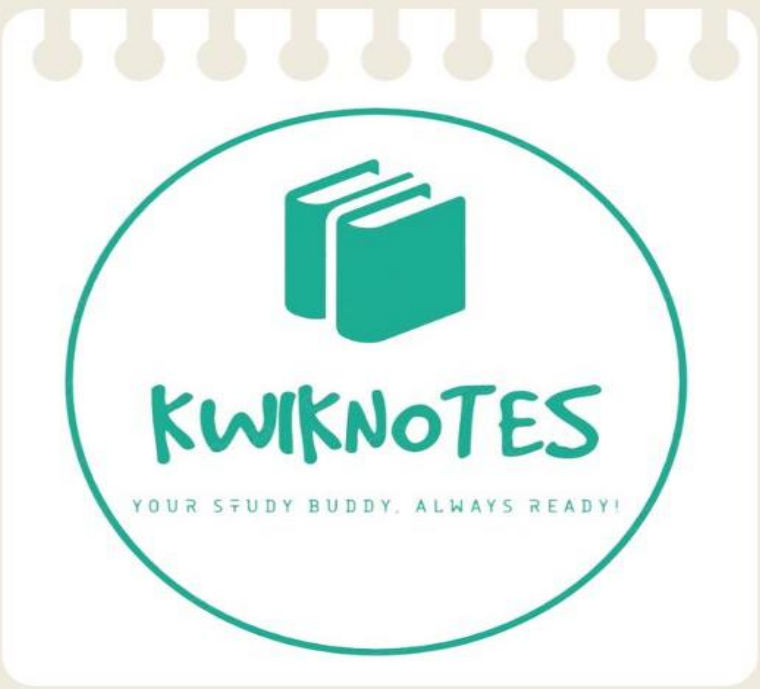


HANDWRITTEN NOTES

Download FREE Notes for Computer Science and related resources only at

Kwiknotes.in

Don't forget to check out our social media handles, do share with your friends.

JAVASCRIPT

Intro
dynamic comp programming lang
Client Side Script
Object Oriented
First Known as LiveScript byt NetScape
Changed to JavaScript
Open & cross platform.

When used with HTML means webpage need not to be static, it can interact, control, etc.

JS is executed when user submit form

JS dont have multithreading

Tools - Notepad, Microsoft frontpage, Macromedia Dreamweaver Mx, MacroMedia Homesite 5.

```
< Script * language "javascript" type ="text/javascript"
    Code
</script>
```

Semicolons are optional.

document.write("text"); ⇒ to print text

Comment <!-- -->(multi) // (Single
/* */

Case Sensitive

<noscript> </noscript> can be used to display text when browser don't support JS.
↳ can be enclosed in head | body.

Datatypes

- Number - String - Boolean - Null - Undefined
- Object

trivial, define single value

Variables
declared with var before
var a = 10;
or var a;
a = 10;

We can't Redeclare variables in JS

Global variable ⇒ Outside function
Local variable ⇒ Inside function

Bitwise operator perform operation with boolean values by conversion.
Index start at 0.

AND && OR || Not !
Bitwise AND & Bitwise OR |
Bitwise XOR ^ Bitwise Not ~
<< leftshit >> Right Shift

Ternary ? :

(100 > 200) ? 100 : 200 ⇒ 100

typeof to know type of var

var b = "abc"
typeof b ⇒ "String" ? "b is string" : "Bismo"
⇒ b is string

Datatype 1) Primitive – undefined, null, no, String, boolean, Symbol

2) Refrence data type - Array & objects

Function

function funcname (parameter) { - - }

```
var a, b;
function avg (a, b) {
        c = (a+b) / 2;
        return c;
    }
c1 = avg (4, 6);
console.log(c1);
```

Ladder

if - else if - else

Loops

```
for (start, end, step) {
    }
for (var i = 0; i < 5; i++) {
        console.log(i);
    }
```

