SER421 Spring 2015 Lab4: HTTP and AJAX  
Assigned 10/04/14, due 10/18/14 at 11:59pm via online submission to Blackboard

**Objectives:**

1. Understand the request/response nature of HTTP.
2. Review the role of HTML and new features in HTML5
3. Demonstrate proficiency in constructing a self-contained HTML5/Javascript application

For this lab you will inspect HTTP traffic, inspect HTML content and HTML5 markup features, and develop an HTML5/Javascript application using AJAX. You must complete Task 1 by yourself. You may complete Task 2 by yourself or with ONE partner.

**Task 1: Web Fundamentals (20 points)**

This task does not involve programming (outside of a little HTML5 markup) but instead asks you to inspect web traffic and content.

Part 1: HTTP

Download the “Tamper Data” plugin from <https://addons.mozilla.org/en-US/firefox/addon/966> for Firefox. There is also a version for Chrome but it is tricky to use. Run this tool and visit URLs:

1. http://www.msnbc.com

2. <http://www.bing.com> (type in any search phrase and hit return)

For each URL, use the Tamper Data tool to inspect the HTTP responses and headers, and answer the following:

1. How many subsequent URLs does the page load?
2. Does the site set a cookie?
3. Does the site use persistent connections?
4. Does the site compress its content?

For #1 you can export the responses to XML by right-clicking in the traffic pane and selecting “Copy All” or “Export All” and then count. You can also use the “Page Info” option under the Tools menu. For 2-4 you can inspect particular responses in TamperData’s bottom panes. You do not have to include the export in your submission, but please include screen captures of the request/response headers you use for parts 2-4.

**Task 2: Construct a self-contained Javascript Application (50 points)**

*openweathermap.org* provides a Web API that returns JSON data (see API off the homepage). The format of the data is described here: <http://bugs.openweathermap.org/projects/api/wiki/Weather_Data>

While an example is given here: <http://api.openweathermap.org/data/2.5/weather?q=London,uk>

Write a complete web application that does the following:

1. Displays a list of 3 cities and their associated data. The data should be retrieved and parsed out of the JSON at URLs like the above via an AJAX call. The data you should display in a table:
   1. A city name and 2-letter country code. Examples: London,UK and Phoenix,US
   2. A timestamp when the data was last updated.
   3. Temperature in Celsius
   4. Humidity – a percentage. Example: 70 means “70% humidity”
   5. Wind speed – miles per hour
   6. Cloudiness – a percentage. Example: 10 means “10% cloudy”

When initially loaded, the application should initialize the rows for 2 the 3 cities, London, UK and Phoenix, US. The 3rd city may be any city and is described next.

1. The 3rd city should be populated by selecting from a set of 5 cities in a dropdown. You may populate the dropdown with any 5 cities you like. When a new city is selected, you should populate its data in the 3rd row.
2. Create a new row underneath each city row that shows how the values have changed from the previous call to the server. Provide a Refresh button that forces a fetch of the 3 cities data. That is, it should show how long ago the previous reading was, how much the temp went up or down, and so on for each data value. Constraint: this functionality should work even if you close the browser and reopen it!
3. At the bottom of the page, display the following lines:
   1. “The average temperature is AAA and the hottest city is TTT”
   2. “The average humidity is BBB and the most humid city is HHH”
   3. “The city with the nicest weather is XXX”
   4. “The city with the worst weather is YYY”

Lines c and d should be custom logic you create – my only constraint is that your computation involve all 4 of the values from 1c to 1f above.

Note the data in this Task is in this JSON format, but in some cases you may have to do a conversion (e.g. Kelvin to Celsius temperatures, wind speed in meters per second instead of MPH).

**Submission:**

Submit via a zipfile to Blackboard by the due date. Name the file <asurite>\_lab4.zip. In the zipfile must be a Word document named Task1.docx and a single html file named lab4\_task2.html (again if you have a non-working later version you want me to look at for partial credit, include it and name it lab4\_task2\_partial.html). You may add a README.txt in the zipfile if there is anything you want us to take into consideration. If you did the Task2 with a partner please indicate so in a header comment of your task 2 file. You and your partner’s submission for Task 2 must match. There will be no extensions for this lab as it is due just before our midterm review session!