PHP

PHP is similar to JavaScript that allows a you -a web developer, to embed some programs into the HTML code of a web page. These programs can give you control to the appearance of web pages in the browser window than what can HTML alone can provide. But the difference between JavaScript and PHP is that the stage of loading the web page upon each embedded program that are executed. JavaScript is a client-side language, which is client-side languages are read and executed by web browser after downloading the web page from the server. While PHP is one of some server-side languages, that these languages are run by the web server, before sending the web page to the browser.

Example of a PHP code:

**<?php echo date(‘ l, F jS Y. ’); ?>**

**<?php** marks as the start of a PHP script and **?>** marks its end.

The web server is asked to interpret everything between the two delimiters and convert it to HTML code before it goes the web page to the requesting browser.

The program appears to be: **Sunday, April 1st 2012.**

PHP Interpreter

Local PHP Interpreter- It is a PHP engine that is already installed on computer

Remote PHP Interpreters- It can be installed on a remote host or in a virtual environment set up in a Vagrant instance.

Basic Syntax and Statements

PHP script consists of a commands or statements. Each PHP statements are always terminated by a semicolon (;).

Example: **echo ‘This is a quiz!’;**

An echo statement, is used to insert the text in the HTML code at the position where the PHP script was.

**echo ‘This is a quiz’** and **echo date(‘l, F jS Y. ‘);**

Quotes are used to mark the beginning and the end of strings while parentheses are used in two purposes, first purpose is to indicate that the date is a function and second purpose is to mark the beginning and end of arguments in order to tell what the function will do.

PHP Variables, Operators and Comments

All variable names in PHP begin with a dollar sign, a single variable contain any data type, number or string. If the variable has an existing value -a number, then the variable was assigned a new value which is a string, it now contains a string value.

The operators of PHP may be used to assign values to variables, other operators can be used to perform mathematical operations on values, there are operators for comparing values and many more. For assigning values to variables, *assignment operator* is being use which is the equal sign ‘=’. There are also ’+=’(Addition), ‘ -= ’(Subtraction), ‘\*=’(Multiplication), ‘/=’(Division), ‘%=’(Modulus) used with numeric values.

Example: <?php

$money=35;

$money +=10;

echo $money; //outputs 45

?>

To perform mathematical operations, *arithmetic operators* are being used like ‘+’ addition operator for adding, ‘ – ‘ subtraction operator for subtracting, ‘ \* ’multiplication operator for multiplying and ‘ / ’ division operator for dividing.

There is also string operators for string values which are *concatenation* ( . ) and *concatenation* *assignment* (.=).Concatenation sticks the string value together and concatenation assignment assigning one variable sticks to another variable.

Example:

<?php

$greet = ‘Hi’. ‘there!’;

echo $greet; //outputs “Hi there!”

$greet = ‘Hello’;

$name = ‘Kevin’;

$greet .= $name;

echo $greet; //outputs “Hello Kevin”

?>

Concatenation can be also applied in echo statement:

<?php

$greet = ‘Hello’;

echo $greet . ‘Mikaela’; // outputs “Hello Mikaela”

?>

Comments are used to indicate what is the code for. It is an explanatory text for every code. Comments begin with ‘//’ -for single comments and for block comments, it begins with ‘/\*’ and end with ‘\*/’. Comments between two delimiters are ignored by PHP interpreters.

There are Logical operators, Comparison operators, Array operators and

Increment and Decrement operators.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Logical operators | | | | | | |
| and, && | | $a and $b  $a && $b | | | True if both variables are true. | |
| or, || | | $f and $g  $f || $g | | | True if either one variable is true. | |
| xor | | $r xor $k | | | True if either one variable is true, but not both. | |
| ! | | !$a | | | False if the variable is true. | |
| Comparison operators | | | |
| == | Equal | | |
| === | Identical | | |
| != , <> | Not equal | | |
| !== | Not identical | | |
| > | Greater than | | |
| < | Less than | | |
| >= | Greater than or equal | | |
| <= | Less than or equal | | |
| Increment or Decrement Operators | | | | | |
| ++$y | | | Pre-increment | | |
| $y++ | | | Post-increment | | |
| --$x | | | Pre-decrement | | |
| $x-- | | | Post-decrement | | |

|  |  |
| --- | --- |
| Array Operators | |
| + | Union |
| == | Equality |
| === | Identity |
| != | Inequality |
| <> | Inequality |
| !== | Non-identity |

PHP Arrays

Array is a variable that contains multiple values. To create array in PHP is to use array command.

Example: $numbers = array (‘one’, 2, 3);

Knowing the index of each value, you can retrieve the value stored in an array. Each element has its index. The string value ‘one’ is first element and its index is zero.

echo $numbers[0]; //outputs ‘one’

An existing value of an array can be change by indicating the index.

$numbers[0] = 1;

And also, can add a new element by adding an index;

$numbers [3]= 4;

Constants

Constant is a name identifier for simple value, constant is case-sensitive and it must be in uppercase. Constants does not begin with dollar sign. Once it is already defined it cannot be redefine unlike the variable. In indicating constants, it can be done using define() function or using const keyword.

In defining constants using define() function, it will be defined to an arbitrary expression. While using const keyword in defining constants, it will define as a scalar expression.

Examples: Defining Constants using define() function

<?php

define(“NICE”, “Lucky day!”);

echo NICE; //outputs “Lucky day!”

?>

Defining Constants using const keyword

<?php

const NICE = ‘Lucky day!’;

echo NICE;

?>

Functions

User -defined functions

The declaration starts with ‘function’. In naming a function, it begins with letter or underscore but not number. Creating a function, an opening curly brace will begin the function code and closing curly brace to end the function.

Example:

<?php

function printMessage(){

echo “This is a function.”;

}

printMessage(); //outputs “This is a function.”

?>

Function Arguments

The information in function can be past by arguments. Arguments are just like variables and it also starts with dollar sign. Arguments are inside of the parentheses in the function, which is after the function name. The arguments can be multiple, the arguments must be comma separated.

Example:

<?php

function nameOfAnimals($animals, $kind){

echo “$animals is an animal. <br>”;

}

nameOfAnimals(“Rabbit”, “mammal”);

nameOfAnimals(“Bear”, “mammal”);

nameOfAnimals(“Snake”, “reptile”);

?>

Returning Values

For the function to return a value, use return statement:

<?php

function divide($dividend, $divisor){

$quotient = $dividend /$divisor;

return $quotient;

}

echo "10 / 5= " . divide(10, 5) . "<br>";

echo "20 / 2 = " . divide(20, 2) ;

?>

Variable functions

Variable functions are which the variable has parentheses attach to the variable name. Variable functions won’t work in echo, print, unset(), isse(), empty().

<?php

function callIt($thing){

echo $thing;

}

$func = ’callIt’; //look for callIt function

$func('phone'); //execute the function

?>

Class and Objects

In creating class, class keyword must be included followed by a class name and pair of curly braces. Inside the pair of curly braces are the definitions of properties, methods and constants which belongs to the class. A class may contain its own properties -variables, methods -functions, and constants.

Example:

<?php

class Demo{

public $statement = ‘This is just a demo.’;

public function $printStatement(){

echo $statement;

}

}

?>

Php is one of the languages that is Object Oriented. Which means objects can be created that contains variables and functions. In creating an object in the class, the new keyword must be use.

<?php

class Demo{

public $statement = ‘This is just a demo.’;

public function $printStatement(){

echo $statement;

}

}

$name = new Demo;

$name->printStatement(); /\*the ‘->’ is an object operator used to access methods and properties of an object\*/

?>