

## **PROGRAMMING IN PHP (SUBJECT CODE - 16SCCCS9-16SCCCA9-16SMBEIT2:2 )**

**Preface** - This course material for programming in PHP has been prepared in the form of question and answers to help the students in preparing for semester exams easily. It contains questions and answers for all the five units of the PHP syllabus of Bharathidasan University, Tiruchirappalli.

**Author:** Dr. S. Chellammal

**Address:**  
Assistant Professor, Department of Computer Science  
Bharathidasan University Constituent Arts & Science College  
Navalurkuttapattu  
Tiruchirappalli-620027

**Date uploaded:** 24.05.2020

**Number of units covered:** All the 5 units of the syllabus  
**SYLLABUS**

**Objective:** To understand the Concepts of PHP and Ajax.

### **Unit I**

Essentials of PHP - Operators and Flow Control - Strings and Arrays.

### **Unit II**

Creating Functions - Reading Data in Web Pages - PHP Browser - Handling Power.

### **Unit III**

Object-Oriented Programming –Advanced Object-Oriented Programming .

### **Unit IV**

File Handling –Working with Databases – Sessions, Cookies, and FTP

### **Unit V**

Ajax – Advanced Ajax – Drawing Images on the Server.

### **Text Book:**

1. The PHP Complete Reference, Steven Holzner, McGrawHillEducation, 2007

## UNIT – I

### 1. What is PHP? List out its features

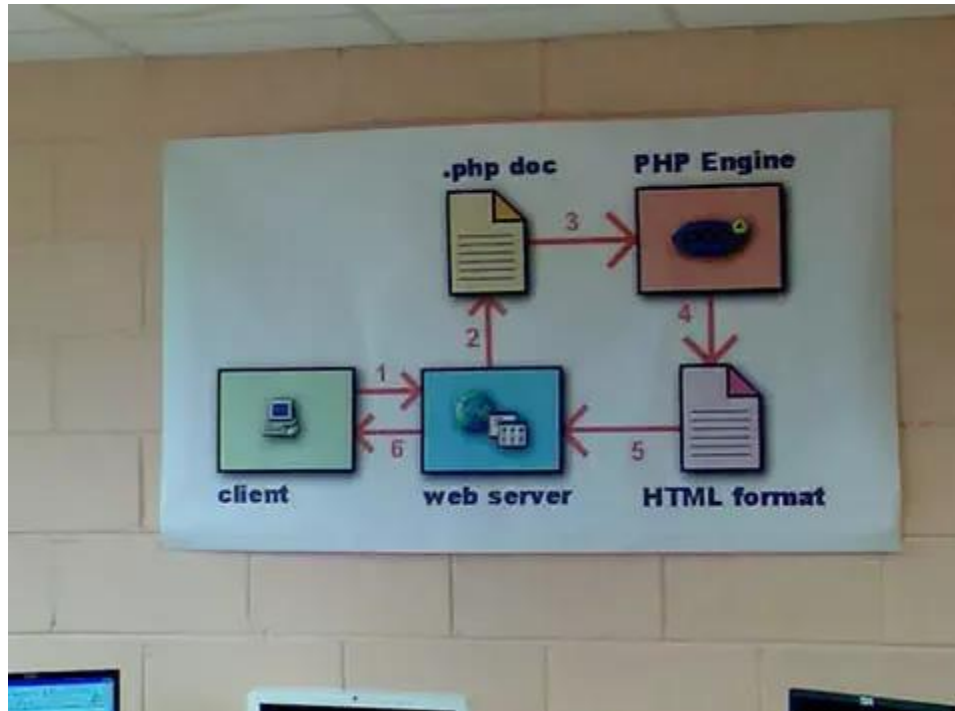
- PHP is an acronym for "PHP: Hypertext Preprocessor" or “Personal Home Page”
- PHP is a widely-used, open source, server side scripting language
- PHP can generate dynamic page content
- PHP scripts are executed on the server
- PHP is free to download and use
- PHP files can contain text, HTML, CSS, JavaScript, and PHP code
- PHP code is executed on the server, and the result is returned to the browser as plain HTML
- PHP files have extension ".php"
- PHP can create, open, read, write, delete, and close files on the server
- PHP can collect form data
- PHP can send and receive cookies
- PHP can add, delete, modify data in your database
- PHP can be used to control user-access
- PHP can encrypt data
- PHP is case sensitive

### 2. List out the advantages of PHP

- Open source
- Speed
- Easy and simple to use
- Reliable
- Can run on any platform
- Built-in relational database modules
- Strong library support

### 3. How does PHP work?

As in the diagram below, we always start with a browser making a request for a web page. This request is going to hit the web server. The web server will then analyze it and determine what to do with it.



If the web server determines that the request is for a PHP file (.php doc) it'll pass that file to the PHP interpreter. The PHP interpreter will read the PHP file, parse it (and other included files) and then execute it. Once the PHP interpreter finishes executing the PHP file, it'll return an output. The web server will take that output and send it back as a response to the browser.

#### 4. Explain the syntax of PHP with example

The syntax of PHP script is given below

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

The file extension is .php

#### Example – Helloworld.php

```
<?php
Echo "Hello world";
?>
```

## 5. List out the syntax of comment line in PHP

### Single comment line

```
<?
# This is a comment, and
# This is the second line of the comment

// This is a comment too. Each style comments only
print "An example with single line comments";
?>
```

### Multi-line comment

```
<?
/* This is a comment with multiline
   Author : Mohammad Mohtashim
   Purpose: Multiline Comments Demo
   Subject: PHP
*/

print "An example with multi line comments";
?>
```

## 6. Explain different data types in PHP

PHP has a total of eight data types which we use to construct our variables –

- **Integers** – are whole numbers, without a decimal point, like 4195.
- **Doubles** – are floating-point numbers, like 3.14159 or 49.1.
- **Booleans** – have only two possible values either true or false.
- **NULL** – is a special type that only has one value: NULL.
- **Strings** – are sequences of characters, like 'PHP supports string operations.'
- **Arrays** – are named and indexed collections of other values.
- **Objects** – are instances of programmer-defined classes, which can package up both other kinds of values and functions that are specific to the class.
- **Resources** – are special variables that hold references to resources external to PHP (such as database connections).

The first five are *simple types*, and the next two (arrays and objects) are compound - the compound types can package up other arbitrary values of arbitrary type, whereas the simple types cannot.

## **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**

```
<?php
$x = "Hello world!";
$y = 'Hello world!';
```

## **Integer**

They are whole numbers, like 4195.

They are the simplest type.

They correspond to simple whole numbers, both positive and negative.

### **Example**

```
$int_var = 12345;
$another_int = -12345
```

Integer can be in decimal (base 10), octal (base 8), and hexadecimal (base 16) format. Decimal format is the default, octal integers are specified with a leading 0(zero), and hexadecimal have a leading 0x(zerox).

## **Double(also called as float)**

A float (floating point number) is a number with a decimal point or a number in exponential form.

```
<?php
$x = 10.365;
var_dump($x);
?>
```

## **Boolean**

A Boolean represents two possible states: TRUE or FALSE.

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

## **PHP Array**

An array stores multiple values in one single variable.

In the following example \$cars is an array.

```
$cars = array("Volvo","BMW","Toyota");
```

## **PHP Object**

An object is a data type which stores data and information on how to process that data.

In PHP, an object must be explicitly declared.

First we must declare a class of object. For this, we use the class keyword. A class is a structure that can contain properties and methods:

```
<?php
class Car {
    function Car() {
        $this->model = "VW";
    }
}
// create an object
$herbie = new Car();

// show object properties
echo $herbie->model;
?>
```

### PHP NULL Value

Null is a special data type which can have only one value: NULL.

A variable of data type NULL is a variable that has no value assigned to it.

Example

```
$x = null;
```

### PHP Resource

The special resource type is not an actual data type. It is the storing of a reference to functions and resources external to PHP.

A common example of using the resource data type is a database call.

## **7. Explain how to use variable in PHP.**

Variables are "containers" for storing information. It is a name given to memory location. In PHP, a variable starts with the \$ sign, followed by the name of the variable

### Example

```
$a=10;//here $a is a variable
```

### Rules

- A variable starts with the \$ sign, followed by the name of the variable
- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and \_)
- Variable names are case-sensitive (\$age and \$AGE are two different variables)

## 8. Explain different types of variables. (or) Explain scope of variables in PHP.

### PHP Variables Scope

In PHP, variables can be declared anywhere in the script.

The scope of a variable is the part of the script where the variable can be referenced/used.

PHP has three different variable scopes:

- local
- global
- static

#### Local

- A variable declared **within** a function has a LOCAL SCOPE and can only be accessed within that function

- ```
<?php
function myTest() {
    $x = 5; // local scope
    echo "<p>Variable x inside function is: $x</p>";
}
myTest();
?>
```

#### Output

Variable x inside function is: 5

#### 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 = 5;
$y = 10;

function myTest() {
    global $x, $y;
    $y = $x + $y;
}
myTest();
echo $y; // outputs 15
?>
```

#### Static

Normally, when a function is completed/executed, all of its variables are deleted. However, sometimes we want a local variable NOT to be deleted. To do this, use the static keyword when you first declare the variable:

Example

