Palette Manager Reference

(Not Recommended)

Carbon > Graphics & Imaging

2006-07-13

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

Simultaneously published in the United States and Canada.

Contents

Palette Manager Reference (Not Recommended) 5

Overview 5
Functions by Task 5
- Animating Palettes 5
- Changing the Pixel Depth for a Video Device 6
- Drawing With Color Palettes 6
- Initializing and Allocating Palettes 6
- Initializing the Palette Manager 7
- Interacting With the Window Manager 7
- Manipulating Palette Entries 7
- Manipulating Palettes and Color Tables 8
- Miscellaneous 8

Functions 8
- SetDepth 8

Data Types 9
- ColorInfo 9
- Palette 10

Constants 11
- Usage Constants 11
- Update Constants 13

Appendix A **Deprecated Palette Manager Reference (Not Recommended) Functions 15**

Deprecated in Mac OS X v10.4 15
- ActivatePalette 15
- AnimateEntry 16
- AnimatePalette 17
- CopyPalette 18
- CTab2Palette 19
- DisposePalette 20
- Entry2Index 20
- GetEntryColor 21
- GetEntryUsage 22
- GetGray 23
- GetNewPalette 23
- GetPalette 24
- GetPaletteUpdates 25
- HasDepth 26
- InitPalettes 27
- NewPalette 27
- NSetPalette 28

Palette2CTab 29
PmBackColor 30
PmForeColor 31
PMgrVersion 32
ResizePalette 32
RestoreBack 33
RestoreDeviceClut 34
RestoreFore 34
SaveBack 35
SaveFore 36
SetEntryColor 36
SetEntryUsage 37
SetPalette 38
SetPaletteUpdates 39

Document Revision History 41

Index 43

Palette Manager Reference (Not Recommended)

Framework:	ApplicationServices/ApplicationServices.h
Declared in	ImageCompression.k.h Palettes.h

Overview

Important: The Palette Manager is deprecated as of Mac OS X v10.4. There is no replacement.

Prior to Mac OS X, applications could use the Palette Manager to ensure that the best set of colors is available when drawing to displays with limited color capabilities (pixel depth of 8 bits or less). For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Functions by Task

Function descriptions are grouped by programming task. For an alphabetical list of functions, see the API index.

Animating Palettes

`AnimateEntry` (page 16) Deprecated in Mac OS X v10.4

Changes the color of a window's palette entry. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`AnimatePalette` (page 17) Deprecated in Mac OS X v10.4

Changes the colors of a series of palette entries; it is similar to the `AnimateEntry` function, but it acts upon a range of entries. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Changing the Pixel Depth for a Video Device

`SetDepth` (page 8)
Changes the pixel depth of a video device. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`HasDepth` (page 26) Deprecated in Mac OS X v10.4
Determines whether a video device supports a specific pixel depth. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Drawing With Color Palettes

`PmBackColor` (page 30) Deprecated in Mac OS X v10.4
Sets the background color field of the current graphics port to a palette color. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`PmForeColor` (page 31) Deprecated in Mac OS X v10.4
Sets the foreground color field of the current graphics port to a palette color. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`RestoreBack` (page 33) Deprecated in Mac OS X v10.4
Sets the current background color to the color you specify. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`RestoreFore` (page 34) Deprecated in Mac OS X v10.4
Sets the current foreground color to the color you supply. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`SaveBack` (page 35) Deprecated in Mac OS X v10.4
Saves the current background color. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`SaveFore` (page 36) Deprecated in Mac OS X v10.4
Saves the current foreground color. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Initializing and Allocating Palettes

`DisposePalette` (page 20) Deprecated in Mac OS X v10.4
Disposes of a palette. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`GetNewPalette` (page 23) Deprecated in Mac OS X v10.4
Creates and initializes a palette from a `'pltt'` resource. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`NewPalette` (page 27) Deprecated in Mac OS X v10.4
Allocates a new palette from colors in the color table. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Initializing the Palette Manager

`InitPalettes` (page 27) Deprecated in Mac OS X v10.4

Initializes the Palette Manager. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`PMgrVersion` (page 32) Deprecated in Mac OS X v10.4

Determines which version of the Palette Manager is executing; it returns an integer specifying the version number. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Interacting With the Window Manager

`ActivatePalette` (page 15) Deprecated in Mac OS X v10.4

Changes the device color tables and generates window updates as needed to meet the color requirements of your window. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`GetPalette` (page 24) Deprecated in Mac OS X v10.4

Obtains a window's palette. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`GetPaletteUpdates` (page 25) Deprecated in Mac OS X v10.4

Obtains the update attribute of a palette. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`NSetPalette` (page 28) Deprecated in Mac OS X v10.4

Associates a new palette with a window. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`SetPalette` (page 38) Deprecated in Mac OS X v10.4

Associates a palette with a window. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`SetPaletteUpdates` (page 39) Deprecated in Mac OS X v10.4

Sets the update attribute of a palette. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Manipulating Palette Entries

`Entry2Index` (page 20) Deprecated in Mac OS X v10.4

Obtains the index for a specified entry in the current graphics port's palette on the current device. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`GetEntryColor` (page 21) Deprecated in Mac OS X v10.4

