Picture Utilities 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

Picture Utilities Reference (Not Recommended) 5

Overview 5
Functions by Task 5
- Collecting Picture Information 5
- Using Universal Procedure Pointers 6

Callbacks 7
- CalcColorTableProcPtr 7
- DisposeColorPickMethodProcPtr 9
- InitPickMethodProcPtr 9
- RecordColorsProcPtr 11

Data Types 12
- CalcColorTableUPP 12
- CommentSpec 12
- DisposeColorPickMethodUPP 13
- FontSpec 13
- InitPickMethodUPP 14
- PictInfo 15
- PictInfoID 18
- RecordColorsUPP 19

Constants 19
- Color Bank Type 19
- Color Selection Method 20
- Color Information Type 20

Result Codes 21

Appendix A **Deprecated Picture Utilities Reference (Not Recommended) Functions 23**

Deprecated in Mac OS X v10.4 23
- DisposeCalcColorTableUPP 23
- DisposeDisposeColorPickMethodUPP 23
- DisposeInitPickMethodUPP 24
- DisposePictInfo 25
- DisposeRecordColorsUPP 25
- GetPictInfo 26
- GetPixMapInfo 28
- InvokeCalcColorTableUPP 30
- InvokeDisposeColorPickMethodUPP 30
- InvokeInitPickMethodUPP 31
- InvokeRecordColorsUPP 31
- NewCalcColorTableUPP 32
- NewDisposeColorPickMethodUPP 32

NewInitPickMethodUPP 33
NewPictInfo 34
NewRecordColorsUPP 35
RecordPictInfo 36
RecordPixMapInfo 36
RetrievePictInfo 37

Document Revision History 39

Index 41

Picture Utilities Reference (Not Recommended)

Framework: ApplicationServices/ApplicationServices.h

Declared in PictUtils.h

Overview

Important: The Picture Utilities are deprecated as of Mac OS X v10.4. The replacement API for all QuickDraw technologies is Quartz 2D (Core Graphics). See *Quartz Programming Guide for QuickDraw Developers* for strategies to replace QuickDraw code with Quartz 2D..

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

QuickDraw pictures are sequences of saved drawing commands. Pictures provide a common medium for the sharing of image data.

The Picture Utilities allow your application to gather information about a picture, such as color, fonts, picture comments, and resolution. You can also use the Picture Utilities to gather information about the colors in pixel maps.

Functions by Task

Collecting Picture Information

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

Disposes of the private data structures allocated by the `NewPictInfo` function. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

`GetPictInfo` (page 26) Deprecated in Mac OS X v10.4

Gathers information about a single picture. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

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

Gathers color information about a single pixel map or bitmap. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

`NewPictInfo` (page 34) Deprecated in Mac OS X v10.4

Begins collecting pictures, pixel maps, and bitmaps for a survey of pictures. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

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

Adds a picture to an informational survey of multiple pictures. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

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

Adds a pixel map or bitmap to an informational survey of multiple pixel maps and bitmaps. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

`RetrievePictInfo` (page 37) Deprecated in Mac OS X v10.4

Returns information about all the pictures, pixel maps, and bitmaps included in a survey. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

Using Universal Procedure Pointers

`DisposeCalcColorTableUPP` (page 23) Deprecated in Mac OS X v10.4

Disposes of a universal procedure pointer (UPP) to a color table calculation callback. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

`DisposeDisposeColorPickMethodUPP` (page 23) Deprecated in Mac OS X v10.4

Disposes of a universal procedure pointer (UPP) to a method disposal callback. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

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

Disposes of a universal procedure pointer (UPP) to a method initialization callback. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

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

Disposes of a universal procedure pointer (UPP) to a color recording callback. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

`InvokeCalcColorTableUPP` (page 30) Deprecated in Mac OS X v10.4

Invokes a color table calculation callback, using a universal procedure pointer. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

`InvokeDisposeColorPickMethodUPP` (page 30) Deprecated in Mac OS X v10.4

Invokes a method disposal callback, using a universal procedure pointer. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

`InvokeInitPickMethodUPP` (page 31) Deprecated in Mac OS X v10.4

Invokes a method initialization callback, using a universal procedure pointer. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

`InvokeRecordColorsUPP` (page 31) Deprecated in Mac OS X v10.4

Invokes a color recording callback, using a universal procedure pointer. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

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

Creates a new universal procedure pointer (UPP) to a color table calculation callback. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

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

Creates a new universal procedure pointer (UPP) to a method disposal callback. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

`NewInitPickMethodUPP` (page 33) Deprecated in Mac OS X v10.4

Creates a new universal procedure pointer (UPP) to a method initialization callback. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

`NewRecordColorsUPP` (page 35) Deprecated in Mac OS X v10.4

Creates a new universal procedure pointer (UPP) to a color recording callback. (Deprecated. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

Callbacks

CalcColorTableProcPtr

Defines a pointer to a color table calculation callback. Your color calculation callback selects as many colors as are requested by your application from the color bank for a picture or pixel map and then fills these colors into an array of `ColorSpec` structures.

```
typedef OSErr (*CalcColorTableProcPtr)
(
    UInt32 dataRef,
    SInt16 colorsRequested,
    void * colorBankPtr,
    CSpecArray resultPtr
);
```

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

```
OSErr CalcColorTableProcPtr (
    UInt32 dataRef,
    SInt16 colorsRequested,
    void * colorBankPtr,
    CSpecArray resultPtr
);
```

Parameters

`dataRef`

A handle to any data your method needs. Your application initially creates this handle using the `InitPickMethodProcPtr` (page 9) function.

`colorsRequested`

The number of colors requested by your application to be gathered for examination in a `ColorTable` or `Palette` structure.

colorBankPtr

If your MyInitPickMethodCallback function returned either the colorBankIsExactAnd555 or colorBankIs555 constant, then this parameter contains a pointer to the 5-5-5 histogram that describes all of the colors in the picture, pixel map, or bitmap being examined. (The format of the 5-5-5 histogram is explained in the function description for the InitPickMethodProcPtr (page 9) function.) Your MyCalcColorTableCallback function should examine these colors and then, using its own criterion for selecting the colors, fill in an array of ColorSpec structures with the number of colors specified in the colorsRequested parameter.

If your MyInitPickMethodCallback function returned the colorBankIsCustom constant, then the value passed in this parameter is invalid. In this case, your MyCalcColorTableCallback function should use the custom color bank that your application created (using the RecordColorsProcPtr (page 11) function) for filling in an array of ColorSpec structures with the number of colors specified in the colorsRequested parameter.

Your MyCalcColorTableCallback function should return a pointer to this array of ColorSpec structures in the next parameter.

resultPtr

A pointer to the array of ColorSpec structures to be filled with the number of colors specified in the colorsRequested parameter. The Picture Utilities function that your application initially called places these colors in a Palette structure or ColorTable structure, as specified by your application.

Return Value

A result code. See "Picture Utilities Result Codes" (page 21). If MyCalcColorTableCallback generates an error, it should return the error as its function result. This error is passed back to the GetPictInfo, GetPixMapInfo, or NewPictInfo function, which in turn passes the error to your application as a function result.

