Optimize Your iPhone App for the Retina Display

Andrew Platzer
iOS Application Frameworks Engineer

Richard Schreyer
iPhone GPU Software

Retina Display

- Refined visuals
- More immersive interface
- Support different resolutions
- Compatibility

Retina Display

UIKit

- Points ≠ pixels
- Images
- Views and drawing
- Core Graphics and Core Animation
- Icons and launch images

Points ≠ Pixels

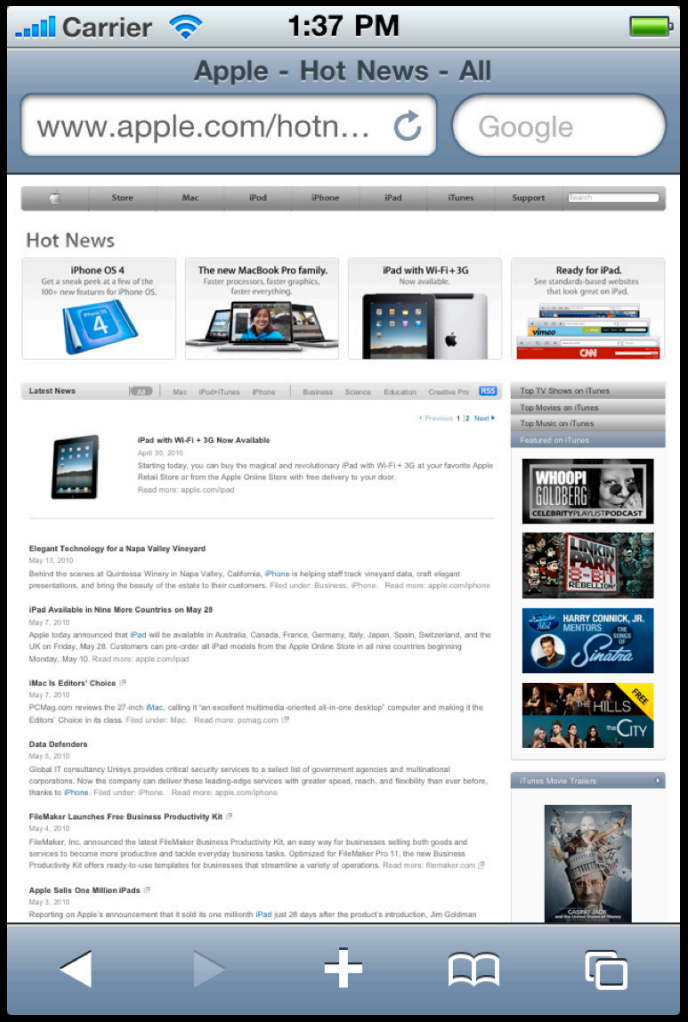
Points ≠ Pixels

- Points and pixels were used interchangeably
- Almost a given
- Not anymore!
- All values are in device-independent points

Points ≠ Pixels

- Retina display has 2× resolution
  - 1 point = 2 pixels
- Advantages to exact 2× scaling
  - No fractional pixels
  - No pixel cracks
  - No fuzzy scaling
- 2× resolution means 4× pixels!
  - Look out for speed and memory hits


Carrier
1:37 PM
Apple - Hot News - All
www.apple.com/hotne...
Google
Hot News


Carrier
1:37 PM
Apple - Hot News - All
www.apple.com/hotn...
Google
Store
Mac
iPod
iPhone
iPad
iTunes
Support
Hot News
iPhone OS 4
Get a sneak peek at a few of the 100+ new features for iPhone OS.
The new MacBook Pro family.
Faster processors, faster graphics, faster everything.
iPad with Wi-Fi+3G
Now available.
Ready for iPad.
See standards-based websites that look great on iPad.
Latest News
iPad with Wi-Fi + 3G Now Available
Top TV Shows on iTunes
Top Movies on iTunes
Top Music on iTunes
Featured on iTunes
iTunes Movie Trailers


Carrier
App
www.apple.cor
Carrier
App
www.apple.co
Store
Mac

What Is a ‘Point’?

- *device-independent measurement of pixel elements*

What Is a ‘Point’?

