JavaScript Assignments”

**Assignment #1** - Using Variables in a Form

Note: I’ll include the HTML files with the email. I tried to Compress and even Zip, but having issues.

Expectations

The JavaScript that you use in the course assessments should:

* Render and function properly in at least two of the following browsers: Chrome, Firefox, Edge, or Safari.
* Be verified to be error-free, well-documented with comments, and appropriately constructed.

Overview

In this assessment, use the Web page called “invitation.html” found in the Required Resources (in the zip file called IT-FP3215.zip) to add functionality to an interactive form that generates an invitation to volunteers for an event. The file will have the following invitation message placeholder and a form below it. You will add JavaScript to the form that will allow a user to dynamically fill out the invitation.

Hello \_\_*recipientName*\_\_\_\_\_!

You have been invited to volunteer for an event held by \_\_*organizationName*\_\_\_\_\_ on \_\_\_*eventDate*\_\_\_\_\_. Please come to the following website: <insert your website URL> to sign up as a volunteer.

Thanks!

\_\_*hostName*\_\_

Hints:

* The placeholders (for example the “recipientName”) will need to be set up as <span> elements with an assigned id attribute. Prompt the user to enter in the recipient’s name, organization name, event date, and host name. Using JavaScript, replace those elements dynamically with what the user has entered in the form.
* Use the form’s input fields (once submitted) to store the values to JavaScript variables. Then manipulate the DOM to replace the <span> element content dynamically.

Tip: Variable names cannot include any special characters or spaces. They cannot start with a number. They also cannot be any of JavaScript’s reserved words.

Preparation

Download and unzip the IT-FP3215.zip file found in the Required Resources. It contains the initial framework for the site. All of the HTML files are located in the root directory. Images are placed in the images subdirectory; CSS files are placed under the css subdirectory. Your JavaScript external files should be placed under the “js” subdirectory (you will need to create it). When you submit your work, be sure to zip up the entire folder, including all of the ancillary files such as the images, CSS, and JavaScript code.

Note: This course requires you to use a text editor to complete your work. There are many free open source options on the Internet from which you may choose. See the Suggested Resources for links to free, open source text editors.

Directions

Read the Overview. Use the invitation.html file in the Resources as a template for completing this assessment.

Write JavaScript that enables the invitation to be dynamically completed using the form. Make sure to do each of the following:

* Declare variables to store the input field data.
* Store the input field data into the variables on form submit.
* Manipulate the DOM to replace placeholder data with the variables.
* Write developer comments to describe the variables being declared and explain the functions and logical blocks of JavaScript code.
* Once completed, view your pages in each of your two selected Web browsers to see if the content renders appropriately and consistently within each. Next, verify that your code is error-free using the appropriate browser-specific development tool found in the Resources.

Take a screen capture of each of your validation results and save it for submission.

Submission Requirements

* Upload all your website files to a hosting service of your choice. Two free sites to consider are Freehosting.com or 000webhost.com. See the Suggested Resources for links.
* Submit your work in the courseroom using a single Zip file containing the following:
  + Your entire Web site (including the updated “invitation.html” file).
  + A Word document with:
    - The url to your Web site so faculty can view your site on a live host.
    - A screen capture of each of your two validations that you completed using the developer tools found in the Resources.

**Assignment #2** - Using Arrays and Loops

Overview

In the previous assessment, you used a static set of named variables to store the data that was entered in a form to be output in a message. For this assessment, you will use the invitation.html file that you modified in the previous assessment to create a more effective process for entering information and creating invitations for volunteers. Rather than having to enter each volunteer and create an invitation one at a time, you will create a script that will prompt the user to enter in the number of volunteers, event date, organization name, and host name. Using the number of volunteers entered (5–10), the script should loop through to ask the user to enter the recipient name and store it in an array. Once the user has entered all of the names and pressed submit, the page should display each of the invitations one after another at the bottom of the page.

Directions

Use the invitation.html file that you submitted in your previous assessment to add functionality to your form. This new functionality should allow the user to enter in the number of volunteers, whereupon the form will display the corresponding number of input fields for the user to enter information for each volunteer. Once the form is submitted, the data should be stored in an array to be looped through to create separate invitations for each volunteer on a new page.

