Web Engineering LAB

Lab # 09
Server-side Scripting (PHP)

Instructor: Hurmat Hidayat

Course Code: SL3003

Semester Spring 2023

Department of Computer Science,
National University of Computer and Emerging Sciences FAST
Peshawar Campus

Content

Server Side Scripting

1. Server-side scripting is responsible for the completion or carrying out a task at the server-end and then sending the result to the client-end.
2. In server-side script, it doesn't matter which browser is being used at client-end, because the server does all the work.
3. Server-side scripting is mainly used when the information is sent to a server and to be processed at the server-end. Some sample uses of server-scripting can be :
    - Password Protection.
    - Browser Customization (sending information as per the requirements of client-end browser)
    - Form Processing
    - Building/Creating and displaying pages created from a database.
    - Dynamically editing changing or adding content to a web-page.
4. Here are some popular server-side scripting languages PHP, Perl, ASP (Active Server Pages), JSP (Java Server Pages).

Setting Up the Server Locally

For Windows: https://phpandmysql.com/extras/installing-xampp/

For Mac: https://phpandmysql.com/extras/installing-mamp/#why-use-mamp

Test PHP file:

1. Open your text editor
2. Add the following php script

3. Save the file as testFile.php in your htdocs folder.
4. Go to your browser, type http://localhost:8888/

5. Click on your file "testFile.php" and you will see the output on the document.

Introduction to PHP

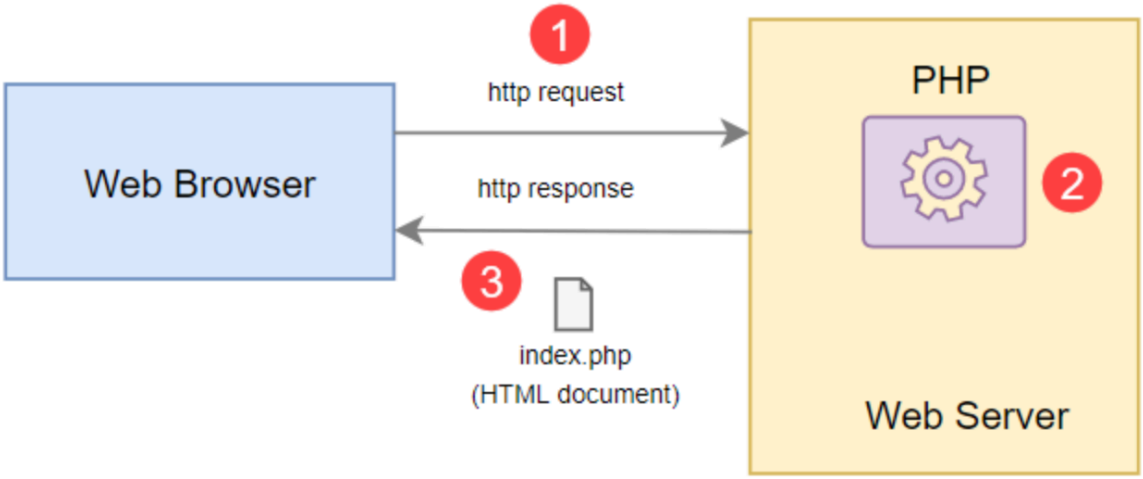
PHP is a server-side and general-purpose scripting language that is especially suited for web development. PHP originally stood for Personal Home Page. However, now, it stands for Hypertext Preprocessor. PHP was created by Rasmus Lerdorf in 1994. It's currently maintained by the PHP Development Team.

PHP runs on the web server, processes the request, and returns the HTML document.

PHP has two main applications:

- Server-side scripting – PHP is well-suited for developing dynamic websites and web applications.
- Command-line scripting – like Python and Perl, you can run PHP script from the command line to perform administrative tasks like sending emails and generating PDF files.

The following illustrates how PHP works:



PHP Syntax

A PHP script can be placed anywhere in the document. A PHP script starts with <?php and ends with ?>:

```
<?php
// PHP code goes here
?>
```

The default file extension for PHP files is ".php". A PHP file normally contains HTML tags, and some PHP scripting code. PHP statements end with a semicolon (;).