- ~~*dimples*~~
- No fixed physical dimension
  - iPhone 3GS ~160 DPI
  - iPad ~130 DPI
  - Retina Display ~320 DPI
- 10 *point* system font is readable
- 40×40 *point* rectangle is a good touch size

Scaling

UIScreen

- `UIScreen.scale` —read-only property
- `UIScreen.bounds` is in **points**
  - Full screen is still 320 × 480
  - No UI relayout
- `UITouch.locationInView:` is in points
  - Non-integer touch point values
- `UIWindow.frame` is in points

UIImage

UIImage

Properties

- `UIImage.size` — points, not pixels
- `UIImage.scale` — always non-zero
- Pixel dimensions = size × scale
  - Or `CGImageGetWidth/Height`

UIImage

Files

- High-resolution images
  - *"@2x"* suffix—`Button@2x.png`
- Automatic selection using screen scale
  - `+imageNamed:`
  - `+imageWithContentsOfFile:`
- Ignores file resolution (DPI)
  - `+imageWithData:` always return 1.0 scaled image

UIImage

- `+imageNamed:` no longer needs explicit extension

```
UIImage *image = [UIImage imageNamed:"Button"];
```

- `UIImage` from `CGImageRef`:
  - `+imageWithCGImage:scale:orientation:`
  - `-initWithCGImage:scale:orientation:`
- `UIImagePNGRepresentation`, `UIImageJPEGRepresentation`
  - Write out a DPI value
  - DPI = scale x 72.0

UIGraphicsBeginImageContext

- `size` — pixels
- *`scale`* — 1.0
- *`opacity`* — not opaque

UIGraphicsBeginImageContextWithOptions

- `size` — points
- `scale` — 0.0 … ∞
  - 0.0 = use main screen scale
- `opacity` — `YES/NO`
  - Drawing performance
  - No space performance (padding)
- **Thread safe drawing** in iOS 4!
  - UIColor, UIFont, UIImage, NSString drawing

Views and Drawing

UIView

- `UIView.contentScaleFactor`
  - **Only** affects content
  - Does not affect view geometry or subviews
  - Always returns non-zero value
- `–drawRect: – window.screen.scale`
- `UIImageView – image.scale`
- Never need to set

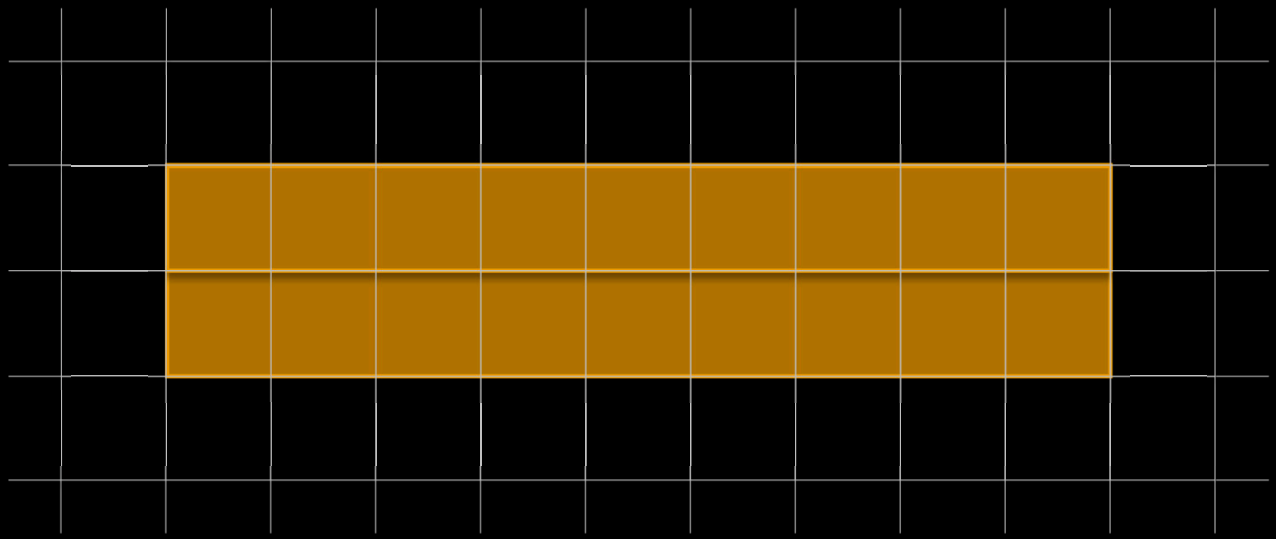
UIView

- `-drawRect:` will pass in a scaled `CGContextRef`
  - `rect` is in points
  - Bitmap buffer size is in pixels
  - Current transform matrix (CTM) scaled
- Text and bezier path drawing are automatically high resolution
- Use `contentScaleFactor` to determine size of a single pixel

