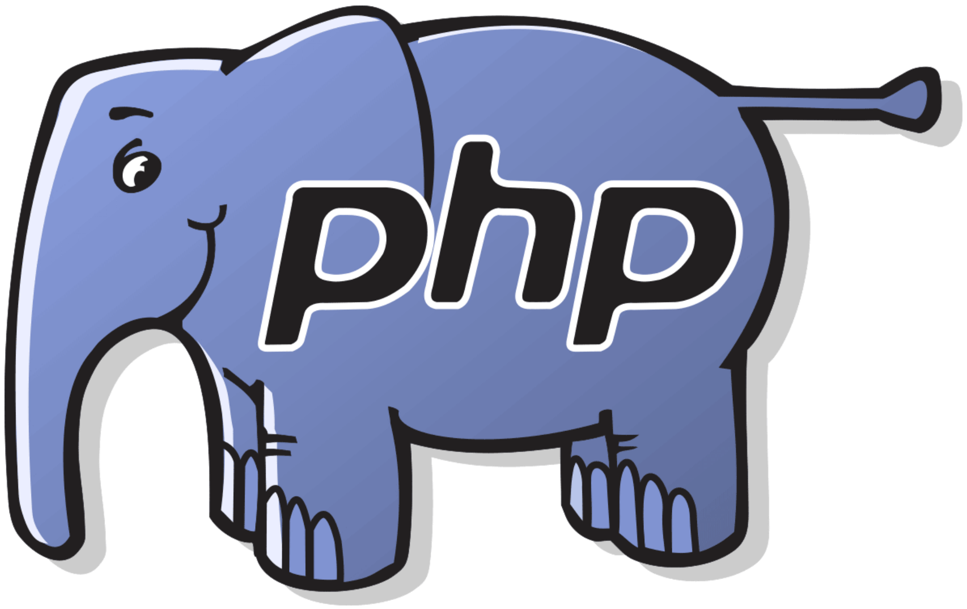

MODCOM
Institute of Technology

Teaching Techniques

- ✓ Class Slides
- ✓ Examples
- ✓ Labs
- ✓ Assignments

PHP- Basics

- ✓Introduction to PHP
- ✓PHP Syntax
- ✓Variables
- ✓Operators
- ✓Control Structures
- ✓Functions
- ✓Arrays

Introduction to PHP

- PHP an acronym for Hypertext Pre-processor
- PHP is a widely-used, open source scripting language
- PHP scripts are executed on the server
- PHP costs nothing, it is free to download and use

What is PHP File

- PHP files can contain text, HTML, CSS, JavaScript, and PHP code

- PHP code are executed on the server, and the result is returned to the browser as plain HTML

- PHP files have extension ".php"

What Can PHP Do ?

- PHP can generate dynamic page content
- PHP can create, open, read, write, and close files on the server
- PHP can collect form data
- PHP can add, delete, modify data in your database
- PHP can restrict users to access some pages on your website
- PHP can encrypt data

Why Php?

- PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- PHP is compatible with almostall servers used today (Apache, IIS, etc.)
- PHP supports a wide range of databases
- PHP is free.
- PHP is easy to learn and runs efficiently on the server side

Tools for development

- Text Editor e.g. Sublime Text
- Web Browser
- Web server e.g. WampServer /Xamp
- Database e.g. MySQL

How it works

- When someone visits your PHP webpage, your web server processes the PHP code.

- It then sees which parts it needs to show to visitors(content and pictures).

  - hides the other stuff(file operations, math calculations, etc.) then translates your PHP into HTML.

- After the translation into HTML, it sends the webpage to your web browser.

PHP Syntax

The PHP script is executed on the server, and the plain HTML result is sent back to the browser.

- ☐ **Syntax -** The rules that must be followed to write properly structured code.

- ☐ PHP's syntax and semantics are similar to most other programming languages.

- ☐ (C, Java,Perl) with the addition that all PHP code is contained with a tag, of sorts.

- ☐ All PHP code must be contained within the following...

