Color Picker Manager Reference

Carbon > Graphics & Imaging

2007-07-02

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

Simultaneously published in the United States and Canada.

Contents

Tables

Color Picker Manager Reference

Framework: Carbon/Carbon.h

Declared in ColorPicker.h

Overview

Using Color Picker Manager functions, your application can use the standard user interface for soliciting color choices from users. Your application can use these functions to display, respond to events within, and close a color picker dialog box. The Color Picker Manager provides standard color pickers (which allow users to select colors from ranges of colors) and a standard dialog box allowing users to interact with the color picker. Your application can also create its own color pickers and dialog boxes to work with the Color Picker Manager.

Carbon supports the Color Picker Manager functions that most applications rely on, `GetColor`, `PickColor`, and `NPickColor`. In addition, Carbon will support all the functions described as supported in "Technote 1100: Color Picker 2.1."

That same Technote specifies certain functions from earlier versions of the Color Picker Manager that are no longer supported by the Mac OS; Carbon will not support these functions. In addition, many of the functions that start with the word "Picker", such as `PickerInit`, `PickerGetProfile`, and `PickerItemHit`, are no longer supported. These routines provided low-level access to the Color Picker Manager that was rarely used.

Functions by Task

Converting Between SmallFract and Fixed Values

`Fix2SmallFract` (page 10)
Converts a fixed integer to a `SmallFract` value.

`SmallFract2Fix` (page 18)
Converts a `SmallFract` value to a fixed integer.

Converting Colors Among Color Models

`CMY2RGB` (page 9)
Converts a CMY color to its equivalent RGB color.

`HSL2RGB` (page 12)
Converts an HSL color to an RGB color.

`HSV2RGB` (page 12)
Converts an HSV color to an RGB color.

`RGB2CMY` (page 17)
Converts an RGB color to a CMY color.

`RGB2HSL` (page 17)
Converts an RGB color to an HSL color.

`RGB2HSV` (page 17)
Converts an RGB color to an HSV color.

Using the Standard Color Picker Dialog Box

`NPickColor` (page 15)
Displays the Color Picker dialog.

`GetColor` (page 11)
Requests the user to choose a color. This function is obsolete; use the `PickColor` function instead.

`PickColor` (page 16)
Requests the user to choose a color from the standard color picker dialog box.

Working With Universal Procedure Pointers

`NewUserEventUPP` (page 15)
Creates a universal procedure pointer (UPP) to an event filter callback.

`DisposeUserEventUPP` (page 10)
Disposes of a a universal procedure pointer (UPP) to an event filter callback.

`InvokeUserEventUPP` (page 13)
Invokes an event filter callback.

`NewNColorChangedUPP` (page 14)
Creates a universal procedure pointer (UPP) to a color-changed callback.

`InvokeNColorChangedUPP` (page 13)
Invokes a color-changed callback.

`DisposeNColorChangedUPP` (page 9)
Disposes of a a universal procedure pointer (UPP) to a color-changed callback.

`NewColorChangedUPP` (page 14)
Creates a universal procedure pointer (UPP) to a color-changed callback.

`InvokeColorChangedUPP` (page 13)
Invokes a color-changed callback.

`DisposeColorChangedUPP` (page 9)
Disposes of a a universal procedure pointer (UPP) to a color-changed callback.

Functions

CMY2RGB

Converts a CMY color to its equivalent RGB color.

```
void CMY2RGB (
   const CMYColor *cColor,
   RGBColor *rColor
);
```

Parameters

cColor

A pointer to a `CMYColor` structure to be converted.

rColor

A pointer to an `RGBColor` structure for the converted color.

Availability

Available in CarbonLib 1.0 and later when running Mac OS 8.1 or later.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`ColorPicker.h`

DisposeColorChangedUPP

Disposes of a a universal procedure pointer (UPP) to a color-changed callback.

```
void DisposeColorChangedUPP (
   ColorChangedUPP userUPP
);
```

Parameters

userUPP

Availability

Available in CarbonLib 1.0 and later.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`ColorPicker.h`

DisposeNColorChangedUPP

Disposes of a a universal procedure pointer (UPP) to a color-changed callback.

```
void DisposeNColorChangedUPP (
   NColorChangedUPP userUPP
);
```

Parameters

userUPP

Availability
Available in CarbonLib 1.0 and later.
Available in Mac OS X 10.0 and later.

Declared In
ColorPicker.h

DisposeUserEventUPP

Disposes of a a universal procedure pointer (UPP) to an event filter callback.

```
void DisposeUserEventUPP (
   UserEventUPP userUPP
);
```

Parameters

userUPP

Availability
Available in CarbonLib 1.0 and later.
Available in Mac OS X 10.0 and later.
Not available to 64-bit applications.

