



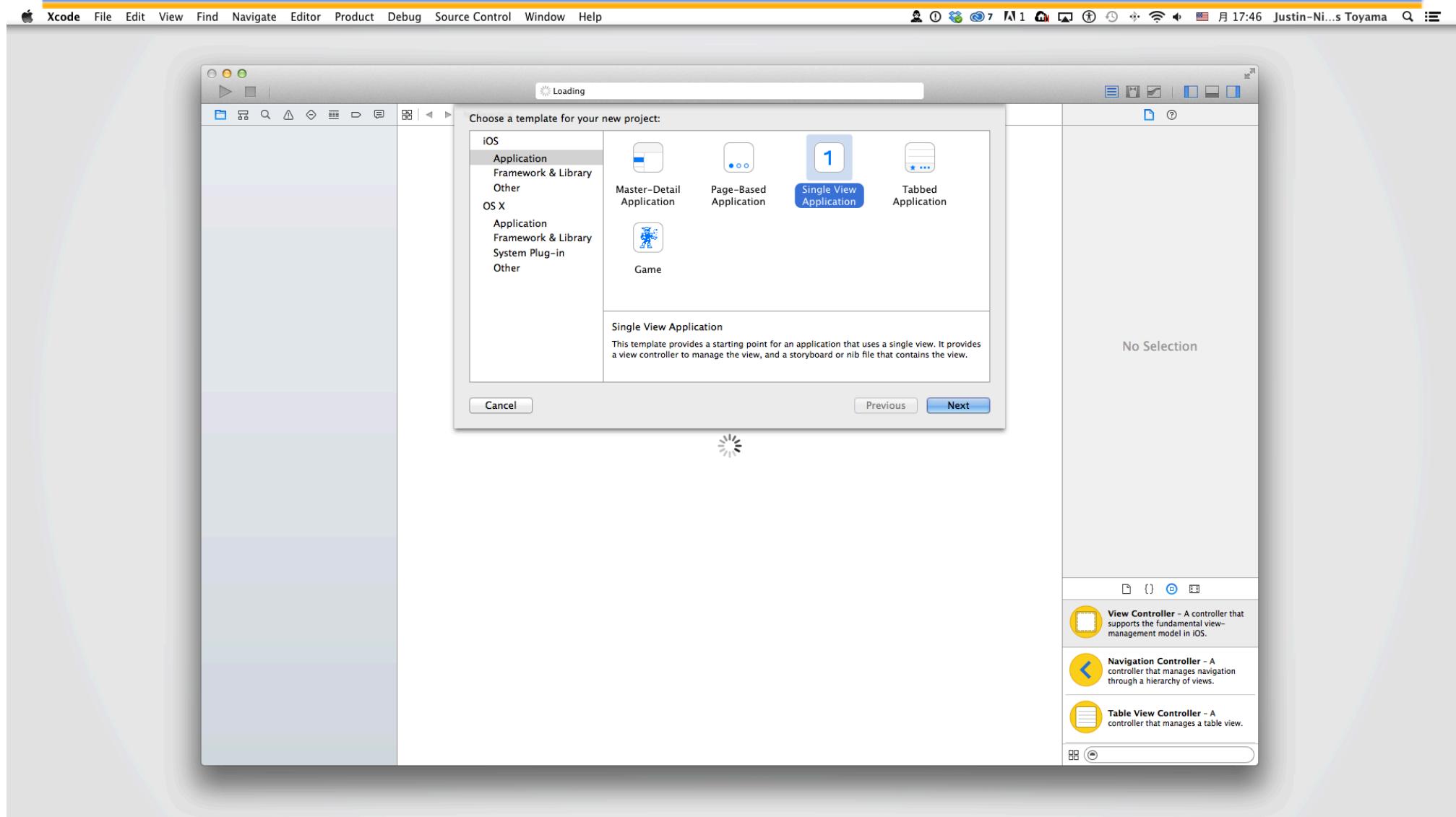
iPhone Development

Justin Toyama

xCode 6 Updates
Intro to Swift with Example

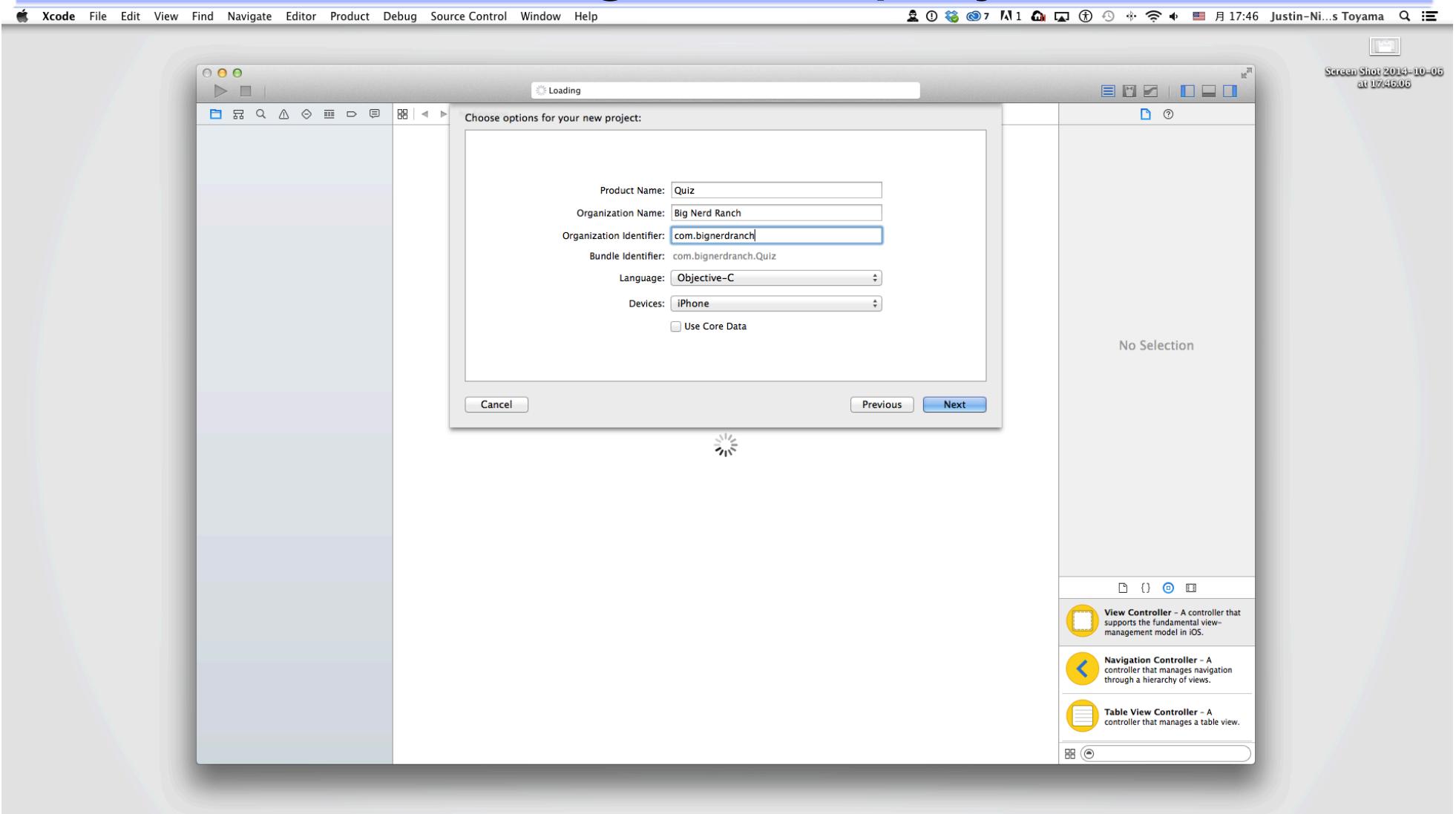


Starting a new project



Select the Single View Application Option (Empty application now in the 'Other' tab.)

Starting a new project



- Don't worry about the Class prefix (but when you create files, prefix manually)
- Select Swift

Starting a new project



Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help

Quiz: Ready | Today at 17:51

Choose a template for your new file:

iOS				
Source				
User Interface				
Core Data				
Resource				
Other				

Cocoa Touch Class
A Cocoa Touch class.

Cancel Previous Next

Identity and Type

- Name: ViewController.m
- Type: Default - Objective-C So...
- Location: Relative to Group
- View Controller.m
- Full Path: /Users/justin/Dropbox/UCI Extension/2014 Fall intro/Quiz/Quiz/ViewController.m

Target Membership

- Quiz
- QuizTests

Text Settings

- Text Encoding: Default - Unicode (UTF-8)
- Line Endings: Default - OS X / Unix (LF)
- Indent Using: Spaces
- Widths: Tab 4 Indent 4
- Wrap lines

Source Control

- Repository --
- Type --
- Current Branch --
- Version --
- Status: No changes
- Location

View Controller - A controller that supports the fundamental view-management model in iOS.

