Introduction to PHP & Features

PHP is a server scripting language, and a powerful tool for making dynamic and interactive
Web pages.
PHP is a widely-used, free, and efficient alternative to competitors such as Microsoft's ASP.

Example

```
<html>
<body>
<?php
echo "My first PHP script!";
?>
</body>
</html>
```

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

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"

Basic PHP Syntax

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

Comments

Example

```
<html>
<body>
<?php
// This is a single-line comment comment
 /*
This is a multiple-lines comment
```

```
block that spans over multiple
lines
*/
?>
</body>
</html>
```

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

```
<html>
<body>
<?php
$x = "Hello world!";
$y = 'Hello world!';
echo $x;
echo "<br>";
echo $y;
?>
</body>
</html>
```

OUTPUT:

Hello world!
Hello world!

String Functions

• Get The Length of a String

• The PHP strlen() function returns the length of a string.
• The example below returns the length of the string "Hello world!":

Example

```
<html>
<body>
<?php
echo strlen("Hello world!"); ?>
</body>
</html>
```

OUTPUT:

12

Count The Number of Words in a String

The PHP str_word_count() function counts the number of words in a string:

Example

```
<html>
<body>
<?php
echo str_word_count("Hello world!");
?>
</body>
</html>
```

OUPUT:

2

Reverse a String

• The PHP strrev() function reverses a string:

Example

```
<html>
<body>
<?php
echo strrev("Hello world!");
?>
</body>
</html>
```

OUTPUT:

!dlrow olleH

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 three formats: decimal (10-based), hexadecimal (16-
based - prefixed with 0x) or octal (8-based - prefixed with 0)
• In the following example $x is an integer. The PHP var_dump() function returns the data type and value:

Example

```
<html>
<body>
<?php
$x = 5985;
var_dump($x);
?>
</body>
</html>
```

OUTPUT:

int(5985)

PHP Boolean

A Boolean represents two possible states: TRUE or FALSE.

```
$x = true;
$y = false;
```

Booleans are often used in conditional testing

PHP Array

• An array stores multiple values in one single variable:
• An array is a special variable, which can hold more than one value at a time.
• If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this:

```
$cars1 = "Volvo";
$cars2 = "BMW";
$cars3 = "Toyota";
```

• However, what if you want to loop through the cars and find a specific one? And what if you had not 3 cars, but 300?
• The solution is to create an array!
• An array can hold many values under a single name, and you can access the

values by referring to an index number.

Example

```
<html>
<body>
<?php
$cars = array("Volvo", "BMW", "Toyota");
echo "I like " . $cars[0] . ", " . $cars[1] . " and " . $cars[2] . ".";
?>
</body>
</html>
```

OUTPUT:

I like Volvo, BMW and Toyota.

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

```
<html>
<body>
<?php
$txt = "Hello world!";
$x = 5;
$y = 10.5;
echo $txt;
echo "<br>";
echo $x;
echo "<br>";
echo $y;
?>
</body>
</html>
```

PHP Operators

Operators are used to perform operations on variables and values.

PHP divides the operators in the following groups:

• Arithmetic operators

• Assignment operators

• Comparison operators
• Increment/Decrement operators
• Logical operators
• String operators
• Array operators

Arithmetic Operators:
The arithmetic operators are used to perform simple mathematical operations like addition, subtraction, multiplication, etc. Below is the list of arithmetic operators along with their syntax and operations in PHP.

Operator	Name	Syntax	Operation
+	Addition	$x + $y	Sum the operands
–	Subtraction	$x – $y	Differences the operands
*	Multiplication	$x * $y	Product of the operands
/	Division	$x / $y	The quotient of the operands
**	Exponentiation	$x ** $y	$x raised to the power $y
%	Modulus	$x % $y	The remainder of the operands

Logical or Relational Operators:
These are basically used to operate with conditional statements and expressions. Conditional statements are based on conditions. Also, a condition can either be met or cannot be met so the result of a conditional statement can either be true or false. Here are the logical operators along with their syntax and operations in PHP.

Operator	Name	Syntax	Operation
and	Logical AND	$x and $y	True if both the operands are true else false
or	Logical OR	$x or $y	True if either of the operands is true else false
xor	Logical XOR	$x xor $y	True if either of the operands is true and false if both are true
&&	Logical AND	$x && $y	True if both the operands are true else false
||	Logical OR	$x || $y	True if either of the operands is true else false
!	Logical NOT	!$x	True if $x is false

Comparison Operators: These operators are used to compare two elements and outputs the result in boolean form. Here are the comparison operators along with their syntax and operations in PHP.

Operator	Name	Syntax	Operation
==	Equal To	$x == $y	Returns True if both the operands are equal
!=	Not Equal To	$x != $y	Returns True if both the operands are not equal
<>	Not Equal To	$x <> $y	Returns True if both the operands are unequal
===	Identical	$x === $y	Returns True if both the operands are equal and are of the same type

Operator	Name	Syntax	Operation
!==	Not Identical	$x == $y	Returns True if both the operands are unequal and are of different types
<	Less Than	$x < $y	Returns True if $x is less than $y
>	Greater Than	$x > $y	Returns True if $x is greater than $y
<=	Less Than or Equal To	$x <= $y	Returns True if $x is less than or equal to $y
>=	Greater Than or Equal To	$x >= $y	Returns True if $x is greater than or equal to $y

Conditional or Ternary Operators:
These operators are used to compare two values and take either of the results simultaneously, depending on whether the outcome is TRUE or FALSE. These are also used as a shorthand notation for *if...else* statement that we will read in the article on decision making.
Syntax:
$var = (condition)? value1 : value2;

Assignment Operators: These operators are used to assign values to different variables, with or without mid-operations. Here are the assignment operators along with their syntax and operations, that PHP provides for the operations.

Operator	Name	Syntax	Operation
=	Assign	$x = $y	Operand on the left obtains the value of the operand on the right
+=	Add then Assign	$x += $y	Simple Addition same as $x = $x + $y

Operator	Name	Syntax	Operation
-=	Subtract then Assign	$x -= $y	Simple subtraction same as $x = $x – $y
*=	Multiply then Assign	$x *= $y	Simple product same as $x = $x * $y
/=	Divide then Assign (quotient)	$x /= $y	Simple division same as $x = $x / $y
%=	Divide then Assign (remainder)	$x %= $y	Simple division same as $x = $x % $y

Increment/Decrement Operators: These are called the unary operators as they work on single operands. These are used to increment or decrement values.

Operator	Name	Syntax	Operation
++	Pre-Increment	++$x	First increments $x by one, then return $x
—	Pre-Decrement	–$x	First decrements $x by one, then return $x
++	Post-Increment	$x++	First returns $x, then increment it by one
—	Post-Decrement	$x–	First returns $x, then decrement it by one

PHP String Operators

• PHP has two operators that are specially designed for strings.

Operator Name Example Result

. Concatenation $txt1 . $txt2 Concatenation of $txt1 and $txt2

.= Concatenation

assignment

$txt1 .= $txt2	Appends $txt2 to $txt1