SYDE 322 Assignment #1

Web Development Languages: JavaScript

Learning Objectives:

- After completing this assignment, students will be able to (1) understand JavaScript notation, and (2) use JavaScript to program interactive web pages.

Join up with your class mates, and form two-person teams.
Please submit your completed assignment on LEARN.
Please include the name and ID for each member of the team.

Web Development Languages: JavaScript

JavaScript is a popular web development and scripting language that works with HTML5. It is supported by all major browsers, and is used to program interactive web pages. Interactive web sites can handle events, such as user mouse clicks, and provide dynamic, on-the-fly updates of the web site content. In this assignment, we will go through several exercises that are intended to familiarize you with JavaScript notation and JavaScript capabilities.

JavaScript is included in HTML5 pages within the **`<script>...</script>`** element, and the code is typed inside the **`<script>`** element. The code can be included as a direct function call, such as

```
    document.getElementById("test").innerHTML = "Hello World!";
```

or indirectly, as a defined function, such as

```
    function myFunction() {
        document.getElementById("test").innerHTML = "Hello World!";
    }
```

that can be invoked later when a specific event occurs (e.g., a mouse click).

Each HTML5 page has an internal **`Document`** object that represents the file that is currently loaded and displayed. JavaScript can access this object using the keyword **`document`** or **`window.document`**, and use various attributes and methods of the object to manipulate what is displayed on the screen. In the above example, **`getElementById("test")`** is used to obtain a reference to the HTML element whose ID equals to “test”. The returned reference is an object of type **`Element`**, with its own attributes and methods. In this case, we modified the internal property of **`Element`** called **`innerHTML`**, which allows us to set the text contained within the element, and we have set it to the specified value of **`"Hello World!".`**

For more details on **`Document`** and **`Element`** objects, please consult the following URLS:

https://developer.mozilla.org/en-US/docs/Web/API/document
https://developer.mozilla.org/en-US/docs/Web/API/element

Next, use Notepad or a similar text editor, and type in the following text. Save this document as “js-test.html” (ensure that the document ends with the “.html” or “.htm” file extension), and then open it in your favourite web browser.

```
<!DOCTYPE html>
<html>
   <header>
         <meta http-equiv="content-type" content="text/html;charset=utf-8" />
         <script>
         function changeText() { // this gets executed when called
         var element = document.getElementById("test");
         if (element.innerHTML.match("SYDE 322")) {
               element.innerHTML = "Hello World!";
         } else {
               element.innerHTML = "My SYDE 322 test script";
         }
         }

         function changeStyle1() {
         // change text style to these specific properties
         var element = document.getElementById("test");
         element.style.fontSize = "60px";
         element.style.fontFamily = "cursive";
         element.style.color = "red";
         }

         function writeText() { // generate a new page
         document.open();
         document.write('<!DOCTYPE html><html><body>');
         document.write('<h2 id="test2" style="color:blue"> \
         My first fully-generated HTML page!</h2>');
         document.write('<p><input type="button" \
         value="Go Back" onclick="window.history.back()"></p>');
         document.write('</body></html>');
         document.close();
         // something’s missing here
         </script>
   </header>
   <body>
         <h1 id="test" onclick="changeText()">My SYDE 322 test script</h1>
         <script>
         // the following gets executed immediately
         document.getElementById("test").innerHTML = "Hello World!";
         </script>

         <p><input type="button" value="Style1" onclick="changeStyle1()"></p>

         <p><input type="button" \
         value="Generate New Page" onclick="writeText()"></p>  </body>
</html>
```

There is an intentional bug in the given code! To find and correct this bug, press “F12” and select “Console” (e.g., in Firefox, you can also click in the top-left corner of the screen, selected “Web Developer” and then click on “Toggle Tools”). Once you have opened the console, you will see that the bug is a missing closing bracket after the else statement. To fix it, replace the line “`// something’s missing here`” with “`}`” (a single closing bracket). The console will be very useful for resolving various errors that may come up as you write your JavaScript code.

Also note that in some versions of Google Chrome, “`Go Back`” function as currently programmed will not function.

Now, it is time to test and understand your script. If you keep clicking on the text in the page, you will see it alternating between **"My SYDE 322 test script"** and **"Hello World!"**. Let's see how that happens.

First, notice that the text is included inside the **<h1>** element, which also includes an attribute **id="test"**. This attribute allows us to more easily identify this particular element inside the page using scripting. Hence, it is important to add attribute **id**, which is set to a unique value, to HTML elements that we may want to access via JavaScript code later. Also, notice that we have added one more attribute to the **<h1>** element, namely **onclick="changeText()"**. This attribute is an event that is generated by the page itself when the user clicks on this element. Other examples of events include **onmouseover**, which activates as the mouse moves over an element; **onmouseout**, which activates as the mouse moves away from an element; **onkeydown**, which activates as the user presses a keyboard key; **onload**, which activates when the document is finished loading; and **onchange**, which activates when an element is changed. In this case, we have linked the event in question with a call to the function **changeText()**, which is defined in the hedear of the document.

