# JavaScript

**JavaScript:-**

JavaScript is a lightweight, interpreted programming language with object oriented capabilities. It is an interpreted language, usually embedded directly onto HTML pages

**Benefits Of Javascript**

1. It is widely supported by web browser.
2. It gives easy access to the document object.
3. It can give interesting animations without long download times
4. It can't get a virus infection directly from javascript
5. Javascript allow many page effects
   a) User page time in/ out
   b) popups & tooltips
   c) Embedded audio
   d) scrolling banners
   e) print pages

**Advantages of javascript:**

1. **An Interpreted Language**: which requires no compilation steps the browser just as it interprets HTML tags
2. **Embedded within HTML**:- Javascript does not requires any special or separate editor for programs written
   It can be written along with HTML tags in notepad
3. **Javascript can react to events**: To execute when something happens, like a user click on HTML buttons
4. **Javascript can read and write HTML elements:** A javascript can read & change the content of HTML elements

**Javascript can be used to validate data**: Using javascript we can validate data for example enter valid email – id, enter valid name, mobile number etc.

**Inserting java script into a HTML Page:**

The following are the basic rules to be followed while developing JavaScript code.

1. Java script statements end with semi-colon
2. Java script is case – sensitive
3. Java script support two forms of comments
   → Single – line comment begin with (//)
   → multi line comment begin with (/*) and end with (*/)
4. Block of code must be surrounded by a pair of braces ({ })
5. Functions have parameters which are passed inside parenthesis ()

**Including Java Script into HTML:**

- Java Script is placed between tags starting with **<script language = "javascript">** and end with **</script>**
- Java script can be placed in various locations in HTML
  - ↪ Java script in **HEAD** section
  - ↪ Java script in **BODY** section
  - ↪ Java script in both **HEAD** and **BODY** section
  - ↪ Java script in **External File (with .js )**
- Java script placed in the HEAD section of HTML will be executed when called.
- Java script placed in the BODY section of HTML will be execute only when the page is loaded

Java script in the **HEAD section:**

```
<html>
<head>
<script language="javascript">
    ----  java script code here-------
</script>
</head>
</body> ----------</body>
</html>
```

Java script in the **BODY section**

```
<html>
<head>    </head>
   <body>
            <script language="javascript">
               -------  java script code here-------
            </script>
      </body>
</html>
```

**Link Java script** file in HTML

```
<html>
<head>
   <script language="javascript"   src ="filename.js">
    </script>
    </head>
      </body> --------------</body>
</html>
```

**VARIABLES in Java Script:-**

- ✓ Variables are used to hold data in memory. It is a name of the memory location.
- ✓ Before using a variable we must declare it in a JavaScript program.
- ✓ Variable are declared with the **var** key word in JavaScript.

**Ex:-**

```
<script   language="javascript">
  var     rollno;
  var     sname;
</scritp>
```

- ✓ Multiple variable declaration var rollno, sname, avg;
- ✓ **Global Variable** : A global variable will be available in entire program.
- ✓ **Local Variable**: A local variable will be visible only within a function where it is defined.

```
<script   language="javascript">
      var     rollno=10;     // global variable
  function  display()
  {
      var  rollno=20;     // local variable
  }
</scritp>
```

**Variable Rules:-**

- Javascript variable name starts with letter(a-z), should not start with number(0-9)
- Cannot use spaces in between names.
- An underscore( _ ) can be used between multiple words.
- Variable names are case sensitive.
- Reserved words are not used as a variable name.

**Ex:- some valid variables: var rno; var s_name;**

**DATA TYPES :-**

JavaScript uses 4 data types number, string, Boolean, null

1) **Number:-** It is possible to express both integers and floating point values.
**Ex:- 23, 44, -5.6, 6.7**
2) **String:-** These are collection of characters. Indicated with single or double quoted.
**Ex:- ‘ murthy’ “sailu”**
3) **String:-** Variables hold the values TRUE and FALSE
**Ex:- boolean res=true;**
4) **NULL :-** The null value represents just that – nothing. It does not mean nil or zero.

