**Javascript tutorial**

Javascript is language which is used in the making of website.

Javascript allow us to change the variable type in the run time.

**Let Vs var**

Variable declared with “Let” keyword having a blocked scope (means access from within that block) and can’t be re declared.

**Ex:** It is wrong way.

Let a=5 ; Let a=6 ;

We can’t re declare the “a” when we declared it with let

Let c=9

{

Let c=7;

}

**Note:** Both c are different

But in another way the variable declared with “var” keyword having the globally scope (means access from anywhere) and can be re declared.

**Ex:** It is possible.

var a=5 ; var a=6 ;

We can re declare the “a” when we declared it with “var”

var c=9

{

var c=7;

}

**Note:** Both c is have same value “c=7”.

Array:

Array is collection of data items.

In javascript we can store different type of element in the same array.

Arrays are mutable so it can be change.

Ex: let a=[12,”umair”,null,134.6];

**Arrays method:**

**Pop ():** it returns the last element of an array and also delete it from an array.

**Shift ():** it returns the starting element of an array and also change the existing array.

**unShift ():** it adds the element at the starting of an array and also change the existing array and give the length of an array.

**Push ():**it add the element at the end of an array and also give the new length of an array.

**delete** arrayname[index]:

it delete the element at given index but it can not change the length of an array.

**foreach() loop:**

foreach loop is a loop which is used for printing array methods or also performing the operation on array.

Foreach loop takes an function as argument or also told as takes the callback.

Foreach loop do not return the new array but it only used to manipulate the existing array element.

**Map function**

Map is a higher order function of array which is exist in advance javascript function.

Map function gives the callback and the return

The new array after the manipulation on existing array.

In callback it takes the argument (value,index,array);

**Filter function:**

Filter function is also the higher order function of javascript. It also returns the new array of that element which satisfied the given condition.

Let a=Arrayname.filter ((value)=> {

return value>5;

)

In which “a” is an array which contain that values which is greater than 5.

**Note:** filter function don’t change the our real array.

**Window object:**

Window object is a browser window object which provide some method to control it.

It is a global object.

Everything in javascript available in window object

**DOM:**

DOM stand for document object model.

Our complete html page is converted into javascript object and name as “document”.

The javascript representation of our page is called DOM.

**BOM**

BOM stand for browser object model which represents additional object provided by browser (host environments) means the browser where our application is running.

**DOM TREE**

Dom tree refers to the html page in which all the nodes are object.

There can be three types of nodes in Dom tree.

* Text node
* Element node
* Comment node.

**Auto correction:**

If any error is encountered, then browser tends to resolve it or fix it.

**Ex:**

If we close the “span” tag with “div” tag

<span>hello</div> => browser will fix it.

**Element accessing in DOM:**

**Children of an element:**

Direct as well as deeply nested elements of an element are called its children.

**Child nodes:**

Elements that are the direct child of another element are called their child nodes.

There are three ways to find the child’s of any element.

**firstChild:**

firstchild give the first node child of any element but it also includes the (text-nodes) as well as.

**Ex:** if we find the first child of body by using **(“document.body.firstChild”)** then it will return the text if this body contains the text as a first node.

**lastChild:**

(document.body.lastChild) will return the last child of body.

**childNodes:**

it will return the list of childs of any element which contains the comments, text, nodes.

**Note:**

Note that the list return by (document.body.childNodes) is not an array it is only a list but we can convert into a list by using Array.form(document.body.childNodes).

**parentNode:**

(document.body.parentNode) will return the parent of body nodes which includes the (text, element, comment).

**parentElement:**

(document.body.parentElement) will return the parent element which includes only tags not the (text, comments) as a parent.

**nextSibling:**

(document.body.nextSibling) will return the upcoming element of body.

**previousSibling:**

(document.body.previousSibling) will return the previous element of body.

**firstElementChild:**

(document.body.firstElemetChild) will only return the first element child if body contain the element as a first child.

**lastElementChild:**

same as firstElemetChild but returning the last element child.

**nextElementSibling:**

(document.body.nextElementSibling) it will return the next or upcoming element if it contain’s the next element.

**Some method which is used in DOM**

**matches:**

this method returns the true or false if the particular element contains the given CSS selector.

**Closest:**

This method returns the that element if its contains the particular CSS selector. This method only checks the CSS selector in their parents and itself.

**Contains:**

This method only returns the true or false if the element Contain the particular element as a child

**innerHTML:**

innerHTML return the inner html of any element. Element may contain text or any element.

It is not applicable of text or comment nodes only for elements.

**outerHTML:**

outerHTML return the inner html and as well as element itself.

**textContent:**

textContent return the text of any element.

**HTML attributes and their methods:**

* **setAttributes ():**

this method take the two argument first is “attribute-name” and second is “attribute-value”.

* **getAttributes ():**

this method takes argument as a attribute name and return the attributes value

* **has Attributes ():**

this method returns the true or false if the particular element contains the specific attribute.

**Note**

javascript allow us to set the attributes according to your choice by using “**data-attributeName”** and which we access by using the **“data.attributeName”** and we can also get the all user-define attributes by using **“elementName.dataset”.**

**Insertion methods:**

**ElementName.append(“elementName”):**

This method adds the element in last of mentioned element.

**ElementName.prepend(“elementName”):**

This method adds the element at the starting of mentioned element.

**ElementName.before(“elementName”):**

This method adds the element before the mentioned element.

**ElementName.after(“elementName”):**

This method adds the element after the mentioned element.

**ElementName.replaceWith(“elementName”):**

This method replaces the element with the mentioned element.

**insertAdjacentHTML:**

We can also insert the element/text/html in the particular element by using the (insertAdjacentHTML) method.

This method takes the two arguments.

1: where you insert the element, text, HTML.

So we have four option

1.beforebegin

2.beforeend

3.afterbegin

4.afterend

**Ex:** elementName.insertAdjacentHTML(“beforeend”, ”<h1>hello world</h1>”)

**Assessing classes of elements and setting them**

**Class List:**

classlist is the method which returns the list of any particular method.

**Note:**

We can set the class using “elementName.CLassName=”

**remove ();**

we can use the remove method to remove any class from particular element.

**Add ();**

we can use the add method to add any class in a particular element.

**Asynchronous Programming:**

Asynchronous programming is the way of doing programming in which we manage multiple tasks at the same time also called doing multiple tasks at the same time.

We can achieve Asynchronous Programming in javascript by using

* Call Backs
* Async/Await
* Promises

Synchronous Programming:

Synchronous Programming is the way of doing programming in which we do one task at a time.

***CallBacks:***

Callbacks are defined as when we pass the function as an argument to another function.

We use the callbacks when we want to call the function or (want to execute a particular block of code) after performing particular task.