CmpSci 491 IP
iOS Programming Seminar

Chip Weems
CS 342, weems@cs.umass.edu
Office Hours: Monday 10:30 - 11:45, More will be added, appointments welcome
Seminar

seminar  |ˈseməˌnær|
noun
a class at a college or university in which a topic is discussed by a teacher and a small group of students.

ORIGIN late 19th cent.: from German Seminar, from Latin seminarium (see seminary)
Motivations

* Mobile is cool, programming it is too
* I have this app I want to build...
* Get rich and retire early
* I’ve wanted to learn Swift since I was 4!
* Mobile environments are a major software engineering segment
Interview each other for the following information:

* Motivations
* Background
* Goals

On return, introduce the class to another member of your group. Group turns on video for their turn – everyone else turns off.
Course Structure

* Some initial lectures to get started
* Mix of student presentations/experimentation/lectures
* Individual and group work on apps, demos, problem solving
* Team project to build an app of your own choosing
What You Will Need

* Access to an Intel-based computer running OSX 10.15
  * Can use hackintosh, but be aware...
* Preferably a machine you can use in class
* Apple ID (e.g., Appstore ID), developer.apple.com account
* Latest XCode and simulator
  * Version 11.6 (don’t go to 12 — likely to release mid-semester)
What You Will Do

- Two in-class API presentations, as part of a group
- Some warm-up projects to get familiar with Xcode, Swift
- Business plan for your team's app
- Proposal presentation, two milestone demos
- Final presentation and report on your app
Grading

- Two API presentations 10% each: Depth, quality, engagement with class, use of time
  - Basic one in 1st half of semester, more advanced one in 2nd half

- Homework
  - Complete two warm-up exercises, keeping a journal of your experience (4% each)
  - "Business plan" consisting of six draft pieces (4% each) and a revised complete final version (8%)

- App implementation
  - Presentation of plan and app design (6%)
  - Two milestone demos (7% each) and a final demo (10%)
  - Final written report (10%) of what you learned, problems you overcame, and who did what

NO EXAMS (this is a seminar)
Grading Philosophy

* Participation - the value of the seminar is in having everyone engage with it
  * If participation is strong, I won't grade for attendance

* The purpose of the homework is to get you thinking in new ways - I want to see where you are in your thinking, even if it's not fully formed, so I can give you feedback

* I am generous with partial credit, but can only do that if you submit something! So late homework will have a 30% deduction (because that's probably more than I would take off for being incomplete)
Books

- Book: Beginning iPhone Development with Swift 5, Wallace Wang, Apress
  - Available as e-book, $30 - watch for sales
- Optional Reference: Pro iPhone Development with Swift 5, Wallace Wang, Apress (I can supply chapters)
- The Swift Programming Language (Apple - free)
Back to the Future

* iOS 14, XCode 12, about to be released
* DON’T go there
* XCode is flaky enough in “stable” versions, and 12 has major changes
* Every time this class runs, they update mid-semester
  * Once you do, you’re on your own, and Apple won’t let you go back
  * Best if we all stay with the same version - turn off auto-update
* Thankfully Swift 5 was a major update and won’t change soon
Other Resources

* XCode contextual help

  * Get XCode 11.6 from App Store (“stable” version)

* [https://developer.apple.com/swift/resources/](https://developer.apple.com/swift/resources/)
  * Significant changes in Swift 5
App Projects

* Two will be assigned as homework
  * Exploring APIs, getting familiar with Swift and Xcode
* Main project is your own choice
  * Team of two preferred - can be individual, particularly if prior experience
  * Project plan presentation with UI mockup, to get class feedback
  * Two milestone demos, with class feedback, then final demo and report
  * Will begin with business/project plan
What Makes a Good App?


* Out of hundreds of pages, basic ideas are in Themes, User Interaction, and Visual Design sections

* The Mobile HIG is a result of many years of actual experience and scientific study — it could be the basis of an entire course

* Sections after Visual Design illustrate the catalog of UI API elements that are available — a great intro to UI design options in iOS
What Makes a Good App?

Focusing on the needs of the user
The App is its UI

* Users experience the app through its interface
* The most incredible app will be seen as worthless if its UI is poor
* A great UI creates a positive feeling
* The UI affects how much people actually use the app, and recommend it
**Metaphors**

* Model the UI and the actions of the app on a familiar real-world analogy
* A natural UI shouldn’t require a user manual for the most common tasks
* Can extend a metaphor at deeper levels
  * But don’t overdo it
* Use standard controls when possible
* Metaphor is the basis of OOP
Direct Manipulation

- Touch interface allows direct control of objects on the screen
- Objects respond to gestures naturally
- Objects stay on screen while touched
- Responses should be immediate
- Orientation, motion also affect UI
See and Point

- Avoid keypad entry
- Present choices, tables, controls
- Easier for user to pick than to type
- Avoids extra error checking
Feedback

- Respond visually to every user action
- Show status progress for lengthy ops
- Audible feedback can’t be primary
  - Could be noisy environment, or sounds off
- Animation enhances experience, but isn’t the feedback focus of most tasks
User Control

- Let the user initiate actions
- Keep actions simple
- Allow cancellation
- Confirm anything irreversible
- Allow stopping at any point (it’s also a phone)
Aesthetics

- The appearance should fit the task
  - Simple and unembellished engineering app
  - Beautiful menu planning guide with food photos

- Keep it simple
  - Use controls in familiar ways
  - Follow iOS standard patterns
  - Aim for intuitiveness, minimal cognitive effort
Consistency

* Be consistent: logically arrange controls and keep in similar places across views
  * Don’t make users hunt for the same control on different views

* Similar controls should do similar things on different views
  * Use different controls for different behaviors
iPhone vs. iPad

- Small screen requires multiple views
  - Transitions have lower cognitive effort

- Larger screen can split view, obtain effect of multiple views with one
  - Full screen transitions have higher cognitive effort

- Cogito ergo some parts of UI design will be different between iPhone and iPad
Examples

- **Adjust settings**
  - iPhone app flips to back view
  - iPad app uses popover

- **Select from list**
  - iPhone app switches to list view
  - iPad app shows list in split view
Size and Resolution

🌟 Goal of iOS is resolution independence

🌟 UI elements drawn with vectors, but some require multiple images (selected automatically)

🌟 High resolution on small screen = better quality. Not more elements. Fingers don’t get smaller

🌟 Screen size change requires UI redesign

🌟 Increases developer effort, code size, user confusion -- keep number of formats small
Screen Sizes

- In most cases, extra space on different iPhone models is used to automatically improve UI experience:
  - Bigger entry area
  - More options in scroll list visible
  - More space between elements
  - Avoid feature differences between models
- Some larger screens have extra features
The App is also its Data Source

- For some apps, a large part of the value is in the back end
- Having a database or model that nobody else has
  - Example: Access to UMTA bus locations
  - Example: Detailed model of solar system orbits
- Updating the data/model can be a huge (i.e., valuable) effort — making that manageable can be a big deal
For Next Time

- Start thinking about ideas for apps you would like to build
- Be prepared to discuss one at start of class
- Get set up for starting development
Get Xcode

- Open App Store
- Select Develop
- Get and install
Developer Access

* [https://developer.apple.com](https://developer.apple.com)

* Select Account -- can use AppStore or iCloud ID
Developer Resources

- Go to https://developer.apple.com/documentation/technologies
- Top-level index to developer technologies and APIs
- Swift, UIKit, SwiftUI, and Foundation cover much of what we’ll use
  - But browse through and see if anything looks especially interesting