View Geometry & Target-Action Controls

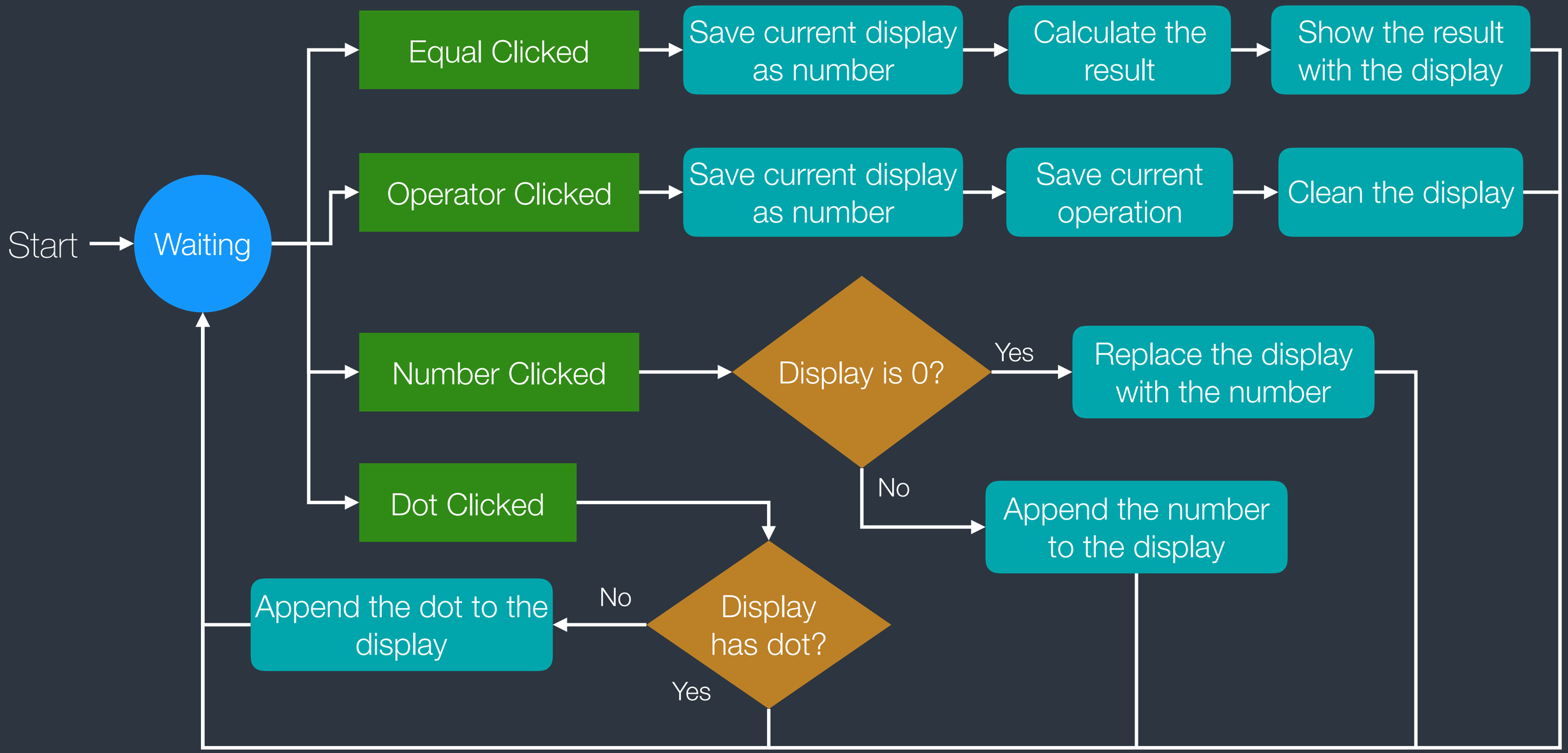
Nov. 17 ’16 - Introduction to iOS SDK

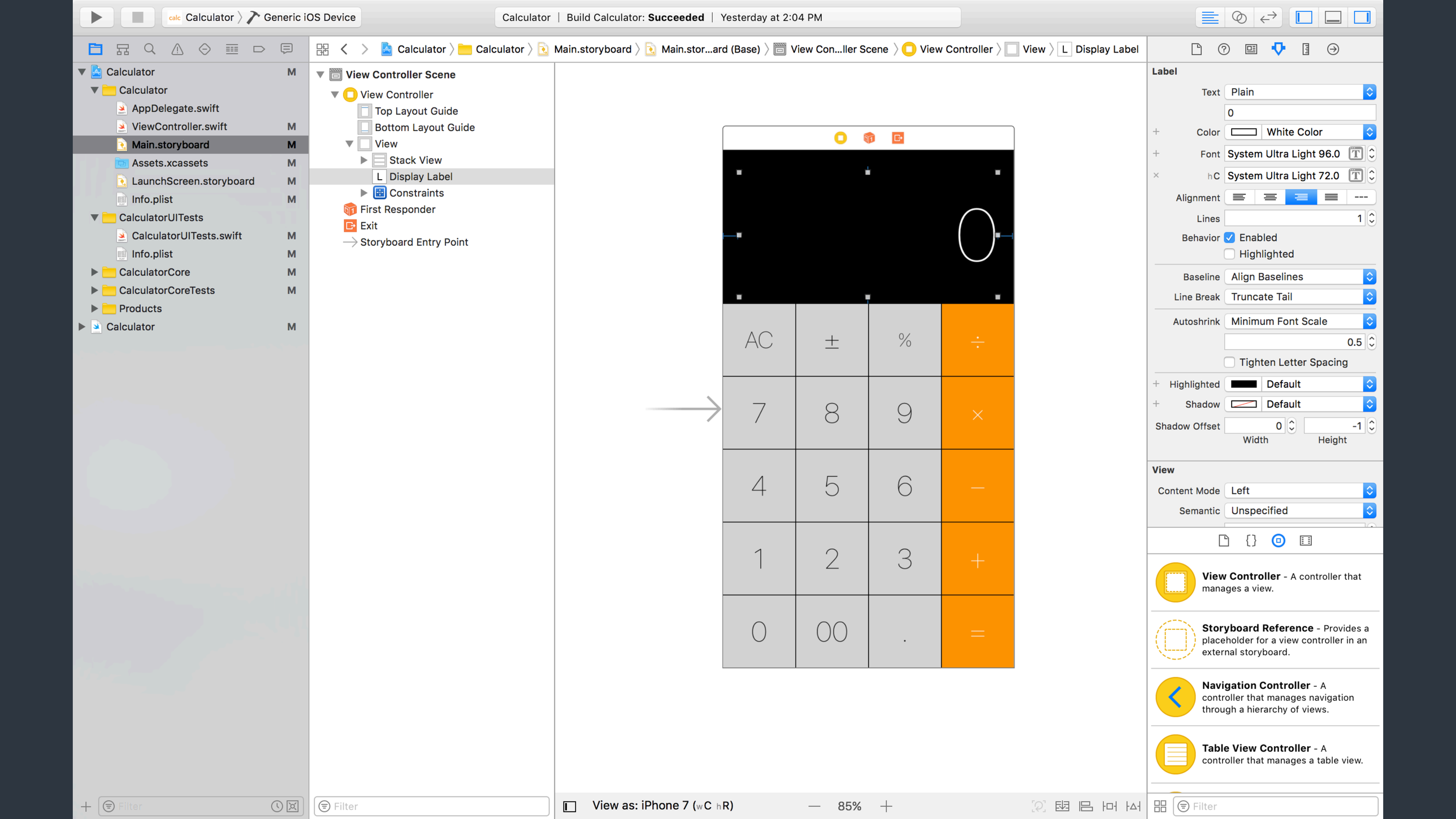
Tien-Che Tsai

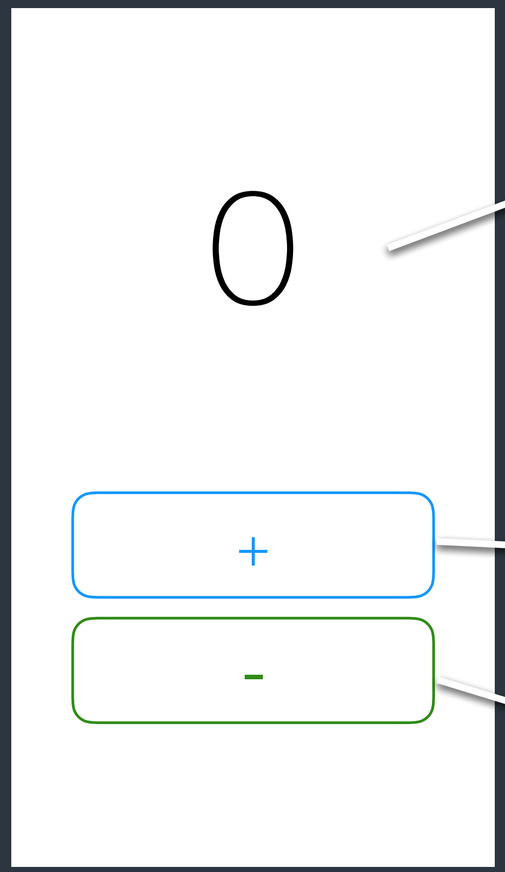
sodas@icloud.com / tctsai@nccu.edu.tw

"Your real job as a software engineer isn't to write code. It's to translate hand-wavy business requirements into detailed specs that a computer can follow. Your job is to ask questions and to find edge cases that the product people didn't think of. Your job is to help operations define processes well enough to be automated."

