Welcome to CSE 438S
Mobile Application Development
“iPhone Class”

Course Information

- **Instructor**
  - Todd Sproull
  - todd@wustl.edu
  - Jolley 536
  - Office Hours by Appointment

- **Classrooms**
  - Steinberg 105
  - Whitaker 316 (Mac Lab)

- **Time**
  - Mondays and Wednesdays 11:30 AM – 1 PM

- **Course Website**
  - http://research.engineering.wustl.edu/~todd/cse438/
  - Also available on Canvas

- **Head TA**
  - Jordie Chisam
  - jachisam@wustl.edu

- **We will use Piazza to answer questions**
  - Please sign up, I emailed everyone an invite
Requirements

- **CSE 247**

- **Access to an Intel-based Macintosh**
  - Running macOS 10.14 or later
  - iPhone SDK Xcode 10.2.1 and iOS 12
    - We will use Xcode 10.2.1 the entire semester, I strongly recommend not upgrading to a newer version of the software

- **Textbook**
  - None, we will use lecture slides and the developer.apple.com website

- **Owning an iPhone or iPod Touch not required**
  - We will use the simulator throughout the semester
  - Final projects may target an iPhone or iPod Touch

Stanford CS193p

- This course is based on cs193p taught at Stanford by Evan Doll and Alan Cannistraro
  - Lectures and slides available on iTunes

- Many of the lectures and programming assignments come from this class
  - Initial assignments are identical
  - Later assignments somewhat different

- Consider taking the iTunes course if that suits your personality
Copyrights, Patents, Fair Use...

- Everything discussed in this class and on the website is completely OPEN and FREE
  - Do whatever you want with it

- The goal of this class is to share as much information as possible
  - Open discussion of topics and ideas

- If you have a great idea and do not want others to implement it and sell it DO NOT discuss it here
  - If you choose to discuss it, we can probably improve it

- You are free to become an Apple Developer ($99/yr) and sell anything you create in this class
  - Or implement another student’s great idea and sell it

What is this class all about?

- Building applications on iOS Devices
  - iPhone, iPad, iPod Touch, Apple Watch, Apple TV
- Learn new programming languages
  - Swift
  - Objective-C
Cocoa Touch and iPhone SDK

- Based on Cocoa
  - API used to develop software on Mac

- Provides rich starting point for exploring app design

- Shows real-world implementations of OO design patterns

- Designs learned on iPhone translate directly to Mac OS X

Swift

- Apple’s latest programming language to develop OS X and iOS applications

- New language only a few years

- Combines many of the latest programming techniques in an easy to learn language
Grading

• 4 lab assignments during the semester
  – 70% of your final grade

• Final Project
  – Work on something that can make a difference
    • Start thinking about your project today!
  – 30% of your final grade

Questions?
iPhone OS Overview

iPhone
Mac OS X

- Cocoa
- Media
- Core Services
- Core OS
iPhone / iPad

• Core OS
  – OS X Kernel
  – BSD
  – Sockets
  – Security
  – Power Mgmt
  – Keychain
  – File System
• **Core Services**
  – Collections
  – Networking
  – SQLite
  – Net Services
  – Threading
  – Preferences

• **Media**
  – Core Audio
  – Audio Mixing
  – Audio Recording
  – Video Playback
  – JPG, PNG, TIFF
  – PDF
  – Quartz (2D)
  – Core Animation
  – OpenGL ES
• **Cocoa Touch**
  – Multi-Touch Events
  – Multi-Touch Controls
  – Accelerometer
  – Localization
  – Alerts
  – Web Views

---

**Development**

• **Tools**
  – Xcode
    • Storyboard (formerly Interface Builder)

• **Frameworks**
  – Foundations
  – UIKit

• **Languages and Runtimes**
  – Swift
  – Objective C
Cocoa Touch Architecture

Cocoa Touch

UIKit
User interface elements
Application runtime
Event handling
Hardware APIs

Foundation
Utility classes
Collection classes
Object wrappers for system services
Subset of Foundation in Cocoa

Object Oriented Programming
Object

Behavior

Thing

behavior
doSomething
Message

“doSomething”

State

state

behavior

Thing

doSomething

count

flag

doSomething
Other Objects as State

- **Thing**
  - count
  - flag
  - helper
  - doSomething
- **otherThing**
  - doMore

Outlets

- **Controller**
  - slider
  - label
  - updateLabel
- **Value:** 100
Target/Action

Controller
- slider label
- updateLabel

target

action - ‘updateLabel’

Value: 100

Demo
Recap

- Keep logic separate from interface elements
- Outlets connect controllers to views
- Use target/action to customize behavior