Appearance Manager Reference

Carbon > User Experience

2007-01-23

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Contents

Appearance Manager Reference

Framework:	Carbon/Carbon.h
Declared in	Appearance.h

Overview

The Appearance Manager coordinates the look of human interface elements in Mac OS X. You can use the Appearance Manager to adapt any nonstandard interface elements in your program to the same coordinated look as the rest of Mac OS X. The Appearance Manager also provides many standard human interface elements, such as focus rings and group boxes, that can eliminate the need to create and maintain your own custom solutions.

In Mac OS X v10.3 and later, the Appearance Manager provides a new API called HITheme for drawing appearance primitives. Currently, this API is documented in the `HITheme.h` interface file. The HITheme API is similar to the legacy Appearance Manager API, but many parameters have been modified to use Quartz 2D types instead of QuickDraw types. For example, HITheme functions draw into a Quartz graphics context instead of the current QuickDraw graphics port. The legacy Appearance Manager API is implemented on top of the HITheme API, so using the new API can provide a significant performance advantage.

Functions by Task

Accessing Theme Information

`CopyThemeIdentifier` (page 17)
Retrieves a string identifying the current theme variant.

`GetTheme` (page 18)
Obtains a collection containing data describing the current theme.

`GetThemeMetric` (page 23)
Retrieves the value of a metric property of a user interface element.

`GetThemeFont` (page 180) Deprecated in Mac OS X v10.5
Obtains information about a system font in the current theme. (Deprecated. Some theme fonts cannot be drawn using QuickDraw; use `HIThemeDrawTextBox` instead.)

`IsValidAppearanceFileType` (page 199) Deprecated in Mac OS X v10.5
Returns whether the system can interpret files of a given file type as appearance files. (Deprecated. There is no replacement function.)

`IterateThemes` (page 200) Deprecated in Mac OS X v10.5
Iterates over all themes installed on a system. (Deprecated. There is no replacement function.)

SetTheme (page 204) Deprecated in Mac OS X v10.5
Sets a specified collection as the current theme. (Deprecated. There is no replacement function.)

UseThemeFont (page 209) Deprecated in Mac OS X v10.5
Sets the font of the current graphics port to one of the current theme's system fonts. (Deprecated. Use HIThemeDrawTextBox instead.)

Drawing Theme-Compliant Controls

GetThemeCheckBoxStyle (page 19)
Obtains the system preference for the type of mark to use in a checkbox.

GetThemeScrollBarArrowStyle (page 24)
Obtains the system preference for the type of scroll bar arrows to be used.

GetThemeScrollBarThumbStyle (page 24)
Obtains the system preference for the type of scroll box to be used.

DrawThemeButton (page 149) Deprecated in Mac OS X v10.5
Draws a button. (Deprecated. Use HIThemeDrawButton instead.)

DrawThemeChasingArrows (page 150) Deprecated in Mac OS X v10.5
Draws an asynchronous arrows indicator. (Deprecated. Use HIThemeDrawChasingArrows instead.)

DrawThemeEditTextFrame (page 151) Deprecated in Mac OS X v10.5
Draws an editable text frame. (Deprecated. Use HIThemeDrawFrame instead.)

DrawThemeFocusRect (page 152) Deprecated in Mac OS X v10.5
Draws or erases a focus ring around a specified rectangle. (Deprecated. Use HIThemeDrawFocusRect instead.)

DrawThemeFocusRegion (page 152) Deprecated in Mac OS X v10.5
Draws or erases a focus ring around a specified region. (Deprecated. Use HIThemeDrawFocusRect instead.)

DrawThemeGenericWell (page 153) Deprecated in Mac OS X v10.5
Draws an image well frame. (Deprecated. Use HIThemeDrawGenericWell instead.)

DrawThemeListBoxFrame (page 154) Deprecated in Mac OS X v10.5
Draws a list box frame. (Deprecated. Use HIThemeDrawFrame instead.)

DrawThemePlacard (page 159) Deprecated in Mac OS X v10.5
Draws a placard. (Deprecated. Use HIThemeDrawPlacard instead.)

DrawThemePopupArrow (page 160) Deprecated in Mac OS X v10.5
Draws a pop-up arrow. (Deprecated. Use HIThemeDrawPopupArrow instead.)

DrawThemePrimaryGroup (page 161) Deprecated in Mac OS X v10.5
Draws a primary group box frame. (Deprecated. Use HIThemeDrawGroupBox instead.)

DrawThemeScrollBarArrows (page 162) Deprecated in Mac OS X v10.5
Draws scroll bar arrows consistent with the current system preferences. (Deprecated. Use HIThemeDrawTrack, which draws the entire scrollbar including both the track and arrows.)

DrawThemeSecondaryGroup (page 164) Deprecated in Mac OS X v10.5
Draws a secondary group box frame. (Deprecated. Use HIThemeDrawGroupBox instead.)

DrawThemeSeparator (page 164) Deprecated in Mac OS X v10.5
Draws a separator line. (Deprecated. Use HIThemeDrawSeparator instead.)

DrawThemeTab (page 167) Deprecated in Mac OS X v10.5
Draws a tab. (Deprecated. Use HIThemeDrawTab instead.)

DrawThemeTabPane (page 168) Deprecated in Mac OS X v10.5
Draws a tab pane. (Deprecated. Use HIThemeDrawTabPane instead.)

DrawThemeTickMark (page 171) Deprecated in Mac OS X v10.5
Draws a tick mark. (Deprecated. Use HIThemeDrawTickMark instead.)

DrawThemeTrack (page 172) Deprecated in Mac OS X v10.5
Draws a track. (Deprecated. Use HIThemeDrawTrack instead.)

DrawThemeTrackTickMarks (page 173) Deprecated in Mac OS X v10.5
Draws tick marks for a track. (Deprecated. Use HIThemeDrawTrackTickMarks instead.)

GetThemeButtonBackgroundBounds (page 178) Deprecated in Mac OS X v10.5
Obtains the rectangle that contains a button. (Deprecated. Use HIThemeGetButtonBackgroundBounds instead.)

GetThemeButtonContentBounds (page 178) Deprecated in Mac OS X v10.5
Obtains the rectangle where content can be drawn for a button. (Deprecated. Use HIThemeGetButtonContentBounds instead.)

GetThemeButtonRegion (page 179) Deprecated in Mac OS X v10.5
Obtains the region occupied by a button. (Deprecated. Use HIThemeGetButtonShape instead.)

GetThemeScrollBarTrackRect (page 182) Deprecated in Mac OS X v10.5
Obtains the area containing the track portion of a scroll bar. (Deprecated. Use HIThemeGetScrollBarTrackRect instead.)

GetThemeTabRegion (page 184) Deprecated in Mac OS X v10.5
Obtains the region occupied by a tab. (Deprecated. Use HIThemeGetTabDrawShape instead.)

GetThemeTrackBounds (page 186) Deprecated in Mac OS X v10.5
Obtains the bounding rectangle of a track. (Deprecated. Use HIThemeGetTrackBounds instead.)

GetThemeTrackDragRect (page 186) Deprecated in Mac OS X v10.5
Obtains the area in which the user may drag a track's indicator. (Deprecated. Use HIThemeGetTrackDragRect instead.)

GetThemeTrackLiveValue (page 187) Deprecated in Mac OS X v10.5
Obtains the current value of a track's indicator, given its relative position. (Deprecated. Use HIThemeGetTrackLiveValue instead.)

GetThemeTrackThumbPositionFromOffset (page 188) Deprecated in Mac OS X v10.5
Obtains the relative position of a track's indicator, given an offset from its prior position. (Deprecated. Use HIThemeGetTrackThumbPositionFromOffset instead.)

GetThemeTrackThumbPositionFromRegion (page 189) Deprecated in Mac OS X v10.5
Obtains the relative position of a track's indicator, given its current position. (Deprecated. use HIThemeGetTrackThumbPositionFromBounds instead.)

GetThemeTrackThumbRgn (page 189) Deprecated in Mac OS X v10.5
Obtains the region containing a track's indicator. (Deprecated. Use HIThemeGetTrackThumbShape instead.)

HitTestThemeScrollBarArrows (page 192) Deprecated in Mac OS X v10.5
Returns whether the user clicked upon the specified scroll bar's arrows. (Deprecated. Use HIThemeHitTestScrollBarArrows instead.)

HitTestThemeTrack (page 194) Deprecated in Mac OS X v10.5
Returns whether the user clicked upon the specified track. (Deprecated. Use HIThemeHitTestTrack instead.)

Drawing Theme-Compliant Menus

GetThemeMenuBarHeight (page 21)
Obtains the height of a menu bar.

GetThemeMenuItemExtra (page 21)
Obtains a measurement of the space surrounding a menu item.

GetThemeMenuSeparatorHeight (page 22)
Obtains the height of a menu separator line.

GetThemeMenuTitleExtra (page 23)
Obtains a measurement of the space to either side of a menu title.

DrawThemeMenuBackground (page 154) Deprecated in Mac OS X v10.5
Draws a menu background. (Deprecated. Use `HIThemeDrawMenuBackground` instead.)

DrawThemeMenuBarBackground (page 155) Deprecated in Mac OS X v10.5
Draws a menu bar background. (Deprecated. Use `HIThemeDrawMenuBarBackground` instead.)

DrawThemeMenuItem (page 156) Deprecated in Mac OS X v10.5
Draws a menu item. (Deprecated. Use `HIThemeDrawMenuItem` instead.)

DrawThemeMenuSeparator (page 157) Deprecated in Mac OS X v10.5
Draws a menu item separator line. (Deprecated. Use `HIThemeDrawMenuSeparator` instead.)

DrawThemeMenuTitle (page 158) Deprecated in Mac OS X v10.5
Draws a menu title. (Deprecated. Use `HIThemeDrawMenuTitle` instead.)

GetThemeMenuBackgroundRegion (page 181) Deprecated in Mac OS X v10.5
Obtains the background region for a menu. (Deprecated. Use `HIThemeGetMenuBackgroundShape` instead.)

Drawing Theme-Compliant Windows

DrawThemeModelessDialogFrame (page 159) Deprecated in Mac OS X v10.5
Draws a beveled outline inside the content area of a modeless dialog box. (Deprecated. Use `HIThemeDrawWindowFrame` instead.)

DrawThemeScrollBarDelimiters (page 163) Deprecated in Mac OS X v10.5
Outlines a window's scroll bars. (Deprecated. Use `HIThemeDrawScrollBarDelimiters` instead.)

DrawThemeStandaloneGrowBox (page 165) Deprecated in Mac OS X v10.5
Draws a size box. (Deprecated. Use `HIThemeDrawGrowBox` instead.)

DrawThemeStandaloneNoGrowBox (page 166) Deprecated in Mac OS X v10.5
Draws a fill image for use in the corner space between scroll bars. (Deprecated. Use `HIThemeDrawGrowBox` instead.)

DrawThemeTitleBarWidget (page 171) Deprecated in Mac OS X v10.5
Draws a close box, zoom box, or collapse box. (Deprecated. Use `HIThemeDrawTitleBarWidget` instead.)

DrawThemeWindowFrame (page 174) Deprecated in Mac OS X v10.5
Draws a window frame. (Deprecated. Use `HIThemeDrawWindowFrame` instead.)

DrawThemeWindowHeader (page 176) Deprecated in Mac OS X v10.5
Draws a window header. (Deprecated. Use `HIThemeDrawHeader` instead.)

DrawThemeWindowListViewHeader (page 176) Deprecated in Mac OS X v10.5
  Draws a window list view header. (Deprecated. Use HIThemeDrawHeader instead.)

GetThemeStandaloneGrowBoxBounds (page 183) Deprecated in Mac OS X v10.5
  Obtains the bounds of a size box. (Deprecated. Use HIThemeGetGrowBoxBounds instead.)

GetThemeWindowRegion (page 190) Deprecated in Mac OS X v10.5
  Obtains the specified window region. (Deprecated. Use HIThemeGetWindowShape instead.)

GetThemeWindowRegionHit (page 191) Deprecated in Mac OS X v10.5
  Obtains the part of the window that the user clicked upon. (Deprecated. Use HIThemeGetWindowRegionHit instead.)

Playing Theme Sounds

The theme sound functions do nothing in Mac OS X.

BeginThemeDragSound (page 16)
  Continuously plays a theme-specific sound associated with the user's movement of a given interface object.

EndThemeDragSound (page 17)
  Terminates the playing of a sound associated with the user's movement of a given interface object.

PlayThemeSound (page 26)
  Plays an asynchronous sound associated with the specified state change.

Registering With the Appearance Manager

IsAppearanceClient (page 198) Deprecated in Mac OS X v10.5
  Returns whether a given process is currently registered as a client of the Appearance Manager. (Deprecated. There is no replacement function.)

RegisterAppearanceClient (page 203) Deprecated in Mac OS X v10.5
  Registers your program with the Appearance Manager. (Deprecated. There is no replacement function.)

UnregisterAppearanceClient (page 209) Deprecated in Mac OS X v10.5
  Informs the Appearance Manager that your program is no longer its client. (Deprecated. There is no replacement function.)

Specifying Theme-Compliant Cursors

SetAnimatedThemeCursor (page 27)
  Animates a version of the specified cursor type that is consistent with the current theme.

SetThemeCursor (page 28)
  Sets the cursor to a version of the specified cursor type that is consistent with the current theme.

Using Theme-Compliant Colors and Patterns

DisposeThemeDrawingState (page 17)
  Releases the memory associated with a reference to a graphics port's drawing state.

GetThemeBrushAsColor (page 19)
Obtains the color that corresponds to a given theme brush type under the current theme.

GetThemeDrawingState (page 20)
Obtains the drawing state of the current graphics port.

GetThemeTextColor (page 25)
Obtains the text color used for a specified element under the current theme.

NormalizeThemeDrawingState (page 26)
Sets the current graphics port to a default drawing state.

SetThemeDrawingState (page 29)
Sets the drawing state of the current graphics port.

ApplyThemeBackground (page 145) Deprecated in Mac OS X v10.5
Sets the background color or pattern of the current port to be consistent with that of an embedding object. (Deprecated. Use HIThemeApplyBackground instead.)

GetThemeAccentColors (page 177) Deprecated in Mac OS X v10.5
Obtains a copy of a theme's accent colors. (Deprecated. There is no replacement function.)

IsThemeInColor (page 199) Deprecated in Mac OS X v10.5
Returns whether the current theme would draw in color in the given environment. (Deprecated. There is no replacement function.)

SetThemeBackground (page 205) Deprecated in Mac OS X v10.5
Applies a theme-compliant color or pattern to the background of the current port. (Deprecated. Use HIThemeSetFill and draw using Quartz 2D.)

SetThemePen (page 206) Deprecated in Mac OS X v10.5
Applies a theme-compliant color or pattern to the foreground of the current port. (Deprecated. Use HIThemeSetStroke and draw using Quartz 2D.)

SetThemeTextColor (page 207) Deprecated in Mac OS X v10.5
Sets the current text color to be consistent with that of a specified element. (Deprecated. Use HIThemeSetTextFill and draw with Quartz 2D, ATSUI, or HIThemeDrawTextBox.)

Drawing Theme-Compliant Text

GetThemeTextShadowOutset (page 25)
Tells you the amount of space taken up by the shadow for a given font and drawing state combination.

DrawThemeTextBox (page 169) Deprecated in Mac OS X v10.5
Draws text into the area you specify. (Deprecated. Use HIThemeDrawTextBox instead.)

GetThemeTextDimensions (page 184) Deprecated in Mac OS X v10.5
Tells you the height, width, and baseline for a string. (Deprecated. Use HIThemeGetTextDimensions instead.)

TruncateThemeText (page 208) Deprecated in Mac OS X v10.5
Truncates text to fit within the width you specify. (Deprecated. There is no replacement function; use HIThemeGetTextDimensions or HIThemeDrawTextBox instead.)

Creating and Disposing Universal Procedure Pointers to Appearance Manager Callbacks

`DisposeMenuItemDrawingUPP` (page 146) Deprecated in Mac OS X v10.5
: Disposes of the UPP to a menu item drawing function. (Deprecated. There is no replacement function.)

`DisposeMenuTitleDrawingUPP` (page 146) Deprecated in Mac OS X v10.5
: Disposes of the UPP to a menu title drawing function. (Deprecated. There is no replacement function.)

`DisposeThemeButtonDrawUPP` (page 147) Deprecated in Mac OS X v10.5
: Disposes of the UPP to a button drawing function. (Deprecated. There is no replacement function.)

`DisposeThemeEraseUPP` (page 147) Deprecated in Mac OS X v10.5
: Disposes of the UPP to a background drawing callback function. (Deprecated. There is no replacement function.)

`DisposeThemeIteratorUPP` (page 147) Deprecated in Mac OS X v10.5
: Disposes of the UPP to a theme iteration callback function. (Deprecated. There is no replacement function.)

`DisposeThemeTabTitleDrawUPP` (page 148) Deprecated in Mac OS X v10.5
: Disposes of the UPP to a tab title drawing function. (Deprecated. There is no replacement function.)

`DisposeWindowTitleDrawingUPP` (page 148) Deprecated in Mac OS X v10.5
: Disposes of the UPP to a window title drawing function. (Deprecated. There is no replacement function.)

`InvokeMenuItemDrawingUPP` (page 194) Deprecated in Mac OS X v10.5
: Invokes your menu item drawing function. (Deprecated. There is no replacement function.)

`InvokeMenuTitleDrawingUPP` (page 195) Deprecated in Mac OS X v10.5
: Invokes your menu title drawing function. (Deprecated. There is no replacement function.)

`InvokeThemeButtonDrawUPP` (page 195) Deprecated in Mac OS X v10.5
: Invokes your button drawing function. (Deprecated. There is no replacement function.)

`InvokeThemeEraseUPP` (page 196) Deprecated in Mac OS X v10.5
: Invokes your background drawing callback function. (Deprecated. There is no replacement function.)

`InvokeThemeIteratorUPP` (page 196) Deprecated in Mac OS X v10.5
: Invokes your theme iteration callback function. (Deprecated. There is no replacement function.)

`InvokeThemeTabTitleDrawUPP` (page 197) Deprecated in Mac OS X v10.5
: Invokes your tab title drawing function. (Deprecated. There is no replacement function.)

`InvokeWindowTitleDrawingUPP` (page 197) Deprecated in Mac OS X v10.5
: Invokes your window title drawing function. (Deprecated. There is no replacement function.)

`NewMenuItemDrawingUPP` (page 201) Deprecated in Mac OS X v10.5
: Creates a new universal procedure pointer (UPP) to a menu item drawing function. (Deprecated. There is no replacement function.)

`NewMenuTitleDrawingUPP` (page 201) Deprecated in Mac OS X v10.5
: Creates a new universal procedure pointer (UPP) to a menu title drawing function. (Deprecated. There is no replacement function.)

`NewThemeButtonDrawUPP` (page 201) Deprecated in Mac OS X v10.5
: Creates a new universal procedure pointer (UPP) to a button drawing function. (Deprecated. There is no replacement function.)

`NewThemeEraseUPP` (page 202) Deprecated in Mac OS X v10.5
: Creates a new universal procedure pointer (UPP) to a background drawing callback function. (Deprecated. There is no replacement function.)

`NewThemeIteratorUPP` (page 202) Deprecated in Mac OS X v10.5

Creates a new universal procedure pointer (UPP) to a theme iteration callback function. (Deprecated. There is no replacement function.)

`NewThemeTabTitleDrawUPP` (page 203) Deprecated in Mac OS X v10.5

Creates a new universal procedure pointer (UPP) to a tab title drawing function. (Deprecated. There is no replacement function.)

`NewWindowTitleDrawingUPP` (page 203) Deprecated in Mac OS X v10.5

Creates a new universal procedure pointer (UPP) to a window title drawing function. (Deprecated. There is no replacement function.)

Functions

BeginThemeDragSound

Continuously plays a theme-specific sound associated with the user's movement of a given interface object.

```
OSStatus BeginThemeDragSound (
   ThemeDragSoundKind kind
);
```

Parameters

kind

A value of type `ThemeDragSoundKind`. Pass a constant specifying the sound to play; see "Theme Drag Sounds" (page 135) for descriptions of possible values.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The Appearance Manager automatically plays drag sounds for standard user interface elements and for Drag Manager drag actions. Your application may call `BeginThemeDragSound`, typically upon detecting a drag initiation, to play a drag sound for a custom element. `BeginThemeDragSound` plays the specified sound in a continuous loop until your application calls the function `EndThemeDragSound` (page 17), typically upon receiving a mouse-up event.

Note that the `BeginThemeDragSound` function automatically tracks the current mouse position and handles any panning or variations in pitch for the sound.

Special Considerations

This function is not implemented in Mac OS X.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

`Appearance.h`

CopyThemeIdentifier

Retrieves a string identifying the current theme variant.

```
OSStatus CopyThemeIdentifier (
   CFStringRef *outIdentifier
);
```

Parameters

outIdentifier

A pointer to a string that, on output, contains the current theme variant. When you no longer need the string, you should release it.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Availability

Available in Mac OS X v10.1 and later.

Declared In

`Appearance.h`

DisposeThemeDrawingState

Releases the memory associated with a reference to a graphics port's drawing state.

```
OSStatus DisposeThemeDrawingState (
   ThemeDrawingState inState
);
```

Parameters

inState

A value of type `ThemeDrawingState`. Pass a value specifying the previous drawing state for the current graphics port. You may obtain this value from the `outState` parameter of `GetThemeDrawingState` (page 20).

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Also see the function `SetThemeDrawingState` (page 29).

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

`Appearance.h`

EndThemeDragSound

Terminates the playing of a sound associated with the user's movement of a given interface object.

```
OSStatus EndThemeDragSound (
   void
);
```

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The Appearance Manager automatically starts and stops drag sounds for standard user interface elements and for Drag Manager drag actions. Your application may call `BeginThemeDragSound` (page 16), typically upon detecting a drag initiation, to play a drag sound for a custom element. Call the `EndThemeDragSound` function to turn off a drag sound when the drag is completed, typically upon receipt of a mouse-up event.

Special Considerations

This function is not implemented in Mac OS X.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

`Appearance.h`

GetTheme

Obtains a collection containing data describing the current theme.

```
OSStatus GetTheme (
   Collection ioCollection
);
```

Parameters

`ioCollection`

A value of type `Collection`. Pass a reference to a collection object, such as that created by calling the Collection Manager function `NewCollection`. On return, the collection contains data describing attributes of the current theme.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The `GetTheme` function obtains a collection containing a copy of the data for the current theme. The theme data is in the form of collection items, each corresponding to an attribute of the theme. For a given theme, the actual number of collection items may vary, depending upon how fully the theme's attributes are specified. See "Theme Collection Tags" (page 47) for descriptions of the possible theme collection items.

Your application can use theme collection tags, along with various Collection Manager functions, to access the data in the collection.

Also see the function `SetTheme` (page 204).

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
`Appearance.h`

GetThemeBrushAsColor

Obtains the color that corresponds to a given theme brush type under the current theme.

```
OSStatus GetThemeBrushAsColor (
   ThemeBrush inBrush,
   SInt16 inDepth,
   Boolean inColorDev,
   RGBColor *outColor
);
```

Parameters

`inBrush`

A value of type `ThemeBrush`. Pass a constant specifying the theme brush type for which you wish to obtain a color; see "Theme Brushes" (page 70) for descriptions of possible values.

`inDepth`

A signed 16-bit integer. Pass a value specifying the bit depth (in bits per pixel) of the current graphics port.

`inColorDev`

A value of type `Boolean`. Pass `true` to indicate that you are drawing on a color device. Pass `false` for a monochrome device.

`outColor`

A pointer to a structure of type `RGBColor`. On return, the structure contains a color corresponding to the color or pattern used by the specified theme brush under the current theme.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
The `GetThemeBrushAsColor` function obtains a color that corresponds to that which is in use for a specified theme brush. If, in the current theme, the specified brush draws with a pattern instead of a color, a theme-specified approximate color is obtained. Your application should call `GetThemeBrushAsColor` only when you must use an `RGBColor` value for a specific operation; typically, your application should call the functions `SetThemeBackground` (page 205) and `SetThemePen` (page 206) for greatest fidelity with the current theme.

Availability
Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
`Appearance.h`

GetThemeCheckBoxStyle

Obtains the system preference for the type of mark to use in a checkbox.

```
OSStatus GetThemeCheckBoxStyle (
   ThemeCheckBoxStyle *outStyle
);
```

Parameters

outStyle

A pointer to a value of type ThemeCheckBoxStyle. On return, the value specifies the type of mark being used. See "Theme Checkbox Styles" (page 86) for descriptions of possible values.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Because international systems may specify the use of one type of mark to use in checkboxes over another, your application should call GetThemeCheckBoxStyle to obtain the correct type of mark to use on the current system.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

Appearance.h

GetThemeDrawingState

Obtains the drawing state of the current graphics port.

```
OSStatus GetThemeDrawingState (
   ThemeDrawingState *outState
);
```

Parameters

outState

A pointer to a value of type ThemeDrawingState. On return, GetThemeDrawingState sets the outState parameter to point to a copy of the drawing state for the current graphics port.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Your application may call the GetThemeDrawingState function before performing an operation that modifies the drawing state of a graphics port. To return the graphics port to its previous drawing state and release the memory allocated for the drawing state reference, your application should call SetThemeDrawingState (page 29), providing the reference obtained in the outState parameter of GetThemeDrawingState. You can also call DisposeThemeDrawingState (page 17) to release the allocated memory.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

Appearance.h

GetThemeMenuBarHeight

Obtains the height of a menu bar.

```
OSStatus GetThemeMenuBarHeight (
   SInt16 *outHeight
);
```

Parameters

outHeight

A pointer to a signed 16-bit integer. On return, the integer value represents the height (in pixels) of the menu bar.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The GetThemeMenuBarHeight function obtains the specified height of a menu bar in the current theme. This is in contrast to the Menu Manager function GetMBarHeight, which obtains the actual space that the menu bar is currently occupying on the screen. In most instances, the values produced by these two functions are the same. But, when the menu bar is hidden, GetMBarHeight produces a value of 0, and GetThemeMenuBarHeight still provides the "ideal" menu bar height.

Special Considerations

Because menu bar heights may vary among appearances by one or more pixels, you should check the current menu bar height after a theme switch. Specifically, your application should respond to the theme-switch Apple event, kAEAppearanceChanged, by checking the current menu bar height. See "Appearance Manager Apple Events" (page 46) for more details on kAEAppearanceChanged.

It is important to check the menu bar height before positioning any windows. Failure to do so may result in the menu bar overlapping your application's windows.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

Appearance.h

GetThemeMenuItemExtra

Obtains a measurement of the space surrounding a menu item.

```
OSStatus GetThemeMenuItemExtra (
   ThemeMenuItemType inItemType,
   SInt16 *outHeight,
   SInt16 *outWidth
);
```

Parameters

inItemType

A value of type ThemeMenuItemType. Pass a constant identifying the type of menu item for which you are interested in getting a measurement. See "Theme Menu Item Types" (page 102).

outHeight

A pointer to a signed 16-bit integer. On return, the integer value represents the total amount of padding between the content of the menu item and the top and bottom of its frame (in pixels). Your content's height plus the measurement provided by the `outHeight` parameter equals the total item height.

outWidth

A pointer to a signed 16-bit integer. On return, the integer value represents the total amount of padding between the content of the menu item and the left and right limits of the menu (in pixels). Your content's width plus the measurement provided by the `outWidth` parameter equals the total item width.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Your application should call the `GetThemeMenuItemExtra` function when you are writing your own menu definition function and wish to be theme-compliant. Once you have determined the height and width of the content of a menu item, call `GetThemeMenuItemExtra` to get a measurement in pixels of the space surrounding a menu item, including any necessary inter-item spacing, in the current theme. By combining the values for your menu item's content and the extra padding needed by the theme, you can derive the size of the rectangle needed to encompass both the content and the theme element together.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Appearance.h`

GetThemeMenuSeparatorHeight

Obtains the height of a menu separator line.

```
OSStatus GetThemeMenuSeparatorHeight (
   SInt16 *outHeight
);
```

Parameters

outHeight

A pointer to a signed 16-bit integer. On return, the integer value represents the height (in pixels) of the menu separator line.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The `GetThemeMenuSeparatorHeight` function obtains the height of a menu separator line under the current theme. Your application should call the `GetThemeMenuSeparatorHeight` function when you are writing your own menu definition function and wish to calculate a menu rectangle for a separator to match the current theme.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Appearance.h`

GetThemeMenuTitleExtra

Obtains a measurement of the space to either side of a menu title.

```
OSStatus GetThemeMenuTitleExtra (
   SInt16 *outWidth,
   Boolean inIsSquished
);
```

Parameters

outWidth

A pointer to a signed 16-bit integer. On return, the integer value represents the horizontal distance (in pixels) between the menu title and the bounds of its containing rectangle.

inIsSquished

A value of type `Boolean`. If all the titles do not fit in the menu bar and you wish to condense the menu title's spacing to fit, pass `true`. If you pass `false`, the menu title is not condensed.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Once you have determined the height and width of the content of a menu title, call `GetThemeMenuTitleExtra` to get the space surrounding the menu title in the current theme.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Appearance.h`

GetThemeMetric

Retrieves the value of a metric property of a user interface element.

```
OSStatus GetThemeMetric (
   ThemeMetric inMetric,
   SInt32 *outMetric
);
```

Parameters

inMetric

The user interface metric to retrieve. See "Theme Metrics" (page 53) for a list of possible metrics.

outMetric

A pointer to an integer value. On output, contains the value of the specified metric property, generally in points.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Appearance.h`

GetThemeScrollBarArrowStyle

Obtains the system preference for the type of scroll bar arrows to be used.

```
OSStatus GetThemeScrollBarArrowStyle (
   ThemeScrollBarArrowStyle *outStyle
);
```

Parameters

outStyle

A pointer to a value of type `ThemeScrollBarArrowStyle`. On return, the value specifies the type of scroll bar arrows being used. See "Theme Scroll Bar Arrow Styles" (page 104) for descriptions of possible values.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Because the user can specify varying types of scroll bar arrows on a theme-specific basis, your application should call `GetThemeScrollBarArrowStyle` to obtain the preferred style under the current theme.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

`Appearance.h`

GetThemeScrollBarThumbStyle

Obtains the system preference for the type of scroll box to be used.

```
OSStatus GetThemeScrollBarThumbStyle (
   ThemeScrollBarThumbStyle *outStyle
);
```

Parameters

outStyle

A pointer to a value of type `ThemeScrollBarThumbStyle`. On return, the value specifies the type of scroll box being used. See "Theme Scroll Box Styles" (page 105) for descriptions of possible values.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Because the user can specify either proportional or fixed-size scroll boxes (also known as "scroll indicators" or "thumbs") on a theme-specific basis, your application should call `GetThemeScrollBarThumbStyle` to obtain the preferred style under the current theme.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

`Appearance.h`

GetThemeTextColor

Obtains the text color used for a specified element under the current theme.

```
OSStatus GetThemeTextColor (
   ThemeTextColor inColor,
   SInt16 inDepth,
   Boolean inColorDev,
   RGBColor *outColor
);
```

Parameters

inColor

A value of type `ThemeTextColor`. Pass a constant specifying the element for which you wish to obtain the current text color; see "Theme Text Colors" (page 94) for descriptions of possible values.

inDepth

A signed 16-bit integer. Pass a value specifying the bit depth (in bits per pixel) of the current graphics port.

inColorDev

A value of type `Boolean`. Pass `true` to indicate that you are drawing on a color device. Pass `false` for a monochrome device.

outColor

A pointer to a structure of type `RGBColor`. On return, the structure contains the text color used for the specified element under the current theme.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Also see the function `SetThemeTextColor` (page 207).

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

`Appearance.h`

GetThemeTextShadowOutset

Tells you the amount of space taken up by the shadow for a given font and drawing state combination.

```
OSStatus GetThemeTextShadowOutset (
   ThemeFontID inFontID,
   ThemeDrawState inState,
   Rect *outOutset
);
```

Parameters

inFontID

The `ThemeFontID` describing the font you'd like the shadow characteristics of. Font and drawing state both determine the amount of shadow that will be used on rendered text. See "Theme Font IDs" (page 91) for the values you can use here.

inState

The `ThemeDrawState` which matches the drawing state you'd like the shadow characteristics of. Font and state both determine the amount of shadow that will be used on rendered text. See "Theme Drawing States" (page 51) for the values you can use here.

outOutset

On output, this parameter contains the amount of space the shadow will take up beyond each edge of the text bounding rectangle returned by `GetThemeTextDimensions`. The fields of this parameter will either be positive values or zero.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
`GetThemeTextShadowOutset` passes back the maximum amount of space the shadow will take up for text drawn in the specified font and state. While `GetThemeTextDimensions` tells you how much space is taken up by the character glyphs themselves, it does not incorporate the font or state shadow into its calculations. If you need to know how much total space including the shadow will be taken up, call `GetThemeTextDimensions` followed by `GetThemeTextShadowOutset`.

Availability
Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
`Appearance.h`

NormalizeThemeDrawingState

Sets the current graphics port to a default drawing state.

```
OSStatus NormalizeThemeDrawingState (
   void
);
```

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
The `NormalizeThemeDrawingState` function sets the background of a graphics port to white; the pen of the port to a size of 1 pixel by 1 pixel, a pattern mode of `patCopy`, and a pattern of black; and the text mode of the port to `srcOr`. `NormalizeThemeDrawingState` also flushes from memory any color foreground or background patterns saved in the port's `GrafPort.pnPat` or `GrafPort.bkPat` fields, respectively.

Availability
Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
`Appearance.h`

PlayThemeSound

Plays an asynchronous sound associated with the specified state change.

```
OSStatus PlayThemeSound (
   ThemeSoundKind kind
);
```

Parameters

kind

A value of type ThemeSoundKind. Pass a constant specifying the sound to play; see "Theme Sounds" (page 119) for descriptions of possible values.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The Appearance Manager automatically plays theme sounds for standard user interface elements. Your application can call the PlayThemeSound function to play a theme sound for a custom element. The sound plays asynchronously until complete, stopping automatically.

Special Considerations

This function is not implemented in Mac OS X.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

Appearance.h

SetAnimatedThemeCursor

Animates a version of the specified cursor type that is consistent with the current theme.

```
OSStatus SetAnimatedThemeCursor (
   ThemeCursor inCursor,
   UInt32 inAnimationStep
);
```

Parameters

inCursor

A value of type ThemeCursor. Pass a constant specifying the type of cursor to set; see "Theme Cursors" (page 87) for a description of the possible values. Note that only cursors designated as able to be animated should be used for this function. If you specify an unanimatable cursor type, SetAnimatedThemeCursor returns the error themeBadCursorIndexErr (–30565).

inAnimationStep

An unsigned 32-bit value. Pass a value specifying the current animation step of the cursor. To animate the cursor, increment the value by 1 with each call to SetAnimatedThemeCursor.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Appearance Manager 1.1 introduces cursors that can change appearance with a theme change. In order to be theme-compliant, your program should use these theme-specific cursors whenever possible, instead of the classic black-and-white cursors.