Declared In
ColorPicker.h

Fix2SmallFract

Converts a fixed integer to a `SmallFract` value.

```
SmallFract Fix2SmallFract (
   Fixed f
);
```

Parameters

f

The value of type `Fixed` to be converted to a `SmallFract` value.

Return Value
A `SmallFract` converted from the fixed integer value specified in the `f` parameter. See the description of the `SmallFract` data type.

Discussion
A `SmallFract` value can represent a value between 0 and 65,535. Introduced by the original Color Picker Package, `SmallFract` values are used in `CMYColor`, `HSLColor`, and `HSVColor` structures. They can be assigned directly to and from integers.

Availability
Available in CarbonLib 1.0 and later when running Mac OS 8.1 or later.
Available in Mac OS X 10.0 and later.
Not available to 64-bit applications.

Declared In
`ColorPicker.h`

GetColor

Requests the user to choose a color. This function is obsolete; use the `PickColor` function instead.

```
Boolean GetColor (
   Point where,
   ConstStr255Param prompt,
   const RGBColor *inColor,
   RGBColor *outColor
);
```

Parameters

where

A point defining the location of the upper-left corner of the dialog box. If you set this parameter to (0,0), the dialog box is centered horizontally on the main screen, with one-third of the empty space above the box and two-thirds below, regardless of the screen size. If you set this parameter to (–1,–1), the `GetColor` function displays the dialog box on the screen supporting the greatest pixel depth.

prompt

Text for prompting the user to choose a color. This string is displayed in the upper-left corner of the dialog box.

inColor

A pointer to an `RGBColor` structure for a color at entry to the picker. This is the original color, which the user may want for comparison.

outColor

A pointer to an `RGBColor` structure describing the new color. This is set to the last color that the user picked before clicking OK. On entry, the `outColor` parameter is treated as undefined, so the output color sample initially matches the input. Although the color being picked may vary widely, the input color sample remains fixed, and clicking the input sample resets the output color sample to match it.

Return Value
`True` if the user clicks the OK button; `false` if the user clicks the Cancel button. In either case, the dialog box is removed.

Discussion
The `GetColor` function does not support ColorSync 1.0 color matching; however, the `PickColor` function does. This function was designed for use for version 1.0 of the Color Picker Package and is still supported for backward compatibility.

Availability
Available in CarbonLib 1.0 and later when running Mac OS 8.1 or later.
Available in Mac OS X 10.0 and later.

Declared In
`ColorPicker.h`

HSL2RGB

Converts an HSL color to an RGB color.

```
void HSL2RGB (
   const HSLColor *hColor,
   RGBColor *rColor
);
```

Parameters

hColor

A pointer to the `HSLColor` structure to be converted.

rColor

A pointer to an `RGBColor` structure for the converted color.

Availability
Available in CarbonLib 1.0 and later when running Mac OS 8.1 or later.
Available in Mac OS X 10.0 and later.
Not available to 64-bit applications.

Declared In
`ColorPicker.h`

HSV2RGB

Converts an HSV color to an RGB color.

```
void HSV2RGB (
   const HSVColor *hColor,
   RGBColor *rColor
);
```

Parameters

hColor

A pointer to the `HSVColor` structure to be converted.

rColor

A pointer to an `RGBColor` structure for the converted color.

Availability
Available in CarbonLib 1.0 and later when running Mac OS 8.1 or later.
Available in Mac OS X 10.0 and later.
Not available to 64-bit applications.

Declared In
`ColorPicker.h`

InvokeColorChangedUPP

Invokes a color-changed callback.

```
void InvokeColorChangedUPP (
   SInt32 userData,
   PMColor *newColor,
   ColorChangedUPP userUPP
);
```

Parameters

newColor

userUPP

Discussion

You should not need to use the function `InvokeColorChangedUPP`, as the Color Picker Manager calls your color-changed callback function for you.

Availability

Available in CarbonLib 1.0 and later.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`ColorPicker.h`

InvokeNColorChangedUPP

Invokes a color-changed callback.

```
void InvokeNColorChangedUPP (
   SRefCon userData,
   NPMColor *newColor,
   NColorChangedUPP userUPP
);
```

Parameters

newColor

userUPP

Discussion

You should not need to use the function `InvokeNColorChangedUPP`, as the Color Picker Manager calls your color-changed callback function for you.

Availability

Available in CarbonLib 1.0 and later.

Available in Mac OS X 10.0 and later.

Declared In

`ColorPicker.h`

InvokeUserEventUPP

Invokes an event filter callback.

```
Boolean InvokeUserEventUPP (
   EventRecord *event,
   UserEventUPP userUPP
);
```

Parameters

event

userUPP

Discussion

You should not need to use the function `InvokeUserEventUPP`, as the Color Picker Manager calls your event filter callback function for you.

