StartGlyphFunc | access_glyph_frame | |
| TT_Loader_EndGlyphFunc | forget_glyph_frame | |
| TT_Loader_ReadGlyphFunc | read_glyph_header | |
| TT_Loader_ReadGlyphFunc | read_simple_glyph | |
| TT_Loader_ReadGlyphFunc | read_composite_glyph | |
| void* | sfnt | |

| | | |
|---|---|---|
| void* | psnames | |
| void* | mm | |
| void* | tt_var | |
| void* | face_var | |
| void* | psaux | |
| TT_GaspRec | gasp | |
| TT_PCLT | pclt | |
| FT_ULong | num_sbit_scales | |
| TT_SBit_Scale | sbit_scales | |
| TT_Post_NamesRec | postscript_names | |
| FT_Palette_Data | palette_data | |
| FT_UShort | palette_index | |
| FT_Color* | palette | |
| FT_Bool | have_foreground_color | |
| FT_Color | foreground_color | |
| FT_ULong | font_program_size | |
| FT_Byte* | font_program | |
| FT_ULong | cvt_program_size | |
| FT_Byte* | cvt_program | |
| FT_ULong | cvt_size | |
| FT_Int32* | cvt | |
| FT_Generic | extra | |
| const char* | postscript_name | |
| FT_ULong | glyf_len | |
| FT_ULong | glyf_offset | |
| FT_Bool | is_cff2 | |
| FT_Bool | doblend | |
| GX_Blend | blend | |
| FT_UInt32 | variation_support | |
| const char* | var_postscript_prefix | |
| FT_UInt | var_postscript_prefix_len | |
| FT_UInt | var_default_named_instance | |
| const char* | non_var_style_name | |
| FT_ULong | horz_metrics_size | |
| FT_ULong | vert_metrics_size | |

| | | |
|---|---|---|
| FT_ULong | num_locations | |
| FT_Byte* | glyph_locations | |
| FT_Byte* | hdmx_table | |
| FT_ULong | hdmx_table_size | |
| FT_UInt | hdmx_record_count | |
| FT_ULong | hdmx_record_size | |
| FT_Byte** | hdmx_records | |
| FT_Byte* | sbit_table | |
| FT_ULong | sbit_table_size | |
| TT_SbitTableType | sbit_table_type | |
| FT_UInt | sbit_num_strikes | |
| FT_UInt* | sbit_strike_map | |
| FT_Byte* | kern_table | |
| FT_ULong | kern_table_size | |
| FT_UInt | num_kern_tables | |
| FT_UInt32 | kern_avail_bits | |
| FT_UInt32 | kern_order_bits | |
| TT_BDFRec | bdf | |
| FT_ULong | horz_metrics_offset | |
| FT_ULong | vert_metrics_offset | |
| FT_ULong | ebdt_start | |
| FT_ULong | ebdt_size | |
| void* | cpal | |
| void* | colr | |
| void* | svg | |
| FT_Byte* | gpos_table | |
| FT_UInt32* | gpos_lookups_kerning | |
| FT_UInt | num_gpos_lookups_kerning | |

**Definition:**

typedef struct TT_FaceRec_ { FT_FaceRec root; TTC_HeaderRec ttc_header; FT_ULong format_tag; FT_UShort num_tables; TT_Table dir_tables; TT_Header header; /* TrueType header table */ TT_HoriHeader horizontal; /* TrueType horizontal header */ TT_MaxProfile max_profile; FT_Bool vertical_info; TT_VertHeader vertical; /* TT Vertical header, if present */ FT_UShort num_names; /* number of name records */ TT_NameTableRec name_table; /* name table */ TT_OS2 os2; /* TrueType OS/2 table */ TT_Postscript postscript; /* TrueType Postscript table */ FT_Byte* cmap_table; /* extracted `cmap' table */ FT_ULong cmap_size; TT_Loader_GotoTableFunc goto_table; TT_Loader_StartGlyphFunc access_glyph_frame; TT_Loader_EndGlyphFunc forget_glyph_frame; TT_Loader_ReadGlyphFunc read_glyph_header; TT_Loader_ReadGlyphFunc read_simple_glyph; TT_Loader_ReadGlyphFunc read_composite_glyph; /* a typeless pointer to the SFNT_Interface table

used to load */ /* the basic TrueType tables in the face object */ void* sfnt; /* a typeless pointer to the FT_Service_PsCMapsRec table used to */ /* handle glyph names <-> unicode & Mac values */ void* psnames; #ifdef TT_CONFIG_OPTION_GX_VAR_SUPPORT /* a typeless pointer to the FT_Service_MultiMasters table used to */ /* handle variation fonts */ void* mm; /* a typeless pointer to the FT_Service_MetricsVariationsRec table */ /* used to handle the HVAR, VVAR, and MVAR OpenType tables by the */ /* "truetype" driver */ void* tt_var; /* a typeless pointer to the FT_Service_MetricsVariationsRec table */ /* used to handle the HVAR, VVAR, and MVAR OpenType tables by this */ /* TT_Face's driver */ void* face_var; /* since 2.13.1 */ #endif /* a typeless pointer to the PostScript Aux service */ void* psaux; /************************************************************************** * * Optional TrueType/OpenType tables * */ /* grid-fitting and scaling table */ TT_GaspRec gasp; /* the `gasp' table */ /* PCL 5 table */ TT_PCLT pclt; /* embedded bitmaps support */ FT_ULong num_sbit_scales; TT_SBit_Scale sbit_scales; /* postscript names table */ TT_Post_NamesRec postscript_names; /* glyph colors */ FT_Palette_Data palette_data; /* since 2.10 */ FT_UShort palette_index; FT_Color* palette; FT_Bool have_foreground_color; FT_Color foreground_color; /************************************************************************** * * TrueType-specific fields (ignored by the CFF driver) * */ /* the font program, if any */ FT_ULong font_program_size; FT_Byte* font_program; /* the cvt program, if any */ FT_ULong cvt_program_size; FT_Byte* cvt_program; /* the original, unscaled, control value table */ FT_ULong cvt_size; FT_Int32* cvt; /************************************************************************** * * Other tables or fields. This is used by derivative formats like * OpenType. * */ FT_Generic extra; const char* postscript_name; FT_ULong glyf_len; FT_ULong glyf_offset; /* since 2.7.1 */ FT_Bool is_cff2; /* since 2.7.1 */ #ifdef TT_CONFIG_OPTION_GX_VAR_SUPPORT FT_Bool doblend; GX_Blend blend; FT_UInt32 variation_support; /* since 2.7.1 */ const char* var_postscript_prefix; /* since 2.7.2 */ FT_UInt var_postscript_prefix_len; /* since 2.7.2 */ FT_UInt var_default_named_instance; /* since 2.13.1 */ const char* non_var_style_name; /* since 2.13.1 */ #endif /* since version 2.2 */ FT_ULong horz_metrics_size; FT_ULong vert_metrics_size; FT_ULong num_locations; /* up to 0xFFFF + 1 */ FT_Byte* glyph_locations; FT_Byte* hdmx_table; FT_ULong hdmx_table_size; FT_UInt hdmx_record_count; FT_ULong hdmx_record_size; FT_Byte** hdmx_records; FT_Byte* sbit_table; FT_ULong sbit_table_size; TT_SbitTableType sbit_table_type; FT_UInt sbit_num_strikes; FT_UInt* sbit_strike_map; FT_Byte* kern_table; FT_ULong kern_table_size; FT_UInt num_kern_tables; FT_UInt32 kern_avail_bits; FT_UInt32 kern_order_bits; #ifdef TT_CONFIG_OPTION_BDF TT_BDFRec bdf; #endif /* TT_CONFIG_OPTION_BDF */ /* since 2.3.0 */ FT_ULong horz_metrics_offset; FT_ULong vert_metrics_offset; #ifdef TT_CONFIG_OPTION_EMBEDDED_BITMAPS /* since 2.7 */ FT_ULong ebdt_start; /* either `CBDT', `EBDT', or `bdat' */ FT_ULong ebdt_size; #endif /* since 2.10 */ void* cpal; void* colr; /* since 2.12 */ void* svg; #ifdef TT_CONFIG_OPTION_GPOS_KERNING /* since 2.13.3 */ FT_Byte* gpos_table; /* since 2.14 */ /* This is actually an array of GPOS lookup subtables. */ FT_UInt32* gpos_lookups_kerning; FT_UInt num_gpos_lookups_kerning; #endif } TT_FaceRec;

## *struct TT_Gasp_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | version | |
| FT_UShort | numRanges | |
| TT_GaspRange | gaspRanges | |

**Definition:**

typedef struct TT_Gasp_ { FT_UShort version; FT_UShort numRanges; TT_GaspRange gaspRanges; } TT_GaspRec;

## *struct TT_GlyphZoneRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | n_points | |
| FT_UShort | n_contours | |
| FT_Vector* | org | |
| FT_Vector* | cur | |
| FT_Vector* | orus | |
| FT_Byte* | tags | |
| FT_UShort* | contours | |
| FT_UShort | first_point | |

**Definition:**

typedef struct TT_GlyphZoneRec_ { FT_UShort n_points; /* number of points in zone */ FT_UShort n_contours; /* number of contours */ FT_Vector* org; /* original point coordinates */ FT_Vector* cur; /* current point coordinates */ FT_Vector* orus; /* original (unscaled) point coordinates */ FT_Byte* tags; /* current touch flags */ FT_UShort* contours; /* contour end points */ FT_UShort first_point; /* offset of first (#0) point */ } TT_GlyphZoneRec, *TT_GlyphZone;

## *struct TT_LangTagRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | stringLength | |
| FT_ULong | stringOffset | |
| FT_Byte* | string | |

**Definition:**

typedef struct TT_LangTagRec_ { FT_UShort stringLength; FT_ULong stringOffset; /* this last field is not defined in the spec */ /* but used by the FreeType engine */ FT_Byte* string; } TT_LangTagRec, *TT_LangTag;

## *struct TT_LongMetricsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | advance | |
| FT_Short | bearing | |

**Definition:**

typedef struct TT_LongMetricsRec_ { FT_UShort advance; FT_Short bearing; } TT_LongMetricsRec, *TT_LongMetrics;

## *struct TT_NameRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | platformID | |
| FT_UShort | encodingID | |
| FT_UShort | languageID | |
| FT_UShort | nameID | |
| FT_UShort | stringLength | |
| FT_ULong | stringOffset | |
| FT_Byte* | string | |

**Definition:**
typedef struct TT_NameRec_ { FT_UShort platformID; FT_UShort encodingID; FT_UShort languageID; FT_UShort nameID; FT_UShort stringLength; FT_ULong stringOffset; /* this last field is not defined in the spec */ /* but used by the FreeType engine */ FT_Byte* string; } TT_NameRec, *TT_Name;

## *struct TT_NameTableRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | format | |
| FT_UInt | numNameRecords | |
| FT_UInt | storageOffset | |
| TT_NameRec* | names | |
| FT_UInt | numLangTagRecords | |
| TT_LangTagRec* | langTags | |
| FT_Stream | stream | |

**Definition:**
typedef struct TT_NameTableRec_ { FT_UShort format; FT_UInt numNameRecords; FT_UInt storageOffset; TT_NameRec* names; FT_UInt numLangTagRecords; TT_LangTagRec* langTags; FT_Stream stream; } TT_NameTableRec, *TT_NameTable;

## *struct TT_SBit_ComponentRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | glyph_code | |
| FT_Char | x_offset | |
| FT_Char | y_offset | |

**Definition:**
typedef struct TT_SBit_ComponentRec_ { FT_UShort glyph_code; FT_Char x_offset; FT_Char y_offset; } TT_SBit_ComponentRec, *TT_SBit_Component;

## *struct TT_SBit_LineMetricsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Char | ascender | |
| FT_Char | descender | |
| FT_Byte | max_width | |
| FT_Char | caret_slope_numerator | |
| FT_Char | caret_slope_denominator | |
| FT_Char | caret_offset | |
| FT_Char | min_origin_SB | |
| FT_Char | min_advance_SB | |
| FT_Char | max_before_BL | |
| FT_Char | min_after_BL | |
| FT_Char | pads | |

**Definition:**

typedef struct TT_SBit_LineMetricsRec_ { FT_Char ascender; FT_Char descender; FT_Byte max_width; FT_Char caret_slope_numerator; FT_Char caret_slope_denominator; FT_Char caret_offset; FT_Char min_origin_SB; FT_Char min_advance_SB; FT_Char max_before_BL; FT_Char min_after_BL; FT_Char pads[2]; } TT_SBit_LineMetricsRec, *TT_SBit_LineMetrics;

## *struct TT_SBit_RangeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | first_glyph | |
| FT_UShort | last_glyph | |
| FT_UShort | index_format | |
| FT_UShort | image_format | |
| FT_ULong | image_offset | |
| FT_ULong | image_size | |
| TT_SBit_MetricsRec | metrics | |
| FT_ULong | num_glyphs | |
| FT_ULong* | glyph_offsets | |
| FT_UShort* | glyph_codes | |
| FT_ULong | table_offset | |

**Definition:**

typedef struct TT_SBit_RangeRec_ { FT_UShort first_glyph; FT_UShort last_glyph; FT_UShort index_format; FT_UShort image_format; FT_ULong image_offset; FT_ULong image_size; TT_SBit_MetricsRec metrics; FT_ULong num_glyphs; FT_ULong* glyph_offsets; FT_UShort*

glyph_codes; FT_ULong table_offset; } TT_SBit_RangeRec, *TT_SBit_Range;

## *struct TT_SBit_ScaleRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| TT_SBit_LineMetricsRec | hori | |
| TT_SBit_LineMetricsRec | vert | |
| FT_Byte | x_ppem | |
| FT_Byte | y_ppem | |
| FT_Byte | x_ppem_substitute | |
| FT_Byte | y_ppem_substitute | |

**Definition:**

typedef struct TT_SBit_ScaleRec_ { TT_SBit_LineMetricsRec hori; TT_SBit_LineMetricsRec vert; FT_Byte x_ppem; FT_Byte y_ppem; FT_Byte x_ppem_substitute; FT_Byte y_ppem_substitute; } TT_SBit_ScaleRec, *TT_SBit_Scale;

## *struct TT_SBit_Small_Metrics_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte | height | |
| FT_Byte | width | |
| FT_Char | bearingX | |
| FT_Char | bearingY | |
| FT_Byte | advance | |

**Definition:**

typedef struct TT_SBit_Small_Metrics_ { FT_Byte height; FT_Byte width; FT_Char bearingX; FT_Char bearingY; FT_Byte advance; } TT_SBit_SmallMetricsRec, *TT_SBit_SmallMetrics;

## *struct TT_SBit_StrikeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Int | num_ranges | |
| TT_SBit_Range | sbit_ranges | |
| FT_ULong | ranges_offset | |
| FT_ULong | color_ref | |
| TT_SBit_LineMetricsRec | hori | |
| TT_SBit_LineMetricsRec | vert | |

| | | |
|---|---|---|
| FT_UShort | start_glyph | |
| FT_UShort | end_glyph | |
| FT_Byte | x_ppem | |
| FT_Byte | y_ppem | |
| FT_Byte | bit_depth | |
| FT_Char | flags | |

**Definition:**
typedef struct TT_SBit_StrikeRec_ { FT_Int num_ranges; TT_SBit_Range sbit_ranges; FT_ULong ranges_offset; FT_ULong color_ref; TT_SBit_LineMetricsRec hori; TT_SBit_LineMetricsRec vert; FT_UShort start_glyph; FT_UShort end_glyph; FT_Byte x_ppem; FT_Byte y_ppem; FT_Byte bit_depth; FT_Char flags; } TT_SBit_StrikeRec, *TT_SBit_Strike;

## *typedef TT_ShortMetrics*

**Definition:**
typedef FT_Short TT_ShortMetrics;

## *struct TT_SizeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| TT_Face | face | |
| TT_Size | size | |
| FT_GlyphSlot | glyph | |
| FT_GlyphLoader | gloader | |
| FT_ULong | load_flags | |
| FT_UInt | glyph_index | |
| FT_Stream | stream | |
| FT_UInt | byte_len | |
| FT_Short | n_contours | |
| FT_BBox | bbox | |
| FT_Int | left_bearing | |
| FT_Int | advance | |
| FT_Int | linear | |
| FT_Bool | linear_def | |
| FT_Vector | pp1 | |
| FT_Vector | pp2 | |
| TT_GlyphZoneRec | base | |
| TT_GlyphZoneRec | zone | |

| | | |
|---|---|---|
| TT_ExecContext | exec | |
| FT_ULong | ins_pos | |
| void* | other | |
| FT_Int | top_bearing | |
| FT_Int | vadvance | |
| FT_Vector | pp3 | |
| FT_Vector | pp4 | |
| FT_Byte* | cursor | |
| FT_Byte* | limit | |
| FT_ListRec | composites | |
| FT_Byte* | widthp | |

**Definition:**
typedef struct TT_SizeRec_* TT_Size;

## *struct TT_TableRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ULong | Tag | |
| FT_ULong | CheckSum | |
| FT_ULong | Offset | |
| FT_ULong | Length | |

**Definition:**
typedef struct TT_TableRec_ { FT_ULong Tag; /* table type */ FT_ULong CheckSum; /* table checksum */ FT_ULong Offset; /* table file offset */ FT_ULong Length; /* table length */ } TT_TableRec, *TT_Table;