Single Pixel Drawing

Using a rectangle

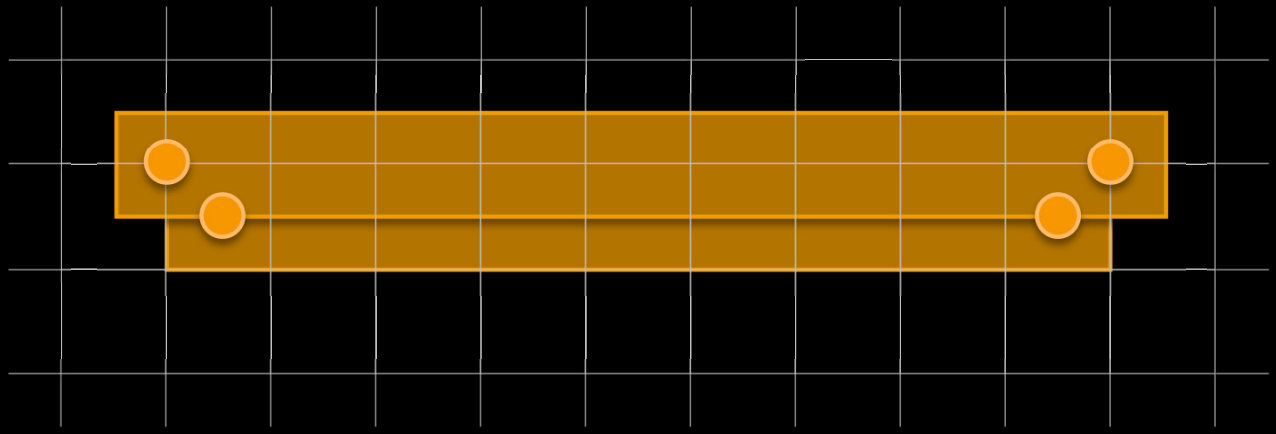
```
CGFloat height = 1.0 / self.contentScaleFactor;
CGRectFill(CGRectMake(0.0, 0.0, width, height));
```



Single Pixel Drawing

Using a bezier path

```
UIBezierPath *path = [UIBezierPath bezierPath];
CGFloat lineWidth = 1.0 / self.contentScaleFactor;
path.lineWidth = lineWidth;
[path moveToPoint:CGPointMake(0.0, 0.0)];
[path addLineToPoint:CGPointMake(width, 0.0)];
[path addLineToPoint:CGPointMake(width - lineWidth, lineWidth / 2.0)];
[path addLineToPoint:CGPointMake(0.0, lineWidth / 2.0)];
[path closePath];
[path stroke];
```



UIImage

- Draw methods use `UIImage.scale`
  - `-drawAtPoint:...`
  - `-drawInRect:...`

UIImageView

- Automatically sets `contentScaleFactor` to `image.scale`
- Highlighted image should have same scale factor
- Animated images should have same scale factor
- Don’t add your own `–drawRect:`

UIScrollView

- `scrollview.contentOffset`
  - Don’t be surprised to see non-integer offsets
  - Avoid rounding

Core Graphics and Core Animation

Core Graphics

Pixels

- Core Graphics has no concept of *image resolution* (DPI)
- Works **only** in pixels
- No place to store a ‘scale’

Core Graphics

CGBitmapContexts

- Please use `UIGraphicsBeginImageContextWithOptions`
- If you use `CGBitmapContextCreate`
  - Set up scale transform
  - Flip co-ordinate system for UIKit drawing
  - Push context to use UIKit drawing
  - Keep track of scale for extracted `CGImageRefs`
- No real performance gain

ImageIO

Reading DPI

- `CGImageSourceCopyPropertiesAtIndex`
  - `kCGImagePropertyDPIWidth/Height (CFNumberRef)`
  - Generally 72.0 DPI = 1× scale
  - PNG uses pixels/cm so watch out for 71.999

Core Animation

- `CALayer.contentsScale`
  - `UIView.contentScaleFactor`
  - Automatically set by UIView, UIImageView
- Set the contentsScale if you set CALayer.contents directly
  - Mask layers