Navigation Controller - A controller that manages navigation through a hierarchy of views.

Table View Controller - A controller that manages a table view.

File > New... > Cocoa Touch Class

Starting a new project



Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help

Quiz: Ready | Today at 17:51

Screen Shot 2014-10-06 at 17:52:00

Choose options for your new file:

Class: TestClass
 Subclass of: NSObject
 Also create XIB file
 iPhone
 Language: Objective-C

Cancel Previous Next

Identity and Type
 Name: ViewController.m
 Type: Default - Objective-C So...
 Location: Relative to Group
 ViewController.m
 Full Path: /Users/justin/Dropbox/UCI Extension/2014 Fall intro/Quiz/Quiz/ViewController.m

Target Membership
 Quiz
 QuizTests

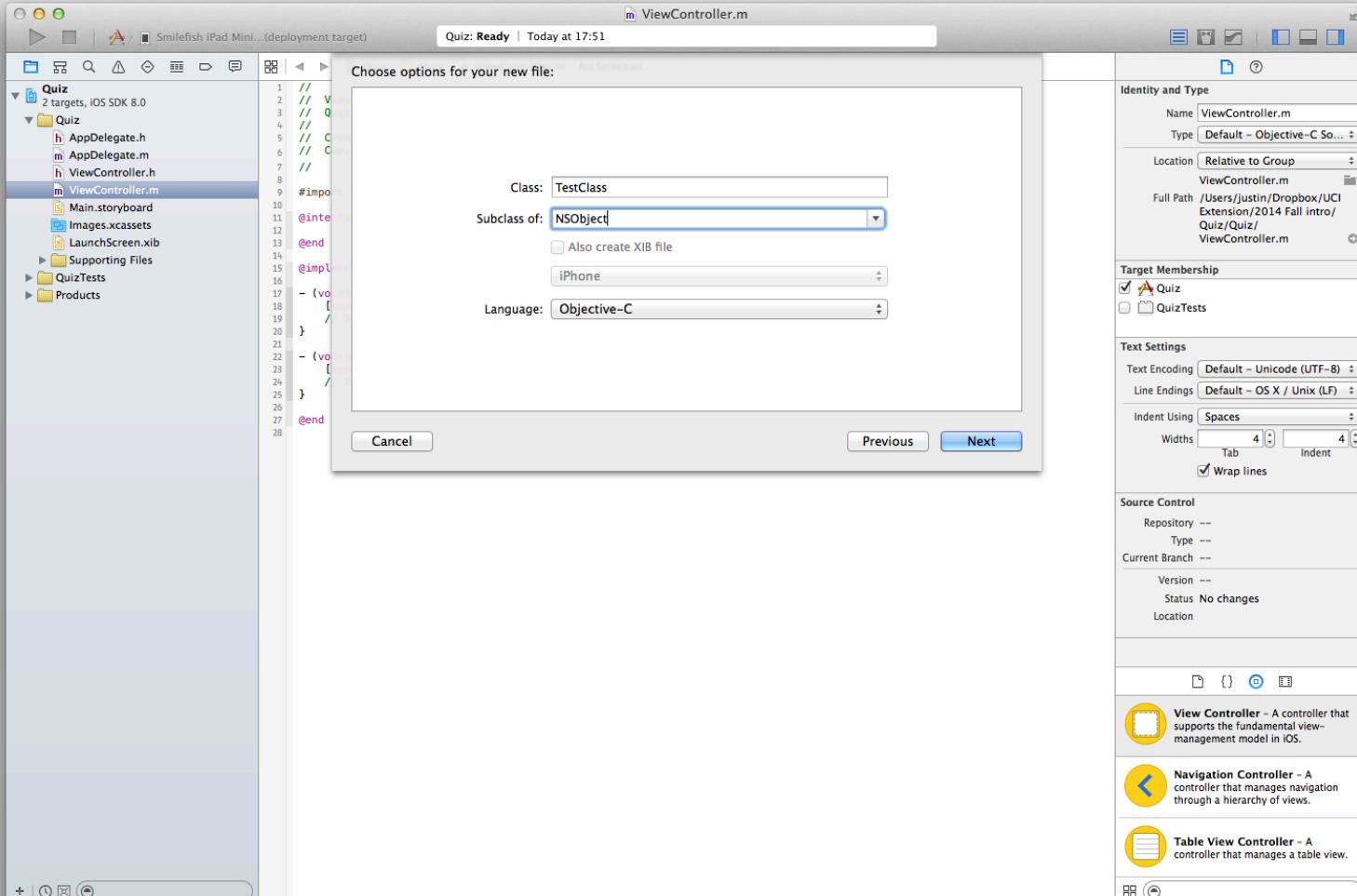
Text Settings
 Text Encoding: Default - Unicode (UTF-8)
 Line Endings: Default - OS X / Unix (LF)
 Indent Using: Spaces
 Widths: Tab 4 Indent 4
 Wrap lines

