

# CSC 214

Internet Technologies and Web Design

Dr Ali Joan Beri Wacka

# What is PHP?

- PHP is an acronym for "PHP: Hypertext Preprocessor"
- PHP is a widely-used, open source scripting language
- PHP scripts are executed on the server
- PHP is free to download and use
- **PHP is an amazing and popular language!**
- It is powerful enough to be at the core of the biggest blogging system on the web (WordPress) and websites development!
- It is deep enough to run the largest social network (Facebook)!
- It is also easy enough to be a beginner's first server side language!

# What is a PHP File?

- What is a 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, delete, and close files on the server
- PHP can collect form data
- PHP can send and receive cookies
- PHP can add, delete, modify data in your database
- PHP can be used to control user-access
- PHP can encrypt data

With PHP you are not limited to output HTML. You can output images, PDF files, and even Flash movies. You can also output any text, such as XHTML and XML.

# Why PHP and what is needed to run PHP

## Why

- PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- PHP is compatible with almost all servers used today (Apache, IIS, etc.)
- PHP supports a wide range of databases
- PHP is free. Download it from the official PHP resource: [www.php.net](http://www.php.net)
- PHP is easy to learn and runs efficiently on the server side

## Needed

- To start using PHP, you can:
- Find a web host with PHP and MySQL support
- Install a web server on your own PC, and then install PHP and MySQL
- Use a Web Host With PHP Support
  - If your server has activated support for PHP you do not need to do anything.
  - Just create some .php files, place them in your web directory, and the server will automatically parse them for you.
  - You do not need to compile anything or install any extra tools.
  - Because PHP is free, most web hosts offer PHP support.

# Set Up PHP on Your Own PC/laptop

- However, if your server does not support PHP, you must:
- install a web server
- install PHP
- install a database, such as MySQL
- The official PHP website (PHP.net) has installation instructions for PHP: <http://php.net/manual/en/install.php>
- A PHP script is executed on the server, and the plain HTML result is sent back to the browser.

# Basic PHP Syntax

- A PHP script can be placed anywhere in the document.
- A PHP script starts with `<?php` and ends with `?>`:

```
<?php
// PHP code goes here
?>
```

- The default file extension for PHP files is ".php".
- A PHP file normally contains HTML tags, and some PHP scripting code.
- Below, we have an example of a simple PHP file, with a PHP script that uses a built-in PHP function "echo" to output the text "Hello World!" on a web page:

## Example

- ```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello World!";
?>

</body>
</html>
```

## Comments in PHP

- A comment 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 looking at the code.
- Comments can be used to:
- Let others understand what you are doing
- Remind yourself of what you did - Most programmers have experienced coming back to their own work a year or two later and having to re-figure out what they did. Comments can remind you of what you were thinking when you wrote the code
- PHP supports several ways of commenting:

## Example

- ```
<!DOCTYPE html>
<html>
<body>

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

# This is also a single-line comment

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

// You can also use comments to leave out parts of a code line
$x = 5 /* + 15 */ + 5;
echo $x;
?>

</body>
</html>
```

### PHP Case Sensitivity

- In PHP, all keywords (e.g. if, else, while, echo, etc.), classes, functions, and user-defined functions are NOT case-sensitive. However; all variable names are case-sensitive.
- In the example below, all three echo statements below are legal (and equal):

## Example

```
<!DOCTYPE html>
<html>
<body>

<?php
ECHO "Hello World!<br>";
echo "Hello World!<br>";
EcHo "Hello World!<br>";
?>

</body>
</html>
```

In the example below, only the first statement will display the value of the \$color variable (this is because \$color, \$COLOR, and \$coLOR are treated as three different variables):

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
$color = "red";
```

```
echo "My car is " . $color . "<br>";
```

```
echo "My house is " . $COLOR . "<br>";
```

```
echo "My boat is " . $coLOR . "<br>";
```

```
?>
```

```
</body>
```

```
</html>
```



# PHP 5 Variables

## Creating (Declaring) PHP Variables

Variables are "containers" for storing information.

In PHP, a variable starts with the \$ sign, followed by the name of the variable:

Example

```
<?php
$txt = "Hello world!";
$x = 5;
$y = 10.5;
?>
```

- After the execution of the statements above, the variable **\$txt** will hold the value **Hello world!**, the variable **\$x** will hold the value **5**, and the variable **\$y** will hold the value **10.5**.
- **Note:** When you assign a text value to a variable, put quotes around the value.
- **Note:** Unlike other programming languages, PHP has no command for declaring a variable. It is created the moment you first assign a value to it.

Think of variables as containers for storing data.

# PHP Variables

- A variable can have a short name (like x and y) or a more descriptive name (age, carname, total\_volume).
- 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)
- Remember that PHP variable names are case-sensitive!

# Output Variables

The PHP echo statement is often used to output data to the screen.

The following example will show how to output text and a variable:

Example

```
<?php
$txt = "ubuea.net";
echo "I love $txt!";
?>
```

The following example will produce the same output as the example above:

```
<?php
$txt = "ubuea.net";
echo "I love " . $txt . "!";
?>
```

The following example will output the sum of two variables:

```
<?php
$x = 5;
$y = 4;
echo $x + $y;
?>
```

# PHP Variables Scope

In PHP, variables can be declared anywhere in the script.

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>";
?>
```

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

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

# PHP The global Keyword

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

To do this, use the global keyword before the variables (inside the function):

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

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

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

- PHP also stores all global variables in an array called `$GLOBALS[index]`. The *index* holds the name of the variable. This array is also accessible from within functions and can be used to update global variables directly.
- The example above can be rewritten like this:

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

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

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

# PHP 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();
?>
```

- Then, each time the function is called, that variable will still have the information it contained from the last time the function was called.
- **Note:** The variable is still local to the function.

## PHP 5 echo and print Statements

- In PHP there are two basic ways to get output: echo and print.
- In this tutorial we use echo (and print) in almost every example. So, this chapter contains a little more info about those two output statements.

### **PHP echo and print Statements**

echo and print are more or less the same. They are both used to output data to the screen.

The differences are small: echo has no return value while print has a return value of 1 so it can be used in expressions. echo can take multiple parameters (although such usage is rare) while print can take one argument. echo is marginally faster than print.

# The PHP echo Statement

- The echo statement can be used with or without parentheses: echo or echo().
- Display Text
- The following example shows how to output text with the echo command (notice that the text can contain HTML markup):

```
<?php
echo "<h2>PHP is Fun!</h2>";
echo "Hello world!<br>";
echo "I'm about to learn PHP!<br>";
echo "This ", "string ", "was ", "made ", "with multiple parameters.";
?>
```



## Display Variables

The following example shows how to output text and variables with the echo statement:

- ```
<?php
$txt1 = "Learn PHP";
$txt2 = "ubuea.net";
$x = 5;
$y = 4;

echo "<h2>" . $txt1 . "</h2>";
echo "Study PHP at " . $txt2 . "<br>";
echo $x + $y;
?>
```