Discussion

Selecting from the color bank created for the picture, bitmap, or pixel map being examined, MyCalcColorTableCallback fills an array of ColorSpec structures with the number of colors requested in the colorsRequested parameter and returns this array in the resultPtr parameter.

If more colors are requested than the picture contains, MyCalcColorTable fills the remaining entries with black (0000 0000 0000).

The colorBankPtr parameter is of type Ptr because the data stored in the color bank is of the type specified by your InitPickMethodProcPtr (page 9) function. Thus, if you specified colorBankIs555 in the colorBankType parameter, the color bank would be an array of integers. However, if the Picture Utilities support other data types in the future, the colorBankPtr parameter could point to completely different data types.

Always coerce the value passed in the colorBankPtr parameter to a pointer to an integer. In the future you may need to coerce this value to a pointer of the type you specify in your MyInitPickMethodCallback function.

Availability

Available in Mac OS X v10.0 and later.

Declared In

PictUtils.h

DisposeColorPickMethodProcPtr

Defines a pointer to a method disposal callback function. Your method disposal function releases the memory for the `'cpmt'` resource allocated by your `MyInitPickMethodCallback` function.

```
typedef OSErr (*DisposeColorPickMethodProcPtr)
(
    UInt32 dataRef
);
```

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

```
OSErr DisposeColorPickMethodProcPtr (
    UInt32 dataRef
);
```

Parameters

dataRef

A handle to any data your method needs. Your application initially creates this handle using the `InitPickMethodProcPtr` (page 9) function.

Return Value

A result code. See "Picture Utilities Result Codes" (page 21). If your `MyDisposeColorPickMethodCallback` function generates an error, it should return the error as its function result. This error is passed back to the `GetPictInfo`, `GetPixMapInfo`, or `NewPictInfo` function, which in turn passes the error to your application as a function result.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`PictUtils.h`

InitPickMethodProcPtr

Defines a pointer to a method initialization callback function. Your method initialization function specifies the color back and allocates whatever data your color-picking method needs.

```
typedef OSErr (*InitPickMethodProcPtr)
(
    SInt16 colorsRequested,
    UInt32 * dataRef,
    SInt16 * colorBankType
);
```

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

```
OSErr InitPickMethodProcPtr (
    SInt16 colorsRequested,
    UInt32 * dataRef,
    SInt16 * colorBankType
);
```

Parameters

colorsRequested

The number of colors requested by your application to be gathered for examination in a `ColorTable` or `Palette` structure.

dataRef

A handle to any data needed by your color-picking method; that is, if your application allocates and uses additional data, it should return a handle to it in this parameter.

colorBankType

The type of color bank your color-picking method uses. Your `MyInitPickMethodCallback` function should return one of three valid color bank types.

Return the `colorBankIs555` constant in this parameter if you want to let the Picture Utilities gather the colors for a picture or a pixel map into a 5-5-5 histogram. When you return the `colorBankIs555` constant, the Picture Utilities call your `MyCalcColorTableCallback` function with a pointer to the color bank (that is, to the 5-5-5 histogram). Your `MyCalcColorTableCallback` function selects whatever colors it needs from this color bank. Then the Picture Utilities function called by your application returns these colors in a `Palette` structure, a `ColorTable` structure, or both, as requested by your application.

Return the `ColorBankIsExactAnd555` constant in this parameter to make the Picture Utilities return exact colors if there are less than 256 unique colors in the picture; otherwise, the Picture Utilities gather the colors for the picture in a 5-5-5 histogram, just as they do when you return the `colorBankIs555` constant. If the picture or pixel map has fewer colors than your application requests when it calls a Picture Utilities function, the Picture Utilities function returns all of the colors contained in the color bank. If the picture or pixel map contains more colors than your application requests, the Picture Utilities call your `MyCalcColorTableCallback` function to select which colors to return.

Return the `colorBankIsCustom` constant in this parameter if you want to implement your own color bank for storing the colors in a picture or a pixel map. For example, because the 5-5-5 histogram that the Picture Utilities provide gathers colors to a resolution of 5 bits per color, your application may want to create a histogram with a resolution of 8 bits per color. When you return the `colorBankIsCustom` constant, the Picture Utilities call your `MyRecordColorsCallback` function to create this color bank. The Picture Utilities also call your `MyCalcColorTableCallback` function to select colors from this color bank.

Return Value

A result code. See "Picture Utilities Result Codes" (page 21). If `MyInitPickMethodCallback` generates any error, it should return the error as its function result. This error is passed back to the `GetPictInfo`, `GetPixMapInfo`, or `NewPictInfo` function, which in turn passes the error to your application as a function result.

Discussion

Your color-picking method (`'cpmt'`) resource should include a function that specifies its color bank (that is, the structure into which all the colors of a picture, pixel map, or bitmap are gathered) and allocates whatever data your color-picking method needs. Your `MyInitPickMethodCallback` can let the Picture Utilities generate a color bank consisting of a histogram (that is, frequency counts of each color) to a resolution of 5 bits per color. Or, your `MyInitPickMethodCallback` function can specify that your application has its own custom color bank—for example, a histogram to a resolution of 8 bits per color.

The 5-5-5 histogram that the Picture Utilities provide if you return the `ColorBankIs555` or `ColorBankIsExactAnd555` constant in the `colorBankType` parameter is like a reversed `cSpecArray` structure, which is an array of `ColorSpec` structures. This 5-5-5 histogram is an array of 32,768 integers, where the index into the array is the color: 5 bits of red, followed by 5 bits of green, followed by 5 bits of blue. Each entry in the array is the number of colors in the picture that are approximated by the index color for that entry.

For example, suppose there were three instances of the following color in the pixel map:

- Red = %1101 1010 1010 1110
- Green = %0111 1010 1011 0001
- Blue = %0101 1011 0110 1010

This color would be represented by index % 0 11011-01111-01011 (in hexadecimal, $6DEB), and the value in the histogram at this index would be 3, because there are three instances of this color.

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

Declared In
`PictUtils.h`

RecordColorsProcPtr

Defines a pointer to a color recording callback function. Your color recording function creates a color bank.

```
typedef OSErr (*RecordColorsProcPtr)
(
    UInt32 dataRef,
    RGBColor * colorsArray,
    SInt32 colorCount,
    SInt32 * uniqueColors
);
```

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

```
OSErr RecordColorsProcPtr (
    UInt32 dataRef,
    RGBColor * colorsArray,
    SInt32 colorCount,
    SInt32 * uniqueColors
);
```

Parameters

`dataRef`
A handle to any data your function needs. Your application initially creates this handle using the `InitPickMethodProcPtr` (page 9) function.

`colorsArray`
An array of `RGBColor` structures. Your `MyRecordColorsCallback` function stores the color information for this array of `RGBColor` structures in a data structure of type `RGBColorArray`.

