CIContext Class Reference

Cocoa > Graphics & Imaging

2007-03-16

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

Simultaneously published in the United States and Canada.

Contents

CIContext Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	Library/Frameworks/QuartzCore.framework
Availability	Mac OS X v10.4 and later
Declared in	CIContext.h
Companion guides	Core Image Programming Guide Image Unit Tutorial
Related sample code	CIAnnotation Reducer UnsharpMask WebKitCIPlugIn WhackedTV

Overview

The `CIContext` class provides an evaluation context for rendering a `CIImage` object through Quartz 2D or OpenGL. You use `CIContext` objects in conjunction with other Core Image classes, such as `CIFilter`, `CIImage`, and `CIColor`, to take advantage of the built-in Core Image filters when processing images.

Tasks

Creating a Context

`+ contextWithCGContext:options:` (page 6)
Creates a Core Image context from a Quartz context, using the specified options.

`+ contextWithCGLContext:pixelFormat:options:` (page 7)
Creates a Core Image context from a CGL context, using the specified options and pixel format object.

Rendering Images

- `createCGImage:fromRect:` (page 8)
  Creates a Quartz 2D image from a region of a `CIImage` object.
- `createCGImage:fromRect:format:colorSpace:` (page 9)
  Creates a Quartz 2D image from a region of a `CIImage` object.
- `createCGLayerWithSize:info:` (page 10)
  Creates a CGLayer object from the provided parameters.
- `drawImage:atPoint:fromRect:` (page 11)
  Renders a region of an image to a point in the context destination.
- `drawImage:inRect:fromRect:` (page 11)
  Renders a region of an image to a rectangle in the context destination.
- `render:toBitmap:rowBytes:bounds:format:colorSpace:` (page 12)
  Renders to the given bitmap.

Managing Resources

- `clearCaches` (page 8)
  Frees any cached data, such as temporary images, associated with the context and runs the garbage collector.
- `reclaimResources` (page 12)
  Runs the garbage collector to reclaim any resources that the context no longer requires.

Class Methods

contextWithCGContext:options:

Creates a Core Image context from a Quartz context, using the specified options.

```
+ (CIContext *)contextWithCGContext:(CGContextRef)ctx options:(NSDictionary *)dict
```

Parameters

ctx

A Quartz graphics context (`CGContextRef` object) either obtained from the system or created using a Quartz function such as `CGBitmapContextCreate`. See *Quartz 2D Programming Guide* for information on creating Quartz graphics contexts.

dict

A dictionary that contains color space information. You can provide the keys `kCIContextOutputColorSpace` (page 13) or `kCIContextWorkingColorSpace` (page 13) along with a `CGColorSpaceRef`object for each color space.

Discussion

After calling this method, Core Image draws content to the specified Quartz graphics context.

When you create a `CIContext` object using a Quartz graphics context, any transformations that are already set on the Quartz graphics context affect drawing to that context.

Availability
Mac OS X v10.4 and later.

See Also
`+ contextWithCGLContext:pixelFormat:options:` (page 7)

Related Sample Code
CIAnnotation
UnsharpMask

Declared In
`CIContext.h`

contextWithCGLContext:pixelFormat:options:

Creates a Core Image context from a CGL context, using the specified options and pixel format object.

```
+ (CIContext *)contextWithCGLContext:(CGLContextObj)ctx
    pixelFormat:(CGLPixelFormatObj)pf options:(NSDictionary *)dict
```

Parameters

ctx

A CGL context (`CGLContextObj` object) obtain by calling the CGL function `CGLCreateContext`.

pf

A CGL pixel format object (`CGLPixelFormatObj` object) created by calling the CGL function `CGLChoosePixelFormat`. This argument must be the same pixel format object used to create the CGL context. The pixel format object must be valid for the lifetime of the Core Image context. Don't release the pixel format object until after you release the Core Image context.

options

A dictionary that contains color space information. You can provide the keys `kCIContextOutputColorSpace` (page 13) or `kCIContextWorkingColorSpace` (page 13) along with a `CGColorSpaceRef` object for each color space.

Discussion
After calling this method, Core Image draws content into the surface (drawable object) attached to the CGL context. A CGL context is an Mac OS X OpenGL context. For more information, see *OpenGL Programming Guide for Mac OS X*.

When you create a `CIContext` object using a CGL context, all OpenGL states set for the CGL context affect rendering to that context. That means that coordinate and viewport transformations set on the CGL context as well as the vertex color.

