LAB Week 7 & 8: Working with JavaScript

Contents

Introduction

In this lab you will practice working with objects and go on to classes in JavaScript. You should practice running and debugging your JavaScript.

Task 1. Follow on from last week colours exercise

1. This exercise follows on from last week where we displayed a list of colours that we retrieved from a variety of different source files using AJAX. ***Ajax Message pattern***.

Build a web page that has either 5 buttons or a menu with 5 options to allow the user to choose the type of data that they want to load and display using AJAX.

1.a) TEXT - The format of the file is just text eg red green blue etc.

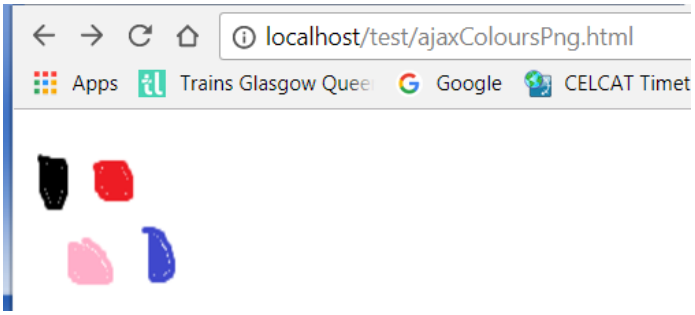
1.b) HTML - a file where the format of the file is an unordered list of colours in html.

1.c) XML - an XMLfile with a list of colours in an appropriate xml format e.g.

```
<colours>
   <colour>black</colour>
   <colour>blue</colour>
   <colour>yellow</colour>
   <colour>green</colour>
</colours>
```

1.d) JSON - a JSON file with a list of colours in an appropriate json format e.g. ["black", "blue", "yellow", "red"]

1.e) IMAGE – a text file that has the name of a file that is a picture with a collection of different colours



NB This just shows that depending on how you structure your files you can load a wide variety of different message types and display their content in the browser using a single application.

Task 2. Objects and methods

Create a new web page to contain the JavaScript you will write.

Add a <script> element to your web page. You can either write your JavaScript in the page, or refer to a separate .js file which you create for your script.

Write JavaScript to accomplish the results described below. You should test your script by viewing the page in your browser. You can use your debugger, e.g. Firebug, to help you.

1. Create an object called *module*, using an object initialiser, which represents a GCU module. The object should have the following properties:

   name -"CSWD",
   level- "4",
   lecturer- "Jim Devon"

2. Write a representation of your object to the web page using:

   ```
   document.write(JSON.stringify(module, null, 4));
   ```

   You can also set a breakpoint in your debugger and inspect the object. You can use these approaches at subsequent stage to check your objects are as they should be.

3. Add a new property *students* to *module*, which consists of an array containing two objects with the following properties:

   name - "Erich Gamma", matriculation number- “1911974”
   name - "Richard Helm", matriculation number- “2741980”

4. Write the *matriculation number* property of the students to the web page.

5. Add a new property *marks* to each student, with the following values:

   [25, 18, 35]
   [12, 10, 25]

6. Add a method *totalMarks* to module which calculates the total number of marks held by both students combined. Call this method and write the result to the web page (should be 135).

7. Add a new marks value, 15 and 18 respectively, to each student. Call the *totalMarks* method again and write the result to the web page (should be 168).

Task 3. Classes

In this task you will define a class, or prototype, to represent a GCU module, and create an instance of this class.

Add a new script to your project, either within a web page or in an external file which is referenced in your page. Write code to accomplish the following:

1. Create a constructor *Module* which takes parameters representing the name, level and lecturer of the module and the names of two students. An example call to this constructor would be:

   ```
   var CSWD = new Module("CSWD", "4", "Jim Devon", "Erich Gamma", "Richard
   Helm");
   ```

   The constructor should initialise an object with the same structure as *module* in Task 1, except that we will not include the *matriculation number* property of the student objects.

2. Add methods to the prototype of *Module* as follows:

   addMarks(student, points), where student is the index of a student, 0 or 1, and marks is the new mark value to be added for this student.

   totalMarks(), which as in Task 1 calculates the total number of marks held by both students combined.

3. Test your class by calling the constructor and the methods as follows:

   ```
   var CSWD = new Module("CSWD", "4", "Jim Devon", "Erich Gamma", "Richard
   Helm");

   CSWD.addMarks(0, 25);
   CSWD.addMarks(0, 18);
   CSWD.addMarks(0, 35);
   CSWD.addMarks(1, 12);
   CSWD.addMarks(1, 10);
   CSWD.addMarks(1, 25);


   document.write(JSON.stringify(CSWD, null, 4));
   document.write("<br />");

   document.write("Points: " + CSWD.totalMarks());
   document.write("<br />");
   ```

   The value returned by *totalMarks* should be as before,135.