Lecture 2 – ICT1233

Server-Side Scripting

Introduction to PHP

Department of ICT

Faculty of Technology

Objectives

- After the successful completion of this lecture, students should be able to,
  - Identify what is server side scripting is
  - Compare PHP with other server side scripting languages and explain importance of PHP
  - Define valid PHP variables and identify data types used in PHP
  - Apply PHP for developing Web Applications

Server Side Scripting

- Server side scripting is,
  - A web server technology in which a user's request is fulfilled by running a script directly on the webserver to generate dynamic web pages
  - Used to provide interactive web sites that interface to databases or other data stores on the server

PHP

- PHP was conceived on 1994 by Rasmus Lerdorf
- PHP is the acronym for "PHP: Hypertext Preprocessor"
- PHP pages contain HTML with embedded code that does "something“
- PHP code is enclosed in special start and end processing instructions that allow you to jump into and out of the PHP mode
    - <?php……… ?>

Features of PHP

- Free to download from www.php.net
- Open source, which is able to view, modify and redistribute source code
- Support different database management systems including MySQL
- Platform independent
- Compatible with almost all servers used today (Apache, IIS)

PHP Variables

- A variable is a representation of a particular value
- PHP variables can be used to hold values or expressions
- PHP has no command for declaring a variable
- It is created the moment you first assign a value to it
- Ex:-

```
$txt="Hello world!";

$x=5;
```

Rules for PHP 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 ($age and $AGE are two different variables)

Example : What is the Output???

```
<?php
        $num1=5;
        $num2=6;
        echo "Number 1 is ".$num1."<br/>";
        echo "Number 2 is ".$num2."<br/>";
        //addition
        $result=$num1+$num2;
        echo "Result is ".$result; //hold value of expression
?>
```

PHP Variables Scope

- The scope of a variable is the part of the script where the variable can be referenced/used.

- PHP has three different variable scopes:

  - ✓ local
  - ✓ global
  - ✓ static

Global and Local Scope

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

```
<?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>";
?>
```

Global and Local Scope

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

```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 Global keyword

The global keyword is used to access a global variable from within a function.

```php
<?php
$x = 5;
$y = 10;

function myTest() {
  global $x, $y;
  $y = $x + $y;
}

myTest();
echo $y; // outputs 15
?>
```

The Static keyword

Normally, when a function is completed/executed, all of its variables are deleted. However, sometimes we want a local variable NOT to be deleted. We need it for a further job.

To do this, use the static keyword when you first declare the variable:

```
<?php
function myTest() {
  static $x = 0;
  echo $x;
  $x++;
}

myTest();
myTest();
myTest();
?>
```

Exercise : What is the output?

