Web Development using PHP and MySQL - Introduction to PHP and MySQL

Rahamathulla K
Assistant Professor
GEC Thrissur

January 17, 2017

Outline

PHP Introduction

- PHP is a scripting language commonly used on web servers
- Stands for PHP: Hypertext Preprocessor
- Open source
- Embedded code
- Multiple operating systems/web servers

PHP Can be used for

- Dynamic generation of web-page content
- Database interaction
- Processing of user supplied data
- Email
- File handling
- Text processing
- Network interaction

PHP Fundamentals

- PHP is embedded within xhtml pages within the tags: <?php and ?>
- The short version of these tags can also be used: <? ?>
- Each line of PHP is terminated, with a semi-colon

Hello World Example

```
<html>
<head>
<title>PHP Hello World Example </title>
</head>

<body>
<?php echo "<p>Hello World! </p>" ?>
</body>
</hmtl>
```

Literals

- All strings must be enclosed in single or double quotes.
- Numbers are not in enclosed in quotes: 1 or 45 or 34.564
- Booleans (true/flase) can be written directly as true or false.

Comments

- // This is a comment
- # This is also a comment
- /* This is a comment
  that is spread over
  multiple lines */
- // recommended over #
- <?php
  //this is a comment
  echo “ Hello World! ”
  ?>

Displaying Data

- There are two language constructs available to display data: print() and echo().
- They can be used with or without brackets
- Note that the data displayed by PHP is actually parsed by your browser as HTML. View source to see actual output
- Some characters are considered special, Escape these with a backslash \

Variables

- When we work in PHP, we often need a labelled place to store a value (be it a string, number, whatever) so we can use it in multiple places in our script
- These labelled places are called VARIABLES
- Variables are named as $ followed by variable name
- Case sensitive - $variable differs from $Variable
- Name must started with a letter or an underscore

Constants

- Constants (unchangeable variables) can also be defined
- Each constant is given a name (note no preceding dollar is applied here)
- By convention, constant names are usually in UPPERCASE
- define("AGE",35);

Single and Double Quotes

- There is a difference between strings written in single and double quotes.
- In a double-quoted string any variable names are expanded to their values
- In a single-quoted string, no variable expansion takes place
- echo "$name is $age"; - Here both name and age are expanded

Quiz

- All variables in PHP start with which symbol?
  1. $
  2. #
  3. &
  4. No need of any symbol

Quiz

- In PHP you can use both single quotes (' ') and double quotes (" ") for strings:
  1. True
  2. False

Expressions

- Variables are automatically initialised when you start to use them.
- Using variables within expressions to do something is what PHP is all about.
  - $name = 'rahaman';
- Use a dot to concatenate two strings:
  - echo $firstname.' '.$surname;

Some Types of Operator

- Arithmetic
- Assignment
- Comparison
- Logical
- String
- Incrementing/decrementing

Comparisons

- Comparison expressions return a value of TRUE (or '1') or FALSE (or '0').
- $a = 10;
- $b = 13;
- echo $a <$b;

Incrementing/Decrementing

- ++$a - Pre-increment - Increments $a by one, then returns $a.
- $a++ - Post-increment
- - -$a - Pre-decrement
- $a- - - Post-decrement

Groups of variables

- So far, we have stored ONE piece of data in each variable
- It is also possible to store multiple pieces of data in ONE variable by using an array
- Each piece of data in an array has a key
  - $name[0] = 'rahaman';
  - $name[1] = 'test';
  - $name[2] = 'another';

Arrays

- Array keys can be strings as well as numbers
  - $name['first'] = 'rahaman';
  - $name['second'] = 'test';
- Note that trying to echo an entire array will not display the data. To print an entire array to screen (for debug, for example) use the function print_r instead.
- print_r($name);

If Statement

- To do something depending on a comparison, use an if statement.
- if (comparison) {
  expressions; // do if TRUE
  }
- if ($a <$b) {
  echo 'a is smaller than b';
  }

Extending IF statements

- if (comparison) {
  expressions; // do if TRUE
  } else {
  expressions; // do otherwise
  }
- if (comparison1) {
  expressions;
  } elif (comparison2) {
  expressions;
  } else {
  expressions; // do otherwise
  }

