iOS DeCal : Lecture 1

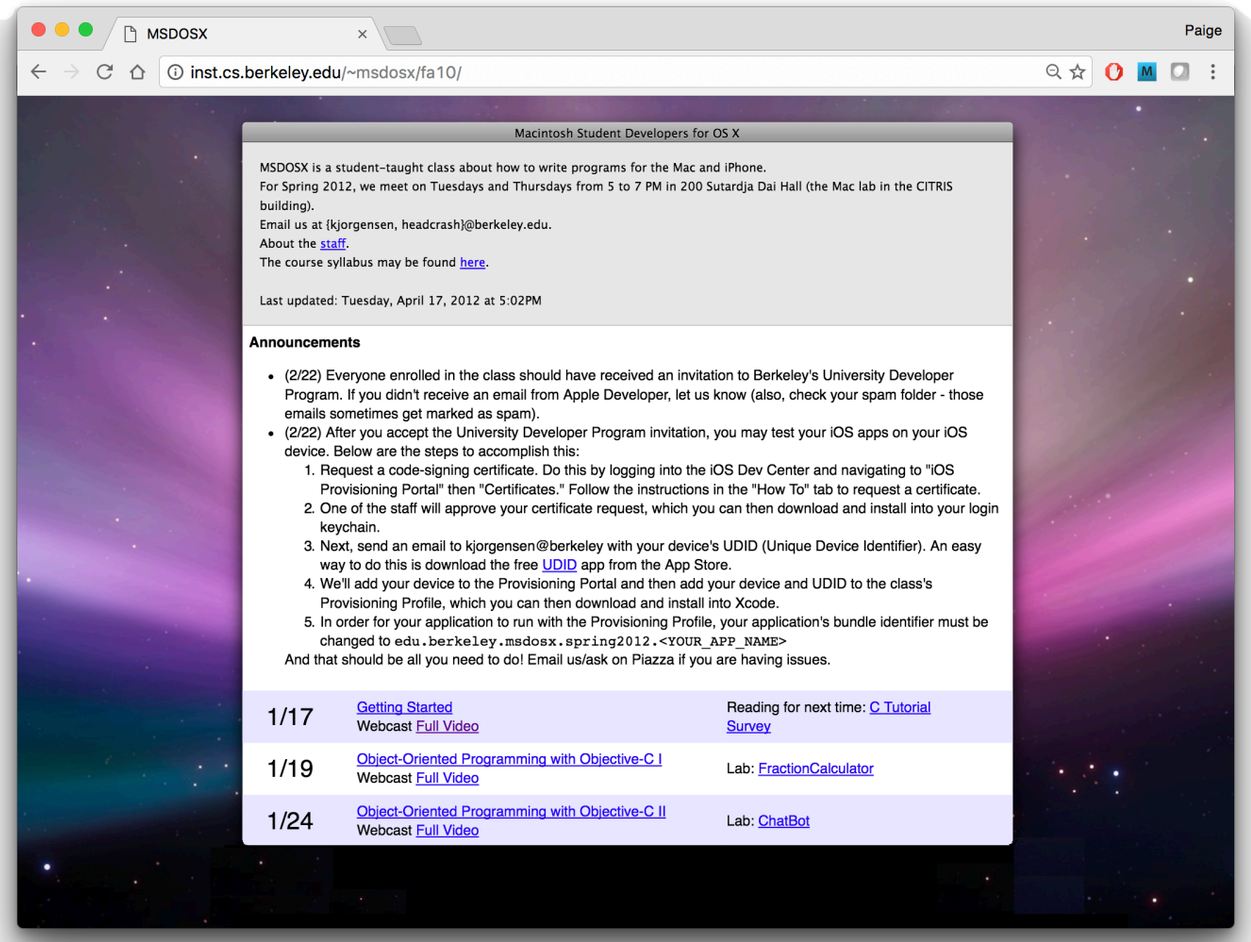
Introduction, Xcode, and Swift

Overview : Today's Lecture

1. **What is this course about?**
2. **Course Logistics**
    - Staff Introduction
    - Lecture / lab format
    - Lab Room Assignments
3. **Introduction to Xcode (+ demo)**
4. **Swift 3 Overview**

Course Background

Course History MSDOSX



2010 Macintosh Student Developers

Macintosh Student Developers for OS X

Started by Dan Garcia in early 2000s

Originated from desire for (the few) passionate Mac users to program for their own devices

Course History MSDOSX

2010 Macintosh
Student Developers

Macintosh Student Developers for OS X

Started by Dan Garcia in early 2000s

Originated from desire for (the few) passionate Mac users to program for their own devices

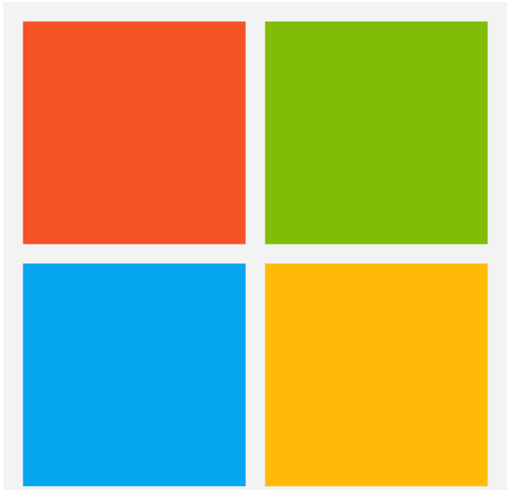
What will you learn in this course?

Swift 3
(Programming
Language)

Xcode 8
(IDE)

iOS Application
Development

iOS Experience : Past Students



Course Logistics

Course Staff : Instructors

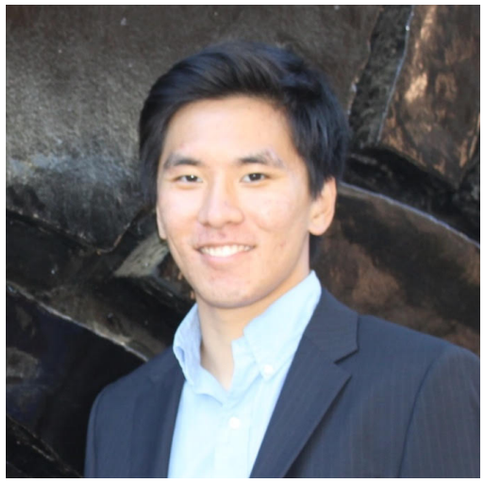
Akilesh Bapu

Paige Plander

Course Staff : Teaching Assistants

Anwar Baroudi
Head TA

Sameer Suresh
Head TA



Chan Hee Park

Maya Reddy

Matt Turk

Chris Zielinski

Maaz Uddin

Nithi Narayanan

Gera Groshev

William Smith

Class Format : Lecture / Labs

Tuesdays : 6:30 - 8:00 PM (HP Auditorium)

Approximately 1 - 1.5 hour of lecture
Attendance check-in during lecture
Optional Office Hour immediately following lecture

Thursdays : 6:30 - 8:00 PM (Soda 310 **OR** Jacobs 220)

Approximately 1 - 1.5 hour lab

- Thursday labs due the following Tuesday, in case you don't finish during the lab period
- May work with a partner
- Submit lab via TA check-off **(recommended)** or Gradescope
- Labs will typically cover the last topic of lecture

Attendance check-in during lab
Work on iOS projects with your peers and instructor guidance!

Enrollment

Rather than enrolling through CalCentral, students will be automatically enrolled (either into the class or onto waitlist)

Since we are dropping students who didn't attend this first lecture, we are waiting to enroll students until after lecture.

E-mail us ASAP or talk to us after lecture if you have a conflict or want to drop.