Icons and Launch Images

Icons

Application

- Create a 2× application icon—114×114
- Add entry in *Info.plist*
  - Icon Files (aka `CFBundleIconFiles`)—array of file names
  - Correct one chosen based on size

Key	Value
Information Property List	(16 items)
Icon files	(2 items)
Item 0	AppIcon.png
Item 1	AppIcon@2x.png

Icons

Documents

- Create a 2× sized document icon—44×58
- Add entry in *Info.plist*
  - `CFBundleDocumentTypes`—dictionary
    - `CFBundleTypeIconFiles`—array of file names
    - Correct one chosen based on size

Key	Value
Information Property List	(16 items)
CFBundleDocumentTypes	(1 item)
Item 0	(8 items)
CFBundleTypeIconFiles	(2 items)
Item 0	DocIcon.png
Item 1	DocIcon@2x.png

Icons

Settings and Search

- Create a 2× settings icon—58×58
- Add entry in *Info.plist*
  - Icon files—along with application icons
  - Correct one chosen based on size

Key	Value
Information Property List	(16 items)
Icon files	(4 items)
Item 0	AppIcon.png
Item 1	AppIcon@2x.png
Item 2	SettingsIcon.png
Item 3	SettingsIcon@2x.png

Launch Images

- Snapshot your launch images
- Add entry in *Info.plist*
  - Launch image (aka `UILaunchImageFile`)

Key	Value
▼ Information Property List	(16 items)
Launch image	Launch.png

  - 'Base' name
  - Orientation, scale, and platform suffixes
    - Launch-Landscape.png
    - Launch@2x.png
    - Launch-Landscape@2x.png

Summary

UIKit and the Retina Display

UIKit and the Retina Display

High resolution "don'ts"

1. Assume points = pixels
2. Expect `UIImage.size` = pixel size
3. Assume `CGContextRefs` have 1.0 scale
4. Cram more content into your interface
5. Create radically different high-resolution artwork and icons

UIKit and the Retina Display

Levels of support

0. Do nothing—high-quality text, bezier paths, system UI
1. Add '@2x' images and icons—automatic selection
2. Generate high-resolution offscreen images—scale of 0.0
3. Draw in high resolution—`view.contentScaleFactor`

OpenGL on the Retina Display

Richard Schreyer
iPhone GPU Software

OpenGL

Overview

- Requires explicit adoption
- OpenGL is a **pixel**-based API
- Perform your own point to pixel conversion

```
pixelSize = bounds.size * contentScaleFactor
```

OpenGL

Adoption steps

1. Allocate high-resolution color buffer
2. Fix hard-coded sizes
3. Load higher-resolution artwork

OpenGL

Setting color buffer size

- `UIView.contentScaleFactor`
  - Defaults to `UIScreen.scale` for most built-in UIKit views
  - Defaults to 1.0 for OpenGL views

OpenGL

Setting color buffer size

```
//  allocate color buffer
view.contentScaleFactor = [UIScreen mainScreen].scale;
[ctx renderbufferStorage: target fromDrawable: view.layer];


//  save color buffer dimensions
glGetRenderbufferParameteriv(target, RENDERBUFFER_WIDTH, &pixelWidth);
glGetRenderbufferParameteriv(target, RENDERBUFFER_HEIGHT, &pixelHeight);
```

OpenGL

Avoid hard-coded sizes

```
glScissor (GLint x, GLint y, GLsizei width, GLsizei height)
glViewport (GLint x, GLint y, GLsizei width, GLsizei height)
glReadPixels (GLint x, GLint y, GLsizei width, GLsizei height, ...)
glTexImage2D (..., GLsizei width, GLsizei height, ...)
glRenderbufferStorage (..., GLsizei width, GLsizei height)
```

- Depth buffer size must match color buffer

```
glRenderbufferStorage(target, DEPTH_COMPONENT24, pixelWidth, pixelHeight);
```

