**CONTENTS**

1.DHTML

2.DOM

3.DOM Objects

3.1.Window

3.2.Screen

3.3.Location

3.4.History

3.5.Navigator

3.6.Document

3.7.Element

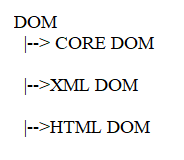
3.8.Collection

3.9.Small Project.

**1.DHTML(Dynamic Hypertext markup language)**

Dynamic HTML is not a markup language or programming language but it is a term that combines the features of various web development technologies for creating the dynamic web pages and interactive web pages. The DHTML consists of four languages or four components.

* HTML
* CSS
* JS
* DOM



**2.DOM(Dynamic/Document Object Model):**The Document Object Model (DOM) is a programming interface(or)API. This API consists of several Interfaces (specifications). Each Interface is blue print of object. It represents that object properties and methods.

Usually these interfaces are given by W3C. These interfaces are implemented as objects by browser vendors. This API becomes part of browser. We can access this API directly from java script. By this API, JS can dynamically.

Change the structure of HTML document/web page.

Add the new element to document.

Remove the existing element from the document.

Change the attributes.

React to event which occurs on web page.

---etc.

Note:- while web page is being loaded ,browser constructs DOM tree for that web page.

Example:

<!DOCTYPEhtml>

<htmllang="en">

<head>

    <title>Document</title>

</head>

<body>

    <p>sukumar</p>

    <div>

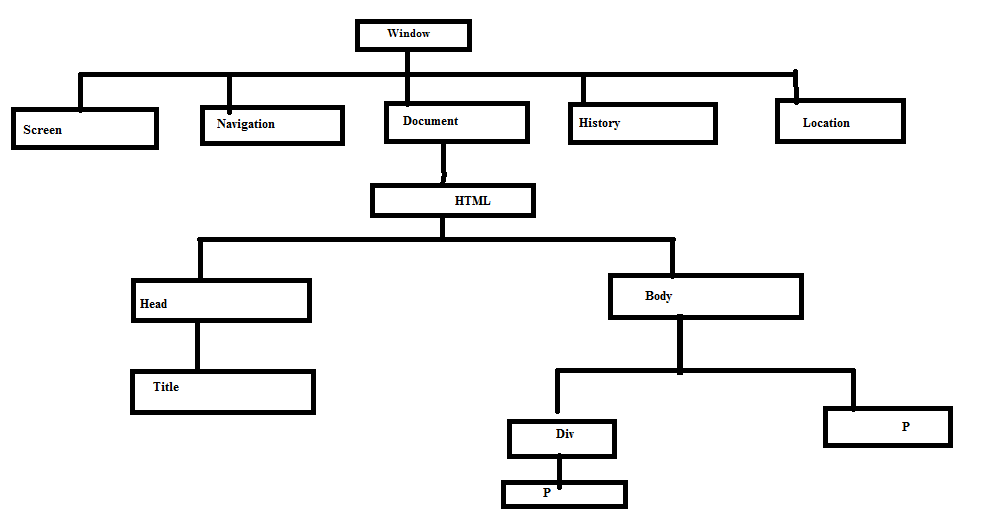
    <p>sv</p>

    </div>

</body>

</html>

Browser will construct following DOM Tree for above code.



**3. DOM objects:-**

**3.1. window:-** In browser, the window is opened. The window is represented by object.The object name is ‘window’.

I.Properties:

* Name:- It set or returns name of window.
* Length:- It set or returns no.of frames in window.
* innerHeight:- Returns inner height of window content area(Excludes scroll/Toolbar).
* InnerWidth:- Returns inner width of windows content area.(Excludes scroll/toolbar).
* outerHeight:- Returns outer height of window(including toolbar/scrollbars).
* Outerwidth:- Returns outer width of window(including toolbar/scrollbar).
* Frames:- Returns array of all frames in current window.
* Length:- It returns no.of frames in window.
* Top:- It returns top-most browser window.
* Parent:- It returns parent window object of current window.
* Navigator:- It returns navigator object of window.
* Screen:- It returns screen object for window.
* Closed:- Returns Boolean value indicating whether window has been closed or not.

