Announcements

• Lab 2 is due next Monday (Sept 25th) by 11:59 PM
  – Late policy is 10% of lab total per day late
    • So -7.5 points per day late for lab 2

• Labs 3 and 4 are posted on the course website

Creating Views
Where do views come from?

- Commonly placed in Storyboard
- Drag out any of the existing view objects (buttons, labels, etc)
- Or drag generic UIView and set custom class

Manual Creation

- Views are initialized using UIView.init(frame:)
  
  ```
  let theFrame = CGRect(x:0, y:0, width:200, height:150)
  let myView = UIView(frame: theFrame)
  ```

- Example:
  
  ```
  let frame = CGRect(x:20, y:45, width: 140, height: 20)
  let myLabel = UILabel(frame:frame)
  myLabel.text = "Hello Class"
  view.addSubview(myLabel)
  ```
Defining Custom Views

- Subclass UIView

- For custom drawing, you override:
  ```swift
  func draw(_ rect: CGRect)
  ```

- For event handling, you override:
  ```swift
  func touchesBegan(_ touches: Set<UITouch> withEvent: UIEvent?)
  func touchesMoved(_ touches: Set<UITouch> withEvent: UIEvent?)
  func touchesEnded(_ touches: Set<UITouch> withEvent: UIEvent?)
  ```

Drawing Views
draw: Method

- **draw: does nothing by default**
  - If not overridden, then backgroundColor is used to fill

- **Override – draw: to draw a custom view**
  - rect argument is area to draw

- **When is it OK to call draw:?**

Be Lazy

- **draw: is invoked automatically**
  - Don’t call it directly!

- **Being lazy is good for performance**

- **When a view needs to be redrawn, use:**
  setNeedsDisplay
CoreGraphics and Quartz 2D

- UIKit offers very basic drawing functionality
  - UIRectFill(CGRect rect)
  - UIRectFrame(CGRect rect)

- CoreGraphics: Drawing APIs

- CG is a C-based API, not Objective-C

- CG and Quartz 2D drawing engine define simple but powerful graphics primitives
  - Graphics context
  - Transformations
  - Paths
  - Colors
  - Fonts
  - Painting operations
CG Wrappers

- Some CG functionality wrapped by UIKit
  - UIColor
    - Convenience for common colors
    - Easily set the fill and/or stroke colors when drawing

    ```
    UIColor.red.set()
    // drawing will be done in red
    ```
  - UIFont
    - Access system font
    - Get font by name
    - Get preferred font for a given text style
      - Best way for font in code
      ```
      class func preferredFont(forTextStyle style: UIFontTextStyle) -> UIFont
      ```
    - A few examples of Text Styles
      - UIFontTextStyle.headline
      - UIFontTextStyle.body
      - UIFontTextStyle.footnote

Simple draw(_: ) example

- Draw a solid color and shape

```swift
override func draw(_ rect: CGRect) {
    let bounds = self.bounds
    UIColor.gray.set()
    UIRectFill (bounds)
    
    let myShape = CGRect(x: 10, y: 10, width: 50, height: 100)
    UIColor.red.set()
    UIRectFill(myShape)
    
    UIColor.black.set()
    UIRectFrame(myShape)
}
```
Drawing More Complex Shapes

- Common steps for draw:
  - Get current graphics context
  - Define a path
  - Set a color
  - Stroke or fill path
  - Repeat, if necessary

Paths

- CoreGraphics paths define shapes
- Made up of lines, arcs, curves and rectangles
- Creation and drawing of paths are two distinct operations
  - Define path first, then draw it
Drawing Shapes using Bezier Paths

- First create a Bezier Path
  let path = UIBezierPath()

- Move around, add lines or arcs to path
  path.move(to: CGPoint(x:60,y:40))
  path.addLine(to: CGPoint(x:100,y:50))

Simple Example

```swift
override func draw(_ rect: CGRect){
    let path = UIBezierPath()
    path.move(to: CGPoint(x: 75,y: 10))
    path.addLine(to: CGPoint(x: 10,y: 150))
    path.addLine(to: CGPoint(x: 160,y: 150))
    path.close()
    UIColor.red.setFill()
    UIColor.black.setStroke()
    path.lineWidth = 3.0
    path.stroke()
    path.fill()
}
```

What shape is this?
More Drawing Information

- UIView Class Reference
- CGContext Reference
- “Quartz 2D Programming Guide”
- Lots of samples in the iPhone Dev Center

Lab 3 Preview
In Class Demo using Views