String

String

- JavaScript uses string data type to represent textual data. It is marked by enclosing in quotes or double quotes.

- Each element in the String occupies a position index. The first element is at index 0, the next at index 1, and so on.

- The length of a String is the number of elements in it.

- 'This is a string'
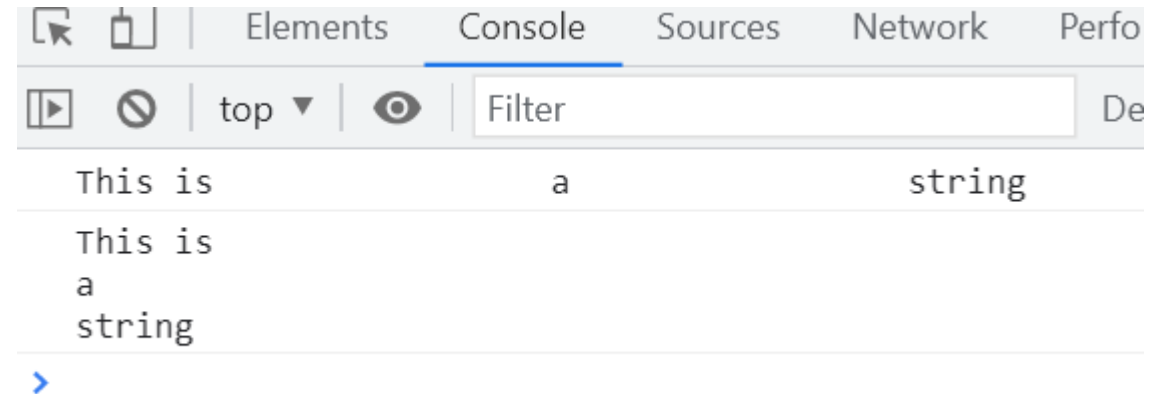
-    "Another string here"

Escape sequences

- Sometimes you need to put quotation mark inside a string. JavaScript provides escape sequences in the form of backslash \ to handle this kind of operation

- 'I\'m a Software Developer and I work with \'JavaScript\''

- Output
  - // I'm a Software Developer and I work with 'JavaScript'

Escape Sequence

- Other forms of backslash escape sequence includes: ' single quote " double quote \ backslash \n new line \t tab \b backspace You might use them when you deal with string data types

```
demo.html
1    <html>
2    <body>
3      <script type="text/javascript">
4           // Multi line string code with backslash
5               var a = "This is \
6                  a \
7                  string"
8           // Multi line string interpretation
9               var b =    "This is \na \nstring"
0          console.log(a);
1          console.log(b);
2      </script>
3    </body>
4    </html>
```

Elements   Console   Sources   Network   Perfo

top ▼   Filter   De

This is   a   string

This is
a
string

>

String template literal

- Template literals are string literals that allows developer to embed expressions in a string. They are marked by the enclosing back-tick mark ``` that surrounds them.

```
// template literals can create multi line string interpretation without \n
 `This is
 a
 string`
```

- Template literals can use variables or expressions as part of the string. This is done by using the ${} to enclose the expression.

```
let name = "Akash";

`Your name is ${name}.`

// normal string equivalent
'Your name is ' + name + ' I Love JS.'

// Using expressions
`The result of 10 + 19 is ${10+19}`
```

== and === operator

- Both double equals == and triple equals === operator is used for comparing between two values on which the operator is used on.

- The difference between the two operators is that the double equals == will compare the values loosely, meaning that it will try to convert values with different types before comparing them.

- The triple equals === won't convert values of different types. It will simply return false when comparing values of different types.

