CoreGraphics + CoreAnimation

for aspiring **Superheroes**

April 3rd, 2012

Session 8

Philip Kluz
Wannabe-Superhero

“How to create pretty stuff and bestow life upon it.”

CoreGraphics (Quartz)

Hardware accelerated 2D rendering engine.

CoreAnimation

Library to simplify creation of animated user interfaces.

CoreGraphics (Quartz)

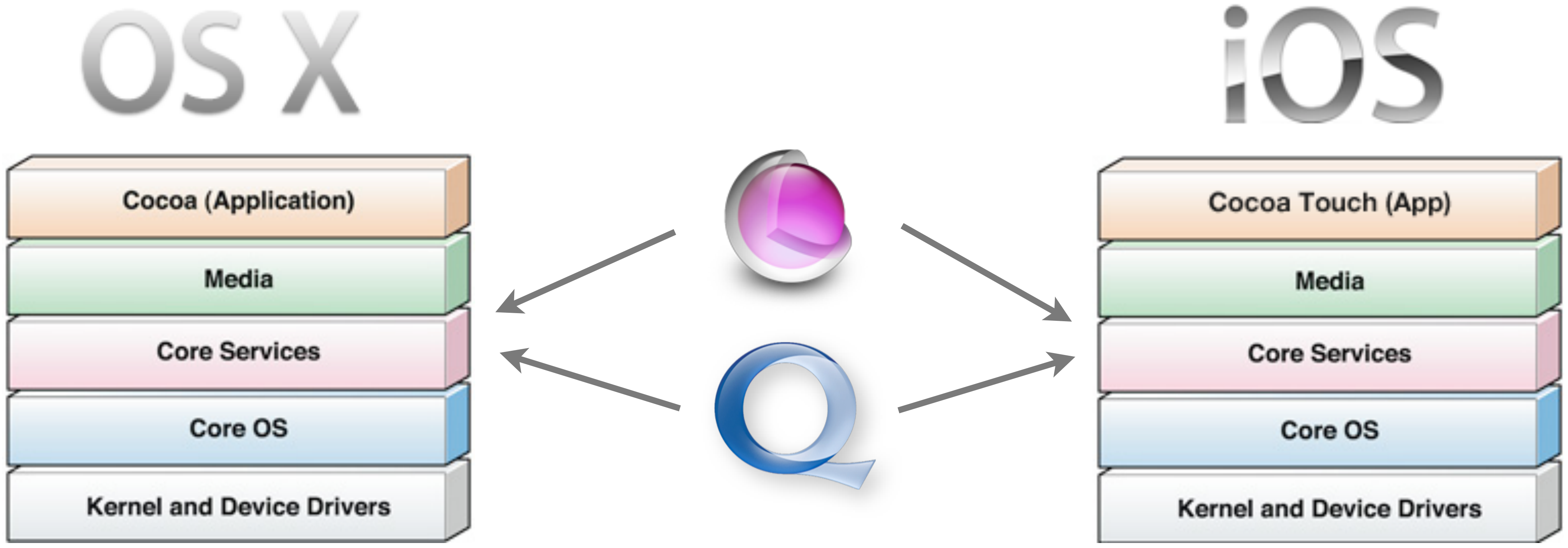
- Introduction
- Mac OS X vs iOS
- CG Crash Course

CoreAnimation

- Introduction
- UIKit Based Animation
- CA Crash Course

System Architectures

Two birds with one shot.

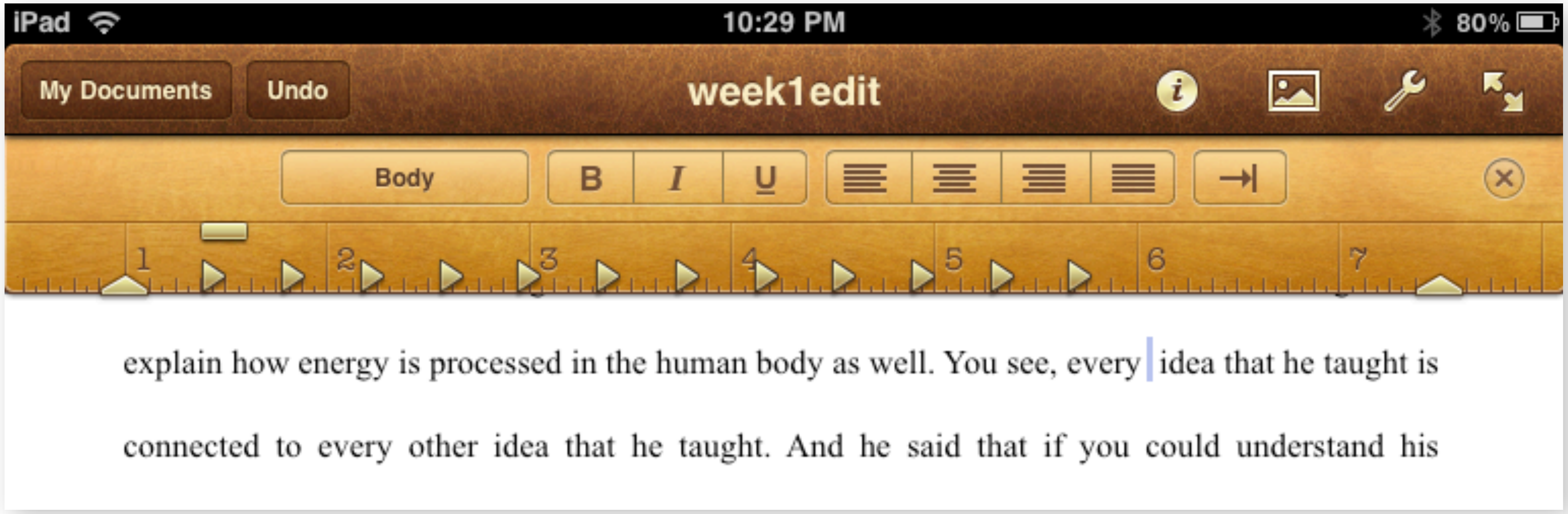


CoreGraphics

Creating “pretty stuff”.

CoreGraphics

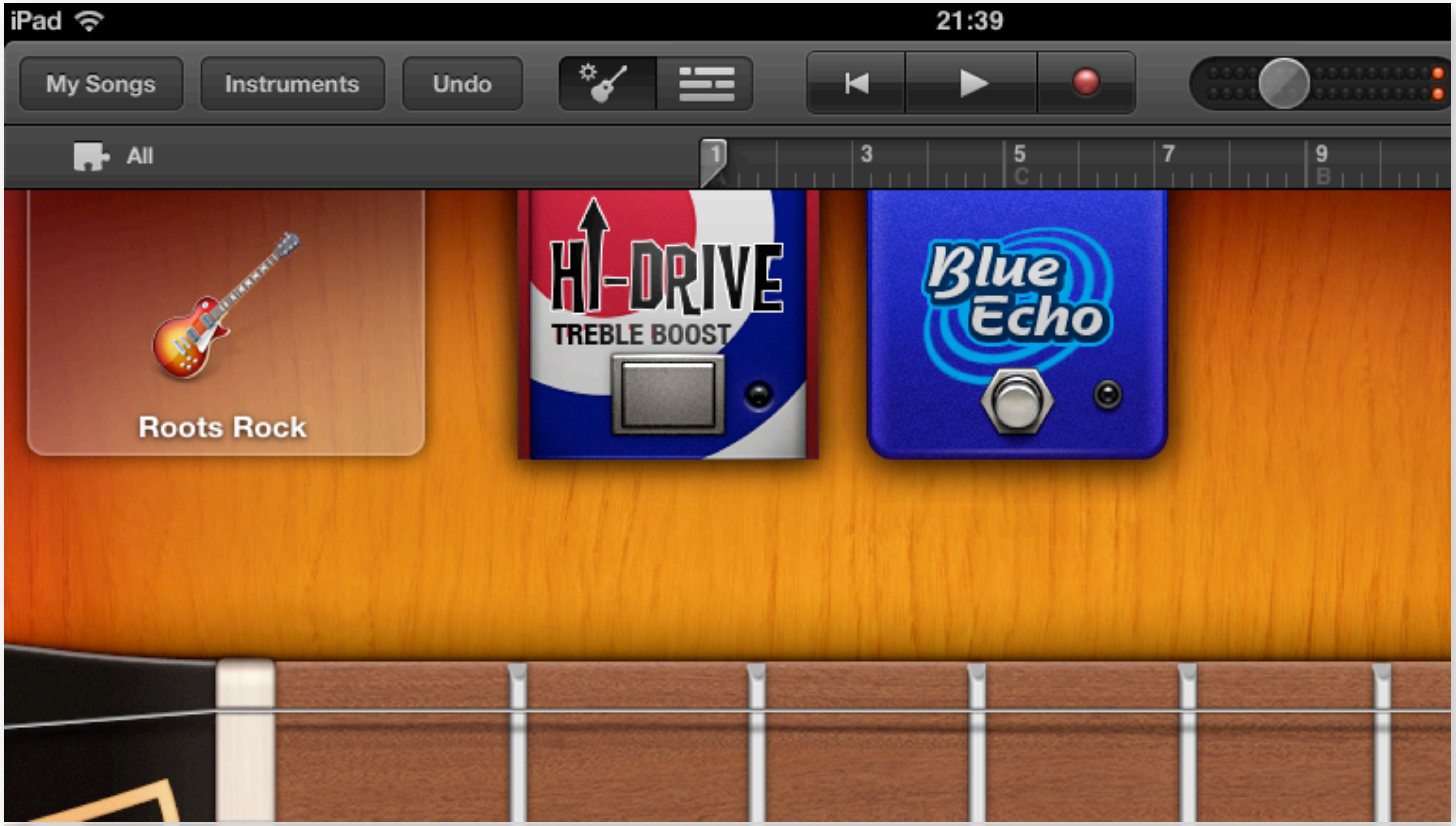
Pages



explain how energy is processed in the human body as well. You see, every idea that he taught is connected to every other idea that he taught. And he said that if you could understand his