Another way

IMP

```
var arr = [5, 6, 7, 8, 9]
arr.forEach (function (element) {
        console.log(element);
    })
```

forEach(value, index, array) => Syntax

```
let num = [3, 5, 1, 2, 4]
num.forEach((element) => {
        console.log(element**2)
})
```

=> 9 25 1 4 16

Or

```
for (let i of num) {
        console.log(i)
}
```

=> 3 5 1 2 4

Or

```
for (let i in num) {
        console.log(i)
}
```

=> In focus only it will display 0 1 2 3 4

To print values using for-in use console.log(num[i])

map() is similar as forEach but creates a new array. We can modify element using map

```
let arr = [45, 20, 21]
let a = arr.map((value) => {
        console.log(value)
})
```

=> 45
20
4

Map & filter Return array
filter Return value.

```
let arr = [45, 23, 21]
let a = arr.map ((value, index, array) => {
        return value + 1          ⇐ ①
})
console.log(a)    ②    ⇒ store result in a
```

① ② 46 24 22

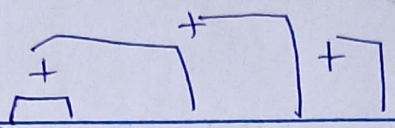
Filter Method ↳ creates a new array to apply condition

```
let arr = [45, 23, 21, 0, 3, 5]
let a = arr.filter((a) => {
        return a < 10
})
console.log(a)
```

⇒ [0, 3, 5]

It donot modify previous array.

Reduce it reduce array to a value using arithmetic Operator.



```
let arr = [1,2,3,4,5]
let newarr = arr.reduce((h1,h2) => {
        return h1+h2
})
console.log(newarr)          => 15
```

1,2 as h1 & h2 then 3 and 3 then 6 + 4

+3
+6
till last element it continues

Prompt sometime don't work in compiler so use this to add a no. in compiler

Variable can start with underscore and $

Constant declaration

```
const a = 10;
```

Function

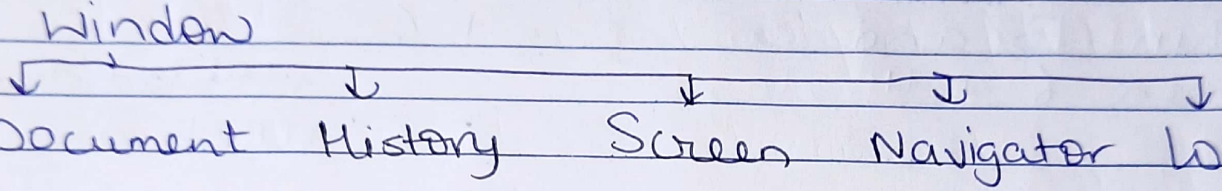
```
var add = new function ("n1", "n2", "return n1+n2");
document.write(add(2,8));   => 10
```

SetTimeout to delay something

Browser Object Model (BOM)

to interact with browser

Default object - window ex- window.alert();
is same as alert();



1) Window object - it is supported by all browsers, it represent windows of browsers.

All JS object funct var become part of window object

Window Size - window.innerHeight (in pixel) to have inner height of browser window
window.innerWidth - for width
window object - alert, confirm, prompt, open, close, openlink
SetTimeout

2) History object - represent an array of URL's visited by user. By using this object, you can load previous, forward or any particular page.

It can be written as window.history or history

history.back() load previous URL in history list

Window.history.back() → will take you to previous page

Window.history.forward() → take you to next page

history.length() → to know no. of URL visited

document.write("History:" + history.length);

history.back → for one previous page

history.go()

history.go(-2) ⇒ perviously 2 page
history.go(2) ⇒ forward 2 page

3) Navigator
navigator.appName
navigator.appCodeName
navigator.CookieEnabled
navigator.appVersion
navigator.plugins