Workload: Labs / Projects

Labs (11)

Approximately 1.5 hours

One for each week's topic (typically last topic of the day)

Projects (2)

Hangman

Snapchat Clone

Final Project

Come up with your own idea!

Work with up to 4 other students.

Lab Room Assignments : (Thursdays)

310 Soda - (Instructor : Paige)

Lastnames beginning with A - L

220 Jacobs - (Instructor : Akilesh)

Lastnames beginning with L - Z

** Lab sections will be quite full - so we may need to move around some students

Grading : Rough Breakdown

30% Projects

Graded on a scale from 1-10 (Rubrics will be available closer to the project due date)

35% Labs

Graded on a Pass/Fail Basis

35% Final Project

Graded out of 35 based on Rubric

Cheating

This is a DeCal.

We don't have any fancy algorithms in place like other courses.

We will know though.

iOS auto-graders are difficult to make so we'll be going through projects manually during grading. There's a pretty good chance we'll be able to find out if you did copy code.

Will be dealt with on a case-by-case basis.

You get out of it what you put in.

Leaving Lecture Early

We will be alternating when check-in forms are uploaded

Labs will primarily cover the LAST topic of lecture

Leaving early will make labs take more time

If you need to leave early, make a private post on Piazza and sit nearby the doors

Links / Contact Info

Course Website - updated regularly

iosdecal.com

Piazza - ask us questions here!

piazza.com/berkeley/spring2017/cs198s17

E-mail

iosdecalstaff@gmail.com

Attendance Check-in : Lecture

There will be a check-in every lecture via Google Form

You must check in with a different person in the class each week (one form per pair).

Excused Absences: Private Post Us on Piazza

Conflicting exams only

Unexcused Absences

Students with 4+ Unexcused Absences will receive an NP for the course

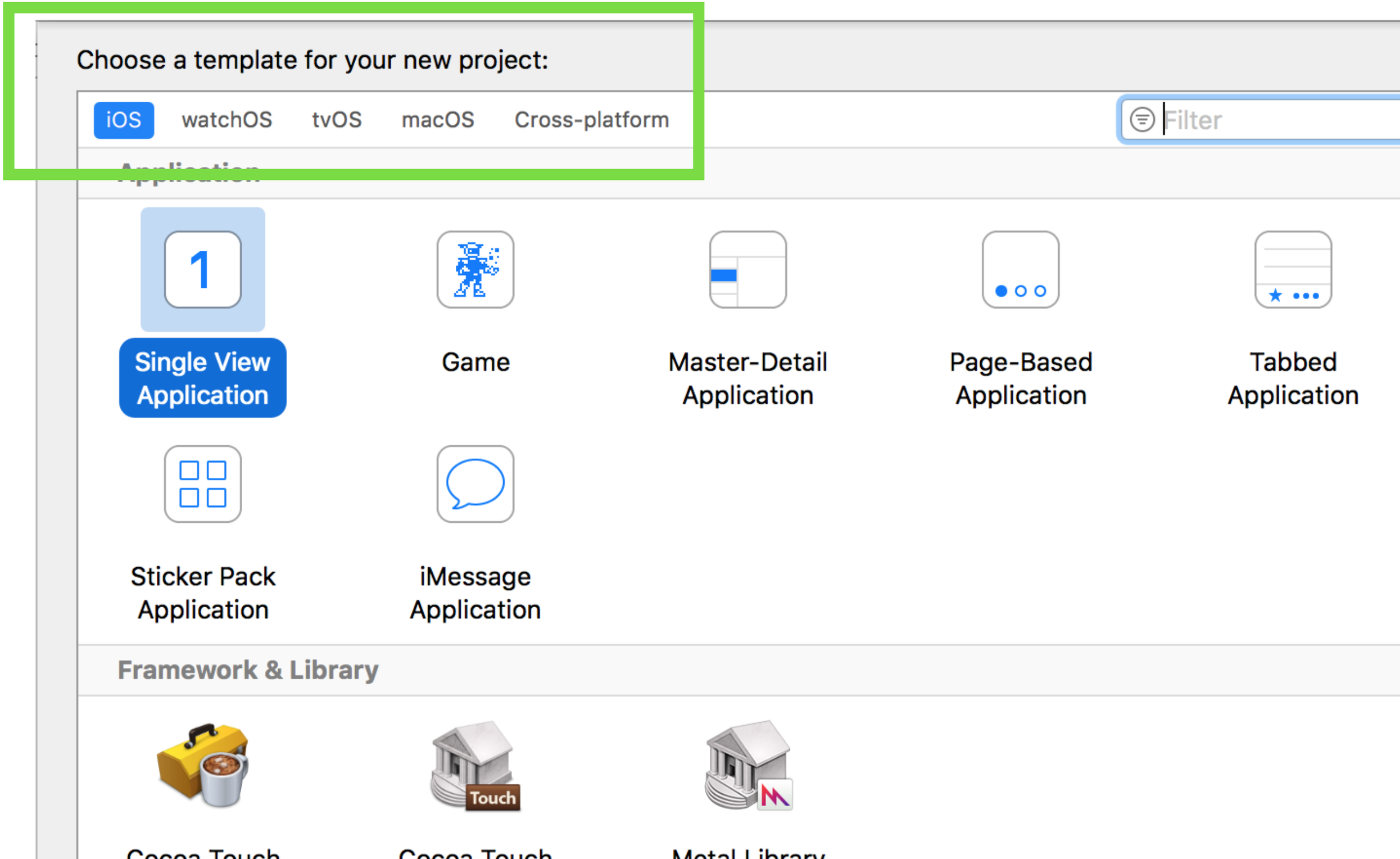
** **More info on Attendance Policy found on Piazza** **

Let’s try it now!

Introduce yourself to another student or TA, and fill out the Google Form found on our course website (**iosdecal.com**).

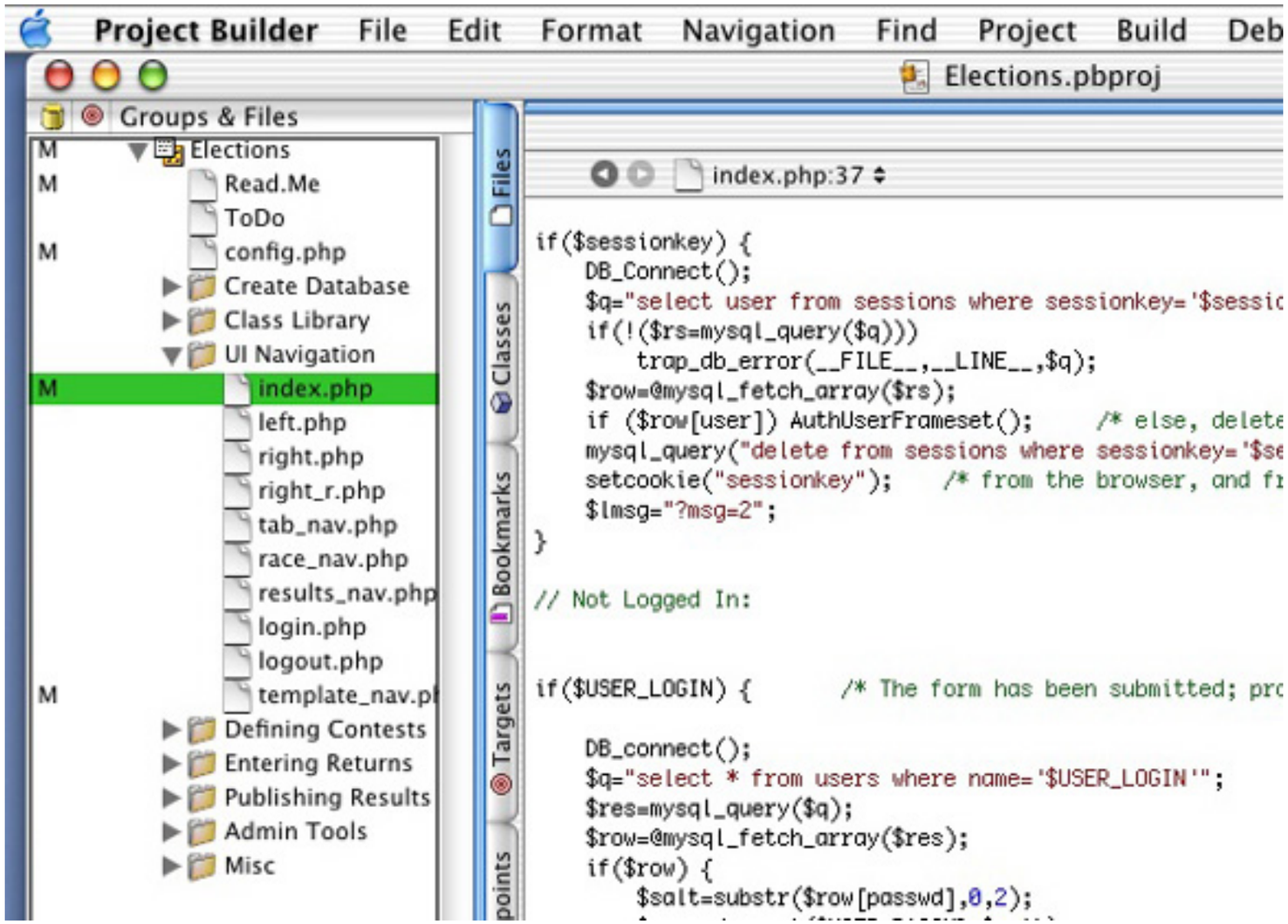
Xcode!!