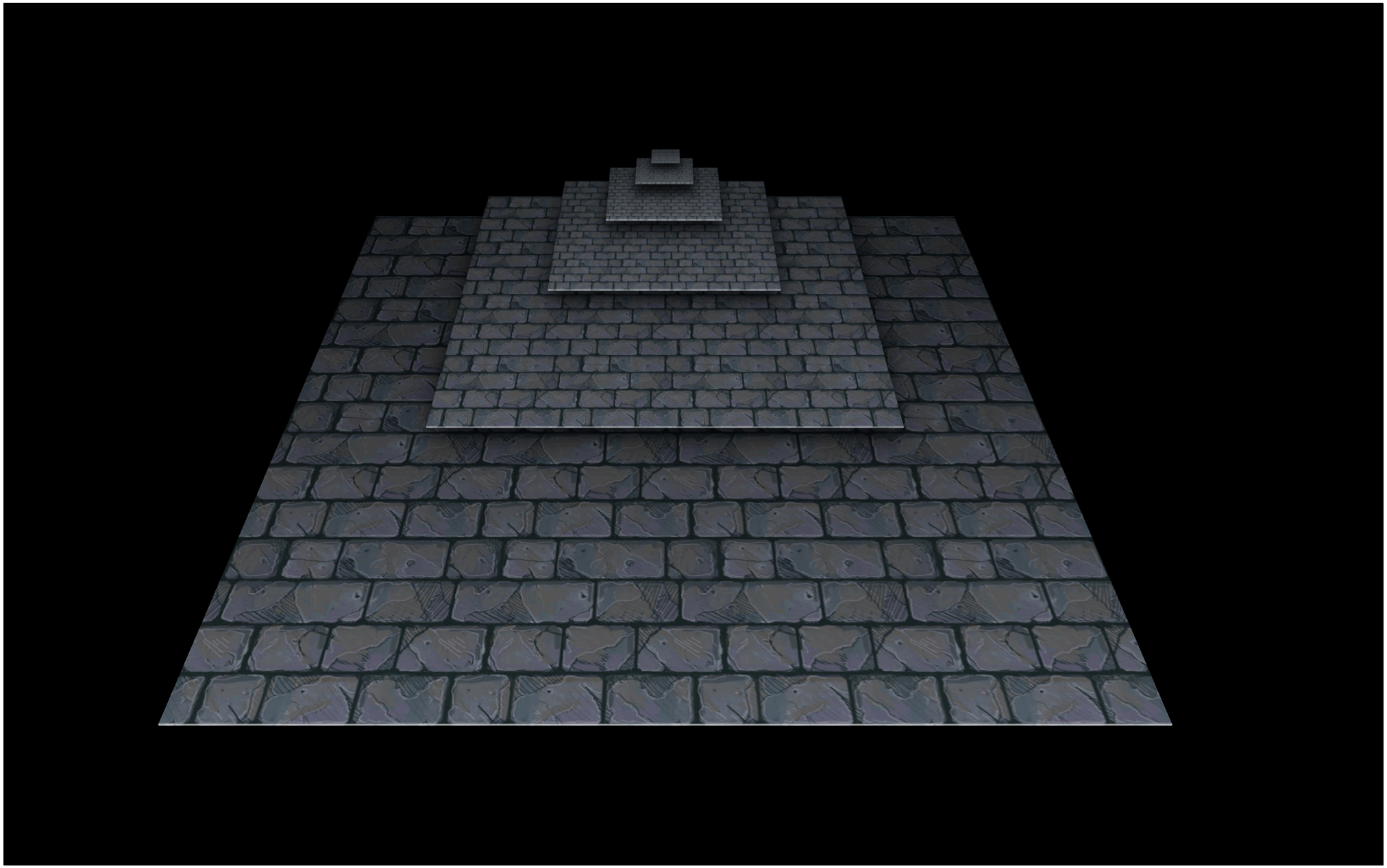
- Most applications use a full-size viewport

```
glViewport(0, 0, pixelWidth, pixelHeight);
```

OpenGL

Texture sizing

- Use higher-resolution textures
- Share assets with iPad
  - Performance and display size are similar



OpenGL

Loading textures with UIImage

- `UIImage.size` returns points, not pixels
  - `glTexImage2D` requires pixels
- Pixel dimensions = image.size × image.scale
  - Or use CGImageGetWidth/Height

