4995-4
WEB MOBILE APP PROG
Fall 2013

Prof. Dan Rubenstein
danr@cs.columbia.edu

WARNING!!!

- This class will be heavily project-based and heavily unstructured and heavily experimental (in terms of the work you will do and that I've never tried it before)
- You will do your own project
- You must come up with your own project idea (I will try to help)
- You will do your project individually (except under very special circumstances)

Feel free to walk out of here (if anyone is actually here) any time this doesn't sound like your cup of tea

About the Course

- I wanted to learn Web/Mobile/App programming but work seems to keep getting in the way
- Solution: make it part of my job
- I learn as I go, i.e.,
  - I’m not very knowledgable (yet)
  - I don’t have much of a lesson plan (yet)
- Expect very little structure – I may be learning from you more than you will from me…
- This course is very experimental!!!!!!
- I (plan to) use Mac products mostly (iphone, apps) so I am biased toward that for app development, but the web stuff is mostly architecture non-specific (e.g., apache) if you have no Mac access. Also, if you can learn other architectures on your own (e.g., Android), feel free – we may also go in that direction if there is sufficient interest

Prof. Rubenstein’s Philosophy

- Learn by doing (not me sitting and lecturing)
- Effort In translates to results out
- Developing student creativity / ingenuity is critical! That’s what this course aspires to “teach” (encourage probably a better word)

- I will make mistakes and don’t mind being corrected.
- I have trouble being organized

Pre-reqs

- Background knowldege
  - Solid coding experience in C / C++
    - We will probably need to learn objective-C (Mac stuff)
  - Basic understanding of HTML
    - We will do javascript / php etc.
  - Some “hardcore CS”: An idea of some project you want to do that can be an app or website that would do some “real” CS (or confidence you can think of something)
- Class attitude:
  - Ability / Willingness to work on your own / drive yourself / get it all to work
  - Willingness to participate in class, take feedback, and give feedback to others
  - Resourcefulness!!!!
  - Self-Motivated: up to you to work on the project over the course of the term
- Like the idea of an unpolished course where you can potentially change its direction / focus
- If you have expertise in something this class is covering that Prof. Rubenstein does not and you want to help present/prepare/set direction, that’s highly encouraged!

Project

- Must contain:
  - Core CS idea (e.g., something technique/ method learned in 3000 or 4000-level class
  - User interface
  - Does something
  - You will need to document / discuss / present design, etc.
  - Documentation (including design, so I know what you did!!)
  - Prototype must be accessible (web, iphone simulator, etc.)
- Doesn’t need:
  - Marketability
  - To be “pretty”

Sample project

- Prof. Rubenstein's "Optimal train scheduler"
- http://uribe.cs.columbia.edu/train/test2.html
- Why an appropriate project?
  - Runs off a webpage (client/server arch)
  - Uses javascript/php to fill in forms, track times, etc.
  - Back-end has some (fairly) sophisticated shortest-path algorithms, optimized to be computed quickly
  - Had to fetch MTA's train feed (Google protocol buffer)

Errata

- Attendance Mandatory! (Especially when other students are presenting)
- No class on Sep 23 (I'm out of town)
- Class dates:
  - 9/9, 9/16, 9/30
  - 10/7(?), 10/14, 10/21, 10/28
  - 11/11, 11/18, 11/25
  - 12/2, 12/9
- Schedule:
  - Sept: basics of php, javascript, Mac(?), iPhone(?) development (other e.g., android?)
  - Oct: Project designs, figure out specific needs, we focus on those topics
  - Nov: refinements, adjustments, additional topics?
  - Dec: presentations, etc.

Grade

- 70% project, 30% class participation
- Project: 50% design doc, 50% execution
- Extra credit if you know an area covered and help with its preparation

- Enrollment: complete the attached mini-project yourself, then we can talk…

Project to get into course

If you can’t do this yourself, you probably should not be taking the course

PART 1: Write a program that

- Reads a text file, one word per line in the file
- Prints out the words in a sorted order, then…
- Repeatedly takes an input (with whitespace)
- Returns all matches whose word matches each substring in the input (that was separated by whitespace)
- E.g., if words are dog, cat, frog, fox, leopard, gorilla, giraffe
  - The sorted list is: cat, dog, fox, frog, giraffe, gorilla, leopard
  - Input “o” returns dog, frog, fox, leopard, gorilla
  - Input “og” returns dog, frog
  - Input “o g” returns dog, frog, gorilla
- Document your design (feel free to use any pre-existing libraries)
- I will give instructions on how/where to send in your code

Project Part 2

- Suppose the word list is very long (e.g., millions of words), and the code must handle a large number of requests:
  - How would you design the search structures and algorithm to minimize time if the input is the prefix?
  - How would you do the design if the input is a set of 1 and 2-letter substrings?
- Just write a paragraph of a basic description.