Make sure to do the following:

* Add an input field that allows the user to input the number of volunteers as a numeric value.
* Once the number of volunteers has been entered (by pressing the enter key while the cursor is in the input field), use JavaScript to display the volunteer input fields based upon the number of volunteers entered.
* Write JavaScript that stores the form into an array of records once the form is submitted.
* Write JavaScript to Loop through the array of records and then display the invitation message for each volunteer. (Unlike a simple array that contains a single variable for each index, an array of records allows us to store related data fields for each index in the array. If we were going to store this in simple arrays, we would need a separate array for each data field.)
* Once completed, view your pages in each of your two selected Web browsers to see if the content renders appropriately and consistently within each. Next, verify that your code is error-free using the appropriate browser-specific development tool found in the Resources. Take a screen capture of each of your validation results and save it for submission.

Submission Requirements

* Upload your Web site files to your Web host.
* Submit your work in the courseroom using a single Zip file containing the following:
  + Your entire Web site, including all associated files.
  + A Word document with:
    - The url to your Web site so faculty can view your site on a live host.
    - A screen capture of each of your two validations that you completed using the developer tools found in the Resources.

**Assignment #3** - Dynamic Images, Events, and the DOM

Overview

Most Web sites have images. There are many different features and functionality that can be used to work with images via the use of JavaScript. These include preloading, rollovers, and cycling banner ads. In this assessment, you will be working with JavaScript, images, and events, and you will be manipulating the DOM to create an interactive image gallery.

*Hint: Preloading your images will only work on a hosting server and not your local drive as there’s no load time for your images locally. Once you have preloaded those images, you may want to clear your cache to test your loading of the images again.*

Tip: In a smaller JavaScript program such as this one, each function is created for a specific purpose. However, in more complex sites, it is better to build functions that we can apply to multiple situations. For example, rather than specifying an element name or id, we can use a variable that is passed into the function.

Directions

Use the gallery.html and index.html files that you downloaded in the Zip file in Assessment 1. Create functionality using JavaScript on the following pages:

index.html

* Preload the images (banner1.jpg, banner2.jpg and banner3.jpg found in the images folder) for the banner at the top of the page.
* Create a cycling two-state banner that cycles every three seconds.

gallery.html

* Preload the gallery images.
* Create rollover functionality for each of the thumbnails in your image gallery. Use appropriate images found in the images folder.
* Write developer comments to describe the variables being declared and explain the functions and logical blocks of JavaScript code pertaining to the gallery.

Make sure to do the following:

* Create an onpageload function to preload all of your images.
* Create a modularized function to cycle the homepage banner.
* Create a modularized rollover function for gallery images.
* Once completed, view your pages in each of your two selected Web browsers to see if the content renders appropriately and consistently within each. Next, verify that your code is error-free using the appropriate browser-specific development tool found in the Resources.

Take a screen capture of each of your validation results and save it for submission.

Note: "Modularized" refers to creating components that can be re-purposed without significant changes to coding. Modularized components have no "hard coding" of image names, urls, et cetera, in the functions. There is an example JavaScript modularized and non-modularized JavaScript found in the Suggested Resources in the file called "Example Code."

Submission Requirements

* Upload your Web site files to your Web host.
* Submit your work in the courseroom using a single Zip file containing the following:
  + Your entire Web site including all associated files.
  + A Word document with:
    - The url to your Web site so faculty can view your site on a live host.
    - A screen capture of each of your two validations that you completed using the developer tools found in the Resources.

**Assignment #4** - Form Field Validation and Error Messages

Overview

You have created a basic form and added interactivity to images using JavaScript. Now it is time to validate information entered into your form fields. You will use the “registration.html” file found in your Zip file. You will see that the page has a form that has the following fields for allowing users to register for an account. The items in parentheses are defined formatting instructions to be observed for each field.

* Username\* (must only contain letters and numbers).
* Password\* (minimum of 8 characters).
* PasswordVerify\* (minimum of 8 characters, must match password).
* FirstName\* (text string).
* LastName\*( text string).
* Email (uses xxx@xxx.xxx format).
* PhoneNumber (uses (xxx) xxx-xxxx format).
* SignUpNewsletter (radio box for yes/no).
* Tips:
* JavaScript is case sensitive so a variable named myVar is different than MyVar and myvar.
* Perform your form validation testing in pieces at a time rather than all at the end. It will be much easier to fix the error if you have a smaller sequence of code to review.

