COMPUTER SCIENCE

Grade: XII

WEB TECHNOLOGY II

REFERENCE NOTE

NEB Important Questions for Computer Science XII

Unit 3- Web Technology II

1. What is web technology? Explain different data types used in JavaScript.
2. ***Differentiate between client side scripting and Server-side scripting.***
3. What is event handling in JavaScript? Give one example.
4. ***What is JQuery? Write its features and Write a program to displaying a message "Hello Class 12" using JQuery.***
5. What is JavaScript? How can you add JavaScript to an HTML page? Describe with example.
6. What is jQuery? Write its features.
7. ***What is Java Script function? Explain with calling function examples.***
8. What is PHP? Write the advantages of PHP.
9. Explain the different operators used in PHP.
10. ***What are the features of PHP? Write a PHP program to display the largest among three numbers.***
11. Define SQL? Write down the SQL queries to create a database, create a table in the database, insert data in the table, and query the data to display it.
12. Write down the server side script to create a database, connect with it, create a table and insert data in it.

Practical Programs

1. Write a java Script to display 1 to 10 using for loop, while loop and do while loop.
2. Create a Page with a button with value "Computer" on clicking the button your page should" Computer Science".
3. ***Write a JavaScript program to display "Welcome Class-12" using onload event.***
4. Design a form with username, address, e-mail, password and submit button. Validated the form using jQuery.
5. ***Design a form with username and password and submit button. Write a PHP code to get value of username and password using a) $_POST variable and b) $_GET variable.***
6. Write a PHP program to check if a string is a palindrome or not.
7. ***Write a Java Script program to calculate the factorial of a given number.***
8. Write a java script program to input three number and find largest one using java and PHP.

Web technology

Introduction

Web Technology is the tools and techniques which enables two or more computing devices to communicate over a network i.e. Internet.

Web Technology consist of two words, the web refers to the World Wide Web generally known as World Wide Web.

WWW is the cyber space containing webpages, documents, and any other resources which are identified and located with the help of their URLs.

Technology refers to the tools and techniques that makes these resources available on the Web such as, web browsers to view content of web, Programming languages and frameworks for the development of websites, Database to store data at back end, protocols for communicating on the web, multimedia elements etc.

Web development

It is the process of designing and developing website which are hosted through internet or intranet. The process of developing web can range from developing static page to a complex such as web based application social media sites, E-commerce.

Web development includes web design, web content development, client side scripting, server side scripting, web engineering etc. Since, web development consists of several inter-related task which can be accomplish by different types of developer who focuses on different aspect of web creation.

Frontend	**Backend**
Client Side	Server Side
Website Design	Database
HTML	PHP
CSS	Java
JavaScript	Python
AJAX	Ruby
UI/UX	Servers
Some UI technologies	Some Backend technologies
Ruby	.NET

Scripting Language:

- JavaScript is a scripting language. A scripting language is a lightweight programming language. JavaScript code can be inserted into any HTML page, and it can be executed by all types of web browsers.

- JavaScript is used to make web pages interactive. It runs on your visitor's computer and doesn't require constant downloads from your website. JavaScript and Java are completely different language, both in concept and design.

Unlike HTML, JavaScript is case sensitive-therefore watch your capitalization closely when you write JavaScript statements, create or call variables, objects and functions.

A scripting language is a programming language designed for integrating and communicating with other programming language. It is used in server side as well as client side. Some of the most widely used scripting language are JavaScript, VBScript, PHP, Perl, Python, Ruby etc.

Application Areas of Scripting Language

- Dynamic web applications
- Game application and Multimedia
- Scripting like Ruby and Python are used in statistics and research
- To automate the process
- Used to create plug ins and extensions for existing applications.

Some of the popular Language are

1. **Python:** Python is a very popular and demanding programming language now because it is suitable for developing very simple to complex applications. It is also used to connect database systems. Python has simple English like syntax.
2. **Perl: P**ractical **E**xtraction and **R**eporting **L**anguage, first released in 1987 is a powerful language with advance features. Perl is a stable, cross platform programming language. It is a general-purpose programming language originally developed for text manipulation and now used for a wide range of tasks including system administration, web development, network programming, GUI development, and more.
3. **Ruby:** Ruby is a dynamic, open source programming language with a focus on simplicity and productivity. It has an elegant syntax that is natural to read and easy to write.
4. **Bash:** A Bash script is a plain text file which contains a series of commands. It is widely available on various operating systems and is a default command interpreter on most GNU/Linux systems.
5. **Node.js:** Node.js is used to create dynamic page contents. It can create, open, read, write, delete, and close file on the server.
6. **ASP.net:** It is used to develop dynamic websites, web applications, and web services. ASP.NET is a web application framework developed and marketed by Microsoft to allow programmers to build dynamic web sites.
7. **VBScript:** Visual Basic Script (VBScript) is an open source web programming language developed by Microsoft. It is superset of JavaScript and adds optional static typing class based object oriented programming VBScript is lightweight scripting language. As VBScript is only used by IE browsers JavaScript is preferred over VBScript.
8. **JavaScript:** JavaScript is most well-known and widely used scripting language for web pages. All java script files are stored in file having is extension. JavaScript and Java programming language are two different things: JavaScript is generally used for making websites interactive and dynamic.

9. **jQuery:** JQuery is a JavaScript library that simplifies writing code and enables rapid web development. JQuery simplifies HTML document traversing and manipulation, and browser event handling. The main concept of jQuery is "write less, do more".
10. **PHP:** Hypertext Preprocessor (PHP) is widely used scripting language. PHP scripts are executed on the server. It is used to manage dynamic content, databases, session tracking and building e-commerce sites. It is integrated with a popular database MySQL.

S.N.	Scripting Language	S.N.	Programming Language
1	Scripting language are platform-specific.	1	Programming language are platform-independent.
2	Most of the scripting languages are interpreted.	2	Most of the programming language are complied.
3	Scripting language runs slower than programming language.	3	Programming languages runs faster than scripting language.
4	Developer has to write less code compared to programming language.	4	Developer has to write much code compared to scripting language.
5	We cannot create standalone application with scripting language only.	5	We can create standalone application with programming language only.
6	Examples, JavaScript, VBScript, Python, Perl, ASP etc.	6	Examples, C, C++, Java etc.
7	Scripting languages run inside other programs. It is dependent on other programming language.	7	It is not dependent on other programs to run. It is independent.

Server side and client side programming

Client-Side Scripting programming

Client-side scripting is performed to generate a code that can run on the client side i.e (front end) browser without needing the server-side (back end) processing. Basically, client-side scripts are placed inside an HTML document.

The client-side scripting can be used to layout the content of the web. For example, when a user makes a request through web browser for a webpage to the server, it just sent the HTML and CSS as plain text, and the browser interprets and renders the content of web in the client end (user).

Client-side scripting is designed to run as a scripting language which can be executed by web browser. Front end developer is someone who design and develop client side of a website. Generally he or she works in user interface (UI) of a website. Front end developer must be at least fluent in three different languages i.e. HTML, CSS, JavaScript whereas, there are several other libraries which can be used for front end development.

Advantages of Client side Scripting:

1. Immediate response to users
2. Enhance the appearance of websites
3. More responsive design and interaction with the user
4. It does not need to send requests to the server hence reduces the load on server
5. Loading time of a page is faster
6. Reduces the network traffic
7. It is reusable

Disadvantages of Client side Scripting:

1. All browsers may not support client side script
2. The code is not secure because anyone can look at the code
3. Users can disable the client side scripts so required content may not be displayed
4. Database connection is not possible with client side scripting.
5. Dynamic content cannot be displayed.

Server-Side Scripting programming

Server-side scripting also known as back-end runs on the server where the application is hosted. Server-side is used to serve content depending upon the user request. Back end helps to create dynamic web based application that allows user to interact and communicate with the application. Back end language also helps to connect front end with data base. So that, User can store and retrieve data as per the requirement. Back-end developer is responsible for server-side programming. Some of the popular server-side (back-end) scripting language are ASP, JavaScript (using SSJS (Server-side JavaScript e.g. node.js), Perl, PHP, Ruby, Python etc.

The client-side scripting emphasizes making the interface of the web application or website (UI) more appealing and functional. Whereas, server-side scripting emphasizes on data accessing methods, error handling and fast processing etc.

Advantages

1) You can create dynamic pages.
2) Can connect to database that resides on the web server.
3) Can access files from the server to client browser Users are not able to block the contents from server
4) The actual code is not visible to the client
5) Authentication and verification of user is possible
6) It supports many databases like MySQL Oracle,
7) Efficient storage and delivery of information
8) Customized user experience.
9) Controlled access to content
10) Notification and communication
11) Users do not need to download plug-in like java or flash
12) The content management system (CMS) makes editing simpler

Disadvantages of Server side Scripting

1. The scripting software has to be installed on the server.
2. The script takes more time to execute.
3. It requires a large amount of memory space in the server computer.
4. Implementation cost is high
5. If a lot of users are accessing server data, server may crash due to overload

Note: *Full-stack developer understand both Front end and back end development process. They can accomplish entire project. Full stack developer must have expertise in client site and server site Scripting language. Moreover, he/she has a great knowledge of integrating database with the application.*

```
<html>
<head>
<title> Inline javascript Example</title>
<body>
        <form>
<center><input type="button" value="ClickMe" onclick="alert('Button Clicked: ')">
        </center></form>
</body>
</html>
```

Comparison between Client-side and Server-side Scripting

This section elaborates the fundamental differences between client-side and server-side scripts:

1. The client-side script is executed at the front-end in the client's browser while the server-side script is executed at the back end with a web server.
2. The client-side script is visible to the user of the web browser while the server-side script is hidden.
3. The client-side script is not secure while the server-side script is secure.
4. The client-side script does not need to interact with the server while the server-side script needs a web server to be processed.
5. The client-side script is executed on a local computer while the server-side script is executed on a remote computer.
6. The client-side script has a faster response time than the server-side script.
7. Client-side script is executed after the browser receives the web pages sent by the server while the server-side script cannot execute the client-side script.
8. The client-side script cannot connect with the database while the server-side script can connect with the database present on the server-side.
9. The client-side script cannot access the files while the server-side script can access and manipulate the files present at the webserver.
10. The client-side script helps create interactive web pages while the server-side script helps create web pages with dynamic data.