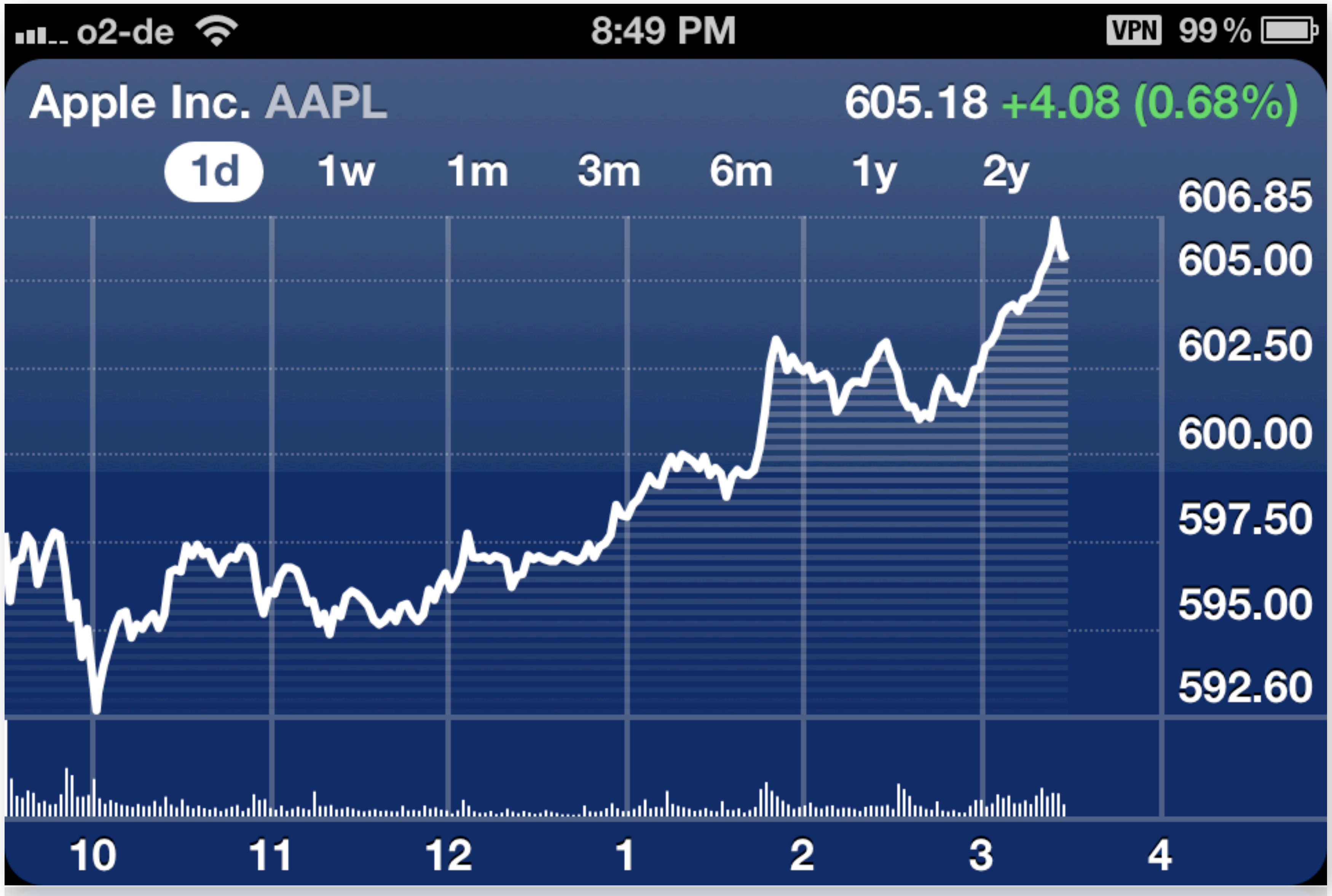
CoreGraphics

Garage Band



CoreGraphics

Stocks App

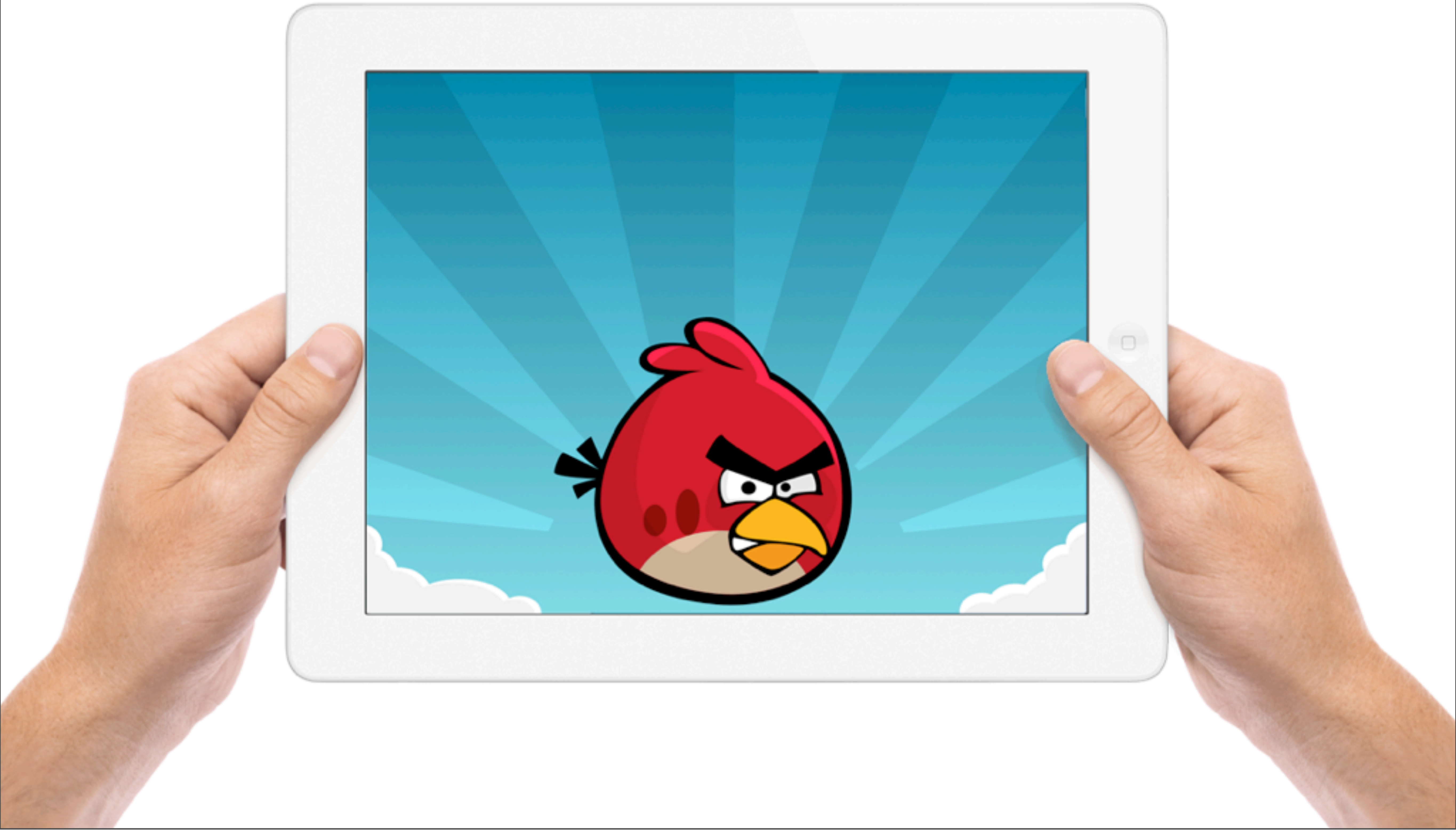


Reference Project (1)

“CoreBirdy”

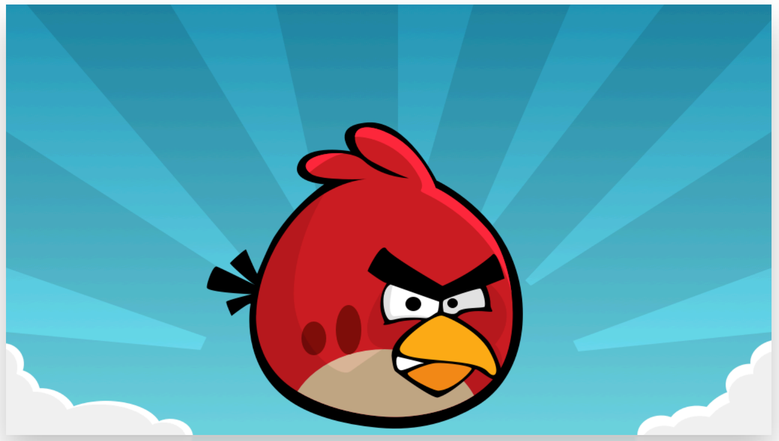
Reference Project (1)

“CoreBirdy”



Reference Project (1)

“CoreBirdy”



- Heavily Bézier curve based.
- Curves of high degree make it slower than using an image.
- Curves too complex to create without help of additional tools.
- Resolution independent!
- Very(!) small.

Essentially these are the pros and cons of CoreGraphics.

CoreGraphics

...should I use it?

Why...?

Reduces size of App! **Resolution independent!** **Blazingly fast!***

**When...?*

Whenever you can but not necessarily for geometrically complex shapes.

"If I can't wrap my head around the geometric description of an object, I will use an image."

Where...?

In the appropriate methods.

"Don't call for CoreGraphics, it calls you."

CoreGraphics

...how do I use it?

- “*In iOS all drawing occurs within the confines of a `UIView` object.*”
  (Source: Apple Documentation - “Graphics and Drawing in iOS”)

- Custom drawing requires subclassing of `UIView` and overriding

  ```
  - (void)drawRect:(CGRect)rect;
  ```

- **Don’t forget Quartz.** It’s a library, thus you need to **add** and **import** it!

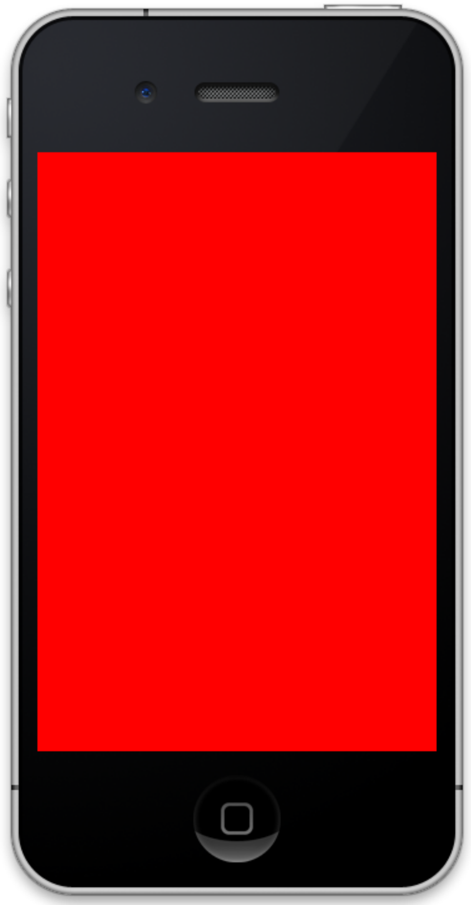
  ```
  #import <QuartzCore/QuartzCore.h>
  ```

CoreGraphics
Preparations

- Create a new Single-View project and call it “ColorFill”
- Subclass `UIView` and call it “`ColorFillView`”
- In your ViewController.xib set the main views class to `ColorFillView`.
- Add **`QuartzCore.framework`** to your project.
- `#import <QuartzCore/QuartzCore.h>` in `ColorFillView.m`
- Override `-drawRect:`

CoreGraphics

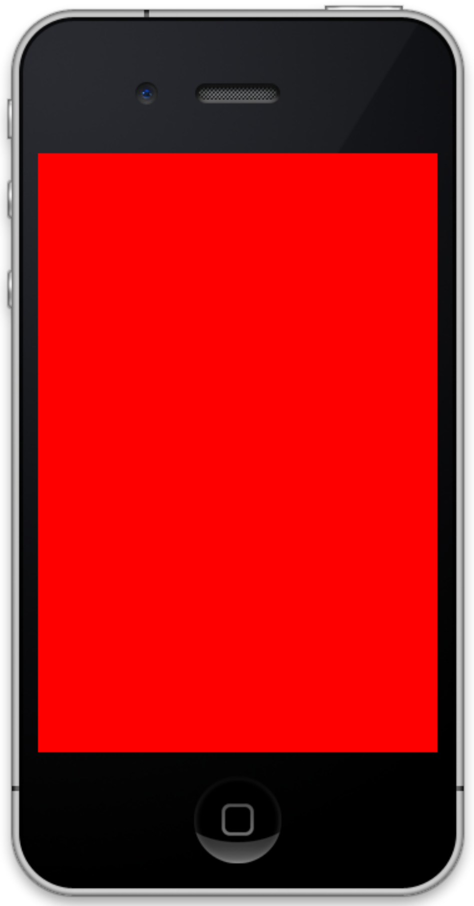
Color Fill



CoreGraphics

Color Fill

```
- (void)drawRect:(CGRect)rect
{
   [super drawRect:rect];
   CGContextRef context = UIGraphicsGetCurrentContext();
   CGFloat redColor[4] = {1.0f,0.0f,0.0f,1.0f};
   CGContextSetFillColor(context, redColor);
   CGContextFillRect(context, rect);
}
```

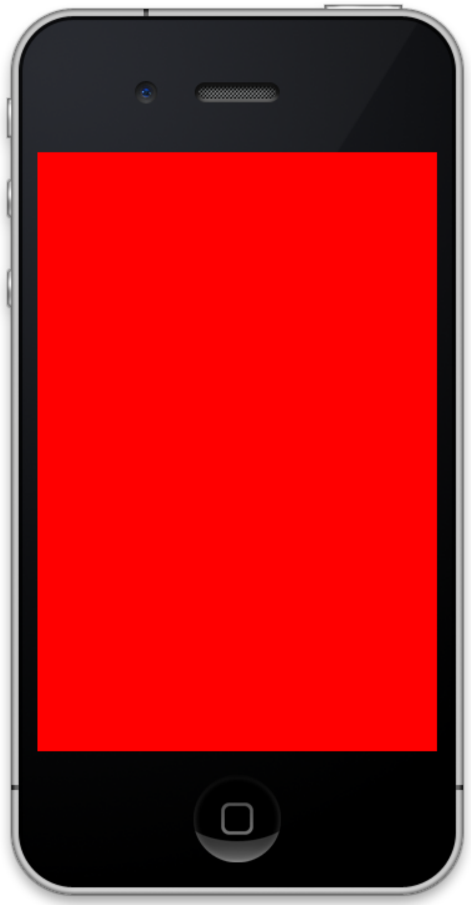


CoreGraphics

Color Fill

```
- (void)drawRect:(CGRect)rect
{
  [super drawRect:rect];
  CGContextRef context = UIGraphicsGetCurrentContext();
  CGFloat redColor[4] = {1.0f,0.0f,0.0f,1.0f};
  CGContextSetFillColor(context, redColor);
  CGContextFillRect(context, rect);
}
```

Grab the current drawing context.



CoreGraphics
Color Fill

```
- (void)drawRect:(CGRect)rect
{
   [super drawRect:rect];
   CGContextRef context = UIGraphicsGetCurrentContext();
   CGFloat redColor[4] = {1.0f,0.0f,0.0f,1.0f};
   CGContextSetFillColor(context, redColor);
   CGContextFillRect(context, rect);
}
```

Get some red color (RGBA)

CoreGraphics

Color Fill

```
- (void)drawRect:(CGRect)rect
{
   [super drawRect:rect];
   CGContextRef context = UIGraphicsGetCurrentContext();
   CGFloat redColor[4] = {1.0f,0.0f,0.0f,1.0f};
   CGContextSetFillColor(context, redColor);
   CGContextFillRect(context, rect);
}
```

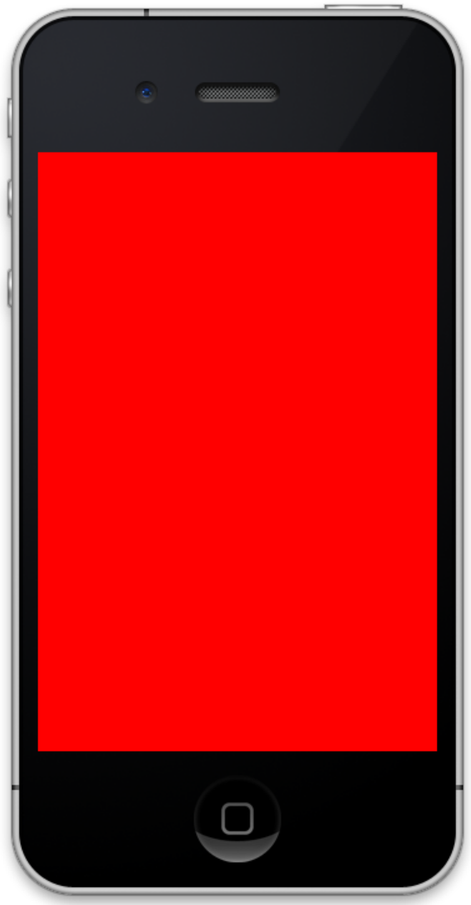
Set red as the filling color for the following actions in the given context.

