# iPhone Application Development

- Fall 2016 - Class #1

# Class 1. iPhone Environment

- Why develop an iPhone app
- Pre-requisites for the class
- HW/SW needs.
- Apple License categories and what will be needed for the class.
- Books
- Grading Policy
- Help
- Demo of SDK environment at high level. Cursory tour of Xcode and files generated by Xcode for "Hello COD" Application.
- Sample Final Projects from Previous Year's Classes.
- Programming exercise 1. Simple application to help you test your environment setup and get your feet wet.

# Why Offer iPod/iPhone/iPad Development Class

- Fastest growing market
  - 1 million iPad sold in record amount of time. Apple has 20+/-% share of mobile smart phones. Drove Blackberry and Nokia out of business. Put AT&T as the predominant carrier. Power of Apple is amazing! Sure there is Android.
  - Integration of iPhone and sensors is the new emerging world.
  - All of this translate to market for developers
- Cookbooks and podcasts available on web do not meet need of many students who need interaction and direction to learn new skills. Steep learning curve as you need to learn:
  - Apple OS in case you have not used mac's before,
  - Objective C - is the language for developing all apple products,
  - Xcode IDE and then start writing apps!
- Younger generation relate to iPhone more than not so young population. This would be something that students would want to learn.
- Because it's fun!

# Pre-requisites for the Class

- Apple leverages Frameworks, and Patterns heavily. Good understanding and working experience with any Object Oriented language is a must.
  - Let me know if you are rusty on some topics, I can point you to exercises to refresh your concepts.
- Apple developed Mac apps and iOS using Objective C. Originally developed by NeXT, Objective C incorporated OO concepts on top of C. Language is somewhat awkward to express and not truly OO when compared to modern C++ or java.
- Apple is moving to Swift - new OO language. Apple is conservative and incrementalist when it comes to introducing new HW or SW.
  - All of iOS Framework is in Objective C, or C libraries. Expect that to change over time.
  - Current official 2.0 version of Swift is being replaced with 3.0 in next release of Xcode
- Our Fall 2016 class will use Objective C and mention Swift as appropriate for easier transition for all of you in future.
  - Quite literally millions of apps out there are written using Objective C. Tons of books and examples on web are in Objective C
  - Most if not all of the employers will require you to have knowledge of Objective C.
  - That said, Apple will slowly transition everything to swift.

# HW and SW

- Access to any Intel based apple computer:
  - MacMini basic model about $600 – user provides keyboard, mouse, and screen
  - iMac 22" about $1100
  - Mac laptops or MacBook Air is fine.
  - Access to COD Mac resources for HW assignments. However recommend students to buy Mac if this is something they would like to do. Apple offers education discount on some of the models in case you are planning to buy Mac.
- No need to have iPod/iPhone/iPad for this class. Apple's simulator embedded with SDK is all that's needed. Topics such as Core Location Services and Accelerometer would not be exercised on the simulator. However software and API discussion will be adequate to understand. Class will be using simulator.
- Need 2 USB drives for the class. Please put a text file with you name on each USB. I may take a week or two to evaluate your homework so better to keep 2 USBs handy. Generally you will need less than 100 MB for each exercise.
- For Fall 2016 term we will use xcode 7.3.1. Please do not upgrade to newer version – it's not easy to roll back to previous version. Apple introduces significant changes to UI in major releases. You will be lost if you try to use upcoming 8.0 version while class is following 7.3.1.
- **Change setting on your personal mac to disable automatic update.** It's coming!

# Apple License Options

- Individual Membership is free. You can download Xcode and use resources for free. You can load app on your USB tethered iOS device for free. You cannot deploy your app on iTunes store for others to use, even if you are selling for free.

- In order to deploy your apps on iTunes, you will need a license. It's a Developer License from Apple for $99 a year and allows licensed developers to deploy their apps on App Store for revenue or free. Deploy some free apps and get your feet wet!

    - Some of the prospective employers look for apps on iTunes. The act of publishing app requires you to be compliant with apple's policy.

- Enterprise membership is for organizations to create proprietary apps designed and distributed exclusively to organization's members. Apps deployed under enterprise license must be free. For e.g. AT&T has app for installers to activate your DVR on new service. App collects all the info and runs several tasks.

- MFi License is for hardware component and documentation. Companies developing iPhone, iPod, iPad, iWatch, AirPlay etc. accessory get access to protocol specifications, hardware connectors and components tools.

- There is University Program where you develop and deploy your apps on your iPhones for free for the duration of the semester. We will not be using this program as most of the testing can be done on the simulator and Apple allows to put apps on your phone.

