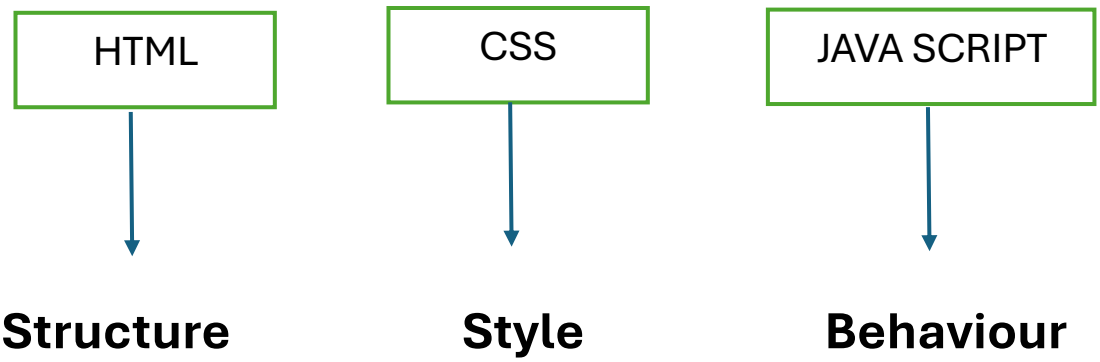
Introduction to JavaScript: Java Script is a computer programming language used to make websites and applications dynamic and interactive. It is unique because it can run directly in your browser and also on a server. Along with Hypertext Markup Language (HTML) and Cascading Style Sheets (CSS), JavaScript is one of the most commonly used programming languages of the Internet. HTML provides the basic layout, structure, and content of a website. CSS provides design, fonts, colors, effects, and other visual elements. JavaScript brings dynamism and interactivity to the website. For example, pop-ups, animations, video and social media embeds, drop-down menus, and many other website components are created using JavaScript.



History of JavaScript: JavaScript is developed in 1995 by Brendan Eich, a computer scientist and programmer at Netscape Communications Corporation. The initial name of the JavaScript was 'Mocha'. After that, it changed to 'LiveScript', and then 'JavaScript'. Over the years, JavaScript has evolved and become more powerful, with the addition of new features such as object-oriented programming, regular expressions, and support for asynchronous programming. Today, JavaScript is used in a wide range of applications, including web development, mobile app development, and server-side programming. JavaScript language is interpreted and executed by the browser.

JavaScript as an Interpreted Language: An interpreted language is one that does not require compiling into machine language. It is executed by an interpreter that reads the source code and converts it into a form that is directly executed. The interpreter executes the code line by line. Any error that is found by the interpreter on any line of code, will stop further execution of the program. JavaScript is directly executed by the browser which interprets

the program instruction by instruction leading to slower execution of the program.

Static Website: A static website is a type of website that delivers the same content to all users. The content is stored on the web server and is shown on the user's browser as is, without any change of content. Static websites are simple to create and host, and are typically used for informational websites, such as brochure websites or portfolios. They don't require any server-side processing or database interactions, making them fast and easy to maintain.
Dynamic Website: A dynamic website is a type of website that generates content based on user interactions or other events. It generates custom content for each user based on their actions or preferences. Dynamic websites are typically created using server-side programming languages such as PHP, JavaScript, or Python, and they interact with a database to retrieve and store data. Examples of dynamic websites include social media platforms, e-commerce sites, and content management systems.

Static Website	Dynamic Website
Content of Web pages cannot be changed at runtime.	Content of Web pages can be changed at runtime.
No interaction with the database possible.	Interaction with database possible.
It is faster to load as compared to dynamic websites as it is more text based.	It is slower than a static website because it has images, audio & video content.
Cheaper Development costs.	More Development costs.
Blog Websites, Newsletter Contents, Brochure websites	Facebook, MakemyTrip, TimesofIndia

Features of JavaScript

1. **Easy to Learn**: JavaScript is easy to learn and the syntax of JavaScript is very simple. A programming beginner can choose JavaScript as their first programming language.

2. **Case Sensitive**: JavaScript is a highly case-sensitive programming language, which means that the identifiers, keywords, variables, and function names must be written correctly.

3. **OS Support**: It is supported by several operating systems including, Windows, macOS, etc.

4. **Control Statements**: It has control statements like if, if-else, if-else-if, switch case, and loop which helps users to write complex code using these control statements.

