PHP

PHP

Contact:8874588766,9598948810 Website: infomaxacademy.com

Notes By: Dinesh Maurya

Introduction

PHP stands for hypertext preprocessor. It is server side scripting language that allows your website to be truly dynamic. It flexibility and relatively learning curve make it one of the most popular scripting language around.

```
<?php

PHP code goes hear

?>
```

PHP Syntax:

PHP can be written within HTML code in a HTML page, just like JavaScript & other Scripting language. Html can also be written inside the PHP section of your page, which allows you to format text by keeping blocks of code together. PHP can also be written as standalone program with no html at all-

- PHP is denoted in the page with opening & closing tag as follows- <?php .. ?>
- PHP lines end with a semicolon (;).
- // used for single line comment
  /* used for multi-line comment */
- The PHP function 'echo' is used to send text to the browser.

```
<html>
<head>
<title>Demo </title>
</head>
<body>
<?php
echo"my first prograame";
?>
</body>
</html>
```

INTEGRATING HTML IN PHP-

Modify the above code as follows-

```
<?php
```

```
echo"<h1>my first<br>prograame</h1>";
?>
```

Note: while using HTML inside PHP

- Since the echo function may involve the use of double code, so whenever HTML uses double code, you can do one of the two things to avoid problem
    - Use single code inside your HTML
      Echo”Dinesh Maurya”;
    - Escape your HTML double code with the backslash as follows
      Echo”Dinesh Maurya”;
- You still have to end your sentence with a semicolon as well as close all codes at end of your echo statement.

Constant in PHP:

- A constant is place holder for a value that your reference within your code.
- Typically named with capital latters
- Constant name must begin with letter or underscore and can’t begin with a number.
- Names are also case sensitive.
- You define a value assigned to the constant with the PHP functions ***define ()***.
- Once you have defined a constant it can’t be changed or undefined.

```
<?php
define(“INSTITUTE”,”Infomax Computer Academy”);
echo “Allahabad Best Computer Training Institute is:”;
echo INSTITUTE;
?>
```

Variable in PHP:

- Variable are meant to be a changed at some point in your program.
- Variables do not need to be defining or declared and can simply assign when needed.
- Variables can store numeric or text value.
- Variable must be denoted by $ sign
- Variable names are case sensitive.
- First character of variable name must underscore or alphabet and can’t be number.

```
<?php
define("SubName","WebTechnology");
echo"Myfavrate Subject:";
echoSubName;
echo"<br>";
$marks=87;
echo"Marks obtain in WT:";
echo $marks;
?>
```

Destroying Variable:

```
<?php
$a="Indian";
echo"Before destroying $a";
echo"<br>";
unset($a);
echo"After destroying $a";
?>
```

Data type in PHP: PHP include 8 data types which are as follows

DATA TYPE	
Primitive Data type	**Compound Data Type**
1. Integer	1. object
2. floating point number	2. array
3.String	
4. Boolean	

In PHP it is not necessary to specify the data type of variable before using it. PHP automatically decides the data type according to the value assigned to the variable.

```
<?php
$testing = 5;
printgettype( $testing ); // integer
print "<br>";
$testing = "five";
printgettype( $testing ); // string
print("<br>");
```

```
$testing = 5.0;
printgettype( $testing ); // double
print("<br>");
$testing = true;
printgettype( $testing ); // boolean
print "<br>";
?>
```

String: Strings are in closed within single code or double code. String inside double code passed while string inside single code is not. It means when variable or special character are in closed in double code with string then the value of variable are printed with specified string. When variable name & special character are in closed with single code then the output is printed in the same as you typed them.

```
<?php
$a=”Indian”;
echo”<br>”;
echo’we are $a’;
ech0”<br>”;
echo”\&”;
echo”<br>”;
echo’\&’;
?>
```

Boolean: it represents a true or false value. All conditions return a true/false value depending on the condition being tested. Some of the statements condition being tested. Some of the statements always return a false value which is as follows.

- The keyword literal False.
- The integer 0
- The floating point 0.0
- The empty string
- The string ”0”
- The NULL value

Object: it allows you to store data as well as information process that data. To declare objects, first you must declare a class of object. Objects also allow you t create your own data types.

