

SOEN287 – Winter 2019 Assignment #3

Due Date: By 11:59 PM Friday **March 22th**, 2019

Evaluation: 4% of final mark (see marking rubric at the end of handout)

Late Submission: none accepted

Purpose: The purpose of this assignment is to help you learn the JavaScript and HTML documents, and dynamic documents with JavaScript.

CEAB/CIPS Attributes: This assignment is primarily evaluating your use of JavaScript and HTML, and dynamic documents with JavaScript. (Use of engineering tools).

Question #1

Create 2 files, named **a3q1.html** and **a3q1.js**, to implement a simple regular expression testing page, like the one shown below. Try to match the layout in the screenshot. There is no need to put any CSS code to achieve this layout. The text area should have 10 rows and 80 columns, and the first 2 input text should have a size of 20 and the last one a size of 5. The title should be an *h1*.

Assignment 3 Question 1

Search in this text:

John;Doe;jd2019@example.com
Jane;Doe;jane@google.com;
Bob;777;bob%777@bob.com

Split the text according to these characters:

Test each word with this regex:

Modifiers for the regex:

There is a total of 8 non-empty words in the text, including 3 words matching the regex.

All the JavaScript code should be put in the js file, except for connecting the search button with a search function located in the js file, which has to be included in the head element of the html file.

Your search function should:

1. start by splitting the text from the text area according to the split characters given in the first input text box. If the first input text box is empty, then don't split the text
2. create a regular expression by providing the *RegExp* constructor function the text of the second and third input boxes (use ***new RegExp()***, not ***/.../...***)
3. test the regular expression on each non-empty word (or element) of the array obtained from step 1, or on the text directly if no splitting characters were given
4. Output the number of non-empty words in the text, plus the number of words matching the regular expression.

Question #2

Create 2 files, named ***a3q2.html*** and ***a3q2.js***, to implement a page validating variable names, and adding them into a list, like in the 3 screenshots included below. Try to match the layout in the screenshots. The only CSS needed is to change the text and background colors of the input text box when the value input is not a valid variable name. What constitutes a valid variable name is given in the screenshots.

Assignment 3 Question 2

Enter a variable name starting with a lowercase letter followed by any number of letters (lower or uppercase), digits or the underscore character. The variable name should not have previously been added.

var

Variables

- abc

Assignment 3 Question 2

Enter a variable name starting with a lowercase letter followed by any number of letters (lower or uppercase), digits or the underscore character. The variable name should not have previously been added.

var

Variables

- abc

Assignment 3 Question 2

Enter a variable name starting with a lowercase letter followed by any number of letters (lower or uppercase), digits or the underscore character. The variable name should not have previously been added.

var

Variables

- abc
- abc_2
- aBc2_
- aBc2_A

You should write at least 2 functions, in your js file, one to validate a variable name (function name: *validateVariable*), and another to add a variable (function name: *addVariable*). The validate function should be called on the *keyup* event of the input text. If the value in the input text is not a valid variable name, then it should change the text color to red and the background color to yellow. The easiest way to do this is to define a CSS class in the html file and set the *className* of the element to this error class. If the value is a valid variable name, then remove the class from the input text element (by setting the *className* to an empty string).

The add button should call the add variable function on the onclick event. It should first check if the variable name is valid, then check if the variable name has already been added to the list. If not, then it should add the variable to the list. If the variable name is already in the list, then it should not do anything. You can keep a global array containing all the variable names added, to make it easier to check, in the add variable function, which variable names have been added so far. After checking if the variable name has already been added or not, you can easily push a new variable name to the global array before or after adding the variable name to the ul.

Question #3

You have a server-side script that cannot handle any ampersand (&) and pipe character (|) in the form data. So we need to convert all the ampersand characters to “and” and pipe characters to “or” at the client side. Write a script to do these conversions in the form field when the field loses **focus** element (**blur**).

A) Your script (**AmpersandPipe.js**) contains one (01) variable and two (02) functions:

1. Global variable: dom.
2. The function **getElementAmpersandPipe()** first reads the field element by using the existing function *getElementById()* to access the element using its id (“field”) specified in your HTML file. Then it uses the *addEventListener()* function to register the event handler. *addEventListener()* takes (03) arguments : (a) the name of event as a string literal (here use “**blur**”), (b) the handler function *convertAmpersandPipe*, and (c) the Boolean value *false*.

3. The function **"convertAmpersandPipe()"** that converts all the ampersands in the form field to " and " and all the pipe characters to "or", when the field loses focus ("**blur**").
 4. The html document called "AmpersandPipe.html", which includes the "AmpersandPipe.js" in the head section.
- B) At the end of the JavaScript file, finish with this line to fire the load event when a resource and its dependent resources have finished loading:

```
window.addEventListener( "load", getElementAmpersandPipe, false );
```

Here is a sample input and output to illustrate the expected behavior of your program:

Sample User Input:

Enter some text that includes at least one ampersand "&" and one pipe sign "|" then click outside the input box

Result:

Sample Output:

Enter some text that includes at least one ampersand "&" and one pipe sign "|" then click outside the input box

Result:

Question #4

Copy and paste the following HTML code into a new document:

```
<!DOCTYPE html>
<html>
<head>
<title>Q4</title>
</head>
<body>
  <div id="container">
    <h2>Create Your Own flower Bouqute</h2>
    <form action="" id="frm">
      <fieldset>
        <table border="0">
          <tr>
            <th>Item</th>
            <th>Unit</th>
            <th>Price</th>
          </tr>
          <tr>
            <td>Red Roses</td>
            <td><input type="text" size="3" id="rose" /></td>
            <td>$5.5</td>
          </tr>
          <tr>
            <td>Yellow Lily</td>
            <td><input type="text" size="3" id="lily" /></td>
            <td>$7.5</td>
          </tr>
          <tr>
            <td>White Mini Calla</td>
            <td><input type="text" size="3" id="scalla" /></td>
            <td>$4.00</td>
          </tr>
          <tr>
            <td>Pink Orchid</td>
            <td><input type="text" size="3" id="orchid" /></td>
            <td>$8.00</td>
          </tr>
          <tr>
            <td>Orange Daisy</td>
            <td><input type="text" size="3" id="daisy" /></td>
            <td>$7.00</td>
          </tr>
        </table>
        <br /><br />
        <label> Choose your delivery method </label> <br/><br/>
        <input type="radio" name="delivery"/> Standard ($2)
        <input type="radio" name="delivery"/> Premium ($6)
      <br/><br/>
      <input type="button" value="Place Order" id="sub">
    </form>
  </div>
</body>
</html>
```

```
onclick="calculateTotal();" />
</fieldset>
</form>
</div>
<script type="text/javascript" src="Q4.js"></script>
</body>
</html>
```

a) Write JavaScript code (Q4.js) that is executed when the place order button is pressed and it calculates the cost of each item (based on quantity specified) entered the online form and the overall total cost. The results should be displayed on the same page under the form submission button.

All JavaScript code must be external. If any fields are left blank or do not contain a number, an alert box should display an appropriate error message upon form submission.

b) The following style rules must be implemented using an external CSS. Ensure that the stylesheet is properly referenced in the HTML code

- All text in the page must be displayed in *MediumBlue*.
- The background color of the form is *DarkOrange*.
- The input form should have a fixed width of 540 pixels.
- The input form should be surrounded by a *blue* border and the distance from the content to the border should be 10 pixels.

The resulted page looks like as the following.

Create Your Own Flower Bouquete

Item	Unit	Price
Red Roses	<input type="text" value="1"/>	\$5.5
Yellow Lily	<input type="text" value="2"/>	\$7.5
White Mini Calla	<input type="text" value="2"/>	\$4.00
Pink Orchid	<input type="text" value="0"/>	\$8.00
Orange Daisy	<input type="text" value="0"/>	\$7.00

Choose your delivery method

☒ Standard (\$2) ☐ Premium (\$6)

Red Roses(units = 1): \$5.5
 Yellow Lily (units = 2): \$15
 White Mini Calla (units = 2): \$8
 Pink Orchid(units = 0): \$0
 Orange Daisy (units = 0): \$0
 Delivery: \$2

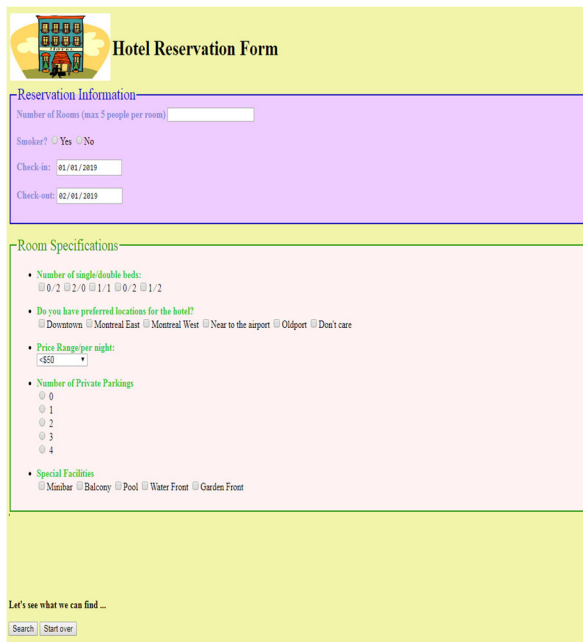
Final Total: \$30.5

Question #5

The running problem to build a website for the Hotel Room Reservation (To be continued in Assignment 4). We resume the Hotel Reservation Web site in Assignment 1.

Please create another section called “Expert suggestion” after the section of “What you are looking for”. Style it properly and make it invisible at first. Once the user’s selections meet the following criteria, use JavaScript to insert the suggested text into this section and make it visible.

- If the Number of price range is \$50 to \$100 and the selected location is Downtown, show the expert suggestion: “It is very difficult to find a hotel room in this price range at Downtown
- Note: make sure the search type is “button”
- Here is the sample output:



Hotel Reservation Form

Reservation Information

Number of Rooms (max 5 people per room)

Smoker? ☐ Yes ☒ No

Check-in:

Check-out:

Room Specifications

- Number of single/double beds:
☐ 0/2 ☐ 2/0 ☐ 1/1 ☐ 0/2 ☐ 1/2
- Do you have preferred locations for the hotel?
☐ Downtown ☐ Montreal East ☐ Montreal West ☐ Near to the airport ☐ Oldport ☐ Don't care
- Price Range/per night:
- Number of Private Parkings
☐ 0
☐ 1
☐ 2
☐ 3
☐ 4
- Special Facilities
☐ Mainbar ☐ Balcony ☐ Pool ☐ Water Front ☐ Garden Front

Let's see what we can find ...

a) Initial form



Hotel Reservation Form

Reservation Information

Number of Rooms (max 5 people per room)

Smoker? ☐ Yes ☒ No

Check-in:

Check-out:

Room Specifications

- Number of single/double beds:
☐ 0/2 ☐ 2/0 ☒ 1/1 ☐ 0/2 ☐ 1/2
- Do you have preferred locations for the hotel?
☒ Downtown ☐ Montreal East ☐ Montreal West ☐ Near to the airport ☐ Oldport ☐ Don't care
- Price Range/per night:
- Number of Private Parkings
☐ 0
☒ 1
☐ 2
☐ 3
☐ 4
- Special Facilities
☒ Mainbar ☐ Balcony ☐ Pool ☐ Water Front ☐ Garden Front

Expert Suggestion

- It is very difficult to find a hotel room in this price range at Downtown

Let's see what we can find ...

b) after filling the form and hitting the search button

Submitting Assignment #3

- Zip the source code (HTML and JavaScript files only please) of this assignment.
- Naming convention for zip file: Create one zip file, containing all source code files for your assignment using the following naming convention:
 - The assignment is done individually:
The zip file should be called *a#_studentID*, where # is the number of the assignment *studentID* is your student ID number. For example, for the first assignment, student 123456 would submit a zip file named *a3_123456.zip*
- For submission instructions please refer to the course web page.
- Assignments not submitted to the correct location or not in the requested format will not be graded.
- Submit only ONE version of an assignment. If more than one version is submitted the *last one will be graded* and all others will be disregarded.

Evaluation Criteria of Assignment #3 (20 points)

HTML Document with JavaScript source code	
Question #1 (4 pts.)	
HTML document, correct elements, layout	1 pt.
Splitting the text correctly	0.5 pt.
Creating the regex and testing the words	2 pts.
Display the results	0.5 pt.
Question #2 (4 pts.)	
HTML document	0.5 pt.
Function validateVariable()	1.5 pts.
Function addVariable()	2 pts.
Question #3 (4.5 pts.)	
HTML form one (01) id	0.5 pt.
Global variables& register the event handler	1 pt.
Function getElementAmpersandpipe()	1 pt.
Function convertAmpersandpipe()	1.5 pts.
Display the result	0.5 pt.
Question #4 (4.5 pts.)	
CSS file	1
Calculate the sum	1.5pts.
Display the result	1.5 pts.
Validation	0.5 pt.
Question #5 (3 pts.)	
checked values test and changing the visibility	1.5 pts.
Display the result& event handler	1.5 pts.