**What You Should Already Know**

**Before you continue you should have a basic understanding of the following**:

* HTML
* CSS
* JavaScript

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)!  
It is deep enough to run large social networks!  
It is also easy enough to be a beginner's first server side language!

What is a PHP File?

* PHP files can contain text, HTML, CSS, JavaScript, and PHP code
* PHP code is 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

Why PHP?

* 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

What's new in PHP -7

* PHP 7 is much faster than the previous popular stable release (PHP 5.6)
* PHP 7 has improved Error Handling
* PHP 7 supports stricter Type Declarations for function arguments
* PHP 7 supports new operators (like the spaceship operator: <=>)

**Install PHP**

However, if your server does not support PHP, you must need following requirements:

* install a web server (xampp, wamp)
* install a database, such as MySQL

## 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 (**index.php**), with a PHP script that uses a built-in PHP function "echo" to output the text "Hello World!" on a web page:

**<?php**

**echo**  **"** Hello World! **"**;

**?>**

**Note:** PHP statements end with a semicolon (;)

## PHP Case Sensitivity

In PHP, keywords (e.g. if, else, while, echo, etc.), classes, functions, and user-defined functions are not case-sensitive.

<?php

ECHO "Hello World!<br>";

echo "Hello World!<br>";

EcHo "Hello World!<br>";

?>

But However all variable names are **case-sensitive**!

Look at 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

$COLOR is not same as $color:

**<?php**

$color = "red";

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

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

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

**?>**

**Comments in PHP**

A comment in PHP code is a line that is not executed as a 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 your code
* 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
* Leave out some parts of your code

// This is a single-line comment

# This is also a single-line comment

/\* This is a multi-line comment \*/

**PHP Variables ?**

Variables are "containers" for storing information.

Creating (Declaring) PHP Variables

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

*Example of variable*

**$x = 5;**

**$y = "John";**

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

## Output via Variables

$txt = "php";

echo "I love $txt!";

output: I love php

## Variable Types

***PHP supports the following data types:***

* String - “im a string”
* Integer - 10
* Float - 10.2
* Boolean - true/false
* Array - [2, 3, 56]
* Object -
* NULL - null
* Resource - files & others

## Get the Type

To get the data type of a variable, use the var\_dump() function.

**var\_dump(5);**

**var\_dump("John");**

**var\_dump(3.14);**

**var\_dump(true);**

**var\_dump([2, 3, 56]);**

**var\_dump(NULL);**

we can get the output after apply this

## PHP echo and print Statements

We can get output: with  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 more faster than print.

## The PHP echo Statement

**Display Text by 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.";

?>

**Display Variables**

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

$txt1 = " with confident";

$x = 5;

$y = 4;

echo "Study PHP " . $txt1 . "<br>";

echo $x + $y;

**output:**

Study PHP with confident

9

**Display Variables**

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

$txt2 = "some text";

$x = 5;

$y = 4;

print "Study PHP at " . $txt2 . "<br>";

print $x + $y;

Output:

Study PHP at some text

9

# PHP Data Types

Variables can store data of different types, and different data types can do different things.

PHP supports the following data types:

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

PHP String

A string is a sequence of characters, like "Hello world!".

A string can be any text inside quotes. You can use **single or double quotes**:

Example

$x = "Hello world!";

$y = 'Hello world!';

var\_dump($x);

echo "<br>";

var\_dump($y);

output:

string(12) "Hello world!"  
string(12) "Hello world!"

PHP Integer

An integer data type is a **non-decimal number** between -2,147,483,648 and 2,147,483,647.

Rules for integers:

* An integer must have at least one digit
* An integer must not have a decimal point
* An integer can be either positive or negative
* Integers can be specified in: decimal (base 10), hexadecimal (base 16), octal (base 8), or binary (base 2) notation

Example:

$x = 5985;

## PHP Float

A float (floating point number) is a number with a decimal point or a number in exponential form.

In the following example $x is a float. The PHP var\_dump() function returns the data type and value:

$x = 10.365;

## PHP Boolean

A Boolean represents two possible states: TRUE or FALSE.

$x = true;

$y = false;

## PHP Array

An array stores multiple values in one single variable.

In the following example $cars is an array. The PHP var\_dump() function returns the data type and value:

$cars = array ("Volvo","BMW","Toyota");

# PHP Strings

Strings in PHP are surrounded by either double quotation marks, or single quotation marks. A string is a sequence of characters, like "Hello world!"

echo "Hello";

echo 'Hello';

Note:  There is a big different between double quotes and single quotes in PHP.

Double quotes process special characters, single quotes does not.

## Double or Single Quotes?

You can use double or single quotes, but you should be aware of the differences between the two.

Double quoted strings perform action on special characters.

E.g. when there is a variable in the string, it returns the value of the variable:

*Example*

Double quoted string literals perform operations for special characters:

$x = "John";

echo "Hello $x";

output: Hello John

Single quoted strings does not perform such actions, it returns the string like it was written, with the variable name:

$x = "John";

echo 'Hello $x';

output: Hello $x

## String Length

The PHP strlen() function returns the length of a string.

## <?php

## echo strlen ("Hello world!");

## ?>

## Output: 12

## Word Count

The PHP str\_word\_count() function counts the number of words in a string.

<?php

echo str\_word\_count ("Hello world!");

?>

# PHP - Modify Strings

**TEXT UPPERCASE**

The strtoupper() function returns the string in upper case:

<?php

$x = "Hello World!";

echo strtoupper($x);

?>

**TEXT lowercase**

The strtolower() function returns the string in lower case:

<?php

$x = "Hello World!";

echo strtolower($x);

?>

**Replace String**

The PHP str\_replace() function replaces some characters with some other characters in a string.

<?php

$x = "Hello World!";

echo str\_replace("World", "Dolly", $x);

?>

Output: Hello Dolly!

## Reverse a String

The PHP strrev() function reverses a string.

<?php

$x = "Hello World!";

echo strrev($x);

?>

Output: !dlroW olleH

## Remove Whitespace

Whitespace is the space before and/or after the actual text, and very often you want to remove this space.

<?php

$x = " Hello World! ";

echo trim($x);

?>

## 