CoreGraphics

Color Fill

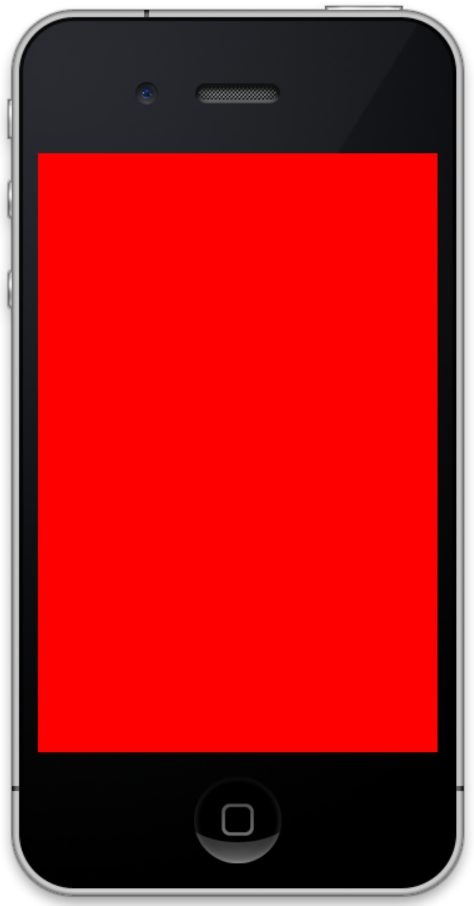
```
- (void)drawRect:(CGRect)rect
{
   [super drawRect:rect];
   CGContextRef context = UIGraphicsGetCurrentContext();
   CGFloat redColor[4] = {1.0f,0.0f,0.0f,1.0f};
   CGContextSetFillColor(context, redColor);
   CGContextFillRect(context, rect);
}
```

Fill the context with whatever color we just specified.



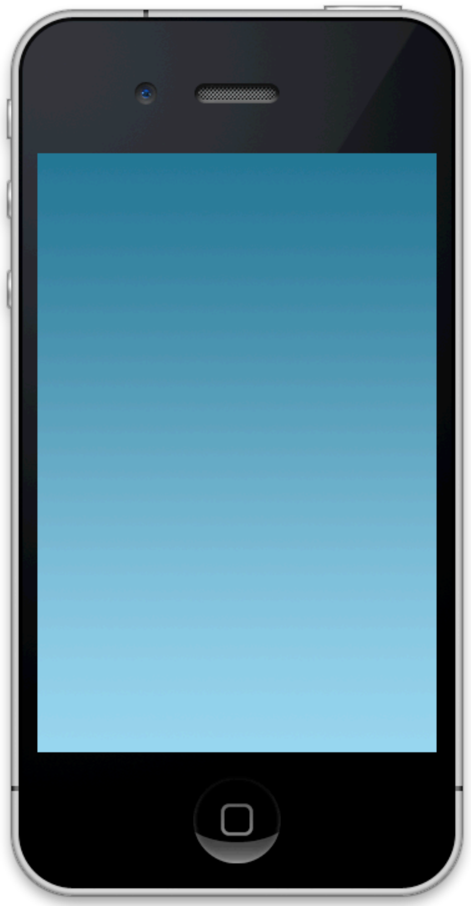
CoreGraphics
Color Fill

```
- (void)drawRect:(CGRect)rect
{
   [super drawRect:rect];
   CGContextRef context = UIGraphicsGetCurrentContext();
   CGFloat redColor[4] = {1.0f,0.0f,0.0f,1.0f};
   CGContextSetFillColor(context, redColor);
   CGContextFillRect(context, rect);
}
```



Again: Only UIView (sub)classes have/get the drawRect: method (inherited/ called).

CoreGraphics
Gradients



CoreGraphics
Gradients

```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];
    CGContextRef context = UIGraphicsGetCurrentContext();
    CGColorSpaceRef colorSpace = CGColorSpaceCreateDeviceRGB();
    CGFloat colors[8] = {33.0f/255.0f, 102.0f/255.0f, 133.0f/255.0f, 1.0f,
                        138.0f/255.0f, 206.0f/255.0f, 236.0f/255.0f, 1.0f};
    CGFloat locations[2] = {0.0f, 1.0f};

    CGGradientRef gradient = CGGradientCreateWithColorComponents(
                           colorSpace, colors, locations, 2);

    CGContextDrawLinearGradient(context, gradient,
                           CGPointMake(0.0f, 0.0f), CGPointMake(0.0f, 320.0f),
                           kCGGradientDrawsAfterEndLocation|
                           kCGGradientDrawsBeforeStartLocation);

    CGGradientRelease(gradient);
    CGColorSpaceRelease(colorSpace);
}
```

CoreGraphics

Gradients

Grab the current drawing context.

```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];
    CGContextRef context = UIGraphicsGetCurrentContext();
    CGColorSpaceRef colorSpace = CGColorSpaceCreateDeviceRGB();
    CGFloat colors[8] = {33.0f/255.0f, 102.0f/255.0f, 133.0f/255.0f, 1.0f,
                         138.0f/255.0f, 206.0f/255.0f, 236.0f/255.0f, 1.0f};
    CGFloat locations[2] = {0.0f, 1.0f};

    CGGradientRef gradient = CGGradientCreateWithColorComponents(
                                 colorSpace, colors, locations, 2);

    CGContextDrawLinearGradient(context, gradient,
                                 CGPointMake(0.0f, 0.0f), CGPointMake(0.0f, 320.0f),
                                 kCGGradientDrawsAfterEndLocation|
                                 kCGGradientDrawsBeforeStartLocation);

    CGGradientRelease(gradient);
    CGColorSpaceRelease(colorSpace);
}
```

CoreGraphics

Gradients

```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];
    CGContextRef context = UIGraphicsGetCurrentContext();
    CGColorSpaceRef colorSpace = CGColorSpaceCreateDeviceRGB();
    CGFloat colors[8] = {33.0f/255.0f, 102.0f/255.0f, 133.0f/255.0f, 1.0f,
                         138.0f/255.0f, 206.0f/255.0f, 236.0f/255.0f, 1.0f};
    CGFloat locations[2] = {0.0f, 1.0f};

    CGGradientRef gradient = CGGradientCreateWithColorComponents(
                               colorSpace, colors, locations, 2);

    CGContextDrawLinearGradient(context, gradient,
                               CGPointMake(0.0f, 0.0f), CGPointMake(0.0f, 320.0f),
                               kCGGradientDrawsAfterEndLocation|
                               kCGGradientDrawsBeforeStartLocation);

    CGGradientRelease(gradient);
    CGColorSpaceRelease(colorSpace);
}
```

Get device color space.

CoreGraphics

Gradients

```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];
    CGContextRef context = UIGraphicsGetCurrentContext();
    CGColorSpaceRef colorSpace = CGColorSpaceCreateDeviceRGB();
    CGFloat colors[8] = {33.0f/255.0f, 102.0f/255.0f, 133.0f/255.0f, 1.0f,
                           138.0f/255.0f, 206.0f/255.0f, 236.0f/255.0f, 1.0f};
    CGFloat locations[2] = {0.0f, 1.0f};

    CGGradientRef gradient = CGGradientCreateWithColorComponents(
                               colorSpace, colors, locations, 2);

    CGContextDrawLinearGradient(context, gradient,
                               CGPointMake(0.0f, 0.0f), CGPointMake(0.0f, 320.0f),
                               kCGGradientDrawsAfterEndLocation|
                               kCGGradientDrawsBeforeStartLocation);

    CGGradientRelease(gradient);
    CGColorSpaceRelease(colorSpace);
}
```

Define two colors (RGBA).

CoreGraphics
Gradients

```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];
    CGContextRef context = UIGraphicsGetCurrentContext();
    CGColorSpaceRef colorSpace = CGColorSpaceCreateDeviceRGB();
    CGFloat colors[8] = {33.0f/255.0f, 102.0f/255.0f, 133.0f/255.0f, 1.0f,
                         138.0f/255.0f, 206.0f/255.0f, 236.0f/255.0f, 1.0f};
    CGFloat locations[2] = {0.0f, 1.0f};

    CGGradientRef gradient = CGGradientCreateWithColorComponents(
                              colorSpace, colors, locations, 2);

    CGContextDrawLinearGradient(context, gradient,
                              CGPointMake(0.0f, 0.0f), CGPointMake(0.0f, 320.0f),
                              kCGGradientDrawsAfterEndLocation|
                              kCGGradientDrawsBeforeStartLocation);

    CGGradientRelease(gradient);
    CGColorSpaceRelease(colorSpace);
}
```

Define gradient locations (unit size).

CoreGraphics
Gradients

```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];
    CGContextRef context = UIGraphicsGetCurrentContext();
    CGColorSpaceRef colorSpace = CGColorSpaceCreateDeviceRGB();
    CGFloat colors[8] = {33.0f/255.0f, 102.0f/255.0f, 133.0f/255.0f, 1.0f,
                         138.0f/255.0f, 206.0f/255.0f, 236.0f/255.0f, 1.0f};
    CGFloat locations[2] = {0.0f, 1.0f};

    CGGradientRef gradient = CGGradientCreateWithColorComponents(
                                  colorSpace, colors, locations, 2);

    CGContextDrawLinearGradient(context, gradient,
                                  CGPointMake(0.0f, 0.0f), CGPointMake(0.0f, 320.0f),
                                  kCGGradientDrawsAfterEndLocation|
                                  kCGGradientDrawsBeforeStartLocation);

    CGGradientRelease(gradient);
    CGColorSpaceRelease(colorSpace);
}
```

Create a gradient with the predefined parameters.

CoreGraphics
Gradients

```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];
    CGContextRef context = UIGraphicsGetCurrentContext();
    CGColorSpaceRef colorSpace = CGColorSpaceCreateDeviceRGB();
    CGFloat colors[8] = {33.0f/255.0f, 102.0f/255.0f, 133.0f/255.0f, 1.0f,
                         138.0f/255.0f, 206.0f/255.0f, 236.0f/255.0f, 1.0f};
    CGFloat locations[2] = {0.0f, 1.0f};

    CGGradientRef gradient = CGGradientCreateWithColorComponents(
                                colorSpace, colors, locations, 2);

    CGContextDrawLinearGradient(context, gradient,
                                CGPointMake(0.0f, 0.0f), CGPointMake(0.0f, 320.0f),
                                kCGGradientDrawsAfterEndLocation|
                                kCGGradientDrawsBeforeStartLocation);

    CGGradientRelease(gradient);
    CGColorSpaceRelease(colorSpace);
}
```

Draw a linear gradient in the current context with a starting and end point. Extend the filling upwards and downwards.