Array: it represents a variable that store the collection of related data elements. The position of an array element is either specified numerically or alphabetically.

NULL: it represents a specified data type that can have only one value 'NULL'.

Note: Determining variable data type by Using gettype() function.

```
<?php
$a="India";
echo gettype($a);
echo"<br>";
$a=99.5;
Echo gettype($a);
echo"<br>";
unset($a);
echogettype($a);
?>
```

Function to check data type of variable:

Function Name	
is_boolean()	is_string()
is_numeric()	is_null()
is_int()	is_array()
is_object()	

Operators in PHP:

Operators are special symbol that are used to perform some arithmetic and logical operation.

PHP provide the following types of operators:

- **Arithmetic Operators**
- **Assignment Operators**
- **Comparison Operators**
- **Increment/ Decrement Operators**
- **Logical Operators**

- **String Operators**
- **Array Operators**
- **Conditional assignment Operators**

Arithmetic Operators

Operator	Name	Example $x=5,$y=2
+	Addition	$x + $y=7
-	Subtraction	$x - $y=3
*	Multiplication	$x -*$y=10
/	Division	$x / $y=2.5
%	Modulus	$x %$y=1
**	Exponentiation	$x **$y=25

Assignment Operators

Operator	Name	Example $x=5	Result
=	Assignment	$x =5	$x=5
+=	Assignment with Addition	$x +=5	$x=10
-=	Assignment with Subtraction	$x -=5	$x=0
*=	Assignment with Multiplication	$x *=5	$x=25
/=	Assignment with Division	$x /=5	$x=1
%=	Assignment with Modules	$x %=3	$x=2

Comparison Operators

Operator	Name	Example $x=5,$y=3	Result
==	Equal	$x ==$y	false
===	Identical (also check for data type)	$x ===5.0	false
!= or <>	Not equal	$x <>$y, $x!=$y	true
!==	Not identical	$x !==5.0	true
>	Greater than	$x >$y	true
<	Less than	$x<$y	false
>=	Greater than or equal to	$x>=$y	true
<=	Less than or equal to	$x<=$y	false

Logical Operators

Operator	Name	Example $x=5,$y=3
&& (and)	And	Return True if All expression is true, otherwise return false
|| (or)	Or	Return true if Any expression are true otherwise return false
!	Not Equal	It work as compliment i.e return true if expression is false.

Increment / Decrement Operators

increment operators are used to increment a variable's value by 1. decrement operators are used to decrement a variable's value by 1.

Operator	Name	Description
++a	Pre Increment	First increment the value of variable then assign value.
a++	Post Increment	First assign the value of variable then increment its value.
--a	Pre-decrement	First decrement the value of variable then assign value.
a--	Post -decrement	First assign the value of variable then decrement its value.

String Operators

Operator	Description	Example
.	**Concatenation**	**$a . $b**
.=	**Concatenation assignment**	**$a .= $b**

Conditional Assignment Operators (? :)

conditional assignment operators are used to set a value depending on conditions: (? :), It is also known as Ternary Operator.

Syntax:

exp1 ? exp2 : exp3;

or

$x = *expr1* ? *expr2* : *expr3*

Example: $a%2==0)? Echo “No is Even” : echo “No is Odd ”;

Example : $max=$a>$b?$a:$b;

Conditional Statement

when you write code, you want to perform different actions for different conditions. You can use conditional statements in your code to do this.

- **if statement** - executes some code if one condition is true
- **if...else statement** - executes some code if a condition is true and another code if that condition is false
- **if...else if...else statement** - executes different codes for more than two conditions
- **Nested if else statement**- when want to place condition inside another condition
- **switch statement** - selects one of many blocks of code to be executed

The if Statement

The if statement executes some code if one condition is true.

Syntax

```
if (condition) {
  code to be executed if condition is true;
}
```

The if...else Statement

The if...else statement executes some code if a condition is true and another code if that condition is false.

Syntax

```
if (condition) {
  code to be executed if condition is true;
} else {
  code to be executed if condition is false;
}
```

The if...else if...else Statement

The if...else if...else statement executes different codes for more than two conditions.

