**Contents**

1.plug-in

2.video

3.audio

4.web Storage

5.canvas

5.1.chartJS

6. Geo Location API

7. Google API

8. Semantic Tags

9. Web workers

10. Index DB -------------> Pending.

11. Drag & Drop

12. WebForm API

**1.plug-in:-** The plug-In is small computer program that extends the functionality of web broser. The plug-in can be added to webpage with <object> or<embed> tag.Upto html4, The browser depends on flash player(plugin)to load/run the videos and audios files.But html5 onwards, video and audio are played with out depending on external plug-in(flash player). Because html5 provided its own player(HTML5 video player/audio player) to load and run audio and video.

**2.Video**: The video tag embed the video in webpage. The video can be played using html5 video player.

Syntax:

<video attr=value,height=’v1’ width=’v2’>

<source src=” “ type=’video/format’>

<source src=” “ type=’video/format’>

…etc.

[<track attr[s]=’v1’>

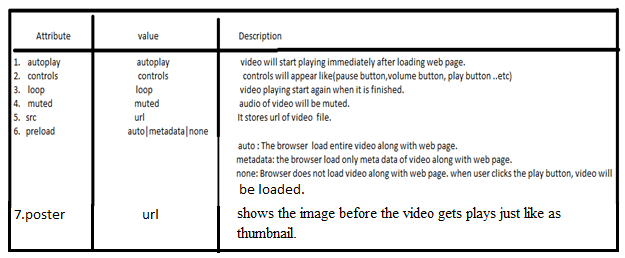
<track attr[s]=’v1’>

…etc. ]

</video>

(or)

<video attr=’v1’></video>



The browser loads first one(video file) which is specified in first source-tag and if it works then it will make playing it. If first one is not available or browser does not support first one file format, browser go to next one.

Example:

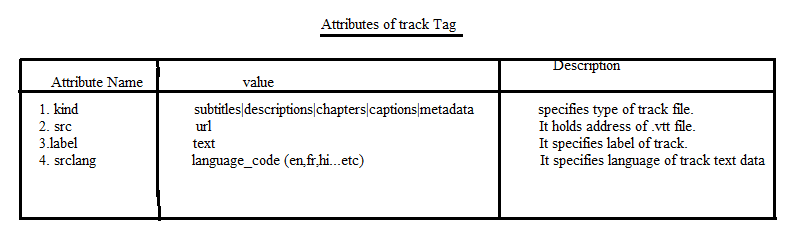
<video >

<source src=’a.mp4’ type=’video/mp4’>

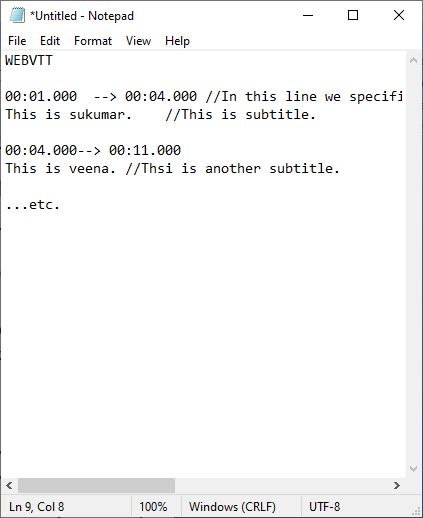
<source src=’a.ogv’ type=’video/ogv’>

</video>

Track:- The track element is used to specify “**text track”**for audio and video elements. The track is formatted in (webvtt format).vtt files. These files may be subtitles files , caption files,chapter file, or description file. The text which is inside the file will appear when media is playing.



Example to Trackfile: abc.vtt



**3.audio:**The audio tag embed the audioin webpage. The audio can be played using plug-in(flash player).

Syntax:

<audio attr=value,height=’v1’ width=’v2’>

<source src=” “ type=’video/format’>

<source src=” “ type=’video/format’>

…etc.