The IDE to Rule Them All

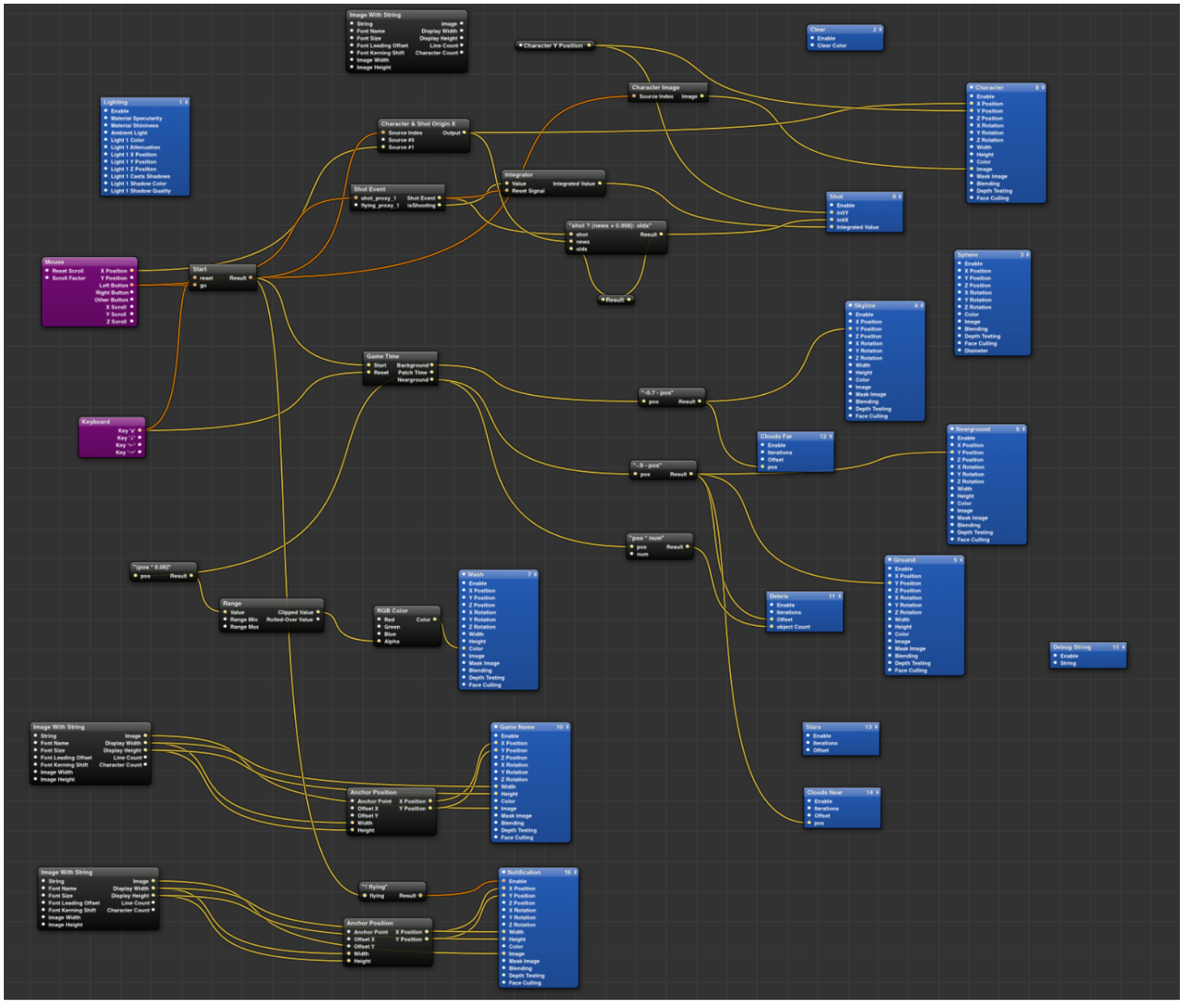


Xcode History

Project Builder

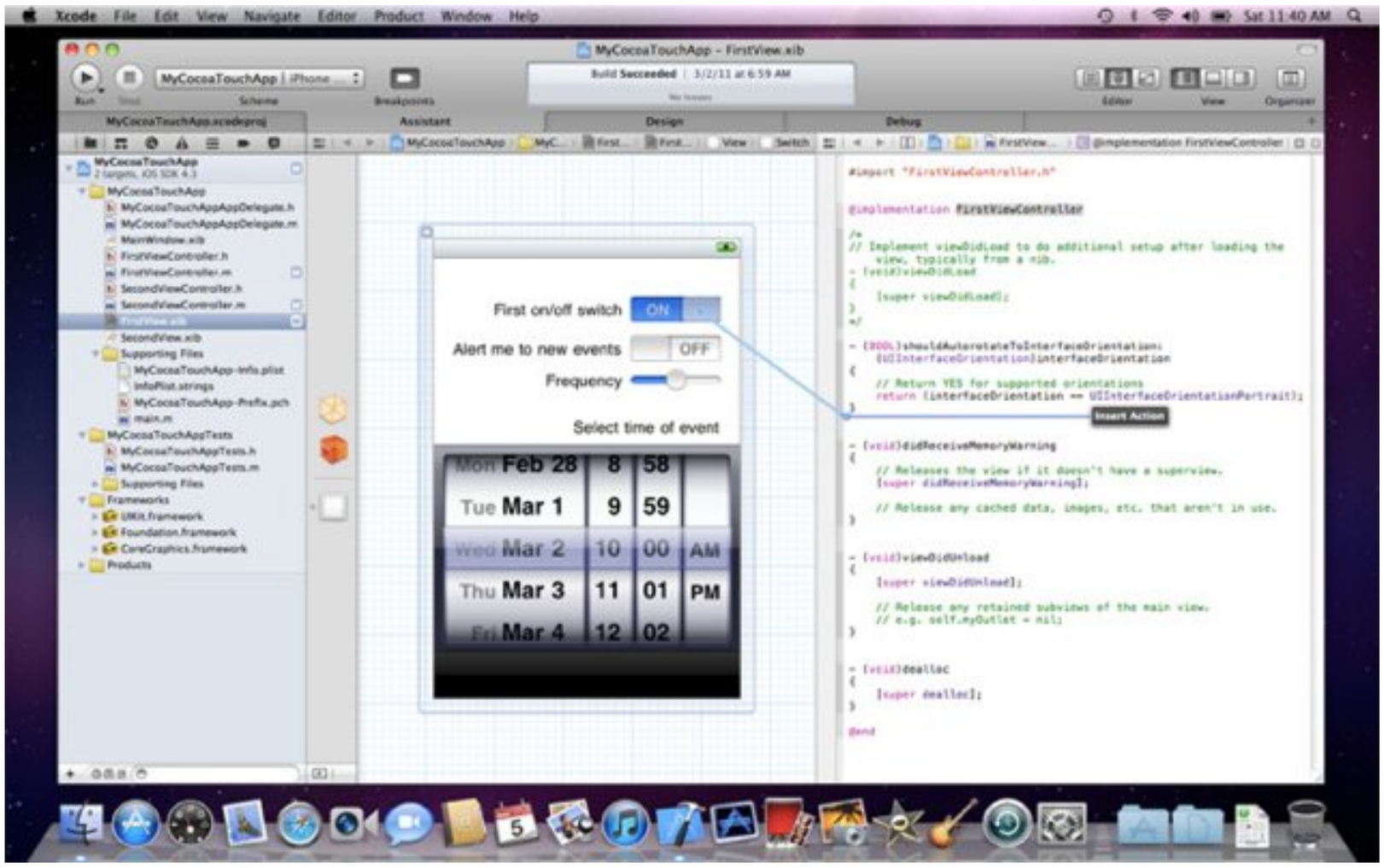


Xcode 2 - Visual Programming Language



Xcode 3 - Beginnings of iOS Dev

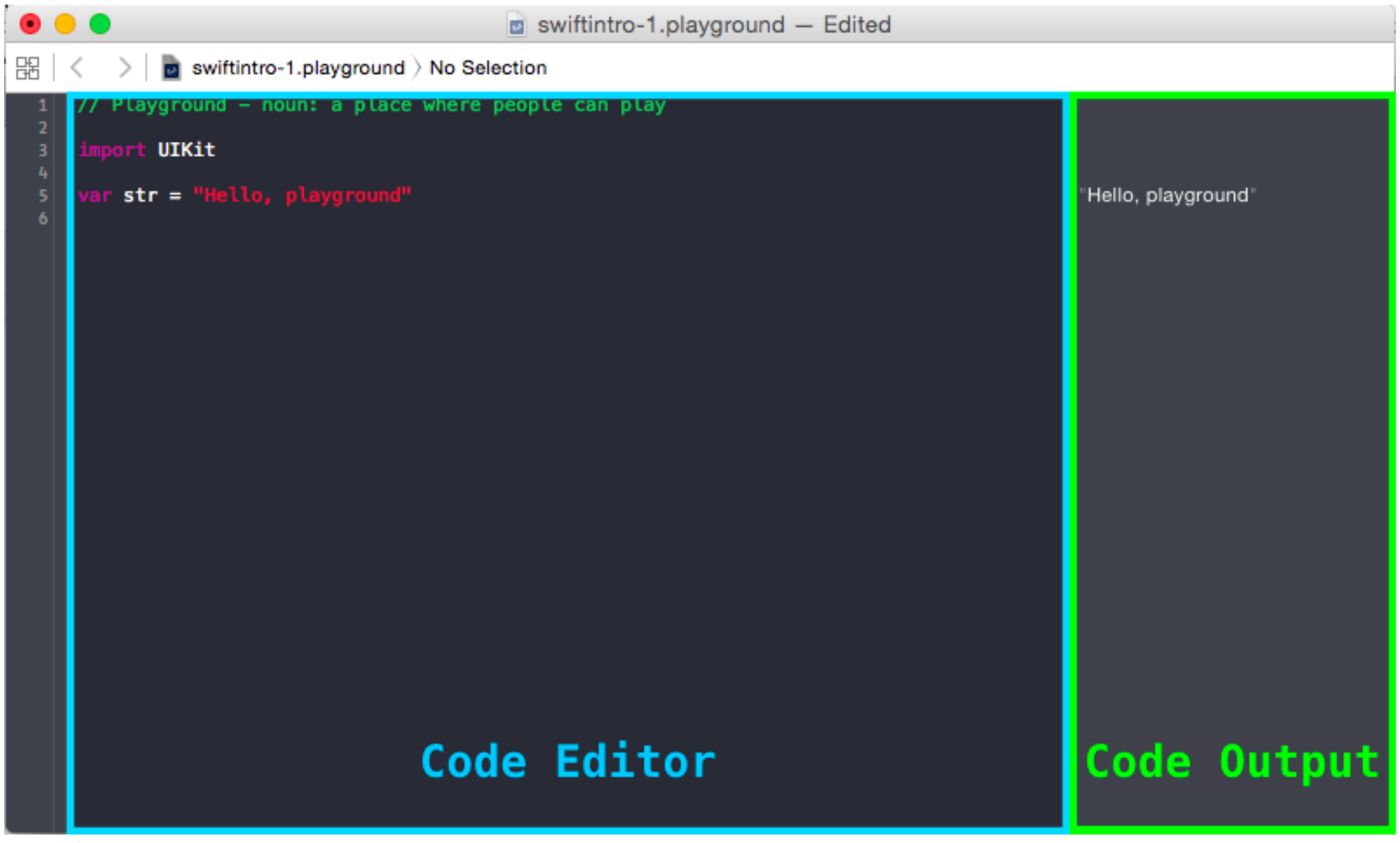
Xcode 4 - It All Comes Together (Literally)



- Brings IDE and Interface Builder together
- Seamless Interface for Creating iOS Apps
- Other than minor UI Changes, Xcode 8 (Todays version) still looks very similar to Xcode 4