Seal Object

3) Screen Object – to get browser screen info

Width, height, pixelDepth, colorDepth, availHeight, availWidth

Screen.width

DOM Document Object Model

Element visible on document ; p, span, div

document
anchor
form → radio, text, checkbox, select, reset, button
link

☆ Display input in alertbox

```
<form name = "form1">
    Name:
        <input type = "text" name = "name">
        <button value = "Click" onclick =
            "func()" >
</form>
```

```
<script type = "text/javaScript">
    function func () {
        var name = document.form1.
                   name.value ;
        alert (name);
    }
</script>
```

getElementById

```
<form name = "form1">
    <input type = "number" id = "phone">
    <input type = "button" value = "Click" onclick = "A()">
</form>
<script>
    function A() {
        var a = document.getElementById
                ("phone").value ;
        alert (a);
</script>
```

↳ it returns array.

getElementByClass

```
<html>
  <div class="abc">
      This is one
  </div>
  <div class="abc">
      This is two
  </div>
  <Script>
      var x = document.getElementsByClassName
          ("abc");
      alert (x[0].innerText);   ⇒ This is one
      alert (x[1].innerText);   ⇒ This is two
  </script>
```

Alert box

getElementsByName() ⇒ used in form

```
<form>
  Male <input type="radio" name="gender"
          value="M"> <br>
  Female <input type="radio" name="gender"
          value="F"> <br>
```

innerHTML to add HTML tags in js by using id.

```
var x = document.getElementById("abc");
x.innerHTML = "<a href='#'> </a>
or
var x = document.querySelector(".compare");
```

```
<input type="button" value="Show data"
onclick="show()">
</form>

<script>
    function show() {
        var x = document.getElementsBy
        Name("gender");
        alert("Total Gender" + x.length);
</script>
```

⇒ 2.

To display Male Female i.e elements.
use

```
alert(x[0].innerHTML);
```

getElementsByTagName()

we can give tag name such as p, img, div, input.

To print on same page

```
var x = document.getElementById("name").value;
document.getElementById("ia").innerHTML = x;
```

Jquery can also validate

return false — to prevent page reload.

Validation JS

Data checked on client side when client enters.

```
<form name = "myform" method = "post"
      onsubmit = "return validateform()">
Name: <input type = "text" name = "name">
Pass: <input type = "password" name = "pass">

<input type = "Submit" value = "Submit">

</form>

<script>
    function validateform() {
    var a = document.myform.name.value;
    var pass = document.myform.pass.value;
    if (name == null || name == "") {
        alert ("Please Enter Name");
        return false;          //to cancel
    }
    else if (pass.length < 6) {
        alert (" Enter 6 digits);
        return false;
    } }
```

</script>

✡ Script to Confirm password

```
<form name = "form1" onsubmit = "return val()">
    <input type = "Password" name = "pass">
    <input type = "Password" name = "confirm">

    <input type = "Submit" value = "Submit">
</form>

<script>
var pass1 = document.form1.pass.value;
var pass2 = document.form1.confirm.value;
if (pass == confirm) {
        return true;
} else {
        alert ("Pass don't match");
        return false;
}
```

Telephone: F45 Enter a valid number

To give validation in HTML itself

```
<form name="f1" onsubmit="return val()">
    Telephone number:
    <input type="number" name="tel">
    <span id="error"></span>
</form>

<script>
    function val()
    var telephone = document.f1.tel.value;
    if(isNaN(telephone) {
        document.getElementById("error")
        .innerText = "Enter valid
            telephone number";
        return false;
    }
    else {
        return true;
```

Return false to stop Execution

Return to it Reloads the page

email validation

```
<form name="form" onsubmit="return e()">
    email:
        <input type="text" name="email">
        <input type="submit" value="Submit">
</form>

<script>
    function e() {
        var email = document.form.
            email.value;
        var at = email.indexOf('@');
        var dot = email.indexOf('.');
        if (at<1 || dot<at+2 ||
            dot+2>=email.length){
            alert("Enter valid email");
            return false;
        }
    }
```

var.back() → to go back
var.forward → to go one page forward
var.go(2) → 2 page forward
var.go(-2) → 2 page back

