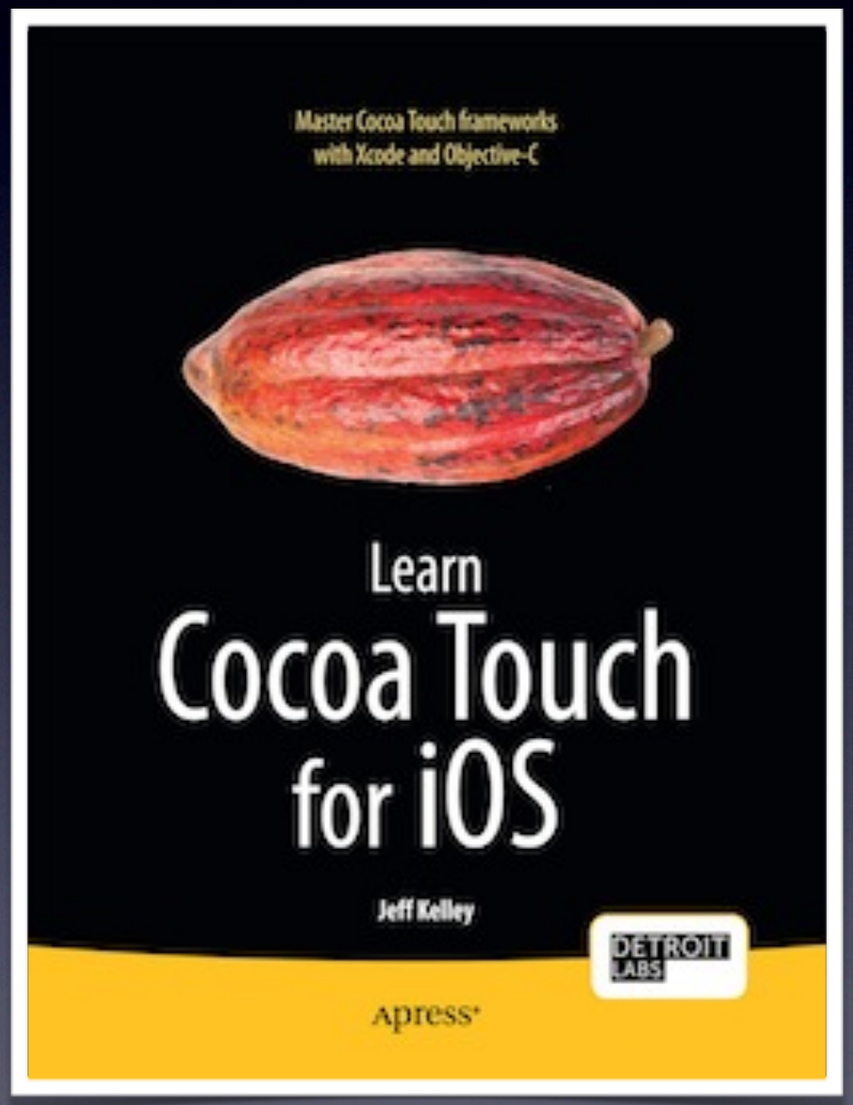
Using Images in iOS

Jeff Kelley | @SlaunchaMan

CodeMash 2013

Who Am I?



DETROIT
LABS

Using Images in iOS

Using Images in iOS

Why A Talk About Images?

iDevice Displays

Image Pitfalls

- One uncompressed Retina iPad image file: ~12 MB
- App Store download limit for non-WiFi connections: 50 MB
- One uncompressed background image for all resolutions: ~18 MB

Special Images in the Bundle

- Icon Files
  - 29x29 (iPhone Settings)
  - 50x50 (iPad Search Results)
  - 57x57 (iPhone Icon)
  - 58x58 (iPhone Settings, Retina)
  - 72x72 (iPad Icon)
  - 100x100 (iPad Search Results, Retina)
  - 114x114 (iPhone Icon, Retina)
  - 144x144 (iPad Icon, Retina)

Special Images in the Bundle

- iTunes Artwork
  - Include as ‘iTunesArtwork’ (no extension) for ad-hoc builds
- now must be 1024x1024
  - Warn your designer!
  - Consider using vector graphics

Special Images in the Bundle

- Default Images
  - Display while your app is launching
  - One for each resolution/rotation
  - Separate image for iPhone 5
    - The presence of this image indicates whether your app has iPhone 5 support

How can we balance good UIs with image sizes?