HelloPHP.php

/Applications/MAMP/htdocs/HelloPHP.php

```
<!DOCTYPE html>
<html>
<head>
    <title>Embed PHP in a .html File</title>
</head>
<body>
    <h1>
        <?php
            echo "Hello World"
        ?>
    </h1>
</body>
</html>
```

Note: In PHP, keywords (e.g. if, else, while, echo, etc.), classes, functions, and user-defined functions are not case-sensitive.

/Applications/MAMP/htdocs/practice09.php

```
<!DOCTYPE html>
<html>
<body>

<?php
    ECHO "Hello World!<br>";
    echo "Hello World!<br>";
    EcHo "Hello World!<br>";
?>

</body>
</html>
```

Write the following script and observe the output:

/Applications/MAMP/htdocs/practice09.php

```php
<!DOCTYPE html>
<html>
<body>

<?php

    $color = "red";

    echo "My car is " . $color . "<br>";
    echo "My house is " . $COLOR . "<br>";
    echo "My boat is " . $coLOR . "<br>";
?>
</body>
</html>
```

Note: $color, $COLOR, and $coLOR are treated as three different variables

Comments in PHP

```php
<!DOCTYPE html>
<html>
<body>

<?php
// This is a single-line comment

# This is also a single-line comment

/*
This is a multiple-lines comment block
that spans over multiple
lines
*/

?>

</body>
</html>
```

PHP Variables

Variables are "containers" for storing information. In PHP, a variable starts with the $ sign, followed by the name of the variable:

```php
<?php
$txt = "Hello world!";
$x = 5;
```

```
$y = 10.5;
?>
```

After the execution of the statements above, the variable $txt will hold the value Hello world!, the variable $x will hold the value 5, and the variable $y will hold the value 10.5.

/Applications/MAMP/htdocs/practice09.php

```
<!-- PHP Variables -->
<?php
    $txt = "Hello world! 2";
    $x = 5;
    $y = 10.5;

    echo "$txt" ."<br>";
    echo $x + $y;
?>
```

Global and Local Scope

A variable declared **outside** a function has a GLOBAL SCOPE and can only be accessed outside a function.

/Applications/MAMP/htdocs/practice09.php

```
<!-- Global and local scope -->

<?php
$x = 5; // global scope

function myTest() {
  // using x inside this function will generate an error
  echo "<p>Variable x inside function is: $x</p>";
}
myTest();

echo "<p>Variable x outside function is: $x</p>";
?>
```

A variable declared **within** a function has a LOCAL SCOPE and can only be accessed within that function:

/Applications/MAMP/htdocs/practice09.php

```
<?php
function myTest() {
  $x = 5; // local scope
  echo "<p>Variable x inside function is: $x</p>";
}
myTest();

// using x outside the function will generate an error
echo "<p>Variable x outside function is: $x</p>";
?>
```

The PHP echo Statement

The echo statement can be used with or without parentheses: echo or echo(). The following example shows how to output text with the echo command (notice that the text can contain HTML markup):

```
<?php
echo "<h2>PHP is Fun!</h2>";
echo "Hello world!<br>";
echo "I'm about to learn PHP!<br>";
echo "This ", "string ", "was ", "made ", "with multiple parameters.";
?>
```

The following example shows how to output text and variables with the echo statement:

```
<?php
$txt1 = "Learn PHP";
$txt2 = "for server side scripting";
$x = 5;
$y = 4;

echo "<h2>" . $txt1 . "</h2>";
echo "Study PHP " . $txt2 . "<br>";
echo $x + $y;
?>
```

PHP Data Types

PHP supports the following data types:

- String : `$x = "Hello world!";`
- Integer: `$x = 5985;`
    - The PHP var_dump() function returns the data type and value:

```
<?php
$x = 5985;
var_dump($x);
?>
```

- Float (floating point numbers - also called double): `$x = 10.365`
- Boolean: `$x = true; $y = false;`
- Array: `$cars = array("Volvo","BMW","Toyota")`
- NULL: `$x = null; var_dump($x);`
- Resource: The special resource type is not an actual data type. It is the storing of a reference to functions and resources external to PHP.
- Objects

Objects

Classes and objects are the two main aspects of object-oriented programming.

A class is a template for objects, and an object is an instance of a class.

When the individual objects are created, they inherit all the properties and behaviors from the class, but each object will have different values for the properties.

Let's assume we have a class named Car. A Car can have properties like model, color, etc. We can define variables like $model, $color, and so on, to hold the values of these properties.