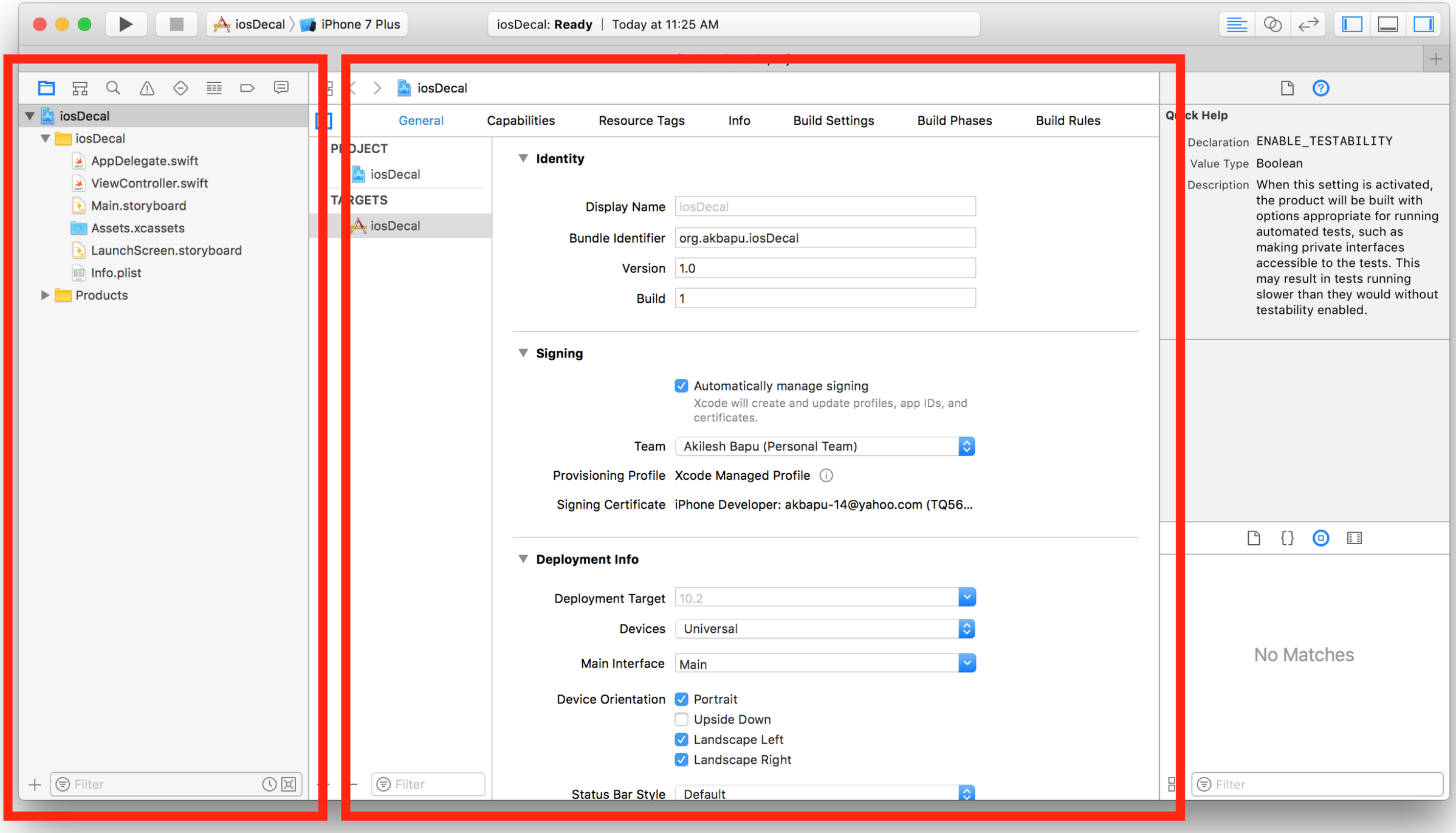
Swift Playgrounds

- Think of it as a Swift Interpreter



Xcode IDE Demo

Let's Create an App

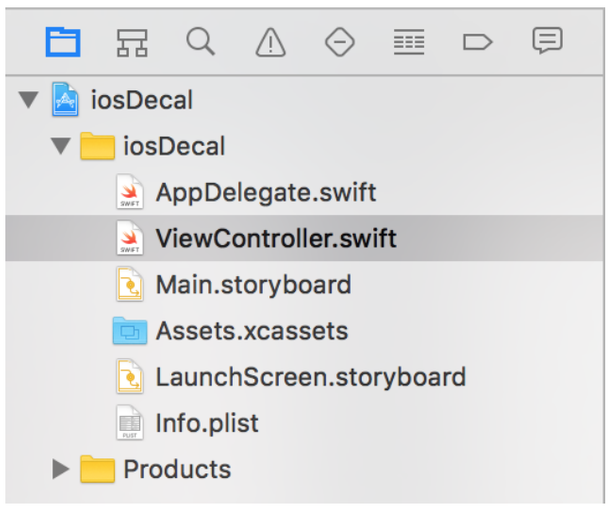


Project Navigator

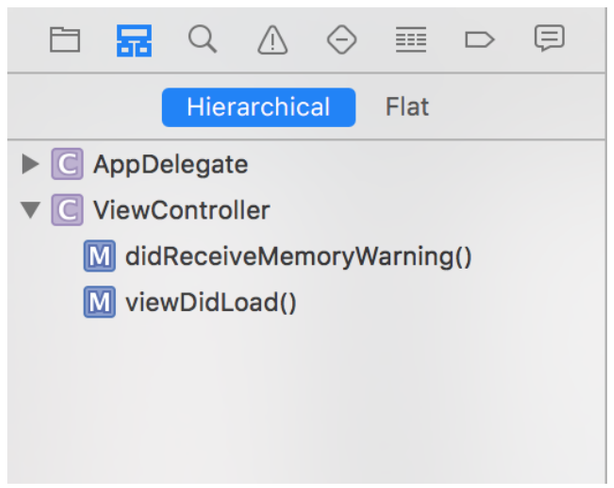
Project/Target Settings

Xcode Structure : Left Panel

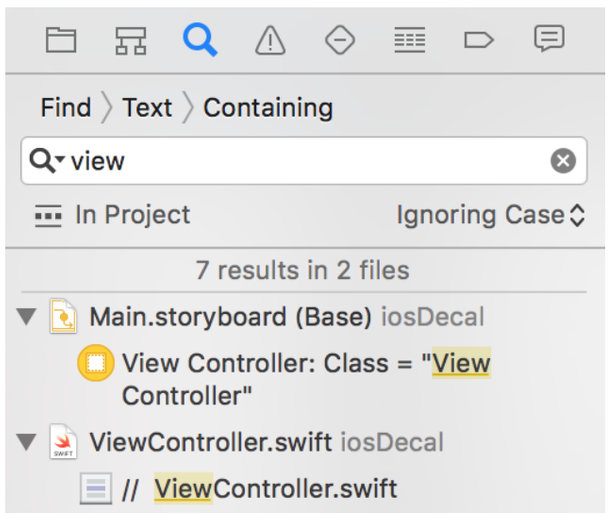
Project Navigator



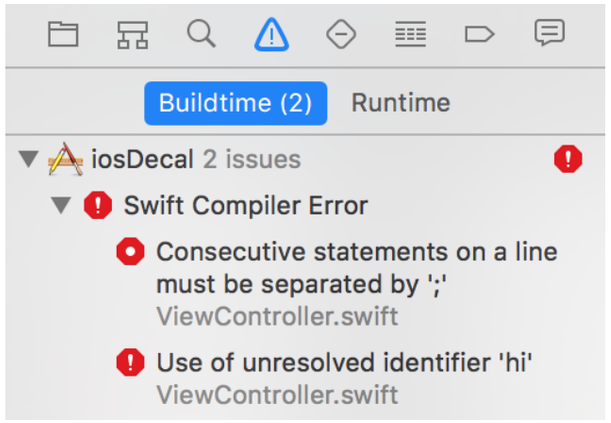
Symbol Navigator



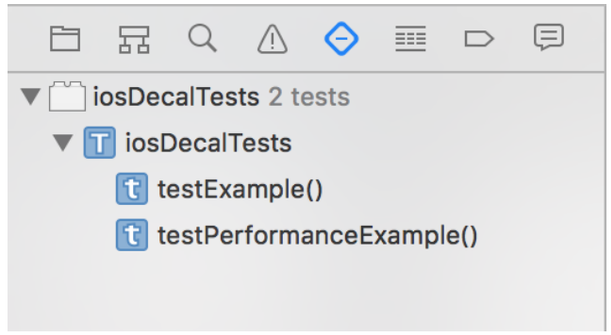