**STRING MANIPULATIONS:**

1. **charAt():** This method returns the character from the specified index. (First character index is 0) (Last character index length-1)
   **Ex:-** var str = new String(“welcome”):
   **str.charAt(2);** **Output:** l
2. **concat():** This method adds two or more Strings and returns a new String.
   **Ex:** var str1=”welcome to”, str2 =”Java”;
   **var str3 = str1. concat(str2);**
3. **indexOf():** This method returns the index within the calling String object of the first occurrence of the specified value if not found returns (-1)
   **Ex**: var str1 = new String (“This is string one”);
   **var index=str1.indexOf(“String”);**
4. **toLowerCase():** This method returns the calling String value converted to lower case
   Syntax: String.toLowerCase()
   Ex: var str = new String(“WELCOME”);
   **str.toLowerCase();**
5. **toUpperCase():** This method returns the calling String value converted to uppercase
   Syntax: String.toUpperCase()
   Ex: var str = new String(“welcome”);
   **str. toUpperCase();**
6. **substring():** This method is used to take a part of a String
   **Syntax**: string.substring(start , length]);
   **Ex**: var str = “welcome”;
   **str.substring(1,2);** **Output:** el
7. **length():** This property returns the number of characters in the given string
   **Ex:** var str = “welcome”;
   **int len = str. length( );**
8. **replace():-** This method replaces one string with another String. Searches for pattern1, if the search is successful pattern1 is replaced with pattern2.
   **Syntax**: string. replace(oldstring, newstring);

Ex: var str = “welcome to java”

**var res = str.replace(“java, ”html” );**

**Output:** welcome to html

```
<html>
        <head><title> string functions</title></head>
<body>
                <h1 align=center><u> String Functions</u></h1>
    <script language="JavaScript">
        var str = new String("Hello World");
        document.writeln("<br><br>Length :"   +str.length);
        document.writeln("<br><br>Upper Case:"   +str.toUpperCase());
        document.writeln("<br>Lower Case :"    +str.toLowerCase());
        document.writeln("<br><br>indexOf String :"  +str.indexOf("l"));
        document.writeln("<br><br>Substring:"  +str.substring(1,2));
        var rep=str.replace("World","JavaScript");
        document.writeln("<br>Replace String :"+rep);
    </script>
 </bdoy>
</html>
```

**MATHEMATICAL FUNCTIONS:**

- The Math object provides properties and methods for mathematical constants and functions
- Math is static and can be called by using Math as an object without creating it

  Syntax:

  **var pival = Math.PI;**

  var sinval = Math.sin(30);

**Mathematical Functions:**

1. **abs():** Returns the absolute value of a given number

   **Ex: var r = Math.abs(-1);**

   document.write(r); output: 1

2. **max():** Returns the largest of zero or more number

   **Ex: var r = Math.max(10, 20, 30);**

   document. write(r) output: 30

3. **min():** Returns the smallest of zero or more number

   **Ex: var r = math.min(10, 20, 30);**

   document. write(r) ; output: 10

4. **pow():** Returns base to the exponent power, that is base exponent

   **Ex: var r = Math.pow(2,3);**

   document. write(r) ; output: 8

5. **round():** Returns the value of a number rounded to the nearest integer

   **Ex: var r = Math.round(0.5);** output: 1

6. **sqrt():** returns the square root of a number

   **Ex: var r = Math.sqrt(16);** output:4

7. **ceil():** Returns the smallest integer greater than or equal to a number
   **Ex: var r = Math. ceil(45.20);** output: 46

8. **floor():** Returns the largest integer less than or equal to a number
   **Ex: var r= Math.floor(10.3);** output:10
   document. write(r);

9. **Math.PI:** Returns the PI value 3.14159
   **Ex: var r = Math.PI;**
   document. write(r); output:3.14159

10. **sin():** Returns the sine of a number
    **Ex: var r = Math.sin(30);**

11. **cos():** Returns the cosine of a number
    **Ex: var r = Math.cos(30);**

12. **tan():** Returns the tangent of a number
    **Ex: var r = Math.tan(45);**

13. **asin():** Returns the arcsine (in radians) of a number
    **Ex: var r = Math. asin(30);**

14. **acos():** Returns the arccosine(in radians) of a number
    **Ex: var r= Math. acos(30);**

15. **atan():**Returns the arctangent(in radians) of a number
    **Ex: var r = Math. atan(45);**

16. **random():** Returns the random number between 0 to 1.
    **Ex: var r = Math. random();**

**OPERATORS IN JavaScript**

**Ans:** Javascript language supports following operators

1. **Arithmetic Operators**
2. **Relational (comparison) operators**
3. **Logical operators**
4. **Assignment operators**
5. **Conditional operators**
6. **The + operator used on strings**

