FUNCTIONS AND PHP SUPERGLOBALS

By Hafiza Alia

PHP Functions

- **PHP Built-in Functions**
  - PHP has over 1000 built-in functions that can be called directly, from within a script, to perform a specific task.

Functions

PHP User Defined Functions

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

Syntax

- function *functionName*() {
  code to be executed;
  }
- **Note:** A function name can start with a letter or underscore (not a number).

Example

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

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

PHP Function Arguments

- Information can be passed to functions through arguments. An argument is just like a variable.
- Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.

Example

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

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

Example

- The following example has a function with two arguments ($fname and $year):
-
```
<?php
function familyName($fname, $year) {
    echo "$fname Refsnes. Born in $year <br>";
}

familyName("Hege", "1975");
familyName("Stale", "1978");
familyName("Kai Jim", "1983");
?>
```

PHP Default Argument Value

```
<?php
function setHeight($minheight = 50) {
    echo "The height is : $minheight <br>";
}

setHeight(350);
setHeight(); // will use the default value of 50
setHeight(135);
setHeight(80);
?>
```

PHP Functions - Returning values

- To let a function return a value, use the return statement:
-
```
<?php
function sum($x, $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 Global Variables - SuperGlobals

- **$GLOBALS**
- **$_SERVER**
- **$_REQUEST**
- **$_POST**
- **$_GET**
- **$_FILES**
- **$_COOKIE**
- **$_SESSION**

PHP $GLOBALS

- **$GLOBALS** is a PHP super global variable which is used to access global variables from anywhere in the PHP script (also from within functions or methods).

- PHP stores all global variables in an array called **$GLOBALS[index].** The *index* holds the name of the variable

Example

```
<?php
$x = 75;
$y = 25;

function addition() {
    $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];
}

addition();
echo $z;
?>
```

PHP $_SERVER

- $_SERVER is a PHP super global variable which holds information about headers, paths, and script locations.

Example

```
<?php
echo $_SERVER['PHP_SELF'];
echo "<br>";
echo $_SERVER['SERVER_NAME'];
echo "<br>";
echo $_SERVER['HTTP_HOST'];
echo "<br>";
echo $_SERVER['HTTP_REFERER'];
echo "<br>";
echo $_SERVER['HTTP_USER_AGENT'];
echo "<br>";
echo $_SERVER['SCRIPT_NAME'];
?>
```

Lists of the most important elements that can go inside $_SERVER

Element/Code	Description
$_SERVER['PHP_SELF']	Returns the filename of the currently executing script
$_SERVER['GATEWAY_INTERFACE']	Returns the version of the Common Gateway Interface (CGI) the server is using
$_SERVER['SERVER_ADDR']	Returns the IP address of the host server
$_SERVER['SERVER_NAME']	Returns the name of the host server (such as www.godaddy.com)
$_SERVER['SERVER_SOFTWARE']	Returns the server identification string (such as Apache/2.2.24)
$_SERVER['SERVER_PROTOCOL']	Returns the name and revision of the information protocol (such as HTTP/1.1)
$_SERVER['REQUEST_METHOD']	Returns the request method used to access the page (such as POST)
$_SERVER['HTTP_REFERER']	Returns the complete URL of the current page (not reliable because not all user-agents support it)

PHP $_REQUEST

- PHP $_REQUEST is used to collect data after submitting an HTML form.
- When a user submits the data by clicking on "Submit", the form data is sent to the file specified in the action attribute of the <form> tag.
- ***$REQUEST contains: $COOKIE, $GET, and $POST variables***

Example

```
<form method="POST" action="<?php echo $_SERVER['PHP_SELF'];?>">
  Name: <input type="text" name="fname">
  <input type="submit">
</form>

<?php
if  $_SERVER["REQUEST_METHOD"] == "POST") {
    // collect value of input field
    $name = $_REQUEST['fname'];
    if (empty($name)) {
        echo "Name is empty";
    } else {
        echo $name;
    }
}
?>
```

PHP Superglobal - $_POST

- PHP $_POST is widely used to collect form data after submitting an HTML form with method="post". $_POST is also widely used to pass variables.
- $_POST is also widely used to pass variables.

PHP Superglobal - $_POST

- **Collect data from a form with method POST**
- <?php
  if ($_SERVER["REQUEST_METHOD"] == "POST") {
      // collect value of input field
      $name = $_POST['fname'];
      if (empty($name)) {
          echo "Name is empty";
      } else {
          echo $name;
      }
  }
  ?>

PHP Superglobal - $_GET

- PHP $_GET is a PHP super global variable which is used to collect form data after submitting an HTML form with method="get".
- $_GET can also collect data sent in the URL.

PHP Superglobal - $_GET

- **Collect data from a form with method GET**
- <?php
  if ($_SERVER["REQUEST_METHOD"] == "POST") {
      // collect value of input field
      $name = $_POST['fname'];
      if (empty($name)) {
          echo "Name is empty";
      } else {
          echo $name;
      }
  }
  ?>

END OF LECTURE 5