[<track attr[s]=’v1’>

<track attr[s]=’v2’>

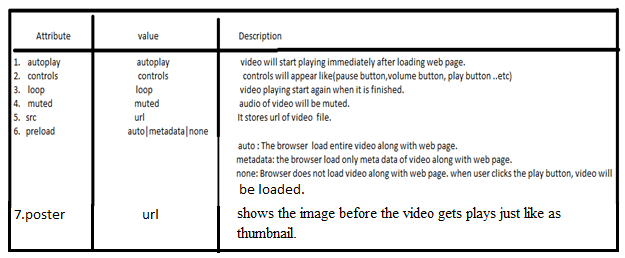
…etc.]

</audio>

(or)

<audio attr[s]=’v1’ ></audio>





Example:

<!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>

       This Page have  videos.

    </h2>

       <video  controlsheight="250px"width="300px"poster="poster.jpg">

           <sourcesrc="video2.mp4"type="video/mp4"></source>

            <sourcesrc="video1.webm"type="video/mp4"></source>

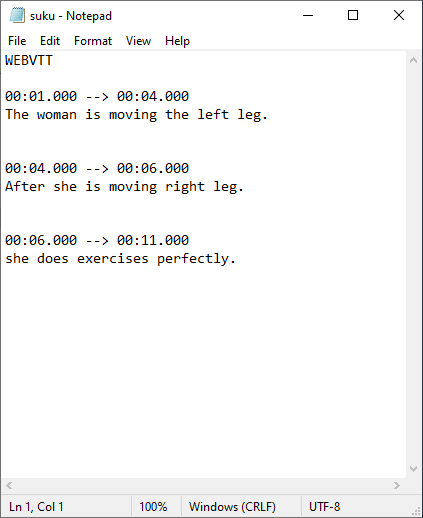
            <tracksrc="suku.vtt"kind="subtitles"label="First\_one"srclang="en"  />