Your application should call the SetAnimatedThemeCursor function to ensure that its animated cursors are theme-compliant, rather than using any QuickDraw cursor utilities functions such as SetCursor, SetCCursor, SpinCursor, or RotateCursor. If you wish a non-animated cursor to be theme-compliant, call the function SetThemeCursor (page 28).

Because these are color cursors, they currently cannot be set from interrupt time. Therefore, if you support animated cursors that are changed at interrupt time you should continue to use your own cursors for now.

Special Considerations

Do not call SetAnimatedThemeCursor at interrupt time.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

Appearance.h

SetThemeCursor

Sets the cursor to a version of the specified cursor type that is consistent with the current theme.

```
OSStatus SetThemeCursor (
   ThemeCursor inCursor
);
```

Parameters

inCursor

A value of type ThemeCursor. Pass a constant specifying the type of cursor to set; see "Theme Cursors" (page 87) for a description of possible values.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Appearance Manager 1.1 introduces cursors that can change appearance with a theme change. In order to be theme-compliant, your program should use these theme-specific cursors whenever possible, instead of the classic black-and-white cursors. Because these are color cursors, they currently cannot be set from interrupt time.

Your application should call the SetThemeCursor function to ensure that its cursors are theme-compliant, rather than the QuickDraw cursor utilities functions SetCursor or SetCCursor. If you wish an animatable cursor to be theme-compliant, call the function SetAnimatedThemeCursor (page 27).

Special Considerations

Do not call SetThemeCursor at interrupt time.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Related Sample Code

CarbonCocoa_PictureCursor

CarbonSketch

QTCarbonShell

Declared In
Appearance.h

SetThemeDrawingState

Sets the drawing state of the current graphics port.

```
OSStatus SetThemeDrawingState (
   ThemeDrawingState inState,
   Boolean inDisposeNow
);
```

Parameters

inState

A value of type ThemeDrawingState. Pass a ThemeDrawingState value such as that produced in the outState parameter of GetThemeDrawingState (page 20).

inDisposeNow

A value of type Boolean. Pass a value of true to release the memory allocated for the drawing state reference. Pass false if you wish to continue using the drawing state and do not want to dispose of the memory at this time; you must call DisposeThemeDrawingState (page 17) to dispose of the memory any time before your application terminates.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
Your application can save the port state by calling the function GetThemeDrawingState (page 20) and restore the port state by calling the function SetThemeDrawingState, supplying the value obtained in the outState parameter of GetThemeDrawingState, after you have completed all of your drawing.

Availability
Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
Appearance.h

Callbacks

MenuItemDrawingProcPtr

Draws a menu item.

```
typedef void ( *MenuItemDrawingProcPtr)
(
    const Rect * inBounds,
    SInt16 inDepth,
    Boolean inIsColorDevice,
    SInt32 inUserData
);
```

If you name your function MyMenuItemDrawingCallback, you would declare it like this:

```
void MyMenuItemDrawingCallback (
    const Rect * inBounds,
    SInt16 inDepth,
    Boolean inIsColorDevice,
    SInt32 inUserData
);
```

Parameters

inBounds

A pointer to a structure of type Rect. You are passed a rectangle specifying the dimensions and position in which you should draw your menu item content. Your menu item drawing function is called clipped to the rectangle in which you are allowed to draw your content; do not draw outside this region.

inDepth

A signed 16-bit integer. You are passed the bit depth (in bits per pixel) of the current graphics port.

inIsColorDevice

A value of type Boolean. You are passed true to indicate that you are drawing on a color device; inIsColorDevice is false for a monochrome device.

inUserData

You are passed data specifying how to draw the menu item content from the inUserData parameter of DrawThemeMenuItem (page 156).

Discussion

At the time your menu item drawing function is called, the foreground text color and mode is already set to draw in the correct state (enabled, selected, disabled) and correct color for the theme. You do not need to set the color unless you have special drawing needs. If you do have special drawing needs, you should supply the inDepth value and the value of the inIsColorDevice parameter to the function IsThemeInColor to determine whether or not you should draw the menu item content in color.

Note that the Appearance Manager calls your MyMenuItemDrawingCallback function for every device that the inBounds rectangle intersects.

You should refer to your MyMenuItemDrawingCallback function using a MenuItemDrawingUPP, which you can create with NewMenuItemDrawingUPP.

You typically use the NewMenuItemDrawingUPP function like this:

```
MenuItemDrawingUPP myMenuItemDrawingUPP;
myMenuItemDrawingUPP = NewMenuItemDrawingUPP(MyMenuItemDrawingCallback);
```

Special Considerations

The Appearance Manager draws the background of the menu item prior to calling your menu item drawing function, so you should not erase the item's background from this function.

Version Notes

This function is available with Appearance Manager 1.0.1 and later.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Appearance.h`

MenuTitleDrawingProcPtr

Draws a menu title.

```
typedef void (*MenuTitleDrawingProcPtr)
(
    const Rect * inBounds,
    SInt16 inDepth,
    Boolean inIsColorDevice,
    SInt32 inUserData
);
```

If you name your function `MyMenuTitleDrawingCallback`, you would declare it like this:

```
void MyMenuTitleDrawingCallback (
    const Rect * inBounds,
    SInt16 inDepth,
    Boolean inIsColorDevice,
    SInt32 inUserData
);
```

Parameters

inBounds

A pointer to a structure of type `Rect`. You are passed a rectangle specifying the dimensions and position in which you should draw your menu title content. Your menu title drawing function is called clipped to the rectangle in which you are allowed to draw your content; do not draw outside this region.

inDepth

A signed 16-bit integer. You are passed the bit depth (in bits per pixel) of the current graphics port.

inIsColorDevice

A value of type `Boolean`. You are passed `true` to indicate that you are drawing on a color device; `inIsColorDevice` is `false` for a monochrome device.

inUserData

You are passed data specifying how to draw the menu title content from the `inTitleData` parameter of `DrawThemeMenuTitle` (page 158).

Discussion

At the time your menu title drawing function is called, the foreground text color and mode is already set to draw in the correct state (enabled, selected, disabled) and correct color for the theme. You do not need to set the color unless you have special drawing needs. If you do have special drawing needs, you should supply the `inDepth` value and the value of the `inIsColorDevice` parameter to the function `IsThemeInColor` to determine whether or not you should draw the menu title content in color.

Note that the Appearance Manager calls your MyMenuTitleDrawingCallback function for every device that the inBounds rectangle intersects.

You should refer to your MyMenuTitleDrawingCallback function using a MenuTitleDrawingUPP, which you can create with the NewMenuTitleDrawingUPP function.

You typically use the NewMenuTitleDrawingUPP function like this:

```
MenuTitleDrawingUPP myMenuTitleDrawingUPP;
myMenuTitleDrawingUPP = NewMenuTitleDrawingUPP(MyMenuTitleDrawingCallback);
```

Special Considerations

The Appearance Manager draws the background of the menu title prior to calling your menu title drawing function, so you should not erase the title's background from this function.

Version Notes

This function is available with Appearance Manager 1.0.1 and later.

Availability

Available in Mac OS X v10.0 and later.

Declared In

Appearance.h

ThemeButtonDrawProcPtr

Draws a button label.

```
typedef void (*ThemeButtonDrawProcPtr)
(
    const Rect * bounds,
    ThemeButtonKind kind,
    const ThemeButtonDrawInfo * info,
    UInt32 userData,
    SInt16 depth,
    Boolean isColorDev
);
```

If you name your function MyThemeButtonDrawCallback, you would declare it like this:

```
void MyThemeButtonDrawCallback (
    const Rect * bounds,
    ThemeButtonKind kind,
    const ThemeButtonDrawInfo * info,
    UInt32 userData,
    SInt16 depth,
    Boolean isColorDev
);
```

Parameters

bounds

A pointer to a structure of type Rect. The rectangle you are passed is set to the area in which you should draw your content. Your button label drawing function is called clipped to the rectangle in which you are allowed to draw your content; do not draw outside this region. Note that if a right-to-left adornment is specified in the ThemeButtonDrawInfo structure passed into the info parameter, you may need to accommodate this orientation when placing your content.

kind

A value of type ThemeButtonKind. You are passed a constant specifying the button type. See "Theme Buttons" (page 78) for descriptions of possible values.

info

A pointer to a structure of type ThemeButtonDrawInfo (page 40). The structure is set to contain the current state, value, and adornment for the button.

userData

An unsigned 32-bit value. You are passed data specifying how to draw the content, from the inUserData parameter of DrawThemeButton (page 149).

depth

A signed 16-bit value. You are passed the bit depth (in bits per pixel) of the current graphics port.

isColorDev

A value of type Boolean. If true, indicates that you are drawing on a color device; a value of false indicates a monochrome device.

Discussion

At the time your button label drawing function is called, the foreground text color and mode is already set to draw in the correct state (active or inactive) and correct color for the theme. You do not need to set the color unless you have special drawing needs. If you do have special drawing needs, you should supply the depth value and the value of the isColorDevice parameter to the function IsThemeInColor to determine whether or not you should draw your content in color. Note that the Appearance Manager calls your MyThemeButtonDrawCallback function for every device that the bounds rectangle intersects.

You should refer to your MyThemeButtonDrawCallback function using a ThemeButtonDrawUPP, which you can create with the NewThemeButtonDrawUPP function.

You typically use the NewThemeButtonDrawUPP function like this:

```
ThemeButtonDrawUPP myThemeButtonDrawUPP;
myThemeButtonDrawUPP = NewThemeButtonDrawUPP(MyThemeButtonDrawCallback);
```

Special Considerations

The Appearance Manager draws the button background prior to calling your button label drawing function, so you should not erase the button background from your label drawing function.

Version Notes

This function is available with Appearance Manager 1.1 and later.

Availability

Available in Mac OS X v10.0 and later.

Declared In

Appearance.h

ThemeEraseProcPtr

Draws a background.

```
typedef void (*ThemeEraseProcPtr) (
    const Rect * bounds,
    UInt32 eraseData,
    SInt16 depth,
    Boolean isColorDev
);
```

If you name your function `MyThemeEraseCallback`, you would declare it like this:

```
void MyThemeEraseCallback (
    const Rect * bounds,
    UInt32 eraseData,
    SInt16 depth,
    Boolean isColorDev
);
```

Parameters

bounds

A pointer to a structure of type `Rect`. The rectangle you are passed is set to the area in which you should draw. Your drawing function is called clipped to the rectangle in which you are allowed to draw; do not draw outside this region.

eraseData

An unsigned 32-bit value. You are passed data specifying how to draw, from the `eraseData` parameter of `DrawThemeChasingArrows` (page 150), `DrawThemePopupArrow` (page 160), `DrawThemeTrack` (page 172), or `DrawThemeTrackTickMarks` (page 173) or from the `inUserData` parameter of `DrawThemeButton` (page 149).

depth

A signed 16-bit value. You are passed the bit depth (in bits per pixel) of the current graphics port.

isColorDev

A value of type `Boolean`. If `true`, indicates that you are drawing on a color device; a value of `false` indicates a monochrome device.

Discussion

At the time your drawing function is called, the foreground text color and mode is already set to draw in the correct state (active or inactive) and correct color for the theme. You do not need to set the color unless you have special drawing needs. If you do have special drawing needs, you should supply the `depth` value and the value of the `isColorDevice` parameter to the function `IsThemeInColor` to determine whether or not you should draw in color. Note that the Appearance Manager calls your `MyThemeEraseCallback` function for every device that the `bounds` rectangle intersects, so your application does not need to call the `DeviceLoop` function itself.

You should refer to your `MyThemeEraseCallback` function using a `ThemeEraseUPP`, which you can create with the `NewThemeEraseUPP` function.

You typically use the `NewThemeEraseUPP` function like this:

```
ThemeEraseUPP myThemeEraseUPP;
myThemeEraseUPP = NewThemeEraseUPP(MyThemeEraseCallback);
```

Version Notes
This function is available with Appearance Manager 1.1 and later.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Appearance.h`

ThemeIteratorProcPtr

Performs a custom response to an iteration over themes installed on a system.

```
typedef Boolean (*ThemeIteratorProcPtr)
(
    ConstStr255Param inFileName,
    SInt16 resID,
    Collection inThemeSettings,
    void * inUserData
);
```

If you name your function `MyThemeIteratorCallback`, you would declare it like this:

```
Boolean MyThemeIteratorCallback (
    ConstStr255Param inFileName,
    SInt16 resID,
    Collection inThemeSettings,
    void * inUserData
);
```

Parameters

inFileName
A value of type `ConstStr255Param`. You are passed the name of the file containing the theme being iterated upon.

resID
A signed 16-bit integer. You are passed the resource ID of the theme.

inThemeSettings
A value of type `Collection`. You are passed a reference to a collection that contains data describing attributes of the theme. Note that the Appearance Manager owns this collection, and that your application should not dispose of it.

inUserData
A pointer to data of any type. You are passed the value specified in the `inUserData` parameter of the function `IterateThemes` (page 200).

Return Value
A value of type `Boolean`. If you return `true`, `IterateThemes` continues iterating. Set to `false` to terminate the iteration.

Discussion
You should refer to your `MyThemeIteratorCallback` function using a `ThemeIteratorUPP`, which you can create using the `NewThemeIteratorUPP` function.

You typically use the `NewThemeIteratorUPP` function like this:

```
ThemeIteratorUPP myThemeIteratorUPP;
myThemeIteratorUPP = NewThemeIteratorUPP(MyThemeIteratorCallback);
```

Special Considerations

Your application should not open and close theme files during this call.

Version Notes

This function is available with Appearance Manager 1.1 and later.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Appearance.h`

ThemeTabTitleDrawProcPtr

Draws a tab title.

```
typedef void (*ThemeTabTitleDrawProcPtr)
(
    const Rect * bounds,
    ThemeTabStyle style,
    ThemeTabDirection direction,
    SInt16 depth,
    Boolean isColorDev,
    UInt32 userData
);
```

If you name your function `MyThemeTabTitleDrawCallback`, you would declare it like this:

```
void MyThemeTabTitleDrawCallback (
    const Rect * bounds,
    ThemeTabStyle style,
    ThemeTabDirection direction,
    SInt16 depth,
    Boolean isColorDev,
    UInt32 userData
);
```

Parameters

bounds

A pointer to a structure of type `Rect`. The rectangle you are passed is set to the area in which you should draw your tab title content. Your tab title drawing function is called clipped to the rectangle in which you are allowed to draw your content; do not draw outside this region.

style

A value of type `ThemeTabStyle`. You are passed a constant specifying the relative position (front or non-front) and state of the tab. See "Theme Tab Styles" (page 108) for descriptions of possible values.

direction

A value of type `ThemeTabDirection`. You are passed a constant specifying the direction in which the tab is oriented. See "Theme Tab Directions" (page 107) for descriptions of possible values.

depth

A signed 16-bit value. You are passed the bit depth (in bits per pixel) of the current graphics port.

isColorDev

A value of type `Boolean`. If `true`, indicates that you are drawing on a color device; a value of `false` indicates a monochrome device.

userData

An unsigned 32-bit value. You are passed data specifying how to draw the tab title content, from the `userData` parameter of `DrawThemeTab` (page 167).

Discussion

At the time your tab title drawing function is called, the foreground text color and mode is already set to draw in the correct state (active or inactive) and correct color for the theme. You do not need to set the color unless you have special drawing needs. If you do have special drawing needs, you should supply the `depth` value and the value of the `isColorDevice` parameter to the function `IsThemeInColor` to determine whether or not you should draw the tab title content in color. Note that the Appearance Manager calls your `MyThemeTabTitleDrawCallback` function for every device that the `bounds` rectangle intersects.

You should refer to your `MyThemeTabTitleDrawCallback` function using a `ThemeTabTitleDrawUPP`, which you can create with the `NewThemeTabTitleDrawUPP` function.

You typically use the `NewThemeTabTitleDrawUPP` function like this:

```
ThemeTabTitleDrawUPP myThemeTabTitleDrawUPP;
myThemeTabTitleDrawUPP = NewThemeTabTitleDrawUPP(MyThemeTabTitleDrawCallback);
```

Special Considerations

The Appearance Manager draws the tab background prior to calling your tab title drawing function, so you should not erase the tab background from your title drawing function.

Version Notes

This function is available with Appearance Manager 1.1 and later.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Appearance.h`

WindowTitleDrawingProcPtr

Draws a window title.

```
typedef void (*WindowTitleDrawingProcPtr)
(
    const Rect * bounds,
    SInt16 depth,
    Boolean colorDevice,
    UInt32 userData
);
```

If you name your function `MyWindowTitleDrawingCallback`, you would declare it like this:

```
void MyWindowTitleDrawingCallback (
    const Rect * bounds,
    SInt16 depth,
    Boolean colorDevice,
    UInt32 userData
);
```

Parameters

bounds

A pointer to a structure of type Rect. The rectangle you are passed is set to the area in which you should draw your window title content. Your window title drawing function is called clipped to the rectangle in which you are allowed to draw your content; do not draw outside this region.

depth

A signed 16-bit value. You are passed the bit depth (in bits per pixel) of the current graphics port.

colorDevice

A value of type Boolean. If true, indicates that you are drawing on a color device; a value of false indicates a monochrome device.

userData

A signed 32-bit value. You are passed data specifying how to draw the window title content, from the titleData parameter of DrawThemeWindowFrame (page 174).

Discussion

At the time your window title drawing function is called, the foreground text color and mode is already set to draw in the correct window state (active or inactive) and correct color for the theme. You do not need to set the color unless you have special drawing needs. If you do have special drawing needs, you should supply the depth value and the value of the colorDevice parameter to the function IsThemeInColor to determine whether or not you should draw the window title content in color. Note that the Appearance Manager calls your MyWindowTitleDrawingCallback function for every device that the bounds rectangle intersects.

You should refer to your MyWindowTitleDrawingCallback function using a WindowTitleDrawingUPP, which you can create with the NewWindowTitleDrawingUPP function.

You typically use the NewWindowTitleDrawingUPP function like this:

```
WindowTitleDrawingUPP myWindowTitleDrawingUPP;
myWindowTitleDrawingUPP = NewWindowTitleDrawingUPP(MyWindowTitleDrawingCallback);
```

Special Considerations

The Appearance Manager draws the background of the window title prior to calling your window title drawing function, so you should not erase the background from this function.

Version Notes

This function is available with Appearance Manager 1.1 and later.

Availability

Available in Mac OS X v10.0 and later.

Declared In

Appearance.h

Data Types

ProgressTrackInfo

Describes the progress bar–specific features of a given track control.

```
struct ProgressTrackInfo {
    UInt8 phase;
};
typedef struct ProgressTrackInfo ProgressTrackInfo;
```

Fields

`phase`

A value specifying the current animation phase for an indeterminate progress bar. You can pass any value of type `UInt8`. Increment this value to animate the progress bar. Set this field to 0 for a determinate progress bar.

Discussion

Your application supplies a `ProgressTrackInfo` structure in the `ThemeTrackDrawInfo` (page 41) structure.

Version Notes

The `ProgressTrackInfo` structure is available with Appearance Manager 1.1 and later.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Appearance.h`

ScrollBarTrackInfo

Describes the scroll bar–specific features of a given track control.

```
struct ScrollBarTrackInfo {
    SInt32 viewsize;
    ThemeTrackPressState pressState;
};
typedef struct ScrollBarTrackInfo ScrollBarTrackInfo;
```

Fields

`viewsize`

A signed 32-bit integer, specifying the size of the content being displayed. This value should be expressed in terms of the same units of measurement as are used for the minimum, maximum, and current settings of the scroll bar.

`pressState`

A value of type `ThemeTrackPressState`, specifying what in the scroll bar is currently pressed. See "Theme Track Press States" (page 112) for descriptions of possible values. Pass 0 if nothing is currently pressed.

Discussion

Your application uses the `ScrollBarTrackInfo` structure in the `ThemeTrackDrawInfo` (page 41) structure.

Version Notes

The `ScrollBarTrackInfo` structure is available with Appearance Manager 1.1 and later.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Appearance.h`

SliderTrackInfo

Describes the slider-specific features of a given track control.

```
struct SliderTrackInfo {
    ThemeThumbDirection thumbDir;
    ThemeTrackPressState pressState;
};
typedef struct SliderTrackInfo SliderTrackInfo;
```

Fields

`thumbDir`

A value of type `ThemeThumbDirection`, specifying the direction in which the slider indicator points. See "Theme Thumb Directions" (page 106) for descriptions of possible values.

`pressState`

A value of type `ThemeTrackPressState`, specifying the part of the slider that is currently pressed. See "Theme Track Press States" (page 112) for descriptions of possible values. Pass 0 if nothing is currently pressed.

Discussion
Your application supplies a `SliderTrackInfo` structure to the `ThemeTrackDrawInfo` (page 41) structure.

Version Notes
The `SliderTrackInfo` structure is available with Appearance Manager 1.1 and later.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Appearance.h`

ThemeButtonDrawInfo

Describes the changeable visual characteristics of a button.

```
struct ThemeButtonDrawInfo {
    ThemeDrawState state;
    ThemeButtonValue value;
    ThemeButtonAdornment adornment;
};
typedef struct ThemeButtonDrawInfo ThemeButtonDrawInfo;
```

Fields

`state`

A value of type `ThemeDrawState`, specifying the state of the button, such as whether it is active, inactive, or pressed. See "Theme Drawing States" (page 51) for descriptions of possible values.

`value`

A value of type `ThemeButtonValue`, specifying the value of the button, such as, in the case of checkbox, whether it is drawn as on, off, or mixed. See "Theme Button Values" (page 84) for descriptions of possible values.

`adornment`

A value of type `ThemeButtonAdornment`, specifying any supplementary characteristics of the button, such as whether it is drawn with a focus ring. See "Theme Button Adornments" (page 81) for descriptions of possible values.

Discussion

Your application can use a `ThemeButtonDrawInfo` structure, together with a constant of type `ThemeButtonKind`, to fully describe the visual characteristics of a given button type at a given point in time. See "Theme Buttons" (page 78) for a description of `ThemeButtonKind` values.

Your application uses the `ThemeButtonDrawInfo` structure in the function `DrawThemeButton` (page 149) to draw a theme-compliant button and in the functions `GetThemeButtonRegion` (page 179) and `GetThemeButtonContentBounds` (page 178) to obtain information about a specific button type.

Version Notes

The `ThemeButtonDrawInfo` structure is available with Appearance Manager 1.1 and later.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Appearance.h`

ThemeTrackDrawInfo

Describes a track control.

```
struct ThemeTrackDrawInfo {
    ThemeTrackKind kind
    Rect bounds
    SInt32 min
    SInt32 max
    SInt32 value
    UInt32 reserved
    ThemeTrackAttributes attributes
    ThemeTrackEnableState enableState
    UInt8 filler1
    union {
        ScrollBarTrackInfo scrollbar;
        SliderTrackInfo slider;
        ProgressTrackInfo progress;
    } trackInfo;
};
typedef struct ThemeTrackDrawInfo ThemeTrackDrawInfo;
```

Fields

`kind`

A value of type `ThemeTrackKind`, specifying the type of track to be drawn. See "Theme Track Kinds" (page 111) for descriptions of possible values.

`bounds`
: A structure of type `Rect` specifying the dimensions and position of the track, in local coordinates.

`min`
: A signed 32-bit integer specifying the minimum value for the track.

`max`
: A signed 32-bit integer specifying the maximum value for the track.

`value`
: A signed 32-bit integer specifying the current value for the track.

`reserved`
: Reserved.

`attributes`
: A value of type `ThemeTrackAttributes` specifying additional attributes of the track, such as whether the track has an indicator. See "Theme Track Attributes" (page 109) for descriptions of possible values.

`enableState`
: A value of type `ThemeTrackEnableState` specifying the current state of the track control; see "Theme Track States" (page 110) for descriptions of possible values.

`filler1`

`trackInfo`
: A union of the `ScrollBarTrackInfo`, `SliderTrackInfo`, and `ProgressTrackInfo` structures. Your application fills in the structure that is appropriate for the kind of track with which you are working. See `ScrollBarTrackInfo` (page 39), `SliderTrackInfo` (page 40), and `ProgressTrackInfo` (page 39) for details on these structures.

Discussion

Your application fills out the applicable fields of a `ThemeTrackDrawInfo` structure to fully describe any given track control.

Version Notes

The `ThemeTrackDrawInfo` structure is available with Appearance Manager 1.1 and later.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Appearance.h`

ThemeWindowMetrics

Describes the dimensions of the parts of a window.

```
struct ThemeWindowMetrics {
    UInt16 metricSize;
    SInt16 titleHeight;
    SInt16 titleWidth;
    SInt16 popupTabOffset;
    SInt16 popupTabWidth;
    UInt16 popupTabPosition;
};
typedef struct ThemeWindowMetrics ThemeWindowMetrics;
```

Fields

`metricSize`
: A value specifying the size of the `ThemeWindowMetrics` structure.

`titleHeight`
: A measurement in pixels of the height of the title text in the current system font, including any icon that may be present in the title region. Set this field to 0 if the window does not contain a title.

`titleWidth`
: A measurement in pixels of the width of the title text in the current system font, including any icon that may be present in the title region. Set this field to 0 if the window does not contain a title.

`popupTabOffset`
: A measurement in pixels of the distance that the left edge of a pop-up window's tab is offset from the left edge of the window. This value is used in conjunction with the value passed in the `popupTabPosition` field to determine the actual position of the tab. Set this field to 0 if the window is not a pop-up window.

`popupTabWidth`
: A measurement in pixels of the width of a pop-up window's tab. Set this field to 0 if the window is not a pop-up window.

`popupTabPosition`
: A value specifying the rule to apply when positioning a pop-up window's tab. Set this field to 0 if the window is not a pop-up window. See "Pop-up Window Tab Positions" (page 118) for the values you can use in this field.

Discussion

Your application uses the `ThemeWindowMetrics` structure to inform the Appearance Manager of the dimensions of specific parts of your window.

Version Notes

The `ThemeWindowMetrics` structure is available with Appearance Manager 1.1 and later.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Appearance.h`

ThemeDrawingState

Defines a reference to a private structure containing information about the current state of a graphics port.

```
typedef struct OpaqueThemeDrawingState * ThemeDrawingState;
```

Discussion
You can use the ThemeDrawingState type with the function GetThemeDrawingState (page 20) to obtain the current graphics port's drawing state and with the function SetThemeDrawingState (page 29) to restore a port's drawing state. You should dispose of the memory allocated to contain a ThemeDrawingState reference by calling DisposeThemeDrawingState (page 17) or passing a value of true in the inDisposeNow parameter of SetThemeDrawingState.

Version Notes
The ThemeDrawingState type is available with Appearance Manager 1.1 and later.

Availability
Available in Mac OS X v10.0 and later.

Declared In
Appearance.h

MenuItemDrawingUPP

Defines a universal procedure pointer (UPP) to a menu item drawing function.

```
typedef MenuItemDrawingProcPtr MenuItemDrawingUPP;
```

Discussion
See MenuItemDrawingProcPtr (page 29) for more information.

Availability
Available in Mac OS X v10.0 and later.

Declared In
Appearance.h

MenuTitleDrawingUPP

Defines a universal procedure pointer (UPP) to a menu title drawing function.

```
typedef MenuTitleDrawingProcPtr MenuTitleDrawingUPP;
```

Discussion
See MenuTitleDrawingProcPtr (page 31) for more information.

Availability
Available in Mac OS X v10.0 and later.

Declared In
Appearance.h

ThemeButtonDrawUPP

Defines a universal procedure pointer (UPP) to a button drawing function.

```
typedef ThemeButtonDrawProcPtr ThemeButtonDrawUPP;
```

Discussion
See `ThemeButtonDrawProcPtr` (page 32) for more information.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Appearance.h`

ThemeEraseUPP

Defines a universal procedure pointer (UPP) to a background drawing callback function.

```
typedef ThemeEraseProcPtr ThemeEraseUPP;
```

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Appearance.h`

ThemeIteratorUPP

Defines a universal procedure pointer (UPP) to a theme iteration callback function.

```
typedef ThemeIteratorProcPtr ThemeIteratorUPP;
```

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Appearance.h`

ThemeTabTitleDrawUPP

Defines a universal procedure pointer (UPP) to a tab title drawing function.

```
typedef ThemeTabTitleDrawProcPtr ThemeTabTitleDrawUPP;
```

Discussion
See `ThemeTabTitleDrawProcPtr` (page 36) for more information.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Appearance.h`

WindowTitleDrawingUPP

Defines a universal procedure pointer (UPP) to a window title drawing function.

```
typedef WindowTitleDrawingProcPtr WindowTitleDrawingUPP;
```

Discussion

See `WindowTitleDrawingProcPtr` (page 37) for more information.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Appearance.h`

Constants

Appearance Manager Apple Events

Identify Apple events sent to Appearance Manager clients when a change occurs in the current appearance.

```
enum {
    kAppearanceEventClass = 'appr',
    kAEAppearanceChanged = 'thme',
    kAESystemFontChanged = 'sysf',
    kAESmallSystemFontChanged = 'ssfn',
    kAEViewsFontChanged = 'vfnt'
};
```

Constants

`kAppearanceEventClass`

The event class of Appearance Manager Apple events.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kAEAppearanceChanged`

The ID of the event indicating the current appearance has changed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kAESystemFontChanged`

The ID of the event indicating the current system font has changed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kAESmallSystemFontChanged`

The ID of the event indicating the current small system font has changed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kAEViewsFontChanged`
The ID of the event indicating the current views font has changed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion
When the user changes the current appearance (that is, when a theme switch occurs), the Appearance Manager may send any of these Apple events to all running applications that are high-level event aware and which are registered as clients of the Appearance Manager. Your application registers itself with the Appearance Manager by calling the function `RegisterAppearanceClient` (page 203).

Because typical results of a theme switch might include a change in menu bar height or window structure dimensions, as well as changes to the system fonts, colors, and patterns that are currently in use, applications should listen for and respond to the Appearance Manager Apple events under most circumstances. Note that none of the Appearance Manager Apple events have parameters and that the return value for each is ignored.

Appearance Manager File Types

Identify the various Appearance Manager file types.

```
enum {
    kThemeDataFileType = 'thme',
    kThemePlatinumFileType = 'pltn',
    kThemeCustomThemesFileType = 'scen',
    kThemeSoundTrackFileType = 'tsnd'
};
```

Constants
`kThemeDataFileType`
The file type of appearances other than the platinum appearance.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemePlatinumFileType`
The file type of the platinum appearance.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeCustomThemesFileType`
The file type of a file that contains user-defined themes. See `SetTheme` (page 204) for a discussion of defining your own theme.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Theme Collection Tags

Identify items in a collection describing a theme.

```
enum {
    kThemeNameTag = 'name',
    kThemeVariantNameTag = 'varn',
    kThemeVariantBaseTintTag = 'tint',
    kThemeHighlightColorTag = 'hcol',
    kThemeScrollBarArrowStyleTag = 'sbar',
    kThemeScrollBarThumbStyleTag = 'sbth',
    kThemeSoundsEnabledTag = 'snds',
    kThemeDblClickCollapseTag = 'coll'
    kThemeAppearanceFileNameTag = 'thme',
    kThemeSystemFontTag = 'lgsf',
    kThemeSmallSystemFontTag = 'smsf',
    kThemeViewsFontTag = 'vfnt',
    kThemeViewsFontSizeTag = 'vfsz',
    kThemeDesktopPatternNameTag = 'patn',
    kThemeDesktopPatternTag = 'patt',
    kThemeDesktopPictureNameTag = 'dpnm',
    kThemeDesktopPictureAliasTag = 'dpal',
    kThemeDesktopPictureAlignmentTag = 'dpan',
    kThemeHighlightColorNameTag = 'hcnm',
    kThemeExamplePictureIDTag = 'epic',
    kThemeSoundTrackNameTag = 'sndt',
    kThemeSoundMaskTag = 'smsk',
    kThemeUserDefinedTag = 'user',
    kThemeSmoothFontEnabledTag = 'smoo',
    kThemeSmoothFontMinSizeTag = 'smos'
};
```

Constants

`kThemeNameTag`

Identifies a collection item containing the name of the theme, e.g. "Mac OS Default". The Appearance Manager only uses this collection item to identify themes within the Appearance control panel, so the `GetTheme` function does not return this collection item. To specify a theme name, you must create a new collection item of this type before calling the function `SetTheme`.

Collection data type: `Str255`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeVariantNameTag`

Identifies a collection item containing the color variation used for menus and controls in the theme.

Collection data type: `Str255`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeHighlightColorTag`

Identifies a collection item containing the text highlight color for the theme.

Collection data type: an `RGBColor` structure

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeScrollBarArrowStyleTag`

Identifies a collection item containing a value of type `ThemeScrollBarArrowStyle` identifying the type of scroll bar arrows used in the theme.

Collection data type: `ThemeScrollBarArrowStyle`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeScrollBarThumbStyleTag`

Identifies a collection item containing a value of type `ThemeScrollBarThumbStyle` identifying the type of scroll boxes used in the theme.

Collection data type: `ThemeScrollBarThumbStyle`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundsEnabledTag`

Identifies a collection item specifying whether theme sounds are enabled for the theme.

Collection data type: `Boolean`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDblClickCollapseTag`

Identifies a collection item specifying whether the ability to double-click to collapse a window is enabled for the theme.

Collection data type: `Boolean`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAppearanceFileNameTag`

Identifies a collection item containing the name of the appearance, e.g. "Apple platinum".

Collection data type: `Str255`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSystemFontTag`

Identifies a collection item containing the name of the large system font for the theme.

Collection data type: `Str255`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSmallSystemFontTag`

Identifies a collection item containing the name of the small system font for the theme.

Collection data type: `Str255`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeViewsFontTag`

Identifies a collection item containing the name of the views font for the theme.

Collection data type: `Str255`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeViewsFontSizeTag`

Identifies a collection item containing the size of the views font for the theme.

Collection data type: `SInt16`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDesktopPatternNameTag`

Identifies a collection item containing the name of the desktop pattern for the theme.

Collection data type: `Str255`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDesktopPatternTag`

Identifies a collection item containing a flattened version of the desktop pattern for the theme.

Collection data type: variable-length data

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDesktopPictureNameTag`

Identifies a collection item containing the name of the desktop picture for the theme.

Collection data type: `Str255`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDesktopPictureAliasTag`

Identifies a collection item containing an alias handle for the desktop picture for the theme.

Collection data type: `AliasHandle`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDesktopPictureAlignmentTag`

Identifies a collection item containing a value specifying how to position the desktop picture for the theme. Possible values are described in "Desktop Picture Alignments" (page 137).

Collection data type: `UInt32`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeHighlightColorNameTag`

Identifies a collection item containing the name of the text highlight color for the theme.

Collection data type: `Str255`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeExamplePictureIDTag`

Identifies a collection item containing the ID of the example picture for the theme.

Collection data type: `SInt16`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

kThemeSoundTrackNameTag

Identifies a collection item containing the name of the soundtrack for the theme.

Collection data type: Str255

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeSoundMaskTag

Identifies a collection item containing an unsigned 32-bit integer whose bits are set to reflect the classes of sounds that are enabled for a theme. Possibilities include sounds for menus, windows, controls, and the Finder. See "Theme Sound Masks" (page 118) for descriptions of possible sound mask values.

