
QCPlugInOutImageProvider Protocol 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, Objective-C, 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

QCPlugInOutputImageProvider Protocol Reference 5

Overview 5

Tasks 5

 Rendering an Image to a Destination 5

 Providing Information About the Image 6

 Providing Information About the Rendering Destination 6

Instance Methods 6

 canRenderWithCGLContext: 6

 copyRenderedTextureForCGLContext:pixelFormat:bounds:isFlipped: 7

 imageBounds 7

 imageColorSpace 8

 releaseRenderedTexture:forCGLContext: 8

 renderToBuffer:withBytesPerRow:pixelFormat:forBounds: 8

 renderWithCGLContext:forBounds: 9

 shouldColorMatch 10

 supportedBufferPixelFormat 10

 supportedRenderedTexturePixelFormat 10

Document Revision History 13

Index 15

QCPlugInOutputImageProvider Protocol Reference

Framework	/System/Library/Frameworks/Quartz.framework/Frameworks/QuartzComposer.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	QCPlugIn.h

Overview

The `QCPlugInOutputImageProvider` protocol eliminates the need to use explicit image types for the image output ports on a custom patch. The methods in this protocol are called by the Quartz Composer engine when the output image is needed. If your custom patch has an image output port, you need to implement the appropriate methods for rendering image data and to supply information about the rendering destination and the image bounds.

Output images are opaque provider objects that comply to this protocol. To create an image output port as an Objective-C 2.0 property, declare it as follows:

```
@property(dynamic) id<QCPlugInOutputImageProvider> outputImage;
```

To create an image input port dynamically use the type `QCPortTypeImage`:

```
[self addOutputPortWithType:QCPortTypeImage  
    forKey:@"outputImage"  
    withAttributes:nil];
```

To write images to that port, you need to implement the methods in this protocol and create an internal class that represents the images produced by the custom patch. For example, a simple interface for an image provider is:

```
@interface MyOutputImage : NSObject <QCPlugInOutputImageProvider>  
{  
    NSUInteger _width;  
    NSUInteger _height;  
}
```

Tasks

Rendering an Image to a Destination

- [renderToBuffer:withBytesPerRow:pixelFormat:forBounds:](#) (page 8)

Renders a subregion of the image into the supplied memory buffer using the specified pixel format.

- [copyRenderedTextureForCGLContext:pixelFormat:bounds:isFlipped:](#) (page 7)
Returns the name of an OpenGL texture of type `GL_TEXTURE_RECTANGLE_EXT` that contains a subregion of the image in a given pixel format.
- [renderWithCGLContext:forBounds:](#) (page 9)
Renders a subregion of the image to the provided CGL context.
- [releaseRenderedTexture:forCGLContext:](#) (page 8)
Releases the previously copied texture.

Providing Information About the Image

- [imageBounds](#) (page 7)
Returns the bounds of the image expressed in pixels and aligned to integer boundaries.
- [imageColorSpace](#) (page 8)
Returns the color space of the image or `NULL` if the image should not be color matched.
- [shouldColorMatch](#) (page 10)
Returns whether the image should be color matched.

Providing Information About the Rendering Destination

- [supportedBufferPixelFormat](#) (page 10)
Returns a list of pixel formats that are supported for rendering to a memory buffer.
- [supportedRenderedTexturePixelFormat](#) (page 10)
Returns a list of pixel formats that are supported for rendering to an onscreen OpenGL context.
- [canRenderWithCGLContext:](#) (page 6)
Returns whether the image data can be rendered into the provided CGL context.

Instance Methods

canRenderWithCGLContext:

Returns whether the image data can be rendered into the provided CGL context.

- (BOOL) `canRenderWithCGLContext:(CGLContextObj)cgl_ctx`

Parameters

ctx

The CGL context that your image will be rendered to.

Return Value

YES if the image can be rendered into this CGL context; otherwise NO, in which case [renderToBuffer:withBytesPerRow:pixelFormat:forBounds:](#) (page 8) is called.

Discussion

If your image can render using any OpenGL context, simply return YES. If your code requires special extensions, you'll need to check for them and then provide the appropriate return value. For more information on checking for OpenGL capabilities supported by the hardware, see *OpenGL Programming Guide for Mac OS X*.

Availability

Available in Mac OS X v10.5 and later.

Declared In

QCPlugIn.h

copyRenderedTextureForCGLContext:pixelFormat:bounds:isFlipped:

Returns the name of an OpenGL texture of type `GL_TEXTURE_RECTANGLE_EXT` that contains a subregion of the image in a given pixel format.

```
- (GLuint) copyRenderedTextureForCGLContext:(CGLContextObj)cgl_ctx
    pixelFormat:(NSString*)format bounds:(NSRect)bounds isFlipped:(BOOL*)flipped
```

Parameters

cgl_ctx

The CGL context to render to.

format

A string that represents the pixel format of the texture.

bounds

The bounds of the subregion of the image.

isFlipped

Set to YES on output if the contents of the returned texture are vertically flipped.

Return Value

The name of an OpenGL texture of type `GL_TEXTURE_RECTANGLE_EXT` that contains a subregion of the image in a given pixel format or 0 if the texture can't be provided.

Discussion

Implement this method if you want to create the texture yourself or use framebuffer objects (FBO). Use `<OpenGL/CGLMacro.h>` to send commands to the OpenGL context. Make sure to preserve all the OpenGL states except the ones defined by `GL_CURRENT_BIT`.

Availability

Available in Mac OS X v10.5 and later.

