JavaScript

JavaScript is a lightweight, cross-platform, and interpreted scripting language. It is well-known for the development of web pages; many non-browser environments also use it. JavaScript can be used for **Client-side** developments as well as **Server-side** developments. JavaScript contains a standard library of objects, like **Array**, **Date**, and **Math**, and a core set of language elements like **operators**, **control structures**, and **statements.**

- ✓ **Client-side:** It supplies objects to control a browser and its Document Object Model (DOM). Like if client-side extensions allow an application to place elements on an HTML form and respond to user events such as **mouse clicks**, **form input**, and **page navigation**. Useful libraries for the client-side are **AngularJS**, **ReactJS**, **VueJS** and so many others.
- ✓ **Server-side:** It supplies objects relevant to running JavaScript on a server. Like if the server-side extensions allow an application to communicate with a database, and provide continuity of information from one invocation to another of the application, or perform file manipulations on a server. The useful framework which is the most famous these days is **node.js**.

- ✓ JavaScript can be added to your HTML file in **two ways:**
- ✓ **Internal JS:** We can add JavaScript directly to our HTML file by writing the code inside the <script> tag. The <script> tag can either be placed inside the <head> or the <body> tag according to the requirement.
- ✓ **External JS:** We can write JavaScript code in other file having an extension .js and then link this file inside the <head> tag of the HTML file in which we want to add this code.

History of JavaScript:

It was created in 1995 by Brendan Eich while he was an engineer at Netscape. It was originally going to be named LiveScript but was renamed. Unlike most programming languages, the JavaScript language has no concept of input or output. It is designed to run as a scripting language in a host environment, and it is up to the host environment to provide mechanisms for communicating with the outside world. The most common host environment is the browser.

Features of JavaScript: According to a recent survey conducted by **Stack Overflow**, JavaScript is the most popular language on earth.
With advances in browser technology and JavaScript having moved into the server with Node.js and other frameworks, JavaScript is capable of so much more.

Here are a few things that we can do with JavaScript:

- ✓ JavaScript was created in the first place for DOM manipulation. Earlier websites were mostly static, after JS was created dynamic Web sites were made.
- ✓ Functions in JS are objects. They may have properties and methods just like another object. They can be passed as arguments in other functions.
- ✓ Can handle date and time.
- ✓ Performs Form Validation although the forms are created using HTML.
- ✓ No compiler needed.
- ✓

Applications of JavaScript:

- ✓ **Web Development:** Adding interactivity and behavior to static sites JavaScript was invented to do this in 1995. By using AngularJS that can be achieved so easily.
- ✓ **Web Applications:** With technology, browsers have improved to the extent that a language was required to create robust web applications. When we explore a map in Google Maps then we only need to click and drag the mouse. All detailed view is just a click away, and this is possible only because of JavaScript. It uses Application Programming Interfaces (APIs) that provide extra power to the code. The Electron and React is helpful in this department.
- ✓ **Server Applications:** With the help of Node.js, JavaScript made its way from client to server and node.js is the most powerful in the server-side.
- ✓ **Games:** Not only in websites, but JavaScript also helps in creating games for leisure. The combination of JavaScript and HTML 5 makes JavaScript popular in game development as well. It provides the EaseJS library which provides solutions for working with rich graphics.
- ✓ **Smartwatches:** JavaScript is being used in all possible devices and applications. It provides a library PebbleJS which is used in smartwatch applications. This framework works for applications that require the internet for its functioning.
- ✓ **Art:** Artists and designers can create whatever they want using JavaScript to draw on HTML 5 canvas, make the sound more effective also can be used p5.js library.

- ✓ **Machine Learning:** This JavaScript ml5.js library can be used in web development by using machine learning.

Where to put JavaScript in an HTML Document?

JavaScript can be placed inside the body or the head section of an HTML page or inside both head and body.

JavaScript in head

```
<html>
   <head>
      <script>
         function gfg() {
           document.getElementById("demo").innerHTML = "Geeks For Geeks";
         }
      </script>
   </head>
   <body>
      <h2>JavaScript in Head</h2>
      <p id="demo" style="color:green;">geeksforgeeks.</p>
      <button type="button" onclick="gfg()">click it</button>
   </body>
</html>
```

JavaScript in body

```
<html>
   <center>
      <body>
         <h2>JavaScript in Body</h2>
         <p id="demo">geeksforgeeks.</p>
         <button type="button" onclick="gfg()">Try it</button>
         <script>
            function gfg() {
              document.getElementById("demo").innerHTML = "Geeks For Geeks";
            }
         </script>
      </body>
   </center>
</html>
```

External JavaScript

```
<html>
   <center>
   <body>
      <h2>External JavaScript</h2>
      <p id="demo">Geeks For Geeks.</p>
      <button type="button" onclick="myFunction()">Try it</button>
      <script src="myScript.js"></script>
   </body>
   <center>
</html>
```

JavaScript | Statements

The programming instructions written in a program in a programming language are known as statements. Order of execution of Statements is the same as they are written.