Source Control
 Repository --
 Type --
 Current Branch --
 Version --
 Status: No changes
 Location

View Controller - A controller that supports the fundamental view-management model in iOS.

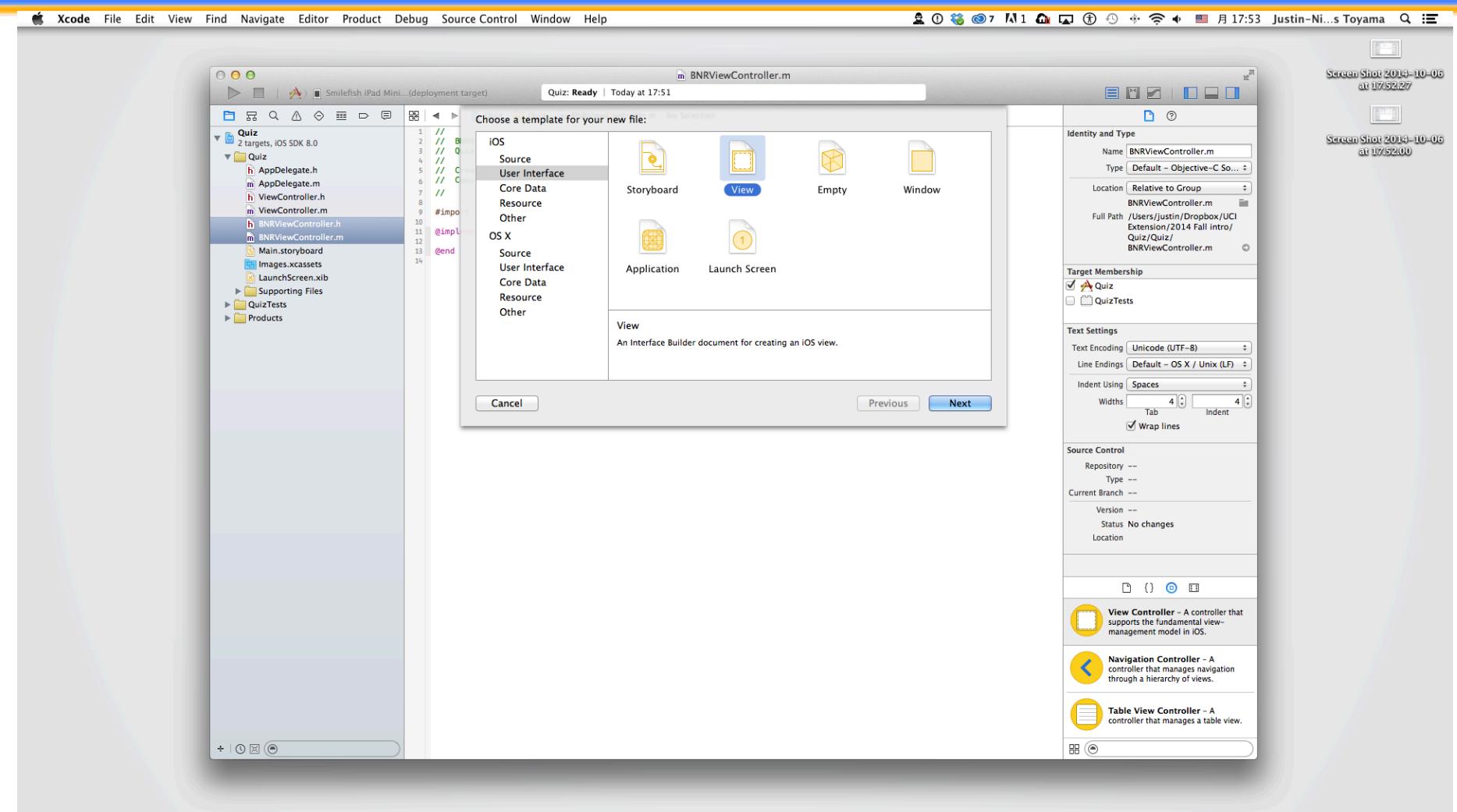
Navigation Controller - A controller that manages navigation through a hierarchy of views.

