

---

# QCRenderer Class Reference

[Cocoa > Graphics & Imaging](#)



2007-05-09



Apple Inc.  
© 2007 Apple Inc.  
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

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

Apple, the Apple logo, Cocoa, Mac, Mac OS, and Quartz are trademarks of Apple Inc., registered in the United States and other countries.

OpenGL is a registered trademark of Silicon Graphics, Inc.

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

---

## **QCRenderer Class Reference 5**

---

Overview 5

Tasks 6

    Creating and Initializing a Renderer 6

    Rendering a Composition 6

    Getting the Composition Object 6

    Taking Snapshot Images 6

Instance Methods 6

    composition 6

    createSnapshotImageOfType: 7

    initWithCGSize:colorSpace:composition: 7

    initWithCGLContext:pixelFormat:colorSpace:composition: 8

    initWithComposition:colorSpace: 8

    initWithOpenGLContext:pixelFormat:file: 9

    renderAtTime:arguments: 9

    snapshotImage 10

Constants 10

    Rendering Arguments 10

---

## **Document Revision History 13**

---

## **Index 15**

---



# QCRenderer Class Reference

---

<b>Inherits from</b>	NSObject
<b>Conforms to</b>	QCCompositionRenderer NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/Quartz.framework/Frameworks/QuartzComposer.framework
<b>Availability</b>	Available in Mac OS X v10.4 and later.
<b>Declared in</b>	QCRenderer.h

## Overview

A `QCRenderer` class is designed for low-level rendering of Quartz Composer compositions. This is the class to use if you want to be in charge of rendering a composition to a specific OpenGL context—either using the `NSOpenGLContext` class or a `CGLContextObj` object. `QCRenderer` also allows you to load, play, and control a composition.

To render a composition to a specific OpenGL context:

- Create an instance of `QCRenderer` using one of the initialization methods, such as `initWithOpenGLContext:pixelFormat:file:` (page 9).
- Render frames by calling the method `renderAtTime:arguments:` (page 9)
- If you use double buffering in OpenGL, you must swap the OpenGL buffers.
- Release the renderer with you no longer need it.

This code snippet shows how to implement these tasks:

```
NSOpenGLContext* context = [myNSOpenGLView openGLContext];
NSOpenGLPixelFormat* format = [myNSOpenGLView pixelFormat];
NSString* path = @"/Users/MyName/MyComposition.qtz";
QCRenderer* myRenderer;
// Create a Quartz Composer renderer.
myRenderer = [[QCRenderer alloc] initWithOpenGLContext:context
                                           pixelFormat:format
                                           file:path];

// Render the first 10 seconds of the composition with steps of 1/25s.
for(double t = 0.0; t <= 10.0; t += 1.0/25.0)
{
    [myRenderer renderAtTime:t arguments:nil];
    [context flushBuffer]; //Required on double-buffered contexts
}
// Clean up
```

```
[renderer release];
```

## Tasks

### Creating and Initializing a Renderer

- [initWithComposition:colorSpace:](#) (page 8)  
Creates a renderer object with a composition object and a color space.
- [initWithOpenGLContext:pixelFormat:file:](#) (page 9)  
Creates a renderer object with an `NSOpenGLContext` object and a composition file.
- [initWithCGLContext:pixelFormat:colorSpace:composition:](#) (page 8)  
Creates a renderer object with a `CGLContextObj` object, a pixel format, a color space, and a composition object.
- [initOffScreenWithSize:colorSpace:composition:](#) (page 7)  
Creates an offscreen renderer of a given size with the provided color space and composition object.

### Rendering a Composition

- [renderAtTime:arguments:](#) (page 9)  
Renders a frame of a composition at the specified time.

### Getting the Composition Object

- [composition](#) (page 6)  
Returns the composition object associated with the renderer.

### Taking Snapshot Images

- [snapshotImage](#) (page 10)  
Returns an `NSImage` object of the current image in the OpenGL context associated with the renderer.
- [createSnapshotImageOfType:](#) (page 7)  
Returns the current image in the OpenGL context associated with the renderer, as an image object of the provided image type.