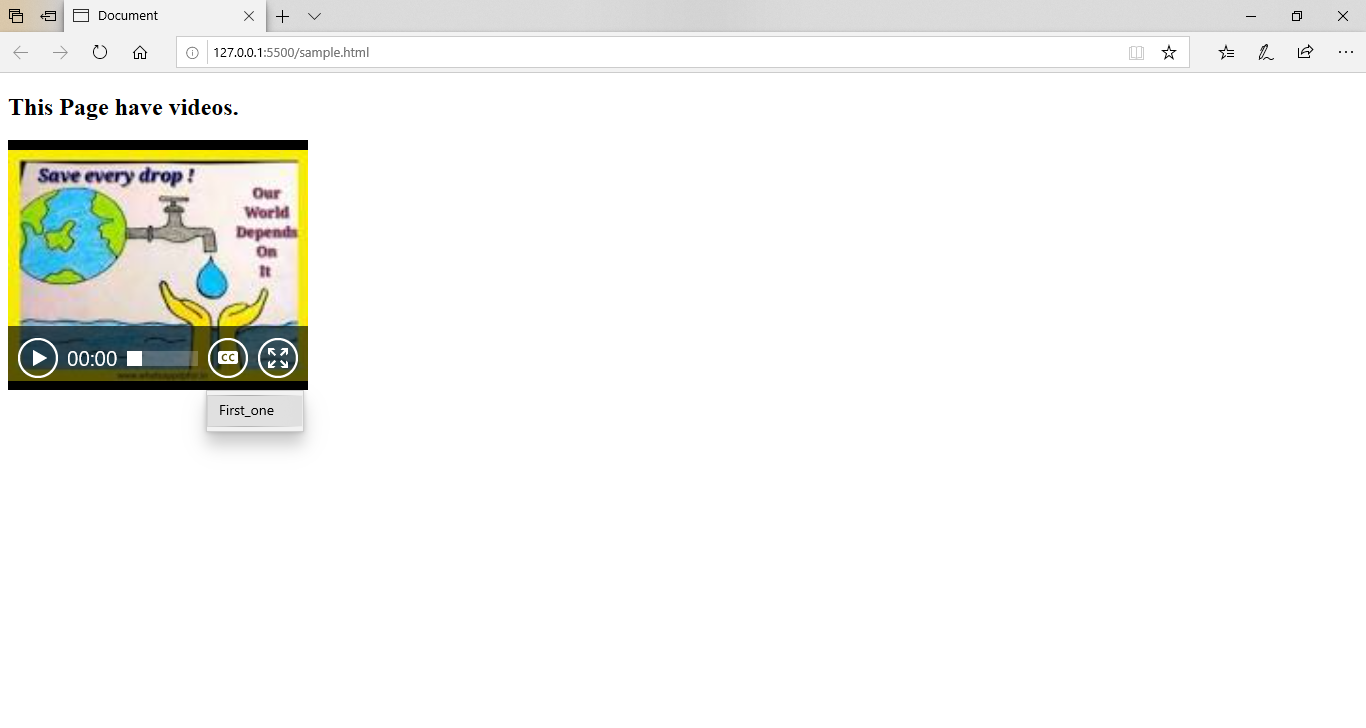
       </video>

</body>

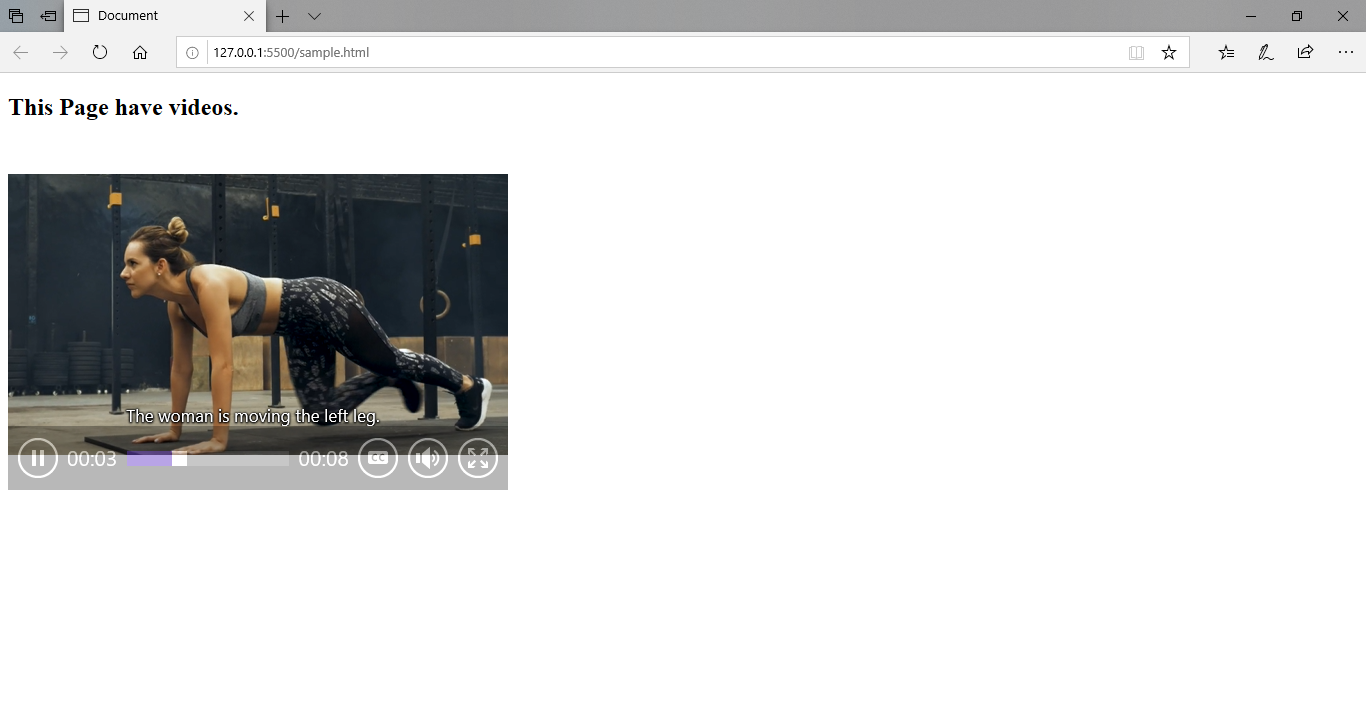
</html>

FileName:suku.vtt.

