Unit – 7 PHP

Web Technology
2160708
Semester 6

Unit – 7 PHP

Book: Developing Web Applications, Ralph Moseley and M. T. Savaliya, Wiley-India

Web link: https://www.w3schools.com/php/default.asp

Web Link: https://www.tutorialspoint.com/php/index.htm

What is PHP?

- The PHP Hypertext Pre processor (PHP) is a server side scripting language that allows web developers to create dynamic content that interacts with databases.
- PHP is basically used for developing web based software applications.
- PHP is a widely-used, free, and efficient alternative to competitors such as Microsoft's ASP.
- PHP is a server side scripting language that is embedded in HTML.
- It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.

Basic Syntax of PHP

- A PHP script can be placed anywhere in the document.
- A PHP script starts with

```
<?php
 // PHP code goes here
?>
```

- The default file extension for PHP files is ".php". (also .php3 & . php4 or .ph3 & .ph4)

Basic Syntax of PHP

- Example

```
<!DOCTYPE html>
<html>
<head>
    <title>HTML with PHP</title>
</head>
<body>
<?php
echo "Hello, I am Computer Engineer";
?>
</body>
</html>
```

- PHP statements end with a semicolon (;).
- `echo "Hello, I am Computer Engineer";` used to output the text on a web page.

Basic Syntax of PHP

- **Comments in PHP**
  - `// This is a single-line comment`
  - `# This is also a single-line comment`

  -
    ```
    /*
    This is a multiple-lines comment block
    that spans over multiple lines */
    ```

  -
    ```
    // You can also use comments to leave out
    parts of a code line
    $x = 5 /* + 15 */ + 5;
    ```
- In PHP, all keywords (e.g. if, else, while, echo, etc.), are NOT case-sensitive.

  -
    ```
    ECHO "Hello World!<br>";
    OR echo "Hello
    World!<br>";
    ```
- **all variable names are case-sensitive.**

Basic Syntax of PHP

- **Variables**
- PHP is loosely typed means we don’t have to state explicitly type of variable.
- variable starts with the $ sign, followed by the name of the variable.
- A variable name cannot start with a number.
- Variable names are case-sensitive.

```
<?php
   $txt = "Hello world!";
   $x = 5;
   $y = 10.5;
?>
```

Basic Syntax of PHP

- **Output Variables**
- The PHP echo and print statement is often used to output data to the screen.

```
<?php
   $txt = "CE Student!";
   echo "I am $txt";
?>
```

- Example : addition of 2 variables

```
<?PHP
   $x=10;
   $y=20;
   echo $x + $y;
?>
```

Basic Syntax of PHP

- **Scope of Variables**
- PHP has three different variable scopes:
- **Local :** A variable declared within a function has a LOCAL SCOPE and can only be accessed within that function.

```
<?php
function myfunction()
{
    $x=10;
    echo "X is accessible inside function $x";
}
myfunction();
//echo "X is  not accessible outside function
    $x";
?>
```

Basic Syntax of PHP

- **Scope of Variables**
- PHP has three different variable scopes:
- **Global:** A variable declared outside a function has a GLOBAL SCOPE and can only be accessed outside a function.

```
<?php
$x=10;
function myfunction()
{
    //echo "X is not accessible inside
    function $x";
}
myfunction();
echo "X is accessible outside function $x";
?>
```

Basic Syntax of PHP

- **Scope of Variables**
- PHP has three different variable scopes:
- **Global:** The global keyword is used to access a global variable from within a function.
- To do this, use the global keyword before the variables (inside the function):

```
<?php
$x=10;
function myfunction()
{    global $x;
     echo "Now X is accessible inside function $x";
}
myfunction();
echo " X is accessible outside function $x";
?>
```

Basic Syntax of PHP

- **Scope of Variables**
- PHP has three different variable scopes:
- **Static:** sometimes we want a local variable NOT to be deleted, use the static keyword when you first declare the variable.

```
<?PHP
function mytest()
{
    static $x=0;
    echo $x . "<br>";
    $x++;
}
mytest();
mytest();
mytest();
?>
```

Basic Syntax of PHP

- **Data Types**
- PHP supports the following data types:
  - String
  - Integer
  - Float (floating point numbers - also called double)
  - Boolean
  - Array
  - Object
  - NULL
  - Resource

Basic Syntax of PHP

- **Data Types – Array**
- An array stores multiple values in one single variable.
- In PHP, the array() function is used to create an array.

