WEB APP PROGRAMMING IN PYTHON
SYLLABUS: FALL 2015

Instructor: Hardeep Johar
Contact: hj2203@columbia.edu

The goal of this class is to give you a working knowledge of how to write computer programs, to get
data from the web, and to build your own web applications. We’ll learn the basic elements of
programming, web scraping and crawling and web app development through the language Python.
At the end of this class, you can expect to be fairly proficient in using Python to build applications,
understand enough about programming to be able to quickly pick up other languages (particularly
scripting languages such as JavaScript, Ruby or Perl), and have a good understanding of what it
takes to plan, analyze, design, implement and support software applications. We’ll put all these
components together to learn how crawl the web for data and to build simple web apps.

Your reward, at the end of the course (assuming you do everything seriously!) will be substantial. You’ll
understand how computer applications - large scale ones or small scale ones - work and you’ll be able
to build working prototypes to illustrate and market your own ideas. But, be aware that this is an
intensive course that will require a lot of work and a substantial time commitment.

Scope of the course

Python and Django: Designed (by Guido van Rossum) to be simple, readable, and uncomplicated,
Python is about as intuitive as a programming language can get. Django is a web-framework, software
that is designed to make the development and maintenance of a website as painless as possible.
Django is written in Python for Python and runs on most web servers. With Python and Django you can
prototype your web app ‘off-server’ and then, once you’ve tested it, easily port it to a web server.

Python2 vs. Python3. Unfortunately, there are two versions of Python and they aren’t compatible. We’ll
be using Python3 (current version: 3.4.2) mainly because Python2 is slated to go away (though, in the
open source world, that doesn’t mean a whole lot!) but also because it is a better language.

HTML, CSS, and JavaScript. HTML is a markup language. Pieces of text are ‘tagged’ (bold, headings,
list elements, buttons, forms) and these tags are interpreted by the browser when it renders a web
page. CSS is a style sheet language that integrates with HTML to create a common ‘look and feel’ for a
website (CSS is what makes all Columbia Business School pages have the blue bar on the top).
JavaScript is a programming language much used on websites (because client computers can run
javascript programs). We won’t have the time to cover JavaScript but you will have the tools and
knowledge to pick it up quickly if necessary.
Course structure

Part 1: The fundamentals of programming

What we’ll learn: the basics of programming
What we’ll do: go the web and get data. Simple stuff for starters.
What you may find surprising: If you’ve never done programming before, you’ll be surprised at the rudimentary nature of a computer’s ‘brain’. And, you may be surprised at how hard it is to change your mind set and ‘think’ like a computer.

Part 2: Repetition, Modularity

What we’ll learn: Structured programming concepts
What we’ll do: Crawl the web and see how we can get lots of data. Examine Python tools available for analyzing that data.
What you may find surprising: That the Web is no mystery and digging around inside it is actually fairly simple!

Part 3: Web Apps

What we’ll learn: How to create our own data types
What we’ll do: Build web apps using Python and Django.
What you may find surprising: Developing a prototype web app is remarkably simple (once we know the basics of programming).
Evaluation and learning components

Mini Quizzes: We’ll have a few, very short, quizzes mainly to reinforce points made in class and also to help you get your hands dirty. Most quizzes will be online ‘do whenever you have the time’, though some may be in-class. All quizzes are open book and you’re welcome to check your solutions on your computers. Quizzes will be lightly graded so make a good faith effort and you’ll do fine. One quiz scores will be dropped in computing your quiz grade so no worries if you mess one up.

Home assignments: We’ll have a few home assignments as well. Like the quizzes, assignments are not meant to be diagnostic but rather to help you practice and learn so they will be very lightly graded. You can consult with others, ask me questions, use google or duckduckgo for help, but do try them on your own first. Because it is important that you do the assignments, I will accept late assignments. But, any assignment submitted after the due date will automatically get a 30% reduction in score.

Project: There is no better way to learn something than to go out and use it so start thinking about a web application that you think you’d like to build. The expectation is that you use the material we’ll cover in this class to plan, design, and implement a small web application. Your project grade will depend on how well your work illustrates your understanding of the course material. Final submission will include a design report, Python code, and an in-class “speed-date” presentation and demonstration.

Participation: Demonstrate engagement in the course by asking questions. Preferably in the “Forums” section of the class website on Canvas. I’ll respond to every question, either online or, if the response is of general interest, in the classroom. Some classes have mandatory attendance (I’ll let you know which ones).

Exams: The course has two exams, a midterm exam sometime late in the course, and a final exam during exam week. The midterm exam will be a short “in-class” closed book exam. The final exam will be a - by definition open book! - exam delivered online.

Computers in class

Computers are a requirement for this course and you are expected to bring one for every class. We’ll do a lot of programming - the best way to learn is to see something in action and Python is an especially good language for making things happen. Make sure that your laptops have sufficient charge for the 3 hour plus class!
There is no required text for this class because, unfortunately, most books are designed for people who want to become computer programmers. **However, I strongly suggest that you purchase the Mark Lutz book below so that you have something to read and work with outside the class.** In the class, our focus is on usability and therefore we'll use a mix of online resources, class slides and notes, and sample programs as a substitute for formal texts. The internet is an almost endless resource with excellent tutorials on almost everything and answers to practically every question you might have and I'll point you to these resources as we move along the course.

**Learning Python, 5th Edition Powerful Object-Oriented Programming, Mark Lutz. O'Reilly Media, 2013.** No one can call this 1600 page tome a lightweight, and it isn’t. Easily the best book to keep around when you want to delve deeper into the language.

**Python Programming Fundamentals:** Kent D. Lee. Springer-Verlag, 2011. This is a slim book but it covers most of what we need.


**Introduction to Computing:** Explorations in Language, Logic, and Machines, David Evans. Createspace, 2011. Only if you really, really want to know how computing works. Also available in pdf form at [http://www.computingbook.org](http://www.computingbook.org)

**Online resources**

**Python** documentation: [http://docs.python.org/3.3/index.html](http://docs.python.org/3.3/index.html)


**Django** documentation: [https://www.djangoproject.com](https://www.djangoproject.com)


David Evans’ computing book: [http://www.computingbook.org](http://www.computingbook.org)

Google. Hard to imagine but almost any question you ask have has likely been answered somewhere. Use google liberally (but intelligently)! [https://www.google.com/](https://www.google.com/)