EVENTS IN JS

To do something through mouse

Event performed	Event Handeler	Description
click	onclick	when mouse click an element
mouseover	onmouseover	when cursor come over element
mouseout	onmouseout	when cursor leaves the element
mousedown	onmousedown	when mouse is pressed over element
mouseup	onmouseup	when mouse button is released over element
mousemove	onmousemove	when mouse movement takes place

onfocus - when we click on input
onblur - when we click outside that input

```
<input type "text" onfocus="f(this)" onblur="b(this)">
<input type="text" onfocus="f(this)" onblur="b(this)">
f(element){
    element.style.background="yellow"; } → will make current parameter element
b(element){
    element.style.background="white"; }
```

Event acts same as hovers in HTML

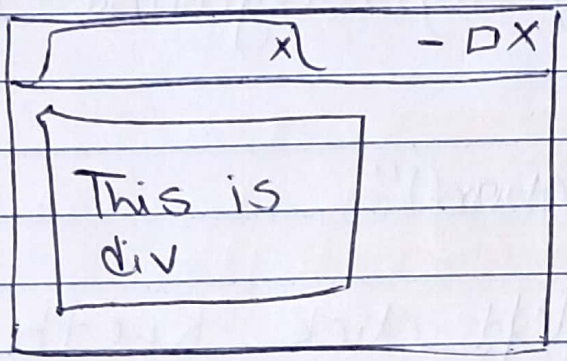
Mouse Event

1) **Onclick** - it is used with buttons

```
<input type="button" value="changebg"
 onclick="color()">
<script>
    function color() {
        document.body.style.background
            = 'black';
        document.body.style.color="white">
```

for bg

for text color

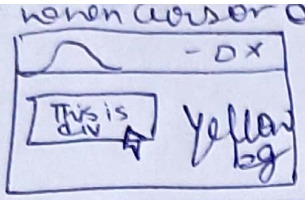


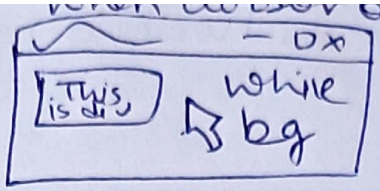
2) **onmouseover** = "func"

```
<div onmouseover="over()">
    This is div
</div>

<script>
    function over() {
        document.body.style.color="yellow";
```

⇒ Pr isme yeah hoga ki cursor upar jate hi bg background yellow hojaega or sirf Reload p hatega. To prevent this we use **onmouseout**





3) onmouseout

```
<div onmouseover = "over()"
     onmouseout = "out">
This is Div
</div>
<script>
    function over() {
        document.body.style.background
                = "yelow";
    }
    function out() {
        document.body.stly.background = "white";
    }
```

4) onmousedown = "function()"

⇒ jese hi mouse p left click Kretb chalega pr tbtk chalega jbtk mouse se click Krke cursor div me h jese hi div se cursor bahar aya change hojaenge

```
<div onmousedown = "down()">
    This is div
</div>
```

```
<script>
    function down() {
        document.body.style.background="pink";
    }
</script>
```

5) onmouseup = " fun() "

```
<script>
    function up() {
        document.body.style.background="yan">
}</script
```

=) jb click Krke div me h
Lekin mousedown K lie Click
Krke hold Kra h.

6) mousemove = Same is mouseover

Same jese hi cursor vha leke jaenge
it will do as directed.

Keyboard Events

1) onKeydown
2) onKeyup

It is for input field when me enter data.