```
<?php
$fruit =
    array("Banana","Apple","Orange","Watermalon");
echo "I like " . $fruit[0] . ", ".
                    $fruit[1] . ", " .
                    $fruit[2] . ", " ."and ".
                    $fruit[3] ;
?>
```

Basic Syntax of PHP

- **Data Types – Array**
- In PHP, there are three types of arrays:
  - **Indexed arrays -** Arrays with a numeric index
  - **Associative arrays -** Arrays with named keys
  - **Multidimensional arrays -** Arrays containing one or more arrays

Question:

What are the different types of arrays in PHP? Explain with example to process the arrays in PHP.

Basic Syntax of PHP

- **Data Types – Array**
- In PHP, there are three types of arrays:
  - **Indexed arrays -** Arrays with a numeric index
  - There are two ways to create indexed arrays:
  - The index can be assigned automatically (index always starts at 0).

```
<?php
$car=array("Volvo","BMW","Toyoto");
echo "I like " . $car[0] .", ". $car[1] .",   ". $car[2];
?>
```

Or

the index can be assigned manually:

```
$cars[0] = "Volvo";
$cars[1] = "BMW";
$cars[2] = "Toyota";
```

Basic Syntax of PHP

- **Data Types – Array**
- In PHP, there are three types of arrays:
  - **Associative arrays -** Arrays with named keys
  - Associative arrays are arrays that use named keys that you assign to them.

```
<?php
$car=array("volvo"=>"20","BMW"=>"12","toyoto"=>"13");
echo "Volvo sold=".$car["volvo"]. "<br>" .
    "BMW sold=" . $car["BMW"] . "<br>".
    "Toyoto sold=" .
        $car["toyoto"];
?>
```

  - or the named key can be assigned manually
    - `$age['volvo'] = "20";`

Basic Syntax of PHP

- **Data Types – Array**
- In PHP, there are three types of arrays:
  - **Multidimensional arrays -** Arrays containing one or more arrays
  - For a two-dimensional array you need two indices to select an element.

  -
    ```
    $cars = array
      (
      array("Volvo",22,18),
      array("BMW",15,13),
      array("Saab",5,2),
      array("Land Rover",17,15)
      );
    ```

Basic Syntax of PHP

- **Multidimensional arrays -** Example

```
<?php
$car=array
      (array("volvo",22,18),
       array("BMW",15,13),
       array("toyoto",5,2),
       array("Land Rover",17,15)
      );
for($i=0;$i<4;$i++)
{
    echo "Car".$i;
    echo "<ul>";
    for($j=0;$j<3;$j++)
    {
          //echo "<li>".$i."-".$j."</li>";
          echo "<li>".$car[$i][$j]."</li>";
    }
    echo "</ul>";
}   ?>
```

Basic Syntax of PHP

- **PHP - Sort Functions For Arrays**
- **sort() -** sort arrays in ascending order
- **rsort() -** sort arrays in descending order
- **asort() -** sort associative arrays in ascending order, according to the value
- **ksort() -** sort associative arrays in ascending order, according to the key
- **arsort() -** sort associative arrays in descending order, according to the value
- **krsort() -** sort associative arrays in descending order, according to the key

Decisions in PHP

- Conditional statements are used to perform different actions based on different conditions.
- In PHP we have the following conditional statements:
  - **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...elseif....else statement** - executes different codes for more than two conditions
  - **switch statement** - selects one of many blocks of code to be executed

Decisions in PHP

- **if statement** - executes some code if one condition is true

- The example below will output "Have a good day!" if the current time (HOUR) is less than 20

```
<?php
$t = date("H");

if ($t < "20") {
    echo "Have a good day!";
}
?>
```

Decisions in PHP

- **if...else statement** - executes some code if a condition is true and another code if that condition is false
- The example below will output "Have a good day!" if the current time is less than 20, and "Have a good night!" otherwise.

```
<?php
$t = date("H");
if ($t < "20")
    echo "Have a good day!";
else
    echo "Good Night";
?>
```