5. **Client and Server-Side Support:** Client-side or front-end web developers use their programming talents to create visually appealing websites for users. This means they build homepages, shopping pages, and slideshows.
Server-side or back-end developers create, design, and manage server-side code responsible for data exchange. You can use JavaScript for both client-side and server-side scripting. This means it's helpful when designing website layouts and managing server-side code for data exchange.
6. **Object-Oriented:** It is an object-oriented programming language, which means that it uses objects to represent real-world entities and concepts.
7. **Browser Support**: All popular web browsers support JavaScript as they provide built-in execution environments.
8. **Functional Programming:** In JavaScript, a function can be assigned to variables just like any other data types. A function can accept another function as a parameter and can also return a function. This provides you the ability to code in functional programming style.
9. **Dynamic:** JavaScript is a high-level and dynamic programming language, which means that the type of a variable is determined at runtime, rather than being specified in the code.

Prerequisites for Executing JavaScript Programs :The biggest advantage of JavaScript is that there is no need to purchase any tool to develop the JavaScript application. These tools are freely available and can be easily obtained. A simple text editor like windows notepad is required to create the JavaScript code. An online text editor can be used alternatively. A HTML editor is another tool which can edit the HTML source code and add the JavaScript code in the web page. An example of an HTML editor is Dreamweaver which has lots of features and drag and drop facilities to make the task easy. Microsoft Visual Web Developer Express is an advanced page editor which can

color the important JavaScript words and validate the code and load the page into the web browser to view the changes or contents.

Introduction to Script Tag :JavaScript can be implemented using **<script>…</script>** tags. The <script> tag containing javascript can be placed anywhere within the web page, but it is normally recommended that it should be kept within the <head> tags. The <script> tag alerts the browser program to start interpreting all the text between these tags as script commands.

The syntax of a JavaScript segment in HyperText Markup Language (HTML) is as follows:

```
<script language="JavaScript" type="text/JavaScript">
    JavaScript code
</script>
```

The <script> tag takes two important attributes:

1. Language- This attribute specifies the scripting language. Typically, its value will be JavaScript. The recent versions of HTML have phased out the use of this attribute and has become optional.

2. Type – This attribute is used to indicate the scripting language and its value should be set to "text/JavaScript". All current browsers are updated and are JavaScript enabled by default. This attribute too is optional.

There are three different places in the HTML document where scripts can be used.

1. **Body of the page**: In this case when page is loaded in the browser then output is displayed as the part of the HTML document.

2. **Header of the page:** In this case code is written in the form of a function (groups of JavaScript statements but treated as a single unit and referred to in the other script in the same page).

3. **As external file:** In this case JavaScript code is written in another file having .js extension. This file is included in a script tag by specifying the file name.

Writing first JavaScript program

<script>
document.write("*hello World!*");
// First JavaScript Program to print Hello World on the screen
</script>
Output on the browser : hello World!

1. Open any editor such as notepad and write the above program
2. Save the program with the .html extension in a proper folder or subfolder on a drive like C:\JavaScript\myprograms.
3. Open the web browser like Google Chrome.
4. Double Click the file you have created and saved in step 2, and then you can see the output of your program.

JavaScript Syntax and Rules: JavaScript syntax is the set of rules that define a structured JavaScript. Here are some tips to remember when writing your first JavaScript program.

1. Case Sensitivity: JavaScript is case sensitive i.e., uppercase letters and lower-case letters have different meanings. For example, the word "alert" has a lowercase "a". so, if we type the word with an uppercase "A", then the alert box will not be displayed and the JavaScript code won't get executed.

2. Whitespace & Line Break: You can use spaces, tabs, and newlines anywhere in the JavaScript Program. The JavaScript interpreter ignores them. Use tabs & spaces to neatly format or indent your code. It makes the code easy to read & understand.

3. Comments: The JavaScript allows us to add single line comments or Multi line comments.
Single-line comments (//) − Any text between a // and the end of a line is treated as a comment.
Multi-line comments (/* */) − These comments may span multiple lines.
Example: <script>
//This program shows single line comments
/*This is an example of
multiline comment */
</script>

Common Errors :Since JavaScript is an interpreted language, there are some common mistakes that won't display the code correctly on your browser.