```
<?php
function myTest() {
```

```

static $x = 0;
echo $x;
$x++;
}

```

```

myTest();
echo "<br>";
myTest();
echo "<br>";
myTest();
?>

```

output

```

0
1
2

```

## 8. Explain in details various operators in PHP

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
- Conditional assignment operators

### Arithmetic operators

The PHP arithmetic operators are used with numeric values to perform common arithmetical operations, such as addition, subtraction, multiplication etc.

| Operator | Name           | Example      | Result                                    |
|----------|----------------|--------------|-------------------------------------------|
| +        | Addition       | $\$x + \$y$  | Sum of \$x and \$y                        |
| -        | Subtraction    | $\$x - \$y$  | Difference of \$x and \$y                 |
| *        | Multiplication | $\$x * \$y$  | Product of \$x and \$y                    |
| /        | Division       | $\$x / \$y$  | Quotient of \$x and \$y                   |
| %        | Modulus        | $\$x \% \$y$ | Remainder of \$x divided by \$y           |
| **       | Exponentiation | $\$x ** \$y$ | Result of raising \$x to the \$y'th power |



### Assignment operators

The PHP assignment operators are used with numeric values to write a value to a variable. The basic assignment operator in PHP is "=". It means that the left operand gets set to the value of the assignment expression on the right.

| Assignment          | Same as...             | Description                                                           |
|---------------------|------------------------|-----------------------------------------------------------------------|
| <code>x = y</code>  | <code>x = y</code>     | The left operand gets set to the value of the expression on the right |
| <code>x += y</code> | <code>x = x + y</code> | Addition                                                              |
| <code>x -= y</code> | <code>x = x - y</code> | Subtraction                                                           |
| <code>x *= y</code> | <code>x = x * y</code> | Multiplication                                                        |
| <code>x /= y</code> | <code>x = x / y</code> | Division                                                              |
| <code>x %= y</code> | <code>x = x % y</code> | Modulus                                                               |

### PHP Comparison operators

The PHP comparison operators are used to compare two values (number or string):

| Operator               | Name                     | Example                        | Result                                                                                                                                              |
|------------------------|--------------------------|--------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>==</code>        | Equal                    | <code>\$x == \$y</code>        | Returns true if \$x is equal to \$y                                                                                                                 |
| <code>===</code>       | Identical                | <code>\$x === \$y</code>       | Returns true if \$x is equal to \$y, and they are of the same type                                                                                  |
| <code>!=</code>        | Not equal                | <code>\$x != \$y</code>        | Returns true if \$x is not equal to \$y                                                                                                             |
| <code>&lt;&gt;</code>  | Not equal                | <code>\$x &lt;&gt; \$y</code>  | Returns true if \$x is not equal to \$y                                                                                                             |
| <code>!==</code>       | Not identical            | <code>\$x !== \$y</code>       | Returns true if \$x is not equal to \$y, or they are not of the same type                                                                           |
| <code>&gt;</code>      | Greater than             | <code>\$x &gt; \$y</code>      | Returns true if \$x is greater than \$y                                                                                                             |
| <code>&lt;</code>      | Less than                | <code>\$x &lt; \$y</code>      | Returns true if \$x is less than \$y                                                                                                                |
| <code>&gt;=</code>     | Greater than or equal to | <code>\$x &gt;= \$y</code>     | Returns true if \$x is greater than or equal to \$y                                                                                                 |
| <code>&lt;=</code>     | Less than or equal to    | <code>\$x &lt;= \$y</code>     | Returns true if \$x is less than or equal to \$y                                                                                                    |
| <code>&lt;=&gt;</code> | Spaceship                | <code>\$x &lt;=&gt; \$y</code> | Returns an integer less than, equal to, or greater than zero, depending on if \$x is less than, equal to, or greater than \$y. Introduced in PHP 7. |

## **Increment and Decrement Operators**

The PHP increment operators are used to increment a variable's value.  
The PHP decrement operators are used to decrement a variable's value.

| <b>Operator</b>    | <b>Name</b>    | <b>Description</b>                      |
|--------------------|----------------|-----------------------------------------|
| <code>++\$x</code> | Pre-increment  | Increments \$x by one, then returns \$x |
| <code>\$x++</code> | Post-increment | Returns \$x, then increments \$x by one |
| <code>--\$x</code> | Pre-decrement  | Decrements \$x by one, then returns \$x |
| <code>\$x--</code> | Post-decrement | Returns \$x, then decrements \$x by one |

## **PHP Logical operators**

The PHP logical operators are used to combine conditional statements.

| <b>Operator</b>   | <b>Name</b> | <b>Example</b>            | <b>Result</b>                                          |
|-------------------|-------------|---------------------------|--------------------------------------------------------|
| <b>and</b>        | <b>And</b>  | <b>\$x and \$y</b>        | <b>True if both \$x and \$y are true</b>               |
| <b>or</b>         | <b>Or</b>   | <b>\$x or \$y</b>         | <b>True if either \$x or \$y is true</b>               |
| <b>xor</b>        | <b>Xor</b>  | <b>\$x xor \$y</b>        | <b>True if either \$x or \$y is true, but not both</b> |
| <b>&amp;&amp;</b> | <b>And</b>  | <b>\$x &amp;&amp; \$y</b> | <b>True if both \$x and \$y are true</b>               |
| <b>  </b>         | <b>Or</b>   | <b>\$x    \$y</b>         | <b>True if either \$x or \$y is true</b>               |
| <b>!</b>          | <b>Not</b>  | <b>!\$x</b>               | <b>True if \$x is not true</b>                         |

## **PHP String Operators**

PHP has two operators that are specially designed for strings.

| <b>Operator</b> | <b>Name</b>              | <b>Example</b>                | <b>Result</b>                      |
|-----------------|--------------------------|-------------------------------|------------------------------------|
| <code>.</code>  | Concatenation            | <code>\$txt1 . \$txt2</code>  | Concatenation of \$txt1 and \$txt2 |
| <code>.=</code> | Concatenation assignment | <code>\$txt1 .= \$txt2</code> | Appends \$txt2 to \$txt1           |

## 9. List out String operators in PHP with example.

There are two string operators.

(i) Concatenation .

(ii) Concatenation assignment .=

### Example for . concatenation

```
<?php
$txt1 = "Hello";
$txt2 = " world!";
echo $txt1 . $txt2;    //output Helloworld!
?>
```

### Example for . Concatenation assignment

```
<?php
$txt1 = "Hello";
$txt2 = " world!";
$txt1 .= $txt2;
echo $txt1;
?>
```

## 10. Explain PHP array operators.

The PHP array operators are used to compare arrays.

| Operator | Name         | Example       | Result                                                                                                |
|----------|--------------|---------------|-------------------------------------------------------------------------------------------------------|
| +        | Union        | $\$x + \$y$   | Union of $\$x$ and $\$y$                                                                              |
| ==       | Equality     | $\$x == \$y$  | Returns true if $\$x$ and $\$y$ have the same key/value pairs                                         |
| ===      | Identity     | $\$x === \$y$ | Returns true if $\$x$ and $\$y$ have the same key/value pairs in the same order and of the same types |
| !=       | Inequality   | $\$x != \$y$  | Returns true if $\$x$ is not equal to $\$y$                                                           |
| <>       | Inequality   | $\$x <> \$y$  | Returns true if $\$x$ is not equal to $\$y$                                                           |
| !==      | Non-identity | $\$x !== \$y$ | Returns true if $\$x$ is not identical to $\$y$                                                       |

## 11. Explain conditional assignment operator in PHP with example

The PHP conditional assignment operators are used to set a value depending on conditions:

| Operator | Name    | Example                                             | Result                                                                                                                                      |
|----------|---------|-----------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| ?:       | Ternary | \$x<br>= <i>expr1</i> ? <i>expr2</i> : <i>expr3</i> | Returns the value of \$x.<br>The value of \$x is <i>expr2</i> if <i>expr1</i> = TRUE.<br>The value of \$x is <i>expr3</i> if <i>expr1</i> = |

### Example for conditional operator

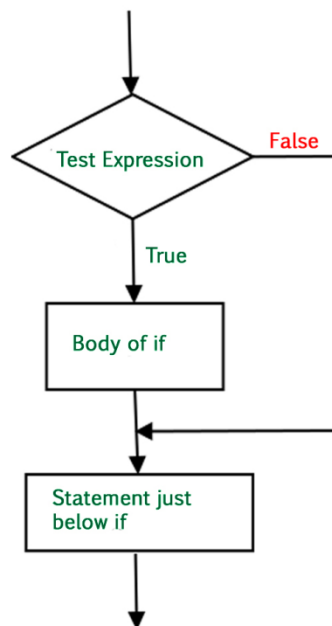
```
<?php
    $a = 10;
    $b = 20;

    /* If condition is true then assign a to result otheriwse b */
    $result = ($a > $b ) ? $a : $b;
    echo "TEST1 : Value of result is $result<br/>";
    /* If condition is true then assign a to result otheriwse b */
    $result = ($a < $b ) ? $a : $b;

    echo "TEST2 : Value of result is $result<br/>";
?>
```

### 12. Explain if and if-else in PHP

Simple if – with simple if statement, a set of code is executed if a condition is true. If the condition is false, the code is not executed



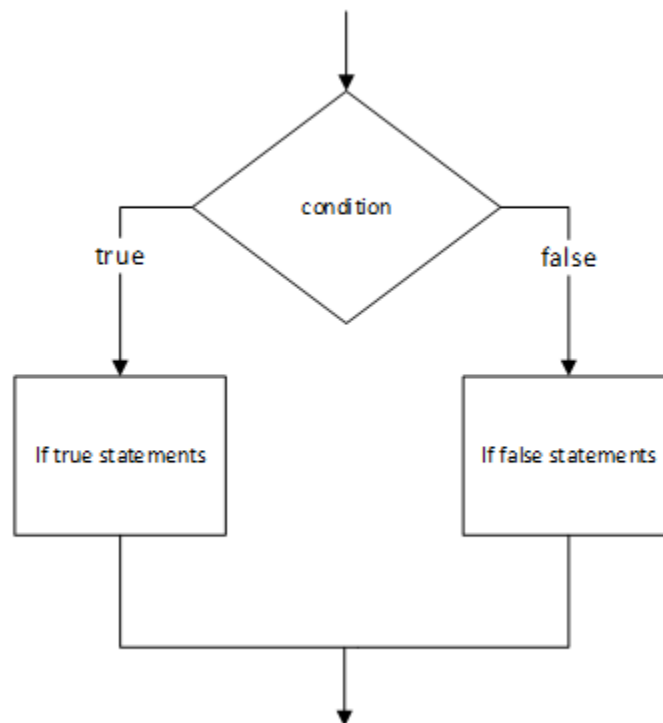
### Example