II.Methods:

* + focus():- It sets current window to front side. It does not return value.
  + blur():-It take back focus from current window.It does not return value.
  + resizeTo(x,y):- It resizes the current window to new height and new width.
  + resizeBy(x,y):-It resize the current window by specified amount.
  + moveBy() :- It moves window relative to its current position.
    - Syntax:- moveBy(x,y)
      * Xand y may be positive or negative value.
  + stop():- It stops the window loading.
  + Prompt()
  + Alert()
  + Comfirm()
  + Close():- It closes the current window.
  + Open():- It opens new browser window.

Syntax- open(url,target,name,specs,replace);

1.url:- It is optional. It specifies url of page to open. If no URL is specified new window about blank is opened.

2.specs:- It is optional. The following values are supported.

a)status=0|1 b) scrollbars=0|1 c)titlebar=0|1 d)Menubar=0|1 e)resizable=0|1 f)height=value

g)width=value h)fullscreen=0|1 i)titlebar=0|1

3.replace:- It allows true or false.

4. The target is

\_blank:Url is loaded into new window.

\_parent:Url is loaded into parent frame.

\_top: utl is replaces any frame sets currently loaded.

\_self:Url replaces the current page.

5. It returns reference of created window.

Example:

<!DOCTYPEhtml>

<htmllang="en">

<head>

    <title>Document</title>

</head>

<body>

    <script>

        varx= window.open('file:///C:/Users/sukumar/Desktop/abc.html','\_blank','statusbar=no,height=400,width=400');

        x.moveTo(200,200);

        x.resizeTo(600,600);

        x.name='Sample';

        x.innerHeight=100;

        x.innerWidth=200;

        x.document.title='Workout'

        x.document.bgColor='red';

        x.document.writeln('windowName:'+x.name);

        x.document.writeln('window outerheight:'+x.outerHeight);

        x.document.writeln('window outerwidth:'+x.outerWidth);

        x.document.writeln('window Innerwidth:'+x.innerWidth);

        x.document.writeln('window InnerHeight:'+x.innerHeight);

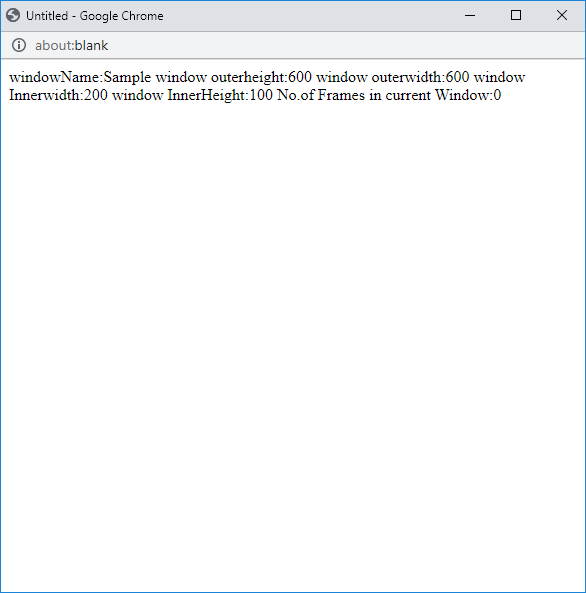
        x.document.writeln('No.of Frames in current Window:'+x.length);

    </script>

    <h1>Tai h r u?</h1>

</body>

</html>



**3.2.Screen**:This object contain information about user screen.

Properties:-

a)availHeight:- It returns height of user screen in pixels minus interfaces features like window taskbar.

b)availwidth:- It returns width of screen in pixels,minus interface features like window taskbar.

c)height:- It returns total height of screen.

d)width:- It returns total width of screen.