`colorCount`
The number of colors in the array specified in the `colorsArray` parameter.

uniqueColors

Upon input, the number of unique colors already added to the array in the `colorsArray` parameter. (The Picture Utilities functions call your `MyRecordColors` function once for every color in the picture, pixel map, or bitmap.) Your `MyRecordColorsCallback` function must calculate the number of unique colors (to the resolution of the color bank) that are added by this call. Your `MyRecordColorsCallback` function should add this amount to the value passed upon input in this parameter and then return the sum in this parameter.

Return Value
A result code. See "Picture Utilities Result Codes" (page 21). If your `MyRecordColorsCallback` function generates any error, it should return the error as its function result. This error is passed back to the `GetPictInfo`, `GetPixMapInfo`, or `NewPictInfo` function, which in turn passes the error to your application as a function result.

Discussion
`MyRecordColorsCallback` stores each color encountered in a picture or pixel into its own color bank. The Picture Utilities call `MyRecordColorsCallback` only if your `MyInitPickMethodCallback` function returns the constant `colorBankIsCustom` in the `colorBankType` parameter. When you return the `colorBankIsCustom` constant in the `colorBankType` parameter to your `MyInitPickMethodCallback` function, your color-picking method (`'cpmt'`) resource must include a function that creates this color bank; for example, your application may want to create a histogram with a resolution of 8 bits per color.

The Picture Utilities functions call `MyRecordColorsCallback` for all the colors in the picture, pixel map, or bitmap.

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

Declared In
`PictUtils.h`

Data Types

CalcColorTableUPP

Defines a universal procedure pointer (UPP) to a color table calculation callback.

```
typedef CalcColorTableProcPtr CalcColorTableUPP;
```

Discussion
For more information, see the description of the callback function `CalcColorTableProcPtr` (page 7).

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

Declared In
`PictUtils.h`

CommentSpec

Contains information about the comments in a picture.

```
struct CommentSpec {
    short count;
    short ID;
};
typedef struct CommentSpec CommentSpec;
typedef CommentSpec * CommentSpecPtr;
typedef CommentSpecPtr * CommentSpecHandle;
```

Fields

`count`

The number of times this kind of picture comment occurs in the picture specified to the `GetPictInfo` function or in all the pictures examined with the `NewPictInfo` function.

`ID`

The value set in the `kind` parameter when the picture comment was created using the function `PicComment`. For a description of this function, see *Inside Mac OS X: Quickdraw Reference*.

Discussion

If you specify the `structureComments` constant in the `verb` parameter to the `GetPictInfo` (page 26) function or the `NewPictInfo` (page 34) function, you receive a `PictInfo` (page 15) structure that includes in its `commentHandle` field a handle to an array of `CommentSpec` structures. The `uniqueComments` field of the `PictInfo` structure indicates the number of `CommentSpec` structures in this array.

When you are finished using the information returned in a `CommentSpec` structure, use the `DisposeHandle` function to dispose of the memory allocated to it.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`PictUtils.h`

DisposeColorPickMethodUPP

Defines a universal procedure pointer (UPP) to a method disposal callback.

```
typedef DisposeColorPickMethodProcPtr DisposeColorPickMethodUPP;
```

Discussion

For more information, see the description of the callback function `DisposeColorPickMethodProcPtr` (page 9).

Availability

Available in Mac OS X v10.0 and later.

Declared In

`PictUtils.h`

FontSpec

Contains information about the fonts in a picture.

```
struct FontSpec {
    short pictFontID;
    short sysFontID;
    long size[4];
    short style;
    long nameOffset;
};
typedef struct FontSpec FontSpec;
typedef FontSpec * FontSpecPtr;
typedef FontSpecPtr * FontSpecHandle;
```

Fields

`pictFontID`

The ID number of the font as it is stored in the picture.

`sysFontID`

The number that identifies the resource file (of type `'FOND'`) that specifies the font family. Every font family, has a unique font family ID, in a range of values that determines the script system to which the font family belongs.

`size`

The point sizes of the fonts in the picture. The field contains 128 bits, in which a bit is set for each point size encountered, from 1 to 127 points. Bit 0 is set if a size larger than 127 is found.

`style`

The styles for this font family at any of its sizes. The values in this field can also be represented with the `Style` data type.

`nameOffset`

The offset into the list of font names (indicated by the `fontNamesHandle` field of the `PictInfo` structure) at which the name for this font family is stored. A font name is given to a font family to distinguish it from other font families.

Discussion

If you specify the `recordFontInfo` constant in the `verb` parameter to the `GetPictInfo` function or the `NewPictInfo` function, your application receives a `PictInfo` structure that includes in its `fontHandle` field a handle to an array of `FontSpec` structures. The `uniqueFonts` field of the `PictInfo` structure indicates the number of `FontSpec` structures in this array. (For bitmap fonts, a font is a complete set of glyphs in one size, typeface, and style. For outline fonts, a font is a complete set of glyphs in one typeface and style.)

When you are finished using the information returned in a `FontSpec` structure, you should use the Memory Manager function `DisposeHandle` to dispose of the memory allocated to it.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`PictUtils.h`

InitPickMethodUPP

Defines a universal procedure pointer (UPP) to a method initialization callback.

```
typedef InitPickMethodProcPtr InitPickMethodUPP;
```

Discussion

For more information, see the description of the callback function `InitPickMethodProcPtr` (page 9).

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

Declared In
`PictUtils.h`

PictInfo

Contains information about a picture.

```
struct PictInfo {
    short version;
    long uniqueColors;
    PaletteHandle thePalette;
    CTabHandle theColorTable;
    Fixed hRes;
    Fixed vRes;
    short depth;
    Rect sourceRect;
    long textCount;
    long lineCount;
    long rectCount;
    long rRectCount;
    long ovalCount;
    long arcCount;
    long polyCount;
    long regionCount;
    long bitMapCount;
    long pixMapCount;
    long commentCount;
    long uniqueComments;
    CommentSpecHandle commentHandle;
    long uniqueFonts;
    FontSpecHandle fontHandle;
    Handle fontNamesHandle;
    long reserved1;
    long reserved2;
};
typedef struct PictInfo PictInfo;
typedef PictInfo * PictInfoPtr;
```

Fields

`version`

The version number of the Picture Utilities, currently set to 0.

`uniqueColors`

The number of colors in the picture specified to the `GetPictInfo` function, or the number of colors in the pixel map or bitmap specified to the `GetPixMapInfo` function, or the total number of colors for all the pictures, pixel maps, and bitmaps returned by the `RetrievePictInfo` function. The number of colors returned in this field is limited by the accuracy of the Picture Utilities' color bank for color storage. See `InitPickMethodProcPtr` (page 9), `RecordColorsProcPtr` (page 11), `CalcColorTableProcPtr` (page 7), and `DisposeColorPickMethodProcPtr` (page 9) for information about the Picture Utility's color bank and about how you can create your own for selecting colors.

`thePalette`

A handle to the resulting `Palette` structure if you specified to the `GetPictInfo`, `GetPixMapInfo`, or `NewPictInfo` function that colors be returned in a `Palette` structure. That `Palette` structure contains either the number of colors you specified to the function or—if there are not that many colors in the pictures, pixel maps, or bitmaps—the number of colors found. Depending on the constant you pass in the `verb` parameter to the function, the `Palette` structure contains either the most used or the widest range of colors in the pictures, pixel maps, and bitmaps. On Macintosh computers running basic QuickDraw only, this field is always returned as `NULL`.

