Week 9 - Web Technologies: HTTP, HTML and JSON

Lets review what we have done so far in the class. Up until this week, we have focused on building up some of your practical skills with Python, including using common modules, processing text and unit testing. We have also focused on some more abstract knowledge, like best OOP design practices and design patterns.

We will take a different approach in the second half of this course, focusing on learning how to build a very simple Python-based web backend. Being comfortable with web technologies, and understanding how to build useful services with them is something most programmers need to be skilled in. Our intention is to give you a some understanding of all the layers in what people call the 'web stack'.

The project for the class will be based on this, so its very important to keep up with the weekly material. Make sure to ask questions in the discussion forum if you are confused or are having difficulty with assignments. This week, we will be starting off by learning some of the basic technologies that form the basis of the web, and how we can use them in Python.

HTTP - The Protocol of the Web

The World Wide Web is based completely on HTTP, or the HyperText Transfer Protocol. In computer science terms, a protocol is simply a set of rules computers must abide by in order to communicate and exchange information. In this case, HTTP is the set of rules that defines how a web browser communicates with a web server in order to retrieve information. Having a good understanding of HTTP is critical for any developer.

For this section, read HTTP: The Protocol Every Web Developer Must Know - Part 1 up until section "Using HTTP in Web Frameworks and Libraries". This will give you a good foundation in understanding HTTP.

Check the "Useful Tools" section in the weekly material to see a list of browser extensions and tools you can use to inspect live HTTP traffic.

Data Representations: HTML and CSS

Before, we stated that HTTP is a protocol for how computers can exchange information. But what is this information we are getting when we request some URL using HTTP? We are going to cover two very important data formats used on the Web: HTML and JSON.

First, lets start with HTML, otherwise known as HyperText Markup Language. Generally speaking, a markup language is a formal way of annotating text so that a computer can understand how to *structure* the text. HTML is just that, a way of 'marking up' text so that a browser will know how to structure and display that content in a certain way. This allows web

developers to create text content that is more than just the text, but that is formatted and presented to an end user (via a web browser).

For this course, we do not need to become HTML experts, as this is not a full course in web development. We just need to understand the basics so we can do useful things. With that in mind, please read The HTML Beginners Guide. Also check the Optional Readings section of this week's material if you want to read and understand more about HTML.

In addition to HTML, you will need to know about *CSS*, or *Cascading Style Sheets*. In conjunction with HTML, CSS allows you to control, in a granular way, how a HTML document will be displayed. You might be wondering: if both HTML and CSS end up changing the display of the document, why have both? Historically, HTML was used for both structuring a document and affecting how it end up being displayed. Over time, however, it became apparent that it would be easier to have HTML focus on the structure of the content, and to have another technology focus specifically on how the document ends up being presented. This is referred to as 'separating structure from presentation'. Using HTML for structuring content and leaving the style and presentation to CSS is generally considered good design. However, in the real world, its usually not possible to keep them perfectly separate.

In conjunction with the above, go over the CSS Beginners Guide, which will cover the basic concepts used in CSS, and how its used to change how HTML documents are displayed. One important thing to remember is that CSS can be used to find and select elements within the HTML document itself.

Parsing HTML with BeautifulSoup module

Now that you have a basic understanding of HTML, one useful thing we can do with it is to parse it, allowing us to extract information from an HTML file. In Python, the best way to do this is to use the BeautifulSoup module. Your first step will be to make sure that this is installed via pip. To do so, you can run the

```
pip install beautifulsoup4 lxml
```

command, which will download and install the module in your Python installation. If you have any trouble with this, please post to the discussion board as soon as possible.

To use the BeautifulSoup module, you need to obtain the HTML document somehow, and pass it to the BeautifulSoup object. This snippet of code uses the *urllib2* module, which we learned about previously, to get the HTML for the Google homepage.

```
from bs4 import BeautifulSoup
```

```
url = "http://www.google.com"
page = urllib2.urlopen(url)
soup = BeautifulSoup(page.read())
```

The last line is where we create a BeautifulSoup object from that HTML, which allows us to look for information.If you wanted to see the actual HTML that was downloaded, you can now run:

```
print soup.prettify()
```

We can now use this soup object to find some interesting pieces of the HTML document. There are quite a few ways to identify content in HTML. At this point, you should read An Introduction to BeautifulSoup, which will take you through an example of how to extract useful data from The Biographical Database of the US Congress. Also check the optional readings for more information and references on BeautifulSoup

Data Representations: JSON

Another very important way to represent data on the web is JSON, or JavaScript Object Notation. Now, again, this is not a web development course, so we do not need to know about Javascript. However, we do need to know about this data format since it is heavily used to exchange information in a clear, more compact way than HTML. JSON is actually a very simple data format, and it quite similar to how we handle dictionaries in Python.

For an overview of how JSON works, check out this tutorial. Please read up until the “Using JSON with Different Platforms” section.

Parsing JSON with json module

Just like with HTML and BeautifulSoup, we can use the *json* module to help parse JSON data in our python programs. The *json* module is described here; please read up until the “Working With Your Own Types” section.

What is Flask?

That sums up everything we need to know for this week. Its a lot to go over, so make sure to start early. However, to give you an understanding of where we are going in this class, we’ll introduce Flask, a Python Web Development Framework. Flask is Python software that allows you to create Python code that can respond to HTTP requests. This effectively allows you to write you own web server. Understanding this week’s topics, which form the basis of modern web technology, is crucial to being successful with Flask and the coming weeks.