Example:

<!DOCTYPEhtml>

<htmllang="en">

<head>

    <title>Document</title>

</head>

<body>

    <script>

        varx= window.open('file:///C:/Users/sukumar/Desktop/abc.html','\_blank','statusbar=no,height=400,width=400');

        x.moveTo(200,200);

        x.resizeTo(600,600);

        vary=x.screen;

        x.document.writeln('Screen Available Height:'+y.availHeight);

        x.document.writeln('Screen Available Width:'+y.availWidth);

        x.document.writeln('Screen Total Height:'+y.height);

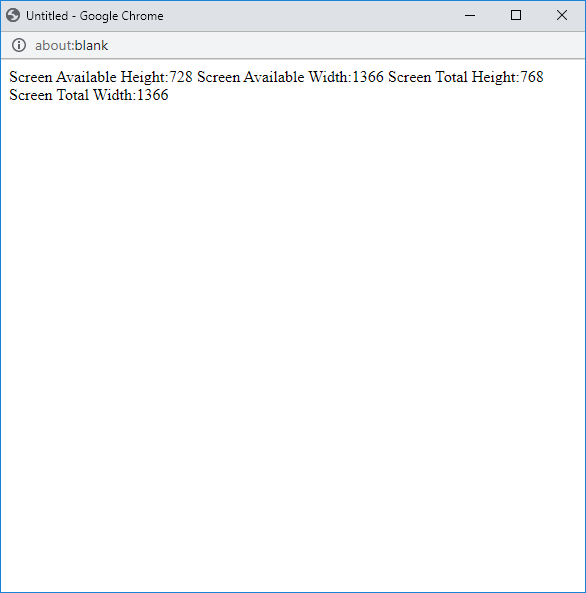
        x.document.writeln('Screen Total Width:'+y.width);

    </script>

    <h1>Tai h r u?</h1>

</body>

</html>



**3.3.Location**:The location object contains information about current URL.

Properties:

* + protocol:- sets or returns the protocol of URL.
  + hostname:- sets or returns the host name of URL.
  + port:- It sets or returns port of URL.
  + pathname:- It sets or returns the absolute pathname of resource in URL.
  + href:- It returns or sets entire URL.
  + hash:- It set or return anchor part of URL.
  + search:- set or returns query string part of URL.

Methods:

assign():-

syntax:- assign(URL)

<URL:-> It specifies URL of page to navigate to .

It loads a new document.

It does not return any value.

reload():-

syntax:- reload(true|false)

The argument is optional.

False:- It is default. It reloads page from cache.

True:- It reloads page from the server.

replace():- It replace a document with new one.

Syntax:- replace(URL)

Diff between replace and assign:- Replace() removes URL of current document from document history, meaning. It is not possible to navigate back to original document.

**3.4.Histroy:**The history object contains URL’s visited by user from the web document/window.

Properties:

1. length: It has no.of urls in history.

Methods:

1.back():- This method loads previous URL in history list. This is same as clikcking “back button” in your browser.

Syntax:- back();

2.forward():- This method loads next url in history list. This is same as clicking “forward button “ in browser.

Syntax::- forward();

3.go():-

Syntax:- go(url|number)

url/number:- This parameter can be which go to url with in specific position. (-1, goes back one page, 1 goes forward one page). This method loads specific URL from history list. We can use this method to load previous or next page.

Example:1

One.html

<!DOCTYPEhtml>

<htmllang="en">

<head>

    <metacharset="UTF-8">

    <metahttp-equiv="X-UA-Compatible" content="IE=edge">

    <metaname="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <h1>Home Page</h1>

    <ahref="C:\Users\sukumar\Desktop\Front-End Technologies\Practise\second.html" >GoTo Second Page</a>

    <buttononclick="two()">History</button>

    <buttononclick="three()">Location</button>

    <script>

         functiontwo()

        {

            console.log('Histrory Length At first page:'+history.length);

        }

        functionthree()

