CGLayer Reference

Graphics & Imaging > Quartz

2006-12-22

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

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

Simultaneously published in the United States and Canada.

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

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

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

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

Contents

CGLayer Reference

Derived From:	CFType
Framework:	ApplicationServices/ApplicationServices.h
Declared in	CGLayer.h

Overview

CGLayer objects are useful for offscreen drawing and can be used in much the same way that a bitmap context can be used. In fact, a CGLayer object is a much better representation than a bitmap context.

Using CGLayer objects can improve performance, particularly when you need to capture a piece of drawing that you stamp repeatedly (using the same scale factor and orientation). Quartz can cache CGLayer objects to the video card, making drawing a CGLayer to a destination much faster than rendering the equivalent image constructed from a bitmap context.

A CGLayer object is created relative to a graphics context. Although layer uses this graphics context as a reference for initialization, you are not restricted to drawing the layer to this graphics context. You can draw the layer to other graphics contexts, although any limitations of the original context are imposed. For example, if you create a CGLayer object using a bitmap context, the layer is rendered as a bitmap when drawn to any other graphics context.

You can use a CGLayer when you want to apply a shadow to a group of objects (such as a group of circles) rather than to individual objects.

Use these layers in your code whenever you can, especially when:

- You need to reuse a filled or stroked shape.
- You are building a scene and at least some of it can be reused. Put the reusable drawing in its own CGLayer.

Any CG object that you draw repeatedly—including CGPath, CGShading, and CGPDFPage—benefit from improved performance if you draw it to a CGLayer object.

Functions by Task

Creating Layer Objects

`CGLayerCreateWithContext` (page 7)
Creates a CGLayer object that is associated with a graphics context.

Drawing Layer Content

`CGContextDrawLayerInRect` (page 7)
: Draws the contents of a CGLayer object into the specified rectangle.

`CGContextDrawLayerAtPoint` (page 6)
: Draws the contents of a CGLayer object at the specified point.

Retaining and Releasing Layers

`CGLayerRelease` (page 9)
: Decrements the retain count of a CGLayer object.

`CGLayerRetain` (page 10)
: Increments the retain count of a CGLayer object.

Getting the CFType ID for a Layer

`CGLayerGetTypeID` (page 9)
: Returns the unique type identifier used for CGLayer objects.

Getting Layer Information

`CGLayerGetSize` (page 9)
: Returns the width and height of a CGLayer object.

`CGLayerGetContext` (page 8)
: Returns the graphics context associated with a CGLayer object.

Functions

CGContextDrawLayerAtPoint

Draws the contents of a CGLayer object at the specified point.

```
void CGContextDrawLayerAtPoint (
   CGContextRef context,
   CGPoint point,
   CGLayerRef layer
);
```

Parameters

`context`
: The graphics context associated with the layer.

`point`
: The location, in current user space coordinates, to use as the origin for the drawing.

layer

The layer whose contents you want to draw.

Discussion

Calling the function `CGContextDrawLayerAtPoint` is equivalent to calling the function `CGContextDrawLayerInRect` with a rectangle that has its origin at `point` and its size equal to the size of the layer.

Availability

Available in Mac OS X version 10.4 and later.

Declared In

`CGLayer.h`

CGContextDrawLayerInRect

Draws the contents of a CGLayer object into the specified rectangle.

```
void CGContextDrawLayerInRect (
   CGContextRef context,
   CGRect rect,
   CGLayerRef layer
);
```

Parameters

context

The graphics context associated with the layer.

rect

The rectangle, in current user space coordinates, to draw to.

layer

The layer whose contents you want to draw.

Discussion

The contents are scaled, if necessary, to fit into the rectangle.

Availability

Available in Mac OS X version 10.4 and later.

