

<http://github.com/GroundSpeed/ParseSwift>

Locking Out the Abominable Snowman in iOS

A CodeMash 2015 Introduction to parse.com using Swift

Don Miller



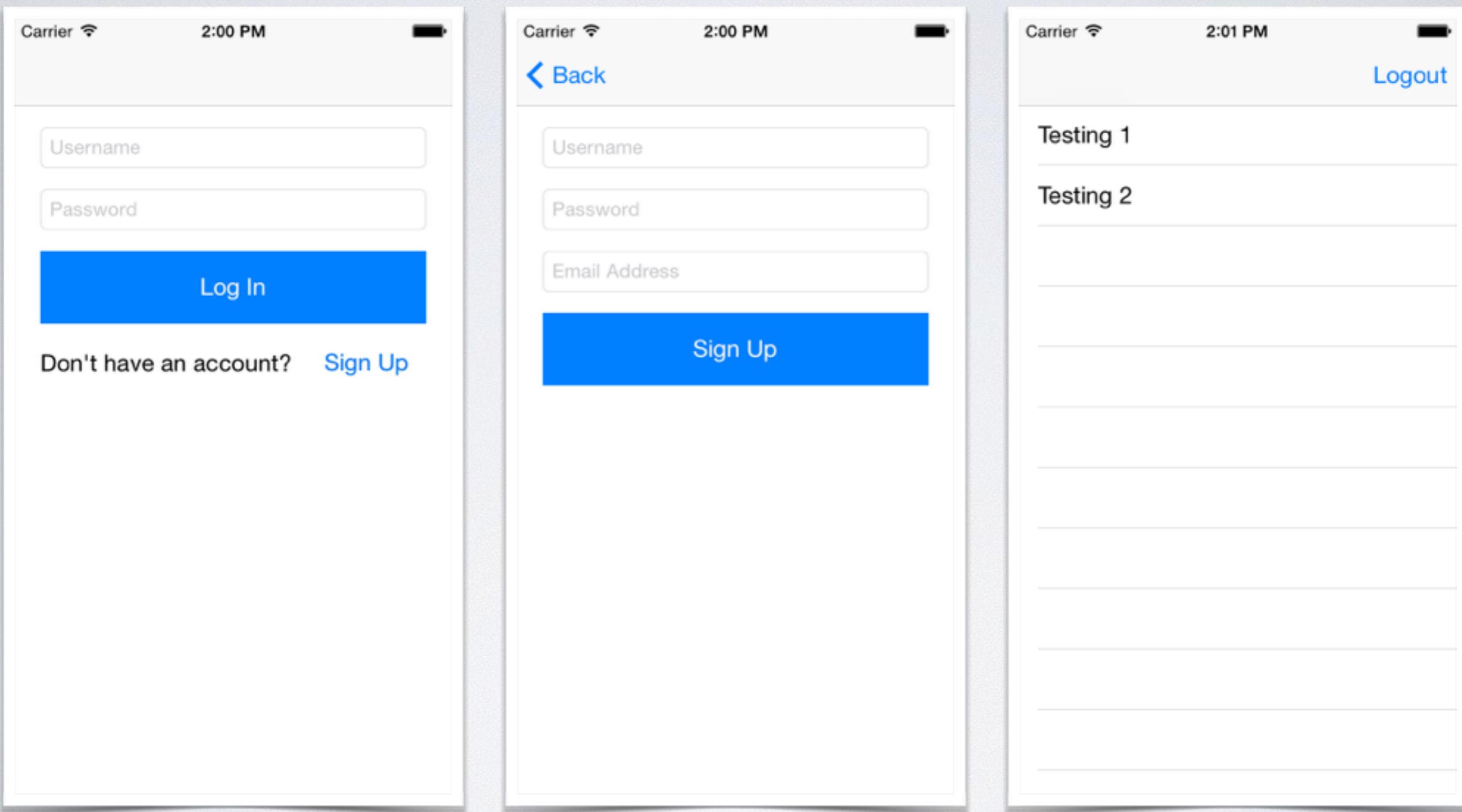
GroundSpeed™
rapid web + mobile software

A bit about Parse

- “Cloud” service to store various pieces of data and notifications
 - Users for Security
 - Classes for data tables (Core)
 - We can pull, push, update, or delete records
 - Analytics
 - Push Notifications



Let's Create a Simple App to Authenticate to Parse



Create Parse App

P

Select an App ▾

Dashboard



Create a new App



GroundSpeed™
rapid web + mobile software

Create Parse App

P

Select an App

oard

[new App](#)

What's the name of your new app?

CodeMash 2015

Create

 cancel



GroundSpeed™
rapid web + mobile software

Create Parse App

P

Select an App

DEV



CodeMash 2015

Created 2 minutes ago

Just getting started?
Check out the [quickstart guide!](#)

Core

Analytics

Push

Quickstart

board

[new App](#)



GroundSpeed™
rapid web + mobile software

Create Parse App

The screenshot shows the Parse Dashboard interface. At the top, there's a blue header bar with the text "Add a new class" and a close button "X". Below this, a large blue button contains the word "User" and a dropdown arrow. A descriptive text block below the button states: "The User class is a special class that can be authenticated, allowing users of your app to log in, sign up, and more." At the bottom of the dialog are two buttons: "Create Class" on the left and "Cancel" on the right. The background of the dialog is white, and it has rounded corners. At the very bottom of the screen, there's a navigation bar with tabs for "Data", "Analytics", "Cloud", and "Generate API".



Create Parse App

P

Add a new class

Custom ▾ Whiskey

Class names must only contain numbers, letters, and underscore, and can only begin with a letter.

Create Class Cancel



Create Parse App

Add a Column



String



name

Must only contain alphanumeric or underscore characters, and must begin with a letter or number.

Create Column

Cancel



GroundSpeed™
rapid web + mobile software

Create A New Single View Application

Choose a template for your new project:

iOS	 Master-Detail Application	 Page-Based Application	 Single View Application	 Tabbed Application
OS X	 Game			

Single View Application

This template provides a starting point for an application that uses a single view. It provides a view controller to manage the view, and a storyboard or nib file that contains the view.

[Cancel](#) [Previous](#) [Next](#)



Enter the Product Name

Choose options for your new project:

Your company's name

Product Name: ParseMash

Organization Name: GroundSpeed

Organization Identifier: com.groundspeedhq

Bundle Identifier: com.groundspeedhq.ParseMash

Language: Swift

Devices: iPhone

Use Core Data

Cancel

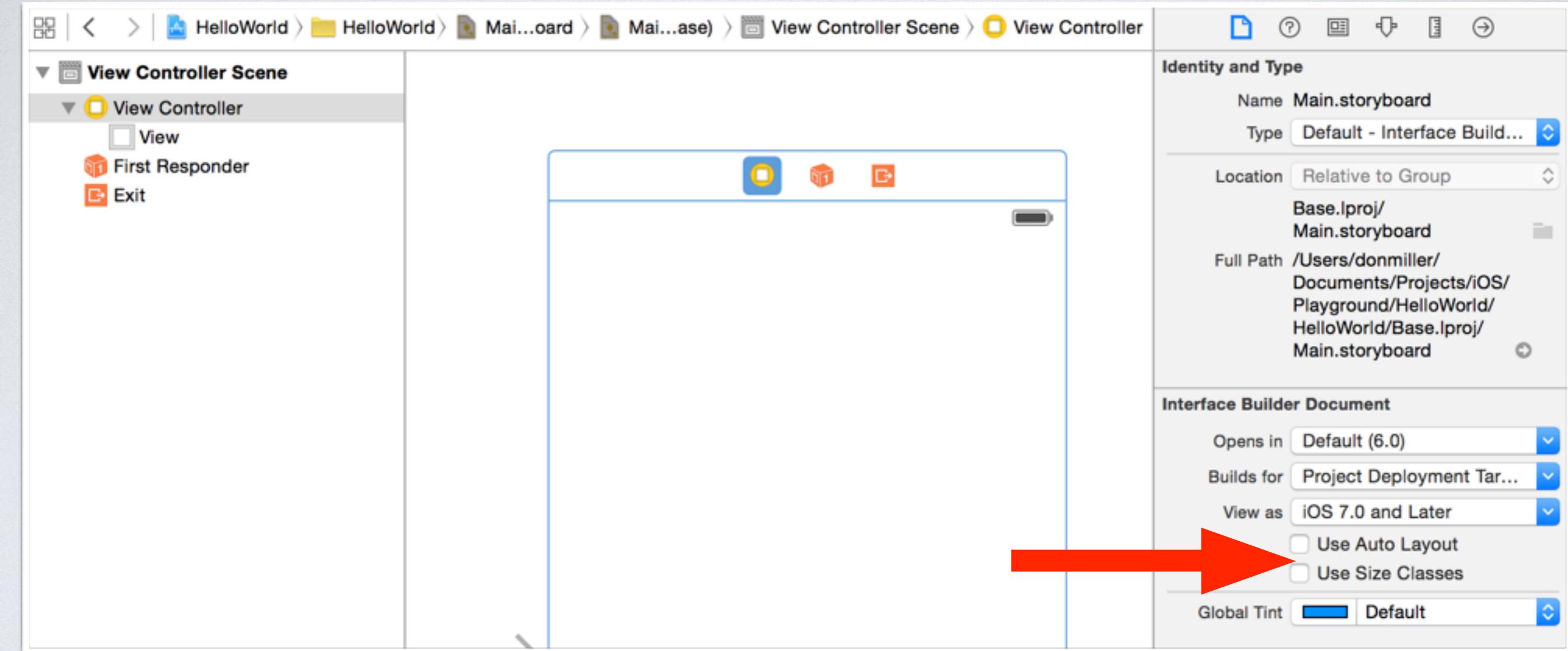
Previous

Next

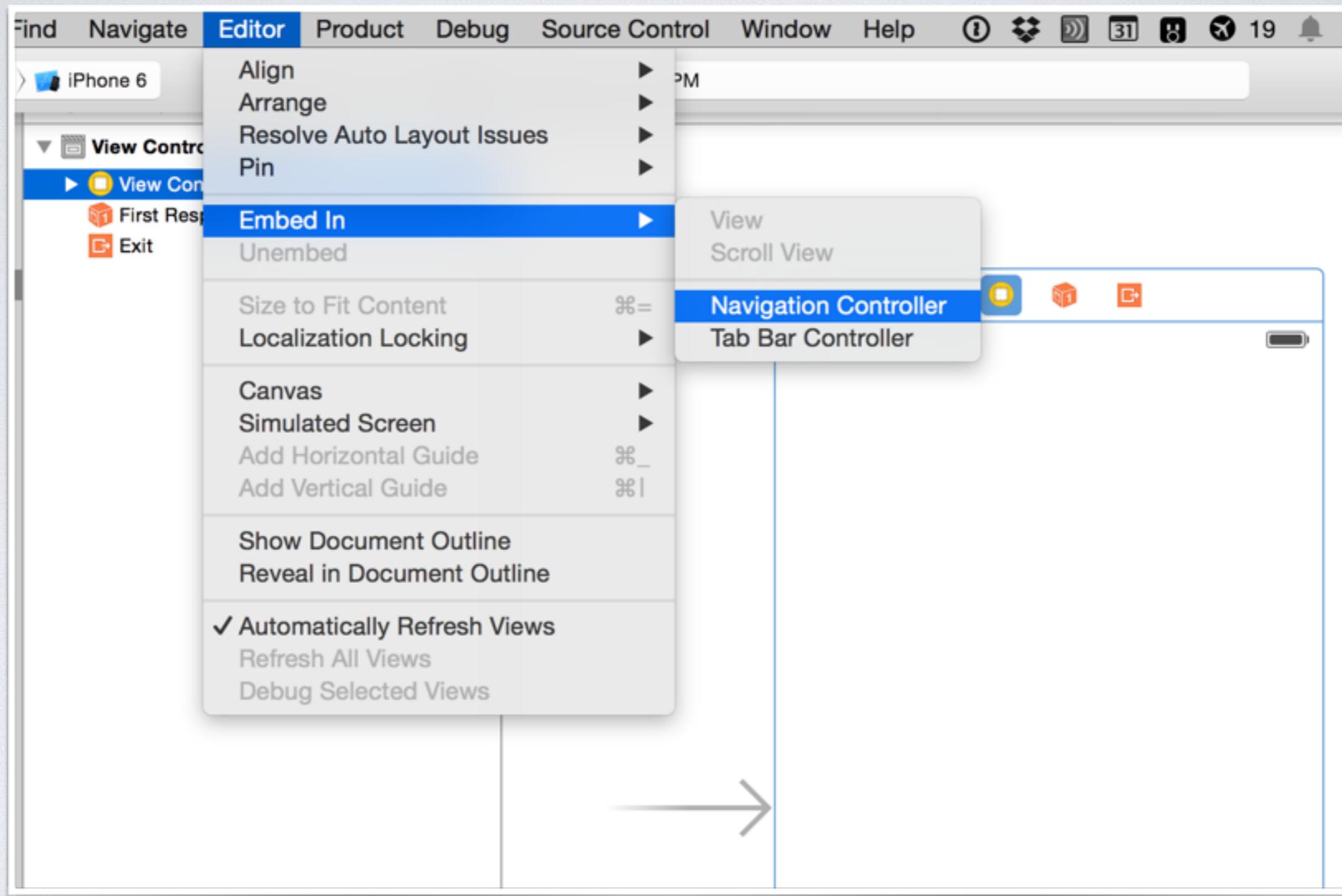


GroundSpeed™
rapid web + mobile software

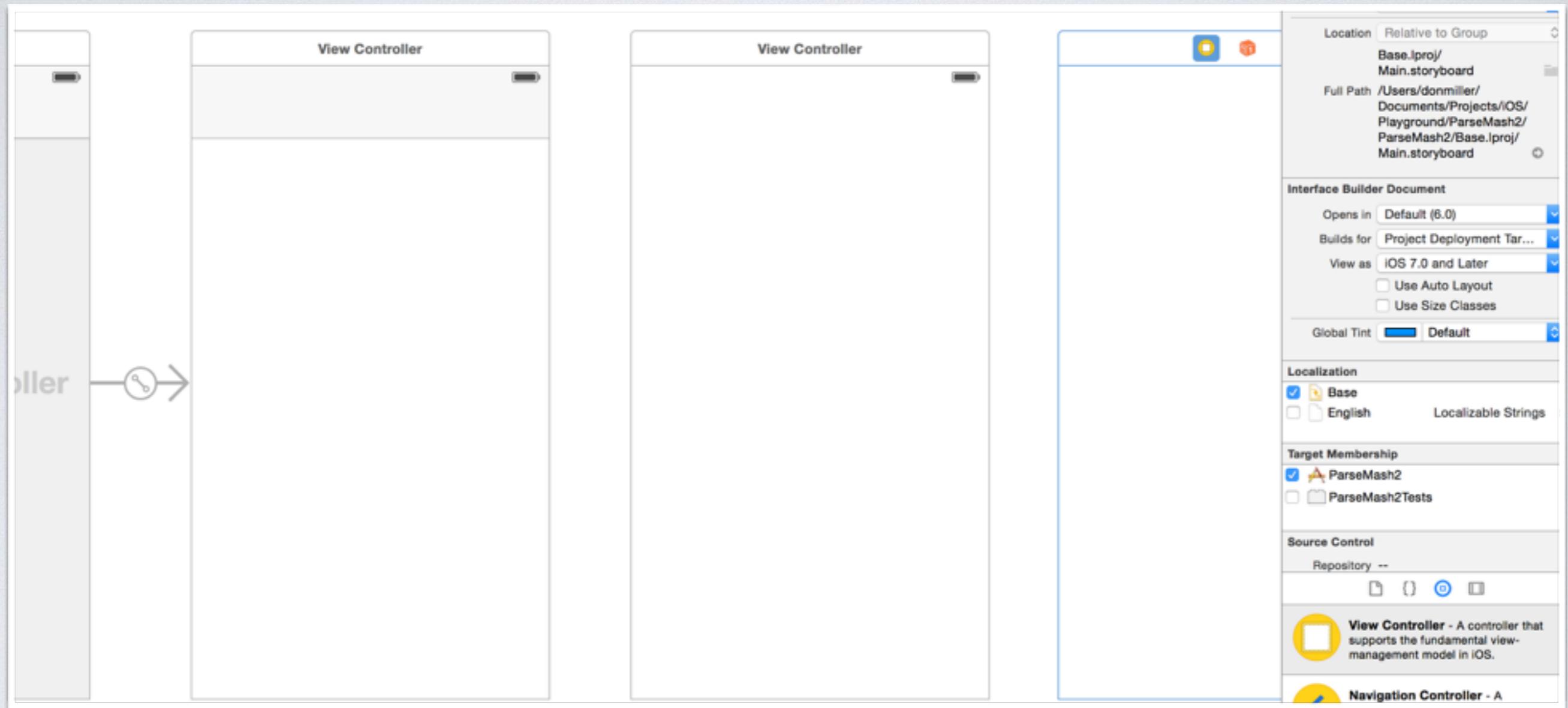
Turn Off Size Classes and Auto Layout



Embed Main View Inside Navigation



Create 2 Additional View Controllers



Add TextFields and Buttons to the Login and Signup

The image shows two mobile application screens side-by-side. Both screens have a light gray header bar at the top with rounded corners. On the left screen, the word "Login" is centered in the header. On the right screen, the word "SignUp" is centered in the header. In the top right corner of both headers, there is a small battery icon. Below the header on the left screen is a large, empty rectangular area. Below the header on the right screen is also a large, empty rectangular area. On the left screen, there are two white rounded rectangular input fields stacked vertically. The top one is labeled "Username" and the bottom one is labeled "Password". Below these input fields is a large blue rectangular button with the word "Login" in white. At the bottom left of the left screen, there is a line of text that says "Don't have an account? [Sign Up](#)". On the right screen, there are three white rounded rectangular input fields stacked vertically. The top two are labeled "Username" and "Password", and the bottom one is labeled "Email Address". Below these input fields is a large blue rectangular button with the word "Sign Up" in white.

Login

Username

Password

Login

Don't have an account? [Sign Up](#)