        {

         vary=window.location;

        console.log('Current Document URL:'+y.href);

        console.log('URL protocol:'+y.protocol);

        console.log('URL domain:'+y.host);

        console.log('URL port:'+y.port);

        console.log('URL quey string:'+y.search);

        console.log('URL absolute Path:'+y.pathname);

        }

    </script>

</body>

</html>

Two.html

<!DOCTYPEhtml>

<htmllang="en">

<head>

    <metacharset="UTF-8">

    <metahttp-equiv="X-UA-Compatible" content="IE=edge">

    <metaname="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <h2>Second Page</h2>

    <ahref="C:\Users\sukumar\Desktop\Front-End Technologies\Practise\third.html">Third page</a>

    <buttononclick="back1()">Goback</button>

    <buttononclick="forward1()">Goforward</button>

    <buttononclick="Hlenth()">History\_Length</button>

    <buttononclick="Loc()">Location-Object</button>

    <script>

        functionback1()

        {

            history.back();

        }

        functionforward1()

        {

            history.forward();

        }

        functionHlenth()

        {

            console.log('History Length At Second page:'+history.length);

        }

        functionLoc()

        {

            console.log("==================Second Page======================");

            vary=window.location;

        console.log('Current Document URL:'+y.href);

        console.log('URL protocol:'+y.protocol);

        console.log('URL domain:'+y.host);

        console.log('URL port:'+y.port);

        console.log('URL quey string:'+y.search);

        console.log('URL absolute Path:'+y.pathname);

        }

    </script>

</body>

</html>

Third.html

<!DOCTYPEhtml>

<htmllang="en">

<head>

    <metacharset="UTF-8">

    <metahttp-equiv="X-UA-Compatible" content="IE=edge">

    <metaname="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <h3>Third page</h3>

    <buttononclick="one()">back</button>

    <buttononclick="two()">History</button>

    <buttononclick="three()">location</button>

    <script>

        functionone()

        {

            history.back();

        }

        functiontwo()

        {

            console.log('History Length At third page:'+history.length);

        }

        functionthree()

        {

            console.log("================Third Page==================");

            vary=window.location;

        console.log('Current Document URL:'+y.href);

        console.log('URL protocol:'+y.protocol);

        console.log('URL domain:'+y.host);

        console.log('URL port:'+y.port);

        console.log('URL quey string:'+y.search);

        console.log('URL absolute Path:'+y.pathname);