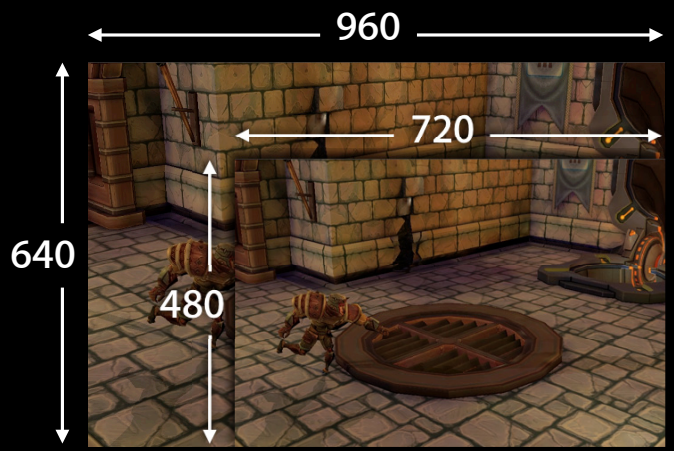
OpenGL

Performance tuning

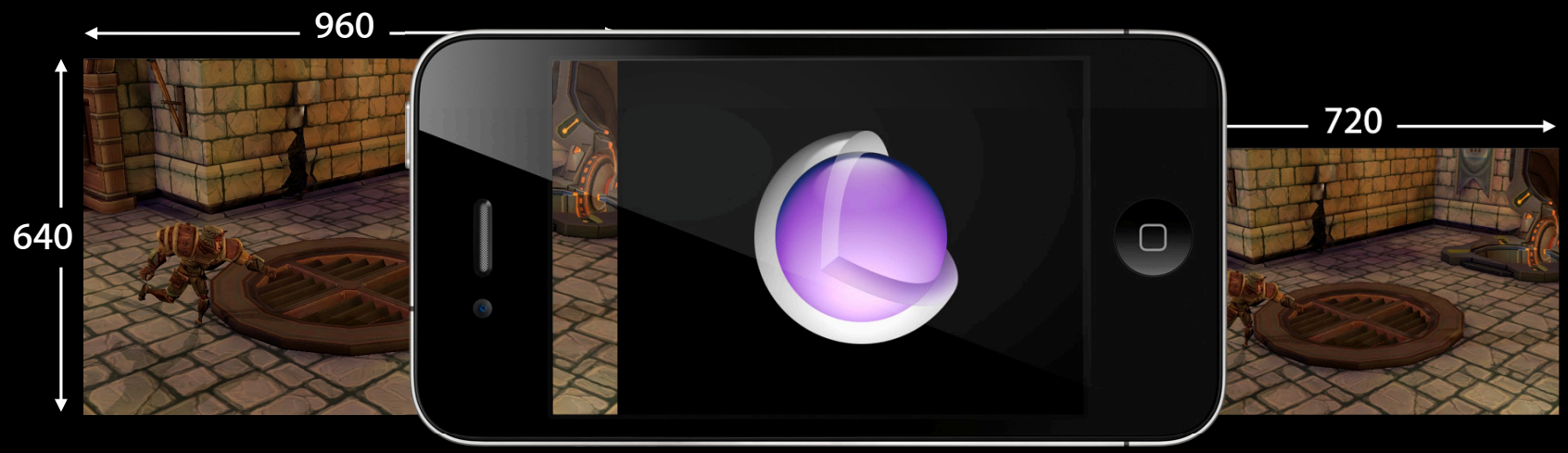
- Increasing GPU workload
- How many pixels are being drawn?
- How expensive is each pixel?
  - Not using mipmaps
  - Long fragment shaders
  - Alpha test

OpenGL ES Tuning & Optimization	Presidio Wednesday 4:30PM

Performance Tuning



Performance Tuning



OpenGL

Performance tuning

- Render at smaller screen resolution
  - contentScaleFactor = 1 (default)
  - contentScaleFactor = 1, add anti-aliasing
  - contentScaleFactor somewhere between 1.0 and UIScreen.scale

OpenGL on Retina Display

Summary

- UIKit uses points, OpenGL uses pixels
- To adopt:
  - Set `glview.contentScaleFactor = screen.scale`
  - Check Depth/Stencil renderbuffer dimensions
  - Check glViewport parameters

More Information

Bill Dudney
Frameworks Evangelist
dudney@apple.com

Documentation
iPhone Application Programming Guide
http://developer.apple.com/iphone

Apple Developer Forums
http://devforums.apple.com

Related Sessions

What's New in Cocoa Touch	Marina Friday 11:30AM
OpenGL ES Tuning & Optimization	Presidio Wednesday 4:30PM

Labs

Retina Display Lab	Application Frameworks Lab D Thursday 4:30PM


WWDC10