Collection data type: UInt32

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeUserDefinedTag

Identifies a collection item specifying whether the theme is user-defined; the value contained in a kThemeUserDefinedTag collection should always be true if the kThemeUserDefinedTag collection is present. The Appearance Manager uses this collection item to identify themes that the user can delete. Note that the GetTheme function does not return this collection item.

Collection data type: Boolean

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeSmoothFontEnabledTag

Identifies a collection item specifying whether font smoothing is enabled in the theme.

Collection data type: Boolean

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeSmoothFontMinSizeTag

Identifies a collection item containing the minimum point size at which font smoothing may be enabled in the theme. Possible values range from 12 to 24, inclusive.

Collection data type: UInt16

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

Discussion

Your application may use these collection tags with the functions SetTheme (page 204) and GetTheme (page 18) to access aspects of a theme. The data type contained in each of the collection items accessed is noted below.

Theme Drawing States

Specify the state in which human interface elements are drawn.

```
enum {
    kThemeStateInactive = 0,
    kThemeStateActive = 1,
    kThemeStatePressed = 2,
    kThemeStateRollover = 6,
    kThemeStateUnavailable = 7,
    kThemeStateUnavailableInactive = 8,
    kThemeStatePressedUp = 2,
    kThemeStatePressedDown = 3
};
typedef UInt32 ThemeDrawState;
```

Constants

`kThemeStateInactive`

The drawing state of elements in an inactive window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeStateActive`

The drawing state of elements in an active window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeStatePressed`

The drawing state of elements in which a mouse click is occurring.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeStateRollover`

The drawing state of elements over which the mouse is positioned.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeStateUnavailable`

The drawing state of elements that are disabled. This state is used to indicate that an element cannot be clicked.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeStateUnavailableInactive`

The drawing state of elements that are disabled and are not in the currently active window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeStatePressedUp`

For stepper controls, the drawing state of the increment button in which a mouse click is occurring.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeStatePressedDown`

For stepper controls, the drawing state of the decrement button in which a mouse click is occurring.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Theme Metrics

Specify metric properties of user interface elements in the current environment.

```
typedef UInt32 ThemeMetric;
enum {
    kThemeMetricScrollBarWidth = 0,
    kThemeMetricSmallScrollBarWidth = 1,
    kThemeMetricCheckBoxHeight = 2,
    kThemeMetricRadioButtonHeight = 3,
    kThemeMetricEditTextWhitespace = 4,
    kThemeMetricEditTextFrameOutset = 5,
    kThemeMetricListBoxFrameOutset = 6,
    kThemeMetricFocusRectOutset = 7,
    kThemeMetricImageWellThickness = 8,
    kThemeMetricScrollBarOverlap = 9,
    kThemeMetricLargeTabHeight = 10,
    kThemeMetricLargeTabCapsWidth = 11,
    kThemeMetricTabFrameOverlap = 12,
    kThemeMetricTabIndentOrStyle = 13,
    kThemeMetricTabOverlap = 14,
    kThemeMetricSmallTabHeight = 15,
    kThemeMetricSmallTabCapsWidth = 16,
    kThemeMetricPushButtonHeight = 19,
    kThemeMetricListHeaderHeight = 20,
    kThemeMetricDisclosureTriangleHeight = 25,
    kThemeMetricDisclosureTriangleWidth = 26,
    kThemeMetricLittleArrowsHeight = 27,
    kThemeMetricLittleArrowsWidth = 28,
    kThemeMetricPopupButtonHeight = 30,
    kThemeMetricSmallPopupButtonHeight = 31,
    kThemeMetricLargeProgressBarThickness = 32,
    kThemeMetricPullDownHeight = 33,
    kThemeMetricSmallPullDownHeight = 34,
    kThemeMetricResizeControlHeight = 38,
    kThemeMetricSmallResizeControlHeight = 39,
    kThemeMetricHSliderHeight = 41,
    kThemeMetricHSliderTickHeight = 42,
    kThemeMetricVSliderWidth = 45,
    kThemeMetricVSliderTickWidth = 46,
    kThemeMetricTitleBarControlsHeight = 49,
    kThemeMetricCheckBoxWidth = 50,
    kThemeMetricRadioButtonWidth = 52,
    kThemeMetricNormalProgressBarThickness = 58,
    kThemeMetricProgressBarShadowOutset = 59,
    kThemeMetricSmallProgressBarShadowOutset = 60,
    kThemeMetricPrimaryGroupBoxContentInset = 61,
    kThemeMetricSecondaryGroupBoxContentInset = 62,
    kThemeMetricMenuMarkColumnWidth = 63,
    kThemeMetricMenuExcludedMarkColumnWidth = 64,
    kThemeMetricMenuMarkIndent = 65,
    kThemeMetricMenuTextLeadingEdgeMargin = 66,
    kThemeMetricMenuTextTrailingEdgeMargin = 67,
    kThemeMetricMenuIndentWidth = 68,
    kThemeMetricMenuIconTrailingEdgeMargin = 69
};
enum {
    kThemeMetricDisclosureButtonHeight = 17,
    kThemeMetricRoundButtonSize = 18,
    kThemeMetricSmallCheckBoxHeight = 21,
    kThemeMetricDisclosureButtonWidth = 22,
    kThemeMetricSmallDisclosureButtonHeight = 23,
    kThemeMetricSmallDisclosureButtonWidth = 24,
    kThemeMetricPaneSplitterHeight = 29,
    kThemeMetricSmallPushButtonHeight = 35,
    kThemeMetricSmallRadioButtonHeight = 36,
    kThemeMetricRelevanceIndicatorHeight = 37,
    kThemeMetricLargeRoundButtonSize = 40,
    kThemeMetricSmallHSliderHeight = 43,
    kThemeMetricSmallHSliderTickHeight = 44,
    kThemeMetricSmallVSliderWidth = 47,
    kThemeMetricSmallVSliderTickWidth = 48,
    kThemeMetricSmallCheckBoxWidth = 51,
    kThemeMetricSmallRadioButtonWidth = 53,
    kThemeMetricSmallHSliderMinThumbWidth = 54,
    kThemeMetricSmallVSliderMinThumbHeight = 55,
    kThemeMetricSmallHSliderTickOffset = 56,
    kThemeMetricSmallVSliderTickOffset = 57
};
enum {
    kThemeMetricComboBoxLargeBottomShadowOffset = 70,
    kThemeMetricComboBoxLargeRightShadowOffset = 71,
    kThemeMetricComboBoxSmallBottomShadowOffset = 72,
    kThemeMetricComboBoxSmallRightShadowOffset = 73,
    kThemeMetricComboBoxLargeDisclosureWidth = 74,
    kThemeMetricComboBoxSmallDisclosureWidth = 75,
    kThemeMetricRoundTextFieldContentInsetLeft = 76,
    kThemeMetricRoundTextFieldContentInsetRight = 77,
    kThemeMetricRoundTextFieldContentInsetBottom = 78,
    kThemeMetricRoundTextFieldContentInsetTop = 79,
    kThemeMetricRoundTextFieldContentHeight = 80,
    kThemeMetricComboBoxMiniBottomShadowOffset = 81,
    kThemeMetricComboBoxMiniDisclosureWidth = 82,
    kThemeMetricComboBoxMiniRightShadowOffset = 83,
    kThemeMetricLittleArrowsMiniHeight = 84,
    kThemeMetricLittleArrowsMiniWidth = 85,
    kThemeMetricLittleArrowsSmallHeight = 86,
    kThemeMetricLittleArrowsSmallWidth = 87,
    kThemeMetricMiniCheckBoxHeight = 88,
    kThemeMetricMiniCheckBoxWidth = 89,
    kThemeMetricMiniDisclosureButtonHeight = 90,
    kThemeMetricMiniDisclosureButtonWidth = 91,
    kThemeMetricMiniHSliderHeight = 92,
    kThemeMetricMiniHSliderMinThumbWidth = 93,
    kThemeMetricMiniHSliderTickHeight = 94,
    kThemeMetricMiniHSliderTickOffset = 95,
    kThemeMetricMiniPopupButtonHeight = 96,
    kThemeMetricMiniPullDownHeight = 97,
    kThemeMetricMiniPushButtonHeight = 98,
    kThemeMetricMiniRadioButtonHeight = 99,
    kThemeMetricMiniRadioButtonWidth = 100,
    kThemeMetricMiniTabCapsWidth  = 101,
    kThemeMetricMiniTabFrameOverlap = 102,
    kThemeMetricMiniTabHeight = 103,
    kThemeMetricMiniTabOverlap = 104,
    kThemeMetricMiniVSliderMinThumbHeight = 105,
    kThemeMetricMiniVSliderTickOffset = 106,
    kThemeMetricMiniVSliderTickWidth = 107,
    kThemeMetricMiniVSliderWidth = 108,
    kThemeMetricRoundTextFieldContentInsetWithIconLeft = 109,
    kThemeMetricRoundTextFieldContentInsetWithIconRight = 110,
    kThemeMetricRoundTextFieldMiniContentHeight = 111,
    kThemeMetricRoundTextFieldMiniContentInsetBottom = 112,
    kThemeMetricRoundTextFieldMiniContentInsetLeft = 113,
    kThemeMetricRoundTextFieldMiniContentInsetRight = 114,
    kThemeMetricRoundTextFieldMiniContentInsetTop = 115,
    kThemeMetricRoundTextFieldMiniContentInsetWithIconLeft = 116,
    kThemeMetricRoundTextFieldMiniContentInsetWithIconRight = 117,
    kThemeMetricRoundTextFieldSmallContentHeight = 118,
    kThemeMetricRoundTextFieldSmallContentInsetBottom = 119,
    kThemeMetricRoundTextFieldSmallContentInsetLeft = 120,
    kThemeMetricRoundTextFieldSmallContentInsetRight = 121,
    kThemeMetricRoundTextFieldSmallContentInsetTop = 122,
    kThemeMetricRoundTextFieldSmallContentInsetWithIconLeft = 123,
    kThemeMetricRoundTextFieldSmallContentInsetWithIconRight = 124,
    kThemeMetricSmallTabFrameOverlap = 125,
    kThemeMetricSmallTabOverlap = 126,
    kThemeMetricSmallPaneSplitterHeight = 127
};
enum {
    kThemeMetricHSliderTickOffset = 128,
    kThemeMetricVSliderTickOffset = 129,
    kThemeMetricSliderMinThumbHeight = 130,
    kThemeMetricSliderMinThumbWidth = 131,
    kThemeMetricScrollBarMinThumbHeight = 132,
    kThemeMetricScrollBarMinThumbWidth = 133,
    kThemeMetricSmallScrollBarMinThumbHeight = 134,
    kThemeMetricSmallScrollBarMinThumbWidth = 135,
    kThemeMetricButtonRoundedHeight = 136,
    kThemeMetricButtonRoundedRecessedHeight = 137
};
enum {
    kThemeMetricSeparatorSize = 138,
    kThemeMetricTexturedPushButtonHeight = 139,
    kThemeMetricTexturedSmallPushButtonHeight = 140
};
```

Constants

`kThemeMetricScrollBarWidth`

The width of a scroll bar. For horizontal scroll bars, this measurement is actually the scroll bar height.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallScrollBarWidth`

The width of a small scroll bar. For horizontal scroll bars, this measurement is actually the scroll bar height.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricCheckBoxHeight`

The height of the non-label part of a check box control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricRadioButtonHeight`

The height of the non-label part of a radio button control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricEditTextWhitespace`

The amount of white space surrounding the text rectangle of the text inside of an Edit Text control. If you select all of the text in an Edit Text control, this white space is visible. The metric is the number of pixels, per side, that the text rectangle is outset to create the whitespace rectangle.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricEditTextFrameOutset`

The thickness of the Edit Text frame that surrounds the whitespace rectangle (which surrounds the text rectangle). The metric is the number of pixels, per side, that the frame rectangle is outset from the whitespace rectangle.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricListBoxFrameOutset`

The number of pixels that the list box frame is outset from the content of the list box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricFocusRectOutset`

This is a deprecated metric and you should not use it. This metric describes how far from a control the focus rectangle was drawn, but control focus drawing no longer uses this information.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricImageWellThickness`

The thickness of the frame drawn by `DrawThemeGenericWell`.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricScrollBarOverlap`

The number of pixels a scrollbar should overlap any bounding box which surrounds it and scrollable content. This also includes the window frame when a scrollbar is along an edge of the window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricLargeTabHeight`

The height of the large tab of a tab control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricLargeTabCapsWidth`

The width of the caps, or end pieces, of the large tabs of a tab control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricTabFrameOverlap`

The amount you must add to the tab height, `kThemeMetricLargeTabHeight`, to find the rectangle height to use with the various tab drawing primitives. This amount is also the amount that each tab overlaps the tab pane.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricTabIndentOrStyle`

If less than zero, this indicates that the text should be centered on each tab. If greater than zero, the text should be justified, according to the system script direction, and the amount is the offset from the edge at which the text should start drawing.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricTabOverlap`

The amount of space that every tab's drawing rectangle overlapsthat of the tab on either side of it.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallTabHeight`

The height of the small tab of a tab control. This includes the pixels that overlap the tab pane and/or tab pane bar.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallTabCapsWidth`

The width of the caps, or end pieces, of the small tabs of a tab control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricPushButtonHeight`

The height and the width of the push button control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricListHeaderHeight`

The height of the list header field of the data browser control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricDisclosureTriangleHeight`

The height of a disclosure triangle control. This triangle is the not the center of the disclosure button, but its own control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricDisclosureTriangleWidth`

The width of a disclosure triangle control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricLittleArrowsHeight`

The height of a little arrows control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricLittleArrowsWidth`

The width of a little arrows control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricPopupButtonHeight`

The height of a popup button control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallPopupButtonHeight`

The height of a small popup button control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricLargeProgressBarThickness`

The height of the large progress bar, not including its shadow.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricPullDownHeight`

This metric is not used.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallPullDownHeight`

This metric is not used.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricResizeControlHeight`

The height of the window grow box control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallResizeControlHeight`

The width of the window grow box control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricHSliderHeight`

The height of the horizontal slider control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricHSliderTickHeight`
: The height of the tick marks for a horizontal slider control.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricVSliderWidth`
: The width of the vertical slider control.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricVSliderTickWidth`
: The width of the tick marks for a vertical slider control.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricTitleBarControlsHeight`
: The height of the title bar widgets (grow, close, and zoom boxes) for a document window.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricCheckBoxWidth`
: The width of the non-label part of a check box control.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricRadioButtonWidth`
: The width of the non-label part of a radio button control.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricNormalProgressBarThickness`
: The height of the normal progress bar, not including its shadow.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricProgressBarShadowOutset`
: The number of pixels of shadow depth drawn below the progress bar.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricSmallProgressBarShadowOutset`
: The number of pixels of shadow depth drawn below the small progress bar.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricPrimaryGroupBoxContentInset`
: The number of pixels that the content of a primary group box is inset from the bounds of the control.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricSecondaryGroupBoxContentInset`

The number of pixels that the content of a secondary group box is from the bounds of the control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricMenuMarkColumnWidth`

The width allocated to draw the mark character in a menu.

Available in Mac OS X v10.1 and later.

Declared in `Appearance.h`.

`kThemeMetricMenuExcludedMarkColumnWidth`

The width allocated for the mark character in a menu item when the menu has the attribute `kMenuAttrExcludesMarkColumn`.

Available in Mac OS X v10.1 and later.

Declared in `Appearance.h`.

`kThemeMetricMenuMarkIndent`

The indent into the interior of the mark column at which the mark character is drawn.

Available in Mac OS X v10.1 and later.

Declared in `Appearance.h`.

`kThemeMetricMenuTextLeadingEdgeMargin`

The whitespace at the leading edge of menu item text.

Available in Mac OS X v10.1 and later.

Declared in `Appearance.h`.

`kThemeMetricMenuTextTrailingEdgeMargin`

The whitespace at the trailing edge of menu item text.

Available in Mac OS X v10.1 and later.

Declared in `Appearance.h`.

`kThemeMetricMenuIndentWidth`

The width per indent level of a menu item. This indent is set by the `SetMenuItemIndent` function.

Available in Mac OS X v10.1 and later.

Declared in `Appearance.h`.

`kThemeMetricMenuIconTrailingEdgeMargin`

The whitespace at the trailing edge of a menu icon, if the item also has text.

Available in Mac OS X v10.1 and later.

Declared in `Appearance.h`.

`kThemeMetricDisclosureButtonHeight`

The height of a disclosure button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundButtonSize`

The height and the width of the round button control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallCheckBoxHeight`
: The height of the non-label part of a small check box control.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricDisclosureButtonWidth`
: The width of a disclosure button.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricSmallDisclosureButtonHeight`
: The height of a small disclosure button.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricSmallDisclosureButtonWidth`
: The width of a small disclosure button.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricPaneSplitterHeight`
: The height (or width if vertical) of a pane splitter.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricSmallPushButtonHeight`
: The height of the small push button control.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricSmallRadioButtonHeight`
: The height of the non-label part of a small radio button control.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricRelevanceIndicatorHeight`
: The height of the relevance indicator control.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricLargeRoundButtonSize`
: The height and the width of the large round button control.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricSmallHSliderHeight`
: The height of the small, horizontal slider control.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeMetricSmallHSliderTickHeight`
The height of the tick marks for a small, horizontal slider control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallVSliderWidth`
The width of the small, vertical slider control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallVSliderTickWidth`
The width of the tick marks for a small, vertical slider control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallCheckBoxWidth`
The width of the non-label part of a small check box control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallRadioButtonWidth`
The width of the non-label part of a small radio button control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallHSliderMinThumbWidth`
The minimum width of the thumb of a small, horizontal slider control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallVSliderMinThumbHeight`
The minimum width of the thumb of a small, vertical slider control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallHSliderTickOffset`
The offset of the tick marks from the appropriate side of a small horizontal slider control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallVSliderTickOffset`
The offset of the tick marks from the appropriate side of a small vertical slider control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMetricComboBoxLargeBottomShadowOffset`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricComboBoxLargeRightShadowOffset`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricComboBoxSmallBottomShadowOffset`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricComboBoxSmallRightShadowOffset`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricComboBoxLargeDisclosureWidth`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricComboBoxSmallDisclosureWidth`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldContentInsetLeft`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldContentInsetRight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldContentInsetBottom`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldContentInsetTop`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldContentHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricComboBoxMiniBottomShadowOffset`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricComboBoxMiniDisclosureWidth`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricComboBoxMiniRightShadowOffset`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricLittleArrowsMiniHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricLittleArrowsMiniWidth`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricLittleArrowsSmallHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricLittleArrowsSmallWidth`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniCheckBoxHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniCheckBoxWidth`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniDisclosureButtonHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniDisclosureButtonWidth`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniHSliderHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniHSliderMinThumbWidth`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniHSliderTickHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniHSliderTickOffset`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniPopupButtonHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniPullDownHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniPushButtonHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniRadioButtonHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniRadioButtonWidth`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniTabCapsWidth`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniTabFrameOverlap`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniTabHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniTabOverlap`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniVSliderMinThumbHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniVSliderTickOffset`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniVSliderTickWidth`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricMiniVSliderWidth`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldContentInsetWithIconLeft`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldContentInsetWithIconRight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldMiniContentHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldMiniContentInsetBottom`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldMiniContentInsetLeft`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldMiniContentInsetRight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldMiniContentInsetTop`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldMiniContentInsetWithIconLeft`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldMiniContentInsetWithIconRight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldSmallContentHeight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldSmallContentInsetBottom`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldSmallContentInsetLeft`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldSmallContentInsetRight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldSmallContentInsetTop`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldSmallContentInsetWithIconLeft`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricRoundTextFieldSmallContentInsetWithIconRight`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallTabFrameOverlap`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallTabOverlap`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallPaneSplitterHeight`

The height of a small pane splitter. Should only be used in a window with thick borders, such as a textured window.

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeMetricHSliderTickOffset`

The horizontal start offset for the first tick mark on a horizontal slider.

Available in Mac OS X v10.4 and later.

Declared in `Appearance.h`.

`kThemeMetricVSliderTickOffset`

The vertical start offset for the first tick mark on a vertical slider.

Available in Mac OS X v10.4 and later.

Declared in `Appearance.h`.

`kThemeMetricSliderMinThumbHeight`

The minimum height for a thumb on a slider.

Available in Mac OS X v10.4 and later.

Declared in `Appearance.h`.

`kThemeMetricSliderMinThumbWidth`

The minimum width for a thumb on a slider.

Available in Mac OS X v10.4 and later.

Declared in `Appearance.h`.

`kThemeMetricScrollBarMinThumbHeight`

The minimum height for a thumb on a scroll bar.

Available in Mac OS X v10.4 and later.

Declared in `Appearance.h`.

`kThemeMetricScrollBarMinThumbWidth`

The minimum width for a thumb on a scroll bar.

Available in Mac OS X v10.4 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallScrollBarMinThumbHeight`

The minimum height for a thumb on a small scroll bar.

Available in Mac OS X v10.4 and later.

Declared in `Appearance.h`.

`kThemeMetricSmallScrollBarMinThumbWidth`

The minimum width for a thumb on a small scroll bar.

Available in Mac OS X v10.4 and later.

Declared in `Appearance.h`.

`kThemeMetricButtonRoundedHeight`

The height of a round-ended button (for example, the Kind button in a Finder Search query.)

Available in Mac OS X v10.5 and later.

Declared in `Appearance.h`.

`kThemeMetricButtonRoundedRecessedHeight`

The height of the inset round-ended button (for example, the Servers button in a Finder Search query.)

Available in Mac OS X v10.5 and later.

Declared in `Appearance.h`.

`kThemeMetricSeparatorSize`

The height of a horizontal separator, or the width of a vertical separator, drawn with the `HIThemeDrawSeparator` theme primitive.

Available in Mac OS X v10.5 and later.

Declared in `Appearance.h`.

`kThemeMetricTexturedPushButtonHeight`

The height of the push button control designed for use in a textured window.

Available in Mac OS X v10.5 and later.

Declared in `Appearance.h`.

`kThemeMetricTexturedSmallPushButtonHeight`

The height of the small push button control designed for use in a textured window.

Available in Mac OS X v10.5 and later.

Declared in `Appearance.h`.

Theme Backgrounds

Identify theme-compliant backgrounds.

```
enum {
    kThemeBackgroundTabPane = 1,
    kThemeBackgroundPlacard = 2,
    kThemeBackgroundWindowHeader = 3,
    kThemeBackgroundListViewWindowHeader = 4,
    kThemeBackgroundSecondaryGroupBox = 5
};
typedef UInt32 ThemeBackgroundKind;
```

Constants

`kThemeBackgroundTabPane`

The background for a tab pane.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBackgroundPlacard`

The background for a placard.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBackgroundWindowHeader`

The background for a window header.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBackgroundListViewWindowHeader`

The background for a window list view header.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

You can pass a constant of type `ThemeBackgroundKind` to the function `ApplyThemeBackground` (page 145) to specify that an embedded object have a background consistent with the current theme and object in which it is visually embedded.

Theme Brushes

Specify theme-compliant colors and patterns for particular human interface elements.

```
enum {
    kThemeBrushDialogBackgroundActive = 1,
    kThemeBrushDialogBackgroundInactive = 2,
    kThemeBrushAlertBackgroundActive = 3,
    kThemeBrushAlertBackgroundInactive = 4,
    kThemeBrushModelessDialogBackgroundActive = 5,
    kThemeBrushModelessDialogBackgroundInactive = 6,
    kThemeBrushUtilityWindowBackgroundActive = 7,
    kThemeBrushUtilityWindowBackgroundInactive = 8,
    kThemeBrushListViewSortColumnBackground = 9,
    kThemeBrushListViewBackground = 10,
    kThemeBrushIconLabelBackground = 11,
    kThemeBrushListViewSeparator = 12,
    kThemeBrushChasingArrows = 13,
    kThemeBrushDragHilite = 14,
    kThemeBrushDocumentWindowBackground = 15,
    kThemeBrushFinderWindowBackground = 16,
    kThemeBrushScrollBarDelimiterActive = 17,
    kThemeBrushScrollBarDelimiterInactive = 18,
    kThemeBrushFocusHighlight = 19,
    kThemeBrushPopupArrowActive = 20,
    kThemeBrushPopupArrowPressed = 21,
    kThemeBrushPopupArrowInactive = 22,
    kThemeBrushAppleGuideCoachmark = 23,
    kThemeBrushIconLabelBackgroundSelected = 24,
    kThemeBrushStaticAreaFill = 25,
    kThemeBrushActiveAreaFill = 26,
    kThemeBrushButtonFrameActive = 27,
    kThemeBrushButtonFrameInactive = 28,
    kThemeBrushButtonFaceActive = 29,
    kThemeBrushButtonFaceInactive = 30,
    kThemeBrushButtonFacePressed = 31,
    kThemeBrushButtonActiveDarkShadow = 32,
    kThemeBrushButtonActiveDarkHighlight = 33,
    kThemeBrushButtonActiveLightShadow = 34,
    kThemeBrushButtonActiveLightHighlight = 35,
    kThemeBrushButtonInactiveDarkShadow = 36,
    kThemeBrushButtonInactiveDarkHighlight = 37,
    kThemeBrushButtonInactiveLightShadow = 38,
    kThemeBrushButtonInactiveLightHighlight = 39,
    kThemeBrushButtonPressedDarkShadow = 40,
    kThemeBrushButtonPressedDarkHighlight = 41,
    kThemeBrushButtonPressedLightShadow = 42,
    kThemeBrushButtonPressedLightHighlight = 43,
    kThemeBrushBevelActiveLight = 44,
    kThemeBrushBevelActiveDark = 45,
    kThemeBrushBevelInactiveLight = 46,
    kThemeBrushBevelInactiveDark = 47,
    kThemeBrushNotificationWindowBackground = 48,
    kThemeBrushMovableModalBackground = 49,
    kThemeBrushSheetBackgroundOpaque = 50,
    kThemeBrushDrawerBackground = 51,
    kThemeBrushToolbarBackground = 52,
    kThemeBrushSheetBackgroundTransparent = 53,
    kThemeBrushMenuBackground = 54,
    kThemeBrushMenuBackgroundSelected = 55,
    kThemeBrushListViewOddRowBackground = 56,
    kThemeBrushListViewEvenRowBackground = 57,
    kThemeBrushListViewColumnDivider = 58,
    kThemeBrushSheetBackground = kThemeBrushSheetBackgroundOpaque,
    kThemeBrushBlack = -1,
    kThemeBrushWhite = -2,
    kThemeBrushPrimaryHighlightColor = -3,
    kThemeBrushSecondaryHighlightColor = -4,
    kThemeBrushAlternatePrimaryHighlightColor = -5
};
typedef SInt16 ThemeBrush;
```

Constants

`kThemeBrushDialogBackgroundActive`

An active dialog box's background color or pattern.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushDialogBackgroundInactive`

An inactive dialog box's background color or pattern.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushAlertBackgroundActive`

An active alert box's background color or pattern.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushAlertBackgroundInactive`

An inactive alert box's background color or pattern.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushModelessDialogBackgroundActive`

An active modeless dialog box's background color or pattern.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushModelessDialogBackgroundInactive`

An inactive modeless dialog box's background color or pattern.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushUtilityWindowBackgroundActive`

An active utility window's background color or pattern.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushUtilityWindowBackgroundInactive`

An inactive utility window's background color or pattern.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushListViewSortColumnBackground`

The background color or pattern of the list view column that is being sorted upon.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushListViewBackground`

The background color or pattern of a list view column that is not being sorted upon.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushIconLabelBackground`

An icon label's color or pattern.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushListViewSeparator`

The color or pattern of the horizontal lines that separate rows of items in list view columns.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushChasingArrows`

Asynchronous arrows' color or pattern.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushDragHilite`

The color or pattern used to indicate that an element is a valid drag-and-drop destination

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushDocumentWindowBackground`

A document window's background color or pattern.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushFinderWindowBackground`

A Finder window's background color or pattern. Generally, you should not use this constant unless you are trying to create a window that matches a Finder window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushScrollBarDelimiterActive`

The color or pattern used to outline the sides of an active scroll bar.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushScrollBarDelimiterInactive`

The color or pattern used to outline the sides of an inactive scroll bar.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushFocusHighlight`

The color or pattern of the focus ring around an element that is selected.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushPopupArrowActive`

The color or pattern of the arrow on an active pop-up menu button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushPopupArrowPressed`

The color or pattern of the arrow on a pop-up menu button that is being clicked on by the user.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushPopupArrowInactive`

The color or pattern of the arrow on an inactive pop-up menu button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushAppleGuideCoachmark`

The color or pattern of an Apple Guide coachmark.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushIconLabelBackgroundSelected`

The color or pattern of the background of an icon's label area, when the icon is selected.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushStaticAreaFill`

The background color or pattern of an element that does not support user interaction.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushActiveAreaFill`

The color or pattern of an element that supports user interaction.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonFrameActive`

The color or pattern that outlines an active button. Your application should draw the button outline outside the edge of the button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonFrameInactive`

The color or pattern that outlines an inactive button. Your application should draw the button outline outside the edge of the button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonFaceActive`
The color or pattern of the face of an active button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonFaceInactive`
The color or pattern of the face of an inactive button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonFacePressed`
The color or pattern of the face of a button that is being clicked on by the user.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonActiveDarkShadow`
For an active button with a 2-pixel-wide edge, the color or pattern of the bottom and right sides of the outer ring of the edge.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonActiveDarkHighlight`
For an active button with a 2-pixel-wide edge, the color or pattern of the top and left sides of the outer ring of the edge.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonActiveLightShadow`
For an active button with a 2-pixel-wide edge, the color or pattern of the bottom and right sides of the inner ring of the edge. For an active button with a 1-pixel-wide edge, the color or pattern of the bottom and right sides of the edge.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonActiveLightHighlight`
For an active button with a 2-pixel-wide edge, the color or pattern of the top and left sides of the inner ring of the edge. For an active button with a 1-pixel-wide edge, the color or pattern of the top and left sides of the edge.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonInactiveDarkShadow`
For an inactive button with a 2-pixel-wide edge, the color or pattern of the bottom and right sides of the outer ring of the edge.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonInactiveDarkHighlight`
For an inactive button with a 2-pixel-wide edge, the color or pattern of the top and left sides of the outer ring of the edge.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonInactiveLightShadow`

For an inactive button with a 2-pixel-wide edge, the color or pattern of the bottom and right sides of the inner ring of the edge. For an inactive button with a 1-pixel-wide edge, the color or pattern of the bottom and right sides of the edge.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonInactiveLightHighlight`

For an inactive button with a 2-pixel-wide edge, the color or pattern of the top and left sides of the inner ring of the edge. For an inactive button with a 1-pixel-wide edge, the color or pattern of the top and left sides of the edge.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonPressedDarkShadow`

For a button with a 2-pixel-wide edge that is being clicked on by the user, the color or pattern of the bottom and right sides of the outer ring of the edge.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonPressedDarkHighlight`

For a button with a 2-pixel-wide edge that is being clicked on by the user, the color or pattern of the top and left sides of the outer ring of the edge.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonPressedLightShadow`

For a button with a 2-pixel-wide edge that is being clicked on by the user, the color or pattern of the bottom and right sides of the inner ring of the edge. For a button with a 1-pixel-wide edge that is being clicked on by the user, the color or pattern of the bottom and right sides of the edge.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushButtonPressedLightHighlight`

For a button with a 2-pixel-wide edge that is being clicked on by the user, the color or pattern of the top and left sides of the inner ring of the edge. For a button with a 1-pixel-wide edge that is being clicked on by the user, the color or pattern of the top and left sides of the edge.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushBevelActiveLight`

For an active bevel button, the color or pattern of the top and left sides of the bevel.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushBevelActiveDark`

For an active bevel button, the color or pattern of the bottom and right sides of the bevel.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushBevelInactiveLight`

For an inactive bevel button, the color or pattern of the top and left sides of the bevel.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushBevelInactiveDark`

For an inactive bevel button, the color or pattern of the bottom and right sides of the bevel.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushNotificationWindowBackground`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushMovableModalBackground`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushSheetBackgroundOpaque`

Available in Mac OS X v10.1 and later.

Declared in `Appearance.h`.

`kThemeBrushDrawerBackground`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushToolbarBackground`

Available in Mac OS X v10.1 and later.

Declared in `Appearance.h`.

`kThemeBrushSheetBackgroundTransparent`

Available in Mac OS X v10.1 and later.

Declared in `Appearance.h`.

`kThemeBrushMenuBackground`

Available in Mac OS X v10.2 and later.

Declared in `Appearance.h`.

`kThemeBrushMenuBackgroundSelected`

Available in Mac OS X v10.2 and later.

Declared in `Appearance.h`.

`kThemeBrushListViewOddRowBackground`

Available in Mac OS X v10.4 and later.

Declared in `Appearance.h`.

`kThemeBrushListViewEvenRowBackground`

Available in Mac OS X v10.4 and later.

Declared in `Appearance.h`.

`kThemeBrushListViewColumnDivider`

Available in Mac OS X v10.4 and later.

Declared in `Appearance.h`.

`kThemeBrushSheetBackground`

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushBlack`

Black; this color does not change from theme to theme. You may use this constant instead of specifying a direct RGB value.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushWhite`

White; this color does not change from theme to theme. You may use this constant instead of specifying a direct RGB value.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBrushPrimaryHighlightColor`

Available in Mac OS X v10.2 and later.

Declared in `Appearance.h`.

`kThemeBrushSecondaryHighlightColor`

Available in Mac OS X v10.2 and later.

Declared in `Appearance.h`.

`kThemeBrushAlternatePrimaryHighlightColor`

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

Discussion

The Appearance Manager provides the underlying support for RGB color data and overrides System 7 color tables such as `'cctb'` and `'mctb'` with an abstract mechanism that allows colors and patterns to be coordinated with the current theme. You can pass constants of type `ThemeBrush` in the `inBrush` parameter of `SetThemeBackground` (page 205), `SetThemePen` (page 206), and `SetThemeWindowBackground` to specify that the Appearance Manager substitute whatever the appropriate color or pattern is for a given human interface element in the current theme.