Ex:1

```
<html>

<body>
    <h2>Welcome</h2>

    <p id="dev"></p>

    <script>

        var a, b, c;


        a = 2;


        b = 3;


        c = a + b;

 document.getElementById("dev").innerHTML = "The value of c is " + c + ".";

    </script>

</body>

</html>
```

Ex:2

```
<html>
<body>

<h2>JavaScript Variables</h2>

<p>In this example, x, y, and z are variables.</p>

<p id="demo"></p>

<script>
var x = 5;
var y = 6;
var z = x + y;
document.getElementById("demo").innerHTML =
"The value of z is: " + z;
</script>
</body>
</html>
```

Code Blocks:

JavaScript statements can be grouped together inside curly brackets. Such groups are known as **code.**

Ex:1

```
<html>
<body>
    <p>Welcome</p>
    <button type="button"
            onclick="myFunction()">
      Click Me!
  </button>

    <p id="dev1"></p>
    <p id="dev2"></p>

    <script>
        function myFunction() {

document.getElementById("dev1").innerHTML = "Hello";

document.getElementById("dev2").innerHTML = "How are you?";
        }
    </script>

</body>

</html>
```

JavaScript Variables: A JavaScript variable is the simple name of storage location where data to be stored. There are two types of variables in JavaScript which are listed below:

- ✓ **Local variables:** Declare a variable inside of block or function.
- ✓ **Global variables:** Declare a variable outside function or with window object.

```
<script>

// Gloabal variable declaration
var Name="Apple";

function MyFunction() {

    // Local variable declaration
    var num = 45;

    // Display the value of Gloabal variable
    document.writeln(Name);

    // Display the value of local variable
    document.writeln("<br>" + num );
}

MyFunction();

</script>
```

JavaScript Operator:

JavaScript operators are symbols that are used to compute the value or in other word we can perform operations on operands. Arithmetic operators (+, -, *, /) are used to compute the value and Assignment operator (=, +=, %=) are used to assign the values to variables.

Ex:1

```
<script>

var x, y, sum;
x = 3;
y = 23;

sum = x + y;

document.write(sum);

</script>
```

JavaScript | Output

JavaScript Output 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:

- ✓ **innerHTML:** It is used to access an element. It defines the HTML content

JS: document.getElementById(id)

Ex:1

```
<html>

<head>
    <title>
        JavaScript Output using innerHTML
    </title>
</head>

<body>
    <h1>Welcome to Javascript </h1>

    <h2>
        JavaScript Display Possibilities
        Using innerHTML
    </h2>

    <p id="DN"></p>

    <!-- Script to uses innerHTML -->
    <script>
        document.getElementById("DN").innerHTML = 10 * 2;
    </script>
</body>

</html>
```

EX:2

```
<html>

<head>
    <title>
        JavaScript Output using window.alert()
    </title>
</head>

<body>
    <h1>Welcome window.alert</h1>

    <h2>
        JavaScript Display Using window.alert()
    </h2>

    <p id="DN"></p>

    <script>
        window.alert(10 * 2);
    </script>
</body>

</html>
```

BASIC FORM OPERATORS IN JAVASCRIPT FUNCTION

USING VARIABLES AND OPERATOR

EX:1

```
<html>

<head>
    <title>
        JavaScript Comments
    </title>

    <script>


        function add() {


            var x, y, z;


            x = Number(document.getElementById("num1").value );

            y = Number(document.getElementById("num2").value );


            z= x +y;


            document.getElementById("sum").value = z;
        }
    </script>
</head>

<body>
    First number: <input id="num1"><br><br>
    Second number: <input id="num2"><br><br>

    <button onclick="add()">Sum</button>

    <input id="sum">
</body>

</html>
```

JavaScript | Events

JavaScript has events to provide a dynamic interface to a webpage. These events are hooked to elements in the Document Object Model (DOM).

There are some JavaScript events:

1) onclick events: This is a mouse event and provokes any logic defined if the user clicks on the element, it is

EX:1

```
<html>
  <head>
    <script>
      function hiThere() {
        alert('Hi there!');
      }
    </script>
  </head>
  <body>
    <button type="button" onclick="hiThere()">Click me event</button>
  </body>
</html>
```

2) onkeyup event: This event is a keyboard event and executes instructions whenever a key is released after pressing.

EX:2

```
<html>
  <head>
    <script>
      var a = 0;
      var b = 0;
      var c = 0;
      function changeBackground() {
        var x = document.getElementById('bg');
        bg.style.backgroundColor = 'rgb('+a+', '+b+', '+c+')';
        a += 1;
        b += a + 1;
        c += b + 1;
        if (a > 255) a = a - b;
        if (b > 255) b = a;
        if (c > 255) c = b;
      }
    </script>
  </head>
  <body>
    <input id="bg" onkeyup="changeBackground()"
      placeholder="write something" style="color:#fff">
  </body>
</html>
```

3) onmouseover event: This event corresponds to hovering the mouse pointer over the element and its children, to which it is bound to.