Availability

Available in CarbonLib 1.0 and later.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`ColorPicker.h`

NewColorChangedUPP

Creates a universal procedure pointer (UPP) to a color-changed callback.

```
ColorChangedUPP NewColorChangedUPP (
   ColorChangedProcPtr userRoutine
);
```

Parameters

userRoutine

Return Value

See the description of the `ColorChangedUPP` data type.

Availability

Available in CarbonLib 1.0 and later.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`ColorPicker.h`

NewNColorChangedUPP

Creates a universal procedure pointer (UPP) to a color-changed callback.

```
NColorChangedUPP NewNColorChangedUPP (
   NColorChangedProcPtr userRoutine
);
```

Parameters

userRoutine

Return Value
See the description of the `NColorChangedUPP` data type.

Availability
Available in CarbonLib 1.0 and later.
Available in Mac OS X 10.0 and later.

Declared In
`ColorPicker.h`

NewUserEventUPP

Creates a universal procedure pointer (UPP) to an event filter callback.

```
UserEventUPP NewUserEventUPP (
   UserEventProcPtr userRoutine
);
```

Parameters

userRoutine

Return Value
See the description of the `UserEventUPP` data type.

Availability
Available in CarbonLib 1.0 and later.
Available in Mac OS X 10.0 and later.
Not available to 64-bit applications.

Declared In
`ColorPicker.h`

NPickColor

Displays the Color Picker dialog.

```
OSErr NPickColor (
   NColorPickerInfo *theColorInfo
);
```

Parameters

theColorInfo

A pointer to a color picker parameter (`NColorPickerInfo`) data structure. On input, you specify information such as the location of the dialog. Make sure that you set `theColor.profile` field in this strucure to the color space that you want the color returned in. On output, the data structure specifies information such as whether the user changed the color.

Return Value
A result code. See "Color Picker Result Codes" (page 34).

Discussion
The `NPickColor` function displays the standard dialog for color pickers. Use the color picker parameter data structure to specify information to and obtain information from the Color Picker Manager.

Availability
Available in CarbonLib 1.0 and later when ColorPicker 2.1 or later is present.
Available in Mac OS X 10.0 and later.

Carbon Porting Notes

This call is identical to the unsupported function `PickColor` , but it replaces the older ColorSync 1.0 data types with new ColorSync 2.x profile references. When filling out the parameter block for a call to the function `NPickColor`, you must replace all profile handles with profile references. The optional color-changed proc you has also changed; a new data structure `NCMColor` replaces the `CMColor` data type and uses profile references.

Related Sample Code
CarbonSketch

Declared In
`ColorPicker.h`

PickColor

Requests the user to choose a color from the standard color picker dialog box.

```
OSErr PickColor (
   ColorPickerInfo *theColorInfo
);
```

Parameters

`theColorInfo`

A pointer to the `ColorPickerInfo (Old)` (page 22) structure, to specify information to and obtain information from the Color Picker Manager.

Return Value
A result code. See "Color Picker Result Codes" (page 34).

Discussion
The `PickColor` function displays the standard modal dialog box for color pickers. When the user clicks the OK button, `PickColor` removes the dialog box and returns `true` in the `newColorChosen` field and the user's selected color in the `theColor` field of `theColorInfo`. When the user clicks the Cancel button, `PickColor` removes the dialog box and returns `false` in the `newColorChosen` field.

Availability
Available in CarbonLib 1.0 and later when ColorPicker 2.0 or later is present.
Available in Mac OS X 10.0 and later.
Not available to 64-bit applications.

Declared In
`ColorPicker.h`

RGB2CMY

Converts an RGB color to a CMY color.

```
void RGB2CMY (
   const RGBColor *rColor,
   CMYColor *cColor
);
```

Parameters

rColor

A pointer to an `RGBColor` structure to be converted.

cColor

A pointer to a `CMYColor` structure for the converted color.

Availability

Available in CarbonLib 1.0 and later when running Mac OS 8.1 or later.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`ColorPicker.h`

RGB2HSL

Converts an RGB color to an HSL color.

```
void RGB2HSL (
   const RGBColor *rColor,
   HSLColor *hColor
);
```

Parameters

rColor

A pointer to the `RGBColor` structure to be converted.

hColor

A pointer to an `HSLColor` structure for the converted color.

Availability

Available in CarbonLib 1.0 and later when running Mac OS 8.1 or later.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`ColorPicker.h`

RGB2HSV

Converts an RGB color to an HSV color.

```
void RGB2HSV (
   const RGBColor *rColor,
   HSVColor *hColor
);
```

Parameters

rColor

A pointer to the `RGBColor` structure to be converted.

hColor

A pointer to an `HSVColor` structure for the converted color.

Availability

Available in CarbonLib 1.0 and later when running Mac OS 8.1 or later.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`ColorPicker.h`

SmallFract2Fix

Converts a `SmallFract` value to a fixed integer.

```
Fixed SmallFract2Fix (
   SmallFract s
);
```

Parameters

s

The value of type `SmallFract` to be converted into a fixed integer.