Theme Buttons

Identify types of buttons.

```
enum {
    kThemePushButton = 0,
    kThemeCheckBox = 1,
    kThemeRadioButton = 2,
    kThemeBevelButton = 3,
    kThemeArrowButton = 4,
    kThemePopupButton = 5,
    kThemeDisclosureButton = 6,
    kThemeIncDecButton = 7,
    kThemeSmallBevelButton = 8,
    kThemeMediumBevelButton = 3,
    kThemeLargeBevelButton = 9,
    kThemeListHeaderButton = 10,
    kThemeRoundButton = 11,
    kThemeLargeRoundButton = 12,
    kThemeSmallCheckBox = 13,
    kThemeSmallRadioButton = 14,
    kThemeRoundedBevelButton = 15,
    kThemeNormalCheckBox = 1,
    kThemeNormalRadioButton = 2
};
typedef UInt16 ThemeButtonKind;
```

Constants

`kThemePushButton`

Identifies a push button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeCheckBox`

Identifies a checkbox.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeRadioButton`

Identifies a radio button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeBevelButton`

Identifies a bevel button with a medium-width bevel; this value is the same as `kThemeMediumBevelButton`.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeArrowButton`

Identifies an arrow button. This button has the appearance of a single button containing small upward- and downward-pointing triangles drawn back to back; the typical use of this button is with an editable text field to create an editable pop-up menu. This button should not be confused with an increment/decrement button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemePopupButton`

Identifies a pop-up menu button. This button has the appearance of a single button made of two parts: a menu item text part and an arrow part.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDisclosureButton`

Identifies a disclosure triangle.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeIncDecButton`

Identifies an increment/decrement or "little arrows" button. This button has the appearance of two separate buttons—one containing an upward-pointing triangle and the other containing a downward-pointing triangle—placed back to back. This button should not be confused with the arrow button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSmallBevelButton`

Identifies a bevel button with a small-width bevel.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMediumBevelButton`

Identifies a bevel button with a medium-width bevel; this value is the same as `kThemeBevelButton`.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeLargeBevelButton`

Identifies a bevel button with a large-width bevel.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeListHeaderButton`

Identifies a sort button for the top of a list.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeRoundButton`

Identifies a round button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeLargeRoundButton`

Identifies a large round button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSmallCheckBox`
  Identifies a small checkbox.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSmallRadioButton`
  Identifies a small radio button.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeRoundedBevelButton`
  Identifies a rounded bevel button.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeNormalCheckBox`
  Identifies a checkbox; this value is the same as `kThemeCheckBox`.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeNormalRadioButton`
  Identifies a radio button; this value is the same as `kThemeRadioButton`.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

Discussion

You can pass constants of type `ThemeButtonKind` to the function `DrawThemeButton` (page 149) to draw a theme-compliant button of a specific type. You can also pass `ThemeButtonKind` constants to the functions `GetThemeButtonRegion` (page 179) and `GetThemeButtonContentBounds` (page 178) to retrieve information about a specific button type.

Theme Button Adornments

Specify the visual characteristics of a button control.

```
enum {
    kThemeAdornmentNone = 0,
    kThemeAdornmentDefault = (1 << 0),
    kThemeAdornmentFocus = (1 << 2),
    kThemeAdornmentRightToLeft = (1 << 4),
    kThemeAdornmentDrawIndicatorOnly = (1 << 5),
    kThemeAdornmentHeaderButtonLeftNeighborSelected = (1 <<
6),
    kThemeAdornmentHeaderButtonRightNeighborSelected = (1 <<
7),
    kThemeAdornmentHeaderButtonSortUp = (1 << 8),
    kThemeAdornmentHeaderMenuButton = (1 << 9),
    kThemeAdornmentHeaderButtonNoShadow = (1 << 10),
    kThemeAdornmentHeaderButtonShadowOnly = (1 << 11),
    kThemeAdornmentNoShadow = kThemeAdornmentHeaderButtonNoShadow,
    kThemeAdornmentShadowOnly = kThemeAdornmentHeaderButtonShadowOnly,
    kThemeAdornmentArrowLeftArrow = (1 << 6),
    kThemeAdornmentArrowDownArrow = (1 << 7),
    kThemeAdornmentArrowDoubleArrow = (1 << 8),
    kThemeAdornmentArrowUpArrow = (1 << 9)
};
typedef UInt16 ThemeButtonAdornment;
```

Constants

`kThemeAdornmentNone`

If no bits are set, the button is drawn with no adornment.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentDefault`

If the bit specified by this mask is set, a default button ring is drawn. This constant applies to push button controls only.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentFocus`

If the bit specified by this mask is set, a focus ring is drawn.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentRightToLeft`

If the bit specified by this mask is set, the button is drawn in a right-to-left orientation.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentDrawIndicatorOnly`

If the bit specified by this mask is set, only the button is drawn, not its label. This characteristic applies to radio buttons, checkboxes, and disclosure triangles.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentHeaderButtonLeftNeighborSelected`

If the bit specified by this mask is set, the left border of the button is drawn as selected (list header button only).

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentHeaderButtonRightNeighborSelected`

If the bit specified by this mask is set, the right border of the button is drawn as selected (list header button only).

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentHeaderButtonSortUp`

If the bit specified by this mask is set the sort indicator is drawn pointing upward (list header button only).

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentHeaderMenuButton`

If the bit specified by this mask is set, the button is drawn as a header menu button (list header button only).

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentHeaderButtonNoShadow`

If the bit specified by this mask is set, the non-shadow area of the button is drawn (list header button only).

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentHeaderButtonShadowOnly`

If the bit specified by this mask is set, only the shadow area of the button is drawn (list header button only).

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentNoShadow`

Use `kThemeAdornmentHeaderButtonNoShadow` instead.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentShadowOnly`

Use `kThemeAdornmentHeaderButtonShadowOnly` instead.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentArrowLeftArrow`

If the bit specified by this mask is set, a left arrow is drawn on the arrow button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAdornmentArrowDownArrow`
    If the bit specified by this mask is set, a down arrow is drawn on the arrow button.

    Available in Mac OS X v10.0 and later.

    Declared in `Appearance.h`.

`kThemeAdornmentArrowDoubleArrow`
    If the bit specified by this mask is set, a double arrow is drawn on the arrow button.

    Available in Mac OS X v10.0 and later.

    Declared in `Appearance.h`.

`kThemeAdornmentArrowUpArrow`
    If the bit specified by this mask is set, an up arrow is drawn on the arrow button.

    Available in Mac OS X v10.0 and later.

    Declared in `Appearance.h`.

Discussion

The `ThemeButtonAdornment` enumeration defines masks your application can use in the `ThemeButtonDrawInfo` (page 40) structure to specify that button controls are drawn with the appropriate human interface characteristics.

Theme Button Values

Specify the value of a button.

```
enum {
    kThemeButtonOff = 0,
    kThemeButtonOn = 1,
    kThemeButtonMixed = 2,
    kThemeDisclosureRight = 0,
    kThemeDisclosureDown = 1,
    kThemeDisclosureLeft = 2
};
typedef UInt16 ThemeButtonValue;
```

Constants

`kThemeButtonOff`
    Identifies a button that is not selected.

    Available in Mac OS X v10.0 and later.

    Declared in `Appearance.h`.

`kThemeButtonOn`
    Identifies a button that is selected.

    Available in Mac OS X v10.0 and later.

    Declared in `Appearance.h`.

`kThemeButtonMixed`
    Identifies a button that is in the mixed state, indicating that a setting is on for some elements in a selection and off for others. This value typically applies to checkboxes and radio buttons.

    Available in Mac OS X v10.0 and later.

    Declared in `Appearance.h`.

`kThemeDisclosureRight`

Identifies a disclosure triangle that is pointing to the right.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDisclosureDown`

Identifies a disclosure triangle that is pointing down.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDisclosureLeft`

Identifies a disclosure triangle that is pointing to the left.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

You can use constants of type `ThemeButtonValue` in the `ThemeButtonDrawInfo` (page 40) structure to specify that button controls are drawn with the correct values.

Theme Pop-Up Arrow Orientations

Specify the direction in which a pop-up arrow is drawn on a button.

```
enum {
    kThemeArrowLeft = 0,
    kThemeArrowDown = 1,
    kThemeArrowRight = 2,
    kThemeArrowUp = 3
};
typedef UInt16 ThemeArrowOrientation;
```

Constants

`kThemeArrowLeft`

A left-pointing arrow.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeArrowDown`

A downward-pointing arrow.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeArrowRight`

A right-pointing arrow.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeArrowUp`

An upward-pointing arrow.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion
You can use a constant of type `ThemeArrowOrientation` in the function `DrawThemePopupArrow` (page 160).

Theme Pop-Up Arrow Sizes

Specify the size of the pop-up arrow that is drawn on a button.

```
enum {
    kThemeArrow3pt = 0,
    kThemeArrow5pt = 1,
    kThemeArrow7pt = 2,
    kThemeArrow9pt = 3
};
typedef UInt16 ThemePopupArrowSize;
```

Constants

`kThemeArrow3pt`

Identifies a pop-up arrow with a 3-pixel base.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeArrow5pt`

Identifies a pop-up arrow with a 5-pixel base.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeArrow7pt`

Identifies a pop-up arrow with a 7-pixel base.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeArrow9pt`

Identifies a pop-up arrow with a 9-pixel base.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion
You can use a constant of type `ThemePopupArrowSize` in the function `DrawThemePopupArrow` (page 160).

Theme Checkbox Styles

Specify types of checkbox marks.

```
enum {
    kThemeCheckBoxClassicX = 0,
    kThemeCheckBoxCheckMark = 1
};
typedef UInt16 ThemeCheckBoxStyle;
```

Constants

`kThemeCheckBoxClassicX`

An “X” type of checkbox mark.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeCheckBoxCheckMark`

A checkmark type of checkbox mark.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

You can call the function `GetThemeCheckBoxStyle` (page 19) to obtain the type of checkbox mark being used in the current theme.

Theme Cursors

Identify types of cursors.

```
enum {
    kThemeArrowCursor = 0,
    kThemeCopyArrowCursor = 1,
    kThemeAliasArrowCursor = 2,
    kThemeContextualMenuArrowCursor = 3,
    kThemeIBeamCursor = 4,
    kThemeCrossCursor = 5,
    kThemePlusCursor = 6,
    kThemeWatchCursor = 7,
    kThemeClosedHandCursor = 8,
    kThemeOpenHandCursor = 9,
    kThemePointingHandCursor = 10,
    kThemeCountingUpHandCursor = 11,
    kThemeCountingDownHandCursor = 12,
    kThemeCountingUpAndDownHandCursor = 13,
    kThemeSpinningCursor = 14,
    kThemeResizeLeftCursor = 15,
    kThemeResizeRightCursor = 16,
    kThemeResizeLeftRightCursor = 17,
    kThemeNotAllowedCursor = 18,
    kThemeResizeUpCursor = 19,
    kThemeResizeDownCursor = 20,
    kThemeResizeUpDownCursor = 21,
    kThemePoofCursor = 22
};
typedef UInt32 ThemeCursor;
```

Constants

`kThemeArrowCursor`

The cursor identified by this constant is typically used as the standard cursor.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeCopyArrowCursor`

The cursor identified by this constant is typically used when the cursor is over a location where a drag action would initiate a copy.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAliasArrowCursor`

The cursor identified by this constant is typically used when the cursor is over a location where a drag action would create an alias or link.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeContextualMenuArrowCursor`

The cursor identified by this constant is typically used when the Control key is being pressed and the cursor is over a location where a contextual menu can be activated.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeIBeamCursor`

The cursor identified by this constant is typically used when the cursor is over an area where the user can select text.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeCrossCursor`

The cursor identified by this constant is typically used when the cursor is over an area where the user can draw graphics.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemePlusCursor`

The cursor identified by this constant is typically used when the cursor is over an area where the user can select table cells.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeWatchCursor`

The cursor identified by this constant is typically used to indicate that an operation is in progress. You can animate this cursor so that a hand of the watch appears to move.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeClosedHandCursor`

The cursor identified by this constant is typically used to indicate that an object has been grabbed and is being moved by the user.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeOpenHandCursor`

The cursor identified by this constant is typically used to indicate that an object may be grabbed or moved by the user.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemePointingHandCursor`

The cursor identified by this constant has the appearance of a pointing hand. You would typically use this constant to indicate that the user may select an object by pressing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeCountingUpHandCursor`

The cursor identified by this constant is typically used to indicate that an operation is in progress. You can animate this cursor so that the fingers appear to open from the palm one by one.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeCountingDownHandCursor`
: The cursor identified by this constant is typically used to indicate that an operation is in progress. You can animate this cursor so that the fingers appear to fold into the palm one by one.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeCountingUpAndDownHandCursor`
: The cursor identified by this constant is typically used to indicate that an operation is in progress. You can animate this cursor so that the fingers appear to alternate between opening from the palm one by one and folding into the palm one by one.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSpinningCursor`
: The cursor identified by this constant is typically used to indicate that an operation is in progress.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeResizeLeftCursor`
: The cursor identified by this constant is typically used to indicate that an object may be resized by dragging to the left.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeResizeRightCursor`
: The cursor identified by this constant is typically used to indicate that an object may be resized by dragging to the right.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeResizeLeftRightCursor`
: The cursor identified by this constant is typically used to indicate that an object may be resized in either direction horizontally.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeNotAllowedCursor`
: The cursor identified by this constant is typically used to indicate that the current action is not allowed. For example, you could use this cursor to indicate that an object being dragged cannot be dropped at the current mouse position.

  Available in Mac OS X v10.2 and later.

  Declared in `Appearance.h`.

`kThemeResizeUpCursor`
: The cursor identified by this constant is typically used to indicate that an object may be resized by dragging vertically in the up direction.

  Available in Mac OS X v10.3 and later.

  Declared in `Appearance.h`.

`kThemeResizeDownCursor`

The cursor identified by this constant is typically used to indicate that an object may be resized by dragging vertically in the down direction.

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemeResizeUpDownCursor`

The cursor identified by this constant is typically used to indicate that an object may be resized by dragging vertically in either direction.

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

`kThemePoofCursor`

The cursor identified by this constant is typically used to indicate that a dragged object will go away if it is released at the current mouse position. When the object goes away, a poof cloud animation should occur.

Available in Mac OS X v10.3 and later.

Declared in `Appearance.h`.

Discussion

You can pass constants of type `ThemeCursor` to the functions `SetThemeCursor` (page 28) and `SetAnimatedThemeCursor` (page 27) to specify the category of cursor to be displayed for your application. The Appearance Manager substitutes the theme-specific instance of the cursor for the cursor category as is appropriate.

Theme Font IDs

Identify types of fonts.

```
enum {
    kThemeSystemFont = 0,
    kThemeSmallSystemFont = 1,
    kThemeSmallEmphasizedSystemFont = 2,
    kThemeViewsFont = 3,
    kThemeEmphasizedSystemFont = 4,
    kThemeApplicationFont = 5,
    kThemeLabelFont = 6,
    kThemeMenuTitleFont = 100,
    kThemeMenuItemFont = 101,
    kThemeMenuItemMarkFont = 102,
    kThemeMenuItemCmdKeyFont = 103,
    kThemeWindowTitleFont = 104,
    kThemePushButtonFont = 105,
    kThemeUtilityWindowTitleFont = 106,
    kThemeAlertHeaderFont = 107,
    kThemeSystemFontDetail = 7,
    kThemeSystemFontDetailEmphasized = 8,
    kThemeCurrentPortFont = 200,
    kThemeToolbarFont = 108
};
typedef UInt16 ThemeFontID;
```

Constants

`kThemeSystemFont`

The current (large) system font. This is the font used to draw most interface elements. If you can't find a more appropriate `ThemeFontID` constant, you should use this one. This font is suitable for drawing titles on most custom widgets and buttons, as well as most static text in dialogs and windows.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSmallSystemFont`

The current small system font. This is the font used to draw interface elements when space is at a premium.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSmallEmphasizedSystemFont`

The current small, emphasized system font. This constant is identical to `kThemeSmallSystemFont`, except it draws bold or otherwise emphasized text, as is appropriate for your application's language and script.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeViewsFont`

The current views font. This is the font used to draw file and folder names in Finder windows or other browsable lists.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeEmphasizedSystemFont`

The current emphasized system font. This constant is identical to `kThemeSystemFont`, except it draws bold or otherwise emphasized text, as is appropriate for your application's language and script.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeApplicationFont`

An analog to the Script Manager's notion of the Application Font. This font is a suitable default choice for your application's document-style text editing areas.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeLabelFont`

Generally smaller than `kThemeSmallSystemFont`, this font is appropriate for drawing text labels next to image content that reinforces the meaning of the text, such as you may use with a bevel button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuTitleFont`

The font used to draw menu titles in the menu bar.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuItemFont`

The font used to draw menu items in menus.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuItemMarkFont`

The font used to draw menu item marks in menus.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuItemCmdKeyFont`

The font used to draw menu item command key equivalents in menus.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeWindowTitleFont`

The font used to draw text in most window title bars.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemePushButtonFont`

The font used to draw text labels on push buttons.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeUtilityWindowTitleFont`

The font used to draw text in utility window title bars.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAlertHeaderFont`

The font used to draw the first and most important message of an alert window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeCurrentPortFont`

Unlike the other `ThemeFontID` constants, `kThemeCurrentPortFont` doesn't map to a font appropriate to your application's language or script. It maps directly to the font, size, and style of the current QuickDraw port. This allows you to get somewhat customized behavior out of the functions that take `ThemeFontID` constants.

Note, however, that `kThemeCurrentPortFont` does not support all QuickDraw styles on all platforms; in particular, outline and shadow style are not supported on Mac OS X. Additionally, `kThemeCurrentPortFont` is not completely unicode savvy; use of `kThemeCurrentPortFont` may result in errors having to do with the current port's font not being appropriate for rendering or measuring all glyphs in a given unicode string.

Because of overhead associated with gathering QuickDraw font information and converting it to the native font format on Mac OS X, using `kThemeCurrentPortFont` may slow down your text drawing and measuring significantly compared to other `ThemeFontID` constants. Use `kThemeCurrentPortFont` only as a last resort.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeToolbarFont`

The font used to draw the label of a toolbar item.

Available in Mac OS X v10.2 and later.

Declared in `Appearance.h`.

Discussion

A `ThemeFontID` constant is a virtual font ID that you can pass to one of the Appearance Manager's text-related functions. Within these functions, the `ThemeFontID` is mapped to the appropriate font or fonts, size, and style based on a number of factors, including:

- the system appearance (Platinum on Mac OS 9 and Aqua on Mac OS X)
- the string to be rendered, if any
- the language or script that the application is running in

The `ThemeFontID` constants allow you to get the correct text appearance for the platform your application is currently running on.

kPublicThemeFontCount

The total number of public `ThemeFontID` constants.

```
enum {
    kPublicThemeFontCount = 17
};
```

Theme Text Colors

Identify the text colors appropriate to the contexts in which text is used.

```
enum {
    kThemeTextColorDialogActive = 1,
    kThemeTextColorDialogInactive = 2,
    kThemeTextColorAlertActive = 3,
    kThemeTextColorAlertInactive = 4,
    kThemeTextColorModelessDialogActive = 5,
    kThemeTextColorModelessDialogInactive = 6,
    kThemeTextColorWindowHeaderActive = 7,
    kThemeTextColorWindowHeaderInactive = 8,
    kThemeTextColorPlacardActive = 9,
    kThemeTextColorPlacardInactive = 10,
    kThemeTextColorPlacardPressed = 11,
    kThemeTextColorPushButtonActive = 12,
    kThemeTextColorPushButtonInactive = 13,
    kThemeTextColorPushButtonPressed = 14,
    kThemeTextColorBevelButtonActive = 15,
    kThemeTextColorBevelButtonInactive = 16,
    kThemeTextColorBevelButtonPressed = 17,
    kThemeTextColorPopupButtonActive = 18,
    kThemeTextColorPopupButtonInactive = 19,
    kThemeTextColorPopupButtonPressed = 20,
    kThemeTextColorIconLabel = 21,
    kThemeTextColorListView = 22

/*Text colors available in Appearance 1.0.1 and later*/
    kThemeTextColorDocumentWindowTitleActive = 23,
    kThemeTextColorDocumentWindowTitleInactive = 24,
    kThemeTextColorMovableModalWindowTitleActive = 25,
    kThemeTextColorMovableModalWindowTitleInactive = 26,
    kThemeTextColorUtilityWindowTitleActive = 27,
    kThemeTextColorUtilityWindowTitleInactive = 28,
    kThemeTextColorPopupWindowTitleActive = 29,
    kThemeTextColorPopupWindowTitleInactive = 30,
    kThemeTextColorRootMenuActive = 31,
    kThemeTextColorRootMenuSelected = 32,
    kThemeTextColorRootMenuDisabled = 33,
    kThemeTextColorMenuItemActive = 34,
    kThemeTextColorMenuItemSelected = 35,
    kThemeTextColorMenuItemDisabled = 36,
    kThemeTextColorPopupLabelActive = 37,
    kThemeTextColorPopupLabelInactive = 38

/* Text colors available in Appearamce 1.1 and later*/
    kThemeTextColorTabFrontActive = 39,
    kThemeTextColorTabNonFrontActive = 40,
    kThemeTextColorTabNonFrontPressed = 41,
    kThemeTextColorTabFrontInactive = 42,
    kThemeTextColorTabNonFrontInactive = 43,
    kThemeTextColorIconLabelSelected = 44,
    kThemeTextColorBevelButtonStickyActive = 45,
    kThemeTextColorBevelButtonStickyInactive = 46

/*Text colors available in Appearance 1.1.1 and later*/
    kThemeTextColorNotification = 47

/*Text colors available later than Mac OS X 10.1.3
    kThemeTextColorSystemDetail = 48
/*Specific colors that do not change from theme to theme*/
    kThemeTextColorBlack = -1,
    kThemeTextColorWhite = -2
};
typedef SInt16 ThemeTextColor;
```

Constants

`kThemeTextColorDialogActive`

Text color for an active dialog box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorDialogInactive`

Text color for an inactive dialog box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorAlertActive`

Text color for an active alert box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorAlertInactive`

Text color for an inactive alert box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorModelessDialogActive`

Text color for an active modeless dialog box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorModelessDialogInactive`

Text color for an inactive modeless dialog box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorWindowHeaderActive`

Text color for the window header of an active window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorWindowHeaderInactive`

Text color for the window header of an inactive window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorPlacardActive`

Text color for a placard in an active window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorPlacardInactive`
: Text color for a placard in an inactive window.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeTextColorPlacardPressed`
: Text color for a placard that is being clicked on by the user.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeTextColorPushButtonActive`
: Text color for an active push button.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeTextColorPushButtonInactive`
: Text color for an inactive push button.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeTextColorPushButtonPressed`
: Text color for a push button that is being clicked on by the user.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeTextColorBevelButtonActive`
: Text color for an active bevel button.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeTextColorBevelButtonInactive`
: Text color for an inactive bevel button.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeTextColorBevelButtonPressed`
: Text color for a bevel button that is being clicked on by the user.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeTextColorPopupButtonActive`
: Text color for the menu of an active pop-up menu button.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeTextColorPopupButtonInactive`
: Text color for the menu of an inactive pop-up menu button.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeTextColorPopupButtonPressed`
Text color for the menu of a pop-up menu button that is being clicked on by the user.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorIconLabel`
Text color for an icon label.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorListView`
Text color for the contents of a list view column.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorDocumentWindowTitleActive`
Text color for the title of an active document window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorDocumentWindowTitleInactive`
Text color for the title of an inactive document window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorMovableModalWindowTitleActive`
Text color for the title of an active movable modal window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorMovableModalWindowTitleInactive`
Text color for the title of inactive movable modal window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorUtilityWindowTitleActive`
Text color for the title of an active utility (floating) window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorUtilityWindowTitleInactive`
Text color for the title of an inactive utility (floating) window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorPopupWindowTitleActive`
Text color for the title of an active pop-up window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorPopupWindowTitleInactive`

Text color for the title of an inactive pop-up window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorRootMenuActive`

Text color for an active menu bar title.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorRootMenuSelected`

Text color for a menu bar title that is being selected by the user.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorRootMenuDisabled`

Text color for a disabled menu bar title.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorMenuItemActive`

Text color for an active menu item.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorMenuItemSelected`

Text color for a menu item that is being selected by the user.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorMenuItemDisabled`

Text color for a disabled menu item.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorPopupLabelActive`

Text color for the label of an active pop-up menu button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorPopupLabelInactive`

Text color for the label of an inactive pop-up menu button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorTabFrontActive`

Text color for the front tab of an active tab control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorTabNonFrontActive`

Text color for an active tab that is not the frontmost of a tab control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorTabNonFrontPressed`

Text color for a tab that is not the frontmost of a tab control, when the tab is being clicked on by the user.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorTabFrontInactive`

Text color for the front tab of an inactive tab control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorTabNonFrontInactive`

Text color for an inactive tab that is not the frontmost of a tab control. The tab may either be inactive because it has been individually disabled or because the tab control as a whole is currently inactive.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorIconLabelSelected`

Text color for the label of an icon that is currently selected.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorBevelButtonStickyActive`

Text color for an active bevel button that is currently on.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorBevelButtonStickyInactive`

Text color for an inactive bevel button that is currently on.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorBlack`

Black; this color does not change from theme to theme. You may use this constant instead of specifying a direct RGB value.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTextColorWhite`

White; this color does not change from theme to theme. You may use this constant instead of specifying a direct RGB value.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

You can pass a constant of type `ThemeTextColor` to the function `SetThemeTextColor` (page 207) to specify that the Appearance Manager substitute whatever the appropriate text color is for a given context under the current theme. You can use the function `GetThemeTextColor` (page 25) to obtain the actual color in use under the current theme for the specified `ThemeTextColor` constant.

Theme Menu Types

Specify a type of menu.

```
enum {
    kThemeMenuTypePullDown = 0,
    kThemeMenuTypePopUp = 1,
    kThemeMenuTypeHierarchical = 2,
    kThemeMenuTypeInactive = 0x0100
};
typedef UInt16 ThemeMenuType;
```

Constants

`kThemeMenuTypePullDown`

A pull-down menu.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuTypePopUp`

A pop-up menu.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuTypeHierarchical`

A hierarchical menu.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuTypeInactive`

An inactive menu. Add this value to any other menu type if the entire menu is inactive.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

You can pass constants of type `ThemeMenuType` in the `inMenuType` parameter of `GetThemeMenuBackgroundRegion` (page 181) and `DrawThemeMenuBackground` (page 154).

Theme Menu States

Specify the state in which theme-compliant menus are drawn.

```
enum {
    kThemeMenuActive = 0,
    kThemeMenuSelected = 1,
    kThemeMenuDisabled = 3
};
typedef UInt16 ThemeMenuState;
```

Constants

kThemeMenuActive

Menu is drawn in its active state.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeMenuSelected

Menu is drawn in its selected state.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

Discussion

You can pass constants of type ThemeMenuState in the inState parameter of DrawThemeMenuItem (page 156) and DrawThemeMenuTitle (page 158). The ThemeMenuState constants are available with Appearance Manager 1.0.1 and later.

Theme Menu Bar States

Specify whether theme-compliant menu bars are drawn as normal or selected.

```
enum {
    kThemeMenuBarNormal = 0,
    kThemeMenuBarSelected = 1
};
typedef UInt16 ThemeMenuBarState;
```

Constants

kThemeMenuBarNormal

Menu bar is drawn in its normal state.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeMenuBarSelected

Menu bar is drawn in its selected state.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

Discussion

You can pass constants of type ThemeMenuBarState in the inState parameter of DrawThemeMenuBarBackground (page 155). The ThemeMenuBarState constants are available with Appearance Manager 1.0.1 and later.

Theme Menu Item Types

Identify types of menu items.

```
enum {
    kThemeMenuItemPlain = 0,
    kThemeMenuItemHierarchical = 1,
    kThemeMenuItemScrollUpArrow = 2,
    kThemeMenuItemScrollDownArrow = 3,
    kThemeMenuItemAtTop = 0x0100,
    kThemeMenuItemAtBottom = 0x0200,
    kThemeMenuItemHierBackground = 0x0400,
    kThemeMenuItemPopUpBackground = 0x0800,
    kThemeMenuItemHasIcon = 0x8000,
    kThemeMenuItemNoBackground = 0x4000
};
typedef UInt16 ThemeMenuItemType;
```

Constants

`kThemeMenuItemPlain`

A plain menu item.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuItemHierarchical`

A hierarchical menu item.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuItemScrollUpArrow`

A scroll-up arrow.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuItemScrollDownArrow`

A scroll-down arrow.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuItemAtTop`

This value may be added to other `ThemeMenuItemType` constants to specify that the item being drawn appears at the top of the menu.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuItemAtBottom`

This value may be added to other `ThemeMenuItemType` constants to specify that the item being drawn appears at the bottom of the menu.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuItemHierBackground`

This value may be added to other `ThemeMenuItemType` constants to specify that the item being drawn is located in a hierarchical menu.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

kThemeMenuItemPopUpBackground

This value may be added to other ThemeMenuItemType constants to specify that the item being drawn is located in a pop-up menu.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeMenuItemHasIcon

This value may be added to the kThemeMenuItemPlain or kThemeMenuItemHierarchical constants, to specify that an icon is drawn along with the item text. This value may not be used with the kThemeMenuItemScrollUpArrow and kThemeMenuItemScrollDownArrow constants.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeMenuItemNoBackground

This value may be added to other ThemeMenuItemType constants to specify that the menu background should not be drawn along with the menu item. Available in Mac OS X.

Available in Mac OS X v10.1 and later.

Declared in Appearance.h.

Discussion

Your application may pass a ThemeMenuItemType constant to the function DrawThemeMenuItem (page 156) to draw a menu item of the specified type, or it may pass a ThemeMenuItemType constant to the function GetThemeMenuItemExtra (page 21) to retrieve spatial information for the given menu item type under the current theme.

kThemeMenuSquareMenuBar

Indicates that the menu bar should be drawn with square corners.

```
enum {
    kThemeMenuSquareMenuBar = (1 << 0)
};
```

Constants

kThemeMenuSquareMenuBar

Menu bar is drawn with square corners.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

Discussion

If you wish the menu bar to be drawn with square upper corners (as for a laptop system) instead of rounded ones (as for a desktop system), your application should set the bit for the attribute kThemeMenuSquareMenuBar.

Theme Scroll Bar Arrow Styles

Specify types of scroll bar arrows.

```
enum {
    kThemeScrollBarArrowsSingle = 0,
    kThemeScrollBarArrowsLowerRight = 1
};
typedef UInt16 ThemeScrollBarArrowStyle;
```

Constants

`kThemeScrollBarArrowsSingle`

Specifies the use of a single arrow at each end of a scroll bar.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeScrollBarArrowsLowerRight`

Specifies the use of double arrows at one end of a scroll bar. For vertical scroll bars, the double arrows are located at the lower end of the scroll bar. For horizontal scroll bars, the double arrows are located at the right end of the scroll bar.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

You can call the function `GetThemeScrollBarArrowStyle` (page 24) to obtain the type of scroll bar arrows being used in the current theme.

Theme Scroll Box Styles

Specify types of scroll boxes.

```
enum {
    kThemeScrollBarThumbNormal = 0,
    kThemeScrollBarThumbProportional = 1
};
typedef UInt16 ThemeScrollBarThumbStyle;
```

Constants

`kThemeScrollBarThumbNormal`

A classic scroll box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeScrollBarThumbProportional`

A proportional scroll box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

You can call the function `GetThemeScrollBarThumbStyle` (page 24) to obtain the type of scroll boxes (also known as "scroll indicators" or "thumbs") being used in the current theme.

Theme Size Box Directions

Identify the directions in which a window may be resized.

```
enum {
    kThemeGrowLeft = (1 << 0),
    kThemeGrowRight = (1 << 1),
    kThemeGrowUp = (1 << 2),
    kThemeGrowDown = (1 << 3)
};
typedef UInt16 ThemeGrowDirection;
```

Constants

`kThemeGrowLeft`

If the bit specified by this mask is set, the object can grow to the left.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeGrowRight`

If the bit specified by this mask is set, the object can grow to the right.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeGrowUp`

If the bit specified by this mask is set, the object can grow upward.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeGrowDown`

If the bit specified by this mask is set, the object can grow downward.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

The `ThemeGrowDirection` enumeration defines masks your application can use to specify the directions in which a window may be resized. You may use constants of type `ThemeGrowDirection` with the function `DrawThemeStandaloneGrowBox` (page 165) to draw a size box and with the function `GetThemeStandaloneGrowBoxBounds` (page 183) to obtain the bounding rectangle of a size box. The constants may be combined to set more than one direction of growth.

Theme Thumb Directions

Specify the direction in which the indicator points in a slider control.

```
enum {
    kThemeThumbPlain = 0,
    kThemeThumbUpward = 1,
    kThemeThumbDownward = 2
};
typedef UInt8 ThemeThumbDirection;
```

Constants

`kThemeThumbPlain`

A plain indicator; that is, one that does not point in any direction.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

kThemeThumbUpward

For a horizontal slider, an upward-pointing indicator. For a vertical slider, a left-pointing indicator.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeThumbDownward

For a horizontal slider, a downward-pointing indicator. For a vertical slider, a right-pointing indicator.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

Discussion

You can use constants of type ThemeThumbDirection in the SliderTrackInfo (page 40) structure. You may use these constants with either horizontal or vertical sliders, and the Appearance Manager interprets the direction of the indicator appropriately.

Theme Tab Directions

Specify the orientation of a tab.

```
enum {
    kThemeTabNorth = 0,
    kThemeTabSouth = 1,
    kThemeTabEast = 2,
    kThemeTabWest = 3
};
typedef UInt16 ThemeTabDirection;
```

Constants

kThemeTabNorth

An upward-pointing tab.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeTabSouth

A downward-pointing tab.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeTabEast

A right-pointing tab.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeTabWest

A left-pointing tab.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

Discussion

You can pass constants of type ThemeTabDirection to the function DrawThemeTab (page 167) to draw theme-compliant tabs that are oriented in various directions. You can also pass a ThemeTabDirection constant to the function GetThemeTabRegion (page 184) to obtain the region containing a tab oriented in a particular direction.

Theme Tab Styles

Specfiy a type of tab.

```
enum {
    kThemeTabNonFront = 0,
    kThemeTabNonFrontPressed = 1,
    kThemeTabNonFrontInactive = 2,
    kThemeTabFront = 3,
    kThemeTabFrontInactive = 4,
    kThemeTabNonFrontUnavailable = 5,
    kThemeTabFrontUnavailable = 6
};
typedef UInt16 ThemeTabStyle;
```

Constants

`kThemeTabNonFront`

An active tab that is not the frontmost in a tab control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTabNonFrontPressed`

A tab that is being clicked on by the user which is not the frontmost tab in a tab control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTabNonFrontInactive`

An inactive tab that is not the frontmost in a tab control. The tab may either be inactive because it has been individually disabled or because the tab control as a whole is currently inactive.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTabFront`

The frontmost tab in an active tab control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTabFrontInactive`

The frontmost tab in an inactive tab control.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

You can pass a constant of type `ThemeTabStyle` to the function `DrawThemeTab` (page 167) to draw a theme-compliant tab in a specific state. You can also pass a `ThemeTabStyle` constant to the function `GetThemeTabRegion` (page 184) to obtain the region containing a tab in a specific state.