```
<?php  
$a=10;  
if($a<20) echo "Hello";  
?>
```

//Output : Hello

### If-else

If-else is a two way branching statement. A set of code is executed if the given condition is true and another set of code is executed if the condition is false.



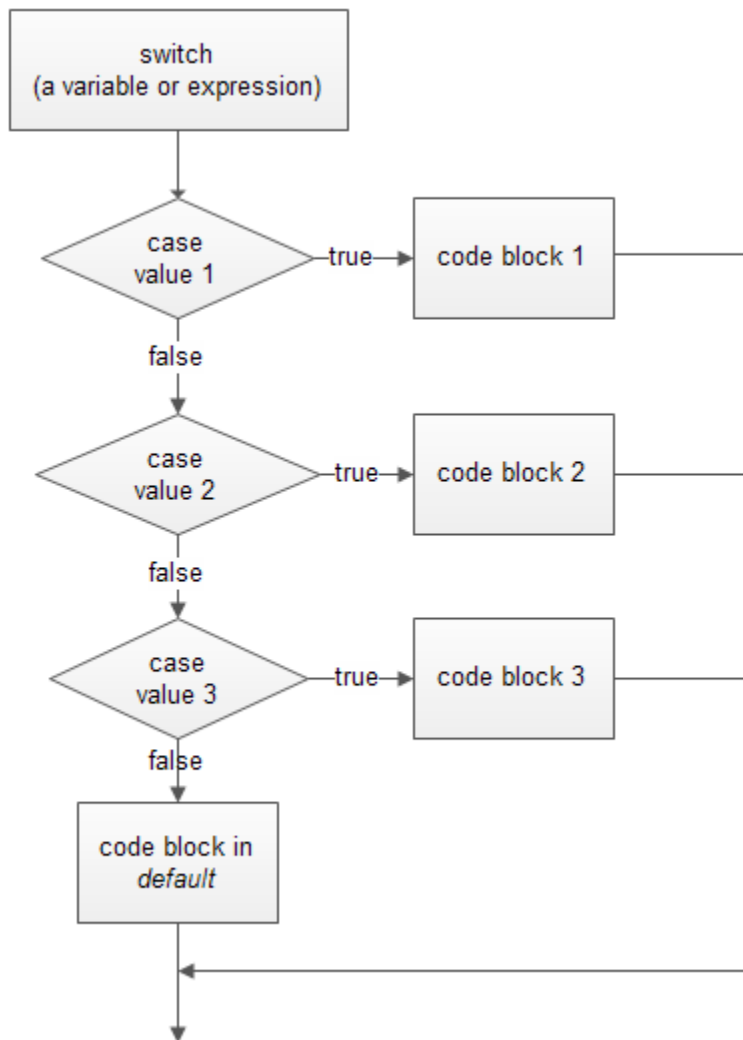
### Example

```
<?php  
$a=30;  
if($a<20) echo "Hello";  
else echo "World";  
?>
```

//output: World

### 13. Explain Switch case in PHP

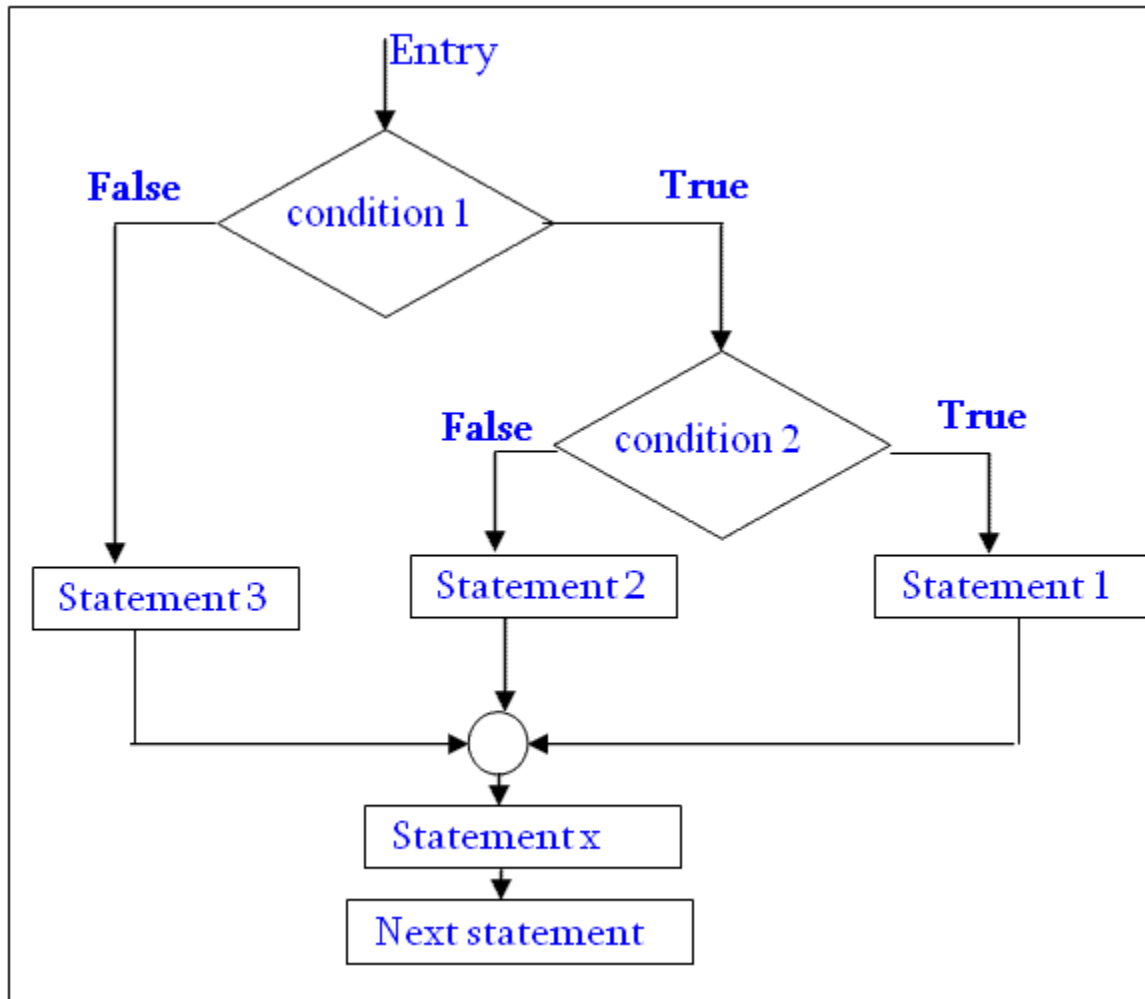
Switch case is a multi-way branching statement. PHP switch statement compares a variable or an expression against many different values and executes a code block based on the value it equals to.



```
<?php
$x = 3;
switch($x){
    case 0: echo 'x is equal 0';
    break;
    case 1: echo 'x is equal 1';
    break;
    case 2: echo 'x is equal 2';
    break;
    case 3: echo 'x is equal 3';    //output is
    break;                        // x is equal to 3
}
```

#### 14. Explain nested if statement in PHP

if statement can be nested in another if statement. Nested is useful when we want to check a condition in another conditions. Nested-if is a multi branch selection.



Example

```
if($a>10)
{
    If($b<10)
    echo "both a and b are less than 10";
    else
    echo "both a and b are not less than 10";

else
{
//code
}
```

## 15. Explain while and do-while in PHP

while is used to perform a task repeatedly based on some condition. It is an entry controlled loop.

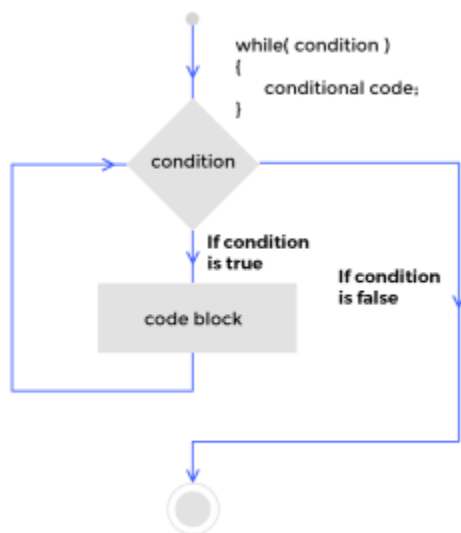
Syntax

```
while(condition)
{
    //Statements
}
```

Example

```
<?php
$i=0;
while($i<5)
{
    echo $i;
    $i++;
}
```

Block diagram



### Do-while

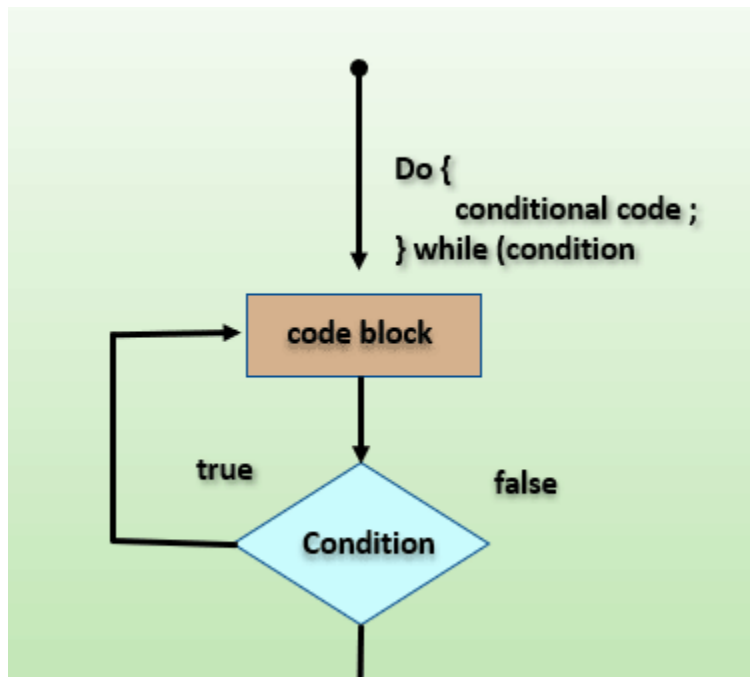
Do while is an exit controlled loop. It is also used to perform a task repeatedly based on some condition. The code inside do-while loop is executed at least once.

Syntax