```
function updateCounter()
{
	static $counter = 0;
	$counter++;
	echo "Static counter is now {$counter}<br>";
}
$counter = 10;
updateCounter();
updateCounter();
echo "Global counter is {$counter}\n";
```

Data Types

- Data type is a classification based on the types of data
- PHP support following data types,
- Scalar(single value) types
    - Boolean
    - integer
    - float
    - string

Data Types

- Compound(collection) types
    - array
    - object
- Special types
    - resources
    - NULL

Operators

- Operators are used to perform operations on variables and values
- An ***operator*** takes some values (the **operands**) and does something
- PHP divides the operators in the following groups:
  - Arithmetic operators
  - Assignment operators
  - Comparison operators
  - Increment/Decrement operators
  - Logical operators

Arithmetic Operators

Operator	Name	Example	Result
+	Addition	$x + $y	Sum of $x and $y
-	Subtraction	$x - $y	Difference of $x and $y
*	Multiplication	$x * $y	Product of $x and $y
/	Division	$x / $y	Quotient of $x and $y
%	Modulus	$x % $y	Remainder of $x divided by $y
**	Exponentiation	$x ** $y	Result of raising $x to the $y'th power

Assignment Operators

Assignment	Same as...	Description
x = y	x = y	The left operand gets set to the value of the expression on the right
x += y	x = x + y	Addition
x -= y	x = x - y	Subtraction
x *= y	x = x * y	Multiplication
x /= y	x = x / y	Division
x %= y	x = x % y	Modulus

Comparison Operators

Operator	Name	Example	Result
==	Equal	$x == $y	Returns true if $x is equal to $y
===	Identical	$x === $y	Returns true if $x is equal to $y, and they are of the same type
!=	Not equal	$x != $y	Returns true if $x is not equal to $y
<>	Not equal	$x <> $y	Returns true if $x is not equal to $y
!==	Not identical	$x !== $y	Returns true if $x is not equal to $y, or they are not of the same type
>	Greater than	$x > $y	Returns true if $x is greater than $y

Comparison Operators

Operator	Name	Example	Result
<	Less than	$x < $y	Returns true if $x is less than $y
>=	Greater than or equal to	$x >= $y	Returns true if $x is greater than or equal to $y
<=	Less than or equal to	$x <= $y	Returns true if $x is less than or equal to $y
<=>	Spaceship	$x <=> $y	Returns an integer less than, equal to, or greater than zero, depending on if $x is less than, equal to, or greater than $y. Introduced in PHP 7.

Increment / Decrement Operators

Operator	Name	Description
++$x	Pre-increment	Increments $x by one, then returns $x
$x++	Post-increment	Returns $x, then increments $x by one
--$x	Pre-decrement	Decrements $x by one, then returns $x
$x--	Post-decrement	Returns $x, then decrements $x by one

Logical Operators

Operator	Name	Example	Result
and	And	$x and $y	True if both $x and $y are true
or	Or	$x or $y	True if either $x or $y is true
xor	Xor	$x xor $y	True if either $x or $y is true, but not both
&&	And	$x && $y	True if both $x and $y are true
||	Or	$x || $y	True if either $x or $y is true
!	Not	!$x	True if $x is not true

Concatenation Operator

- Use . (period)
- Combine strings: putting two string values together
- Ex:

```
$n = 5;
$s = 'There were ' . $n . ' ducks.';

// $s is 'There were 5 ducks'
```

Comments

- There are 2 ways of using comments in PHP.
  - Single line Comments

    //

    #

  - Multiline Comments

    /* comment text */

Conditional Statements

- Use to perform different actions for different conditions
- PHP supports following conditional statements,
  - if statement
  - if..else statement
  - if ...elseif..else statement
  - switch statement

If Statement

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

- **Ex:-**

```
<?php
    $num = 0;
    if ($num == 0){
        echo "Number is equal to 0";
    }
?>
```

If….else Statement

- Execute some code if a condition is true and another code if the condition is false
- Ex:-

```
<?php
      $num = 0;
      if ($num == 0){
            echo "Number is equal to 0";
      }
      else{
            echo "Number is not equal to 0";
      }
?>
```

If… else if… else statement

- Select one from the several blocks of code to be executed

- **Ex:-**

```
<?php
        $num = 0;
        if ($num > 5){
                echo "Number is greater than 5";
        }
        elseif ($num > 2){
                echo "Number is greater than 2";
        }
        else {
                echo "Number is 0 or less than 0";
        }
?>
```

Switch

- Select one of many blocks of code to be executed
- Ex:

```
switch($name) {
        case ‘John':
        echo “Name is John”;
        break;
        case 'Mary':
        echo “Name is John”;
        break;
        default:
        echo “No name to display”;
        break;

}
```

Exercise

- Write down the php code for the following requirement
  - Initialize the variable $city with the value “Matara”
  - If city=Matara, then print, “You belong to Southern province”
  - If city=Colombo, then print, “You belong to Western province”
  - Otherwise print “You are not belong to Southern or Western province”

Answer

```
<?php
	$city="Matara";
	switch ($city) {
		case "Colombo":
		echo "You belong to Western province";
		break;
		case "Matara":
		echo "You belong to Southern province";
		break;
		default:
		echo "You are not belong to either Western or Southern province";
	}
?>
```

Exercise – Home work

- A student got 55 marks for the Science subject. Students are given the comment according to following table. Write a PHP program to print the Comment as per the table

Marks Range	Comments
100>Marks>75	Excellent
75>Marks>50	Good
50>Marks>40	Pass
Non of the above	Improve your knowledge

Summary

- Features of PHP
- Basic syntax of PHP
- How to run PHP scripts
- PHP variables
- PHP Data Types
- Basic conditional statements in PHP