## *typedef height*

**Definition:**
* @struct: * TT_SBit_MetricsRec * * @description: * A structure used to hold the big metrics of a given glyph bitmap in a * TrueType or OpenType font. These are usually found in the 'EBDT' * (Microsoft) or 'bloc' (Apple) table. * * @fields: * height :: * The glyph height in pixels. * * width :: * The glyph width in pixels. * * horiBearingX :: * The horizontal left bearing. * * horiBearingY :: * The horizontal top bearing. * * horiAdvance :: * The horizontal advance. * * vertBearingX :: * The vertical left bearing. * * vertBearingY :: * The vertical top bearing. * * vertAdvance :: * The vertical advance. */ typedef struct TT_SBit_MetricsRec_ { FT_UShort height; FT_UShort width; FT_Short horiBearingX; FT_Short horiBearingY; FT_UShort horiAdvance; FT_Short vertBearingX; FT_Short vertBearingY; FT_UShort vertAdvance; } TT_SBit_MetricsRec, *TT_SBit_Metrics;

## *typedef loaded*

**Definition:**

* @struct: * TT_Post_NamesRec * * @description: * Postscript names table, either format 2.0 or 2.5. * * @fields: * loaded :: * A flag to indicate whether the PS names are loaded. * * num_glyphs :: * The number of named glyphs in the table. * * num_names :: * The number of PS names stored in the table. * * glyph_indices :: * The indices of the glyphs in the names arrays. * * glyph_names :: * The PS names not in Mac Encoding. */ typedef struct TT_Post_NamesRec_ { FT_Bool loaded; FT_UShort num_glyphs; FT_UShort num_names; FT_UShort* glyph_indices; FT_Byte** glyph_names; } TT_Post_NamesRec, *TT_Post_Names;

## *typedef maxPPEM*

**Definition:**
* @struct: * TT_GaspRangeRec * * @description: * A tiny structure used to model a gasp range according to the TrueType * specification. * * @fields: * maxPPEM :: * The maximum ppem value to which `gaspFlag` applies. * * gaspFlag :: * A flag describing the grid-fitting and anti-aliasing modes to be * used. */ typedef struct TT_GaspRangeRec_ { FT_UShort maxPPEM; FT_UShort gaspFlag; } TT_GaspRangeRec, *TT_GaspRange;

## *typedef tag*

**Definition:**
* @struct: * TTC_HeaderRec * * @description: * TrueType collection header. This table contains the offsets of the * font headers of each distinct TrueType face in the file. * * @fields: * tag :: * Must be 'ttc~' to indicate a TrueType collection. * * version :: * The version number. * * count :: * The number of faces in the collection. The specification says this * should be an unsigned long, but we use a signed long since we need * the value -1 for specific purposes. * * offsets :: * The offsets of the font headers, one per face. */ typedef struct TTC_HeaderRec_ { FT_ULong tag; FT_Fixed version; FT_Long count; FT_ULong* offsets; } TTC_HeaderRec;

# wofftypes.h

## *Data Types*

### *struct WOFF2_HeaderRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ULong | signature | |
| FT_ULong | flavor | |
| FT_ULong | length | |
| FT_UShort | num_tables | |
| FT_ULong | totalSfntSize | |
| FT_ULong | totalCompressedSize | |
| FT_ULong | metaOffset | |
| FT_ULong | metaLength | |
| FT_ULong | metaOrigLength | |
| FT_ULong | privOffset | |
| FT_ULong | privLength | |
| FT_ULong | uncompressed_size | |
| FT_ULong | compressed_offset | |
| FT_ULong | header_version | |
| FT_UShort | num_fonts | |
| FT_ULong | actual_sfnt_size | |
| WOFF2_TtcFont | ttc_fonts | |

**Definition:**

typedef struct WOFF2_HeaderRec_ { FT_ULong signature; FT_ULong flavor; FT_ULong length; FT_UShort num_tables; FT_ULong totalSfntSize; FT_ULong totalCompressedSize; FT_ULong metaOffset; FT_ULong metaLength; FT_ULong metaOrigLength; FT_ULong privOffset; FT_ULong privLength; FT_ULong uncompressed_size; /* uncompressed brotli stream size */ FT_ULong compressed_offset; /* compressed stream offset */ FT_ULong header_version; /* version of original TTC Header */ FT_UShort num_fonts; /* number of fonts in TTC */ FT_ULong actual_sfnt_size; /* actual size of sfnt stream */ WOFF2_TtcFont ttc_fonts; /* metadata for fonts in a TTC */ } WOFF2_HeaderRec, *WOFF2_Header;

### *struct WOFF2_InfoRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ULong | header_checksum | |

| | | |
|---|---|---|
| FT_UShort | num_glyphs | |
| FT_UShort | num_hmetrics | |
| FT_Short* | x_mins | |
| WOFF2_Table | glyf_table | |
| WOFF2_Table | loca_table | |
| WOFF2_Table | head_table | |

**Definition:**
typedef struct WOFF2_InfoRec_ { FT_ULong header_checksum; FT_UShort num_glyphs; FT_UShort num_hmetrics; FT_Short* x_mins; WOFF2_Table glyf_table; WOFF2_Table loca_table; WOFF2_Table head_table; } WOFF2_InfoRec, *WOFF2_Info;

## *struct WOFF2_PointRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Int | x | |
| FT_Int | y | |
| FT_Bool | on_curve | |

**Definition:**
typedef struct WOFF2_PointRec_ { FT_Int x; FT_Int y; FT_Bool on_curve; } WOFF2_PointRec, *WOFF2_Point;

## *struct WOFF2_SubstreamRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ULong | start | |
| FT_ULong | offset | |
| FT_ULong | size | |

**Definition:**
typedef struct WOFF2_SubstreamRec_ { FT_ULong start; FT_ULong offset; FT_ULong size; } WOFF2_SubstreamRec, *WOFF2_Substream;

## *struct WOFF2_TableRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Byte | FlagByte | |
| FT_Tag | Tag | |
| FT_ULong | dst_length | |
| FT_ULong | TransformLength | |

| | | |
|---|---|---|
| FT_ULong | flags | |
| FT_ULong | src_offset | |
| FT_ULong | src_length | |
| FT_ULong | dst_offset | |

**Definition:**
typedef struct WOFF2_TableRec_ { FT_Byte FlagByte; /* table type and flags */ FT_Tag Tag; /* table file offset */ FT_ULong dst_length; /* uncompressed table length */ FT_ULong TransformLength; /* transformed length */ FT_ULong flags; /* calculated flags */ FT_ULong src_offset; /* compressed table offset */ FT_ULong src_length; /* compressed table length */ FT_ULong dst_offset; /* uncompressed table offset */ } WOFF2_TableRec, *WOFF2_Table;

## *struct WOFF2_TtcFontRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_ULong | flavor | |
| FT_UShort | num_tables | |
| FT_UShort* | table_indices | |

**Definition:**
typedef struct WOFF2_TtcFontRec_ { FT_ULong flavor; FT_UShort num_tables; FT_UShort* table_indices; } WOFF2_TtcFontRec, *WOFF2_TtcFont;

## *struct WOFF_TableRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Tag | Tag | |
| FT_ULong | Offset | |
| FT_ULong | CompLength | |
| FT_ULong | OrigLength | |
| FT_ULong | CheckSum | |
| FT_ULong | OrigOffset | |

**Definition:**
typedef struct WOFF_TableRec_ { FT_Tag Tag; /* table ID */ FT_ULong Offset; /* table file offset */ FT_ULong CompLength; /* compressed table length */ FT_ULong OrigLength; /* uncompressed table length */ FT_ULong CheckSum; /* uncompressed checksum */ FT_ULong OrigOffset; /* uncompressed table file offset */ /* (not in the WOFF file) */ } WOFF_TableRec, *WOFF_Table;

## *typedef signature*

**Definition:**
* @struct: * WOFF_HeaderRec * * @description: * WOFF file format header. * * @fields: * See * * https://www.w3.org/TR/WOFF/#WOFFHeader */ typedef struct WOFF_HeaderRec_ { FT_ULong

signature; FT_ULong flavor; FT_ULong length; FT_UShort num_tables; FT_UShort reserved;
FT_ULong totalSfntSize; FT_UShort majorVersion; FT_UShort minorVersion; FT_ULong metaOffset;
FT_ULong metaLength; FT_ULong metaOrigLength; FT_ULong privOffset; FT_ULong privLength; }
WOFF_HeaderRec, *WOFF_Header;

# svttcmap.h

## *Data Types*

## *typedef language*

**Definition:**
* @struct: * TT_CMapInfo * * @description: * A structure used to store TrueType/sfnt specific cmap information * which is not covered by the generic @FT_CharMap structure. This * structure can be accessed with the @FT_Get_TT_CMap_Info function. * * @fields: * language :: * The language ID used in Mac fonts. Definitions of values are in * `ttnameid.h`. * * format :: * The cmap format. OpenType 1.6 defines the formats 0 (byte encoding * table), 2~(high-byte mapping through table), 4~(segment mapping to * delta values), 6~(trimmed table mapping), 8~(mixed 16-bit and 32-bit * coverage), 10~(trimmed array), 12~(segmented coverage), 13~(last * resort font), and 14 (Unicode Variation Sequences). */ typedef struct TT_CMapInfo_ { FT_ULong language; FT_Long format; } TT_CMapInfo;

# afblue.h

## *Data Types*

### *enum AF_Blue_String_*

**Definition:**
typedef enum AF_Blue_String_ { AF_BLUE_STRING_ADLAM_CAPITAL_TOP = 0,
AF_BLUE_STRING_ADLAM_CAPITAL_BOTTOM = 30, AF_BLUE_STRING_ADLAM_SMALL_TOP =
40, AF_BLUE_STRING_ADLAM_SMALL_BOTTOM = 65, AF_BLUE_STRING_ARABIC_TOP = 105,
AF_BLUE_STRING_ARABIC_BOTTOM = 123, AF_BLUE_STRING_ARABIC_JOIN = 138,
AF_BLUE_STRING_ARMENIAN_CAPITAL_TOP = 141,
AF_BLUE_STRING_ARMENIAN_CAPITAL_BOTTOM = 165,
AF_BLUE_STRING_ARMENIAN_SMALL_ASCENDER = 189,
AF_BLUE_STRING_ARMENIAN_SMALL_TOP = 210,
AF_BLUE_STRING_ARMENIAN_SMALL_BOTTOM = 234,
AF_BLUE_STRING_ARMENIAN_SMALL_DESCENDER = 258, AF_BLUE_STRING_AVESTAN_TOP
= 282, AF_BLUE_STRING_AVESTAN_BOTTOM = 302, AF_BLUE_STRING_BAMUM_TOP = 312,
AF_BLUE_STRING_BAMUM_BOTTOM = 344, AF_BLUE_STRING_BENGALI_BASE = 376,
AF_BLUE_STRING_BENGALI_TOP = 408, AF_BLUE_STRING_BENGALI_HEAD = 436,
AF_BLUE_STRING_BUHID_TOP = 468, AF_BLUE_STRING_BUHID_LARGE = 476,
AF_BLUE_STRING_BUHID_SMALL = 488, AF_BLUE_STRING_BUHID_BOTTOM = 504,
AF_BLUE_STRING_CANADIAN_SYLLABICS_TOP = 532,
AF_BLUE_STRING_CANADIAN_SYLLABICS_BOTTOM = 564,
AF_BLUE_STRING_CANADIAN_SYLLABICS_SMALL_TOP = 596,
AF_BLUE_STRING_CANADIAN_SYLLABICS_SMALL_BOTTOM = 628,
AF_BLUE_STRING_CANADIAN_SYLLABICS_SUPS_TOP = 660,
AF_BLUE_STRING_CANADIAN_SYLLABICS_SUPS_BOTTOM = 688,
AF_BLUE_STRING_CARIAN_TOP = 720, AF_BLUE_STRING_CARIAN_BOTTOM = 760,
AF_BLUE_STRING_CHAKMA_TOP = 795, AF_BLUE_STRING_CHAKMA_BOTTOM = 820,
AF_BLUE_STRING_CHAKMA_DESCENDER = 845, AF_BLUE_STRING_CHEROKEE_CAPITAL =
910, AF_BLUE_STRING_CHEROKEE_SMALL_ASCENDER = 942,
AF_BLUE_STRING_CHEROKEE_SMALL = 974,
AF_BLUE_STRING_CHEROKEE_SMALL_DESCENDER = 1006,
AF_BLUE_STRING_COPTIC_CAPITAL_TOP = 1022,
AF_BLUE_STRING_COPTIC_CAPITAL_BOTTOM = 1054,
AF_BLUE_STRING_COPTIC_SMALL_TOP = 1086, AF_BLUE_STRING_COPTIC_SMALL_BOTTOM
= 1118, AF_BLUE_STRING_CYPRIOT_TOP = 1150, AF_BLUE_STRING_CYPRIOT_BOTTOM =
1190, AF_BLUE_STRING_CYPRIOT_SMALL = 1225, AF_BLUE_STRING_CYRILLIC_CAPITAL_TOP
= 1240, AF_BLUE_STRING_CYRILLIC_CAPITAL_BOTTOM = 1264,
AF_BLUE_STRING_CYRILLIC_SMALL = 1288,
AF_BLUE_STRING_CYRILLIC_SMALL_DESCENDER = 1312,
AF_BLUE_STRING_DESERET_CAPITAL_TOP = 1321,
AF_BLUE_STRING_DESERET_CAPITAL_BOTTOM = 1346,
AF_BLUE_STRING_DESERET_SMALL_TOP = 1371,
AF_BLUE_STRING_DESERET_SMALL_BOTTOM = 1396,
AF_BLUE_STRING_DEVANAGARI_BASE = 1421, AF_BLUE_STRING_DEVANAGARI_TOP = 1453,
AF_BLUE_STRING_DEVANAGARI_HEAD = 1485, AF_BLUE_STRING_DEVANAGARI_BOTTOM =
1517, AF_BLUE_STRING_ETHIOPIC_TOP = 1525, AF_BLUE_STRING_ETHIOPIC_BOTTOM =
1557, AF_BLUE_STRING_GEORGIAN_MKHEDRULI_TOP = 1589,
AF_BLUE_STRING_GEORGIAN_MKHEDRULI_BOTTOM = 1621,

AF_BLUE_STRING_GEORGIAN_MKHEDRULI_ASCENDER = 1653,
AF_BLUE_STRING_GEORGIAN_MKHEDRULI_DESCENDER = 1685,
AF_BLUE_STRING_GEORGIAN_ASOMTAVRULI_TOP = 1717,
AF_BLUE_STRING_GEORGIAN_ASOMTAVRULI_BOTTOM = 1749,
AF_BLUE_STRING_GEORGIAN_NUSKHURI_TOP = 1781,
AF_BLUE_STRING_GEORGIAN_NUSKHURI_BOTTOM = 1813,
AF_BLUE_STRING_GEORGIAN_NUSKHURI_ASCENDER = 1845,
AF_BLUE_STRING_GEORGIAN_NUSKHURI_DESCENDER = 1877,
AF_BLUE_STRING_GEORGIAN_MTAVRULI_TOP = 1909,
AF_BLUE_STRING_GEORGIAN_MTAVRULI_BOTTOM = 1941,
AF_BLUE_STRING_GLAGOLITIC_CAPITAL_TOP = 1973,
AF_BLUE_STRING_GLAGOLITIC_CAPITAL_BOTTOM = 2005,
AF_BLUE_STRING_GLAGOLITIC_SMALL_TOP = 2037,
AF_BLUE_STRING_GLAGOLITIC_SMALL_BOTTOM = 2069, AF_BLUE_STRING_GOTHIC_TOP = 2101, AF_BLUE_STRING_GOTHIC_BOTTOM = 2141, AF_BLUE_STRING_GREEK_CAPITAL_TOP = 2161, AF_BLUE_STRING_GREEK_CAPITAL_BOTTOM = 2182,
AF_BLUE_STRING_GREEK_SMALL_BETA_TOP = 2200, AF_BLUE_STRING_GREEK_SMALL = 2218, AF_BLUE_STRING_GREEK_SMALL_DESCENDER = 2242,
AF_BLUE_STRING_GUJARATI_TOP = 2266, AF_BLUE_STRING_GUJARATI_BOTTOM = 2298,
AF_BLUE_STRING_GUJARATI_ASCENDER = 2330, AF_BLUE_STRING_GUJARATI_DESCENDER = 2380, AF_BLUE_STRING_GUJARATI_DIGIT_TOP = 2413,
AF_BLUE_STRING_GURMUKHI_BASE = 2433, AF_BLUE_STRING_GURMUKHI_HEAD = 2465,
AF_BLUE_STRING_GURMUKHI_TOP = 2497, AF_BLUE_STRING_GURMUKHI_BOTTOM = 2529,
AF_BLUE_STRING_GURMUKHI_DIGIT_TOP = 2561, AF_BLUE_STRING_HEBREW_TOP = 2581,
AF_BLUE_STRING_HEBREW_BOTTOM = 2605, AF_BLUE_STRING_HEBREW_DESCENDER = 2623, AF_BLUE_STRING_KANNADA_TOP = 2638, AF_BLUE_STRING_KANNADA_BOTTOM = 2682, AF_BLUE_STRING_KAYAH_LI_TOP = 2714, AF_BLUE_STRING_KAYAH_LI_BOTTOM = 2738, AF_BLUE_STRING_KAYAH_LI_ASCENDER = 2758,
AF_BLUE_STRING_KAYAH_LI_DESCENDER = 2766,
AF_BLUE_STRING_KAYAH_LI_LARGE_DESCENDER = 2778, AF_BLUE_STRING_KHMER_TOP = 2799, AF_BLUE_STRING_KHMER_SUBSCRIPT_TOP = 2823,
AF_BLUE_STRING_KHMER_BOTTOM = 2863, AF_BLUE_STRING_KHMER_DESCENDER = 2895,
AF_BLUE_STRING_KHMER_LARGE_DESCENDER = 2929,
AF_BLUE_STRING_KHMER_SYMBOLS_WAXING_TOP = 3016,
AF_BLUE_STRING_KHMER_SYMBOLS_WANING_BOTTOM = 3024,
AF_BLUE_STRING_LAO_TOP = 3032, AF_BLUE_STRING_LAO_BOTTOM = 3064,
AF_BLUE_STRING_LAO_ASCENDER = 3096, AF_BLUE_STRING_LAO_LARGE_ASCENDER = 3112, AF_BLUE_STRING_LAO_DESCENDER = 3124, AF_BLUE_STRING_LATIN_CAPITAL_TOP = 3148, AF_BLUE_STRING_LATIN_CAPITAL_BOTTOM = 3164,
AF_BLUE_STRING_LATIN_SMALL_F_TOP = 3180, AF_BLUE_STRING_LATIN_SMALL_TOP = 3194, AF_BLUE_STRING_LATIN_SMALL_BOTTOM = 3210,
AF_BLUE_STRING_LATIN_SMALL_DESCENDER = 3226,
AF_BLUE_STRING_LATIN_SUBS_CAPITAL_TOP = 3236,
AF_BLUE_STRING_LATIN_SUBS_CAPITAL_BOTTOM = 3256,
AF_BLUE_STRING_LATIN_SUBS_SMALL_F_TOP = 3276,
AF_BLUE_STRING_LATIN_SUBS_SMALL = 3296,
AF_BLUE_STRING_LATIN_SUBS_SMALL_DESCENDER = 3332,
AF_BLUE_STRING_LATIN_SUPS_CAPITAL_TOP = 3352,
AF_BLUE_STRING_LATIN_SUPS_CAPITAL_BOTTOM = 3383,
AF_BLUE_STRING_LATIN_SUPS_SMALL_F_TOP = 3412,
AF_BLUE_STRING_LATIN_SUPS_SMALL = 3438,
AF_BLUE_STRING_LATIN_SUPS_SMALL_DESCENDER = 3463, AF_BLUE_STRING_LISU_TOP = 3474, AF_BLUE_STRING_LISU_BOTTOM = 3506, AF_BLUE_STRING_MALAYALAM_TOP = 3538,