Return Value

A fixed integer converted from the `SmallFract` value specified in the `s` parameter.

Availability

Available in CarbonLib 1.0 and later when running Mac OS 8.1 or later.

Available in Mac OS X 10.0 and later.

Not available to 64-bit applications.

Declared In

`ColorPicker.h`

Callbacks by Task

Changing Colors in a Document

`NColorChangedProcPtr` (page 20)

Defines a pointer to your color-changed callback function, which applies a new color to a user's document.

`ColorChangedProcPtr` (page 19)
Defines a pointer to your color-changed callback function, which applies a new color to a user document.

Handling Application-Directed Events in a Color Picker

`UserEventProcPtr` (page 20)
Defines a pointer to your event filter callback function, which determines whether your application or the Color Picker Manager handles this user event.

Callbacks

ColorChangedProcPtr

Defines a pointer to your color-changed callback function, which applies a new color to a user document.

```
typedef void (*ColorChangedProcPtr) (
    SInt32 userData,
    PMColor *newColor
);
```

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

```
void MyColorChangedProc (
    SInt32 userData,
    PMColor *newColor
);
```

Parameters

userData
Data that your application supplies in the `colorProcData` field of `ColorPickerInfo (Old)` (page 22). Your application can use this value for any purpose it needs.

newColor
A pointer to a `PMColor` (page 28) structure that contains the new color selected by the user. Your color-changed function should update the user's document to use this color.

Discussion

Your application should supply the `colorProc` field of the color picker parameter with a pointer to your color change callback function.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

`ColorPicker.h`

NColorChangedProcPtr

Defines a pointer to your color-changed callback function, which applies a new color to a user's document.

```
typedef void (*NColorChangedProcPtr)
(
    SInt32 userData,
    NPMColor *newColor
);
```

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

```
void MyNColorChangedProc (
    SInt32 userData,
    NPMColor *newColor
);
```

Parameters

newColor

Availability

Available in Mac OS X v10.0 and later.

Declared In

`ColorPicker.h`

UserEventProcPtr

Defines a pointer to your event filter callback function, which determines whether your application or the Color Picker Manager handles this user event.

```
typedef Boolean (*UserEventProcPtr) (
    EventRecord *event
);
```

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

```
Boolean MyUserEventProc (
    EventRecord *event
);
```

Parameters

event

A pointer to an event record.

Return Value

`True` if your application handles the event, `false` otherwise. The Color Picker Manager will process the event further only if `false` is returned.

Discussion

Your application should supply the `eventProc` field of the color picker parameter block with a pointer to your filter function. Your filter function should examine the event record passed in the first parameter to determine whether your application needs to handle the event contained in the record.

Applications can generally allow the Color Picker Manager to handle all events that might occur while displaying the standard dialog box. Update events are exceptions to this, however.

The `PickColor` function calls the Dialog Manager function `DialogSelect`. `DialogSelect` does not allow background windows to receive update events; therefore, at a minimum, your event filter function should handle update events. If your application needs to filter or preprocess other events before `DialogSelect` handles them, your application should do so in its event filter function.

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

Declared In
`ColorPicker.h`

Data Types

CMYColor

Contains cyan, magenta, and yellow values for a color.

```
struct CMYColor {
    SmallFract cyan;
    SmallFract magenta;
    SmallFract yellow;
};
typedef struct CMYColor CMYColor;
```

Fields

`cyan`
The `SmallFract` (page 29) value for the cyan component.

`magenta`
The `SmallFract` value for the magenta component.

`yellow`
The `SmallFract` value for the yellow component.

Discussion
The `CMYColor` structure contains cyan, magenta, and yellow colors. Your application can use a `CMYColor` structure to specify a color in a `PMColor` (page 28) structure. For example, your application supplies a `PMColor` structure in a `ColorPickerInfo (Old)` (page 22) block that it passes to the `PickColor` (page 16) function. Note that CMY and RGB colors are complementary.

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

Declared In
`ColorPicker.h`

ColorChangedUPP

Defines a universal procedure pointer to a color-changed callback.

```
typedef ColorChangedProcPtr ColorChangedUPP;
```

Discussion
For more information, see the description of the `ColorChangedProcPtr` (page 19) callback function.

