# *Web Programming V (420-C50-HR)*

# *Lab 02 – AJAX*

Date assigned: Wednesday, September 6, 2017

Date due: **Wednesday, September 6, 2017, 6:00 p.m.**

**Learning Objectives**

Upon successful completion of this lab exercise, the student will have:

* Work with AJAX

Lab Setup

1. Create a folder to use for all parts of the lab called *username\_C50L02*. Add all the files from the C50L02\_Files zip file to this folder.
2. You can NOT use jQuery for this lab, but must use JavaScript.
3. You must NOT include any JavaScript commands in your HTML. Link you JavaScript file at the BOTTOM of the html page (just before or after the </body>)
4. You are to use Chrome or FireFox for this exercise. Some features will NOT work in IE.
5. You can use whatever environment you want from Notepad++ to Dreamweaver to WebStorm to PHPStorm to Visual Studio. At this point you get to decide.

Marking Scheme

|  |  |
| --- | --- |
| **Part A: Movies** |  |
| xhr object created, opened and sent | 3 |
| readystate event handled | 3 |
| Each movie displayed | 6 |
| Flex boxes used | 2 |
| Page design (header, formatting, etc | 4 |
| **Part B: Staff List** |  |
| xhr object to make request | 4 |
| Fills dropdown menu properly | 6 |
| Displays details | 8 |
| Output format | 6 |
| **Part C: Movies Part Deux** |  |
| fetch with promises | 3 |
| after promise, loop through JSON array | 6 |
| Each movie displayed | 3 |
| Page design | 3 |
| **Lab Organization** |  |
| Handed in correctly in correct location | 3 |
| **Total** | 60 |

To do:

**Part A – Movies**

Open the file movies.xml. It is a list of movies titles with the genre, year, country and director or list of directors. Note: although all the movies exist and most of the genres are correct, the rest of the information is randomly generated. Familiarize yourself with the structure.

You need to create a program called movies.html. In this movie you need to do the following.

1. Use XML Http Request object to read the XML file. You must create, open, use and process the results from the file (that is, you cannot use fetch).
2. When the file is read in, display a list of the individual movies nicely formatted. Include the title in a slightly larger font. Make sure you list all of the directors for the film. Try to use flexboxes/content to display each movie (you can try CSS grid layout, but is not completely supported). Here is a possible layout:

Straight Talk

Directed By: Hilliary Matyja,

Erina Whitmell,

Jerri Streat  
Genre: Comedy

Madagascar, 1985

1. For an extra challenge group the movies by genre

**Part B – Staff List**

Use AJAX to construct a web page that allows you to look up the details of a group of staff members. These staff members will be randomly generated from the site randomuser.me.

To see what a record from the site looks like enter <https://randomuser.me/api/?format=xml> in a browser window. As you can see, an XML record is returned with various fields. You will need

You need:

* The file staff.html as a starting point (with the linked file staff.css). You should not have to change staff.html, but may add to the CSS file if you wish.

Your solution must accomplish the following:

* When the page is loaded, make an (asynchronous) XML http request to the page <https://randomuser.me/api/?format=xml&results=10> to retrieve a list of 10 staff members.
* Populate the dropdown menu with the first and last names of the staff members returned. (From <results><name><first> and <results><name><last>)
* When the entries in the dropdown menu are selected, the details of the lecturer should be displayed in div id="staffdisplay." The information must be formatted nicely. Marks will be allocated for design. The minimum information (fields) that must be displayed are:
  + <name><title>
  + <name><first>
  + <name><last>
  + <email>
  + <username>
  + <cell>
  + <dob> (Date of birth)
  + <picture><large>

I suggest the following plan of action:

* Familiarize yourself with the return from the request. Make sure you understand the layout and what fields you need to extract
* Create a function which populates the dropdown menu. The dropdown (select) is defined for you with the id staffsel, but not the options. In order to add options you need to create elements so something like (there are other solutions):

function loadDropDown() {

var staff = xmlDoc.getElementsByTagName("results");

var staffsel = document.getElementById("staffsel");

for(i = 0; i < staff.length; i++) {

var newOpt = document.createElement("option");

newOpt.value = //put unique value here (username works)

newOpt.text = //put first and last name here

staffsel.add(newOpt);

}

}

* You will need to add an event listener that is called when a value from the dropdown is selected; that is, when there is a change to the select element. This event listener will require you to get the information corresponding to the selected element out of the XML document and display it appropriately on the web page.
* There are many ways to do this; a start to a possible function is listed below. This assumes that the value of the selected option is passed to this function as “id”.

var getStaffMemberById = function(id) {

let x = xmlDoc.getElementsByTagName("results");

for(let i=0; i<staff.length; i++) {

let p = x[i].getElementsByTagName(“username”).firstChild.data;

if(p==id) {

return x[i];

} } }

* Remember if you use innerHTML to display the information in the div with the id staffdisplay then the HTML will be rendered correctly, so you can create and format an img tag with the appropriate src attribute to display the photo.

For an extra challenge, do the “correct” thing and sort the list of names by last name so that the list looks professional.

**Part C – Movies – Part Deux**

Do the same as part A, but use the file movies.json and use the fetch command with promises.

**To submit**

When you have completed the lab exercise, show me the result and create a zip file (username\_C50L02) of the folder and submit it to Moodle.