–Swizec Teller, in a Medium post


Start
Waiting
Equal Clicked
Save current display as number
Calculate the result
Show the result with the display
Operator Clicked
Save current display as number
Save current operation
Clean the display
Number Clicked
Display is 0?
Yes
Replace the display with the number
No
Append the number to the display
Dot Clicked
Display has dot?
No
Append the dot to the display
Yes


Calculator | Generic iOS Device
Calculator | Build Calculator: Succeeded | Yesterday at 2:04 PM
Calculator
Calculator
AppDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
CalculatorUITests
CalculatorUITests.swift
Info.plist
CalculatorCore
CalculatorCoreTests
Products
Calculator
Calculator › Calculator › Main.storyboard › Main.stor...ard (Base) › View Con...ller Scene › View Controller › View › Display Label
View Controller Scene
View Controller
Top Layout Guide
Bottom Layout Guide
View
Stack View
Display Label
Constraints
First Responder
Exit
Storyboard Entry Point
0
AC ± % ÷
7 8 9 ×
4 5 6 −
1 2 3 +
0 00 . =
View as: iPhone 7 (wC hR)
85%
Label
Text Plain
0
Color White Color
Font System Ultra Light 96.0
hC System Ultra Light 72.0
Alignment
Lines 1
Behavior Enabled
Highlighted
Baseline Align Baselines
Line Break Truncate Tail
Autoshrink Minimum Font Scale
0.5
Tighten Letter Spacing
Highlighted Default
Shadow Default
Shadow Offset 0 -1
Width Height
View
Content Mode Left
Semantic Unspecified
View Controller - A controller that manages a view.
Storyboard Reference - Provides a placeholder for a view controller in an external storyboard.
Navigation Controller - A controller that manages navigation through a hierarchy of views.
Table View Controller - A controller that manages a table view.



Outlet

Action

Action

CounterViewController.swift

```swift
import UIKit

class ViewController: UIViewController {

    @IBOutlet weak var resultLabel: UILabel!

    var currentValue = 0 {
        didSet {
            self.resultLabel.text = "\(self.currentValue)"
        }
    }

    @IBAction func increaseBtnClicked(_ sender: UIButton) {
        self.currentValue += 1
    }

    @IBAction func decreaseBtnClicked(_ sender: UIButton) {
        self.currentValue -= 1
    }

}
```

Calculator | Build Calculator: **Succeeded** | Yesterday at 2:04 PM

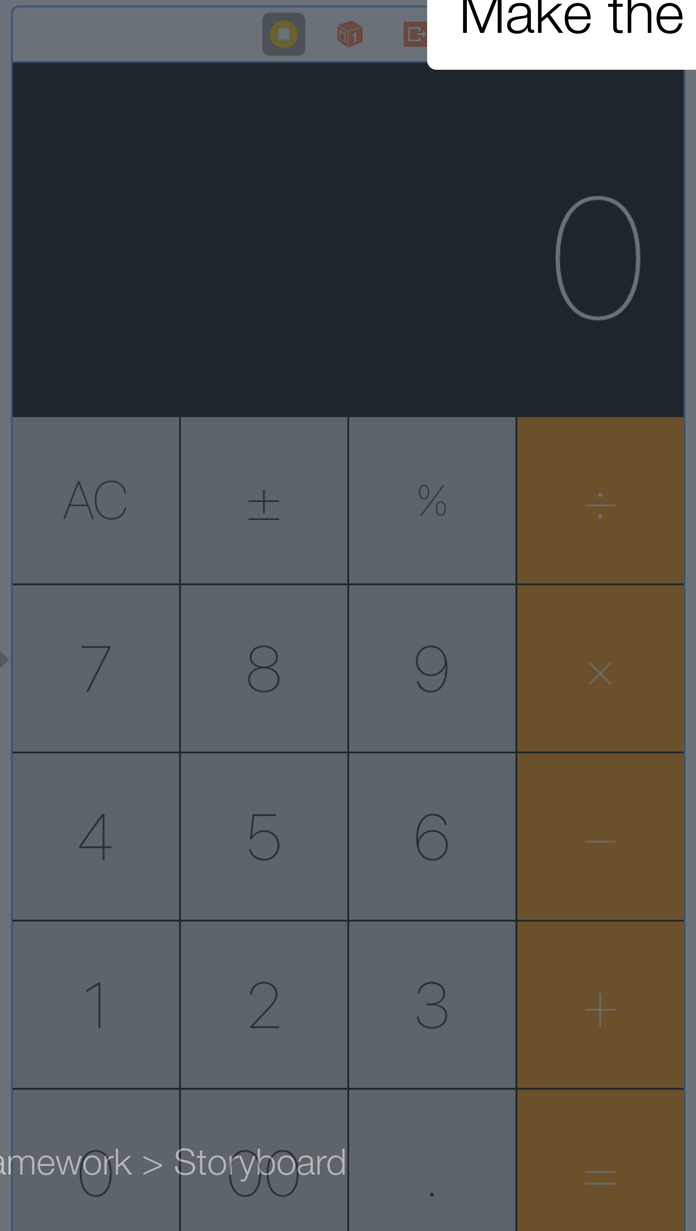
Calculator › Calculator › Main.s...board › Main.s...(Base) › View Controller Scene › View Controller

Automatic › ViewController.swift › ViewController

Make the jumpbar select Automatic.