## Instance Methods

### **composition**

Returns the composition object associated with the renderer.

```
- (QCComposition*) composition
```

**Return Value**

The composition object.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

QCRenderer.h

**createSnapshotImageOfType:**

Returns the current image in the OpenGL context associated with the renderer, as an image object of the provided image type.

```
- (id) createSnapshotImageOfType:(NSString*)type
```

**Parameters**

*type*

A string that specifies any of the following image types: NSBitmapImageRep, NSImage, CIImage, CGImage, CVOpenGLBuffer, CVPixelBuffer.

**Return Value**

The snapshot image in the provided image type. You are responsible for releasing this object when you no longer need it.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

QCRenderer.h

**initWithSize:colorSpace:composition:**

Creates an offscreen renderer of a given size with the provided color space and composition object.

```
- (id) initWithSize:(NSSize)size colorSpace:(CGColorSpaceRef)colorSpace
    composition:(QCComposition*)composition
```

**Parameters**

*size*

The size of the offscreen renderer.

*colorSpace*

A Quartz color space object. This must be an RGB color space. Pass `NULL` to use the default RGB color space. For more information on Quartz color spaces, see *Quartz 2D Programming Guide*.

*composition*

A `QCComposition` object.

**Return Value**

The initialized `QCRenderer` object or `nil` if initialization is not successful.

**Discussion**

This method creates an internal OpenGL context and pixel buffer. Because offscreen rendering is performed on the GPU, the maximum rendering size is limited to the GPU capacity. On typical hardware, the limit is at least 2048 by 2048, but is often 4096 by 4096. The available VRAM affects performance.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

QCRenderer.h

**initWithCGLContext:pixelFormat:colorSpace:composition:**

Creates a renderer object with a `CGLContextObj` object, a pixel format, a color space, and a composition object.

```
- (id) initWithCGLContext:(CGLContextObj)context
    pixelFormat:(CGLPixelFormatObj)format colorSpace:(CGColorSpaceRef)colorSpace
    composition:(QCComposition*)composition;
```

**Parameters**

*context*

A `CGLContextObj` object. The object that you supply must have both a color and a depth buffer.

*format*

A `CGLPixelFormatObj` object.

*colorSpace*

A Quartz color space object. This must be an RGB color space. Pass `NULL` to use the default RGB color space. For more information on Quartz color spaces, see *Quartz 2D Programming Guide*.

*composition*

A `QCComposition` object.

**Return Value**

The initialized `QCRenderer` object or `nil` if initialization is not successful.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

QCRenderer.h

**initWithComposition:colorSpace:**

Creates a renderer object with a composition object and a color space.

```
- (id) initWithComposition:(QCComposition*)composition
    colorSpace:(CGColorSpaceRef)colorSpace;
```

**Parameters**

*composition*

A `QCComposition` object. The composition must not contain any consumer patches. That is, the composition can receive data, process it, and produce output values, but it cannot perform any rendering.



*colorSpace*

A Quartz color space object. This must be an RGB color space. Pass `NULL` to use the default RGB color space. The color space is used only for the images produced by the output image ports of the composition. For more information on Quartz color spaces, see *Quartz 2D Programming Guide*.

**Return Value**

The initialized `QCRenderer` object or `nil` if initialization is not successful.

**Discussion**

Note that `snapshotImage` (page 10) and `createSnapshotImageOfType:` (page 7) always returns `nil` on such `QCRenderer` instances.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

`QCRenderer.h`

**initWithOpenGLContext:pixelFormat:file:**

Creates a renderer object with an `NSOpenGLContext` object and a composition file.

```
- (id)initWithOpenGLContext:(NSOpenGLContext *)context
    pixelFormat:(NSOpenGLPixelFormat *)format file:(NSString *)path
```

**Parameters***context*

An `NSOpenGLContext` object. The object that you supply must have both a color and a depth buffer.

*format*

An `NSOpenGLPixelFormat` object.

*path*

A string that specifies the location of a composition (`.qtz`) file.