For best results, follow these guidelines when you use Core Image to render into an OpenGL context:

- Ensure that the a single unit in the coordinate space of the OpenGL context represents a single pixel in the output device.
- The Core Image coordinate space has the origin in the bottom left corner of the screen. You should configure the OpenGL context in the same way.
- The OpenGL context blending state is respected by Core Image. If the image you want to render contains translucent pixels, it's best to enable blending using a blend function with the parameters `GL_ONE`, `GL_ONE_MINUS_SRC_ALPHA`, as shown in the following code example.

Some typical initialization code for a view with width `W` and height `H` is:

```
glViewport (0, 0, W, H);
glMatrixMode (GL_PROJECTION);
glLoadIdentity ();
glOrtho (0, W, 0, H, -1, 1);
glMatrixMode (GL_MODELVIEW);
glLoadIdentity ();
glBlendFunc (GL_ONE, GL_ONE_MINUS_SRC_ALPHA);
glEnable (GL_BLEND);
```

Availability
Mac OS X v10.4 and later.

See Also
`+ contextWithCGContext:options:` (page 6)

Related Sample Code
CIVideoDemoGL
QTCoreImage101
VideoViewer
WebKitCIPlugIn
WhackedTV

Declared In
`CIContext.h`

Instance Methods

clearCaches

Frees any cached data, such as temporary images, associated with the context and runs the garbage collector.

```
- (void)clearCaches
```

Discussion
You can use this method to remove textures from the texture cache that reference deleted images.

Availability
Mac OS X v10.4 and later.

See Also
`- reclaimResources` (page 12)

Declared In
`CIContext.h`

createCGImage:fromRect:

Creates a Quartz 2D image from a region of a `CIImage` object.

```
- (CGImageRef)createCGImage:(CIImage *)im fromRect:(CGRect)r
```

Parameters

im

A `CIImage` object.

r

The region of the image to render.

Return Value
A Quartz 2D (`CGImageRef`) image. You are responsible for releasing the returned image when you no longer need it.

Discussion
Renders a region of an image into a temporary buffer using the context, then creates and returns a Quartz 2D image with the results.

Availability
Mac OS X v10.4 and later.

See Also
`- createCGImage:fromRect:format:colorSpace:` (page 9)

Related Sample Code
CIAnnotation

Declared In
`CIContext.h`

createCGImage:fromRect:format:colorSpace:

Creates a Quartz 2D image from a region of a `CIImage` object.

```
- (CGImageRef)createCGImage:(CIImage *)im fromRect:(CGRect)r
        format:(CIFormat)f colorSpace:(CGColorSpaceRef)cs
```

Parameters

im

A `CIImage` object.

r

The region of the image to render.

f

The format of the image.

cs

The color space of the image.

Return Value
A Quartz 2D (`CGImageRef`) image. You are responsible for releasing the returned image when you no longer need it.

Discussion
Renders a region of an image into a temporary buffer using the context, then creates and returns a Quartz 2D image with the results.

Availability
Mac OS X v10.5 and later.

See Also
- `createCGImage:fromRect:` (page 8)

Declared In
`CIContext.h`

createCGLayerWithSize:info:

Creates a CGLayer object from the provided parameters.

```
- (CGLayerRef)createCGLayerWithSize:(CGSize)size info:(CFDictionaryRef)d
```

Parameters

size

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

d

A dictionary, which is passed to `CGLayerCreateWithContext` as the `auxiliaryInfo` parameter. Pass `NULL` as this parameter is reserved for future use.

Return Value
A CGLayer (`CGLayerRef`) object.

Discussion
After calling this method, Core Image draws content into the CGLayer object. Core Image creates a CGLayer object by calling the Quartz 2D function `CGLayerCreateWithContext`, whose prototype is:

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

Core Image passes the `CIContext` object as the `context` parameter, the size as the `size` parameter, and the dictionary as the `auxiliaryInfo` parameter. For more information on CGLayer objects, see *Quartz 2D Programming Guide* and *CGLayer Reference*.

Availability
Mac OS X v10.4 and later.

