CS 326
UIViews and Custom Views

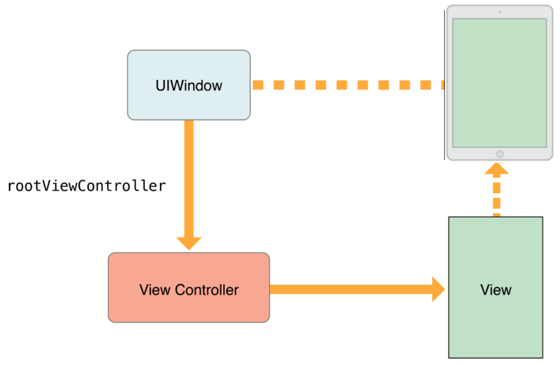
Stephen Freund

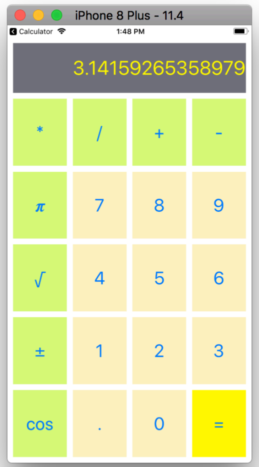
1

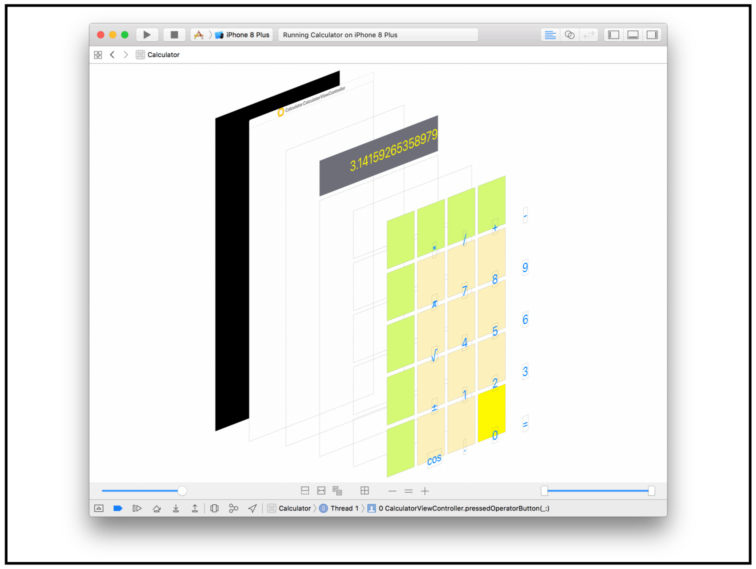
UIViews

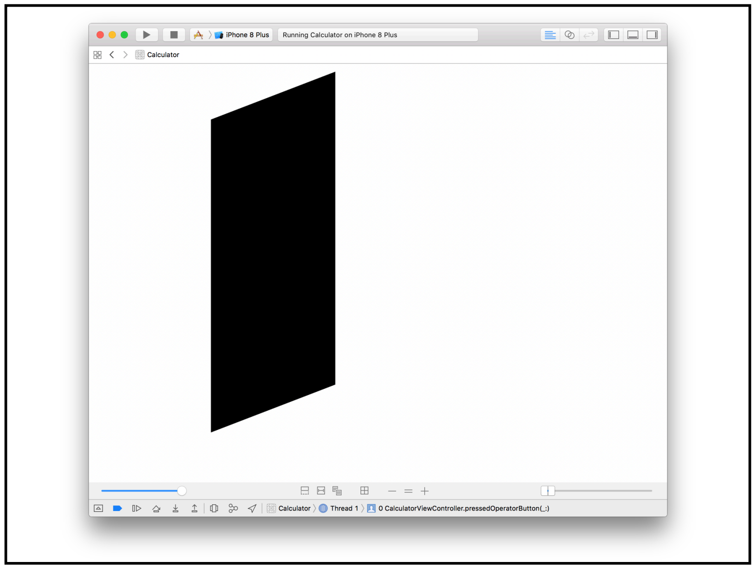
- Rectangular area for
  - drawing
  - handling touch events
- Hierarchical
  - labels, buttons, etc.
  - stack views...
  - starts with **`view`** property of **`UIViewControllers`**

