What is PHP?

- PHP stands Hypertext Preprocessor.
- PHP is a server-side scripting language.
- PHP scripts are executed on the server
- PHP supports many databases (MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, etc.)

What is a PHP File?

- PHP files can contain text, HTML,CSS and scripts
- PHP files are returned to the browser as plain HTML
- PHP files have a file extension of ".php", ".php3", or ".php7" etc
- C:\xampp\htdocs

PHP Syntax

A PHP script starts with <?php and ends with ?>:

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

PHP Case Sensitivity

In PHP, NO keywords (e.g. `if`, `else`, `while`, `echo`, etc.), classes, functions, and user-defined functions are case-sensitive.

Example

```
<!DOCTYPE html>
<html>
<body>
        <?php
        ECHO "Hello World!<br>";
        echo "Hello World!<br>";
        EcHo "Hello World!<br>";
        ?>
```

```
            </body>
            </html>
```

Note: However; all variable names are case-sensitive!

PHP echo and print Statements

With PHP, there are two basic ways to get output: echo and print. 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().

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.";
?>
```

Output of Example

```
PHP is Fun!

Hello world!
I'm about to learn PHP!
This string was made with multiple parameters.
```

The PHP print Statement

The print statement can be used with or without parentheses: print or print().

The following example shows how to output text with the print command (notice that the text can contain HTML markup):

```
<?php
print "<h2>PHP is Fun!</h2>";
print "Hello world!<br>";
print "I'm about to learn PHP!";
?>
```

Output of Example

```
PHP is Fun!
```

```
Hello world!
I'm about to learn PHP!
```

Comments in PHP

Examples:

Syntax for single-line comments:

```
<!DOCTYPE html>
<html>
<body>
    <?php
    // This is a single-line comment

    # This is also a single-line comment
    ?>
</body>
</html>
```

Syntax for multiple-line comments:

```
<!DOCTYPE html>
<html>
<body>
    <?php
        /*
        This is a multiple-lines comment block that
        spans over multiple lines

        */
</body>
</html>
```

Using comments to leave out parts of the code:

```
<!DOCTYPE html>
<html>
<body>
    <?php
        // You can also use comments to leave out parts of a code line
        $x = 5 /* + 15 */ + 5;
        echo $x;
    ?>
</body>
</html>
```

PHP Variables

Creating (Declaring) PHP Variables

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;
?>
```

Note: When you assign a text value to a variable, put quotes around the value.

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!

PHP is a Loosely Typed Language

PHP automatically associates a data type to the variable, depending on its value. Since the data types are not set in a strict sense, you can do things like adding a string to an integer without causing an error.

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:

1. local
2. global
3. static.

i. Global and Local Scope

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

Example:

Variable with global scope:

```
<?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:

Example:

Variable with local scope:

```
<?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>";

?>
```

You can have local variables with the same name in different functions, because local variables are only recognized by the function in which they are declared.

ii. 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):

Example:

```
<?php

        $x = 5;

        $y = 10;

        function myTest() {
          global $x, $y;
          $y = $x + $y;
        }
        myTest();
        echo $y; // outputs 15
    ?>
```

iii. 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:

Example:

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

1. PHP String Functions

- strlen() - Return the Length of a String
- str_word_count() - Count Words in a String
- strrev() - Reverse a String
- strpos() - Search For a Text Within a String
  - The PHP strpos() function searches for a specific text within a string. If a match is found, the function returns the character position of the first match. If no match is found, it will return FALSE
- str_replace() - Replace Text Within a String