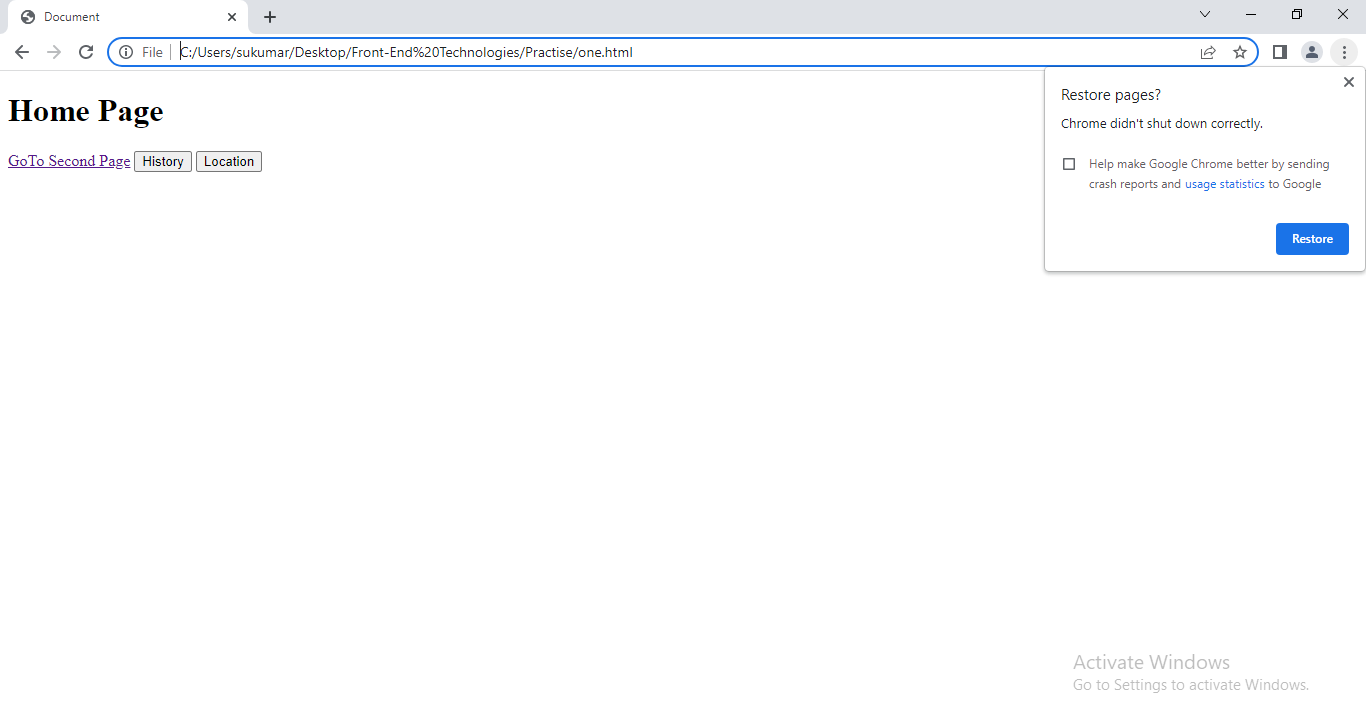
        }

    </script>

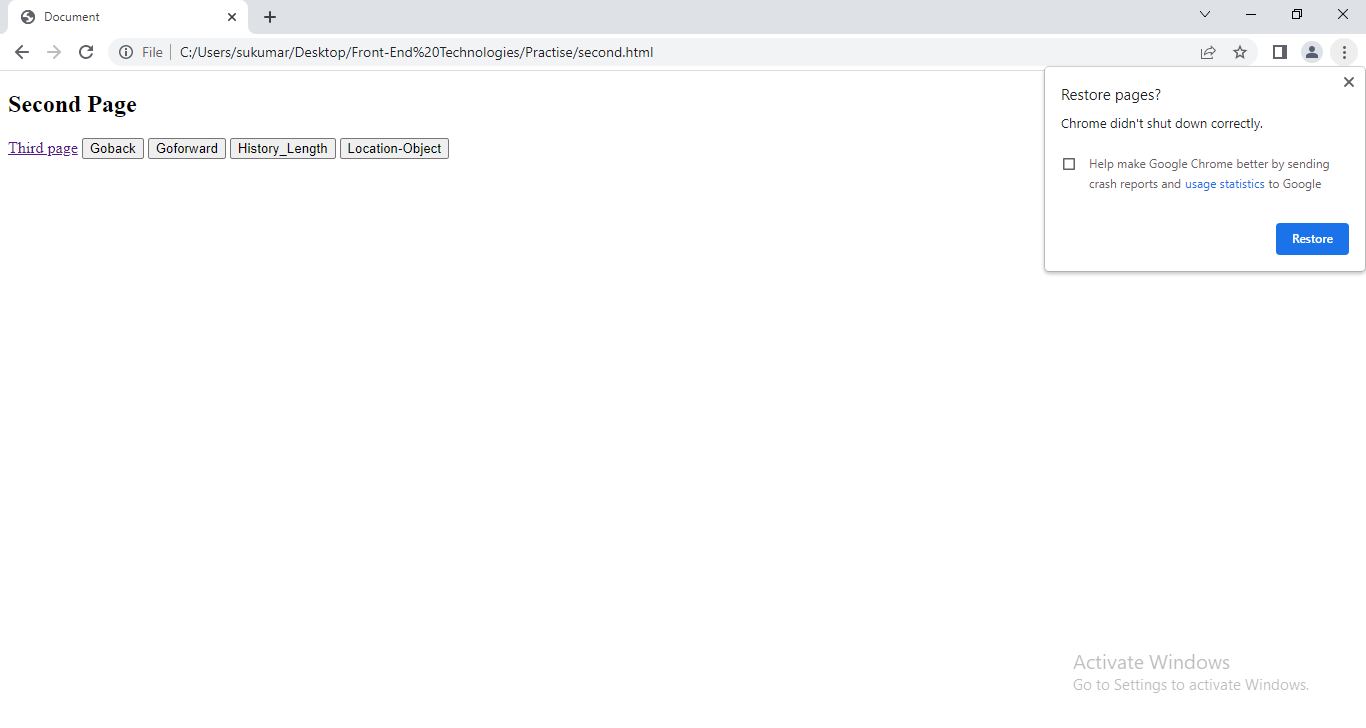
</body>

</html>

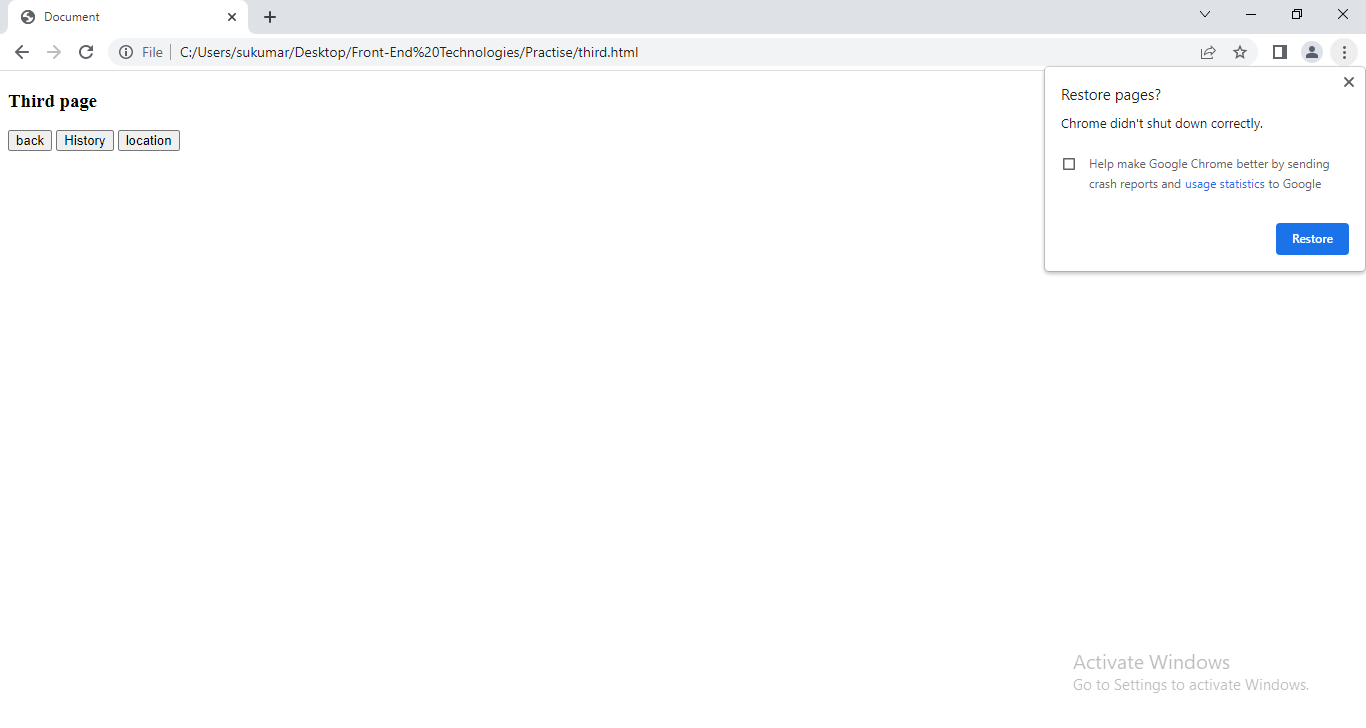
Output:



After pressing the history, location & goTo Secondpage



After pressing the history\_length, location\_object & thirdpage



**3.5.Navigator**:- The navigator object contains browser information.

Properties:-

1.appName:- It returns browser name.

