Core Animation Cookbook

(Legacy)

Developer

Contents

Listings

Core Animation Cookbook

Important: This document may not represent best practices for current development. Links to downloads and other resources may no longer be valid.

This document provides instructions and code fragments that describe how to perform common Core Animation tasks.

Organization of This Document

This document has the following chapters:

- Drawing (page 5) describes various drawing techniques when working with layers.
- Timing (page 7) describes various timing techniques when working with animations.

See Also

These programming guides discuss some of the technologies that are used by Core Animation:

- *Core Animation Programming Guide* describes the Core Animation technology and shows how to use the Core Animation API.
- *Quartz 2D Programming Guide* describes the two-dimensional drawing engine used to draw the content of an `CALayer` instance.
- *Core Image Programming Guide* describes the OS X image processing technology and shows how to use the Core Image API.

Drawing

This chapter discusses drawing issues when using Core Animation and other technologies.

Drawing Layer Content With Application Kit Classes

Core Animation `CALayer` class defines a delegate method, `drawLayer:inContext:`, that you can implement and draw your layer content using Quartz 2D drawing functions. However, Cocoa developers who have complete and working drawing solutions based on the Application Kit drawing classes may wish to continue using that code.

Listing 1 shows an implementation of the CALayer delegate method `drawLayer:inContext:` that creates an `NSGraphicsContext` from the `CGContextRef` passed as the `inContext:` parameter. Layer delegates can use this technique to display content created using `NSBezierPath`, `NSColor`, `NSImage` and other Application Kit classes.

Listing 1 Drawing into a layer using Application Kit classes

```
- (void)drawLayer:(CALayer *)layer inContext:(CGContextRef)ctx
{
   NSGraphicsContext *nsGraphicsContext;
   nsGraphicsContext = [NSGraphicsContext graphicsContextWithGraphicsPort:ctx
                                                                  flipped:NO];
   [NSGraphicsContext saveGraphicsState];
   [NSGraphicsContext setCurrentContext:nsGraphicsContext];

   // ...Draw content using NS APIs...
   NSRect aRect=NSMakeRect(10.0,10.0,30.0,30.0);
   NSBezierPath *thePath=[NSBezierPath bezierPathWithRect:aRect];
   [[NSColor redColor] set];
   [thePath fill];
```



```
    [NSGraphicsContext restoreGraphicsState];
}
```


Timing

This chapter discusses timing issues when using Core Animation.

Using a Single Timing Function For a Keyframe Animation

The `CAKeyframeAnimation` class provides a powerful means of animating layer properties. However, `CAKeyframeAnimation` does not allow you to specify a single animation timing function that is used for the entire path. Instead you are required to specify the timing using the `keyTimes` property, or by specifying an array of timing functions in the `timingFunctions` property.

You can provide a single timing function for the animation by grouping the keyframe animation in a `CAAnimationGroup`, and setting the group animation's timing function to the desired `CAMediaTimingFunction`. The animation group's timing function and duration take precedence over the keyframe animation's timing properties.

A code fragment that implements this strategy is shown in Listing 1.

Listing 1 Using a single timing function for a keyframe animation

```
// create the path for the keyframe animation
CGMutablePathRef thePath = CGPathCreateMutable();
CGPathMoveToPoint(thePath,NULL,15.0f,15.f);
CGPathAddCurveToPoint(thePath,NULL,
                      15.f,250.0f,
                      295.0f,250.0f,
                      295.0f,15.0f);

// create an explicit keyframe animation that
// animates the target layer's position property
// and set the animation's path property
CAKeyframeAnimation *theAnimation=[CAKeyframeAnimation

                                   animationWithKeyPath:@"position"];
```



```
theAnimation.path=thePath;

// create an animation group and add the keyframe animation
CAAnimationGroup *theGroup = [CAAnimationGroup animation];
theGroup.animations=[NSArray arrayWithObject:theAnimation];

// set the timing function for the group and the animation duration
theGroup.timingFunction=[CAMediaTimingFunction

                                functionWithName:kCAMediaTimingFunctionEaseIn];
theGroup.duration=15.0;
// release the path
CFRelease(thePath);



// adding the animation to the target layer causes it
// to begin animating
[theLayer addAnimation:theGroup forKey:@"animatePosition"];
```


Document Revision History

This table describes the changes to *Core Animation Cookbook*.

Date	Notes
2014-09-17	Moved to Retired Documents Library.
2008-03-11	Corrected typos.
2007-10-31	Reorganized the content. Added new examples.
2007-05-15	New document that demonstrates common Core Animation tasks.

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