See Also
- `+ imageWithCGLayer:`
- `+ imageWithCGLayer:options:`

Related Sample Code
QTCarbonCoreImage101

Declared In
`CIContext.h`

drawImage:atPoint:fromRect:

Renders a region of an image to a point in the context destination.

```
- (void)drawImage:(CIImage *)im atPoint:(CGPoint)p fromRect:(CGRect)src
```

Parameters

im

A `CIImage` object.

p

The point in the context destination to draw to.

src

The region of the image to draw.

Discussion

You can call this method to force evaluation of the result after you apply a filter using one of the methods of the `CIFilter` class, such as `apply:`, `apply:arguments:options:`, and `apply:k, . . .`.

Availability

Mac OS X v10.4 and later.

See Also

`- drawImage:inRect:fromRect:` (page 11)

Related Sample Code

QTCarbonCoreImage101

Reducer

Declared In

`CIContext.h`

drawImage:inRect:fromRect:

Renders a region of an image to a rectangle in the context destination.

```
- (void)drawImage:(CIImage *)im inRect:(CGRect)dest fromRect:(CGRect)src
```

Parameters

im

A `CIImage` object.

dest

The rectangle in the context destination to draw into.

src

The subregion of the image that you want to draw into the context, with the origin and target size defined by the `dest` parameter.

Discussion

You can call this method to force evaluation of the result after you you apply a filter using one of the methods of the `CIFilter` class, such as `apply:`, `apply:arguments:options:`, and `apply:k, . . .`.

Availability

Mac OS X v10.4 and later.

See Also
- drawImage:atPoint:fromRect: (page 11)

Related Sample Code
QTCarbonCoreImage101

Declared In
CIContext.h

reclaimResources

Runs the garbage collector to reclaim any resources that the context no longer requires.

```
- (void)reclaimResources
```

Discussion
The system calls this method automatically after every rendering operation. You can use this method to remove textures from the texture cache that reference deleted images.

Availability
Mac OS X v10.4 and later.

See Also
- clearCaches (page 8)

Declared In
CIContext.h

render:toBitmap:rowBytes:bounds:format:colorSpace:

Renders to the given bitmap.

```
- (void)render:(CIImage *)im toBitmap:(void *)data rowBytes:(ptrdiff_t)rb
    bounds:(CGRect)r format:(CIFormat)f colorSpace:(CGColorSpaceRef)cs
```

Parameters

im
A CIImage object.

data
Storage for the bitmap data.

rb
The bytes per row.

r
The bounds of the bitmap data.

f
The format of the bitmap data.

cs
The color space for the data. Pass NULL if you want to use the output color space of the context.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CIContext.h`

Constants

Context Options

Keys in the options dictionary for a `CIContext` object.

```
extern NSString *kCIContextOutputColorSpace;
extern NSString *kCIContextWorkingColorSpace;
extern NSString *kCIContextUseSoftwareRenderer;
```

Constants

`kCIContextOutputColorSpace`
: A key for the color space to use for images before they are rendered to the context. By default, Core Image uses the GenericRGB color space, which leaves color matching to the system. You can specify a different output color space by providing a Quartz 2D `CGColorSpace` object (`CGColorSpaceRef`). (See *Quartz 2D Programming Guide* for information on creating and using `CGColorSpace` objects.)

`kCIContextWorkingColorSpace`
: A key for the color space to use for image operations. By default, Core Image assumes that processing nodes are 128 bits-per-pixel, linear light, premultiplied RGBA floating-point values that use the GenericRGB color space. You can specify a different working color space by providing a Quartz 2D `CGColorSpace` object (`CGColorSpaceRef`). Note that the working color space must be RGB-based. If you have YUV data as input (or other data that is not RGB-based), you can use ColorSync functions to convert to the working color space. (See *Quartz 2D Programming Guide* for information on creating and using `CGColorSpace` objects.)

`kCIContextUseSoftwareRenderer`
: A key for enabling software renderer use. If the associated `NSNumber` object is `YES`, then the software renderer is required.

Declared In
`CIContext.h`

Document Revision History

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

Date	Notes
2007-03-16	Updated for Mac OS Xv10.5.
	Added `createCGImage:fromRect: format:colorSpace:` (page 9).
2006-06-28	Added more information on color spaces.
	Added default color space information to the constants `kCIContextOutputColorSpace` (page 13) and `kCIContextWorkingColorSpace` (page 13).
2006-05-23	First publication of this content as a separate document.
	Added parameter descriptions and updated Class Description.
	Added the constant `kCIContextUseSoftwareRenderer` (page 13).

Document Revision History

Index