Table View Controller - A controller that manages a table view.



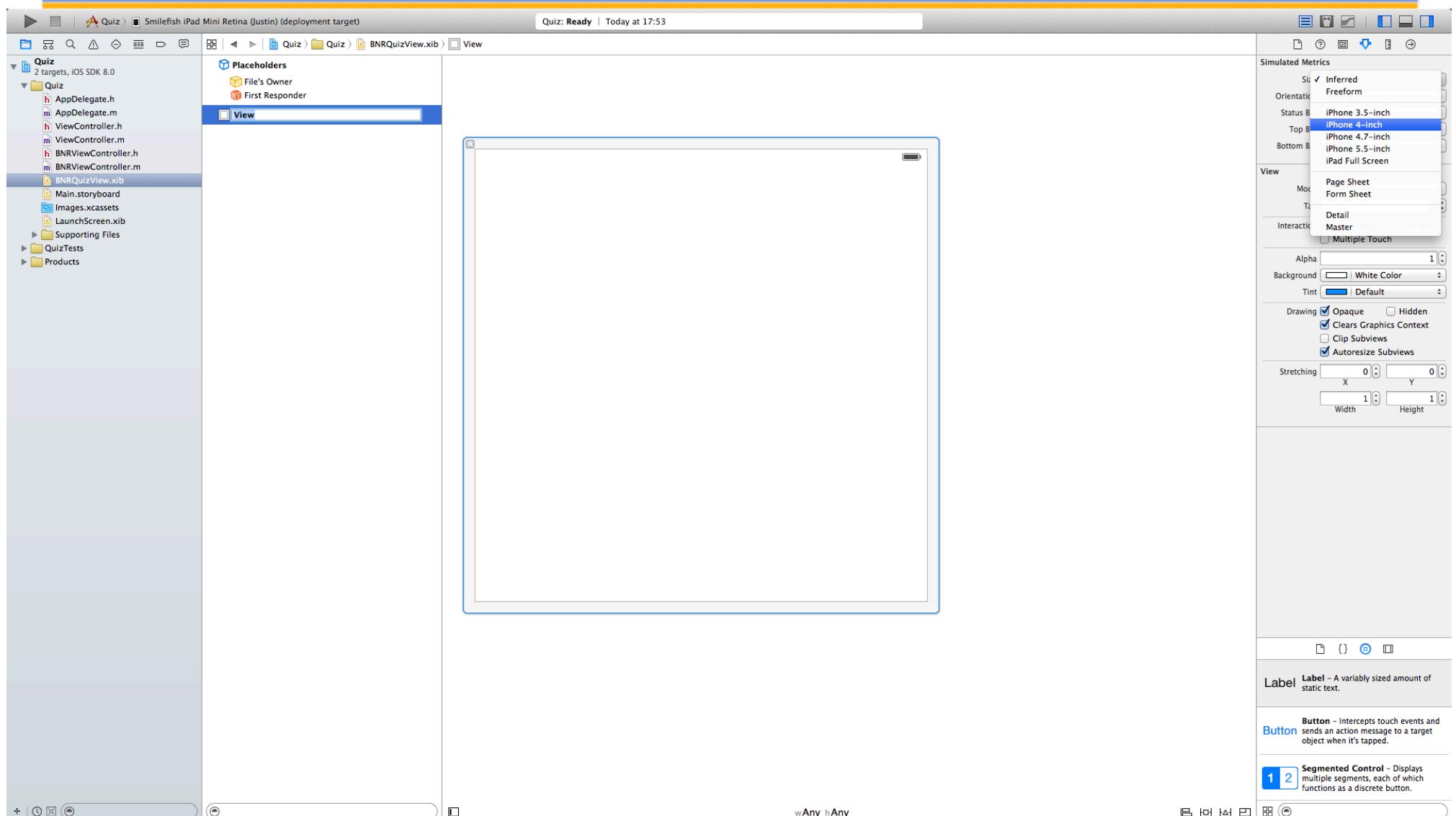
Set the base class (aka “the class from which to inherit” aka “subclass of:”)