```
$i=0;  
do  
{  
echo $i;  
$i++  
}while($i<5);
```

### Block diagram



### 16. Explain for loop in PHP

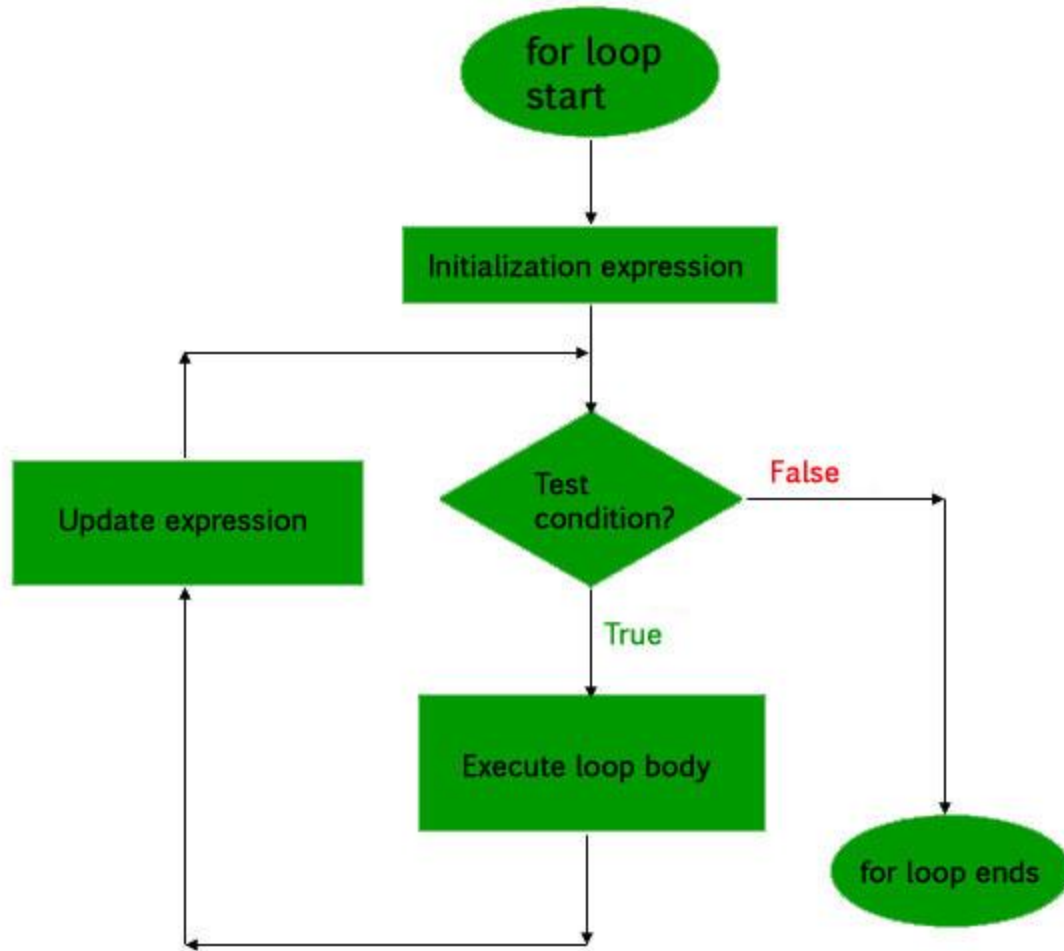
When the number of iterations to be performed is known in advance, for loop is used. For loop is an entry-controlled loop. There are three main parameters to the code, namely the initialization, the test condition and the counter.

#### Syntax

```
for (initialization expression; test condition; update expression) {  
    // code to be executed  
}
```

#### Example

```
<?php  
for($i=0;$i<10;$i++)  
{  
echo I;  
}
```



### 17. Explain foreach in PHP

The foreach construct provides an easy way to iterate over arrays. foreach works only on arrays and objects

```
foreach (array_expression as $value)
    statement
```

```
<?php
$arr = array(1, 2, 3, 4);
foreach ($arr as &$value) {
    echo $value;
}
```

### 18. Explain the use of continue statement in PHP

Continue state is used within loop structure. It is used to skip the rest of current iteration and begins the next iteration

Example

```
<?php
for ($i = 0; $i < 5; ++$i) {
    if ($i == 2)
        continue
    print "$i\n";
}
?>
```

Output

```
0
1
3
4
```

## 19. Explain break statement in PHP

Break statement is used inside the conditional statements or loops and it is used to exit the loop

Example

```
<?php
for ($i = 0; $i < 5; ++$i) {
    if ($i == 2)
        break;
    print "$i\n";
}
?>
```

Output

```
0
1
```

## 20. Explain strings and some string functions in PHP

String is a sequence of characters enclosed within double quotes.

Example

```
<?php
$a="hello";
?>
```

### String functions

(i) strlen() function returns the length of a string.

```
<?php
```

```
echo strlen("Hello world!"); // outputs 12
```

```
?>
```

(ii) `str_word_count()` function counts the number of words in a string.

```
<?php
```

```
echo str_word_count("Hello world!"); // outputs 2
```

```
?>
```

(iii) `strrev()` function reverses a string.

```
<?php
```

```
echo strrev("Hello world!"); // outputs !dlrow olleH
```

```
?>
```

(iv) `str_replace()` function replaces some characters with some other characters in a string.

```
<?php
```

```
echo str_replace("world", "Dolly", "Hello world!"); // outputs Hello Dolly!
```

```
?>
```

## **21. Explain how to create array in PHP**

Array is a subscripted variable used to hold more than one value of similar data type. Its size is fixed.

Array is created using `array()`.

Example

```
<?php
```

```
$cars = array("Volvo", "BMW", "Toyota");
```

```
echo count($cars); //output is 3
```

```
?>
```

## 22. Explain different types of arrays in PHP

In PHP, there are 3 types of arrays

- Indexed arrays - Arrays with a numeric index
- Associative arrays - Arrays with named keys
- Multidimensional arrays - Arrays containing one or more arrays

### Indexed array

Each element of the array has an numeric index

Example

```
<?php
```

```
$cars[0] = "Volvo";
```

```
$cars[1] = "BMW";
```

```
$cars[2] = "Toyota";
```

```
?>
```

### Associative array

Each element has named key as its index

Example

```
$age['Peter'] = "35";
```

```
$age['Ben'] = "37";
```

```
$age['Joe'] = "43";
```

### Multi dimensional array

A multidimensional array is an array containing one or more arrays. PHP supports multidimensional arrays that are two, three, four, five, or more levels deep

## 23. Explain some array functions in PHP

(i) `array_keys()` Returns all the keys of an array

- (ii) `array_merge()`      Merges one or more arrays into one array
- (iii) `array_pop()`      Deletes the last element of an array
- (iv) `array_push()`      Inserts one or more elements to the end of an array
- (v ) `array_reverse()`      Returns an array in the reverse order
- (vi) `array_search()`      Searches an array for a given value and returns the key
- (vii) `array_values()`      Returns all the values of an array
- (viii) `count()`      Returns the number of elements in an array
- (ix) `key()`      Fetches a key from an array
- (x) `sort()`      Sorts an indexed array in ascending order

24. What is a constant in PHP? Give example. How to create a constant in PHP

A constant is an identifier (name) for a simple value. The value cannot be changed during the script.

To create a constant, use the `define()` function.

`define(name, value)`

Example

```
define("greetings", "have good day");
```

## UNIT II

### **1. Define function. How do you create function in PHP?**

A function is a block of statements that can be used repeatedly in a program. A function will be executed by a call to the function.

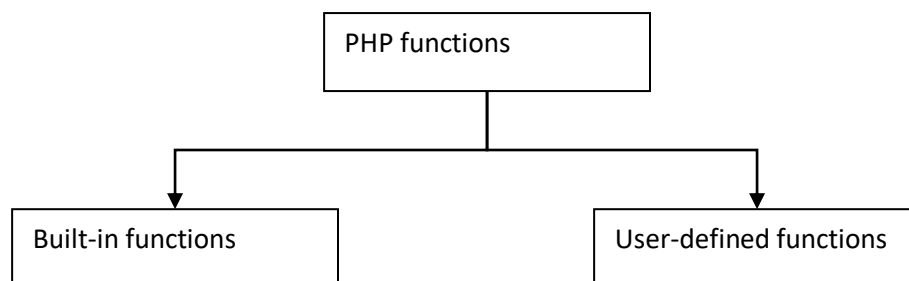
In PHP a function is created using the following syntax

```
function functionName() {  
  
    //code to be executed;  
  
}
```

#### Example

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

### **2. Explain different types of function in PHP**



- Built-in functions : PHP provides us with huge collection of built-in library functions. These functions are already coded and stored in form of functions.
- User Defined Functions : PHP allows us to create our own customised functions called the user-defined functions.

The user-defined functions can be of different types

- (i) functions with no arguments, no return types
- (ii) functions with arguments but no return types
- (iii) functions with no arguments but with return types
- (iv) functions with both arguments and return type

**(i) function with no arguments and no return types**

Example

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

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

**(ii) function with arguments but no return types**

```
<?php
function add($a, $b) {
    echo $a+$b;
}

add(3,6); // call the function & output is 9
?>
```

**(iii) function without arguments but with return type**

```
<?php
function greetings() {
    return "god is great";
}

greetings(); // call the function output- god is great
?>
```

**(iv) function with both argument and return type**

```
<?php
function add($a, $b) {
```



```
    return $a+$b;
}
```

```
$c=add(3,6); //call the function
```

```
Echo $c; // output is 9
?>
```

### 3. How do you read data from web pages in PHP?

There are two methods, used to send data from client(browser) to server

(i) GET Method -

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

(ii) POST Method - The POST method transfers information via HTTP headers.

Example

```
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 upto 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.

Example

```
<html>
  <body>

    <form action = "<?php $_PHP_SELF ?>" method = "GET">
      Name: <input type = "text" name = "name" />
    </body>
  </html>

<?php
if( $_GET["name"])
    echo$_GET['name'];
```

## \$ 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.

```
<html>
  <body>

    <form action = "<?php $_PHP_SELF ?>" method = "POST">
      Name: <input type = "text" name = "name" />
    </body>
</html>

<?php
if( $_POST["name"])
    echo $_POST['name'];
```

## **4. Explain \$\_REQUEST variable in PHP**

The variable \$\_REQUEST contains the contents of both \$\_GET, \$\_POST and \$\_COOKIE. PHP \$\_REQUEST variable can be used to get the result from form data sent with both the GET and POST methods.

```
?php
    if( $_REQUEST["name"]) {
        echo "Welcome ". $_REQUEST['name'];
    }
?>
<html>
  <body>

    <form action = "<?php $_PHP_SELF ?>" method = "POST">
      Name: <input type = "text" name = "name" />
           <input type = "submit" />
    </form>
  </body>
</html>
```

## 5. Differentiate POST and GET methods

POST	GET
Values not visible in the URL	Values visible in the URL
Has not limitation of the length of the values since they are submitted via the body of HTTP	Has limitation on the length of the values usually 255 characters. This is because the values are displayed in the URL. Note the upper limit of the characters is dependent on the browser.
Has lower performance compared to Php_GET method due to time spent encapsulation the Php_POST values in the HTTP body	Has high performance compared to POST method due to the simple nature of appending the values in the URL.
Supports many different data types such as string, numeric, binary etc.	Supports only string data types because the values are displayed in the URL
Results cannot be book marked	Results can be book marked due to the visibility of the values in the URL

## 6. How to read data from text field in a web page or HTML form?

### Reading data from text field

Text field is used to get a single line of text input from the user.

It is a HTML input type.

```
<form method="post">
<label>Please enter your Name:</label>
<input type="text" name="username"/><br>
<input name="form" type="submit" value="Submit"/><br>
</form>
```

The above HTML code create as shown below

**Please enter your Name:**

To read the data from text field in PHP, following code is used

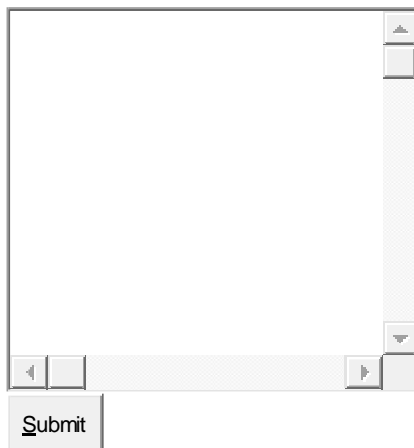
```
<?php
$name= $_POST['username'];
?>
```

## 7. How to retrieve data from textarea in a web page or HTML form?

Textarea in a input control used to receive multi-line text input from user.  
The following HTML code creates textarea in HTML form

```
<form action="" method="post">
<label>Additional Comments:</label><br>
<textarea cols="35" rows="12" name="comments">
</textarea><br>
<input type="submit" name="button" value="Submit"/></form>
```

Additional Comments:

A screenshot of a web browser displaying a form. The form has a label 'Additional Comments:' followed by a large, empty text area with a light gray border and a vertical scrollbar on the right. Below the text area is a 'Submit' button. The text area is currently empty.

This is the PHP code to retrieve the data entered into the textarea.

```
<?php
$comments= $_POST['comments'];
?>
```

## 8. How to retrieve data from radio button in a web page or HTML form?

Radio button is used to select one choice from the multiple choices given.

The following HTML code creates three radio buttons and group them together to choose the type of card with the help of radio buttons

```
<form action="" method="post">
<input type="radio" name="card" value="MasterCard" checked="checked">MasterCard<br>
<input type="radio" name="card" value="Visa">Visa<br>
<input type="radio" name="card" value="American Express">American Express<br>
<input type="submit" name="button" value="Submit"/></form>
```

Which Credit Card would you like to use?

- ☒ MasterCard
- ☐ Visa
- ☐ American Express

Now, the following php code is used to read the value from radio button

```
<?php
if(isset($_POST['card'])){
    $card_type= $_POST ['card'];
} else {
    $card_type = "No Button Selected";
}
?>
```

## 9. How to retrieve data from checkbox in a web page or HTML form?

Checkboxes are to select more than one option from a list of choices. check boxes allow multiple options to be selected.

The following HTML code creates 3 three check boxes

```
<form method="post">
<input type="checkbox" name="Java">Snorkeling<br>
<input type="checkbox" name="Cplusplus">C++<br>
<input type="checkbox" name="Python">Python<br>
<input type="submit" name="button" value="Submit"/></form>
```

☐ **Java**  
☐ **C++**  
☐ **Python**

The PHP code to retrieve the data from checkbox is given below

```

<?php

$java= $_POST['Java'];
$cplus= $_POST[Cplus];
$python= $_POST['Python'];

echo"The activities you want to do are' . '<br>';
if(isset($java)){
echo $java. '<br>';
}
if(isset($cplus)){
echo '$cplus'. '<br>';
}
if(isset($python)){
echo '$python'. '<br>';
}
?>

```

## 10. How to retrieve data from listbox in a web page or HTML form?

Listbox allows to select one or multiple options from the list of items

The following HTML code creates a listbox consisting of 3 fruits which allows multiple options to be selected.

```

<form method="post">
<select name="foods[]" size="3" multiple="multiple">
<option value="Tomatoes">Tomatoes</option>
<option value="Cucumbers">Cucumbers</option>
<option value="Celery">Celery</option>
</select><br>
<input type="submit" name="submit" value="Submit"/>
</form>

```

Tomatoes Cucumbers Celery
---------------------------------

Submit
--------

The following PHP reads the data from listbox

```
<?php

$choices= $_POST['foods'];
if(isset($choices)) {
foreach ($choices as $key => $value)
{
echo $value . '<br>';
}
}
else
{
echo "No item selected";
}
?>
```

## 11. What is HTML form? Explain HTML form processing in PHP

HTML forms are used to get input from user using different controls such as textfield, textarea, button, checkbox, radio button, listbox, etc.

HTML forms are used to send the user information to the server and returns the result back to the browser

To create a HTML form, form tag should be used.

Syntax

```
<form action=".....php" method="GET">
```

```
//input controls and submit button
```

```
</form>
```

### Attributes of form

**Action** - It specifies the location to which the form data has to be sent when the form is submitted

**Method** - It specifies the HTTP method that is to be used when the form is submitted. The possible values are get and post. If get method is used, the form data are visible to the users in the url. Default HTTP method is get.

## Controls used in forms

- **Textbox:** Textbox allows the user to provide single-line input, which can be used for getting values such as names, search menu and etc.
- **Textarea:** Textarea allows the user to provide multi-line input, which can be used for getting values such as an address, message etc.
- **DropDown:** Dropdown or combobox allows the user to provide select a value from a list of values.
- **Radio Buttons:** Radio buttons allow the user to select only one option from the given set of options.
- **CheckBox:** Checkbox allows the user to select multiple options from the set of given options.
- **Buttons:** Buttons are the clickable controls that can be used to submit the form

## 12. Explain how to create a html form?

To create a HTML form, form tag should be used.

Syntax

```
<form action=".....php" method="GET">
```

```
//input controls and submit button
```

```
</form>
```

### Example

```
<html>
<body>
<form method="GET">
<input type="text" name="username"/><br/>
<input type="submit" value="click"/><br/>
</form>
</body>
</html>
```

## 13. Explain form processing in PHP

The data entered in HTML forms are sent to server using \$\_GET[], \$\_POST[] and \$\_REQUEST

**\$\_GET[]:** It is used to retrieve the information from the form control through the parameters sent in the URL. It takes the attribute given in the url as the parameter.



`$_POST[]`: It is used to retrieve the information from the form control through the HTTP POST method. It takes name attribute of corresponding form control as the parameter.

`$_REQUEST[]`: It is used to retrieve an information which are contained in `$_GET`, `$_POST` and `$_COOKIE`

`isset()`: This function is used to determine whether the variable or a form control is having a value or not.

### Example

```
<html>
<body>
<form method="POST">
<input type="text" name="username"/><br/>
<input type="submit" value="click"/><br/>
</form>

<?php

if (isset($_POST['username']))
{
echo $_POST['username']

}
?>
</body>
</html>
```

### 14. Explain `isset()` function in PHP with example

The `isset()` function is an inbuilt function in PHP which is used to determine if the variable is declared and its value is not equal to NULL

#### Syntax

```
boolean isset($var)
```

It returns TRUE if `$var` exists and its value not equal to NULL and FALSE otherwise.

#### Example

```
<?php
    $str = "God is great";

// Check value of variable is set or not
if(isset($str)) {
    echo "Value of variable is set";
}
```

```

else {
    echo "Value of variable is not set";
}
?>

```

## 15. Explain how to get browser power in PHP?

The `get_browser()` function in PHP is an inbuilt function which is used to tell the user about the browser's capabilities.

The `get_browser()` function looks up the user's `browscap.ini` file and returns the capabilities of the user's browser.

The `user_agent` and the `return_array` are passed as parameters to the `get_browser()` function and it returns an object or an array with information about the user's browser on success, or `FALSE` on failure.

```
get_browser(user_agent, return_array)
```

**user\_agent** : It is an optional parameter which specifies the name of an HTTP user agent. Default is the value of `$HTTP_USER_AGENT`.

**return\_array** : It is an optional parameter which returns an array instead of an object if it is set to `True`.

```

<?php
echo $_SERVER['HTTP_USER_AGENT'];
//using get_browser() to display capabilities of the user browser
$mybrowser = get_browser();
print_r($mybrowser);
?>

```

### Output

```

[parent] => IE 6.0
[platform] => WinXP
[netclr] => 1
[browser] => IE
[version] => 6
[majorver] => 6
[minorver] => 0
=> 2
[frames] => 1
[iframes] => 1

```

## UNIT – III

### 1. What is OOP? List out advantages of OOP?

- OOP stands for Object-Oriented Programming
- Object-oriented programming is about creating objects that contain both data and functions

Object-oriented programming has several advantages

1. OOP is faster and easier to execute
2. OOP provides a clear structure for the programs
3. OOP helps to reduce redundancy of coding and makes the code easier to maintain, modify and debug
4. OOP makes it possible to create full reusable applications with less code and shorter development time.

### 2. What is a Class? How to create a class in PHP?

A class is defined by using the class keyword, followed by the name of the class and a pair of curly braces ({}). All its properties and methods goes inside the braces:

Syntax:

```
<?php
class Fruit {
    // code goes here...
}
?>
```

**Example:**

```
<?php
class Fruit {
    // Properties
    public $name;

    // Methods
    function set_name($name) {
        $this->name = $name;
    }
    function get_name() {
        return $this->name;
    }
}
?>
```

There is one property (\$name) and two methods set\_name() and get\_name().

### 3. What is an object? How to create an object in PHP?

We can create multiple objects from a class. Each object has all the properties and methods defined in the class, but they will have different property values.

Objects of a class is created using the **new** keyword.

In the example below, \$apple and \$banana are instances of the class Fruit:

```
<?php
class Fruit {
    // Properties
    public $name;
    public $color;

    // Methods
    function set_name($name) {
        $this->name = $name;
    }
    function get_name() {
        return $this->name;
    }
}

$apple = new Fruit();
$banana = new Fruit();
$apple->set_name('Apple');
$banana->set_name('Banana');

echo $apple->get_name();
echo "<br>";
echo $banana->get_name();
?>
```

### 4. What is constructor? Give example

A constructor is a special function which is automatically called when an object is created. The basic syntax of a constructor is

```
public function __construct()
{
    /code to initialize parameters
}
```

### Example

```
<?php
class Animal
{
    public $name = "No-name animal";
    public function __construct($name)
    {
        $this->name = $name;
    }
}

$animal = new Animal("Bob the Dog");
echo $animal->name;
?>
```

### 5. What is destructor? Give example

A destructor is called when the object is destroyed. The syntax of a destructor is given below

```
public function __destruct()
{
    //code to destruct
}
```

### Example

```
<?php
class Animal
{
    public $name = "No-name animal";
    public function __construct($name)
    {
        // $this->name = $name;
    }

    public function __destruct()
    {
        //code for destructor
    }
}

?>
```

## 6. Explain encapsulation in PHP with example

Encapsulation is a concept of wrapping up or binding up related data members and methods in a single module known as encapsulation And hiding the essential internal property of that module known as data abstraction.

### Example

```
<?php
class person
{
    public $name;
    public $age;

    function __construct($n, $a)
    {
        $this->name=$n;
        $this->age=$a;
    }

    public function setAge($ag)
    {
        $this->ag=$ag;
    }

    public function display()
    {
        echo "welcome ".$this->name."<br/>";
        return $this->age-$this->ag;
    }
}
$person=new person("Pankaj",25);
$person->setAge(20);
echo "You are ".$person->display()." years old";
?>
```

## 7. Explain inheritance in PHP with example

**Inheritance** is a mechanism of extending an existing class by inheriting a class we create a new class with all functionality of that existing class, and we can add new members to the new class. When we inherit one class from another we say that inherited class is a subclass and the class who has inherit is called parent class. We declare a new class with additional keyword **extends**.

```
<?php
class BaseClass
```

```

{
    function add()
    {
        //code
    }
}
class child extends BaseClass
{
    //code in subclass
}
}
?>

```

### 8. Explain interface in PHP with example.

An Interface allows the users to create programs, specifying the public methods that a class must implement.

- An interface consists of methods that have no implementations, which means the interface methods are abstract methods.
- All the methods in interfaces must have public visibility scope.
- Interface is defined using the keyword interface
- Interfaces cannot be instantiated

```

<?php
interface MyInterface{
    public function methodA();
}
?>

```

```

<?php
class MyClass implements MyInterface{
    public function methodA() {
        // method A implementation
    }
}
?>

```

### 9. Explain how an interface extends another interface in PHP. (or) Explain inheritance in interfaces in PHP

In PHP, one interface can extend another interface.

#### Example

```
<?php
```

```
interface MyInterface1{  
    public function methodA();  
}
```

```
interface MyInterface2 extends MyInterface1 {  
    public function methodB();  
}
```

```
?>
```

## 10. Explain abstract class in PHP with example

- Abstract classes are the classes in which at least one method is abstract.
- Abstract classes in PHP are declared with the help of abstract keyword
- An abstract class can contain abstract as well as non abstract methods.
- An instance of abstract class cannot be created.

```
<?php
```

```
// Abstract class example in PHP  
abstract class MyAbstractClass  
{  
  
    // This is abstract function  
    Abstract function printdata();  
  
}
```

```
?>
```

In general, the abstract classes will be extended by other classes to provide implementation for the abstract methods

```
class MySubclass extends MyAbstractClass {  
  
    function printdata() {  
        echo "\n Derived class printdata function";  
    }  
}  
$b1 = new MySubclass; //object for subclass
```



## 11. Compare Interface and Abstract class

Interface	Abstract class
Interface support multiple inheritance	Abstract class does not support multiple inheritance
Interface does'n Contains Data Member	Abstract class contains Data Member
Interface does'n contains Cunstructors	Abstract class contains Cunstructors
An interface Contains only incomplete member (signature of member)	An abstract class Contains both incomplete (abstract) and complete member
An interface cannot have access modifiers by default everything is assumed as public	An abstract class can contain access modifiers for the subs, functions, properties
Member of interface can not be Static	Only Complete Member of abstract class can be Static

## 12. What is static class? Explain static class in PHP with example

The variables and methods that are declared and defined within a class are to be declared as static with the use of static keyword, so that they can be used without instantiating the class first. This means that a class variable can be accessed without a specific instance, it also means that there will only be one version of this variable. Another consequence is that a static method cannot access non-static variables and methods since these require an instance of the class.

To access the static class and it's method use the following syntax:

ClassName::MethodName();

```
<?php
class MyStaticClass {
    public static $x = 13;
    public static function myfunction()
    {
        echo MyStaticClass::$x;
    }
}

MyStaticClass::MethodName();    //output 13
```

## UNIT IV

### 1. List out file handling functions in PHP

- (i) `fopen()` – PHP `fopen()` function is used to open a file. First parameter of `fopen()` contains name of the file which is to be opened and second parameter tells about mode in which file needs to be opened
- (ii) `fread()` — After file is opened using `fopen()` the contents of data are read using `fread()`.
- (iii) `fwrite()` – New file can be created or text can be appended to an existing file using `fwrite()` function
- (iv) `fclose()` – file is closed using `fclose()` function. Its argument is file which needs to be closed

### 2. Discuss in detail different file handling functions with example

**fopen()** – PHP `fopen()` function is used to open a file. First parameter of `fopen()` contains name of the file which is to be opened and second parameter tells about mode in which file needs to be opened

Example

```
<?php
$file = fopen("demo.txt", 'w');
?>
```

Files can be opened in any of the following **file modes**

“w” – Opens a file for write only. If file not exist then new file is created and if file already exists then contents of file is erased.

“r” – File is opened for read only.

“a” – File is opened for write only. File pointer points to end of file. Existing data in file is preserved.

“w+” – Opens file for read and write. If file not exist then new file is created and if file already exists then contents of file is erased.

“r+” – File is opened for read/write.

“a+” – File is opened for write/read. File pointer points to end of file. Existing data in file is preserved. If file is not there then new file is created.

“x” – New file is created for write only.

**fread()** — After file is opened using fopen() the contents of data are read using fread(). It takes two arguments. One is file pointer and another is file size in bytes

```
<?php
$filename = "demo.txt";
$file = fopen( $filename, 'r' );
$size = filesize( $filename );
$filedata = fread( $file, $size );
?>
```

**fwrite()** – New file can be created or text can be appended to an existing file using fwrite() function. Arguments for fwrite() function are file pointer and text that is to written to file. It can contain optional third argument where length of text to written is specified

```
<?php
$file = fopen("demo.txt", 'w');
$text = "Hello world\n";
fwrite($file, $text);
?>
```

**fclose()** – file is closed using fclose() function. Its argument is file which needs to be closed

```
<?php
$file = fopen("demo.txt", 'r');
//some code to be executed
fclose($file);
?>
```

### 3. Explain different file modes in PHP

Files can be opened in any of the following **file modes**

“w” – Opens a file for write only. If file not exist then new file is created and if file already exists then contents of file is erased.

“r” – File is opened for read only.

“a” – File is opened for write only. File pointer points to end of file. Existing data in file is preserved.

“w+” – Opens file for read and write. If file not exist then new file is created and if file already exists then contents of file is erased.

“r+” – File is opened for read/write.

“a+” – File is opened for write/read. File pointer points to end of file. Existing data in file is preserved. If file is not there then new file is created.

“x” – New file is created for write only.

#### Example

```
<?php
$file = fopen("demo.txt", 'w');
?>
```

### 4. What is MYSQL?

- MySQL is a open, relational database system
- MySQL is a database system that runs on a server
- MySQL is ideal for both small and large applications
- MySQL is very fast, reliable, and easy to use
- MySQL uses standard SQL
- MySQL compiles on a number of platforms
- MySQL is free to download and use
- MySQL is developed, distributed, and supported by Oracle Corporation

### 5. What is a table?

A table is a collection of related data, and it consists of columns and rows.

### 6. List out different ways by which one can connect MYSQL from PHP

There are three ways of working with PHP and MySQL:

- MySQLi (object-oriented) (the "i" stands for improved)
- MySQLi (procedural)
- PHP Data Objects (PDO)

**7. Explain how do you connect to mysql database from PHP using mysqli(object oriented method ) method**

MySQLi(object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
// Create connection
$conn = new mysqli($servername, $username, $password);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
// Create database
$sql = "CREATE DATABASE myDB";
$conn->query($sql)

$sql = "CREATE TABLE Employee (empid VARCHAR(4) UNSIGNED, name
VARCHAR(30) NOT NULL)";

$conn->query($sql)
$sql = "INSERT INTO Employee (empid, name) VALUES ('1001', 'Doe')";

$conn->query($sql)
$sql = "SELECT id, empid, name FROM Employee";
$result = $conn->query($sql);
$conn->close();

?>
```

**8. Explain how do you connect to mysql database from PHP using mysqli(procedural) method**

MySQLi(procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
```

```
// Create connection
$conn = mysqli_connect($servername, $username, $password);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
$sql = "CREATE TABLE Employee (empid VARCHAR(5) UNSIGNED,name
VARCHAR(30))";
mysqli_query($conn, $sql)

$sql = "INSERT INTO Employee (empid, name) VALUES ('1001', 'Doe')";
mysqli_query($conn, $sql)
$sql = "SELECT id, empid, name FROM Employee";
mysqli_query($conn, $sql)
$conn->close();

?>
```

## 9. Explain how do you connect to mysql database from PHP using PHP Data Objects(PDO) method

PDO

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
try {
    $conn = new PDO("mysql:host=$servername, $username, $password);
    // set the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $sql = "CREATE DATABASE myDBPDO";
    $conn->exec($sql);}
catch(PDOException $e) {echo "Connection failed";}

$sql = "CREATE TABLE Employee (empid VARCHAR(5) UNSIGNED, name
VARCHAR(30))";
$conn->exec($sql);
$sql = "INSERT INTO Employee (empid, name) VALUES ('1001', 'Doe')";
$conn->exec($sql);
$sql = "SELECT id, empid, name FROM Employee";
```

```
$conn->exec($sql);  
$conn=null;
```

```
?>
```

## 10. What is SQL?

Structured Query Language. SQL lets us access and manipulate databases.

- SQL can execute queries against a database
- SQL can retrieve data from a database
- SQL can insert records in a database
- SQL can update records in a database
- SQL can delete records from a database
- SQL can create new databases
- SQL can create new tables in a database
- SQL can create stored procedures in a database
- SQL can create views in a database
- SQL can set permissions on tables, procedures, and views

## 11. Explain different SQL queries that can be used with databases

(i) Select query – used to select records from a table

Example

```
Select * from customers;
```

(ii) insert query – used to insert record into a table

Example

```
Insert into customers customerId, customerName values ('1001', 'Stephen')
```

(iii) update query – used to update the values of a record

Example

```
Update customers set customerName="Stephen Raj" where customerId='1001'
```

(iv) delete – used to delete a record from a table

```
delete customer where customerId='1001'
```

## 12. What is a session variable?

A PHP session stores data on the server rather than user's computer.

In a session based environment, every user is identified through a unique number called session identifier or SID.

This unique session ID is used to link each user with their own information on the server like emails, posts, etc

### **13. Explain how to start a session in PHP?**

Starting a session in PHP

To begin a new session, simply call the PHP session\_start() function. It will create a new session and generate a unique session ID for the user.

```
<?php  
  
// Starting session  
  
session_start();  
  
?>
```

The session\_start() function first checks to see if a session already exists by looking for the presence of a session ID. If it finds one, i.e. if the session is already started, it sets up the session variables and if doesn't, it starts a new session by creating a new session ID

### **14. How to store, access, remove and destroy Session?**

**Step -1** Before storing data into session, it has to be created using session\_start() method.

**Step-2** Values can be stored as key-value pairs as given in the example

#### **Example**

```
<?php  
  
// Starting session  
  
session_start();  
  
  
  
// Storing session data  
  
$_SESSION["firstname"] = "Peter";  
  
$_SESSION["lastname"] = "Parker";
```



?>

Step -3 Values in a Session variable can be access as shown below

```
<?php
```

```
// Starting session
```

```
session_start();
```

```
// Accessing session data
```

```
echo 'Hi, ' . $_SESSION["firstname"] . ' ' . $_SESSION["lastname"];
```

?>

Step – 4 To remove a session data

To remove certain session data, simply unset the corresponding key of the \$\_SESSION

```
<?php
```

```
// Starting session
```

```
session_start();
```

```
// Removing session data
```

```
if(isset($_SESSION["lastname"])){
```

```
    unset($_SESSION["lastname"]);
```

```
}
```

?>

**Step 5 – Destroying a session**

To destroy a session completely, simply call the session\_destroy() function. This function does not need any argument and a single call destroys all the session data.

```
<?php
```

```
// Starting session
```

```
session_start();
```

```
// Destroying session
```

```
session_destroy();
```

```
?>
```

### **15. What is a cookie? Explain with example**

Cookies are text files stored on the client computer and they are kept of use tracking purpose.

- Server script sends a set of cookies to the browser. For example name, age, or identification number etc.
- Browser stores this information on local machine for future use.
- When next time browser sends any request to web server then it sends those cookies information to the server and server uses that information to identify the user

Cookies are usually set in an HTTP header. A PHP script that sets a cookie might send headers

```
HTTP/1.1 200 OK
Date: Fri, 04 Feb 2000 21:03:38 GMT
Server: Apache/1.3.9 (UNIX) PHP/4.0b3
Set-Cookie: name=xyz; expires=Friday, 04-Feb-07 22:03:38 GMT;
            path=/; domain=tutorialspoint.com
Content-Type: text/html
```

PHP provided setcookie() function to set a cookie.

### **16. How to create a cookie?**

PHP provided setcookie() function to set a cookie.

```
setcookie(name, value, expire, path, domain, security);
```

- Name – This sets the name of the cookie and is stored in an environment variable called HTTP\_COOKIE\_VARS. This variable is used while accessing cookies.
- Value – This sets the value of the named variable and is the content that you actually want to store.

- **Expiry** – This specifies a future time in seconds since 00:00:00 GMT on 1st Jan 1970. After this time cookie will become inaccessible. If this parameter is not set then cookie will automatically expire when the Web Browser is closed.
- **Path** – This specifies the directories for which the cookie is valid. A single forward slash character permits the cookie to be valid for all directories.
- **Domain** – This can be used to specify the domain name in very large domains and must contain at least two periods to be valid. All cookies are only valid for the host and domain which created them.
- **Security** – This can be set to 1 to specify that the cookie should only be sent by secure transmission using HTTPS otherwise set to 0 which means cookie can be sent by regular HTTP.

```
<?php
    setcookie("name", "John Watkin", time()+3600, "/", "", 0);
    setcookie("age", "36", time()+3600, "/", "", 0);
?>
```

## 17. How to access a cookie?

PHP provides many ways to access cookies. Simplest way is to use either `$_COOKIE` or `$HTTP_COOKIE_VARS` variables.

Example

```
<?php
    setcookie("name", "John Watkin", time()+3600, "/", "", 0);
    echo $_COOKIE["name"]. "<br />";

    /* is equivalent to */
    echo $HTTP_COOKIE_VARS["name"]. "<br />";

?>
```

## 18. How to use `isset()` with cookie?

`isset()` function is used to check if a cookie is set or not.

```
<?php
    if( isset($_COOKIE["name"]))
        echo "Welcome " . $_COOKIE["name"] . "<br />";

    else
        echo "Sorry... Not recognized" . "<br />";
```

?>

### 19. How to delete a cookie?

to delete a cookie you should call `setcookie()` with the name argument and set the cookie with a date that has already expired

Example