- Comparing two different values between the number value 0 and boolean value false:

Example

- console.log(0 == false); // true

- console.log(0 === false); // false
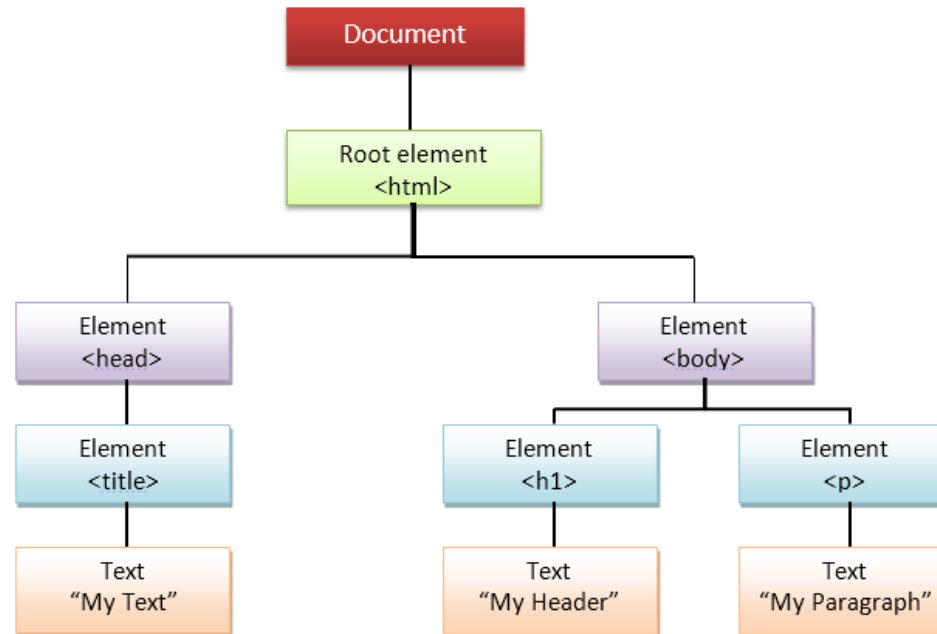

- console.log(0 === 0); // true

JS Dom

JS

What is DOM in JavaScript?

- JavaScript can access all the elements in a webpage making use of Document Object Model (DOM). In fact, the web browser creates a DOM of the webpage when the page is loaded. The DOM model is created as a tree of objects like this:

Selecting the Topmost Elements

- The topmost elements in an HTML document are available directly as document properties. For example, the <html> element can be accessed with document.documentElement property,

- whereas the <head> element can be accessed with document.head property, and the <body> element can be accessed with document.body property.

Example

```html
demo.html
1    <!DOCTYPE html>
2    <html lang="en">
3    <head>
4        <meta charset="utf-8">
5        <title>JS Select Topmost Elements</title>
6    </head>
7    <body>
8        <script>
9        // Display lang attribute value of html element
10       alert(document.documentElement.getAttribute("lang")); // Outputs: en
11
12       // Set background color of body element
13       document.body.style.background = "yellow";
14
15       // Display tag name of the head element's first child
16       alert(document.head.firstElementChild.nodeName); // Outputs: meta
17       </script>
18   </body>
19   </html>
```

How to use DOM and Events

- Using DOM, JavaScript can perform multiple tasks.

- It can create new elements and attributes, change the existing elements and attributes and even remove existing elements and attributes.

- JavaScript can also react to existing events and create new events in the page.

Method

- getElementById()

- getElementsByName()

- getElementsByClassName()

- getElementsByTagName()

getElementById()

- To access elements and attributes whose id is set.

- The id is case-sensitive. For example, the 'root' and 'Root' are totally different id.

- id is unique within a document. However, HTML is a forgiving language. If a document has more than one element with the same id, the getElementById() method returns the first one it encounters.

```html
demo.html
1  <html>
2  <body>
3    <h1 id="one">Welcome</h1>
4     <script type="text/javascript">
5          var text = document.getElementById("one").innerHTML;
6          alert("Value is " + text);
7    </script>
8  </body>
9  </html>
```

```html
<html>
<body>
 <h1 id="one">Welcome</h1>
  <script type="text/javascript">
      var text = document.getElementById("one").innerHTML;
      alert("Value is " + text);
  </script>
</body>
</html>
```