`theColorTable`

A handle to the resulting `ColorTable` structure if you specified to the `GetPictInfo`, `GetPixMapInfo`, or `NewPictInfo` function that colors be returned in a `ColorTable` structure. If the pictures, pixel maps, or bitmaps contain fewer colors found than you specified to the function, the unused entries in the `ColorTable` structure are filled with black. Depending on the constant you pass in the `verb` parameter to the function, the `ColorTable` structure contains either the most used or the widest range of colors in the pictures, pixel maps, and bitmaps. On Macintosh computers running basic QuickDraw only, this field is always returned as `NULL`.

If a picture has more than 256 colors or has pixel depths of 32 bits, then Color QuickDraw translates the colors in the `ColorTable` structure to 16-bit depths. In such a case, the returned colors might have a slight loss of resolution, and the `uniqueColors` field reflects the number of colors distinguishable at that pixel depth.

`hRes`

The horizontal resolution of the current picture, pixel map, or bitmap retrieved by the `GetPictInfo` or `GetPixMapInfo` function or the greatest horizontal resolution from all pictures, pixel maps, and bitmaps retrieved by the `RetrievePictInfo` function.

`vRes`

The vertical resolution of the current picture, pixel map, or bitmap retrieved by the `GetPictInfo` or `GetPixMapInfo` function or the greatest vertical resolution of all pictures, pixel maps, and bitmaps retrieved by the `RetrievePictInfo` function. Although the values of the `hRes` and `vRes` fields are usually the same, they do not have to be.

`depth`

The pixel depth of the picture specified to the `GetPictInfo` function or the pixel map specified to the `GetPixMapInfo` function. When you use the `RetrievePictInfo` function, this field contains the deepest pixel depth of all pictures or pixel maps retrieved by the function.

`sourceRect`

The optimal bounding rectangle for displaying the picture at the resolution indicated by the `hRes` and `vRes` fields. The upper-left corner of the rectangle is always (0,0). Pictures created with the `OpenCPicture` function have the `hRes`, `vRes`, and `sourceRect` fields built into their `Picture` structures. For pictures created by `OpenPicture`, the `hRes` and `vRes` fields are set to 72 dpi, and the source rectangle is calculated using the `picFrame` field of the `Picture` structure for the picture.

`textCount`

The number of text strings in the picture specified to the `GetPictInfo` function, or the total number of text objects in all the pictures retrieved by the `RetrievePictInfo` function. For pixel maps and bitmaps specified to `GetPixMapInfo` or `RetrievePictInfo`, this field is set to 0.

`lineCount`

The number of lines in the picture specified to the `GetPictInfo` function, or the total number of lines in all the pictures retrieved by the `RetrievePictInfo` function. For pixel maps and bitmaps, this field is set to 0.

`rectCount`
: The number of rectangles in the picture specified to the `GetPictInfo` function, or the total number of rectangles in all the pictures retrieved by the `RetrievePictInfo` function. For pixel maps and bitmaps, this field is set to 0.

`rRectCount`
: The number of rounded rectangles in the picture specified to the `GetPictInfo` function, or the total number of rounded rectangles in all the pictures retrieved by the `RetrievePictInfo` function. For pixel maps and bitmaps, this field is set to 0.

`ovalCount`
: The number of ovals in the picture specified to the `GetPictInfo` function, or the total number of ovals in all the pictures retrieved by the `RetrievePictInfo` function. For pixel maps and bitmaps, this field is set to 0.

`arcCount`
: The number of arcs and wedges in the picture specified to the `GetPictInfo` function, or the total number of arcs and wedges in all the pictures retrieved by the `RetrievePictInfo` function. For pixel maps and bitmaps, this field is set to 0.

`polyCount`
: The number of polygons in the picture specified to the `GetPictInfo` function, or the total number of polygons in all the pictures retrieved by the `RetrievePictInfo` function. For pixel maps and bitmaps, this field is set to 0.

`regionCount`
: The number of regions in the picture specified to the `GetPictInfo` function, or the total number of regions in all the pictures retrieved by the `RetrievePictInfo` function. For pixel maps and bitmaps, this field is set to 0.

`bitMapCount`
: The total number of bitmaps in the survey.

`pixMapCount`
: The total number of pixel maps in the survey.

`commentCount`
: The number of comments in the picture specified to the `GetPictInfo` function, or the total number of comments in all the pictures retrieved by the `RetrievePictInfo` function. This field is valid only if you specified to the `GetPictInfo` or `NewPictInfo` function that comments be returned in a `CommentSpec` structure. For pixel maps and bitmaps, this field is set to 0.

`uniqueComments`
: The number of picture comments that have different IDs in the picture specified to the `GetPictInfo` function, or the total number of picture comments with different IDs in all the pictures retrieved by the `RetrievePictInfo` function. This field is valid only if you specify that comments be returned in a `CommentSpec` (page 12) structure. For pixel maps and bitmaps, this field is set to 0.

`commentHandle`
: A handle to an array of `CommentSpec` structures. For pixel maps and bitmaps, this field is set to `NULL`. See `CommentSpec` (page 12).

`uniqueFonts`

The number of different fonts in the picture specified to the `GetPictInfo` function, or the total number of different fonts in all the pictures retrieved by the `RetrievePictInfo` function. For bitmap fonts, a font is a complete set of glyphs in one size, typeface, and style. For outline fonts, a font is a complete set of glyphs in one typeface and style—for example, 12-point Geneva italic. For outline fonts, a font is a complete set of glyphs in one typeface and style—for example, Geneva italic.

This field is valid only if you specify that fonts be returned in a `FontSpec` (page 13) structure. For pixel maps and bitmaps, this field is set to 0.

`fontHandle`

A handle to a list of `FontSpec` structures. For pixel maps and bitmaps, this field is set to `NULL`.

`fontNamesHandle`

A handle to the names of the fonts in the picture retrieved by the `GetPictInfo` function or the pictures retrieved by the `RetrievePictInfo` function. The offset to a particular name is stored in the `nameOffset` field of the `FontSpec` structure for that font. A font name is a name given to one font family to distinguish it from other font families.

`reserved1`

`reserved2`

Discussion

When you use the `GetPictInfo` (page 26) function to collect information about a picture, or when you use the `GetPixMapInfo` (page 28) function to collect color information about a pixel map or bitmap, the function returns the information in a `PictInfo` structure. When you gather this information for multiple pictures, pixel maps, or bitmaps, the `RetrievePictInfo` (page 37) function also returns a `PictInfo` structure containing this information.

Initially, all of the fields in a new `PictInfo` structure are set to `NULL`. Relevant fields are set to appropriate values depending on the information you request using the Picture Utilities functions.