```
setcookie( "name", "", time()- 60, "/", "", 0);
```

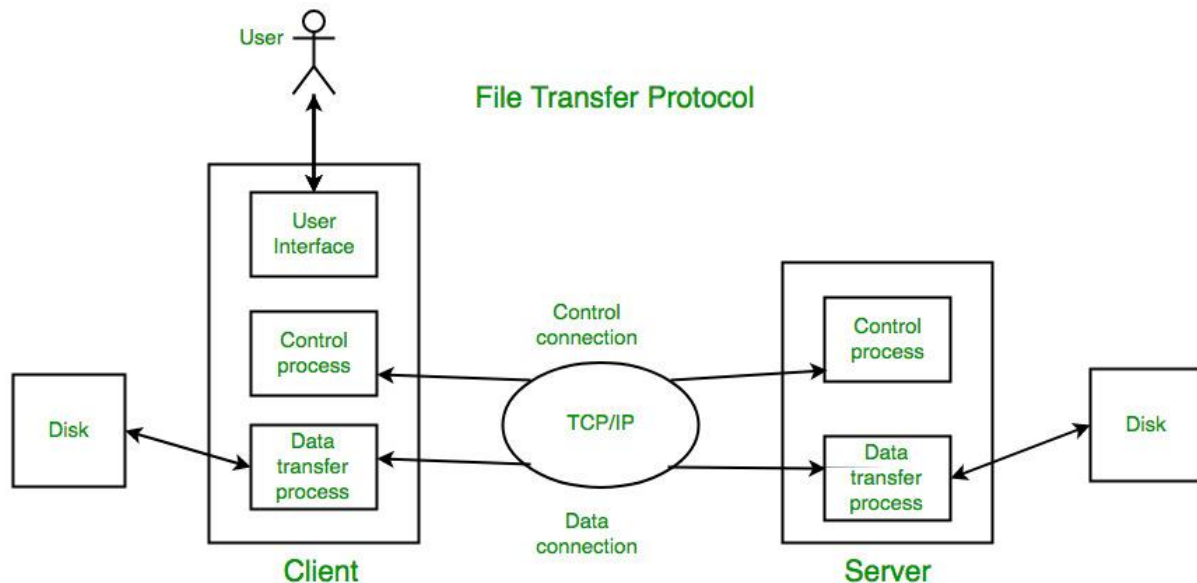
### 20. What is FTP?

File Transfer Protocol (FTP). The File Transfer Protocol (FTP) is a standard network protocol used for the transfer of computer files between a client and server on a computer network. FTP is built on a client-server model architecture using separate control and data connections between the client and the server.

### 21. Give the architecture of FTP with a diagram

File Transfer Protocol (FTP). The File Transfer Protocol (FTP) is a standard network protocol used for the transfer of computer files between a client and server on a computer network. FTP is built on a client-server model architecture using separate control and data connections between the client and the server.

File Transfer Protocol(FTP) is an application layer protocol which moves files between local and remote file systems. It runs on the top of TCP, like HTTP. To transfer a file, 2 TCP connections are used by FTP in parallel: control connection and data connection.



### Control connection

For sending control information like user identification, password, commands to change the remote directory, commands to retrieve and store files, etc., FTP makes use of control connection. The control connection is initiated on port number 21.

### Data Connection

For sending the actual file, FTP makes use of data connection. A data connection is initiated on port number 20.

### FTP Session

When a FTP session is started between a client and a server, the client initiates a control TCP connection with the server side. The client sends control information over this. When the server receives this, it initiates a data connection to the client side. Only one file can be sent over one data connection.

## 22. Explain FTP functions in PHP

- The FTP functions give the client access to file servers through the File Transfer Protocol (FTP).
- PHP FTP functions are used to access FTP files servers through a PHP script.
- FTP functions can be used to login, close connections, upload, rename, delete, download, get info from the file server.

## PHP FTP functions

ftp_alloc()	Allocates space for a file to be uploaded
ftp_cdup()	Changes to the parent directory
ftp_chdir()	Changes the current directory on a FTP server
ftp_chmod()	Set permissions on a file via FTP
ftp_close()	Closes an FTP connection
ftp_connect()	Opens an FTP connection
ftp_delete()	Deletes a file on the FTP server
ftp_exec()	Requests execution of a command on the FTP server
ftp_fget()	Downloads a file from the FTP server and saves to an open file
ftp_fput()	Uploads from an open file to the FTP server
ftp_get_option()	Retrieves various runtime behaviours of the current FTP stream
ftp_get()	Downloads a file from the FTP server
ftp_login()	Logs in to an FTP connection
ftp_mdtm()	Returns the last modified time of the given file
ftp_mkdir()	Creates a directory
ftp_nb_continue()	Continues retrieving/sending a file (non-blocking)
ftp_nb_fget()	Retrieves a file from the FTP server and writes it to an open file (non blocking)
ftp_nb_fput()	Stores a file from an open file to the FTP server (non-blocking)
ftp_nb_get()	Retrieves a file from the FTP server and writes it to a local file (non-blocking)
ftp_nb_put()	Stores a file on the FTP server (non-blocking)
ftp_nlist()	Returns a list of files in the given directory
ftp_pasv()	Turns passive mode on or off
ftp_put()	Uploads a file to the FTP server
ftp_pwd()	Returns the current directory name
ftp_quit()	Alias of ftp_close
ftp_raw()	Sends an arbitrary command to an FTP server
ftp_rawlist()	Returns a detailed list of files in the given directory
ftp_rename()	Renames a file or a directory on the FTP server
ftp_rmdir()	Removes a directory
ftp_set_option()	Set miscellaneous runtime FTP options
ftp_site()	Sends a SITE command to the server
ftp_size()	Returns the size of the given file
ftp_ssl_connect()	Opens an Secure SSL-FTP connection
ftp_systype()	Returns the system type identifier of the remote FTP server

## UNIT-V

### 1. What is Ajax?

- Ajax is abbreviated as Asynchronous Javascript and XML.
- It is new technique used to create better, faster and more interactive web systems or applications.
- Ajax uses asynchronous data transfer between the Browser and the web server.
- This technique is used to make internet faster and user friendly.

### 2. What are Ajax applications?

Browser based applications and platform independent applications are used by Ajax.

### 3. What are the advantages of Ajax?

- Bandwidth utilization – It saves memory when the data is fetched from the same page.
- More interactive
- Speeder retrieval of data

### 4. What are the disadvantages of Ajax?

- AJAX is dependent on Javascript. If there is some Javascript problem with the browser or in the OS, Ajax will not support
- Ajax can be problematic in Search engines as it uses Javascript for most of its parts.
- Source code written in AJAX is easily human readable. There will be some security issues in Ajax.
- Debugging is difficult
- Increases size of the requests
- Slow and unreliable network connection.
- Problem with browser back button when using AJAX enabled pages.

### 5. What are all the technologies used by Ajax?

- JavaScript
- XMLHttpRequest
- Document Object Model (DOM)
- Extensible HTML (XHTML)
- Cascading Style Sheets (CSS)

### 6. What are all the features of Ajax?

- Live data binding
- Client-side template rendering
- Declarative instantiation of client components
- AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS and Java Script.

- Conventional web application transmit information to and from the sever using synchronous requests.
- With AJAX, contents are updated asynchronously.
- XML is commonly used as the format for receiving server data, although any format, including plain text, can be used.
- Ajax uses XHTML for content, CSS for presentation, along with Document Object Model and JavaScript for dynamic content display.
- AJAX is a web browser technology independent of web server software.
- A user can continue to use the application while the client program requests information from the server in the background.
- Intuitive and natural user interaction. Clicking is not required, mouse movement is a sufficient event trigger.
- Data-driven as opposed to page-driven

#### 7. What is JSON in Ajax?

- JSON is abbreviated as JavaScript Object Notation.
- JSON is a safe and reliable data interchange format in JavaScript, which is easy to understand for both users and machines.

#### 8. What are the difference between AJAX and Javascript?

AJAX	Javascript
AJAX sends request to the server and does not wait for the response. It performs other operations on the page during that time	JavaScript make a request to the server and waits for response
AJAX does not require the page to refresh for downloading the whole page	JavaScript manages and controls a Web page after being downloaded
AJAX minimizes the overload on the server since the script needs to request once	JavaScript posts a request that updates the script every time

#### 9. What are the protocols used by Ajax?

- HTTP's GET or POST
- XMLHttpRequest for placing a request with the web server
- Uses JSON to communicate between the client and server



- UED or URL encoded data

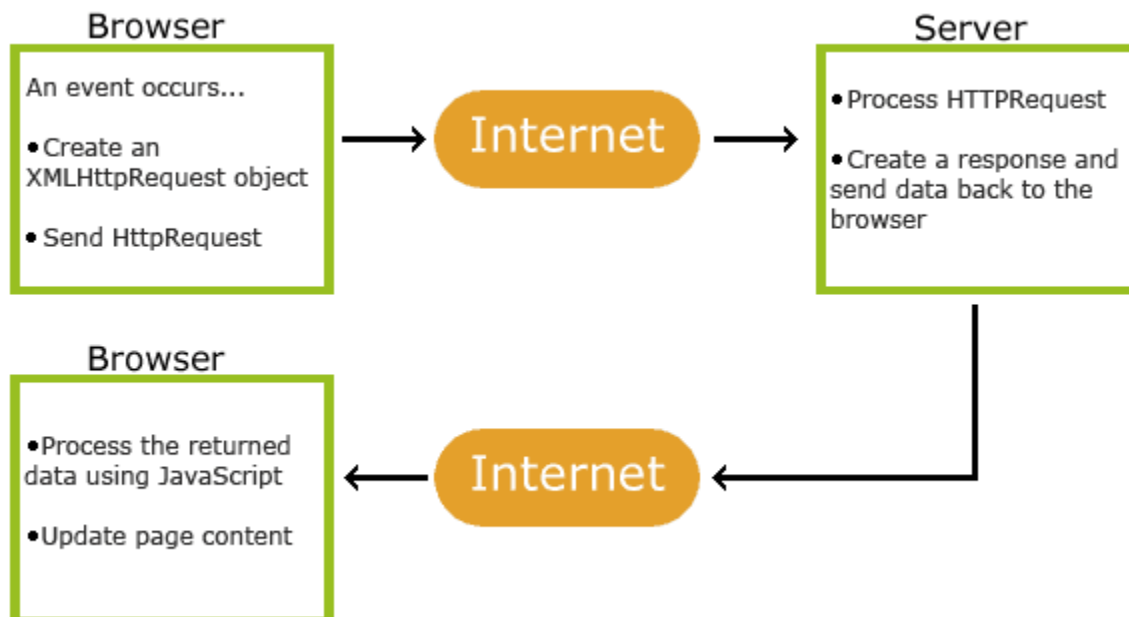
#### 10. What are the types of post back in Ajax?

- Synchronous Postback
- Asynchronous Postback

#### 11. How does Ajax work?

AJAX communicates with the server using XMLHttpRequest object.

1. User sends a request from the UI and a javascript call goes to XMLHttpRequest object.
2. HTTP Request is sent to the server by XMLHttpRequest object.
3. Server interacts with the database using JSP, PHP, Servlet, ASP.net etc.
4. Data is retrieved.
5. Server sends XML data or JSON data to the XMLHttpRequest callback function.
6. HTML and CSS data is displayed on the browser.



#### 12. What are the properties of XMLHttpRequest?

- onReadyStateChange - It is called whenever readystate attribute changes.
- readyState - It represents the state of the request.
- responseText - It returns response as text.

- responseXML - It returns response as XML.
- status - It returns the status number of a request.
- statusText - It returns the details of status.

### **13. What are the important methods of XMLHttpRequest?**

- abort() - It is used to cancel the current request.
- getAllResponseHeaders() - It returns the header details.
- getResponseHeader() - It returns the specific header details.
- open() - It is used to open the request.
- send() - It is used to send the request.
- setRequestHeader() - It adds request header.

### **14. What are the types of open() method used for XMLHttpRequest?**

open(method, URL) - It opens the request specifying get or post method and URL.

open(method, URL, async) - It is same as above but specifies asynchronous or not.

open(method, URL, async, username, password) - It is same as above but specifies the username and password.

### **15. What are the types of send() method used for XMLHttpRequest?**

send() - It sends get request

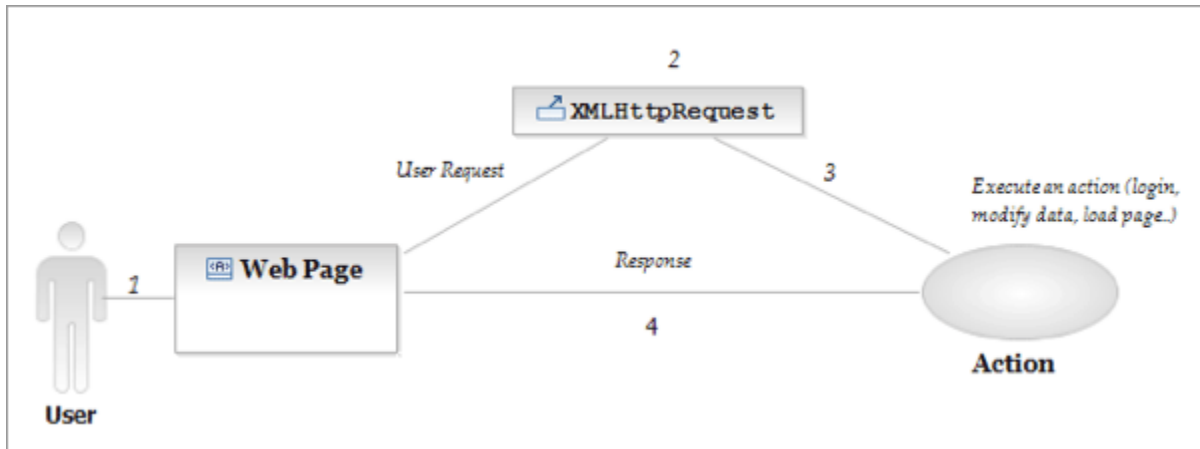
send(string) - It sends post request.

### **16. What is the role of the callback function in AJAX?**

The callback function passes a function as a parameter to another function. If we have to perform various AJAX tasks on a website, then we can create one function for executing XMLHttpRequest and a callback function to execute each AJAX task.

### **17. Elaborate how Ajax works with details of all steps involved.**

With an HTTP request, a web page can make a request to, and get a response from a web server, without reloading the page. The user will stay on the same page, and he or she will not notice that scripts request pages, or send data to a server in the background



### Step 1- Create XMLHttpRequest

```

function ajaxFunction()
{
var xmlhttp;
try
{
xmlhttp=new XMLHttpRequest();
}
catch (e){}

```

### Step-2. Sending request to the server

To send off a request to the server, we use the open() method and the send() method.

```

xmlhttp.open("GET","time.asp",true);
xmlhttp.send(null);

```

### Step -3 Writing server side script

The responseText will store the data returned from the server. Here we want to send back the current time. The code in "time.asp" looks like this:

```

<%
response.expires=-1
response.write(time)
%>

```

### Step – 4. Consuming the response

we need to consume the response received and display it to the user.

```

xmlhttp.onreadystatechange=function()
{
if(xmlhttp.readyState==4)
{
document.myForm.time.value=xmlhttp.responseText;
}
}

```

```
xmlHttp.open("GET","time.asp",true);  
xmlHttp.send(null);  
}
```

## **REFERENCES:**

<https://www.javatpoint.com/php-tutorial>  
<https://www.javatpoint.com/ajax-tutorial>  
<https://career.guru99.com/>  
[https://www.w3schools.com/js/js\\_ajax\\_intro.asp](https://www.w3schools.com/js/js_ajax_intro.asp)  
<https://www.geeksforgeeks.org/php/>  
<https://tutorialspoint.dev/>  
<https://www.tutorialrepublic.com/>