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

Declared In
`ColorPicker.h`

ColorPickerInfo (Old)

Contains information needed to display a standard color picker dialog box.

```
struct ColorPickerInfo {
    PMColor theColor;
    CMProfileHandle dstProfile;
    UInt32 flags;
    DialogPlacementSpec placeWhere;
    Point dialogOrigin;
    OSType pickerType;
    UserEventUPP eventProc;
    ColorChangedUPP colorProc;
    UInt32 colorProcData;
    Str255 prompt;
    PickerMenuItemInfo mInfo;
    Boolean newColorChosen;
    SInt8 filler;
};
typedef struct ColorPickerInfo ColorPickerInfo;
```

Fields

`theColor`
: A `PMColor` (page 28) structure. The color that your application supplies in this field is passed to the color picker for editing. This becomes the original color for the color picker. After the user clicks either the OK or Cancel button to close the dialog box, this field contains the new color, that is, the color last chosen by the user. Although the new colors selected by the user may vary widely, the original color remains fixed for comparison.

`dstProfile`
: A handle to a ColorSync 1.0 profile for the final output device. To use the default system profile, set this field to `NULL`.

`flags`
: Bits representing the color picker flags, which are described in detail in "Color Picker Flags" (page 30). Your application can set any of the following flags: `CanModifyPalette`, `CanAnimatePalette`, `AppIsColorSyncAware`. The color picker may set any of the following flags and override your application settings: `InSystemDialog`, `InApplicationDialog`, `InPickerDialog`, `DetachedFromChoices`.

`placeWhere`
: A specification for where to position the dialog box. Your application uses one of the following constants (described in "Dialog Placement Constants" (page 32)) to specify the position for the color picker dialog box: `kAtSpecifiedOrigin`, `kDeepestColorScreen`, `kCenterOnMainScreen`.

`dialogOrigin`
: The point, in global coordinates, at which to locate the upper-left corner of the dialog box. This origin point is used only if your application supplies the `kAtSpecifiedOrigin` specifier in the `placeWhere` field.

`pickerType`
: The component subtype of the initial color picker. If your application sets this field to 0, the default color picker is used (that is, the last color picker chosen by the user). When `PickColor` returns, this field contains the component subtype of the color picker that was chosen when the user closed the color picker dialog box.

`eventProc`
: A pointer to an application-defined event filter function for handling user events meant for your application. If your filter function returns `true`, the Color Picker Manager will not process the event any further. If your filter function returns `false`, the Color Picker Manager handles the event as if it were meant for the color picker. The event filter function you can supply here is described in `UserEventProcPtr` (page 20).

`colorProc`
: A pointer to an application-defined function to handle color changes. This function, described in `ColorChangedProcPtr` (page 19), should support the updating of colors in a document as the user selects them.

`colorProcData`
: A long integer that the Color Picker Manager passes to the application-defined function supplied in the `colorProc` field. Your application-defined function can use this value for any purpose it needs.

`prompt`
: A text string prompting the user to choose a color for a particular use (for example, "Choose a highlight color:").

`mInfo`
: A `PickerMenuItemInfo` (page 27) structure. This structure allows your application to specify your Edit menu for use when a color picker dialog box is displayed.

`newColorChosen`
: `True` if the user chose a color and clicked the OK button, and `false` if the user clicked Cancel.

`filler`

Discussion

When your application calls the `PickColor` (page 16) function to display a standard color picker dialog box, your application uses a color picker parameter block to specify information to and obtain information from the Color Picker Manager. The color picker parameter block is defined by the `ColorPickerInfo` data type.

This version of the Color Picker Manager uses ColorSync 1.0 profiles only. The ColorSync 1.0 profile is a handle-based profile. The profile format is defined by Apple Computer. You cannot use version 2.0 profiles, which are identified by profile references, with this version of the Color Picker Manager. ColorSync 1.0 profiles typically reside in the ColorSyncTM Profiles folder (within the Preferences folder of the System Folder). They may also be embedded with the images to which they pertain in graphics files.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
`ColorPicker.h`

HSLColor

Contains hue, saturation and lightness information for a color.

```
struct HSLColor {
    SmallFract hue;
    SmallFract saturation;
    SmallFract lightness;
};
typedef struct HSLColor HSLColor;
```

Fields

`hue`
The `SmallFract` (page 29) value for the hue.

`saturation`
The `SmallFract` value for the saturation, where 1 is full color.

`lightness`
The `SmallFract` value for lightness, where 1 is full white.

Discussion
The `HSLColor` structure contains a color's hue, saturation, and lightness values. Your application can use an `HSLColor` structure to specify a color in a `PMColor` (page 28) structure. For example, your application supplies a `PMColor` structure in a `ColorPickerInfo (Old)` (page 22) block that it passes to the `PickColor` (page 16) function. Note that the standard HLS order is altered to HSL.

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

Declared In
`ColorPicker.h`

HSVColor

Contains hue, saturation and value information for a color.

```
struct HSVColor {
    SmallFract hue;
    SmallFract saturation;
    SmallFract value;
};
typedef struct HSVColor HSVColor;
```

Fields

`hue`
The `SmallFract` (page 29) value for the hue.

`saturation`
The `SmallFract` value for the saturation, where 1 is full color.

`value`

The `SmallFract` value for the color's value, where 1 is maximum intensity.