SignUp

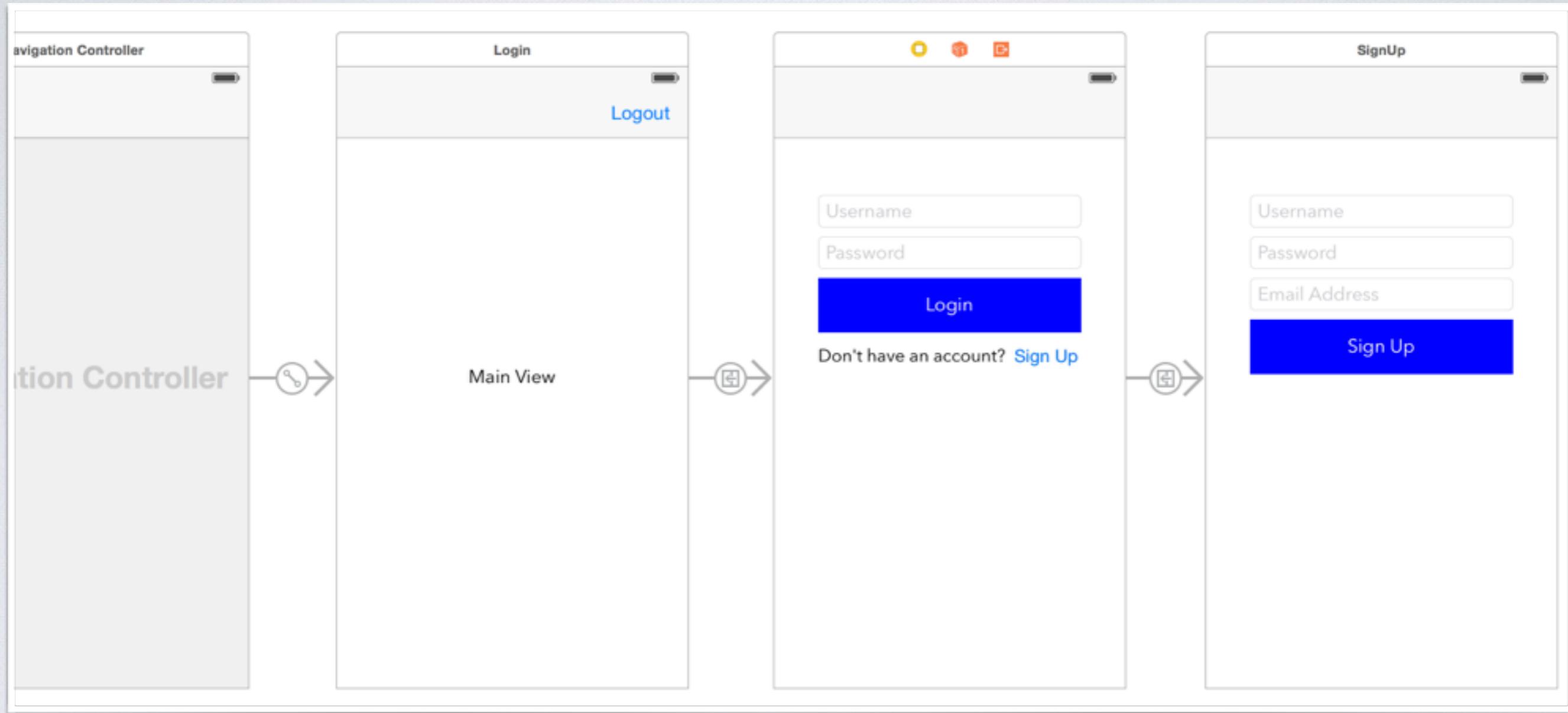
Username

Password

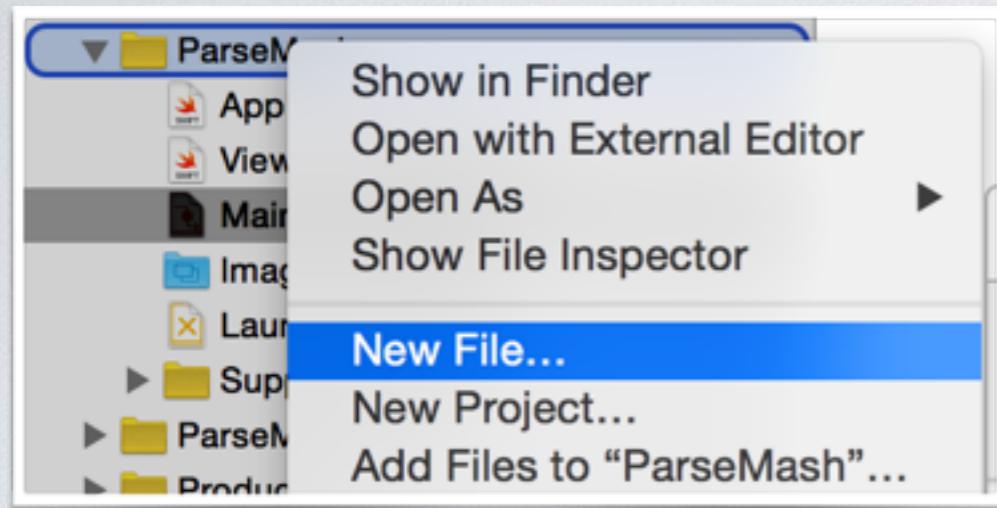
Email Address

Sign Up

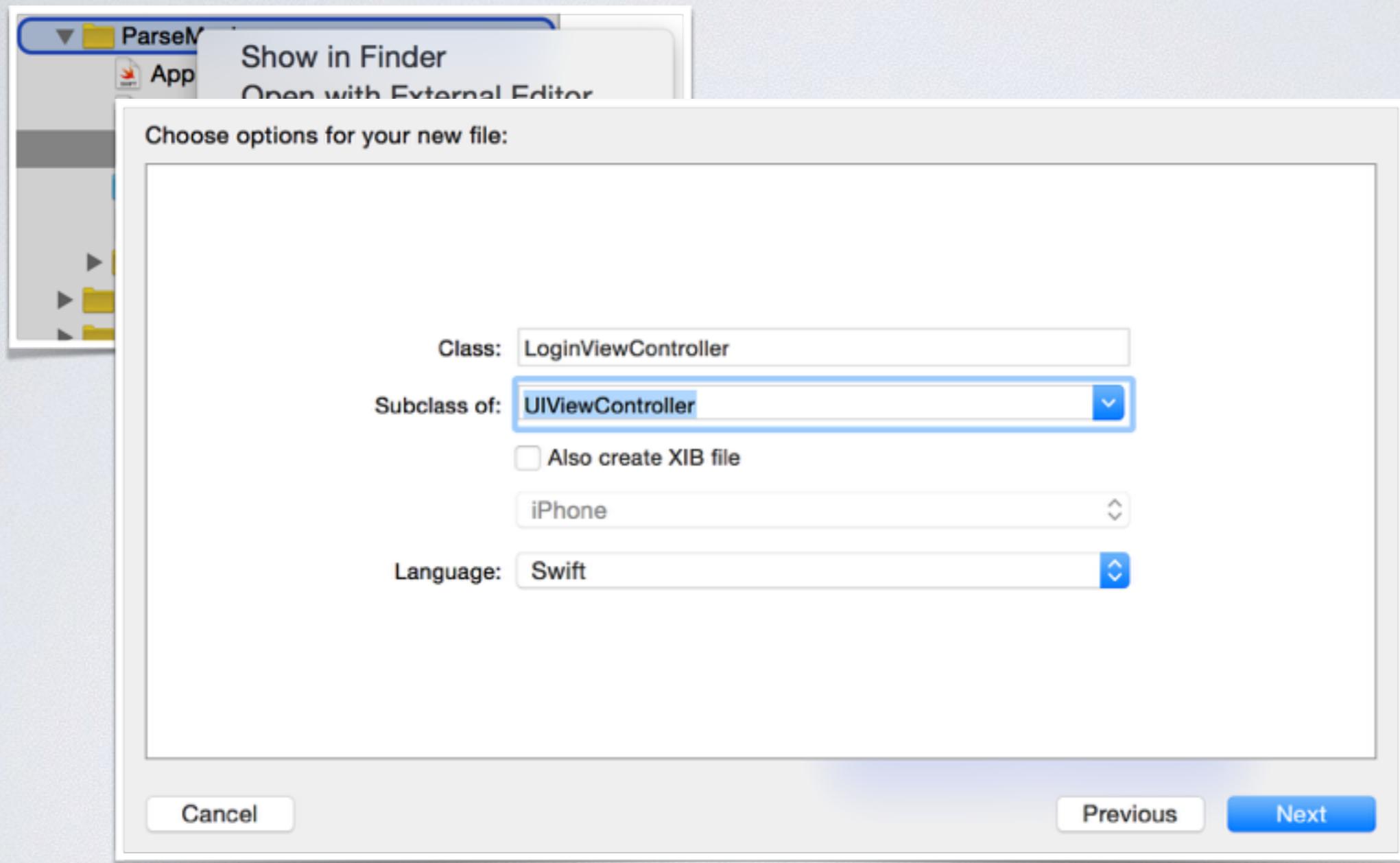
Add Logout Bar Button and Push Segues



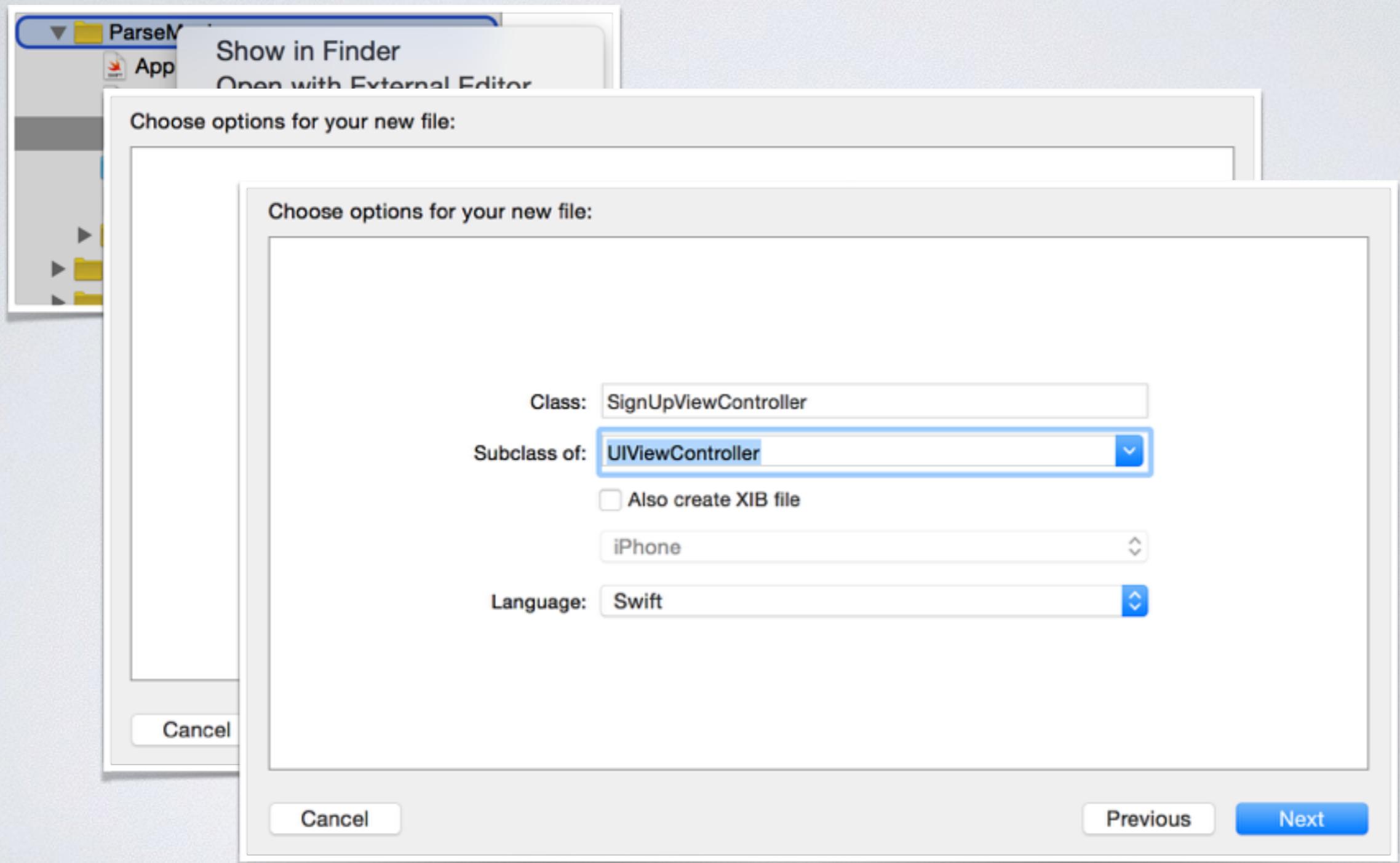
Create Login and Signup View Controllers



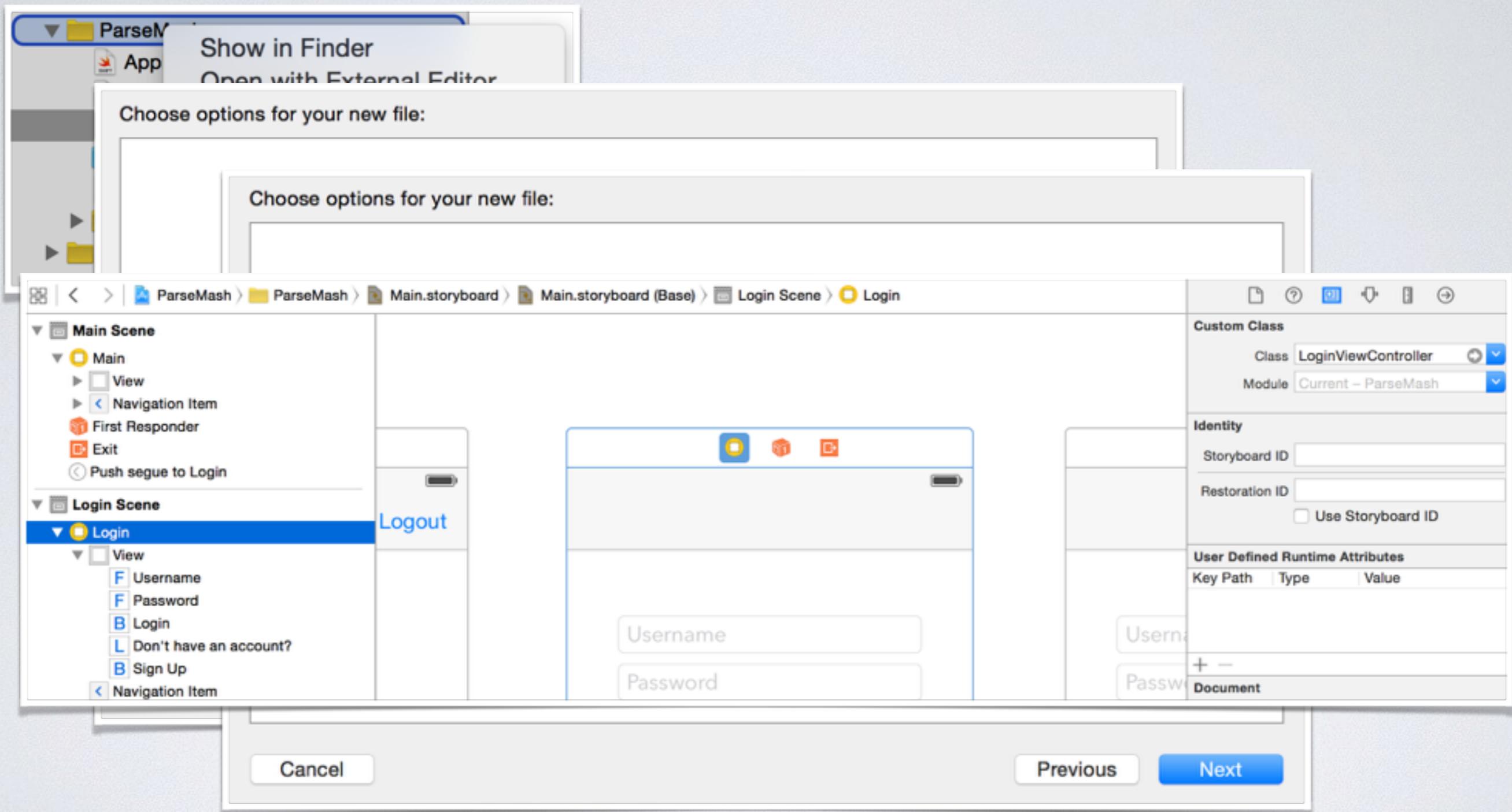
Create Login and Signup View Controllers



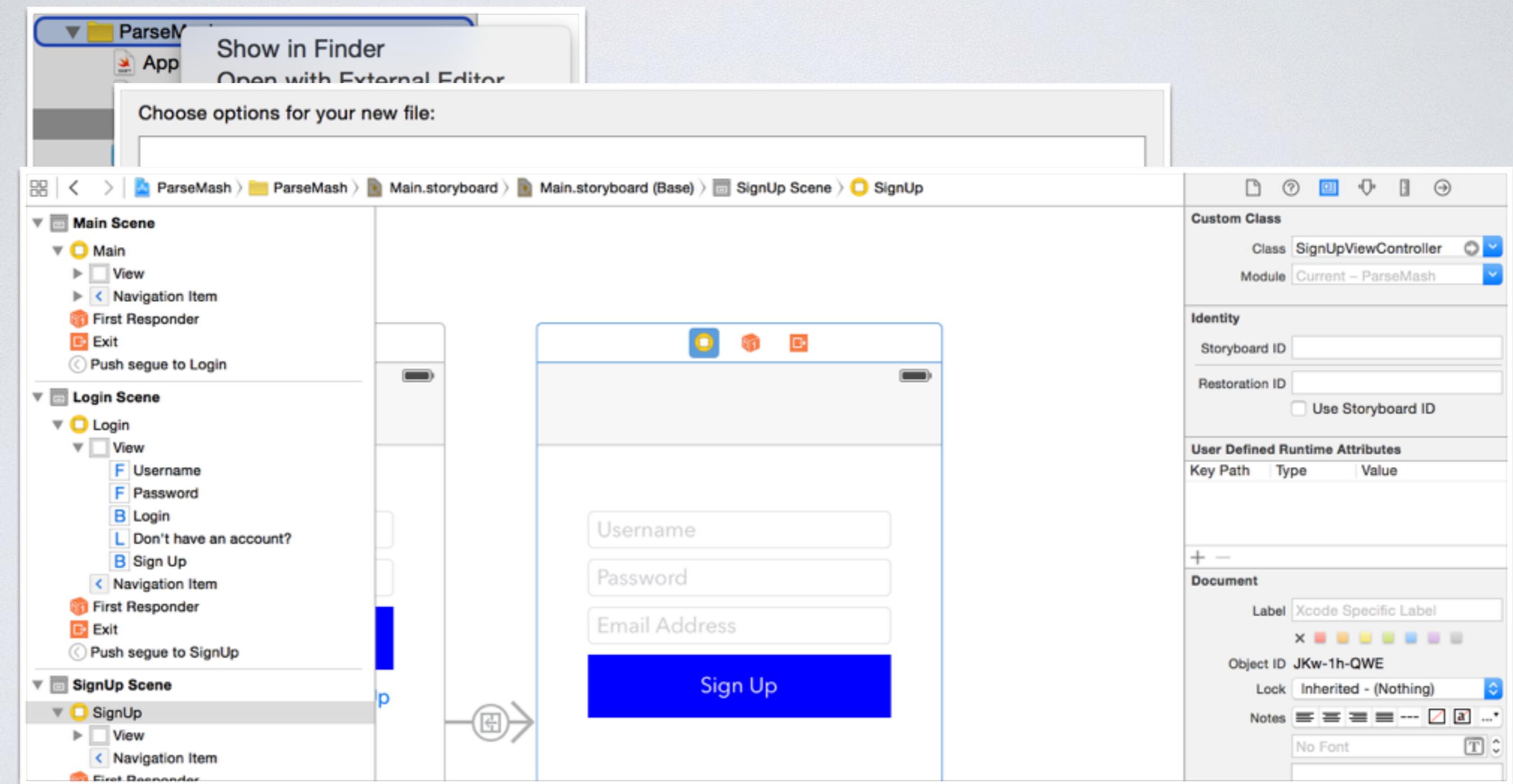
Create Login and Signup View Controllers