Find Navigator



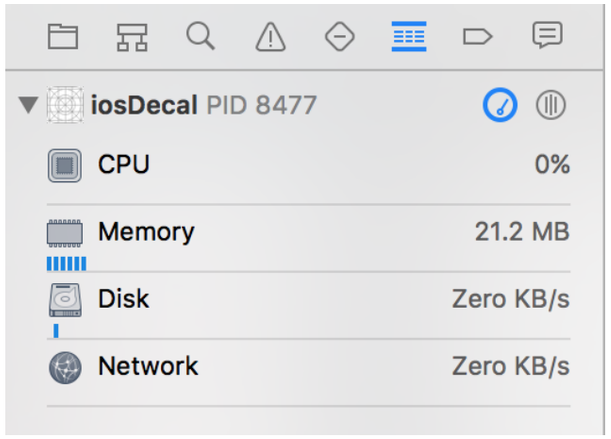
Issue Navigator



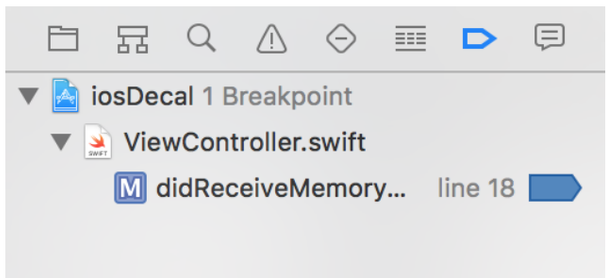
Test Navigator



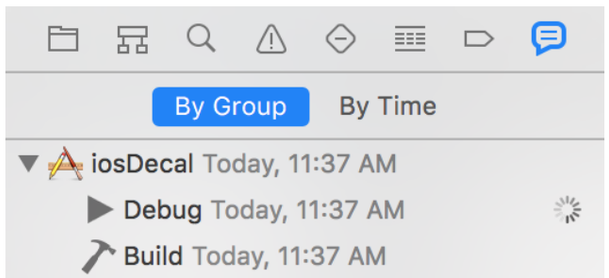
Debug Navigator



Breakpoint Navigator

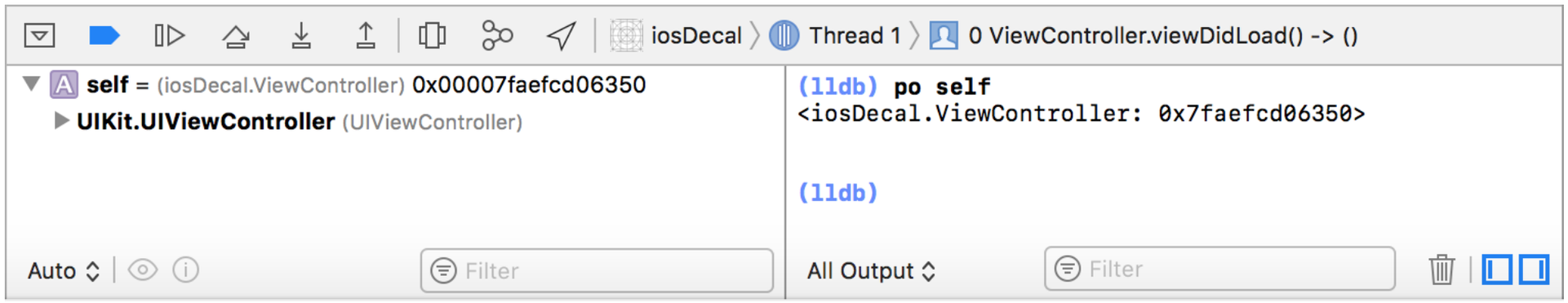


Report Navigator

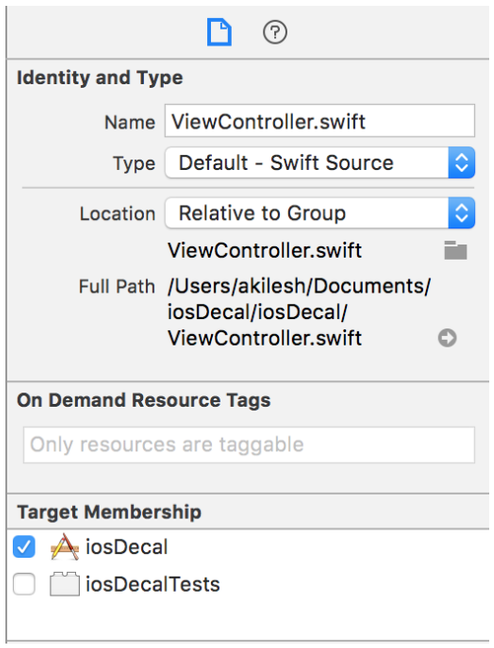


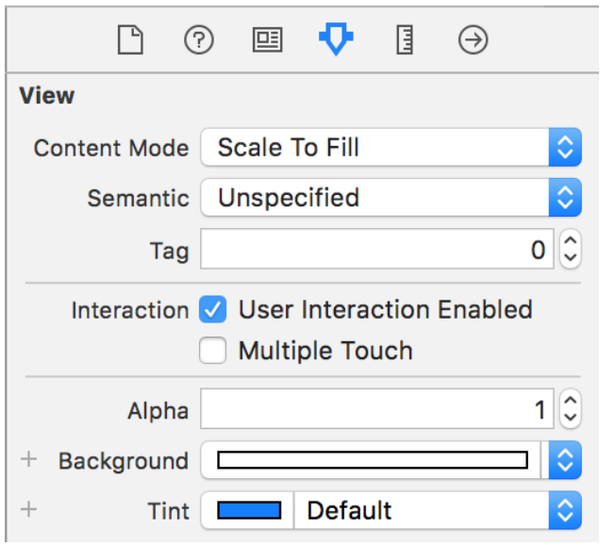
Xcode Navigator

Debug Area

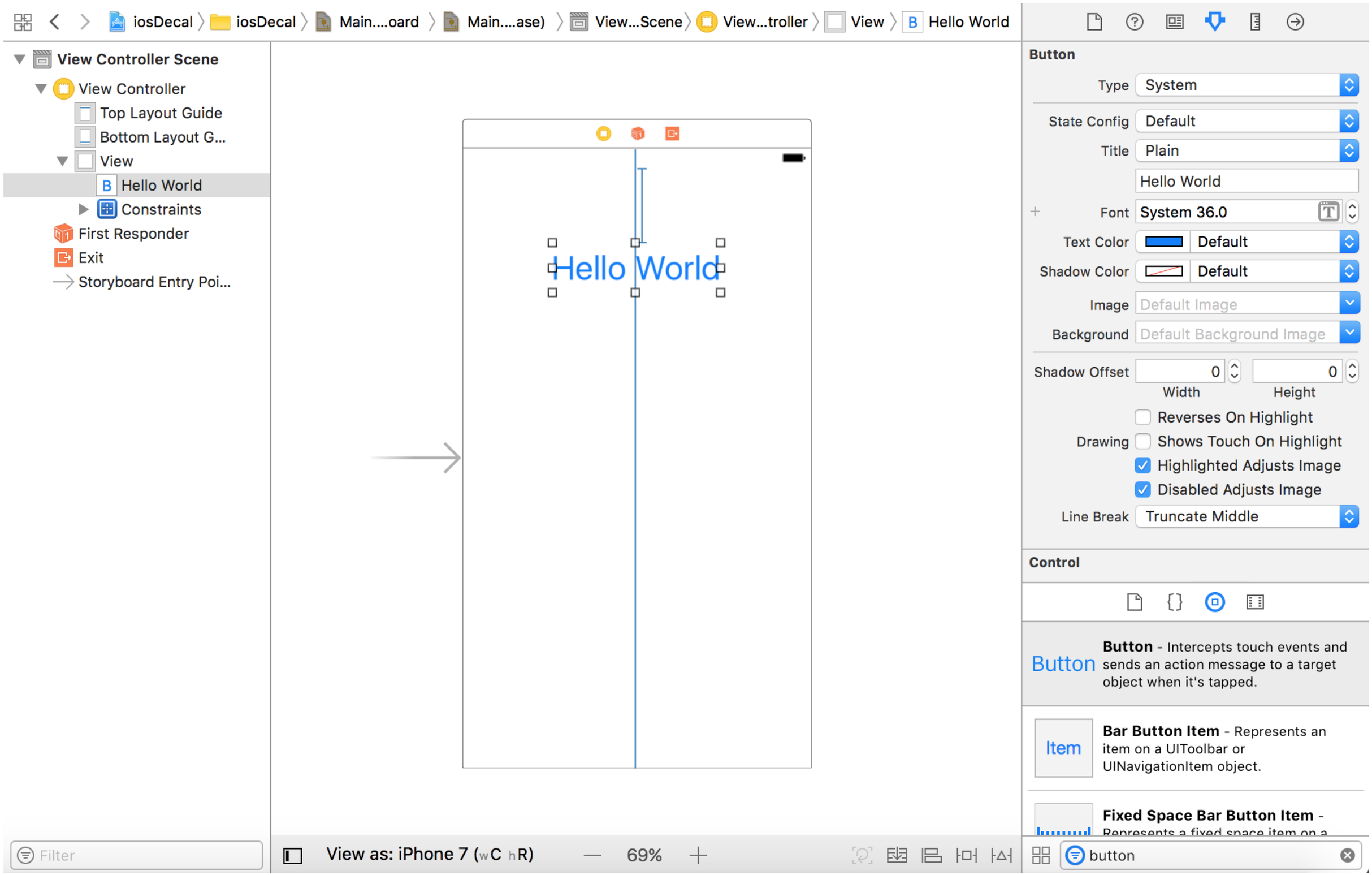


Utilities Area