When you are finished with this information, be sure to dispose of it. You can dispose of `Palette` structures by using the Palette Manager function, `DisposePalette`. Dispose of `ColorTable` structures by using the QuickDraw function, `DisposeCTable`. Dispose of other allocations with the Memory Manager function, `DisposeHandle`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`PictUtils.h`

PictInfoID

Defines an identifier for a collection of pictures, pixel maps, or bitmaps in an application.

```
typedef long PictInfoID;
```

Discussion

Picture Utilities returns a `PictInfoID` value when you call the function `NewPictInfo` (page 34). It serves as a unique identifier for a collection of pictures, pixel maps, or bitmaps defined in your application. You use this ID when calling other Picture Utilities functions to manage and survey your collection.

Availability

Available in Mac OS X v10.0 and later.

Declared In
PictUtils.h

RecordColorsUPP

Defines a universal procedure pointer (UPP) to a color recording callback.

```
typedef RecordColorsProcPtr RecordColorsUPP;
```

Discussion
For more information, see the description of the callback function RecordColorsProcPtr (page 11).

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

Declared In
PictUtils.h

Constants

Color Bank Type

Specifies the type of color bank used in a color-picking method.

```
enum {
    ColorBankIsCustom = -1,
    ColorBankIsExactAnd555 = 0,
    ColorBankIs555 = 1
};
```

Constants

ColorBankIsCustom
: Gathers colors into a custom color bank. Picture Utilities gathers the colors for a picture or a pixel map into a 5-5-5 histogram. When you return the colorBankIs555 constant, the Picture Utilities call your RecordColorsProcPtr (page 11) function with a pointer to the color bank (that is, to the 5-5-5 histogram). Your CalcColorTableProcPtr function selects whatever colors it needs from this color bank. Then the Picture Utilities function called by your application returns these colors in a Palette structure, a ColorTable structure, or both, as requested by your application.

Available in Mac OS X v10.0 and later.

Declared in PictUtils.h.

ColorBankIsExactAnd555
: Gathers exact colors if there are less than 256 unique colors in picture; otherwise gathers colors for picture in a 5-5-5 histogram. If the picture or pixel map has fewer colors than your application requests when it calls a Picture Utilities function, the Picture Utilities function returns all of the colors contained in the color bank. If the picture or pixel map contains more colors than your application requests, the Picture Utilities call your CalcColorTableProcPtr function to select which colors to return.

Available in Mac OS X v10.0 and later.

Declared in PictUtils.h.

`ColorBankIs555`
Gathers colors into a 5-5-5 histogram. Specify `colorBankIsCustom` constant if you want to implement your own color bank for storing the colors in a picture or a pixel map. For example, because the 5-5-5 histogram that the Picture Utilities provide gathers colors to a resolution of 5 bits per color, your application may want to create a histogram with a resolution of 8 bits per color. When you return the `colorBankIsCustom` constant, the Picture Utilities call your `RecordColorsProcPtr` (page 11) function to create this color bank. The Picture Utilities also call your `CalcColorTableProcPtr` function to select colors from this color bank.

Available in Mac OS X v10.0 and later.

Declared in `PictUtils.h`.

Discussion
Your `InitPickMethodProcPtr` (page 9) function returns these constants in the `colorBankType` parameter to indicate the type of color bank used in your color-picking method.

Color Selection Method

Indicates the color selection method used in a `PictInfo` record.

```
enum {
    systemMethod = 0,
    popularMethod = 1,
    medianMethod = 2
};
```

Constants
`systemMethod`
Lets Picture Utilities choose the method. Currently they always choose `popularMethod`.

Available in Mac OS X v10.0 and later.

Declared in `PictUtils.h`.

`popularMethod`
Returns the most frequently used colors.

Available in Mac OS X v10.0 and later.

Declared in `PictUtils.h`.

`medianMethod`
Returns a weighted distribution of colors.

Available in Mac OS X v10.0 and later.

Declared in `PictUtils.h`.

Discussion
These constants are used to indicate the method by which colors are selected for the `ColorTable` or `Palette` structure returned via the `PictInfo` structure, by the functions `NewPictInfo` (page 34) , `GetPixMapInfo` (page 28) , or `GetPictInfo` (page 26).

Color Information Type

Indicates the type of color information returned in a `PictInfo` record.

```
enum {
    returnColorTable = 0x0001,
    returnPalette = 0x0002,
    recordComments = 0x0004,
    recordFontInfo = 0x0008,
    suppressBlackAndWhite = 0x0010
};
```

Constants

`returnColorTable`
Specify to return a Color Table.
Available in Mac OS X v10.0 and later.
Declared in `PictUtils.h`.

`returnPalette`
Specify to return a Palette structure.
Available in Mac OS X v10.0 and later.
Declared in `PictUtils.h`.

`recordComments`
Specify to return comment information.
Available in Mac OS X v10.0 and later.
Declared in `PictUtils.h`.

`recordFontInfo`
Specify to return font information.
Available in Mac OS X v10.0 and later.
Declared in `PictUtils.h`.

`suppressBlackAndWhite`
Don't include black and white with returned colors.
Available in Mac OS X v10.0 and later.
Declared in `PictUtils.h`.

Discussion

These constants are used in the `verb` parameter of the `GetPictInfo` (page 26) , `GetPixMapInfo` (page 28) ,and `NewPictInfo` (page 34) functions to indicate the type of information those functions should return. You can use any or all of these constants or the sum of the integers they represent.

Result Codes

The table below lists the most common result codes returned by Picture Utilities.

Result Code	Value	Description
`pictInfoVersionErr`	-11000	Wrong version of the PictInfo structure. Available in Mac OS X v10.0 and later.
`pictInfoIDErr`	-11001	The internal consistancy check for the PictInfoID is wrong. Available in Mac OS X v10.0 and later.

Result Code	Value	Description
`pictInfoVerbErr`	-11002	The PictInfo verb is not valid. Available in Mac OS X v10.0 and later.
`cantLoadPickMethodErr`	-11003	Unable to load the custom pick method resource. Available in Mac OS X v10.0 and later.
`colorsRequestedErr`	-11004	The number of colors requested is illegal. Available in Mac OS X v10.0 and later.
`pictureDataErr`	-11005	The picture data is not valid. Available in Mac OS X v10.0 and later.

Deprecated Picture Utilities 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

DisposeCalcColorTableUPP

Disposes of a universal procedure pointer (UPP) to a color table calculation callback. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
void DisposeCalcColorTableUPP (
   CalcColorTableUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer.

Discussion

For more information, see `CalcColorTableProcPtr` (page 7).

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`PictUtils.h`

DisposeDisposeColorPickMethodUPP

Disposes of a universal procedure pointer (UPP) to a method disposal callback. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
void DisposeDisposeColorPickMethodUPP (
   DisposeColorPickMethodUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer.

Discussion

For more information, see `DisposeColorPickMethodProcPtr` (page 9).

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`PictUtils.h`

DisposeInitPickMethodUPP

Disposes of a universal procedure pointer (UPP) to a method initialization callback. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
void DisposeInitPickMethodUPP (
   InitPickMethodUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer.

Discussion

For more information, see `InitPickMethodProcPtr` (page 9).