UIImage

- The highest-level image representation in Cocoa Touch
- Create from a file, from data, or from lower-level representations
- Represents an *image*, not image *data* (more on that later)

Creating a UIImage

- From a File:

```
UIImage *myImage = [[UIImage alloc]
initWithContentsOfFile:myPath];
```

- From Data:

```
UIImage *myImage = [[UIImage alloc]
initWithData:myData];
```

Cached Images

- Using the class method `+imageNamed:`, you can create an image from the main bundle
- These images are cached automatically
- For PNG images, you can leave off the extension

Image Cache Behavior

- In low-memory situations, a UIImage's image data will be purged
- This data will be reloaded once the image is needed again

Using Images

- Usually: UIImageView
- Occasionally: Create an NSData object using `UIImagePNGRepresentation()` or `UIImageJPEGRepresentation()`.

Retina Graphics

- Everything on an iOS display is measured in *points*, not *pixels*
- To get the pixel dimensions of an image, multiply *scale* by *size*
- Images created with +imageNamed: automatically seek out image files with the "@2x" suffix before the extension
  - "image.png" and "image@2x.png"

Retina Graphics

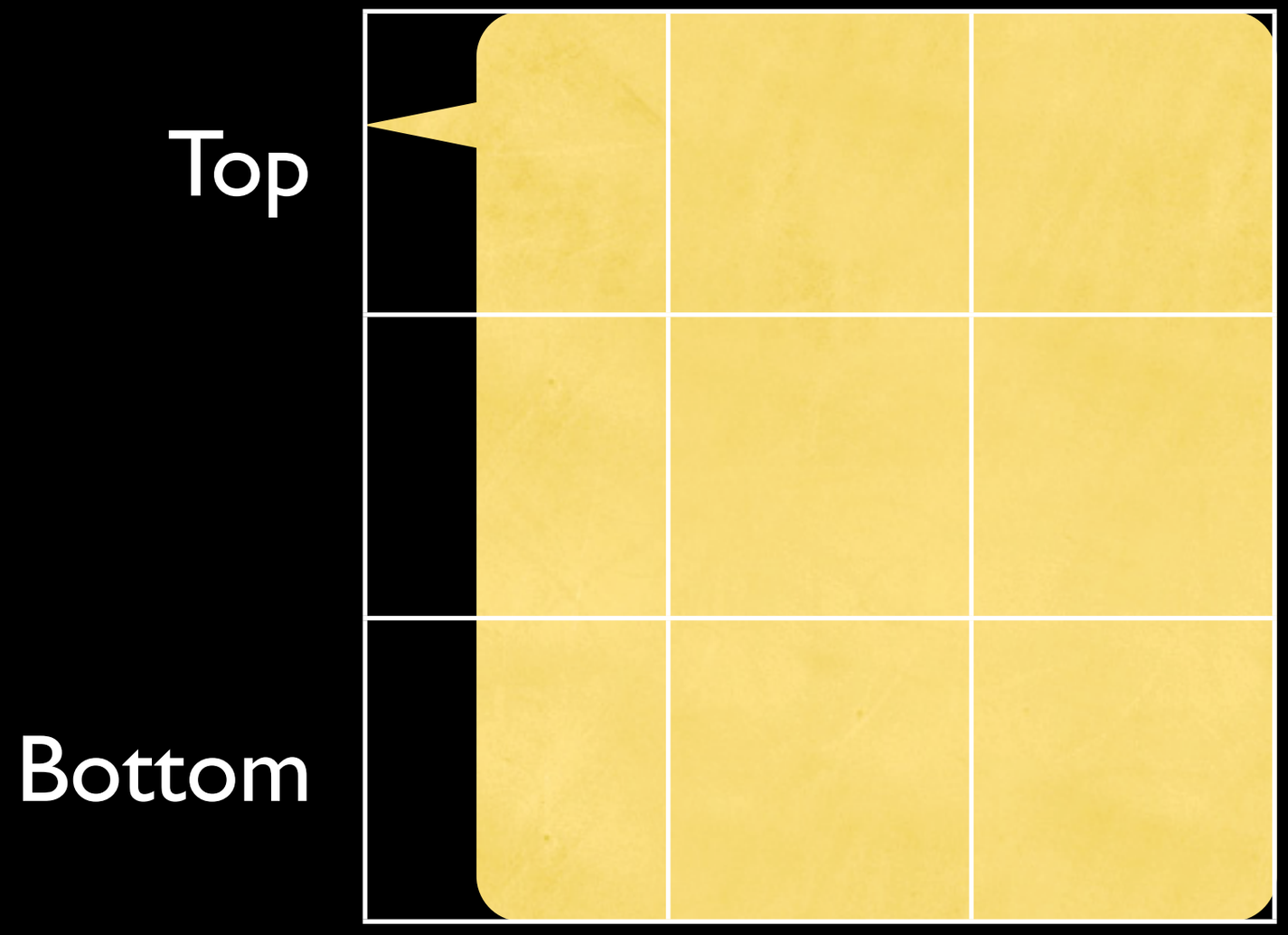
- Tell your designers: design for non-Retina, create for Retina
  - Yes, this is hard.
- Retina graphics must be exactly twice the scale—*no odd dimensions*.
  - Even with drop shadows!