Syntax

```
if (condition) {
  code to be executed if this condition is true;
} elseif (condition) {
  code to be executed if first condition is false and this condition is true;
} else {
  code to be executed if all conditions are false;
}
```

The Nested if else Statement

The nested used to place condition inside another conditional statement.

Syntax

```
if (condition1) {
        if (condition2) {
                code to be executed if both condition1 & condition2 is true;
        } else {
                code to be executed if condition1 is true and condition2 is false;
        }
}

else{
                if (condition3) {
                code to be executed if condition1 is false and condition3 is true;
        } else {
                 code to be executed if both condition1 and condition2 is false;
        }
}
```

Switch Case Statement:

switch statement to **select one of many blocks of code to be executed**.

Syntax

```
switch (n) {
  case label1:
    code to be executed if n=label1;
    break;
```

```
  case label2:
    code to be executed if n=label2;
    break;
  case label3:
    code to be executed if n=label3;
    break;
    ...
  default:
    code to be executed if n is different from all labels;
}
```

ARRAY IN PHP

$array=array();//empty array declaration

Creating an array with elements-

```
<? php
$cars=array("i10","A-star","Scrpio","farari");
print_r($cars)
?>
```

Types of Array

There are 3 types of array

1) **Numeric Array:** Numeric array use integer values as there index numbers

2) **Associative Array** - An array with strings as index. This stores element values in association with key values rather than in a strict linear index order.

NOTE: Built-in array functions is given in function reference PHP Array Functions

Numeric Array
These arrays can store numbers, strings and any object but their index will be represented by numbers. By default, the array index starts from zero.

Example

```
<?php
/* First method to create array. */
$numbers = array( 1, 2, 3, 4, 5);
foreach( $numbers as $value )
{
        echo "Value is $value <br />";
}
/* Second method to create array. */
$numbers[0] = "one";
$numbers[1] = "two";
$numbers[2] = "three";
$numbers[3] = "four";
$numbers[4] = "five";
foreach( $numbers as $value )
{
        echo "Value is $value <br />";
}
?>
```

Associative Arrays

The associative arrays are very similar to numeric arrays in term of functionality but they are different in terms of their index. Associative array will have their index as string so that you can establish a strong association between key and values.

Example:

```
<?php
/* First method to associate create array. */
$salaries = array(
"mohammad" => 2000,
"qadir" => 1000,
"zara" => 500
);
echo "Salary of mohammad is ". $salaries['mohammad'] . "<br />";
echo "Salary of qadir is ". $salaries['qadir']. "<br />";
echo "Salary of zara is ". $salaries['zara']. "<br />";
/* Second method to create array. */
$salaries['mohammad'] = "high";
```

```
$salaries['qadir'] = "medium";
$salaries['zara'] = "low";
echo "Salary of mohammad is ". $salaries['mohammad'] . "<br />";
echo "Salary of qadir is ". $salaries['qadir']. "<br />";
echo "Salary of zara is ". $salaries['zara']. "<br />";
?>
```

USER DEFINED FUNCTION IN PHP

```
function <function name>(arg1,arg2,…. arg n)
{
Return;
}
```

Naming convention:

- Functions name not case sensitive.
- Function name can contain only letters from the ASCII char set digit & underscore.
- Function name cannot begin with digit.
- Two or more function not have the same name as PHP does not support function overloading.
- Reserve keyword can't be used for function name.

BUILT IN FUNCTION IN PHP

String manipulation functions

Function Name	
strlen()	To find the length of string
explods()	
implode()	
strpos()	
str_repeat()	
strrev()	

Mathematical Function

Function Name	Syntax
random	rand(int $min, int $max)
log	log($arg,$base)
round	round(float $val,int $precision)
ceil	ceil(float $value)
floor	
max	max($arg1, $arg2, $arg3, $arg4)
min	min($arg1, $arg2, $arg3, $arg4)
number_format	number_format(number[,dec place][,des point][,thousand])
	Ex. number_format($price,2,”.”,””);

The PHP Date() Function

The PHP date() function formats a timestamp to a more readable date and time.