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`PictUtils.h`

DisposePictInfo

Disposes of the private data structures allocated by the `NewPictInfo` function. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
OSErr DisposePictInfo (
   PictInfoID thePictInfoID
);
```

Parameters

thePictInfoID

The unique identifier returned by `NewPictInfo`.

Return Value

A result code. See "Picture Utilities Result Codes" (page 21).

Discussion

The `DisposePictInfo` function does not dispose of any of the handles returned to you in a `PictInfo` structure by the `RetrievePictInfo` (page 37) function. Instead, you can dispose of a `Palette` structure by using the `DisposePalette` function. You can dispose of a `ColorTable` structure by using the `DisposeCTable` function. Dispose of other allocations with the `DisposeHandle` function.

Use this function when you are finished gathering information from a survey of pictures, pixel maps, or bitmaps.

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

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

`PictUtils.h`

DisposeRecordColorsUPP

Disposes of a universal procedure pointer (UPP) to a color recording callback. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
void DisposeRecordColorsUPP (
   RecordColorsUPP userUPP
);
```

Parameters

userUPP

The universal procedure pointer.

Discussion

For more information, see `RecordColorsProcPtr` (page 11).

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`PictUtils.h`

GetPictInfo

Gathers information about a single picture. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
OSErr GetPictInfo (
   PicHandle thePictHandle,
   PictInfo *thePictInfo,
   short verb,
   short colorsRequested,
   short colorPickMethod,
   short version
);
```

Parameters

thePictHandle

A handle to a picture.

thePictInfo

On return, a pointer to a `PictInfo` (page 15) structure, which holds information about the picture. Initially, all of the fields in the new `PictInfo` structure are set to `NULL`. Relevant fields are set to appropriate values depending on the information you request using the `GetPictInfo` function.

This function collects information from black-and-white pictures and bitmaps, and is supported in System 7 even by computers running only basic QuickDraw. However, when collecting color information on a computer running only basic QuickDraw, the function returns `NULL` instead of a handle to a `Palette` or `ColorTable` structure.

verb

A value indicating what type of information you want `GetPictInfo` to return in the `PictInfo` structure. See "Color Information Type" (page 20) for a description of the values you can use in this parameter.

You can specify whether you want color information (in a `ColorTable` structure, a `Palette` structure, or both), whether you want picture comment information, and whether you want font information. If you want color information, be sure to use the `colorPickMethod` parameter to specify the method by which to select colors.

Because the Palette Manager adds black and white when creating a `Palette` structure, you can specify the number of colors you want minus 2 in the `colorsRequested` parameter and specify the `suppressBlackAndWhite` constant in the `verb` parameter when gathering colors destined for a `Palette` structure or a screen.

colorsRequested

From 1 to 256, the number of colors you want in the `ColorTable` or `Palette` structure returned via the `PictInfo` structure. If you are not requesting colors (that is, if you pass the `recordComments` or `recordFontInfo` constant in the `verb` parameter), specify 0 in this parameter.

colorPickMethod

The method by which colors are selected for the `ColorTable` or `Palette` structure returned via the `PictInfo` structure. See "Color Selection Method" (page 20) for a description of the values you can use here.

You can also create your own color-picking method in a resource file of type `'cpmt'` and pass its resource ID in the `colorPickMethod` parameter. The resource ID must be greater than 127.

version

Always set this parameter to 0.

Return Value

A result code. See "Picture Utilities Result Codes" (page 21).

Discussion

The Picture Utilities provide two color-picking methods: one (specified by the `popularMethod` constant) that gives you the most frequently used colors and one (specified by the `medianMethod` constant) that gives you the widest range of colors. Each has advantages in different situations. For example, suppose the picture of a forest image contains 400 colors, of which 300 are greens, 80 are browns, and the rest are a scattering of golden sunlight effects. If you ask for the 250 most used colors, you will probably receive all greens. If you ask for a range of 250 colors, you will receive an assortment stretching from the greens and golds to the browns, including colors in between that might not actually appear in the image. If you specify the `systemMethod` constant, the Picture Utilities choose the method; currently they always choose `popularMethod`. You can also supply a color-picking method of your own.

If your application uses more than one color-picking method, it should present the user with a choice of which method to use.

When you are finished with the information in the `PictInfo` structure, use the Memory Manager function `DisposeHandle` to dispose of the `PictInfo`, `CommentSpec`, and `FontSpec` structures. Dispose of the `Palette` structure by using the `DisposePalette` function. Dispose of the `ColorTable` structure by using the `DisposeCTable` function.

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

When you ask for color information, `GetPictInfo` takes into account only the version 2 and extended version 2 picture opcodes `RGBFgCol`, `RGBBkCol`, `BkPixPat`, `PnPixPat`, `FillPixPat`, `HiliteColor` and pixel map or bitmap data. Each occurrence of these opcodes is treated as 1 pixel, regardless of the number and sizes of the objects drawn with that color. If you need an accurate set of colors from a complex picture, create an image of the picture in an offscreen pixel map, and then call the `GetPixMapInfo` (page 28) function to obtain color information about that pixel map.

The `GetPictInfo` function returns a bit depth of 1 on QuickTime-compressed `'PICT'` files. However, when QuickTime is installed, QuickTime decompresses and displays the image correctly.

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

`PictUtils.h`

GetPixMapInfo

Gathers color information about a single pixel map or bitmap. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
OSErr GetPixMapInfo (
   PixMapHandle thePixMapHandle,
   PictInfo *thePictInfo,
   short verb,
   short colorsRequested,
   short colorPickMethod,
   short version
);
```

Parameters

`thePixMapHandle`

A handle to a pixel map or bitmap.

thePictInfo

On return, a pointer to a `PictInfo` (page 15) structure, which holds information about a pixel map or bitmap. Initially, all of the fields in a new `PictInfo` structure are set to `NULL`. Relevant fields are set to appropriate values depending on the information you request using the `GetPixMapInfo` function.

This function also collects information from black-and-white pictures and bitmaps, and is supported in System 7 even by computers running only basic QuickDraw. However, when collecting color information on a computer running only basic QuickDraw, this function returns `NULL` instead of a handle to a `Palette` or `ColorTable` structure.

verb

A value indicating whether you want color information returned in a `ColorTable` structure, a `Palette` structure, or both. You can also request that black and white not be included among the returned colors. See "Color Information Type" (page 20) for a description of the values you can use here.

Because the Palette Manager adds black and white when creating a `Palette` structure, you can specify the number of colors you want minus 2 in the `colorsRequested` parameter and specify the constant `suppressBlackAndWhite` in the `verb` parameter when gathering colors destined for a `Palette` structure or a screen.

colorsRequested

From 1 to 256, the number of colors you want in the `ColorTable` or `Palette` structure returned via the `PictInfo` structure.

colorPickMethod

The method by which colors are selected for the `ColorTable` or `Palette` structure returned via the `PictInfo` structure. See "Color Selection Method" (page 20) for a description of the values you can use here.

You can also create your own color-picking method in a resource file of type `'cpmt'` and pass its resource ID in the `colorPickMethod` parameter. The resource ID must be greater than 127.

version

Always set this parameter to 0.

Return Value
A result code. See "Picture Utilities Result Codes" (page 21).