1) onkeydown ⇒ to have change when me type, jbtk koi be key press hogi iski body chalegi

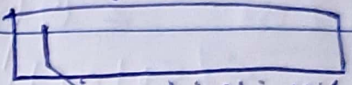
2) <input type="text" onKeyup="green()" onKeydown="yellow">

```
<script>
    function yellow(){
        document.body.style.background
                    ='yellow';
    }
    function green(){
        document.body.style.background=
                    'green';
    }
</script>
```

FORM EVENTS used with input tags or selection

1) onfocus => jab input tag k ander click krenge to function kaam krega

```
<input type="text" onfocus="yellow">
<script>
    function yellow(){
        document.body.style.background
            ='yellow';
    }
</script>
```



jese hi click kreng

2) onblur = jb input p click krke color change ho, to reverse that jb page p bahar khi b click kre except input to it will trigger

click, then it will execute function body

```
<input onblur="green">
function green(){
    document.body
    style.bg="green";
```

3) Onsubmit ⇒ used with form tag to execute a js function

4) onchange ⇒ used with selection tag.

```
<selection onchange="color()">
    <option> Male </option>
    <option> Female </option>
</selection

<script>
    function color() {
        document.body.style.
            background = "yelow";
    }
</script>
```

3) Ways to handle events: html, js, addEvent

i) onclick, etc html me hi dedo with function name

ii) add EventListener ("click", function)

iii) give id in html, in js through doc.getElementById("id").onclick = function() {
— — body }

Window / Document Events

- load
  ↳ when we load webpage
- unload
  ↳ practically impossible
- resize
  ↳ when maximis change window size

```
<body onload="all()" onresize="resize()">
</body>
<script>
    function all() {
        alert("Window is Reloaded");
    }
    function resize() {
        document.body.style.background
            ='green';
    }
```

addEventListener → used within js to add event using js
It does not override event handelers such as onclick

Syntax:

```
element.addEventListener(event, function,
    use(apture));
```

and function
event ^ and required parameter

use apture is optional, boolean value

* can perform bubbling & capturing

Code:

↳ In this onclick is defined in JS. ID is taken from html.

Onclick Events in JavaScript

dont work on html head title style meta, br, script

Double click Event — ondblclick

↳ Also can be implemented using 3 methods only

1) <body>
<h2 ondblclick="dbclick()"> Double Click here </h>

<script>
function dbclick() {
document.getElementById("p1").innerHTML = "This is db para";
</script>

2) <script>
document.getElementById("p1").ondb click = function() { — body }
in this we dont give func name

3) document.getElementById("p1").addEventListener("dbclick", fun);
function fun() { —— }

Onload Event

Can be used with ~~Button~~ body, or any other tag

1) document.onload
triggers before loading of image & external content
It is fired before window.onload.

2) window.onload
trigger when entire page loads including CSS, Script, image, etc.

Syntax
window.onload = fun();

1)
```
<Script>
    document.onload ("This is Doc.onload");
</Script>
```

2)
```
<Script>
    document.onload = alert("Doc.onload");
    window.onload = alert("Window.onload");
</Script>
```

Pehle doc wala alert hoga fir window wala

getElementsByTagName('body')[0] ⇒ coz it works as a array, we need its first indx.

Onresize Jese hi window maximize ~~main~~ hoke resize hogi, it will trigger

```
<body onresize = "fun()" >
    <p id = "p1" > </p>
    <p> This window is resized
    <span id = "s1" > </span>
    </p>

<script>
    var s1 = 0;
    function fun() {
        document.getElementById("p1").
        innerHTML = "Window width = " +
        window.outerWidth + " Window
        height = " + window.outerHeight;
        s1 = s1+1;
        document.getElementById("s1").
        innerHTML = s1;
    }
```

Set ~~interval~~ & clear ~~interval~~ for Animation.

```
setInterval(var, 1000 in ms);
function var () {
    var a=0; a = a+10;
    target.style.marginleft = a+'px';
}
```

↳ div id

EXCEPTION HANDELING

Types of Errors

Syntax Logical Runtime

2 types – throw statement , try catch

1 try {
    var a = [1,2,3];
    document.write(a[9]);
}
catch (e) {
    document.write("error =" + e.message);
}

2 With throw

<script>

<html>
<body>
Enter a no b/w 5 & 10 :
<input type = "text" id = "num";
<input type = "submit"
    onclick = "val()" >

```
<script>
    function val() {
      var message = document.getElement
        ById("para");
      message.innerHTML = "OK";
      try {
        var x = document.getElementById
          ("num").value;
        if (x=="") throw "is empty";
        if (x<5) throw "less than 5";
        if (x>10) throw "greter than 10";
        if (isNaN(x)) throw "not a no";
      }
      catch(e) {
          message.innerHTML =
            "error" + e;
      }
    }
</script>
```

