Layers and Animations

Layers and animations are bound in by Apple in a very unique way. Layers are special drawing areas, which can be animated.

Thus animations, as in CoreAnimation as part of Cocoa, are always based on layers.

Layer and Animation

Layers

Brief history:

Layers as part of an OS were initially developed for the iPhone. As soon as others realized how powerful they are, the technology was adapted for OS X and is now part of all operating system by Apple.

Layer and Animation

Layers

Layers are:

- rectangular drawing areas
- connected to an instance of (NS/UI)View
- in three-dimensional space

Layers have:

- drawable properties
- animatable content

Layer and Animation

Layers

Layers are not:

- three-dimensional objects
- independent drawing areas on screen
- mandatory for drawings on screen

Layer's Drawing

3 methods to provide content:

- set the layer's contents-property (e.g. as image)
- provide a drawing delegate
- subclass CALayer and override 'display'

Layer and Animation

Layer's Content

```
// create the layer and set the bounds and position

CALayer *theLayer = [CALayer layer];

theLayer.position = CGPointMake(50.0f,50.0f);
theLayer.bounds = CGRectMake(0.0f,0.0f,100.0f,100.0f);

 // set the contents property to a CGImageRef from elsewhere

theLayer.contents = theImage;
```

Layer with a Delegate

Method to show content:

```
...
// create the layer and set the bounds and position
CALayer *theLayer = [CALayer layer];
theLayer.position = ...
 // set the delegate property
theLayer.delegate = self;

...
- (void)displayLayer:(CALayer *)theLayer
{// Do some checks and set the content

    theLayer.contents = theImage;
}
```

Layer with a Delegate (draw)

Method to draw:

```
- (void)drawLayer:(CALayer *)theLayer inContext:
(CGContextRef)theContext
{
// Do drawing here
}
```

Subclassing CALayer

Again 2 methods, one to show content, and one to draw. One needs to be overwritten:

```
- (void)display
{
// Set the contents here
}


- (void)drawInContext:(CGContextRef)theContext
{
// Do drawing here
}
```

Layer's Drawing

Preferred methods:

- set the layer's contents-property directly

or

- provide a drawing delegate

Layer and Animation

Layers and their View

Both types of views, UIView and NSView, provide similar properties to access their layers.

On OS X, you need to implement:

```
[myView setWantsLayer: YES];
```

There is a distinction between layer-hosting and layer-backing on OS X.

...

Layer and Animation

Hosted Layer

While backed views are only for implicit drawings and animations, hosted layers are directly accessed. All properties, the layer's tree and the animations are at hand as a layer can provide them.

Layer and Animation

Layer's Tree

Layers inside a view can have their own hierarchy. All CALayer have the property "sublayers". If set, a complex tree can be build, with tens or hundreds or thousands of sublayers, each containing their own drawing, own properties, animation and sublayers.

Layer and Animation

Animation

An animation, as transformation over a given time, is the most prominent feature of CALayers:

All layers can be animated, and all properties of a layer are animatable.

Layer and Animation

Demo

Hornstein Impact

Fireworks:

http://developer.apple.com/mac/library/samplecode/Fireworks/Introduction/Intro.html

Layer and Animation