Adding a view



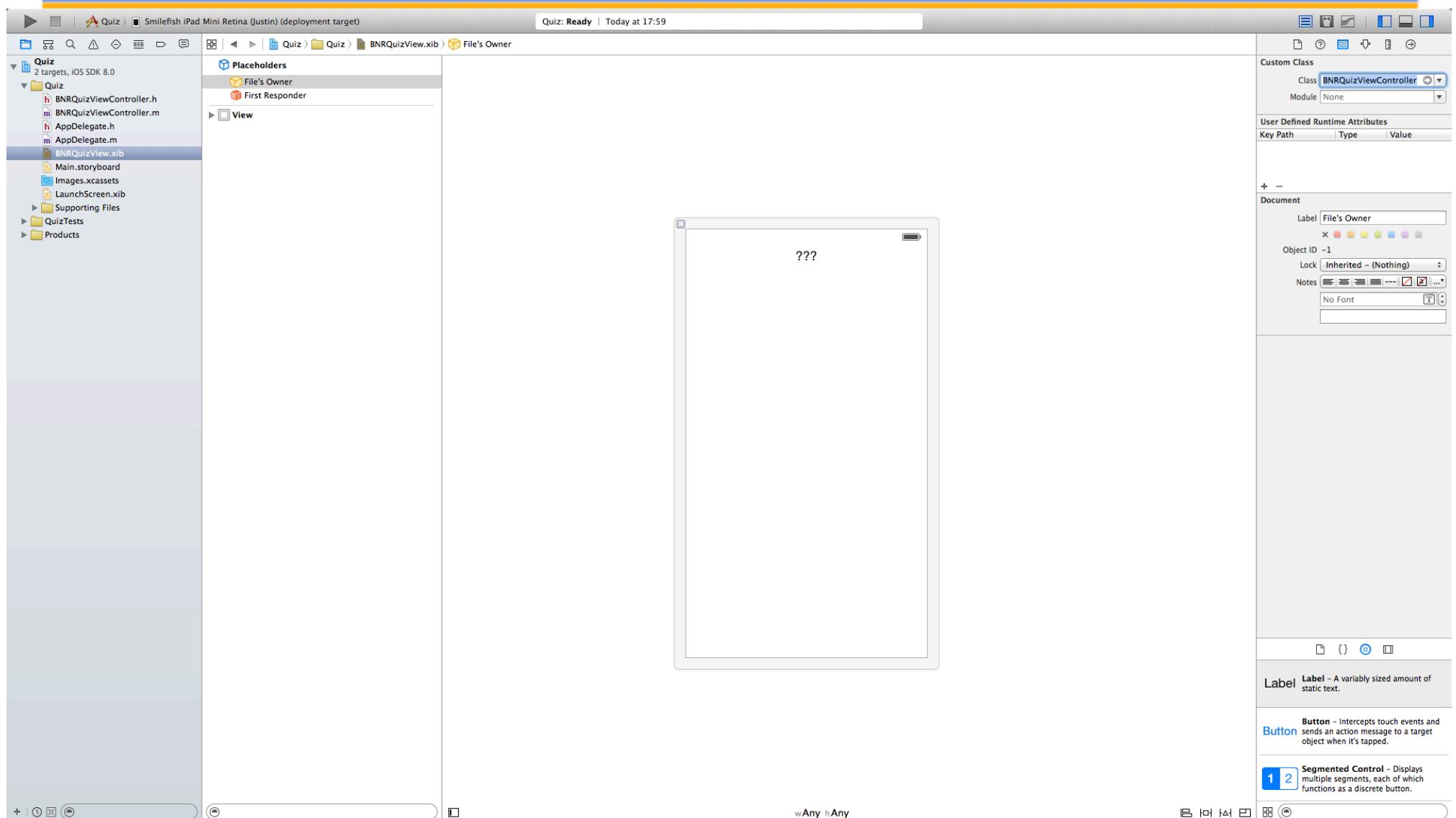
- Select File > New > View.
- Link to custom subclass