Obtains the color of a palette entry. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`GetEntryUsage` (page 22) Deprecated in Mac OS X v10.4

Obtains the usage and tolerance fields of a palette entry. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`SetEntryColor` (page 36) Deprecated in Mac OS X v10.4

Changes the color of a palette entry. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`SetEntryUsage` (page 37) Deprecated in Mac OS X v10.4
: Modifies the usage category and tolerance values of a palette entry. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Manipulating Palettes and Color Tables

`CopyPalette` (page 18) Deprecated in Mac OS X v10.4
: Copies entries from one palette to another. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`CTab2Palette` (page 19) Deprecated in Mac OS X v10.4
: Copies the colors of a color table into a palette. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`Palette2CTab` (page 29) Deprecated in Mac OS X v10.4
: Copies the colors of a palette into a color table. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`ResizePalette` (page 32) Deprecated in Mac OS X v10.4
: Changes the size of a palette. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

`RestoreDeviceClut` (page 34) Deprecated in Mac OS X v10.4
: Sets the color table of a graphics device to its default state. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Miscellaneous

`GetGray` (page 23) Deprecated in Mac OS X v10.4
: Determines the best intermediate color between two colors on a given graphics device. (Deprecated. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Functions

SetDepth

Changes the pixel depth of a video device. (**Deprecated.** There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

```
OSErr SetDepth (
   GDHandle gd,
   short depth,
   short whichFlags,
   short flags
);
```

Parameters

gd
: A handle to the `GDevice` structure of the video device whose pixel depth you wish to change.

depth

The mode ID returned by the `HasDepth` (page 26) function indicating that the video device supports the desired pixel depth. Alternatively, you can pass the desired pixel depth directly in this parameter, although you should use the `HasDepth` function to ensure that the device supports this depth.

whichFlags

The `gdDevType` constant, which represents a bit in the `gdFlags` field of the `GDevice` structure. (If this bit is set to 0 in the `GDevice` structure, the video device is black and white; if the bit is set to 1, the device supports color.

flags

The value 0 or 1. If you pass 0 in this parameter, the `SetDepth` function changes the video device to black and white; if you pass 1 in this parameter, `SetDepth` changes the video device to color.

Return Value
Returns zero if successful, or it returns a nonzero value if it cannot impose the desired depth and display mode on the requested device.

Discussion
The `SetDepth` function does not change the `'scrn'` resource; when the system is restarted, the original depth for this device is restored.

The Monitors control panel is the user interface for changing the pixel depth, color capabilities, and positions of video devices. Since the user can control the capabilities of the video device, your application should be flexible: although it may have a preferred pixel depth, your application should do its best to accommodate less than ideal conditions.

Use `SetDepth` only if your application can run on devices of a particular pixel depth and is unable to adapt to any other depth. Your application should display a dialog box that offers the user a choice between changing to that depth or canceling display of the image before your application uses `SetDepth`. Such a dialog box saves the user the trouble of going to the Monitors control panel before returning to your application.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

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

Declared In
`ImageCompression.k.h`

Data Types

ColorInfo

Specifies color information for each color in a palette.

```
struct ColorInfo {
    RGBColor ciRGB;
    short ciUsage;
    short ciTolerance;
    short ciDataFields[3];
};
typedef struct ColorInfo ColorInfo;
typedef ColorInfo * ColorInfoPtr;
```

Fields

`ciRGB`

An RGB color value, which is defined by the `RGBColor` structure. It contains three fields that contain integer values for defining, respectively, the red, green, and blue values of the color.

`ciUsage`

One or more of the usage constants, specifying how this entry is to be used. The `ciUsage` field can contain any of the `Usage Constants` (page 11).

`ciTolerance`

An integer expressing the range in RGB space within which the red, green, and blue values must fall to satisfy this entry. A tolerance value of $0000 means that only an exact match is acceptable. Values of $0xxx other than $0000 are reserved and should not be used in applications.

`ciDataFields`

Private fields.

Discussion

Each color information structure in a palette comprises an RGB color value, information describing how the color is to be used, a tolerance value for colors that need only be approximated, and private fields. You should not create and modify the public fields directly; instead, use Palette Manager functions such as `SetEntryColor` and `SetEntryUsage`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Palettes.h`

Palette

Represents a set of colors optimized for use on display devices with a limited number of colors.

```
struct Palette {
    short pmEntries;
    short pmDataFields[7];
    ColorInfo pmInfo[1];
};
typedef struct Palette Palette;
typedef Palette * PalettePtr;
```

Fields

`pmEntries`

The number of `ColorInfo` structures in the `pmInfo` array.

`pmDataFields`

Private fields used by the Palette Manager.

`pmInfo`
An array of `ColorInfo` (page 9) structures.

Discussion
A palette structure contains a header and a collection of color information structures, one for each color in the palette.

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

Declared In
`Palettes.h`

Constants

Usage Constants

Constants used to determine how the Palette Manager uses a specific palette color.

```
enum {
    pmCourteous = 0,
    pmDithered = 0x0001,
    pmTolerant = 0x0002,
    pmAnimated = 0x0004,
    pmExplicit = 0x0008,
    pmWhite = 0x0010,
    pmBlack = 0x0020,
    pmInhibitG2 = 0x0100,
    pmInhibitC2 = 0x0200,
    pmInhibitG4 = 0x0400,
    pmInhibitC4 = 0x0800,
    pmInhibitG8 = 0x1000,
    pmInhibitC8 = 0x2000
};
```

Constants
`pmCourteous`
The color accepts whatever value the Color Manager determines to be the closest match currently available in the device color table.