Directions

Read the assessment overview.

Use the registration.html file to add functionality to the form found on the page. The JavaScript that you write should validate each field and display appropriate error messages (if errors are made) that direct the user to complete the form properly. Once the form is completed, your script should open the confirmation page (confirmation.php).

Note: The input fields in the form with an \* are required fields.

Make sure to do the following:

* Write JavaScript that defines that a field is required and generates an appropriate error message if the field has not been completed.
* Write JavaScript to validate all input fields per the formatting definitions that the field values should be checked against (found in the overview) after each field.
* Write JavaScript that displays an appropriate error correction message (next to the field) in the event a form entry error has been made.
* Write a JavaScript that will default the user's cursor to the first erroneous input field in the event that there is an input error.
* Create a submit button that executes the validation when submitted.
* Once completed, view your pages in each of your two selected Web browsers to see if the content renders appropriately and consistently within each. Next, verify that your code is error-free using the appropriate browser-specific development tool found in the Resources.

Take a screen capture of each of your validation results and save it for submission.

Submission Requirements

* Upload your Web site files to your Web host.
* Submit your work in the courseroom using a single Zip file containing the following:
  + Your entire Web site and all associated files.
  + A Word document with:
    - The url to your Web site so faculty can view your site on a live host.
    - A screen capture of each of your two validations that you completed using the developer tools found in the Resources.

**Assignment #5** - Query Strings and Storing Persistent Information

Overview

As Web forms get longer, we very commonly see them split across multiple pages to collect all of the necessary information. There are different reasons for this. A long form can be daunting for users and a large/long form can be difficult for users to fill out on a mobile device. These forms need to be designed so that the data entered by the user on the forms on each page will be submitted to the Web server simultaneously. This makes more sense as they are part of the same data set. The problem with stateless pages is that if the user moves from one page to the next, the data entered is lost. To bypass this issue, you will need to use query strings, hidden input fields, and cookies.

In this assessment you will use the previously created registration.html file to send information to a second page (interests.html) that has another form for the user to complete. You will write a script on that form that will save the information from both forms to a cookie and then display it on a third page.

Tips:

* It will help to output the array into the browser console so that you can verify that the string is being correctly parsed. Details on the browser console can be found in the Resources.
* To skip having to enter data into the form each time to test, it may help to create a JavaScript function that automatically fills in the fields for you and comment it out when completed.

Directions

Read the Overview.

Modify the "registration.html" page created in the prior assessment to send a query (that has all input field information from that form) to a second page (interests.html (created by you)). The information should be stored in hidden input fields (in the interests.html page) using the same field id/name. The interests page should ask the user to enter the following in optional fields:

* Interests (list at least three using a checkbox).
* Newsletter sign up (radio box with a yes/no option).
* Comments (free form text area).
* Referred by (text field).
* When the user presses submit, all of the input fields from this form as well as the registration.html form will be saved into a cookie. The user should then be forwarded to a third page (confirm.html (created by you)) that will read the cookie information and display it in a name/value pair using JavaScript.

Make sure to do the following:

* Create and integrate a script on the registration.html page that stores the input field data into hidden fields to be used in the interests.html page once the submit button is pressed.
* Create an interests.html page with a form that has the fields listed above. This interests.html page will read in the input from the query string data from the registration.html page and store them into hidden input fields.
* Write a script that runs in response to the submit event, from the interests.html page, that saves the input from both pages to a series of cookies to store each input, and opens a third page called confirm.html that reads and displays information from all the fields.
* Once completed, view your pages in each of your two selected Web browsers to see if the content renders appropriately and consistently within each. Next, verify that your code is error-free using the appropriate browser-specific development tool found in the Resources.

Take a screen capture of each of your validation results and save it for submission.

Submission Requirements

* Upload your Web site files to your Web host.
* Submit your work in the course room using a single Zip file containing the following:
  + Your entire Web site and all associated files.
  + A Word document with:
    - The url to your Web site so faculty can view your site on a live host.
    - A screen capture of each of your two validations that you completed using the developer tools found in the Resources.