CoreGraphics

Gradients

```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];
    CGContextRef context = UIGraphicsGetCurrentContext();
    CGColorSpaceRef colorSpace = CGColorSpaceCreateDeviceRGB();
    CGFloat colors[8] = {33.0f/255.0f, 102.0f/255.0f, 133.0f/255.0f, 1.0f,
                         138.0f/255.0f, 206.0f/255.0f, 236.0f/255.0f, 1.0f};
    CGFloat locations[2] = {0.0f, 1.0f};

    CGGradientRef gradient = CGGradientCreateWithColorComponents(
                               colorSpace, colors, locations, 2);

    CGContextDrawLinearGradient(context, gradient,
                               CGPointMake(0.0f, 0.0f), CGPointMake(0.0f, 320.0f),
                               kCGGradientDrawsAfterEndLocation|
                               kCGGradientDrawsBeforeStartLocation);

    CGGradientRelease(gradient);
    CGColorSpaceRelease(colorSpace);
}
```

There is **no ARC** for CF Objects!
You have to release memory you allocated manually!
(Keywords: **Create** & **Copy**)

CoreGraphics

Points and Paths

```
- (void)drawRect:(CGRect)rect
{
   [super drawRect:rect];
   UIBezierPath *bezierPath = [UIBezierPath bezierPath];
   [bezierPath moveToPoint:CGPointMake(1.0f, 2.0f)];
   [bezierPath addLineToPoint:CGPointMake(5.0f, 2.0f)];
   [bezierPath setLineWidth:1.0f];
   [bezierPath stroke];
}
```

CoreGraphics
Points and Paths

```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];
    UIBezierPath *bezierPath = [UIBezierPath bezierPath];
    [bezierPath moveToPoint:CGPointMake(1.0f, 2.0f)];
    [bezierPath addLineToPoint:CGPointMake(5.0f, 2.0f)];
    [bezierPath setLineWidth:1.0f];
    [bezierPath stroke];
}
```

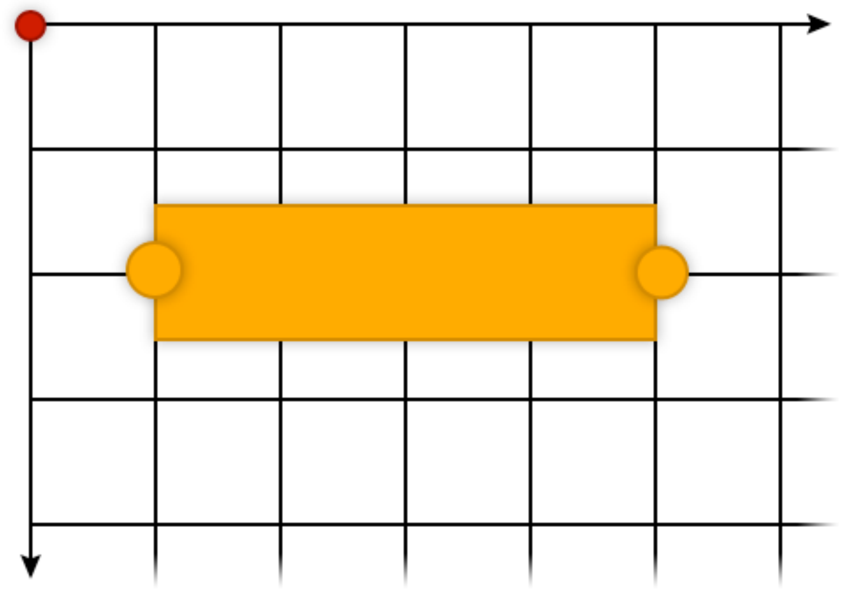
Note: This is UIKit!

CoreGraphics

Points and Paths

```
- (void)drawRect:(CGRect)rect
{
   [super drawRect:rect];
   UIBezierPath *bezierPath = [UIBezierPath bezierPath];
   [bezierPath moveToPoint:CGPointMake(1.0f, 2.0f)];
   [bezierPath addLineToPoint:CGPointMake(5.0f, 2.0f)];
   [bezierPath setLineWidth:1.0f];
   [bezierPath stroke];
}
```

Note: This is UIKit!



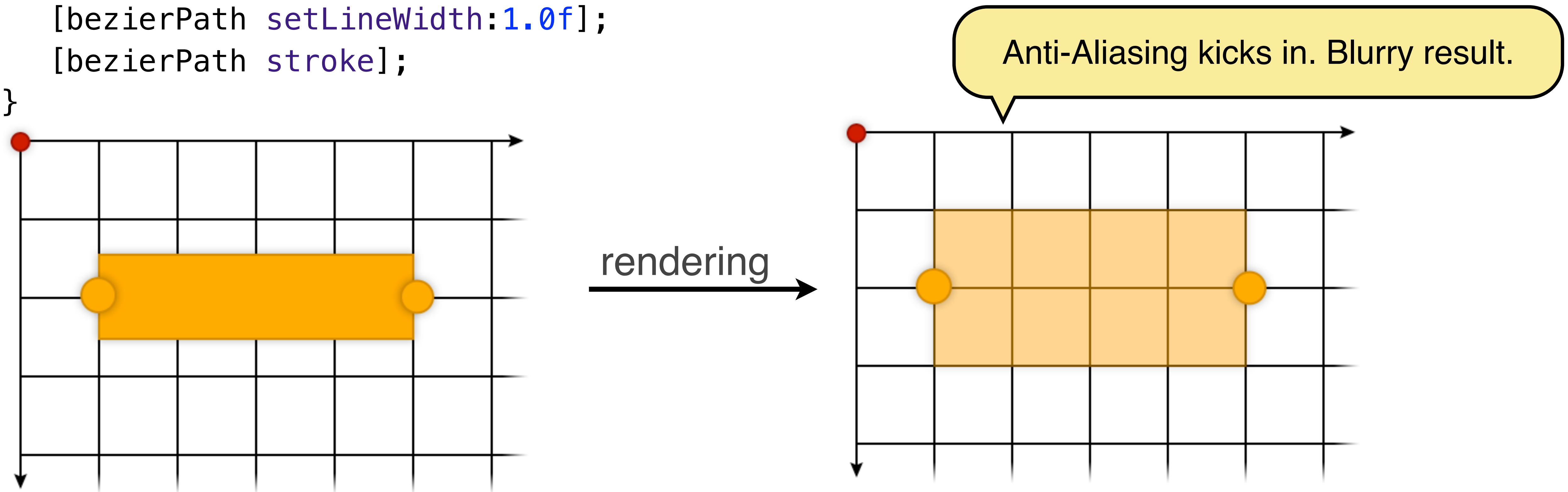
Points are not pixels!

A point is defined at an intersection.

CoreGraphics

Points and Paths

```
- (void)drawRect:(CGRect)rect
{
   [super drawRect:rect];
   UIBezierPath *bezierPath = [UIBezierPath bezierPath];
   [bezierPath moveToPoint:CGPointMake(1.0f, 2.0f)];
   [bezierPath addLineToPoint:CGPointMake(5.0f, 2.0f)];
   [bezierPath setLineWidth:1.0f];
   [bezierPath stroke];
}
```



Geometrical (Points)

Rendered (Pixels) on non-retina displays

CoreGraphics

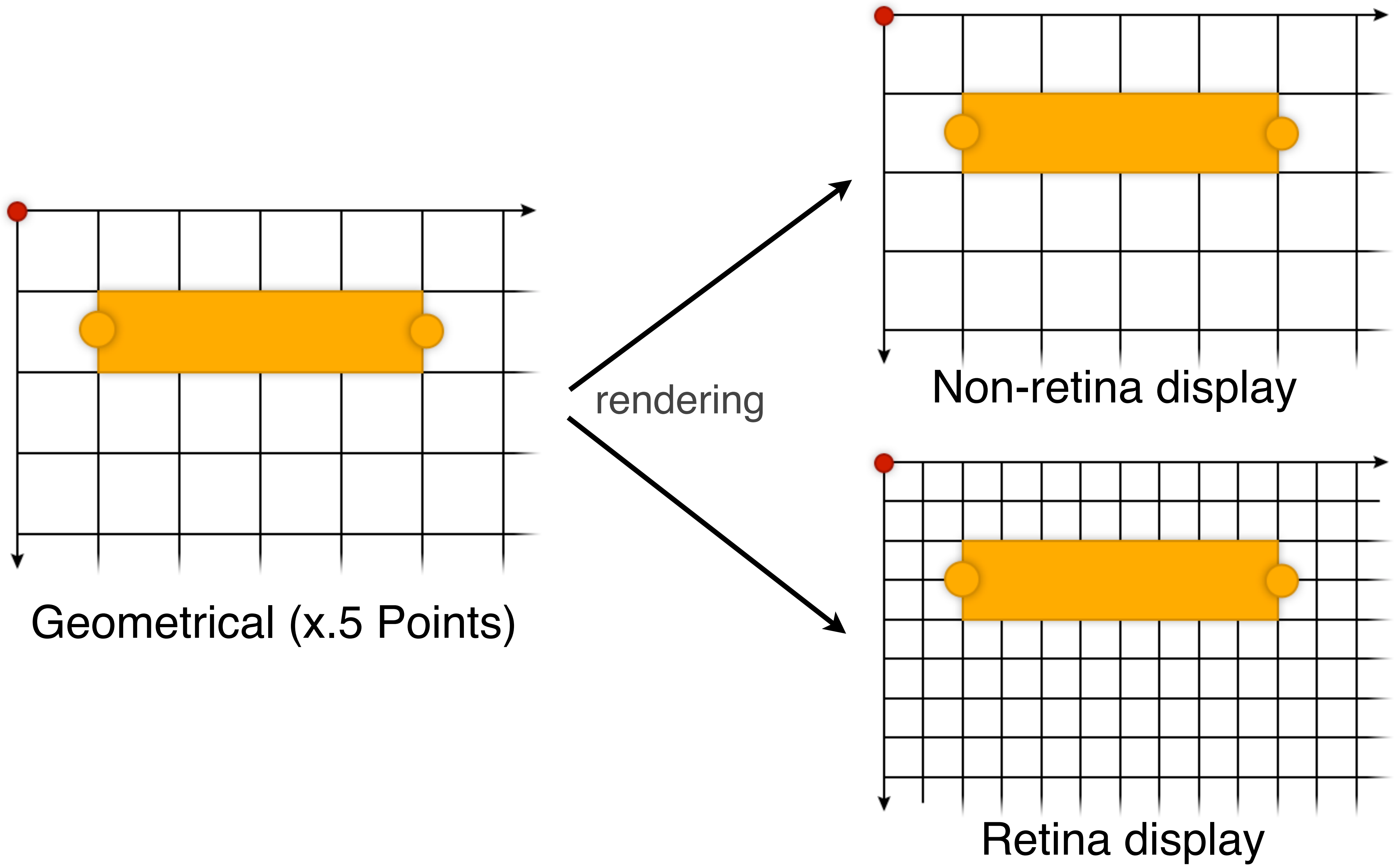
Pixel Precision

Solution: x.5 point offsets!

```
- (void)drawRect:(CGRect)rect
{
   [super drawRect:rect];
   UIBezierPath *bezierPath = [UIBezierPath bezierPath];
   [bezierPath moveToPoint:CGPointMake(1.0f, 1.5f)];
   [bezierPath addLineToPoint:CGPointMake(5.0f, 1.5f)];
   [bezierPath setLineWidth:1.0f];
   [bezierPath stroke];
}
```

CoreGraphics

Pixel Precision



Mission One

Turn the stroked line into a stroked triangle.

Mission One - Solution

"Turn the stroked line into a stroked triangle."

```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];

    // From now on we'll be drawing with red paint!
    [[UIColor redColor] setStroke];

    // Creating a triangle
    UIBezierPath *path = [UIBezierPath bezierPath];
    path.lineWidth = 5.0f;
    [path moveToPoint:CGPointMake(50.0f, 50.0f)];
    [path addLineToPoint:CGPointMake(400.0f, 400.0f)];
    [path addLineToPoint:CGPointMake(400.0f, 50.0f)];
    [path closePath];
    [path stroke];
}
```

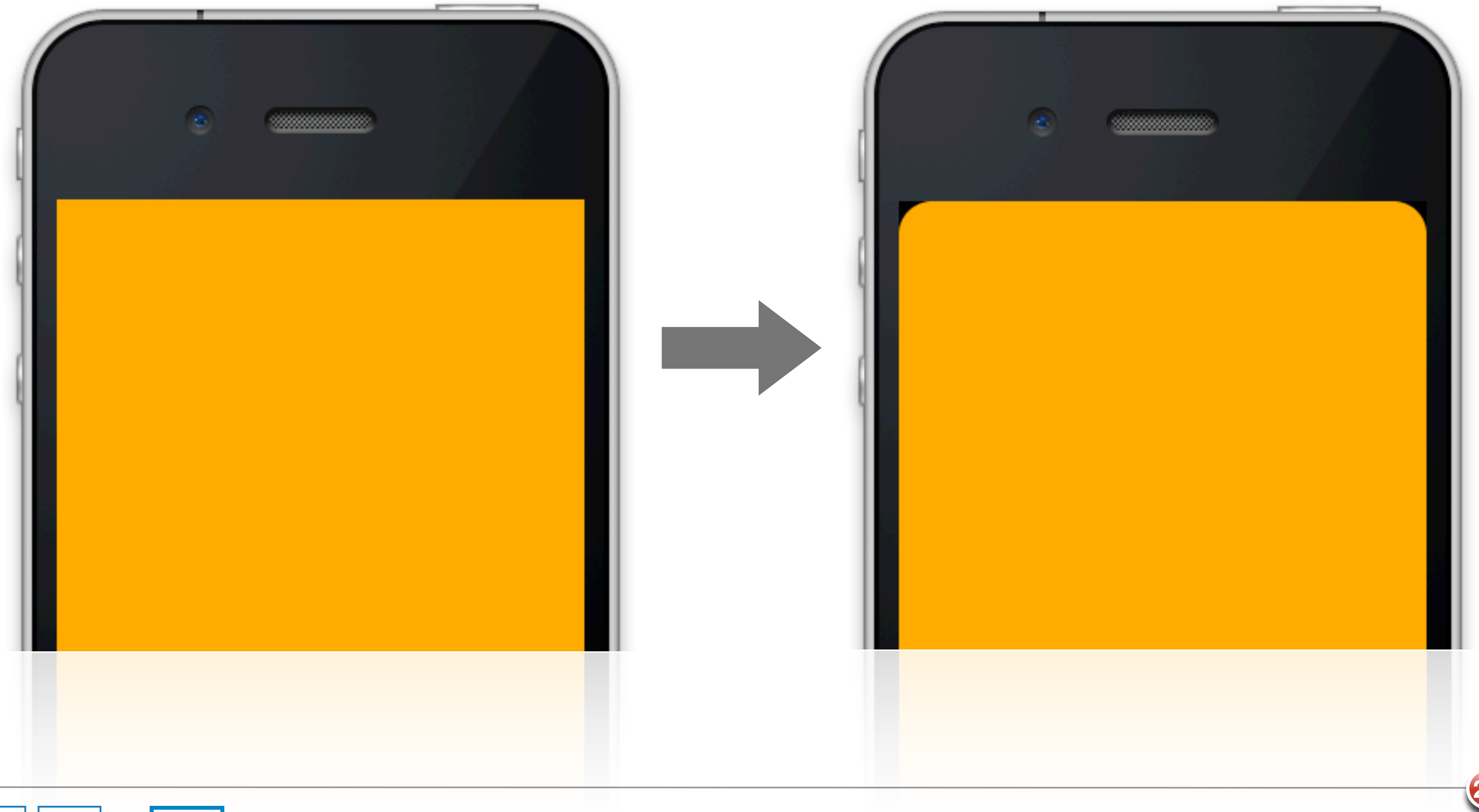
Mission One - Solution

“Turn the stroked line into a stroked triangle.”

```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];

    // From now on we'll be drawing with red paint!
    [[UIColor redColor] setStroke];

    // Creating a triangle
    UIBezierPath *path = [UIBezierPath bezierPath];
    path.lineWidth = 5.0f;
    [path moveToPoint:CGPointMake(50.0f, 50.0f)];
    [path addLineToPoint:CGPointMake(400.0f, 400.0f)];
    [path addLineToPoint:CGPointMake(400.0f, 50.0f)];
    [path closePath];
    [path stroke];
}
```

You needed to add this.