Decisions in PHP

- **if...elseif....else statement** - executes different codes for more than two conditions

```
<?php
$t = date("H");
if ($t < "13")
    echo "Have a good day!";
elseif ($t < 20)
{
    echo "Have a good day";
}
else
{
    echo "Good Night";
}
?>
```

Decisions in PHP

- **switch statement** - selects one of many blocks of code to be executed

```
<?php
 $favcolor= "blue";
switch ($favcolor) {
     case "red":
          echo "Your favorite color is red!";
          break;
    case "blue":
          echo "Your favorite color is blue!";
          break;
    case "green":
          echo "Your favorite color is green!";
          break;
    default:
          echo "Your favorite color is neither red, blue,
    nor green!";
}   ?>
```

Looping in PHP

- In PHP, we have the following looping statements:
  - **while- loops** through a block of code as long as the specified condition is true
  - **do...while - loops** through a block of code once, and then repeats the loop as long as the specified condition is true
  - **for- loops** through a block of code a specified number of times
  - **foreach- loops** through a block of code for each element in an array

Looping in PHP

- **while- loops** through a block of code as long as the specified condition is true

```
<?php
$x = 1;
while($x <= 5)
{
    echo "The number is: $x <br>";
    $x++;
}
?>
```

Question:

Write PHP program to print sum of 1 to 10 number.

Looping in PHP

- **do...while - loops** through a block of code once, and then repeats the loop as long as the specified condition is true

```
<?php
$x = 1;

do {
    echo "The number is: $x <br>";
    $x++;
} while ($x <= 5);
?>
```

Looping in PHP

- **for- loops** through a block of code a specified number of times

```
<?php
 for ($x = 0; $x <= 10; $x++)
 {
     echo "The number is: $x <br>";
 }
 ?>
```

Question:

Write PHP program to print following pattern.

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

Looping in PHP

- **foreach- loops** works only on arrays, and is used to loop through each key/value pair in an array.

```
<?php
$colors = array("red", "green", "blue", "yellow");
foreach ($colors as $value)
{
    echo "$value <br>";
}
?>
```

Functions in PHP

- The real power of PHP comes from its functions; it has more than 1000 built-in functions.
- A user-defined function declaration starts with the word `function`

```
<?php
function writeMsg()
{
    echo "Hello world!";
}

writeMsg();     // call the function
?>
```

Functions in PHP

- PHP Function Arguments

```
<?php
function familyName($fname, $year)
{
echo "$fname Born in $year <br>";
}

familyName("Heet", "2005");
familyName("Sila", "2008");
familyName("Jimy", "2010");
?>
```

Functions in PHP

- PHP Default Argument Value

```
<?php
function setHeight($minheight = 50)
{
echo "The height is : $minheight <br>";
}

setHeight(350);
setHeight(); // will use the default value of 50
setHeight(135);
setHeight(80);
?>
```

Functions in PHP

- PHP Functions - Returning values

```
<?php
function sum($x,$y)
{
    $z = $x + $y;
    return $z;
}

echo "5 + 10 = " . sum(5, 10) . "<br>";
echo "7 + 13 = " . sum(7, 13) . "<br>";
echo "2 + 4 = " . sum(2, 4);
?>
```

Strings in PHP

- A string is a sequence of characters, like "Hello world!".
- String have some commonly used functions to manipulate strings.

```
<?php
    $str ="This  is Testing";
    print$str[0]  . "<br>";
    print$str[2]  . "<br>";
    print$str  . "<br>";
?>
```

Strings in PHP

- **PHP String Functions**

```
<?php
echo "Number of Characters=". strlen("Hello world!") .
    "<br>"; // outputs 12

echo "Number of Words=" . str_word_count("Hello world!").
    "<br>"; // outputs 2

echo "Reverse String=". strrev("Hello world!")."<br>"; //
    outputs !dlrow olleH

echo "Position of World=" . strpos("Hello world!",
    "world")."<br>"; // outputs 6

echo str_replace("world", "India", "Hello
    world!")."<br>"; // outputs Hello India!
?>
```

PHP Global Variables - Superglobals

- Several predefined variables in PHP are "superglobals", which means that they are always accessible, regardless of scope - and you can access them from any function, class or file without having to do anything special.