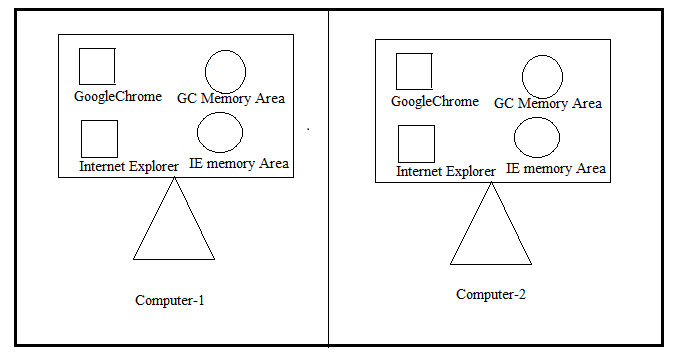




Click on first-one and click the play button. After clicking,



**4.Web Storage:-**

****

Every Browser has individual private Memory area in same computer. The web storage exists in Browser memory area. we store information locally on user’s computer browser memory. If we open the same website on another machine, then in that machine the previous information is not available.This information is not sent to server by itself. The web storage allows us to storeupto 10MB data.The data is in the form of key/value(or)item/value pairs. Simply it is called as client side database.There are two types of web storages.

**4.1. Session Storage**:- The session storage uses **sessionStorage object** to store data on temporary basis for single window/tab or browser session. This data is automatically deleted when user closes browser window or tab.

RealTimeUsage: In Banks, The employee(cashier,manager) credentials are usually stored in session storage instead of local storage because of security reasons.

**4.2.Local Storage**:- The local storage uses localStorage object to store data permanintely for our entire website. The data is avilable , even if user close browser/tab. The data is in local machine until we remove it.

The Session Storage and LocalStorage have following common methods and properties.

1. setItem(key,value):- It stores key/value pair in webstorage.
2. getItem(key):- It returns value of corresponding key from webstorage.
3. removeItem(key):- It removes value from webstorage,if key exists in webstorage.
4. Clear():- It removes the all key/value pairs from web storage.
5. Length:- It is a property. It returns no.of items which is in storage.

Note: The local storage object has provided a one event which is **‘storage’.** This event is fired when object is changed.

**4.3.RealTime Usage of SessionStorage**:

**1**.A shopping cart allows a website’s visitor to view product pages and add items to their basket. The visitor can review all of their items and update their basket (such as to add or remove items). To achieve this, the website needs to store the visitor’s data and pass them from one page to another, until the visitor goes to the checkout page and makes a purchase. Storing data can be done via a server-side language or a client-side one. With a server-side language, the server bears the weight of the data storage, whereas with a client-side language, the visitor’s computer (desktop, tablet or smartphone) stores and processes the data. JS uses web storage api to store and process data at client side.

 If you open multiple tabs on a web app, session storage syncs the data between all opened tabs. If you add an item on one tab, you will sync with the others.

**RealTime Usage of Local storage:** Web storage is used to store user preferences(like color,font-size,..etc)of elements on web page and user setting like username and password.

**Example:1**

<!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>

    <style>

        h2{

            text-align: center;

            color:brown;

            text-decoration: underline;

        }

        div:nth-child(2){

            height:20vh;

            width:50vw;

            border: 0.5pxsolidblack;

            margin:10px0px;

            position: relative;

            left:25vw;

            padding:10px;

            background-color: bisque;

            box-sizing: border-box;

        }

        div:nth-child(2)>input{

            margin-top: 10px;

            border:1pxsolidbrown;

        }

        div:nth-child(2)>button{

            margin:5px5px;

            height: 5vh;

            width:10vw;

            background-color: blueviolet;

            color:aliceblue;

        }

        div:nth-child(3)

        {

            width:50vw;

            position: relative;

            left:25vw;

            border: 1pxsolidred;

            background-color: aqua;

            box-sizing: border-box;

        }

        div:nth-child(3)>h5

        {

            text-align: center;

            text-decoration: underline;

            font-size: 1.2em;