Get Value of Form

```
demo.html
1  <html>
2  <body>
3      <script type="text/javascript">
4          function getnumber(){
5              var number=document.getElementById("number").value;
6          alert(number);
7      }
8      </script>
9  <form>
10      Enter No:<input type="text" id="number" name="number"/><br/>
11      <input type="button" value="Click" onclick="getnumber()"/>
12  </form>
13  </body>
14  </html>
```

```html
<html>
<body>
    <script type="text/javascript">
        function getnumber(){
            var number=document.getElementById("number").value;
            alert(number);
        }
    </script>
<form>
    Enter No:<input type="text" id="number" name="number"/><br/>
    <input type="button" value="Click" onclick="getnumber()"/>
</form>
</body>
</html>
```

demo.html   ×   +

← → C ⌂ ⓘ File | D:/Delete/test/demo.html

Enter No: 10

Click

This page says

10

OK

getElementsByName()

- The **document.getElementsByName()** method returns all the element of specified name.

- The getElementsByName() method returns an array-like object called HTMLCollection which stores all elements that matches the value passed as the method's argument.

Example

```
demo.html
1   <html>
2   <body>
3
4     <p name="opening">Opening paragraph</p>
5
6     <script type="text/javascript">
7             let elements = document.getElementsByName("opening");
8             console.log("a"+elements); // [p]
9             console.log(elements[0]);
10      </script>
11  </body>
12  </html>
```
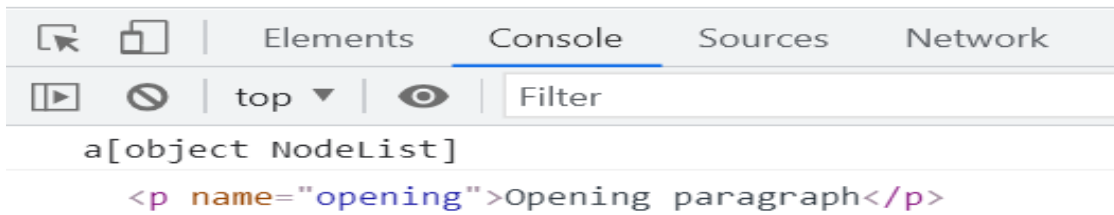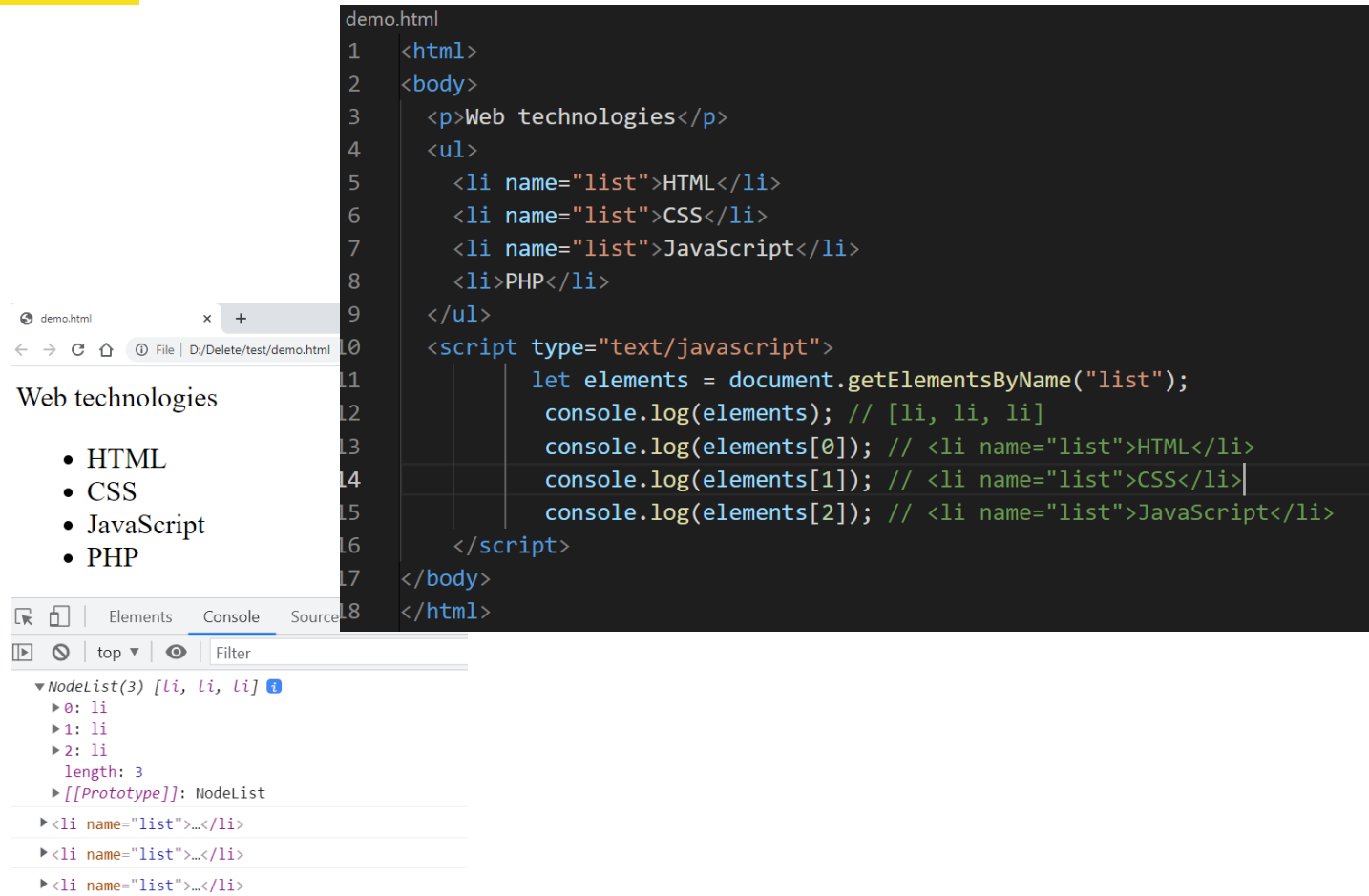
demo.html   ×   +