Starting a new project



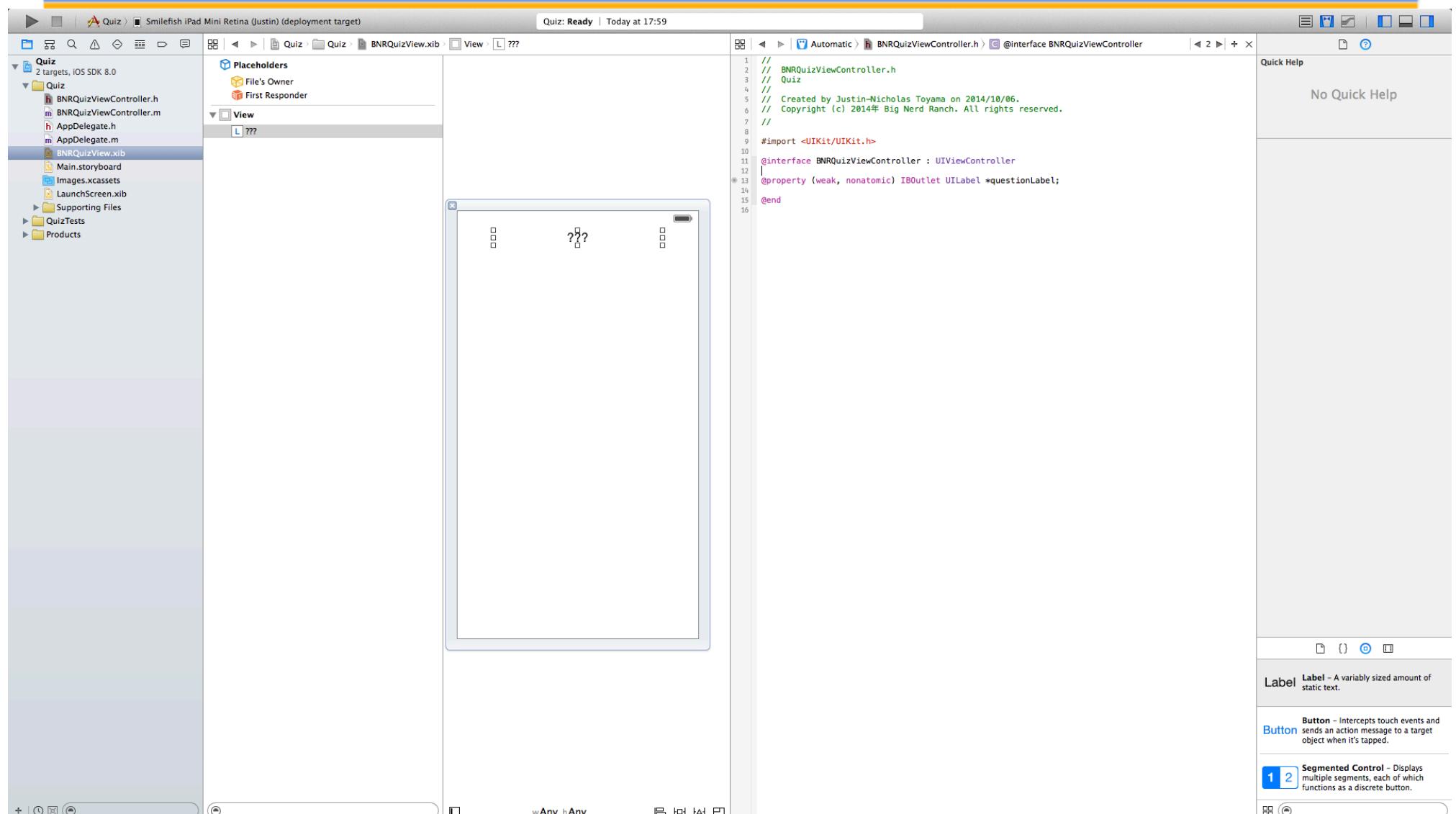
- Be sure to set the Simulated Size (but we will eventually use auto layout...)

Starting a new project



Select the File's Owner object and declare the custom subclass

Creating XIBs

The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure for "Quiz" with targets "Quiz" and "Smilefish iPad Mini Retina (Justin)".
- XIB Editor:** Displays "BNRQuizView.xib" for the iPad Mini Retina target. The view is currently empty, showing a placeholder question mark icon.
- Code Editor:** Shows the header file "BNRQuizViewController.h" with the following code:

```

1 // BNRQuizViewController.h
2 // Quiz
3 //
4 // Created by Justin-Nicholas Toyama on 2014/10/06.
5 // Copyright (c) 2014年 Big Nerd Ranch. All rights reserved.
6 //
7 //
8 #import <UIKit/UIKit.h>
9
10 @interface BNRQuizViewController : UIViewController
11 {
12     @property (weak, nonatomic) IBOutlet UILabel *questionLabel;
13 }
14
15 @end
16 
```
- Assistant Editor:** Shows the "Quick Help" panel for the "Label" component, which defines it as "A variably sized amount of static text".
- Sidebar:** Lists UI components: Label, Button, and Segmented Control.