Tab Heights

Specify the height of a tab.

```
enum {
    kThemeSmallTabHeight = 16,
    kThemeLargeTabHeight = 21,
    kThemeTabPaneOverlap = 3,
    kThemeSmallTabHeightMax = 19,
    kThemeLargeTabHeightMax = 24
};
```

Constants

`kThemeSmallTabHeight`

The amount that small tabs protrude from the frame.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeLargeTabHeight`

The amount that large tabs protrude from the frame.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTabPaneOverlap`

The amount that tabs overlap the frame.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSmallTabHeightMax`

The small tab height, including the overlap.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeLargeTabHeightMax`

The large tab height, including the overlap.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

Use the `kThemeSmallTabHeightMax` and `kThemeLargeTabHeightMax` constants when calculating the rectangles to draw tabs into. This height includes the tab frame overlap. Tabs that are not in the front are only drawn to where they meet the frame, as if the height was just `kThemeLargeTabHeight`, for example. Remember that for east and west tabs, the height referred to is actually the width.

Theme Track Attributes

Specify attributes of a track control.

```
enum {
    kThemeTrackHorizontal = (1 << 0),
    kThemeTrackRightToLeft = (1 << 1),
    kThemeTrackShowThumb = (1 << 2),
    kThemeTrackThumbRgnIsNotGhost = (1 << 3),
    kThemeTrackNoScrollBarArrows = (1 << 4),
    kThemeTrackHasFocus = (1 << 5)
};
typedef UInt16 ThemeTrackAttributes;
```

Constants

`kThemeTrackHorizontal`

If the bit specified by this mask is set, the track is horizontally, not vertically, oriented.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTrackRightToLeft`

If the bit specified by this mask is set, values for the track increase from right to left if the track is horizontally oriented, or from bottom to top if the track is vertically oriented.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTrackShowThumb`

If the bit specified by this mask is set, an indicator is drawn for this track.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTrackThumbRgnIsNotGhost`

If the bit specified by this mask is set, the thumb region is drawn opaque, rather than as a ghost.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTrackNoScrollBarArrows`

If the bit specified by this mask is set, the track scroll bar is drawn without arrows. This attribute currently has no effect.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTrackHasFocus`

If the bit specified by this mask is set, the thumb has focus. This attribute currently has effect only on sliders.

Available in Mac OS X v10.2 and later.

Declared in `Appearance.h`.

Discussion

The `ThemeTrackAttributes` enumeration defines masks your application can use in the `ThemeTrackDrawInfo` (page 41) structure to specify various attributes of track controls.

Theme Track States

Specify the state of a track control.

```
enum {
    kThemeTrackActive = 0,
    kThemeTrackDisabled = 1,
    kThemeTrackNothingToScroll = 2,
    kThemeTrackInactive = 3
};
typedef UInt8 ThemeTrackEnableState;
```

Constants

`kThemeTrackActive`

A track in the active state.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTrackDisabled`

A track in the disabled state.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTrackNothingToScroll`

For scroll bars, the window containing the track is expanded to a sufficiently large state such that all the content is viewable and there is nothing remaining to scroll.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

You can use constants of type `ThemeTrackEnableState` in the `ThemeTrackDrawInfo` (page 41) structure and in the functions `GetThemeScrollBarTrackRect` (page 182) and `HitTestThemeScrollBarArrows` (page 192).

Theme Track Kinds

Identify specific kinds of track-based controls to the Appearance Manager.

```
enum {
    kThemeMediumScrollBar = 0,
    kThemeSmallScrollBar = 1,
    kThemeMediumSlider = 2,
    kThemeMediumProgressBar = 3,
    kThemeMediumIndeterminateBar = 4,
    kThemeRelevanceBar = 5,
    kThemeSmallSlider = 6,
    kThemeLargeProgressBar = 7,
    kThemeLargeIndeterminateBar = 8
};
typedef UInt16 ThemeTrackKind;
```

Constants

`kThemeMediumScrollBar`

A scroll bar.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSmallScrollBar`
A small scroll bar.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMediumSlider`
A slider bar.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMediumProgressBar`
A progress bar.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMediumIndeterminateBar`
An indeterminate progress bar.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Theme Track Press States

Identify what is pressed in an active scroll bar or slider.

```
enum {
    kThemeLeftOutsideArrowPressed = 0x01,
    kThemeLeftInsideArrowPressed = 0x02,
    kThemeLeftTrackPressed = 0x04,
    kThemeThumbPressed = 0x08,
    kThemeRightTrackPressed = 0x10,
    kThemeRightInsideArrowPressed = 0x20,
    kThemeRightOutsideArrowPressed = 0x40,
    kThemeTopOutsideArrowPressed = 0x01,
    kThemeTopInsideArrowPressed = 0x02,
    kThemeTopTrackPressed = 0x04,
    kThemeBottomTrackPressed = 0x10,
    kThemeBottomInsideArrowPressed = 0x20,
    kThemeBottomOutsideArrowPressed = 0x40
};
typedef UInt8 ThemeTrackPressState;
```

Constants

`kThemeLeftOutsideArrowPressed`
For a horizontal scroll bar containing a single pair of arrows, this constant indicates that the arrow on the left is selected.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeLeftInsideArrowPressed`

For a horizontal scroll bar containing a single pair of arrows, this constant should not be used. For a horizontal scroll bar containing two pairs of arrows with one pair at each end, this constant indicates that the inner arrow at the left end of the scroll bar is selected.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeLeftTrackPressed`

For a horizontal scroll bar or slider, indicates that the end of the track to the left of the scroll box or indicator is selected.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeThumbPressed`

Indicates that the scroll box or indicator is selected.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeRightTrackPressed`

For a horizontal scroll bar or slider, indicates that the end of the track to the right of the scroll box or indicator is selected.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeRightInsideArrowPressed`

For a horizontal scroll bar containing a single pair of arrows, this constant should not be used. For a horizontal scroll bar containing two pairs of arrows with one pair at each end, this constant indicates that the inner arrow at the right end of the scroll bar is selected.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeRightOutsideArrowPressed`

For a horizontal scroll bar containing a single pair of arrows, this constant indicates that the arrow on the right is selected.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTopOutsideArrowPressed`

For a vertical scroll bar containing a single pair of arrows, this constant indicates that the arrow on the top is selected.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTopInsideArrowPressed`

For a vertical scroll bar containing a single pair of arrows, this constant should not be used. For a vertical scroll bar containing two pairs of arrows with one pair at each end, this constant indicates that the inner arrow at the top end of the scroll bar is selected.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeTopTrackPressed`
: For a vertical scroll bar or slider, indicates that the end of the track above the scroll box or indicator is selected.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeBottomTrackPressed`
: For a vertical scroll bar or slider, indicates that the end of the track beneath the scroll box or indicator is selected.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeBottomInsideArrowPressed`
: For a vertical scroll bar containing a single pair of arrows, this constant should not be used. For a vertical scroll bar containing two pairs of arrows with one pair at each end, this constant indicates that the inner arrow at the bottom end of the scroll bar is selected.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeBottomOutsideArrowPressed`
: For a vertical scroll bar containing a single pair of arrows, this constant indicates that the arrow on the bottom is selected.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

Discussion

You can use constants of type `ThemeTrackPressState` in structures of type `ScrollBarTrackInfo` (page 39) and `SliderTrackInfo` (page 40) to identify what is pressed in an active scroll bar or slider; the press state is ignored if the control is not active.

Note that some constants are undefined when the corresponding arrows do not exist in the current visual appearance. Prior to using the `ThemeTrackPressState` constants, your application should call the function `GetThemeScrollBarArrowStyle` (page 24) to obtain the type of scroll bar arrows currently being used.

Theme Window Types

Identify windows of specific visual categories.

```
enum {
    kThemeDocumentWindow = 0,
    kThemeDialogWindow = 1,
    kThemeMovableDialogWindow = 2,
    kThemeAlertWindow = 3,
    kThemeMovableAlertWindow = 4,
    kThemePlainDialogWindow = 5,
    kThemeShadowDialogWindow = 6,
    kThemePopupWindow = 7,
    kThemeUtilityWindow = 8,
    kThemeUtilitySideWindow = 9,
    kThemeSheetWindow = 10,
    kThemeDrawerWindow = 11
};
typedef UInt16 ThemeWindowType;
```

Constants

`kThemeDocumentWindow`

A document window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDialogWindow`

A modal dialog box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMovableDialogWindow`

A movable modal dialog box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeAlertWindow`

An alert box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMovableAlertWindow`

A movable alert box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemePlainDialogWindow`

A plain modal dialog box. This window visually corresponds to that produced by the `plainDBox` pre–Appearance Manager window definition ID and does not change appearance by theme.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeShadowDialogWindow`

A dialog box with shadowing.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

kThemePopupWindow

A pop-up window.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeUtilityWindow

A utility window.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeUtilitySideWindow

A utility window with a side title bar.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

Theme Window Attributes

Specify interface elements in a window.

```
enum {
    kThemeWindowHasGrow = (1 << 0),
    kThemeWindowHasHorizontalZoom = (1 << 3),
    kThemeWindowHasVerticalZoom = (1 << 4),
    kThemeWindowHasFullZoom = kThemeWindowHasHorizontalZoom +
kThemeWindowHasVerticalZoom,
    kThemeWindowHasCloseBox = (1 << 5),
    kThemeWindowHasCollapseBox = (1 << 6),
    kThemeWindowHasTitleText = (1 << 7),
    kThemeWindowIsCollapsed = (1 << 8),
    kThemeWindowHasDirty = (1 << 9)
};
typedef UInt32 ThemeWindowAttributes;
```

Constants

kThemeWindowHasGrow

If the bit specified by this mask is set, the window contains a size box.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeWindowHasHorizontalZoom

If the bit specified by this mask is set, the window contains a horizontal zoom box.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

kThemeWindowHasVerticalZoom

If the bit specified by this mask is set, the window contains a vertical zoom box.

Available in Mac OS X v10.0 and later.

Declared in Appearance.h.

`kThemeWindowHasFullZoom`
: If the bit specified by this mask is set, the window contains a full (horizontal and vertical) zoom box.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeWindowHasCloseBox`
: If the bit specified by this mask is set, the window contains a close box.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeWindowHasCollapseBox`
: If the bit specified by this mask is set, the window contains a collapse box.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeWindowHasTitleText`
: If the bit specified by this mask is set, the window contains a title.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeWindowIsCollapsed`
: If the bit specified by this mask is set, the window is currently collapsed.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

Discussion

The `ThemeWindowAttributes` enumeration defines masks your application can use to specify the various interface elements that a given window contains.

Theme Title Bar Items

Identify specific types of window title bar items.

```
enum {
    kThemeWidgetCloseBox = 0,
    kThemeWidgetZoomBox = 1,
    kThemeWidgetCollapseBox = 2,
    kThemeWidgetDirtyCloseBox = 6
};
typedef UInt16 ThemeTitleBarWidget;
```

Constants

`kThemeWidgetCloseBox`
: Identifies a close box.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeWidgetZoomBox`
: Identifies a zoom box.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeWidgetCollapseBox`

Identifies a collapse box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

You may pass constants of type `ThemeTitleBarWidget` to the function `DrawThemeTitleBarWidget` (page 171) to draw specific types of window title bar items. The Appearance Manager draws a theme-compliant version of the title bar item type, as is appropriate.

Pop-up Window Tab Positions

Indicate how a pop-up window's tab is positioned.

```
enum {
    kThemePopupTabNormalPosition = 0,
    kThemePopupTabCenterOnWindow = 1,
    kThemePopupTabCenterOnOffset = 2
};
```

Constants

`kThemePopupTabNormalPosition`

Specifies that the left edge of the tab is to be drawn at the position indicated by the `popupTabOffset` field of the `ThemeWindowMetrics` structure.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemePopupTabCenterOnWindow`

Specifies that the tab is to be drawn centered on the window; the `popupTabOffset` field is ignored.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemePopupTabCenterOnOffset`

Specifies that the tab is to be drawn centered at the position indicated by the `popupTabOffset` field.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

You can use these constants in the `popupTabPosition` field of the `ThemeWindowMetrics` structure.

Theme Sound Masks

Define masks that are used to specify the classes of sounds that are enabled for a theme.

```
enum {
    kThemeNoSounds = 0,
    kThemeWindowSoundsMask = (1 << 0),
    kThemeMenuSoundsMask = (1 << 1),
    kThemeControlSoundsMask = (1 << 2),
    kThemeFinderSoundsMask = (1 << 3)
};
```

Constants

`kThemeNoSounds`

If no bits are set, no theme sounds are enabled.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeWindowSoundsMask`

If the bit specified by this mask is set, window sounds are enabled.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeMenuSoundsMask`

If the bit specified by this mask is set, menu sounds are enabled.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeControlSoundsMask`

If the bit specified by this mask is set, control sounds are enabled.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeFinderSoundsMask`

If the bit specified by this mask is set, Finder sounds are enabled.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

You can use these masks to operate upon the unsigned 32-bit integer contained in the `kThemeSoundMaskTag` collection item, which is described in "Theme Collection Tags" (page 47).

Theme Sounds

Identify theme-specific sounds played when an interface object changes state.

```
enum {
    kThemeSoundNone = 0,
    kThemeSoundMenuOpen = 'mnuo',
    kThemeSoundMenuClose = 'mnuc',
    kThemeSoundMenuItemHilite = 'mnui',
    kThemeSoundMenuItemRelease = 'mnus',
    kThemeSoundWindowClosePress = 'wclp',
    kThemeSoundWindowCloseEnter = 'wcle',
    kThemeSoundWindowCloseExit = 'wclx',
    kThemeSoundWindowCloseRelease = 'wclr',
    kThemeSoundWindowZoomPress = 'wzmp',
    kThemeSoundWindowZoomEnter = 'wzme',
    kThemeSoundWindowZoomExit = 'wzmx',
    kThemeSoundWindowZoomRelease = 'wzmr',
    kThemeSoundWindowCollapsePress = 'wcop',
    kThemeSoundWindowCollapseEnter = 'wcoe',
    kThemeSoundWindowCollapseExit = 'wcox',
    kThemeSoundWindowCollapseRelease = 'wcor',
    kThemeSoundWindowDragBoundary = 'wdbd',
    kThemeSoundUtilWinClosePress = 'uclp',
    kThemeSoundUtilWinCloseEnter = 'ucle',
    kThemeSoundUtilWinCloseExit = 'uclx',
    kThemeSoundUtilWinCloseRelease = 'uclr',
    kThemeSoundUtilWinZoomPress = 'uzmp',
    kThemeSoundUtilWinZoomEnter = 'uzme',
    kThemeSoundUtilWinZoomExit = 'uzmx',
    kThemeSoundUtilWinZoomRelease = 'uzmr',
    kThemeSoundUtilWinCollapsePress = 'ucop',
    kThemeSoundUtilWinCollapseEnter = 'ucoe',
    kThemeSoundUtilWinCollapseExit = 'ucox',
    kThemeSoundUtilWinCollapseRelease = 'ucor',
    kThemeSoundUtilWinDragBoundary = 'udbd',
    kThemeSoundWindowOpen = 'wopn',
    kThemeSoundWindowClose = 'wcls',
    kThemeSoundWindowZoomIn = 'wzmi',
    kThemeSoundWindowZoomOut = 'wzmo',
    kThemeSoundWindowCollapseUp = 'wcol',
    kThemeSoundWindowCollapseDown = 'wexp',
    kThemeSoundWindowActivate = 'wact',
    kThemeSoundUtilWindowOpen = 'uopn',
    kThemeSoundUtilWindowClose = 'ucls',
    kThemeSoundUtilWindowZoomIn = 'uzmi',
    kThemeSoundUtilWindowZoomOut = 'uzmo',
    kThemeSoundUtilWindowCollapseUp = 'ucol',
    kThemeSoundUtilWindowCollapseDown = 'uexp',
    kThemeSoundUtilWindowActivate = 'uact',
    kThemeSoundDialogOpen = 'dopn',
    kThemeSoundDialogClose = 'dlgc',
    kThemeSoundAlertOpen = 'aopn',
    kThemeSoundAlertClose = 'altc',
    kThemeSoundPopupWindowOpen = 'pwop',
    kThemeSoundPopupWindowClose = 'pwcl',
    kThemeSoundButtonPress = 'btnp',
    kThemeSoundButtonEnter = 'btne',
    kThemeSoundButtonExit = 'btnx',
    kThemeSoundButtonRelease = 'btnr',
    kThemeSoundDefaultButtonPress = 'dbtp',
    kThemeSoundDefaultButtonEnter = 'dbte',
    kThemeSoundDefaultButtonExit = 'dbtx',
    kThemeSoundDefaultButtonRelease = 'dbtr',
    kThemeSoundCancelButtonPress = 'cbtp',
    kThemeSoundCancelButtonEnter = 'cbte',
    kThemeSoundCancelButtonExit = 'cbtx',
    kThemeSoundCancelButtonRelease = 'cbtr',
    kThemeSoundCheckboxPress = 'chkp',
    kThemeSoundCheckboxEnter = 'chke',
    kThemeSoundCheckboxExit = 'chkx',
    kThemeSoundCheckboxRelease = 'chkr',
    kThemeSoundRadioPress = 'radp',
    kThemeSoundRadioEnter = 'rade',
    kThemeSoundRadioExit = 'radx',
    kThemeSoundRadioRelease = 'radr',
    kThemeSoundScrollArrowPress = 'sbap',
    kThemeSoundScrollArrowEnter = 'sbae',
    kThemeSoundScrollArrowExit = 'sbax',
    kThemeSoundScrollArrowRelease = 'sbar',
    kThemeSoundScrollEndOfTrack = 'sbte',
    kThemeSoundScrollTrackPress = 'sbtp',
    kThemeSoundSliderEndOfTrack = 'slte',
    kThemeSoundSliderTrackPress = 'sltp',
    kThemeSoundBalloonOpen = 'blno',
    kThemeSoundBalloonClose = 'blnc',
    kThemeSoundBevelPress = 'bevp',
    kThemeSoundBevelEnter = 'beve',
    kThemeSoundBevelExit = 'bevx',
    kThemeSoundBevelRelease = 'bevr',
    kThemeSoundLittleArrowUpPress = 'laup',
    kThemeSoundLittleArrowDnPress = 'ladp',
    kThemeSoundLittleArrowEnter = 'lare',
    kThemeSoundLittleArrowExit = 'larx',
    kThemeSoundLittleArrowUpRelease = 'laur',
    kThemeSoundLittleArrowDnRelease = 'ladr',
    kThemeSoundPopupPress = 'popp',
    kThemeSoundPopupEnter = 'pope',
    kThemeSoundPopupExit = 'popx',
    kThemeSoundPopupRelease = 'popr',
    kThemeSoundDisclosurePress = 'dscp',
    kThemeSoundDisclosureEnter = 'dsce',
    kThemeSoundDisclosureExit = 'dscx',
    kThemeSoundDisclosureRelease = 'dscr',
    kThemeSoundTabPressed = 'tabp',
    kThemeSoundTabEnter = 'tabe',
    kThemeSoundTabExit = 'tabx',
    kThemeSoundTabRelease = 'tabr',
    kThemeSoundDragTargetHilite = 'dthi',
    kThemeSoundDragTargetUnhilite = 'dtuh',
    kThemeSoundDragTargetDrop = 'dtdr',
    kThemeSoundEmptyTrash = 'ftrs',
    kThemeSoundSelectItem = 'fsel',
    kThemeSoundNewItem = 'fnew',
    kThemeSoundReceiveDrop = 'fdrp',
    kThemeSoundCopyDone = 'fcpd',
    kThemeSoundResolveAlias = 'fral',
    kThemeSoundLaunchApp = 'flap',
    kThemeSoundDiskInsert = 'dski',
    kThemeSoundDiskEject = 'dske',
    kThemeSoundFinderDragOnIcon = 'fdon',
    kThemeSoundFinderDragOffIcon = 'fdof'
};
typedef OSType ThemeSoundKind;
```

Constants

`kThemeSoundNone`

Specifies that no sound is played.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundMenuOpen`

Identifies a sound to be played when the user opens a menu.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundMenuClose`

Identifies a sound to be played when the user closes a menu.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundMenuItemHilite`

Identifies a sound to be played when the user highlights a menu item.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundMenuItemRelease`

Identifies a sound to be played when the user selects a menu item.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowClosePress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a window's close box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowCloseEnter`

Identifies a sound to be played when the user moves the cursor over a window's close box after having moved the cursor away from the close box without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowCloseExit`

Identifies a sound to be played when the user moves the cursor away from a position over a window's close box, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowCloseRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a window's close box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowZoomPress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a window's zoom box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowZoomEnter`

Identifies a sound to be played when the user moves the cursor over a window's zoom box after having moved the cursor away from the zoom box without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowZoomExit`

Identifies a sound to be played when the user moves the cursor away from a position over a window's zoom box, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowZoomRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a window's zoom box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowCollapsePress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a window's collapse box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowCollapseEnter`

Identifies a sound to be played when the user moves the cursor over a window's collapse box after having moved the cursor away from the collapse box without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowCollapseExit`

Identifies a sound to be played when the user moves the cursor away from a position over a window's collapse box, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowCollapseRelease`
: Identifies a sound to be played when the user releases the mouse button while the cursor is over a window's collapse box.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundWindowDragBoundary`
: Identifies a sound to be played when the user drags a window to the edge of the area where it can be dragged. Note: This functionality is not available under Appearance Manager 1.1 or prior versions of Appearance.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundUtilWinClosePress`
: Identifies a sound to be played when the user presses the mouse button while the cursor is over a utility (floating) window's close box.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundUtilWinCloseEnter`
: Identifies a sound to be played when the user moves the cursor over a utility (floating) window's close box after having moved the cursor away from the close box without releasing the mouse button.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundUtilWinCloseExit`
: Identifies a sound to be played when the user moves the cursor away from a position over a utility (floating) window's close box, while the mouse button remains pressed.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundUtilWinCloseRelease`
: Identifies a sound to be played when the user releases the mouse button while the cursor is over a utility (floating) window's close box.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundUtilWinZoomPress`
: Identifies a sound to be played when the user presses the mouse button while the cursor is over a utility (floating) window's zoom box.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundUtilWinZoomEnter`
: Identifies a sound to be played when the user moves the cursor over a utility (floating) window's zoom box after having moved the cursor away from the zoom box without releasing the mouse button.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundUtilWinZoomExit`

Identifies a sound to be played when the user moves the cursor away from a position over a utility (floating) window's zoom box, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundUtilWinZoomRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a utility (floating) window's zoom box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundUtilWinCollapsePress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a utility (floating) window's collapse box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundUtilWinCollapseEnter`

Identifies a sound to be played when the user moves the cursor over a utility (floating) window's collapse box after having moved the cursor away from the collapse box without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundUtilWinCollapseExit`

Identifies a sound to be played when the user moves the cursor away from a position over a utility (floating) window's collapse box, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundUtilWinCollapseRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a utility (floating) window's collapse box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundUtilWinDragBoundary`

Identifies a sound to be played when the user drags a utility (floating) window to the edge of the area where it can be dragged.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowOpen`

Identifies a sound to be played when the user opens a window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowClose`

Identifies a sound to be played when the user closes a window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundWindowZoomIn`
: Identifies a sound to be played when the user zooms a window in, that is, to the user state.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundWindowZoomOut`
: Identifies a sound to be played when the user zooms a window out, that is, to the standard state.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundWindowCollapseUp`
: Identifies a sound to be played when the user collapses a window.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundWindowCollapseDown`
: Identifies a sound to be played when the user uncollapses a window.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundWindowActivate`
: Identifies a sound to be played when the user presses the mouse button while the cursor is over an inactive window, thus activating it.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundUtilWindowOpen`
: Identifies a sound to be played when the user opens a utility (floating) window.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundUtilWindowClose`
: Identifies a sound to be played when the user closes a utility (floating) window.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundUtilWindowZoomIn`
: Identifies a sound to be played when the user zooms a utility (floating) window in, that is, to the user state.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundUtilWindowZoomOut`
: Identifies a sound to be played when the user zooms a utility (floating) window out, that is, to the standard state.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundUtilWindowCollapseUp`
: Identifies a sound to be played when the user collapses a utility (floating) window.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeSoundUtilWindowCollapseDown`

Identifies a sound to be played when the user uncollapses a utility (floating) window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundUtilWindowActivate`

Identifies a sound to be played when the user presses the mouse button while the cursor is over an inactive utility (floating) window, thus activating it.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDialogOpen`

Identifies a sound to be played when a dialog box opens.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDialogClose`

Identifies a sound to be played when a dialog box closes.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundAlertOpen`

Identifies a sound to be played when an alert box opens.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundAlertClose`

Identifies a sound to be played when an alert box closes.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundPopupWindowOpen`

Identifies a sound to be played when a pop-up window opens.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundPopupWindowClose`

Identifies a sound to be played when a pop-up window closes.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundButtonPress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a push button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundButtonEnter`

Identifies a sound to be played when the user moves the cursor over a push button after having moved the cursor away from the button without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundButtonExit`

Identifies a sound to be played when the user moves the cursor away from a position over a push button, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundButtonRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a push button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDefaultButtonPress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a default button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDefaultButtonEnter`

Identifies a sound to be played when the user moves the cursor over a default button after having moved the cursor away from the button without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDefaultButtonExit`

Identifies a sound to be played when the user moves the cursor away from a position over a default button, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDefaultButtonRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a default button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundCancelButtonPress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a Cancel button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundCancelButtonEnter`

Identifies a sound to be played when the user moves the cursor over a Cancel button after having moved the cursor away from the button without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundCancelButtonExit`

Identifies a sound to be played when the user moves the cursor away from a position over a Cancel button, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundCancelButtonRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a Cancel button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundCheckboxPress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a checkbox.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundCheckboxEnter`

Identifies a sound to be played when the user moves the cursor over a checkbox after having moved the cursor away from the checkbox without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundCheckboxExit`

Identifies a sound to be played when the user moves the cursor away from a position over a checkbox, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundCheckboxRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a checkbox.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundRadioPress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a radio button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundRadioEnter`

Identifies a sound to be played when the user moves the cursor over a radio button after having moved the cursor away from the radio button without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundRadioExit`

Identifies a sound to be played when the user moves the cursor away from a position over a radio button, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundRadioRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a radio button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundScrollArrowPress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a scroll bar arrow.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundScrollArrowEnter`

Identifies a sound to be played when the user moves the cursor over a scroll bar arrow after having moved the cursor away from the arrow without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundScrollArrowExit`

Identifies a sound to be played when the user moves the cursor away from a position over a scroll bar arrow, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundScrollArrowRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a scroll bar arrow.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundScrollEndOfTrack`

Identifies a sound to be played when a scroll box arrives at the end of a scroll bar and can go no further.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundScrollTrackPress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over the track part of a scroll bar (this area does not include the scroll box or scroll bar arrows).

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundSliderEndOfTrack`

Identifies a sound to be played when a slider indicator arrives at the end of a slider track and can go no further. Note: This functionality is not available under Appearance Manager 1.1 or prior versions of Appearance.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundSliderTrackPress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over the track part of a slider (this area does not include the slider indicator).

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundBalloonOpen`

Identifies a sound to be played when a help balloon appears.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundBalloonClose`

Identifies a sound to be played when a help balloon disappears.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundBevelPress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a bevel button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundBevelEnter`

Identifies a sound to be played when the user moves the cursor over a bevel button after having moved the cursor away from the bevel button without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundBevelExit`

Identifies a sound to be played when the user moves the cursor away from a position over a bevel button, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundBevelRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a bevel button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundLittleArrowUpPress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over the upward-pointing arrow of an increment/decrement button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundLittleArrowDnPress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over the downward-pointing arrow of an increment/decrement button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundLittleArrowEnter`

Identifies a sound to be played when the user moves the cursor over an increment/decrement button after having moved the cursor away from the button without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundLittleArrowExit`

Identifies a sound to be played when the user moves the cursor away from a position over an increment/decrement button, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundLittleArrowUpRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over the upward-pointing arrow of an increment/decrement button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundLittleArrowDnRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over the downward-pointing arrow of an increment/decrement button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundPopupPress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a pop-up menu button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundPopupEnter`

Identifies a sound to be played when the user moves the cursor over a pop-up menu button after having moved the cursor away from the button without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundPopupExit`

Identifies a sound to be played when the user moves the cursor away from a position over a pop-up menu button, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundPopupRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a pop-up menu button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDisclosurePress`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a disclosure triangle.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDisclosureEnter`

Identifies a sound to be played when the user moves the cursor over a disclosure triangle after having moved the cursor away from the disclosure triangle without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDisclosureExit`

Identifies a sound to be played when the user moves the cursor away from a position over a disclosure triangle, while the mouse button remains pressed.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDisclosureRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a disclosure triangle.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundTabPressed`

Identifies a sound to be played when the user presses the mouse button while the cursor is over a tab.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundTabEnter`

Identifies a sound to be played when the user places the cursor over a tab.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundTabExit`

Identifies a sound to be played when the user moves the cursor over a tab after having moved the cursor away from the tab without releasing the mouse button.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundTabRelease`

Identifies a sound to be played when the user releases the mouse button while the cursor is over a tab.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDragTargetHilite`

Identifies a sound to be played when the user drags an object over a valid drag-and-drop destination.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDragTargetUnhilite`

Identifies a sound to be played when the user drags an object away from a valid drag-and-drop destination.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDragTargetDrop`

Identifies a sound to be played when the user drops an object on a valid drag-and-drop destination.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundEmptyTrash`

Identifies a sound to be played when the Finder completes emptying the Trash directory.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundSelectItem`

Identifies a sound to be played when the user presses the mouse button while the cursor is over an item in the Finder.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundNewItem`

Identifies a sound to be played when the user creates a new item.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundReceiveDrop`

Identifies a sound to be played when a Finder object changes parents, such as when the user drops an icon on a folder.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundCopyDone`

Identifies a sound to be played when the Finder completes a copy operation.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundResolveAlias`

Identifies a sound to be played when the Finder resolves an alias.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundLaunchApp`

Identifies a sound to be played when the Finder launches an application.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDiskInsert`
Identifies a sound to be played when a disk is inserted.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundDiskEject`
Identifies a sound to be played when a disk is ejected.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundFinderDragOnIcon`
Identifies a sound to be played when the user drags an object over an icon.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeSoundFinderDragOffIcon`
Identifies a sound to be played when the user drags an object off of an icon.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

Discussion

Your application can pass constants of type `ThemeSoundKind` to the function `PlayThemeSound` (page 26) to play a theme-specific sound for an interface object when it changes state. Each sound plays asynchronously until complete, then stops automatically.

Theme Drag Sounds

Identify theme-specific sounds played when the user performs a drag.

```
enum {
    kThemeDragSoundNone = 0,
    kThemeDragSoundMoveWindow = 'wmov',
    kThemeDragSoundGrowWindow = 'wgro',
    kThemeDragSoundMoveUtilWindow = 'umov',
    kThemeDragSoundGrowUtilWindow = 'ugro',
    kThemeDragSoundMoveDialog = 'dmov',
    kThemeDragSoundMoveAlert = 'amov',
    kThemeDragSoundMoveIcon = 'imov',
    kThemeDragSoundSliderThumb = 'slth',
    kThemeDragSoundSliderGhost = 'slgh',
    kThemeDragSoundScrollBarThumb = 'sbth',
    kThemeDragSoundScrollBarGhost = 'sbgh',
    kThemeDragSoundScrollBarArrowDecreasing = 'sbad',
    kThemeDragSoundScrollBarArrowIncreasing = 'sbai',
    kThemeDragSoundDragging = 'drag'
};
typedef OSType ThemeDragSoundKind;
```

Constants

`kThemeDragSoundNone`
Specifies that no drag sound is used.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDragSoundMoveWindow`
Specifies a sound to be played when the user moves a document window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDragSoundGrowWindow`
Specifies a sound to be played when the user resizes a window by dragging the size box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDragSoundMoveUtilWindow`
Specifies a sound to be played when the user moves a utility window.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDragSoundGrowUtilWindow`
Specifies a sound to be played when the user resizes a utility window by dragging the size box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDragSoundMoveDialog`
Specifies a sound to be played when the user moves a dialog box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDragSoundMoveAlert`
Specifies a sound to be played when the user moves an alert box.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDragSoundMoveIcon`
Specifies a sound to be played when the user moves an icon.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDragSoundSliderThumb`
Specifies a sound to be played when the user drags the indicator of a slider control that supports live feedback.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDragSoundSliderGhost`
Specifies a sound to be played when the user drags the indicator of a slider control that does not support live feedback.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDragSoundScrollBarThumb`
Specifies a sound to be played when the user drags a scroll box belonging to a scroll bar that supports live feedback.

Available in Mac OS X v10.0 and later.

Declared in `Appearance.h`.

`kThemeDragSoundScrollBarGhost`
: Specifies a sound to be played when the user drags a scroll box belonging to a scroll bar that does not support live feedback.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeDragSoundScrollBarArrowDecreasing`
: Specifies a sound to be played when the user presses and holds the mouse button while the cursor is over the scroll bar arrow that decreases the scroll bar's value.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeDragSoundScrollBarArrowIncreasing`
: Specifies a sound to be played when the user presses and holds the mouse button while the cursor is over the scroll bar arrow that increases the scroll bar's value.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

`kThemeDragSoundDragging`
: Specifies a sound to be played during a Drag Manager drag.

  Available in Mac OS X v10.0 and later.

  Declared in `Appearance.h`.

Discussion

Your application can pass constants of type `ThemeDragSoundKind` to the function `BeginThemeDragSound` (page 16) to play a theme-specific sound when a user drags an interface object or otherwise holds the mouse button down for an extended action. Dragging sounds are looped for the duration of the drag and cease when your application calls `EndThemeDragSound` (page 17) when the drag has finished. Only one drag sound may occur at a time.

Desktop Picture Alignments

Represent picture alignments that might be reported in the data for the `kThemeDesktopPictureAlignmentTag` theme collection tag.

```
enum {
    kTiledOnScreen = 1,
    kCenterOnScreen = 2,
    kFitToScreen = 3,
    kFillScreen = 4,
    kUseBestGuess = 5
};
```

Constants

`kTiledOnScreen`
: The picture draws repeatedly.

  Available in Mac OS X v10.2 and later.

  Declared in `Appearance.h`.

`kCenterOnScreen`

The picture is its actual size, or clipped if necessary, with the desktop pattern showing to the side of the picture if it is smaller than the desktop

Available in Mac OS X v10.2 and later.

Declared in `Appearance.h`.

`kFitToScreen`

The picture is reduced if necessary.

Available in Mac OS X v10.2 and later.

Declared in `Appearance.h`.

`kFillScreen`

The picture's aspect ratio is altered if necessary.

Available in Mac OS X v10.2 and later.

Declared in `Appearance.h`.

`kUseBestGuess`

The picture is automatically positioned, based on picture and monitor sizes.

Available in Mac OS X v10.2 and later.