← → C ⌂   ⓘ File | D:/Delete/test/demo.html

Opening paragraph

Elements   Console   Sources   Network

▷| ⊘ | top ▼ | 👁 | Filter

a[object NodeList]

`<p name="opening">Opening paragraph</p>`

>

```
<html>
<body>

    <p name="opening">Opening paragraph</p>

    <script type="text/javascript">
            let elements = document.getElements
ByName("opening");
            console.log("a"+elements); // [p]
            console.log(elements[0]);
    </script>
</body>
</html>
```

Example

```html
<html>
<body>
  <p>Web technologies</p>
  <ul>
    <li name="list">HTML</li>
    <li name="list">CSS</li>
    <li name="list">JavaScript</li>
    <li>PHP</li>
  </ul>
  <script type="text/javascript">
      let elements = document.getElementsByName("list");
      console.log(elements); // [li, li, li]
      console.log(elements[0]); // <li name="list">HTML</li>
      console.log(elements[1]); // <li name="list">CSS</li>
      console.log(elements[2]); // <li name="list">JavaScript</li>
  </script>
</body>
</html>
```

getElementsByClassName()

- getElementsByClassName() method returns an object containing all the elements with the specified class names in the document as objects.

- Each element in the returned object can be accessed by its index.

- This method can be called upon any individual element to search for its descendant elements with the specified class names.

```
document.getElementsByClassName(classnames);
```

Example

```html
demo.html
1   <html>
2   <body>
3
4   <button onclick="red()" class="red">
5           Click to change to red button
6   </button>
7
8     <script type="text/javascript">
9         function red() {
10            document.getElementsByClassName('red')[0].style.backgroundColor='red';
11        }
12    </script>
13  </body>
14  </html>
```
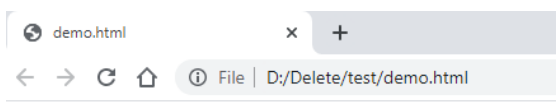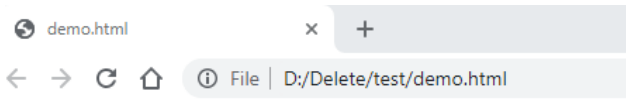
```html
<html>
<body>

<button onclick="red()" class="red">
        Click to change to red button
</button>

  <script type="text/javascript">
       function red() {
           document.getElementsByClassName('red')[0].style.backgroundColor='red';
       }
   </script>
</body>
</html>
```

demo.html   ×   +
← → C ⌂   ⓘ File | D:/Delete/test/demo.html

Click to change to red button

demo.html   ×   +
← → C ⌂   ⓘ File | D:/Delete/test/demo.html

Click to change to red button

```
demo.html
1   <html>
2   <body>
3   <p class="test">This is a paragraph of text.</p>
4     <script type="text/javascript">
5           var matches = document.getElementsByClassName("test");
6           console.log(matches);
7           // Displaying the selected elements count
8           document.write("Number of selected elements: " + matches.length);
9           // Applying bold style to first element in selection
10          matches[0].style.fontWeight = "bold";
11          matches[0].style.border="5px solid red";
12          matches[0].style.background = "yellow";
13          matches[0].style.fontStyle = "italic";
14      </script>
15  </body>
16  </html>
```

This is a paragraph of text.

Number of selected elements: 1

HTMLCollection [p.test]
  ▶ 0: p.test
    length: 1
  ▶ [[Prototype]]: HTMLCollection

```html
<html>
<body>
<p class="test">This is a paragraph of text.</p>
  <script type="text/javascript">
      var matches = document.getElementsByClassName("test");
      console.log(matches);
      // Displaying the selected elements count
      document.write("Number of selected elements: " + matches.length);
      // Applying bold style to first element in selection
      matches[0].style.fontWeight = "bold";
      matches[0].style.border="5px solid red";
      matches[0].style.background = "yellow";
      matches[0].style.fontStyle = "italic";
    </script>
</body>
</html>
```
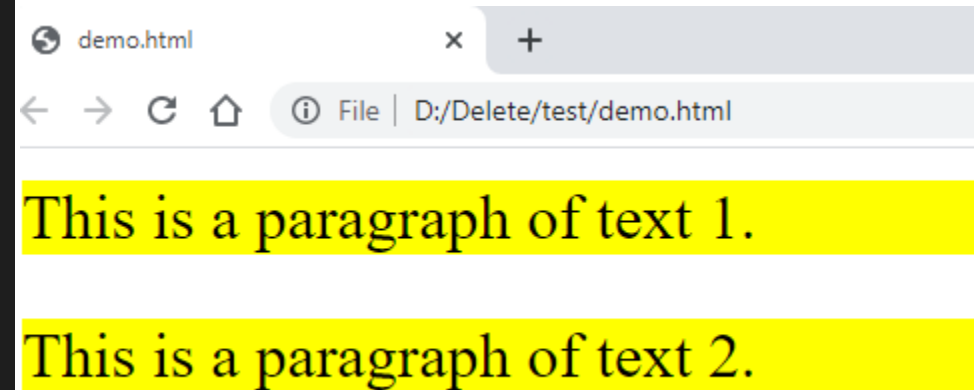
getElementsByTagName()

- You can also select HTML elements by tag name using the getElementsByTagName() method. This method also returns an array-like object of all child elements with the given tag name.

```
demo.html
1    <html>
2    <body>
3     <p>This is a paragraph of text 1.</p>
4    <p>This is a paragraph of text 2.</p>
5
6     <script type="text/javascript">
7             //Single Element Color Change
8             document.getElementsByTagName('p')[0].style.backgroundColor='red';
9             //All Element Color Change
10            var matches = document.getElementsByTagName("p");
11             for(var elem in matches) {
12                matches[elem].style.background = "yellow";
13             }
14
15        </script>
16    </body>
17    </html>
```

demo.html   ×   +

← → C ⌂   ⓘ File | D:/Delete/test/demo.html

This is a paragraph of text 1.