Available in Mac OS X v10.0 and later.

Declared in `Palettes.h`.

`pmDithered`
[description forthcoming]

Available in Mac OS X v10.0 and later.

Declared in `Palettes.h`.

`pmTolerant`
The color accepts the Color Manager's choices on an indexed device.

Available in Mac OS X v10.0 and later.

Declared in `Palettes.h`.

`pmAnimated`
: The color is used for special color animation effects.

Available in Mac OS X v10.0 and later.

Declared in `Palettes.h`.

`pmExplicit`
: The color specifies an index value rather than an RGB color and always generates the corresponding entry from the device's color table.

Available in Mac OS X v10.0 and later.

Declared in `Palettes.h`.

`pmWhite`
: Assign color to white on a 1-bit device.

Available in Mac OS X v10.0 and later.

Declared in `Palettes.h`.

`pmBlack`
: Assign color to black on a 1-bit device.

Available in Mac OS X v10.0 and later.

Declared in `Palettes.h`.

`pmInhibitG2`
: Inhibit on 2-bit grayscale device.

Available in Mac OS X v10.0 and later.

Declared in `Palettes.h`.

`pmInhibitC2`
: Inhibit on 2-bit color device.

Available in Mac OS X v10.0 and later.

Declared in `Palettes.h`.

`pmInhibitG4`
: Inhibit on 4-bit grayscale device.

Available in Mac OS X v10.0 and later.

Declared in `Palettes.h`.

`pmInhibitC4`
: Inhibit on 4-bit color device.

Available in Mac OS X v10.0 and later.

Declared in `Palettes.h`.

`pmInhibitG8`
: Inhibit on 8-bit grayscale device.

Available in Mac OS X v10.0 and later.

Declared in `Palettes.h`.

`pmInhibitC8`
: Inhibit on 8-bit color device.

Available in Mac OS X v10.0 and later.

Declared in `Palettes.h`.

Discussion
You can logically `AND` these constants in certain combinations to specify the value of `ciUsage` in a `ColorInfo` (page 9) record.

The inhibit constants are used to indicate that a graphics device should be prevented from displaying the color at specified pixel depths. They are always used in combination with other usage constants.

Update Constants

Constants used to determine whether a window is updated based on various changes to the color environment.

```
enum {
    pmNoUpdates = 0x8000,
    pmBkUpdates = 0xA000,
    pmFgUpdates = 0xC000,
    pmAllUpdates = 0xE000
};
```

Constants

`pmNoUpdates`
Do not update the window when its color environment changes.
Available in Mac OS X v10.0 and later.
Declared in `Palettes.h`.

`pmBkUpdates`
Update the window only when it is not the active window.
Available in Mac OS X v10.0 and later.
Declared in `Palettes.h`.

`pmFgUpdates`
Update the window only when it is the active window.
Available in Mac OS X v10.0 and later.
Declared in `Palettes.h`.

`pmAllUpdates`
Update the window whenever its color environment changes.
Available in Mac OS X v10.0 and later.
Declared in `Palettes.h`.

Discussion
You use these constants in the `nCUpdates` parameter of the `NSetPalette` (page 28) function and the `updates` parameter of the `SetPaletteUpdates` (page 39) function.

Deprecated Palette Manager Reference (Not Recommended) Functions

A function identified as deprecated has been superseded and may become unsupported in the future.

Deprecated in Mac OS X v10.4

ActivatePalette

Changes the device color tables and generates window updates as needed to meet the color requirements of your window. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void ActivatePalette (
   WindowRef srcWindow
);
```

Parameters

srcWindow

A pointer to the window for which you want status changes reported.

Discussion

The Window Manager calls `ActivatePalette` when your window's status changes—for example, when your window opens, closes, moves, or becomes frontmost. You need to call the `ActivatePalette` function yourself if you change a palette—for example, by changing a color with the `SetEntryColor` function—and you want the changes to take place immediately, before the Window Manager would do it.

If the window specified in the `srcWindow` parameter is frontmost, `ActivatePalette` examines the information stored in the window's palette and attempts to provide the color environment described therein. It determines a list of devices on which to render the palette by intersecting the port rectangle of the window with each device. If the intersection is not empty and if the device has a color table, then `ActivatePalette` checks to see if the color environment is sufficient. If a change is required, `ActivatePalette` calls QuickDraw to reserve or modify the device's color entries as needed. The `ActivatePalette` function then generates update events for all windows that need color updates.

Calling `ActivatePalette` with an offscreen graphics world has no effect.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

AnimateEntry

Changes the color of a window's palette entry. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void AnimateEntry (
   WindowRef dstWindow,
   short dstEntry,
   const RGBColor *srcRGB
);
```

Parameters

dstWindow
A pointer to the window whose palette color is to be changed.

dstEntry
The palette entry to be changed.

srcRGB
A pointer to the new RGB value.