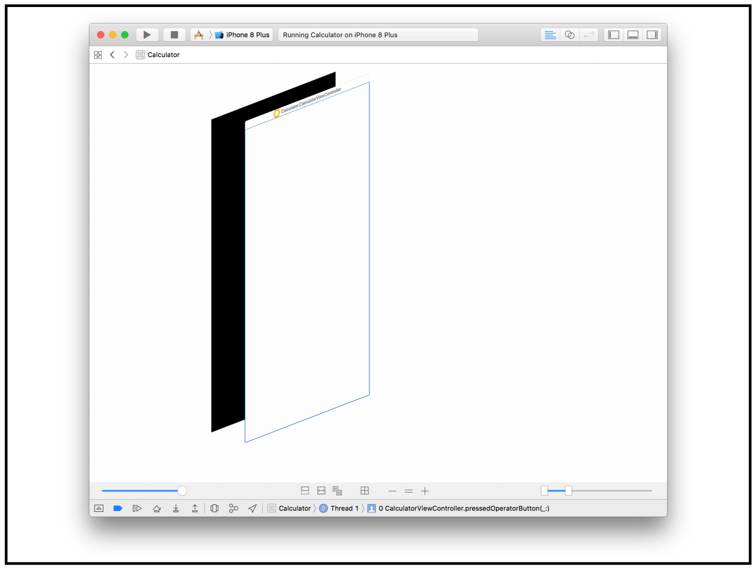
UIWindow, ViewController, and View

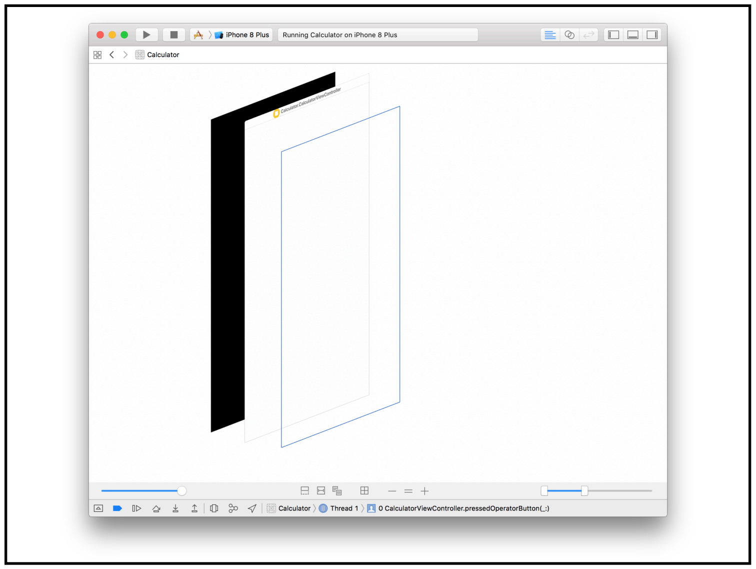


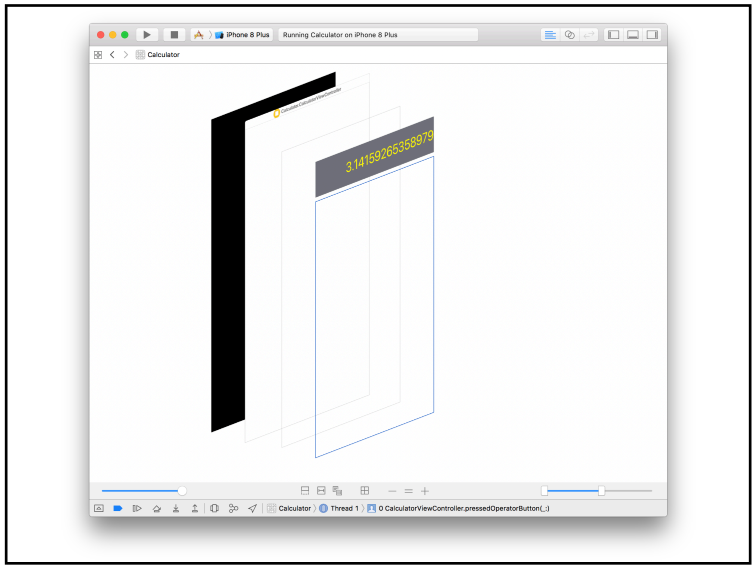


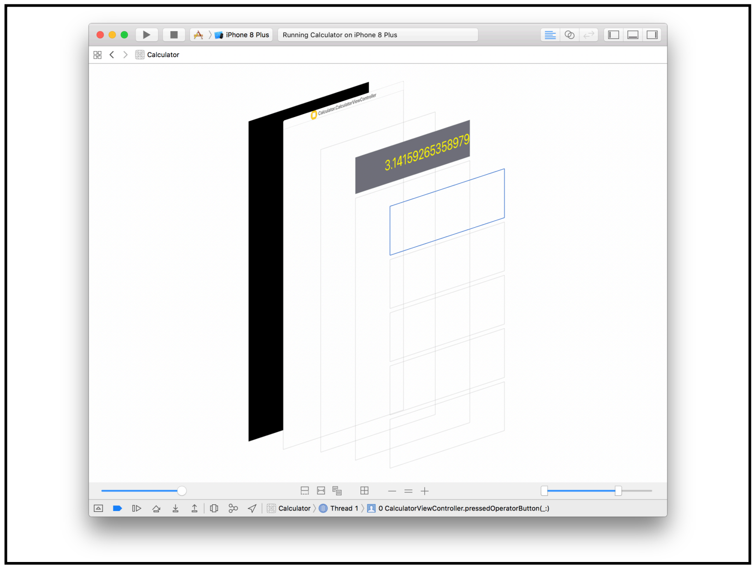

iPhone 8 Plus
Running Calculator on iPhone 8 Plus
Calculator