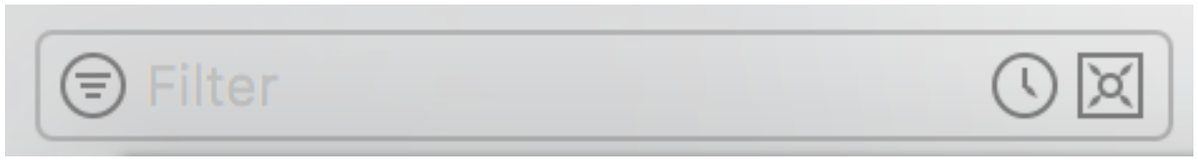


The Storyboard

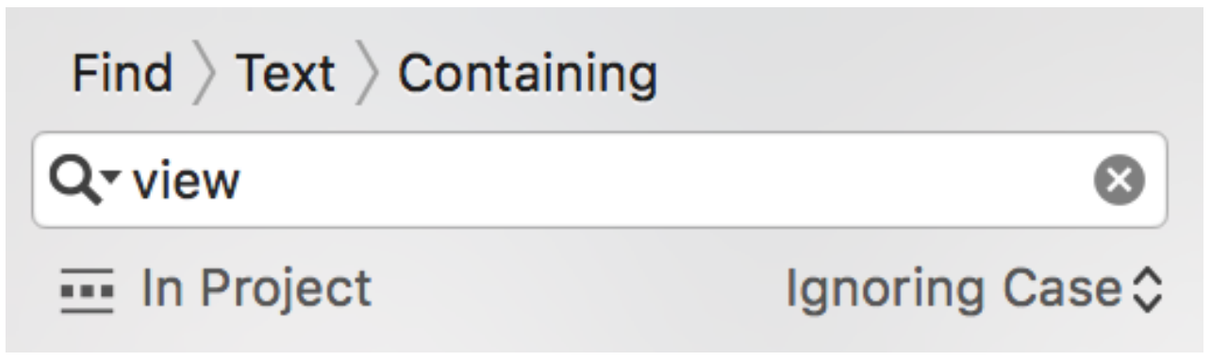


Xcode Tips and Tricks

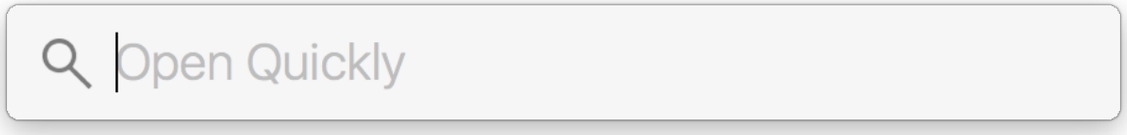
Search



Filter files by name, most recently edited, or whether it's source controlled



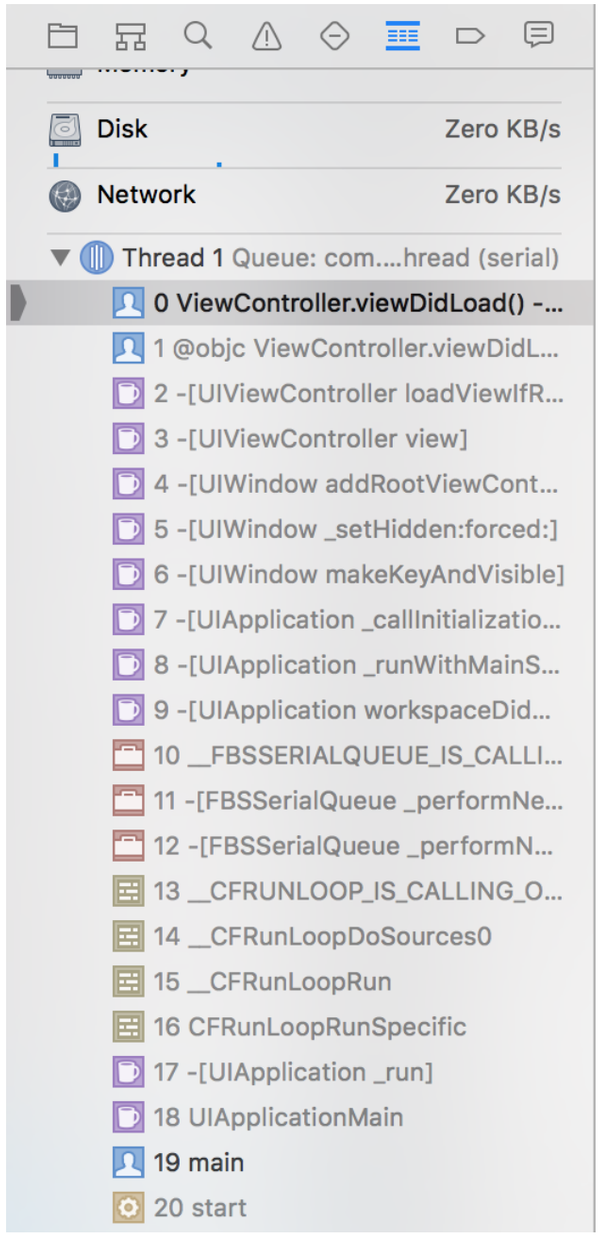
"Find" - Advanced search for strings in entire project (Can be SLOW)



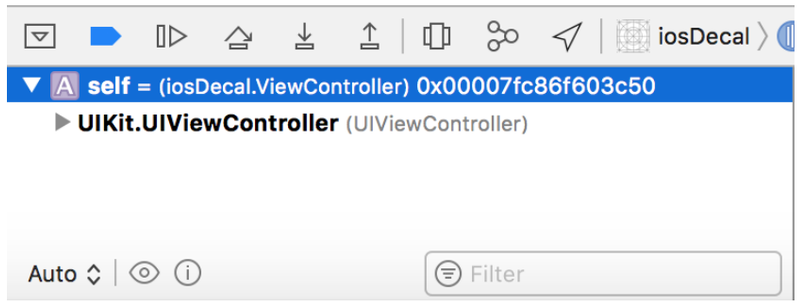
SHIFT-CMD-O - Jump to a certain file (SUPER HELPFUL and FAST)