Enable the assistant editor

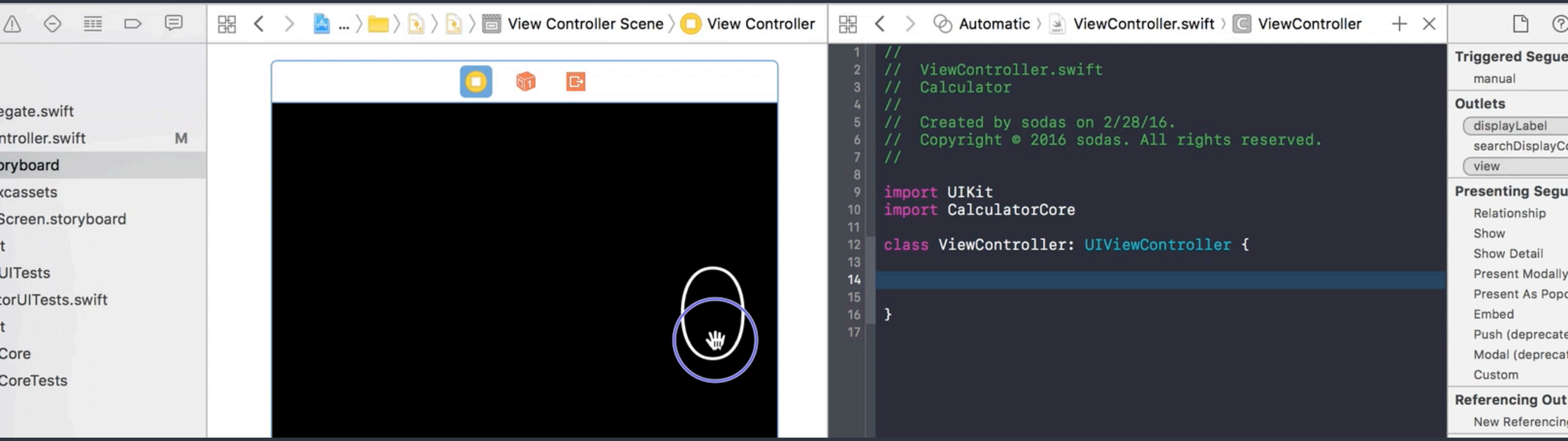
The middle one of the left group.