This is a paragraph of text 2.

innerHTML

- The **innerHTML** property can be used to write the dynamic html on the html document.

- It is used mostly in the web pages to generate the dynamic html such as registration form, comment form, links etc.

-

Load Data

```
demo.html
1   <html>
2   <body>
3       <div id='data'></div>
4     <input type="button" value="Click Me" onclick="showdata()">
5
6   <script type="text/javascript">
7       function showdata() {
8           document.getElementById('data').innerHTML= "Data Loaded";
9       }
10  </script>
11
12  </body>
13  </html>
```
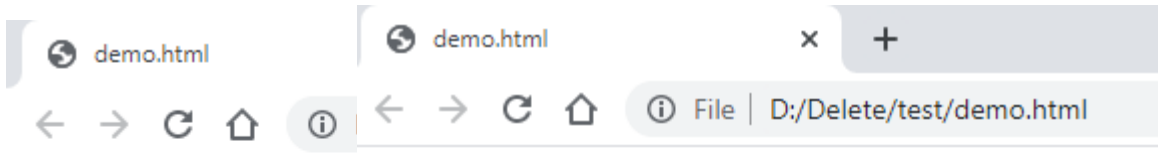
```html
<html>
<body>
   <div id='data'></div>
  <input type="button" value="Click Me" onclick="showdata()">

<script type="text/javascript">
   function showdata() {
       document.getElementById('data').innerHTML= "Data Loaded";
   }
</script>

</body>
</html>
```

demo.html

Click Me

demo.html   ×   +

File | D:/Delete/test/demo.html

Data Loaded
Click Me

Show Hide Form

```
demo.html
1  <html>
2  <body>
3  <script type="text/javascript">
4      var flag=true;
5
6      function showdata() {
7          var cform="<form action='Comment'>Enter Name:<br><input type='text' name='name'/>  ";
8          document.getElementById('data').innerHTML= "Data Loaded";
9          if(flag){
0              document.getElementById("data").innerHTML=cform;
1              flag=false;
2          }else{
3              document.getElementById("data").innerHTML="";
4              flag=true;
5          }
6      }
7  </script>
8      <div id='data'></div>
9      <input type="button" value="Click Me" onclick="showdata()">
0
1  </body>
2  </html>
```
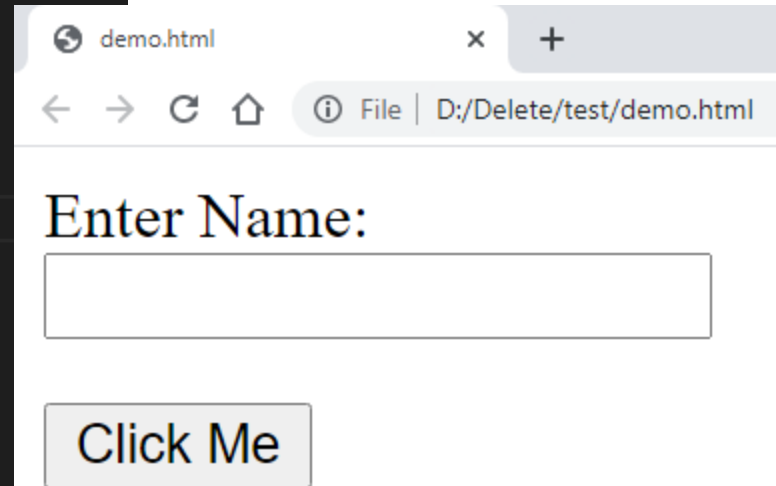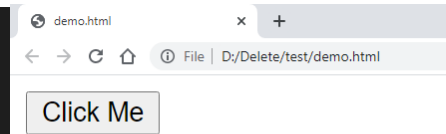
demo.html × +
← → C ⌂ ⓘ File | D:/Delete/test/demo.html

Click Me

demo.html × +
← → C ⌂ ⓘ File | D:/Delete/test/demo.html

Enter Name:

Click Me

innerText

- The **innerText** property can be used to write the dynamic text on the html document. Here, text will not be interpreted as html text but a normal text.