While Loops

- Might want to do something repeatedly while a comparison is true
- while (comparison) {
  expressions;
  }
- $i = 1;
  while ($i <= 10) {
  echo $i;
  $i++;
  }

For Loops

- Sometimes we want to loop around the same bit of code a number of times.. Use a for loop.
- for (expr1; expr2; expr3) { statements; }
- expr1 evaluated/executed initially
- expr2 evaluated at beginning of each iteration (Continues if TRUE)
- expr3 evaluated/executed at end of each iteration
- for ($i=1; $i<=10; $i++) { echo $i; }

Other Control Structures

- Do-While loop
- foreach loop
- switch statement
- break, continue, and return

require, include

- require('header.php')
- include('config.php')
- require_once / include_once

Functions

- A function takes some arguments (inputs) and does something with them (echo, for example, outputs the text input to the user).
- As well as the inbuilt PHP functions, we can define our own functions
- Definition: Before using a function, that function must be defined i.e. what inputs does it need, and what does it do with them?
- Calling: When you call a function, you actually execute the code in the function.

Functions

- Function to join first and last names together with a space..

```
function make_name($first,$last)
{
$fullname = $first.' '.$last;
return $fullname;
}
```

Calling functions

- myfunction($arg1,$arg2,,$argN)
- $answer = myfunction($arg1,$arg2,,$argN)
- eg. echo make_name(rahaman,k);

Scope

- A function executes within its own little protected bubble, or local scope.
- Variables within a function Are local to that function Disappear when function execution ends
- Variables outside a function Are not available within the function Unless set as global
- Remembering variables - Not stored between function calls Unless set as static

File Handling

- Open File - fopen()
- $handle = @fopen("textfile.txt", "r");
- feof() - check end of file
- fgetc() - Read Single Character
- fgets() - Read Single Line
- $buffer = fgets($handle);

CSV File Handling

- Comma Separated Values
- $handle = @fopen("textfile.txt", "r");
- $arr= fgetcsv($file);
- echo "$arr[1];

HTML Forms

- The form is enclosed in form tags
- <form action=path/to/submit/page method=get>
  <- form contents ->
  </form>

Form tags

- action="..." is the page that the form should submit its data to
- method="..." is the method by which the form data is submitted. The option are either get or post. If the method is get the data is passed in the url string, if the method is post it is passed as a separate file.

Form fields: text input

- Use a text input within form tags for a single line freeform text input.
- <label for=fn">First Name </label >
  <input type="text"
  name="firstname"
  id=fn"
  size="20" />
- name=".." is the name of the field. You will use this name in PHP to access the data.
- id=".." is label reference string this should be the same as that referenced in the ¡label¿ tag.
- size=".." is the length of the displayed text box (number of characters).

Form fields: password input

- Use a starred text input for passwords.
- <label for=pw”>Password </label >
  <input type=”password”
  name=”password”
  id=pw”
  size=”20” />

Form fields: drop down

- <label for="tn" >Where do you live </label >
  <select name="town" id="tn">
  <option value="swindon" >Swindon </option >
  <option value="london selected="selected">London
  </option >
  <option value=bristol" >Bristol </option >
  </select >

Form fields: radio buttons

- <input type="radio"
  name="age"
  id="u30
  checked=checked value="Under30" />
  <label for="u30">Under 30 </label>
  <input type="radio"
  name="age"
  id="thirty40"
  value="30to40" />
  <label for="thirty40">30 to 40 </label>

Submit button

- <input type="submit"
  name="submit"
  value="Submit" />

Form Variables in PHP

- The form variables are available to PHP in the page to which they have been submitted.
- The variables are available in two superglobal arrays created by PHP called $_POST and $_GET.
- Access submitted data in the relevant array for the submission type, using the input name as a key.
- eg. $email = $_GET['email'];

Registration Page

- Create HTML form with Username field
- Password field
- date of Birth
- Department
- Organization

Login Page

- Username
- Password
- Login Button

MySQL

- Relational DBMS
- Open Source
- phpmyadmin - web based tool
  - Manage databases and tables
  - Manage users

MySQL Connection

- MySQL Commands
  - $link=mysql_connect($dbhost ,$dbuser ,$dbpasswd)
  - mysql_select_db($db,$link)
  - $resource = mysql_query($query,$link)
  - $result = mysql_fetch_array($resource)