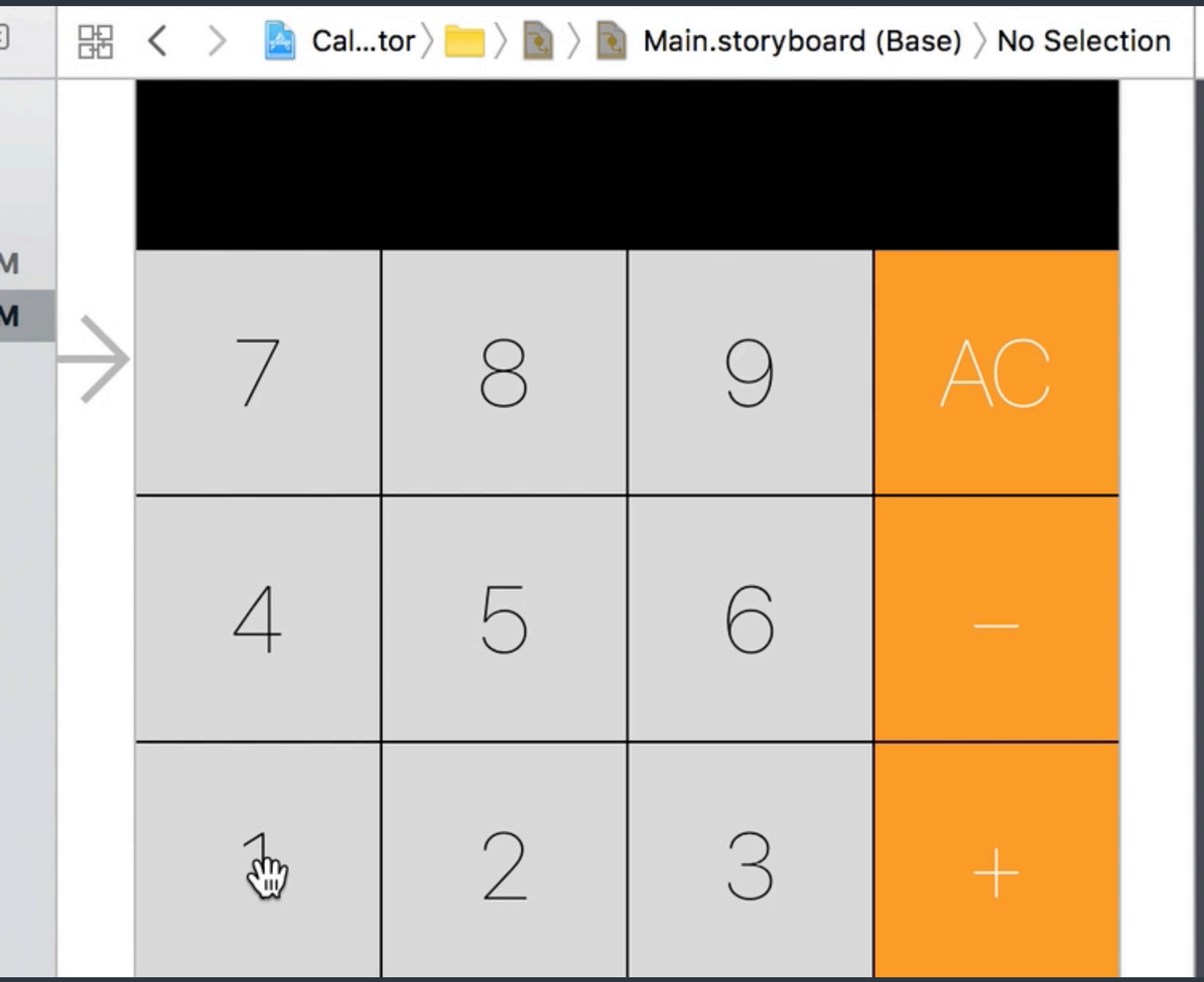
UIKit Framework > Storyboard

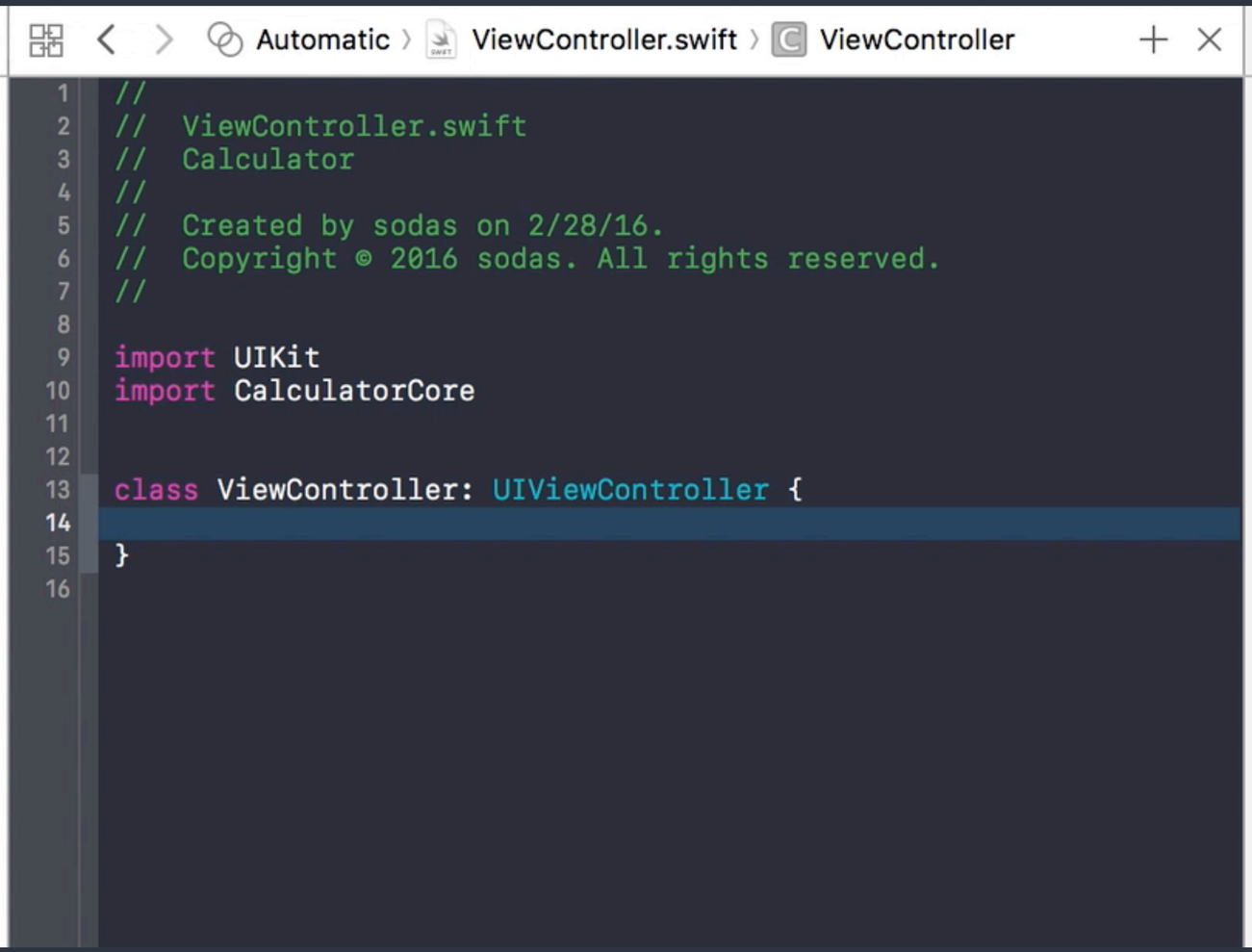
```
//
//  ViewCo ller.swift
//
//  Created by Chris Eidhof on 28/16.
//  Copyright © 2016 Chris Eidhof. All righ
//

import UIKit
import CalculatorCore

class ViewController: UIViewController {
    var core = Core<Double>()

    @IBOutlet weak var displayLabel: UILabel!

    var currentText: String {
        return self.displayLabel.text ?? "0"
    }
    var currentNumber: Double {
        return Double(self.currentText)!
    }

    // MARK: - View Controller Setup

    override var preferredStatusBarStyle: UIStatusBarStyle {
        return .lightContent
    }

    // MARK: - Input

    @IBAction func numericButtonClicked(_ sender: UIButton) {
        let currentText = self.currentText
        let buttonLabel = sender.title(for: .normal)!
        if currentText == "0" {
            self.displayLabel.text = buttonLabel
        } else {
            self.displayLabel.text = currentText + buttonLabel
        }
    }

    @IBAction func doubleZeroButtonClicked(_ sender: UIButton) {
        let currentText = self.currentText
        guard currentText != "0" else {
            return
        }
        self.displayLabel.text = currentText + "00"
    }

    @IBAction func dotButtonClicked(_ sender: Any) {
        let currentText = self.currentText
        guard !currentText.contains(".") else {
            return
        }
        self.displayLabel.text = currentText + "."
    }
```

