

## Assignment #5

# AJAX & JavaScript Libraries and Frameworks

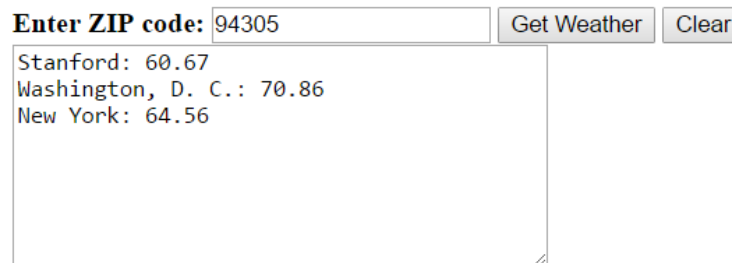
*CS193C Summer 2018, Young*

For our last assignment we will experiment with AJAX and JavaScript Libraries and Frameworks. This assignment is due Thursday August 16<sup>th</sup> at 1:30pm. In order to get all your assignments graded in time to make the end-of-quarter grades deadline, no assignments will be accepted after Thursday the 16<sup>th</sup> at 11:59pm.

## AJAX

For the AJAX section of our assignment we'll retrieve weather information from openweathermap.org. Here is a screenshot showing the AJAX assignment in action:

### Open Weather via AJAX



Enter ZIP code:

Stanford: 60.67  
Washington, D. C.: 70.86  
New York: 64.56

The user enters in a ZIP code and clicks on “Get Weather”. The weather for that ZIP code is retrieved from openweathermap.org and added to a textarea. Weather information listed should include the City corresponding to the ZIP code and the temperature (which is provided to us by openweathermap.org). The textarea should list all requests made. The user can click on the “Clear” button to clear the textarea.

We'll need to teach you a few things in order to get this assignment up and running.

### Creating an Account with OpenWeatherMap.org

Go to:

<http://openweathermap.org/>

Sign Up for an account (upper-right of website). Once you've created an account, go to your account summary and switch to the “API Keys” tab. You'll need to have an API Key to communicate with the server.

### Requesting Weather Reports

To make a request to the server for the weather at a particular zipcode the request format is:

`http://api.openweathermap.org/data/2.5/weather?zip=zipcodeDesired,us&units=imperial&APPID=yourAPIKey`

where *zipcodeDesired* is replaced by the zipcode you want and *yourAPIKey* is replaced by your account's API key.

### Getting Weather Information

You'll pass your URL to OpenWeatherMap.org. After a brief delay, you should get a response providing the weather at the location.

If you use the request above, you'll get your request back as a JSON. This JSON may be found on the XMLHttpRequest object's responseText property. To convert that to an actual object you can do something like this:

```
var result = JSON.parse(requestObj.responseText);
```

If you prefer working with XML you may add the following to the end of your request URL:

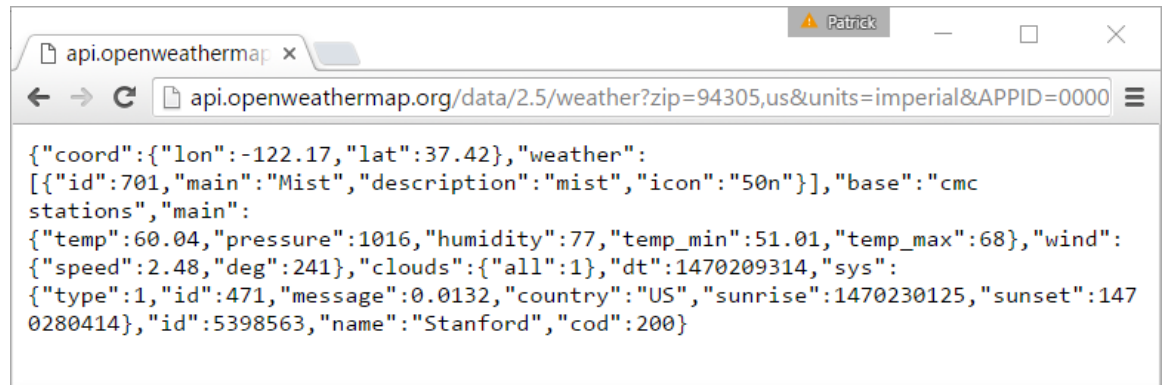
```
&mode=xml
```

In this case the resulting XML may be found at:

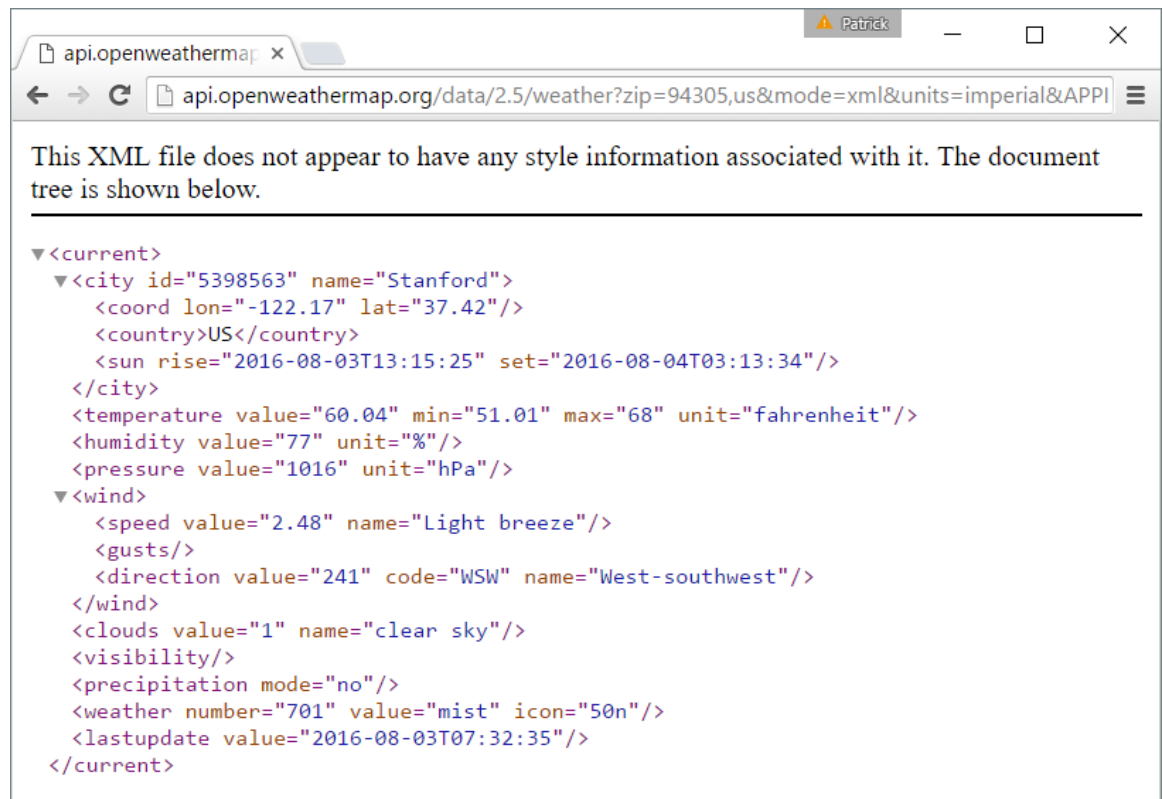
```
var result = requestObj.responseXML;
```

you can then move around the resulting XML tree using the getElementById, getElementsByTagName, and the other standard operations available for moving around the HTML tree.

To take a closer look at what the JSON or the XML look like, instead of using the URL above from JavaScript, instead just enter it directly into the web browser location bar and you should get a result back looking something like this:



Or like this:



This is what's in the responseText or responseXML of your XMLHttpRequestObject. You will probably find JSON easier to use, but you may use either for this assignment.

**Don't forget Ajax uses a callback because often a server will take a while before it responds.** You may have to wait quite a few seconds before your results show up.

## JQuery

We'll keep the JQuery section simple and similar to the in-class examples. Our objective is to just give you just a bit of hands on experience so that you'll remember what you saw in lecture a bit better.

Start out with the jquery-practice.html file provided with this assignment's downloads. This file contains no JavaScript and no JQuery. It does contain HTML along with some CSS Styles. It also contains several buttons which you'll need to wire up to carry out various JQuery tasks.

### Get JQuery Loaded

First things first, you need to get JQuery loaded. I've provided a JQuery file with the assignment downloads. Load it in using a standard <script> tag.

### Turn Headings Red

Wire up the first button so that when the user clicks on it, all headings (h1, h2, and h3) turn red. Note that I've provided a style rule which you may find helpful for carrying out this task.

### Fading Items

Wire up the second button so that when the user clicks on it the heading "Speakers" fades out over a 1 second (1,000 millisecond) period. Fade just the <h3> tag, not the subsequent paragraph on speakers. Once the heading has completely faded out, it will be removed from

the normal text flow and the subsequent paragraph will be bumped up. This is normal and not something you need to correct.

## Node

For our last problem, we're going to get a little bit of experience with Node. In lecture, I showed you how to write and execute a simple Node problem that accessed the file system (see the handout "h06 Basic Node Hands On").

Follow along with the handout and get Node installed on your computer. Experiment with using Node both as an interactive JavaScript environment, and by writing JavaScript files and executing them with Node.

Write a program named "getos.js" that uses the a very short program that retrieves and prints the type of OS you are using by use the "platform" method from the "os" module. See the documentation here.

<https://nodejs.org/dist/latest-v8.x/docs/api/os.html>

What is the value printed?

Turn in the "getos.js" file.