        }

        div:nth-child(3)>table

        {

            position:relative;

            left:10vw;

            width:30vw

        }

    </style>

</head>

<bodybgcolor='green'>

    <h2>

       Shopping Cart

    </h2>

    <div>

    Key:<inputtype="text"><br>

    Value:<inputtype="text"><br>

    <buttononclick="btnEvent(event)">set</button>

    <buttononclick="btnEvent(event)">Update</button>

    <buttononclick="btnEvent(event)">Delete</button>

    <buttononclick="btnEvent(event)">Clear</button>

    </div>

    <div>

        <h5>Session Storage</h5>

        <tableborder="1">

            <thead>

            <tr>

                <th>Key</th>

                <th>Value</th>

            </tr>

            </thead>

            <tbody>

            </tbody>

        </table>

    </div>

   <script>

    classwStorage{

        constructor(){}

        put(a,b){

          localStorage.setItem(a,b);

          fwKeys.push(a);

        }

        del(a){

          localStorage.removeItem(a);

          fwKeys.splice(fwKeys.indexOf(a),1);

        }

        uPut(a,b){

            localStorage.removeItem(a);

            localStorage.setItem(a,b);

        }

        cAll()

        {

            localStorage.clear();

            fwKeys=[];

        }

        display()

        {

            lett1=document.querySelector('tbody');

            while(t1.firstChild)

            {

                t1.removeChild(t1.firstChild);

            }

            if(fwKeys.length>0){

            for(letioffwKeys)

            {

                letd1=document.createElement('tr');

                letd2=document.createElement('td');

                letd3=document.createElement('td');

                d2.innerHTML=i;

                d3.innerHTML=localStorage.getItem(i);

                d1.appendChild(d2);

                d1.appendChild(d3);

                t1.appendChild(d1);

            }

        }

        }

    }

    letfWStorage=newwStorage();

    letfwKeys=[];

    functionbtnEvent(event){

        letiEle=document.querySelectorAll('input');

        letk1=iEle[0].value;

        letv1=iEle[1].value;

        iEle[0].value='';

        iEle[1].value='';

        if(event.target.innerHTML=='set')

        {

          fWStorage.put(k1,v1);

        }

        elseif(event.target.innerHTML=='Update')

        {

            fWStorage.uPut(k1,v1)

        }

        elseif(event.target.innerHTML=='Delete')

        {

            fWStorage.del(k1);

        }

        else

        {

            fWStorage.cAll();

        }

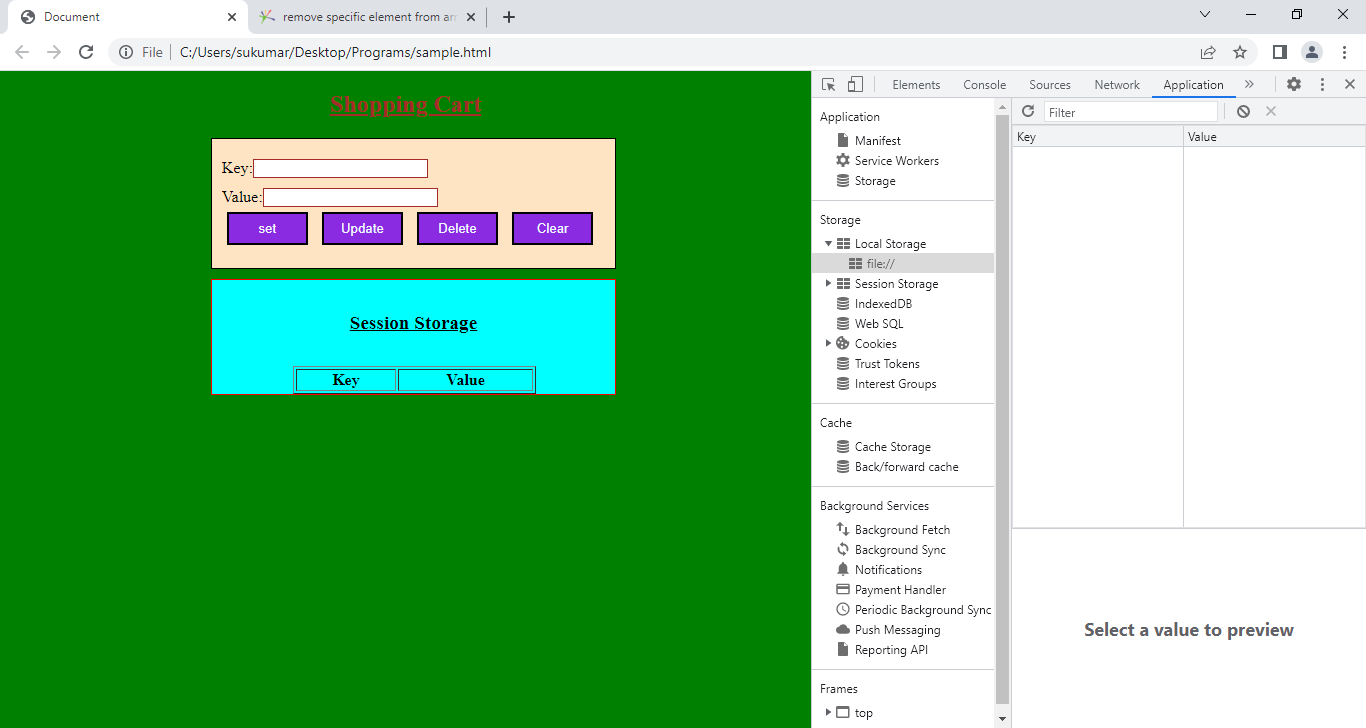
        fWStorage.display();