Create Login and Signup View Controllers



Create Login and Signup View Controllers



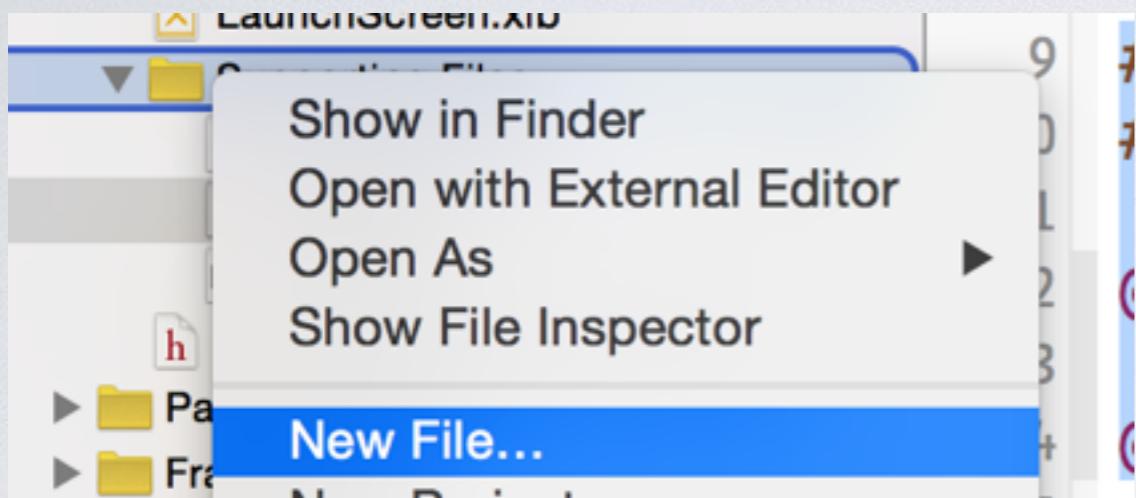
Add the Needed Parse Framework and Prerequisites

The red are required by Parse, but the screenshot is required for our needs

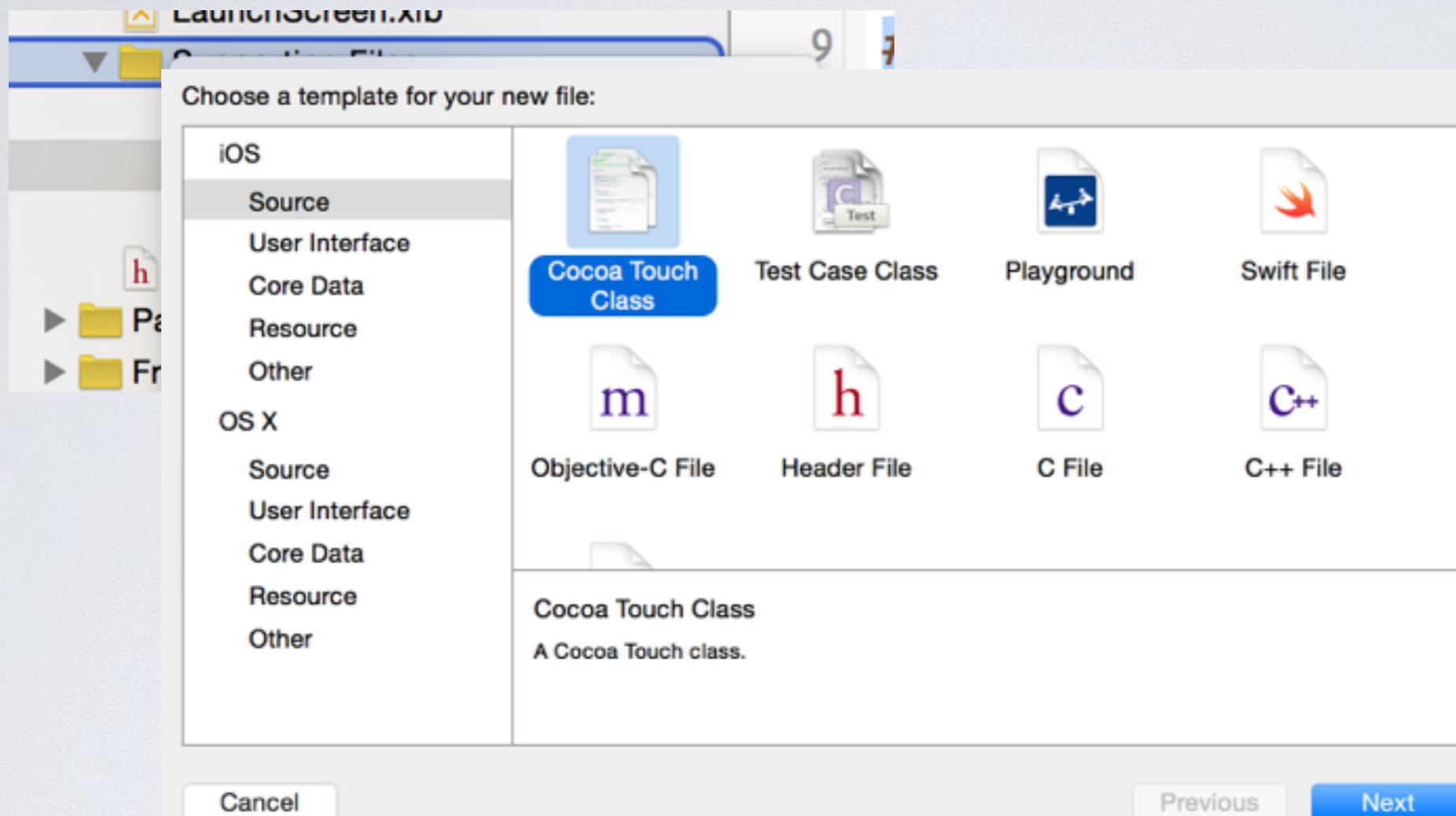
- `AudioToolbox.framework`
- `CFNetwork.framework`
- `CoreGraphics.framework`
- `CoreLocation.framework`
- `libz.dylib`
- `sqlite3.dylib`
- `MobileCoreServices.framework`
- `QuartzCore.framework`
- `Security.framework`
- `StoreKit.framework`
- `SystemConfiguration.framework`
- `Bolts.framework`
- `Parse.framework` (obviously)



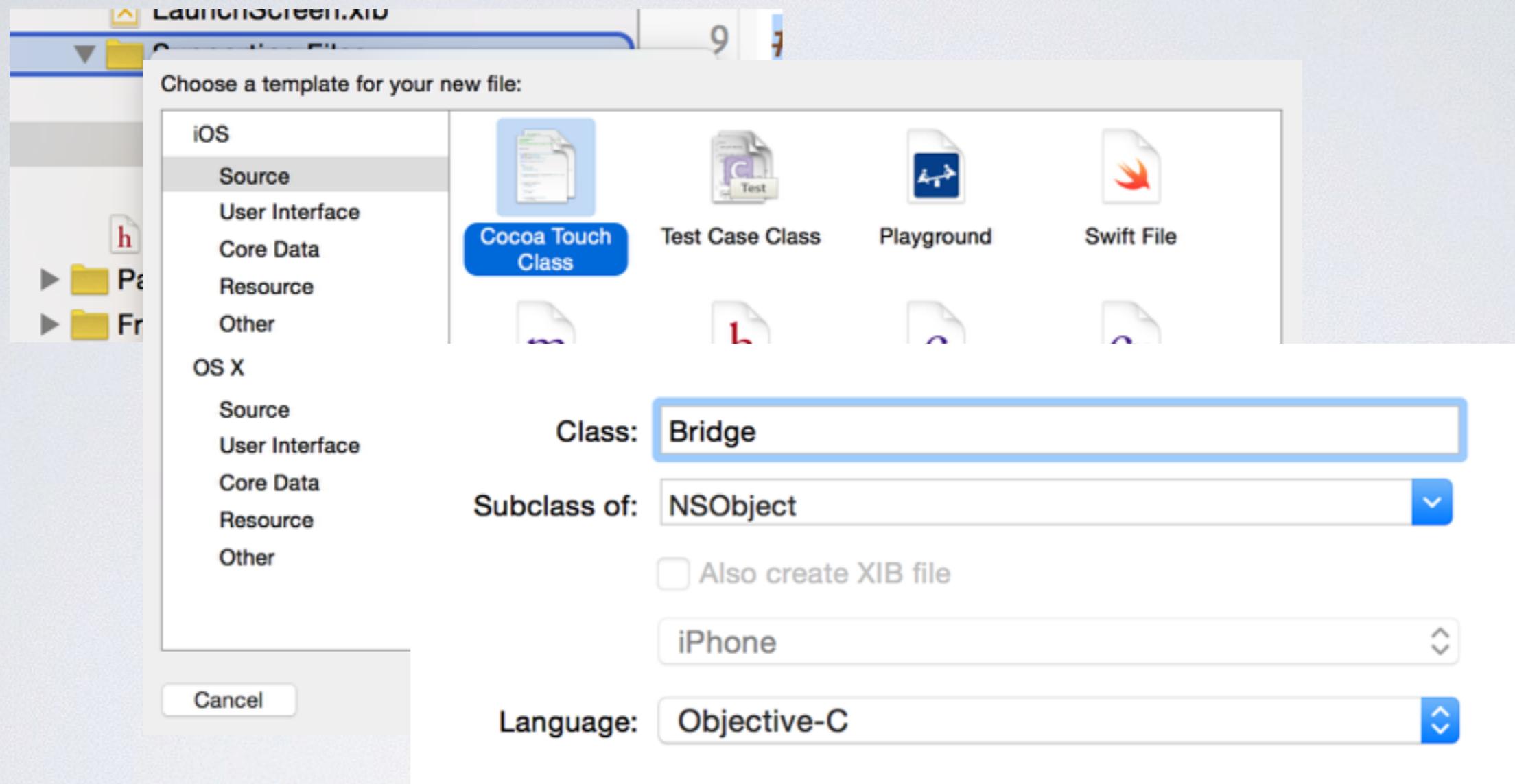
Create Objective-C Bridge



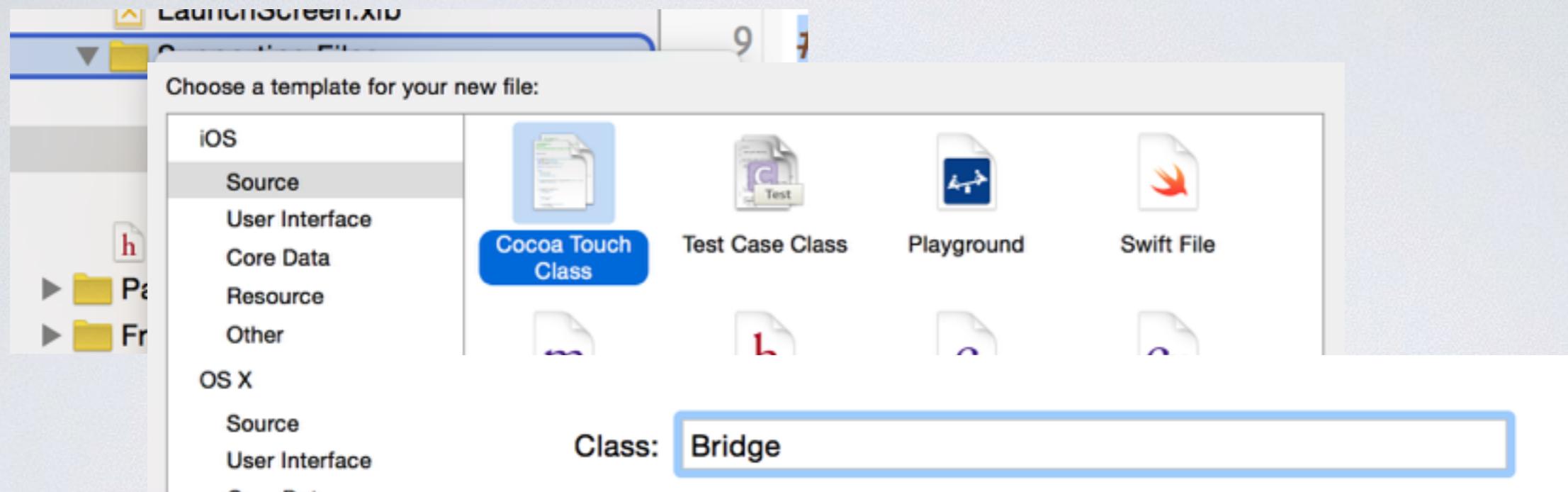
Create Objective-C Bridge



Create Objective-C Bridge



Create Objective-C Bridge



The screenshot shows the Xcode interface with a modal dialog for creating a new file. The dialog title is "Choose a template for your new file:". On the left, under the "iOS" section, the "Source" option is selected. In the center, there are five icons: "Cocoa Touch Class" (selected), "Test Case Class", "Playground", and "Swift File". On the right, there are three more icons. Below the icons, the "Class:" field contains the text "Bridge".

