IMY 210
Practical 6: JSON and AJAX

AJAX is Asynchronous JavaScript and XML, which is used on the client side as a group of interrelated web development techniques, to create asynchronous web applications.

JSON (JavaScript Object Notation) is a lightweight data-interchange format like XML, JSON is human readable and writeable and is easy for machines to parse and generate.

This practical aim to provide you with a better understanding of JSON data structure as well as provide a basic framework for creating AJAX calls using both XML and JSON.

Important!

- Work from your **flash drive** to prevent data loss in case of a technical issue.
- This practical covers **Theme 8**. Knowledge regarding earlier work is assumed.
- If your final HTML file does not retrieve/display any data, you will not receive more than 50% for this practical.

Files provided

- update.rss – A demo file to access XML data
- update.json – A demo file to access JSON data
- screenshot.png – A demo of the output

Tasks

1. In this practical you create a simple function that will update a page occasionally from data found in an XML and JSON file.

2. Creating an asynchronous method to get data from an XML file:
    a. Start by creating a JavaScript function getUpdateXML().
    b. Create an XMLHttpRequest object.
    c. Set the onreadystatechange of the XMLHttpRequest object as a function that does the following:
        i. Add a conditional statement that test for the object's readyState and status.
        ii. **If** the readyState is equivalent to 4 (the request has been completed) **AND** the status is equivalent to 200 (HTTP status of OK)…
        iii. … retrieve the document with xml.responseXML and process the document accordingly to be displayed.

    * You can check this resource to understand more about and how to use responseXML.
    * To access the data in the XML file you can look at the HTML DOM functions (getElementsByTagName and getElementById). A good resource is the Mozilla developer API under web technologies.
    * The most ideal way of printing large chunk of data, to construct the format on how it should be displayed before appending it to the HTML with the innerHTML found as part of the HTML DOM.

    d. Outside the onreadystatechange function, create an open call with the XMLHttpRequest object. Pass the variable "GET" the provided "update.xml".
    e. Call the send function with the object.

3. Creating an asynchronous method to get data from a JSON file:

    a. Everything will be the same from *Acquiring data from XML* with a few minor differences:
        i. Instead of using getElementsByTagName, you can access the json object's nodes directly.
        e.g. *jsonObject.items[0].title;*

ii. Remember to change the file being opened by the XMLHttpRequest object.
iii. Before sending the request object, add a responseType to the object and set it as json.
e.g. *xmlhttprequestObject.responseType = 'json';*

4. Lastly, you can simply test this by linking the function to a button in HTML **but**, for this practical we will run this update function automatically.
In any regular instances we can simply call this function after a user action or page update **but¸** for this practical will simply run a time out and link the update function within our timeout.
    a. A timeout can be created in JavaScript by setting an interval.
    e.g. windows.setInterval(do me!, duration in milliseconds);
    b. We can pass a function to the interval as a variable.
    e.g. ….. (function(){ myFunctionName(); }, 1000);

5. **Bonus**: *Make it pretty. (CSS,1994)*

6. Submit your work on clickUP:
    - Compress only your final html file into an archive named **P6.zip**.
    - If you added any CSS please add them internally to the HTML.
    - Make a final backup of all your files.
    - Submit your ZIP file to the **Practical 6** assignment on the clickUP website before the end of **31 May, Monday**.

Good luck