Declared In

QCPlugIn.h

imageBounds

Returns the bounds of the image expressed in pixels and aligned to integer boundaries.

```
- (NSRect) imageBounds;
```

Return Value

The bounds of the image. Note that the `QCPlugIn` class does not support images that have infinite bounds.

Availability

Available in Mac OS X v10.5 and later.

Declared In

QCPlugIn.h

imageColorSpace

Returns the color space of the image or NULL if the image should not be color matched.

```
- (CGColorSpaceRef) imageColorSpace
```

Return Value

The color space of the image or NULL.

Availability

Available in Mac OS X v10.5 and later.

Declared In

QCPlugIn.h

releaseRenderedTexture:forCGLContext:

Releases the previously copied texture.

```
- (void) releaseRenderedTexture:(GLuint)name forCGLContext:(CGLContextObj)cgl_ctx;
```

Parameters

name

The name of the previously bound texture.

cgl_ctx

The CGL context.

Discussion

Your OpenGL code should save and restore all states *except* for those that are part of `GL_CURRENT_BIT` (vertex position, color, texture, and so on). Also use CGL macros instead of changing the current context, by including this statement:

```
#import <OpenGL/CGLMacro.h>
```

For more details, see *Quartz Composer Custom Patch Programming Guide*.

Availability

Available in Mac OS X v10.5 and later.

Declared In

QCPlugIn.h

renderToBuffer:withBytesPerRow:pixelFormat:forBounds:

Renders a subregion of the image into the supplied memory buffer using the specified pixel format.


```
- (BOOL) renderToBuffer:(void*)baseAddress withBytesPerRow:(NSUInteger)rowBytes
    pixelFormat:(NSString*)format forBounds:(NSRect)bounds
```

Parameters*baseAddress*

The base address of the memory buffer. The Quartz Composer engine passes you an address that is aligned on a 16-byte boundary.

rowBytes

The number of bytes per row of the image data. The Quartz Composer engine guarantees this value is a multiple of 16.

format

The pixel format of the image data.

bounds

The bounds of the subregion.

Return Value

YES if the image is rendered successfully into the buffer; NO on failure or if the image provider doesn't support CPU rendering.

Discussion

The Quartz Composer engine calls this method when it needs pixels. It gives you the base address, the number of row bytes, and the format. Then, you write pixels to the buffer.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [renderWithCGLContext:forBounds:](#) (page 9)

Declared In

QCPlugIn.h

renderWithCGLContext:forBounds:

Renders a subregion of the image to the provided CGL context.

```
- (BOOL) renderWithCGLContext:(CGLContextObj)cgl_ctx forBounds:(NSRect)bounds
```

Parameters*cgl_ctx*

The CGL context to render to.

bounds

The bounds of the subregion.

Return Value

YES if successful; NO on failure or if the image provider doesn't support GPU rendering.

Discussion

The view port is set for you. The model view and projection matrixes are set to the identity.

Your OpenGL code should save and restore all states *except* for those that are part of `GL_CURRENT_BIT` (vertex position, color, texture, and so on). Also use CGL macros instead of changing the current context, by including this statement:

```
#import <OpenGL/CGLMacro.h>
```

For more details, see *Quartz Composer Custom Patch Programming Guide*.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [renderToBuffer:withBytesPerRow:pixelFormat:forBounds:](#) (page 8)

Declared In

QCPlugIn.h

shouldColorMatch

Returns whether the image should be color matched.

- (BOOL) shouldColorMatch

Return Value

NO if the image is a mask or gradient; otherwise YES, which is the default.

Availability

Available in Mac OS X v10.5 and later.

Declared In

QCPlugIn.h

supportedBufferPixelFormat

Returns a list of pixel formats that are supported for rendering to a memory buffer.

- (NSArray*) supportedBufferPixelFormat

Return Value

A list of pixel formats, in order of preference, that the image can be rendered to in memory, or nil if the image provider does not support rendering to the CPU.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [supportedRenderedTexturePixelFormat](#) (page 10)

Declared In

QCPlugIn.h

supportedRenderedTexturePixelFormat

Returns a list of pixel formats that are supported for rendering to an onscreen OpenGL context.

- (NSArray*) supportedRenderedTexturePixelFormat

Return Value

Returns the list of texture pixel formats supported by [copyRenderedTextureForCGLContext:pixelFormat:bounds:isFlipped:](#) (page 7) or `nil` if not supported.

Discussion

If this method returns `nil`, then Quartz Composer calls [canRenderWithCGLContext:](#) (page 6) / [/renderWithCGLContext:forBounds:](#) (page 9).

Availability

Available in Mac OS X v10.5 and later.

See Also

- [supportedBufferPixelFormatFormats](#) (page 10)

Declared In

QCPlugIn.h

Document Revision History

This table describes the changes to *QCPlugInOutputImageProvider Protocol Reference*.

Date	Notes
2007-05-09	New document that describes the methods for managing image data that's produced as output from a QCPlugIn object.

REVISION HISTORY

Document Revision History

Index

C

canRenderWithCGLContext: **protocol instance method 6**
copyRenderedTextureForCGLContext:pixelFormat:
 bounds:isFlipped: **protocol instance method 7**

I

imageBounds **protocol instance method 7**
imageColorSpace **protocol instance method 8**

R

releaseRenderedTexture:forCGLContext: **protocol instance method 8**
renderToBuffer:withBytesPerRow:pixelFormat:
 forBounds: **protocol instance method 8**
renderWithCGLContext:forBounds: **protocol instance method 9**

S

shouldColorMatch **protocol instance method 10**
supportedBufferPixelFormat **protocol instance method 10**
supportedRenderedTexturePixelFormat **protocol instance method 10**