Discussion
The `AnimateEntry` function changes the RGB value of an animated entry for a window's palette. Each device for which that index has been reserved is immediately modified to contain the new value. This is not considered to be a change to the device's color environment because no other windows should be using the animated entry.

If the palette entry is not an animated color or if the associated indexes are no longer reserved, no animation occurs.

If you have blocked color updates in a window by using `SetPalette` with `cUpdates` set to `FALSE`, you may observe unintentional animation. This occurs when `ActivatePalette` reserves for animation device indexes that are already used in the window. Redrawing the window, which normally is the result of a color update event, removes any animated colors that do not belong to the window.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In
`Palettes.h`

AnimatePalette

Changes the colors of a series of palette entries; it is similar to the `AnimateEntry` function, but it acts upon a range of entries. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void AnimatePalette (
    WindowRef dstWindow,
    CTabHandle srcCTab,
    short srcIndex,
    short dstEntry,
    short dstLength
);
```

Parameters

dstWindow
A pointer to the window whose palette colors are to be changed.

srcCTab
A handle to the color table containing the new colors.

srcIndex
The source color table entry at which copying starts.

dstEntry
The palette entry at which copying starts.

dstLength
The number of palette entries to be changed.

Discussion
The `AnimatePalette` function changes the colors of a series of palette entries. Beginning at the index specified by the `srcIndex` parameter (which has a minimum value of 0), the number of entries specified in `dstLength` are copied from the source color table to the destination window's palette, beginning at the entry specified in the `dstEntry` parameter. If the source color table specified in `srcCTab` is not sufficiently large to accommodate the request, `AnimatePalette` modifies as many entries as possible and leaves the remaining entries unchanged.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

CopyPalette

Copies entries from one palette to another. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void CopyPalette (
   PaletteHandle srcPalette,
   PaletteHandle dstPalette,
   short srcEntry,
   short dstEntry,
   short dstLength
);
```

Parameters

srcPalette
A handle to the palette from which colors are copied.

dstPalette
A handle to the palette to which colors are copied.

srcEntry
The source palette entry at which copying starts.

dstEntry
The destination palette entry at which copying starts.

dstLength
The number of destination palette entries to change.

Discussion
The `CopyPalette` function copies entries from the source palette into the destination palette. The copy operation begins at the values specified by the `srcEntry` and `dstEntry` parameters, copying into as many entries as are specified by the `dstLength` parameter. `CopyPalette` resizes the destination palette when the number of entries after the copy operation is greater than it was before the copy operation.

`CopyPalette` does not call `ActivatePalette`, so your application is free to change the palette a number of times without causing a series of intermediate changes to the color environment. Your application should call `ActivatePalette` after completing all palette changes.

If either of the palette handles is `NULL`, `CopyPalette` does nothing.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In
`Palettes.h`

CTab2Palette

Copies the colors of a color table into a palette. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void CTab2Palette (
   CTabHandle srcCTab,
   PaletteHandle dstPalette,
   short srcUsage,
   short srcTolerance
);
```

Parameters

srcCTab

A handle to the color table whose colors are to be copied.

dstPalette

The palette to receive the colors.

srcUsage

A usage constant to be assigned the palette entries. Usage constants are described in "Update Constants" (page 13).

srcTolerance

A tolerance value to be assigned the palette entries.

Discussion
The `CTab2Palette` function copies the fields from an existing color-table structure into an existing palette structure. If the structures are not the same size, then `CTab2Palette` resizes the palette structure to match the number of entries in the color-table structure. If the palette in `dstPalette` has any entries allocated for animation on any screen device, they are relinquished before the new colors are copied. The `srcUsage` and `srcTolerance` parameters are the value that you assign to the new colors.

If you want to use color-table animation, you can use `AnimateEntry` (page 16) and `AnimatePalette` (page 17) to change the colors in a palette and on corresponding devices. Changes made to a palette by `CTab2Palette` do not take effect until the next `ActivatePalette` function is performed. If either the color-table handle or the palette handle is `NULL`, `CTab2Palette` does nothing.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

DisposePalette

Disposes of a palette. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void DisposePalette (
   PaletteHandle srcPalette
);
```

Parameters

srcPalette

A handle to the palette to be disposed of.

Discussion
If the palette has any entries allocated for animation on any screen device, then `DisposePalette` relinquishes these entries before the palette's memory is released.

If a palette is attached to a window automatically—because the palette resource and the window have the same ID—you do not have to call the `DisposePalette` function to dispose of the function. The Palette Manager and Window Manager dispose of the palette automatically if the palette is replaced or if the window goes away.

However, if you explicitly attach a palette to a window with the `SetPalette` or `NSetPalette` function, your application owns the palette and is responsible for disposing of it. It is possible to attach a single palette to multiple windows; therefore, even when a window goes away and no longer needs a palette, other windows may still need it.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

Entry2Index

Obtains the index for a specified entry in the current graphics port's palette on the current device. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
SInt32 Entry2Index (
   short entry
);
```

Parameters

entry

The palette entry whose equivalent device index is to be returned.

Return Value

The index of the given `entry`.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Palettes.h`

GetEntryColor

Obtains the color of a palette entry. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void GetEntryColor (
   PaletteHandle srcPalette,
   short srcEntry,
   RGBColor *dstRGB
);
```

Parameters

srcPalette

A handle to the palette to be accessed.

srcEntry

The palette entry whose color is desired.

dstRGB

A pointer to an RGB color structure to receive the palette color.