Discussion
The Picture Utilities provide two color-picking methods: one that gives you the most frequently used colors and one that gives you the widest range of colors. If you specify the `systemMethod` constant, the Picture Utilities choose that method. Currently they always choose `popularMethod`. You can also supply a color-picking method of your own.

When you are finished with the information in the `PictInfo` structure, be sure to dispose of it. Use the Memory Manager function `DisposeHandle` to dispose of the `PictInfo` structure. Dispose of the `Palette` structure by using the `DisposePalette` function. Dispose of the `ColorTable` structure by using the `DisposeCTable` function.

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

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
`PictUtils.h`

InvokeCalcColorTableUPP

Invokes a color table calculation callback, using a universal procedure pointer. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
OSErr InvokeCalcColorTableUPP (
   UInt32 dataRef,
   SInt16 colorsRequested,
   void *colorBankPtr,
   CSpecArray resultPtr,
   CalcColorTableUPP userUPP
);
```

Discussion
For parameter descriptions, see `CalcColorTableProcPtr` (page 7).

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

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

Declared In
`PictUtils.h`

InvokeDisposeColorPickMethodUPP

Invokes a method disposal callback, using a universal procedure pointer. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
OSErr InvokeDisposeColorPickMethodUPP (
   UInt32 dataRef,
   DisposeColorPickMethodUPP userUPP
);
```

Discussion
For more information, see `DisposeColorPickMethodProcPtr` (page 9).

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

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

Declared In
`PictUtils.h`

InvokeInitPickMethodUPP

Invokes a method initialization callback, using a universal procedure pointer. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
OSErr InvokeInitPickMethodUPP (
   SInt16 colorsRequested,
   UInt32 *dataRef,
   SInt16 *colorBankType,
   InitPickMethodUPP userUPP
);
```

Discussion
For parameter descriptions, see `InitPickMethodProcPtr` (page 9).

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

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

Declared In
`PictUtils.h`

InvokeRecordColorsUPP

Invokes a color recording callback, using a universal procedure pointer. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
OSErr InvokeRecordColorsUPP (
   UInt32 dataRef,
   RGBColor *colorsArray,
   SInt32 colorCount,
   SInt32 *uniqueColors,
   RecordColorsUPP userUPP
);
```

Discussion
For parameter descriptions, see `RecordColorsProcPtr` (page 11).

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

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

Declared In
`PictUtils.h`

NewCalcColorTableUPP

Creates a new universal procedure pointer (UPP) to a color table calculation callback. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
CalcColorTableUPP NewCalcColorTableUPP (
   CalcColorTableProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your color table calculation callback. For more information, see `CalcColorTableProcPtr` (page 7).

Return Value
A UPP to the callback.

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

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

Declared In
`PictUtils.h`

NewDisposeColorPickMethodUPP

Creates a new universal procedure pointer (UPP) to a method disposal callback. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
DisposeColorPickMethodUPP NewDisposeColorPickMethodUPP (
   DisposeColorPickMethodProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your method disposal callback. For more information, see `DisposeColorPickMethodProcPtr` (page 9).

Return Value
A UPP to the callback.

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

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

Declared In
`PictUtils.h`

NewInitPickMethodUPP

Creates a new universal procedure pointer (UPP) to a method initialization callback. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
InitPickMethodUPP NewInitPickMethodUPP (
   InitPickMethodProcPtr userRoutine
);
```

Parameters

userRoutine

A pointer to your method initialization callback. For more information, see `InitPickMethodProcPtr` (page 9).

Return Value
A UPP to the callback.

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

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

Declared In
`PictUtils.h`

NewPictInfo

Begins collecting pictures, pixel maps, and bitmaps for a survey of pictures. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
OSErr NewPictInfo (
   PictInfoID *thePictInfoID,
   short verb,
   short colorsRequested,
   short colorPickMethod,
   short version
);
```

Parameters

thePictInfoID

On return, a value that uniquely identifies your collection of pictures, pixel maps, or bitmaps.

verb

A value indicating what type of information you want the `RetrievePictInfo` (page 37) function to return in a `PictInfo` (page 15) structure. See "Color Information Type" (page 20) for a description of the values you can use here.

The constants `recordComments` and `recordFontInfo` and the values they represent have no effect when gathering information about the pixel maps and bitmaps included in your survey.

Because the Palette Manager adds black and white when creating a palette, you can specify the number of colors you want minus 2 in the `colorsRequested` parameter and specify the constant `suppressBlackAndWhite` in the `verb` parameter when gathering colors destined for a `Palette` structure or a screen.

colorsRequested

From 1 to 256, the number of colors you want included in the `ColorTable` or `Palette` structure returned by the `RetrievePictInfo` function via a `PictInfo` structure.

colorPickMethod

The method by which colors are selected for the `ColorTable` or `Palette` structure included in the `PictInfo` structure returned by the `RetrievePictInfo` function. See "Color Selection Method" (page 20) for a description of the values you can use here.

You can also create your own color-picking method in a resource file of type `'cpmt'` and pass its resource ID in the `colorPickMethod` parameter. The resource ID must be greater than 127.

version

Always set this parameter to 0.

Return Value

A result code. See "Picture Utilities Result Codes" (page 21).

Discussion

To add the information for a picture to your survey, use the `RecordPictInfo` function. To add the information for a pixel map or a bitmap to your survey, use the `RecordPixMapInfo` (page 36) function. For each of these functions, identify the survey with the ID number returned by `NewPictInfo`.

Use the `RetrievePictInfo` function to return information about the pictures, pixel maps, and bitmaps in the survey. The `RetrievePictInfo` function returns your requested information in a `PictInfo` structure.

Use the `verb` parameter for `NewPictInfo` to specify whether you want to gather comment or font information for the pictures in the survey. If you want to gather color information, use the `verb` parameter for `NewPictInfo` to specify whether you want this information in a `ColorTable` structure, a `Palette` structure,

or both. The `PictInfo` structure returned by the `RetrievePictInfo` function will then include a handle to a `ColorTable` structure or a `Palette` structure, or handles to both. If you want color information, be sure to use the `colorPickMethod` parameter to specify the method by which to select colors.

The Picture Utilities provide two color-picking methods: one (specified by the `popularMethod` constant) that gives you the most frequently used colors and one (specified by the `medianMethod` constant) that gives you the widest range of colors. If you specify the `systemMethod` constant, the Picture Utilities choose the method; currently they always choose `popularMethod`. You can also supply a color-picking method of your own.

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

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
`PictUtils.h`

NewRecordColorsUPP

Creates a new universal procedure pointer (UPP) to a color recording callback. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
RecordColorsUPP NewRecordColorsUPP (
   RecordColorsProcPtr userRoutine
);
```

Parameters

`userRoutine`
A pointer to your color recording callback. For more information, see `RecordColorsProcPtr` (page 11).

Return Value
A UPP to the callback.

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

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

Declared In
`PictUtils.h`

RecordPictInfo

Adds a picture to an informational survey of multiple pictures. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
OSErr RecordPictInfo (
   PictInfoID thePictInfoID,
   PicHandle thePictHandle
);
```

Parameters

thePictInfoID

The ID number—returned by the `NewPictInfo` (page 34) function—that identifies the survey to which you are adding the picture.

thePictHandle

A handle to the picture being added to the survey.