Declared in `Appearance.h`.

appearanceBadBrushIndexErr

Obsolete error codes. Use the result codes listed in the section "Appearance Manager Result Codes" (page 142) instead.

```
enum {
    appearanceBadBrushIndexErr = -30560,
    appearanceProcessRegisteredErr = -30561,
    appearanceProcessNotRegisteredErr = -30562,
    appearanceBadTextColorIndexErr = -30563,
    appearanceThemeHasNoAccents = -30564,
    appearanceBadCursorIndexErr = -30565
};
```

kAEThemeSwitch

Obsolete. Use `kAEAppearanceChanged` , described in "Appearance Manager Apple Events," instead.

```
enum {
    kAEThemeSwitch = kAEAppearanceChanged
};
```

kThemeActiveDialogBackgroundBrush

Obsolete. Use the constants described in "Theme Brushes" instead.

```
enum {
    kThemeActiveDialogBackgroundBrush = kThemeBrushDialogBackgroundActive,
    kThemeInactiveDialogBackgroundBrush =
            kThemeBrushDialogBackgroundInactive,
    kThemeActiveAlertBackgroundBrush = kThemeBrushAlertBackgroundActive,
    kThemeInactiveAlertBackgroundBrush = kThemeBrushAlertBackgroundInactive,
    kThemeActiveModelessDialogBackgroundBrush =
            kThemeBrushModelessDialogBackgroundActive,
    kThemeInactiveModelessDialogBackgroundBrush =
        kThemeBrushModelessDialogBackgroundInactive,
    kThemeActiveUtilityWindowBackgroundBrush =
        kThemeBrushUtilityWindowBackgroundActive,
    kThemeInactiveUtilityWindowBackgroundBrush =
        kThemeBrushUtilityWindowBackgroundInactive,
    kThemeListViewSortColumnBackgroundBrush =
        kThemeBrushListViewSortColumnBackground,
    kThemeListViewBackgroundBrush = kThemeBrushListViewBackground,
    kThemeIconLabelBackgroundBrush = kThemeBrushIconLabelBackground,
    kThemeListViewSeparatorBrush = kThemeBrushListViewSeparator,
    kThemeChasingArrowsBrush = kThemeBrushChasingArrows,
    kThemeDragHiliteBrush = kThemeBrushDragHilite,
    kThemeDocumentWindowBackgroundBrush =
        kThemeBrushDocumentWindowBackground,
    kThemeFinderWindowBackgroundBrush = kThemeBrushFinderWindowBackground
};
```

kThemeActiveScrollBarDelimiterBrush

Obsolete. Use the constants described in "Theme Brushes" instead.

```
enum {
    kThemeActiveScrollBarDelimiterBrush =
        kThemeBrushScrollBarDelimiterActive,
    kThemeInactiveScrollBarDelimiterBrush =
        kThemeBrushScrollBarDelimiterInactive,
    kThemeFocusHighlightBrush     = kThemeBrushFocusHighlight,
    kThemeActivePopupArrowBrush   = kThemeBrushPopupArrowActive,
    kThemePressedPopupArrowBrush  = kThemeBrushPopupArrowPressed,
    kThemeInactivePopupArrowBrush = kThemeBrushPopupArrowInactive,
    kThemeAppleGuideCoachmarkBrush = kThemeBrushAppleGuideCoachmark
};
```

kThemeBrushPassiveAreaFill

Obsolete. Use the `kThemeBrushStaticAreaFill` constant, described in "Theme Brushes," instead.

```
enum {
    kThemeBrushPassiveAreaFill = kThemeBrushStaticAreaFill
};
```

kThemeActiveDialogTextColor

Obsolete. Use the constants described in "Theme Text Colors" instead.

```
enum {
    kThemeActiveDialogTextColor   = kThemeTextColorDialogActive,
    kThemeInactiveDialogTextColor = kThemeTextColorDialogInactive,
    kThemeActiveAlertTextColor    = kThemeTextColorAlertActive,
    kThemeInactiveAlertTextColor  = kThemeTextColorAlertInactive,
    kThemeActiveModelessDialogTextColor =
        kThemeTextColorModelessDialogActive,
    kThemeInactiveModelessDialogTextColor =
        kThemeTextColorModelessDialogInactive,
    kThemeActiveWindowHeaderTextColor = kThemeTextColorWindowHeaderActive,
    kThemeInactiveWindowHeaderTextColor =
        kThemeTextColorWindowHeaderInactive,
    kThemeActivePlacardTextColor  = kThemeTextColorPlacardActive,
    kThemeInactivePlacardTextColor = kThemeTextColorPlacardInactive,
    kThemePressedPlacardTextColor = kThemeTextColorPlacardPressed,
    kThemeActivePushButtonTextColor = kThemeTextColorPushButtonActive,
    kThemeInactivePushButtonTextColor = kThemeTextColorPushButtonInactive,
    kThemePressedPushButtonTextColor = kThemeTextColorPushButtonPressed,
    kThemeActiveBevelButtonTextColor = kThemeTextColorBevelButtonActive,
    kThemeInactiveBevelButtonTextColor = kThemeTextColorBevelButtonInactive,
    kThemePressedBevelButtonTextColor = kThemeTextColorBevelButtonPressed,
    kThemeActivePopupButtonTextColor = kThemeTextColorPopupButtonActive,
    kThemeInactivePopupButtonTextColor = kThemeTextColorPopupButtonInactive,
    kThemePressedPopupButtonTextColor = kThemeTextColorPopupButtonPressed,
    kThemeIconLabelTextColor      = kThemeTextColorIconLabel,
    kThemeListViewTextColor       = kThemeTextColorListView
};
```

kThemeActiveDocumentWindowTitleTextColor

Obsolete. Use the constants described in "Theme Text Colors" instead.

```
enum {
    kThemeActiveDocumentWindowTitleTextColor =
        kThemeTextColorDocumentWindowTitleActive,
    kThemeInactiveDocumentWindowTitleTextColor =
        kThemeTextColorDocumentWindowTitleInactive,
    kThemeActiveMovableModalWindowTitleTextColor =
        kThemeTextColorMovableModalWindowTitleActive,
    kThemeInactiveMovableModalWindowTitleTextColor =
        kThemeTextColorMovableModalWindowTitleInactive,
    kThemeActiveUtilityWindowTitleTextColor =
        kThemeTextColorUtilityWindowTitleActive,
    kThemeInactiveUtilityWindowTitleTextColor =
        kThemeTextColorUtilityWindowTitleInactive,
    kThemeActivePopupWindowTitleColor =
        kThemeTextColorPopupWindowTitleActive,
    kThemeInactivePopupWindowTitleColor =
        kThemeTextColorPopupWindowTitleInactive,
    kThemeActiveRootMenuTextColor = kThemeTextColorRootMenuActive,
    kThemeSelectedRootMenuTextColor = kThemeTextColorRootMenuSelected,
    kThemeDisabledRootMenuTextColor = kThemeTextColorRootMenuDisabled,
    kThemeActiveMenuItemTextColor = kThemeTextColorMenuItemActive,
    kThemeSelectedMenuItemTextColor = kThemeTextColorMenuItemSelected,
    kThemeDisabledMenuItemTextColor = kThemeTextColorMenuItemDisabled,
    kThemeActivePopupLabelTextColor = kThemeTextColorPopupLabelActive,
    kThemeInactivePopupLabelTextColor = kThemeTextColorPopupLabelInactive
};
```

kThemeScrollBar

Obsolete. Use the constants described in "Theme Track Kinds" instead.

```
enum {
    kThemeScrollBar                = kThemeMediumScrollBar,
    kThemeSlider                   = kThemeMediumSlider,
    kThemeProgressBar              = kThemeMediumProgressBar,
    kThemeIndeterminateBar         = kThemeMediumIndeterminateBar
};
```

kThemeMetricCheckBoxGlyphHeight

Obsolete. Use the constants described in "Theme Metrics" instead.

```
enum {
    kThemeMetricCheckBoxGlyphHeight = kThemeMetricCheckBoxHeight,
    kThemeMetricRadioButtonGlyphHeight = kThemeMetricRadioButtonHeight,
    kThemeMetricDisclosureButtonSize = kThemeMetricDisclosureButtonHeight,
    kThemeMetricBestListHeaderHeight = kThemeMetricListHeaderHeight,
    kThemeMetricSmallProgressBarThickness =
        kThemeMetricNormalProgressBarThickness,
    kThemeMetricProgressBarThickness = kThemeMetricLargeProgressBarThickness };
```

kThemeNoAdornment

Obsolete. Use the constants described in "Theme Button Adornments" instead.

```
enum {
    kThemeNoAdornment               = kThemeAdornmentNone,
    kThemeDefaultAdornment          = kThemeAdornmentDefault,
    kThemeFocusAdornment            = kThemeAdornmentFocus,
    kThemeRightToLeftAdornment      = kThemeAdornmentRightToLeft,
    kThemeDrawIndicatorOnly         = kThemeAdornmentDrawIndicatorOnly
};
```

kThemeStateDisabled

Obsolete. Use the constants described in "Theme Drawing States" instead.

```
enum {
    kThemeStateDisabled = 0
};
```

kThemeWidgetABox

Obsolete. Use the constants described in "Theme Title Bar Items" instead.

```
enum {
    kThemeWidgetABox = 3,
    kThemeWidgetBBox = 4,
    kThemeWidgetBOffBox = 5
};
```

Result Codes

The most common result codes returned by Appearance Manager are listed below.

Result Code	Value	Description
`themeInvalidBrushErr`	-30560	Invalid brush color constant Available in Mac OS X v10.0 and later.

Result Code	Value	Description
`themeProcessRegisteredErr`	-30561	Application already registered as an Appearance Manager client. Available in Mac OS X v10.0 and later.
`themeProcessNotRegisteredErr`	-30562	Application not registered as Appearance Manager client. Available in Mac OS X v10.0 and later.
`themeBadTextColorErr`	-30563	Invalid text color constant Available in Mac OS X v10.0 and later.
`themeHasNoAccentsErr`	-30564	Theme does not support accent colors Available in Mac OS X v10.0 and later.
`themeBadCursorIndexErr`	-30565	Invalid cursor constant Available in Mac OS X v10.0 and later.
`themeScriptFontNotFoundErr`	-30566	No font record for specified script. Available in Mac OS X v10.0 and later.
`themeMonitorDepthNotSupportedErr`	-30567	Theme cannot be supported on all monitors at their current bit depth Available in Mac OS X v10.0 and later.
`themeNoAppropriateBrushErr`	-30568	Theme brush has no corresponding theme text color Available in Mac OS X v10.2 and later.

Gestalt Constants

You can check for version and feature availability information by using the Appearance Manager selectors defined in the Gestalt Manager. For more information, see *Gestalt Manager Reference*.

Deprecated Appearance Manager Functions

A function identified as deprecated has been superseded and may become unsupported in the future.

Deprecated in Mac OS X v10.5

ApplyThemeBackground

Sets the background color or pattern of the current port to be consistent with that of an embedding object. (Deprecated in Mac OS X v10.5. Use `HIThemeApplyBackground` instead.)

```
OSStatus ApplyThemeBackground (
   ThemeBackgroundKind inKind,
   const Rect *bounds,
   ThemeDrawState inState,
   SInt16 inDepth,
   Boolean inColorDev
);
```

Parameters

inKind

A value of type `ThemeBackgroundKind`. Pass a constant specifying the type of embedding object. See "Theme Backgrounds" (page 69) for descriptions of possible values.

bounds

A pointer to a structure of type `Rect`. Before calling `ApplyThemeBackground`, set the rectangle to a size and position that contains the embedding object, in local coordinates.

inState

A value of type `ThemeDrawState`. Pass a constant specifying the current state of the embedding object. See "Theme Drawing States" (page 51) for descriptions of possible values.

inDepth

A signed 16-bit integer. Pass a value specifying the bit depth (in bits per pixel) of the current graphics port.

inColorDev

A value of type `Boolean`. Pass `true` to indicate that you are drawing on a color device, or `false` for a monochrome device.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The `ApplyThemeBackground` function sets the background color or pattern of the current port to match the background of an embedding object, such as a placard or tab control. Your application should call `ApplyThemeBackground` before erasing the background of your application's content to ensure that the content background matches that of the object in which it is visually embedded.

ApplyThemeBackground aligns patterns based on the rectangle passed in the bounds parameter. This is in contrast to the function SetThemeBackground (page 205), which aligns patterns based on the origin of the current port.

You do not need to call ApplyThemeBackground if your content is an embedded part within a control hierarchy and is logically as well as visually embedded in its container; in this case, the Control Manager automatically requests the embedding control to set up the background before drawing the embedded control.

If you have a custom control definition function that erases its background before drawing, you should use the Control Manager function SetUpControlBackground before erasing. SetUpControlBackground calls ApplyThemeBackground if necessary.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
Appearance.h

DisposeMenuItemDrawingUPP

Disposes of the UPP to a menu item drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
void DisposeMenuItemDrawingUPP (
   MenuItemDrawingUPP userUPP
);
```

Parameters

userUPP

The UPP to dispose of.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In
Appearance.h

DisposeMenuTitleDrawingUPP

Disposes of the UPP to a menu title drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
void DisposeMenuTitleDrawingUPP (
   MenuTitleDrawingUPP userUPP
);
```

Parameters

userUPP

The UPP to dispose of.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In

`Appearance.h`

DisposeThemeButtonDrawUPP

Disposes of the UPP to a button drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
void DisposeThemeButtonDrawUPP (
   ThemeButtonDrawUPP userUPP
);
```

Parameters

userUPP

The UPP to dispose of.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In

`Appearance.h`

DisposeThemeEraseUPP

Disposes of the UPP to a background drawing callback function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
void DisposeThemeEraseUPP (
   ThemeEraseUPP userUPP
);
```

Parameters

userUPP

The UPP to dispose of.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In

`Appearance.h`

DisposeThemeIteratorUPP

Disposes of the UPP to a theme iteration callback function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
void DisposeThemeIteratorUPP (
   ThemeIteratorUPP userUPP
);
```

Parameters

userUPP

The UPP to dispose of.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In

`Appearance.h`

DisposeThemeTabTitleDrawUPP

Disposes of the UPP to a tab title drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
void DisposeThemeTabTitleDrawUPP (
   ThemeTabTitleDrawUPP userUPP
);
```

Parameters

userUPP

The UPP to dispose of.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In

`Appearance.h`

DisposeWindowTitleDrawingUPP

Disposes of the UPP to a window title drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
void DisposeWindowTitleDrawingUPP (
   WindowTitleDrawingUPP userUPP
);
```

Parameters

userUPP

The UPP to dispose of.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In

`Appearance.h`

DrawThemeButton

Draws a button. (Deprecated in Mac OS X v10.5. Use HIThemeDrawButton instead.)

```
OSStatus DrawThemeButton (
   const Rect *inBounds,
   ThemeButtonKind inKind,
   const ThemeButtonDrawInfo *inNewInfo,
   const ThemeButtonDrawInfo *inPrevInfo,
   ThemeEraseUPP inEraseProc,
   ThemeButtonDrawUPP inLabelProc,
   URefCon inUserData
);
```

Parameters

inBounds

A pointer to a structure of type Rect. Pass a rectangle specifying the boundary of the button, in local coordinates.

inKind

A value of type ThemeButtonKind. Pass a constant specifying the type of button to draw. See "Theme Buttons" (page 78) for descriptions of possible values.

inNewInfo

A pointer to a structure of type ThemeButtonDrawInfo. Before calling DrawThemeButton, set the structure to contain the new state, value, and adornment for the button. DrawThemeButton uses the information passed in the inNewInfo and inPrevInfo parameters to apply transitional animation or sound effects as the button state changes, if such are specified under the current theme.

inPrevInfo

A pointer to a structure of type ThemeButtonDrawInfo. If the button state is changing, set the structure to contain the previous state, value, and adornment for the button, to allow DrawThemeButton to apply any transitional effects. If the button state is not changing, you can pass NULL.

inEraseProc

A value of type ThemeEraseUPP. If you have a custom background, use this parameter to pass a universal function pointer to an application-defined function such as that described in ThemeEraseProcPtr (page 34). DrawThemeButton calls this function to erase the background before drawing the button. If you pass NULL, DrawThemeButton's default behavior is to erase the background for you.

inLabelProc

A value of type ThemeButtonDrawUPP. If you pass a universal function pointer to an application-defined function such as that described in ThemeButtonDrawProcPtr (page 32), DrawThemeButton calls that function to draw the label of the button. If you pass NULL, no label is drawn.

inUserData

An unsigned 32-bit integer. Provide any data to be passed in to the callback functions specified in the inLabelProc and inEraseProc parameters. Pass NULL if you do not wish to provide any data.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The DrawThemeButton function draws a theme-compliant button. If a ThemeEraseProcPtr is specified in the inEraseProc parameter, DrawThemeButton uses that function to erase the background of the button before drawing the button. After the button is drawn, if a ThemeButtonDrawProcPtr is specified in the inLabelProc parameter, DrawThemeButton calls that function to draw the button's label.

Note that DrawThemeButton also draws any appearance adornments for the button and that these can extend beyond the button's basic bounding rectangle, as specified in the inBounds parameter, and may be of variable shape. You may therefore wish to call the function GetThemeButtonBackgroundBounds (page 178) to obtain the actual rectangle containing the pixels belonging to a button under the current theme.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeChasingArrows

Draws an asynchronous arrows indicator. (Deprecated in Mac OS X v10.5. Use HIThemeDrawChasingArrows instead.)

```
OSStatus DrawThemeChasingArrows (
   const Rect *bounds,
   UInt32 index,
   ThemeDrawState state,
   ThemeEraseUPP eraseProc,
   URefCon eraseData
);
```

Parameters

bounds

A pointer to a structure of type Rect. Before calling DrawThemeChasingArrows, set the rectangle to contain the asynchronous arrows, in local coordinates.

index

An unsigned 32-bit value. Pass a value specifying the current animation step of the arrows. To animate the arrows, increment the initial value by 1 with each call to DrawThemeChasingArrows.

state

A value of type ThemeDrawState. Pass a constant specifying the state in which to draw the asynchronous arrows indicator; see "Theme Drawing States" (page 51). The asynchronous arrows indicator can be drawn as active or inactive; passing kThemeStatePressed produces an error.

eraseProc

A value of type ThemeEraseUPP. If you have a custom background, pass a universal function pointer to an application-defined function such as that described in ThemeEraseProcPtr (page 34). DrawThemeChasingArrows calls that function to erase the background before drawing the asynchronous arrows. If you pass NULL, no erasing occurs.

eraseData

An unsigned 32-bit integer. Provide any data to be passed in to the eraseData parameter of the callback function specified in the eraseProc parameter.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
The DrawThemeChasingArrows function draws a theme-compliant asynchronous arrows (also known as "chasing arrows") indicator.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
Appearance.h

DrawThemeEditTextFrame

Draws an editable text frame. (Deprecated in Mac OS X v10.5. Use HIThemeDrawFrame instead.)

```
OSStatus DrawThemeEditTextFrame (
   const Rect *inRect,
   ThemeDrawState inState
);
```

Parameters

inRect

A pointer to a structure of type Rect. Before calling DrawThemeEditTextFrame, set the rectangle to the position around which to draw the editable text frame, in local coordinates.

inState

A value of type ThemeDrawState. Pass a constant specifying the state in which to draw the editable text frame; see "Theme Drawing States" (page 51). The frame can be drawn as active or inactive; passing kThemeStatePressed produces an error.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
The DrawThemeEditTextFrame function draws a theme-compliant frame for an editable text field. The frame is a maximum of 2 pixels thick and is drawn outside the specified rectangle. You should not use this function to draw frames for items other than editable text fields.

To ensure that you get an appropriate focus ring for your editable text field, you should pass the same rectangle that you use with DrawThemeEditTextFrame function to the function DrawThemeFocusRect (page 152).

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
Appearance.h

DrawThemeFocusRect

Draws or erases a focus ring around a specified rectangle. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawFocusRect` instead.)

```
OSStatus DrawThemeFocusRect (
   const Rect *inRect,
   Boolean inHasFocus
);
```

Parameters

inRect

A pointer to a structure of type `Rect`. Before calling `DrawThemeFocusRect`, set the rectangle to the position around which to draw the focus ring, in local coordinates. The focus ring is drawn outside the rectangle that is passed in, and it can be outset a maximum of 3 pixels.

inHasFocus

A value of type `Boolean`. Pass `true` to draw the focus ring. Pass `false` to erase the focus ring.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Your application can use the `DrawThemeFocusRect` function to draw a theme-compliant focus ring. The presence of a focus ring indicates whether an item has keyboard focus.

If you are drawing a focus ring around an element for which you have drawn a frame using `DrawThemeEditTextFrame` (page 151) or `DrawThemeListBoxFrame` (page 154), you must coordinate your drawing sequence to achieve the correct look. When drawing the element, your application should first call `DrawThemeEditTextFrame` or `DrawThemeListBoxFrame` and then call `DrawThemeFocusRect`, passing the same rectangle in the `inRect` parameter. If you use `DrawThemeFocusRect` to erase the focus ring around an editable text frame or list box frame, you must redraw the editable text frame or list box frame after calling `DrawThemeFocusRect`, because there is typically an overlap.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

DrawThemeFocusRegion

Draws or erases a focus ring around a specified region. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawFocusRect` instead.)

```
OSStatus DrawThemeFocusRegion (
   RgnHandle inRegion,
   Boolean inHasFocus
);
```

Parameters

inRegion

A value of type RgnHandle. Before calling DrawThemeFocusRegion, set the region to the position around which to draw the focus ring, in local coordinates. The focus ring is drawn outside the region that is passed in, and it can be outset a maximum of 3 pixels.

inHasFocus

A value of type Boolean. Pass true to draw the focus region. Pass false to erase the focus region.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Your application can use the DrawThemeFocusRegion function to draw a theme-compliant focus ring. The presence of a focus ring indicates whether an item has keyboard focus.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeGenericWell

Draws an image well frame. (Deprecated in Mac OS X v10.5. Use HIThemeDrawGenericWell instead.)

```
OSStatus DrawThemeGenericWell (
   const Rect *inRect,
   ThemeDrawState inState,
   Boolean inFillCenter
);
```

Parameters

inRect

A pointer to a structure of type Rect. Before calling DrawThemeGenericWell, set the rectangle to the position around which to draw the image well frame, in local coordinates.

inState

A value of type ThemeDrawState. Pass a constant specifying the state in which to draw the image well frame; see "Theme Drawing States" (page 51). The well can be drawn as active or inactive; passing kThemeStatePressed produces an error.

inFillCenter

A value of type Boolean. Set to true to fill the image well frame with white; otherwise, false.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The DrawThemeGenericWell function draws a theme-compliant image well frame. You can specify that the center of the well be filled in with white.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeListBoxFrame

Draws a list box frame. (Deprecated in Mac OS X v10.5. Use HIThemeDrawFrame instead.)

```
OSStatus DrawThemeListBoxFrame (
   const Rect *inRect,
   ThemeDrawState inState
);
```

Parameters

inRect

A pointer to a structure of type Rect. Before calling DrawThemeListBoxFrame, set the rectangle to the position around which to draw the list box frame, in local coordinates.

inState

A value of type ThemeDrawState. Pass a constant specifying the state in which to draw the list box frame; see "Theme Drawing States" (page 51). The frame can be drawn as active or inactive; passing kThemeStatePressed produces an error.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The DrawThemeListBoxFrame function draws a theme-compliant list box frame. The frame is a maximum of 2 pixels thick and is drawn outside the specified rectangle. To ensure that you get an appropriate focus ring for your list box, you should pass the same rectangle that you use with the DrawThemeListBoxFrame function to the function DrawThemeFocusRect (page 152).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeMenuBackground

Draws a menu background. (Deprecated in Mac OS X v10.5. Use HIThemeDrawMenuBackground instead.)

```
OSStatus DrawThemeMenuBackground (
   const Rect *inMenuRect,
   ThemeMenuType inMenuType
);
```

Parameters

inMenuRect

A pointer to a structure of type `Rect`. Before calling `DrawThemeMenuBackground`, set the rectangle to contain the entire menu, in global coordinates.

inMenuType

A value of type `ThemeMenuType`. Pass a constant specifying the type of menu for which to draw a background; see "Theme Menu Types" (page 101) for descriptions of possible values.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The `DrawThemeMenuBackground` function draws a theme-compliant menu background in the specified rectangle.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

DrawThemeMenuBarBackground

Draws a menu bar background. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawMenuBarBackground` instead.)

```
OSStatus DrawThemeMenuBarBackground (
   const Rect *inBounds,
   ThemeMenuBarState inState,
   UInt32 inAttributes
);
```

Parameters

inBounds

A pointer to a structure of type `Rect`. Before calling `DrawThemeMenuBarBackground`, set the rectangle to specify the menu bar's initial size and location, in global coordinates.

inState

A value of type `ThemeMenuBarState`. Pass a constant specifying the state (active or selected) in which to draw the menu bar; see "Theme Menu Bar States" (page 102).

inAttributes

A value indicating the attributes of the menu bar. Pass 0 for a standard menu bar or `kThemeMenuSquareMenuBar` for a menu bar with square corners.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The DrawThemeMenuBarBackground function draws a theme-compliant menu bar background in the specified rectangle.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeMenuItem

Draws a menu item. (Deprecated in Mac OS X v10.5. Use HIThemeDrawMenuItem instead.)

```
OSStatus DrawThemeMenuItem (
   const Rect *inMenuRect,
   const Rect *inItemRect,
   SInt16 inVirtualMenuTop,
   SInt16 inVirtualMenuBottom,
   ThemeMenuState inState,
   ThemeMenuItemType inItemType,
   MenuItemDrawingUPP inDrawProc,
   URefCon inUserData
);
```

Parameters

inMenuRect

A pointer to a structure of type Rect. Before calling DrawThemeMenuItem, set the rectangle to contain the entire menu, in global coordinates. This is the actual menu rectangle as used in your menu definition function.

inItemRect

A pointer to a structure of type Rect. Before calling DrawThemeMenuItem, set the rectangle to contain the menu item, in global coordinates. The menu item's background is drawn in the rectangle passed in the inItemRect parameter. You should calculate the size of the menu item's content and then call GetThemeMenuItemExtra (page 21) to get the amount of padding surrounding menu items in the current theme; the width and height of the menu item rectangle are determined by adding these values together.

inVirtualMenuTop

A signed 16-bit integer. Pass a value representing the actual top of the menu. Normally this value is the top coordinate of the rectangle supplied in the inMenuRect parameter. This value could be different, however, if a menu is scrolled or bigger than can be displayed in the menu rectangle. You typically pass the value of the global variable TopMenuItem into this parameter if you are writing a custom menu definition function.

inVirtualMenuBottom

A signed 16-bit integer. Pass a value representing the actual bottom of the menu. Typically this value is the bottom coordinate of the rectangle supplied in the inMenuRect parameter. This value could be different, however, if a menu is scrolled or bigger than can be displayed in the menu rectangle. You typically pass the value of the global variable AtMenuBottom into this parameter if you are writing a custom menu definition function.

inState

A value of type ThemeMenuState. Pass a constant specifying the state (active, selected, or disabled) in which to draw the menu item; see "Theme Menu States" (page 101).

inItemType

A value of type ThemeMenuItemType. If you pass kThemeMenuItemScrollUpArrow or kThemeMenuItemScrollDownArrow, then you should pass NULL for the inDrawProc parameter, since there is no content to be drawn. If you pass kThemeMenuItemHierarchical, the hierarchical arrow is drawn for you. See "Theme Menu Item Types" (page 102) for descriptions of possible values.

inDrawProc

A value of type MenuItemDrawingUPP. Pass a universal function pointer to a menu item drawing function such as MenuItemDrawingProcPtr (page 29). The value of the inDrawProc parameter can be a valid universal function pointer or NULL.

inUserData

An unsigned 32-bit integer. Provide any data to be passed in to the inUserData parameter of MenuItemDrawingProcPtr (page 29).

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The DrawThemeMenuItem function draws a theme-compliant menu item.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeMenuSeparator

Draws a menu item separator line. (Deprecated in Mac OS X v10.5. Use HIThemeDrawMenuSeparator instead.)

```
OSStatus DrawThemeMenuSeparator (
   const Rect *inItemRect
);
```

Parameters

inItemRect

A pointer to a structure of type Rect. Before calling DrawThemeMenuSeparator, set the rectangle to contain the menu item separator to be drawn, in global coordinates. The rectangle should be the same height as the height returned by the function GetThemeMenuSeparatorHeight (page 22).

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The DrawThemeMenuSeparator function draws a theme-compliant menu item separator line.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

DrawThemeMenuTitle

Draws a menu title. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawMenuTitle` instead.)

```
OSStatus DrawThemeMenuTitle (
   const Rect *inMenuBarRect,
   const Rect *inTitleRect,
   ThemeMenuState inState,
   UInt32 inAttributes,
   MenuTitleDrawingUPP inTitleProc,
   URefCon inTitleData
);
```

Parameters

`inMenuBarRect`

A pointer to a structure of type `Rect`. Before calling `DrawThemeMenuTitle`, set the rectangle to contain the entire menu bar in which the title is to be drawn, in global coordinates. The menu bar background is drawn in the rectangle passed in the `inMenuBarRect` parameter. Your application can call `GetThemeMenuBarHeight` (page 21) to get the height of the menu bar.

`inTitleRect`

A pointer to a structure of type `Rect`. Before calling `DrawThemeMenuTitle`, set the rectangle to contain the menu title, in global coordinates. The title background is drawn in the rectangle passed in the `inTitleRect` parameter. The width of this rectangle is determined by calculating the width of the menu title's content and then calling `GetThemeMenuTitleExtra` (page 23) to get the amount of padding between menu titles in the current theme; these two values are added together and added to the left edge of where the title should be drawn. The top and bottom coordinates of this rectangle should be the same as those of the `inMenuBarRect` parameter.

`inState`

A value of type `ThemeMenuState`. Pass a constant specifying the state (active, selected, or disabled) in which to draw the menu title; see "Theme Menu States" (page 101).

`inAttributes`

Reserved. Pass 0.

`inTitleProc`

A value of type `MenuTitleDrawingUPP`. Pass a universal function pointer to a menu title drawing function such as `MenuTitleDrawingProcPtr` (page 31), defining how to draw the contents of the menu title. The value of the `inTitleProc` parameter can be a valid universal function pointer or `NULL`.

`inTitleData`

An unsigned 32-bit integer. Provide any data to be passed in to the `inUserData` parameter of `MenuTitleDrawingProcPtr` (page 31).

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The `DrawThemeMenuTitle` function draws a theme-compliant menu title.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
`Appearance.h`

DrawThemeModelessDialogFrame

Draws a beveled outline inside the content area of a modeless dialog box. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawWindowFrame` instead.)

```
OSStatus DrawThemeModelessDialogFrame (
   const Rect *inRect,
   ThemeDrawState inState
);
```

Parameters

`inRect`

A pointer to a structure of type `Rect`. Before calling `DrawThemeModelessDialogFrame`, set the rectangle to the boundary of the window's content area (that is, its port rectangle), inset by 1 pixel on each side, in local coordinates.

`inState`

A value of type `ThemeDrawState`. Pass a constant specifying the state in which to draw the modeless dialog box frame; see "Theme Drawing States" (page 51) for descriptions of possible values. The frame can be drawn as active or inactive; passing `kThemeStatePressed` produces an error.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
The `DrawThemeModelessDialogFrame` function draws a beveled frame, no more than 2 pixels wide, that bounds the window's content area. You can use this function to make a custom modeless dialog box theme-compliant the Dialog Manager automatically draws the interior frame for standard dialog boxes.

If you use `DrawThemeModelessDialogFrame` to draw a frame for a modeless dialog box, your application must explicitly invalidate and redraw the frame area if the dialog box is resized.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
`Appearance.h`

DrawThemePlacard

Draws a placard. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawPlacard` instead.)

```
OSStatus DrawThemePlacard (
   const Rect *inRect,
   ThemeDrawState inState
);
```

Parameters

inRect

A pointer to a structure of type `Rect`. Before calling `DrawThemePlacard`, set the rectangle to a size and position that contains the placard, in local coordinates.

inState

A value of type `ThemeDrawState`. Pass a constant specifying the state in which to draw the placard; see "Theme Drawing States" (page 51). The placard can be drawn as active, inactive, or pressed.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The `DrawThemePlacard` function draws a theme-compliant placard inside the specified rectangle.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

DrawThemePopupArrow

Draws a pop-up arrow. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawPopupArrow` instead.)

```
OSStatus DrawThemePopupArrow (
   const Rect *bounds,
   ThemeArrowOrientation orientation,
   ThemePopupArrowSize size,
   ThemeDrawState state,
   ThemeEraseUPP eraseProc,
   URefCon eraseData
);
```

Parameters

bounds

A pointer to a structure of type `Rect`. Before calling `DrawThemePopupArrow`, set the rectangle to contain the arrow, in local coordinates. `DrawThemePopupArrow` positions the arrow relative to the top left corner of the rectangle.

orientation

A value of type `ThemeArrowOrientation`. Pass a constant specifying the direction in which the pop-up arrow points. See "Theme Pop-Up Arrow Orientations" (page 85) for descriptions of possible values.

size

A value of type `ThemePopupArrowSize`. Pass a constant specifying the size of the pop-up arrow to draw. See "Theme Pop-Up Arrow Sizes" (page 86) for descriptions of possible values.

state

A value of type ThemeDrawState. Pass a constant specifying the current state of the button containing the pop-up arrow. See "Theme Drawing States" (page 51) for descriptions of possible values.

eraseProc

A value of type ThemeEraseUPP. If you have a custom background, pass a universal function pointer to an application-defined function such as that described in ThemeEraseProcPtr (page 34). DrawThemePopupArrow calls that function to erase the background before drawing the pop-up arrow. If you pass NULL, no erasing occurs.

eraseData

An unsigned 32-bit integer. Provide any data to be passed in to the eraseData parameter of the callback function specified in the eraseProc parameter.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The DrawThemePopupArrow function draws a theme-compliant pop-up arrow. A pop-up arrow is an image drawn onto another control to indicate that when the control is clicked, you get a pop-up menu. A pop-up arrow is not a separate button itself. Typically, a pop-up arrow is used in conjunction with a button, such as a push button or bevel button. Bevel button controls automatically draw a pop-up arrow if a menu is associated with the control.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemePrimaryGroup

Draws a primary group box frame. (Deprecated in Mac OS X v10.5. Use HIThemeDrawGroupBox instead.)

```
OSStatus DrawThemePrimaryGroup (
   const Rect *inRect,
   ThemeDrawState inState
);
```

Parameters

inRect

A pointer to a structure of type Rect. Before calling DrawThemePrimaryGroup, set the rectangle to the bounds of the primary group box frame, in local coordinates.

inState

A value of type ThemeDrawState. Pass a constant specifying the state in which to draw the primary group box frame; see "Theme Drawing States" (page 51). The frame can be drawn as active or inactive; passing kThemeStatePressed produces an error.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
The DrawThemePrimaryGroup function draws a theme-compliant primary group box frame. The primary group box frame is drawn inside the specified rectangle and is a maximum of 2 pixels thick.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
Appearance.h

DrawThemeScrollBarArrows

Draws scroll bar arrows consistent with the current system preferences. (Deprecated in Mac OS X v10.5. Use HIThemeDrawTrack, which draws the entire scrollbar including both the track and arrows.)