When the individual objects (Volvo, BMW, Toyota, etc.) are created, they inherit all the properties and behaviors from the class, but each object will have different values for the properties.

If you create a __construct() function, PHP will automatically call this function when you create an object from a class.

/Applications/MAMP/htdocs/ClassAndObjects.php

```php
<?php
class Car {
  public $color;
  public $model;
  public function __construct($color, $model) {
    $this->color = $color;
    $this->model = $model;
  }
  public function message() {
    return "My car is a " . $this->color . " " . $this->mo
  }
}

$myCar = new Car("black", "Volvo");
echo $myCar -> message();
echo "<br>";
$myCar = new Car("red", "Toyota");
echo $myCar -> message();
?>
```

PHP Functions

The real power of PHP comes from its functions. PHP has more than 1000 built-in functions, and in addition you can create your own custom functions. Built-in functions that can be called directly, from within a script, to perform a specific task.

Besides the built-in PHP functions, it is possible to create your own functions.

- A function is a block of statements that can be used repeatedly in a program.
- A function will not execute automatically when a page loads.
- A function will be executed by a call to the function.

A user-defined function declaration starts with the word function:

```
function functionName() {
  code to be executed;
}
```

```
<?php
function writeMsg() {
  echo "Hello world!";
}

writeMsg(); // call the function
?>
```

Information can be passed to functions through arguments. An argument is just like a variable.

```
<?php
function familyName($fname) {
  echo "$fname Refsnes.<br>";
}

familyName("Jani");
familyName("Hege");
familyName("Stale");
familyName("Kai Jim");
familyName("Borge");
?>
```

PHP automatically associates a data type to the variable, depending on its value. Since the data types are not set in a strict sense, you can do things like adding a string to an integer without causing an error.

In PHP 7, type declarations were added. This gives us an option to specify the expected data type when declaring a function, and by adding the strict declaration, it will throw a "Fatal Error" if the data type mismatches.

```
<?php
function addNumbers(int $a, int $b) {
  return $a + $b;
}
echo addNumbers(5, "5 days");
// since strict is NOT enabled "5 days" is changed to int(5), and it
will return 10
?>
```

To specify strict we need to set declare(strict_types=1);. This must be on the very first line of the PHP file.

```
<?php declare(strict_types=1); // strict requirement

function addNumbers(int $a, int $b) {
  return $a + $b;
```

```
}
echo addNumbers(5, "5 days");
// since strict is enabled and "5 days" is not an integer, an error
will be thrown
?>
```

To let a function return a value, use the return statement:

```
<?php declare(strict_types=1); // strict requirement
function sum(int $x, int $y) {
  $z = $x + $y;
  return $z;
}

echo "5 + 10 = " . sum(5, 10) . "<br>";
echo "7 + 13 = " . sum(7, 13) . "<br>";
echo "2 + 4 = " . sum(2, 4);
?>
```

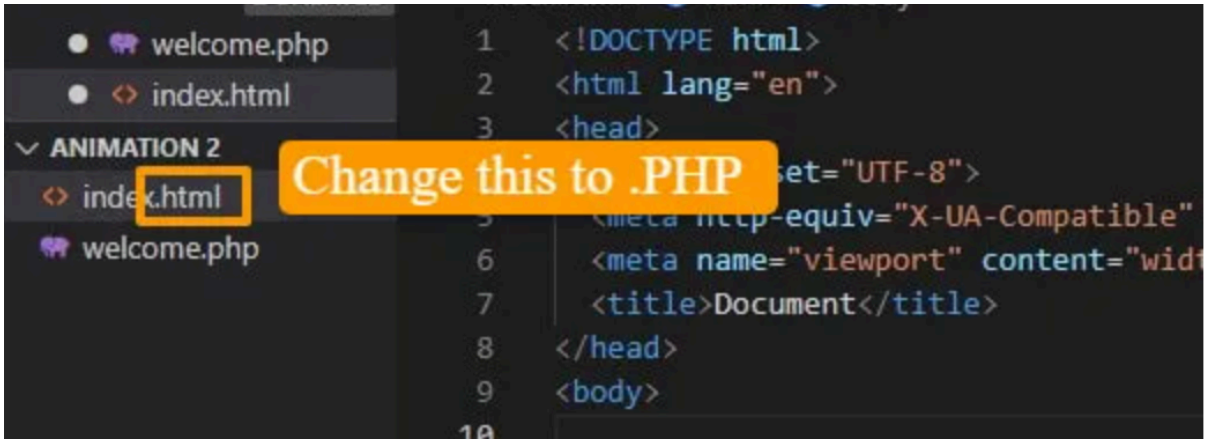
PHP 7 also supports Type Declarations for the return statement. Like with the type declaration for function arguments, by enabling the strict requirement, it will throw a "Fatal Error" on a type mismatch.

