# Chapter One

## Server-Side Scripting Basics

Introduction

In this chapter will cover about basics of server-side scripting:

* Introduction
* Syntax,
* Variable,
* Constant,
* Send data to the browser,
* Data types,
* Date,
* Operators,
* Array,
* Conditionals,
* Loops,
* Data manipulation and validation,
* Function

## Introduction

Web site can be categorized into two; static and dynamic. This is based on the content delivered to clients. If the content of a web site is constant at any time this web site is called as static web site. On the other hand, if the content of a web site update through time this web site can be called as dynamic web site. This is based on the business requirement.

To build or develop static web site HTML, CSS and JavaScript are enough as developing technologies. Where as to develop dynamic web site Database and server scripting technologies are required additionally. For this matter you are learning Interne programming I and II.

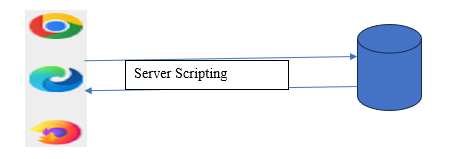


Figure 1 Server Scripting in Web site

As shown in figure 1 the main purpose of server scripting language is making a web site dynamic by sending and receiving data from client to the database and vice versa.

There are a lots of server scripting languages like PHP, Node, ASP, JSP, Python, Golang etc. Even though, for this course PHP is selected.

PHP is created by Rasmus Lerdorf in 1995 by using Perl programming and he gave a name Personal Home Page Tools. Later with his team developed the second version by using C programming language and called Hypertext Preprocessor but the acronym still the same PHP. Now in 2024 the latest version of PHP is 8.3. It used to develop dynamic web site even content management software such as WordPress, Joomla, Drupal those will see in last chapter of this course.

## Features of PHP

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

## Syntax

The grammatical rule to write php code is open a tag and write correct syntax code as aa boddy of code and close is properly with php close tag then save it as .php file as file extension. There four alternatives to write php code and described as follows.

1. Standard

<?php //start tag

/\*

Body

\*/

?>// end tag

1. Short hand

<? //start tag

/\*

Body

\*/

?>// end tag

1. ASP

<% //start tag

/\*

Body

\*/

%>// end tag

1. Script

<script language=”php”>

/\*

Body

\*/

</script>

## Variable

A variable in PHP is a name of memory location that holds data. In other word it is a temporary storage that is used to store data temporarily. In PHP, a variable is declared using $ sign followed by variable name.

Syntax of declaring a variable in PHP is given below:

**$variablename=value;**

In PHP, a variable does not need to be declared before adding a value to it. The variable is declared automatically when you use it. PHP automatically converts the variable to the correct data type, depending on its value. Variable is mutable that means the value can be changed.

**Naming Rules for Variables**

* A variable name must start with a letter or an underscore "\_"
* A variable name can only contain alpha-numeric characters and underscores (a-z, A-Z and )
* A variable name should not contain spaces.
* If a variable name is more than one word, it should be separated with an underscore ($my\_string)or with capitalization ($myString).

**Example**

<?php

$name = “Yared Henok”;

$age = 25;

$department = “Information Technology”;

$height = 1.75;

$gender = “M”;

?>

## Variable scope

The visibility of variable or the availability of a variable is called scope. It can be local or global. Local variables are visible in a block of code only whereas global variable is through the php file. Will see in detail in upcoming topics.

## Constant

Unlike variable the value of constant is immutable that means can not be changed. There are two ways of declaring constants.

1. Using define method
   1. define (“PI”, 3.14);
2. Using const key word
   1. const PI = 3.14

the convention of naming a constant is by all upper case like the above examples.

## Send data to the browser

Static data or data from database can be send to the browser through PHP. There are two main was such as print and echo.

<?php

echo “Hello World \n”;

echo 123;

print (“Hello World”);

print (123);

?>

Then save it as .php and open by web browser like Chrome, Fire Fox or MS Edge

PHP can be embedded in any HTML tag. If the above code embeds in HTML, it is better to embed in body tag.

<!DOCTYPE html>

<html>

<head>

<title>Hello Worl </title>

</head>

<body>

<?php

echo “Hello World \n”;

echo 123;

print (“Hello World”);

print (123);

?>

</body>

</html>

## Data types

In PHP the type of variable can be identified by itself in variable initialization no need to specify the type of a variable. Even though, when necessary, it is possible to specify the type of data as integer, float, string, Boolean, array, null.

