

Mobile Apps Development for Journalists

Week 5 - Fall 2019
University of Texas at Austin
School of Journalism

Jeff Linwood
jlinwood@gmail.com
@jefflinwood

Agenda

- Storyboard, Outlets, Actions
- Working with Maps and Locations

Storyboards

- User Interface Tool for iOS App Development
- Can connect to Swift code using Outlets and Actions

Outlets

- Connections between Storyboard and Code
- Variables in Swift View Controller classes
- Properties can be changed
- Easiest to add in Assistant View

Assistant View

The screenshot displays the Xcode Assistant View interface. On the left, the Project Navigator shows the file structure for 'OutletsAndActions', including 'Main.storyboard' and 'MainViewController.swift'. The center pane shows the 'Main View Controller Scene' with a storyboard layout containing a 'View' and a 'Storyboard Entry Point'. The right pane shows the Swift code for 'MainViewController.swift', which includes comments and method overrides for 'viewDidLoad()', 'didReceiveMemoryWarning()', and 'prepareForSegue()'. The right sidebar contains the 'Custom Class' section (set to 'MainViewController'), 'Identity' fields, 'User Defined Runtime Attributes', and a 'Document' section with a 'Label' and 'Object ID'. At the bottom right, there is a 'View Controller' - A controller that manages a view. section with icons for 'View Controller', 'Storyboard Reference', and 'Navigation Controller'.

```
1 //
2 // MainViewController.swift
3 // OutletsAndActions
4 //
5 // Created by Jeffrey Linwood on 9/20/16.
6 // Copyright © 2016 Jeff Linwood. All rights reserved.
7 //
8
9 import UIKit
10
11 class MainViewController: UIViewController {
12
13     override func viewDidLoad() {
14         super.viewDidLoad()
15
16         // Do any additional setup after loading the
17         // view.
18     }
19
20     override func didReceiveMemoryWarning() {
21         super.didReceiveMemoryWarning()
22         // Dispose of any resources that can be
23         // recreated.
24     }
25
26     /*
27     // MARK: - Navigation
28
29     // In a storyboard-based application, you will
30     // often want to do a little preparation before
31     // navigation
32     override func prepareForSegue(segue:
33     UIStoryboardSegue, sender: AnyObject?) {
34         // Get the new view controller using
35         segue.destinationViewController.
36         // Pass the selected object to the new view
37         // controller.
38     }
39     */
40 }
```

Custom Class
Class: MainViewController
Module: Current - OutletsAndA...

Identity
Storyboard ID:
Restoration ID:
 Use Storyboard ID

User Defined Runtime Attributes
Key Path | Type | Value

Document
Label: Xcode Specific Label
Object ID: BYZ-38-tOr
Lock: Inherited - (Nothing)
Notes:
No Font

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.

Creating an Outlet

- Drag an image view, button, label, or other element onto the storyboard
- Hold the control button down and click and drag between the new item and the view controller
- Release and name the outlet
- Practice!

Actions

- Swift functions that run when something happens - a user taps a button, a switch is changed
- The **func** keyword
- Similar to outlets, but creates a function, not a variable
- Be sure to change Outlet to Action in Pop Up
- Use Assistant View

Location and Mapping Frameworks

- CoreLocation - <https://developer.apple.com/documentation/corelocation?language=swift>
- MapKit - <https://developer.apple.com/documentation/mapkit>
- <https://developer.apple.com/maps/>

Location Functionality

- Get user's current location at a point in time
- Get notified when user moves a significant amount of distance
- Get notified when the user enters a geofence

Map Providers for iOS

- Apple MapKit - <https://developer.apple.com/maps/>
- Google Maps SDK for iOS - <https://developers.google.com/maps/documentation/ios-sdk/>
- MapBox - <https://www.mapbox.com/>
- ArcGIS SDK for iOS - <https://developers.arcgis.com/ios/>

Location and Privacy

- Your iOS App will have to ask the user for permission to get their location
- Choices for tracking user's location
 - Always - the application will get location updates in the background
 - When in use - only when the application is running

Provide Privacy in Info.plist

- Add a row, with a key value of:
- `NSLocationWhenInUseUsageDescription`
- `NSLocationAlwaysAndWhenInUseUsageDescription`
- And a String value with a message such as "This app would like to use your location for...."

Location and Privacy Concerns

- What are some concerns that users might have about their location being used by an app?
- How could these be alleviated or mitigated?

Getting Started with Location

- import **CoreLocation** framework
- Initialize a Location Manager
- Set the delegate on the location manager to self
- Request authorization from the user (the popup dialog)
- Tell the Location Manager to Start Updating Locations

Managers and Delegates

- We haven't worked with managers or delegates before
- **CLLocationManager** - responsible for getting location updates from the phone and passing them to the delegate
- **CLLocationManagerDelegate** - your code that does something with updates
- Need to set the delegate on the manager to **self**

Implementing a Delegate

```
import UIKit
import CoreLocation

class ViewController: UIViewController, CLLocationManagerDelegate {

    let locationManager = CLLocationManager()

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
        locationManager.delegate = self
        if (CLLocationManager.authorizationStatus() == .notDetermined) {
            locationManager.requestWhenInUseAuthorization()
            locationManager.startUpdatingLocation()
        }
        if (CLLocationManager.authorizationStatus() == .authorizedWhenInUse) {
            locationManager.startUpdatingLocation()
        }
    }

    func locationManager(_ manager: CLLocationManager, didUpdateLocations locations:
[CLLocation]) {
        print("Received Location Update")
    }
}
```


Apple Documentation

- CLLocationManager - <https://developer.apple.com/reference/corelocation/cllocationmanager>
- CLLocationManagerDelegate - <https://developer.apple.com/reference/corelocation/cllocationmanagerdelegate>

Testing the App

- In Simulator, go to the Debug Menu, and choose the Location sub menu, then pick the run, bike, or freeway drive.
- You can also set a custom location, or even use location track files

Review of CoreLocation

- Privacy Permission in Info.plist
- import CoreLocation framework
- Initialize a Location Manager
- Set the delegate on the location manager to self
- Request authorization from the user (the popup dialog)
- Tell the Location Manager to Start Updating Locations

Map Tutorial

- Building Mapping Apps with Swift e-book
- Map Tutorial - Chapter 1

Map User Location Tutorial

- Building Mapping Apps with Swift e-book
- User Location Tutorial - Chapter 2

In Class Exercise

- Team up in groups of 2 or 3, and brainstorm several ideas for journalism-related applications you could build that would use location or mapping technology
- Will share with class