- Use control+drag to create connections between Storyboard and Swift source code



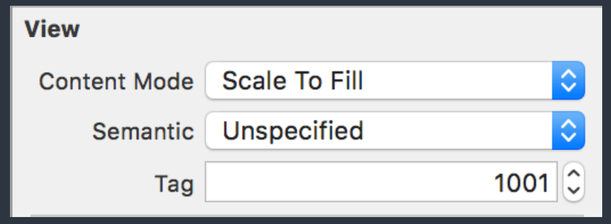
- Use control+drag to create connections between Storyboard and Swift source code

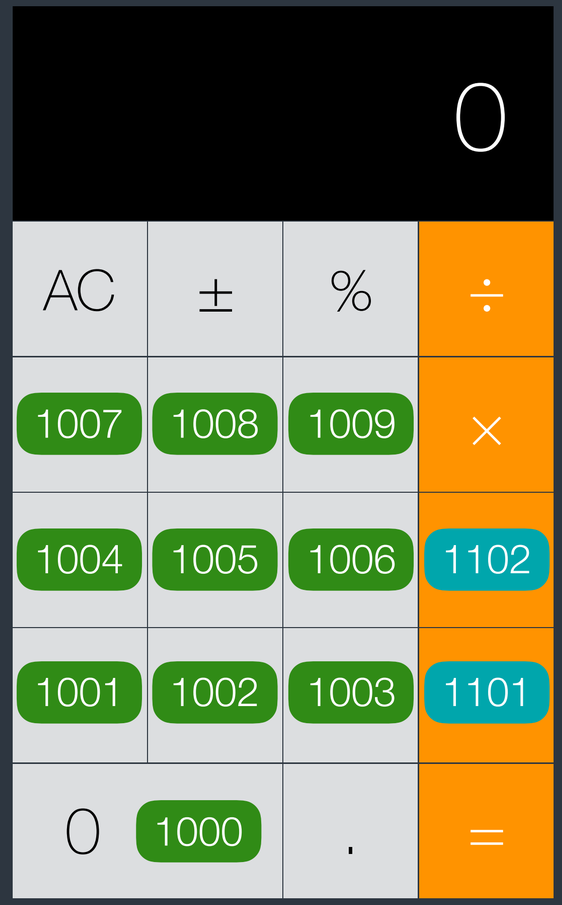




CalculatorViewController.swift

```swift
import UIKit

class ViewController: UIViewController {
    @IBOutlet weak var displayLabel: UILabel!

    @IBAction func numButtonClicked(_ sender: UIButton) {
        let buttonValue = sender.tag - 1000
        // Append `buttonValue` to the number for input.
    }

    @IBAction func opButtonClicked(_ sender: UIButton) {
        switch sender.tag {
            case 1101:  // Add
                ...
            case 1102:  // Subtract
                ...
        }
    }

    ...
}
```

