**JAVASCRIPT**

ECMA is a standard to writing code.

**Prompt ():** This function is used to take input from user on run time.

* Prompt function is defined in window.
* Window object is primary object of javascript.

**Objectname.member**

**Object:** object is the collection of property(variable) and methods(function).

Variable, function: member of object

// var obj={key:value;key:value}

Members of window

Prompt()

Alert()

Confirm()

Location()

Open()

Print()

Play()

Pause()

setInterval()

setTimeout()

clearInterval()

etc.

--------------------------------------------

**Syntax:**

Window.prompt(“FIleName”,”Placeholder”);

Var name=window.prompt(“Name“,”Enter your name”)

------------------------------------------------------------------------------------

getElementById()

getElementByTagName

getElementByClassName

querySelector()

querySelectorAll()

**getElementById():**

* getElementById Function is used to select html element on the basis of HTML Id.
* getElementById function is defined in DOM.
* Return type of getElementById function is current object.

**Syntax:** Document.getElementById(“IdOfHTMLElement”)

**Example:**

<h1 id=”h”>TECHPILE</h1>

Document.getElementById(“h”)

**Step to apply CSS with JAVASCRIPT:**

Object.style.propertyName=”value”

document.getElementById(“h”).style.color=”red”

var x=document.getElementById(“h”)

x.style.color=”red”

x.style.backbround=”aqua”

x.style.borderRadius=”2px”

**Get:**

Var x=HTMLOBJECT.value

Var x=document.getElementById(“idofFormControl”).value

**Example:**

**<input type=”Text” id=”txt”/>**

**<input type=”button” value=”GET VALUE”/>**

**SET:**

Objectname.value=”RAM”

**setInterval:**  setInterval function is used to execute a group of statement for every given time interval.

setInterval function is defined in window.

Example:

**Syntax 1:**

Window.setInterval(functionname,time is millisecond)setInterval(functionname,time in millisecong)

**setInterval:** Window.setInterval(functionname,time is millisecond)

**setTimeout():** this function is used to once a group of statement after given time interval.

**Syntax 1:**

Window.setInterval(functionname,time is millisecond)setTimeout()(functionname,time in millisecong)

**setTimeout:** Window.setTimeout()(functionname,time is millisecond)

**Array of JAVASCRIPT:** array is the collection of heterogeneous data types element. Indexing of array is starts from 0 and last index is n-1 where n is length of array.

Syntax:

Var arrayName=[item1,item2,……..itemN]

Example:

Var myInfo=[”RAM”,30,20000.5,true]

How to access element of array:

myInfo[0]

myInfo[1]

myInfo[2]

myInfo[3]

myInfo[5]

**Syntax 2:**

Var arrname=new Array(item1,item2,……,itemN)

Var courseList=new Array(“HTML”,”CSS”,”JS”,”JQUERY”)

**Example:**

<script>

var courseList=new Array("HTML","CSS","JS","JQUERY","BOOTSTRAP","REACT JS","EJS")

for(var i=0;i<courseList.length;i++)

{

alert(courseList[i])

}

</script>

For …in loop

For …of loop

Var arrName=new Array(item1,item2,item3…….,itemN)

If we want to execute a group of statement for every elements present in given collection then we have to use **for.. in** or **for ..of** loop.

**Syntax:**

For(var variable of collection)

{

Statement

statement

statement

statement

}

**Date object:-**

Data is object providing some methods related date and time.

**Syntax:**

Var d=new Date()

d.getHours() 🡺0-24

d.getMinutes() 🡺0-60

d.getSeconds()🡺0-60

d.getMilliseconds()🡺0-999

d.getDate() 🡺 5

d.getMonth() 🡺 0-11

d.getFullYear() 🡺 2022

d.getDay() 🡺 0-6

d.toLocalTimeString() 🡺11:40 AM/PM

d.toLocalDateString() 🡺 MM/DD/YYYY 🡺 05/12/2022

etc……

**HTMLCollection:-** HTMLCollection is just like array but element of HTMLCollection can be access by index only.

Item

Length

**NamedItem: -** namedItem(“abc”) => id=abc

**NodeList: -** NodeList is just like HTMLCollection but it has own function like entries, values, forEach, key, nameItem.

getElementByClassName: -getElementByClassName is used to return HTMLCollection with specified className.

<p class=”abc”>PARA1</p>

<p class=”abc”>PARA2</p>

<p class=”abc”>PARA3</p>

<p class=”abc”>PARA4</p>

**getElementByTagName: -**

getElementByTagName function is used to return HTMLCollection with specified tagname.

<p class=”abc”>PARA1</p>

<p class=”abc” id=”a”>PARA2</p>

<p class=”abc”>PARA3</p>

<p id=”box” class=”abc”>PARA4</p>

Var x=document.getElementsByTagName(“p”)

X=object HTMLCollection

Var x=document.getElementByTagName(“p”).nameedItem(“box”).innerText

alert(x) //para4

document.title

document.images

document.head

document.body

**Location: -** location is used to redirect location.

🡺Location is defined in window object

Window.location=<https://techpile.in/st>

**open(): -** open function is used to open URL on new tab or new window.

window.open(“url”,\_”blank”, “height=value,width=value”)

**Example:** window.open([https://www.techpile.in,”\_blank”,”height=600,width=400](https://www.techpile.in,)”)

**Print() : -** print function is used to print hole page to print.

**Syntax: -** window.print()

**querySelectorAll ():-** querySelectorAll return nodeList with specified selector Name.

<p>HTML</p>

<p id=”a”>CSS</p>

p class=”a”>JS</p>

<p class=”a”>JQUERY</p>

document.querySelectorAll(“p”)

document.querySelectorAll(“#b”)

document.querySelectorAll(“.a”)

var x=document.querySelectorAll(“p”) //nodeList

**getElementByName: -** getElementByName is used to return Nodelist with specified name.

<p name="a">PARA1</p>

<p name="b">PARA2</p>

<p name="a">PARA3</p>

<p name="a">PARA4</p>

<p name="b">PARA5</p>

document.getElementByName("a") //p1,p3,p4

**setAttribute: -** setAttribute function is used to set attribute of selected HTML Element.

**Syntax: -** var element=document.querySelector(“selector”)

element.setAttribute(“attributename”,”value”)

**getAttribute(): -** It is used to get Attribute value of selected HTML element.

<img src=”images/img-6.png”/>

document.querySelector(“img”).getAttribute(“src”)

**has Attribute :-** it returns true if specified attribute exist on selected HTML element otherwise it returns false.

**Example :-**

<h2 id=”box”>HTML</h2>

<script>

document.querySelector(“h2”).hasAttribute(“class”) // false

document.querySelector(“h2”).hasAttribute(“id”) //true

</script>

**hasAttributes: -** it returns true if selected HTML element have no attribute otherwise it returns true.

var x=document.querySelector().hasAttributes() //false

**Math :-** math is a object providing some property(variable) and method(function) related to mathematical operation.

CONSTANT VALUE: -

PI

**LN2: -** natural logarithm of 2 base e

**LN10: -** natural logarithm of 10 base e

Floor():- floor function is used to return lowest integer value of given decimal value.

x = 12.45675

math.floor(x) //12

**Random(): -**random function is used to return random number between 0(encluding) to 1(excluding)

Example :- alert(Math.random())

Ceil()

Round()

Min()

Max()

Sqrt()

Cbrt()

Paw()

**Log(): -** it is used to find logarithem value given value base on e.

Floor()

Etc…

**Trim () :-** trim function is used to remove both side white space.

Var str=” techpile ”

Var str1=”techpile”

**addEventListener() :-** it is used to add event to the selected HTML element.

It does not override old event.

Var element=document.querySelector(“button”)

Element.addEventListener(Event, functionName, useCapture)

**Event :-** click, mouseover, dblclick

**Function :-**

**useCapture : -** bubbling 🡺 false by default

capturing 🡺 true

element.addEventListener(“click”, demo)

function demo()

{

alert(“okk”)

}

**contextmenu :-** contextmenu event executed when user right click on window

**preventDefault() :-** preventDefault method is used to prevent(stop) default action.

event.preventDefault()

window.addEventListener(“contextmenu”,function(){

alert(“Sorry right click disabled here...”)

event.preventDefault()

})

**copy event:** - copy event executed when user select copy option with keyword/mouse.

Window.addEventListener(“copy”,function(){

alert(“okk“)

})

**cut event:** - copy event executed when user select cut option with keyword/mouse.

Window.addEventListener(“cut”,function(){

alert(“okk“)

})

**paste event:** - copy event executed when user select paste option with keyword/mouse.

Window.addEventListener(“paste”,function(){

alert(“okk“)

})

**String in JS :-**

* In JavaScript string is collection character enclosed with single or double or backtique.
* “RAM”
* ‘RAM’
* `RAM`

**Function of string in javaScript: -**

trimStart

trimEnd

padEnd

**Slice :-** Slice function is used to return a part o string based on start and end index.

🡺here negative index is acceptable.

**Syntax: -**

Var str=”Techpile Technology”

Str.slice(startIndex,endIndex)

**Example: -**

str.slice(2,5) // CHP length= end - start 3

str.slice(0,10) // CHP length= end - start 10

str.slice(-10,-5) // CHP length= end - start 5

replaceAll

**substring:-** substring function is used to return a part o string based on start and end index.

🡺here negative index is not acceptable.

**Syntax:-**

str.substring(startIndex,endIndex)

**substr:-** substr function is used to return a part of string of given string based on start and length.

**Syntax: -**

String.substr(startIndex,length)

startIndex can be negative but length can’t possible

var str =”MOHAN”

alert (str.substr(1,3))

**charAt: -** charAt Function is used to return character at specified index.

🡺Index can’t be negative or greater than or equal to length of string.

var str=”TECHPILE TECHNOLOGY”

alert(str[0]) // T

alert(str[1]) // 1

alert(str.charAt(0)) // T

alert(str.charAt(str.length-1)) //Y

alert(str.charAt(str.length-2)) //G

**lastIndexOf : -** lastIndexOf function is used to return index of last occurrence character in a given string.

* It returns -1 if character not present in given string

**Syntax: -**

* var str=”TECHPILE TECHNOLOGY”
* alert(str.lastIndexOf)

**indexOf : -** indexOf function is used to return index of first occurrence of character in a string.

It returns -1 if character is not present in given string.

**Syntax: -**

var str=”TECHPILE Techpile”

document.write(str.lastIndexOf(“T“)) //9

document.write(str.lastIndexOf(“o“)) //16

**charCodeAt :-** charCodeAt function is used to return at Unicode value(ASCII Value) at specified index

Var str=”ABCabc0123”

alert(str.charAt(0)) //A

alert(str.charCodeAt(3)) //97

alert(str.charCodeAt(5)) //99

alert(str.charCodeAt(0)) //65

alert(str.charCodeAt(str.length)) //NaN

**ASCII value or Character / Unicode value :-** ASCII stands for American Standard Code for Information Interchange

A-Z 🡺 65 – 90

a-z 🡺 97 – 122

space 🡺 32

0 – 9 🡺 48 – 57

var str=”TechpileT”

alert (str.indexOf(“T”)) //0

alert (str.seach(“T”)) //0

alert (str.search(/t/)) //-1 case sensitive

alert (str.search(/t/i)) //0 case In-sensitive

**search : -** search function is used to return index of first occurrence of regular expression in string.

By default it searches in case-sensitive mode.

If we want search case-insensitive mode then we have to use i flag with regular expression.

It returns -1 if if regular expression is not present string.

**toUpperCase :-** to convert string into uppercase letter.

**Syntax: -** var str=”Techpile”

Str=str.toUpperCase()

Document.write(str) // TECHPILE

**toLowerCase :-** to convert string into lowercase.

**Syntax: -** var str=” TECHPILE”

Str=str.toLowerCase()

Document.write(str) // techpile

**Trim() :-** trim function is used to remove both side white space.

**Syntax :-** Var str=” Techpile ”

Document.write(str.length) // 14

Str=str.trim()

Document.write(str.length) //8

Document.write(str) // “techpile”

**trimStart(): -**to remeve left side of white space of a string.

**Syntax :-**

Var str=” Techpile ”

Document.write(str.length) // 14

Str=str.trimStart()

Document.write(str.length) /11

Document.write(str) // “techpile ”

**trimEnd(): -**to remeve right side of white space of a string.

**Syntax :-**

Var str=” Techpile ”

Document.write(str.length) // 14

Str=str.trimEnd()

Document.write(str.length) /11

Document.write(str) // “ techpile”

**padStart() :-** padStart function is used to add content at beginning position of given string.

**Syntax: -** str.padStart(“length of string after add content”,content)

var str="RAM"

str=str.padStart(10,"X")

document.write(str) // XXXXXXXRAM

**padEnd() :-** padEnd function is used to add content at End position of given string.

**Syntax: -** str.padEnd(“length of string after add content”,content)

var str="RAM"

str=str.padEnd(10,"X")

document.write(str) // RAMXXXXXXX

**replace():-** replace function is used o replace a part of string with new string.

=> By default it replace first matches element with new string.

=> if we want to replace all matches of string then we have to use global flag with regular expression.

=>It does not update original string.

**Syntax :-** var str=”Techpile Technology”

Str.replace(oldstring,newString)

Str.replace(“tech”,”make”)

Document.write(str) //Techpile Technology //because string not match.

Str.replace(“Tech”,”XYZ”)

Document.write(str) //XYZpile Technology

Str.replace(/Tech/g,”abc”) //abcpile abcnology

Str.replace(/tech/gi,”abc”) //abcpile abcnology

Str.replace(/tech/i,”abc”) //abcpile Technology

**replaceAll() :-** replaceAll function is used to replace all matches of given string with new string.

* If we want to use regular expression in replaceAll then compulsory we have to global flag(g).

**Syntax: -**

var str=”Techpile techology”

var str=str.replaceAll(“Tech”,”abc”) //abcpile technology

**concat ():-** concat function is ued to merge of multiple string in one string.

Syntax: - var str1=”Techpile”

var str2=”Technology”

var str3=”Pvt Ltd.”

var newString=str1.concat(str2,str3)

**match: -** match function is used to match a part of string in main string and retuen first element.

* If searches element not present in main string then it return null.

**Example: -**

var str="Techpile Technology Pvt. Ltd."

//var x=str.match(/tech/) //null

//var x=str.match(/Tech/i) //Tech

var x=str.match(/Tech/gi) //Tech

document.write(x)

**includes:-** includes function is used to check whether given string present in main string or not.

* It returns true if given string is present in main string otherwise it returns false.

**Syntax:-** var str=”techpile Technology pvt. Ltd.”

**str.**includes(“Tech”) // true

**str.**includes(“O”) // false

**str.**includes(“Ltd”) //true

**Split:-** split function is used to split a string into array based on separator.

**Syntax :-** str.split(separator)

Var str=”Techpile Technology pvt. ltd.”

Var arr=str.split(“”) //

Document.write(arr) // [“Techpile”, “Technology”, “pvt.”, “ltd.”]

**startsWith():-** startsWith function is used to check whether main string starts with given string or not.

* It return true if main string starts with given string otherwise it returns false.

Var str=”Techpile Technology Pvt. Ltd.”

Var x=str.startWith(“Tech”) //true

Var x=str.startWith(“XYZ”) //false

**endsWith():-** endsWith function is used to check whether main string ends with given string or not.

* It return true if main string ends with given string otherwise it returns false.

Var str=”Techpile Technology Pvt. Ltd.”

Var x=str.endsWith(“Tech”) //true

Var x=str.endsWith(“XYZ”) //false

**Array Functions :-**

**Push() :- push fu**nction is used to add element at last position of array and return length of update

Example:- var arr=[“RAM”,”MOHAN”,”SOHAN”]

var x=arr.push(“TECHPILE”)

**filter(): -** filter function is used to filter items of array and return new Array based on some condition.

* Filter function is not applicable for empty array.

Syntax: - arr.filter(function)

Function(currentItem,index,arr)

Here index and arr is optional parameter.

**map():-** if we want to add some functionality to every elements of an array and generate new array then we have to use map function.

* Map function is not applicable for empty array.
* It does not update original array.

Syntax: - arr.map(function)

Function(currentValue,index,arr)

**Pop(): -** pop function is used to remove last element of an array and return removed element.

Arr=[“Ram”,”Mohan”,”Raj”]

Alert(arr.pop())

**Shift():-** shift function is used to remove first element of an array and return removed element.

Var arr=[1,2,3,4,5,6]

X=arr.shift() //1

Alert(x)

**Unshift():-** shift function is used to add an element of an array at beggning position.

=>it returns length of updated array.

Var arr=[1,2,3,4,5,6]

Var x=arr.unshift(“12”)

Alert(x) //7

Alert(arr) //12, 1,2,3,4,5,6

**Delete:-** delete property is used to delete value of an array without index.

**Example: -**

var arr=[“RAM”,”MOHAN”,”SOHAN”]

delete arr[1]

**join() :-** join function is used to join array elements on the basis of separator and return new string.

🡺It does not modified(update) original array

**Example: -**

var arr=["TECHPILE","TECHNOLOGY","PVT","LTD"]

var str=arr.join(" ")

alert(str) //TECHPILE TECHNOLOGY PVT LTD

**every:-** every function is used to check whether given condition satisfy all element of array or not.

🡺it returns true if all elements of array satisfy given condition otherwise it returns false.

Syntax:- arr.every((data)=>{

Return condition

})

**Some :-** some function is used to check whether any elements of satisfied given condition or not.

* It returns true if any elements of array satisfying given condition otherwise it returns false.

**Example:-** var arr=[“RAM”,”MOHAN”,”ROHAN”,”SHYAM”]

arr.some(function(data,index,arr){

return condition

})

var a=arr.some(()=>{

return data[0]==”R”

})

**reduce :- reduce** function is used to reduce elements of array in single valueby applying some functionality.

var arr=[1,2,3,4,5,6,7,8,9,] //trailing comma (extra comma)

**Syntax:- arr**.reduce((t,data,index,arr)=>{

})

T 🡺 initial value / previously return value

**Splice():-** splice function is used to add/remove element to/from array and removed element

var arr=[“RAM”,”MOHAN”,”SOHAN”]

arr.splice(start,length,item1,item2,item3,…..,itemN)

length🡺how many elements we wants to remove(length of removing elements)

arr.splice(1,1,”SHYAMU”)

**keys:-** keys function is used to create array iterator of array key(index.)

var arr=["RAM","MOHAN","SOHAN"]

var x=arr.keys()

for(var i of x){

alert("keys : "+i)

**entries():-**entries function is used to create array iterator of array keys(index) and values

var arr=["RAM","MOHAN","SOHAN"]

var x=arr.entries()

for(var i of x){

alert("key and values : "+i)

**values():-**values function is used to create array iterator of array values

var arr=["RAM","MOHAN","SOHAN"]

var x=arr.values()

for(var i of x){

alert("values : "+i)

**toString:-** toString function function is used to convert elements of array into string separated by comma.

var arr=[“RAM”,”JON”,”MOHIT”]

var x=arr.toString()

document.write(x)

**concat:-** concat function is used to convert(merge) multiple array into one array.

var arr1=[1,2,3,4,5]

var arr2=[10,20,30,40,50]

var arr3=[“RAM”,”MOHAN”,”SOHAN”]

var nArr=arr1.concat(arr2,arr3)

**flat(): -** flat function is used to convert multi-dimensional array into single dimensional array.

var arr=[“RAM”,45,[“Lucknow”,226022,[1,2]]]

nArr=arr.flat()

document.write(nArr) //=[“RAM”,45,”Lucknow”,226022,1,2]

**find():-** find function is used to return first element of array that satisfying condition.

**Syntax:-**

arr.find((data,index,arr)=>{

return condition

})

**Example:**

var arr=["ram","mohan","john"]

        var a=arr.find((data)=>{

        return data[1]=="a"

        })

        document.write(a)

**findIndex():-** findIndex function I used to return index of first element that satisfying condition.

**Syntax:-**

arr.findIndex((data,index,arr)=>{

return condition

})

**Example:-**

var arr=["ram","mohan","john"]

        var a=arr.find((data)=>{

        return data[1]=="a"

        })

        document.write(a)

**Module:-** module is the collection of variable,function and class save to a single file.(.js extension)

Every .js extension file is treated like module.

**File1.js:-**

export var a=10

export function demo(){

alert(“ok”)

}

export class Test{

demo1(){

}

}

**File2.js:-**

import {a,demo,Test} from ‘./file1.js’;

alert(a)

demo()

var obj=new Test()

obj.demo1()

member aliasing || rename

import {firstname as f, lastname as l,fun1 as f1,fun2 as f2,test as t} from “./m1.js”

**Destructuring :-** Destructuring is a process to hold element of array/object into the corresponding variable.

**Object :-** Object is the collection of property(variable) and methods(function).

* If we want represent a group of value in form of key/value then we have to use object data type.
* In JavaScript object is denoted{}

**Syntax:-**

var objectName={key1:value1,key2:value2,……..,keyN:valueN}

**Example:-** var studentInfo() = {name : ”SHIVAM”, branch : ”CS”, salary :100000, age:45 }

=>here value can be duplicate but key can’t be duplicate.

=>if we are trying to duplicates key then first value of key will be replace with new value of key.

Var obj={1:”RAM”,2:”Mohan”,1:”ROHAN”}

Alert(obj[1]) //ROHAN

**How to access elements of Object.**

**Syntax :** objectName[key] //value of key

studentInfo[name]

studentInfo[branch]

studentInfo[age]

studentInfo[salary]

var obj1={name:”RAM”,age:45,salary:500000}

**How to change value of key**

Obj1.name ="ROHIT"

Console.log(obj1)

**How to add key in a object**

**Syntax:**

objectName.key=value

obj1.Salary=500000

console.log(obj1) // {name:”ROHIT”,age:45}

obj1.salary=100000

console.log(obj1)// {name:”ROHIT”,age:45,salary:100000}

var empInfo={

name:”SHIVAM”

salary:50000

address:{city:”DELHI”,pincode:”226022”}

}

Var obj={name:”RAM”,fun:function(){

Console.log(“I am from fun key”)

}}

**Console.log(`{obj.name}{obj.fun}`)**

var obj1={

City : "Lucknow"

Pincode : 226022

}

console.log("Name" is obj1) //false

console.log("City" is obj1) //true

console.log("Pincode" is obj1) //true

**hasOwnProperty :-**

hasOwnProperty is used to check weather given key present in object or not.

It returns true if given key present in given object otherwise it return false.

   var obj1={

    City : "Lucknow",

    Pincode : 226022}

console.log(obj1.hasOwnProperty("propertyName"))  //true || false

console.log(obj1.hasOwnProperty("City"))  //true

console.log(obj1.hasOwnProperty("Name"))  //false

 var obj={

            Name:"Shivam",

            Age:20,

            Salary:100000,

}

var arr=[1,2,3,4,5,6]

**keys() :-**

keys function is used to create an array iterate with key of an object.

var obj1={

1:1,

2:2,

3:45,

}

[1,2,3]

**Syntax:-**

var arr=Object.keys(objectName)

Example :-

var x=object.keys(obj1) //[1,2,3]

values :-

var obj1={

1:1,

2:2,

3:45,

}

Object.values(obj1) // [1,2,45]

for (var i of arr){

console.log(i)

}

var obj = new Object({Name : "SHIVAM", Age : 19})

console.log(obj.Name)

console.log(obj.Age)

Array of Object :-

It is collection of object.

var arr =[{},{},{},{}{},………{}]

{} 🡺 {keyN:valueN}

var studentInfo[

{

Name : "RAM",

Age : 23,

Branch : "CS",

City : "Lucknow"

},

{

Name : "Shivam",

Age : 34,

Branch : "IT",

City : "Kanpur"

},

{

Name : "Rohan",

Age : 20,

Branch : "32",

City : "Varanasi"

},

{

Name : "aaaa",

Age : 23,

Branch : "EE",

City : "Lucknow"

},

]

**How to access element of array of object.**

studentInfo[0].Name // RAM

studentInfo[0][Name] // RAM

studentInfo[0].Age // 34

studentInfo[0].Branch // CS

studentInfo[0].City // Lucknow

**hoising:-** hoisting is a process to move all declaration to top in it’s scope.

* Hoisting process is not applicable for javascript class.
* Hoisting is default process done by javascript engine.

Class Test {

demo(){

console.log(“okk”)

}

}

var obj = new Test()

obj.demo()

**OOP’s (object oriented programming system)**

Oops is programming paradigm or pattern to create program by using class and object.

Advance of oops is code reusability.

Class

Object

Method

Inheritance

encapsulation

etc

**class:-** class is a collection of property(variable) and method(function). Class keyword is used to create class in javascript.

Syntax :- class className{

Property

method

}

**2nd method**

Var className = class {

Property

Method

}

**Object :-** object is class type variable.

To access members of class compulsory we have create object of that class.

**Reference variable :-** reference variable is used to refer (store) object of class.

In javascript new keyword is used to create object of an class.

**Syntax:-**

var reference variable = new className()

**constructor :-** constructor is a special type of method.

Constructor method called automatically at time of object creation of class.

Constructor main purpose of constructor method is used to create property of a class.

Constructor keyword is used to create constructor in javaScript.

**Syntax:-**

Class className{

constructor()

}

**Constructor with parameter :-**

class myclass{

constructor(name,age){

this.name=name

this.age = age

this.salary = 20000

}

}

var obj = new myclass("RAM",20)

**Example:-**

class myClass{

            constructor(){  // automatic invoke

                console.log("hello i am from constructor method")

            }

            demo(){

                console.log("hello i am from demo function")

            }

        }

        var obj = new myClass()

        obj.demo()

**Method vs Constructor :**

|  |  |
| --- | --- |
| Name of methods can be any name  Method will be execute if we call that method  Per object, method can be called any number of times  Inside method we can write business logic | Constructor name should be constructor  Constructor will be executed automatically at the time of object creation  Per object constructor will be execute only once  Inside constructor we have to declare and initialize in instance variable(property) |

**Instance variable vs static variable**

|  |  |
| --- | --- |
| **Instance Variable** | **Static variable** |
| 1. These are object level variable. 2. For every object separated copy will be created. 3. By using one object reference, if we are trying to perform any changes to the instance variable then those changes won’t be reflected to the remaining object. |  |