For more details of events, consult the following URL:
http://www.w3schools.com/js/js_events.asp

The script that is inserted inside the **<body>** element of the page contains a direct call to **document.getElementById("test").innerHTML**, which is used to change the text contained within the **<h1>** element. As we click on that same text, on mouse click the function **changeText()**, which is defined in the **<header>** element of the page, is invoked.

The **changeText()** function declares a variable called **element** as **var element,** which is a common way for declaring variables in JavaScript, and then sets it to contain a reference to the **Element** object returned by the **document.getElementById("test")** call. Next, we use **element.innerHTML.match("SYDE 322")**, and specifically the function **match()**, to check if the text of the element that was clicked on contains the string "SYDE 322". If it does contain that string, we change the text of the same element to **"Hello World!"**. If it does not contain the string, we change it to **"My SYDE 322 test script"**.

Pay special attention to quotation marks, where in a word processor, such as Word, opening " and " closing quotation marks are used, but in HTML only unqualified " quotations need to be used.

Next, observe the input button named **"Style1"** that when clicked executes the method **changeStyle1()**. The method changes the style of the previously displayed text to a specific, font, and colour. This method demonstrates how JavaScript can dynamically update the styling of a web page.

Finally, observe the input button named "**Generate New Page"** that when clicked executes the method **writeText()**. The method will replace the current page with a new HTML page, and provide one input button that will allow you to go back to the current page. This method demonstrates how JavaScript can dynamically generate new web pages and new content from existing content.

To learn more about JavaScript functions, and JavaScript in general, please visit the following URLs:
http://www.w3schools.com/js/js_statements.asp
http://www.w3schools.com/js/js_best_practices.asp

To learn JavaScript in detail, go through additional examples on math operations, string manipulation, and control flow, including conditional statements and loops.

JavaScript Exercise1: (submit as js-page1.html)
Use the code from the sample page given above, and alter the **`changeStyle1()`** function to change the page background colour, font style, text colour, and text size to something that you find visually appealing. Create two more buttons that link to two other functions named **`changeStyle2()`** and **`changeStyle3()`** that provide different style settings than **`changeStyle1()`**. That is, as you click on each of these three buttons, the style of your web page changes accordingly.

For more details on how to set CSS properties using JavaScript, see the following URL:
http://www.w3schools.com/jsref/dom_obj_style.asp

[Optional Objective] Define each style as a separate CSS file, and load each file separately.

JavaScript Exercise2: (submit as js-page2.html)
Create a new page with three input buttons named after your favourite YouTube videos or SoundCloud songs. Each button needs to be linked to a function similar to **`writeText()`** function from the code given above, where each new function will generate a new HTML page. Within each generated page, embed your favourite video from YouTube, or your favourite song from SoundCloud, and make the video or the song start playing as soon as the page is loaded. Embed three different videos and/or songs in three different functions. Also provide “Go Back” button in each page.

For more details on how to embed video and sound in HTML5, see the following URLs:
http://www.w3schools.com/html/html5_video.asp
http://www.w3schools.com/html/html5_audio.asp

[Optional Objective] Add a random-video-or-song button that will randomly choose a media file from an extensible list, and play it in a separate window.

JavaScript Exercise3: (submit as js-page3.html)
Create a simple web-based calculator application. The application should have two text input fields, named “**`x`**” and “**`y`**”, one output field named “**`result`**”, and several input buttons for common arithmetic operations, such as addition, subtraction, multiplication, and division. For example, when **`x = 5`** and **`y = 4`** and the button for multiplication is pressed, **`result = 20`**.

For more details on number operations in JavaScript, see the following URLs:
http://www.w3schools.com/js/js_number_methods.asp
http://www.w3schools.com/js/js_operators.asp

[Optional Objective] Spruce up your calculator with modern styling using CSS, and consider using only one field for both inputs and the output, such as in the more realistic calculator applications. Make your calculator represent something that other people may want to use.

To qualify for full marks, complete at least one of the three optional objectives. You may use additional JavaScript libraries in your code, such as JQuery, provided that the library calls do not trivialize the assignment question.

Deliverables:
Three HTML5 documents named as specified above and all the referenced files (include all files as one zipped archive) submitted to the dropbox on LEARN (one submission per team).

Include the name and ID for each member of the team.

Submit your completed assignment before 3:30pm on Tue Jan 26th.