Return Value

A result code. See "Picture Utilities Result Codes" (page 21).

Discussion

The `RecordPictInfo` function adds the picture you specify in the parameter `thePictHandle` to the survey of pictures identified by the parameter `thePictInfoID`. Use `RecordPictInfo` repeatedly to add additional pictures to your survey.

After you have collected all of the pictures you need, use the `RetrievePictInfo` (page 37) function to return information about pictures in the survey.

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

When you ask for color information, `RecordPictInfo` takes into account only the version 2 and extended version picture opcodes `RGBFgCol`, `RGBBkCol`, `BkPixPat`, `PnPixPat`, `FillPixPat`, and `HiliteColor`. Each occurrence of these opcodes is treated as 1 pixel, regardless of the number and sizes of the objects drawn with that color. If you need an accurate set of colors from a complex picture, create an image of the picture in an offscreen pixel map, and then call the `GetPixMapInfo` (page 28) function to obtain color information about that pixel map.

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

`PictUtils.h`

RecordPixMapInfo

Adds a pixel map or bitmap to an informational survey of multiple pixel maps and bitmaps. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
OSErr RecordPixMapInfo (
   PictInfoID thePictInfoID,
   PixMapHandle thePixMapHandle
);
```

Parameters

thePictInfoID

The ID number—returned by the NewPictInfo (page 34) function—that identifies the survey to which you are adding the pixel map or bitmap.

thePixMapHandle

A handle to a pixel map or bitmap to be added to the survey.

Return Value

A result code. See "Picture Utilities Result Codes" (page 21).

Discussion

The RecordPixMapInfo function adds the pixel map or bitmap you specify in the parameter thePixMapHandle to the survey identified by the parameter thePictInfoID. Use RecordPictInfo repeatedly to add additional pixel maps and bitmaps to your survey.

After you have collected all of the images you need, use the RetrievePictInfo (page 37) function to return information about all the images in the survey.

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

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

PictUtils.h

RetrievePictInfo

Returns information about all the pictures, pixel maps, and bitmaps included in a survey. (Deprecated in Mac OS X v10.4. Use Quartz 2D instead; see *Quartz Programming Guide for QuickDraw Developers*.)

```
OSErr RetrievePictInfo (
   PictInfoID thePictInfoID,
   PictInfo *thePictInfo,
   short colorsRequested
);
```

Parameters

thePictInfoID

The ID number, returned by the NewPictInfo (page 34) function, that identifies the survey of pictures, pixel maps, and bitmaps.

thePictInfo

On return, a pointer to the `PictInfo` (page 15) structure that holds information about the pictures or images in the survey.

This function also collects information from black-and-white pictures and bitmaps, and is supported in System 7 even by computers running only basic QuickDraw. However, when collecting color information on a computer running only basic QuickDraw, the function returns `NULL` instead of a handle to a `Palette` or `ColorTable` structure.

colorsRequested

From 1 to 256, the number of colors you want returned in the `ColorTable` or `Palette` structure included in the `PictInfo` structure.

Return Value
A result code. See "Picture Utilities Result Codes" (page 21).

Discussion
After using the `NewPictInfo` function to create a new survey, and then using `RecordPictInfo` to add pictures to your survey and `RecordPixMapInfo` to add pixel maps and bitmaps to your survey, call `RetrievePictInfo`.

When you are finished with the information in the `PictInfo` structure, dispose of the `Palette` structure by using the `DisposePalette` function. Dispose of the `ColorTable` structure with the `DisposeCTable` function. Dispose of other allocations with the `DisposeHandle` function. Use the `DisposePictInfo` function to dispose of the private data structures created by the `NewPictInfo` function.

Special Considerations

Because Quartz 2D uses an entirely different approach to graphics than used by QuickDraw, there is no one-to-one correlation between QuickDraw and Quartz 2D functions. However, because Quartz offers many new features and improved performance compared to QuickDraw, it is worthwhile making the effort to convert your graphics code to Quartz.

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
`PictUtils.h`

Document Revision History

This table describes the changes to *Picture Utilities Reference*.

Date	Notes
2006-07-13	Updated for Mac OS X v10.5.
2006-07-24	Deprecated the manager. Use Quarz 2D instead.
2003-02-01	Updated to include Mac OS X availability information.

Document Revision History

Index

C

CalcColorTableProcPtr callback 7
CalcColorTableUPP data type 12
cantLoadPickMethodErr constant 22
Color Bank Type 19
Color Information Type 20
Color Selection Method 20
ColorBankIs555 constant 20
ColorBankIsCustom constant 19
ColorBankIsExactAnd555 constant 19
colorsRequestedErr constant 22
CommentSpec structure 12

D

DisposeCalcColorTableUPP function (Deprecated in Mac OS X v10.4) 23
DisposeColorPickMethodProcPtr callback 9
DisposeColorPickMethodUPP data type 13
DisposeDisposeColorPickMethodUPP function (Deprecated in Mac OS X v10.4) 23
DisposeInitPickMethodUPP function (Deprecated in Mac OS X v10.4) 24
DisposePictInfo function (Deprecated in Mac OS X v10.4) 25
DisposeRecordColorsUPP function (Deprecated in Mac OS X v10.4) 25

F

FontSpec structure 13

G

GetPictInfo function (Deprecated in Mac OS X v10.4) 26
GetPixMapInfo function (Deprecated in Mac OS X v10.4) 28

I

InitPickMethodProcPtr callback 9
InitPickMethodUPP data type 14
InvokeCalcColorTableUPP function (Deprecated in Mac OS X v10.4) 30
InvokeDisposeColorPickMethodUPP function (Deprecated in Mac OS X v10.4) 30
InvokeInitPickMethodUPP function (Deprecated in Mac OS X v10.4) 31
InvokeRecordColorsUPP function (Deprecated in Mac OS X v10.4) 31

M

medianMethod constant 20

N

NewCalcColorTableUPP function (Deprecated in Mac OS X v10.4) 32
NewDisposeColorPickMethodUPP function (Deprecated in Mac OS X v10.4) 32
NewInitPickMethodUPP function (Deprecated in Mac OS X v10.4) 33
NewPictInfo function (Deprecated in Mac OS X v10.4) 34
NewRecordColorsUPP function (Deprecated in Mac OS X v10.4) 35

P

PictInfo structure 15
PictInfoID data type 18
pictInfoIDErr constant 21
pictInfoVerbErr constant 22
pictInfoVersionErr constant 21
pictureDataErr constant 22
popularMethod constant 20

R

RecordColorsProcPtr callback 11
RecordColorsUPP data type 19
recordComments constant 21
recordFontInfo constant 21
RecordPictInfo function (Deprecated in Mac OS X v10.4) 36
RecordPixMapInfo function (Deprecated in Mac OS X v10.4) 36
RetrievePictInfo function (Deprecated in Mac OS X v10.4) 37
returnColorTable constant 21
returnPalette constant 21

S

suppressBlackAndWhite constant 21
systemMethod constant 20