2.language:- It returns language of browser.

3.appVersion:- It returns version info about browser.

4.online:- It returns Boolean value that specifies whether browser is in online or offline mode.

5.platForm:- It returns for which platform the browser in compiled.

6.product:- It returns engine name of browser.

7.cookieEnabled:- It retuns boolena value that specifies whether cookie is enabled in browser.

8.plugIns:- It is array. It contains information of installed plugins in browser.

9.bluetooth.

..etc.

**3.6.Document:**A Document object represents the HTML document that is displayed in that window.

Properties:

* + - Title: It holds the title of document.
    - URL: It contains url of document. It is read only property.
    - bgColor: we set/get the background color of html document.
    - fgColor: we set/get the text color of document.
    - Head: It returns <head> node object.
    - Body: It returns <body> node object.
    - Lastmodified: It is read only string that specifies date of most recent changes to document.
    - Images: It returns Collection object. It contains image objects. The images have been defined in web page.

Methods:

1.createElement:-

Syntax:- Var element=document.createElement(tagName[,-options]);

\* This method creates HTML element specified by tag name or an html unknown element if tag name is not specified.

\*TagName is name of element you want to create.

\* It returns Element object.

2. appendChild():- This method adds element to document as last child.

Syntax:

appendChild(element object);

3.removeChild():- This method removes existing child element from document . If element does not exist, this method returns null.

Syntax:

removeChild(element object);

4.replaceChild():- This method replaces the existing element with new element in document. It old element is not in document, then it returns null value.

Syntax:

repalceChild(newelement,oldelement);

5.getElementById():-

Syntax:

Var name= getElementById(element-id);

It returns Element object having given id value.

6. getElementByName():-

Syntax:

Var name=getElementByName(name of element);

It returns Collection object which having all element with given name.

7. getElementByTagName():

Syntax:

Var name=getElementByTagName(tagname (or) \*)

If argument is \* then this method returns all elements object in HTML document as collection object.

If argument is specific tag name then It returns Collection object which having only specific element objects.

8. getElementByClassName()

Syntax:

Var name=getElementByClassname(class);

It returns the collection object.

9.QuerySelector():The Document's querySelector method returns the first Element within the document that matches the specified selector or a group of selectors. If no matches are found, null is returned.

Syntax:

Var name=querySelector(‘#id/.class/tagname’);

10.QuerySelectorAll():-The querySelectorAll returns a static NodeList representing a list of the document's elements that match the specified group of selectors.

Syntax:

Var name=querySelectory(“.class/tagname’);

11.setTimeOut():-This method calls a function after a number of milliseconds.

Syntax:

Var name=setTimeout(function, milliseconds, param1, param2, ..).

param1,param2 are optional.

These are arguments to function.

It returns id of timer.

12.setInterval():-This method continues calling the function until  clearInterval()  is called, or the window is closed.

Syntax:

Var name=setInterval(function,milliseconds);

13. clearInterval():- This method stops repeated execution of function.

Syntax:

clearInterval(Id which is returned by setInterval);

**3.7. Element**:- The Element object represent Document element.

Properties:

* Id: It set or returns id of element.
* tagName: It returns only tagName of element.
* className: It returns value of class attribute of element.
* firstElementChild: It holds first Child element of element.
* laseElementChild: It holds last child element of element.
* firstSibling: It stores the first sibling of element.
* lastSibling: It stores the last sibling of element.
* previousSibling:- It contains the previous sibling of element.
* nextSibling: It store the next sibling of element.
* Parent : It hold parent element of element.
* childElementCount:- we get the no.of children of current element.
* Style:- we can set/get style attribute value to element.
* innerHTML: we can set/get the content of element.
* innerText: we can set/get the content of element.