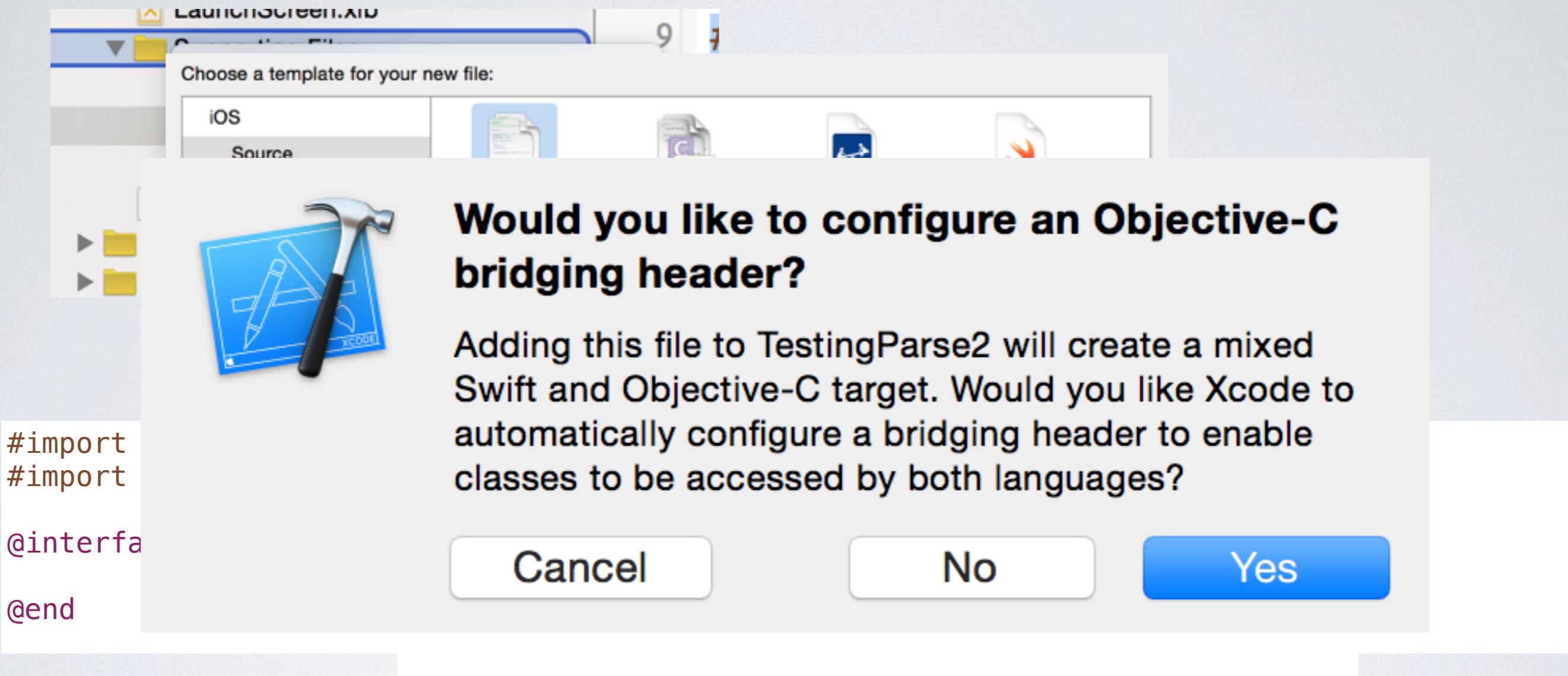
```
#import <Foundation/Foundation.h>
#import <Parse/Parse.h>

@interface Bridge : NSObject

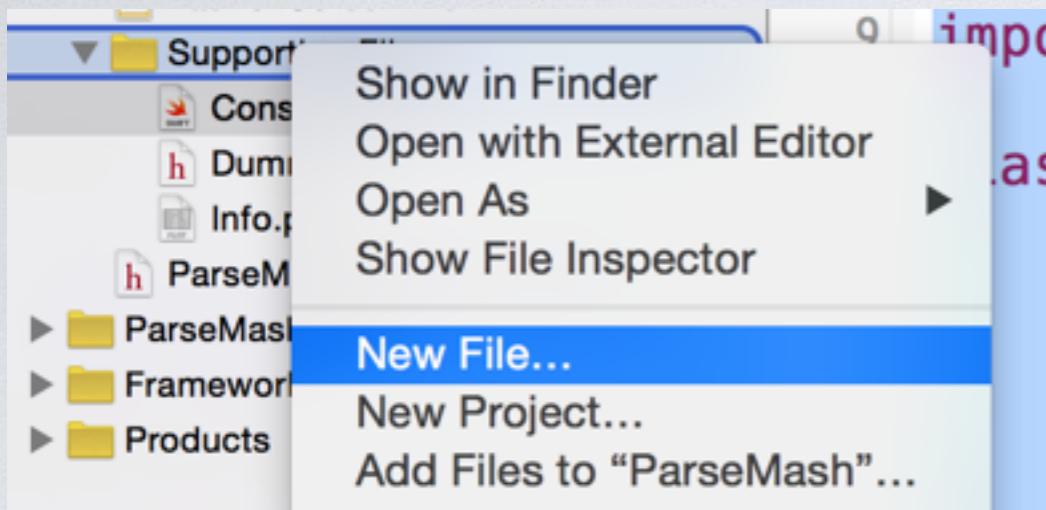
@end
```



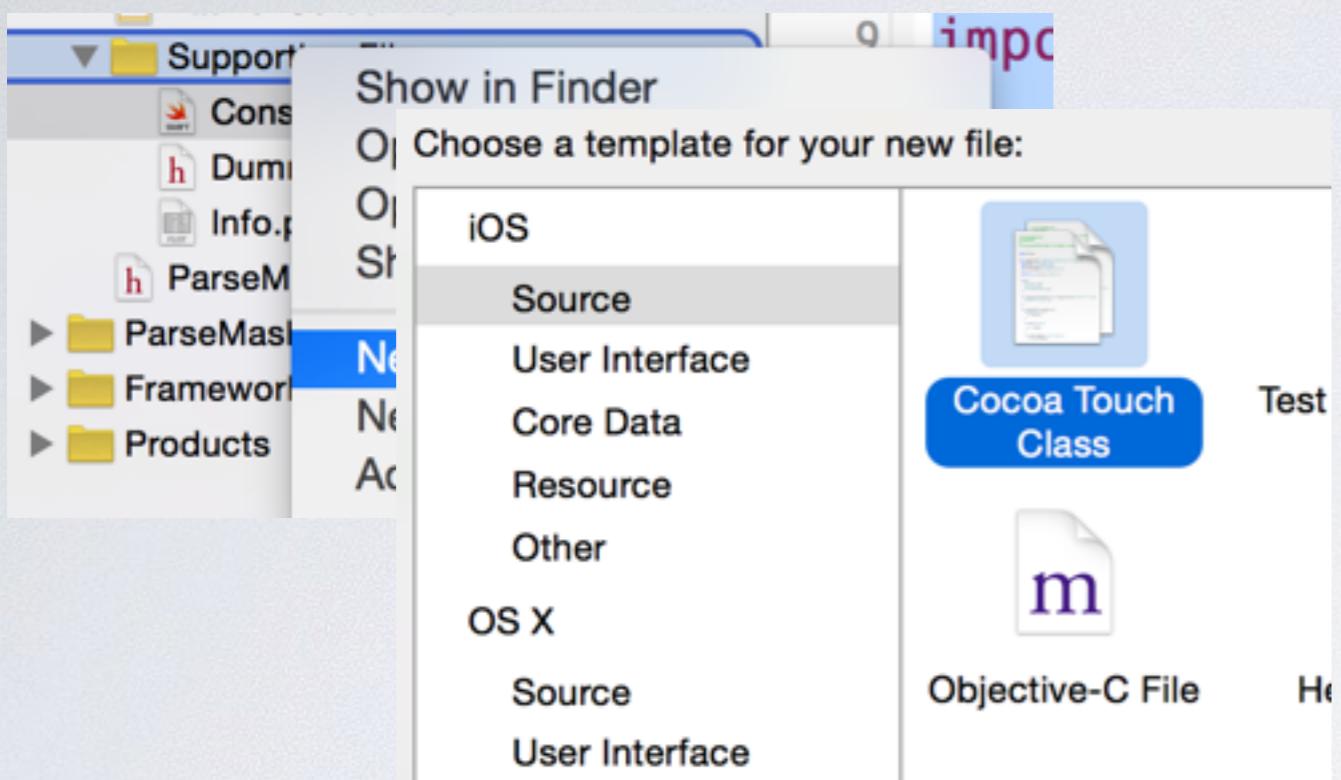
Create Objective-C Bridge



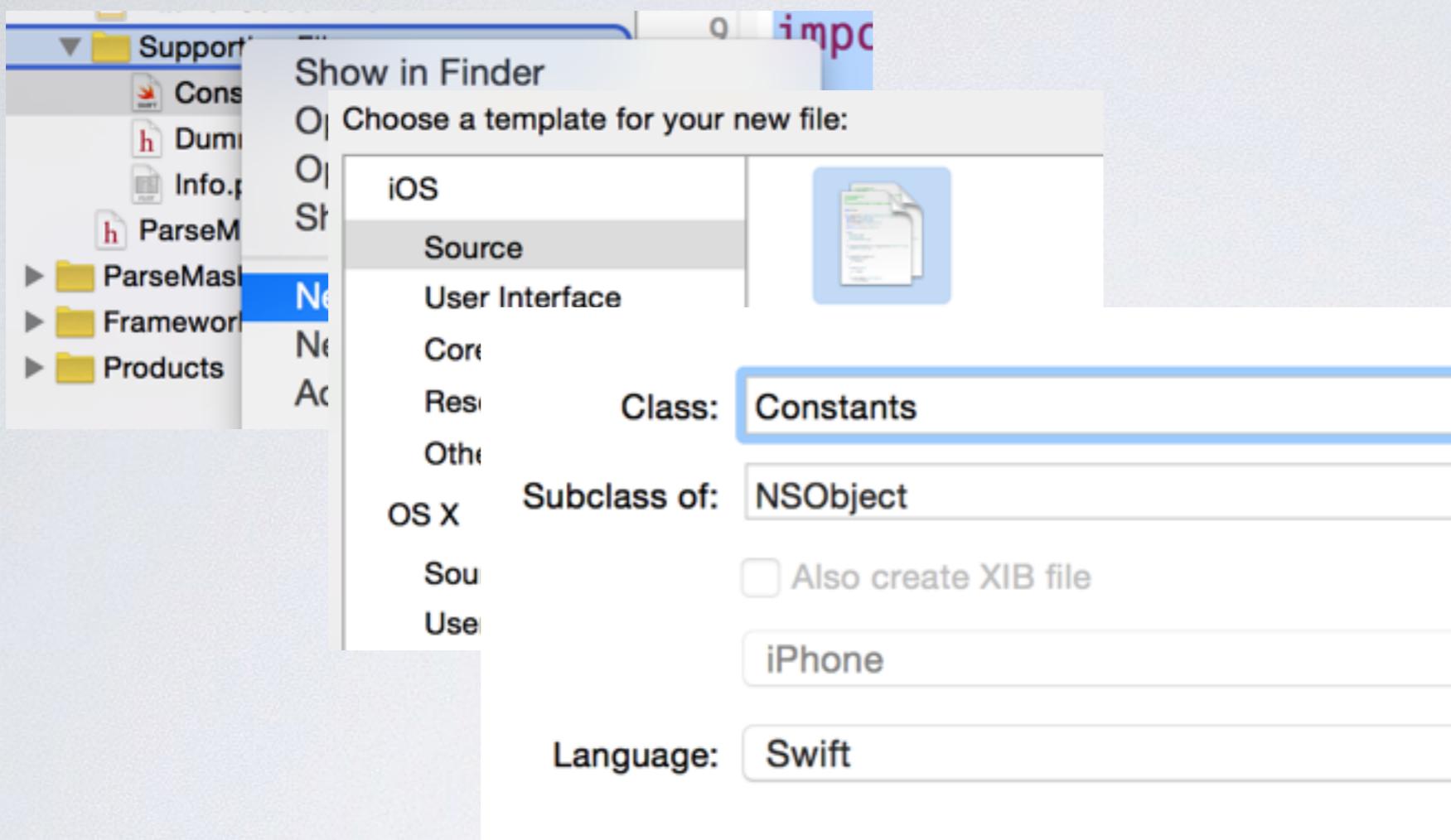
Create a Constants Class



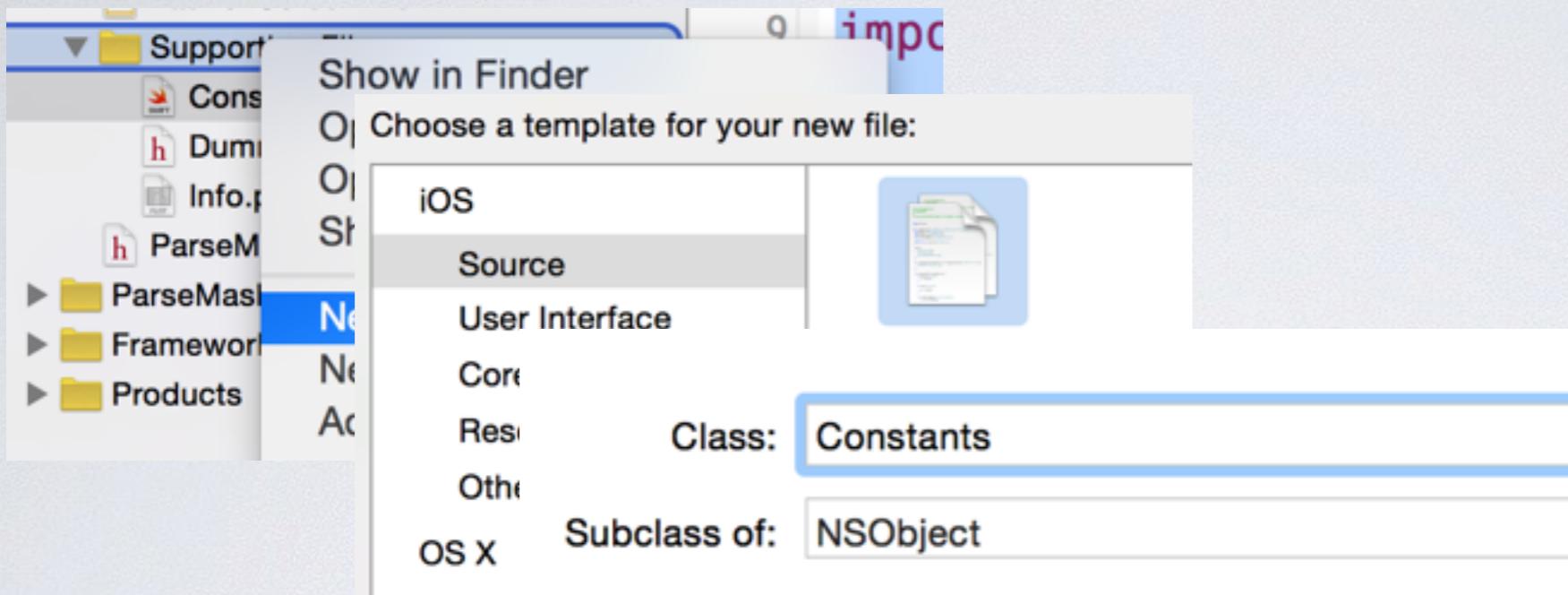
Create a Constants Class



Create a Constants Class



Create a Constants Class

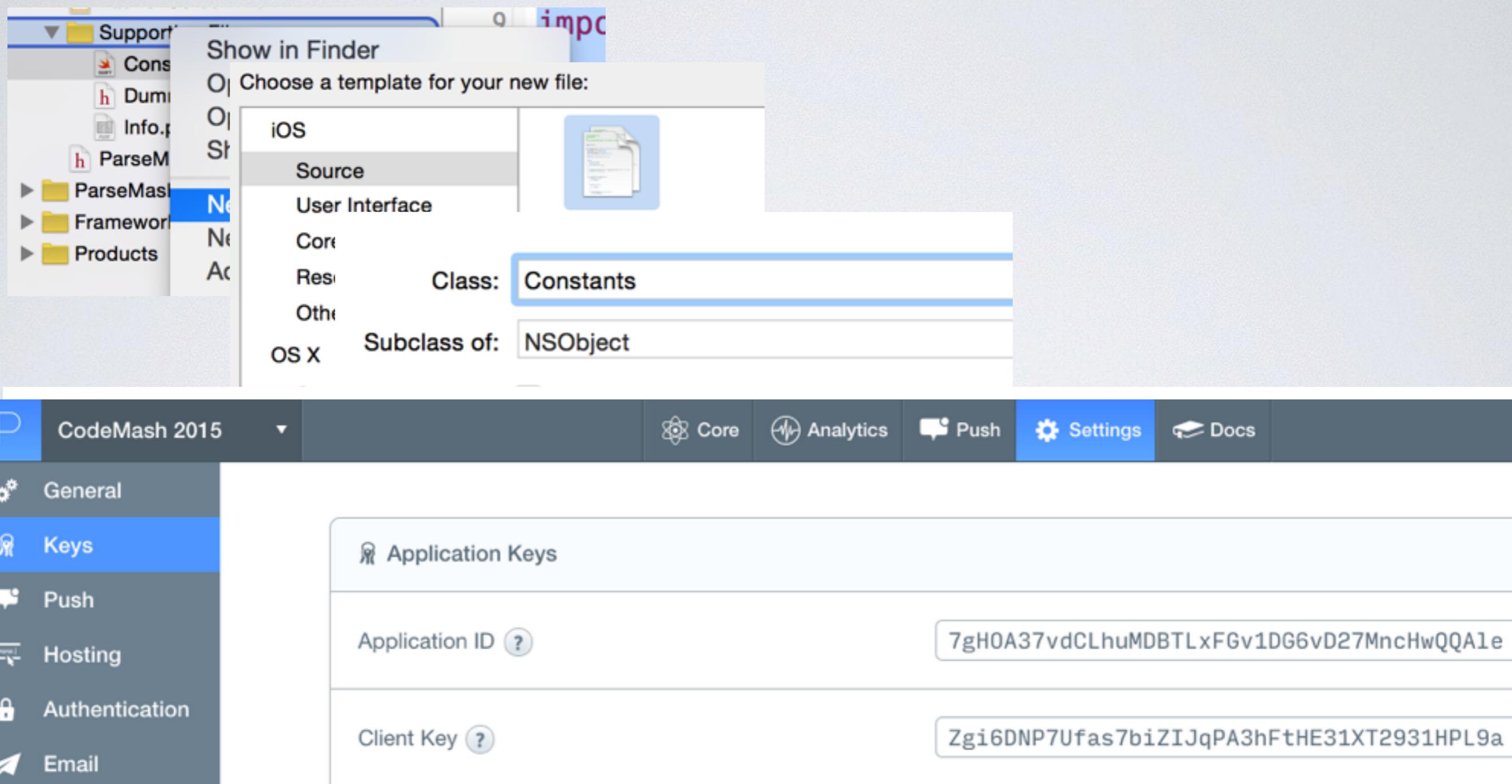


```
import UIKit

class Constants: NSObject {
    let kParseClassName = "Whiskey"
    let kApplicationId = "Copy and paste from settings"
    let kClientKey      = "Copy and paste from settings"
}
```



Create a Constants Class



Add Parse to the App Delegate



GroundSpeed™
rapid web + mobile software

Add Parse to the App Delegate

```
import Parse

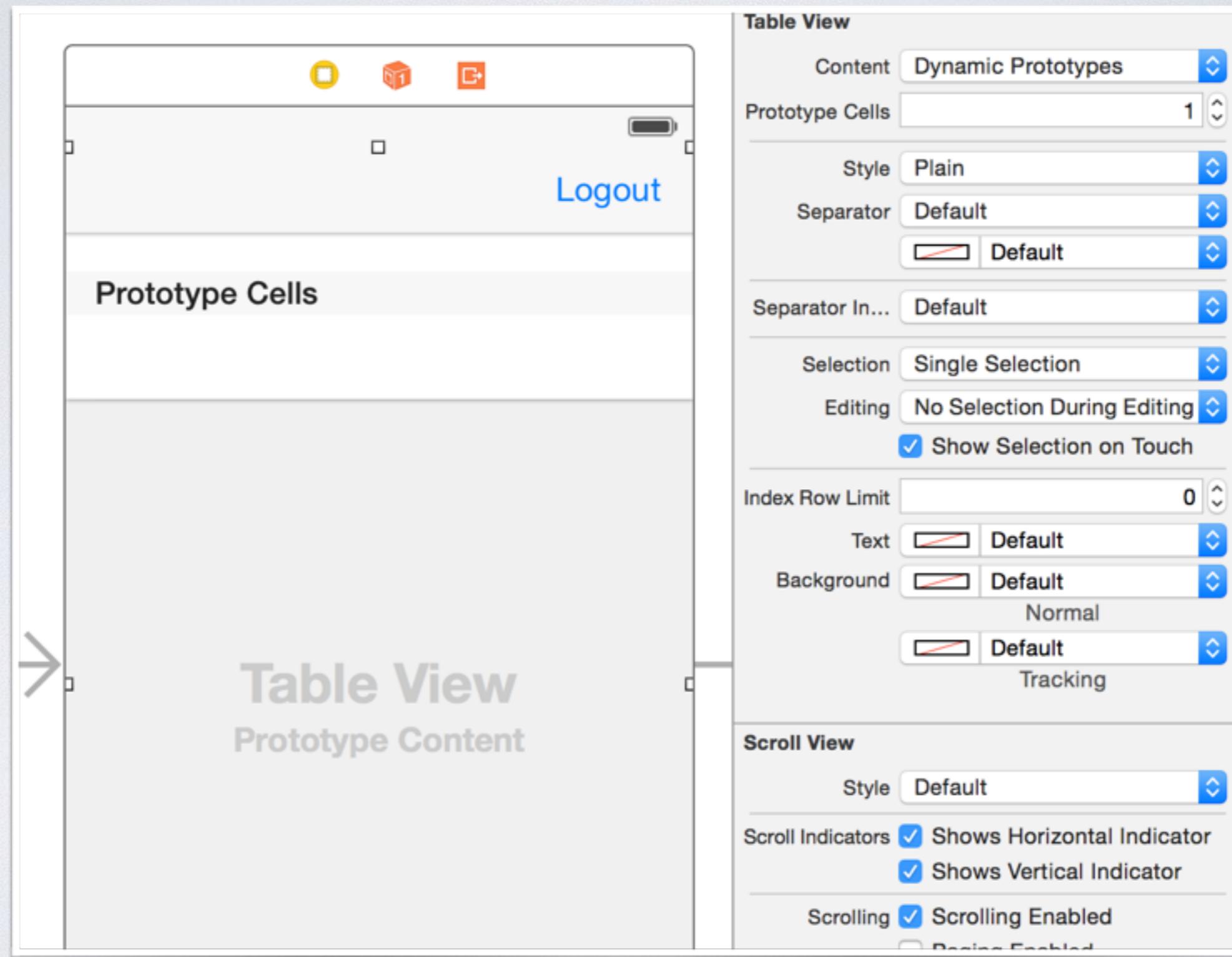
@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {

    var window: UIWindow?
    let c = Constants()

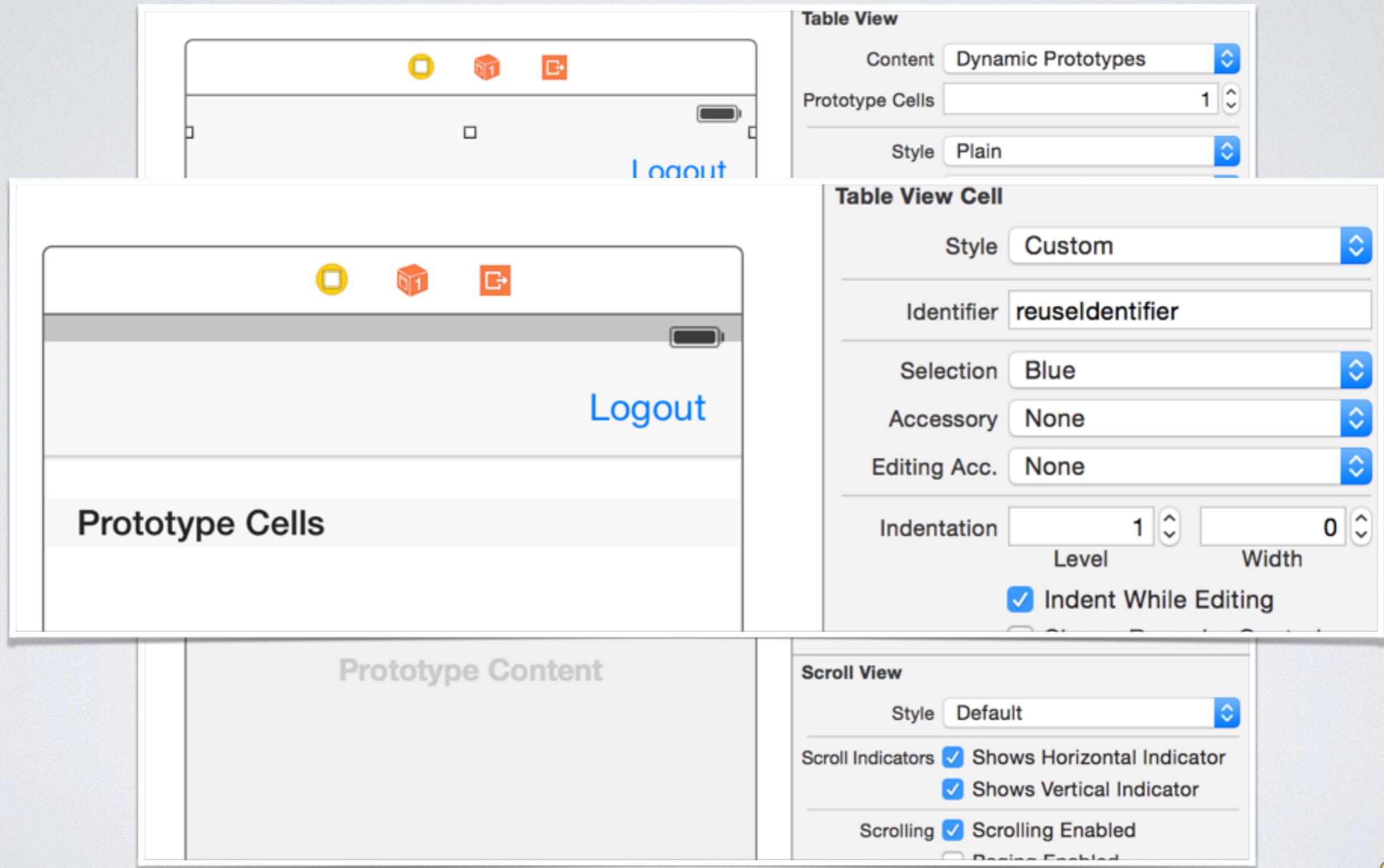
    func application(application: UIApplication, didFinishLaunchingWithOptions
launchOptions: [NSObject: AnyObject]?) -> Bool {
        // Override point for customization after application launch.
        Parse.setApplicationId(c.kApplicationId, clientKey: c.kClientKey)
        return true
    }
}
```