EX:3

```
<html>
  <head>
    <script>
       function hov() {
         var e = document.getElementById('hover');
         e.style.display = 'none';
      }
    </script>
  </head>
  <body>
    <div id="hover" onmouseover="hov()"
      style="background-color:green;height:200px;width:200px;">
    </div>
  </body>
</html>
```

4) onmouseout event: Whenever the mouse cursor leaves the element which handles a mouseout event, a function associated with it is executed.

```
<html>
  <head>
    <script>
      function out() {
        var e = document.getElementById('hover');
        e.style.display = 'none';
      }
    </script>
  </head>
  <body>
    <div id="hover" onmouseout="out()"
      style="background-color:green;height:200px;width:200px;">
    </div>
  </body>
</html>
```

5) onchange event: This event detects the change in value of any element listing to this event.

EX:1

```
<html>
  <head></head>
  <body>
    <input onchange="alert(this.value)" type="number">
  </body>
</html>
```

6) onload event: When an element is loaded completely, this event is evoked.

```
<html>

  <head></head>

  <body>

    <img onload="alert('Image completely loaded')"

      alt="Logo"

      src="AddImage">

  </body>

</html>
```

7) onfocus event: An element listing to this event executes instructions whenever it receives focus.

```
<html>
  <head>
    <script>
      function focused() {
        var e = document.getElementById('inp');
        if (confirm('Got it?')) {
          e.blur();
        }
      }
    </script>
  </head>
  <body>
    <p >Take the focus into the input box below:</p>

    <input id="inp" onfocus="focused()">
  </body>
</html>
```

8) onblur event: This event is evoked when an element loses focus.

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

<p>Write something in the input box and then click elsewhere
       in the document body.</p>

    <input onblur="alert(this.value)">
  </body>
</html>
```

JavaScript Data Types

Data types basically specify what kind of data can be stored and manipulated within a program.

There are **six basic** data types in JavaScript which can be divided into three main categories:

primitive (primary**), composite** (reference) and **special** data types.

String, Number and Boolean are **primitive** data types.

Object, Array, and Function are **composite** data types.

Whereas Undefined and Null are **special** data types.

Primitive data types can hold only one value at a time, whereas composite data types can hold collections of values and more complex entities.

The null type is the second primitive data type that also has only one value: **null**. JavaScript defines that null is equal to undefined.

JavaScript was originally developed as LiveScript by Netscape in the mid-1990s. It was later renamed to JavaScript in 1995 and became an ECMA standard in 1997. Now JavaScript is the standard client-side scripting language for web-based applications, and it is supported by virtually all web browsers available today, such as Google Chrome, Mozilla Firefox.

JavaScript is an object-oriented language, and it also has some similarities in syntax to Java programming language. But JavaScript is not related to Java in any way.

JavaScript is officially maintained by ECMA (European Computer Manufacturers Association).

(ES6) is the latest major version of the ECMAScript standard.

Except VAR One of the features that came with ES6 is the addition of LET and CONST, which can be used for variable declaration as well.

Data Types

In programming, data types are an important concept. To be able to operate on variables, it is important to know something about the types:

STRING DATA:

```
<body>

<h2>JavaScript Strings</h2>

<p>Strings are written </p>

<p id="demo"></p>

<script>

let carName1 = "Volvo XC60";

let carName2 = 'Volvo XC60';

document.getElementById("demo").innerHTML =

carName1 + "<br>" +

carName2;

</script>

</body>

</html>
```

NUMBER DATA TYPE

Ex:2

```
<html>
<body>
<h2>JavaScript Numbers</h2>
<p>Numbers can be written </p>
<p id="demo"></p>
<script>
let x1 = 34.00;
let x2 = 34;
let x3 = 3.14;
document.getElementById("demo").innerHTML =
x1 + "<br>" + x2 + "<br>" + x3;
</script>
</body>
</html>
```

Booleans Data Type

Ex:3

```
<html>
<body>
<h2>JavaScript Booleans</h2>
<p>Booleans can have two values: true or false:</p>
<p id="demo"></p>
<script>
let x = 5;
let y = 5;
let z = 6;
document.getElementById("demo").innerHTML =
(x == y) + "<br>" + (x == z);
</script>
</body>
</html>
```

Array Data Type:

Ex: 4

```
<html>
<body>

<h2>JavaScript Arrays</h2>
<p>Array indexes are zero-based, which means the first item is [0].</p>
<p id="demo"></p>
<script>
const cars = ["Saab","Volvo","BMW"];
document.getElementById("demo").innerHTML = cars[0];
</script>
</body>
</html>
```

Object Data Type:

Ex:5

```
<html>
<body>
<h2>JavaScript Objects</h2>
<p id="demo"></p>
<script>
const person = {
  firstName : "Durganand",
  lastName  : "Panjiyar",
  age     : 39,
  eyeColor  : "Royal Blue"
};
document.getElementById("demo").innerHTML =
person.firstName + " is " + person.age + " years old.";
</script>
</body>
</html>
```

Undefined

```
<html>

<body>


<h2>JavaScript</h2>


<p>The value of a variable with no value is <b>undefined</b>.</p>


<p id="demo"></p>


<script>

let car;

document.getElementById("demo").innerHTML =

car + "<br>" + typeof car;

</script>


</body>

</html>
```

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