# Books

- Most of the material from this class is from Apple's SDK website. It's the most reliable and precise document you will ever find. However it may not meet needs of student who could benefit from detailed direction with detailed reasoning and students who are new to Objective C.
- Week 2 and Week 3 on Objective C as well as reference guide to Objective C through out the class:
  - Programming in Objective-C 2.0 (3+ Edition) by Stephen G. Cohen
- I have used few books in past. You don't need to get these books. Few worth a mention are:
  - iOS 4 Programming Cookbook (1st Edition) by Vandad Nahavandipoor
  - The iPhone Developer's Cookbook: Building Applications with the iPhone 3.0 SDK (2nd Edition) by Erica Sadun – used during Fall 2010 offering of the class
  - The Big Nerd Ranch Guide - was favorite of some students in 2013.
  - iOS programming books generally are out of date by the time they get published. Some are good some are not. Our COD Library carries some books, check them out.
- Recipe and Concepts – difference between language focused and application focused books. Need good mix of both.
- Internet is another rich source of information. Realize that it can confuse you.
- I will post slides on blackboard after the class as I generally update them just before our class.

# Grading Policy and Contact Info

- Class Participation - 10%
- HW projects to build sample apps to check your understanding of the material.  About 5 projects - 60%
- Final project - 30%

- No exams, no quizzes. All hands on app building.
- Best way to get hold of me is by e-mail. My e-mail address is parikhj@cod.edu. I read mail twice a day on most days.
- About me.
- Your back ground in OO, Mac OS, and Integrated development environment

# Tour of Mac - Spaces, Zoom

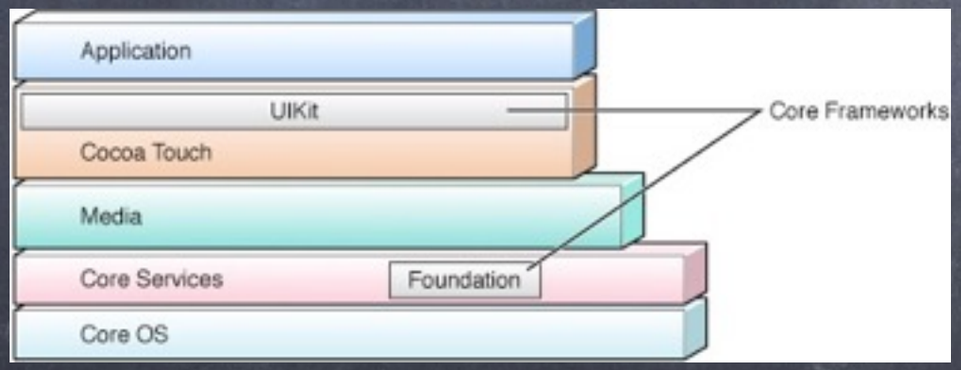
- Spaces provides you virtual screen. You can set up two, four, six, eight or even sixteen full size virtual screens. You can then move around between virtual screens and get a screen full of real estate. Web windows in one screen, Mail in another, iTunes in third, xcode in another one and no clutter.
  - Mission Control will provide a list of current Desktops. To add virtual desktops Mission Control->Click + icon on upper right corner.
  - To switch between spaces (work spaces or virtual desktops) click Ctrl-left/right/up/down keys.
  - To Move an app window goto Mission Control, drag app window to the desktop/space you desire.
  - To delete a desktop goto Mission Control and move your mouse pointer to desktop your desire to close/delete. You will see X sign, click on it and it's gone!
  - Refer to Apple's documentation at https://support.apple.com/en-us/HT204100
  - Expose and Spaces are very useful features imported from 1990's Unix workstations. You will love it once you get used to it.
- Set up keyboard or mouse triggers for automatic choices. Control - Up arrow takes you one screen up, Control - Down arrow take you one screen down and so on for right and left arrows.
- Zoom or Magnify screen
  - Very important feature to me!! Be sure that you have it set up - system Preferences -> Accessibility -> Zoom and select scroll gesture with Control modifier key to zoom.
  - Once setup, you can show me code you are having difficulty with quite easily.

# Tour of Mac

- File Manager and files on USB/drive for your project
- Delete key is really backspace and forward delete key as you are accustomed to is ? Can program this but remember once you reboot COD computer will go back to what it was before.
- Force Quit -> Control-click dock icon and find force quit
- Clover or Apple Key. A cloverleaf-like symbol commonly used in Scandinavia as an indicator of cultural locations and places of interest. This is command key - a modifier key to things you would enter. Apple short cuts are achieved with this key. Try - Apple-tab key combo - it is equivalent to Alt-tab in windows world.
- Try Settings for Mission Control, Application Windows, Show Desktop, Show Dashboard
- Settings on your COD Mac will revert back to original after you reboot.