AF_BLUE_STRING_MALAYALAM_BOTTOM = 3582,
AF_BLUE_STRING_MEDEFAIDRIN_CAPITAL_TOP = 3614,
AF_BLUE_STRING_MEDEFAIDRIN_CAPITAL_BOTTOM = 3649,
AF_BLUE_STRING_MEDEFAIDRIN_SMALL_F_TOP = 3689,
AF_BLUE_STRING_MEDEFAIDRIN_SMALL_TOP = 3719,
AF_BLUE_STRING_MEDEFAIDRIN_SMALL_BOTTOM = 3749,
AF_BLUE_STRING_MEDEFAIDRIN_SMALL_DESCENDER = 3779,
AF_BLUE_STRING_MEDEFAIDRIN_DIGIT_TOP = 3794,
AF_BLUE_STRING_MONGOLIAN_TOP_BASE = 3819,
AF_BLUE_STRING_MONGOLIAN_BOTTOM_BASE = 3863, AF_BLUE_STRING_MYANMAR_TOP = 3867, AF_BLUE_STRING_MYANMAR_BOTTOM = 3899,
AF_BLUE_STRING_MYANMAR_ASCENDER = 3931,
AF_BLUE_STRING_MYANMAR_DESCENDER = 3959, AF_BLUE_STRING_NKO_TOP = 3991,
AF_BLUE_STRING_NKO_BOTTOM = 4015, AF_BLUE_STRING_NKO_SMALL_TOP = 4030,
AF_BLUE_STRING_NKO_SMALL_BOTTOM = 4039, AF_BLUE_STRING_OL_CHIKI = 4051,
AF_BLUE_STRING_OLD_TURKIC_TOP = 4075, AF_BLUE_STRING_OLD_TURKIC_BOTTOM = 4090, AF_BLUE_STRING_OSAGE_CAPITAL_TOP = 4110,
AF_BLUE_STRING_OSAGE_CAPITAL_BOTTOM = 4150,
AF_BLUE_STRING_OSAGE_CAPITAL_DESCENDER = 4180,
AF_BLUE_STRING_OSAGE_SMALL_TOP = 4195, AF_BLUE_STRING_OSAGE_SMALL_BOTTOM = 4235, AF_BLUE_STRING_OSAGE_SMALL_ASCENDER = 4275,
AF_BLUE_STRING_OSAGE_SMALL_DESCENDER = 4300, AF_BLUE_STRING_OSMANYA_TOP = 4315, AF_BLUE_STRING_OSMANYA_BOTTOM = 4355, AF_BLUE_STRING_ROHINGYA_TOP = 4395, AF_BLUE_STRING_ROHINGYA_BOTTOM = 4420, AF_BLUE_STRING_ROHINGYA_JOIN = 4445, AF_BLUE_STRING_SAURASHTRA_TOP = 4448,
AF_BLUE_STRING_SAURASHTRA_BOTTOM = 4480, AF_BLUE_STRING_SHAVIAN_TOP = 4500,
AF_BLUE_STRING_SHAVIAN_BOTTOM = 4510, AF_BLUE_STRING_SHAVIAN_DESCENDER = 4535, AF_BLUE_STRING_SHAVIAN_SMALL_TOP = 4545,
AF_BLUE_STRING_SHAVIAN_SMALL_BOTTOM = 4580, AF_BLUE_STRING_SINHALA_TOP = 4595, AF_BLUE_STRING_SINHALA_BOTTOM = 4627, AF_BLUE_STRING_SINHALA_DESCENDER = 4659, AF_BLUE_STRING_SUNDANESE_TOP = 4703,
AF_BLUE_STRING_SUNDANESE_BOTTOM = 4727,
AF_BLUE_STRING_SUNDANESE_DESCENDER = 4759, AF_BLUE_STRING_TAI_VIET_TOP = 4767, AF_BLUE_STRING_TAI_VIET_BOTTOM = 4787, AF_BLUE_STRING_TAMIL_TOP = 4799,
AF_BLUE_STRING_TAMIL_BOTTOM = 4831, AF_BLUE_STRING_TELUGU_TOP = 4863,
AF_BLUE_STRING_TELUGU_BOTTOM = 4891, AF_BLUE_STRING_THAI_TOP = 4919,
AF_BLUE_STRING_THAI_BOTTOM = 4943, AF_BLUE_STRING_THAI_ASCENDER = 4971,
AF_BLUE_STRING_THAI_LARGE_ASCENDER = 4983, AF_BLUE_STRING_THAI_DESCENDER = 4995, AF_BLUE_STRING_THAI_LARGE_DESCENDER = 5011,
AF_BLUE_STRING_THAI_DIGIT_TOP = 5019, AF_BLUE_STRING_TIFINAGH = 5031,
AF_BLUE_STRING_VAI_TOP = 5063, AF_BLUE_STRING_VAI_BOTTOM = 5095, af_blue_1_1 = 5126, #ifdef AF_CONFIG_OPTION_CJK AF_BLUE_STRING_CJK_TOP = af_blue_1_1 + 1,

## *enum AF_Blue_Stringset_*

**Definition:**
typedef enum AF_Blue_Stringset_ { AF_BLUE_STRINGSET_ADLM = 0,
AF_BLUE_STRINGSET_ARAB = 5, AF_BLUE_STRINGSET_ARMN = 9,
AF_BLUE_STRINGSET_AVST = 16, AF_BLUE_STRINGSET_BAMU = 19,
AF_BLUE_STRINGSET_BENG = 22, AF_BLUE_STRINGSET_BUHD = 27,
AF_BLUE_STRINGSET_CAKM = 32, AF_BLUE_STRINGSET_CANS = 36,
AF_BLUE_STRINGSET_CARI = 43, AF_BLUE_STRINGSET_CHER = 46,

```
AF_BLUE_STRINGSET_COPT = 53, AF_BLUE_STRINGSET_CPRT = 58,
AF_BLUE_STRINGSET_CYRL = 63, AF_BLUE_STRINGSET_DEVA = 69,
AF_BLUE_STRINGSET_DSRT = 75, AF_BLUE_STRINGSET_ETHI = 80,
AF_BLUE_STRINGSET_GEOR = 83, AF_BLUE_STRINGSET_GEOK = 90,
AF_BLUE_STRINGSET_GLAG = 97, AF_BLUE_STRINGSET_GOTH = 102,
AF_BLUE_STRINGSET_GREK = 105, AF_BLUE_STRINGSET_GUJR = 112,
AF_BLUE_STRINGSET_GURU = 118, AF_BLUE_STRINGSET_HEBR = 124,
AF_BLUE_STRINGSET_KNDA = 128, AF_BLUE_STRINGSET_KALI = 131,
AF_BLUE_STRINGSET_KHMR = 137, AF_BLUE_STRINGSET_KHMS = 143,
AF_BLUE_STRINGSET_LAO = 146, AF_BLUE_STRINGSET_LATN = 152,
AF_BLUE_STRINGSET_LATB = 159, AF_BLUE_STRINGSET_LATP = 166,
AF_BLUE_STRINGSET_LISU = 173, AF_BLUE_STRINGSET_MLYM = 176,
AF_BLUE_STRINGSET_MEDF = 179, AF_BLUE_STRINGSET_MONG = 187,
AF_BLUE_STRINGSET_MYMR = 190, AF_BLUE_STRINGSET_NKOO = 195,
AF_BLUE_STRINGSET_NONE = 200, AF_BLUE_STRINGSET_OLCK = 201,
AF_BLUE_STRINGSET_ORKH = 204, AF_BLUE_STRINGSET_OSGE = 207,
AF_BLUE_STRINGSET_OSMA = 215, AF_BLUE_STRINGSET_ROHG = 218,
AF_BLUE_STRINGSET_SAUR = 222, AF_BLUE_STRINGSET_SHAW = 225,
AF_BLUE_STRINGSET_SINH = 231, AF_BLUE_STRINGSET_SUND = 235,
AF_BLUE_STRINGSET_TAML = 239, AF_BLUE_STRINGSET_TAVT = 242,
AF_BLUE_STRINGSET_TELU = 245, AF_BLUE_STRINGSET_THAI = 248,
AF_BLUE_STRINGSET_TFNG = 256, AF_BLUE_STRINGSET_VAII = 259, af_blue_2_1 = 262, #ifdef
AF_CONFIG_OPTION_CJK AF_BLUE_STRINGSET_HANI = af_blue_2_1 + 0, af_blue_2_1_1 =
af_blue_2_1 + 2, #ifdef AF_CONFIG_OPTION_CJK_BLUE_HANI_VERT af_blue_2_1_2 =
af_blue_2_1_1 + 2, #else af_blue_2_1_2 = af_blue_2_1_1 + 0, #endif /*
AF_CONFIG_OPTION_CJK_BLUE_HANI_VERT */ af_blue_2_2 = af_blue_2_1_2 + 1, #else
af_blue_2_2 = af_blue_2_1 + 0, #endif /* AF_CONFIG_OPTION_CJK */ AF_BLUE_STRINGSET_MAX
/* do not remove */ } AF_Blue_Stringset;
```

# afcjk.h

## *Data Types*

### *struct AF_CJKBlueRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| AF_WidthRec | ref | |
| AF_WidthRec | shoot | |
| FT_UInt | flags | |

**Definition:**

typedef struct AF_CJKBlueRec_ { AF_WidthRec ref; AF_WidthRec shoot; /* undershoot */ FT_UInt flags; } AF_CJKBlueRec, *AF_CJKBlue;

# afglobal.h

## *Data Types*

### *struct AF_FaceGlobalsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Face | face | |
| FT_UInt | glyph_count | |
| FT_UShort* | glyph_styles | |
| hb_font_t* | hb_font | |
| hb_buffer_t* | hb_buf | |
| FT_Byte* | gsub | |
| FT_UInt32* | gsub_lookups_single_alternate | |
| FT_UInt | increase_x_height | |
| AF_StyleMetrics | metrics | |
| FT_UShort | stem_darkening_for_ppem | |
| FT_Pos | standard_vertical_width | |
| FT_Pos | standard_horizontal_width | |
| FT_Pos | darken_x | |
| FT_Pos | darken_y | |
| FT_Fixed | scale_down_factor | |
| AF_Module | module | |

**Definition:**

typedef struct AF_FaceGlobalsRec_ { FT_Face face; FT_UInt glyph_count; /* unsigned face->num_glyphs */ FT_UShort* glyph_styles; #ifdef FT_CONFIG_OPTION_USE_HARFBUZZ hb_font_t* hb_font; hb_buffer_t* hb_buf; /* for feature comparison */ /* The GSUB table. */ FT_Byte* gsub; /* Lookup offsets, with only SingleSubst and AlternateSubst non-NULL. */ FT_UInt32* gsub_lookups_single_alternate; #endif /* per-face auto-hinter properties */ FT_UInt increase_x_height; AF_StyleMetrics metrics[AF_STYLE_MAX]; /* Compute darkening amount once per size. Use this to check whether */ /* darken_{x,y} needs to be recomputed. */ FT_UShort stem_darkening_for_ppem; /* Copy from e.g. AF_LatinMetrics.axis[AF_DIMENSION_HORZ] */ /* to compute the darkening amount. */ FT_Pos standard_vertical_width; /* Copy from e.g. AF_LatinMetrics.axis[AF_DIMENSION_VERT] */ /* to compute the darkening amount. */ FT_Pos standard_horizontal_width; /* The actual amount to darken a glyph along the X axis. */ FT_Pos darken_x; /* The actual amount to darken a glyph along the Y axis. */ FT_Pos darken_y; /* Amount to scale down by to keep emboldened points */ /* on the Y-axis in pre-computed blue zones. */ FT_Fixed scale_down_factor; AF_Module module; /* to access global properties */ } AF_FaceGlobalsRec;

# afhints.h

## *Data Types*

### *enum AF_Dimension_*

**Definition:**
typedef enum AF_Dimension_ { AF_DIMENSION_HORZ = 0, /* x coordinates, */ /* i.e., vertical segments & edges */ AF_DIMENSION_VERT = 1, /* y coordinates, */ /* i.e., horizontal segments & edges */ AF_DIMENSION_MAX /* do not remove */ } AF_Dimension;

# aflatin.h

## *Data Types*

### *struct AF_LatinBlueRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| AF_WidthRec | ref | |
| AF_WidthRec | shoot | |
| FT_Pos | ascender | |
| FT_Pos | descender | |
| FT_UInt | flags | |

**Definition:**

typedef struct AF_LatinBlueRec_ { AF_WidthRec ref; AF_WidthRec shoot; FT_Pos ascender; FT_Pos descender; FT_UInt flags; } AF_LatinBlueRec, *AF_LatinBlue;

# afloader.h

## *Data Types*

## *typedef face*

**Definition:**
* The autofitter module's (global) data structure to communicate with * actual fonts. If necessary, `local' data like the current face, the * current face's auto-hint data, or the current glyph's parameters * relevant to auto-hinting are `swapped in'. Cf. functions like * `af_loader_reset' and `af_loader_load_g'. */ typedef struct AF_LoaderRec_ { /* current face data */ FT_Face face; AF_FaceGlobals globals; /* current glyph data */ AF_GlyphHints hints; AF_StyleMetrics metrics; FT_Bool transformed; FT_Matrix trans_matrix; FT_Vector trans_delta; FT_Vector pp1; FT_Vector pp2; /* we don't handle vertical phantom points */ } AF_LoaderRec, *AF_Loader;

# afmodule.h

## *Data Types*

### *typedef root*

**Definition:**
* This is the `extended' FT_Module structure that holds the * autofitter's global data. */ typedef struct AF_ModuleRec_ { FT_ModuleRec root; FT_UInt fallback_style; AF_Script default_script; FT_Bool no_stem_darkening; FT_Int darken_params[8]; #if defined( FT_CONFIG_OPTION_USE_HARFBUZZ ) && \ defined( FT_CONFIG_OPTION_USE_HARFBUZZ_DYNAMIC ) ft_hb_funcs_t* hb_funcs; #endif } AF_ModuleRec, *AF_Module;

# aftypes.h

## *Data Types*

### *enum AF_Coverage_*

**Definition:**
typedef enum AF_Coverage_ { #include "afcover.h" AF_COVERAGE_DEFAULT } AF_Coverage;

### *struct AF_FaceGlobalsRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| AF_StyleClass | style_class | |
| AF_ScalerRec | scaler | |
| FT_Bool | digits_have_same_width | |
| AF_FaceGlobals | globals | |
| FT_Hash | reverse_charmap | |

**Definition:**
typedef struct AF_FaceGlobalsRec_* AF_FaceGlobals;

### *struct AF_ScalerRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Face | face | |
| FT_Fixed | x_scale | |
| FT_Fixed | y_scale | |
| FT_Pos | x_delta | |
| FT_Pos | y_delta | |
| FT_Render_Mode | render_mode | |
| FT_UInt32 | flags | |

**Definition:**
typedef struct AF_ScalerRec_ { FT_Face face; /* source font face */ FT_Fixed x_scale; /* from font units to 1/64 device pixels */ FT_Fixed y_scale; /* from font units to 1/64 device pixels */ FT_Pos x_delta; /* in 1/64 device pixels */ FT_Pos y_delta; /* in 1/64 device pixels */ FT_Render_Mode render_mode; /* monochrome, anti-aliased, LCD, etc. */ FT_UInt32 flags; /* additional control flags, see above */ } AF_ScalerRec, *AF_Scaler;

### *enum AF_Script_*

**Definition:**

typedef enum AF_Script_ { #include "afscript.h" AF_SCRIPT_MAX /* do not remove */ } AF_Script;

## *typedef AF_Style*

**Definition:**
* The topmost structure for modelling the auto-hinter glyph input data * is a `style class', grouping everything together. */ #undef STYLE #define STYLE( s, S, d, ws, sc, ss, c ) \ AF_STYLE_ ## S, /* The list of known styles. */ typedef enum AF_Style_ { #include "afstyles.h" AF_STYLE_MAX /* do not remove */ } AF_Style;