Methods:-

1. has Attributes():- Returns true, If element has any attribute, otherwise false.

Syntax:-

Boolean Element. has Attributes();

2.has Attribute():-

Syntax: -

Element.hasAttribute(attributename);

It returns true , if element has specific attribute other wise false,

3.hasChildNodes:-

Syntax:-

Element.hasChildNodes();

It returns true if specified Element has child nodes, otherwise false.

4.setAttribute:- This method adds specified attribute to an element and gives it specified value.

If specified attribute already exists, only value is set/changed.

Bad Syntax:- Element.setAttribute(attributeName,value)

It does not return any value.

Good Syntax:-

Element.style.attributeName=value;

5.getAttribute():-

Syntax:

Element.getAttribute(Attribute name);

It returns a String, representing specified attribute value. If attribute does not exist, the return value is NULL or EMPTY String.

6.removeAttribute():- This method removes specified attribute from an element.

Syntax:-

Element.removeAttribute(attributeName)

It does not return any value.

7. appendChild()

8. removeChild()

3.8.Collection: DOM collection object is an array that contains list of HTML elements. DOM collection object contains the following properties and methods.

1.length:- It returns number of elements in the collections.

2. item(index):- Returns element at the specified index in the collections.

3.namedItem:- returns element with the specified ID, or name, in the Collections.

<html>

<head>

<title>sample</title>

</head>

<body>

<ponclick="abc()">one</p>

<p>two</p>

<p>three</p>

<script>

functionabc()

{

 vara=document.getElementsByTagName('p');

 for(i=0;i<a.length;i++)

    {

       a[i].style='color:red';

    }

}

</script>

</body>

</html>

3.9.SMALL PROJECT

I created small project using only JS and DOM.

<!DOCTYPEhtml>

<htmllang="en">

<head>

   <metacharset="UTF-8">

   <metahttp-equiv="X-UA-Compatible" content="IE=edge">

   <metaname="viewport" content="width=device-width, initial-scale=1.0">

   <title>Products</title>

</head>

<body>

   <script>

      document.bgColor='red';

      vartEle=document.createElement("h1");

      tEle.innerHTML='Electronic Goods';

      tEle.style='text-align:center;color:white;border:1px solid black';

      document.body.appendChild(tEle);

      vareData={'prod1':{'Name':'Laptap','Price':2000,'Company':'Sony','Image':'laptap.jpg'},'Prod2':{'Name':'Mobile','Price':1000,'Company':'Vivo','Image':'mobile.jpg'}};

      vari;

      vardisplay=function(data)

      {

         vardCon=document.createElement("div") ;

         dCon.style='border:1px solid blue;padding:0px;float:left; margin-left:5px;';

         varUlist=document.createElement("ul");

         Ulist.style='list-style-type:none;margin:15px;padding:0px;';

         for(varj in data)

         {

            varlItem=document.createElement("li");

            if(j=='Image')

            {

              varij=document.createElement("img");

              ij.setAttribute('src',data[j]);

              ij.setAttribute('height','100px');

              ij.style='margin-left:5px;margin-top:5px;';

              lItem.appendChild(ij);

              lItem.style='border:1px solid green;margin-top:2px;'

            }

            else{

               lItem.style='border:1px solid green;margin-top:2px;width:250px;height:25px;'

               lItem.innerHTML=j+':'+data[j];

            }

            Ulist.appendChild(lItem);

         }

         dCon.append(Ulist);

         document.body.appendChild(dCon);

      }

      for(i in eData)

      {

         display(eData[i]);

      }

   </script>

</body>

</html>

Output:

