# JAVA SCRIPT

- Java script is a programming language designed to add interactivity in the HTML pages.

- Using HTML we cannot create interactive web pages.so JavaScript is designed to add interactivity in the HTML pages.
- JavaScript is a scripting language that will allow you to add real programming to your webpages.
- JavaScript is most commonly used as a client side scripting language. This means that JavaScript code is written into an HTML page. When a user requests an HTML page with JavaScript in it, the script is sent to the browser and it's up to the browser to do something with it.

## HISTORY OF JAVASCRIPT

- JavaScript was originally named Mocha and the first version was written by Brendan Eich at Netscape in May 1995.
- The name was changed to LiveScript in September, 1995, and became JavaScript sometime after that.

## Basic Structure of JavaScript

```
<head>
    <script language= "javascript" >
        Code
    </script>
</head>
```

**Simple program using JavaScript**

```
<html>
<head>
<title>my first javascript </title>
</head>
<body>
<script language="javascript">
        document.write("hello javascript");
</script>
</body>
</html>
```

## JavaScript Comments

**COMMENT helps the readers to understand the meaning of the code**

There are two types of comments in JavaScript.

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

**Single line Comment**

**Single line comments start with // (double forward slashes).**

It can be used before and after the statement.

```
<html>
<head>
<title>single line comment </title>
</head>
<body>
<script language="javascript">      //this line tells web browser that this is in javascript
document.write("hello  javascript"); //this line prints the statement which is in double quotes
</script>
</body>
</html>
```

**Multi line Comment**

Multi-line comments start with /* and end with */.

**/* your code here  */**

Any text between /* and */ will be ignored by JavaScript.

It can be used before  and after  of the statement.

```
<html>
<head>
<title>multi line comment </title>
</head>
<body>
```

```
<script language="javascript">

/* It is multi line comment.

It will not be displayed */

   document.write("hello javascript");

</script>

</body>

</html>
```

## JavaScript Variables

- **In a programming language, variables are used to store data values.**
- **JavaScript uses the var keyword to declare variables.**

```
<html>
  <body>
    <script language="javascript" >

          var a,b,c;

          a=10;
           b=20;
           c=30;

          c=a+b;

         document.write("sum of the two variables is " +c);

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

## Statements (or) Control Statements in javascript

**Statements in javascript are of two types namely**

1. **Conditional statements (decision control statements)**
2. **Looping control statements (Iterative control statements)**

### Conditional statements in java script

In JavaScript we have the following conditional statements:

- Use **if** to specify a block of code to be executed, if a specified condition is true
- Use **else** to specify a block of code to be executed, if the same condition is false
- Use **else if** to specify a new condition to test, if the first condition is false
- Use **switch** to specify many alternative blocks of code to be executed.

### If statement

```
<script language="javascript"  >

  var mood = "happy";

  if(mood == "happy" )
     {
        document.write("i am in happy mood");
     }

</script>
```

### If –else statement

```
<script language="javascript" >

   var mood = "happy";

    if(mood == "happy" )
       {
          document.write("i am in happy mood");
       }
 else
     {
        document.write("i am in a sad mood");
     }

</script>
```

**If-elseif-else statement (nested if-else)**

```
<script language="javascript"  >

    var mood = "happy";

    if(mood == "happy" )
        {
      document.write("i am in happy mood");
    }

    else if( mood == "sad" )
    {
      document.write("i am in a sad mood");
    }

    else
    {
      document.write("i am neither happy nor sad ");
    }
</script>
```

**Switch statement**

```
<script language="javascript" >

    var favcolor = "pink";

      switch(favcolor)
              {

                      case "red":

                                  document.write("your favourite color is red");
                                  break;

                     case "blue":

                                 document.write("your favourite color is blue");
                                 break;
```

```
                case "green":

                        document.write("your favourite color is green");
                        break;

                default:

                   document.write("your favourite color is neither red,blue nor green");

            }

        </script>
```

**Loops in java script**

**Looping control statements are also called as iterative control statements**

**While:** loop through a block of code as long as a specified condition is true

**Do-while**: loop through a block of code once,and then repeats the loop as long as the specified condition is true

**For** : loop through a block of code a specified number of times.

**For loop**

```
<script language="javascript">

    var x;

     for(x=0;x<=5;x++)
      {

      document.write("the number is" +x);

      document.write("<br>");
      }

    </script>
```

**While loop**

```
<script language="javascript">

    var x=0;

     while(x<=5)
        {

              document.write("the number is" +x);

              document.write("<br>");

              x++;
        }
     </script>
```

**Do-while loop**

```
    <script language="javascript">


      var x=0;

    do
          {
                document.write("the number is" +x);

                  document.write("<br>");

                    x++;

           }while(x<=5);

     </script>
```

# Operators in java script

JavaScript supports the following types of operators.

1. Arithmetic Operators
2. Comparision Operators
3. Logical (or Relational) Operators
4. Assignment Operators
5. Conditional (or ternary) Operators

## 1. Arithmetic Operators

JavaScript supports the following arithmetic operators

Assume **variable A holds 10 and variable B holds 20**, then

| Sr.No | Operator and Description |
|---|---|
| 1 | **+ (Addition)**<br>Adds two operands<br>**Ex:** A + B will give 30 |
| 2 | **- (Subtraction)**<br>Subtracts the second operand from the first<br>**Ex:** A - B will give -10 |
| 3 | *** (Multiplication)**<br>Multiply both operands<br>**Ex:** A * B will give 200 |
| 4 | **/ (Division)**<br>Divide the numerator by the denominator<br>**Ex:** B / A will give 2 |

| | |
|---|---|
| 5 | **% (Modulus)**<br>Outputs the remainder of an integer division<br>**Ex:** B % A will give 0 |
| 6 | **++ (Increment)**<br>Increases an integer value by one<br>**Ex:** A++ will give 11 |
| 7 | **-- (Decrement)**<br>Decreases an integer value by one<br>**Ex:** A-- will give 9 |

**2. Comparison Operators**

JavaScript supports the following comparison operators

Assume **variable A holds 10 and variable B holds 20**, then

| Sr.No | Operator and Description |
|---|---|
| 1 | = = **(Equal)**<br>Checks if the value of two operands are equal or not, if yes, then the condition becomes true.<br>**Ex:** (A == B) is not true. |
| 2 | **!= (Not Equal)**<br>Checks if the value of two operands are equal or not, if the values are not equal, then the condition becomes true.<br>**Ex:** (A != B) is true. |

| | |
|---|---|
| 3 | **> (Greater than)**<br>Checks if the value of the left operand is greater than the value of the right operand, if yes, then the condition becomes true.<br>**Ex:** (A > B) is not true. |
| 4 | **< (Less than)**<br>Checks if the value of the left operand is less than the value of the right operand, if yes, then the condition becomes true.<br>**Ex:** (A < B) is true. |
| 5 | **>= (Greater than or Equal to)**<br>Checks if the value of the left operand is greater than or equal to the value of the right operand, if yes, then the condition becomes true.<br>**Ex:** (A >= B) is not true. |
| 6 | **<= (Less than or Equal to)**<br>Checks if the value of the left operand is less than or equal to the value of the right operand, if yes, then the condition becomes true.<br>**Ex:** (A <= B) is true. |

### 3. Logical Operators

JavaScript supports the following logical operators

Assume **variable A holds 10 and variable B holds 20**, then

| Sr.No | Operator and Description |
|---|---|
| 1 | **&& (Logical AND)**<br>If both the operands are non-zero, then the condition becomes true.<br>**Ex:** (A && B) is true. |

| | |
|---|---|
| 2 | **\|\| (Logical OR)**<br>If any of the two operands are non-zero, then the condition becomes true.<br>**Ex:** (A \|\| B) is true. |
| 3 | **! (Logical NOT)**<br>Reverses the logical state of its operand. If a condition is true, then the Logical NOT operator will make it false.<br>**Ex:** ! (A && B) is false. |

**4. Assignment Operators**

JavaScript supports the following assignment operators

| Sr.No | Operator and Description |
|---|---|
| 1 | **= (Simple Assignment )**<br>Assigns values from the right side operand to the left side operand<br>**Ex:** C = A + B will assign the value of A + B into C |
| 2 | **+= (Add and Assignment)**<br>It adds the right operand to the left operand and assigns the result to the left operand.<br>**Ex:** C += A is equivalent to C = C + A |
| 3 | **–= (Subtract and Assignment)**<br>It subtracts the right operand from the left operand and assigns the result to the left operand.<br>**Ex:** C -= A is equivalent to C = C - A |

| | |
|---|---|
| 4 | ***= (Multiply and Assignment)**<br>It multiplies the right operand with the left operand and assigns the result to the left operand.<br>**Ex:** C *= A is equivalent to C = C * A |
| 5 | **/= (Divide and Assignment)**<br>It divides the left operand with the right operand and assigns the result to the left operand.<br>**Ex:** C /= A is equivalent to C = C / A |
| 6 | **%= (Modules and Assignment)**<br>It takes modulus using two operands and assigns the result to the left operand.<br>**Ex:** C %= A is equivalent to C = C % A |

**5. <u>Conditional Operator (? :)</u>**

The conditional operator first evaluates an expression for a true or false value and then executes one of the two given statements depending upon the result of the evaluation.

| Sr.No | Operator and Description |
|---|---|
| 1 | **? : (Conditional )**<br>If Condition is true? Then value X : Otherwise value Y |

# JavaScript Popup Boxes(Dialog boxes or message boxes)

**JavaScript Popup boxes is three types**

## Alert Box :

**Alert Box simply display a message to user on browser.**

The alert box pops up with an OK button which user has to press to continue

Syntax : **alert("textmessage");**

```
<script type="text/javascript">
    alert("welcome to java script");
</script>
```

## Confirm Box:

**A confirm box is often use to verify or accept some confirm message and display on browser.**
When a confirm box display on browser, the user allow to click either "OK" or "Cancel" to proceed.

If the user clicks "OK", the box returns true. If the user clicks "Cancel", the box returns false.

Syntax: **confirm("textmessage");**

```
<script language="javascript">

        msg="if you do hard work";

        x = confirm(msg);
        if (x == true)
                {
                        alert("You will get good marks.");
                }
        else
                {
                        alert("You will get less marks.");
                }
</script>
```

## Prompt Box :.

**It is used when the user should input a value before entering the page.**

The user must click either "OK" or "Cancel" to proceed after entering the text

Syntax: **prompt("textmessage","Default_Value");**

```
<script language="javascript">

        visiter_name = prompt("Input your name : ");

        if( visiter_name != " ")
            {
                alert("Your Name is : "+visiter_name);
            }

        else
            {

                alert("Blank name ...!");
            }

</script>
```

# JavaScript Strings

**The JavaScript string is an object that represents a sequence of characters**.

There are 2 ways to create string in JavaScript

1. **By string literal**
2. **By string object (using new keyword)**

### 1) By string literal

The string literal is created using double quotes.

Syntax:

**var stringname="string value";**

```
<script language="javascript">
    var str="This is string literal";
    document.write(str);
</script>
```

### 2) By string object (using new keyword)

The syntax of creating string object using new keyword is given below:

**var stringname=new String("string literal");**

```
<script>
    var stringname=new String("hello javascript string");
    document.write(stringname);
</script>
```

# JavaScript String Methods

## (String manipulation (Or) string handling Functions)

**1) charAt(index)**

The JavaScript String charAt() method returns the character at the given index.

```
<script>
        var str="javascript";
        document.write(str.charAt(2));       // v
</script>
```

**2) concat(str)**

The JavaScript String concat(str) method concatenates or joins two strings.

```
<script>
        var s1="javascript ";
        var s2="is a scripting language";
        var s3=s1.concat(s2);
        document.write(s3);    //javascript is a scripting language
</script>
```

**3) indexOf(str)**

indexOf() method returns the position of the first occurrence of a specified value in a string.

```
<script language="javascript">
                var str = "hello hello hello";
                var n = str.indexOf("hello");
                document.write(n);                  // 0
</script>
```

**4) lastIndexOf(str)**

lastIndexOf() method returns the position of the last occurrence of a specified value in a string.

```
<script language="javascript">
        var str = "hello hello hello";
        var n = str.lastIndexOf("hello");
        document.write(n);              // 12
</script>
```

**5) toLowerCase()**

The JavaScript String toLowerCase() method returns the given string in lowercase letters.

```
<script>
    var s1="HELLO WORLD";
    var s2=s1.toLowerCase();
    document.write(s2);          //hello world
</script>
```

**6) toUpperCase()**

The JavaScript String toUpperCase() method returns the given string in uppercase letters.

```
<script>
    var s1="hello world";
    var s2=s1.toUpperCase();
    document.write(s2);                 //HELLO WORLD
</script>
```

**7) slice(beginIndex, endIndex)**

The slice() method extracts parts of a string and returns the extracted parts in a new string.

Use the start and end parameters to specify the part of the string you want to extract.

The first character has the position 0, the second has position 1, and so on.

**Note:** Use a negative number to select from the end of the string.

```
<script>
        var s1="abcdefgh";
        var s2=s1.slice(2,5);
        document.write(s2);
</script>
```

**Output: cde**

**8) trim()**

The JavaScript String trim() method removes leading and trailing whitespaces from the string.

```
<script>
        var s1="    javascript trim    ";
        var s2=s1.trim();
        document.write(s2);
</script>
```

**Output: javascript trim**

# Mathematical Functions in java script

Math object allows you to perform several mathematical tasks.

**Math.abs(x)**

Returns the absolute value of x

**Math.ceil(x)**

returns the value of x rounded **up** to its nearest integer

**Math.exp(x)**

Returns the value of Ex

**Math.floor(x)**

returns the value of x rounded **down** to its nearest integer

**Math.log(x)**

Returns the natural logarithm (base E) of x

**Math.max(x, y, z, ..., n)**

Returns the number with the highest value

**Math.min(x, y, z, ..., n)**

Returns the number with the lowest value

**Math.pow(x, y)**

Returns the value of x to the power of y

**Math.random()**

Returns a random number between 0 and 1

**Math.round(x)**

Rounds x to the nearest integer

**Math.sqrt(x)**

Returns the square root of x

**Math.sin(x)**

Returns the sine of x (x is in radians).

If you want to use degrees instead of radians, you have to convert degrees to radians:

**Angle in radians = Angle in degrees x PI / 180.**

**Math.cos(x)**

Returns the cosine of x (x is in radians)

**Math.tan(x)**

Returns the tangent of an angle

```
<script language="javascript">
        var  x = 25;
        document.write( Math.log(x)+ "<br>");          // 3.2188758248682006
        document.write( Math.sqrt(x)+ "<br>");          //5
        document.write(Math.pow(x,2)+ "<br>");          //625

        var x = 25.67;
        document.write(Math.floor(x)+ "<br>");          //25
        document.write(Math.ceil(x)+ "<br>");           //26
        document.write(Math.round(x)+ "<br>");          //26

        var x = -25
        document.write(Math.abs(x)+ "<br>");            //25

        var maximum = Math.max(23, 34, 43, 89);         //89
        document.write( maximum + " <br>");

        var minimum = Math.min(23, 34, 43, 89);         // 23
        document.write( minimum + " <br>");

        document.write( Math.PI+ "<br>");               //3.141592653589793

        document.write(Math.sin(90 * Math.PI / 180)+"<br>");            // 1
</script>
```

**OUTPUT:**

```
3.2188758248682006
5
625
25
26
26
25
89
23
3.141592653589793
1
```

## Arrays in Javascript

JavaScript array is an object that represents a collection of similar type of elements.

There are 3 ways to construct array in JavaScript

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

**1) JavaScript array literal**

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

**var arrayname=[value1,value2.....valueN];**

As you can see, values are contained inside [ ] and separated by , (comma).

**Example of creating and using array in JavaScript**.

```
<script>
        var emp=["Sonoo","Vimal","Ratan"];
        for (i=0;i<emp.length;i++)
                {
                        document.write(emp[i] + "<br>");
                }
</script>
```

**Note:**

The .length property returns the length of an array.

**Output of the above example**

Sonoo

Vimal

Ratan

### 2) JavaScript Array directly (new keyword)

The syntax of creating array directly is given below:

```
var arrayname=new Array();
```

Here, new keyword is used to create instance of array.

Let's see the example of creating array directly.

```
<script language="javascript">
        var i;
        var emp = new Array();
        emp[0] = "Arun";
        emp[1] = "Varun";
        emp[2] = "John";
        for (i=0;i<emp.length;i++)
                {
                        document.write(emp[i] + "<br>");
                }
</script>
```

**3) JavaScript array constructor (new keyword)**

Here, you need to create instance of array by passing arguments in constructor so that we don't have to provide value explicitly.

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

```
<script>
        var emp=new Array("Jai","Vijay","Smith");
        for (i=0;i<emp.length;i++)
                {
                        document.write(emp[i] + "<br>");
                }
</script>
```

## MultiDimensional Array

**A JavaScript multidimensional array is an array of arrays**

**(or)**

**An array whose elements consist of arrays**.

```
<script language="javascript">
var A = [
            ['apple', 'orange', 'pear'],
            ['carrots', 'beans', 'peas'],
             ['cookies', 'cake', 'muffins', 'pie']
      ];
alert( A[2][1] );              // cake
</script> >
```

**NOTE:**

- The first square bracket references the desired element in the outer array.
- The second square bracket references the desired element in the inner array.
- ar[2][1] references the second element in the third sub-array.
- JavaScript array indexes start at zero.

# Functions in javascript

A JavaScript function is a block of code designed to perform a particular task.

A JavaScript function is executed when we calls it.

## JavaScript Function Syntax

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 of parameters (that might be empty), and a statement block surrounded by curly braces.

## Syntax

```
<script type="text/javascript">

    function functionname(parameter-list)
    {
      statements
    }

</script>
```

## Return Statement

A JavaScript function can have an optional **return** statement. This is required if you want to return a value from a function. This statement should be the last statement in a function.

**passing arguments to function (or) function using return statement**

```
<script language="javascript">

        function add(x,y)
        {
         return(x+y);
        }

        var a= add(2,3);
        var b= add(4,5);
        var c= add(5,10);

        alert(a);              //defined function can be called more than once
        alert(b);
        alert(c);


</script>
```

**Scope of the variable**

**Scope** is the set of variables you have access to.

Variables declared within a function is called **local scope**.They can only be accessed within the function

Variables declared outside a function is called **global scope**. All functions on a web page can access it.

**Global scope:**

```
<script language="javascript">

        var x = 5;

        function myTest()
                {
                        document.write("variable inside the function is"+x);
                }

         myTest();

          document.write("<br> variable outside the function is"+x);

</script>
```

**Local Scope:**

```
<script language="javascript">

        function myTest()

                {
                        var x=5;
                        document.write("variable inside the function is"+x);
                }

          myTest();

         document.write("variable outside the function is"+x);

</script>
```

**Function Literal:**

A Function Literal is an expression that defines an unnamed function**.**

**syntax:**

```
<script type="text/javascript">

    var variablename = function (Argument List)
        {
            function Body
        };

</script>
```

Note: **var abc = function() { ... }** is known as a **function expression**

**Example:**

```
<script language="javascript">

    var sqr=function(x)
        {
            return(x*x);
        }

    var result=sqr(3);
    alert(result);

</script>
```

**Function Constructor:**

The function statement is not the only way to define a new function; you can define your function dynamically using **Function()** constructor along with the **new** operator.

**Syntax:**

```
<script language="javascript">

        var variablename = new Function(Arg1, Arg2..., "Function Body");

</script>
```

**NOTE:**

The Function() constructor expects any number of string arguments.
The last argument is the body of the function

**Example:**

```
<script language="javascript">
        var square=new Function("x","return x*x");
        var result=square(3);
        alert(result);
 </script>
```

**Function Pointer:**

Function Pointer is just like any other variable out there, but instead of pointing to an integer or string type, it points to the actual function

```
<script language="javascript">
        function square(x)
                {
                        return (x*x);
                }
                f=square;
                alert(square(3));
                alert(f(3));
</script>
```

**Nested functions(inner functions):**

Declaring a function within another function is called **nested function(inner function).**

**Properties of nested functions**

1.inner function can only be accessed from statements of outer function.

2. inner function can access the variables that have been declared in the outer function.

**Example:**

```
<script language="javascript">

        function addSquares(a, b)
                {

                        return square(a) + square(b);

                                function square(x)
                                        {
                                                return x * x;
                                        }

                }
        var a = addSquares(2, 3);               // returns 13
         var b = addSquares(3, 4);              // returns 25
         var c = addSquares(4, 5);               // returns 41

        alert(a);
        alert(b);
        alert(c);
</script>
```