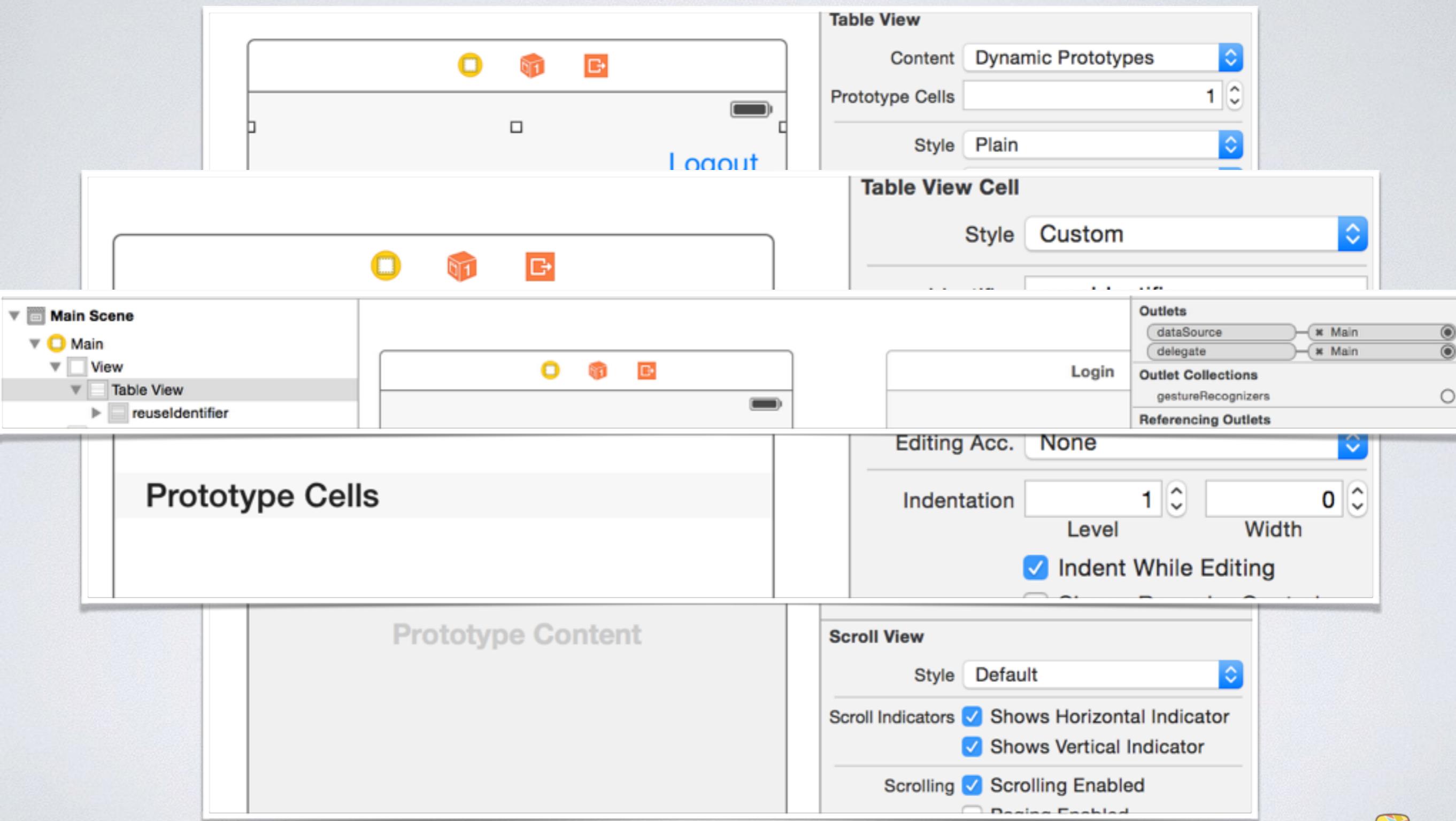
Add a TableView to Your Main Controller



Add a TableView to Your Main Controller



Add a TableView to Your Main Controller



Program Your Main ViewController

```
import UIKit
import Parse

class MainViewController: UIViewController, UITableViewDelegate, UITableViewDataSource {
    .....
}
```



Program Your Main ViewController

```
@IBOutlet var tblWhiskey: UITableView!
let c = Constants()
var whiskeys = [String]()

override func viewDidLoad() {
    super.viewDidLoad()
    // Do any additional setup after loading the view.
    var currentUser = PFUser.currentUser()
    if currentUser != nil {
        var query = PFQuery(className:c.kParseClassName)
        query.findObjectsInBackgroundWithBlock({(NSArray objects, NSError error) in
            if (error != nil) {
                NSLog("error " + error.localizedDescription)
            }
            else {
                NSLog("objects %@", objects as NSArray)
                var whiskeyNames = NSArray(array: objects)

                for whiskey in whiskeyNames {
                    self.whiskeys.append(whiskey.objectForKey("Name") as String);
                }
                self.tblWhiskey.reloadData()
            }
        })
    } else {
        self.performSegueWithIdentifier("showLogin", sender: self)
    }
}
```

Program Your Main ViewController

```
override func viewDidAppear(animated: Bool) {
    tblWhiskey.reloadData()
}

func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    return self.whiskeys.count
}

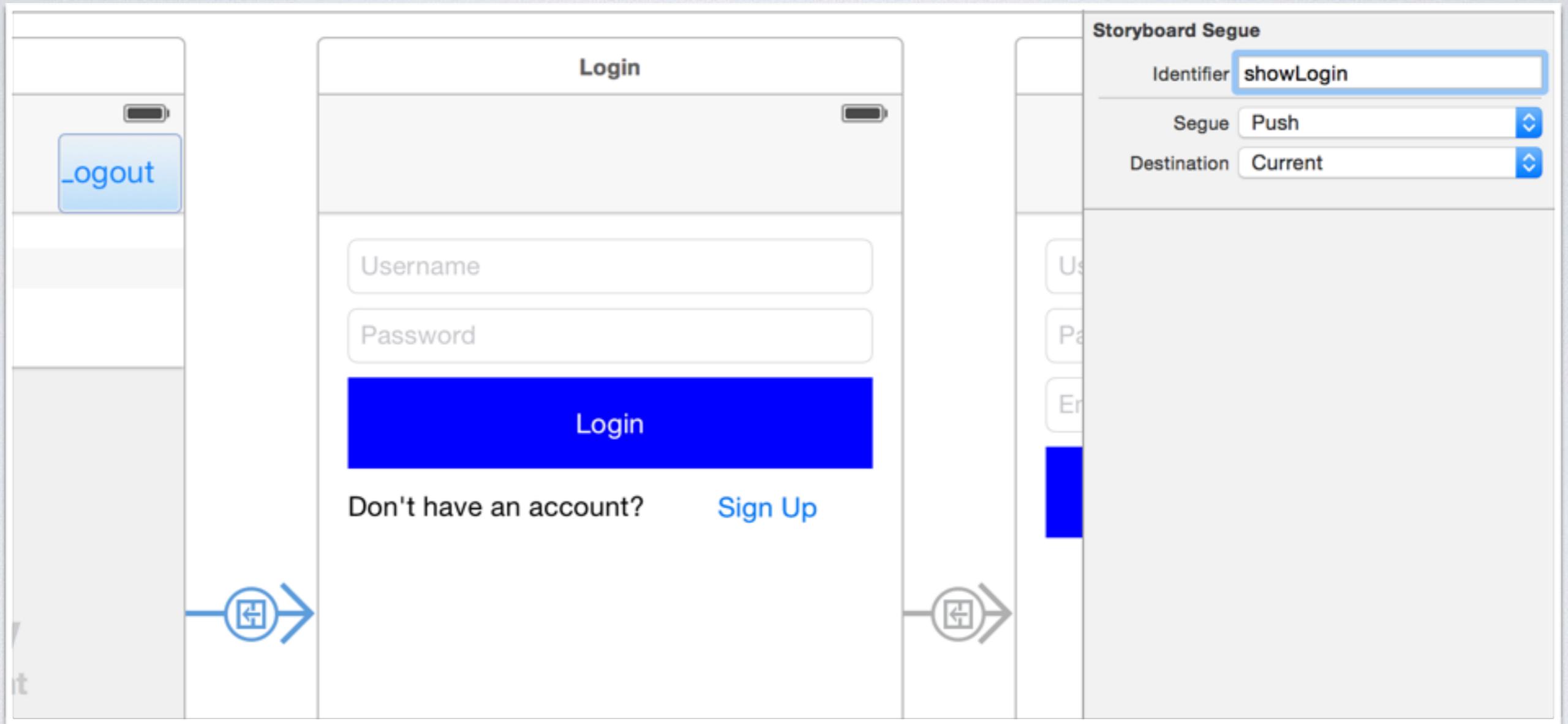
func tableView(tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
    let cell = tableView.dequeueReusableCell(withIdentifier:"reuseIdentifier",
forIndexPath: indexPath) as UITableViewCell

    // Configure the cell...
    cell.textLabel?.text = self.whiskeys[indexPath.row]

    return cell
}

@IBAction func btnLogout(sender: AnyObject) {
    PFUser.logout()
    PFUser.currentUser().save()
    self.performSegueWithIdentifier("showLogin", sender: self)
}
```

Update Identifier in Login Segue



Add your IBAction and IBOulet Connections to the Main View

The screenshot shows the Xcode storyboard editor. On the left, a main view controller is displayed with a navigation bar at the top containing three icons: a blue square with a yellow circle, an orange cube with a white '1', and an orange square with a white 'C'. Below the navigation bar is a battery icon. In the bottom right corner of the main view, there is a blue "Logout" button. To the left of the main view, there is a section labeled "Prototype Cells". The right side of the screen features a sidebar titled "Connections" which lists several categories:

- Triggered Segues**: manual (radio button)
- Outlets**:
 - searchDisplayController (radio button)
 - tblWhiskey (radio button) — connected to * Table View
 - view (radio button) — connected to * View
- Presenting Segues**:
 - relationship (radio button) — connected to * Navigation Co... root view contr...
 - embed (radio button)
 - push (radio button)
 - modal (radio button)
 - custom (radio button)
- Referencing Outlets**: New Referencing Outlet (radio button)
- Referencing Outlet Collections**: New Referencing Outlet Collection (radio button)
- Received Actions**: btnLogout: (radio button) — connected to * Logout



Program Your LoginViewController



GroundSpeed™
rapid web + mobile software

Program Your LoginViewController

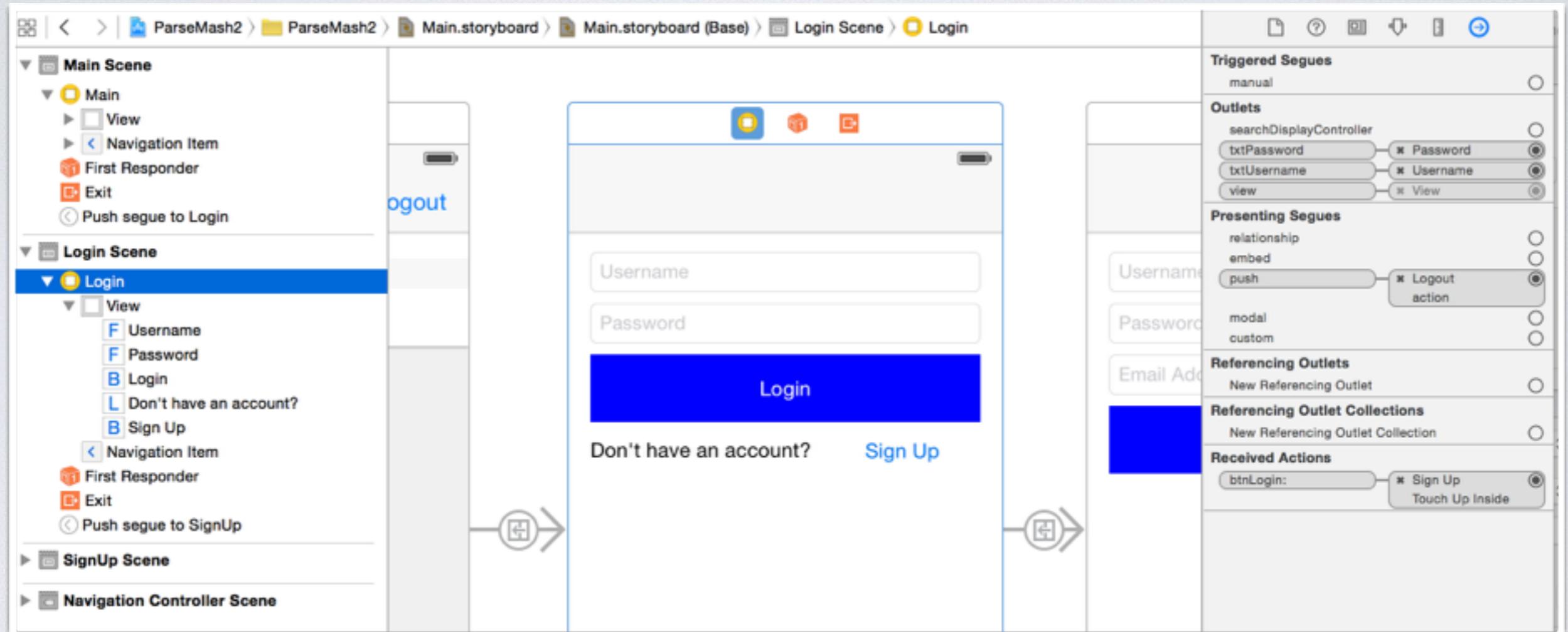
```
@IBOutlet var txtUsername: UITextField!
@IBOutlet var txtPassword: UITextField!

override func viewDidLoad() {
    super.viewDidLoad()
    self.navigationItem.hidesBackButton = true
}

override func didReceiveMemoryWarning() {
    super.didReceiveMemoryWarning()
    // Dispose of any resources that can be recreated.
}

@IBAction func btnLogin(sender: AnyObject) {
    PFUser.logInWithUsernameInBackground(txtUsername.text, password:txtPassword.text)
{
    (user: PFUser!, error: NSError!) -> Void in
    if user != nil {
        self.navigationController?.popToRootViewControllerAnimated(true)
    } else {
        let alert = UIAlertView()
        alert.title = "Sorry!"
        alert.message = error.userInfo!["error"] as NSString
        alert.addButtonWithTitle("OK")
        alert.show()
    }
}
}
```