# The Cocoa Environment

- Cocoa is an Object Oriented framework that provides runtime environment for application running on Mac OS, and iOS.
- Framework is something that you build upon. An incomplete architecture without your code to form a complete application. Cocoa provides fantastic framework upon which you will build applications for iPhone, iPad, iPod, or Mac devices.
- Cocoa in iOS world. Picture from http://developer.apple.com website. We will be doing development in Application layer of the Cocoa environment. Apple Cocoa iOS provides all of the other layers to us.




Xcode is Apple's IDE that integrates Cocoa and provides an integrated development and test environment.

Rich environment that supports all aspect of Software development, documentation, modeling, data,

writing, building and testing code, performance measurements, debugging etc. etc.

# Xcode SDK Environment

- General Look and Feel.
    - Toolbar => Overview, Action, Build, Build and Go, Tasks, Info, Editor, Search
    - Favorite Bar - turn on if you don't see it => View > Layout > Show Favorites Bar
    - Groups and Files list
        - Everything related to the project - files, folders, sounds, the whole kitchen sink
        - Execution Targets
        - Errors and Warnings
        - Linked files such as NIB, implementation
        - Bookmarks, search results for easy API lookup
        - SCM - mentioned for posterity but not used in the class
    - Status Bar
    - Detail View - splits e.g. to show code

# Xcode SDK Environment II

- Development environment offers very powerful tools. We will cover these tools as we get more introduce more topics.
- Documentation
- XIB, Storyboard
- Debugger
- Instruments to keep track of memory usage
- Instruments to keep track of objects
- Third party tools such as Clang static analyzer.
- We will be using xcode 7.3.1 version. DO NOT UPDATE VERSION on your person mac. Apple introduces new products during Sept/Oct timeframe and upgrade Xcode at that time. I have been playing with the next beta version which will become official version around sep/oct. It is very important for you to not update your version on your machine. Be careful when you click on OS Updates, read description and uncheck xcode updates. Many Many students in past have struggled!!!
    - Changes to iOS code and interfaces are extensive in addition to changes to Xcode GUI for Swift 3.0 and iOS 10.0.

# iPhone and iPad Simulator

- Xcode supports a built in simulators for iPhone and iPad. It should be first platform to start testing your app.
- Simulator is compiled for Intel chips where as code runs on ARM chips. There will be differences. That said, Simulators are darn close to the real thing.
- Simulators can not mimic hand gestures at pace a human hand would do – there are commands to simulate physical actions such as pinch, tap etc. are fairly good. On simulator menu you will see commands to simulate gestures. Good idea to make a web page out of it for your quick reference.
- On Xcode 7, simulators for iPhones with different resolutions is available too – it's as real as it gets.
- HW features such as GPS, gyro, signal strength etc. are not available on simulators – they are ways to simulate some.
- If you plan to deploy a real app, you must test on real phone, simulator is not a substitute for phone.
- Realize that once you update OS version on the actual phone, there is no way to turn back. If you are planning to deploy application that support various phone version, you need to keep one of each OS version phone around.

# Mini Tour of Xcode

- Header file lookup
- API Reference lookup
- Code folding - collapse what you don't care to see and un-clutter the space
- Code completion - if not enabled goto Xcode->Preferences, check Immediate from Automatically suggest pop-up on the Code Completion->Code sense pane
- Access Documentation
- The Documentation Window
  - Option double click takes you to API search of a symbol in the code window
  - Search other ways to access documentation for the iPhone OS Library symbols and get comfortable with what works for you

# Brief Summary of Sections

- Refer to the exercise "Hello World" on xcode.
- @interface – describes the class, its data components, and its methods. @end signals end of interface description for the interface section.
- @implementation – contains the code that implements the interface.
- @program – contains the program code to carry out the purpose of the program.
- Typically professional programs will keep interface in one file, and implementation of individual interfaces in separate files. For this class we will keep all three in one file most of the time to easily look up things.
- Rules for naming variables, classes etc. xcode will help you check with an incorrect name starting with $name. Reserved words such as int, short etc. can't obviously be used. Once again xcode will help

# Reference Material

- Xcode Overview:
  - https://developer.apple.com/library/prerelease/ios/documentation/ToolsLanguages/Conceptual/Xcode_Overview/index.html#//apple_ref/doc/uid/TP40010215-CH1-SW1
- Introduction to Developing iOS Apps
  - https://developer.apple.com/library/prerelease/ios/referencelibrary/GettingStarted/RoadMapiOS/index.html#//apple_ref/doc/uid/TP40011343-CH2-SW1

# Xcode Version 7 (was Xcode Version 6, and was Xcode Version 5 before that)

- Change is constant!
- Current version on COD machines is 5.1.1 for Xcode. Apple has released 4.2.x version for OS X Leopard and 4.2 for Lion. We are trying to go to 4.x version but there are different paths - and IT is evaluating those paths. We will know for sure which way we are going - Lion and then to 4.x or 4.x on Leopard.
- Regardless of the version exercise is the same - newer version is very convenient to use and offers nifty features so good to keep up with new version but for first few weeks it doesn't really matter much.
- Apple has published Xcode 6 beta version. We will not use beta version for this class. It's possible that during the duration of this class Apple may make 6.1 + as an official version. We will start with 5.1.1 version and end this class with 5.1.1 version. Please do not update xcode on your home machines. There are migration issues and students who migrated to beta version in past semester have spent a lot of additional hrs.
- I will mention new upcoming features in Xcode 6.1 as we work exercises so you will get a good idea of what's coming.