Discussion
The `HSVColor` structure contains the hue, saturation, and value of a color. Your application can use an `HSVColor` structure to specify a color in a `PMColor` (page 28) structure. For example, your application supplies a `PMColor` structure in a `ColorPickerInfo (Old)` (page 22) block that it passes to the `PickColor` (page 16) function.

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

Declared In
`ColorPicker.h`

NColorChangedUPP

Defines a universal procedure pointer to your color-changed callback.

```
typedef NColorChangedProcPtr NColorChangedUPP;
```

Discussion
For more information, see the description of the `NColorChangedProcPtr` (page 20) callback function.

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

Declared In
`ColorPicker.h`

NColorPickerInfo

Contains information needed to display a standard color picker dialog.

```
struct NColorPickerInfo {
    NPMColor theColor;
    CMProfileRef dstProfile;
    UInt32 flags;
    DialogPlacementSpec placeWhere;
    Point dialogOrigin;
    OSType pickerType;
    UserEventUPP eventProc;
    NColorChangedUPP colorProc;
    UInt32 colorProcData;
    Str255 prompt;
    PickerMenuItemInfo mInfo;
    Boolean newColorChosen;
    UInt8 reserved;
};
typedef struct NColorPickerInfo NColorPickerInfo;
```

Fields

`theColor`
: An `NPMColor` structure that contains a color profile and a color.

`dstProfile`
: The destination profile.

`flags`
: The options to apply when displaying the color picker dialog. For information on the flags that you can supply, see "Color Picker Flags" (page 30).

`dialogOrigin`
: A constant that specifies where on the screen to place the color picker dialog. The constant `kAtSpecifiedOrigin` specifies to place the top-left corner of the color picker dialog at the point specified in the `dialogOrigin` field of the color picker parameter block. The constant `kDeepestColorScreen` specifies to center the color picker dialog on the screen with the greatest color depth. The constant `kCenterOnMainScreen` specifies to center the color picker dialog on the main screen.

`pickerType`
: The type of color picker.

`eventProc`
: A a universal procedure pointer to an event-filter callback.

`colorProc`
: A universal procedure pointer to a color-changed callback.

`colorProcData`
: Data needed for your color-changed callback.

`prompt`
: A string to use as a prompt.

`mInfo`
: A structure that contains information needed for your application to specify an Edit menu for use when a color picker dialog box is displayed.

`newColorChosen`
: A Boolean value that specifies whether or not a new color was chosen.

`reserved`
: Reserved for future use.

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

Declared In
`ColorPicker.h`

NPMColor

Contains a color profile and a color.

```
struct NPMColor {
    CMProfileRef profile;
    CMColor color;
};
typedef struct NPMColor NPMColor;
```

Fields
`profile`
A color-matching profile.

`color`
A color, as specified in a color-matching structure.

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

Declared In
`ColorPicker.h`

NPMColorPtr

A pointer to an `NPMColor` data structure.

```
typedef NPMColor * NPMColorPtr;
```

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

Declared In
`ColorPicker.h`

PickerMenuItemInfo

Contains information needed for your application to specify an Edit menu for use when a color picker dialog box is displayed.

```
struct PickerMenuItemInfo {
    short editMenuID;
    short cutItem;
    short copyItem;
    short pasteItem;
    short clearItem;
    short undoItem;
};
typedef struct PickerMenuItemInfo PickerMenuItemInfo;
```

Fields

`editMenuID`
: The resource ID of the Edit menu for the color picker.

`cutItem`
: The item number of the Cut command.

`copyItem`
: The item number of the Copy command.

`pasteItem`
: The item number of the Paste command.

`clearItem`
: The item number of the Clear command.

`undoItem`
: The item number of the Undo command.

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

Declared In
`ColorPicker.h`

PMColor

Contains a color profile and a color.

```
 {
    CMProfileHandle profile;
    CMColor color;
};
typedef struct PMColor PMColor;
```

Fields

`profile`
: A handle to a color-matching profile (`CMProfile` structure). If your application sets this field to `NULL`, then the Color Picker Manager uses the default system profile.

`color`
: A color, as specified in a color-matching structure. (`CMColor` is a union data type defined in the ColorSync programming interface.

Discussion
For defining colors, version 2.0 of the Color Picker Manager uses a picker color structure. For example, when your application creates a `ColorPickerInfo` parameter block to pass to `PickColor` (page 16) , your application supplies a picker color structure. The color that your application supplies in this field is passed to the color picker for editing. After the user clicks either the OK or Cancel button to close the dialog box, this field contains the color last chosen by the user.