## *struct AF_WidthRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Pos | org | |
| FT_Pos | cur | |
| FT_Pos | fit | |

**Definition:**
typedef struct AF_WidthRec_ { FT_Pos org; /* original position/width in font units */ FT_Pos cur; /* current/scaled position/width in device subpixels */ FT_Pos fit; /* current/fitted position/width in device subpixels */ } AF_WidthRec, *AF_Width;

## *enum AF_WritingSystem_*

**Definition:**
typedef enum AF_WritingSystem_ { #include "afws-iter.h" AF_WRITING_SYSTEM_MAX /* do not remove */ } AF_WritingSystem;

# ft-hb.h

## *Data Types*

### *typedef args*

**Definition:**
typedef ret (*ft_ ## name ## _func_t) args;

# bdf.h

## *Data Types*

### *struct bdf_bbx_t_*

**Members:**

| Type | Name | Description |
|---|---|---|
| unsigned short | width | |
| unsigned short | height | |
| short | x_offset | |
| short | y_offset | |
| short | ascent | |
| short | descent | |

**Definition:**
typedef struct bdf_bbx_t_ { unsigned short width; unsigned short height; short x_offset; short y_offset; short ascent; short descent; } bdf_bbx_t;

### *struct bdf_property_t_*

**Members:**

| Type | Name | Description |
|---|---|---|
| const char* | name | |
| int | format | |
| int | builtin | |
| char* | atom | |
| long | l | |
| unsigned long | ul | |

**Definition:**
typedef struct bdf_property_t_ { const char* name; /* Name of the property. */ int format; /* Format of the property. */ int builtin; /* A builtin property. */ union { char* atom; long l; unsigned long ul; } value; /* Value of the property. */ } bdf_property_t;

# ftccache.h

## *Data Types*

### *struct FTC_CacheRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FTC_MruNodeRec | mru | |
| FTC_Node | link | |
| FT_Offset | hash | |
| FT_UShort | cache_index | |
| FT_Short | ref_count | |

**Definition:**
typedef struct FTC_CacheRec_* FTC_Cache;

### *struct FTC_NodeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FTC_MruNodeRec | mru | |
| FTC_Node | link | |
| FT_Offset | hash | |
| FT_UShort | cache_index | |
| FT_Short | ref_count | |

**Definition:**
typedef struct FTC_NodeRec_ { FTC_MruNodeRec mru; /* circular mru list pointer */ FTC_Node link; /* used for hashing */ FT_Offset hash; /* used for hashing too */ FT_UShort cache_index; /* index of cache the node belongs to */ FT_Short ref_count; /* reference count for this node */ } FTC_NodeRec;

# ftcglyph.h

## *Data Types*

### *struct FTC_FamilyRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FTC_MruNodeRec | mrunode | |
| FT_UInt | num_nodes | |
| FTC_Cache | cache | |
| FTC_MruListClass | clazz | |

**Definition:**

typedef struct FTC_FamilyRec_ { FTC_MruNodeRec mrunode; FT_UInt num_nodes; /* current number of nodes in this family */ FTC_Cache cache; FTC_MruListClass clazz; } FTC_FamilyRec, *FTC_Family;

### *struct FTC_GCacheRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FTC_CacheRec | cache | |
| FTC_MruListRec | families | |

**Definition:**

typedef struct FTC_GCacheRec_ { FTC_CacheRec cache; FTC_MruListRec families; } FTC_GCacheRec, *FTC_GCache;

# ftcimage.h

## *Data Types*

### *struct FTC_INodeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FTC_GNodeRec | gnode | |
| FT_Glyph | glyph | |

**Definition:**
typedef struct FTC_INodeRec_ { FTC_GNodeRec gnode; FT_Glyph glyph; } FTC_INodeRec, *FTC_INode;

# ftcmanag.h

## *Data Types*

### *struct FTC_ManagerRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Library | library | |
| FT_Memory | memory | |
| FTC_Node | nodes_list | |
| FT_Offset | max_weight | |
| FT_Offset | cur_weight | |
| FT_UInt | num_nodes | |
| FTC_Cache | caches | |
| FT_UInt | num_caches | |
| FTC_MruListRec | faces | |
| FTC_MruListRec | sizes | |
| FT_Pointer | request_data | |
| FTC_Face_Requester | request_face | |

**Definition:**

typedef struct FTC_ManagerRec_ { FT_Library library; FT_Memory memory; FTC_Node nodes_list; FT_Offset max_weight; FT_Offset cur_weight; FT_UInt num_nodes; FTC_Cache caches[FTC_MAX_CACHES]; FT_UInt num_caches; FTC_MruListRec faces; FTC_MruListRec sizes; FT_Pointer request_data; FTC_Face_Requester request_face; } FTC_ManagerRec;

# ftcmru.h

## *Data Types*

### *struct FTC_MruNodeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FTC_MruNode | next | |
| FTC_MruNode | prev | |

**Definition:**
typedef struct FTC_MruNodeRec_* FTC_MruNode;

# ftcsbits.h

## *Data Types*

### *struct FTC_SNodeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FTC_GNodeRec | gnode | |
| FT_UInt | count | |
| FTC_SBitRec | sbits | |

**Definition:**

typedef struct FTC_SNodeRec_ { FTC_GNodeRec gnode; FT_UInt count; FTC_SBitRec sbits[FTC_SBIT_ITEMS_PER_NODE]; } FTC_SNodeRec, *FTC_SNode;

# cffcmap.h

## *Data Types*

### *struct CFF_CMapStdRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_CMapRec | cmap | |
| FT_UShort* | gids | |

**Definition:**

typedef struct CFF_CMapStdRec_* CFF_CMapStd;

# cffparse.h

## *Data Types*

### *struct CFF_ParserRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Library | library | |
| FT_Byte* | start | |
| FT_Byte* | limit | |
| FT_Byte* | cursor | |
| FT_Byte** | stack | |
| FT_Byte** | top | |
| FT_UInt | stackSize | |
| FT_ListRec | t2_strings | |
| FT_UInt | object_code | |
| void* | object | |
| FT_UShort | num_designs | |
| FT_UShort | num_axes | |

**Definition:**

typedef struct CFF_ParserRec_ { FT_Library library; FT_Byte* start; FT_Byte* limit; FT_Byte* cursor; FT_Byte** stack; FT_Byte** top; FT_UInt stackSize; /* allocated size */ #ifdef CFF_CONFIG_OPTION_OLD_ENGINE FT_ListRec t2_strings; #endif /* CFF_CONFIG_OPTION_OLD_ENGINE */ FT_UInt object_code; void* object; FT_UShort num_designs; /* a copy of `CFF_FontRecDict->num_designs' */ FT_UShort num_axes; /* a copy of `CFF_FontRecDict->num_axes' */ } CFF_ParserRec, *CFF_Parser;

# cidload.h

## *Data Types*

### *struct CID_Loader_*

**Members:**

| Type | Name | Description |
|---|---|---|
| CID_Parser | parser | |
| FT_Int | num_chars | |

**Definition:**

typedef struct CID_Loader_ { CID_Parser parser; /* parser used to read the stream */ FT_Int num_chars; /* number of characters in encoding */ } CID_Loader;

# cidobjs.h

## *Data Types*

### *struct CID_CharMapRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |
| FT_Bool | valid | |

**Definition:**
typedef struct CID_CharMapRec_* CID_CharMap;

### *struct CID_DriverRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |
| FT_Bool | valid | |

**Definition:**
typedef struct CID_DriverRec_* CID_Driver;

### *struct CID_GlyphSlotRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |
| FT_Bool | valid | |

**Definition:**
typedef struct CID_GlyphSlotRec_* CID_GlyphSlot;

### *struct CID_SizeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |
| FT_Bool | valid | |

**Definition:**
typedef struct CID_SizeRec_* CID_Size;

### *struct CID_SizeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |
| FT_Bool | valid | |

**Definition:**
typedef struct CID_SizeRec_ { FT_SizeRec root; FT_Bool valid; } CID_SizeRec;

## *struct CID_Size_Hints_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |
| FT_Bool | valid | |

**Definition:**
typedef struct CID_Size_Hints_ CID_Size_Hints;

# cidparse.h

## *Data Types*

### *typedef root*

**Definition:**
* A structure which holds the information about * the current font. * * num_dict :: * The number of font dictionaries. */ typedef struct CID_Parser_ { PS_ParserRec root; FT_Stream stream; FT_Byte* postscript; FT_ULong postscript_len; FT_ULong data_offset; FT_ULong binary_length; CID_FaceInfo cid; FT_UInt num_dict; } CID_Parser;

# gxvcommn.h

## *Data Types*

### *struct GXV_ValidatorRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | u | |
| FT_Short | s | |

**Definition:**
typedef struct GXV_ValidatorRec_* GXV_Validator;

### *struct GXV_odtect_DataRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Bytes | start | |
| FT_ULong | length | |
| FT_String* | name | |

**Definition:**
typedef struct GXV_odtect_DataRec_ { FT_Bytes start; FT_ULong length; FT_String* name; } GXV_odtect_DataRec, *GXV_odtect_Data;

# gxvfeat.h

## *Data Types*

### *struct GXV_Feature_RegistryRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Bool | existence | |
| FT_Bool | apple_reserved | |
| FT_Bool | exclusive | |
| FT_Byte | nSettings | |

**Definition:**

typedef struct GXV_Feature_RegistryRec_ { FT_Bool existence; FT_Bool apple_reserved; FT_Bool exclusive; FT_Byte nSettings; } GX_Feature_RegistryRec;

# gxvmort.h

## *Data Types*

### *struct GXV_mort_featureRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | featureType | |
| FT_UShort | featureSetting | |
| FT_ULong | enableFlags | |
| FT_ULong | disableFlags | |

**Definition:**

typedef struct GXV_mort_featureRec_ { FT_UShort featureType; FT_UShort featureSetting; FT_ULong enableFlags; FT_ULong disableFlags; } GXV_mort_featureRec, *GXV_mort_feature;

# ftzopen.h

## *Data Types*

### *enum FT_LzwPhase_*

**Definition:**
typedef enum FT_LzwPhase_ { FT_LZW_PHASE_START = 0, FT_LZW_PHASE_CODE, FT_LZW_PHASE_STACK, FT_LZW_PHASE_EOF } FT_LzwPhase;

# otvcommn.h

## *Data Types*

### *struct OTV_ValidatorRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Validator | root | |
| FT_UInt | type_count | |
| OTV_Validate_Func* | type_funcs | |
| FT_UInt | lookup_count | |
| FT_UInt | glyph_count | |
| FT_UInt | nesting_level | |
| OTV_Validate_Func | func | |
| FT_UInt | extra1 | |
| FT_UInt | extra2 | |
| FT_Bytes | extra3 | |
| FT_UInt | debug_indent | |
| const FT_String* | debug_function_name | |

**Definition:**

typedef struct OTV_ValidatorRec_* OTV_Validator;

# pfrcmap.h

## *Data Types*

### *struct PFR_CMapRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_CMapRec | cmap | |
| FT_UInt | num_chars | |
| PFR_Char | chars | |

**Definition:**
typedef struct PFR_CMapRec_ { FT_CMapRec cmap; FT_UInt num_chars; PFR_Char chars; } PFR_CMapRec, *PFR_CMap;

# pfrobjs.h

## *Data Types*

### *struct PFR_FaceRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_FaceRec | root | |
| PFR_HeaderRec | header | |
| PFR_LogFontRec | log_font | |
| PFR_PhyFontRec | phy_font | |

**Definition:**

typedef struct PFR_FaceRec_* PFR_Face;

# pfrtypes.h

## *Data Types*

### *struct PFR_BitmapCharRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UInt | char_code | |
| FT_UInt | gps_size | |
| FT_UInt32 | gps_offset | |

**Definition:**
typedef struct PFR_BitmapCharRec_ { FT_UInt char_code; FT_UInt gps_size; FT_UInt32 gps_offset; } PFR_BitmapCharRec, *PFR_BitmapChar;

### *struct PFR_CharRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UInt | char_code | |
| FT_Int | advance | |
| FT_UInt | gps_size | |
| FT_UInt32 | gps_offset | |

**Definition:**
typedef struct PFR_CharRec_ { FT_UInt char_code; FT_Int advance; FT_UInt gps_size; FT_UInt32 gps_offset; } PFR_CharRec, *PFR_Char;

### *struct PFR_CoordRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UInt | org | |
| FT_UInt | cur | |

**Definition:**
typedef struct PFR_CoordRec_ { FT_UInt org; FT_UInt cur; } PFR_CoordRec, *PFR_Coord;

### *struct PFR_DimensionRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UInt | standard | |
| FT_UInt | num_stem_snaps | |

| | | |
|---|---|---|
| FT_Int* | stem_snaps | |

**Definition:**
typedef struct PFR_DimensionRec_ { FT_UInt standard; FT_UInt num_stem_snaps; FT_Int* stem_snaps; } PFR_DimensionRec, *PFR_Dimension;

## *struct PFR_HeaderRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UInt32 | signature | |
| FT_UInt | version | |
| FT_UInt | signature2 | |
| FT_UInt | header_size | |
| FT_UInt | log_dir_size | |
| FT_UInt | log_dir_offset | |
| FT_UInt | log_font_max_size | |
| FT_UInt32 | log_font_section_size | |
| FT_UInt32 | log_font_section_offset | |
| FT_UInt32 | phy_font_max_size | |
| FT_UInt32 | phy_font_section_size | |
| FT_UInt32 | phy_font_section_offset | |
| FT_UInt | gps_max_size | |
| FT_UInt32 | gps_section_size | |
| FT_UInt32 | gps_section_offset | |
| FT_UInt | max_blue_values | |
| FT_UInt | max_x_orus | |
| FT_UInt | max_y_orus | |
| FT_UInt | phy_font_max_size_high | |
| FT_UInt | color_flags | |
| FT_UInt32 | bct_max_size | |
| FT_UInt32 | bct_set_max_size | |
| FT_UInt32 | phy_bct_set_max_size | |
| FT_UInt | num_phy_fonts | |
| FT_UInt | max_vert_stem_snap | |
| FT_UInt | max_horz_stem_snap | |
| FT_UInt | max_chars | |

**Definition:**

typedef struct PFR_HeaderRec_ { FT_UInt32 signature; FT_UInt version; FT_UInt signature2; FT_UInt header_size; FT_UInt log_dir_size; FT_UInt log_dir_offset; FT_UInt log_font_max_size; FT_UInt32 log_font_section_size; FT_UInt32 log_font_section_offset; FT_UInt32 phy_font_max_size; FT_UInt32 phy_font_section_size; FT_UInt32 phy_font_section_offset; FT_UInt gps_max_size; FT_UInt32 gps_section_size; FT_UInt32 gps_section_offset; FT_UInt max_blue_values; FT_UInt max_x_orus; FT_UInt max_y_orus; FT_UInt phy_font_max_size_high; FT_UInt color_flags; FT_UInt32 bct_max_size; FT_UInt32 bct_set_max_size; FT_UInt32 phy_bct_set_max_size; FT_UInt num_phy_fonts; FT_UInt max_vert_stem_snap; FT_UInt max_horz_stem_snap; FT_UInt max_chars; } PFR_HeaderRec, *PFR_Header;

## *struct PFR_KernItemRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| PFR_KernItem | next | |
| FT_Byte | pair_count | |
| FT_Byte | flags | |
| FT_Short | base_adj | |
| FT_UInt | pair_size | |
| FT_Offset | offset | |
| FT_UInt32 | pair1 | |
| FT_UInt32 | pair2 | |

**Definition:**

typedef struct PFR_KernItemRec_* PFR_KernItem;

## *struct PFR_LogFontRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UInt32 | size | |
| FT_UInt32 | offset | |
| FT_Int32 | matrix | |
| FT_UInt | stroke_flags | |
| FT_Int | stroke_thickness | |
| FT_Int | bold_thickness | |
| FT_Int32 | miter_limit | |
| FT_UInt32 | phys_size | |
| FT_UInt32 | phys_offset | |

**Definition:**

typedef struct PFR_LogFontRec_ { FT_UInt32 size; FT_UInt32 offset; FT_Int32 matrix[4]; FT_UInt stroke_flags; FT_Int stroke_thickness; FT_Int bold_thickness; FT_Int32 miter_limit; FT_UInt32 phys_size; FT_UInt32 phys_offset; } PFR_LogFontRec, *PFR_LogFont;

## *struct PFR_PhyFontRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Memory | memory | |
| FT_UInt32 | offset | |
| FT_UInt | font_ref_number | |
| FT_UInt | outline_resolution | |
| FT_UInt | metrics_resolution | |
| FT_BBox | bbox | |
| FT_UInt | flags | |
| FT_Int | standard_advance | |
| FT_Int | ascent | |
| FT_Int | descent | |
| FT_Int | leading | |
| PFR_DimensionRec | horizontal | |
| PFR_DimensionRec | vertical | |
| FT_String* | font_id | |
| FT_String* | family_name | |
| FT_String* | style_name | |
| FT_UInt | num_strikes | |
| FT_UInt | max_strikes | |
| PFR_StrikeRec* | strikes | |
| FT_UInt | num_blue_values | |
| FT_Int | *blue_values | |
| FT_UInt | blue_fuzz | |
| FT_UInt | blue_scale | |
| FT_UInt | num_chars | |
| FT_Offset | chars_offset | |
| PFR_Char | chars | |
| FT_UInt | num_kern_pairs | |
| PFR_KernItem | kern_items | |
| PFR_KernItem* | kern_items_tail | |
| FT_ULong | bct_offset | |
| FT_Byte* | cursor | |