We can also use finally{ }

COOKIES

document.cookie = "cookiename";

1) Setting a cookie

```
<body>
  <input type = "button" value = "setcookie"
         onclick = "setcookie()" >
  <input type = "button" value = "getcookie"
         onclick = "getcookie()" >
<script>
    function setcookie() {
        document.cookie = "Cookie is set";
    }
    function getcookie() {
        if (document.cookie.length != 0) {
            alert (document.cookie);
        else {
            alert ("no cookie");
        }
    }
```

2) Set Expiry date.

document.cookie = "CookiSet ; expires = 27 Mar 2023 00:00:00 UTC";

Now, it will show no cookie set.

3) Cookie Max-age — set max time of cookie in seconds

document.cookie = "cookie set ; max-age =" + (20) + " ;" ;

4) Path

document.cookie = "cookie set ; path = / /";

5) Domain

document.cookie = "cookie set" ;
domain = websitename.com

☆ One cookie cannot be assigned Multiple values. To store in one cookie make it a object, or make n no. of cookies

1)

* Setting Multiple cookies at once which will not happen

```
<input type = "button" value "Set"
        onclick = "setcookie()";
<input type = "button" value = "get"
        onclick = "getcookie()";

<script>
    function setcookie () {
        document.cookie = "cookie1"
        ; "cookie2" ; "cookie3";
    }
    function getcookie() {
        if(document.cookie.length!=0) {
            alert(document.cookie);
        }else{
            alrt ("no cookie");
        }
    }
</script>
```

Output when we click on Set it will set & get cookie will always give cookie1 in alrt beoz it cannot have multiple values, it only takes first value

Agar input k value to cookie me store kna hto use id, cookie = doc.getElById(v).value;

2)

instead he can write it 3 time to view all cookies

```
function setcookie(){
        document.cookie = "cookie1";
        document.cookie = "cookie2";
        document.cookie = "cookie3";
```

=) it will work

3) By making object

```
function setcookie(){
        var object1 = {};
        object1.one = "Cookie1",      // name
        object1.two = "Cookie2";      // Rno
        object1.three = "Cookie3";    // dept

        var jsonstring =
                JSON.stringify(object1);
        document.cookie = jsonstring;
}

function getcookie(){
        if(document.cookie.length != 0){
                var obj2 = JSON.parse(document.cookie);
                alert(obj2.name, obj2.two, obj2.three)
        else
                alert("No cookie");
```

to make it view properly

Deleting A Cookie

1) Expires = date & time with UTC
2) max-age = 0
3) Explicitely by chrome Settings.

To make array of cookie

```
var array = document.cookie.split("=");
alrt("Cookie Name:" + array[0] + "\n
  value:" + array[1]);
```

=> Name: Kashish output

We can change color of webpage & store it in cookie & next time when user open it, it views same color using window.load().

Cookie = "doc.value" →.

PROMISES Function calling another function is called call back

If promise is fulfilled Resolve is executed, otherwise Reject is executed.

Promise are executed for asynchronous functions.

A promise is special JS object that links producing code and consuming code together. The producing code takes whatever time it needs to produce promised Result, and then promise makes result available to all combined codes, when its ready.

Syntax -

```
let promise = new Promise(function(resolve, reject){
    //Code
});
```

Promise maintain both producing & consuming code.

Consuming code gets executed when promise code gives any result.

```
let myPromise = new Promise (function (resolve, reject)
{
    resolve();  // when successful
    reject();   // when error
});
// Consuming Code ( Must wait for fulfilled Promise)
myPromise.then(
    function(value) { /* code if successful */ },
    function(error) { /* code if error */ }
);
```

A Promise can be
- Pending - fullfiled - Rejected