To declare a type for the function return, add a colon (:) and the type right before the opening curly ({)bracket when declaring the function.

```
<?php declare(strict_types=1); // strict requirement
function addNumbers(float $a, float $b) : float {
  return $a + $b;
}
echo addNumbers(1.2, 5.2);
?>
```

Link External PHP File to HTML

This is probably the most straightforward way to link external PHP file to HTML. All you need to do is rename your HTML file so that it has a .php extension, such as “test.html” renamed as “test.php”



After the file has been renamed to a .php extension, we can either use the include or require functions to link an external PHP script to HTML.
The include() function is used when the file is not necessary, and the program should continue when the file is not found.

```
//like
<?php
include(" welcome.php ");
?>
```

Reasons to link external PHP files include:

- Your code will be cleaner and more organized, and you will be less likely to make coding mistakes.
- Easily maintain multiple scripts as you only need to make changes in one location rather than search for and change it on each page of the site.
- Allow different pages within a website to share common functions/variables without having to repeat the code multiple times.
- It makes debugging and testing functions easier as you can test your scripts from a separate file rather than having them embedded in HTML, which complicates things.

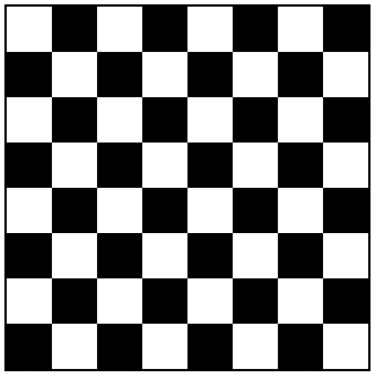
References

1. https://www.geeksforgeeks.org/web-scripting-and-its-types/
2. https://www.w3schools.com/php/default.asp

Lab Tasks:

1. Setup Xampp and create a test file.

2. Write a program to create Chess board in PHP using for loop.



3. **Classes and Objects:** Create a class named as Account (bank account), add required properties and methods that will perform withdraw and deposit functions. Create an object of the class, set initial balance, perform some deposits and a withdrawal. Display the final balance. Also add header and footer to your document using include().

4. **PHP basics and Functions:** Rewrite, understand and run the following code, add comments and briefly describe the code.

```
<?php
declare(strict_types = 1);

$candy = [
   'Toffee' => ['price' => 3, 'stock' => 12],
   'Mints'  => ['price' => 2, 'stock' => 26],
   'Fudge'  => ['price' => 4, 'stock' => 8],
];

$tax = 20;

function get_reorder_message(int $stock): string
{
   return ($stock < 10) ? 'Yes' : 'No';
}

function get_total_value(float $price, int $quantity): float
{
```

```
    return $price * $quantity;
}

function get_tax_due(float $price, int $quantity, int $tax = 0): float
{
    return ($price * $quantity) * ($tax / 100);
}
?>

<!DOCTYPE html>
<html>
  <head>
    <title>Functions Example</title>
    <link rel="stylesheet" href="css/styles.css">
  </head>
  <body>
    <h1>The Candy Store</h1>
    <h2>Stock Control</h2>
    <table>
      <tr>
        <th>Product</th><th>Stock</th><th>Re-order</th><th>Total value</th><th>Tax
due</th>
      </tr>
      <?php foreach ($candy as $product => $data) { ?>
        <tr>
          <td><?=  $product ?></td>
          <td><?=  $data['stock'] ?></td>
          <td><?=  get_reorder_message($data['stock']) ?></td>
          <td>$<?= get_total_value($data['price'], $data['stock']) ?></td>
          <td>$<?= get_tax_due($data['price'], $data['stock'], $tax) ?></td>
       </tr>
      <?php } ?>
    </table>
  </body>
</html>
```