**Return Value**

An initialized `QCRenderer` object or `nil` if initialization is not successful.

**Availability**

Available in Mac OS X v10.4 and later.

**Declared In**

`QCRenderer.h`

**renderAtTime:arguments:**

Renders a frame of a composition at the specified time.

```
- (BOOL)renderAtTime:(NSTimeInterval)time arguments:(NSDictionary *)arguments
```

**Parameters***time*

The time, in seconds, at which to render a composition frame. The time must be a positive value or zero.

*arguments*

An optional dictionary that can have any of the entries defined in “[Rendering Arguments](#)” (page 10).

**Return Value**

YES if successful.

**Discussion**

You need to call this method each time you want to render a frame of the composition.

All OpenGL states are preserved *except* the following:

- States defined by `GL_CURRENT_BIT`
- Textures on each unit and the environment mode
- Matrix mode

If you are using double buffers, keep in mind that the `renderAtTime:arguments:` method does not swap the front and back buffers of the OpenGL context. You must perform the swap yourself by calling the OpenGL command `flushBuffer` on the context associated with the renderer.

If you are interleaving OpenGL code with rendering of a composition, make sure that the OpenGL context is current. If you are using the `NSOpenGLContext` class, call the `makeCurrentContext` method prior to rendering. If you are using the CGL API, call the function `CGLSetCurrentContext`.

**Availability**

Available in Mac OS X v10.4 and later.

**Declared In**

`QCRenderer.h`

**snapshotImage**

Returns an `NSImage` object of the current image in the OpenGL context associated with the renderer.

```
- (NSImage*) snapshotImage
```

**Return Value**

The snapshot image.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

`QCRenderer.h`

## Constants

**Rendering Arguments**

Arguments that you can pass to the `renderAtTime:arguments:` (page 9) method.

```
extern NSString* const QCRendererEventKey;  
extern NSString* const QCRendererMouseLocationKey;
```

### Constants

`QCRendererEventKey`

A key for a renderer event. The associated value is an `NSEvent` object.

Available in Mac OS X v10.4 and later.

Declared in `QCRenderer.h`.

`QCRendererMouseLocationKey`

A key for the mouse location. The associated value is an `NSPoint` object stored in an `NSValue` object. The mouse location is in normalized coordinates relative to the OpenGL context viewport  $[0, 1] \times [0, 1]$  with the origin  $(0, 0)$  at the lower-left corner.

Available in Mac OS X v10.4 and later.

Declared in `QCRenderer.h`.

### Declared In

`QCRenderer.h`



# Document Revision History

---

This table describes the changes to *QCRenderer Class Reference*.

Date	Notes
2007-05-09	Updated for Mac OS X v10.5.
	Added <a href="#">composition</a> (page 6), <a href="#">snapshotImage</a> (page 10), <a href="#">createSnapshotImageOfType:</a> (page 7), <a href="#">initWithComposition:colorSpace:</a> (page 8), <a href="#">initWithScreenWithSize:colorSpace:composition:</a> (page 7), and <a href="#">initWithCGLContext:pixelFormat:colorSpace:composition:</a> (page 8).
2006-06-28	Added hyperlinks.
2006-05-23	First publication of this content as a separate document.
	Edited content to comply with new guidelines. Added See Also sections and parameter descriptions.

## REVISION HISTORY

### Document Revision History

# Index

---

## C

---

composition **instance method** [6](#)  
createSnapshotImageOfType: **instance method** [7](#)

## I

---

initWithScreenWithSize:colorSpace:composition:  
**instance method** [7](#)  
initWithCGLContext:pixelFormat:colorSpace:  
composition: **instance method** [8](#)  
initWithComposition:colorSpace: **instance method**  
[8](#)  
initWithOpenGLContext:pixelFormat:file:  
**instance method** [9](#)

## Q

---

QCRendererEventKey **constant** [11](#)  
QCRendererMouseLocationKey **constant** [11](#)

## R

---

renderAtTime:arguments: **instance method** [9](#)  
Rendering Arguments [10](#)

## S

---

snapshotImage **instance method** [10](#)