1. Missing quotation marks - document.write("hello world)
2. Case sensitive - document -> cannot be written as Document
3. Missing parenthesis/brackets - document.write("hello world"
4. Missing <script> tag in the program (either opening or closing)
5. Variable names misspelt.

Internal & External Java Script

JavaScript can be added directly to the HTML file by writing the code inside the <script> tag . We can place the <script> tag either inside <head> or the <body> tag according to the need.

Example:

```
<html>
<head>
   <title>Internal Javascript</title>
</head>
<script>
/*Internal Javascript*/
Document.write("Hello Friends!How are you");
</script>
</html>
```

Output is: Hello Friends!How are you

External JavaScript: A JavaScript program can be written in a file and saved with the .js extension. This file can be then linked inside the <head> and </head> tags of the HTML document in which we want to add this code. The SRC attribute of the <script> tag allows to give the path of the JavaScript file.

Example:

Open the text editor and type the following:

```
document. write("Good Morning Friends!")
```

Save the above file as tryExternal.js

Open another document in your text editor and type the following code:

```
<html>
```

```
<head>
<script src="tryExternal.js">
</script>
</head>
</html>
```

Save the above file as Show.html and open it in your browser to see the output.

Output: Good Morning Friends!

Input and Output from the Script

1. document.write method

JavaScript has access to the document property of the window object. The document property returns the document object of the window which is being used. When an HTML element is loaded onto the web browser then it becomes a part of the document object. The document object is the root node of the HTML page. One can access the document object either by using window.document or just document. For example: window.document.close() or document.close() The document object has a large range of properties or methods which a developer can make use of. The document.write() in JavaScript helps to write a JavaScript program or HTML expression into the document**.**

Example 1: We can write a string into a web page

```
<html>
<head>
<script>document. write("Beware of Cyber Scams")</script>
</head>
</html>
```

Example 2: We can also write HTML content and display them on a web page.

```
<html>
<head>
<script>document. write("<b>Beware of Cyber Scams</b>")</script>
</head>
</html>
```

2. Dialog boxes

(i) alert() dialog box: An alert dialog box displays a short message or notification to the user. Alert box gives only one button "OK" to select and

proceed. It just gives some information, necessary to the user and it can be invoked at various events possible in JavaScript.

Example:

```
<html>
<head>
<title>JavaScript Alert Dialog Box</title>
</head>
<body>
<script>
alert("You have chosen an Alert box");
 </script>
</body>
</html>
```

(ii) prompt() dialog box: The prompt dialog box is used to get some input from the user. It has two buttons, one for “OK” and the other for “Cancel”. This method returns the text entered in the input field when the user clicks the OK button. It will return null if the user clicks the Cancel button. If the user clicks the OK button without entering any text, an empty string is returned. For this reason, its result is usually assigned to a variable when it is used. If you enter a number in the prompt box, it will be stored as a string, as it always returns a string.

Example:

```
<html>
<head>
<script>
var yourname = prompt("Enter your name : ");
var age = prompt("Enter your age :");
document.write("Your name is : " + yourname+"<br>");
document.write("Your age is : " + age);
</script>
</head>
</html>
```

(iii) confirm() dialog box: A confirmation dialog box is mostly used to take the user's confirmation on any option. It displays a dialog box with two buttons: OK

and Cancel. If the user clicks on the OK button, it will return true. If the user clicks on the Cancel button, it will return false. It returns a Boolean value depending on whether the user clicks OK(true) or CANCEL(false) button.
Example:

```
<html>
<head>
<title>JavaScript Confirm Dialog Box</title></head>
<body>
<script>
var result = confirm("Are you sure you want to cancel the order?");
if(result==true) {
document.write("Order IS Canceled!");
 } else {
 document.write("Order NOT Canceled");
}
</script>
</body>
</html>
```

3. Interacting with HTML

The HTML DOM (Document Object Model) provides a way to interact with and manipulate the elements of an HTML document using JavaScript. It allows you to access, modify, and add elements dynamically, change styles and classes, handle events, and perform other operations on the document. One of the most common DOM functions you'll use is the document.getElementById() function, which returns the element with the ID you pass in as a parameter.
Example:

```
<html>
<head>
<script>
function myFunction()
{
 document.getElementById("demo").innerHTML="Hello World";
 }
</script>
</head>
```

```
<body>
<p>click the button</p>
< button onclick="myFunction()">Click Me< /button>
< p id="demo"></p>
</body>
</html>
```