**Definition:**
typedef struct PFR_PhyFontRec_ { FT_Memory memory; FT_UInt32 offset; FT_UInt font_ref_number; FT_UInt outline_resolution; FT_UInt metrics_resolution; FT_BBox bbox; FT_UInt flags; FT_Int standard_advance; FT_Int ascent; /* optional, bbox.yMax if not present */ FT_Int descent; /* optional, bbox.yMin if not present */ FT_Int leading; /* optional, 0 if not present */ PFR_DimensionRec horizontal; PFR_DimensionRec vertical; FT_String* font_id; FT_String* family_name; FT_String* style_name; FT_UInt num_strikes; FT_UInt max_strikes; PFR_StrikeRec* strikes; FT_UInt num_blue_values; FT_Int *blue_values; FT_UInt blue_fuzz; FT_UInt blue_scale; FT_UInt num_chars; FT_Offset chars_offset; PFR_Char chars; FT_UInt num_kern_pairs; PFR_KernItem kern_items; PFR_KernItem* kern_items_tail; /* not part of the spec, but used during load */ FT_ULong bct_offset; FT_Byte* cursor; } PFR_PhyFontRec, *PFR_PhyFont;

# afmparse.h

## *Data Types*

### *enum AFM_ValueType_*

**Definition:**
enum AFM_ValueType_ { AFM_VALUE_TYPE_STRING, AFM_VALUE_TYPE_NAME, AFM_VALUE_TYPE_FIXED, /* real number */ AFM_VALUE_TYPE_INTEGER, AFM_VALUE_TYPE_BOOL, AFM_VALUE_TYPE_INDEX /* glyph index */ };

# psarrst.h

## *Data Types*

### *struct CF2_ArrStackRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Memory | memory | |
| FT_Error* | error | |
| size_t | sizeItem | |
| size_t | allocated | |
| size_t | count | |
| size_t | totalSize | |
| void* | ptr | |

**Definition:**

typedef struct CF2_ArrStackRec_ { FT_Memory memory; FT_Error* error; size_t sizeItem; /* bytes per element */ size_t allocated; /* items allocated */ size_t count; /* number of elements allocated */ size_t totalSize; /* total bytes allocated */ void* ptr; /* ptr to data */ } CF2_ArrStackRec, *CF2_ArrStack;

# psfixed.h

## *Data Types*

### *typedef CF2_Frac*

**Definition:**
typedef FT_Int32 CF2_Frac; /* 2.30 fixed-point */

# psfont.h

## *Data Types*

### *struct CF2_FontRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Memory | memory | |
| FT_Error | error | |
| FT_Bool | isT1 | |
| FT_Bool | isCFF2 | |
| CF2_RenderingFlags | renderingFlags | |
| CF2_Matrix | currentTransform | |
| CF2_Matrix | innerTransform | |
| CF2_Matrix | outerTransform | |
| CF2_Fixed | ppem | |
| CFF_BlendRec | blend | |
| CF2_UInt | vsindex | |
| CF2_UInt | lenNDV | |
| FT_Fixed* | NDV | |
| CF2_Int | unitsPerEm | |
| CF2_Fixed | syntheticEmboldeningAmountX | |
| CF2_Fixed | syntheticEmboldeningAmountY | |
| CF2_OutlineRec | outline | |
| PS_Decoder* | decoder | |
| CFF_SubFont | lastSubfont | |
| FT_Bool | hinted | |
| FT_Bool | darkened | |
| FT_Bool | stemDarkened | |
| FT_Int | darkenParams | |
| CF2_Fixed | stdVW | |
| CF2_Fixed | stdHW | |
| CF2_Fixed | darkenX | |
| CF2_Fixed | darkenY | |
| FT_Bool | reverseWinding | |

| | | |
|---|---|---|
| CF2_BluesRec | blues | |
| FT_Service_CFFLoad | cffload | |

**Definition:**

struct CF2_FontRec_ { FT_Memory memory; FT_Error error; /* shared error for this instance */ FT_Bool isT1; FT_Bool isCFF2; CF2_RenderingFlags renderingFlags; /* variables that depend on Transform: */ /* the following have zero translation; */ /* inner * outer = font * original */ CF2_Matrix currentTransform; /* original client matrix */ CF2_Matrix innerTransform; /* for hinting; erect, scaled */ CF2_Matrix outerTransform; /* post hinting; includes rotations */ CF2_Fixed ppem; /* transform-dependent */ /* variation data */ CFF_BlendRec blend; /* cached charstring blend vector */ CF2_UInt vsindex; /* current vsindex */ CF2_UInt lenNDV; /* current length NDV or zero */ FT_Fixed* NDV; /* ptr to current NDV or NULL */ CF2_Int unitsPerEm; CF2_Fixed syntheticEmboldeningAmountX; /* character space units */ CF2_Fixed syntheticEmboldeningAmountY; /* character space units */ /* FreeType related members */ CF2_OutlineRec outline; /* freetype glyph outline functions */ PS_Decoder* decoder; CFF_SubFont lastSubfont; /* FreeType parsed data; */ /* top font or subfont */ /* these flags can vary from one call to the next */ FT_Bool hinted; FT_Bool darkened; /* true if stemDarkened or synthetic bold */ /* i.e. darkenX != 0 || darkenY != 0 */ FT_Bool stemDarkened; FT_Int darkenParams[8]; /* 1000 unit character space */ /* variables that depend on both FontDict and Transform */ CF2_Fixed stdVW; /* in character space; depends on dict entry */ CF2_Fixed stdHW; /* in character space; depends on dict entry */ CF2_Fixed darkenX; /* character space units */ CF2_Fixed darkenY; /* depends on transform */ /* and private dict (StdVW) */ FT_Bool reverseWinding; /* darken assuming */ /* counterclockwise winding */ CF2_BluesRec blues; /* computed zone data */ FT_Service_CFFLoad cffload; /* pointer to cff functions */ };

# psglue.h

## *Data Types*

### *typedef CF2_RenderingFlags*

**Definition:**
typedef CF2_Int CF2_RenderingFlags;

# psread.h

## *Data Types*

### *struct CF2_BufferRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Error* | error | |
| const FT_Byte* | start | |
| const FT_Byte* | end | |
| const FT_Byte* | ptr | |

**Definition:**

typedef struct CF2_BufferRec_ { FT_Error* error; const FT_Byte* start; const FT_Byte* end; const FT_Byte* ptr; } CF2_BufferRec, *CF2_Buffer;

# psstack.h

## *Data Types*

### *struct CF2_StackNumber_*

**Members:**

| Type | Name | Description |
|---|---|---|
| CF2_Fixed | r | |
| CF2_Frac | f | |
| CF2_Int | i | |
| CF2_NumberType | type | |

**Definition:**

typedef struct CF2_StackNumber_ { union { CF2_Fixed r; /* 16.16 fixed-point */ CF2_Frac f; /* 2.30 fixed-point (for font matrix) */ CF2_Int i; } u; CF2_NumberType type; } CF2_StackNumber;

# pstypes.h

## *Data Types*

### *typedef CF2_F16Dot16*

**Definition:**
typedef FT_Int32 CF2_F16Dot16;

# t1cmap.h

## *Data Types*

### *struct T1_CMapCustomRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_CMapRec | cmap | |
| FT_UInt | first | |
| FT_UInt | count | |
| FT_UShort* | indices | |

**Definition:**
typedef struct T1_CMapCustomRec_* T1_CMapCustom;

### *struct T1_CMapStdRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_CMapRec | cmap | |
| const FT_UShort* | code_to_sid | |
| PS_Adobe_Std_StringsFunc | sid_to_string | |
| FT_UInt | num_glyphs | |
| const char* const* | glyph_names | |

**Definition:**
typedef struct T1_CMapStdRec_* T1_CMapStd;

# pshalgo.h

## *Data Types*

### *struct PSH_HintRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Int | org_pos | |
| FT_Int | org_len | |
| FT_Pos | cur_pos | |
| FT_Pos | cur_len | |
| FT_UInt | flags | |
| PSH_Hint | parent | |
| FT_Int | order | |

**Definition:**

typedef struct PSH_HintRec_* PSH_Hint;

# pshglob.h

## *Data Types*

### *struct PSH_WidthRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Int | org | |
| FT_Pos | cur | |
| FT_Pos | fit | |

**Definition:**
typedef struct PSH_WidthRec_ { FT_Int org; FT_Pos cur; FT_Pos fit; } PSH_WidthRec, *PSH_Width;

### *typedef org*

**Definition:**
* The maximum number of blue zones in a font global hints structure. * See @PS_Globals_BluesRec. */ #define PS_GLOBALS_MAX_BLUE_ZONES 16 /********************************************************************** * * @constant: * PS_GLOBALS_MAX_STD_WIDTHS * * @description: * The maximum number of standard and snap widths in either the * horizontal or vertical direction. See @PS_Globals_WidthsRec. */ #define PS_GLOBALS_MAX_STD_WIDTHS 16 /* standard and snap width */ typedef struct PSH_WidthRec_ { FT_Int org; FT_Pos cur; FT_Pos fit; } PSH_WidthRec, *PSH_Width;

# pshrec.h

## *Data Types*

### *struct PS_HintRec_*

**Definition:**
typedef struct PS_HintRec_* PS_Hint;

# ftmisc.h

## *Data Types*

### *typedef FT_Byte*

**Definition:**
typedef unsigned char FT_Byte;

# ftsdf.h

## *Data Types*

### *struct SDF_Raster_Params_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Raster_Params | root | |
| FT_UInt | spread | |
| FT_Bool | flip_sign | |
| FT_Bool | flip_y | |
| FT_Bool | overlaps | |

**Definition:**

typedef struct SDF_Raster_Params_ { FT_Raster_Params root; FT_UInt spread; FT_Bool flip_sign; FT_Bool flip_y; FT_Bool overlaps; } SDF_Raster_Params;

# ftsdfcommon.h

## *Data Types*

### *typedef FT_26D6_Vec*

**Definition:**
* common typedefs * */ typedef FT_Vector FT_26D6_Vec; /* with 26.6 fixed-point components */

# ftsdfrend.h

## *Data Types*

### *struct SDF_Renderer_Module_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_RendererRec | root | |
| FT_UInt | spread | |
| FT_Bool | flip_sign | |
| FT_Bool | flip_y | |
| FT_Bool | overlaps | |

**Definition:**

typedef struct SDF_Renderer_Module_ { FT_RendererRec root; FT_UInt spread; FT_Bool flip_sign; FT_Bool flip_y; FT_Bool overlaps; } SDF_Renderer_Module, *SDF_Renderer;

# ttcmap.h

## *Data Types*

### *struct TT_CMapRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_CMapRec | cmap | |
| FT_Byte* | data | |
| FT_Int | flags | |

**Definition:**

typedef struct TT_CMapRec_ { FT_CMapRec cmap; FT_Byte* data; /* pointer to in-memory cmap table */ FT_Int flags; /* for format 4 only */ } TT_CMapRec, *TT_CMap;

# svgtypes.h

## *Data Types*

### *struct SVG_RendererRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_RendererRec | root | |
| FT_Bool | loaded | |
| FT_Bool | hooks_set | |
| SVG_RendererHooks | hooks | |
| FT_Pointer | state | |

**Definition:**

typedef struct SVG_RendererRec_ { FT_RendererRec root; /* this inherits FT_RendererRec */ FT_Bool loaded; FT_Bool hooks_set; SVG_RendererHooks hooks; /* this holds hooks for SVG rendering */ FT_Pointer state; /* a place for hooks to store state, if needed */ } SVG_RendererRec;

# ttgxvar.h

## *Data Types*

### *struct GX_AVarSegmentRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | pairCount | |
| GX_AVarCorrespondence | correspondence | |

**Definition:**
typedef struct GX_AVarSegmentRec_ { FT_UShort pairCount; GX_AVarCorrespondence correspondence; /* array with pairCount entries */ } GX_AVarSegmentRec, *GX_AVarSegment;

### *struct GX_AVarTableRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| GX_AVarSegment | avar_segment | |
| GX_ItemVarStoreRec | itemStore | |
| GX_DeltaSetIdxMapRec | axisMap | |

**Definition:**
typedef struct GX_AVarTableRec_ { GX_AVarSegment avar_segment; /* avar_segment[num_axis] */ GX_ItemVarStoreRec itemStore; /* Item Variation Store */ GX_DeltaSetIdxMapRec axisMap; /* Axis Mapping */ } GX_AVarTableRec, *GX_AVarTable;

### *struct GX_BlendRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UInt | num_axis | |
| FT_Fixed* | coords | |
| FT_Fixed* | normalizedcoords | |
| FT_MM_Var* | mmvar | |
| FT_Offset | mmvar_len | |
| FT_Fixed* | normalized_stylecoords | |
| FT_Bool | avar_loaded | |
| GX_AVarTable | avar_table | |
| FT_Bool | hvar_loaded | |
| FT_Bool | hvar_checked | |

| | | |
|---|---|---|
| FT_Error | hvar_error | |
| GX_HVVarTable | hvar_table | |
| FT_Bool | vvar_loaded | |
| FT_Bool | vvar_checked | |
| FT_Error | vvar_error | |
| GX_HVVarTable | vvar_table | |
| GX_MVarTable | mvar_table | |
| FT_UInt | tuplecount | |
| FT_Fixed* | tuplecoords | |
| FT_Fixed* | tuplescalars | |
| FT_UInt | gv_glyphcnt | |
| FT_ULong* | glyphoffsets | |
| FT_ULong | gvar_size | |

**Definition:**

typedef struct GX_BlendRec_ { FT_UInt num_axis; FT_Fixed* coords; FT_Fixed* normalizedcoords; FT_MM_Var* mmvar; FT_Offset mmvar_len; FT_Fixed* normalized_stylecoords; /* normalized_stylecoords[num_namedstyles][num_axis] */ FT_Bool avar_loaded; GX_AVarTable avar_table; FT_Bool hvar_loaded; FT_Bool hvar_checked; FT_Error hvar_error; GX_HVVarTable hvar_table; FT_Bool vvar_loaded; FT_Bool vvar_checked; FT_Error vvar_error; GX_HVVarTable vvar_table; GX_MVarTable mvar_table; FT_UInt tuplecount; FT_Fixed* tuplecoords; /* tuplecoords[tuplecount][num_axis] */ FT_Fixed* tuplescalars; /* tuplescalars[tuplecount] */ FT_UInt gv_glyphcnt; FT_ULong* glyphoffsets; /* glyphoffsets[gv_glyphcnt + 1] */ FT_ULong gvar_size; } GX_BlendRec;

## *struct GX_HVVarTableRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| GX_ItemVarStoreRec | itemStore | |
| GX_DeltaSetIdxMapRec | widthMap | |
| GX_DeltaSetIdxMapRec | lsbMap | |
| GX_DeltaSetIdxMapRec | rsbMap | |
| GX_DeltaSetIdxMapRec | tsbMap | |
| GX_DeltaSetIdxMapRec | bsbMap | |
| GX_DeltaSetIdxMapRec | vorgMap | |

**Definition:**

typedef struct GX_HVVarTableRec_ { GX_ItemVarStoreRec itemStore; /* Item Variation Store */ GX_DeltaSetIdxMapRec widthMap; /* Advance Width Mapping */ #if 0 GX_DeltaSetIdxMapRec lsbMap; /* not implemented */ GX_DeltaSetIdxMapRec rsbMap; /* not implemented */ GX_DeltaSetIdxMapRec tsbMap; /* not implemented */ GX_DeltaSetIdxMapRec bsbMap; /* not implemented */ GX_DeltaSetIdxMapRec vorgMap; /* not implemented */ #endif } GX_HVVarTableRec,

*GX_HVVarTable;

### *struct GX_MVarTableRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | valueCount | |
| GX_ItemVarStoreRec | itemStore | |
| GX_Value | values | |

**Definition:**
typedef struct GX_MVarTableRec_ { FT_UShort valueCount; GX_ItemVarStoreRec itemStore; /* Item Variation Store */ GX_Value values; /* Value Records */ } GX_MVarTableRec, *GX_MVarTable;

### *enum GX_TupleCountFlags_*

**Definition:**
typedef enum GX_TupleCountFlags_ { GX_TC_TUPLES_SHARE_POINT_NUMBERS = 0x8000, GX_TC_RESERVED_TUPLE_FLAGS = 0x7000, GX_TC_TUPLE_COUNT_MASK = 0x0FFF } GX_TupleCountFlags;

### *enum GX_TupleIndexFlags_*

**Definition:**
typedef enum GX_TupleIndexFlags_ { GX_TI_EMBEDDED_TUPLE_COORD = 0x8000, GX_TI_INTERMEDIATE_TUPLE = 0x4000, GX_TI_PRIVATE_POINT_NUMBERS = 0x2000, GX_TI_RESERVED_TUPLE_FLAG = 0x1000, GX_TI_TUPLE_INDEX_MASK = 0x0FFF } GX_TupleIndexFlags;

### *typedef fromCoord*

**Definition:**
* A data structure representing `shortFracCorrespondence' in `avar' * table according to the specifications from Apple. */ typedef struct GX_AVarCorrespondenceRec_ { FT_Fixed fromCoord; FT_Fixed toCoord; } GX_AVarCorrespondenceRec_, *GX_AVarCorrespondence;

# ttinterp.h

## *Data Types*

### *struct TT_CallRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Int | Caller_Range | |
| FT_Long | Caller_IP | |
| FT_Long | Cur_Count | |
| TT_DefRecord | *Def | |

**Definition:**
typedef struct TT_CallRec_ { FT_Int Caller_Range; FT_Long Caller_IP; FT_Long Cur_Count; TT_DefRecord *Def; /* either FDEF or IDEF */ } TT_CallRec, *TT_CallStack;