Date (format, timestamp)

Parameters Date & Time Function

PHP Date() - Format the Date

The required *format* parameter in the date() function specifies how to format the date/time.

Here are some characters that can be used:

- d - Represents the day of the month (01 to 31)
- m - Represents a month (01 to 12)
- Y - Represents a year (in four digits)

A list of all the characters that can be used in the *format* parameter, can be found in our PHP Date reference, date() function.

Other characters, like"/", ".", or "-" can also be inserted between the letters to add additional formatting:

Example:

```
<?php
echo date("Y/m/d") . "<br>";
```

```
echo date("Y.m.d") . "<br>";
echo date("Y-m-d");
?>
```

Output:

```
2009/05/11
2009.05.11
2009-05-11
```

Date Timestamp

Format	Description
d	Specifies the day of the month from 1 to 31
D	Specifies the textual representation of a day(three latter eg. Sun,mon)
F	Specifies a full textual representation of month eg. January
t	Specifies the number of days in the given month eg. 30 ,31
g	Specifies 12 hour format of an hour.)(1-12)
L	Specifies whether it's a leep year or not.(1 if leep year otherwise 0)
l	Specifies the full textual representation of a day.
h	Specifies 12 hour format of an hour.)(01-12)
i	Specifies minute with leading zeros ie. 00-59
s	Specifies second with leading zero ie. 00-59
Y	Specifies four digit representation of year
a	Specifies lowercase am & pm
S	Specifies the suffix for the day of the month ie. St,th,nd,rd

Example:

```
<? PHP
echo date("l")."<br>";
echo (date("l d s \0 f f y h : I: s a")."");
?>
```

Output: Thursday

Thursday 16th of July 2009 11:37:03 am

GET AND POST METHOD

There are two ways the browser client can send information to the web server.

- The GET Method
- The POST Method

The GET Method

The GET method sends the encoded user information appended to the page request. The page and the encoded information are separated by the ? character.

```
http://www.test.com/index.htm?name1=value1&name2=value2
```

- The GET method produces a long string that appears in your server logs, in the browser's Location: box.
- The GET method is restricted to send up to 1024 characters only.
- Never use GET method if you have password or other sensitive information to be sent to the server.
- GET can't be used to send binary data, like images or word documents, to the server.
- The data sent by GET method can be accessed using QUERY_STRING environment variable.
- The PHP provides $_GET associative array to access all the sent information using GET method.

The POST Method

The POST method transfers information via HTTP headers. The information is encoded as described in case of GET method and put into a header called QUERY_STRING.

- The POST method does not have any restriction on data size to be sent.
- The POST method can be used to send ASCII as well as binary data.
- The data sent by POST method goes through HTTP header so security depends on HTTP protocol. By using Secure HTTP you can make sure that your information is secure.
- The PHP provides $_POST associative array to access all the sent information using POST method.

The $_REQUEST variable

The PHP $_REQUEST variable contains the contents of both $_GET, $_POST, and $_COOKIE. We will discuss $_COOKIE variable when we will explain about cookies.

The PHP $_REQUEST variable can be used to get the result from form data sent with both the GET and POST methods.

FILE INCLUSION

You can include the content of a PHP file into another PHP file before the server executes it. There are four PHP functions which can be used to included one PHP file into another PHP file.

- include() Function
- require() Function
- include_once() Function
- require_once() Function

The include() and include_once() Function

The include() function takes all the text in a specified file and copies it into the file that uses the include function. If there is any problem in loading a file then the include() function generates a warning but the script will continue execution.

The require() and require_once() Function

The require() function takes all the text in a specified file and copies it into the file that uses the include function. If there is any problem in loading a file then the require() function generates a fatal error and halt the execution of the script.

Browser Redirection

The PHP header() function supplies raw HTTP headers to the browser and can be used to redirect it to another location. The redirection script should be at the very top of the page to prevent any other part of the page from loading.
The target is specified by the Location: header as the argument to the header() function. After calling this function the exit() function can be used to halt parsing of rest of the code.