```
OSStatus DrawThemeScrollBarArrows (
   const Rect *bounds,
   ThemeTrackEnableState enableState,
   ThemeTrackPressState pressState,
   Boolean isHoriz,
   Rect *trackBounds
);
```

Parameters

bounds

A pointer to a structure of type Rect. Before calling DrawThemeScrollBarArrows, set the rectangle to contain the scroll bar for which to draw arrows, in local coordinates. Typically, the rectangle you specify is the entire base control rectangle—that is, the value contained in the contrlRect field of the scroll bar's ControlRecord structure.

enableState

A value of type ThemeTrackEnableState. Pass a constant specifying the current state of the scroll bar; see "Theme Track States" (page 110) for descriptions of possible values.

pressState

A value of type ThemeTrackPressState. Pass a constant specifying what is pressed in an active scroll bar or 0 if nothing is pressed. The press state is ignored if the scroll bar is not active. See "Theme Track Press States" (page 112) for descriptions of possible values.

isHoriz

A value of type Boolean. Pass true if the scroll bar is horizontal; pass false if it is vertical.

trackBounds

A pointer to a structure of type Rect. On return, the rectangle is set to the bounds of the track portion of the scroll bar; this rectangle excludes the area containing the scroll bar arrows. Pass NULL if you do not wish to obtain this information.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
The DrawThemeScrollBarArrows function draws a set of theme-compliant scroll bar arrows for the scroll bar whose position and dimensions are specified in the bounds parameter. Depending upon the current system preferences, DrawThemeScrollBarArrows draws the arrows in one of the following configurations:

- one arrow at either end of the scroll bar
- two arrows at the same end of the scroll bar

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
`Appearance.h`

DrawThemeScrollBarDelimiters

Outlines a window's scroll bars. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawScrollBarDelimiters` instead.)

```
OSStatus DrawThemeScrollBarDelimiters (
   ThemeWindowType flavor,
   const Rect *inContRect,
   ThemeDrawState state,
   ThemeWindowAttributes attributes
);
```

Parameters

`flavor`

A value of type `ThemeWindowType`. Pass a constant specifying the type of window for which to draw scroll bar delimiters. See "Theme Window Types" (page 114) for descriptions of possible values.

`inContRect`

A pointer to a structure of type `Rect`. Before calling `DrawThemeScrollBarDelimiters`, set the rectangle to the boundary of the content rectangle of the window, in local coordinates.

`state`

A value of type `ThemeDrawState`. Pass a constant—either `kThemeStateActive` or `kThemeStateInactive`—appropriate to the current state of the window. The scroll bar delimiters can be drawn as active or inactive passing `kThemeStatePressed` produces an error. See "Theme Drawing States" (page 51) for descriptions of these values.

`attributes`

A value of type `ThemeWindowAttributes`. Pass one or more constants corresponding to the window's current visual attributes. See "Theme Window Attributes" (page 116) for descriptions of possible values. Pass 0 if the window has none of the enumerated attributes.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
The `DrawThemeScrollBarDelimiters` function draws theme-compliant outlines for both the horizontal and vertical scroll bars in a given window. The scroll bars are each assumed to cover the full length of their respective sides of the window's content region if the scroll bars for which you wish delimiters to be drawn are not full length, or if only one scroll bar exists for a given window, `DrawThemeScrollBarDelimiters` should not be used.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
`Appearance.h`

DrawThemeSecondaryGroup

Draws a secondary group box frame. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawGroupBox` instead.)

```
OSStatus DrawThemeSecondaryGroup (
   const Rect *inRect,
   ThemeDrawState inState
);
```

Parameters

`inRect`

A pointer to a structure of type `Rect`. Before calling `DrawThemeSecondaryGroup`, set the rectangle to the bounds of the secondary group box frame to be drawn, in local coordinates.

`inState`

A value of type `ThemeDrawState`. Pass a constant specifying the state in which to draw the secondary group box frame; see "Theme Drawing States" (page 51). The frame can be drawn as active or inactive; passing `kThemeStatePressed` produces an error.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
The `DrawThemeSecondaryGroup` function draws a theme-compliant secondary group box frame. The secondary group box frame is drawn inside the specified rectangle and is a maximum of 2 pixels thick. Note that a secondary group box frame is typically nested within a primary group box frame.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
`Appearance.h`

DrawThemeSeparator

Draws a separator line. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawSeparator` instead.)

```
OSStatus DrawThemeSeparator (
   const Rect *inRect,
   ThemeDrawState inState
);
```

Parameters

inRect

A pointer to a structure of type Rect. Before calling DrawThemeSeparator, set the rectangle to contain the separator line, in local coordinates. The orientation of the rectangle determines where the separator line is drawn. If the rectangle is wider than it is tall, the separator line is horizontal; otherwise it is vertical.

inState

A value of type ThemeDrawState. Pass a constant specifying the state in which to draw the separator line; see "Theme Drawing States" (page 51). The separator line can be drawn as active or inactive; passing kThemeStatePressed produces an error.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The DrawThemeSeparator function draws a theme-compliant separator line. The separator line is a maximum of 2 pixels thick and is drawn inside the specified rectangle.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeStandaloneGrowBox

Draws a size box. (Deprecated in Mac OS X v10.5. Use HIThemeDrawGrowBox instead.)

```
OSStatus DrawThemeStandaloneGrowBox (
   Point origin,
   ThemeGrowDirection growDirection,
   Boolean isSmall,
   ThemeDrawState state
);
```

Parameters

origin

A structure of type Point. Pass the origin point of the size box rectangle. For example, the origin point of the size box for an object that can grow downward and to the right is the size box's upper-left corner. Typically, you use the coordinates of the corner of whatever object owns the size box for the origin value. For example, if you are drawing a scrolling list that can grow downward and to the right, the origin value would be the coordinates of the bottom-right corner of the list.

growDirection

A value of type ThemeGrowDirection. Pass a constant specifying the direction(s) in which the resizeable object can grow. See "Theme Size Box Directions" (page 105) for descriptions of possible values. The Appearance Manager uses the growDirection parameter to establish which corner of the size box is the origin.

isSmall

A value of type Boolean. Pass a value of true to specify a small size box (typically for use with small scroll bars) or false to specify a standard size box.

state

A value of type ThemeDrawState. Pass a constant—either kThemeStateActive or kThemeStateInactive—appropriate to the current state of the size box the size box cannot be drawn as pressed. See "Theme Drawing States" (page 51) for descriptions of these values.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The DrawThemeStandaloneGrowBox function draws a theme-compliant size box that is suitable for use inside the content area of a window. The image is designed to fit between scroll bars and does not have to be abutted with the window frame.

Also see the function DrawThemeStandaloneNoGrowBox (page 166).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeStandaloneNoGrowBox

Draws a fill image for use in the corner space between scroll bars. (Deprecated in Mac OS X v10.5. Use HIThemeDrawGrowBox instead.)

```
OSStatus DrawThemeStandaloneNoGrowBox (
   Point origin,
   ThemeGrowDirection growDirection,
   Boolean isSmall,
   ThemeDrawState state
);
```

Parameters

origin

A structure of type Point. Pass the origin point of the rectangle in which to draw the image. Typically, you use the coordinates of the corner of whatever object owns the image for the origin value. For example, if you are drawing the image in the bottom-right corner of a window between the scroll bars of a non-resizeable scrolling list, the origin value would be the coordinates of the bottom-right corner of the list.

growDirection

A value of type ThemeGrowDirection. See "Theme Size Box Directions" (page 105) for descriptions of possible values. The Appearance Manager uses the growDirection parameter to establish which corner of the rectangle that contains the image is the origin.

isSmall

A value of type Boolean. Pass a value of true to specify a small image (for use with small scroll bars) or false to specify a large image (for use with standard scroll bars).

state

A value of type ThemeDrawState. Pass a constant—either kThemeStateActive or kThemeStateInactive—appropriate to the current state of the window containing the fill image the image cannot be drawn as pressed. See "Theme Drawing States" (page 51) for descriptions of these values.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The DrawThemeStandaloneNoGrowBox function draws a theme-compliant image for use as filler in the corner space between scroll bars that

- abut the frame of a window that is not resizeable and which therefore lacks a size box to fill the intervening space
- do not abut the window frame

Also see the function DrawThemeStandaloneGrowBox (page 165).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeTab

Draws a tab. (Deprecated in Mac OS X v10.5. Use HIThemeDrawTab instead.)

```
OSStatus DrawThemeTab (
   const Rect *inRect,
   ThemeTabStyle inStyle,
   ThemeTabDirection inDirection,
   ThemeTabTitleDrawUPP labelProc,
   URefCon userData
);
```

Parameters

inRect

A pointer to a structure of type `Rect`. Before calling `DrawThemeTab`, set the rectangle to the bounds of the tab, in local coordinates. There are two standard sizes (or heights) for tabs that should be used in your calculation of the tab rectangle—these are measured by the distance the tabs protrude from the pane. Small tabs have a height of 16 pixels large tabs have a height of 21 pixels. (The widths of tabs are variable.) Additionally, the distance that the tab overlaps the pane must be included in the tab rectangle this overlap distance is always 3 pixels, although the 3-pixel overlap is only drawn for the front tab. The tab rectangle should reflect the orientation of the tab that is specified in the `inDirection` parameter.

inStyle

A value of type `ThemeTabStyle`. Pass a constant specifying the relative position (front or non-front) and state of the tab. See "Theme Tab Styles" (page 108) for descriptions of possible values.

inDirection

A value of type `ThemeTabDirection`. Pass a constant specifying the direction in which to orient the tab. See "Theme Tab Directions" (page 107) for descriptions of possible values.

labelProc

A value of type `ThemeTabTitleDrawUPP`. Pass a universal function pointer to an application-defined function such as that described in `ThemeTabTitleDrawProcPtr` (page 36). `DrawThemeTab` calls your function to draw the title of the tab. If you pass `NULL`, no drawing occurs.

userData

An unsigned 32-bit integer. Provide any data to be passed in to the `userData` parameter of the callback function specified in the `labelProc` parameter.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The `DrawThemeTab` function draws a theme-compliant tab. A tab control consists of two basic components: multiple tabs that label the various content pages that can be displayed and a single pane upon which the content for each tab is drawn. Use the function `DrawThemeTabPane` (page 168) to draw the tab pane. The Appearance Manager coordinates the appearance of the pane and frontmost tab automatically.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

DrawThemeTabPane

Draws a tab pane. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawTabPane` instead.)

```
OSStatus DrawThemeTabPane (
   const Rect *inRect,
   ThemeDrawState inState
);
```

Parameters

inRect

A pointer to a structure of type `Rect`. Before calling `DrawThemeTabPane`, set the rectangle to the bounds of the tab pane, in local coordinates.

inState

A value of type `ThemeDrawState`. Pass a constant specifying the state in which to draw the tab pane; see "Theme Drawing States" (page 51). The tab pane can be drawn as active or inactive; passing `kThemeStatePressed` produces an error.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The `DrawThemeTabPane` function draws a theme-compliant tab pane. A tab control consists of two basic components: multiple tabs that label the various content pages that can be displayed and a single pane upon which the content for each tab is drawn. Use the function `DrawThemeTab` (page 167) to draw the tab. The Appearance Manager coordinates the appearance of the pane and frontmost tab automatically.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

DrawThemeTextBox

Draws text into the area you specify. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawTextBox` instead.)

```
OSStatus DrawThemeTextBox (
   CFStringRef inString,
   ThemeFontID inFontID,
   ThemeDrawState inState,
   Boolean inWrapToWidth,
   const Rect *inBoundingBox,
   SInt16 inJust,
   CGContextRef inContext
);
```

Parameters

inString

A `CFStringRef` containing the unicode characters you wish to render. You must not pass in a `CFStringRef` that was allocated with any of the "NoCopy" CFString creation functions; a string created with a "NoCopy" function has transient storage which is incompatible with `DrawThemeTextBox`'s caches.

inFontID

The ThemeFontID describing the font you'd like to render the text with. See "Theme Font IDs" (page 91) for the values you can use here.

inState

The ThemeDrawState describing the state of the interface element you are drawing the text for. If, for example, you are drawing text for an inactive window, you would pass kThemeStateInactive. The ThemeDrawState is generally only used to determine the shadow characteristics for the text on Mac OS X.

See "Theme Drawing States" (page 51) for the values you can use here.

Note that the ThemeDrawState does not imply a color. It is not used as a mechanism for graying the text. If you wish to draw grayed text, you must set up the desired gray color and apply it to either the current QuickDraw port or the CGContextRef, as appropriate.

inWrapToWidth

A Boolean value indicating whether you want to draw multiple lines of text wrapped to a bounding box. False indicates that only one line of text should be drawn without any sort of wrapping.

inBoundingBox

The rectangle, in coordinates relative to the current QuickDraw port, describing the area to draw the text within. The first line of text will be top-justified to this rectangle. Wrapping, if desired, will happen at the horizontal extent of this rectangle. Regardless of the amount of text in your CFStringRef, all drawn text will be clipped to this rectangle.

inJust

The horizontal alignment you would like for your text. You can use one of the standard alignment constants from TextEdit.h.

inContext

The CGContextRef into which you would like to draw the text. On Mac OS X, all text drawing happens in CGContextRefs; if you pass NULL, a transient CGContextRef will be allocated and deallocated for use within the single function call. Relying on the system behavior if transiently creating CGContextRefs may result in performance problems. On Mac OS 9, this parameter is ignored.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

DrawThemeTextBox allows you to draw theme-savvy —that is, Aqua-savvy on Mac OS X— text. It is unicode savvy, although only partially so under CarbonLib, and allows you to customize certain text rendering characteristics such as the font, wrapping behavior, and justification. The text is drawn into the CGContextRef you provide, or into the current QuickDraw port if no CGContextRef is provided. None of DrawThemeTextBox's parameters imply a color, so you must set up the desired text color separately before calling DrawThemeTextBox. If you provide a CGContextRef, its fill color will be used to draw the text. If you do not provide a CGContextRef, a color based on the current QuickDraw port's foreground color and the grayishTextOr mode, if set, will be used to draw the text.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeTickMark

Draws a tick mark. (Deprecated in Mac OS X v10.5. Use HIThemeDrawTickMark instead.)

```
OSStatus DrawThemeTickMark (
   const Rect *bounds,
   ThemeDrawState state
);
```

Parameters

bounds

A pointer to a structure of type Rect. Before calling DrawThemeTickMark, set the rectangle to the position that contains the tick mark, in local coordinates. Note that tick marks are of a fixed—3 pixel by 8 pixel—size.

state

A value of type ThemeDrawState. Pass a constant specifying the state in which to draw the tick mark; see "Theme Drawing States" (page 51). The tick mark can be drawn as active or inactive; passing kThemeStatePressed produces an error.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The DrawThemeTickMark function draws a single theme-compliant tick mark. To draw a complete set of tick marks for a track, call the function DrawThemeTrackTickMarks (page 173).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeTitleBarWidget

Draws a close box, zoom box, or collapse box. (Deprecated in Mac OS X v10.5. Use HIThemeDrawTitleBarWidget instead.)

```
OSStatus DrawThemeTitleBarWidget (
   ThemeWindowType flavor,
   const Rect *contRect,
   ThemeDrawState state,
   const ThemeWindowMetrics *metrics,
   ThemeWindowAttributes attributes,
   ThemeTitleBarWidget widget
);
```

Parameters

flavor

A value of type ThemeWindowType. Pass a constant specifying the type of window for which to draw a title bar item. See "Theme Window Types" (page 114) for descriptions of possible values.

contRect

A pointer to a structure of type `Rect`. Before calling `DrawThemeTitleBarWidget`, specify the rectangle for which you wish to draw a title bar item, in coordinates local to the current port. This rectangle is typically the content rectangle of a window.

state

A value of type `ThemeDrawState`. Pass a constant— `kThemeStateActive`, `kThemeStateInactive`, or `kThemeStatePressed`—appropriate to the current state of the title bar item. See "Theme Drawing States" (page 51) for descriptions of these values.

metrics

A pointer to a structure of type `ThemeWindowMetrics`. Before calling `DrawThemeTitleBarWidget`, set the structure to contain information describing the window for which you wish to draw a title bar item.

attributes

A value of type `ThemeWindowAttributes`. Pass one or more constants corresponding to the window's current visual attributes. See "Theme Window Attributes" (page 116) for descriptions of possible values. Pass 0 if the window has none of the enumerated attributes.

widget

A value of type `ThemeTitleBarWidget`. Pass a constant—`kThemeWidgetCloseBox`, `kThemeWidgetZoomBox`, or `kThemeWidgetCollapseBox`—appropriate to the type of title bar item you wish to draw. See "Theme Title Bar Items" (page 117) for descriptions of these values.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The `DrawThemeTitleBarWidget` function draws theme-compliant title bar items. Your application should not typically need to call this function; `DrawThemeTitleBarWidget` is typically of use only for applications that need to draw title bar items of simulated windows. Note that while the `DrawThemeWindowFrame` function automatically draws all title bar items, your application must call the `DrawThemeTitleBarWidget` function during tracking, to ensure that the title bar items' states are drawn correctly.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

DrawThemeTrack

Draws a track. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawTrack` instead.)

```
OSStatus DrawThemeTrack (
   const ThemeTrackDrawInfo *drawInfo,
   RgnHandle rgnGhost,
   ThemeEraseUPP eraseProc,
   URefCon eraseData
);
```

Parameters

drawInfo

A pointer to a structure of type `ThemeTrackDrawInfo`. Before calling `DrawThemeTrack`, set the structure to contain the current visual characteristics of the track.

rgnGhost

A value of type `RgnHandle`. If the track is of a type that contains an indicator, such as a scroll bar or slider, you may pass a handle to the region where `DrawThemeTrack` is to draw a ghost image of the track indicator. Your application should only use a ghost image with the indicator when a track does not support live feedback. Pass `NULL` if you do not want to draw a ghost image.

eraseProc

A value of type `ThemeEraseUPP`. If you have a custom background, pass a universal function pointer to an application-defined function such as that described in `ThemeEraseProcPtr` (page 34). `DrawThemeTrack` calls that function to erase the background before drawing the track. If you pass `NULL`, no erasing occurs.

eraseData

An unsigned 32-bit integer. Provide any data to be passed in to the callback function specified in the `eraseProc` parameter.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Your application may use the `DrawThemeTrack` function to draw a theme-compliant slider, progress bar, or scroll bar. If you use `DrawThemeTrack` to draw a scroll bar, use the function `DrawThemeScrollBarArrows` (page 162) to draw the scroll bar's arrows. If you use `DrawThemeTrack` to draw a slider, use `DrawThemeTrackTickMarks` (page 173) to draw any tick marks for the slider.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

DrawThemeTrackTickMarks

Draws tick marks for a track. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawTrackTickMarks` instead.)

```
OSStatus DrawThemeTrackTickMarks (
   const ThemeTrackDrawInfo *drawInfo,
   ItemCount numTicks,
   ThemeEraseUPP eraseProc,
   URefCon eraseData
);
```

Parameters

drawInfo

A pointer to a structure of type ThemeTrackDrawInfo. Before calling DrawThemeTrackTickMarks, set the structure to describe the current visual characteristics of the track. Because, under Appearance Manager 1.1, sliders are the only track type to support tick marks, you should set the kind field of the ThemeTrackDrawInfo structure to kThemeSlider and fill out the remainder of the structure appropriately for a slider track. You should set the bounds field of the ThemeTrackDrawInfo structure to the boundary of the track itself, not including the area that contains the tick marks; you can obtain the actual bounding rectangle of the track by calling the function GetThemeTrackBounds (page 186). DrawThemeTrackTickMarks draws the tick marks outside the track's bounding rectangle, above or below the track depending on the thumb direction indicated by the drawInfo.trackInfo.slider.thumbDir field.

numTicks

A value of type ItemCount. Pass an unsigned 32-bit value specifying the number of tick marks to be drawn.

eraseProc

A value of type ThemeEraseUPP. If you have a custom background, pass a universal function pointer to an application-defined function such as that described in ThemeEraseProcPtr (page 34). DrawThemeTrackTickMarks calls that function to erase the background before drawing tick marks. If you pass NULL, no erasing occurs.

eraseData

An unsigned 32-bit integer. Provide any data to be passed in to the callback function specified in the eraseProc parameter.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Your application can call the DrawThemeTrackTickMarks function to draw theme-compliant tick marks for a slider control. (Under Appearance Manager 1.1, sliders are the only track type that supports tick marks.) To draw a track control, call the function DrawThemeTrack (page 172). To draw a single tick mark, call the function DrawThemeTickMark (page 171).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeWindowFrame

Draws a window frame. (Deprecated in Mac OS X v10.5. Use HIThemeDrawWindowFrame instead.)

```
OSStatus DrawThemeWindowFrame (
   ThemeWindowType flavor,
   const Rect *contRect,
   ThemeDrawState state,
   const ThemeWindowMetrics *metrics,
   ThemeWindowAttributes attributes,
   WindowTitleDrawingUPP titleProc,
   URefCon titleData
);
```

Parameters

flavor

A value of type ThemeWindowType. Pass a constant specifying the type of window for which to draw a frame. See "Theme Window Types" (page 114) for descriptions of possible values.

contRect

A pointer to a structure of type Rect. Before calling DrawThemeWindowFrame, specify the rectangle for which you wish to draw a window frame, in coordinates local to the current port. This rectangle is typically the content rectangle of a window.

state

A value of type ThemeDrawState. Pass a constant—either kThemeStateActive or kThemeStateInactive—appropriate to the current state of the window. See "Theme Drawing States" (page 51) for descriptions of these values.

metrics

A pointer to a structure of type ThemeWindowMetrics. Before calling DrawThemeWindowFrame, set the structure to describe the window for which to draw a frame.

attributes

A value of type ThemeWindowAttributes. Pass one or more constants corresponding to the window's current visual attributes. See "Theme Window Attributes" (page 116) for descriptions of possible values. Pass 0 if the window has none of the enumerated attributes.

titleProc

A value of type WindowTitleDrawingUPP. If you pass the value kThemeWindowHasTitleText in the attributes parameter, you should pass a universal function pointer to an application-defined function such as that described in WindowTitleDrawingProcPtr (page 37) in the titleProc parameter. DrawThemeWindowFrame calls that function to draw the window's title. Pass NULL if there is no title to be drawn.

titleData

An unsigned 32-bit integer. Provide any data to be passed in to the userData parameter of the callback function specified in the titleProc parameter.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The DrawThemeWindowFrame function draws a window frame appropriate to the specified window type. You may use DrawThemeWindowFrame to make a custom window theme-compliant.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeWindowHeader

Draws a window header. (Deprecated in Mac OS X v10.5. Use HIThemeDrawHeader instead.)

```
OSStatus DrawThemeWindowHeader (
   const Rect *inRect,
   ThemeDrawState inState
);
```

Parameters

inRect

A pointer to a structure of type Rect. Before calling DrawThemeWindowHeader, specify the rectangle containing the window header, in local coordinates.

inState

A value of type ThemeDrawState. Pass a constant specifying the state in which to draw the window header; see "Theme Drawing States" (page 51). The header can be drawn as active or inactive; passing kThemeStatePressed produces an error.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The DrawThemeWindowHeader function draws a theme-compliant window header, such as that used by the Finder.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

DrawThemeWindowListViewHeader

Draws a window list view header. (Deprecated in Mac OS X v10.5. Use HIThemeDrawHeader instead.)

```
OSStatus DrawThemeWindowListViewHeader (
   const Rect *inRect,
   ThemeDrawState inState
);
```

Parameters

inRect

A pointer to a structure of type Rect. Before calling DrawThemeWindowListViewHeader, specify the rectangle in which to draw the window list view header, in local coordinates.

inState

A value of type `ThemeDrawState`. Pass a constant specifying the state in which to draw the window list view header; see "Theme Drawing States" (page 51). The header can be drawn as active or inactive; passing `kThemeStatePressed` produces an error.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The `DrawThemeWindowListViewHeader` function draws a theme-compliant window list view header, such as that used by the Finder. A window list view header is drawn without a line on its bottom edge, so that bevel buttons can be placed against it without overlapping.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

GetThemeAccentColors

Obtains a copy of a theme's accent colors. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
OSStatus GetThemeAccentColors (
   CTabHandle *outColors
);
```

Parameters

outColors

A pointer to a value of type `CTabHandle`. On return, the handle refers to a `ColorTable` structure containing the current accent colors.

Return Value

A result code. `GetThemeAccentColors` returns the result `appearanceThemeHasNoAccents` if the current theme has no accent colors.

Discussion

Note that the Appearance Manager does not currently define semantics for any indexes into the color table produced by the `GetThemeAccentColors` function.

Special Considerations

In Mac OS X, theme accent colors are not supported.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

GetThemeButtonBackgroundBounds

Obtains the rectangle that contains a button. (Deprecated in Mac OS X v10.5. Use `HIThemeGetButtonBackgroundBounds` instead.)

```
OSStatus GetThemeButtonBackgroundBounds (
   const Rect *inBounds,
   ThemeButtonKind inKind,
   const ThemeButtonDrawInfo *inDrawInfo,
   Rect *outBounds
);
```

Parameters

inBounds

A pointer to a structure of type `Rect`. Before calling `GetThemeButtonBackgroundBounds`, set the rectangle to the boundary of the button without any adornments, in local coordinates.

inKind

A value of type `ThemeButtonKind`. Pass a constant specifying the type of button being examined. See "Theme Buttons" (page 78) for descriptions of possible values.

inDrawInfo

A pointer to a structure of type `ThemeButtonDrawInfo`. Before calling `GetThemeButtonBackgroundBounds`, set the structure to contain the state, value, and adornment for the button.

outBounds

A pointer to a structure of type `Rect`. On return, the rectangle contains the actual boundary of the button, including any adornments, in local coordinates.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Appearance adornments can extend beyond the basic bounding rectangle of a button and may be of variable shape. Your application may call the `GetThemeButtonBackgroundBounds` function to obtain the actual rectangle containing the pixels belonging to a button under the current theme.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

GetThemeButtonContentBounds

Obtains the rectangle where content can be drawn for a button. (Deprecated in Mac OS X v10.5. Use `HIThemeGetButtonContentBounds` instead.)

```
OSStatus GetThemeButtonContentBounds (
   const Rect *inBounds,
   ThemeButtonKind inKind,
   const ThemeButtonDrawInfo *inDrawInfo,
   Rect *outBounds
);
```

Parameters

inBounds

A pointer to a structure of type Rect. Before calling GetThemeButtonContentBounds, set the rectangle to contain the boundary of the button, in local coordinates.

inKind

A value of type ThemeButtonKind. Pass a constant specifying the type of button being examined. See "Theme Buttons" (page 78) for descriptions of possible values.

inDrawInfo

A pointer to a structure of type ThemeButtonDrawInfo. Before calling GetThemeButtonContentBounds, set the structure to contain the state, value, and adornment for the button.

outBounds

A pointer to a structure of type Rect. On return, the rectangle contains the actual boundary, in local coordinates, of the area of the button's face in which content can be drawn.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The GetThemeButtonContentBounds function obtains the rectangle where content can be drawn for a button under the current theme.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

GetThemeButtonRegion

Obtains the region occupied by a button. (Deprecated in Mac OS X v10.5. Use HIThemeGetButtonShape instead.)

```
OSStatus GetThemeButtonRegion (
   const Rect *inBounds,
   ThemeButtonKind inKind,
   const ThemeButtonDrawInfo *inNewInfo,
   RgnHandle outRegion
);
```

Parameters

inBounds

A pointer to a structure of type Rect. Before calling GetThemeButtonRegion, set the rectangle to the boundary of the button without any adornments, in local coordinates.

inKind

A value of type ThemeButtonKind. Pass a constant specifying the type of button being examined. See "Theme Buttons" (page 78) for descriptions of possible values.

inNewInfo

A pointer to a structure of type ThemeButtonDrawInfo. Before calling GetThemeButtonRegion, set the structure to contain the state, value, and adornment for the button.

outRegion

A value of type RgnHandle. On return, the region contains the actual dimensions and position of the button and any adornments, in local coordinates.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Appearance adornments can extend beyond the basic bounding rectangle of a button and may be of variable shape. Your application may call the GetThemeButtonRegion function to obtain the exact area covered by pixels belonging to a specific button under the current theme.

Special Considerations

This function is available with Appearance Manager 1.1 and later.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

GetThemeFont

Obtains information about a system font in the current theme. (Deprecated in Mac OS X v10.5. Some theme fonts cannot be drawn using QuickDraw; use HIThemeDrawTextBox instead.)

```
OSStatus GetThemeFont (
   ThemeFontID inFontID,
   ScriptCode inScript,
   Str255 outFontName,
   SInt16 *outFontSize,
   Style *outStyle
);
```

Parameters

inFontID

A value of type ThemeFontID. Pass a constant specifying the kind of font (that is, the current large, small, or small emphasized system fonts or the views font) for which you wish to retrieve the current font name, size, and style in use.

inScript

A value of type ScriptCode. Pass a script code identifying the script system for which you wish obtain font information. You may pass the metascript code smSystemScript to specify the system script.

outFontName

A value of type StringPtr. Pass a pointer to a Pascal string. On return, the string contains the name of the font in use. Pass NULL if you do not wish to obtain this information.

outFontSize

A pointer to a signed 16-bit integer. On return, the integer value specifies the size of the font in use. Pass NULL if you do not wish to obtain this information.

outStyle

A pointer to a value of type Style. On return, the value specifies the style of the font in use. Pass NULL if you do not wish to obtain this information.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Your application can call the GetThemeFont function to obtain the precise font settings (font name, size, and style) used by a system font under the current theme.

Also see the function UseThemeFont (page 209).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

GetThemeMenuBackgroundRegion

Obtains the background region for a menu. (Deprecated in Mac OS X v10.5. Use HIThemeGetMenuBackgroundShape instead.)

```
OSStatus GetThemeMenuBackgroundRegion (
   const Rect *inMenuRect,
   ThemeMenuType menuType,
   RgnHandle region
);
```

Parameters

inMenuRect

A pointer to a structure of type Rect. Before calling GetThemeMenuBackgroundRegion, set the rectangle to contain the entire menu, in global coordinates.

menuType

A value of type ThemeMenuType. Pass a constant specifying the type of menu (pull-down, pop-up, or hierarchical) whose background you wish to obtain; see "Theme Menu Types" (page 101) for descriptions of possible values.

region

A value of type RgnHandle. Pass a region handle created by your application. On return, the region is set to that of the rectangle specified in the inMenuRect parameter, that is, the menu's background region.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The GetThemeMenuBackgroundRegion function obtains the background region that a menu occupies under the current theme.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

GetThemeScrollBarTrackRect

Obtains the area containing the track portion of a scroll bar. (Deprecated in Mac OS X v10.5. Use HIThemeGetScrollBarTrackRect instead.)

```
OSStatus GetThemeScrollBarTrackRect (
   const Rect *bounds,
   ThemeTrackEnableState enableState,
   ThemeTrackPressState pressState,
   Boolean isHoriz,
   Rect *trackBounds
);
```

Parameters

bounds

A pointer to a structure of type Rect. Before calling GetThemeScrollBarTrackRect, set the rectangle to the boundary of the scroll bar, in local coordinates. Typically, the rectangle you specify is the entire base control rectangle—that is, the value contained in the contrlRect field of the track's ControlRecord structure.

enableState

A value of type ThemeTrackEnableState. Pass a constant specifying the current state of the scroll bar; see "Theme Track States" (page 110) for descriptions of possible values.

pressState

A value of type ThemeTrackPressState. Pass a constant specifying what is pressed in an active scroll bar or 0 if nothing is pressed; the press state is ignored if the scroll bar is not active. See "Theme Track Press States" (page 112) for descriptions of possible values.

isHoriz

A value of type Boolean. Pass true if the scroll bar is horizontal; pass false if it is vertical.

trackBounds

A pointer to a structure of type Rect. On return, the structure contains the rectangle that bounds the track portion of the scroll bar. Note that the rectangle produced does not include in its bounds any tick marks that a track (such as a slider) might have; tick marks are drawn outside the track rectangle. Similarly, for a scroll bar, the rectangle produced does not contain the scroll bar arrows, just the track itself.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
Your application may call the `GetThemeScrollBarTrackRect` function to obtain the actual rectangle containing the track portion of a scroll bar under the current theme.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
`Appearance.h`

GetThemeStandaloneGrowBoxBounds

Obtains the bounds of a size box. (Deprecated in Mac OS X v10.5. Use `HIThemeGetGrowBoxBounds` instead.)

```
OSStatus GetThemeStandaloneGrowBoxBounds (
   Point origin,
   ThemeGrowDirection growDirection,
   Boolean isSmall,
   Rect *bounds
);
```

Parameters

`origin`

A structure of type `Point`. Pass the origin point of the size box rectangle. For example, the origin point of the size box for an object that can grow downward and to the right is the size box's upper-left corner. Typically, you use the coordinates of the corner of whatever object owns the size box for the `origin` value; for instance, if you are drawing a scrolling list that can grow downward and to the right, the `origin` value would be the coordinates of the bottom-right corner of the list.

`growDirection`

A value of type `ThemeGrowDirection`. For a size box, pass a constant specifying the direction(s) in which the window can grow. See "Theme Size Box Directions" (page 105) for descriptions of possible values. The Appearance Manager uses the `growDirection` parameter to establish which corner of the size box is the origin.

`isSmall`

A value of type `Boolean`. Pass a value of `true` to specify a small size box or fill image. Pass `false` to specify a large size box or fill image.

`bounds`

A pointer to a structure of type `Rect`. On return, the rectangle contains the boundary of the size box or fill image.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
The `GetThemeStandaloneGrowBoxBounds` function obtains the bounds of a size box under the current theme. Note that you can also use `GetThemeStandaloneGrowBoxBounds` to obtain the bounds of the fill image drawn by the function `DrawThemeStandaloneNoGrowBox` (page 166).

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
`Appearance.h`

GetThemeTabRegion

Obtains the region occupied by a tab. (Deprecated in Mac OS X v10.5. Use `HIThemeGetTabDrawShape` instead.)

```
OSStatus GetThemeTabRegion (
   const Rect *inRect,
   ThemeTabStyle inStyle,
   ThemeTabDirection inDirection,
   RgnHandle ioRgn
);
```

Parameters

`inRect`

A pointer to a structure of type `Rect`. Before calling `GetThemeTabRegion`, set the rectangle to the boundary of the tab, in local coordinates.

`inStyle`

A value of type `ThemeTabStyle`. Pass a constant specifying the relative position (front or non-front) and state of the tab to be examined. See "Theme Tab Styles" (page 108) for descriptions of possible values.

`inDirection`

A value of type `ThemeTabDirection`. Pass a constant specifying the direction in which the tab is oriented. See "Theme Tab Directions" (page 107) for descriptions of possible values.