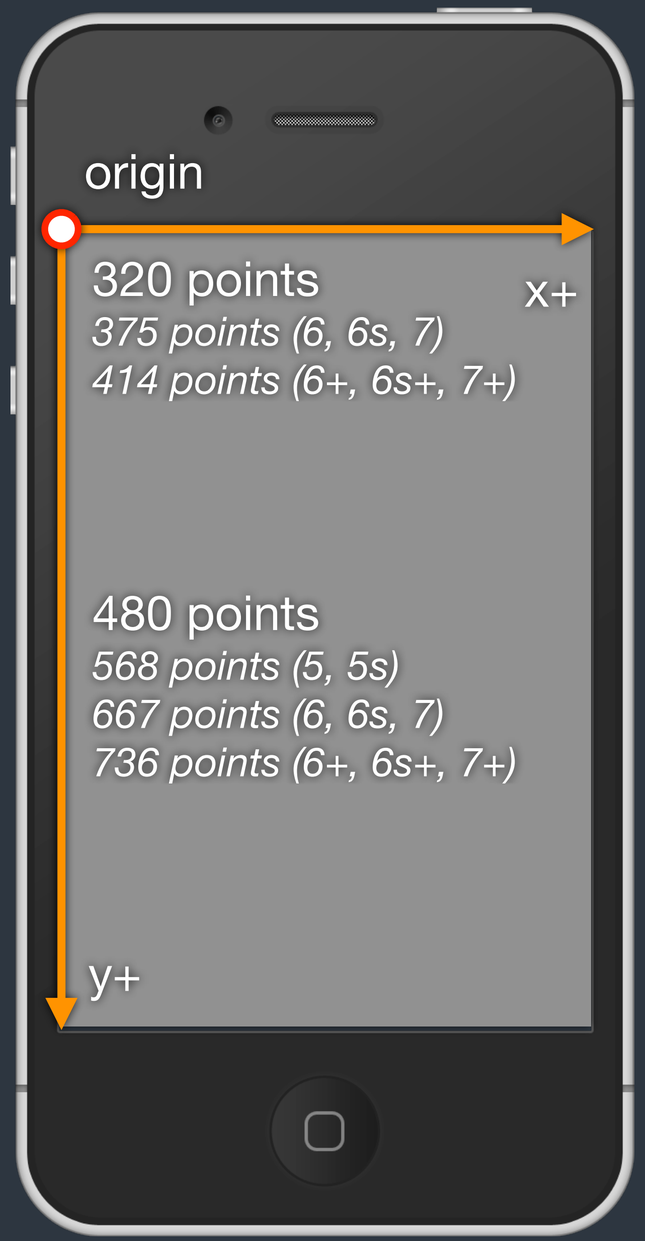




View Geometry

Coordinates System of Views

- Points are used when programming with user coordinate space.
  UIKit and CoreGraphics *objects* use “points”.
- Pixels are used when working with device coordinate space.
  Like *image drawing* and OpenGL ES.
- Points to pixels are not always 1:1.
  For iPhone 5s, 1 point equals to 2 pixels.

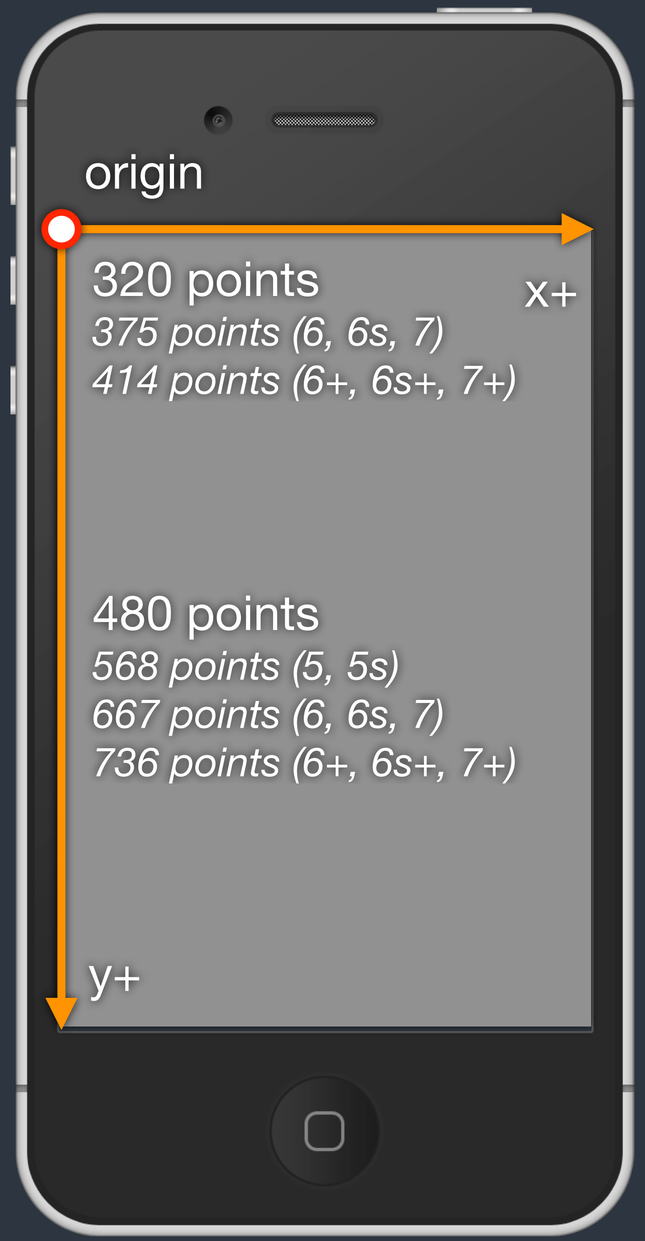


http://www.paintcodeapp.com/news/ultimate-guide-to-iphone-resolutions

Coordinates System of Views

- In UIKit, the origin is at the top-left corner of the view. x+ is from left to right and y+ is from top to bottom.
  In AppKit and CoreGraphics, the origin is at “bottom-left” and “y+” is from bottom to top.