- $GLOBALS: PHP super global variable which is used to access global variables from anywhere in the PHP script.

- $_SERVER: PHP super global variable which holds information about headers, paths, and script locations.

- $_REQUEST: is used to collect data after submitting an HTML form.

PHP Global Variables - Superglobals

- $_POST : used to collect form data after submitting an HTML form with method="post".

- $_GET : used to collect form data after submitting an HTML form with method="get".

- $_FILES: array to store all the information about files. File will be uploaded in a temporary folder on web server.

- $_ENV: An associative array of variables passed to the current script via the environment method.

PHP Global Variables - Superglobals

- $_COOKIE: which holds all cookie names and values.

- $_SESSION: array variable used to store session information.

Form processing in PHP

form.html

```
<html>
<body>
    <center>
<form action="welcome.php" method="post">
User Name: <input type="text" name="name"><br><br><br>
Password: <input type="password"
    name="password"><br><br><br>
<input type="submit" value="Submit">
</form>
</center>
</body>
</html>
```

When the user fills out the form above and clicks the submit button, the form data is sent for processing to a PHP file named "welcome.php". The form data is sent with the HTTP POST method.

Form processing in PHP

- To display the submitted data you could simply echo all the variables.

Welcome.php

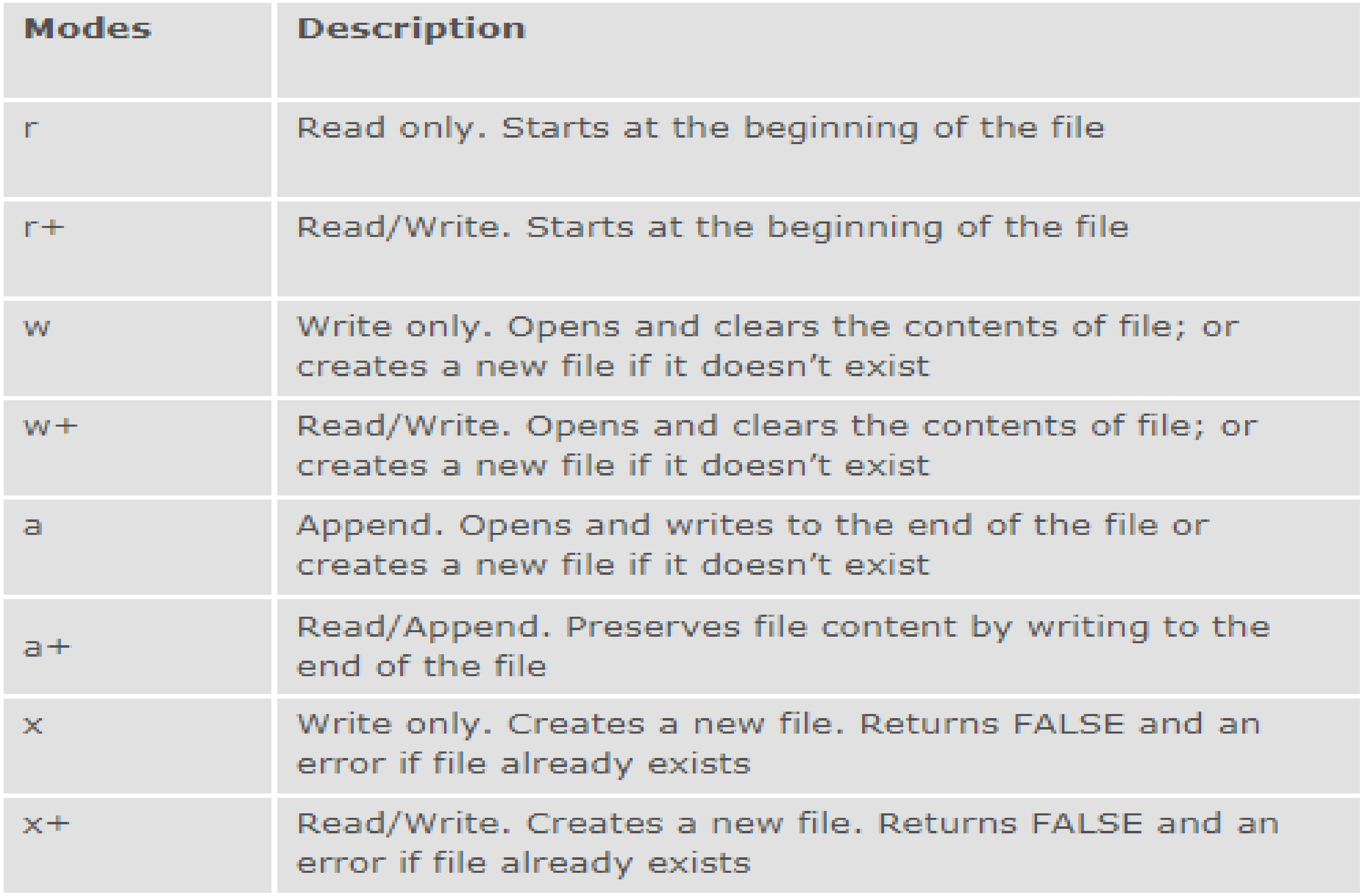
```
<html>
<body>
Welcome <?php echo $_POST["name"]; ?><br>
You are successfully logged in.<br>
</body>
</html>
```