Image Tips & Tricks

Stretchable Images

- Once you have an image, you can create a resizable version that can resize to any arbitrary size
- `- (UIImage *)resizableImageWithCapInsets: (UIEdgeInsets)capInsets;`
- The inner portion is tiled or stretched

Stretchable Images



Demo

Animated Images

- Load individual frames of an animation as separate images and cycle through them
- Can initialize with an array of images or with a prefix
- Up to 1,024 frames

Animated Images

- Creating from a prefix:

```
+ (UIImage *)animatedImageNamed:(NSString *)name
duration:(NSTimeInterval)duration;
```

- Creating from an array:

```
+ (UIImage *)animatedImageWithImages:(NSArray
*)images duration:(NSTimeInterval)duration;
```

Demo

Animated Images

- If you're really crazy:
  - `+ (UIImage *)animatedResizableImageNamed:(NSString *)name capInsets:(UIEdgeInsets)capInsets duration:(NSTimeInterval)duration;`

Image Patterns

- Sometimes you want to make a tiny image repeat, such as in a background
- Create a UIColor object with the image as a pattern:

```
UIColor *myPatternColor = [UIColor
colorWithPatternImage:myImage];
```

- Great way to save space

Demo

“We’ll Do It Live!”: Rendering with CoreGraphics

CoreGraphics

- AKA QuartzCore (<QuartzCore/QuartzCore.h>)
- Allows you to draw graphics *yourself* as opposed to including images

Custom View Rendering

- The easiest path to custom drawing
- Override `-drawRect:`

Graphics Contexts

- Graphics contexts, represented by the `CGContext` opaque type, are the thing you draw *into*
- In `-drawRect:`, you can get the current context with `UIGraphicsGetCurrentContext()`

Demo

Custom Drawing

- A view doesn't always redraw itself
- Call `-setNeedsDisplay` to mark it as dirty
- When marked as dirty, will re-draw in the event loop

Custom Drawing

- Can be a performance problem
- Lots of duplicate rendering effort

Problem: Interpolation

- Images resized in UIImageViews have no control over interpolation quality
- Bad solution: multiple images in the bundle, resized to the necessary sizes
- How can we resize an image with good quality?

Interpolating with CoreGraphics

- Create a new bitmap graphics context
- Set its interpolation quality to high
- Draw the original image into the context
- Get a new image

Demo

Image Data

- CoreGraphics is fun and all, but sometimes you need to access pixel data
  - What is pixel data? How is it formatted
  - iOS uses (mostly) RGBA8888 format
    - What the heck does that mean?

Color Spaces

- `CGColorSpace` opaque type
- Two main color spaces to worry about:
  - `CGColorSpaceCreateDeviceGray()`
  - `CGColorSpaceCreateDeviceRGB()`
- Defines the number of components

Graphics Context Options

- CGContextRef CGBitmapContextCreate
  (
    void *data,
    size_t width,
    size_t height,
    size_t bitsPerComponent,
    size_t bytesPerRow,
    CGColorSpaceRef colorspace,
    CGBitmapInfo bitmapInfo
  );

Getting Pixel Data

- Create a buffer of pixel data to draw into
- We'll use uint8_t for unsigned 8-bit integers
- Create a bitmap image context
- Draw the image into the context
- Voilà!

Getting Pixel Data

- Why didn't we just ask the image for its data?
- Drawing it into our own context give us a mutable version of it
- We know the color space we're using

Demo

Generating Pixel Data

- Remember before when we made a bitmap graphics context with data?
- Turns out you can pre-fill that data with your own pixels

Demo

Additional Topics

- OpenGL
- CoreImage
- Vector Graphics

AmazeKit

- Image rendering library for iOS
- Uses stacked image effects, analogous to Photoshop layer styles
- Renders UI into image, caches heavily
- Allows for small bundle sizes, rendered-on-demand UI
- Soon to be open-sourced

AmazeKit Example