### *typedef TT_CodeRange_Tag*

**Definition:**
* These sub-tables relate to instruction execution. * */ #define TT_MAX_CODE_RANGES 3 /********************************************************************** * * There can only be 3 active code ranges at once: * - the Font Program * - the CVT Program * - a glyph's instructions set */ typedef enum TT_CodeRange_Tag_ { tt_coderange_none = 0, tt_coderange_font, tt_coderange_cvt, tt_coderange_glyph } TT_CodeRange_Tag;

### *enum TT_CodeRange_Tag_*

**Definition:**
typedef enum TT_CodeRange_Tag_ { tt_coderange_none = 0, tt_coderange_font, tt_coderange_cvt, tt_coderange_glyph } TT_CodeRange_Tag;

### *struct TT_DefRecord_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Int | range | |
| FT_Long | start | |
| FT_Long | end | |
| FT_UInt | opc | |
| FT_Bool | active | |

**Definition:**
typedef struct TT_DefRecord_ { FT_Int range; /* in which code range is it located? */ FT_Long start; /* where does it start? */ FT_Long end; /* where does it end? */ FT_UInt opc; /* function #, or instruction code */ FT_Bool active; /* is it active? */ } TT_DefRecord, *TT_DefArray;

## *struct TT_ExecContextRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| TT_Face | face | |
| TT_Size | size | |
| FT_Memory | memory | |
| TT_Interpreter | interpreter | |
| FT_Error | error | |
| FT_Long | top | |
| FT_Long | stackSize | |
| FT_Long* | stack | |
| FT_Long | args | |
| FT_Long | new_top | |
| FT_Long | pointSize | |
| FT_Size_Metrics | metrics | |
| TT_Size_Metrics | tt_metrics | |
| TT_GraphicsState | GS | |
| FT_Int | iniRange | |
| FT_Int | curRange | |
| FT_Byte* | code | |
| FT_Long | IP | |
| FT_Long | codeSize | |
| FT_Byte | opcode | |
| FT_Int | length | |
| FT_ULong | cvtSize | |
| FT_Long* | cvt | |
| FT_ULong | glyfCvtSize | |
| FT_Long* | glyfCvt | |
| FT_UInt | glyphSize | |
| FT_Byte* | glyphIns | |
| FT_UInt | numFDefs | |
| FT_UInt | maxFDefs | |
| TT_DefArray | FDefs | |
| FT_UInt | numIDefs | |

| | | |
|---|---|---|
| FT_UInt | maxIDefs | |
| TT_DefArray | IDefs | |
| FT_UInt | maxFunc | |
| FT_UInt | maxIns | |
| TT_CallStack | callStack | |
| FT_UShort | maxPoints | |
| FT_Short | maxContours | |
| TT_CodeRangeTable | codeRangeTable | |
| FT_UShort | storeSize | |
| FT_Long* | storage | |
| FT_UShort | glyfStoreSize | |
| FT_Long* | glyfStorage | |
| FT_F26Dot6 | period | |
| FT_F26Dot6 | phase | |
| FT_F26Dot6 | threshold | |
| FT_Bool | instruction_trap | |
| FT_Bool | is_composite | |
| FT_Bool | pedantic_hinting | |
| TT_Round_Func | func_round | |
| FT_Vector | moveVector | |
| TT_Move_Func | func_move | |
| TT_Move_Func | func_move_orig | |
| TT_Cur_Ppem_Func | func_cur_ppem | |
| TT_Get_CVT_Func | func_read_cvt | |
| TT_Set_CVT_Func | func_write_cvt | |
| TT_Set_CVT_Func | func_move_cvt | |
| FT_Bool | grayscale | |
| FT_Int | backward_compatibility | |
| FT_Render_Mode | mode | |
| FT_ULong | loopcall_counter | |
| FT_ULong | loopcall_counter_max | |
| FT_ULong | neg_jump_counter | |
| FT_ULong | neg_jump_counter_max | |

**Definition:**

typedef struct TT_ExecContextRec_ { TT_Face face; /* ! */ TT_Size size; /* ! */ FT_Memory memory; TT_Interpreter interpreter; /* instructions state */ FT_Error error; /* last execution error */ FT_Long top;

/* @! top of exec. stack */ FT_Long stackSize; /* ! size of exec. stack */ FT_Long* stack; /* ! current exec. stack */ FT_Long args; FT_Long new_top; /* new top after exec. */ TT_GlyphZoneRec zp0, /* @! zone records */ zp1, /* @! */ zp2, /* @! */ pts, /* ! */ twilight; /* ! */ FT_Long pointSize; /* ! in 26.6 format */ FT_Size_Metrics metrics; /* ! */ TT_Size_Metrics tt_metrics; /* ! size metrics */ TT_GraphicsState GS; /* !@ current graphics state */ FT_Int iniRange; /* initial code range number */ FT_Int curRange; /* current code range number */ FT_Byte* code; /* current code range */ FT_Long IP; /* current instruction pointer */ FT_Long codeSize; /* size of current range */ FT_Byte opcode; /* current opcode */ FT_Int length; /* opcode length or increment */ FT_ULong cvtSize; /* ! */ FT_Long* cvt; /* ! */ FT_ULong glyfCvtSize; FT_Long* glyfCvt; /* cvt working copy for glyph */ FT_UInt glyphSize; /* ! glyph instructions buffer size */ FT_Byte* glyphIns; /* ! glyph instructions buffer */ FT_UInt numFDefs; /* ! number of function defs */ FT_UInt maxFDefs; /* ! maximum number of function defs */ TT_DefArray FDefs; /* table of FDefs entries */ FT_UInt numIDefs; /* ! number of instruction defs */ FT_UInt maxIDefs; /* ! maximum number of ins defs */ TT_DefArray IDefs; /* table of IDefs entries */ FT_UInt maxFunc; /* ! maximum function index */ FT_UInt maxIns; /* ! maximum instruction index */ FT_Int callTop, /* @! top of call stack during execution */ callSize; /* size of call stack */ TT_CallStack callStack; /* call stack */ FT_UShort maxPoints; /* capacity of this context's `pts' */ FT_Short maxContours; /* record, expressed in points and */ /* contours. */ TT_CodeRangeTable codeRangeTable; /* ! table of valid code ranges */ /* useful for the debugger */ FT_UShort storeSize; /* ! size of current storage */ FT_Long* storage; /* ! storage area */ FT_UShort glyfStoreSize; FT_Long* glyfStorage; /* storage working copy for glyph */ FT_F26Dot6 period; /* values used for the */ FT_F26Dot6 phase; /* `SuperRounding' */ FT_F26Dot6 threshold; FT_Bool instruction_trap; /* ! If `True', the interpreter */ /* exits after each instruction */ FT_Bool is_composite; /* true if the glyph is composite */ FT_Bool pedantic_hinting; /* true if pedantic interpretation */ /* latest interpreter additions */ TT_Round_Func func_round; /* current rounding function */ FT_Vector moveVector; /* "projected" freedom vector */ TT_Project_Func func_project, /* current projection function */ func_dualproj, /* current dual proj. function */ func_freeProj; /* current freedom proj. func */ TT_Move_Func func_move; /* current point move function */ TT_Move_Func func_move_orig; /* move original position function */ TT_Cur_Ppem_Func func_cur_ppem; /* get current proj. ppem value */ TT_Get_CVT_Func func_read_cvt; /* read a cvt entry */ TT_Set_CVT_Func func_write_cvt; /* write a cvt entry (in pixels) */ TT_Set_CVT_Func func_move_cvt; /* incr a cvt entry (in pixels) */ FT_Bool grayscale; /* bi-level hinting and */ /* grayscale rendering */ #ifdef TT_SUPPORT_SUBPIXEL_HINTING_MINIMAL /* * FreeType supports ClearType-like hinting of TrueType fonts through * the version 40 interpreter. This is achieved through several hacks * in the base (v35) interpreter, as detailed below. * * ClearType is an umbrella term for several rendering techniques * employed by Microsoft's various GUI and rendering toolkit * implementations, most importantly: subpixel rendering for using the * RGB subpixels of LCDs to approximately triple the perceived * resolution on the x-axis and subpixel hinting for positioning stems * on subpixel borders. TrueType programming is explicit, i.e., fonts * must be programmed to take advantage of ClearType's possibilities. * * When ClearType was introduced, it seemed unlikely that all fonts * would be reprogrammed, so Microsoft decided to implement a backward * compatibility mode. It employs several simple to complicated * assumptions and tricks, many of them font-dependent, that modify the * interpretation of the bytecode contained in these fonts to retrofit * them into a ClearType-y look. The quality of the results varies. * Most (web)fonts that were released since then have come to rely on * these hacks to render correctly, even some of Microsoft's flagship * fonts (e.g., Calibri, Cambria, Segoe UI). * * FreeType's minimal subpixel hinting code (interpreter version 40) * employs a small list of font-agnostic hacks loosely based on the * public information available on Microsoft's compatibility mode[2]. * The focus is on modern (web)fonts rather than legacy fonts that were * made for monochrome rendering. It will not match ClearType rendering * exactly. Unlike the `Infinality' code (interpreter version 38) that * came before, it will not try to toggle hacks for specific fonts for * performance and complexity reasons. It will fall back to version 35 * behavior for tricky fonts[1] or when monochrome rendering is * requested. * * Major hacks * * - Any point movement on the x axis is ignored (cf. `Direct_Move' and * `Direct_Move_X'). This has the smallest code footprint and single * biggest effect. The ClearType way to increase resolution is * supersampling the x axis, the FreeType way is ignoring instructions * on the x axis, which gives the same result in the majority of * cases. * * - Points

are not moved post-IUP (neither on the x nor on the y axis), * except the x component of diagonal moves post-IUP (cf. * `Direct_Move', `Direct_Move_Y', `Move_Zp2_Point'). Post-IUP * changes are commonly used to `fix' pixel patterns which has little * use outside monochrome rendering. * * - SHPIX and DELTAP don't execute unless moving a composite on the * y axis or moving a previously y touched point. SHPIX additionally * denies movement on the x axis (cf. `Ins_SHPIX' and `Ins_DELTAP'). * Both instructions are commonly used to `fix' pixel patterns for * monochrome or Windows's GDI rendering but make little sense for * FreeType rendering. Both can distort the outline. See [2] for * details. * * - The hdmx table and modifications to phantom points are ignored. * Bearings and advance widths remain unchanged (except rounding them * outside the interpreter!), cf. `compute_glyph_metrics' and * `TT_Hint_Glyph'. Letting non-native-ClearType fonts modify spacing * might mess up spacing. * * Minor hacks * * - FLIPRGON, FLIPRGOFF, and FLIPPT don't execute post-IUP. This * prevents dents in e.g. Arial-Regular's `D' and `G' glyphs at * various sizes. * * (Post-IUP is the state after both IUP[x] and IUP[y] have been * executed.) * * The best results are achieved for fonts that were from the outset * designed with ClearType in mind, meaning they leave the x axis mostly * alone and don't mess with the `final' outline to produce more * pleasing pixel patterns. The harder the designer tried to produce * very specific patterns (`superhinting') for pre-ClearType-displays, * the worse the results. * * Microsoft defines a way to turn off backward compatibility and * interpret instructions as before (called `native ClearType')[2][3]. * The font designer then regains full control and is responsible for * making the font work correctly with ClearType without any * hand-holding by the interpreter or rasterizer[4]. The v40 * interpreter assumes backward compatibility by default, which can be * turned off the same way by executing the following in the control * program (cf. `Ins_INSTCTRL'). * * #PUSH 4,3 * INSTCTRL[] * * [1] Tricky fonts as FreeType defines them rely on the bytecode * interpreter to display correctly. Hacks can interfere with them,

# ttobjs.h

## *Data Types*

### *struct TT_DriverRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | rp0 | |
| FT_UShort | rp1 | |
| FT_UShort | rp2 | |
| FT_UShort | gep0 | |
| FT_UShort | gep1 | |
| FT_UShort | gep2 | |
| FT_UnitVector | dualVector | |
| FT_UnitVector | projVector | |
| FT_UnitVector | freeVector | |
| FT_Long | loop | |
| FT_Int | round_state | |
| FT_F26Dot6 | compensation | |
| FT_F26Dot6 | minimum_distance | |
| FT_F26Dot6 | control_value_cutin | |
| FT_F26Dot6 | single_width_cutin | |
| FT_F26Dot6 | single_width_value | |
| FT_UShort | delta_base | |
| FT_UShort | delta_shift | |
| FT_Bool | auto_flip | |
| FT_Byte | instruct_control | |
| FT_Bool | scan_control | |
| FT_Int | scan_type | |

**Definition:**

typedef struct TT_DriverRec_* TT_Driver;

### *struct TT_DriverRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_DriverRec | root | |

| | | |
|---|---|---|
| TT_GlyphZoneRec | zone | |
| FT_UInt | interpreter_version | |

**Definition:**
typedef struct TT_DriverRec_ { FT_DriverRec root; TT_GlyphZoneRec zone; /* glyph loader points zone */ FT_UInt interpreter_version; } TT_DriverRec;

## *typedef TT_GlyphSlot*

**Definition:**
typedef FT_GlyphSlot TT_GlyphSlot;

## *struct TT_GraphicsState_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | rp0 | |
| FT_UShort | rp1 | |
| FT_UShort | rp2 | |
| FT_UShort | gep0 | |
| FT_UShort | gep1 | |
| FT_UShort | gep2 | |
| FT_UnitVector | dualVector | |
| FT_UnitVector | projVector | |
| FT_UnitVector | freeVector | |
| FT_Long | loop | |
| FT_Int | round_state | |
| FT_F26Dot6 | compensation | |
| FT_F26Dot6 | minimum_distance | |
| FT_F26Dot6 | control_value_cutin | |
| FT_F26Dot6 | single_width_cutin | |
| FT_F26Dot6 | single_width_value | |
| FT_UShort | delta_base | |
| FT_UShort | delta_shift | |
| FT_Bool | auto_flip | |
| FT_Byte | instruct_control | |
| FT_Bool | scan_control | |
| FT_Int | scan_type | |

**Definition:**

typedef struct TT_GraphicsState_ { FT_UShort rp0; FT_UShort rp1; FT_UShort rp2; FT_UShort gep0; FT_UShort gep1; FT_UShort gep2; FT_UnitVector dualVector; FT_UnitVector projVector; FT_UnitVector freeVector; FT_Long loop; FT_Int round_state; FT_F26Dot6 compensation[4]; /* device-specific compensations */ /* default values below can be modified by 'fpgm' and 'prep' */ FT_F26Dot6 minimum_distance; FT_F26Dot6 control_value_cutin; FT_F26Dot6 single_width_cutin; FT_F26Dot6 single_width_value; FT_UShort delta_base; FT_UShort delta_shift; FT_Bool auto_flip; FT_Byte instruct_control; /* According to Greg Hitchcock from Microsoft, the `scan_control' */ /* variable as documented in the TrueType specification is a 32-bit */ /* integer; the high-word part holds the SCANTYPE value, the low-word */ /* part the SCANCTRL value. We separate it into two fields. */ FT_Bool scan_control; FT_Int scan_type; } TT_GraphicsState;

## *struct TT_SizeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |
| FT_Size_Metrics* | metrics | |
| FT_Size_Metrics | hinted_metrics | |
| TT_Size_Metrics | ttmetrics | |
| FT_Byte* | widthp | |
| FT_ULong | strike_index | |
| FT_Long | point_size | |
| TT_GraphicsState | GS | |
| TT_GlyphZoneRec | twilight | |
| TT_ExecContext | context | |
| FT_Error | bytecode_ready | |
| FT_Error | cvt_ready | |

**Definition:**

typedef struct TT_SizeRec_ { FT_SizeRec root; /* we have our own copy of metrics so that we can modify */ /* it without affecting auto-hinting (when used) */ FT_Size_Metrics* metrics; /* for the current rendering mode */ FT_Size_Metrics hinted_metrics; /* for the hinted rendering mode */ TT_Size_Metrics ttmetrics; FT_Byte* widthp; /* glyph widths from the hdmx table */ FT_ULong strike_index; /* 0xFFFFFFFF to indicate invalid */ #ifdef TT_USE_BYTECODE_INTERPRETER FT_Long point_size; /* for the `MPS' bytecode instruction */ TT_GraphicsState GS; TT_GlyphZoneRec twilight; /* The instance's twilight zone */ TT_ExecContext context; /* if negative, `fpgm' (resp. `prep'), wasn't executed yet; */ /* otherwise it is the returned error code */ FT_Error bytecode_ready; FT_Error cvt_ready; #endif /* TT_USE_BYTECODE_INTERPRETER */ } TT_SizeRec;

## *struct TT_Size_Metrics_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_Long | x_ratio | |

| | | |
|---|---|---|
| FT_Long | y_ratio | |
| FT_Long | ratio | |
| FT_Fixed | scale | |
| FT_UShort | ppem | |
| FT_Bool | rotated | |
| FT_Bool | stretched | |

**Definition:**

typedef struct TT_Size_Metrics_ { /* for non-square pixels */ FT_Long x_ratio; FT_Long y_ratio; FT_Long ratio; /* current ratio */ FT_Fixed scale; FT_UShort ppem; /* maximum ppem size */ FT_Bool rotated; /* `is the glyph rotated?'-flag */ FT_Bool stretched; /* `is the glyph stretched?'-flag */ } TT_Size_Metrics;

# t1load.h

## *Data Types*

### *struct T1_Loader_*

**Members:**

| Type | Name | Description |
|---|---|---|
| T1_ParserRec | parser | |
| FT_Int | num_chars | |
| PS_TableRec | encoding_table | |
| FT_Int | num_glyphs | |
| PS_TableRec | glyph_names | |
| PS_TableRec | charstrings | |
| PS_TableRec | swap_table | |
| FT_Int | num_subrs | |
| PS_TableRec | subrs | |
| FT_Hash | subrs_hash | |
| FT_Bool | fontdata | |
| FT_UInt | keywords_encountered | |