CoreGraphics
Clipping



CoreGraphics

Clipping

```
- (void)drawRect:(CGRect)rect
{
   [super drawRect:rect];

   CGContextRef context = UIGraphicsGetCurrentContext();
   CGFloat yellowColor[4] = {1.0f, 1.0f, 0.0f, 1.0f};
   CGContextSetFillColor(context, yellowColor);

   UIBezierPath *clipPath = [UIBezierPath
                             bezierPathWithRoundedRect:rect
                             cornerRadius:15.0f];
   CGContextAddPath(context, clipPath.CGPath);
   CGContextClip(context);

   CGContextFillRect(context, rect);
}
```

CoreGraphics

Clipping

```
- (void)drawRect:(CGRect)rect
{
  [super drawRect:rect];

  CGContextRef context = UIGraphicsGetCurrentContext();
  CGFloat yellowColor[4] = {1.0f, 1.0f, 0.0f, 1.0f};
  CGContextSetFillColor(context, yellowColor);

  UIBezierPath *clipPath = [UIBezierPath
                             bezierPathWithRoundedRect:rect
                             cornerRadius:15.0f];
  CGContextAddPath(context, clipPath.CGPath);
  CGContextClip(context);

  CGContextFillRect(context, rect);
}
```

Grab the current drawing context.

CoreGraphics

Clipping

```
- (void)drawRect:(CGRect)rect
{
  [super drawRect:rect];

  CGContextRef context = UIGraphicsGetCurrentContext();
  CGFloat yellowColor[4] = {1.0f, 1.0f, 0.0f, 1.0f};
  CGContextSetFillColor(context, yellowColor);

  UIBezierPath *clipPath = [UIBezierPath
                             bezierPathWithRoundedRect:rect
                             cornerRadius:15.0f];
  CGContextAddPath(context, clipPath.CGPath);
  CGContextClip(context);

  CGContextFillRect(context, rect);
}
```

Define a color.

CoreGraphics
Clipping

```
- (void)drawRect:(CGRect)rect
{
  [super drawRect:rect];

  CGContextRef context = UIGraphicsGetCurrentContext();
  CGFloat yellowColor[4] = {1.0f, 1.0f, 0.0f, 1.0f};
  CGContextSetFillColor(context, yellowColor);

  UIBezierPath *clipPath = [UIBezierPath
                           bezierPathWithRoundedRect:rect
                           cornerRadius:15.0f];
  CGContextAddPath(context, clipPath.CGPath);
  CGContextClip(context);

  CGContextFillRect(context, rect);
}
```

Set the color as the filling color.

CoreGraphics

Clipping

```
- (void)drawRect:(CGRect)rect
{
   [super drawRect:rect];

   CGContextRef context = UIGraphicsGetCurrentContext();
   CGFloat yellowColor[4] = {1.0f, 1.0f, 0.0f, 1.0f};
   CGContextSetFillColor(context, yellowColor);

   UIBezierPath *clipPath = [UIBezierPath
                              bezierPathWithRoundedRect:rect
                              cornerRadius:15.0f];
   CGContextAddPath(context, clipPath.CGPath);
   CGContextClip(context);

   CGContextFillRect(context, rect);
}
```

Build a bezier curve out of the our drawing area that has rounded corners.

CoreGraphics

Clipping

```
- (void)drawRect:(CGRect)rect
{
  [super drawRect:rect];

  CGContextRef context = UIGraphicsGetCurrentContext();
  CGFloat yellowColor[4] = {1.0f, 1.0f, 0.0f, 1.0f};
  CGContextSetFillColor(context, yellowColor);

  UIBezierPath *clipPath = [UIBezierPath
                             bezierPathWithRoundedRect:rect
                             cornerRadius:15.0f];
  CGContextAddPath(context, clipPath.CGPath);
  CGContextClip(context);

  CGContextFillRect(context, rect);
}
```

Add path to context and clip with it.

CoreGraphics

Clipping

```
- (void)drawRect:(CGRect)rect
{
   [super drawRect:rect];

   CGContextRef context = UIGraphicsGetCurrentContext();
   CGFloat yellowColor[4] = {1.0f, 1.0f, 0.0f, 1.0f};
   CGContextSetFillColor(context, yellowColor);

   UIBezierPath *clipPath = [UIBezierPath
                             bezierPathWithRoundedRect:rect
                             cornerRadius:15.0f];
   CGContextAddPath(context, clipPath.CGPath);
   CGContextClip(context);

   CGContextFillRect(context, rect);
}
```

Fill the rect with the predefined color. Drawing outside the clipping area will be ignored.

CoreGraphics
Clipping

“What if I’m done clipping and want to go back?”

CoreGraphics

Clipping

```
- (void)drawRect:(CGRect)rect
{
   [super drawRect:rect];

   CGContextRef context = UIGraphicsGetCurrentContext();

   CGContextSaveGState(context);

   // Do stuff.

   CGContextRestoreGState(context);
}
```

Save and revert the context sate. - Works like a stack.

“What if I’m done clipping and want to go back?”

CoreGraphics
Even-Odd Clipping

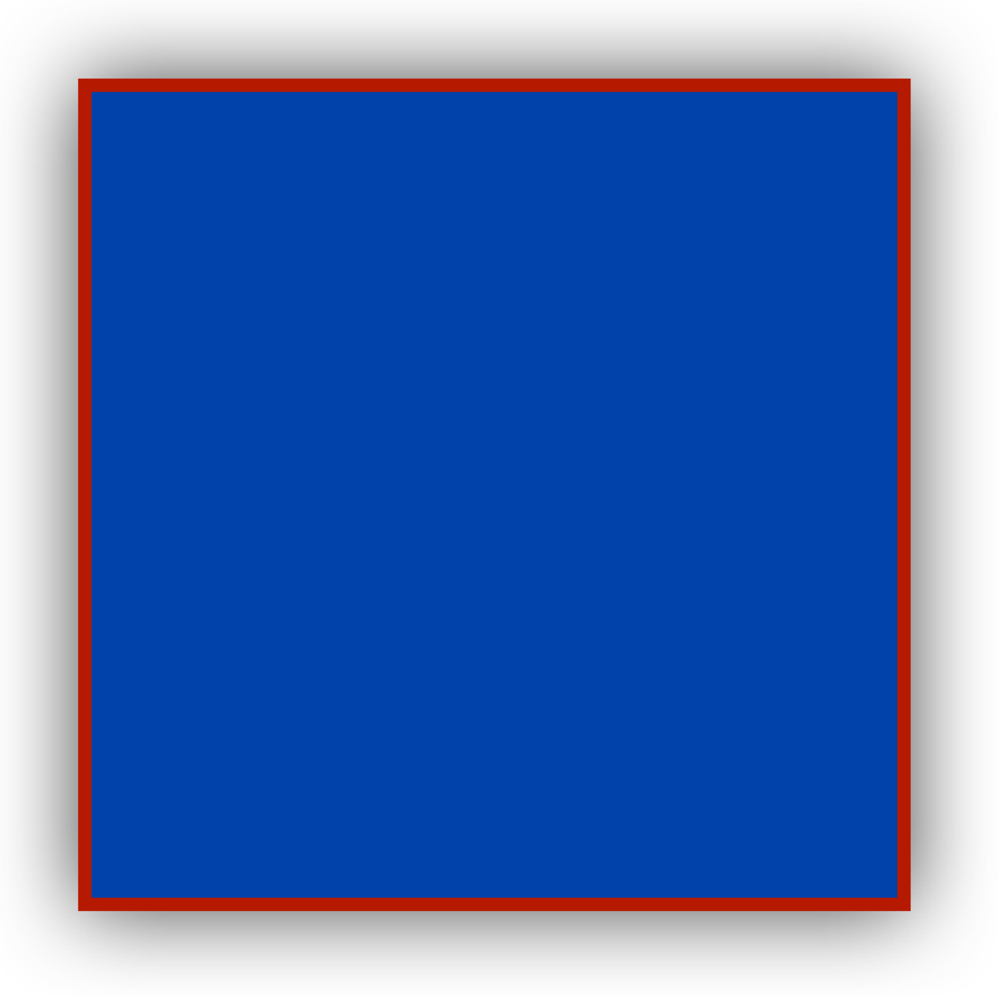
```
- (void)drawRect:(CGRect)rect
{
   // [...]

   CGPathRef outerPath = CGPathCreateWithRect(outerRect, NULL);
   CGPathRef innerPath = CGPathCreateWithRect(innerRect, NULL);

   CGContextAddPath(context, outerPath);
   CGContextAddPath(context, innerPath);

   CGContextEOClip(context);

   CGContextFillRect(context, rect);

   // [...]
}
```

"When you use this function instead of `CGContextClip`, *subsequent nested regions [...] continue to toggle clipping on and off."*

(Source: http://cocoawithlove.com/2009/09/creating-alpha-masks-from-text-on.html)

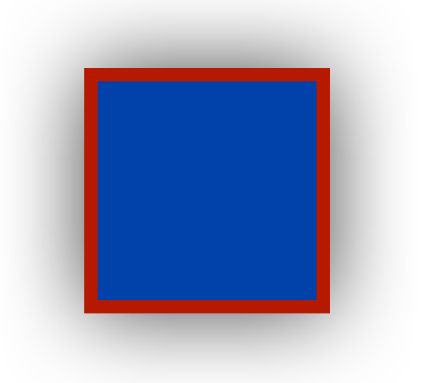
Mission Two

“Recreate the image below.”



Mission Two

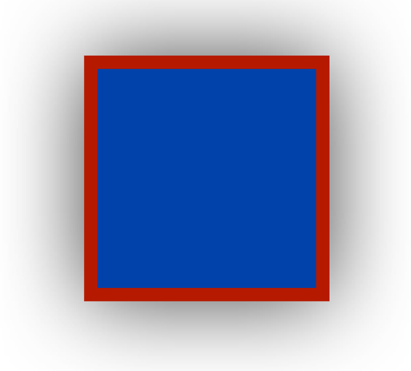
“Recreate the image below.”



```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];

    CGRect box = CGRectMake(200.0f, 200.0f, 400.0f, 400.0f);

    CGContextRef context = UIGraphicsGetCurrentContext();

    // Define some colors
    CGFloat redColor[4] = {1.0f, 0.0f, 0.0f, 1.0f};
    CGFloat blueColor[4] = {0.1f, 0.1f, 0.5f, 1.0f};

    // TODO 1 + 2 + 3;
    CGContextSetFillColor(context, blueColor);
    CGContextSetStrokeColor(context, redColor);
    CGContextSetLineWidth(context, 5.0f);

    CGContextSetShadowWithColor(context, CGSizeZero, 20.0f, [UIColor blackColor].CGColor);

    CGContextFillRect(context, box);
    CGContextStrokeRect(context, box);
}
```

Mission Two

“Recreate the image below.”

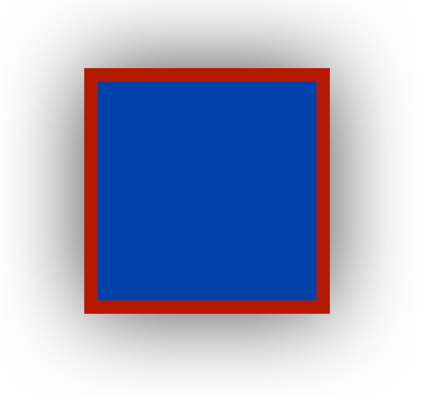


```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];

    CGRect box = CGRectMake(200.0f, 200.0f, 400.0f, 400.0f);

    CGContextRef context = UIGraphicsGetCurrentContext();

    // Define some colors
    CGFloat redColor[4] = {1.0f, 0.0f, 0.0f, 1.0f};
    CGFloat blueColor[4] = {0.1f, 0.1f, 0.5f, 1.0f};

    // TODO 1 + 2 + 3;
    CGContextSetFillColor(context, blueColor);
    CGContextSetStrokeColor(context, redColor);
    CGContextSetLineWidth(context, 5.0f);

    CGContextSetShadowWithColor(context, CGSizeZero, 20.0f, [UIColor blackColor].CGColor);

    CGContextFillRect(context, box);
    CGContextStrokeRect(context, box);
}
```

Who can tell me whats “not-so-cool” about this answer?