Debugging

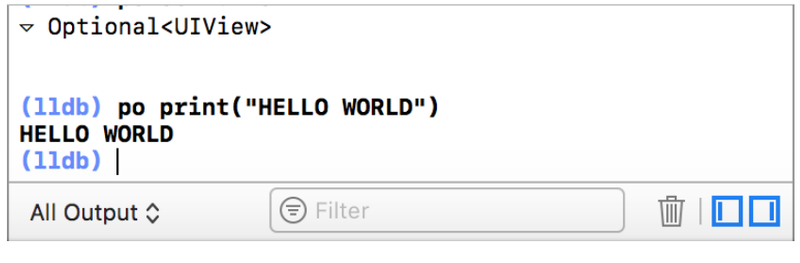
Call Hierarchy - Great if you step through code and end up at some random file and want to see how it's relevant



Currently Variables in Use - You can actually preview views

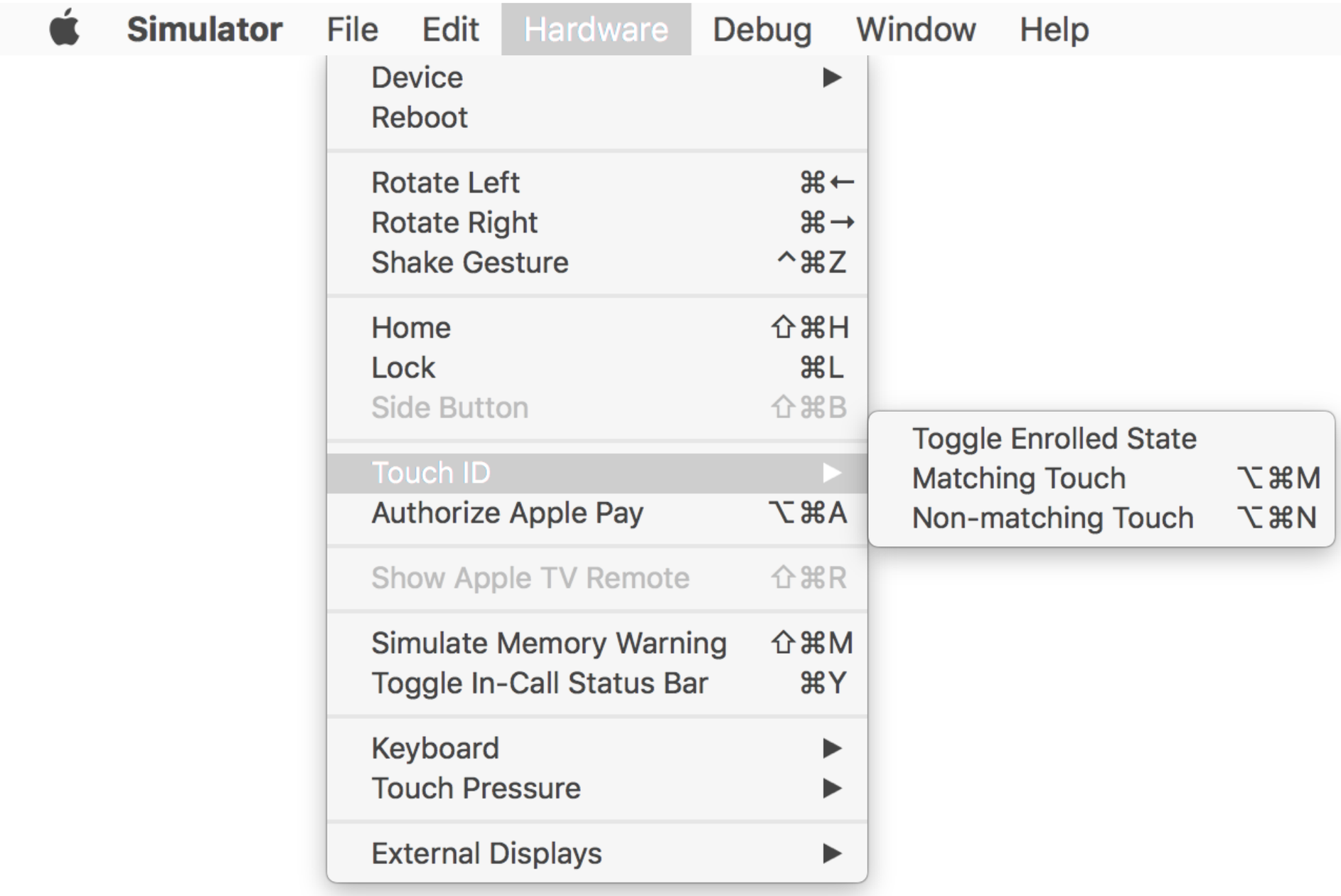


LLDB - Console Output + "po" (RUN CODE AS THE APP IS GOING)



Simulator Settings

Hardware and Debug Menus - Easily ignorable but amazingly helpful



Swift 3 Overview

Code Samples : functions

```
// Defining functions
func update(withNewData data: [String]) -> Bool {
    if data[0] == "Error" {
        return false
    }
    // ...
    return true
}

// Calling Functions
update(withNewData: ["iOS", “DeCal"])
```

Code Samples : functions

```
// Defining functions
func update(withNewData data: [String]) -> Bool {
    if data[0] == "Error" {
        return false
    }
    // ...
    return true
}

// Calling Functions
update(withNewData: ["iOS", “DeCal"])
```

Note: Internal Parameter (used in function) - `data`
External Parameter (used when calling function) - `withNewData`

Code Samples : Classes

```
class User { }

class Student: User {

    var enrolled: Bool
    let year: Int
    var favoriteDog: String?

        init(enrolled: Bool, year: Int) {
            self.enrolled = enrolled
            self.year = year
        }
}
```

Code Samples : Classes

? —> Optional Type

```
class User { }

class Student: User {

    var enrolled: Bool
    let year: Int
    var favoriteDog: String?

        init(enrolled: Bool, year: Int) {
            self.enrolled = enrolled
            self.year = year
        }
}
```

Optionals : Overview

A type that is logically allowed to have “no value”

- Able to be set to a value or `nil`

Properties of optional type are automatically initialized with a value of nil

“Unwrap” optionals with a “`!`” (Careful! If `nil` -> error)

```
var response: String? = "Hello World"
print(response)
```

Optionals : Overview

A type that is logically allowed to have “no value”

- Able to be set to a value or `nil`

Properties of optional type are automatically initialized with a value of nil

“Unwrap” optionals with a “`!`” (Careful! If `nil` -> error)

```
var response: String? = "Hello World"
print(response)
```

Console Output

```
Optional("Hello World")
```

Optionals : Overview

A type that is logically allowed to have “no value”

- Able to be set to a value or `nil`

Properties of optional type are automatically initialized with a value of nil

“Unwrap” optionals with a “`!`” (Careful! If `nil` -> error)

```
var response: String? = "Hello World"
print(response!)
```

Console Output

```
“Hello World!”
```

Optionals : Overview

A type that is logically allowed to have “no value”

- **Able to be set to a value or `nil`**

Properties of optional type are automatically initialized with a value of nil

“Unwrap” optionals with a “`!`” (Careful! If `nil` -> error)

```
var response: String?
print(response!)
```

Console Output

Optionals : Overview

A type that is logically allowed to have “no value”

- **Able to be set to a value or `nil`**

Properties of optional type are automatically initialized with a value of nil

“Unwrap” optionals with a “`!`” (Careful! If `nil` -> error)

```
var response: String?
print(response!)
```

Console Output

```
fatal error: unexpectedly found nil while unwrapping an
Optional value
```

Optionals : Usage

```
let dan = User()

func getWelcomeText(user: User) -> String? {
    if user.signedIn {
        return "Hello World!"
    }
    else {
        return nil
    }
}

if let response = getWelcomeText(user: dan) {
    print("The user responded: \(response)")
}
```

HW0 : Swift 3 Tour (Reading)

Due this Thursday (before lab)

Next Lab : Xcode/Swift Tutorial Lab

Next Lecture : MVC and AutoLayout