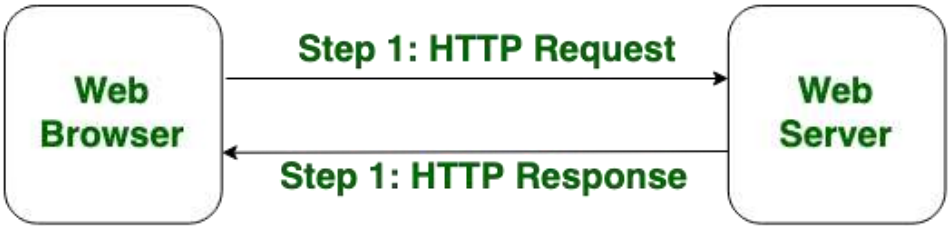
S.N.	Server Side Scripting	S.N.	Client side scripting
1	The Server executes the server side scripting.	1	The client (web browser) executes the client side scripting.
2	It can be used to connect database on the web server.	2	It cannot be used to connect to the database on the web server.
3	Server side scripting response is slower.	3	Client side scripting response is faster.
4	Source code is not visible to the user so it is secure.	4	Source code is visible to user so it is relatively insecure.
5	Users can not block server side scripting.	5	Users can block client side scripting
6	Examples: PHP, ASP.NET, ASP, Ruby on rails, Python, Perl.	6	Examples: JavaScript, VBScript
7	It does not depend on client. Any server side technology can be used.	7	It depends on browser and version of the browser.

Internet Technology

The Internet is the global system of interconnected computer networks that uses the Internet protocol suite to communicate between networks and devices. It is a network of networks that consists of private, public, academic, business, and government networks of local to global scope, linked by a broad array of electronic, wireless, and optical networking technologies. The Internet carries a vast range of information resources and services, such as the inter-linked hypertext documents and applications of the World Wide Web, electronic mail, telephony, and file sharing.

Static Website:

Static Web pages are very simple. It is written in languages such as HTML, JavaScript, CSS, etc. **Static websites** are the websites that doesn't change the content or layout dynamically with every request to the web server. Static websites display exactly the same information whenever anyone visits it. User sees the updated content of Static Website only when a web author manually updates them with a **text editor** or any web editing tool used for creating websites. Static webpages do not have to be simple plain text. They can feature multiple design and even videos.



Static Web Page

Features of Static Websites:

- Static websites are the simplest kind of website that you can build.
- Every viewer will see the exactly same text, multimedia design or video every time he/she visits the website until you alter that page's source code.
- Static websites are written with the help of **HTML and CSS.**
- The only form of interactively on a static website is **hyperlink**.
- Static website can be used for the information that doesn't change substantially over months or even years.
- Static pages are easy and simple to understand, secure, less prone to technology errors and breakdown and easily visible by search engines.
- **HTML** was the first tool with which people had begun to create static web pages.
- Static websites provide **flexibility**.
- **Lightweight**.
- Static websites perform **faster** and **well** than dynamic ones.

Advantages of Static websites

- Static websites are highly cost-effective for publishing.
- They require less coding and technical knowledge.
- Static websites are easier to make.
- Static websites are quick to develop.
- Static websites are cheap to host.
- A static website contains data which is immutable.
- Static websites are beginner level. A programmer with knowledge of HTML, CSS, and JavaScript can build static websites.
- It's easy to create and host online.
- Static websites provide security.

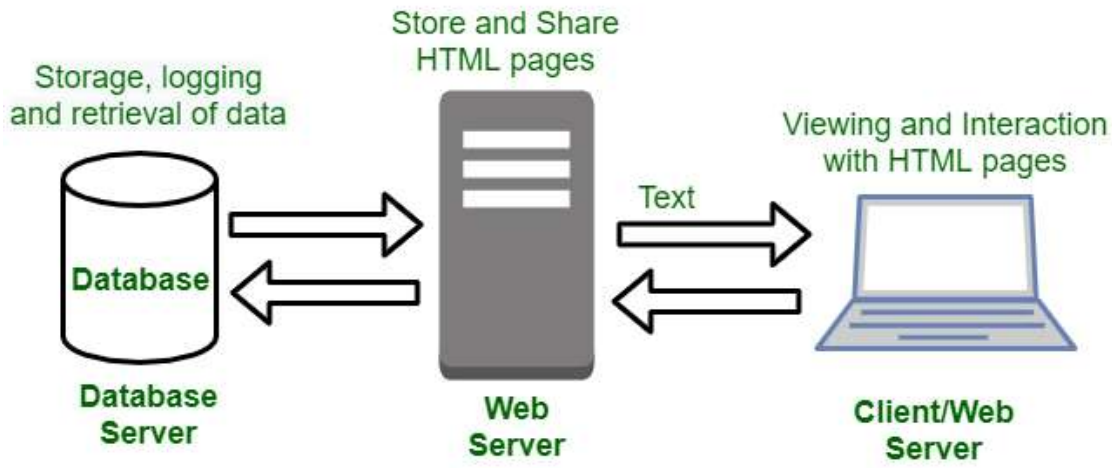
Disadvantages of Static Websites:

- Requires web development expertise to update site.
- Site not as useful for the user.
- Content can get stagnant.
- Send the same response for every request.
- Dynamic functionality works only on the client side.

Dynamic website:

Dynamic websites are those websites that changes the content or layout with every request to the webserver. These websites have the capability of producing different content for different visitors from the same source code file. There are two kinds of dynamic web pages i.e. client side scripting and server side scripting. The client-side web pages changes according to your activity on the web page. On the server-side, web pages are changed whenever a web page is loaded.

Example: login & signup pages, application & submission forms, inquiry and shopping cart pages. A Typical Architecture of dynamic website



There are different languages used to create dynamic web pages like PHP, ASP, .NET and JSP. Whenever a dynamic page loads in browser, it requests the database to give information depending upon user's input. On receiving information from the database, the resulting web page is applied to the user after applying the styling codes.

Features of dynamic webpage:

- These websites are very flexible.
- In these websites the content can be quickly changed on the user's computer without new page request to the web browser.
- In these websites the owner have the ability to simply update and add new content to the site.
- These websites are featured with content management system, e-commerce system and intranet or extranet facilities.
- Most of the dynamic web content, is assembled on the web using server-scripting languages.

Advantages of dynamic webpage:

- It provides more functional websites.
- It is very easy to update.
- It helps in the search engines because new content brings people back to the site.
- These are interactive websites because these can be customized.
- These websites can work as a system to allow staff or users to collaborate.

- It's easy to modify or update data.
- It provides a user-friendly interactive interface for users.
- Proves smooth navigation.
- provide interactive user interface
- It provides a better user experience.
- It provides real-time data.

Disadvantages of dynamic webpages:

- These types of websites are complex.
- These are more expensive to develop.
- Hosting of these websites is also costlier.
- It requires a rapid, high-end web server.
- High production costs.
- Slow to load content.
- Client will require a skilled programmer to build a dynamic website.
- Hosting a website is costly as compared to a dynamic website.
- Low speed compared to a static website

Application of Dynamic Website:

- Online booking system:
- E-commerce website.
- Voting or polls,
- Forums
- E-newsletter.

JavaScript

JavaScript is a lightweight, interpreted programming language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java. JavaScript is very easy to implement because it is integrated with HTML. It is open and cross-platform.

This tutorial has been prepared for JavaScript beginners to help them understand the basic functionality of JavaScript to build dynamic web pages and web applications.

Features

- JavaScript is a lightweight, interpreted programming language.
- Designed for creating network-centric applications.
- Complementary to and integrated with Java.
- Complementary to and integrated with HTML.
- It is case sensitive language.
- JavaScript is supportable in operating system.
- It provides good control to the users over the web browsers.

```
<script>
document.write("Hello JavaScript by JavaScript");
</script>
```

Advantages of JavaScript

1. **Less server interaction**: You can validate user input before sending the page off to the server. This saves server traffic, which means fewer loads on your server.
2. **Easy to learn:** By learning few commands and simple rules of syntax, you can easily build applications using JavaScript.

3. **Immediate feedback to the visitors:** They don't have to wait for a page reload to see if they have forgotten to enter something.
4. **Increased interactivity**: You can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.
5. **Quick Development:** Scripts can be developed in short period of time
6. **Richer interfaces:** You can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.
7. **Transmitting information:** About the users' reading habits and browsing activities to various websites. Web pages frequently do this for web analytics, ad tracking personalization or other purposes.
8. **Easy Debugging and Testing**: As JavaScript is interpreted line by line, it is easy to find error and make changes.
9. **Interactive content,** for example games, and playing audio and video.
10. **Validating input values** of a Web form to make sure that they are acceptable before being submitted to the server.

Uses of JavaScript

- Client side validation
- Dynamic drop down menus.
- Displaying date and time.
- Displaying pop-up windows and dialog boxes
- Displaying clocks.
- Event handling.
- Developing Mobile applications
- Creating web browser based games.
- Building web servers
- Adding interactivity to website.

Adding JavaScript to HTML

JavaScript, also known as JS, is one of the scripting (client-side scripting) languages, that is usually used in web development to create modern and interactive web-pages. The term "script" is used to refer to the languages that are not standalone in nature and here it refers to JavaScript which run on the client machine.

In other words, we can say that the term scripting is used for languages that require the support of another language to get executed. For example, JavaScript programs cannot get executed without the help of HTML or without integrated into HTML code.

1. Embedding code
2. Inline code
3. External file

1. Embedding code:-

To add the JavaScript code into the HTML pages, we can use the <script>.....</script> tag of the HTML that wrap around JavaScript code inside the HTML program. Users can also define JavaScript code in the <body> tag

(or we can say body section) or <head> tag

because it completely depends on the structure of the web page that the users use.

```
<html>
<head>
<title> page title</title>
<script>
document.write("Welcome to Java Script Programming");
</script>
</head>
<body>
<p>In this example we saw how to add JavaScript in the head section </p>
</body>
</html>
```

We can also define the JavaScript code in the <body> tags or body section.
Let's understand through an example.

```
<html>
<head>
<title> Body Section Example</title> </head>
<body>
<script>
document.write("Welcome to Javatpoint");
</script>
<p> In this example we saw how to add  JavaScript in the body section </p>
</body>
</html>
```

2. Inline code:-

Generally, this method is used when we have to call a function in the HTML event attributes. There are many cases (or events) in which we have to add JavaScript code directly eg., OnMover event, OnClick, etc. Let's see with the help of an example, how we can add JavaScript directly in the html without using the **<script>.... </script>** tag.

Let's look at the example.

```
<html>
<head>
<title> Inline code example</title>  </head>
<body>
<p>
<a href="#" onClick="alert('Welcome !');">Click Me</a>
<button onclick="alert('Hello Class 12')">Click Here</button>
</p>
<p> In this example we saw how to use inline JavaScript or directly in an HTML tag.
  </p>   </body>  </html>
```

3. External file:-

We can also create a separate file to hold the code of JavaScript with the (.js) extension and later incorporate/include it into our HTML document using the **src** attribute of the <script> tag. It becomes very helpful if we want to use the same code in multiple HTML documents. It also saves us from the task of writing the same code over and over again and makes it easier to maintain web pages.