Mission Two

"Recreate the image below."



```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];

    CGRect box = CGRectMake(200.0f, 200.0f, 400.0f, 400.0f);

    CGContextRef context = UIGraphicsGetCurrentContext();

    // Define some colors
    CGFloat redColor[4] = {1.0f, 0.0f, 0.0f, 1.0f};
    CGFloat blueColor[4] = {0.1f, 0.1f, 0.5f, 1.0f};

    // TODO 1 + 2 + 3;
    CGContextSetFillColor(context, blueColor);
    CGContextSetStrokeColor(context, redColor);
    CGContextSetLineWidth(context, 5.0f);

    CGContextSetShadowWithColor(context, CGSizeZero, 20.0f, [UIColor blackColor].CGColor);

    CGContextFillRect(context, box);
    CGContextStrokeRect(context, box);
}
```

Who can tell me whats "not-so-cool" about this answer?

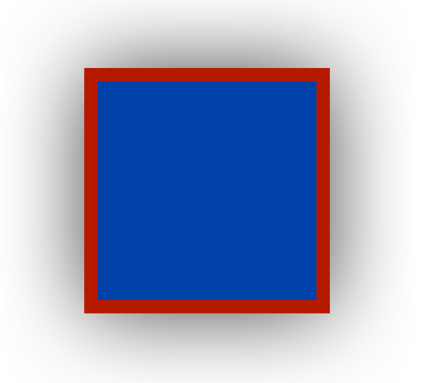
Notice how we're drawing the shadow twice?

Mission 2.5

Find a way to fix the shadow drawing.

Mission Two

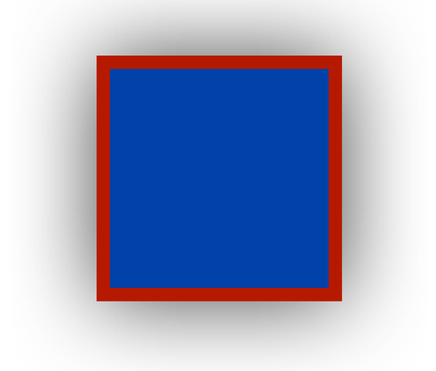
“Recreate the image below.”



```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];

    CGRect box = CGRectMake(200.0f, 200.0f, 400.0f, 400.0f);

    CGContextRef context = UIGraphicsGetCurrentContext();

    // Define some colors
    CGFloat redColor[4] = {1.0f, 0.0f, 0.0f, 1.0f};
    CGFloat blueColor[4] = {0.1f, 0.1f, 0.5f, 1.0f};

    // TODO 1 + 2 + 3;
    CGContextSetFillColor(context, blueColor);
    CGContextSetStrokeColor(context, redColor);
    CGContextSetLineWidth(context, 5.0f);

    CGContextSaveGState(context);
    CGContextSetShadowWithColor(context, CGSizeZero, 20.0f, [UIColor blackColor].CGColor);
    CGContextFillRect(context, box);
    CGContextRestoreGState(context);

    CGContextStrokeRect(context, box);
}
```

Mission Two

“Recreate the image below.”



```
- (void)drawRect:(CGRect)rect
{
    [super drawRect:rect];

    CGRect box = CGRectMake(200.0f, 200.0f, 400.0f, 400.0f);

    CGContextRef context = UIGraphicsGetCurrentContext();

    // Define some colors
    CGFloat redColor[4] = {1.0f, 0.0f, 0.0f, 1.0f};
    CGFloat blueColor[4] = {0.1f, 0.1f, 0.5f, 1.0f};

    // TODO 1 + 2 + 3;
    CGContextSetFillColor(context, blueColor);
    CGContextSetStrokeColor(context, redColor);
    CGContextSetLineWidth(context, 5.0f);

    CGContextSaveGState(context);
    CGContextSetShadowWithColor(context, CGSizeZero, 20.0f, [UIColor blackColor].CGColor);
    CGContextFillRect(context, box);
    CGContextRestoreGState(context);

    CGContextStrokeRect(context, box);
}
```

Save and restore the graphics state!

CoreGraphics
Outlook

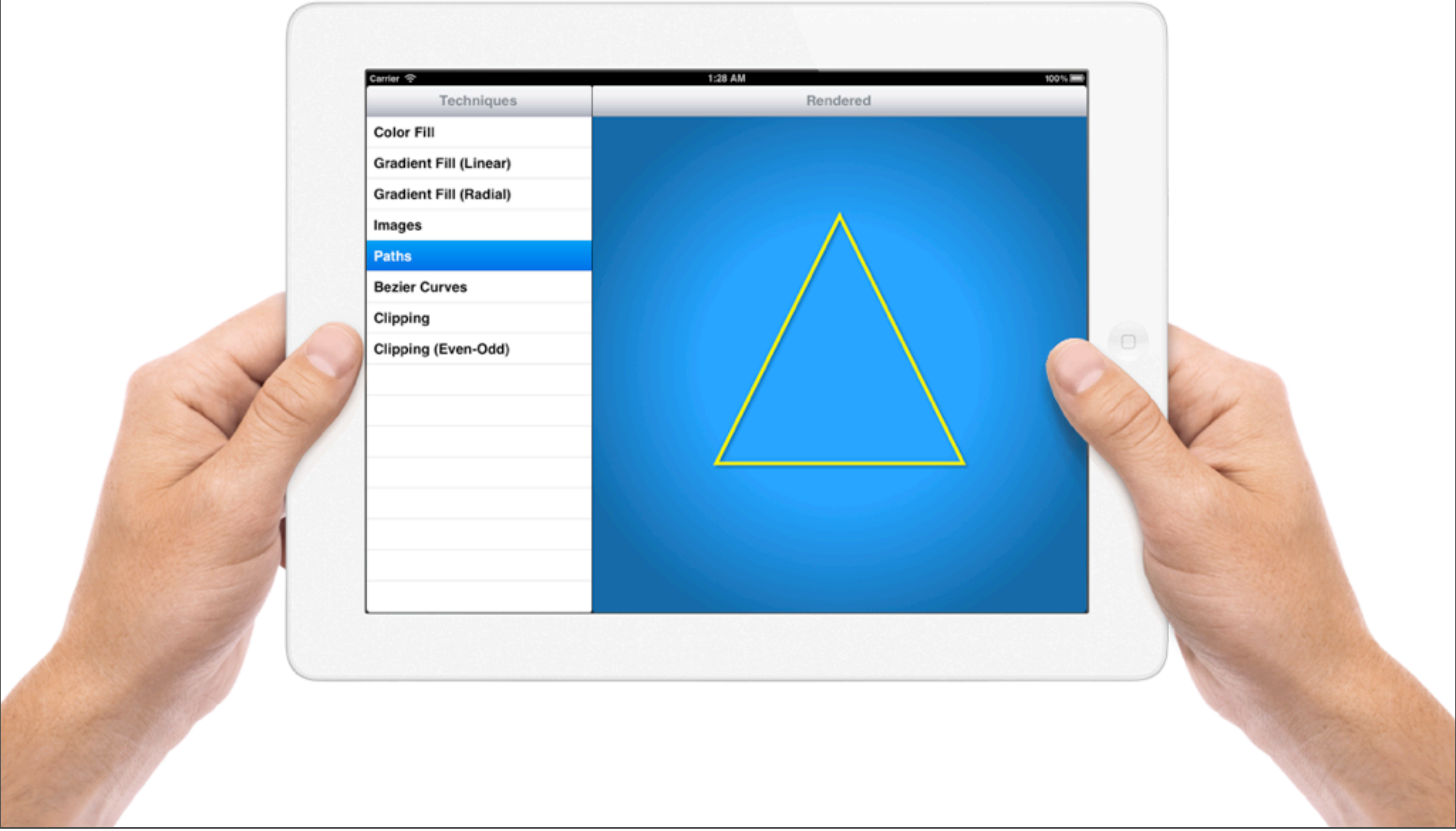
- These were only the very basics.
- CoreGraphics is inherently more powerful.
- ...even more so together with CoreImage.

Reference Project (2)

"CoreTechniques"

Reference Project (2)

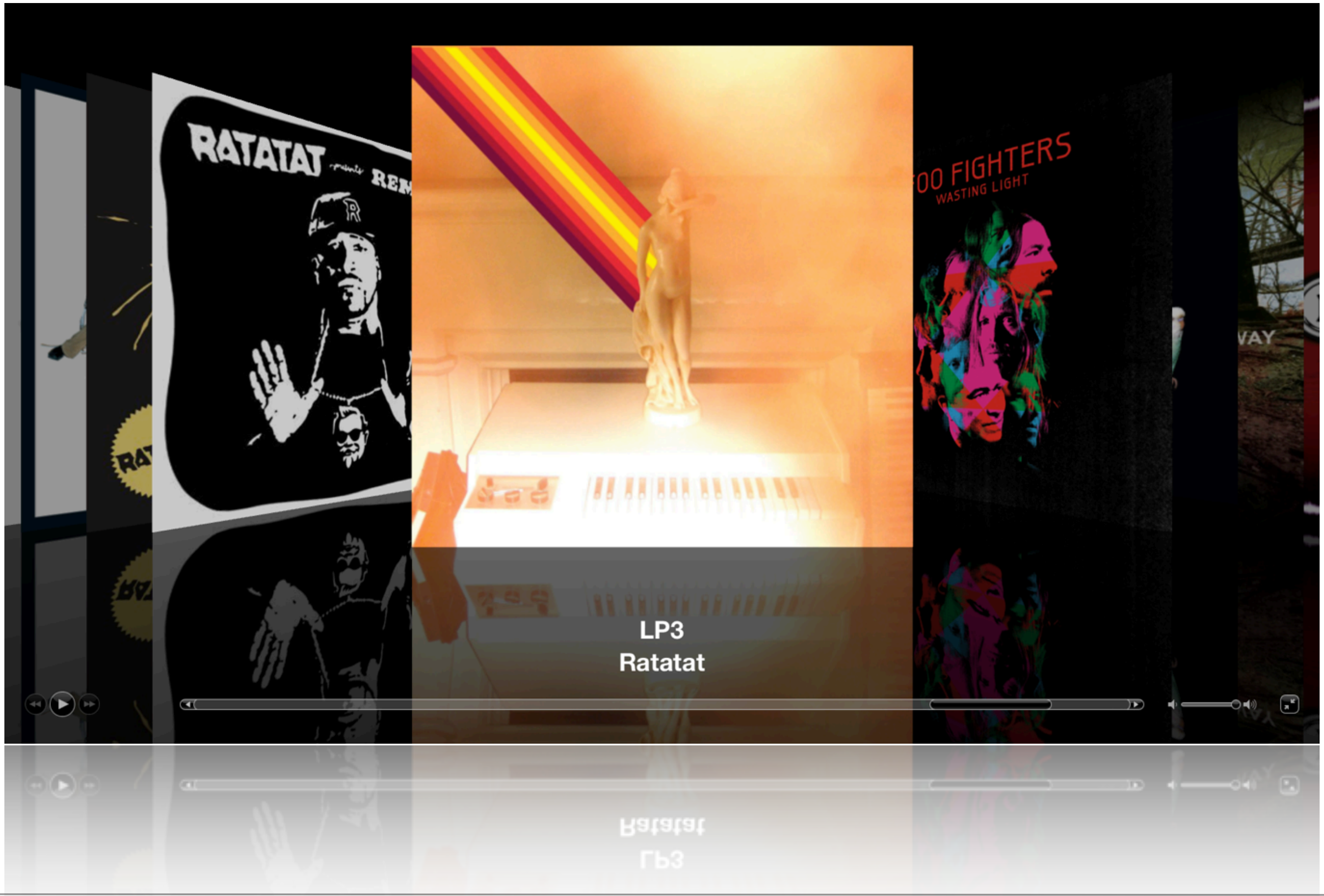
“CoreTechniques”



CoreAnimation

CoreAnimation

CoverFlow



CoreAnimation
QuickTime

CoreAnimation
Tapbots (Various)

CoreAnimation

Introduction at WWDC 2007

CoreAnimation

...should I use it?

Why...?

Good looks <u>ARE</u> important. **Visual cues enhance UX!**

When...?

Whenever UIKit does not suffice.

Where...?

Whenever you're working with the layer of a UIView.

UIKit (Animations)

Sometimes it’s easier - Simple translation.

“When a tap is recognized, animate a subview to the right by 200px”

UIKit (Animations)

Sometimes it’s easier - Simple translation.

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
  [UIView animateWithDuration:3.0f animations:^
  {
     self.subV.frame = CGRectOffset(self.subV.frame, -200.0f, 0.0f);
  }];
}
```

“When a tap is recognized, animate a subview to the right by 200px”

UIKit (Animations)

Sometimes it’s easier - Simple translation.

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
    [UIView animateWithDuration:3.0f animations:^
    {
        self.subV.frame = CGRectOffset(self.subV.frame, -200.0f, 0.0f);
    }];
}
```

Objective-C “Block” aka “Anonymous Function”.