The picker color structure includes a ColorSync 1.0 profile, a structure that matches colors among hardware devices such as displays, printers, and scanners. This color-matching profile (a data structure of type `CMProfile`) defines the color space of the color (which includes the type of color—RGB, CMYK, HSL, and so on). Using the `dstProfile` field of `ColorPickerInfo (Old)` (page 22) or the PickerSetProfile function, your application can specify a destination color-matching profile, which describes the color space of the device for which the color is being chosen. Given information about the destination profile, color pickers that are ColorSync aware can help the user choose a color that's within the destination device's gamut.

This version of the Color Picker Manager uses ColorSync 1.0 profiles only. The ColorSync 1.0 profile is a handle-based profile. The profile format is defined by Apple Computer. You cannot use version 2.0 profiles, which are identified by profile references, with this version of the Color Picker Manager. ColorSync 1.0 profiles typically reside in the ColorSyncTM Profiles folder (within the Preferences folder of the System Folder). They may also be embedded with the images to which they pertain in graphics files.

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

Declared In
`ColorPicker.h`

PMColorPtr

A pointer to a `PMColor` data structure.

```
typedef PMColor * PMColorPtr;
```

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

Declared In
`ColorPicker.h`

SmallFract

Defines a data type for an unsigned short value.

```
typedef  unsigned short SmallFract;
```

Discussion
The `SmallFract` type is derived from the low-order word of a fixed integer. The Color Picker Manager uses `SmallFract` values to save memory and to be compatible with the Color QuickDraw `RGBColor` structure. You can use the `Fix2SmallFract` (page 10) function to convert a fixed integer to a `SmallFract` value. You can use the `SmallFract2Fix` (page 18) function to convert a `SmallFract` value to a fixed integer.

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

Declared In
`ColorPicker.h`

UserEventUPP

Defines a universal procedure pointer to an event-filter callback.

```
typedef UserEventProcPtr UserEventUPP;
```

Discussion
For more information, see the description of the `UserEventProcPtr` (page 20) callback function.

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

Declared In
`ColorPicker.h`

Constants

Color Picker Flags

Specify a variety of options to apply when displaying the color picker dialog.

```
enum {
    kColorPickerDialogIsMoveable = 1,
    kColorPickerDialogIsModal = 2,
    kColorPickerCanModifyPalette = 4,
    kColorPickerCanAnimatePalette = 8,
    kColorPickerAppIsColorSyncAware = 16,
    kColorPickerInSystemDialog = 32,
    kColorPickerInApplicationDialog = 64,
    kColorPickerInPickerDialog = 128,
    kColorPickerDetachedFromChoices = 256,
    kColorPickerCallColorProcLive = 512
};
```

Constants
`kColorPickerDialogIsMoveable`
If your application sets the bit represented by this constant when creating a custom dialog box, then the color picker dialog box is moveable by the user.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

`kColorPickerDialogIsModal`

If your application sets the bit represented by this constant when creating a custom dialog box, then the color picker dialog box is a modal dialog box.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

`kColorPickerCanModifyPalette`

Your application should set the bit represented by this constant if your application can install a palette of its own that may modify (but not animate) the current color table. If you do not want the colors in the document to change as the user makes choices in the color picker dialog box, do not set this flag.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

`kColorPickerCanAnimatePalette`

If your application sets the bit represented by this constant, then the color picker may modify or animate the palette.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

`kColorPickerAppIsColorSyncAware`

Your application should set the bit represented by this constant if your application uses ColorSync color matching. If your application sets this bit, a color may be returned to your application in a different color space than the one initially passed to the `PickColor` function. For example, your application could pass an RGB color with no color-matching profile in the field `theColor` in `ColorPickerInfo`, and the Color Picker Manager could return a CMYK color with its associated profile. If your application does not set this flag, the Color Picker Manager automatically converts any color it receives back from the color picker to an RGB color.

This version of the Color Picker Manager uses ColorSync 1.0 profiles only. The ColorSync 1.0 profile is a handle-based profile. The profile format is defined by Apple Computer. You cannot use version 2.0 profiles, which are identified by profile references, with this version of the Color Picker Manager. ColorSync 1.0 profiles typically reside in the ColorSyncTM Profiles folder (within the Preferences folder of the System Folder). They may also be embedded with the images to which they pertain in graphics files.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

`kColorPickerInSystemDialog`

The color picker sets this flag to indicate that the color picker is in a system-owned dialog box.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

`kColorPickerInApplicationDialog`

The color picker sets this flag to indicate that the color picker is in an application-owned dialog box.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

`kColorPickerInPickerDialog`

The color picker sets this flag to indicate that the color picker is in its own dialog box.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

`kColorPickerDetachedFromChoices`
The color picker sets this flag to indicate that the color picker has been detached from the choices list.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

`kColorPickerCallColorProcLive`
Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

Discussion
In the `flags` field of the `ColorPickerInfo (Old)` (page 22) parameter block your application specifies characteristics for the color picker dialog box.