- It is used mostly in the web pages to generate the dynamic content such as writing the validation message, password strength etc.

Example

```html
<html>
<body>
<script type="text/javascript" >
    function validate() {
    var msg;
    if(document.myForm.userPass.value.length>5){
        msg="Great";
    }
    else{
        msg="Sorry";
    }

        document.getElementById('mylocation').innerText=msg;
    }

</script>
<form name="myForm">
<input type="password" value="" name="userPass" onkeyup="validate()">
Strength:<span id="mylocation">no strength</span>
</form>
</body>
</html>
```

```
demo.html
1   <html>
2   <body>
3   <script type="text/javascript" >
4       function validate() {
5       var msg;
6       if(document.myForm.userPass.value.length>5){
7           msg="Great";
8       }
9       else{
10          msg="Sorry";
11      }
12          document.getElementById('mylocation').innerText=msg;
13      }
14
15  </script>
16  <form name="myForm">
17  <input type="password" value="" name="userPass" onkeyup="validate()">
18  Strength:<span id="mylocation">no strength</span>
19  </form>
20  </body>
21  </html>
```

Strength:no strength

Strength:Sorry

Strength:Great

JavaScript DOM Styling

JS

Styling DOM Elements in JavaScript

- You can also apply style on HTML elements to change the visual presentation of HTML documents dynamically using JavaScript.

- You can set almost all the styles for the elements like, fonts, colors, margins, borders, background images, text alignment, width and height, position, and so on.

Example

```
demo.html
1   <html>
2   <body>
3    <p id="test">This is a paragraph of text .</p>
4   <script type="text/javascript">
5       // Selecting element
6       var elem = document.getElementById("test");
7       // Appling styles on element
8       elem.style.color = "blue";
9       elem.style.fontSize = "18px";
10      elem.style.fontWeight = "bold";
11  </script>
12  </body>
13  </html>
```

```
<html>
<body>
 <p id="test">This is a paragraph of text .</p>
<script type="text/javascript">
    // Selecting element
    var elem = document.getElementById("test");
    // Appling styles on element
    elem.style.color = "blue";
    elem.style.fontSize = "18px";
    elem.style.fontWeight = "bold";
</script>
</body>
</html>
```

demo.html   ×   +

← → C ⌂   ⓘ File | D:/Delete/test/demo.html

This is a paragraph of text .

Naming Conventions of CSS Properties in JavaScript

- Many CSS properties, such as font-size, background-image, text-decoration, etc. contain hyphens (-) in their names. Since, in JavaScript hyphen is a reserved operator and it is interpreted as a minus sign, so it is not possible to write an expression, like: elem.style.font-size

- Therefore, in JavaScript, the CSS property names that contain one or more hyphens are converted to intercapitalized style word. It is done by removing the hyphens and capitalizing the letter immediately following each hyphen, thus the CSS property font-size becomes the DOM property fontSize, border-left-style becomes borderLeftStyle, and so on.