Now you can link up your UI to your code

- Get Xcode 6 from the Mac Appstore
 - Allows you to code and compile Swift code
 - Backwards compatible with Obj-C projects
 - Introduces Playgrounds – see your code output “live” without building or running
- Swift Resources
 - The Swift Programming Language (iBooks) (read the tour)
 - Also online (free):
https://developer.apple.com/library/prerelease/ios/documentation/Swift/Conceptual/Swift_Programming_Language/index.html
 - Using Swift with Cocoa and Objective-C (iBooks)
 - Also online
(free):https://developer.apple.com/library/prerelease/ios/documentation/Swift/Conceptual/BuildingCocoaApps/index.html#/apple_ref/doc/uid/TP40014216



- Swift closely parallels Obj-C – we will explore the Swift Programming Language over the next 5 weeks while we **focus on learning the APIs and functionality provided by the iOS SDK.**
- Provides its own versions of all fundamental C and Obj-C types (**Int, Double, Float, Bool, String, Array, Dictionary**)
- Introduces advanced types (tuples – multiple return values)
- Introduces optional types (aka “optionals”) to protect from runtime exceptions caused by **nuls**
- Type-safety (strictly and statically typed) is stressed by the language and the compiler but you don’t have to specify type if the type is expected/can be inferred
- Importing the Foundation framework or UIKit maps Obj-C classes to Swift classes “automagically”. For example NSString works nicely with String.

```
import Foundation
Class MyViewController : UIViewController {
    func describeMyself(name:String) -> String {
        return "My name is \(name)."
    }
    let myLabel = UILabel(frame: CGRectZero)
    myLabel.text = describeMyself("Justin")
    println(myLabel.text)
}
```

Obj-C

```
int myVariable = 42;  
  
//-----  
  
const int myConstant = 50;  
  
//-----  
  
UITableView *myTableView = [[UITableView  
alloc] initWithFrame:CGRectZero  
style:UITableViewStyleGrouped];  
  
myTableView.backgroundColor = [UIColor  
darkGrayColor];  
  
NSLog(@"%@",  
myTableView.backgroundColor);  
  
[myTableView reloadData];  
  
////
```

Swift

```
var myVariable = 42  
  
//-----  
  
let myConstant = 50  
  
//-----  
  
let myTableView =  
UITableView(frame:CGRectZero, style:  
.Grouped)  
  
myTableView.backgroundColor =  
UIColor.darkGrayColor()  
  
println(myTableView.backgroundColor)  
  
myTableView.reloadData()  
  
////
```

cheat sheet



Swift

type is inferred; no *

```

var myVariable = 42
//-----

let myConstant = 50
//-----

let myTableView = UITableView(frame:CGRectZero,
style: .Grouped)

myTableView.backgroundColor =
UIColor.darkGrayColor()

println("The color is" +
myTableView.backgroundColor)

myTableView.reloadData()

```

 Table view will not be modified, so we use `let`

 no semicolons! `=`

 no need to call `alloc init`

method arguments in parentheses, separated by comma

reference constants by dot syntax (type inferred)

- With just this much, we can create a view hierarchy, access properties, get and set the values of variables we define...in other words, 90% of your life as a developer
- As we go through the advanced course, I will pepper the slides with the Swift way of doing things

Still use parentheses even if no arguments (this makes it a method call)

- In this application, we will use Swift to build a very simple app that displays a question and the corresponding answer.
- This will let us study a little Swift code and also get familiar with Interface Builder so that we can start making more sophisticated view hierarchies next week.
- Implementation:
 - We will create two arrays – one to hold the questions and another to hold the answers
 - We will generate a simple user interface with a label to display the question and a label to display the corresponding answer when the appropriate button is pressed



- Read the 'Welcome to Swift' section from [The Swift Programming Language](#)
- Read 'Getting Started' from [Using Swift with Cocoa and Objective-C](#)
- This is enough for you to start a simple project in Swift or perhaps convert an existing project to Swift
- Some things may go over your head but we will cover the material over the next 5 weeks, so don't worry. Some topics—like blocks or closures—will be covered in the advanced class.
- Challenge: write the Quiz sample application from BNR Ch.1 as a Swift project following the video and lecture notes.
- Look at the Tutorials in the Homework Guide