    }

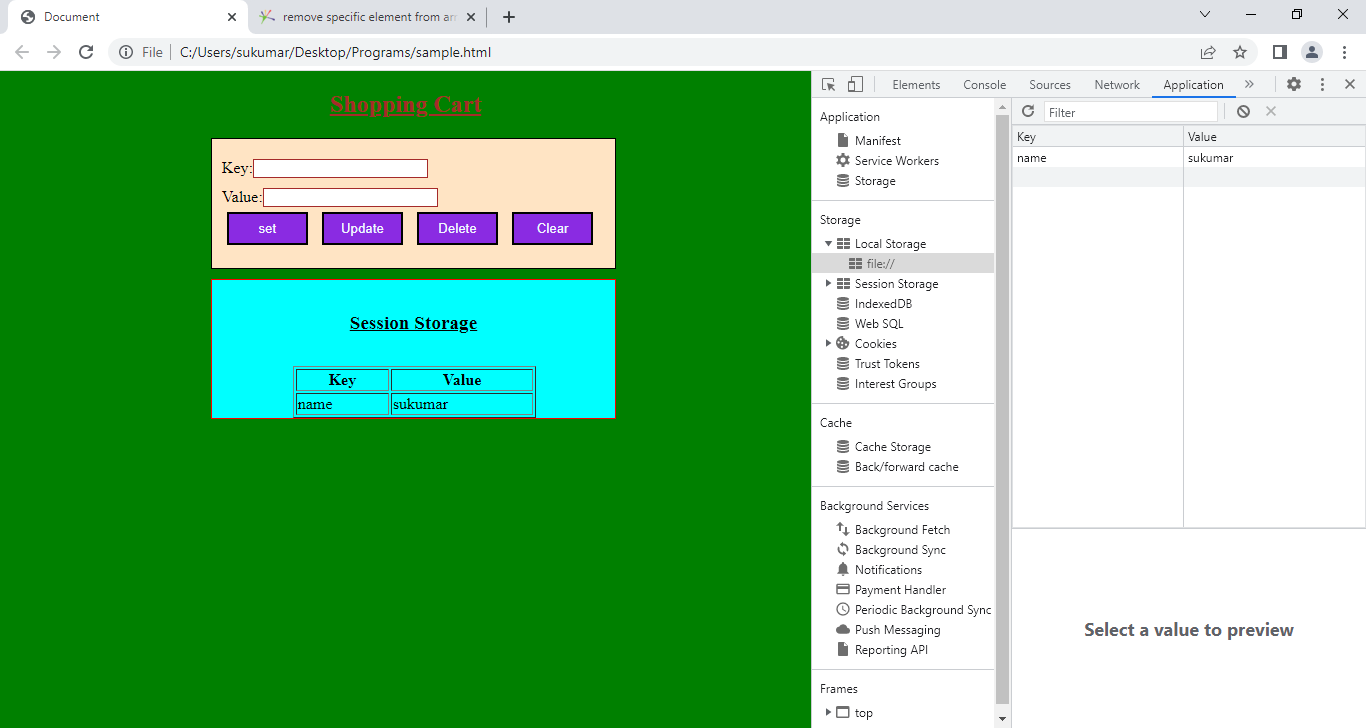
   </script>

</body>

</html>



Enter key:name ,value:sukumar and press the set button.