Discussion

You can modify the entry's color using the `SetEntryColor` (page 36) function.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

GetEntryUsage

Obtains the usage and tolerance fields of a palette entry. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void GetEntryUsage (
   PaletteHandle srcPalette,
   short srcEntry,
   short *dstUsage,
   short *dstTolerance
);
```

Parameters

srcPalette
A handle to the palette to be accessed.

srcEntry
The palette entry whose usage and tolerance are desired.

dstUsage
A pointer to the usage value of the palette entry.

dstTolerance
A pointer to the tolerance value of the palette entry.

Discussion
You can modify the entry's usage and tolerance values by using the `SetEntryUsage` (page 37) function.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

GetGray

Determines the best intermediate color between two colors on a given graphics device. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

```
Boolean GetGray (
   GDHandle device,
   const RGBColor *backGround,
   RGBColor *foreGround
);
```

Parameters

device

A handle to the graphics device for which an intermediate color or gray is needed.

backGround

The `RGBColor` structure for one of the two colors for which you want an intermediate color.

foreGround

On input, the `RGBColor` structure for the other of the two colors; upon completion, the best intermediate color between these two.

Return Value

If no gray is available (or if no distinguishable third color is available), the `foreGround` parameter is unchanged, and the function returns `FALSE`. If at least one gray or intermediate color is available, it is returned in the `foreGround` parameter, and the function returns `TRUE`.

Discussion

The `GetGray` function determines the midpoint values for the red, green, and blue values of the two colors you specify in the `backGround` and `foreGround` parameters.

You can also use `GetGray` to return the best gray. For example, when dimming an object, supply black and white as the two colors, and `GetGray` returns the best available gray that lies between them.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Palettes.h`

GetNewPalette

Creates and initializes a palette from a `'pltt'` resource. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
PaletteHandle GetNewPalette (
   short PaletteID
);
```

Parameters

PaletteID

The resource ID of the source palette.

Return Value

A handle to the new palette.

Discussion

The `GetNewPalette` function detaches the resource when it creates the new palette, so you do not need to call the `ReleaseResource` function.

If you open a new color window with `GetNewCWindow`, the Window Manager calls `GetNewPalette` automatically, with `paletteID` equal to the window's resource ID. Therefore, if you have created a palette resource with the same ID as a window, the Window Manager and Palette Manager automatically create the palette for you and your application need not call `GetNewPalette` to create the palette.

To attach a palette to a window after creating it, use the `SetPalette` (page 38) function. To change the entries in a palette after creating it, use the `SetEntryColor` (page 36) and `SetEntryUsage` (page 37) functions.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Palettes.h`

GetPalette

Obtains a window's palette. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
PaletteHandle GetPalette (
   WindowRef srcWindow
);
```

Parameters

srcWindow

A pointer to the window for which you want the associated palette.

Return Value
A handle to the palette associated with the window specified in the `srcWindow` parameter or `NULL` if the window has no associated palette or is not a color window.

Discussion
Normally, the `GetPalette` function does not allocate memory, with one exception. When your application calls `GetPalette` to get a copy of the default application palette, the Palette Manager looks at the `AppPalette` global variable. If `AppPalette` is `NULL`, `GetPalette` makes a copy of the default system palette and returns a handle to this copy.

You request the default palette as follows:

myPaletteHndl = GetPalette ((WindowPtr) -1);

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

GetPaletteUpdates

Obtains the update attribute of a palette. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
short GetPaletteUpdates (
   PaletteHandle p
);
```

Parameters

p
A handle to the palette.

Return Value
One of the update attributes described in "Update Constants" (page 13).

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

HasDepth

Determines whether a video device supports a specific pixel depth. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

```
short HasDepth (
   GDHandle gd,
   short depth,
   short whichFlags,
   short flags
);
```

Parameters

gd
A handle to the `GDevice` structure of the video device.

depth
The pixel depth for which you're testing.

whichFlags
The `gdDevType` constant, which represents a bit in the `gdFlags` field of the `GDevice` structure. If this bit is set to 0 in the `GDevice` structure, the video device is black and white; if the bit is set to 1, the device supports color.

flags
The value 0 or 1. If you pass 0 in this parameter, the `HasDepth` function tests whether the video device is black and white. If you pass 1 in this parameter, `HasDepth` tests whether the video device supports color.

Return Value
Returns 0 if the device does not support the depth you specify in the `depth` parameter or the display mode you specify in the `flags` parameter. Any other value indicates that the device supports the specified depth and display mode. The function result contains the mode ID that QuickDraw passes to the video driver to set its pixel depth and to specify color or black and white. You can pass this mode ID in the `depth` parameter for the `SetDepth` function to set the graphics device to the pixel depth and display mode for which you tested.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In
`Palettes.h`

InitPalettes

Initializes the Palette Manager. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void InitPalettes (
   void
);
```

Discussion
The `InitPalettes` function searches for devices that support a device color table and initializes an internal data structure for each one. Your application does not have to call `InitPalettes` because the Window Manager's `InitWindows` function calls it automatically.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

NewPalette

Allocates a new palette from colors in the color table. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
PaletteHandle NewPalette (
   short entries,
   CTabHandle srcColors,
   short srcUsage,
   short srcTolerance
);
```

Parameters

entries
The number of `ColorInfo` structures to be created in the new palette.

srcColors
A handle to the color table from which the colors are to be obtained. If no color table is provided (srcColors = NULL), then all colors in the palette are set to black (red, green, and blue equal to $0000).

srcUsage
The usage value to be assigned each ColorInfo structure in the palette.

srcTolerance
The tolerance value to be assigned each ColorInfo structure in the palette.