`ioRgn`

A value of type `RgnHandle`. On return, the region contains the actual dimensions and position of the tab, in local coordinates.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
Because a tab can have a non-rectangular shape, your application should call `GetThemeTabRegion` to get the actual region containing the tab under the current theme, in order to perform accurate hit testing.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
`Appearance.h`

GetThemeTextDimensions

Tells you the height, width, and baseline for a string. (Deprecated in Mac OS X v10.5. Use `HIThemeGetTextDimensions` instead.)

```
OSStatus GetThemeTextDimensions (
   CFStringRef inString,
   ThemeFontID inFontID,
   ThemeDrawState inState,
   Boolean inWrapToWidth,
   Point *ioBounds,
   SInt16 *outBaseline
);
```

Parameters

inString

A CFStringRef containing the unicode characters you wish to measure. You must not pass in a CFStringRef that was allocated with any of the "NoCopy" CFString creation functions, as mentioned in the description of the DrawThemeTextBox (page 169) function.

inFontID

The ThemeFontID describing the font you'd like to measure the text with. See "Theme Font IDs" (page 91) for the values you can use here.

inState

The ThemeDrawState which matches the state you will ultimately render the string with. Drawing state may affect text measurement, so you should be sure the value you pass to GetThemeTextDimensions matches the value you will eventually use for drawing. See "Theme Drawing States" (page 51) for the values you can use here.

inWrapToWidth

A Boolean indicating whether you want the measurements based on wrapping the text to a specific width. If you pass true, you must specify the desired width in ioBounds->h.

ioBounds

On output, ioBounds->v contains the height of the text. If you pass false in the inWrapToWidth parameter, ioBounds->h will contain the width of the text on output. If you pass true in inWrapToWidth, ioBounds->h must (on input) contain the desired width for wrapping; on output, ioBounds->h contains the same value you specified on input.

outBaseline

On output, outBaseline contains the offset (in QuickDraw space) from the bottom edge of the last line of text to the baseline of the first line of text. outBaseline will generally be a negative value.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

GetThemeTextDimensions measures the given string using the font and drawing state you specify. It always reports the actual height and baseline. It sometimes reports the actual width. It can measure a string that wraps. It is unicode savvy, although only partially so under CarbonLib.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

GetThemeTrackBounds

Obtains the bounding rectangle of a track. (Deprecated in Mac OS X v10.5. Use `HIThemeGetTrackBounds` instead.)

```
OSStatus GetThemeTrackBounds (
   const ThemeTrackDrawInfo *drawInfo,
   Rect *bounds
);
```

Parameters

drawInfo

A pointer to a structure of type `ThemeTrackDrawInfo`. Before calling `GetThemeTrackBounds`, set the structure to describe the current visual characteristics of the track. Typically, the rectangle you specify in `ThemeTrackDrawInfo.bounds` is the proposed bounding rectangle for the track. `GetThemeTrackBounds` examines this rectangle to determine the actual bounds that the track would occupy. Depending on the track type, the actual bounding rectangle for a track might contain an absolute or fixed value (as for the height of a progress bar, which is always 14 pixels). Or, the track bounds might scale (as for a scroll bar) to fit the proposed bounds.

bounds

A pointer to a structure of type `Rect`. On return, the rectangle contains the actual boundary of the track, in local coordinates. Note that the rectangle produced does not include in its bounds any tick marks that a track (such as a slider) might have; tick marks are drawn outside the track rectangle. Similarly, for a scroll bar, the rectangle produced does not contain the scroll bar arrows, just the track itself.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Your application may call the `GetThemeTrackBounds` function to obtain the actual rectangle containing a track under the current theme.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

GetThemeTrackDragRect

Obtains the area in which the user may drag a track's indicator. (Deprecated in Mac OS X v10.5. Use `HIThemeGetTrackDragRect` instead.)

```
OSStatus GetThemeTrackDragRect (
   const ThemeTrackDrawInfo *drawInfo,
   Rect *dragRect
);
```

Parameters

drawInfo

A pointer to a structure of type ThemeTrackDrawInfo. Before calling GetThemeTrackDragRect, set the structure to contain the current visual characteristics of the track.

dragRect

A pointer to a structure of type Rect. On return, the rectangle contains the actual boundary of the indicator's drag rectangle, in local coordinates.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Because of varying indicator geometries and theme designs, the draggable area for an indicator is not typically exactly the same as the track rectangle. Your application should call GetThemeTrackDragRect to obtain the actual area within a track where an indicator can be dragged under the current theme.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

GetThemeTrackLiveValue

Obtains the current value of a track's indicator, given its relative position. (Deprecated in Mac OS X v10.5. Use HIThemeGetTrackLiveValue instead.)

```
OSStatus GetThemeTrackLiveValue (
   const ThemeTrackDrawInfo *drawInfo,
   SInt32 relativePosition,
   SInt32 *value
);
```

Parameters

drawInfo

A pointer to a structure of type ThemeTrackDrawInfo. Before calling GetThemeTrackLiveValue, set the structure to contain the current visual characteristics of the track.

relativePosition

A signed 32-bit value. Pass the distance, in pixels, between the minimum end of the track and the near side of the indicator. You may obtain this value by calling either of the functions GetThemeTrackThumbPositionFromOffset (page 188) or GetThemeTrackThumbPositionFromRegion (page 189).

value

A pointer to a signed 32-bit value. On return, this value contains the new value of the indicator.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
Your application can use the `GetThemeTrackLiveValue` function to respond to the `posCntl` and `kControlMsgCalcValueFromPos` control definition message.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
`Appearance.h`

GetThemeTrackThumbPositionFromOffset

Obtains the relative position of a track's indicator, given an offset from its prior position. (Deprecated in Mac OS X v10.5. Use `HIThemeGetTrackThumbPositionFromOffset` instead.)

```
OSStatus GetThemeTrackThumbPositionFromOffset (
   const ThemeTrackDrawInfo *drawInfo,
   Point thumbOffset,
   SInt32 *relativePosition
);
```

Parameters

drawInfo

A pointer to a structure of type `ThemeTrackDrawInfo`. Before calling `GetThemeTrackThumbPositionFromOffset`, set the structure to contain the current visual characteristics of the track.

thumbOffset

A structure of type `Point`. Pass the point (in coordinates local to the control's window) that specifies the vertical and horizontal offset, in pixels, by which the indicator has moved from its current position. Typically, this is the offset between the locations where the cursor was when the user pressed and released the mouse button while dragging the indicator.

relativePosition

A pointer to a signed 32-bit value. On return, this value contains the new distance, in pixels, between the minimum end of the track and the near side of the indicator.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
Your application can use the `GetThemeTrackThumbPositionFromOffset` function to respond to the `posCntl` control definition message.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

GetThemeTrackThumbPositionFromRegion

Obtains the relative position of a track's indicator, given its current position. (Deprecated in Mac OS X v10.5. use `HIThemeGetTrackThumbPositionFromBounds` instead.)

```
OSStatus GetThemeTrackThumbPositionFromRegion (
   const ThemeTrackDrawInfo *drawInfo,
   RgnHandle thumbRgn,
   SInt32 *relativePosition
);
```

Parameters

drawInfo

A pointer to a structure of type `ThemeTrackDrawInfo`. Before calling `GetThemeTrackThumbPositionFromRegion`, set the structure to contain the current visual characteristics of the track.

thumbRgn

A value of type `RgnHandle`. Before calling `GetThemeTrackThumbPositionFromRegion` set the region to contain the actual dimensions and position of the indicator, in local coordinates.

relativePosition

A pointer to a signed 32-bit value. On return, this value contains the new distance, in pixels, between the minimum end of the track and the near side of the indicator.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Your application can use the `GetThemeTrackThumbPositionFromRegion` function to respond to the `kControlMsgCalcValueFromPos` control definition message.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

GetThemeTrackThumbRgn

Obtains the region containing a track's indicator. (Deprecated in Mac OS X v10.5. Use `HIThemeGetTrackThumbShape` instead.)

```
OSStatus GetThemeTrackThumbRgn (
   const ThemeTrackDrawInfo *drawInfo,
   RgnHandle thumbRgn
);
```

Parameters

drawInfo

A pointer to a structure of type `ThemeTrackDrawInfo`. Before calling `GetThemeTrackThumbRgn`, set the structure to contain the current visual characteristics of the track.

thumbRgn

A value of type `RgnHandle`. On return, the region contains the actual dimensions and position of the indicator, in local coordinates.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Your application can use the `GetThemeTrackThumbRgn` function to obtain the indicator region for tracks that have indicators, such as sliders and scroll bars.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

GetThemeWindowRegion

Obtains the specified window region. (Deprecated in Mac OS X v10.5. Use `HIThemeGetWindowShape` instead.)

```
OSStatus GetThemeWindowRegion (
   ThemeWindowType flavor,
   const Rect *contRect,
   ThemeDrawState state,
   const ThemeWindowMetrics *metrics,
   ThemeWindowAttributes attributes,
   AppearanceRegionCode winRegion,
   RgnHandle rgn
);
```

Parameters

flavor

A value of type `ThemeWindowType`. Pass a constant specifying the type of window to be examined. See "Theme Window Types" (page 114) for descriptions of possible values.

contRect

A pointer to a structure of type `Rect`. Before calling `GetThemeWindowRegion`, set the rectangle to the content area of the window, specified in coordinates local to the current port.

state

A value of type `ThemeDrawState`. Pass a constant—either `kThemeStateActive` or `kThemeStateInactive`—appropriate to the current state of the window. See "Theme Drawing States" (page 51) for descriptions of these values.

metrics

A pointer to a structure of type `ThemeWindowMetrics`. Before calling `GetThemeWindowRegion`, set the structure to contain information describing the window.

attributes

A value of type `ThemeWindowAttributes`. Pass one or more constants corresponding to the window's current visual attributes. See "Theme Window Attributes" (page 116) for descriptions of possible values. Pass 0 if the window has none of the enumerated attributes.

winRegion

A value of type `WindowRegionCode`. Pass a constant specifying the region of the window whose dimensions you wish to obtain.

rgn

A value of type `RgnHandle`. Pass a handle to a valid region. On return, the region represents the actual region requested.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The `GetThemeWindowRegion` function obtains the dimensions of the specified window region under the current theme.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

GetThemeWindowRegionHit

Obtains the part of the window that the user clicked upon. (Deprecated in Mac OS X v10.5. Use `HIThemeGetWindowRegionHit` instead.)

```
Boolean GetThemeWindowRegionHit (
   ThemeWindowType flavor,
   const Rect *inContRect,
   ThemeDrawState state,
   const ThemeWindowMetrics *metrics,
   ThemeWindowAttributes inAttributes,
   Point inPoint,
   AppearanceRegionCode *outRegionHit
);
```

Parameters

flavor

A value of type `ThemeWindowType`. Pass a constant specifying the type of window to be examined. See "Theme Window Types" (page 114) for descriptions of possible values.

inContRect

A pointer to a structure of type `Rect`. Before calling `GetThemeWindowRegionHit`, set rectangle to the content area of the window, specified in coordinates local to the current port.

state

A value of type ThemeDrawState. Pass a constant—either kThemeStateActive or kThemeStateInactive—appropriate to the current state of the window. See "Theme Drawing States" (page 51) for descriptions of these values.

metrics

A pointer to a structure of type ThemeWindowMetrics. Before calling GetThemeWindowRegionHit, set the structure to contain information describing the window.

inAttributes

A value of type ThemeWindowAttributes. Pass one or more constants corresponding to the window's current visual attributes. See "Theme Window Attributes" (page 116) for descriptions of possible values. Pass 0 if the window has none of the enumerated attributes.

inPoint

A structure of type Point. Pass the point, specified in specified in coordinates local to the current port, where the mouse-down event occurred. Your application may retrieve this value from the where field of the event structure.

outRegionHit

A pointer to a value of type WindowRegionCode. On return, the value is set to the region code of the window part in which the point passed in the inPoint parameter is located.

Return Value

A value of type Boolean. If true, the mouse-down event occurred inside the window; otherwise, false.

Discussion

Your window definition function should call the GetThemeWindowRegionHit function to determine where a specified mouse-down event occurred.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

HitTestThemeScrollBarArrows

Returns whether the user clicked upon the specified scroll bar's arrows. (Deprecated in Mac OS X v10.5. Use HIThemeHitTestScrollBarArrows instead.)

```
Boolean HitTestThemeScrollBarArrows (
   const Rect *scrollBarBounds,
   ThemeTrackEnableState enableState,
   ThemeTrackPressState pressState,
   Boolean isHoriz,
   Point ptHit,
   Rect *trackBounds,
   AppearancePartCode *partcode
);
```

Parameters

scrollBarBounds

A pointer to a structure of type Rect. Before calling HitTestThemeScrollBarArrows, set the rectangle to the boundary of the scroll bar, in local coordinates. Typically, the rectangle you specify is the entire base control rectangle—that is, the value contained in the contrlRect field of the scroll bar's ControlRecord structure.

enableState

A value of type ThemeTrackEnableState. Pass a constant specifying the current state of the scroll bar; see "Theme Track States" (page 110) for descriptions of possible values.

pressState

A value of type ThemeTrackPressState. Pass a constant specifying what is pressed in an active scroll bar or 0 if nothing is pressed; the press state is ignored if the scroll bar is not active. See "Theme Track Press States" (page 112) for descriptions of possible values.

isHoriz

A value of type Boolean. Pass true if the scroll bar is horizontal; pass false if it is vertical.

ptHit

A structure of type Point. Pass the point, specified in local coordinates, where the mouse-down event occurred. Your application may retrieve this value from the where field of the event structure.

trackBounds

A pointer to a structure of type Rect. On return, the rectangle contains the bounds of the track portion of the scroll bar; this rectangle excludes the area containing the scroll bar arrows. Pass NULL if you do not wish to obtain this information.

partcode

A pointer to a value of type ControlPartCode. On return, this value specifies the arrow in which the mouse-down event occurred.

Return Value

A value of type Boolean. If true, the mouse-down event occurred inside the scroll bar arrows; otherwise, false.

Discussion

Your application may use the HitTestThemeScrollBarArrow function to test whether a given mouse-down event occurred on a scroll bar's arrows. If not, you may then use the rectangle produced in the trackBounds parameter of HitTestThemeScrollBarArrows as the bounds of the track for the function HitTestThemeTrack (page 194), in order to determine whether the mouse-down event occurred in the track part of the scroll bar.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

HitTestThemeTrack

Returns whether the user clicked upon the specified track. (Deprecated in Mac OS X v10.5. Use HIThemeHitTestTrack instead.)

```
Boolean HitTestThemeTrack (
   const ThemeTrackDrawInfo *drawInfo,
   Point mousePoint,
   AppearancePartCode *partHit
);
```

Parameters

drawInfo

A pointer to a structure of type ThemeTrackDrawInfo. Before calling HitTestThemeTrack, set the structure to contain the current visual characteristics of the track.

mousePoint

A structure of type Point. Pass the point, specified in local coordinates, where the mouse-down event occurred. Your application may retrieve this value from the where field of the event structure.

partHit

A pointer to a value of type ControlPartCode. On return, this value specifies the part of the track in which the mouse-down event occurred.

Return Value

A value of type Boolean. If true, the mouse-down event occurred inside the track; otherwise, false.

Discussion

The HitTestThemeTrack function checks to see whether a given track contains the specified point at which a mouse-down event occurred.

For a scroll bar–type track, your application should also check to see whether the mouse-down event occurred in the scroll bar's arrows, which are not considered part of the track and are not tested by this function. To do this, your application should first use the function HitTestThemeScrollBarArrows (page 192) to test whether a given mouse-down event occurred on a scroll bar's arrows. If not, you may then use the rectangle produced in the rTrack parameter of HitTestThemeScrollBarArrows as the bounds of the track for HitTestThemeTrack, in order to determine whether the mouse-down event occurred in the track part of the scroll bar.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

InvokeMenuItemDrawingUPP

Invokes your menu item drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
void InvokeMenuItemDrawingUPP (
   const Rect *inBounds,
   SInt16 inDepth,
   Boolean inIsColorDevice,
   SRefCon inUserData,
   MenuItemDrawingUPP userUPP
);
```

Discussion
You should not need to use the function `InvokeMenuItemDrawingUPP`, as the system calls your menu item drawing function for you.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In
`Appearance.h`

InvokeMenuTitleDrawingUPP

Invokes your menu title drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
void InvokeMenuTitleDrawingUPP (
   const Rect *inBounds,
   SInt16 inDepth,
   Boolean inIsColorDevice,
   SRefCon inUserData,
   MenuTitleDrawingUPP userUPP
);
```

Discussion
You should not need to use the function `InvokeMenuTitleDrawingUPP`, as the system calls your menu title drawing function for you.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In
`Appearance.h`

InvokeThemeButtonDrawUPP

Invokes your button drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
void InvokeThemeButtonDrawUPP (
   const Rect *bounds,
   ThemeButtonKind kind,
   const ThemeButtonDrawInfo *info,
   URefCon userData,
   SInt16 depth,
   Boolean isColorDev,
   ThemeButtonDrawUPP userUPP
);
```

Discussion

You should not need to use the function `InvokeThemeButtonDrawUPP`, as the system calls your button drawing function for you.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In

`Appearance.h`

InvokeThemeEraseUPP

Invokes your background drawing callback function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
void InvokeThemeEraseUPP (
   const Rect *bounds,
   URefCon eraseData,
   SInt16 depth,
   Boolean isColorDev,
   ThemeEraseUPP userUPP
);
```

Discussion

You should not need to use the function `InvokeThemeEraseUPP`, as the system calls your `ThemeEraseProcPtr` (page 34) callback function for you.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In

`Appearance.h`

InvokeThemeIteratorUPP

Invokes your theme iteration callback function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
Boolean InvokeThemeIteratorUPP (
   ConstStr255Param inFileName,
   SInt16 resID,
   Collection inThemeSettings,
   PRefCon inUserData,
   ThemeIteratorUPP userUPP
);
```

Discussion
You should not need to use the function InvokeThemeIteratorUPP, as the system calls your ThemeIteratorProcPtr (page 35) callback function for you.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In
Appearance.h

InvokeThemeTabTitleDrawUPP

Invokes your tab title drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
void InvokeThemeTabTitleDrawUPP (
   const Rect *bounds,
   ThemeTabStyle style,
   ThemeTabDirection direction,
   SInt16 depth,
   Boolean isColorDev,
   URefCon userData,
   ThemeTabTitleDrawUPP userUPP
);
```

Discussion
You should not need to use the function InvokeThemeTabTitleDrawUPP, as the system calls your tab title drawing function for you.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In
Appearance.h

InvokeWindowTitleDrawingUPP

Invokes your window title drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
void InvokeWindowTitleDrawingUPP (
   const Rect *bounds,
   SInt16 depth,
   Boolean colorDevice,
   URefCon userData,
   WindowTitleDrawingUPP userUPP
);
```

Discussion

You should not need to use the function InvokeWindowTitleDrawingUPP, as the system calls your window title drawing function for you.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In

Appearance.h

IsAppearanceClient

Returns whether a given process is currently registered as a client of the Appearance Manager. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
Boolean IsAppearanceClient (
   const ProcessSerialNumber *process
);
```

Parameters

process

A pointer to a value of type ProcessSerialNumber. Pass the serial number of the process to examine.

Return Value

A value of type Boolean. If true, the specified process is currently registered as a client of the Appearance Manager; otherwise, false.

Discussion

Applications typically do not need to call the IsAppearanceClient function. A plug-in could call IsAppearanceClient to determine whether the process in which it is running is a registered Appearance Manager client. To register with the Appearance Manager, call the function RegisterAppearanceClient (page 203).

Special Considerations

This function always returns true in Mac OS X.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

IsThemeInColor

Returns whether the current theme would draw in color in the given environment. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
Boolean IsThemeInColor (
   SInt16 inDepth,
   Boolean inIsColorDevice
);
```

Parameters

inDepth

A signed 16-bit integer. Pass a value specifying the bit depth (in bits per pixel) of the current graphics port.

inIsColorDevice

A value of type `Boolean`. Pass `true` to indicate that you are drawing on a color device, or `false` for a monochrome device.

Return Value

A value of type `Boolean`. `IsThemeInColor` returns `true` if, given the depth and color device information, the theme would draw in color; otherwise, `false`.

Discussion

To be consistent with the current theme, your application can call the `IsThemeInColor` function to determine whether or not the Appearance Manager is drawing the theme in color or black and white. If the function returns `true`, you should draw in color; if it returns `false`, you should draw in black and white. Note that the Appearance Manager may draw a theme in black and white not only because of the current bit depth or device type, but also because the theme may have defined black-and-white elements, such as the "Black & White" accent color in the platinum appearance.

Special Considerations

In Mac OS X, this function always returns true because the Aqua theme is always drawn in color.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

IsValidAppearanceFileType

Returns whether the system can interpret files of a given file type as appearance files. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
Boolean IsValidAppearanceFileType (
   OSType fileType
);
```

Parameters

fileType

A four-character code. Pass the file type to be examined.

Return Value
A value of type `Boolean`. If `true`, files of the specified file type can be used as appearance files; otherwise, `false`.

Discussion
Under Appearance Manager 1.1, only the `'thme'` and `'pltn'` file types, described in "Appearance Manager File Types" (page 47), are valid appearance file types. Your application typically does not need to call this function.

Special Considerations

This function always returns false in Mac OS X.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
`Appearance.h`

IterateThemes

Iterates over all themes installed on a system. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
OSStatus IterateThemes (
   ThemeIteratorUPP inProc,
   PRefCon inUserData
);
```

Parameters

inProc

A universal function pointer to an application-defined function such as that described in `ThemeIteratorProcPtr` (page 35). `IterateThemes` calls the specified function for each theme found installed in the system.

inUserData

A pointer to data of any type. Provide any data to be passed in to the `inUserData` parameter of the callback function specified in the `inProc` parameter. Pass `NULL`, if you do not wish to provide any data.

Return Value
A result code. See "Appearance Manager Result Codes" (page 142).

Discussion
The `IterateThemes` function continues to iterate until the function specified in the `inProc` parameter returns `false` or until there are no more themes.

Special Considerations

This function does nothing in Mac OS X; it does not call the theme iterator callback function.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
`Appearance.h`

NewMenuItemDrawingUPP

Creates a new universal procedure pointer (UPP) to a menu item drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
MenuItemDrawingUPP NewMenuItemDrawingUPP (
   MenuItemDrawingProcPtr userRoutine
);
```

Return Value
A UPP. See `MenuItemDrawingProcPtr` (page 29) for information on the menu item drawing function. See the description of the `MenuItemDrawingUPP` data type.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In
`Appearance.h`

NewMenuTitleDrawingUPP

Creates a new universal procedure pointer (UPP) to a menu title drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
MenuTitleDrawingUPP NewMenuTitleDrawingUPP (
   MenuTitleDrawingProcPtr userRoutine
);
```

Return Value
A UPP. See `MenuTitleDrawingProcPtr` (page 31) for information on the menu title drawing function. See the description of the `MenuTitleDrawingUPP` data type.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In
`Appearance.h`

NewThemeButtonDrawUPP

Creates a new universal procedure pointer (UPP) to a button drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
ThemeButtonDrawUPP NewThemeButtonDrawUPP (
   ThemeButtonDrawProcPtr userRoutine
);
```

Return Value

A UPP. See `ThemeButtonDrawProcPtr` (page 32) for information on the button drawing function. See the description of the `ThemeButtonDrawUPP` data type.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In

`Appearance.h`

NewThemeEraseUPP

Creates a new universal procedure pointer (UPP) to a background drawing callback function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
ThemeEraseUPP NewThemeEraseUPP (
   ThemeEraseProcPtr userRoutine
);
```

Return Value

A UPP. See the description of the `ThemeEraseUPP` data type.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In

`Appearance.h`

NewThemeIteratorUPP

Creates a new universal procedure pointer (UPP) to a theme iteration callback function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
ThemeIteratorUPP NewThemeIteratorUPP (
   ThemeIteratorProcPtr userRoutine
);
```

Return Value

A UPP. See the description of the `ThemeIteratorUPP` data type.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In

`Appearance.h`

NewThemeTabTitleDrawUPP

Creates a new universal procedure pointer (UPP) to a tab title drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
ThemeTabTitleDrawUPP NewThemeTabTitleDrawUPP (
   ThemeTabTitleDrawProcPtr userRoutine
);
```

Return Value
A UPP. See ThemeTabTitleDrawProcPtr (page 36) for information on the tab title drawing function. See the description of the ThemeTabTitleDrawUPP data type.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In
Appearance.h

NewWindowTitleDrawingUPP

Creates a new universal procedure pointer (UPP) to a window title drawing function. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
WindowTitleDrawingUPP NewWindowTitleDrawingUPP (
   WindowTitleDrawingProcPtr userRoutine
);
```

Return Value
A UPP. See WindowTitleDrawingProcPtr (page 37) for information on the window title drawing function. See the description of the WindowTitleDrawingUPP data type.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared In
Appearance.h

RegisterAppearanceClient

Registers your program with the Appearance Manager. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
OSStatus RegisterAppearanceClient (
   void
);
```

Return Value
A result code. The result code appearanceProcessRegisteredErr indicates that your program was already registered when you called the RegisterAppearanceClient function.

Discussion

The `RegisterAppearanceClient` function must be called at the beginning of your program, prior to initializing or drawing any onscreen elements or invoking any definition functions, such as the menu bar.

You should call `RegisterAppearanceClient` in order to receive Appearance Manager Apple events. Under Appearance Manager 1.1 and later, when the user changes the current appearance (that is, when a theme switch occurs), the Appearance Manager sends Apple events to all running applications that are registered as clients of the Appearance Manager and which are high-level event aware. Because typical results of a theme switch might include a change in menu bar height or window structure dimensions, as well as changes to the system fonts, colors, and patterns currently in use, you should listen for and respond to the Appearance Manager Apple events under most circumstances. See "Appearance Manager Apple Events" (page 46) for more details.

When your program calls `RegisterAppearanceClient`, the Appearance Manager also automatically maps standard pre–Appearance Manager definition functions to their theme-compliant equivalents for your program, whether or not systemwide appearance is active.

See also the function `UnregisterAppearanceClient` (page 209).

Special Considerations

This function does nothing in Mac OS X.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

SetTheme

Sets a specified collection as the current theme. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
OSStatus SetTheme (
   Collection ioCollection
);
```

Parameters

`ioCollection`

A value of type `Collection`. Pass a reference to a collection object, such as that created by calling the Collection Manager function `NewCollection`. Before calling `SetTheme`, set the collection to contain theme data that you wish to use for the current theme. The theme data is in the form of collection items, each corresponding to an attribute of the theme. For a given theme, the actual number of collection items may vary, depending upon how fully the theme's attributes are specified. Your application can use theme collection tags, along with various Collection Manager functions, to access the data in the collection. See "Theme Collection Tags" (page 47) for descriptions of the possible theme collection items.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

The `SetTheme` function sets the attributes of the current theme. You may use `SetTheme` to set up a custom theme environment for your application, to be used only when your application is active. You may also use `SetTheme` to create a theme environment that you want to be user-selectable and to have systemwide effect.

Your application can use the `GetTheme` (page 18) function to obtain a collection containing a copy of the data for the current theme.

Special Considerations

This function does not modify the current theme in Mac OS X.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

SetThemeBackground

Applies a theme-compliant color or pattern to the background of the current port. (Deprecated in Mac OS X v10.5. Use `HIThemeSetFill` and draw using Quartz 2D.)

```
OSStatus SetThemeBackground (
   ThemeBrush inBrush,
   SInt16 inDepth,
   Boolean inIsColorDevice
);
```

Parameters

`inBrush`

A value of type `ThemeBrush`. Pass a constant specifying the theme brush to which to set the background; see "Theme Brushes" (page 70) for descriptions of possible values.

`inDepth`

A signed 16-bit integer. Pass a value specifying the bit depth (in bits per pixel) of the current graphics port.

`inIsColorDevice`

A value of type `Boolean`. Pass `true` to indicate that you are drawing on a color device. Pass `false` for a monochrome device.

Return Value

A result code. The result code `appearanceBadBrushIndexErr` indicates that the brush constant passed was not valid.

Discussion

Your application should call the `SetThemeBackground` function each time you wish to draw in a specified brush type. Note that the `SetThemeBackground` function aligns patterns with 0,0 in the current port. To align a pattern independently of the port origin, use the function `ApplyThemeBackground` (page 145).

Because the constant in the inBrush parameter can specify a color or pattern, depending on the current theme, your application must save and restore the current drawing state of the graphics port around calls to SetThemeBackground. Under Appearance Manager 1.1 and later, you can use the functions GetThemeDrawingState (page 20) and SetThemeDrawingState (page 29) to do this.

Prior to Appearance Manager 1.1, you must save and restore the pnPixPat and bkPixPat fields of your graphics port when saving the text and background colors. Because patterns in the bkPixPat field override the background color of the window, call the QuickDraw function BackPat to set your background pattern to a normal white pattern. This ensures that you can use RGBBackColor to set your background color to white, call the QuickDraw function EraseRect, and get the expected results.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

Appearance.h

SetThemePen

Applies a theme-compliant color or pattern to the foreground of the current port. (Deprecated in Mac OS X v10.5. Use HIThemeSetStroke and draw using Quartz 2D.)

```
OSStatus SetThemePen (
   ThemeBrush inBrush,
   SInt16 inDepth,
   Boolean inIsColorDevice
);
```

Parameters

inBrush

A value of type ThemeBrush. Pass a constant specifying the theme brush type to which to set the pen; see "Theme Brushes" (page 70) for descriptions of possible values.

inDepth

A signed 16-bit integer. Pass a value specifying the bit depth (in bits per pixel) of the current graphics port.

inIsColorDevice

A value of type Boolean. Pass true to indicate that you are drawing on a color device. Pass false for a monochrome device.

Return Value

A result code. The result code appearanceBadBrushIndexErr indicates that the brush constant passed in was not valid.

Discussion

Your application should call the SetThemePen function each time you wish to draw an element in a specified brush constant.

Because the constant in the inBrush parameter can represent a color or pattern, depending on the current theme, your application must save and restore the current drawing state of the graphics port around calls to SetThemePen. Under Appearance Manager 1.1 and later, you can use the functions GetThemeDrawingState (page 20) and SetThemeDrawingState (page 29) to do this. Prior to Appearance

Manager 1.1, you must save and restore the pnPixPat and bkPixPat fields of your graphics port when saving the text and background colors. Because patterns in the pnPixPat field override the foreground color of the window, call the QuickDraw function PenPat to set your foreground pattern to a normal white pattern. This ensures that you can use RGBForeColor to set your foreground color to white, call the QuickDraw function PaintRect, and get the expected results.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
Appearance.h

SetThemeTextColor

Sets the current text color to be consistent with that of a specified element. (Deprecated in Mac OS X v10.5. Use HIThemeSetTextFill and draw with Quartz 2D, ATSUI, or HIThemeDrawTextBox.)

```
OSStatus SetThemeTextColor (
   ThemeTextColor inColor,
   SInt16 inDepth,
   Boolean inIsColorDevice
);
```

Parameters

inColor

A value of type ThemeTextColor. Pass a constant specifying an interface element. SetThemeTextColor sets the current text color to be the same as the color of that element's text. See "Theme Text Colors" (page 94) for descriptions of possible values.

inDepth

A signed 16-bit integer. Pass a value specifying the bit depth (in bits per pixel) of the current graphics port.

inIsColorDevice

A value of type Boolean. Pass true to indicate that you are drawing on a color device. Pass false for a monochrome device.

Return Value
A result code. The result code appearanceBadTextColorIndexErr indicates that the text color index passed was not valid.

Discussion
Your application typically uses the SetThemeTextColor function inside a DeviceLoop drawing function to set the foreground color to a theme-compliant value.

Also see the function GetThemeTextColor (page 25).

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

TruncateThemeText

Truncates text to fit within the width you specify. (Deprecated in Mac OS X v10.5. There is no replacement function; use `HIThemeGetTextDimensions` or `HIThemeDrawTextBox` instead.)

```
OSStatus TruncateThemeText (
   CFMutableStringRef inString,
   ThemeFontID inFontID,
   ThemeDrawState inState,
   SInt16 inPixelWidthLimit,
   TruncCode inTruncWhere,
   Boolean *outTruncated
);
```

Parameters

inString

A `CFMutableStringRef` containing the unicode characters you wish to truncate. On output, this string may have been altered to fit within the specified width.

You must not pass in a CFString that was allocated with any of the "NoCopy" CFString creation functions, as mentioned in the description of the `DrawThemeTextBox` (page 169) function.

inFontID

The `ThemeFontID` to use for text measurements. See "Theme Font IDs" (page 91) for the values you can use here.

inState

The `ThemeDrawState` which matches the state you will ultimately render the string with. This may affect text measurement during truncation, so you should be sure the value you pass to `TruncateThemeText` matches the value you will eventually use for drawing. See "Theme Drawing States" (page 51) for the values you can use here.

inPixelWidthLimit

The maximum width, in pixels, that the resulting truncated string may have.

inTruncWhere

A `TruncCode` indicating where you would like truncation to occur.

outTruncated

On output, this Boolean value indicates whether the string was truncated. `True` means the string was truncated. `False` means the string was not—and did not need to be—truncated.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

`TruncateThemeText` alters a unicode string to fit within a width that you specify. It is unicode savvy, although only partially so under CarbonLib, and makes its calculations—and any subsequent string alteration—based on the font and drawing state you specify. If the string needs to be truncated, it will be reduced to the maximum number of characters which, with the addition of an ellipsis character, fits within the specified width.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
`Appearance.h`

UnregisterAppearanceClient

Informs the Appearance Manager that your program is no longer its client. (Deprecated in Mac OS X v10.5. There is no replacement function.)

```
OSStatus UnregisterAppearanceClient (
   void
);
```

Return Value
A result code. The result code `appearanceProcessNotRegisteredErr` indicates that your program was not registered when you called the `UnregisterAppearanceClient` function.

Discussion
The `UnregisterAppearanceClient` function is automatically called for you when your program terminates. While you do not typically need to call this function, you might want to call `UnregisterAppearanceClient` if you are running a plug-in architecture, and you know that a given plug-in is not theme-compliant. In this case you would bracket your use of the plug-in with calls to `UnregisterAppearanceClient` (before the plug-in is used) and `RegisterAppearanceClient` (after the plug-in is used), so that the Appearance Manager is turned off for the duration of the plug-in's usage.

See also the function `RegisterAppearanceClient` (page 203).

Special Considerations

This function does nothing in Mac OS X.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In
`Appearance.h`

UseThemeFont

Sets the font of the current graphics port to one of the current theme's system fonts. (Deprecated in Mac OS X v10.5. Use `HIThemeDrawTextBox` instead.)

```
OSStatus UseThemeFont (
   ThemeFontID inFontID,
   ScriptCode inScript
);
```

Parameters

inFontID

A value of type `ThemeFontID`. Pass a constant specifying the kind of font (that is, the current large, small, or small emphasized system fonts or the views font) to be applied to the current port. See "Theme Font IDs" (page 91) for descriptions of possible values.

inScript

A value of type `ScriptCode`. Pass a script code specifying the script system for which you wish to set the current font; you may pass the metascript code `smSystemScript` to specify the system script.

Return Value

A result code. See "Appearance Manager Result Codes" (page 142).

Discussion

Your application can call the `UseThemeFont` function to draw text in one of the current theme's system fonts.

Also see the function `GetThemeFont` (page 180).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Not available to 64-bit applications.

Declared In

`Appearance.h`

Document Revision History

This table describes the changes to *Appearance Manager Reference*.

Date	Notes
2007-01-23	Updated for Mac OS X v10.5.
2003-02-01	Updated formatting

Index