The color picker may set any of the `InSystemDialog`, `InApplicationDialog`, `InPickerDialog`, or `DetachedFromChoices` flags and override your application settings.

Special Considerations

Dialog Placement Constants

Specify where on the screen to place the color picker dialog.

```
typedef SInt16 DialogPlacementSpec;
enum {
    kAtSpecifiedOrigin = 0,
    kDeepestColorScreen = 1,
    kCenterOnMainScreen = 2
};
```

Constants
`kAtSpecifiedOrigin`
Specify to place the top-left corner of the color picker dialog at the point specified in the `dialogOrigin` field of the color picker parameter block.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

`kDeepestColorScreen`
Specify to center the color picker dialog on the screen with the greatest color depth.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

`kCenterOnMainScreen`
Specify to center the color picker dialog on the main screen.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

Discussion
In the `placeWhere` field of the `ColorPickerInfo (Old)` (page 22) , your application specifies where to place the color picker dialog box. Your application uses the `DialogPlacementSpec` enumeration to specify the position of the color picker dialog box.

Maximum Small Fraction

Defines the maximum value for the `SmallFract` data type.

```
enum {
    kMaximumSmallFract = 0x0000FFFF
};
```

Width and Height Constants

Specify the width and height of the color picker dialog.

```
enum {
    kDefaultColorPickerWidth = 383,
    kDefaultColorPickerHeight = 238
};
```

Constants

`kDefaultColorPickerWidth`

Specifies the width of the color picker dialog.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

`kDefaultColorPickerHeight`

Specifies the height of the color picker dialog.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

Old Maximum Small Fraction

Defines the maximum value for the `SmallFract` data type.

```
enum {
    MaxSmallFract = 0x0000FFFF
};
```

Constants

`MaxSmallFract`

The maximum value of the `SmallFract` data type, as a long integer.

Available in Mac OS X v10.0 and later.

Declared in `ColorPicker.h`.

Old Color Picker Flags

Specify a variety of options to apply when displaying the color picker dialog.

```
enum {
    DialogIsMoveable = 1,
    DialogIsModal = 2,
    CanModifyPalette = 4,
    CanAnimatePalette = 8,
    AppIsColorSyncAware = 16,
    InSystemDialog = 32,
    InApplicationDialog = 64,
    InPickerDialog = 128,
    DetachedFromChoices = 256,
    CallColorProcLive = 512
};
```

Result Codes

The most common result codes returned by Color Picker are listed below.

Result Code	Value	Description
`firstPickerError`	-4000	Available in Mac OS X v10.0 and later.
`invalidPickerType`	-4000	Available in Mac OS X v10.0 and later.
`requiredFlagsDontMatch`	-4001	Available in Mac OS X v10.0 and later.
`pickerResourceError`	-4002	Available in Mac OS X v10.0 and later.
`cantLoadPicker`	-4003	Available in Mac OS X v10.0 and later.
`cantCreatePickerWindow`	-4004	Available in Mac OS X v10.0 and later.
`cantLoadPackage`	-4005	Available in Mac OS X v10.0 and later.
`pickerCantLive`	-4006	Available in Mac OS X v10.0 and later.
`colorSyncNotInstalled`	-4007	Available in Mac OS X v10.0 and later.
`badProfileError`	-4008	Available in Mac OS X v10.0 and later.
`noHelpForItem`	-4009	Available in Mac OS X v10.0 and later.

Unsupported Functions

This section lists functions that are unsupported in Mac OS X. "Unsupported Functions" provides information on what you should do in place of using these functions.

Table A-1 Porting notes for unsupported Color Picker functions

Unsupported functions	Porting notes
PickerDisplay	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
PickerEdit	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
PickerEvent	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
PickerExtractHelpItem	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
PickerGetDialog	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
PickerGetEditMenuState	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
PickerGetIconData	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
PickerGetItemList	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
PickerGetPrompt	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
PickerInit	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
PickerItemHit	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
PickerSetBaseItem	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
PickerSetColorChangedProc	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.

Unsupported functions	Porting notes
`PickerSetOrigin`	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
`PickerSetPrompt`	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
`PickerSetVisibility`	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.
`PickerTestGraphics-World`	This low-level function was rarely used. Use high-level functions such as `GetColor`, `PickColor`, and `NPickColor` instead.

Document Revision History

This table describes the changes to *Color Picker Manager Reference*.

Date	Notes
2007-07-02	Corrected serveral technical and typographical errors.
	Added information to `PMColor` concerning the `CMColor` data structure.
	Added `NPMColorPtr` and `PMColorPtr`.
	Corrected typographical errors.
2005-07-07	Added documentation for the fields of the NColorPickerInfo data structure.
2003-04-01	Added abstracts to many functions.
2003-02-01	Updated formatting and linking. Moved unsupported functions to Appendix A.

Document Revision History

Index

N

O

P

R

S

U

W