Return Value
A handle to the new palette.

Discussion
The NewPalette function fills the palette with as many RGB values from the color table as it has or can fit.

To attach a palette to a window after creating it, use the SetPalette (page 38) function. To change the entries in a palette after creating it, use the SetEntryColor (page 36) and SetEntryUsage (page 37) functions.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
Palettes.h

NSetPalette

Associates a new palette with a window. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void NSetPalette (
   WindowRef dstWindow,
   PaletteHandle srcPalette,
   short nCUpdates
);
```

Parameters

dstWindow
A pointer to the window to which you want to assign a new palette.

srcPalette
A handle to the palette you want to assign.

nCUpdates

An integer value in which you specify whether the window is to receive updates as a result of various changes to the color environment. See "Update Constants" (page 13) for a description of the update options.

Discussion

`NSetPalette` changes the palette associated with the window specified in the `dstWindow` parameter to the palette specified by `srcPalette`. `NSetPalette` also records whether the window is to receive updates as a result of changes to its color environment. The update constants, which you pass to the `nCUpdates` parameter, determine when the window is updated.

This function is identical to the `SetPalette` (page 38) function except that the `nCUpdates` parameter is an integer rather than a Boolean value, so that a variety of conditions can trigger an update event.

Use the `SetPalette` (page 38) function if you do not need the flexibility that `NSetPalette` provides for update events.

Use the `GetNewPalette` (page 23) function or the `NewPalette` (page 27) function to create a new palette. To dispose of a palette, use the `DisposePalette` (page 20) function.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Palettes.h`

Palette2CTab

Copies the colors of a palette into a color table. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void Palette2CTab (
   PaletteHandle srcPalette,
   CTabHandle dstCTab
);
```

Parameters

srcPalette

A handle to the palette whose colors are to be used.

dstCTab

A handle to the color table to receive the colors.

Discussion
The `Palette2CTab` function copies all of the colors from an existing palette structure into an existing color-table structure. If the structures are not the same size, then `Palette2CTab` resizes the color-table structure to match the number of entries in the palette structure. If either the palette handle or the color-table handle is `NULL`, `Palette2CTab` does nothing.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

PmBackColor

Sets the background color field of the current graphics port to a palette color. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void PmBackColor (
   short dstEntry
);
```

Parameters
`dstEntry`
The palette entry whose color is to be used as the background color.

Discussion
The `PmBackColor` function sets the current color graphics port's `rgbBkColor` field to match the color in the entry specified by the `dstEntry` parameter of the palette associated with the current window structure. For courteous and tolerant entries, `PmBackColor` calls the `RGBBackColor` function using the RGB color of the palette entry. For animated colors, `PmBackColor` selects the recorded device index previously reserved for animation (if still present) and installs it in the color graphics port. The `rgbBgColor` field is set to the value from the palette entry. For explicit colors, `PmBackColor` places the value

```
dstEntry modulo (maxIndex +1)
```

into the color graphics port, where `maxIndex` is the largest index available in a device's color table. When multiple devices with different depths are present, `maxIndex` varies appropriately for each device.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

PmForeColor

Sets the foreground color field of the current graphics port to a palette color. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void PmForeColor (
   short dstEntry
);
```

Parameters

`dstEntry`

The palette entry whose color is to be used as the foreground color.

Discussion
The `PmForeColor` function sets the current color graphics port's `rgbFgColor` field to match the color in the entry specified by the `dstEntry` parameter of the palette associated with the current window structure. For courteous and tolerant entries, `PmForeColor` calls the `RGBForeColor` function using the RGB color of the palette entry. For animated colors, `PmForeColor` selects the recorded device index previously reserved for animation (if still present) and installs it in the color graphics port. The RGB foreground color field is set to the value from the palette entry. For explicit colors, `PmForeColor` places the value

```
dstEntry modulo (maxIndex +1)
```

into the color graphics port, where `maxIndex` is the largest index available in a device's color table. When multiple devices with different depths are present, the value of `maxIndex` varies appropriately for each device.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

PMgrVersion

Determines which version of the Palette Manager is executing; it returns an integer specifying the version number. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

```
short PMgrVersion (
   void
);
```

Return Value

`PMgrVersion` returns $0202 if system software version 7.0 is executing; $0201 if system software version 6.0.5 is executing; and $0200 if the original 32-Bit QuickDraw system extension is executing.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Palettes.h`