PHP Syntax

```
<?php
   Your code here;
 ?>
```

Php uses a .php extension

A PHP file normally contains HTML tags, and some PHP scripting code.

PHP Syntax

If you have PHP inserted into your HTML and want the web browser to interpret it correctly, then you must save the file with a *.php* extension.

PHP Syntax

```
<!DOCTYPE html>
<html>
<head>
<title>My first PHPpage</title>
<head>
<body>
<?php
   echo "Hello World!";
 ?>


</body>
</html>
```

Save the file with .php extension

Note: *PHP statements are terminated by semicolon (;)*

PHP Example

```
<?php
  echo "Hello World!   Welcome to php";
?>
```

Display

```
Hello World!   Welcome to php
```

PHP-Echo

- ✓ The PHP function *echo* is a means of outputting text to the web browser.

- ✓ Throughout your PHP career you will be using the echo function more than any other.

The Semicolon!

- As you mayor may not have noticed in the above example, there was a semicolon after the line of PHP code.

- The semicolon signifies the end of a PHP statement and should never be forgotten.

Comments in PHP

Acomment in PHP code is a line that is not read/executed as part of the program.

Its only purpose is to be read by someone who is editing the code!
Comments are useful for:

✓To let others understand whatyou are doing

✓To remind yourself what youdid

PHP supportsthree waysof commenting:

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

# This is also a single line comment

/*
This is a multiple lines comment block
that spans over more than
one line
*/
?>
```

PHP Variables

Variables

Variables are "*containers*" for storing information:

Much Like Algebra

x=5
y=6
z=x+y
In algebra we use letters (like x) to hold values (like 5).
From the expression z=x+y above, we can calculate the value of z to be 11.

In PHP these letters are called **variables.**

Example

In PHP you define a variable with the following form

```
$variable_name =Value;
```

```
<?php

  $x=15;
  $y=26;
  $z=$x+$y;
  echo $z;

?>
```

A variable can have a short name (like x and y) or a more descriptive name (age,car, quantity etc).

Rules for Variables

- ✓ A variable starts with the $ sign, followed by the name of the variable
- ✓ A variable name must start with a letter or the underscore character
- ✓ A variable name cannot start with a number
- ✓ A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- ✓ Variable names are case sensitive ($y and $Y are two different variables)

Echoing Variables

Here are the most important things to know about variables in PHP

- No quotations are required, even if the variable does not hold a string.
- Below is the correct format for echoing a variable.

```
<?php
$my_string = "Hello Bob. My name is: ";
$my_name = “John”;
echo $my_string;
echo $my_name;
?>
```

Display

```
Hello Bob. My name is: John
```

Task

Write a php program to calculate the netpay of an employee given the following:

Basic pay: 80000

Deductions

NHIF=3000
NSSF=4000
LOAN=15000
P.A.Y.E=10000

The program should display the output on the browser window in the following format

```
Basic pay=
------------------
Deductions
------------------
NHIF=
NSSF=
LOAN=
P.A.Y.E=
------------------
Net  Pay=
-------------------
```

Echoing Variables and Text Strings

- ✓ You can also combine text strings and variables.
- ✓ By doing such a conjunction you save yourself from having to do a large number of echo statements.
- ✓ The example below shows how to do such a combination
- ✓ “Period”. Is used for combination

Echoing Variables and Text Strings

```
<?php
$my_string = "Hello Bob. My name is: ";
$newline = "<br />";
echo $my_string.“Eric".$newline;
echo "Hi, I'm Bob. Who are you?".$my_string.$newline;
echo "Hi, I'm Bob. Who are you?".$my_string.“Eric";
?>
```

NB: while using html tags in php they must be quoted i.e “<br/>”

PHP - Strings

Throughout your PHP career you will be using strings a great deal.

So it is important to have a basic understanding of PHP strings

PHP - String Creation

- ✓Before you can use a string you have to create it!
- ✓A string can be used directly in a function or it can be stored in a variable.
- ✓Below we create the exact same string twice: first storing it into a variable, in the second case we place the string directly into function

PHP - String Creation

