Concepts in Database-Backed Web Programming
CSE 491, Fall '09
Syllabus 9/3/09

Lecture - Tuesdays 4:30 - 5:50pm, Egr 1225
Lab - Thursdays 4:30 - 5:50pm, Egr 3353

Instructor: C. Titus Brown, ctb@msu.edu

Primary office: Giltner 318(a).
Office hours and meetings by appointment: Egr 3137.

Course Web site: http://ged.msu.edu/courses/2009-fall-cse-491/

Objectives:

The goal of this course is to teach you just enough to be dangerous in Web development, and to gain some experience in Python programming and sort-of-"real" software development. Here are some of the concepts I hope to introduce:

- HTML and markup, CSS, separation of content and style
- basic server/client concepts and concerns
- server-side data storage (SQL databases, in particular)
- client-side behavior with JavaScript
- asynchronous JavaScript queries to servers (AJAX)
- automated testing: unit tests, Web testing
- version control
- security (XSS and SQL injection, especially)
- scalability concerns and multiprocessing
- debugging
- refactoring

I aim to make this course 95% buzzword compliant -- the only buzzword I'm leaving off is "enterprise"...

Materials: There are no required books. Lecture notes will be provided in outline form, as will certain demonstration code. If you must have a book, I suggest "Philip and Alex's Guide to Web Publishing" by Philip Greenspun. It's available online for free, and you can also buy paper copies quite cheaply. But it's only indirectly relevant to the course. The book "Python Web Programming" (Holden and Beazley, 2002) also covers some relevant material.

Attendance: You are responsible for learning the material presented in class whether or not you attend class. This should not in general be impossible, but it will be easier if you show up for lecture; plus, I believe I'm widely regarded as entertaining. I will provide an outline of the material covered in each lecture, in any case.

Registering: It is MSU's official policy that you must be registered for the course, either for a grade or as an auditor.

Office hours and help: I'll plan to make myself available the night before the homework is due -- so, currently, Monday 9:15pm-11:15pm. If the homework due date changes I'll do my best to change the office hours to match. There is a course mailing list, and I encourage you to ask and answer questions there. Please contact me via e-mail if you want to set up a private meeting. For e-mail, please make sure that 'titus@idyll.org' and 'cse491-fall-2009@lists.idyll.org' make it through your spam filters! (If you use gmail, you may have problems.)

Homework & grading: Weekly programming assignments will make up at least 75% of your grade. The programming assignments will generally be progressive, so you will need to fix whatever bugs or mistakes were present in the previous week's work. I plan to hand out assignments on Tuesdays in lecture and they will be due the following Tuesday at the beginning of lecture. Assignments will be graded on depth of understanding (do you know what you're doing and why?), correctness (do they work?) and English (spelling matters, folks). The remaining 25% of your grade will be determined ... somehow. I haven't figured out how, yet.

You are welcome to work in groups, and you are welcome to use code from anyone and anywhere for your projects - this explicitly includes the Internet and the Python Cookbook. (Please cite your sources for any substantial borrowed pieces of code.) However, you must turn in your assignments individually and you will be individually responsible for the quality of your work. You may also be asked to explain your work to me in private or in public in front of the class. **The primary goal is for you to do the work and to understand what you have done and why.**

All assignments must work in Python 2.6, which is freely available and cross-platform. Your work will be tested under Linux.

Late or incomplete homework will not be graded and will receive a score of 0. However, because programming assignments will build upon previous work, you will have to do it anyway...

One warning: if you hand in homework that isn't functioning at all, you will not get any credit. Always run your code before checking it in; I've had people lose all their points due to a small indentation and commenting change.

Another warning: this will be a programming-heavy course. If you are taking other programming-heavy courses (CSE 335, in particular) you may want to

rethink your life choices.

A third warning: everybody does their !#%!$ homework at the last minute, apparently. That was occasionally a bad idea in this course last year; it will be a much worse idea this year, I think

Commercialization (my notes) and intellectual property (your code): My notes will be licensed under a Creative Commons license, attribution required (http://creativecommons.org/licenses/by/2.5/). This means you can do whatever you want with them as long as you properly attribute their origin to me. Copyright to your code will be retained by you, but in order to take the course you must give me and all other students in the course a license to share and adapt your work for non-commercial purposes (Creative Commons, Attribution-Noncommercial-Share Alike; http://creativecommons.org/licenses/by-nc-sa/3.0/). Please see me privately *before drop day* if you want to discuss this.

Course project: The most likely candidate for the last 25% of your grade is some sort of class project, where you can work in groups or alone on any one of a variety of projects. I have a bunch of ideas, and I want yours, too. Some ideas:

1. Install reddit's source code and add a feature or two.
2. Utilize some form of open data & build a new interface to it (e.g. Toronto's open data system; CATA schedules in MI; MSU student directory or class schedule; google 'ny times "city invites software developers'; etc.)
3. Create a Django or Google AppEngine site.
4. Telescope time assignment fairness (see arXiv paper on "web-PLOP").

One option for grading this is to have everyone write blogs each week about what they're working on and what they've done.

Course Outline (Approximate)

Week 2: How the Web works. Getting set up; CGI, HTML forms, UNIX tools.

Week 3: How clients work. Client Web API; Templating.

Week 4: Networking. Network programming (TCP/IP) in Python; CSS, more templating.

Week 5: Web back-ends. Server API; writing tests (unittest, nose, doctest, twill).

Week 6: Multiprocessing. Processes, threads; JavaScript.

Week 7: Client-server communication channels. Cookies; asynchronous JavaScript (AJAX).

Week 8: Server-side data storage. SQL databases (sqlite, MySQL?); more testing tools and approaches (Selenium; figleaf).

Week 9: Server-side security. Logins and access control; OpenID; proper paranoia.

Week 10: Client-side security. XSS, SQL injection (yes, they're server side), HTTPS. Miscellaneous other topics.

Questions:

1. Should I be available via some chat system during office hours? What are kids using this week? (Use e-mail outside of office hours.)
2. What kind of Web dev project ideas do you have? Anything that you've always wanted to do?
3. Are the office hours good?
4. Vote: how many want paper copies of notes?