“When a tap is recognized, animate a subview to the right by 200px”

UIKit (Animations)

Sometimes it’s easier - Simple translation.

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
   [UIView animateWithDuration:3.0f animations:^
   {
       self.subV.frame = CGRectOffset(self.subV.frame, -200.0f, 0.0f);
   }];
}
```

Describe the state you want your object to end up in.

“When a tap is recognized, animate a subview to the right by 200px”

UIKit (Animations)

Sometimes it’s easier - Simple translation.

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
  [UIView animateWithDuration:3.0f animations:^
  {
     self.subV.frame = CGRectOffset(self.subV.frame, -200.0f, 0.0f);
  }];
}
```

“When a tap is recognized, animate a subview to the right by 200px”

UIKit (Animations)

Sometimes it's easier - Simple translation.

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
   [UIView animateWithDuration:3.0f animations:^
   {
      self.subV.frame = CGRectOffset(self.subV.frame, -200.0f, 0.0f);
   }];
}
```

"When a tap is recognized, animate a subview to the right by 200px"

UIKit (Animations)

Sometimes it’s easier - Options & Completion Handler.

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
   [UIView animateWithDuration:2.0f delay:10.0f
                          options:UIViewAnimationCurveEaseInOut
    animations:^
   {
       // Do this.
   }
   completion:^(BOOL finished)
   {
       // And do this when you’re finished.
   }];
}
```

UIKit (Animations)

Sometimes it's easier - Options & Completion Handler.

Additional delay parameter.

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
   [UIView animateWithDuration:2.0f delay:10.0f
                               options:UIViewAnimationCurveEaseInOut
    animations:^
   {
       // Do this.
   }
   completion:^(BOOL finished)
   {
       // And do this when you're finished.
   }];
}
```

UIKit (Animations)

Sometimes it’s easier - Options & Completion Handler.

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
   [UIView animateWithDuration:2.0f delay:10.0f
                           options:UIViewAnimationCurveEaseInOut
    animations:^
   {
       // Do this.
   }
   completion:^(BOOL finished)
   {
       // And do this when you’re finished.
   }];
}
```

Constant value that defines the animation curve. Can be linear, ease in, ...

UIKit (Animations)

Sometimes it’s easier - Options & Completion Handler.

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
  [UIView animateWithDuration:2.0f delay:10.0f
                             options:UIViewAnimationCurveEaseInOut
   animations:^
  {
      // Do this.
  }
  completion:^(BOOL finished)
  {
      // And do this when you’re finished.
  }];
}
```

Completion handler.

UIKit (Animations)

Sometimes it’s easier - Nesting.

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
  [UIView animateWithDuration:2.0f delay:10.0f
                          options:UIViewAnimationCurveEaseInOut
   animations:^
  {
      self.subV.alpha = 1.0f;
  }
  completion:^(BOOL finished)
  {
      [UIView animateWithDuration:2.0f animations:^
      {
         self.subV.frame = SOME_RECT;
         self.subV.alpha = 0.0f;
      }];
  }];
}
```

UIKit (Animations)

Sometimes it’s easier - Nesting.

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
   [UIView animateWithDuration:2.0f delay:10.0f
                        options:UIViewAnimationCurveEaseInOut
    animations:^
   {
       self.subV.alpha = 1.0f;
   }
   completion:^(BOOL finished)
   {
       [UIView animateWithDuration:2.0f animations:^
       {
           self.subV.frame = SOME_RECT;
           self.subV.alpha = 0.0f;
       }];
   }];
}
```

Completion handler containing another animation enables primitive chaining.

Mission Three

Make the red box disappear (fade) with a 1 second duration.

Mission Three

“Make the red box disappear (fade) with 1 seconds duration.”

```
- (IBAction)toggleBoxVisibility:(id)sender
{
    CGFloat destinationAlpha = 0.0f;

    if (self.redBox.alpha == 0.0f)
    {
       destinationAlpha = 1.0f;
    }
    else
    {
       destinationAlpha = 0.0f;
    }

    [UIView animateWithDuration:1.0f animations:^
    {
        self.redBox.alpha = destinationAlpha;
    }];
}
```

Mission Three

"Make the red box disappear (fade) with 1 seconds duration."

```
- (IBAction)toggleBoxVisibility:(id)sender
{
    CGFloat destinationAlpha = 0.0f;

    if (self.redBox.alpha == 0.0f)
    {
       destinationAlpha = 1.0f;
    }
    else
    {
       destinationAlpha = 0.0f;
    }

    [UIView animateWithDuration:1.0f animations:^
    {
        self.redBox.alpha = destinationAlpha;
    }];
}
```

Notice how if you tap quickly the animation skips forward to the next state?

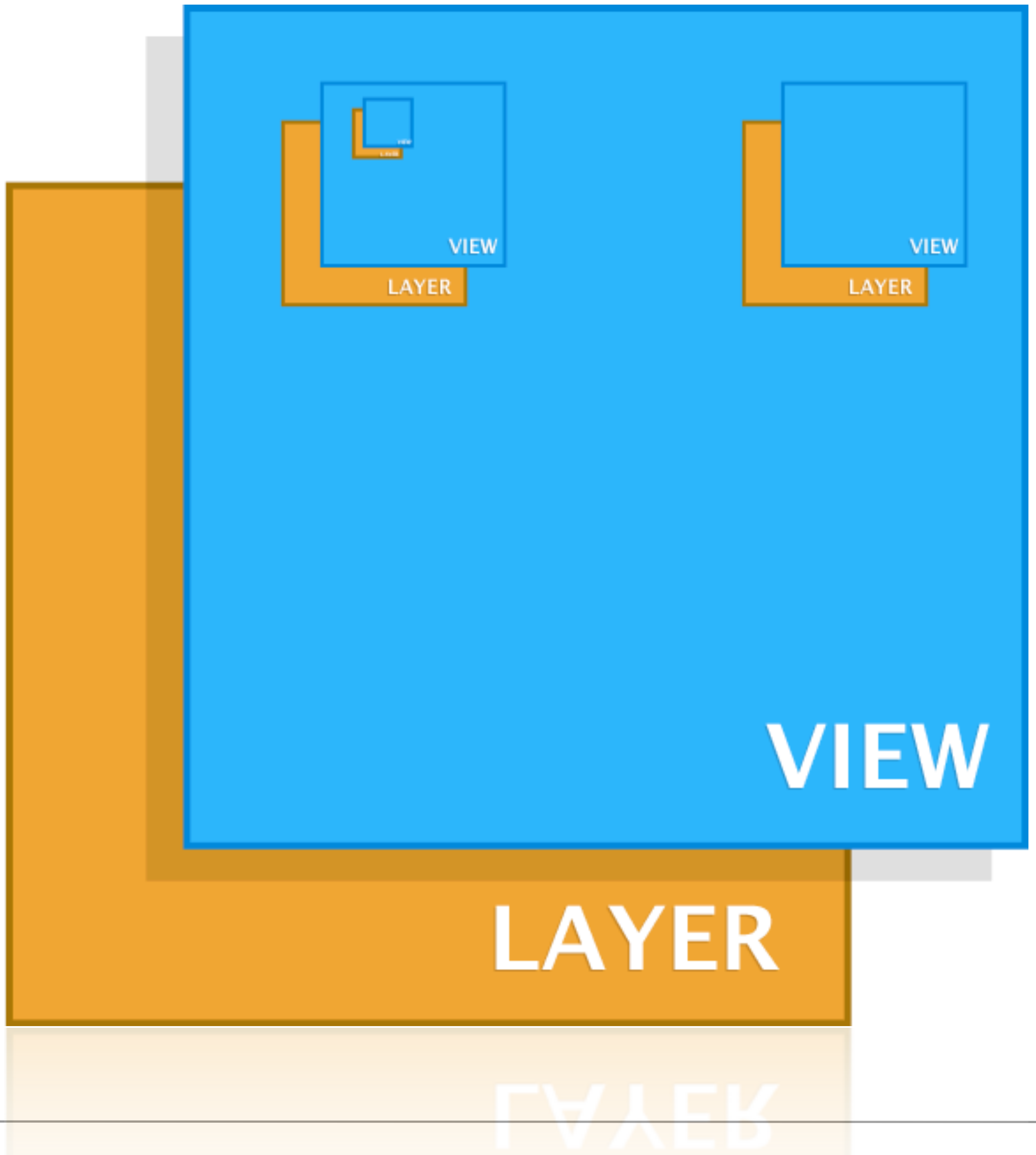
CoreAnimation
UIViews and CALayers

- Every `UIView` has an associated `CALayer` (backing it).
- `CALayer` hierarchy works almost exactly like the `UIView` hierarchy.
    - `addSubview:` **vs**. `addSublayer:`
- Internally, whenever you add a subview iOS mirrors the action to the corresponding layer.

WHY?

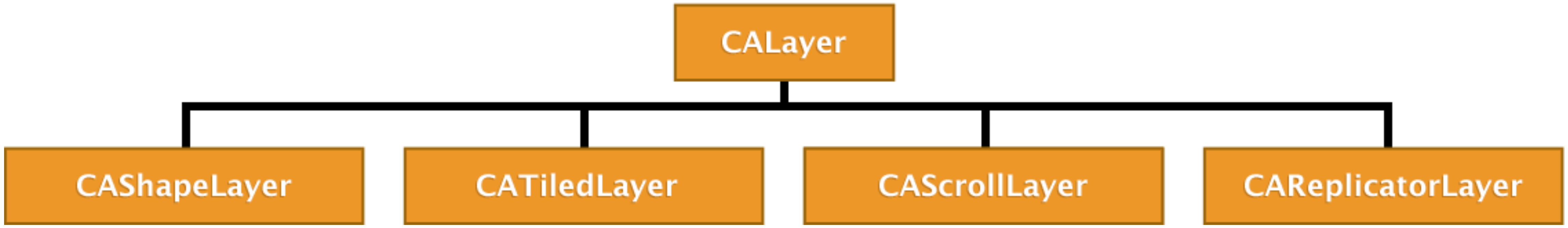
CoreAnimation

UIViews and CALayers



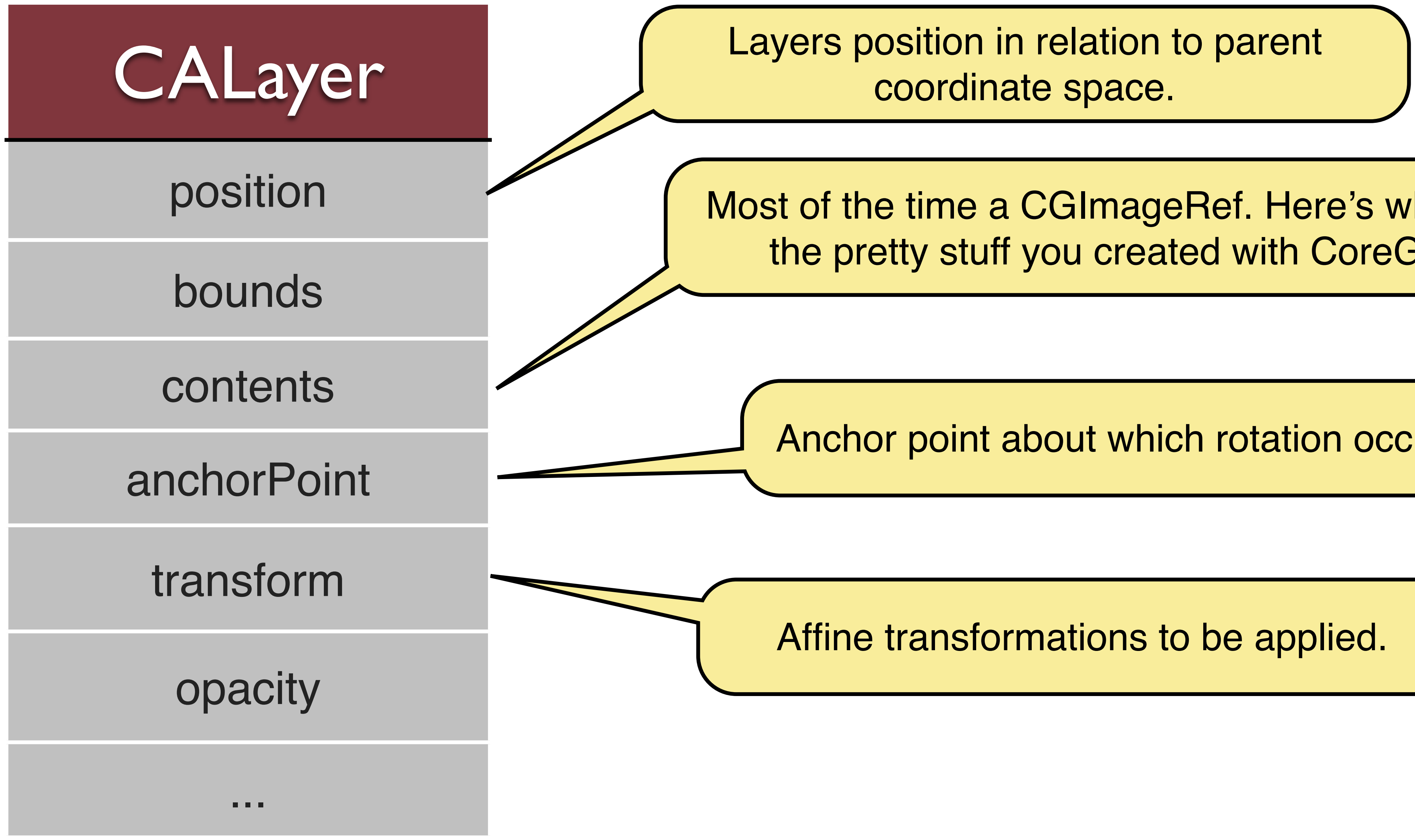
CoreAnimation

Various layers for different purposes



CoreAnimation

UIViews and CALayers


CALayer
position
bounds
contents
anchorPoint
transform
opacity
...
Layers position in relation to parent coordinate space.
Most of the time a CGImageRef. Here's where you put the pretty stuff you created with CoreGraphics.
Anchor point about which rotation occurs.
Affine transformations to be applied.

CoreAnimation
Implicit Animation

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
    UITouch *touch = (UITouch *)[touches anyObject];
    CALayer *hitLayer = [self.view.layer hitTest:[touch locationInView:self.view]];

    if ([hitLayer isEqual:self.redBox])
    {
        [self toggleRedBoxScaling];
    }
}

- (void)toggleRedBoxScaling
{
    if (CATransform3DIsIdentity(self.redBox.transform))
    {
        self.redBox.transform = CATransform3DMakeScale(2.0f, 2.0f, 1.0f);
    }
    else
    {
        self.redBox.transform = CATransform3DIdentity;
    }
}
```

Simply check on whether the layer was hit.