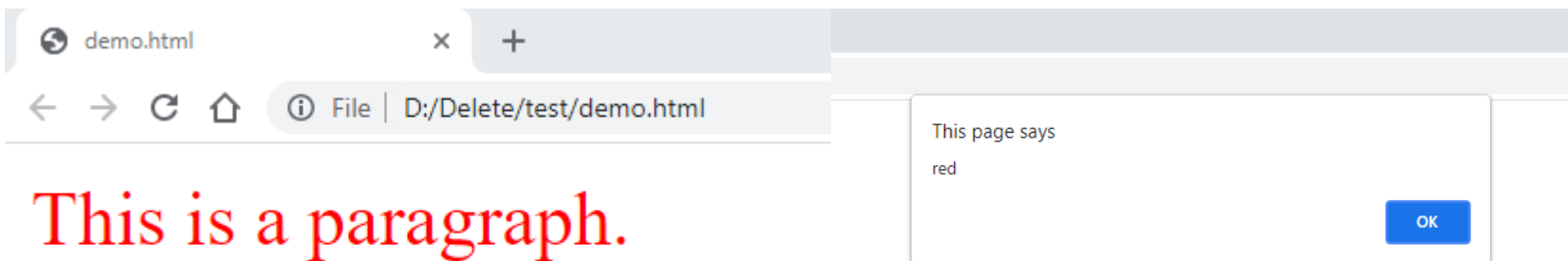# Getting Style Information from Elements

- you get the styles applied on the HTML elements using the style property.

```
demo.html
1    <html>
2    <body>
3       <p id="test" style="color:red; font-size:20px;">This is a paragraph.</p>
4    <script type="text/javascript">
5        var elem = document.getElementById("test");
6        // Getting style information from element
7        alert(elem.style.color);   // Outputs: red
8        alert(elem.style.fontSize);   // Outputs: 20px
9        alert(elem.style.fontStyle);   // Outputs nothing
10   </script>
11   </body>
12   </html>
```

```html
<html>
<body>
 <p id="test" style="color:red; font-size:20px;">This is a paragraph.</p>
<script type="text/javascript">
    var elem = document.getElementById("test");
    // Getting style information from element
    alert(elem.style.color);  // Outputs: red
    alert(elem.style.fontSize);  // Outputs: 20px
    alert(elem.style.fontStyle);  // Outputs nothing
</script>
</body>
</html>
```

demo.html   ×   +

← → C ⌂   ⓘ File | D:/Delete/test/demo.html

This is a paragraph.

This page says
red

OK

Get Exclusive
Video Tutorials

www.aptutorials.com

https://www.youtube.com/user/Akashtips

Get More Details

www.akashsir.com

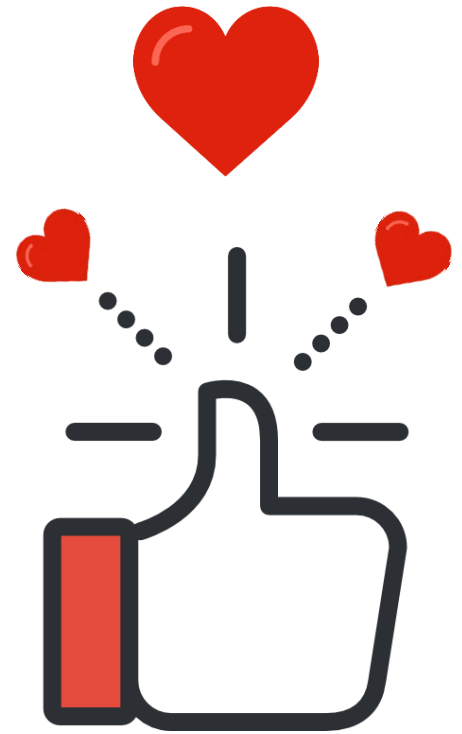# If You Liked It !
Rating Us Now

Just Dial
https://www.justdial.com/Ahmedabad/Akash-Technolabs-Navrangpura-Bus-Stop-Navrangpura/079PXX79-XX79-170615221520-S5C4_BZDET

Sulekha
https://www.sulekha.com/akash-technolabs-navrangpura-ahmedabad-contact-address/ahmedabad

Connect With Me

Akash Padhiyar
#AkashSir

www.akashsir.com
www.akashtechnolabs.com
www.akashpadhiyar.com
www.aptutorials.com

Social Info

Akash.padhiyar

Akashpadhiyar

Akash_padhiyar

+91 99786-21654

#Akashpadhiyar
#aptutorials