

# iPhone/iPad DevCon 2010



## CUSTOM UI VIEWS

Nathan Eror - Free Time Studios  
@neror

WHO AM I?



<http://www.freetimestudios.com>

WTF?

# WTF?

- Creating custom views using Core Animation and Quartz
- How to choose the best solution
- What to watch out for

# THE ROLE OF A UIVIEW

- Drawing and animation
- Layout and subview management
- Event handling

# CUSTOM INTEGRATION POINTS

- Event handling methods:
  - `touchesBegan:withEvent:`
  - `touchesMoved:withEvent:`
  - `touchesEnded:withEvent:`
  - `touchesCancelled:withEvent:`
- The `layoutSubviews` method
- The `drawRect:` method
- The `layerClass` method

# WHICH DO I CHOOSE?

- The `layoutSubviews` method
  - Simplest option
  - Create, update, move subviews
- The `layerClass` method
  - Change the type of the backing layer
  - Specialized layers can be more powerful and focused
- The `drawRect:` method
  - Draw with Quartz into a `CGContext`

# DEMO: DRAW RECT

# CALAYERS & UIVIEWS

## UIView.h

```
UIKIT_EXTERN_CLASS @interface UIView : UIResponder<NSCoding> {  
    @package  
    CALayer *_layer;  
}  
  
+ (Class)layerClass;  
  
@property(n nonatomic, readonly, retain) CALayer *layer;
```

# CHANGE THE BACKING LAYER

## Your UIView Subclass

```
@implementation MyView

+ (Class)layerClass {
    return [CAShapeLayer class];
}
```

# SPECIAL CALAYER TYPES

- `CATiledLayer`: For displaying very large layers (bigger than 1024x1024) at multiple levels of detail
- `CAShapeLayer`: Draws an animatable `CGPath` (since 3.0)
- `CAGradientLayer`: Draws a gradient over its background color (since 3.0)
- `CATransformLayer`: A container layer for true 3D hierarchy (since 3.0)
- `CAReplicatorLayer`: Creates copies of its sublayers with each copy potentially having geometric, temporal and color transformations applied to it (since 3.0)
- `CATextLayer`: Renders text using Core Text. Supports loadable fonts and `NSAttributedString`. (since 3.2)

# SPECIAL LAYER EXAMPLES

# GRADIENT LAYERS

## GradientLayers.m

```
gradientLayer_.backgroundColor = [[UIColor blackColor] CGColor];
gradientLayer_.bounds = CGRectMake(0., 0., 200., 200.);
gradientLayer_.position = self.view.center;
gradientLayer_.cornerRadius = 12.;
gradientLayer_.borderWidth = 2.;
gradientLayer_.borderColor = [[UIColor blackColor] CGColor];
gradientLayer_.startPoint = CGPointZero;
gradientLayer_.endPoint = CGPointMake(0., 1.);
gradientLayer_.colors = [NSArray arrayWithObjects:
    (id)[[UIColor whiteColor] CGColor],
    (id)[UIColorFromRGBA(0xFFFFFFFF, .1) CGColor],
    nil];
```

# SHAPE LAYERS

## ShapeLayers.m

```
shapeLayer_.backgroundColor = [[UIColor clearColor] CGColor];
shapeLayer_.frame = CGRectMake(0., 0., 200., 200.);
shapeLayer_.position = self.view.center;

CGMutablePathRef path = CGPathCreateMutable();
CGPathAddEllipseInRect(path, NULL, rect);
shapeLayer_.path = path;
CGPathRelease(path);

shapeLayer_.fillColor = [[UIColor blueColor] CGColor];
shapeLayer_.strokeColor = [[UIColor blackColor] CGColor];
shapeLayer_.lineWidth = 4.;
shapeLayer_.lineDashPattern = [NSArray arrayWithObjects:
                                [NSNumber numberWithInt:8],
                                [NSNumber numberWithInt:8],
                                nil];
shapeLayer_.lineCap = kCALineCapRound;
```

- Shape layers allow for optimized drawing of simple CGPaths
- The changing of the 'path' property can be animated

# TEXT LAYERS

## TextLayers.m

```
CTFontRef font = CTFontCreateWithName(CFSTR("Courier"), 16.f, NULL);
normalTextLayer_ = [[CATextLayer alloc] init];
normalTextLayer_.font = font;
normalTextLayer_.string = @"This is just a plain old CATextLayer";
normalTextLayer_.wrapped = YES;
normalTextLayer_.foregroundColor = [[UIColor purpleColor] CGColor];
normalTextLayer_.fontSize = 20.f;
normalTextLayer_.alignmentMode = kCAAlignmentCenter;
normalTextLayer_.frame = CGRectMake(0.f, 10.f, 320.f, 32.f);
CFRelease(font);
```



- 'string' property can be either an NSString or an NSAttributedString
- If using NSAttributedString, the text styling properties (fontSize, foregroundColor, etc) are ignored (see next slide)

# TEXT LAYERS

## TextLayers.m

```
NSDictionary *fontAttributes = [NSDictionary dictionaryWithObjectsAndKeys:
    @"Courier", (NSString *)kCTFontFamilyNameAttribute,
    @"Bold", (NSString *)kCTFontStyleNameAttribute,
    [NSNumber numberWithFloat:16.f], (NSString *)kCTFontSizeAttribute,
    nil];

CTFontDescriptorRef descriptor =
    CTFontDescriptorCreateWithAttributes((CFDictionaryRef)fontAttributes);
CTFontRef courierFont = CTFontCreateWithFontDescriptor(descriptor, 0, NULL);
CFRelease(descriptor);

NSDictionary *stringAttributes =
    [NSDictionary dictionaryWithObject:(id)courierFont
    forKey:(NSString *)kCTFontAttributeName];

NSMutableAttributedString *attrString =
    [[NSMutableAttributedString alloc] initWithString:EXAMPLE_STRING
    attributes:stringAttributes];

NSRange rangeOfClassName = [[attrString string] rangeOfString:@"CATextLayer"];
[attrString addAttribute:(NSString *)kCTForegroundColorAttributeName
    value:(id)[UIColor redColor] CGColor]
    range:rangeOfClassName];

CFRelease(courierFont);
attributedTextLayer_.string = attrString;
```

# RESOURCES

- <http://github.com/neror/CA360>
- <http://github.com/neror/CustomUIViews>
- <http://ftutils.com>
- <http://www.freetimestudios.com/resources/>

# THANK YOU

Nathan Eror

[neror@freetimestudios.com](mailto:neror@freetimestudios.com)

<http://twitter.com/freetimestudios>

<http://twitter.com/neror>

<http://www.freetimestudios.com>

<http://www.neror.com>