DOM:-

-DOM stands for Document Object Model.

-it is not a part of JavaScript, it is a part of browser.

-dom is a tree like hierarchical structure.

-each elements of the html document are stored in the dom structure as nodes.

-dom is used to access the html elements as well as the css properties, and manipulate them.

-it is used to add behaviors in our UI.

-to communicate with the browser dom we have an object in js called document object.

-all the dom methods, properties and events is available within this document object.

Structure of dom:-

-all the elements are stored in this tree like structure as nodes.

properties to manipulate text content of the elements:-

1. textContent

2.innerText

3.innerHTML

1. textContent:-

- it is used to manipulate the text content of any html elements.

-it is used to access the visible text as well as the hidden text of any html elements.

-it reads all the text content as string.

-html tags are not readable as an element by textContent.

example:- <h1 id="hello">hello</h1>

let h1Tag=document.getElementById("hello");

h1Tag.textContent='good afternoon';

2. innerText:-

- it is used to manipulate the text content of any html elements.

-it is used to access the visible text but not the hidden text of any html elements.

-it reads all the text content as string.

-html tags are not readable as an element by innerText.

example:- <h1 id="hello">hello</h1>

let h1Tag=document.getElementById("hello");

h1Tag.innerText='good afternoon';

3.innerHTML:-

- it is used to manipulate the text content of any html elements.

-it is used to access the visible text but not the hidden text of any html elements.

-it reads all the text content as string.

-html tags are readable as an element by innerHTML.

-we can add any html tags also as a child elements inside of the targetted element.

example:- <h1 id="hello">hello</h1>

let h1Tag=document.getElementById("hello");

h1Tag.innerHtml='<span>good afternoon</span>';

Selectors in js

-If we want to target Html elements in javascript to perform some manipulation process for that we can use selectors in js.

-in js there are some dom methods using which we will target html elements.

following are the different types of selectors in js:-

1. getElementById

2. getElementsByClassName

3. getElmentsByTagName

4. querySelector

5. querySelectorAll

1. getElementById:-

-if we want to target an element using id, then we use getElementById method.

-syntax:- <h1 id="idName">Hello</h1>

```
 let variableName=document.getElementById("idName");
```

-the element which we want to target that element will be returned and stored in the variableName.

-we can access that element via the variable.

example:- <p id="para">hello</p>

```
 var pTag= document.getElementById("para");

 console.log(pTag);

 pTag.innerText="i am targetted using id"
```

2. getElementsByClassName:-

-if we want to target any element or multiple elements using class name, then we use getElementsByClassName method.

-syntax:- <h1 class="className">i am h1</h1>

```
   <p class="className">i am p</p>

  <div class="className">i am div</div>

 let variableName=document.getElementsByClassName("className");
```

-the elements which we want to target those elements will be returned as HtmlCollection in the format of an arraylist and stored in the variableName.

-we can access that HtmlCollection via the variable.and we can use indices to access indivisual element.

example:- <div class="cont">1</div>

```
  <div class="cont">2</div>

  <div class="cont">3</div>

  <div class="cont">4</div>

  <div class="cont">5</div>

  let divs= document.getElementsByClassName("cont");

    console.log(divs);

    for(let i=0;i<divs.length;i++){

      divs[i].innerText=i+2;    }
```

3. getElementsByTagName:-

-if we want to target any element or multiple elements using Tag name, then we use getElementsByTagName method.

-syntax:- <h1>i am h1</h1>

```
     <h1>i am h1</h1>
     <h1>i am h1</h1>
    let variableName=document.getElementsByTagName("TagName");
```

-the elements which we want to target those elements will be returned as HtmlCollection in the format of an arraylist and stored in the variableName.

-we can access that HtmlCollection via the variable.and we can use indices to access indivisual element.

example:- <button>click1</button>

```
      <button>click2</button>
      <button>click3</button>
      <button>click4</button>
      let btn=  document.getElementsByTagName("button");
      console.log(btn);
      for(let i=0;i<btn.length;i++){
        btn[i].innerHTML="next";
      }
```

4. querySelector:-

-if we want to target an element based on the given query. then we use querySelector.

-using querySelector, we can target an element based on id,classname and tagname as well.

-we just pass the #id or .classname or tagname within the args as a string.

syntax:- <h1 id="query">hello</h1>

```
     <p class="query">hello</p>
           <div>hello</div>
     let variableName=document.querySelector("query");
```

-it will return the first element which will match with the query.

-it will return a single element as value.

-we can directly access that element using the variableName.

example:-

```
<h1 id="hello">hello</h1>

    <p class="para">i am para1</p>
    <p class="para">i am para2</p>
    <p class="para">i am para3</p>
    <p class="para">i am para4</p>

    <button>click1</button>
    <button>click2</button>
    <button>click3</button>

   let h1Tag=document.querySelector("#hello");
    console.log(h1Tag)

   let pTag=document.querySelector(".para")
   console.log(pTag);

    let btnTag=document.querySelector("button")
    console.log(btnTag);
```

5. querySelectorAll:-

-if we want to target any elements based on the given query. then we use querySelectorAll.

-using querySelectorAll, we can target any element based on id,classname and tagname as well.

-we just pass the #id or .classname or tagname within the args as a string.

syntax:-

```
<h1 id="query">hello</h1>
   <p class="query">hello</p>
         <div>hello</div>
   let variableName=document.querySelectorAll("query");
```

-it will return all the element which will match with the query.

-it will return a nodeList with each matching element as element of the nodelist.

-we can access that nodelist using the variableName.

-we can access the indivisual elements via indices.

-we can also use forEach method as well as length property to access each element of the nodeList.

example:-

```
<h1 id="hello">hello</h1>


<p class="para">i am para1</p>
<p class="para">i am para2</p>
<p class="para">i am para3</p>
<p class="para">i am para4</p>


<button>click1</button>
<button>click2</button>
<button>click3</button>
let h1Tag=document.querySelectorAll("#hello");
console.log(h1Tag);
h1Tag[0].innerHTML="i am changed"


let pTags=document.querySelectorAll(".para");
console.log(pTags);
pTags.forEach(ele=>console.log(ele)
)
let btnTags=document.querySelectorAll("button");
console.log(btnTags);
```

createElement Method :-

-if we want to create any element in js, then we use createElement method.

-after creating the element we can add the content using properties like innerText, innerHtml or text content.

-after creating the element we have to append or insert the element in an already existing element.

-it accepts one args as a string, that is the name of the tag we want to create.

syntax:- let tag=document.createElement("tagName");

example:- let h1Tag=document.createElement("h1")

```
        console.log(h1Tag);
         h1Tag.innerHTML="Hello";
         document.body.appendChild(h1Tag);
```

appendChild method:-

-if we want to append or insert any element inside of a parent element, we use appendChild method.

-After creating an element we have to append that created element within an already existing element, that can be done by using appendChild method.

-it accepts one args, which is the element which we want to append.

syntax:- document.parentElement.appendChild(elementName);

example:- let h1Tag=document.createElement("h1")

```
        console.log(h1Tag);
        h1Tag.innerHTML="Hello";
        document.body.appendChild(h1Tag);
```

remove method:-

-if we want to delete any element, we use remove method.

-it is a no args method.

syntax:- elementName.remove();

example:-

```
      h1Tag.remove();
```