3.14159265358979
Calculator > Thread 1 > 0 CalculatorViewController.pressedOperatorButton(_:)

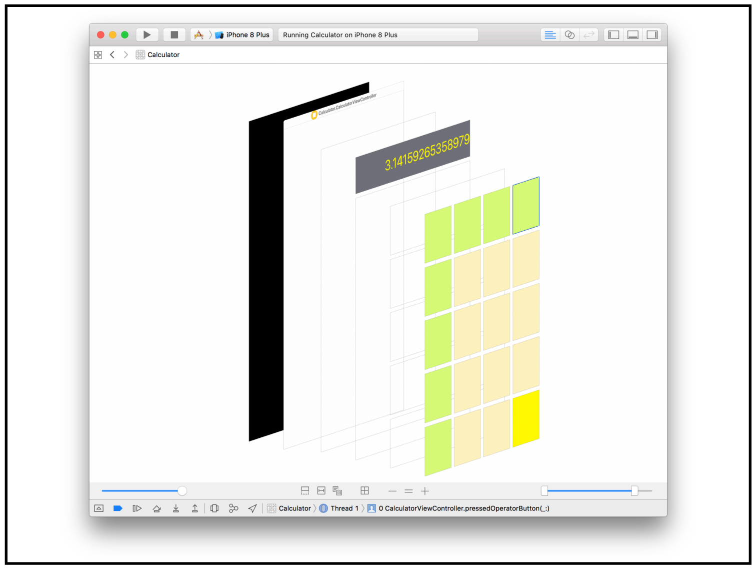

iPhone 8 Plus
Running Calculator on iPhone 8 Plus
Calculator
Calculator > Thread 1 > 0 CalculatorViewController.pressedOperatorButton(_:)

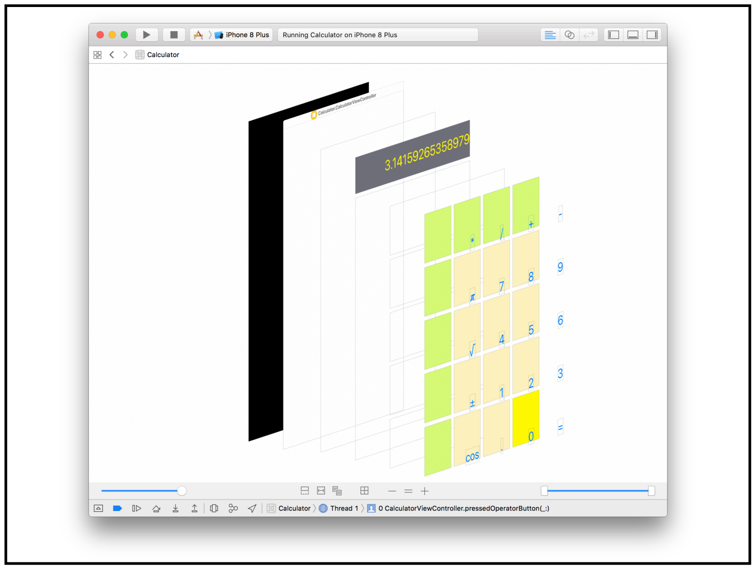

iPhone 8 Plus
Running Calculator on iPhone 8 Plus
Calculator
Calculator > Thread 1 > 0 CalculatorViewController.pressedOperatorButton(_:)


iPhone 8 Plus
Running Calculator on iPhone 8 Plus
Calculator
Calculator > Thread 1 > 0 CalculatorViewController.pressedOperatorButton(_:)


iPhone 8 Plus
Running Calculator on iPhone 8 Plus
Calculator
3.14159265358979
Calculator
Thread 1
0 CalculatorViewController.pressedOperatorButton(_:)


iPhone 8 Plus
Running Calculator on iPhone 8 Plus
Calculator
3.14159265358979
Calculator
Thread 1
0 CalculatorViewController.pressedOperatorButton(_:)


iPhone 8 Plus
Running Calculator on iPhone 8 Plus
Calculator
3.14159265358979
Calculator
Thread 1
0 CalculatorViewController.pressedOperatorButton(_:)


iPhone 8 Plus
Running Calculator on iPhone 8 Plus
Calculator
3.14159265358979
+
9
8
7
π
6
5
4
√
3
2
1
±
=
0
cos
Calculator
Thread 1
0 CalculatorViewController.pressedOperatorButton(_:)

Initializing Views and Controllers

- UIControllers or UIViews are created directly from storyboard data.
- Do not add initializers to them!
- We'll see ways to initialize parts of them later
  - eg: viewDidLoad()

UIView Coordinate System

- `CGFloat`
  - Use this instead of Double or Float.
  - Conversions exist: `CGFloat(myDouble)`
- `CGPoint`
  - `var point = CGPoint(x: 10.0, y: 13.4)`
  - `point.x += 2`
  - `point.y -= 22.2`
- `CGSize`
  - `let size = CGSize(width: 10.5, height: 50)`
  - `let area = size.width * size.height`

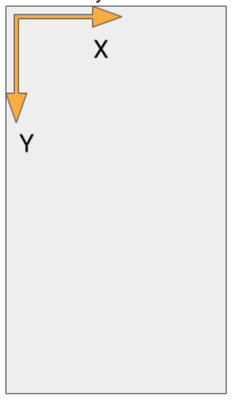
UIView Coordinate System

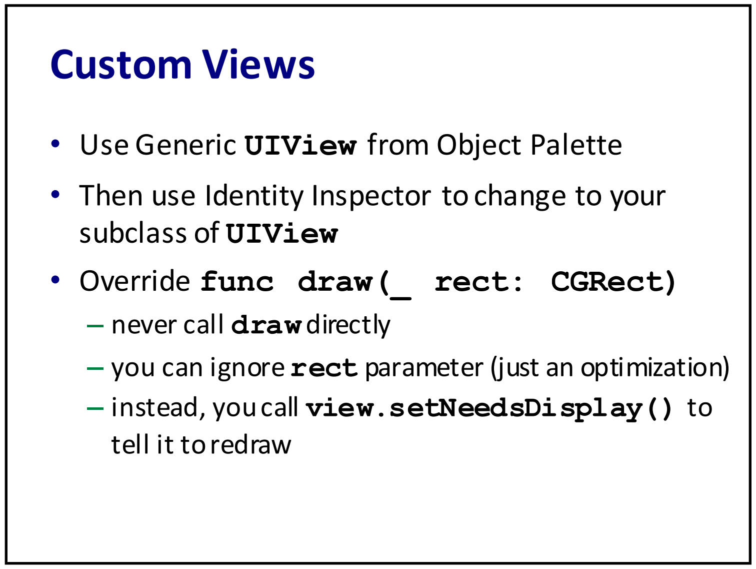
- `CGRect`
  - point + size
  - `var rect = CGRect(origin: aPt, size: aSize)`
- Lots of properties / methods:
  - `origin, size`
  - `minX, midX, maxX, minY, midY, maxY`
  - `intersects: (CGRect) -> Bool`
  - `intersect: (CGRect) -> CGRect`
  - `contains: (CGPoint) -> Bool`
  - ...
  - see docs

UIView Coordinate System

- Coordinate System
  - Origin is top left
  - Units are points, not pixels (Typically 2 pixels per point)
- `UIView` Properties
  - `bounds`: boundaries of where drawing happens
  - `frame`: where it is in parent's coordinate system
    - never use this in CS326...

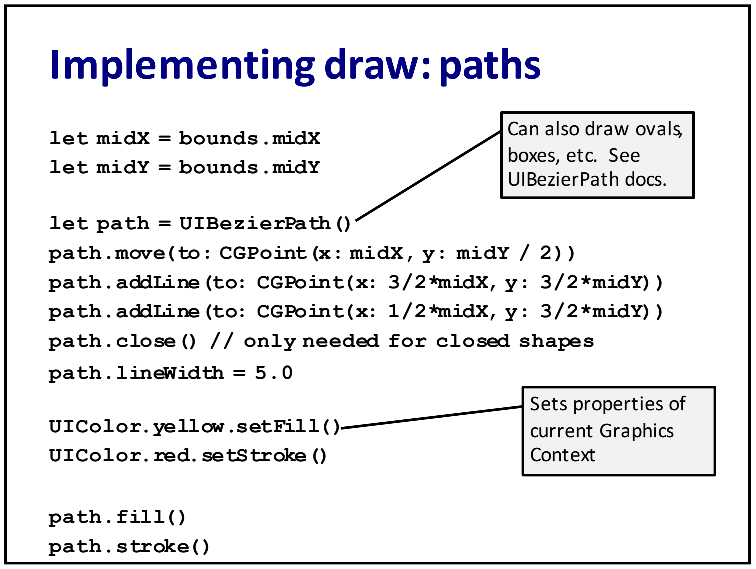
UIView Coordinate System





Custom Views

- Use Generic **`UIView`** from Object Palette
- Then use Identity Inspector to change to your subclass of **`UIView`**
- Override **`func draw(_ rect: CGRect)`**
  - never call **`draw`** directly
  - you can ignore **`rect`** parameter (just an optimization)
  - instead, you call **`view.setNeedsDisplay()`** to tell it to redraw

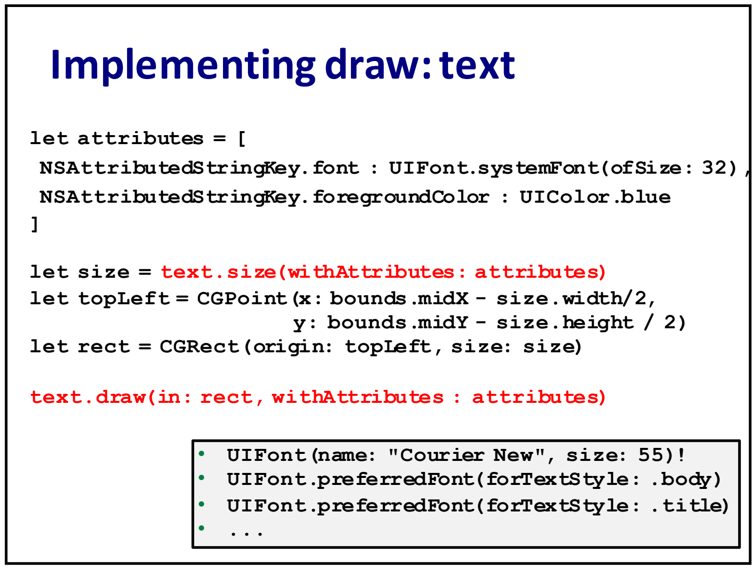


Implementing draw: paths

```
let midX = bounds.midX
let midY = bounds.midY

let path = UIBezierPath()
path.move(to: CGPoint(x: midX, y: midY / 2))
path.addLine(to: CGPoint(x: 3/2*midX, y: 3/2*midY))
path.addLine(to: CGPoint(x: 1/2*midX, y: 3/2*midY))
path.close() // only needed for closed shapes
path.lineWidth = 5.0

UIColor.yellow.setFill()
UIColor.red.setStroke()

path.fill()
path.stroke()
```

Can also draw ovals, boxes, etc. See UIBezierPath docs.

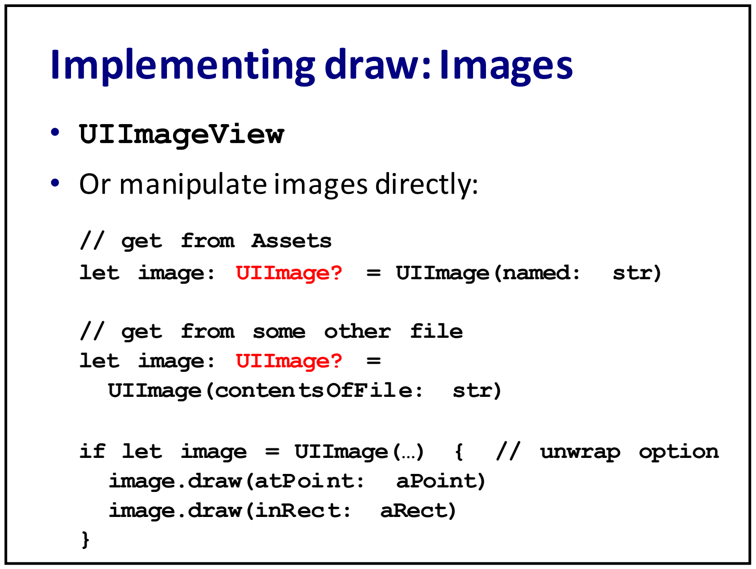
Sets properties of current Graphics Context



Implementing draw: text

```
let attributes = [
 NSAttributedStringKey.font : UIFont.systemFont(ofSize: 32),
 NSAttributedStringKey.foregroundColor : UIColor.blue
]

let size = text.size(withAttributes: attributes)
let topLeft = CGPoint(x: bounds.midX - size.width/2,
                      y: bounds.midY - size.height / 2)
let rect = CGRect(origin: topLeft, size: size)

text.draw(in: rect, withAttributes : attributes)
```

- `UIFont(name: "Courier New", size: 55)!`
- `UIFont.preferredFont(forTextStyle: .body)`
- `UIFont.preferredFont(forTextStyle: .title)`
- `...`



Implementing draw: Images

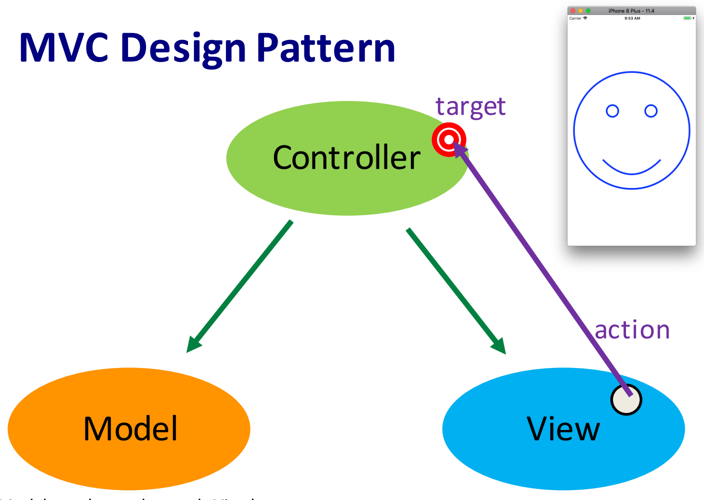
- **`UIImageView`**
- Or manipulate images directly:

```
// get from Assets
let image: UIImage? = UIImage(named: str)

// get from some other file
let image: UIImage? =
  UIImage(contentsOfFile: str)

if let image = UIImage(…) { // unwrap option
  image.draw(atPoint: aPoint)
  image.draw(inRect: aRect)
}
```

UIView Attributes

- Need to force a **`UIView`** to be redrawn when device orientation changes.
  - In Attributes Inspector, set **`UIView`**'s "Content Mode" to "redraw"
- Others
  - hidden
  - backgroundColor
  - transparency
- Experiment!

MVC Design Pattern



Model won't exactly match View's representation of a face...