

## EECS 448 Software Engineering Lab

### Lab #6: PHP Programming (*continuation of Lab #5*)

#### Due Date and Deliverables

The deliverables for this lab will be due on **Sunday, November 13, 2022 at 11:59 PM**. These include:

- GitHub repository link containing source code for the required files (see below)
- people.eecs.ku.edu link

#### Before Starting

Since PHP runs on the server side, all code must be run from the EECS website to verify functionality.

In the root of your public\_html folder, create a helloWorld.php file and add the following:

```
<?php
    echo "<p>Hello, World!</p>";
?>
```

Then try to access

- [http://people.eecs.ku.edu/~YOUR\\_USERNAME/helloWorld.php](http://people.eecs.ku.edu/~YOUR_USERNAME/helloWorld.php)

If the text "Hello, World!" appears, your public\_html directory is working correctly.

#### Embedding PHP code in a .html file

You can write chunks of PHP code, surround by `<?php ?>`, inside an html file, but you have to configure your public\_html folder to do so. Sometimes you may want to write PHP embedded in HTML and other times you might want to just echo out HTML through PHP.

Create a file if it is not already there, called .htaccess inside of your public\_html folder. Edit that file and add the following line:

```
AddHandler application/x-httpd-php.html
```

Save and exit the file.

#### Basic Syntax

PHP is syntactically similar to C++ and Java. Some very important differences:

- All variables begin with a \$ (e.g. \$x)
- Variables don't have to be declared before using them

- Variables are dynamically typed, meaning a variable can be a number on one line, then a string on the next

```
$x = 10;  
$x = "letters";
```

PHP still has "if" statements, loops, methods, and classes. You can check the references for examples and further syntactical details.

## Print HTML

PHP stands for "Hypertext Preprocessor". Hypertext is in reference to the 'H' in HTML. In other words, the goal of PHP is to be a programmatic way to generate HTML. How does this work? Let's see.

First, here is a basic printing statement:

```
echo "Hello, world!";
```

But you can echo anything, even HTML.

In your HTML file, load the PHP file using a line similar to the following:

```
<?php include 'myfirstprogram.php';?>
```

Here's a sample program that can print a series of sums.

```
<?php  
//Inside myfirstprogram.php  
function sum($x, $y) {  
    $z = $x + $y;  
    return $z;  
}  
  
echo "5 + 10 = <b>" . sum(5, 10) . "</b><br>";  
echo "7 + 13 = <b>" . sum(7, 13) . "</b><br>";  
echo "2 + 4 = <b>" . sum(2, 4) . "</b>";  
  
?>
```

## Error Reporting

Should you need to, you can add the following code at the top of your PHP script (inside the <?php ?>) to output possible errors.

```
error_reporting(E_ALL);  
ini_set("display_errors", 1);
```

If your PHP script displays no output or you receive an HTTP 500 error, look for possible syntax errors. PHP will not display anything if there is a syntax error. If your PHP code is displayed instead of being run, make sure the code is in your public\_html folder and you are accessing the page from your people.eecs.ku.edu address.

## Exercise 1: Multiplication Table

Create a PHP program that displays a 100x100 multiplication table. The first row and first column should be the 1...100 and the inner parts of the table should be the multiples. Below is an ASCII approximation:

```
 1 2 3 4 5 ... 100
1 1 2 3 4 5
2 2 4 6 8 10
3 3 6 9 12 15
.
.
100
```

You need to create and populate an HTML table, not just print text values. This means adding tags.

### HTML Forms

PHP combined with HTML, CSS, and JavaScript can make for some very powerful web pages. We will setup some basic HTML front-ends and PHP back-ends.

Contents of the HTML front-end:

- Form tag
  - defines (as its attributes and values)
    - Where to send the data
    - How to send the data (via the get or post method)
  - surrounds
    - Input tags (has name attributes that will be the keys in the array PHP receives)
    - Submit button (triggers submission)

Simple form (HTML front-end):

```
<html>
  <head>
  </head>
  <body>
    <form action="myBackend.php" method="post">
      Name: <input type="text" name="name"><br>
      E-mail: <input type="text" name="email"><br>
      <input type="submit">
    </form>
  </body>
</html>
```

Simple form (PHP back-end)

```
<?php
```

```
//access the global array called $_POST to get the values from the text fields
```

```
$name = $_POST["name"];  
$email = $_POST["email"];
```

```
echo "Name: " . $name . "<br>";  
echo "Email: " . $email . "<br>";
```

```
?>
```

## Exercise 2: Quiz Creation

Required files:

- Quiz.html
- Quiz.php

### Quiz.html

This HTML file will consist of the following:

- At least five questions that are multiple choice
  - Each question must have four possible answers. The answers will be in the form of radio buttons like [this picture](#)
- Below the question there will be a single submit button labeled "Submit quiz"

### Quiz.php

This file will grade the quiz.

You must display the following to the user:

- All of the questions
- Their answers
- The correct answer in the form:  
    Question 1: What was the capital of Spain?  
    You answered: Lisbon  
    Correct answer: Madrid
- The total they answered correctly
- Their score in a percent (if they get all questions right they receive a 100%, if they only get 1 right that's a 20%, etc)

## Exercise 3: Web Store

### Customer Interaction

Required files overview:

- customerFrontend.html
  - The form the user interacts with to make purchase choices, give a username and password, and choose shipping options
  - You will get to decide what your store sells, but you need to sell at least three items of different prices
  - You will need to offer 3 types of shipping (radio buttons are great for this):
    - Free 7 day
    - \$50.00 over night
    - \$5.00 three day
  - Reset button
  - Submit button
  - HINT: to give JavaScript the ability to stop submission, use the event *onsubmit*
- style.css
  - Styles the store file (including the backend!)
- formChecker.js
  - Validates the input from the customer before submitting
  - Quantities cannot be blank or negative (zero is fine)
  - Username must be in the form of an email address (name@domain.com)
  - password field cannot be blank
    - We do not have a database to check against, so the password can be anything
  - They must pick a shipping option
- customerBackend.php
  - Processes the purchase
  - Prints a welcome message to the user and displays their password (never do this in production! This is only practice to get used to transmitting form information)
  - Prints a receipt ([example](#))

### What to submit for grading

Just like in Lab 5, add links to your index.html page to create a menu to the three exercises.

Create a GitHub repository containing the source code for the required files. Submit the link and your people.eecs.ku.edu link to Blackboard by **Sunday, November 13 at 11:59 PM.**

### Grading

[15 pts] Exercise 1: Multiplication Table

[20 pts] Exercise 2: Quiz Creation

[20 pts] Exercise 3: Web Store

[5 pts] Links to your exercises in index.html