# INTRODUCTION TO IOS APPS AND OBJECTIVE-C

CS 496 - SPRING 2014

# Overview

- Why go native?
- iOS Overview
- Objective-C
- Simple example (time permitting)

# Why go native?

- Performance
- Features
  - Newest libraries
  - Hardware access
  - Tools

# iOS Overview

- Object-Oriented
- Model-View-Controller
- Libraries
- Developer Community (Cocoapods)
- Tools

https://www.apple.com/pr/products/ios/ios.html

# Model View Controller

# Libraries

- UIKit — User Interface
- Foundation — Data structures
- CoreMotion — Accelerometer and Gyro
- CoreLocation — GPS, Compass, & iBeacon
- CoreBluetooth — Low level Bluetooth
- So many more…

# Open Source Libraries

- Active developer community
- 8th most popular programming language on Github in 2013
- 36k+ open source repositories in 2013
- Cocoapods — easy way to incorporate third party repositories (cocoapods.org)

http://adambard.com/blog/top-github-languages-for-2013-so-far/

# Tools

- Free on Mac App Store
  - Xcode
    - IDE
    - Instruments
    - Interface Builder/ Storyboard

http://upload.wikimedia.org/wikipedia/fr/d/da/Logo_xcode.png

# Objective-C

- Strict superset of C
  - C++ also supported
- Additions to make C object-oriented
  - Classes
  - Properties
  - Dynamic

# Classes

- Inheritance — Ex: Dog <- Animal
- Method Overloading — Ex: [aDog run] != [aCat run]
  - Specialize methods to class
- Properties — Ex: aDog.age
  - Attributes of an instance
- Protocols — Ex: Dog <TrainedDog>
  - Can respond to certain methods i.e. sit or stay

# Methods

- Class Methods
  - + (instancetype) array
  - NSArray *newArray = [NSArray array];
- Instance Methods
  - - (id) objectAtIndex:(NSUInteger)index
  - id aObject = [newArray objectAtIndex:0]

# Generics — id

- id: a very special type in Objective-C
  - Represents any object type: SomeClass *
- Defers type checking to runtime
- Alternative to C++ templates
- Example:
  - NSArray stores and returns type id

# Memory Management

- Dynamic Memory Allocation
  - [[SomeClass alloc] init]
    - Allocate memory
    - Initialize for given class
- (Automatic) Reference Counting
  - +1 count when using memory, -1 when done, free when (count == 0)
- All memory on heap; always use (SomeClass*)
  - Pointers, Pointers, Pointers

# Properties

- @property (nonatomic,strong) NSDate *birthday
  - nonatomic: not thread safe (Opposite: atomic)
  - strong: +1 reference count (Opposite: weak)
  - Object type: NSDate *
  - Property name: birthday
  - Access
    - Get: anInstance.birthday or [anInstance birthday]
    - Set: anInstance.birthday = ?? or [anInstance setBirthday:??]

# DEMO