Example:

```
<?php
        header('Location:index.php');
?>
```

PHP MySQL Database

With PHP, you can connect to and manipulate databases. MySQL is the most popular database system used with PHP.

Function for database Connectivity and Basic Operation

To connect from MYSQL server:

mysql_connect() Function:

It is used for connect from the MYSQL database

Syntax:

mysql_connect(“host_name”,”user_name”,”password”);

Example:

mysql_connect(“localhost”,”root”,””);

mysql_select_db() Function:

it is used from the select database from database server for operation

Syntax:

mysql_select_db(“database_name”);

Example:

mysql_select_db(“chail_computer”);

mysql_query() Function:

it is used to execute the query (sql Statement)

Syntax:

mysql_query(“sql statemenet”);

Example:

mysql_query("select * from student");

mysql_fetch_array() Function:

It is used to fetch the one row from dataset (table).

Syntax:

$row_var=mysql_fetch_array($data_var);

Example:

```
$qs="select * from student";
$data=mysql_query($qs);
while($row=mysql_fetch_array($data))
{
        Echo $row['name']."<br/>";
}
```

mysql_num_rows() Function:

It is used to count the number of rows in dataset.
Synatx:

Mysql_num_rows($data_var);

Example:

```
$qs="select  *  from user_master ";
$data=mysql_query($qs);
if(mysql_num_rows($data)>0)
{
        echo "record found";
}
```

PHP - File Uploading

Step 1: Creating an upload form

The following HTM code below creates an uploader form. This form is having method attribute set to post and enctype attribute is set to multipart/form-data

Step 2: Extract the FILE

There is one global PHP variable called $_FILES. This variable is an associate double dimension array and keeps all the information related to uploaded file. So if the value assigned to the input's name attribute in uploading form was file, then PHP would create following five variables −

$_FILES['file']['tmp_name'] − the uploaded file in the temporary directory on the web server.
$_FILES['file']['name'] − the actual name of the uploaded file.
$_FILES['file']['size'] − the size in bytes of the uploaded file.
$_FILES['file']['type'] − the MIME type of the uploaded file.
$_FILES['file']['error'] − the error code associated with this file upload.

Example:

```
<?php
   if(isset($_FILES['image'])){
      $errors= array();
      $file_name = $_FILES['image']['name'];
      $file_size =$_FILES['image']['size'];
      $file_tmp =$_FILES['image']['tmp_name'];
      $file_type=$_FILES['image']['type'];
      $file_ext=strtolower(end(explode('.',$_FILES['image']['name'])));

      $expensions= array("jpeg","jpg","png");

      if(in_array($file_ext,$expensions)=== false){
         $errors[]="extension not allowed, please choose a JPEG or PNG file.";
      }

      if($file_size > 2097152){
         $errors[]='File size must be excately 2 MB';
      }

      if(empty($errors)==true){
         move_uploaded_file($file_tmp,"images/".$file_name);
         echo "Success";
      }else{
         print_r($errors);
      }
   }
?>
<html>
   <body>

      <form action="" method="POST" enctype="multipart/form-data">
         <input type="file" name="image" />
```

```
         <input type="submit"/>
      </form>

   </body>
</html>
```

PHP - Sessions

An alternative way to make data accessible across the various pages of an entire website is to use a PHP Session.

A session creates a file in a temporary directory on the server where registered session variables and their values are stored. This data will be available to all pages on the site during that visit.

Starting a PHP Session

A PHP session is easily started by making a call to the session_start() function.This function first checks if a session is already started and if none is started then it starts one. It is recommended to put the call to session_start() at the beginning of the page.

Session variables are stored in associative array called $_SESSION[]. These variables can be accessed during lifetime of a session.

The following example starts a session then register a variable called counter that is incremented each time the page is visited during the session.

Make use of isset() function to check if session variable is already set or not.