Files in PHP

- File handling is an important part of any web application. You often need to open and process a file for different tasks.
- PHP has several functions for creating, reading, uploading, and editing files.
- **Opening and Closing Files**
  - fopen() function is used to open a file.
  - Syntax: `fopen( $filename, "mode" );`
- After making a changes to the opened file it is important to close it with the **fclose() function**. The fclose() function requires a file pointer as its argument and then returns true when the closure succeeds or false if it fails.

Files in PHP

- File opening modes



Modes	Description
r	Read only. Starts at the beginning of the file
r+	Read/Write. Starts at the beginning of the file
w	Write only. Opens and clears the contents of file; or creates a new file if it doesn't exist
w+	Read/Write. Opens and clears the contents of file; or creates a new file if it doesn't exist
a	Append. Opens and writes to the end of the file or creates a new file if it doesn't exist
a+	Read/Append. Preserves file content by writing to the end of the file
x	Write only. Creates a new file. Returns FALSE and an error if file already exists
x+	Read/Write. Creates a new file. Returns FALSE and an error if file already exists

Files in PHP

Example:

```
<?php
$fp = fopen("myfile.txt","a+");
if( $fp == false)
{
  echo ( "Error in opening file" );
}
else{
 echo ( "File Open for reading and
   writting\n" );
 }
$txt="JS = JavaScript";
fwrite($fp,$txt);
fclose($fp);
echo readfile("myfile.txt");
?>
```

File Upload in PHP

- The main use of PHP file handling is to upload files on web server.

- PHP allows you to upload single and multiple files through few lines of code only.

- PHP file upload features allows you to upload binary and text files both.

- Moreover, you can have the full control over the file to be uploaded through PHP authentication and file operation functions.

File Upload in PHP

- $_FILES: array to store all the information about files.
- The PHP global **$_FILES** contains all the information of file.
- By the help of $_FILES global, we can get file name, file type, file size, temp file name and errors associated with file.

- $_FILES['filename']['name']
  - returns file name.

- $_FILES['filename']['type']
  - returns MIME type of the file.

File Upload in PHP

- $_FILES['filename']['size']
  - returns size of the file (in bytes).

- $_FILES['filename']['tmp_name']
  - returns temporary file name of the file which was stored on the server.

- $_FILES['filename']['error']
  - returns error code associated with this file.

File Upload in PHP

- Example: upload.php

```
<!DOCTYPE html>
<html><head>
  <title>Upload your files</title>
</head>
<body>
  <form enctype="multipart/form-data"
   action="upload.php" method="POST">
    <p>Upload your file</p>
    <input type="file"
   name="uploaded_file"></input><br />
    <input type="submit"
   value="Upload"></input>
  </form>
</body>
</html>
```

File Upload in PHP

- Example: upload.php (file continue…)

```
<?PHP
  if(!empty($_FILES['uploaded_file']))
  { $path = "uploads/";
    $path = $path . basename(
   $_FILES['uploaded_file']['name']);

   if(move_uploaded_file($_FILES['uploaded_f
   i   le']['tmp_name'],      $path))     {
      echo "The file ". basename(
   $_FILES['uploaded_file']['name']).
      " has been uploaded";
    } else{
        echo "There was an error uploading
   the file, please try again!";
    }    }   ?>
```