```
<?php
$my_string=“Dell - Unlock your potential!";
echo “Dell - Unlock your potential!";
echo $my_string;
?>
```

Output

```
Dell - Unlock your potential! Dell - Unlock your potential!
```

Cont..

- In the above example the first string will be stored into the variable *$my_string*.
- While the second string will be used in the echo function and **not** be stored.
- Remember to save your strings into variables if you plan on using them more than once!

PHP - String Creation Single Quotes..

Thus far we have created strings using double-quotes.

But it is just as correct to create a string using single-quotes, otherwise known as apostrophes.

PHP - String Creation Single Quotes..

```
<?php

$my_string = 'Tizag - Unlock your potential!';
echo 'Tizag - Unlock yourpotential!';
echo $my_string;

?>
```

Operators

Arithmetic Operator

Operator	Description	Example	Result
+	Addition	x=2 x+2	4
-	Subtraction	x=2 5-x	3
*	Multiplication	x=4 x*5	20
/	Division	15/5 5/2	3 2.5
%	Modulus (division remainder)	5%2 10%8 10%2	1 2 0
++	Increment	x=5 x++	x=6
--	Decrement	x=5 x--	x=4

Comparison Operator

Operator	Description	Example
==	is equal to	5==8 returns false
!=	is not equal	5!=8 returns true
<>	is not equal	5<>8 returns true
>	is greater than	5>8 returns false
<	is less than	5<8 returns true
>=	is greater than or equal to	5>=8 returns false
<=	is less than or equal to	5<=8 returns true

Logical Operator

Operator	Description	Example

&&	and	x=6
		y=3 (x < 10 && y > 1) returns true

||	or	x=6
		y=3 (x==5 ||y==5) returns false

!	not	x=6
		y=3 !(x==y) returns true

Control Structures

Control Structures

Control structures are statements that control the order of program execution.

The following are the most commonly used control structures

- ✓ **if statement**
- ✓ **if...else statement**
- ✓ **if...elseif....else statement**
- ✓ **Switch statement**
- ✓ **while loop**
- ✓ **for loop**
- ✓ **do... While loop**

if statement

The if statement is used to execute some code **only if a specified condition is true**.

Syntax

```
if (condition)
  {
  code to be executed if condition is true;
  }
```

if statement

```
<?php
$meangrade=85;
if ($meangrade>“50”)
   {
  echo “Pass!";
   }
?>
```

if. . ..else statement

The if....else statement is used to execute some code **if a condition is true and another code if the condition is false**.

Syntax

```
if (condition)
 {
  code to be executed if condition is true;
 }
else
 {
  code to be executed if condition is false;
 }
```

If. . ..else statement

```
<?php
$meangrade=85;
if ($meangrade>50)
   {
      echo "Pass!";
   }
else
   {
       echo "Fail!";
   }
?>
```

if. . ..elseif. . .else statement

The if....elseif...else statement to **select one of several blocks of code to be executed**

Syntax

```
if (condition)
  {
  code to be executed if condition is true;
  }
elseif (condition)
  {
  code to be executed if condition is true;
 }
else
  {
  code to be executed if condition is false;
 }
```

if. . ..elseif. . ..else statement

The example below will output Distinction if the mean grade is >80 and Credit if the mean grade is >70, otherwise it will output pass.

Example

```
<?php
$meangrade=85;
if ($meangrade>80)
   {
    echo “Distinction!";
   }
elseif($meangrade>70)
   {
        echo “Credit!";
   }
 else
   {
       echo ‘Pass!";
   }
?>
```

MODCOM Institute of Technology

PHP assignment I

- Using if… elseif…. Else and comparison operators create a grading system that will give a student his/her grade and average Mark based on the following criteria. The input of five subject should be done via a variable initialization.

Average Mark	Grade
80-100	Distinction
70-79	Credit
60-69	Pass
0-59	Fail

PHP Loops

PHP Loops

Often when you write code, you want the same block of code to run over and over again in a row.

In PHP, we have the following looping statements:

- ✓**while** - Execute a block of code as long as the specified condition is true
- ✓**do...while** - loops through a block of code once, and then repeats the loop as long as the specified condition is true.
- ✓**for** - loops through a block of code a specified number of times
- ✓**foreach** - loops through a block of code for each element in an array

While loop

Syntax