**Definition:**

typedef struct T1_Loader_ { T1_ParserRec parser; /* parser used to read the stream */ FT_Int num_chars; /* number of characters in encoding */ PS_TableRec encoding_table; /* PS_Table used to store the */ /* encoding character names */ FT_Int num_glyphs; PS_TableRec glyph_names; PS_TableRec charstrings; PS_TableRec swap_table; /* For moving .notdef glyph to index 0. */ FT_Int num_subrs; PS_TableRec subrs; FT_Hash subrs_hash; FT_Bool fontdata; FT_UInt keywords_encountered; /* T1_LOADER_ENCOUNTERED_XXX */ } T1_LoaderRec, *T1_Loader;

# t1objs.h

## *Data Types*

### *struct T1_CharMapRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |

**Definition:**
typedef struct T1_CharMapRec_* T1_CharMap;

### *struct T1_GlyphSlotRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |

**Definition:**
typedef struct T1_GlyphSlotRec_* T1_GlyphSlot;

### *struct T1_GlyphSlotRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_GlyphSlotRec | root | |
| FT_Bool | hint | |
| FT_Bool | scaled | |
| FT_Fixed | x_scale | |
| FT_Fixed | y_scale | |
| FT_Int | max_points | |
| FT_Int | max_contours | |

**Definition:**
typedef struct T1_GlyphSlotRec_ { FT_GlyphSlotRec root; FT_Bool hint; FT_Bool scaled; FT_Fixed x_scale; FT_Fixed y_scale; FT_Int max_points; FT_Int max_contours; } T1_GlyphSlotRec;

### *struct T1_SizeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |

**Definition:**
typedef struct T1_SizeRec_* T1_Size;

### *struct T1_SizeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |

**Definition:**

typedef struct T1_SizeRec_ { FT_SizeRec root; } T1_SizeRec;

### *struct T1_Size_Hints_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |

**Definition:**

typedef struct T1_Size_Hints_ T1_Size_Hints;

# t1parse.h

## *Data Types*

### *struct T1_ParserRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| PS_ParserRec | root | |
| FT_Stream | stream | |
| FT_Byte* | base_dict | |
| FT_ULong | base_len | |
| FT_Byte* | private_dict | |
| FT_ULong | private_len | |
| FT_Bool | in_pfb | |
| FT_Bool | in_memory | |
| FT_Bool | single_block | |

**Definition:**

typedef struct T1_ParserRec_ { PS_ParserRec root; FT_Stream stream; FT_Byte* base_dict; FT_ULong base_len; FT_Byte* private_dict; FT_ULong private_len; FT_Bool in_pfb; FT_Bool in_memory; FT_Bool single_block; } T1_ParserRec, *T1_Parser;

# t42objs.h

## *Data Types*

### *struct T42_SizeRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_SizeRec | root | |
| FT_Size | ttsize | |

**Definition:**
typedef struct T42_SizeRec_ { FT_SizeRec root; FT_Size ttsize; } T42_SizeRec, *T42_Size;

# t42parse.h

## *Data Types*

### *struct T42_ParserRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| PS_ParserRec | root | |
| FT_Stream | stream | |
| FT_Byte* | base_dict | |
| FT_Long | base_len | |
| FT_Bool | in_memory | |

**Definition:**

typedef struct T42_ParserRec_ { PS_ParserRec root; FT_Stream stream; FT_Byte* base_dict; FT_Long base_len; FT_Bool in_memory; } T42_ParserRec, *T42_Parser;

# t42types.h

## *Data Types*

### *struct T42_FaceRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_FaceRec | root | |
| T1_FontRec | type1 | |
| const void* | psnames | |
| const void* | psaux | |
| const void* | afm_data | |
| FT_Byte* | ttf_data | |
| FT_Long | ttf_size | |
| FT_Face | ttf_face | |
| FT_CharMapRec | charmaprecs | |
| FT_CharMap | charmaps | |
| PS_UnicodesRec | unicode_map | |

**Definition:**

typedef struct T42_FaceRec_ { FT_FaceRec root; T1_FontRec type1; const void* psnames; const void* psaux; #if 0 const void* afm_data; #endif FT_Byte* ttf_data; FT_Long ttf_size; FT_Face ttf_face; FT_CharMapRec charmaprecs[2]; FT_CharMap charmaps[2]; PS_UnicodesRec unicode_map; } T42_FaceRec, *T42_Face;

# winfnt.h

## *Data Types*

### *struct WinMZ_HeaderRec_*

**Members:**

| Type | Name | Description |
|---|---|---|
| FT_UShort | magic | |
| FT_UShort | lfanew | |

**Definition:**
typedef struct WinMZ_HeaderRec_ { FT_UShort magic; /* skipped content */ FT_UShort lfanew; } WinMZ_HeaderRec;

# glps_audio_stream.h

## *Functions*

### *glps_audio_stream_destroy*

**Signature:**
void glps_audio_stream_destroy(glps_audio_stream * am)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| am | glps_audio_stream * | Audio manager instance |

### *glps_audio_stream_pause*

**Signature:**
void glps_audio_stream_pause(glps_audio_stream * am)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| am | glps_audio_stream * | Audio manager instance |

### *glps_audio_stream_resume*

**Signature:**
void glps_audio_stream_resume(glps_audio_stream * am)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| am | glps_audio_stream * | Audio manager instance |

### *glps_audio_stream_set_position*

**Signature:**
void glps_audio_stream_set_position(glps_audio_stream * am, unsigned int position)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| am | glps_audio_stream * | Audio manager instance |
| position | unsigned int | Position in samples |

### *glps_audio_stream_set_volume*

**Signature:**
void glps_audio_stream_set_volume(glps_audio_stream * am, float volume)
**Parameters:**

| Name | Type | Description |
|---|---|---|

| | | |
|---|---|---|
| am | glps_audio_stream * | Audio manager instance |
| volume | float | Volume level (0.0 to 1.0) |

### *glps_audio_stream_stop*

**Signature:**
void glps_audio_stream_stop(glps_audio_stream * am)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| am | glps_audio_stream * | |

# glps_window_manager.h

## *Functions*

### **glps_wm_init*

**Description:** Header file for the GLPS window manager, handling window creation, rendering, and input events.
**Signature:**
glps_WindowManager *glps_wm_init()

### *glps_wm_cursor_change*

**Description:** Sets the callback for mouse movement events.
**Signature:**
void glps_wm_cursor_change(glps_WindowManager* wm, GLPS_CURSOR_TYPE cursor_type)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| wm | glps_WindowManager* | Pointer to the GLPS Window Manager. |
| cursor_type | GLPS_CURSOR_TYPE | |

### *glps_wm_destroy*

**Description:** Cleans up and destroys the GLPS Window Manager.
**Signature:**
void glps_wm_destroy(glps_WindowManager * wm)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| wm | glps_WindowManager * | Pointer to the GLPS Window Manager. |

### *glps_wm_get_fps*

**Description:** Attaches data to Clipboard.
**Signature:**
double glps_wm_get_fps(glps_WindowManager * wm, size_t window_id)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| wm | glps_WindowManager * | Pointer to the window manager struct. |
| window_id | size_t | |

### *glps_wm_get_window_count*

**Description:** Returns the total window count.
**Signature:**
size_t glps_wm_get_window_count(glps_WindowManager * wm)

**Parameters:**

| Name | Type | Description |
|---|---|---|
| wm | glps_WindowManager * | Pointer to the GLPS Window Manager. |

### *glps_wm_set_window_ctx_curr*

**Description:** Sets the OpenGL context of a specific window as the current context.
**Signature:**
void glps_wm_set_window_ctx_curr(glps_WindowManager * wm, size_t window_id)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| wm | glps_WindowManager * | Pointer to the GLPS Window Manager. |
| window_id | size_t | ID of the window to set the context for. |

### *glps_wm_swap_buffers*

**Description:** Swaps the front and back buffers for the specified window.
**Signature:**
void glps_wm_swap_buffers(glps_WindowManager * wm, size_t window_id)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| wm | glps_WindowManager * | Pointer to the GLPS Window Manager. |
| window_id | size_t | ID of the window to swap buffers for. |

### *glps_wm_swap_interval*

**Description:** Sets the swap interval for buffer swaps.
**Signature:**
void glps_wm_swap_interval(glps_WindowManager * wm, unsigned int swap_interval)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| wm | glps_WindowManager * | Pointer to the GLPS Window Manager. |
| swap_interval | unsigned int | Number of vertical refreshes between buffer swaps. |

### *glps_wm_window_destroy*

**Description:** Destroys the specified window.
**Signature:**
void glps_wm_window_destroy(glps_WindowManager * wm, size_t window_id)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| wm | glps_WindowManager * | Pointer to the GLPS Window Manager. |
| window_id | size_t | ID of the window to destroy. |

### *glps_wm_window_is_resizable*

**Description:** Creates a new window with the specified title and dimensions.
**Signature:**
void glps_wm_window_is_resizable(glps_WindowManager * wm, bool state, size_t window_id)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| wm | glps_WindowManager * | Pointer to the GLPS Window Manager. |
| state | bool | |
| window_id | size_t | |

# glps_common.h

## *Data Types*

### *enum GLPS_CURSOR_TYPE*

**Definition:**
* @enum GLPS_CURSOR_TYPE * @brief Cursor types. */ typedef enum { GLPS_CURSOR_ARROW, GLPS_CURSOR_IBEAM, GLPS_CURSOR_CROSSHAIR, GLPS_CURSOR_HAND, GLPS_CURSOR_HRESIZE, GLPS_CURSOR_VRESIZE, GLPS_CURSOR_NOT_ALLOWED } GLPS_CURSOR_TYPE;

### *enum GLPS_SCROLL_AXES*

**Definition:**
* @enum GLPS_SCROLL_AXES * @brief Scroll axis definitions. */ typedef enum { GLPS_SCROLL_H_AXIS, /**< Horizontal scroll axis. */ GLPS_SCROLL_V_AXIS /**< Vertical scroll axis. */ } GLPS_SCROLL_AXES;

### *enum GLPS_SCROLL_SOURCE*

**Definition:**
* @enum GLPS_SCROLL_SOURCE * @brief Scroll source definitions. */ typedef enum { GLPS_SCROLL_SOURCE_FINGER, /**< Scroll generated by a finger. */ GLPS_SCROLL_SOURCE_WHEEL, /**< Scroll generated by a wheel. */ GLPS_SCROLL_SOURCE_CONTINUOUS, /**< Continuous scrolling source. */ GLPS_SCROLL_SOURCE_WHEEL_TILT, /**< Tilted wheel scroll source. */ GLPS_SCROLL_SOURCE_OTHER /**< Other scroll source. */ } GLPS_SCROLL_SOURCE;

### *typedef buffer_frames*

**Definition:**
typedef struct { // Thread management void *thread; /**< Thread for audio playback. */ void *volume_mutex; /**< Mutex for volume control. */ void *state_mutex; /**< Mutex for state control. */ const char *audio_file_path; /**< Path to the audio file. */ const char *device_name; /**< Name of the audio device. */ int buffer_frames; /**< Number of frames in the buffer. */ float *pcm_buffer; /**< Pointer to the PCM buffer. */ int sample_rate; /**< Sample rate of the audio. */ int channels; /**< Number of audio channels. */ int bits_per_sample; /**< Number of bits per sample. */ int buffer_size; /**< Size of the audio buffer. */ float volume; /**< Volume level. */ int position; /**< Position in the audio stream. */ int is_playing; /**< Flag indicating if audio is playing. */ int is_paused; /**< Flag indicating if audio is paused. */ int is_stopped; /**< Flag indicating if audio is stopped. */ void *pcm_handle; /**< Pointer to the PCM handle. */ void *mp3; /**< Pointer to the MP3 decoder. */ } glps_audio_stream;

### *struct clipboard_data*

**Members:**

| Type | Name | Description |
|---|---|---|
| glps_WindowManager | *wm | |

| | | |
|---|---|---|
| size_t | window_id | |

**Definition:**
struct clipboard_data clipboard; /**< Current clipboard data. */

## *typedef current_serial*

**Description:** Represents the Wayland context for GLPS.
**Definition:**
typedef struct { struct wl_display *wl_display; /**< Wayland display. */ struct wl_registry *wl_registry; /**< Wayland registry. */ struct wl_compositor *wl_compositor; /**< Wayland compositor. */ struct wl_seat *wl_seat; /**< Wayland seat. */ struct xdg_wm_base *xdg_wm_base; /**< XDG WM base. */ struct zxdg_decoration_manager_v1 *decoration_manager; /**< Decoration manager. */ struct xdg_toplevel_tag_manager_v1 *tag_manager; /**< Tag manager. */ struct wl_data_device_manager *data_dvc_manager; /**< Data control Manager. */ struct wl_data_device *data_dvc; /**< Data device to interact with Clipboard and Drag&Drop; operations. */ struct wl_data_source *data_src; /**< Clipboard data source.*/ struct wl_pointer *wl_pointer; /**< Wayland pointer. */ struct wl_keyboard *wl_keyboard; /**< Wayland keyboard. */ struct xkb_state *xkb_state; /**< Keyboard state. */ struct xkb_context *xkb_context; /**< Keyboard context. */ struct xkb_keymap *xkb_keymap; /**< Keyboard keymap. */ struct wl_touch *wl_touch; /**< Wayland touch interface. */ struct wl_data_offer *current_drag_offer; uint32_t current_serial; uint32_t keyboard_serial; size_t keyboard_window_id; size_t mouse_window_id; size_t touch_window_id; size_t current_drag_n_drop_window; glps_DropCoordinates drop_coordinates; } glps_WaylandContext;

## *typedef dpy*

**Description:** EGL context for rendering.
**Definition:**
typedef struct { EGLDisplay dpy; /**< EGL display. */ EGLContext ctx; /**< EGL context. */ EGLConfig conf; /**< EGL configuration. */ } glps_EGLContext;

## *typedef egl_surface*

**Definition:**
typedef struct { EGLSurface egl_surface; /**< EGL surface. */ struct wl_egl_window *egl_window; /**< X11 EGL window. */ Window window; /**< X11 window identifier. */ bool fps_is_init; struct timespec fps_start_time; bool is_desktop; // New field to mark desktop window } glps_X11Window;

## *struct glps_WaylandWindow*

**Members:**

| Type | Name | Description |
|---|---|---|
| struct xdg_surface | *xdg_surface | |
| struct xdg_toplevel | *xdg_toplevel | |
| struct wl_surface | *wl_surface | |
| EGLSurface | egl_surface | |
| struct wl_egl_window | *egl_window | |

| | | |
|---|---|---|
| glps_WindowProperties | properties | |
| struct zxdg_toplevel_decoration_v1 | *zxdg_toplevel_decoration | |
| struct wl_callback | *frame_callback | |
| struct timespec | fps_start_time | |
| bool | fps_is_init | |
| void | *frame_args | |
| uint32_t | serial | |

**Definition:**
* @struct glps_WaylandWindow * @brief Represents a Wayland window in GLPS. */ typedef struct { struct xdg_surface *xdg_surface; /**< XDG surface. */ struct xdg_toplevel *xdg_toplevel; /**< XDG toplevel. */ struct wl_surface *wl_surface; /**< Wayland surface. */ EGLSurface egl_surface; /**< EGL surface. */ struct wl_egl_window *egl_window; /**< Wayland EGL window. */ glps_WindowProperties properties; /**< Window properties. */ struct zxdg_toplevel_decoration_v1 *zxdg_toplevel_decoration; struct wl_callback *frame_callback; // FPS COUNTER struct timespec fps_start_time; bool fps_is_init; void *frame_args; uint32_t serial; } glps_WaylandWindow;

## *struct glps_debug*

**Members:**

| Type | Name | Description |
|---|---|---|
| glps_WindowManager | *wm | |
| size_t | window_id | |

**Definition:**
struct glps_debug debug_utilities;

## *typedef hwnd*

**Definition:**
typedef struct { HWND hwnd; HDC hdc; glps_WindowProperties properties; LARGE_INTEGER fps_start_time; LARGE_INTEGER fps_freq; bool fps_is_init; } glps_Win32Window;

## *struct pointer_event*

**Members:**

| Type | Name | Description |
|---|---|---|
| uint32_t | event_mask | |
| wl_fixed_t | surface_x | |
| wl_fixed_t | surface_y | |
| uint32_t | button | |
| uint32_t | state | |
| uint32_t | time | |
| uint32_t | serial | |

| | | |
|---|---|---|
| bool | valid | |
| wl_fixed_t | value | |
| int32_t | discrete | |
| uint32_t | axis_source | |
| size_t | window_id | |

**Definition:**
* @struct pointer_event * @brief Represents pointer event data. */ struct pointer_event { uint32_t event_mask; /**< Pointer event mask. */ wl_fixed_t surface_x; /**< X-coordinate of pointer. */ wl_fixed_t surface_y; /**< Y-coordinate of pointer. */ uint32_t button; /**< Button identifier. */ uint32_t state; /**< State of the button (pressed/released). */ uint32_t time; /**< Timestamp of the event. */ uint32_t serial; /**< Serial number of the event. */ struct { bool valid; /**< Indicates if the axis data is valid. */ wl_fixed_t value; /**< Axis value. */ int32_t discrete; /**< Discrete axis value. */ } axes[2]; /**< Data for horizontal and vertical axes. */ uint32_t axis_source; /**< Source of the axis event. */ size_t window_id; };

## *struct pointer_event*

**Members:**

| Type | Name | Description |
|---|---|---|
| glps_WindowManager | *wm | |
| size_t | window_id | |