ResizePalette

Changes the size of a palette. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void ResizePalette (
    PaletteHandle p,
    short size
);
```

Parameters

p

A handle to the palette to be resized.

size

The number of resulting entries in the palette.

Discussion

The `ResizePalette` function sets the palette specified in `srcPalette` to the number of entries indicated in the `size` parameter. If `ResizePalette` adds entries at the end of the palette, it sets them to `pmCourteous`, with the RGB values set to (0,0,0)—that is, black. If `ResizePalette` deletes entries from the end of the palette, it safely disposes of them.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Palettes.h`

RestoreBack

Sets the current background color to the color you specify. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void RestoreBack (
   const ColorSpec *c
);
```

Parameters

c

A pointer to the `ColorSpec` structure containing the RGB color to be set as the background color. If you specify 0 in the `value` field of the `ColorSpec` structure, the `RestoreBack` function stores the RGB value in the `rgbFgColor` field of the current `CGrafPort` structure. If you specify 1 in the `value` field of the `ColorSpec` structure, the `RestoreBack` function stores the RGB value in the `pmBkColor` field of the `GrafVars` structure.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Palettes.h`

RestoreDeviceClut

Sets the color table of a graphics device to its default state. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void RestoreDeviceClut (
   GDHandle gd
);
```

Parameters

gd

A handle to the `GDevice` structure. Pass `NULL` in the `gdh` parameter to restore all screens.

Discussion

The `RestoreDeviceClut` function changes the color table of the device specified by the `gdh` parameter to its default state. If this process changes any entries, the Palette Manager posts color updates to windows intersecting the device.

You do not need to use this function to restore the Finder's desktop colors, because its colors are automatically restored upon switching from applications that use the Palette Manager. Likewise, you need not worry when switching to another application that uses the Palette Manager. Although colors are not automatically restored in this case, if that application needs a certain set of colors, the Palette Manager provides them the moment that application comes to the front.

The reason to use `RestoreDeviceClut` is that you may be switching to an application that does not use the Palette Manager, in which case that application inherits your palette unless you restore the default color lookup tables for all the available display devices.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Palettes.h`

RestoreFore

Sets the current foreground color to the color you supply. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void RestoreFore (
   const ColorSpec *c
);
```

Parameters

c

A pointer to the `ColorSpec` structure containing the RGB color to be set as the foreground color. If you specify 0 in the `value` field of the `ColorSpec` structure, the `RestoreFore` function stores the RGB value in the `rgbFgColor` field of the current `CGrafPort` structure. If you specify 1 in the `value` field of the `ColorSpec` structure, the `RestoreFore` function stores the RGB value in the `pmFgColor` field of the `GrafVars` structure.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

SaveBack

Saves the current background color. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void SaveBack (
   ColorSpec *c
);
```

Parameters

c

A pointer to the `ColorSpec` structure to receive the current background color. A value of 0 in the `value` field of the `ColorSpec` structure specifies retrieving the RGB color from the `rgbBkColor` field of the `CGrafPort` structure; a value of 1 in the `value` field specifies retrieving the palette entry from the `pmBkColor` field of the `GrafVars` structure.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

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

Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

SaveFore

Saves the current foreground color. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void SaveFore (
   ColorSpec *c
);
```

Parameters

c

A pointer to the `ColorSpec` structure to hold the current foreground color. A value of 0 in the `value` field of the `ColorSpec` structure specifies retrieving the RGB color from the `rgbFgColor` field of the `CGrafPort` structure; a value of 1 in the `value` field specifies retrieving the palette entry from the `pmFgColor` field of the `GrafVars` structure.On return, `ColorSpec` structure holds the current foreground color. You can save either QuickDraw's foreground color from the `CGrafPort` structure or the Palette Manager's foreground color from the `GrafVars` structure.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

SetEntryColor

Changes the color of a palette entry. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void SetEntryColor (
   PaletteHandle dstPalette,
   short dstEntry,
   const RGBColor *srcRGB
);
```

Parameters

dstPalette

The palette whose entry color is to be changed.

dstEntry

The palette entry to be changed.

srcRGB

A pointer to the new RGB color value.

Discussion

`SetEntryColor` marks the entry as having changed, but it does not change the color environment. That change occurs upon the next call to `ActivatePalette`. `SetEntryColor` marks modified entries such that the palette is updated, even though no update is required by a change in the color environment.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Palettes.h`

SetEntryUsage

Modifies the usage category and tolerance values of a palette entry. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void SetEntryUsage (
   PaletteHandle dstPalette,
   short dstEntry,
   short srcUsage,
   short srcTolerance
);
```

Parameters

dstPalette

A handle to the palette to be modified.

dstEntry

The palette entry.

srcUsage
The new usage value; one or more usage constants.

srcTolerance
The new tolerance value.

Discussion
`SetEntryUsage` marks the entry as having changed, but it does not change the color environment. That change occurs upon the next call to `ActivatePalette`. Modified entries are marked such that the palette is updated even though no update is required by a change in the color environment. If either `srcUsage` or `srcTolerance` is set to $FFFF (–1), the entries are not changed.

