Lecture 1 Notes
6:30-9:30PM, 22 June 2015

6:30 Introduction

Learn how to make Apps on the iPhone (and iPad)

Swift: New language replacing Objective-C (20%)

Xcode: Development and deployment environment (30%)

Foundation: Common w/Mac OS X (10%)

Cocoa Touch: Libraries specific to iOS, especially UIKit (50%)

Management: https://developer.apple.com/account/overview.action (3%)

Rapidly evolving environment: 4 versions of Swift; 4 releases of Xcode since intro

6:40 Logistics:

- Prerequisites
    - Experience with a strongly typed language
    - Experience with an object oriented language

        - 3 college semesters or 1 year professional experience

- Meeting times and dates
- Required sections: solidify this week
- Teacher Intro, TF intros
- Please read syllabus cover-to-cover
- The Strict Parts: Attendance, Academic honesty, Assignments
- Time commitment
- Grading
- Final Project
- Course website
- Apple Developer sign-up from your registered e-mail

6:55 Interdisciplinary class (in no particular order)

iOS is the center – all else will serve the practical application of iOS

- Traditional software engineering: breaking up and re-assembling a problem
- Learning a new language

- Navigating a complex IDE: 100's of bells and whistles
- Release engineering: Apple's conventions, code signing
- Event driven programming: handlers, background threads, observers, notifications
- Mobile visual design
    - Dividing up functionality, creating transitions
    - Obeying Apple's UI design conventions
- Working with a large library of Frameworks
- Staying connected: working with networks and servers
- Other specifics of a smart handheld
    - Security: Code signing App sandboxing
    - Notifications, barge-in phone calls
    - GPS: in-app, geo-fencing, variable accuracy
    - Limited power
    - Limited / highly variable Internet
    - Complex finger gestures
    - 3-axis accelerometer

7:20 Swift: A modern grab-bag (in no particular order)

- With enough to worry about (see above): dump as much “baggage” of C as possible
- Built-in dynamic storage types: Arrays and Dictionaries
- Type safety
- Type inference
- Syntactic sugar for compact code
- Rich standard library
- Automatic memory management
- Solving the sentinel value problem: Optionals
- Object oriented
- Functional programming
    - Anonymous Functions / Closures
    - Overloaded operators

    - Read-only enforcement

- Speed & small footprint: fully compiled
- Generics
- Available for not just iOS – Mac OS X, and Linux coming soon
- Readings: See course syllabus

7:45 Break, fill out surveys

8:00 Continue Swift discussion as needed, questions on everything so far

8:15 Xcode

- Starting a playground
- Entering code
- Seeing the results

- Seeing errors
- Starting a project
- Laying something out in Storyboard
- Running it in the simulator

8:30 Quick demo, and 2nd week preview: Game of Life

8:40-end AMSAP: As Much Swift As Possible