Put this code in a test.php file and load this file many times to see the result –

```
<?php
  session_start();

  if( isset( $_SESSION['counter'] ) ) {
    $_SESSION['counter'] += 1;
  }else {
    $_SESSION['counter'] = 1;
  }

  $msg = "You have visited this page ".  $_SESSION['counter'];
  $msg .= "in this session.";
?>

<html>

  <head>
    <title>Setting up a PHP session</title>
```

```
  </head>

  <body>
    <?php  echo ( $msg ); ?>
  </body>

</html>
```

Destroying a PHP Session

A PHP session can be destroyed by session_destroy() function. This function does not need any argument and a single call can destroy all the session variables. If you want to destroy a single session variable then you can use unset() function to unset a session variable.

Here is the example to unset a single variable −

```
<?php

  unset($_SESSION['counter']);

?>
```

Here is the call which will destroy all the session variables –

```
<?php

  session_destroy();

?>
```

Cookie

A cookie is often used to identify a user. A cookie is a small file that the server embeds on the user's computer. Each time the same computer requests a page with a browser, it will send the cookie too. With PHP, you can both create and retrieve cookie values.

Create Cookies variable

A cookie is created with the setcookie() function.

Syntax

setcookie(*name, value, expire, path, domain, secure, httponly*);

Only the *name* parameter is required. All other parameters are optional.

Retrieve a Cookie

The following example creates a cookie named "user" with the value "John Doe". The cookie will expire after 30 days (86400 * 30). The "/" means that the cookie is available in entire website (otherwise, select the directory you prefer).

We then retrieve the value of the cookie "user" (using the global variable $_COOKIE). We also use the isset() function to find out if the cookie is set:

Example:

```
<?php
$cookie_name = "user";
$cookie_value = "John Doe";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/"); // 86400 = 1 day
?>
<html>
<body>

<?php
if(!isset($_COOKIE[$cookie_name])) {
   echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
   echo "Cookie '" . $cookie_name . "' is set!<br>";
   echo "Value is: " . $_COOKIE[$cookie_name];
}
?>

</body>
</html>
```

AJAX

AJAX= Asynchronous javaScript and XAML. AJAX is a technique of communication with serever, and updating parts of a webpage- without reloading the whole page. The XMLHttpRequest object is used to exchange data with server behind the scenes.

Create a XMLHttpRequest Object:

```
var reqObj;
if(window.XMLHttpRequest)
{
```

```
    // code for IE7+,Firefox,Chrome,Opera,Safari
    reqObj=new XMLHttpRequest();
  }
  else
  {
      reqObj=new ActiveXObject("Microsoft.XMLHTTP");
   }
```

Send a Request To a Server:

Method	Description
Open(method,url,async):	Specifies the type of request,the URL, and if the request should be handled asynchronously or not. Method: the type of request: GET or POST URL: the location of the file on the server Async: true (asynchronous)or false (synchronous)
Send(String):	Send the request off to the server. String: only used for POST request.

The onreadystatechange Event:

When a requet to a server is sent, we want to perform some action based on the response. The onreadystatechange event is triggered every time the ready state change. The ready sate property holds the status of the XMLHttpRequest. Three important properties of the XMLHttpRequest Object:

Property	Description
readyState:	Hold the status of the XMLHttpRequest. 0: request not initialized 1: server connection established 2: request received 3: processing request

4: request finished and response is ready

Status: 200: “OK”

404: Page Not Found.

Server Response

To get the response from a server, use the responseText or responseXML property of the XMLHTttpRequest Object.

Property	Description
responseText	get the response data as a string
responseXML	get the response data as XML data

Example:

```
<html>
<head>
<script>
function showHint(str) {
   if (str.length == 0) {
      document.getElementById("txtHint").innerHTML = "";
      return;
   } else {
      var xmlhttp = new XMLHttpRequest();
      xmlhttp.onreadystatechange = function() {
         if (this.readyState == 4 && this.status == 200) {
            document.getElementById("txtHint").innerHTML = this.responseText;
         }
      };
      xmlhttp.open("GET", "gethint.php?q=" + str, true);
      xmlhttp.send();
   }
}
</script>
</head>
```

```
<body>

<p><b>Start typing a name in the input field below:</b></p>
<form>
First name: <input type="text" onkeyup="showHint(this.value)">
</form>
<p>Suggestions: <span id="txtHint"></span></p>
</body>
</html>
```