```
while (condition is true)
    {
    code to be executed;
    }
```

While loop

Syntax

```
while (condition is true)
    {
    code to be executed;
    }
```

While loop

Example

The example below first sets a variable $x to 5 ($x=5;). Then, the while loop will continue to run as long as $x is less than, or equal to 15. $x will increase by 1 each time the loop runs ($x++;):

While loop

Example

```
<?php
  $x=5;
  while($x<=15)
    {
    echo "The number is: $x <br>";
    $x++;
    }
  ?>
```

Assignment

Using while loop write a PHP program that will generate all the years between 1970 and 2014

PHP do. . .while loop

Example

The do...while loop will always execute the block of code once, it will then check the condition, and repeat the loop while the specified condition is true.

Syntax

```
do
  {
  code to be executed;
  }
while (condition is true);
```

MODCOM Institute of Technology

PHP do. . .while loop

Example

The example below first sets a variable $x to 20 ($x=20;).

Then, the do while loop will write some output, and then increment the variable $x with 1.

Then the condition is checked (is $x less than, or equal to 50?), and the loop will continue to run as long as $x is less than, or equal to 50:

PHP do. . .while loop

Example

```
<?php
  $x=20;
  do
    {
      echo "The number is: $x <br>";
      $x++;
    }
  while ($x<=50);
  ?>
```

PHP do. . .while loop

Note that in a do while loop the condition is tested AFTER executing the statements within the loop.

This means that the do while loop would execute its statements at least once, even if the condition fails the first time.

PHP for. . .loop

The for loop is used when you know in advance how many times the script should run.

Syntax

```
for (init counter; test counter; increment counter)
    {

    code to be executed;

    }
```

PHP for. . .loop

Parameters:

init counter: Initialize the loop counter value

test counter: Evaluated for each loop iteration. If it evaluates to TRUE, the loop continues. If it evaluates to FALSE, the loop ends.

increment counter: Increases the loop counter value

PHP for. . .loop

Example:

The example below displays the message welcome to php 10 times

```
<?php

  for ($counter=1; $counter<=10; $counter++)
    {
    echo “Welcome to PHP <br>";
    }

?>
```

PHP for. . .loop

Example:

The example below displays the message welcome to php 10 times

```
<?php

   for ($counter=1; $counter<=10; $counter++)
     {
     echo “Welcome to PHP <br>";
     }

?>
```

PHP foreach loop

The foreach loop works only on arrays, and is used to loop through each key/value pair in an array.

Syntax

```
<?php

foreach ($arrayName as $value)
    {
    code to be executed;
    }

?>
```

PHP foreach loop

For every loop iteration, the value of the current array element is assigned to $value and the array pointer is moved by one, until it reaches the last array element.

The following example demonstrates a loop that will output the values of the given array ($cars)

Syntax

```
<?php
   $cars = array(“BMW",“Toyota",“Nissan");
   foreach ($cars as $value)
     {
     echo "$value <br>";
     }
?>
```

PHP Arrays

PHP Arrays

An array is a special type of a variable that stores one or more values in a single variable.

An array can hold many values under a single name, and you can access the values by referring to an index number.

In PHP, there are 2 types of commonly used arrays:

- ✓ **Indexed arrays** - Arrays with numeric index
- ✓ **Associative arrays** - Arrays with string index

PHP Indexed Arrays

There are two waysto create indexed arrays:
The index can be assigned automatically (index alwaysstarts at 0):

```
$cars=array( “Nissan","BMW","Toyota“);
```

or the index can be assigned manually:

```
$cars[0]=“Nissan";
$cars[1]="BMW";
$cars[2]="Toyota";
```

PHP Indexed Arrays

The following example creates an indexed array named $cars, assigns three elements to it, and then prints a text containing the array values:

```
<?php

$cars=array(“Nissan","BMW","Toyota");
echo "I like" . $cars[0] . ", " . $cars[1] . " and " . $cars[2] . ".";

?>
```

Output

```
I like Nissan, BMW, and Toyota
```

PHP Indexed Arrays

The following example creates an indexed array named $cars, assigns three elements to it, and then prints a text containing the array values:

```
<?php

$cars=array(“Nissan","BMW","Toyota");
echo "I like" . $cars[0] . ", " . $cars[1] . " and " . $cars[2] . ".";

?>
```

Output

```
I like Nissan, BMW,and Toyota
```

Loop through an indexed array

Toloop through and print all the values of an indexed array,
you could use a for loop, like this:

Example

```
<?php
$cars=array(“Nissan","BMW","Toyota");
$arrlength=count($cars);

for($x=0;$x<$arrlength;$x++)
 {
 echo $cars[$x];
 echo "<br>";
 }
?>
```

PHP Associative array

Associative arrays are arrays that use named keysthat you assign to them. There are two waysto create an associative array:

```
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");
```

or:

```
$age['Peter']="35";
$age['Ben']="37";
$age['Joe']="43";
```

```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");
echo "Peter is " . $age['Peter'] . " years old.";
?>
```

Loop through an associative Array

To loop through and print all the values of an associative array, you could use a foreach loop, like this:

```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");

foreach($age as $x=>$x_value)
 {
 echo "Key=". $x . ", Value=" . $x_value;
 echo "<br>";
 }
?>
```

PHP Functions

PHP Functions- user defined Functions

Afunction is a piece of code which takes one or more input in the form of parameter and does some processing and returns a value.

There are two parts which should be clear to you:

- Creating a PHP Function
- Calling a PHP Function

Creating PHP Functions

Suppose you want to create a PHP function which will simply write a simple message on your browser when you will call it.

Following example below creates a function called writeMessage() and then calls it just after creating it.

```
<?php
//Defininga PHP Function
function writeMessage()
 {
echo "You are really a nice person, Have a nice time!";
 }
 //Calling a PHPFunction
writeMessage();

?>
```

Output

```
You are really a nice person, Have a nice time!
```

Note

- ✓Function name should start with keyword **function**
- ✓All the PHP code should be put inside { and } braces
- ✓Use function name to call a function

Output

```
You are really a nice person, Have a nice time!
```

Note

- ✓Function name should start with keyword **function**
- ✓All the PHP code should be put inside { and } braces
- ✓Use function name to call a function

PHP Functions

We have already looked at user defined functions.

This lesson we are going to look on PHP inbuild functions.

These include; the date function, include, strpos,strtlower, date etc.

PHP Predefined Functions

We have already looked at user defined functions.

This lesson we are going to look on PHP inbuild functions.

- INCLUDE FUNCTION
- DATE FUNCTION

PHP Date

While PHP's *date()* function may seem to have an overwhelming amount of options available.

Isn't it always better to have more choices than not enough?

With PHP's date function you format timestamps, so they are more human readable.

PHP Date

This lesson will teach you how to display the current time.

Formating PHP's timestamp, and show you all the various date arguments for reference purposes.

PHP Date

- The date function uses letters of the alphabet to represent various parts of a typical date and time format.

- The letters we will be using in our first example are:

  - **d**: The day of the month. The type of output you can expect is 01 through 31.
  - **m**: The current month, as a number. You can expect 01 through 12.
  - **y**: The current year in two digits ##. You can expect 00 through 99

Date

```
<?php
echo date("m/d/y");
echo date("m-d-y");


?>
```

Display:

```
02/27/10
02-27-10
```

Square Root - sqrt

```
<?php
echo sqrt(9); // 3
echo sqrt(10); // 3.16227766 ...
?>
```

Display:

```
3
3.16227766 ...
```

Check out more on PHP tutorial
http://www.w3schools.com/php/
http://www.tutorialspoint.com/php/
http://www.tizag.com/phpT/