This function allows you to easily modify a palette created with `NewPalette` or modified by `CTab2Palette`. For such palettes the `ciUsage` and `ciTolerance` fields of the `ColorInfo` structure are the same because you can designate only one value for each. You typically call `SetEntryUsage` after `NewPalette` or `CTab2Palette` to adjust and customize your palette.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Palettes.h`

SetPalette

Associates a palette with a window. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void SetPalette (
   WindowRef dstWindow,
   PaletteHandle srcPalette,
   Boolean cUpdates
);
```

Parameters

dstWindow
A pointer to the window to which you want to assign a new palette.

srcPalette
A handle to the palette you want to assign.

cUpdates

A Boolean value in which your application specifies whether the window is to receive updates as a result of changes to the color environment. Specify `TRUE` if you want the window to be updated, even if the window is not the frontmost window. When a window is the frontmost window, changes to its palette cause it to get an update event regardless of how the `cUpdates` parameter is set.

Discussion

You can use the `NSetPalette` (page 28) function, which does the same thing as `SetPalette`, when you need greater flexibility in setting criteria for updates. The `nCUpdates` parameter for the `NSetPalette` function includes the option of turning off updates when the window is the frontmost window.

Use the `NSetPalette` function to associate a palette with a window but with additional options as to when an update event is triggered by changes to the color environment.

Use the `GetNewPalette` (page 23) function or the `NewPalette` (page 27) function to create a new palette. To dispose of a palette, use the `DisposePalette` (page 20) function.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Palettes.h`

SetPaletteUpdates

Sets the update attribute of a palette. (Deprecated in Mac OS X v10.4. There is no replacement; 8-bit graphics mode is not supported by the Mac OS X GUI.)

Not recommended.

```
void SetPaletteUpdates (
   PaletteHandle p,
   short updates
);
```

Parameters

p

A handle to the palette.

updates

One of the update attributes for the `NSetPalette` function. See "Update Constants" (page 13) for a description of the update attributes.

Special Considerations

For applications running in Mac OS X, the Palette Manager is no longer relevant because display devices always support direct color (pixel depth of 16 or 32 bits). The palette-based graphics model only works in 256-color (8-bit pseudocolor) modes of operation, which are not supported for the Mac OS X GUI.

There is some support for palettes in Quartz Services; see *Quartz Display Services Reference*.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Palettes.h`

Document Revision History

This table describes the changes to *Palette Manager Reference*.

Date	Notes
2006-07-13	Updated for Mac OS X v10.5.
2006-07-24	Deprecated the manager. There is no replacement.
2003-07-31	Most functions are not recommended for new application development. For more information, see "Palette Manager Reference" (page 8).
2003-01-01	The information in this document replaces information published previously in *Inside Macintosh: Advanced Color Imaging*.

Document Revision History

Index

A

ActivatePalette function (Deprecated in Mac OS X v10.4) 15
AnimateEntry function (Deprecated in Mac OS X v10.4) 16
AnimatePalette function (Deprecated in Mac OS X v10.4) 17

C

ColorInfo structure 9
CopyPalette function (Deprecated in Mac OS X v10.4) 18
CTab2Palette function (Deprecated in Mac OS X v10.4) 19

D

DisposePalette function (Deprecated in Mac OS X v10.4) 20

E

Entry2Index function (Deprecated in Mac OS X v10.4) 20

G

GetEntryColor function (Deprecated in Mac OS X v10.4) 21
GetEntryUsage function (Deprecated in Mac OS X v10.4) 22
GetGray function (Deprecated in Mac OS X v10.4) 23
GetNewPalette function (Deprecated in Mac OS X v10.4) 23
GetPalette function (Deprecated in Mac OS X v10.4) 24
GetPaletteUpdates function (Deprecated in Mac OS X v10.4) 25

H

HasDepth function (Deprecated in Mac OS X v10.4) 26

I

InitPalettes function (Deprecated in Mac OS X v10.4) 27

N

NewPalette function (Deprecated in Mac OS X v10.4) 27
NSetPalette function (Deprecated in Mac OS X v10.4) 28

P

Palette structure 10
Palette2CTab function (Deprecated in Mac OS X v10.4) 29
pmAllUpdates constant 13
pmAnimated constant 12
PmBackColor function (Deprecated in Mac OS X v10.4) 30
pmBkUpdates constant 13
pmBlack constant 12
pmCourteous constant 11
pmDithered constant 11
pmExplicit constant 12
pmFgUpdates constant 13

PmForeColor function (Deprecated in Mac OS X v10.4) 31
PMgrVersion function (Deprecated in Mac OS X v10.4) 32
pmInhibitC2 constant 12
pmInhibitC4 constant 12
pmInhibitC8 constant 12
pmInhibitG2 constant 12
pmInhibitG4 constant 12
pmInhibitG8 constant 12
pmNoUpdates constant 13
pmTolerant constant 11
pmWhite constant 12

R

ResizePalette function (Deprecated in Mac OS X v10.4) 32
RestoreBack function (Deprecated in Mac OS X v10.4) 33
RestoreDeviceClut function (Deprecated in Mac OS X v10.4) 34
RestoreFore function (Deprecated in Mac OS X v10.4) 34

S

SaveBack function (Deprecated in Mac OS X v10.4) 35
SaveFore function (Deprecated in Mac OS X v10.4) 36
SetDepth function 8
SetEntryColor function (Deprecated in Mac OS X v10.4) 36
SetEntryUsage function (Deprecated in Mac OS X v10.4) 37
SetPalette function (Deprecated in Mac OS X v10.4) 38
SetPaletteUpdates function (Deprecated in Mac OS X v10.4) 39

U

Update Constants 13
Usage Constants 11