Add your IBAction and IBOulet Connections to the Login View



Program Your SignUpViewController

```
@IBOutlet var txtUsername: UITextField!
@IBOutlet var txtPassword: UITextField!
@IBOutlet var txtEmail: UITextField!

@IBAction func btnSignUp(sender: AnyObject) {
    let username: String =
self.txtUsername.text!.stringByTrimmingCharactersInSet(NSCharacterSet.whitespaceAndNewline
CharacterSet())
    let password: String =
self.txtPassword.text!.stringByTrimmingCharactersInSet(NSCharacterSet.whitespaceAndNewline
CharacterSet())
    let email: String =
self.txtEmail.text!.stringByTrimmingCharactersInSet(NSCharacterSet.whitespaceAndNewlineCha
racterSet())
```

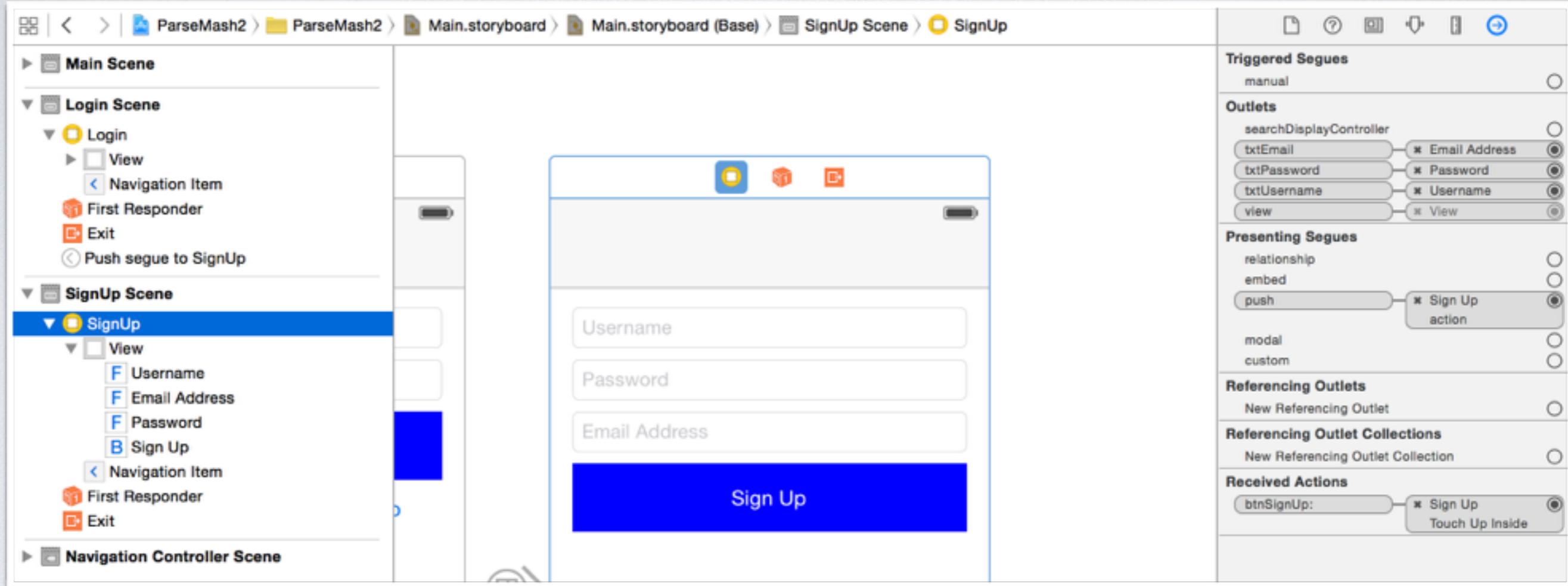


Program Your SignUpViewController

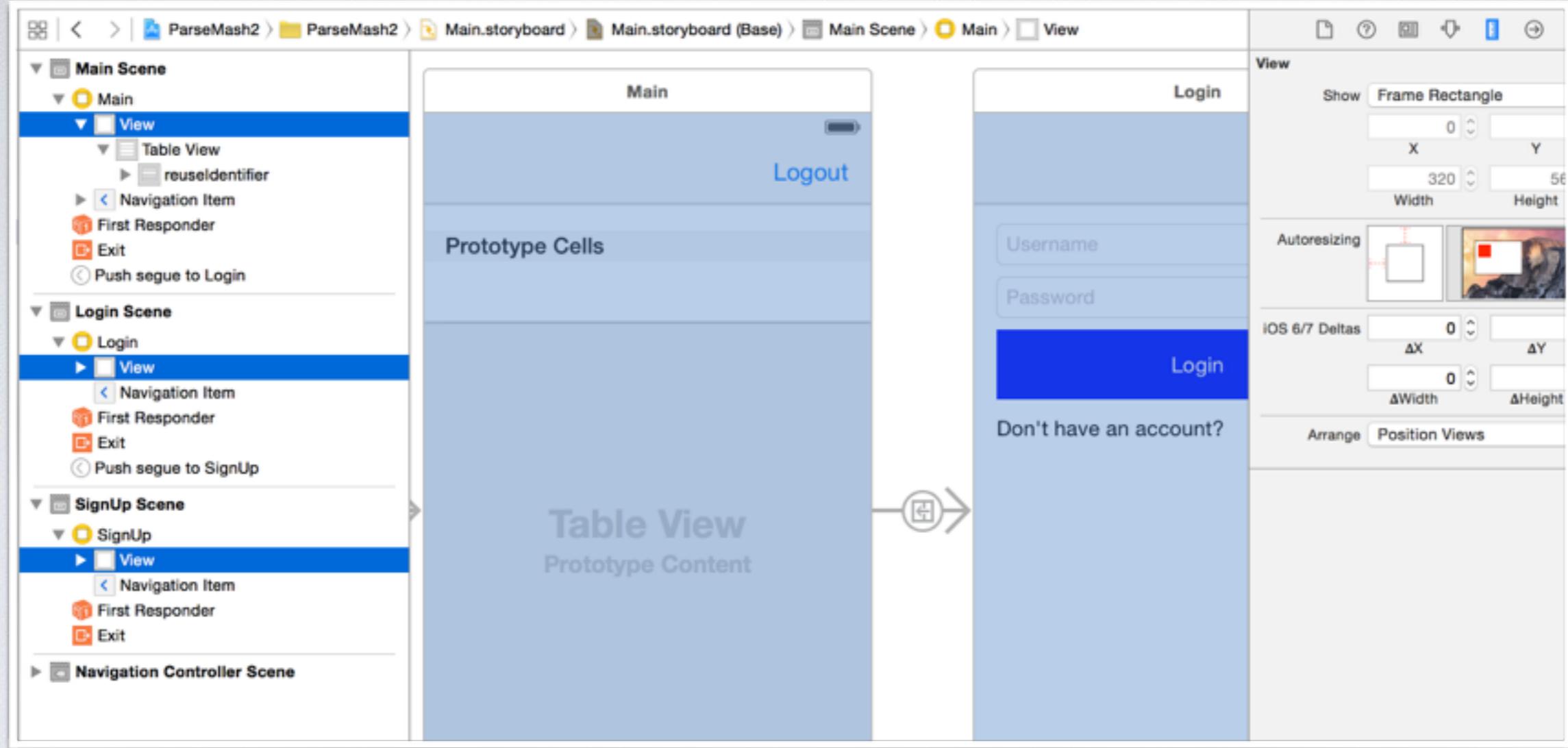
```
if (countElements(username) == 0 || countElements(password) == 0 || countElements(email) == 0)
{
    let alert = UIAlertView()
    alert.title = "Oops!"
    alert.message = "Make sure you enter an username, password, and email address!"
    alert.addButtonWithTitle("OK")
    alert.show()
}
else
{
    var newUser = PFUser()
    newUser.username = username
    newUser.password = password
    newUser.email = email

    newUser.signUpInBackgroundWithBlock {
        (succeeded: Bool!, error: NSError!) -> Void in
        if error == nil {
            self.navigationController?.popToRootViewControllerAnimated(true)
        } else {
            let alert = UIAlertView()
            alert.title = "Sorry!"
            alert.message = error.userInfo!["error"] as NSString
            alert.addButtonWithTitle("OK")
            alert.show()
        }
    }
}
```

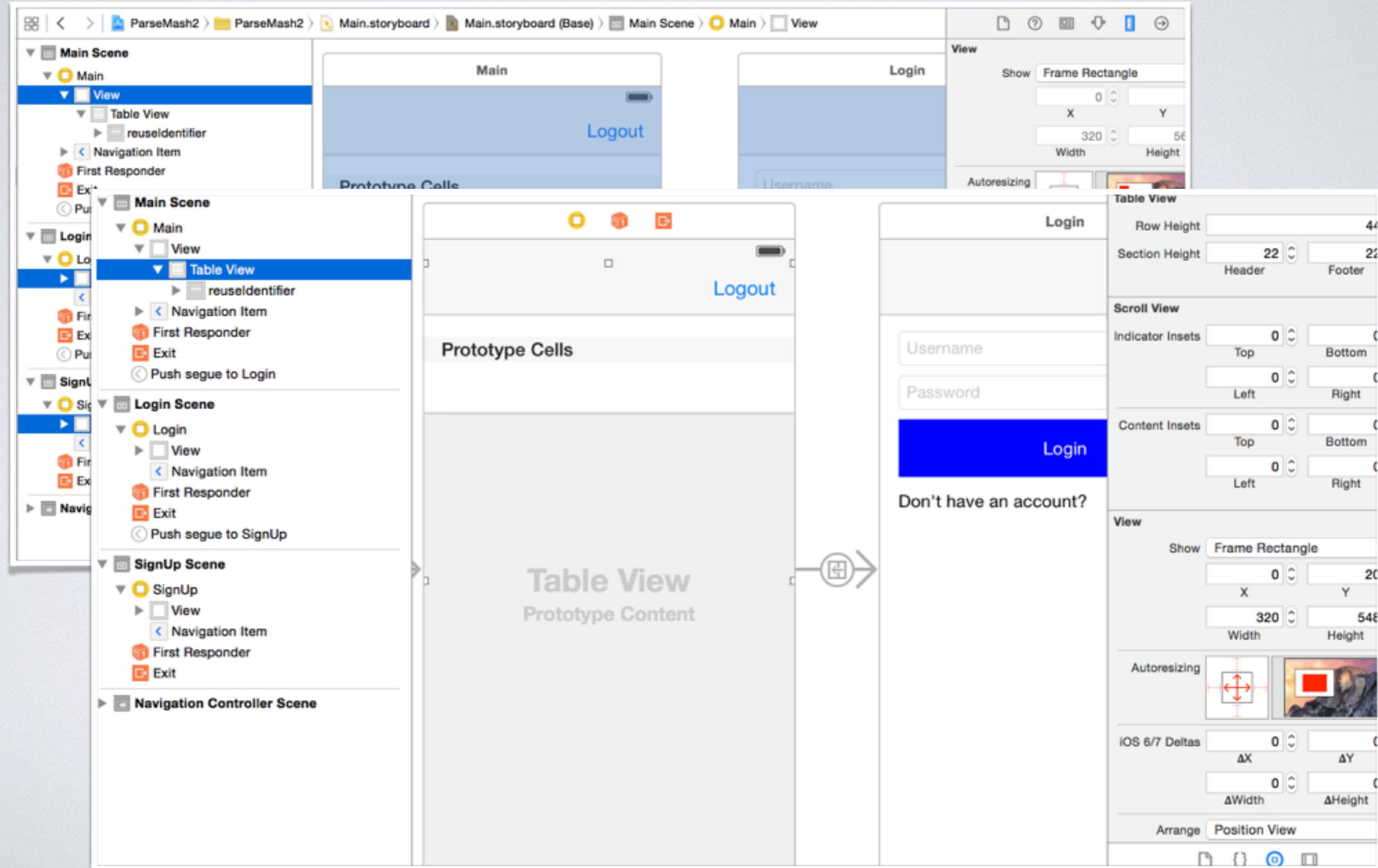
Add your IBAction and IBOulet Connections to the SignUp View



Clean Up - Because We Aren't Using Auto Layout



Clean Up - Because We Aren't Using Auto Layout



Don Miller
don@groundspeedhq.com
@donmiller

**Or follow @groundspeedhq
groundspeedhq.com**