1. **Arithmetic Operator:-** These operators take numerical values as their operands and return a single value. (a, b variables holds a= 10, b=20 then)

| Operator | Description | Example |
|---|---|---|
| + | Addition | a + b will give 30 |
| - | Subtraction | a - b will give -10 |
| * | Multiplication | a * b will give 200 |
| / | Division | b/a will give 2 |
| % | Modulation | b%a will give 0 |
| ++ | Increment operator, increases one value | a++ will give 11 |
| -- | Decrement operator decrease one value | a - -will give 9 |

2. **Relational Operators:** These operators compare operands and returns a logical value (true/false) (Assume variable a=10 and b = 20 then)

| Operator | Description | Example |
|---|---|---|
| = = | Equal operator | (a==b) is not true |
| != | Not equal operator | (a!=b) is true |
| > | Greater than | (a>b) is not true |
| < | Less than t | (a<b) is true |
| >= | Greater than or equal operator | (a>=b) is not true |
| <= | Less than or equal operator | (a<=b) is true |

3. **Logical Operators:** These operators are typically used with Boolean (logical) values returns (true/False) values
(Assume variable a = 10, b=20 then)

| Operator | Description | Example |
|---|---|---|
| && | Logical AND operator (all conditions must true) | (a&&b) is true |
| \|\| | Logical OR operator. If any one condition is true | (a\|\|b) is true |
| ! | Logical NOT operator (it converts reverse results) | !(a&&b) is false |

4. **Assignment Operators:** The assignment operators(=) to assign a value to a variable or constant or expression assigned for given variable

| Operator | Description | Example |
|---|---|---|
| = | Assign operator | c=a+b |
| += | Shortcut Addition assignment | c+=a is equal to(c=c+a) |
| -= | Shortcut subtraction assignment | c-=a is equal to (c=c-a) |
| *= | Shortcut multiplication assignment | C*=a is equal to (c=c*a) |
| /= | Shortcut division assignment | c/=a is equal to(c=c/a) |

5. **Conditional Operator(?:) :** Which is used for comparing two expressions, also contains a conditional operator that assigns a value to a variable based on some condition
**Syntax:** variable = (condition)? Value1:value2;
**Ex:** result = (marks>=35)?"passed":"failed";

6. **The "+" operator used on strings:** The '+' operator can also be used to add string variables or text values together
**Ex**: txt1 = 'aditya";
txt2 = "Degree college";
txt3 = txt1+txt2;

**FUNCTIONS IN JavaScript**

A function is a piece of code that performs a specific task. You may call a function from anywhere within a page.

Functions can be defined both in the <head> and in the <body> section of a document.

**Defining Functions:**

Functions are defined using the function key word. The function name can be any combination of digits, letters and underscore but not a white space

**Syntax:**

```
function  funname(parameters….)
{
    Body of functioin…..
}
```

**Example:-**

```
< script language = "javascript">
            fuction display()
             {
                alert("hello");
             }
  </script>
```

**Parameter passing to functions:** When a function receives a value as a parameter. The parameter are taken from the function definition.

The "return" key word is used to return values from functions.

- The function can return only single value

**Example:-**

```
  <script language = "javascript">
        document. write("The product of 2 no's:"+product(5,3));
       function product(a,b)
        {
            return a * b;
        }
 </script>
```

**Program-1:-**

```
        <html>
           <head>
                <script language = "javascript">
        document. write("The product of 2 no's:"+product(5,3));
        function product(a, b)
        {
            return a * b;
        }
 </script>
</head>
<body> <h1> Function Returning Demo </h1>
</body>
</html>
```

**Program-2 :**

```
    <html>
         <head><title> function demo</title>
          < script language = "javascript">
            fuction display()
             {
                alert("hello");
             }
          </script>
        </head>
     <body>
       <form name = "f1">
         <input type = "botton"   value = "click me"   onclick = "display()">
       </form>
 </body>
 </html>
```

The above program whenever user hit the input button immediately display() function will be execute and the message box with "hello" message will be displayed on the browser window.