In this example, we will see how we can include an external JavaScript file in an HTML document.
Let's understand through a simple example.

```
<html>
<head>
<title>Including a External JavaScript File</title>
</head>
<body>
<form>
<input type="button" value="Result" onclick="display()"/>
</form>
<script src="hello.js">
</script>
</body>
</html>
```

Now let's create separate JavaScript file = Hello.js

```
function display()
 {
alert("Hello Friends!");
}
```

JavaScript Fundamentals

Script

JavaScript statements are written within <script>…..</script>. The <script> tag alerts the browser program to start interpreting all the statements between these tags as a script. A simple syntax of your JavaScript will appear as follows:

Syntax: <script>
block of Statements
</script>

Statements:

JavaScript statements are composed of Values, Operators, Expressions, Keywords, and Comments. The program consists of many statements. The statements are executed, one by one, in the same order as they are written. It is often called JavaScript code.

```
var area = l*b;
```

JavaScript comments

The **JavaScript comments** are meaningful way to deliver message. It is used to add information about the code, warnings or suggestions so that end user can easily interpret the code.

The JavaScript comment is ignored by the JavaScript engine i.e. embedded in the browser.

There are two types of comments in JavaScript.

1. Single-line Comment
2. Multi-line Comment

1. Single line Comment

It is represented by double forward slashes (//). It can be used before and after the statement.

Let's see the example of single-line comment i.e. added before the statement.

```
<script>
// It is single line comment
document.write("hello javascript");
</script>
```

Let's see the example of single-line comment i.e. added after the statement.

```
<script>
var a=10;
var b=20;
var c=a+b;  //It adds values of a and b variable
document.write(c); //prints sum of 10 and 20
</script>
```

2. Multi line Comment

It can be used to add single as well as multi line comments. So, it is more convenient.

It is represented by forward slash with asterisk then asterisk with forward slash. For example:

```
/* your comment here  */
```

It can be used before, after and middle of the statement.

```
<script>
/* It is multi line comment.
It will not be displayed */
document.write("Example of javascript multiline comment");
</script>
```

Whitespace and Line Breaks:

JavaScript ignores spaces, tabs, and newlines that appear in JavaScript programs. To make a code readable and understandable, formatting and indenting the code by using several spaces, tabs, and newlines can be used freely in the program.

Case Sensitivity:

JavaScript is a case-sensitive language. The keywords, variables, functions names, and any other identifiers must always be typed with a consistent capitalization of letters. So the identifiers Time and TIME will convey different meanings in JavaScript.

Output:

Java Script can "display" data in different ways: It defines the ways to display the output of a given code. The output can be display by using four different ways which are listed below:

Basic Display function in JavaScript

1. Document.wirte

Using document.write( ), the output can be displayed in browser page.

```
<html>
<head>    <title> Display functions in Javascript </title>
</head>
<body>
    <h1> Output with document.write</h1>        <br>
    Hello Class 12 <br>
    <script>
            document.write(7+5)
    </script>
    <input type="button" value="Click Me" onclick="document.write(7+5)">
</body>
</html>
```

Inner HTML

Using inner HTML, the output can be displayed into the HTML element. The innerHTML property sets or returns the HTML content (inner HTML) of an element.

```
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML = "Hello JavaScript!";
</script>
```

2. Window. Alert

Using window.alert( ), the output can be displayed into an alert box. window.confirm() and window.prompt() are also other method to output into an alert box.

```
<html>
<head>        <title> Display functions in Javascript </title>
</head>
<body>
        <h1> Output Message with window alert</h1> <br>
        <script>
                alert(7+5);
        </script>
</body></html>
```

3. Console.log

Using console.log( ), the output can be displayed into the browser console. To use console of web browser, right click in web browser and select inspect. The console will display.

```
<html>
<head>        <title>Display functions in Javascript </title>
</head>
<body>
        <h1> Demo of console.log</h1> <br>
        <script>
                Console.log("Gurukul College");
                Console.log("7+5");
        </script>
</body>
</html>
```

Reserved Words:

Lists of all the reserved words in JavaScript are given in the following table. They cannot be used as Java Script variables, functions, methods, loop labels, or any object names.

Abstract	**Else**	**instance of**	**Switch**
boolean	enum	int	synchronized
break	export	interface	this
byte	extends	long	throw
Case	False	Native	Throws
Catch	Final	New	transient
Char	Finally	Null	True
Class	Float	Package	Try
Const	For	Private	Type of
Continue	Function	Protected	Var
Debugger	Go to	Public	Void
Default	If	Return	Volatile
Delete	Implements	Short	While
Do	Import	Static	With
Double	In	Super	

JavaScript Data Types

JavaScript provides different **data types** to hold different types of values. There are two types of data types in JavaScript.

1. Primitive data type
2. Non-primitive (reference) data type

JavaScript is a **dynamic type language**, means you don't need to specify type of the variable because it is dynamically used by JavaScript engine. You need to use **var** here to specify the data type. It can hold any type of values such as numbers, strings etc. For example:

```
var a=40;        //holding number
var b="Rahul";   //holding string
```

JavaScript primitive data types

There are five types of primitive data types in JavaScript. They are as follows:

Data Type	**Description**
1. String	represents sequence of characters e.g. var str="hello";
2. Number	represents numeric values e.g. var a=100;
3. Boolean	represents Boolean value either false or true
4. Undefined	represents undefined value. E.g. var address="Bharaptur" address= undefined
5. Null	represents null i.e. no value at all.

JavaScript non-primitive (reference) data types

The non-primitive data types are as follows:

Data Type	**Description**
1. Object	represents instance through which we can access members
2. Array	represents group of similar values
3. RegExp	represents regular expression

JavaScript Variable

1. JavaScript Local variable
2. JavaScript Global variable

A **JavaScript variable** is simply a name of storage location. There are two types of variables in JavaScript: local variable and global variable.

There are some rules while declaring a JavaScript variable (also known as identifiers).

1. Name must start with a letter (a to z or A to Z), underscore (_), or dollar($) sign.
2. After first letter we can use digits (0 to 9), for example value1.
3. Reserved words cannot be used as names.
4. JavaScript variables are case sensitive, for example x and X are different variables.

Correct JavaScript variables

1. var x = 10;
2. var _value="Hari";

Incorrect JavaScript variables

1. var 123=30;
2. var *aa=320;

Example of JavaScript variable

Let's see a simple example of JavaScript variable.

```
<html>
<head>
<title> Example of Variable </title>
<body>
<script>
var x = 10;
var y = 20;
var z=x+y;
document.write(z);
</script>
</body>
</html>
```

JavaScript Operators

JavaScript operators are symbols that are used to perform operations on operands. For example:

```
var sum=10+20;
```

Here, + is the arithmetic operator and = is the assignment operator.

There are following types of operators in JavaScript.

1. Arithmetic Operators
2. Comparison (Relational) Operators
3. Bitwise Operators
4. Logical Operators
5. Assignment Operators
6. Special Operators
7. Unary Operator
8. String Operator
9. Conditional Operator

JavaScript Arithmetic Operators

Arithmetic operators are used to perform arithmetic operations on the operands. The following operators are known as JavaScript arithmetic operators.

Operator	Description	Example
1) +	Addition	10+20 = 30
2) -	Subtraction	20-10 = 10
3) *	Multiplication	10*20 = 200
4) /	Division	20/10 = 2
5) %	Modulus (Remainder)	20%10 = 0
6) ++	Increment	var a=10; a++; Now a = 11
7) --	Decrement	var a=10; a--; Now a = 9

Examples

```
<html>
<body>
<h2>Arithmetic Operations</h2>
<p>A typical arithmetic operation takes two numbers (or expressions) and produces a new number.</p>
<p id="Hello"></p>
<script>
let a = 3;
let x = (100 + 50) * a;
document.getElementById("Hello").innerHTML=x;    // document.write(x); OR  window.alert(x);
</script>
</body>
</html>
```

JavaScript Comparison Operators

The JavaScript comparison operator compares the two operands. The comparison operators are as follows:

Operator	Description	Example
1) ==	Is equal to	10==20 = false
2) !=	Not equal to	10!=20 = true
3) >	Greater than	20>10 = true
4) >=	Greater than or equal to	20>=10 = true
5) <	Less than	20<10 = false
6) <=	Less than or equal to	20<=10 = false

```
<html>
<body>
<h1>JavaScript Operators</h1>
<h2>The + Operator</h2>
<p>The + operator concatenates (adds) strings.</p>
<script>
let text1 = "Rajesh";
let text2 = "Hamal";
let text3 = text1 + " " + text2;
document.write(text3);
</script>
</body>
</html>
```

```
<html>
<body>
<h1>JavaScript Arithmetic</h1>
<h2>The += Operator</h2>
<script>
        var x = 10;
        x += 5;
document.write(x);
</script>
</body>
</html>
```

JavaScript Bitwise Operators

The bitwise operators perform bitwise operations on operands. The bitwise operators are as follows:

Operator	Description	Example
1) &	Bitwise AND	(10==20 & 20==33) = false
2) |	Bitwise OR	(10==20 | 20==33) = false
3) ^	Bitwise XOR	(10==20 ^ 20==33) = false
4) ~	Bitwise NOT	(~10) = -10
5) <<	Bitwise Left Shift	(10<<2) = 40
6) >>	Bitwise Right Shift	(10>>2) = 2
7) >>>	Bitwise Right Shift with Zero	(10>>>2) = 2

JavaScript Logical Operators

The following operators are known as JavaScript logical operators.

Operator	Description	Example
1) &&	Logical AND	(10==20 && 20==33) = false
2) ||	Logical OR	(10==20 || 20==33) = false
3) !	Logical Not	!(10==20) = true

JavaScript Assignment Operators

The following operators are known as JavaScript assignment operators.

Operator	Description	Example
1) =	Assign	10+10 = 20
2) +=	Add and assign	var a=10; a+=20; Now a = 30
3) -=	Subtract and assign	var a=20; a-=10; Now a = 10
4) *=	Multiply and assign	var a=10; a*=20; Now a = 200
5) /=	Divide and assign	var a=10; a/=2; Now a = 5
6) %=	Modulus and assign	var a=10; a%=2; Now a = 0

JavaScript Special Operators

The following operators are known as JavaScript special operators.

Operator	Description
1) (?:)	Conditional Operator returns value based on the condition. It is like if-else.
2) ,	Comma Operator allows multiple expressions to be evaluated as single statement.
3) Delete	Delete Operator deletes a property from the object.
4) In	In Operator checks if object has the given property
5) Instanceof	checks if the object is an instance of given type
6) new	creates an instance (object)
7) typeof	checks the type of object.
8) Void	it discards the expression's return value.

JavaScript Bitwise Operators

Bit operators work on 32 bits numbers.

Any numeric operand in the operation is converted into a 32 bit number. The result is converted back to a JavaScript number.

Operator	Description	Example	Same as	Result	Decimal
&	AND	5 & 1	0101 & 0001	0001	1
|	OR	5 | 1	0101 | 0001	0101	5
~	NOT	~ 5	~0101	1010	10
^	XOR	5 ^ 1	0101 ^ 0001	0100	4
<<	left shift	5 << 1	0101 << 1	1010	10
>>	right shift	5 >> 1	0101 >> 1	0010	2

Functions and Function Return

A **JavaScript function** is a block of code designed to perform a particular task. A function is a go of reusable code which can be called anywhere in your program. This eliminates the need of writing the same code again and again. It helps programmers in writing modular codes. Function allow a programmer to divide a big program into a number of small and manageable function.