File Upload in PHP

- <form enctype="multipart/form-data“ : This value is required when you are using forms that have a file upload control.
- <input type="file" name="uploaded_file"> for file upload form control.
- $path = "uploads/";: Specify file uploading path
- basename( ) : basename(path,suffix)
- move_uploaded_file
  - Moves an uploaded file to a new location
  - bool move_uploaded_file (string $filename , string $destination);

Cookie and Sessions in PHP

- **Why and when to use Cookies and Sessions?**
  - HTTP is a stateless protocol does not require the server to retain information or status about each user for the duration of multiple requests.
  - But some web applications may have to track the user's progress from page to page, for example when a web server is required to customize the content of a web page for a user. Solutions for these cases include:
    - **the use of HTTP cookies.**
    - **server side sessions**

Cookie and Sessions in PHP

- **Cookie:** A cookie is a small text file that the web server stores on the client computer.

- the php **set cookie function** must be executed before the HTML opening tag

```
<?php
setcookie(cookie_name, cookie_value,
[expiry_time], [cookie_path], [domain],
[secure], [httponly]);
?>
```

- php“**setcookie**” is the PHP function used to create the cookie.

Cookie and Sessions in PHP

- **Cookies**
  - **“cookie_name**” is the name of the cookie that the server will use when retrieving its value from the $_COOKIE array variable. It’s mandatory.
  - **“cookie_value”** is the value of the cookie and its mandatory.
  - **“[expiry_time]”** is optional; it can be used to set the expiry time for the cookie such as 1 hour. i.e. time() + 3600 for 1 hour.
  - **“[cookie_path]”** is optional; it can be used to set the cookie path on the server. The **forward slash “/”** means that the cookie will be made available on the entire domain. Sub directories limit the cookie access to the subdomain.

Cookie and Sessions in PHP

- **Cookies**
  - **“[domain]”** is optional, it can be used to define the cookie access hierarchy.
  - **“[secure]”** is optional, the default is false. It is used to determine whether the cookie is sent via https if it is set to true or http if it is set to false.
  - **“[Httponly]”** is optional. If it is set to true, then only client side scripting languages i.e. JavaScript cannot access them.

Cookie and Sessions in PHP

- **Creating Cookies – Example**

```
<?php
   setcookie("name", "John", time()+10, "/","",
0);
   setcookie("age", "36", time()+3600, "/", "",
0);
?>
<html>
      <head>
      <title>Setting Cookies with PHP</title>
   </head>
   <body>
      <?php echo "Set Cookies"?>
   </body>
</html>
```

Cookie and Sessions in PHP

- **Accessing Cookies with PHP**
- $_COOKIE or $HTTP_COOKIE_VARS variables used to access cookies.

```
<html>
<head><title>Accessing Cookies with PHP</title>
</head>
<body>
<?php
 if( isset($_COOKIE["name"]))
   echo "Welcome " . $_COOKIE["name"] . "<br />";
 else
   echo "Sorry... Not recognized" . "<br />";
?>
</body>
</html>
```

Cookie and Sessions in PHP

- **Deleting Cookie with PHP**
  - delete a cookie you should set the cookie with a date that has already expired

```
<?php
   setcookie( "name", "", time()- 60, "/","", 0);
   setcookie( "age", "", time()- 60, "/","", 0);
?>
<html>
<head><title>Deleting Cookies with PHP</title>
</head>
   <body>
      <?php echo "Deleted Cookies" ?>
   </body>
</html>
```

Cookie and Sessions in PHP

- **Sessions:**
- A session is a global variable **stored on the server.**

- Each session is assigned a unique id which is used to retrieve stored values.

- Whenever a session is created, a cookie containing the unique session id is stored on the user’s computer and returned with every request to the server. If the client browser does not support cookies, the unique php session id is displayed in the URL.

Cookie and Sessions in PHP

- **Sessions:**
- Sessions have the capacity to store relatively large data compared to cookies.

- The session values are automatically deleted when the browser is closed.

- $_SESSION array variable used to store session information.

- Just like cookies, the session must be started before any HTML tags.

Cookie and Sessions in PHP

- **Create Sessions:**
- In order to create a session, you must first call the PHP session_start function and then store your values in the $_SESSION array variable.

```
<?php
        session_start();
                if(isset($_SESSION['page_count']))
                        $_SESSION['page_count']  += 1;
                else
                        $_SESSION[' page_count ' ]   = 1;
        echo 'You are visitor number'.
        $_SESSION['page_count'];
?>
```

Cookie and Sessions in PHP

- **Create Sessions: (continue...)**

```
<!DOCTYPE html>
<html>
<head>
    <title>Session in PHP</title>
</head>
<body>
<?php
echo "Session started";
?>
</body>
</html>
```

Cookie and Sessions in PHP

- **Destroying Session Variables**

```
<?php

 session_destroy(); //destroy entire session

?>
```

OR

```
<?php

unset($_SESSION['product']); //destroy product
session item

?>
```

Cookie and Sessions in PHP

Cookies	Sessions
Cookies are stored in browser as text file format.	Sessions are stored in server side.
It is stored limit amount of data.	It is stored unlimited data
It is only allowing 4kb[4096bytes].	It is holding the multiple variable in sessions.
It is not holding the multiple variable in cookies.	It is holding the multiple variable in sessions.
we can accessing the cookies values in easily. So it is less secure.	we cannot accessing the session values in easily. So it is more secure.
setting the cookie time to expire the cookie.	using session_destory(), we will destroyed the sessions.
The setcookie() function must appear BEFORE the <html> tag.	The session_start() function must be first thing in your document.

Object Oriented Programming with PHP

- **Object oriented programming concepts:**
  - Class
  - Object
  - Member Variable
  - Member function
  - Inheritance
  - Polymorphism
  - Data Abstraction
  - Constructor

Object Oriented Programming with PHP

- **Defining PHP Classes**

```
<?php
    class phpClass
{
        var $var1;
        var $var2 = "constant string";

        function myfunc ($arg1, $arg2)
         {
            [..]
        }
        [..]
}
?>
```

Object Oriented Programming with PHP

- **Example : creating class**

```
<?php
class Books
{
 var $price;
 var $title;
 function setPrice($par){$this->price = $par;}
 function getPrice(){echo $this->price."<br/>";}
 function setTitle($par){$this->title = $par;}
 function getTitle(){echo $this->title."<br/>";}
   }
?>
```

- The variable $this is a special variable and it refers to the same object ie. itself.

Object Oriented Programming with PHP

- **Creating object**

```
$physics = new Books;
$maths = new Books;
$chemistry = new Books;
```

- **Calling Member Functions**

```
$physics->setTitle( "Physics for High School" );
$physics->setPrice( 10 );
$physics->getTitle();
$physics->getPrice();
```

Object Oriented Programming with PHP

- **Constructor Functions**
- PHP provides a special function called **__construct()** to define a constructor.

```
function __construct( $par1, $par2 )
{
    $this->title = $par1;
    $this->price = $par2;
}

//create object
$physics = new Books( "Physics for High School",
10 );
```

- Like a constructor function you can define a destructor function using function **__destruct().**

Object Oriented Programming with PHP

- **Inheritance**

```
class Novel extends Books
 {
   var $publisher;
   function setPublisher($par)
   {
      $this->publisher = $par;
   }
   function getPublisher()
   {
      echo $this->publisher. "<br />";
   }
}
```

Object Oriented Programming with PHP

- **Function Overriding**
- Function definitions in child classes override definitions with the same name in parent classes.

- Class members may be **public, private or protected**.

```
class MyClass {
   protected $car = "skoda";
```

Object Oriented Programming with PHP

- **Interfaces**
- interfaces are skeletons which are implemented by developers.

```
interface Mail
{
    public function sendMail();
}
class Report implements Mail
{
    // sendMail() Definition goes here
}
```

Object Oriented Programming with PHP

- **Abstract Classes**
- An abstract class is one that cannot be instantiated, only inherited. You declare an abstract class with the keyword **abstract.**
- When inheriting from an abstract class, all methods marked abstract in the parent's class declaration must be defined by the child.

*****************END******************