The next time the run-loop checks animate-able properties an (implicit) interpolation will occur.

Mission Four

Figure out how to disable implicit animations.

Mission Four - Solution

Figure out how to disable implicit animations.

```
- (void)toggleRedBoxScaling
{
    [CATransaction setDisableActions:YES];
    if (CATransform3DIsIdentity(self.redBox.transform))
    {
       self.redBox.transform = CATransform3DMakeScale(2.0f, 2.0f, 1.0f);
    }
    else
    {
       self.redBox.transform = CATransform3DIdentity;
    }
    [CATransaction setDisableActions:NO];
}
```

Disallows property changes to trigger a chain of events that leads to flagging-for-interpolation.

Enable it again if you don’t want to permanently disable it globally.

CoreAnimation

Explicit Animation

So what if I want the animation to last for 3 seconds and a flip to occur?

CoreAnimation

Explicit Animation

```
- (void)toggleRedBoxScalingExplicitDemo
{
    if (CATransform3DIsIdentity(self.redBox.transform))
    {
       CATransform3D rotation = CATransform3DMakeRotation(M_PI, 0.0f, 1.0f, 0.0f);
       CATransform3D scaling = CATransform3DMakeScale(2.0f, 2.0f, 1.0f);
       CATransform3D concatented = CATransform3DConcat(rotation, scaling);

       CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"transform"];
       [animation setFromValue:[NSValue valueWithCATransform3D:self.redBox.transform]];
       [animation setToValue:[NSValue valueWithCATransform3D:concatented]];
       animation.duration = 3.0f;

       [self.redBox addAnimation:animation forKey:@"transform"];
    }
    else
    {
       self.redBox.transform = CATransform3DIdentity;
    }
}
```

CoreAnimation
Explicit Animation

Create a 3D transformation that scales and flips.

```
- (void)toggleRedBoxScalingExplicitDemo
{
    if (CATransform3DIsIdentity(self.redBox.transform))
    {
        CATransform3D rotation = CATransform3DMakeRotation(M_PI, 0.0f, 1.0f, 0.0f);
        CATransform3D scaling = CATransform3DMakeScale(2.0f, 2.0f, 1.0f);
        CATransform3D concatented = CATransform3DConcat(rotation, scaling);

        CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"transform"];
        [animation setFromValue:[NSValue valueWithCATransform3D:self.redBox.transform]];
        [animation setToValue:[NSValue valueWithCATransform3D:concatented]];
        animation.duration = 3.0f;

        [self.redBox addAnimation:animation forKey:@"transform"];
    }
    else
    {
        self.redBox.transform = CATransform3DIdentity;
    }
}
```

CoreAnimation

Explicit Animation

```
- (void)toggleRedBoxScalingExplicitDemo
{
    if (CATransform3DIsIdentity(self.redBox.transform))
    {
       CATransform3D rotation = CATransform3DMakeRotation(M_PI, 0.0f, 1.0f, 0.0f);
       CATransform3D scaling = CATransform3DMakeScale(2.0f, 2.0f, 1.0f);
       CATransform3D concatented = CATransform3DConcat(rotation, scaling);

       CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"transform"];
       [animation setFromValue:[NSValue valueWithCATransform3D:self.redBox.transform]];
       [animation setToValue:[NSValue valueWithCATransform3D:concatented]];
       animation.duration = 3.0f;

       [self.redBox addAnimation:animation forKey:@"transform"];
    }
    else
    {
       self.redBox.transform = CATransform3DIdentity;
    }
}
```

Create a “Basic Animation” for the “transform” property.

CoreAnimation

Explicit Animation

```
- (void)toggleRedBoxScalingExplicitDemo
{
    if (CATransform3DIsIdentity(self.redBox.transform))
    {
       CATransform3D rotation = CATransform3DMakeRotation(M_PI, 0.0f, 1.0f, 0.0f);
       CATransform3D scaling = CATransform3DMakeScale(2.0f, 2.0f, 1.0f);
       CATransform3D concatented = CATransform3DConcat(rotation, scaling);

       CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"transform"];
       [animation setFromValue:[NSValue valueWithCATransform3D:self.redBox.transform]];
       [animation setToValue:[NSValue valueWithCATransform3D:concatented]];
       animation.duration = 3.0f;

       [self.redBox addAnimation:animation forKey:@"transform"];
    }
    else
    {
       self.redBox.transform = CATransform3DIdentity;
    }
}
```

Set the value at which the animations should start.

CoreAnimation

Explicit Animation

```
- (void)toggleRedBoxScalingExplicitDemo
{
    if (CATransform3DIsIdentity(self.redBox.transform))
    {
        CATransform3D rotation = CATransform3DMakeRotation(M_PI, 0.0f, 1.0f, 0.0f);
        CATransform3D scaling = CATransform3DMakeScale(2.0f, 2.0f, 1.0f);
        CATransform3D concatented = CATransform3DConcat(rotation, scaling);

        CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"transform"];
        [animation setFromValue:[NSValue valueWithCATransform3D:self.redBox.transform]];
        [animation setToValue:[NSValue valueWithCATransform3D:concatented]];
        animation.duration = 3.0f;

        [self.redBox addAnimation:animation forKey:@"transform"];
    }
    else
    {
        self.redBox.transform = CATransform3DIdentity;
    }
}
```

Set the value to which you'd like to interpolate to.

CoreAnimation

Explicit Animation

```
- (void)toggleRedBoxScalingExplicitDemo
{
    if (CATransform3DIsIdentity(self.redBox.transform))
    {
       CATransform3D rotation = CATransform3DMakeRotation(M_PI, 0.0f, 1.0f, 0.0f);
       CATransform3D scaling = CATransform3DMakeScale(2.0f, 2.0f, 1.0f);
       CATransform3D concatented = CATransform3DConcat(rotation, scaling);

       CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"transform"];
       [animation setFromValue:[NSValue valueWithCATransform3D:self.redBox.transform]];
       [animation setToValue:[NSValue valueWithCATransform3D:concatented]];
       animation.duration = 3.0f;

       [self.redBox addAnimation:animation forKey:@"transform"];
    }
    else
    {
       self.redBox.transform = CATransform3DIdentity;
    }
}
```

Add the animation for it to be executed next time the animate-able properties are checked.

CoreAnimation
Explicit Animation

```
- (void)toggleRedBoxScalingExplicitDemo
{
    if (CATransform3DIsIdentity(self.redBox.transform))
    {
       CATransform3D rotation = CATransform3DMakeRotation(M_PI, 0.0f, 1.0f, 0.0f);
       CATransform3D scaling = CATransform3DMakeScale(2.0f, 2.0f, 1.0f);
       CATransform3D concatented = CATransform3DConcat(rotation, scaling);

       CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"transform"];
       [animation setFromValue:[NSValue valueWithCATransform3D:self.redBox.transform]];
       [animation setToValue:[NSValue valueWithCATransform3D:concatented]];
       animation.duration = 3.0f;

       [self.redBox addAnimation:animation forKey:@"transform"];
    }
    else
    {
       self.redBox.transform = CATransform3DIdentity;
    }
}
```

Applying the identity matrix returns to base size.

Mission 4.5

Find out what’s wrong with the animation.

CoreAnimation
Model and Presentation Layer

- Whenever an animation of a `CALayer` is in progress, it happens on the **presentation layer**!
- The actual **model** of our `CALayer` is ***not*** being altered by animations!
- That's because the model shouldn't ever be caught stateless or transiting to another state.
- You can access the presentation layer via `[layer presentationLayer];`
  - This comes in handy if you want to interrupt an animation and proceed from where the animation is at that particular point in time!

Mission 4.5 - Solution

Find out what's wrong with the animation.

Mission 4.5 - Solution

Find out what's wrong with the animation.

```
- (void)toggleRedBoxScalingExplicitDemo
{
    if (CATransform3DIsIdentity(self.redBox.transform))
    {
       CATransform3D rotation = CATransform3DMakeRotation(M_PI, 0.0f, 1.0f, 0.0f);
       CATransform3D scaling = CATransform3DMakeScale(2.0f, 2.0f, 1.0f);
       CATransform3D concatenated = CATransform3DConcat(rotation, scaling);

       CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"transform"];
       [animation setFromValue:[NSValue valueWithCATransform3D:self.redBox.transform]];
       [animation setToValue:[NSValue valueWithCATransform3D: concatenated]];
       animation.duration = 3.0f;

       self.redBox.transform = concatenated; // Alter the model.

       [self.redBox addAnimation:animation forKey:@"transform"];
    }
    else
    {
       self.redBox.transform = CATransform3DIdentity;
    }
}
```

Mission 4.5 - Solution

Find out what's wrong with the animation.

```
- (void)toggleRedBoxScalingExplicitDemo
{
    if (CATransform3DIsIdentity(self.redBox.transform))
    {
       CATransform3D rotation = CATransform3DMakeRotation(M_PI, 0.0f, 1.0f, 0.0f);
       CATransform3D scaling = CATransform3DMakeScale(2.0f, 2.0f, 1.0f);
       CATransform3D concatenated = CATransform3DConcat(rotation, scaling);

       CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"transform"];
       [animation setFromValue:[NSValue valueWithCATransform3D:self.redBox.transform]];
       [animation setToValue:[NSValue valueWithCATransform3D: concatenated]];
       animation.duration = 3.0f;

       self.redBox.transform = concatenated; // Alter the model.

       [self.redBox addAnimation:animation forKey:@"transform"];
    }
    else
    {
       self.redBox.transform = CATransform3DIdentity;
    }
}
```

Alters the model and triggers implicit animations to be generated - we don't want that!

Mission 4.5 - Solution

Find out what's wrong with the animation.

```
- (void)toggleRedBoxScalingExplicitDemo
{
    if (CATransform3DIsIdentity(self.redBox.transform))
    {
       CATransform3D rotation = CATransform3DMakeRotation(M_PI, 0.0f, 1.0f, 0.0f);
       CATransform3D scaling = CATransform3DMakeScale(2.0f, 2.0f, 1.0f);
       CATransform3D concatenated = CATransform3DConcat(rotation, scaling);

       CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"transform"];
       [animation setFromValue:[NSValue valueWithCATransform3D:self.redBox.transform]];
       [animation setToValue:[NSValue valueWithCATransform3D: concatenated]];
       animation.duration = 3.0f;

       self.redBox.transform = concatenated; // Alter the model.

       [self.redBox addAnimation:animation forKey:@"transform"];
    }
    else
    {
       self.redBox.transform = CATransform3DIdentity;
    }
}
```

Thus we override the animation (that was generated by assigning a value to the model) with our own version!

CoreAnimation
Outlook

- CoreAnimation is incredibly powerful and **extensive**.
- Which makes it rather difficult at first but it’s no rocket science.
- I recommend “CoreAnimation for Mac OS X and iPhone” by Bill Dudney. Fabulous (and brief :)) book.
  (Source: http://pragprog.com/book/bdcora/core-animation-for-mac-os-x-and-the-iphone)

CoreAnimation+CoreGraphics

Why it matters in Game Development

- Every game has a way it interfaces with the user.
- That's where Apple's frameworks shine!
- You don't want to implement UI directly in OpenGL. Really. - Just don't.
- No matter if it's an OpenGL view or some other rendering canvas. Everything is wrapped in a `UIView`.
    - Example: "Quest" (iOS Game - OpenGL) has its entire UI created with methods I illustrated throughout the talk.
- Whether it's a good idea to implement an entire game in CA+CG depends on how crazy your calculations are.

Thank you :)!