Data types in php:

1. Integer: $age = 30;
2. Float : $height = 1.75;
3. Boolean: $active = true;
4. String : $fname = “Yohannes”;
5. Null: $grade = null;
6. Resource : $conn = mysqli\_connect(“localhost”, “root”, “”);
7. Array : $books = [“PHP”, “Java”, “Python”];
8. Object: $abebe = new Employee();

All these data types are called scalar data types;

NB. Quotation is required for string literal. It can be single quotation or double quotation.

## Data type casting

When converting one data type into other data type is called casting.

Example: $height = (int) 1.75;

## Date

Date is the crucial part of dynamic web development like reporting, log control. So, knowing and operating it can help a lot. There is built in function data that and generate date in several formats.

Example

$rdate = new date(‘Y/M/D’);

$date\_time = new date(‘h/m/d H:s’);

## Operators

In 1.3, 1.4 and 1.6 discussed about data. Then after having data there will be operation or manipulation of data. The symbols used to manipulate data are called operators. There are several operators based on operations and data types:

## Arithmetic operators

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Operation | Symbol | example | result |
| 1 | Addition | + | $x = 3; $y = 4;  echo $x + $y | 7 |
| 2 | Subtraction | - | $x = 3; $y = 4;  echo $x - $y | -1 |
| 3 | Multiplication | \* | $x = 3; $y = 4;  echo $x \* $y | 12 |
| 4 | Division | / | $x = 3; $y = 4;  echo $x / $y | 0.75 |
| 5 | Module | % | $x = 3; $y = 4;  echo $x % $y | 3 |

## Assignment operator

An assignment operator is (=) used in variable initialization. It used to assign value or literal in a computer memory or a variable or a constant.

Example: $i = 9;

Compound assignments: combining arithmetic operators with assignment can simplify operations like $i += 8; is equal to $i = $i +8; So, in place of + it is possible to use other arithmetic operators.

## Comparison or relational operators

These operators are used to compare value or operands.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Operation | Symbol | example | result |
| 1 | Equal to | ===, == | $x = 3; $y = 4;  echo $x == $y | 0 or false |
| 2 | No equal to | != | $x = 3; $y = 4;  echo $x != $y | 1 or true |
| 3 | Greater than | > | $x = 3; $y = 4;  echo $x > $y | 0 or false |
| 4 | Greater than or equal to | >= | $x = 3; $y = 4;  echo $x >= $y | 0 or false |
| 5 | Less than | < | $x = 3; $y = 4;  echo $x < $y | 1 or true |
| 6 | Less than or equal to | <= | $x = 3; $y = 4;  echo $x < = $y | 1 or true |
| 6 | Not equal to | <> | $x = 3; $y = 4;  echo $x <> $y | 0 or false |

## Logical operators

These operators are used to validate conditions.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Operation | Symbol | example | result |
| 1 | and | && | $x = 3; $y = 4; $z = 5;  echo $x == $y && $x < $z | 0 or false |
| 2 | OR | || | $x = 3; $y = 4; $z = 5;  echo $x == $y && $x < $z | 1 or true |
| 3 | Not | ! | $x = 3; $y = 4;  echo !($x > $y) | 1 or true |

## Bitwise operators

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Operation | Symbol | example | result |
| 1 | and | & | $x = 3; $y = 4;  echo $x & $y | 0 or false |
| 2 | OR | | | $x = 3; $y = 4; $z = 5;  echo $x == $y && $x < $z | 1 or true |
| 3 | Shift right | >> | $x = 3; $y = 4;  echo !($x > $y) | 1 or true |
| 4 | Shift left | << |  |  |

## Other operators

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Operation | Symbol | purpose | Example |
| 1 | Parenthesis | () | For precedence, function definition, function call | 0 or false |
| 2 | Square bracket | [] | Array initialization | $books = [“PHP”,”Java”]; |
| 3 | Curly brace | {} | Block the code | Function |
| 4 | Dot(period) | . | Concatenation | $fname.” “.$lname |
| 5 | Semicolon | ; | End of statement | $x = 10; |
| 6 | Coma | , | Separate elements | $books = [“PHP”,”Java”]; |