CHAPTER 2 INTRODUCTION TO PHP

Part 2

What is PHP?

- PHP (stands for: Hypertext Preprocessor") is a widely-used Open Source Scripting language that is especially suited for Web development and can be embedded into HTML
- PHP is a server-side, cross-platform, HTML embedded scripting language

PHP

- Server-side - PHP scripts execute on the Web server, *not within the browser* on the local machine.
- Cross-platform - PHP scripts can run on many different operating systems and Web servers.
- HTML embedded scripting language - PHP statements and commands are embedded in the HTML documents.

Features of PHP

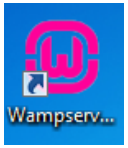
- Build templates to ease site maintenance
- Create graphics
- Connect to databases like Oracle, Informix or MySQL to the Web
- Communicate with external Web sites
- Build discussion forums or Web-based e-mail programs
- etc…

Basic Syntax

- PHP script is executed on the server, and the plain HTML result is sent back to the browser.
- PHP script can be placed anywhere in the document.
- PHP script starts with **<?php** and ends with **?>**
- The default file extension for PHP files is ".php".
- PHP file normally contains HTML tags, and some PHP scripting code.
- PHP statements end with a semicolon (;).

Sending data to the browser

- Double click Wampserver64

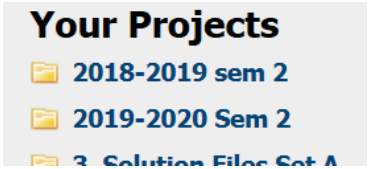


- Ensure that the server is running

- Click the Green icon and choose localhost

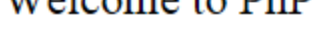
- The WAMPServer home page will be displayed. (http://localhost/)

- Check the folder in **www** under **Your Projects**



- Click the folder to view your files

- Click the file to view the output



Basics of PHP

- PHP programs are written using a text editor such as Notepad++ or IDE like Microsoft Expression Web 4.

Writing Comments

- PHP supports several ways of comments: Single Line and Multi-lines comments.
- Example of single line comment

```
// This is a single-line comment
```

- Example of multiple line comment

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

Example

- Open the editor (e.g. Microsoft Expression Web 4) and type the following statements

```
<?php
echo "Welcome to PhP";
?>
```

- Create a folder in C:\wamp64\www and save the program (eg1.php) in the newly created folder.

Integrating HTML with PHP

- echo "Welcome to
 HCT ";
- echo "<h1> My PHP Website </h1>";

Output

Welcome to
HCT

My PHP Website

Variables

- Used to store both numeric and non-numeric data
- Variable name should be preceded by a dollar [$] sign and must begin with a letter, optionally followed by more letters, numbers or underscore character.
- Variable names are case-sensitive ($age and $AGE are two different variables).

How to define a variable?

- **Example 1**

$country="Oman"; – country is a variable and its value is Oman.

echo $country – will print the value Oman

- **Example 2**

$n = 5; – n is the name of the variable and 5 is its value.

echo $n – will print the value 5

Data types

- String
- Integer
- Float (floating point numbers - also called double)
- Boolean
- Array
- Object
- NULL
- Resource

var_dump() function

- Returns the data type and value
- Example

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

Output

```
C:\wamp64\www\2020-2021 Sem 1\1.php:16:int 5985
```

Introducing Strings

Strings can be seen as a stream of characters. For example, ‘G’ is a character and ‘GeeksforGeeks’ is a string.

Operator	Name	Example	Result
.	Concatenation	$txt1 . $txt2	Concatenation of $txt1 and $txt2
.=	Concatenation assignment	$txt1 .= $txt2	Appends $txt2 to $txt1

String function

- **strlen()** function returns the length of a string.
- **str_word_count()** function counts the number of words in a string.
- **strrev()** function reverses a string.

Example (String)

```
<?php
echo strlen("Hello world!")."<br>";
echo str_word_count("Hello world!")."<br>";
echo strrev("Hello world!");
?>
```

Output

```
12
2
!dlrow olleH
```

PHP echo() Function (string)

The echo() function outputs one or more strings.

Note: The echo() function is not actually a function, so you are not required to use parentheses with it. However, if you want to pass more than one parameter to echo(), using parentheses will generate a parse error.

Syntax

```
echo(strings)
```

One or more strings to be sent to the output

Introducing Numbers

PHP has both integer and floating-point (decimal) number types.

these two types can be classified under the generic title numbers without losing much valuable distinction. Valid numbers in PHP can be anything like

8

3.14 10980843985

–4.2398508 4.

4e2

Notice that these values are never quoted—quoted numbers are strings with numeric values—nor do they include commas to indicate thousands. Also, a number is assumed to be positive unless it is preceded by the minus sign (–).

Introducing Constants

- A constant is a placeholder for a value that you refer within your code.
- Constants are typically named with capital letters
- Constant names must begin with a letter or an underscore and cannot begin with a number.
- The value cannot be changed during the script.

How to define a constant?

- **Example 1**

define(“A”, 10); – A is a constant and its value is 10.

echo A – will print the value 10

- **Example 2**

define (“College”, “HCT”); – College is the name of the constant and HCT is its value.

echo College – will print the value HCT

Single vs. Double Quotation Marks

In PHP, it's important to understand how single quotation marks differ from double quotation marks.

- When writing an expression, a constant or a variable in PHP which needs quotation marks you can always choose between using single quotation marks or double quotation marks. It really doesn't matter if you choose single or double quotation marks, as long as it are straight marks (' of ") and not italic marks (‘ of “).

- When your variable or expression consists of text (a string) you do need to differentiate single and double quotation marks. It is sometimes required to ‘escape’ the normal text processing. Therefore an escape can be used. When using double quotation marks multiple escape characters can be used contrary to using single quotation marks. An escape character needs to be indicated by means of a backslash \.

Single vs. Double Quotation Marks

Single Quotes

Single quotes are the simplest way to make a string. They just display what they are given, no bells and whistles, no special "powers" like being able to show variable values (see below in the Double Quotes section).

```
// Using single quotes to save a string in a variable:
$recipe_title = 'Meatball Spaghetti';

// Using single quotes to write something on the screen:
echo '<h1>Meatball Spaghetti</h1>';

// The line above will get output as-is in your code:
<h1>Meatball Spaghetti</h1>
```

Double Quotes

One big difference about double quotes vs single quotes is that you can use double quotes to include variables directly inside the string. If you use single quotes, you would have to concatenate the pieces together. Let's see an example.

Let's say you have recipes and you save the titles into a variable called `$recipe_title`:

```
$recipe_title = 'Meatball Spaghetti';
```

If you want to create the HTML for recipe titles so that they look like this (and you aren't embedding PHP directly into HTML files, in which you might use `<?php ?>` tags instead to spit out variables):

```
<h1>Meatball Spaghetti</h1>
```

Using single quotes you need to add the different parts together:

```
echo '<h1>' . $recipe_title . '</h1>';
```

With double quotes, however, you can put the variable directly inside the quotes:

```
echo "<h1>$recipe_title</h1>";
```