- For iPad, the width is 768 points and the height is 1,024 points.
  For iPad Pro, it’s 1,024 points × 1,366 points.



Geometry of Views

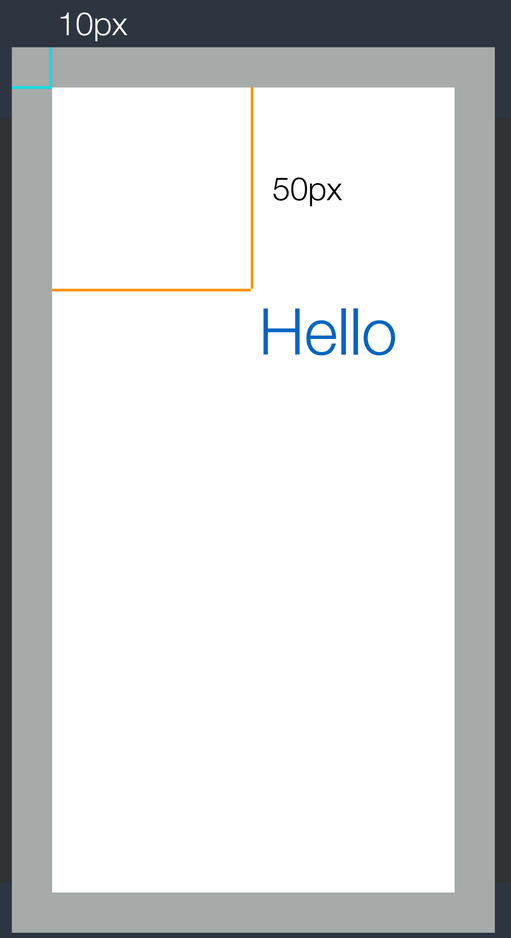
- a CGRect is a rectangle with origin and size.
  a CGPoint represents points and a CGSize is an area.

- A view has a frame and a bounds, both are CGRect.
  - Frame is in terms of the superview.
    When using a view, use frame.
  - Bounds is in terms of the view (local).
    When implementing (drawing) a view, use bounds.
    The origin of bounds is always (0, 0).

Create views by code

View Creation.swift

```swift
import UIKit

let rect = CGRect(x: 10, y: 10, width: 100, height: 200)
let view = UIView(frame: rect)
view.backgroundColor = UIColor.white

let button = UIButton(type: .system)
button.setTitle("Hello", for: .normal)
button.frame.origin = CGPoint(x: 50, y:50)
view.addSubview(button)
```



View Creation.swift

```swift
import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

        let rect = CGRect(x: 10, y: 10, width: 100, height: 200)
        let whitePanelView = UIView(frame: rect)
        whitePanelView.backgroundColor = UIColor.white
        self.view.addSubview(whitePanelView)

        let button = UIButton(type: .system)
        button.setTitle("Hello", for: .normal)
        button.frame.origin = CGPoint(x: 50, y:50)
        whitePanelView.addSubview(button)
    }
}
```

The `viewDidLoad` method would be called by the view controller when the base view is ready.
You can customize your views from here.

Target-Action Pattern

- It's a design pattern in which an object holds the information necessary to send a message *(action)* to another object *(target)* when an event occurs.
- For example, when a button *clicked* (event), it *calls a method* (action) of its *view controller* (target).
- `UIControl` class provides the capability to use this pattern. Check `UIControlEvents` enum for events.

View Creation.swift

```
import UIKit
class ViewController: UIViewController {
    override func viewDidLoad() {
        super.viewDidLoad()
        let button = UIButton(type: .system)
        button.setTitle("Hello", for: .normal)
        button.frame.origin = CGPoint(x: 50, y:50)
        self.view.addSubview(button)

        button.addTarget(self,
                         action: #selector(ViewController.buttonClicked(_:)),
                         for: .touchUpInside)
    }

    func buttonClicked(_ sender: UIButton) {
        print("Button Clicked")
    }
}
```

View Creation.swift

```swift
import UIKit
class ViewController: UIViewController {
    override func viewDidLoad() {
        super.viewDidLoad()
        let button = UIButton(type: .system)
        button.setTitle("Hello", for: .normal)
        button.frame.origin = CGPoint(x: 50, y
        self.view.addSubview(button)

        button.addTarget(self,
                         action: #selector(ViewController.buttonClicked(_:)),
                         for: .touchUpInside)
    }

    func buttonClicked(_ sender: UIButton) {
        print("Button Clicked")
    }
}
```

A `selector` is a reference to a method in a class, *it comes from Objective-C*.

When the button's event occurs, call the method of the target.
target: **`self`**
method: **`selector`**
event: **`.touchUpInside`**