**Definition:**
struct pointer_event pointer_event; /**< Current pointer event data. */

## *enum pointer_event_mask*

**Definition:**
* @enum pointer_event_mask * @brief Bitmask for pointer event types. */ enum pointer_event_mask { POINTER_EVENT_ENTER = 1 << 0, /**< Pointer entered surface. */ POINTER_EVENT_LEAVE = 1 << 1, /**< Pointer left surface. */ POINTER_EVENT_MOTION = 1 << 2, /**< Pointer motion. */ POINTER_EVENT_BUTTON = 1 << 3, /**< Pointer button pressed/released. */ POINTER_EVENT_AXIS = 1 << 4, /**< Pointer axis event. */ POINTER_EVENT_AXIS_SOURCE = 1 << 5, /**< Pointer axis source event. */ POINTER_EVENT_AXIS_STOP = 1 << 6, /**< Pointer axis stop event. */ POINTER_EVENT_AXIS_DISCRETE = 1 << 7, /**< Pointer axis discrete event. */ };

## *struct timespec*

**Members:**

| Type | Name | Description |
|---|---|---|
| char | mime_type | |
| char | buff | |

**Definition:**
struct timespec fps_start_time;

## *struct touch_event*

**Members:**

| Type | Name | Description |
|---|---|---|
| uint32_t | event_mask | |
| uint32_t | time | |
| uint32_t | serial | |
| struct touch_point | points | |
| size_t | window_id | |

**Definition:**
* @struct touch_event * @brief Represents touch event data. */ struct touch_event { uint32_t event_mask; /**< Event mask for the touch event. */ uint32_t time; /**< Timestamp of the event. */ uint32_t serial; /**< Serial number of the event. */ struct touch_point points[10]; /**< Array of touch points. */ size_t window_id; };

## *typedef touch_event*

**Description:** Represents the manager for GLPS windows.
**Definition:**
typedef struct { #ifdef GLPS_USE_WAYLAND glps_WaylandContext *wayland_ctx; /**< Wayland context. */ glps_WaylandWindow **windows; /**< Array of Wayland window pointers. */ glps_EGLContext *egl_ctx; /**< EGL context. */ struct touch_event touch_event; /**< Current touch event data. */ struct pointer_event pointer_event; /**< Current pointer event data. */ struct clipboard_data clipboard; /**< Current clipboard data. */ #endif #ifdef GLPS_USE_WIN32 glps_Win32Context *win32_ctx; /**< Win32 Context */ glps_Win32Window **windows; /**< Array of Win32 window pointers. */ WNDCLASSEX wc; #endif #ifdef GLPS_USE_X11 glps_EGLContext *egl_ctx; /**< EGL context. */ glps_X11Context *x11_ctx; glps_X11Window **windows; /**< Array of X11 window pointers. */ #endif char font_path[256]; /**< Path to the font file. */ size_t window_count; /**< Number of managed windows. */ bool inhibit_reset; /**< Indicates if reset should be inhibited. */ unsigned int selected_color; /**< Selected color value. */ struct glps_debug debug_utilities; struct glps_Callback callbacks; bool should_close; } glps_WindowManager;

## *struct touch_event*

**Members:**

| Type | Name | Description |
|---|---|---|
| glps_WindowManager | *wm | |
| size_t | window_id | |

**Definition:**
struct touch_event touch_event; /**< Current touch event data. */

## *enum touch_event_mask*

**Definition:**
* @enum touch_event_mask * @brief Bitmask for touch event types. */ enum touch_event_mask { TOUCH_EVENT_DOWN = 1 << 0, /**< Touch down event. */ TOUCH_EVENT_UP = 1 << 1, /**< Touch up event. */ TOUCH_EVENT_MOTION = 1 << 2, /**< Touch motion event. */ TOUCH_EVENT_CANCEL = 1 << 3, /**< Touch cancel event. */ TOUCH_EVENT_SHAPE = 1 << 4,

/**< Touch shape event. */ TOUCH_EVENT_ORIENTATION = 1 << 5, /**< Touch orientation event. */ };

## *struct touch_point*

**Members:**

| Type | Name | Description |
|---|---|---|
| bool | valid | |
| int32_t | id | |
| uint32_t | event_mask | |
| wl_fixed_t | surface_x | |
| wl_fixed_t | surface_y | |
| wl_fixed_t | major | |
| wl_fixed_t | minor | |
| wl_fixed_t | orientation | |

**Definition:**

* @struct touch_point * @brief Represents a single touch point. */ struct touch_point { bool valid; /**< Indicates if the touch point is valid. */ int32_t id; /**< Identifier of the touch point. */ uint32_t event_mask; /**< Event mask for the touch point. */ wl_fixed_t surface_x; /**< X-coordinate of the touch point. */ wl_fixed_t surface_y; /**< Y-coordinate of the touch point. */ wl_fixed_t major; /**< Major axis of the touch point. */ wl_fixed_t minor; /**< Minor axis of the touch point. */ wl_fixed_t orientation; /**< Orientation of the touch point. */ };

## *struct touch_point*

**Members:**

| Type | Name | Description |
|---|---|---|
| struct xdg_surface | *xdg_surface | |
| struct xdg_toplevel | *xdg_toplevel | |
| struct wl_surface | *wl_surface | |
| EGLSurface | egl_surface | |
| struct wl_egl_window | *egl_window | |
| glps_WindowProperties | properties | |
| struct zxdg_toplevel_decoration_v1 | *zxdg_toplevel_decoration | |
| struct wl_callback | *frame_callback | |
| struct timespec | fps_start_time | |
| bool | fps_is_init | |
| void | *frame_args | |
| uint32_t | serial | |

**Definition:**

struct touch_point points[10]; /**< Array of touch points. */

### *typedef width*

**Description:** Properties for a GLPS window.
**Definition:**
typedef struct { char title[64]; /**< Title of the window. */ int width; int height; } glps_WindowProperties;

### *typedef window_id*

**Description:** Arguments for frame callbacks.
**Definition:**
typedef struct { glps_WindowManager *wm; /**< Window Manager. */ size_t window_id; /**< ID of the window. */ } frame_callback_args;

### *struct wl_compositor*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct wl_compositor *wl_compositor; /**< Wayland compositor. */

### *struct wl_data_device*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct wl_data_device *data_dvc; /**< Data device to interact with Clipboard

### *struct wl_data_device_manager*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct wl_data_device_manager *data_dvc_manager; /**< Data control Manager. */

### *struct wl_data_offer*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**

struct wl_data_offer *current_drag_offer;

### *struct wl_data_source*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**

struct wl_data_source *data_src; /**< Clipboard data source.*/

### *struct wl_egl_window*

**Members:**

| Type | Name | Description |
|---|---|---|
| int | x | |
| int | y | |

**Definition:**

struct wl_egl_window *egl_window; /**< Wayland EGL window. */

### *struct wl_egl_window*

**Members:**

| Type | Name | Description |
|---|---|---|
| char | mime_type | |
| char | buff | |

**Definition:**

struct wl_egl_window *egl_window; /**< X11 EGL window. */

### *struct wl_keyboard*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct wl_keyboard *wl_keyboard; /**< Wayland keyboard. */

### *struct wl_pointer*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct wl_pointer *wl_pointer; /**< Wayland pointer. */

### *struct wl_registry*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct wl_registry *wl_registry; /**< Wayland registry. */

### *struct wl_seat*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct wl_seat *wl_seat; /**< Wayland seat. */

### *struct wl_surface*

**Members:**

| Type | Name | Description |
|---|---|---|

| | | |
|---|---|---|
| int | x | |
| int | y | |

**Definition:**
struct wl_surface *wl_surface; /**< Wayland surface. */

### *struct wl_touch*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct wl_touch *wl_touch; /**< Wayland touch interface. */

### *struct xdg_toplevel*

**Members:**

| Type | Name | Description |
|---|---|---|
| int | x | |
| int | y | |

**Definition:**
struct xdg_toplevel *xdg_toplevel; /**< XDG toplevel. */

### *struct xdg_toplevel_tag_manager_v1*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct xdg_toplevel_tag_manager_v1 *tag_manager; /**< Tag manager. */

### *struct xdg_wm_base*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct xdg_wm_base *xdg_wm_base; /**< XDG WM base. */

### *struct xkb_context*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct xkb_context *xkb_context; /**< Keyboard context. */

### *struct xkb_keymap*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct xkb_keymap *xkb_keymap; /**< Keyboard keymap. */

### *struct xkb_state*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct xkb_state *xkb_state; /**< Keyboard state. */

### *struct zxdg_decoration_manager_v1*

**Members:**

| Type | Name | Description |
|---|---|---|
| EGLDisplay | dpy | |
| EGLContext | ctx | |
| EGLConfig | conf | |

**Definition:**
struct zxdg_decoration_manager_v1 *decoration_manager; /**< Decoration manager. */

### *struct zxdg_toplevel_decoration_v1*

**Members:**

| Type | Name | Description |
|---|---|---|
| int | x | |
| int | y | |

**Definition:**

struct zxdg_toplevel_decoration_v1 *zxdg_toplevel_decoration;

# glps_wayland.h

## *Functions*

### *wl_update*

**Description:** Updates the specified window, handling rendering and events.
**Signature:**
void wl_update(glps_WindowManager * wm, size_t window_id)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| wm | glps_WindowManager * | Pointer to the GLPS Window Manager. |
| window_id | size_t | ID of the window to update. |

## *Data Types*

### *struct wl_pointer*

**Description:** Updates the specified window, handling rendering and events.
**Definition:**
void wl_pointer_enter(void *data, struct wl_pointer *wl_pointer, uint32_t serial, struct wl_surface *surface, wl_fixed_t surface_x, wl_fixed_t surface_y);

# pico_logger.h

## *Functions*

### *dump_memory*

**Description:** Dumps the memory contents of a buffer. This function outputs the contents of a memory buffer in a human-readable hexadecimal format, useful for debugging low-level memory issues.
**Signature:**
void dump_memory(const char * label, const void * buffer, size_t size)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| label | const char * | A label describing the memory buffer. |
| buffer | const void * | A pointer to the memory buffer to be dumped. |
| size | size_t | The size of the memory buffer in bytes. |

### *log_message*

**Description:** Provides logging and debugging utilities with configurable log levels.
**Signature:**
void log_message(DebugLevel level, const char * file, int line, const char * func, const char * fmt)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| level | DebugLevel | |
| file | const char * | |
| line | int | |
| func | const char * | |
| fmt | const char * | |

### *log_performance*

**Description:** Logs a performance metric message. This function logs a performance-related message, useful for timing and profiling.
**Signature:**
void log_performance(char * message)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| message | char * | The performance-related message to log. |

### *print_stack_trace*

**Description:** Prints the current stack trace. This function outputs the current call stack to aid in debugging and tracing the program's execution flow.

**Signature:**
void print_stack_trace()

### *save_log_file*

**Description:** Saves the logged messages to a file. This function writes all the accumulated log messages to a specified file. Each log message is written to a new line in the file. If the file cannot be opened, an error is printed to the standard output.
**Signature:**
void save_log_file(const char * path)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| path | const char * | The path to the file where the log messages will be saved. If the file does not e |

### *set_logging_enabled*

**Description:** Macro to log a performance-related message. This macro wraps the `log_performance` function for easier usage.
**Signature:**
void set_logging_enabled(bool enabled)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| enabled | bool | |

### *set_minimum_log_level*

**Description:** Sets the minimum log level for logging. This function defines the minimum log level for messages to be logged. Messages below the specified level will not be logged.
**Signature:**
void set_minimum_log_level(DebugLevel level)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| level | DebugLevel | The minimum log level (e.g., DEBUG_LEVEL_WARNING). |

## *Data Types*

### *enum DebugLevel*

**Description:** Provides logging and debugging utilities with configurable log levels.
**Definition:**
* @enum DebugLevel * @brief Log levels used for message categorization. * * The log levels allow messages to be classified by severity, ranging from * general informational messages to critical errors. */ typedef enum { DEBUG_LEVEL_INFO, /**< Informational messages */ DEBUG_LEVEL_WARNING, /**< Warnings indicating potential issues */ DEBUG_LEVEL_ERROR, /**< Error messages indicating a problem */ DEBUG_LEVEL_CRITICAL /**< Critical error messages indicating a failure */ } DebugLevel;

# nfd.h

## *Functions*

### *NFD_ClearError*

**Signature:**
NFD_API void NFD_ClearError()

### *NFD_FreePathN*

**Signature:**
NFD_API void NFD_FreePathN(nfdnchar_t* filePath)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| filePath | nfdnchar_t* | |

### *NFD_FreePathU8*

**Signature:**
NFD_API void NFD_FreePathU8(nfdu8char_t* filePath)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| filePath | nfdu8char_t* | |

### *NFD_GetError*

**Signature:**
NFD_API const char* NFD_GetError()

### *NFD_Init*

**Signature:**
NFD_API nfdresult_t NFD_Init()

### *NFD_OpenDialogMultipleN_With_Impl*

**Signature:**
return NFD_OpenDialogMultipleN_With_Impl()

### *NFD_OpenDialogMultipleU8_With_Impl*

**Signature:**
return NFD_OpenDialogMultipleU8_With_Impl()

### *NFD_OpenDialogN_With_Impl*

**Signature:**
return NFD_OpenDialogN_With_Impl()

### *NFD_OpenDialogU8_With_Impl*

**Signature:**
return NFD_OpenDialogU8_With_Impl()

### *NFD_PathSet_EnumNextN*

**Signature:**
NFD_API nfdresult_t NFD_PathSet_EnumNextN(nfdpathsetenum_t* enumerator, nfdnchar_t** outPath)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| enumerator | nfdpathsetenum_t* | |
| outPath | nfdnchar_t** | |

### *NFD_PathSet_EnumNextU8*

**Signature:**
NFD_API nfdresult_t NFD_PathSet_EnumNextU8(nfdpathsetenum_t* enumerator, nfdu8char_t** outPath)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| enumerator | nfdpathsetenum_t* | |
| outPath | nfdu8char_t** | |

### *NFD_PathSet_Free*

**Signature:**
NFD_API void NFD_PathSet_Free(const nfdpathset_t* pathSet)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| pathSet | const nfdpathset_t* | |

### *NFD_PathSet_FreeEnum*

**Signature:**
NFD_API void NFD_PathSet_FreeEnum(nfdpathsetenum_t* enumerator)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| enumerator | nfdpathsetenum_t* | |

### *NFD_PathSet_FreePathN*

**Signature:**
NFD_API void NFD_PathSet_FreePathN(const nfdnchar_t* filePath)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| filePath | const nfdnchar_t* | |

### *NFD_PathSet_FreePathU8*

**Signature:**
NFD_API void NFD_PathSet_FreePathU8(const nfdu8char_t* filePath)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| filePath | const nfdu8char_t* | |

### *NFD_PathSet_GetCount*

**Signature:**
NFD_API nfdresult_t NFD_PathSet_GetCount(const nfdpathset_t* pathSet, nfdpathsetsize_t* count)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| pathSet | const nfdpathset_t* | |
| count | nfdpathsetsize_t* | |

### *NFD_PickFolderMultipleN_With_Impl*

**Signature:**
return NFD_PickFolderMultipleN_With_Impl()

### *NFD_PickFolderMultipleU8_With_Impl*

**Signature:**
return NFD_PickFolderMultipleU8_With_Impl()

### *NFD_PickFolderN*

**Signature:**
NFD_API nfdresult_t NFD_PickFolderN(nfdnchar_t** outPath, const nfdnchar_t* defaultPath)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| outPath | nfdnchar_t** | |
| defaultPath | const nfdnchar_t* | If null, the operating system will decide. |

### *NFD_PickFolderN_With_Impl*

**Signature:**
return NFD_PickFolderN_With_Impl()

### *NFD_PickFolderU8*

**Signature:**
NFD_API nfdresult_t NFD_PickFolderU8(nfdu8char_t** outPath, const nfdu8char_t* defaultPath)
**Parameters:**

| Name | Type | Description |
|---|---|---|
| outPath | nfdu8char_t** | |
| defaultPath | const nfdu8char_t* | If null, the operating system will decide. |

### *NFD_PickFolderU8_With_Impl*

**Signature:**
return NFD_PickFolderU8_With_Impl()

### *NFD_Quit*

**Signature:**
NFD_API void NFD_Quit()

### *NFD_SaveDialogN_With_Impl*

**Signature:**
return NFD_SaveDialogN_With_Impl()

### *NFD_SaveDialogU8_With_Impl*

**Signature:**
return NFD_SaveDialogU8_With_Impl()

## *Data Types*

### *typedef nfdchar_t*

**Definition:**
typedef nfdnchar_t nfdchar_t;

### *typedef nfdnchar_t*

**Definition:**
typedef wchar_t nfdnchar_t;

### *typedef nfdnchar_t*

**Definition:**
typedef nfdu8char_t nfdnchar_t;

### *struct nfdnfilteritem_t*

**Members:**

| Type | Name | Description |
|---|---|---|
| const nfdnchar_t* | name | |
| const nfdnchar_t* | spec | |

**Definition:**
typedef struct { const nfdnchar_t* name; const nfdnchar_t* spec; } nfdnfilteritem_t;

### *typedef nfdnfilteritem_t*

**Definition:**
typedef nfdu8filteritem_t nfdnfilteritem_t;

### *typedef nfdpathset_t*

**Definition:**
typedef void nfdpathset_t;

### *typedef nfdpathsetsize_t*

**Definition:**
typedef unsigned long nfdpathsetsize_t;

### *struct nfdu8filteritem_t*

**Members:**

| Type | Name | Description |
|---|---|---|
| const nfdu8char_t* | name | |
| const nfdu8char_t* | spec | |

**Definition:**
typedef struct { const nfdu8char_t* name; const nfdu8char_t* spec; } nfdu8filteritem_t;