Advantages of Using Function:

- You can't do anything in JavaScript without function.
- Functions increases code reusability.
- Function helps to structure the code properly.
- Functions make code less complex.

- Functions make code more readable and extendable.
- Functions can be called anywhere in the program.
- Functions helps program developers to write the modular codes.
- Function also allows a programmer to divide a big program into a number of small and manageable functions.

Function Definition

Before we use a function, we need to define it. The most common way to define a function in JavaScript is by using the ***function*** keyword, followed by a unique function name, a list di parameters (that might be empty), and a statement block surrounded by curly braces.

Syntax:

```
<script type="text/javascript">
function function_name (parameter-list)
{
//statements
}
</script>
```

Calling a Function

The code written inside a function does not execute unless it is called. To call a function somewhere later in the script, you would simply need to write the name of that function as shown in the following code.

```
<html>
<title>
</title>
<head>
<script type="text/javascript">
function Hello()
  {
document.write ("Hello Class 12");
  }
</script>
</head>
<body>
    <p>Click the following button to call the function </p>
     <form>
            <input type="button" onclick="Hello()" value= "Click Here">
    </form>
</body>
</html>
```

Example 2

```
<html>
<body>
<script>
    function showMessage()
            {
    document.write ("Hello Class 12");
    document.write ("Good Morning");
            }
               showMessage()
</script>
</body>
</html>
```

Function Parameters

A function can take multiple parameters separated by comma. The function parameters can have value of any data type.

```
<html>
<body>
      <h2> Function with Parameters </h2>
<script>
      var a= 3
      var b= 4
            add(a,b);
               function add(a,b)
               {
               var sum=a+b;
document.write("sum of two number is="+sum);    // OR     alert("sum of two number is="+sum);
               }
</script>
</body>
<html>
```

Function Return

The **return** statement is used to return a particular value from the function to the function caller. The function will stop executing when the **return** statement is called. The **return** statement should be the last statement in a function because the code after the **return** statement will be unreachable.

```
<html>
<head>
<title> Example of Function Return</title>  </head>
<body>
   <h1> Example of the JavaScript's return statement </h1>
   <script>
      var a=12;
      var b= 20;
      var res = fun(a, b);
```

```
function fun(x, y)
{
var r= x*y;
return (r);
}
document.write(res);
   </script>
</body>
</html>
```

Above program is a simple example of using the **return** statement. Here, returning the result of the product of two numbers and returned the value to the function caller.

The variable **res** is the function caller; it calls the function **fun()** and passes two parameters as the arguments of the function.

The result will be store in the **res** variable. Final output 240 is the product of arguments **12** and **20**.

Control Structures

1. If-else
2. switch case
3. for loop
4. while loop
5. do while loop

The **JavaScript if-else statement** is used *to execute the code whether condition is true or false*. There are three forms of if statement in JavaScript.

1. If Statement
2. If else statement
3. if else if statement

JavaScript If statement

It evaluates the content only if expression is true. The signature of JavaScript if statement is given below.

```
if(expression)
{
//content to be evaluated
}
```

Let's see the simple example of if statement in JavaScript.

```
<script>
var a=20;
if(a>10)
{
document.write("value of a is greater than 10");
}
</script>
```

```
<html>
<head>
<title> Number to check Positive or not </title>
</head>
<body>
<script type= "text/javascript">
var num= prompt("Enter Number");
if (num>0)
{
        document.write("Given number is Positive");
}
</script>
</body>
</html>
```

JavaScript If...else Statement

It evaluates the content whether condition is true of false. The syntax of JavaScript if-else statement is given below.

```
if(expression)
{
//content to be evaluated if condition is true
}
else
{
//content to be evaluated if condition is false
}
```

```
<html>
<body>
<script>
var num= prompt("Enter Number");
if (num>0)
{
 document.write("Given number is Positive");
}
```

```
else
{
 document.write("Given number is Negative");
}
</script>
</body>
</html>
```

Let's see the example of if-else statement in JavaScript to find out the even or odd number.

```
<script>
var a=17;
if(a%2==0)
{
document.write("a is even number");
}
else
{
document.write("a is odd number");
}
</script>
```

```
<html>
<body>
<script>
var num= prompt("Enter Number");
var x=num%2;
if (x==0)
{
    document.write("Given number is Even");
}
else
{
        document.write("Given number is Odd");
}
</script>
</body>
</html>
```

JavaScript If...else if statement

It evaluates the content only if expression is true from several expressions. The signature of JavaScript if else if statement is given below.

```
if(expression1)
{
//content to be evaluated if expression1 is true
}
else if(expression2)
{
//content to be evaluated if expression2 is true
}
else if(expression3)
{
//content to be evaluated if expression3 is true
}
else
{
//content to be evaluated if no expression is true
}
```

Let's see the simple example of if else if statement in JavaScript.

```
<script>
var a=20;
if(a==10)
{
document.write("a is equal to 10");
}
else if(a==15)
{
document.write("a is equal to 15");
}
else if(a==20)
{
document.write("a is equal to 20");
}
else
{
document.write("a is not equal to 10, 15 or 20");
}
</script>
```

Write a Program to input a number and check that is positive, negative or not.

```
<html>
<body>
<script>
var a=prompt("Enter Number");
if(a>0)
{
document.write("Number is positive");
}
else if(a<0)
{
document.write("Number is Negative");
}
else
{
document.write("Number is Zero");
}
</script>
</body>
</html>
```

JavaScript Switch

The **JavaScript switch statement** is used *to execute one code from multiple expressions*. It is just like else if statement that allow us to choose only one option among the many given options. But it is convenient than *if..else..if* because it can be used with numbers, characters etc.

The signature of JavaScript switch statement is given below.

```
switch(expression)
{
case value1:
 code to be executed;
 break;
case value2:
 code to be executed;
 break;
.....
default:
 code to be executed if above values are not matched;
}
```

Let's see the simple example of switch statement in JavaScript.

```
<html>
<head>
<script>
var grade='B';
var result;
switch(grade)
{
case 'A':
result="A Grade";
break;
case 'B':
result="B Grade";
break;
case 'C':
result="C Grade";
break;
default:
result="No Grade";
}
document.write(result);
</script>
</head>
</html>
```

```
<html>
<head>
<title> Switch Case Example For to find Name of the day.</title>
</head>
<body>
<script>
    var n=prompt("Enter a number between 1 and 7");
    switch(n)
    {
        case(n="1");
        document.write("The day is Sunday");
        break;
        case(n="2");
        document.write("The day is Monday");
        break;
        case(n="3");
        document.write("The day is Tuesday");
```

```
break;
case(n="4");
document.write("The day is Wednesday");
break;
case(n="5");
document.write("The day is Thursday");
break;
case(n="6");
document.write("The day is Friday");
break;
default:
document.write("Invalid day");
break;
</script>
</body>
</html>
```

JavaScript Loops / Iterations

The processing of repeatedly executing the block of statements is called iteration or looping. Loops are used to repeat the execution of a block of code. The **JavaScript loops** are used *to iterate the piece of code* using for, while, do while. It makes the code compact. It is mostly used in array.

There are four types of loops in JavaScript.

1. for loop
2. while loop
3. do-while loop

1) JavaScript For loop

For loop is used to execute a set of statements for given number of times. The **JavaScript for loop** *iterates the elements for the fixed number of times*. It should be used if number of iteration is known. The syntax of for loop is given below.

```
for (initialization; condition; increment)
{
  code to be executed
}
```

Let's see the simple example of for loop in JavaScript.

```
<html>
<head>
<script>
for (i=1; i<=5; i++)
{
document.write(i + "<br/>")  // OR   document.write(i);
}
```

```
</script>
</head>
</html>
```

Output: 12345

JavaScript While loop:

The while loop is an entry controlled loop in which condition is checked before the execution of loop.

Syntax

```
while (condition)
{
   // code block to be executed
 }
```

Example, WAP to display even number from 1 to 20.

```
<script>
    var i=2;
       while (i<=20)
        {
           document.write(i);
            i=i+2;
         }
</script>
```

JavaScript do while loop

The **JavaScript do while loop** *iterates the elements for the infinite number of times* like while loop. But, code is *executed at least* once whether condition is true or false. The syntax of do while loop is given below.

```
do
{
    code to be executed
}while (condition);
```

Example,

```
<script>
var i=1;
do{
document.write(i + "<br/>");
i=i+2;
}while (i<=20);
</script>
```

Some JavaScript Program Examples

1. Write a JavaScript program to input two numbers and find out its sum.

```
<html>
<head>
<script type=”text/javascript”>
var a = prompt(“Enter the first number”);
var b = prompt(“Enter the second number”);
a = parseInt(a);
b = parseInt(b);
var sum=a+b
document.write(sum)
</head>
<body>
</body>
</html>
```

Or

```
<html>
<head>
        <title> To find out sum of Given two numbers </title>
<script type="text/JavaScript">
function add()
{
        var a,b,c;
a=Number(document.getElementaryById("first").value);
b=Number(document.getElementaryById("second").value);
c=a+b;
document.getElementaryById("answer").value=c;
}
</script>
</head>
<body>
        Enter the First Numebr:      <input id="first">      <br>
        Enter the Second Number:   <input id="second">   <br> <hr>
        <button onclick="add()"> Addition </button>
        <input id="answer">
</body>
</html>
```

Object based Programming with Java Script and Event Handling

Object

Object is a non-primitive data type in JavaScript. Object is like any other variable but it is quite differ from variable. The variable can hold only one value but object can hold multiple values. The object holds multiple values in terms of properties and methods. **Properties** can hold values of primitive data types and **methods** are functions.

An object is defined simple way as like a variable. The object is defined with a unique object name and the curly brackets { } is used to include properties and methods. The variables within the object is known as properties and the function within the object is known as methods. The properties or method are written as name: value pairs. The name and value separated by a colon. The period '.' is used to access value to object.

Syntax:

```
var object_name = { name: value1, name2: value2, nameN: valueN };
```

Example

```
<html>
<body>
<script>
    var emp={id:101,name:"Ram Prasad",salary:50000}
    document.write(emp.id+" "+emp.name+" "+emp.salary
    document.write(emp.id+" "+emp.name+" "+emp.salary);
</script>
</body>
</html>
```

In the above example, object name is defined as emp and it has three properties id, name and salary. The value of object can be accessed easily using object name period and properties name.

Creating Objects in JavaScript

There are 3 ways to create objects.

1. By object literal
2. By creating instance of Object directly (using new keyword)
3. By using an object constructor (using new keyword)

1) JavaScript Object by object literal

The syntax of creating object using object literal is given below:

```
object={property1:value1,property2:value2.....propertyN:valueN}
```

As you can see, property and value is separated by : (colon). Let's see the simple example of creating object in JavaScript.

```
<html>
<body>
<h2>JavaScript Objects</h2>
<p>Creating a JavaScript Object:</p>
<p id="demo"></p>
<script>
const person = {firstName: "John", lastName: "Doe", age: 50, eyeColor: "blue"};
document.getElementById("demo").innerHTML = person.firstName + " is " + person.age + " years old.";
</script>
</body>
</html>
```

```
<html>
<head>
< title></title>
< script>
    var vehicle = {
                color:"green",
                weight:50,
                height:5,
                move:function()
                  {
                  document.write("vehicle moves");
                  }
              };
      vehicle.move();
  </script>
</head>
<body>
</body>
</html>
```

- *In the above example, method is defined as move. The method can be accessed easily using object name period and method name.*

2) By creating instance of Object

The syntax of creating object directly is given below:

```
var objectname=new Object();
```

Here, **new keyword** is used to create object. Let's see the example of creating object directly.

```
<html>
<body>
<script>
var study=new Object();
study.roll=10;
study.name="Rajesh Hamal";
study.class=12;
document.write(study.roll +"<br>");
document.write(study.name +"<br>");
document.write(study.class);
</script>
</body>
</html>
```

3) By using an Object constructor

Here, you need to create function with arguments. Each argument value can be assigned in the current object by using this keyword.

The **this keyword** refers to the current object.

The example of creating object by object constructor is given below.

```
<script>
function emp(id,name,salary)
{
this.id=id;
this.name=name;
this.salary=salary;
}
e=new emp(103,"Anjan Bista",30000);
document.write(e.id+" "+e.name+" "+e.salary);
</script
```

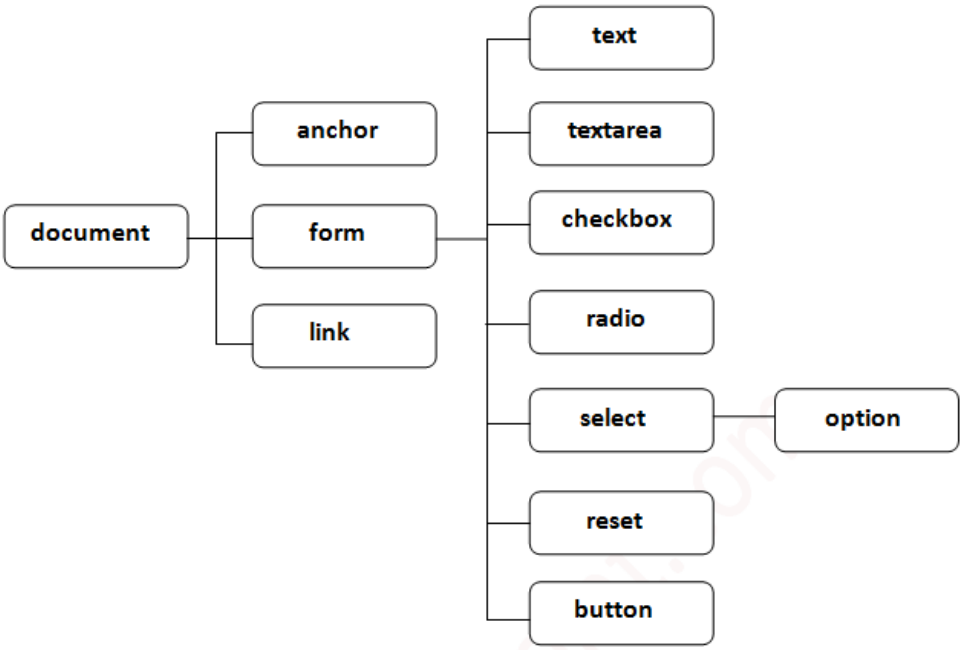
Document Object Model

The **document object** represents the whole html document. When html document is loaded in the browser, it becomes a document object. It is the **root element** that represents the html document. It has properties and methods. By the help of document object, we can add dynamic content to our web page.

As mentioned earlier, it is the object of window. So **window.document** is same as a **document**

Properties of document object

Let's see the properties of document object that can be accessed and modified by the document object.



JavaScript gets all the power it needs to create dynamic HTML:

- JavaScript can change all the HTML elements in the page
- JavaScript can change all the HTML attributes in the page
- JavaScript can change all the CSS styles on the page
- JavaScript can remove existing HTML elements and attributes
- JavaScript can add new HTML elements and attributes
- JavaScript can react to all existing HTML events in the page
- JavaScript can create new HTML events in the page

Methods of document object

We can access and change the contents of document by its methods. The important methods of document object are as follows:

Method	Description
write("string")	writes the given string on the doucment.
writeln("string")	writes the given string on the doucment with newline character at the end.
getElementById()	returns the element having the given id value.
getElementsByName()	returns all the elements having the given name value.
getElementsByTagName()	returns all the elements having the given tag name.
getElementsByClassName()	returns all the elements having the given class name.

Accessing field value by document object

In this example, we are going to get the value of input text by user. Here, we are using **document.form1.name.value** to get the value of name field. Here, **document** is the root element that represents the html document. **form1** is the name of the form. **name** is the attribute name of the input text. **value** is the property, that returns the value of the input text.

Let's see the simple example of document object that prints name with welcome message.

```
<script type="text/javascript">
function printvalue()
{
var name=document.form1.name.value;
alert("Welcome: "+name);
}
</script>
  <form name="form1">
     Enter Name:<input type="text" name="name"/>
<input type="button" onclick="printvalue()" value="print name"/>
</form>
```

1. getElementById( )

The **document.getElementById()** method returns the element of specified id. We can use document.getElementById() method to get value of the input text. But we need to define id for the input field.

Example

```
<html>
<body>
<p id="demo"></p>
<p id="try"></p>
<script>
var x="How are you"
document.getElementById("demo").innerHTML = x;
document.getElementById("try").innerHTML= " I am fine and you"
</script>
```

*In the example above, getElementById is a **method**, while innerHTML is a **property**.*

Let's see the simple example of **document.getElementById()** method that prints cube of the given number.

```
<script>
function getcube()
{
var n=document.getElementById("number").value;
alert(n*n*n);
}
</script>
<form>
Enter No:<input type="text" id="number" name="number"/><br/>
<input type="button" value="cube" onclick="getcube()"/>
</form>
```

2. getElementsByName()

The **document.getElementsByName()** method returns all the element of specified name. The syntax of the getElementsByName() method is given below:

`document.getElementsByName("name")` Here, name is required.

Example of document.getElementsByName() method

In this example, we going to count total number of genders. Here, we are using getElementsByName() method to get all the genders.

```
<script>
function totalelements()
{
var g=document.getElementsByName("gender");
alert("Total Genders:"+g.length);
}
</script>
<form>
Male:<input type="radio" name="gender" value="male">
Female:<input type="radio" name="gender" value="female">

<input type="button" onclick="totalelements()" value="Total Genders">
</form>
```

3. document.getElementsByTagName()

The **document.getElementsByTagName()** method returns all the element of specified tag name.

The syntax of the getElementsByTagName() method is given below:

```
document.getElementsByTagName("name")
```

Here, name is required.

Example of document.getElementsByTagName() method

In this example, we going to count total number of paragraphs used in the document. To do this, we have called the document.getElementsByTagName("p") method that returns the total paragraphs.

```
<script type="text/javascript">
function countpara()
{
var totalpara=document.getElementsByTagName("p");
alert("total p tags are: "+totalpara.length);
}
</script>
<p>This is a pragraph</p>
<p>Here we are going to count total number of paragraphs by getElementByTagName() method.
</p>
<p>Let's see the simple example</p>
<button onclick="countpara()">count paragraph</button>
```

Another example of document.getElementsByTagName() method

In this example, we going to count total number of h2 and h3 tags used in the document.

```
<script type="text/javascript">
function counth2()
{
var totalh2=document.getElementsByTagName("h2");
alert("total h2 tags are: "+totalh2.length);
}
function counth3()
{
var totalh3=document.getElementsByTagName("h3");
alert("total h3 tags are: "+totalh3.length);
}
</script>
<h2>This is h2 tag</h2>
<h2>This is h2 tag</h2>
<h3>This is h3 tag</h3>
<h3>This is h3 tag</h3>
<h3>This is h3 tag</h3>
<button onclick="counth2()">count h2</button>
<button onclick="counth3()">count h3</button>
```

Javascript - innerHTML

The **innerHTML** property can be used to write the dynamic html on the html document.

It is used mostly in the web pages to generate the dynamic html such as registration form, comment form, links etc.

Example of innerHTML property

In this example, we are going to create the html form when user clicks on the button.

In this example, we are dynamically writing the html form inside the div name having the id mylocation. We are identifying this position by calling the document.getElementById() method.

```
<script type="text/javascript" >
function showcommentform()
 {
var data="Name:<input type='text' name='name'><br>
Comment:<br><textarea rows='5' cols='80'></textarea>
<br><input type='submit' value='Post Comment'>";
document.getElementById('mylocation').innerHTML=data;
}
</script>
<form name="myForm">
<input type="button" value="comment" onclick="showcommentform()">
<div id="mylocation"></div>  </form>
```

JavaScript Events

The change in the state of an object is known as an **Event**. In html, there are various events which represents that some activity is performed by the user or by the browser. When JavaScript code is included in HTML, JavaScript react over these events and allow the execution. This process of reacting over the events is called **Event Handling**. Thus, JavaScript handles the HTML events via **Event Handlers**.

For example, when a user clicks over the browser, add js code, which will execute the task to be performed on the event.

Some of the HTML events and their event handlers are:

Mouse events:

Event Performed	Event Handler	Description
Click	Onclick	When mouse click on an element
Mouseover	Onmouseover	When the cursor of the mouse comes over the element
Mouseout	Onmouseout	When the cursor of the mouse leaves an element
Mousedown	Onmousedown	When the mouse button is pressed over the element
Mouseup	Onmouseup	When the mouse button is released over the element
Mousemove	Onmousemove	When the mouse movement takes place.

Keyboard events:

Event Performed	Event Handler	Description
Keydown & Keyup	onkeydown & onkeyup	When the user press and then release the key

Form events:

Event Performed	Event Handler	Description
Focus	onfocus	When the user focuses on an element
Submit	onsubmit	When the user submits the form
Blur	Onblur	When the focus is away from a form element
Change	onchange	When the user modifies or changes the value of a form element

Window/Document events

Event Performed	Event Handler	Description
Load	Onload	When the browser finishes the loading of the page
Unload	onunload	When the visitor leaves the current webpage, the browser unloads it
Resize	Onresize	When the visitor resizes the window of the browser

Let's discuss some examples over events and their handlers.

Click Event

```
<html>
<head> Javascript Events </head>
<body>
<script>
      function clickevent()
         {
            document.write("This is Java Click Events");
         }
</script>
        <form>
                  <input type="button" onclick="clickevent()" value="Click To View"/>
        </form>
```

```
</body>
</html>
```

Mouse Over Event

```
<html>
<head>
<h1> Javascript Events </h1>
</head>
<body>
<script>
      function mouseoverevent()
   {
      alert("This is Java MouseOverEvent");
   }
  </script>
<p onmouseover="mouseoverevent()"> Keep Mouse's cursor over me</p>
</body>
</html>
```

Focus Event

```
<html>
<head> Javascript Events</head>
<body>
<h2> Enter something here</h2>
<input type="text" id="input1" onfocus="focusevent()"/>
<script>
   function focusevent()
   {
      document.getElementById("input1").style.background=" aqua";
   }
</script>
</body>
</html>
```

Keydown Event

```
<html>
<head> Javascript Events</head>
<body>
<h2> Enter something here</h2>
<input type="text" id="input1" onkeydown="keydownevent()"/>
<script>
   function keydownevent()
   {
      document.getElementById("input1");
      alert("Pressed a key");
   }
</script>
</body>
</html>
```

Load event

```
<html>
<head>Javascript Events</head>
```

```
</br>
<body onload="window.alert('Page successfully loaded');">
<script>
   document.write("The page is loaded successfully");
</script>
</body>
</html>
```

Image Object

Form Validation

It is important to validate the form submitted by the user because it can have inappropriate values. So, validation is must to authenticate user.

JavaScript provides facility to validate the form on the client-side so data processing will be faster than server-side validation. Most of the web developers prefer JavaScript form validation.

Through JavaScript, we can validate name, password, email, date, mobile numbers and more fields.

In this example, we are going to validate the name and password. The name can't be empty and password can't be less than 6 characters long.

Here, we are validating the form on form submit. The user will not be forwarded to the next page until given values are correct.

```
<html>
<body>
<script>
function validateform()
{
var name=document.myform.name.value;
var password=document.myform.password.value;

if (name==null || name=="")
{
 alert("Name can't be blank");
 return false;
}
else if(password.length<6)
{
 alert("Password must be at least 6 characters long.");
 return false;
 }
}
</script>
<body>
```

```
<form name="myform" method="post"
action="http://www.javatpoint.com/javascriptpages/valid.jsp"
onsubmit="return validateform()" >
Name: <input type="text" name="name"><br/>
Password: <input type="password" name="password"><br/>
<input type="submit" value="register">
</form>
</body>
</html>
```

JavaScript Retype Password Validation

```
<script type="text/javascript">
function matchpass(){
var firstpassword=document.f1.password.value;
var secondpassword=document.f1.password2.value;

if(firstpassword==secondpassword){
return true;
}
else{
alert("password must be same!");
return false;
}
}
</script>

<form name="f1" action="register.jsp" onsubmit="return matchpass()">
Password:<input type="password" name="password" /><br/>
Re-enter Password:<input type="password" name="password2"/><br/>
<input type="submit">
</form>
```

Number Validation

Let's validate the textfield for numeric value only. Here, we are using isNaN() function.

```
<script>
function validate()
{
var num=document.myform.num.value;
if (isNaN(num))
{
  document.getElementById("numloc").innerHTML="Enter Numeric value only";
```

```
  return false;
}
Else
{
  return true;
  }
}
</script>
<form name="myform" onsubmit="return validate()" >
Number: <input type="text" name="num"><span id="numloc"></span><br/>
<input type="submit" value="submit">
</form>
```

JavaScript validation with image

Let's see an interactive JavaScript form validation example that displays correct and incorrect image if input is correct or incorrect.

```
<script>
function validate()
{
var name=document.f1.name.value;
var password=document.f1.password.value;
var status=false;

if(name.length<1)
{
document.getElementById("nameloc").innerHTML=
" <img src='unchecked.gif'/> Please enter your name";
status=false;
}
Else
{
document.getElementById("nameloc").innerHTML=" <img src='checked.gif'/>";
status=true;
}
if(password.length<6)
{
document.getElementById("passwordloc").innerHTML=
" <img src='unchecked.gif'/> Password must be at least 6 char long";
status=false;
```

```
}
Else
{
document.getElementById("passwordloc").innerHTML=" <img src='checked.gif'/>";

}
return status;
}
</script>
<form name="f1" action="#" onsubmit="return validate()">
<table>
<tr><td>Enter Name:</td><td><input type="text" name="name"/>
<span id="nameloc"></span></td></tr>
<tr><td>Enter Password:</td><td><input type="password" name="password"
/>
<span id="passwordloc"></span></td></tr>
<tr><td colspan="2"><input type="submit" value="register"/></td></tr>
</table>
</form>
```

JavaScript email validation

We can validate the email by the help of JavaScript.

There are many criteria that need to be follow to validate the email id such as:

- Email id must contain the @ and . character
- There must be at least one character before and after the @.
- There must be at least two characters after. (Dot).

Let's see the simple example to validate the email field.

```
<script>
function validateemail()
{
var x=document.myform.email.value;
var atposition=x.indexOf("@");
var dotposition=x.lastIndexOf(".");
if (atposition<1 || dotposition<atposition+2 || dotposition+2>=x.length)
{
  alert("Please enter a valid e-
mail address \n atpostion:"+atposition+"\n dotposition:"+dotposition);
```

```
  return false;
  }
}
</script>
<body>
<form name="myform"  method="post" action="#" onsubmit="return validateemail
();">
Email: <input type="text" name="email"><br/>
  <input type="submit" value="register">
</form>
```

JavaScript Programs

1. **This program adds 5 numbers and writes the answer as an HTML output.**

```
<!doctype html>
<html> <body>
<script>
var i, n=5, sum=0; arr = new Array(1, 2, 3, 5, 9);
for(i=0; i<5; i++)
sum = sum + arr[i]; document.write(sum);
</script></body></html>
```

2. **JavaScript program to check whether a number is a prime number or not. This program does not take input from user.**

```
<!doctype html>
<html>
<body> <script>
var num, i, c=0;
num=9;
for(i=2; i<num; i++)
{
if(num%2==0)
{
C++;
break;
}
}
if(c==0)
document.write(num+" is a Prime Number");
else
document.write(num+" is not a Prime Number");
</script> </body>
</html>
```

Data Format Validation:

Secondly, the data that is entered must be checked for correct form and value. Your code must include appropriate logic to test the correctness of data.

```
<html>
<body>
<script>
function myFormValidate ( )
{
var name = document.myform.name.value; var password =
document.myform.password.value;
    if (name==null || name=="")
    {
    alert("Please, Enter Name");
    return false;
}
else if (password.length < 5)
{
 alert ("Password must be at least 5 characters long.");
return false;
    }
}
</script>
<body>
<form name = "myform" method = "post" action = "abc.php" onsubmit = "return myFormValidate ( )" >
Name: <input type = "text" name = "name"><br/>
Password: <input type = "password" name = "password"><br/>
<input type = "submit" value = "Login">
</form>
</body>
</html>
```

JQuery

JQuery is a fast, small, lightweight, "write less, do more", and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling animation, and Ajax much simpler with an easy-to-use API (Application Programming Interface) that works across browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

JQuery is a JavaScript library. The purpose of jQuery is to make it much easier to use JavaScript on your website. JQuery simplifies a lot of the complicated things from JavaScript like AJAX (Asynchronous JavaScript and XML) calls and DOM (Document Object Model) manipulation. Some of the biggest companies which uses jQuery on the Web are

1. Google 2. Microsoft 3. IBM

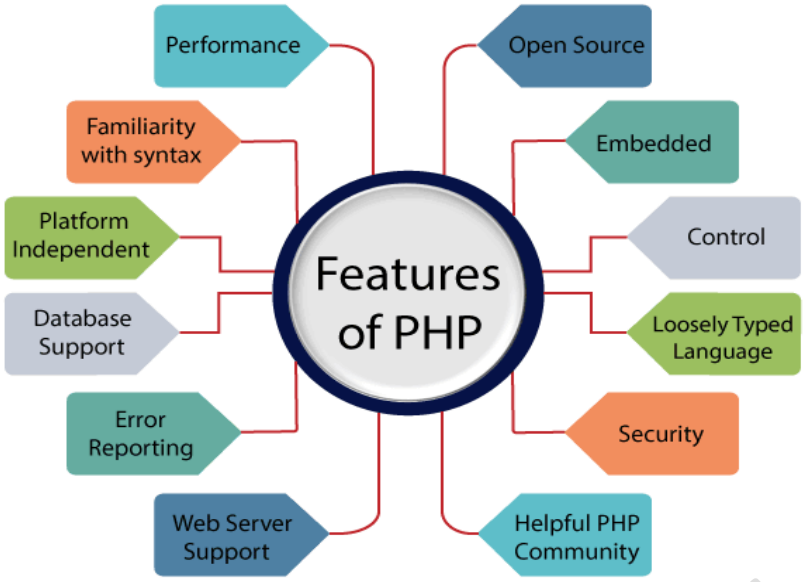
Features of jQuery:

Some of important feature of jQuery are listed as follows:

1. The jQuery is very small, fast, lightweight JavaScript library
2. It is very fast and extensible.
3. It has a lot of built-in animation effects which can use directly in websites.
4. It is used to improve the performance of an application.
5. It is used to develop browser's compatible web applications effectively.
6. It uses mostly new features of new browsers.
7. It is platform-independent.
8. It is used to develop a responsive web application.

Server Side Scripting Using PHP:

Server side script are discrete blocks of program code that execute on a web server (as opposed to a web client) computer. They are generally used to create Dynamic web pages. This means that the page displayed to the user does not exist as a document on the server in its own right, and is only brought into existence in response to a user request. Often, a server-side script provides the interface between a web-based user interface and a database that resides on a web server. PHP is a server side scripting language used to create dynamic web pages. PHP is well suited for web development. PHP stands for hypertext preprocessor. It is an interpreted language. It is embedded in HTML. It is an open source language.



Some important points need to be notice about PHP are as followed:

1. PHP stands for Hypertext Preprocessor.
2. PHP is an interpreted language, i.e., there is no need for compilation.
3. PHP is faster than other scripting languages, for example, ASP and JSP.
4. PHP is a server-side scripting language, which is used to manage the dynamic content of the website.
5. PHP can be embedded into HTML.
6. PHP is an object-oriented language.
7. PHP is an open-source scripting language.
8. PHP is simple and easy to learn language.

Features of PHP

1. Simple
2. Interpreted
3. Faster
4. Open Source
5. Platform Independent
6. Case Sensitive
7. Error Reporting
8. Real-Time Access Monitoring
9. Loosely Typed Language

Characteristics of PHP

1. Simplicity
2. Efficiency
3. Security
4. Flexibility
5. Familiarity

Advantages of PHP / what problem does it solve.

1. PHP is Free
2. PHP is Cross Platform
3. PHP is widely used
4. PHP hides its complexity
5. PHP is built for Web Programming

Introduction to PHP:

PHP is an open-source, interpreted, and object-oriented scripting language that can be executed at the server-side. PHP is well suited for web development. Therefore, it is used to develop web applications (an application that executes on the server and generates the dynamic page.).

Before learning PHP, you must have the basic knowledge of **HTML, CSS,** and **JavaScript**. So, learn these technologies for better implementation of PHP.

HTML - HTML is used to design static webpage.

CSS - CSS helps to make the webpage content more effective and attractive.

JavaScript - JavaScript is used to design an interactive website.

PHP is widely used in web development nowadays. PHP can develop dynamic websites easily. But you must have the basic the knowledge of following technologies for web development as well.

1. HTML
2. CSS
3. JavaScript
4. Ajax
5. XML and JSON
6. jQuery

Why use PHP / Strengths of PHP

PHP is a server-side scripting language, which is used to design the dynamic web applications with MySQL database.

1. It handles dynamic content, database as well as session tracking for the website.
2. You can create sessions in PHP.
3. It can access cookies variable and also set cookies.
4. It helps to encrypt the data and apply validation.
5. PHP supports several protocols such as HTTP (Hypertext Transfer Protocol), POP3 (Post office Protocol-3), SNMP (Simple Network Management Protocol), LDAP (Lightweight Directory Access Protocol), IMAP (Internet Message Access Protocol), and many more.
6. Using PHP language, you can control the user to access some pages of your website.
7. As PHP is easy to install and set up, this is the main reason why PHP is the best language to learn.
8. PHP can handle the forms, such as - collect the data from users using forms, save it into the database, and return useful information to the user. **For example** - Registration form.

Weaknesses of PHP

1. PHP's main strength flexibility is also its weakness. It can be a little too forgiving of errors.
2. With no strict rules, inexperienced programmers have the freedom to create some very bad solutions to simple problems.
3. Bad packages that got popular in the community are still reused when developers are trying something new or rushed for time. Some of these mistakes lead to security risks.

4. As a mature tool PHP has some legacy baggage. There are lots of internal consistencies, especially surrounding references and values.
5. This is mostly due to updates, which add features that clash with earlier features,
6. PHP is an interpreted language, which can reduce speed.
7. PHP 7 increased performance over previous versions by a significant amount while maintaining most language compatibility.
8. The changes didn't affect the learning curve or existing applications much while improving performance. Still, it executes more slowly than compiled languages.
9. Scaling and maintaining PHP is a complicated endeavor.
10. Context matters a great deal in dynamically typed languages, so tracking down errors gets harder the larger an application grows.
11. Experienced PHP developers can mitigate this problem y planning for scalability, but there's only so much they can do to reduce maintenance issues down the road.

Common uses of PHP

1. PHP performs system functions, i.e., from files on a system it can create, open, read, write, and close them.
2. PHP can handle forms, i.e., gather data from files, save data to a file, through email we can send data, return data to the user.
3. We add, delete, and modify elements within the database through PHP.
4. Access cookies variables and set cookies.
5. Using PHP, we can restrict users to access some pages of your website.
6. It can encrypt data.

3.15 Object Oriented Programming with Server Side Scripting

Object Oriented is an approach to software development that models application around real world objects such as employees, cars, bank accounts, etc. A class defines the properties and methods of a real world object. An object is an occurrence of a class. There are three major principles of OOP are:

1. Encapsulation: This is concerned with hiding the implementation details and only exposing the methods. It used to reduce software development complexity.

2. Inheritance: This is concerned with the relationship between classes. The main purpose of inheritance is the re-usability.

3. Polymorphism: This is concerned with having a single form but many different implementation ways. The main purpose of polymorphism is to simplify maintaining applications and making them more extendable.

PHP in Object oriented programming PHP is an object oriented scripting language. It supports all of the above principles. The above principles are achieved via:

Encapsulation - using "get" and "set" methods.

Inheritance - using extends keyword.

Polymorphism - using implements keyword

Hardware and Software Requirements

Hardware

Hardware requirement are not much, we just need a laptop or desktop computer. PHP 5.5+ require a computer with at least window 2008/vista.

The computer with Pentium IV processor, RAM 2GB, hard disk 256 GB, color monitor or above is more than enough.

Software

To run PHP code, we need the following software on our local machine.

1. Browser
2. Web server (eg. Apache)
3. PHP (interpreter)
4. MySQL Database (it is optional)

Basics of PHP:

Setting up the environment

To run PHP a web development is needed. This needs a PHP compatible web server and interpreter. Package like WAMP, LAMP, and XAMP etc. can be used which includes a web server.

Writing the code and running the script

PHP scripts are plain text. A PHP script begins with <?php and ends with ?>. The PHP code is saved with extension. PHP and is saved in the root directory of web server.

Basic PHP Syntax

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

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

- The default file extension for PHP files is ".php".
- A PHP file normally contains HTML tags, and some PHP scripting code.
- Below, we have an example of a simple PHP file, with a PHP script that uses a built-in PHP function "echo" to output the text "Hello World!" on a web page:

```
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello Class 12";
?>

</body>
</html>
```

Note: PHP statements end with a semicolon (;).

PHP Case Sensitivity

In PHP, keywords (e.g. if, else, while, echo, etc.), classes, functions, and user-defined functions are not case-sensitive.

In the example below, all three echo statements below are equal and legal:

```
<html>
<body>

<?php
ECHO "Hello Class 12<br>";
echo "Hello Class 12<br>";
EcHo "Hello Class 12<br>";
?>

</body>
</html>
```

Comments in PHP

Comments are used to make code more readable. There are two types of comments –single line and multi-line comments. A single line comments starts with // while multi-line comment begins with /* and end with */.

```
<html>
<body>

<?php
// This is a single-line comment

# This is also a single-line comment
?>

</body>
</html>
```

```
<html>
 <body>

 <?php
 /*
 This is a multiple-lines comment block
 that spans over multiple
 lines
 */
 ?>

 </body>
 </html>
```

Loosely typed language: PHP is a loosely typed language, it means PHP automatically converts the variable to its correct data type.

PHP Variables

A variable can have a short name (like x and y) or a more descriptive name (age, carname, total_volume).

Rules for PHP variables:

- 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)

PHP Variable: Declaring string, integer, and float

Let's see the example to store string, integer, and float values in PHP variables.

```
<?php
$str="hello string";
$x=200;
$y=44.6;
echo "string is: $str <br/>";
echo "integer is: $x <br/>";
echo "float is: $y <br/>";
?>
```

PHP Variable Scope

The scope of a variable is defined as its range in the program under which it can be accessed. In other words, "The scope of a variable is the portion of the program within which it is defined and can be accessed."

PHP has three types of variable scopes:

1. Local variable 2. Global variable 3. Static variable

Local variable

The variables that are declared within a function are called local variables for that function. These local variables have their scope only in that particular function in which they are declared. This means that these variables cannot be accessed outside the function, as they have local scope.

A variable declaration outside the function with the same name is completely different from the variable declared inside the function. Let's understand the local variables with the help of an example:

Global variable

The global variables are the variables that are declared outside the function. These variables can be accessed anywhere in the program. To access the global variable within a function, use the GLOBAL keyword before the variable. However, these variables can be directly accessed or used outside the function without any keyword. Therefore there is no need to use any keyword to access a global variable outside the function.

Let's understand the global variables with the help of an example:

```
<?php
   function local_var()
   {
      $num = 45;  //local variable
      echo "Local variable declared inside the function is: ". $num;
   }
   local_var();
?>
```

```
<?php
$name = "Sanaya Sharma";        //Global Variable
function global_var()
{
    global $name;
    echo "Variable inside the function: ". $name;
    echo "</br>";
}
global_var();
echo "Variable outside the function: ". $name;
?>
```

Basic Programming in PHP

PHP echo and print Statements

We frequently use the echo statement to display the output. There are two basic ways to get the output in PHP:

- echo
- print

echo and print are language constructs, and they never behave like a function. Therefore, there is no requirement for parentheses. However, both the statements can be used with or without parentheses. We can use these statements to output variables or strings.

```
<?php
echo "<h2>PHP is Fun!</h2>";
echo "Hello world!<br>";
echo "I'm about to learn PHP!<br>";
echo "This ", "string ", "was ", "made ", "with multiple parameters."; <br> <hr>

print "<h2>PHP is Case Sensitive!</h2>";
print "Hello Class 12!<br>";
print "I'm about to learn PHP!";
?>
```

Difference between echo and print

echo

- echo is a statement, which is used to display the output.
- echo can be used with or without parentheses.
- echo does not return any value.
- We can pass multiple strings separated by comma (,) in echo.
- echo is faster than print statement.

print

- print is also a statement, used as an alternative to echo at many times to display the output.
- print can be used with or without parentheses.
- print always returns an integer value, which is 1.

- Using print, we cannot pass multiple arguments.
- print is slower than echo statement.

PHP Data Types

Variables can store data of different types, and different data types can do different things. PHP supports the following data types:

1. String
2. Integer
3. Float (floating point numbers - also called double)
4. Boolean
5. Array
6. Object
7. NULL
8. Resource

PHP Data Types: Compound Types

It can hold multiple values. There are 2 compound data types in PHP.

1. array
2. object

PHP Data Types: Special Types

There are 2 special data types in PHP.

1. resource
2. NULL

PHP Boolean

Booleans are the simplest data type works like switch. It holds only two values: **TRUE (1)** or **FALSE (0)**. It is often used with conditional statements. If the condition is correct, it returns TRUE otherwise FALSE.

Example:

```
<?php
   if (TRUE)
      echo "This condition is TRUE.";
   if (FALSE)
      echo "This condition is FALSE.";
?>
```

Output:

```
This condition is TRUE.
```

PHP Integer

Integer means numeric data with a negative or positive sign. It holds only whole numbers, i.e., numbers without fractional part or decimal points.

Rules for integer:

- An integer can be either positive or negative.
- An integer must not contain decimal point.
- Integer can be decimal (base 10), octal (base 8), or hexadecimal (base 16).
- The range of an integer must be lie between 2,147,483,648 and 2,147,483,647 i.e., -2^31 to 2^31.

Example:

```
<?php
   $dec1 = 34;
   $oct1 = 0243;
   $hexa1 = 0x45;
   echo "Decimal number: " .$dec1. "</br>";
   echo "Octal number: " .$oct1. "</br>";
   echo "HexaDecimal number: " .$hexa1. "</br>";
?>
```

Output:

```
Decimal number: 34
Octal number: 163
Hexadecimal number: 69
```

PHP Float

A floating-point number is a number with a decimal point. Unlike integer, it can hold numbers with a fractional or decimal point, including a negative or positive sign.

Example:

```
<?php
   $n1 = 19.34;
   $n2 = 54.472;
   $sum = $n1 + $n2;
   echo "Addition of floating numbers: " .$sum;
?>
```

Output:

```
Addition of floating numbers: 73.812
```

PHP String

A string is a non-numeric data type. It holds letters or any alphabets, numbers, and even special characters. String values must be enclosed either within **single quotes** or in **double quotes**. But both are treated differently. To clarify this, see the example below:

Example:

```
<?php
   $company = "Javatpoint";
   //both single and double quote statements will treat different
   echo "Hello $company";
   echo "</br>";
   echo 'Hello $company';
?>
```

Output:

```
Hello Javatpoint
Hello $company
```

PHP Array

An array is a compound data type. It can store multiple values of same data type in a single variable.

Example:

```
<?php
   $bikes = array ("Royal Enfield", "Yamaha", "KTM");
   var_dump($bikes);   //the var_dump() function returns the datatype and values
   echo "</br>";
   echo "Array Element1: $bikes[0] </br>";
   echo "Array Element2: $bikes[1] </br>";
   echo "Array Element3: $bikes[2] </br>";
?>
```

Output:

```
array(3) { [0]=> string(13) "Royal Enfield" [1]=> string(6) "Yamaha" [2]=> string(3) "KTM" }
Array Element1: Royal Enfield
Array Element2: Yamaha
Array Element3: KTM
```

PHP object

Objects are the instances of user-defined classes that can store both values and functions. They must be explicitly declared.

Example:

```
<?php
    class bike {
        function model() {
            $model_name = "Royal Enfield";
            echo "Bike Model: " .$model_name;
          }
    }
    $obj = new bike();
    $obj -> model();
?>
```

Output:

```
Bike Model: Royal Enfield
```

PHP Resource

Resources are not the exact data type in PHP. Basically, these are used to store some function calls or references to external PHP resources. **For example** - a database call. It is an external resource.
This is an advanced topic of PHP, so we will discuss it later in detail with examples.

PHP Null

Null is a special data type that has only one value: **NULL**. There is a convention of writing it in capital letters as it is case sensitive.
The special type of data type NULL defined a variable with no value.

Example:

```
<?php
   $n = NULL;
```

```
   echo $n;   //it will not give any output
?>
```

PHP Operators

PHP Operator is a symbol i.e used to perform operations on operands. In simple words, operators are used to perform operations on variables or values. For example:

$num=10+20; //+ is the operator and 10,20 are operands

In the above example, + is the binary + operator, 10 and 20 are operands and $num is variable.

PHP Operators can be categorized in following forms:

1. Arithmetic Operators
2. Assignment Operators
3. Bitwise Operators
4. Comparison Operators
5. Incrementing/Decrementing Operators
6. Logical Operators
7. String Operators
8. Array Operators
9. Type Operators
10. Execution Operators

Arithmetic Operators

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

Operator	Name	Example	Explanation
+	Addition	$a + $b	Sum of operands
-	Subtraction	$a - $b	Difference of operands
*	Multiplication	$a * $b	Product of operands
/	Division	$a / $b	Quotient of operands
%	Modulus	$a % $b	Remainder of operands
**	Exponentiation	$a ** $b	$a raised to the power $b

Assignment Operators

The assignment operators are used to assign value to different variables. The basic assignment operator is "=".

Operator	Name	Example	Explanation
=	Assign	$a = $b	The value of right operand is assigned to the left operand.
+=	Add then Assign	$a += $b	Addition same as $a = $a + $b
-=	Subtract then Assign	$a -= $b	Subtraction same as $a = $a - $b
*=	Multiply then Assign	$a *= $b	Multiplication same as $a = $a * $b
/=	Divide then Assign (quotient)	$a /= $b	Find quotient same as $a = $a / $b

%=	Divide then Assign (remainder)	$a %= $b	Find remainder same as $a = $a % $b

Bitwise Operators

The bitwise operators are used to perform bit-level operations on operands. These operators allow the evaluation and manipulation of specific bits within the integer.

Operator	Name	Example	Explanation
&	And	$a & $b	Bits that are 1 in both $a and $b are set to 1, otherwise 0.
|	Or (Inclusive or)	$a | $b	Bits that are 1 in either $a or $b are set to 1
^	Xor (Exclusive or)	$a ^ $b	Bits that are 1 in either $a or $b are set to 0.
~	Not	~$a	Bits that are 1 set to 0 and bits that are 0 are set to 1
<<	Shift left	$a << $b	Left shift the bits of operand $a $b steps
>>	Shift right	$a >> $b	Right shift the bits of $a operand by $b number of places

Comparison Operators

Comparison operators allow comparing two values, such as number or string. Below the list of comparison operators are given:

Operator	Name	Example	Explanation
==	Equal	$a == $b	Return TRUE if $a is equal to $b
===	Identical	$a === $b	Return TRUE if $a is equal to $b, and they are of same data type
!==	Not identical	$a !== $b	Return TRUE if $a is not equal to $b, and they are not of same data type
!=	Not equal	$a != $b	Return TRUE if $a is not equal to $b
<>	Not equal	$a <> $b	Return TRUE if $a is not equal to $b
<	Less than	$a < $b	Return TRUE if $a is less than $b
>	Greater than	$a > $b	Return TRUE if $a is greater than $b
<=	Less than or equal to	$a <= $b	Return TRUE if $a is less than or equal $b
>=	Greater than or equal to	$a >= $b	Return TRUE if $a is greater than or equal $b
<=>	Spaceship	$a <=>$b	Return -1 if $a is less than $b Return 0 if $a is equal $b Return 1 if $a is greater than $b

Incrementing/Decrementing Operators

The increment and decrement operators are used to increase and decrease the value of a variable.

Operator	Name	Example	Explanation
++	Increment	++$a	Increment the value of $a by one, then return $a
		$a++	Return $a, then increment the value of $a by one
--	Decrement	--$a	Decrement the value of $a by one, then return $a
		$a--	Return $a, then decrement the value of $a by one

Logical Operators

The logical operators are used to perform bit-level operations on operands. These operators allow the evaluation and manipulation of specific bits within the integer.

Operator	Name	Example	Explanation
And	And	$a and $b	Return TRUE if both $a and $b are true
Or	Or	$a or $b	Return TRUE if either $a or $b is true
Xor	Xor	$a xor $b	Return TRUE if either $ or $b is true but not both
!	Not	! $a	Return TRUE if $a is not true
&&	And	$a && $b	Return TRUE if either $a and $b are true
||	Or	$a || $b	Return TRUE if either $a or $b is true

PHP Form Handling

We can create and use forms in PHP. To get form data, we need to use PHP super global $_GET and $_POST. The form request may be get or post. To retrieve data from get request, we need to use $_GET, for post request $_POST.

PHP Get Form

Get request is the default form request. The data passed through get request is visible on the URL browser so it is not secured. You can send limited amount of data through get request.

Let's see a simple example to receive data from get request in PHP.

File: form1.html

```
<form action="welcome.php" method="get">
Name: <input type="text" name="name"/>
<input type="submit" value="visit"/>
</form>
```

File: welcome.php

```
<?php
$name=$_GET["name"];//receiving name field value in $name variable
echo "Welcome, $name";
?>
```

PHP Post Form

Post request is widely used to submit form that have large amount of data such as file upload, image upload, login form, registration form etc.

The data passed through post request is not visible on the URL browser so it is secured. You can send large amount of data through post request.

Let's see a simple example to receive data from post request in PHP.

File: form1.html

```
<html>
<head> <title>  Post Method </title>
</head>
```

```
<body>
<form action="login.php" method="post">
<table>
<tr><td>Name:</td><td> <input type="text" name="name"/></td></tr>
<tr><td>Password:</td><td> <input type="password" name="password"/></td></tr>
<tr><td colspan="2"><input type="submit" value="login"/>  </td></tr>
</table>
</form> </body> </html>
```

File: login.php

```
<?php
$name=$_POST["name"];
$password=$_POST["password"];
  echo "Welcome: $name, your password is: $password";
?>
```

S.N.	GET Method	S.N.	POST Method
1	Only limited amount of data can be sent because is sent in header.	1	Large amount of data can be sent because data is sent in body.
2	GET request is not secured because data is exposed in URL bar.	2	POST request is secured because data is not exposed in URL bar.
3	GET request can be bookmarked.	3	POST request cannot be bookmarked.
4	GET is essentially used for fetching the information.	4	The purpose of POST method is to update the data.
5	It can be cached.	5	It cannot be cached.
6	GET method is the default method if the method is not specified in the form.	6	POST method must be specified in the form. It is not default method.

PHP Programs

1. Write a php code to enter your name and display it.

```
<html>
<head>
<title> Your name </title>
</head>
<body>
        <B> Enter your Name</B> <br/>
<form method="POST">
<input type="text" name="name"/> <br/>
<input type="submit" value="submit"/>
</form>
<?php
$name=$_POST['name'];
echo "Your Name=$name";
?>
</body>
```

```
<html>
```

2. Write a php code to display the factorial of a number given by user.

```
<html>
<head>
<title> Your name </title>
</head>
<body>
        <B> Factorial Calculaiton</B> <br/>
<form method="POST">
        Input a Number: <input type="Number" name="number"/> <br/>
        <input type="submit" value="Calculate"/>
</form>
<?php
        $n=$_POST['number'];
        $fact=1;
for($i=1; $i<=$n;$i++)
{
        $fact=$fact*$i;
}
echo "The factorial of $n=$fact";
?>
</body>
</html>
```

3. Write a php code to display all even numbers up to 50.

```
<html>
        <head>
        <title> Your name </title>
        </head>
        <body>
                <B> Display Even number up to 50</B> <br/> <hr>
        <?php
                $a=2;
                for($i=1;$i<=25;$i++)
                {
                        echo "$a, ";
                        $a=$a+2;
                }
        ?>
        </body>
        </html>
```

4. Write a PHP code to display the simple interest.

```
<html>
<head>
<title> Calculate simple Interest </title>
</head>
<body>
        <B> Calculate Simple Interest</B> <br/>
<form method="POST">
```

```
        Enter Principal:<input type="Number" name="p"/> <br/>
        Enter Time:<input type="Number" name="t"/> <br/>
        Enter Rate: <input type="Number" name="r"/> <br/>
        <input type="submit" value="Calculate SI"/>
</form>
<?php
        $p=$_POST['p'];
        $t=$_POST['t'];
        $r=$_POST['r'];
        $si=($p*$t*$r)/100;
        echo "Simple Interest=$si";
?>
</body>
<html>
```

Database Connectivity

Database: To organize the data in systematic and manageable form, the database is required. The database is the system where data and information are stored and organized systematically.

- The stored data can be retrieved easily when required.
- Data are stored in tabular form.
- The number of columns and rows forms the table.

The column of the table is known as a field that has a unique field name and the row of the table is known as records which represent individual information. Each table has a unique field known as a primary key. Each of the records of the primary field is different which make distinct from other records for identification.

MySQL is the most popular database system used with PHP. With PHP, we can connect to and manipulate MYSQL databases. It is a database system used on the web that runs on a server. It can be used for organized databases for both small and large applications. It is comparatively more fast, reliable, and easy to use than other databases. It uses standard SQL and compiles on a number of platforms.

```
<?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";
?>
```

Web References:

- https://www.javatpoint.com
- https://www.w3schools.com
- https://www.tutorialspoint.com
- https://www.google.com
- https://www.wikipedia.org