**blackCSIS210 - Data Structures**

Web App. Dev.

### Laboratory 6

**Lab 1**

# Names \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#

# General Lab Procedures

* You should create a directory (folder) in your home account called csis390. At the beginning of each lab, create a new sub-directory called labX, where X is the lab number.
* Files used in the lab can be found on the course canvas webpages.
* Turn in this lab sheet stapled to print outs of the code you produce as needed in each assigned section from the laboratory manual. These sheets should be in order. One lab submission is sufficient for each group.
* You can find documentation for the HTML 5 specification, the CSS, the Javascript, and the php references respectively at

<https://html.spec.whatwg.org/>

<https://developer.mozilla.org/en-US/docs/Web/CSS/Reference>

<https://developer.mozilla.org/en-US/docs/Web/JavaScript>

https://www.php.net/manual/en/langref.php

**Lab Objectives**

* Practice using php basics

## Lab 6

#### Part 1 – Welcome Script

In this part of the lab, you will practice working with a basic coloring php script.

* Download the file **welcome.txt** from Canvas.
* Click on one of the following PHP scripts (depending on your assigned ftp site), experiment with it, and notice it has a flaw.

<http://www.sienasellbacks.com/sandbox/lab6/welcome.php>

<http://www.breimer.net/sandbox/lab6/welcome.php>

* The script asks for your name and displays a welcome message. Then, the script asks for your favorite color and it is supposed to change the welcome message to your chosen color, but your name disappears. This is because the server can only remember your very last submission. In order to fix this problem, we must use sessions.
* In Notepad++, create a new file called **welcome.php** and save it in your lab6 folder
* Copy and paste the PHP code of the welcome.txt file into your file and save welcome.php to your lab6 folder.
* Using the connection setting that was assigned to you, upload your entire lab6 folder to the remote server and then disconnect from the server.
* Open the script in Chrome and make sure the script is working on the server.
* Once completed, demonstrate your webpage for your instructor and have him initial here. If you do not finish during the lab period, then demonstrate your webpage at the beginning of the next lab period.

Instructor’s Initials \_\_\_\_\_\_\_\_

* Now, it is time for you to fix the script. In Notepad++, edit the script to fix the flaw by using SESSION variables.
* Start a session by calling the session\_start() function.
	+ Important: session\_start() is typically the first line of code inside the first set of PHP tags.
* Then, we can copy the submitted form values to PHP SESSION variables as follows:

$\_SESSION['username'] = $\_GET['username'];

* + Important: You only want to copy the username if the user just submitted it. If $\_GET['username'] is null, you do not want to set the session variable.
* Instead of using the GET variables to print the welcome message, we can use the SESSION variables as follows:

Welcome '.$\_SESSION['username'].'

* Save welcome.php.
* Upload your lab6 folder to the server. Disconnect from the server.
* Reload your script in Chrome.
* Once completed, demonstrate your webpage for your instructor and have him initial here. If you do not finish during the lab period, then demonstrate your webpage at the beginning of the next lab period.

Instructor’s Initials \_\_\_\_\_\_\_\_

* Open welcome.php in Notepad++.
* First, extend the script so that you can also add the font-size of the welcome message as follows:

<http://www.sienasellbacks.com/sandbox/lab6/welcome_solution.php>

<http://www.breimer.net/sandbox/lab6/welcome_solution.php>

* + When the user clicks "Add Color" change the submit button value to "Add Size" and the input text field's name to "size".
	+ Add another else if clause to handle when the user clicks "Add Size"
	+ Add more inline CSS to the welcome message:

echo '<h2 style="font-size: '.$\_SESSION['size'].'px;

* Second, improve the application so that it logs the users input in a text file calledwelcomedata.txt.
	+ A sample file is on Canvas.
	+ After printing/echoing the welcome message with the username, color and font-size, we can construct a comma-separate data string as follows:

$datastring = $\_SESSION['username'].",".$\_SESSION['color'].",".$\_SESSION['size']."\n";

* Then we can append this string to a file on the server as follows:

$filename = "welcomedata.txt";

$myfile = fopen($filename, "a");

fwrite($myfile, $datastring);

fclose($myfile);

* Save welcome.php.
* Upload your lab6 folder to the server. Disconnect from the server.
* Reload your script in Chrome.
* Once completed, demonstrate your webpage for your instructor and have him initial here. If you do not finish during the lab period, then demonstrate your webpage at the beginning of the next lab period.

Instructor’s Initials \_\_\_\_\_\_\_\_

* The welcome.php script shows a typical PHP application that generates or modifies a series of forms. It also illustrates how PHP can dynamically change a web page's content and style based on submitted user information.
	+ PHP Pros: PHP can receive form data easily using the $\_GET variable and your application code remains hidden making it difficult for others to "steal" your core algorithms. PHP can store information temporarily via SESSIONS and permanently via files and databases.
	+ PHP Cons: Updating a page requires a full HTTP request/response cycle where an entire page has to be transmitted, loaded and rendered. In fact, the extended welcome.php application requires 4 HTTP request/response cycles (one to load the initial form and one for each of the three entered values).
* Generating web pages with PHP can lead to an unstructured mix of hard-coded HTML/CSS and dynamically-generated HTML/CSS.
* Rather than generate and control the form with PHP, we can implement the form and all the behavior using the following JavaScript application:

<http://www.sienasellbacks.com/sandbox/lab6/welcome.html>

http://www.breimer.net/sandbox/lab6/welcome.html

* This JavaScript mini-application will send the submitted data using AJAX to the server-side script: add\_welcome\_data.php. The code for this is on Canvas in the file **add\_welcome\_data.txt**.
* This script writes to the same file as the PHP script.
	+ Examine the source code of welcome.html. Notice how it uses a similar if-else-if statement to control the form elements to gather the username, color and font-size. Sadly, if you refresh the page, everything is lost. However, we only need one full HTTP request/response to load the page and one AJAX request to transmit the data.
	+ Notice how we transmit the data by "stringifing" a JavaScript object into a JSON-encoded string.
	+ Look at the add\_welcome\_data code that appends to the file. Notice how we can parse the JSON-encoded string to create a PHP object.
	+ Note that in Part 2, you can use this as a model for completing the last part.

#### Part 2 – Mad Libs

In this part of the lab, you will develop a Mad Libs application using php.

* Connect to the following URL and examine the php script. The code is available on canvas in madlibs.txt.

<http://www.sienasellbacks.com/sandbox/lab6/madlibs.php>

http://www.breimer.net/sandbox/lab6/madlibs.php

* The script takes the user input and inserts it into one of two silly sentences. This script is annoying because each time you press a button the page reloads, the input disappears and you have to retype it.
* In Notepad++, create a new file called madlibs.php and save it in your lab6 folder.
* Copy and paste the PHP code of the [mad libs script](http://breimer.sienacs.com/courses/csis-390-s17/labs/lab6/madlibs.txt) above into your file.
* Save your copy of madlibs.php to your lab6 folder.
* Using the connection setting that were emailed to you, upload your entire lab6 folder to the remote server and then **disconnect from the server**.
* Open the script in Chrome and verify that the script is working on the server.
* We will now try to fix the php script. In Notepad++, edit the script by using the POST method and by using SESSION variables to remember the posted data.
	+ Consult <https://www.w3schools.com/php/php_sessions.asp> for more information on session variables.
* Change the form method from "get" to "post"
* Throughout the PHP code, use the $\_POST variable instead of the $\_GET variable.
* Add a block of PHP code at the top of the file to start a session and save the posted form data to $\_SESSION variables:

<?php

session\_start();

$\_SESSION['name'] = $\_POST['name'];

$\_SESSION['verb'] = $\_POST['verb'];

?>

* When generating the form, use the session variable to give each input element the previously stored value:

<input type="text" name="name" value="<?php echo $\_SESSION['name'] ?>">

* + Note that this illustrate one of the most complex syntax issues you will ever see as a Computer Scientist. We are embedding a PHP echo statement inside of the quotes of an HTML attribute value. PHP cannot access the document objects because the document is not rendered on the server. The only way to put values into attributes is to literally print/echo the value inside of hard-coded HTML.
* Instead of setting the local PHP variables based on the $\_GET or $\_POST variable, which might have been forgotten, use the values stored in the session variables: $name = $\_SESSION['name'];
* Upload your lab6 folder to the server. Disconnect from the server.
* Reload your php script in Chrome.
* Once completed, demonstrate your webpage for your instructor and have him initial here. If you do not finish during the lab period, then demonstrate your webpage at the beginning of the next lab period.

Instructor’s Initials \_\_\_\_\_\_\_\_

* Now you will extend the php script so that you can enter six word and can select three different sentences. Your code will eventually work like the script found at

<http://www.sienasellbacks.com/sandbox/lab6/madlibs_solution.php>

* This script also logs the users input in a text file called [sentencedata.txt](http://breimer.sienacs.com/courses/csis-390-s17/labs/lab6/sentencedata.txt%22%20%5Ct%20%22_blank)
* Add other additional form elements and session variables for six different words.
* Add an additional sentence button and else if to generate a third sentence from the six words.
* In addition to printing/echoing the sentence, store the sentence as a string called $sentence and append it to a file as follows:

$filename = "sentencedata.txt";

$myfile = fopen($filename, "a");

fwrite($myfile, $sentence);

fclose($myfile);

* Save madlibs.php. Upload your lab6 folder to the server. Disconnect from the server.
* Reload your php script in Chrome.
* Once completed, demonstrate your webpage for your instructor and have him initial here. If you do not finish during the lab period, then demonstrate your webpage at the beginning of the next lab period.

Instructor’s Initials \_\_\_\_\_\_\_\_

* Finally, rather than generate and control the form with PHP, we can implement the form and all the behavior using JavaScript.
* Using the JavaScript Welcome application as a model, implement the Mad Libs application using JavaScript and AJAX.
	+ Your file should be called madlibs.html. Use the welcome.html application as a model.
	+ This JavaScript mini-application will send the submitted sentences using AJAX to a server-side script called [add\_sentence\_data.php](http://breimer.sienacs.com/courses/csis-390-s17/labs/lab6/add_sentence_data.txt%22%20%5Ct%20%22_blank). Sample code is available on Canvas; download this to your lab6 folder.
	+ Unlike the Welcome application which sent an object with three fields, this application sends a single string. Thus, you do not have to use JSON to stringify the data object.
* **Create a zip file of your lab6 folder called lab6.zip and submit the file in Canvas. The zip file should contain the following files: madlibs.php, madlibs.html, add\_sentence\_data.php, welcome.php and welcome.html. In the comment area of Canvas put your partner's name.**