**Objects in JavaScript:**

1. Built-in objects provide information about
   - ➔ Currently loaded web page
   - ➔ Its contents
   - ➔ Current session of Navigator
   - ➔ Methods for working with properties.
2. Built-in objects in java script are part of the Navigator object hierarchy.
3. The following are some of the built-in objects in Java script.

a. Navigator – The navigator object provides information about the current browser.
b. Window – The window object provides methods and properties to work with navigator window which includes objects for each frame.
c. Location – The location object provides methods and properties to work with the URL.
d. History – The history object provides methods and properties about the history list and previous, next visited web page information.
e. Document – The document object is the most frequently used java script object, which contains methods and properties to work with form elements, links and applets.

**DOCUMENT OBJECT:**

The following are the some of the properties associated with document object:

1. bgColor -- used to set or get the background color
2. fgColor -- used to set or get the foreground color
3. anchor -- object reference of an anchor contained in the document
4. alinkColor -- used to set or get the value of alink attribute
5. form -- object reference of a form contained in the document
6. forms -- used to get an array of all the form objects contained in the document
7. image -- object reference of an image contained in the document
8. images -- used to get an array of all the image objects contained in the document
9. link -- object reference of a link contained in the document
10. links -- get an array of all the link object references contained in the document

The following are the some of the Methods associated with document object:

1. close( ) -- used to close a document stream opened using document.open( )
2. getElementById( ) -- used to access any element on the page using ID attribute.
   **document.getElementBbyId("ID").value;**
3. open( ) -- used to open a document stream to display or write something.
   **document.open( )**
4. write( ) -- used to write a given string on to the document.
   **document.write("String");**
5. writeln( ) -- used to write a given string on the document and inserts a new line character at the end.
   **document.writeln("STRING");**

**WINDOW OBJECT:**

Window object is the top-level object for each document, location and history object. The following are the some of the properties associated with window object.

1. closed -- used to identify whether a window is closed or not. Returns true if closed.
2. status -- used to get the current status of the browser window.
3. document -- returns the document object of the current browser page.
4. length -- returns number of frames in a window.
5. name -- used to find the name of the browser window.
6. opener -- used to identify the object from which the window was created.

The following are the methods associated with window object.

1. alert( ) -- used to pop up an alert message.
   **alert("Message");**

2. confirm( ) -- used to display a confirm dialog box.
   **confirm("Message");**
3. prompt( ) -- used to get user input through keyboard.
   **prompt( "Message");**
4. open( ) -- used to open a new window using javascript.
   **window.open(url,name);**
5. close( ) -- used to close a window using javascript.
   **window.close( );**

**DATE OBJECT:**

Date object is used to date and time in milliseconds from 1st January 1970 UTC. The Date object is created in different ways as shown below.

1. Date( ) -- Creates an empty date object.
2. Date(milli-seconds) -- creates a new date object based up on the number of milliseconds since 00:00:00 hours on 1-1-1970.
3. Date(year,month,day[,hour,minute,second]) – Create a new Date object based up on the specified values.

The following are the methods associated with Date Object:

1. date( ) -- returns the current date.
2. getDate( ) -- returns the date between the days 1 and 31.
3. getDay( ) -- returns the day of the week between 0 and 6.
4. getMonth( ) -- returns the month between 0 and 11.
5. getYear( ) -- returns the last two digits of the year.
6. getFullYear( ) -- returns the year as four digit number.
7. getHours( ) -- returns the current hours from 0 to 23
8. getMinutes( ) -- returns the current minutes from 0 to 59
9. getSeconds( ) -- returns the current seconds from 0 to 59
10. getMilliseconds( ) – returns the milliseconds from 0 to 999
11. setDate( ) -- specifies the day of the month from 1 to 31
12. setMonth( ) -- specifies the month from 0 to 11
13. setFullYear( ) -- specifies the four digit year in a date object
14. setHours( ) -- specifies the hours from 0 to 23 in a date object
15. setMinutes( ) -- specifies the minutes from 0 to 59 in a date object.
16. setSeconds( ) -- specifies the seconds from 0 to 59 in a date object.
17. setMilliseconds( ) – specifies the milliseconds from 0 to 999 in a date object.
18. toString( ) -- converts a date object into a string.

**HISTORY OBJECT:**

1. History object contains the complete list of url/links visited during a session.
2. History object methods are used to move forward or backward.

next – represents the next page URL in the history object list.
previous -- represent the previous page URL in the history object list.
current -- represent the current page URL in the history object.
length -- returns the number of objects in the history list.