# Help!
# My Contact Info

- Best way to get hold of me is via e-mail. I read e-mails posted to parikhj@cod.edu twice a day during weekdays.
- I can stop by 30 min before the class if you need one on one help with the material, lab or project. Do let me know couple of days ahead of time so I can plan my day accordingly.

# Hello World Step by Step

- **First app**
  - From Xcode - File->New Project
  - Under iOS Application choose Single View Application. Click Next.
  - Name it and store where you prefer - let's call it class 1. Provide Organization Name (I provided my own name), Organization identifier (I provided "mine"), pick language as Objective-C, Devices as iPhone, and check only Include Unit Tests and Include UI Tests boxes. Click Next and Save the project on desktop.
  - Hit Run and app should compile without writing one line of code. After build Xcode will bring up iPhone simulator, provided you have chosen right target.
  - You will see a  blank screen. You just wrote and ran first app!
- **XIB and Storyboard**
  - NeXT Interface Builder file extension. It's spec file for GUI written in XML.
  - You can edit XML or better yet use Interface Builder program to write specs to xib for an app.
    - In Xcode 3.2 and prior version IB was an independent program to design interface. Over the time it has evolved and in Xcode 4.x it's part of Xcode IDE.
    - A lot of books use XIB as they update their material so worth a mention.
    - On involved graphics design, Graphics designer would use IB/SB and generate xib for programmers to use it in their programs.
    - SB was introduced in 4.0 version and support for XIB will disappear from Xcode in future version

# Hello World Label

- GUI design canvass is Story Board. You can have several story boards in your app. On launch you get Main.storyboard
- Put Label on our Main.storyboard
- Click on label and put it on the view window (of controller.xib). Name the label. Get a feel for how you can align label around (tools->attributes inspector), change fonts (Control T to get font panel - universal in Mac environment)
- Build and Run!!!
- Change different target - for e.g. iPad or different version of the iPhone. No code changes generally needed for different targets.

# Hello World - Click and Action

- Add a Button from object library by dragging round rect to Main.storyboard window.
- Notice Button attributes: color, back ground, font etc. Here is where you can customize button. Change name from Button to "Hello"
- Add code for click action - open "Class-1ViewController.h" file and add "-(IBAction) btnClicked:(id) sender;" statement to .h file before @end statement.
- Above statement created what's called an event handler - "btnClicked". We need to connect this handler to touch action with iOS framework. Control click from button and you will see a line connect it to "View Controller" - a little icon on top of the Main.storyboard canvas window. More discussion on First Responder, Exit, and View Controller coming up in later classes.
- A popup with "btnClicked" will show up select it - Xcode will blink "View Controller" to tell you that it has now linked touch action on the screen to the handler. If we had more "IBAction" in .h file we would get list of them to select which handler to hook.
- Now if user click on "Touch Me", iOS framework will do it's magic and invoke your event handler called "btnClicked".
- Next we need to add code to take some action when button is clicked.
- Open ViewController.m file under class 1. It has blank template for an Objective C file that contains code for the handler.
- Notice a little yellow triangle that tells you that class implementation in not complete!

# Hello World - Code

- Add following code to .m file

```
-(IBAction) btnClicked:(id) sender {
 // use alert box to display message
        UIAlertView *alert =

    [[UIAlertView alloc] initWithTitle:@"Hello
        CIS 2840!"
              message:@"here I come!"
                 delegate:self
        cancelButtonTitle:@"OK"


        otherButtonTitles:nil];
          [alert show];

}
```

- Next we need to make connection between the Hello Button and our class method. Hold Control key and drag from the button to File's owner (yellow box) on the story board. You will get a menu with a section called "Sent Events" and you will see our method btnClicked: there. Select btnClicked. When user touches our Hello button, the framework will invoke btnClicked: method and code that we have in box earlier will get executed.

# Build and Run!!!

# Summary

- Xcode and iOS versions
- Apple's strategy and how these tools help you deploy applications fast!
- Tour of Basic Xcode capabilities and some discussion on Objective C
- No HW this week but do figure out where you will be doing HW at COD or home and allocate time. This class can be a lot of fun if you put effort.
- We will discuss objective C fundaments for next two classes.
- http://developer.apple.com is your best documentation source.
- Check out iTunes classes on iPhone Application Development - more time on app development material can only help!