Declared In

`CGLayer.h`

CGLayerCreateWithContext

Creates a CGLayer object that is associated with a graphics context.

```
CGLayerRef CGLayerCreateWithContext (
    CGContextRef context,
    CGSize size,
    CFDictionaryRef auxiliaryInfo
);
```

Parameters

context

The graphics context you want to create the layer relative to. The layer uses this graphics context as a reference for initialization.

size

The size, in default user space units, of the layer relative to the graphics context.

auxiliaryInfo

Reserved for future use. Pass `NULL`.

Return Value

A CGLayer object. You are responsible for releasing this object using the function `CGLayerRelease` (page 9) when you no longer need the layer.

Discussion

After you create a CGLayer object, you should reuse it whenever you can to facilitate the Quartz caching strategy. Quartz caches any objects that are reused, including CGLayer objects. Objects that are reused frequently remain in the cache. In contrast, objects that are used once in a while may be moved in and out of the cache according to their frequency of use. If you don't reuse CGLayer objects, Quartz won't cache them. This means that you lose an opportunity to improve the performance of your application.

Availability

Available in Mac OS X version 10.4 and later.

Declared In

`CGLayer.h`

CGLayerGetContext

Returns the graphics context associated with a CGLayer object.

```
CGContextRef CGLayerGetContext (
    CGLayerRef layer
);
```

Parameters

layer

The layer whose graphics context you want to obtain.

Return Value

The graphics context associated with the layer.

Discussion

The context that's returned is the context for the layer itself, not the context that you specified when you created the layer.

Availability

Available in Mac OS X version 10.4 and later.

Declared In
`CGLayer.h`

CGLayerGetSize

Returns the width and height of a CGLayer object.

```
CGSize CGLayerGetSize (
    CGLayerRef layer
);
```

Parameters

`layer`

The layer whose width and height you want to obtain.

Return Value
The width and height of the layer, in default user space coordinates.

Availability
Available in Mac OS X version 10.4 and later.

Declared In
`CGLayer.h`

CGLayerGetTypeID

Returns the unique type identifier used for CGLayer objects.

```
CFTypeID CGLayerGetTypeID (
    void
);
```

Return Value
The type identifier for CGLayer objects.

Discussion
A type identifier is an integer that identifies the opaque type to which a Core Foundation object belongs. You use type IDs in various contexts, such as when you are operating on heterogeneous collections.

Availability
Available in Mac OS X version 10.4 and later.

Declared In
`CGLayer.h`

CGLayerRelease

Decrements the retain count of a CGLayer object.

```
void CGLayerRelease (
    CGLayerRef layer
);
```

Parameters

layer

The layer to release.

Discussion

This function is equivalent to calling `CFRelease (layer)` except that it does not crash (as `CFRetain` does) if the `layer` parameter is `null`.

Availability

Available in Mac OS X version 10.4 and later.

Declared In

`CGLayer.h`

CGLayerRetain

Increments the retain count of a CGLayer object.

```
CGLayerRef CGLayerRetain (
    CGLayerRef layer
);
```

Parameters

layer

The layer to retain.

Return Value

The same layer you passed in as the `layer` parameter.

Discussion

This function is equivalent to calling `CFRetain (layer)` except that it does not crash (as `CFRetain` does) if the `layer` parameter is `null`.

Availability

Available in Mac OS X version 10.4 and later.

Declared In

`CGLayer.h`

Data Types

CGLayerRef

An opaque type used for offscreen drawing.

```
typedef struct CGLayer *CGLayerRef;
```

Availability

Available in Mac OS X v10.4 and later.

Declared In

`CGLayer.h`

Document Revision History

This table describes the changes to *CGLayer Reference*.

Date	Notes
2006-12-22	Minor technical corrections and editorial changes.
	Modified the description of the `size` parameter used by `CGLayerCreateWithContext` (page 7).
	Grouped functions by their use.
	Modified the description of the size parameter for `CGLayerCreateWithContext` (page 7).
2005-07-07	Fixed typographical errors and revised wording of the function CGLayerGetContext.
2005-04-29	Corrected typos.
	First version.

Index