Like do all other operations.

Example:2 setting User Preferences.

<!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>

<bodyonload="xyz()">

  <h2>User Preferences.</h2>

  <div>

  <form>

    Give BackGroundColor:<inputtype="text">

    <buttononClick="abc()">Set</button>

  </form>

 </div>

 <script>

  functionabc(){

  vard=localStorage.setItem('bgcolor',document.querySelector("input").value);

  }

  functionxyz()

  {

    varx=localStorage.getItem('bgcolor');

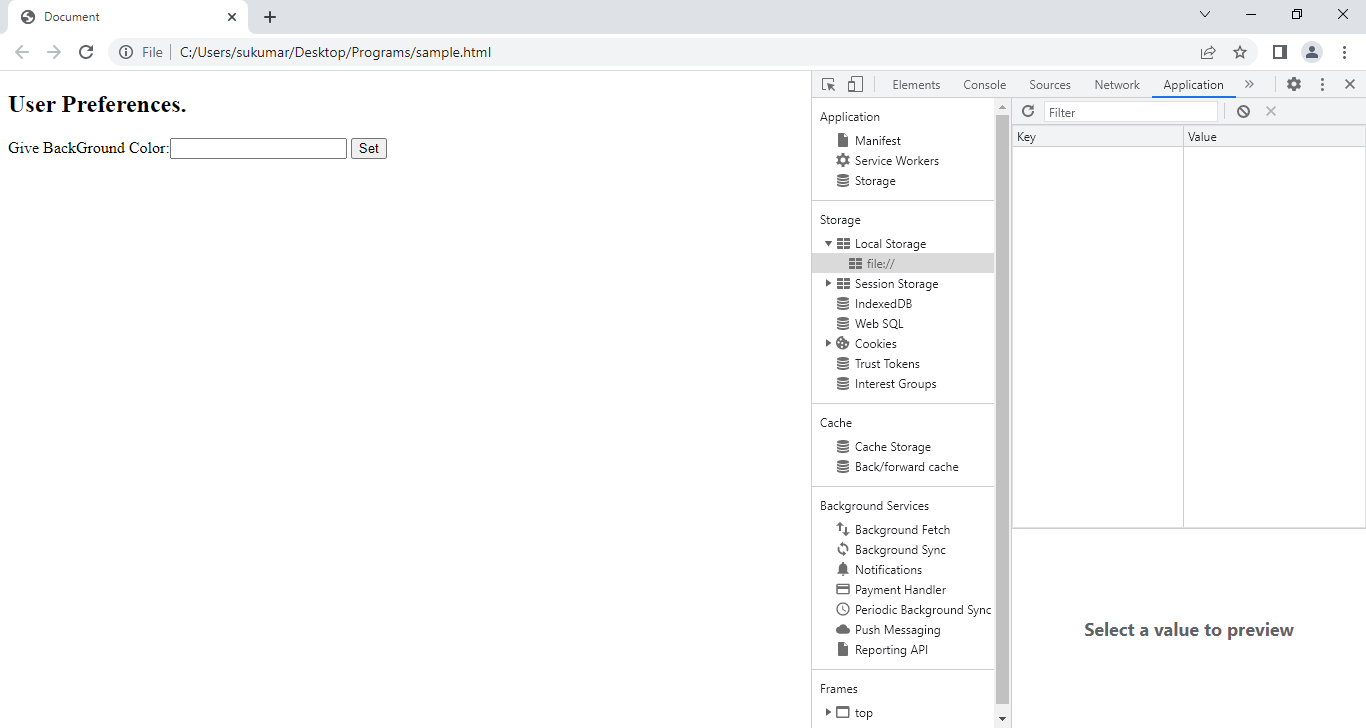
    document.bgColor=x;

  }

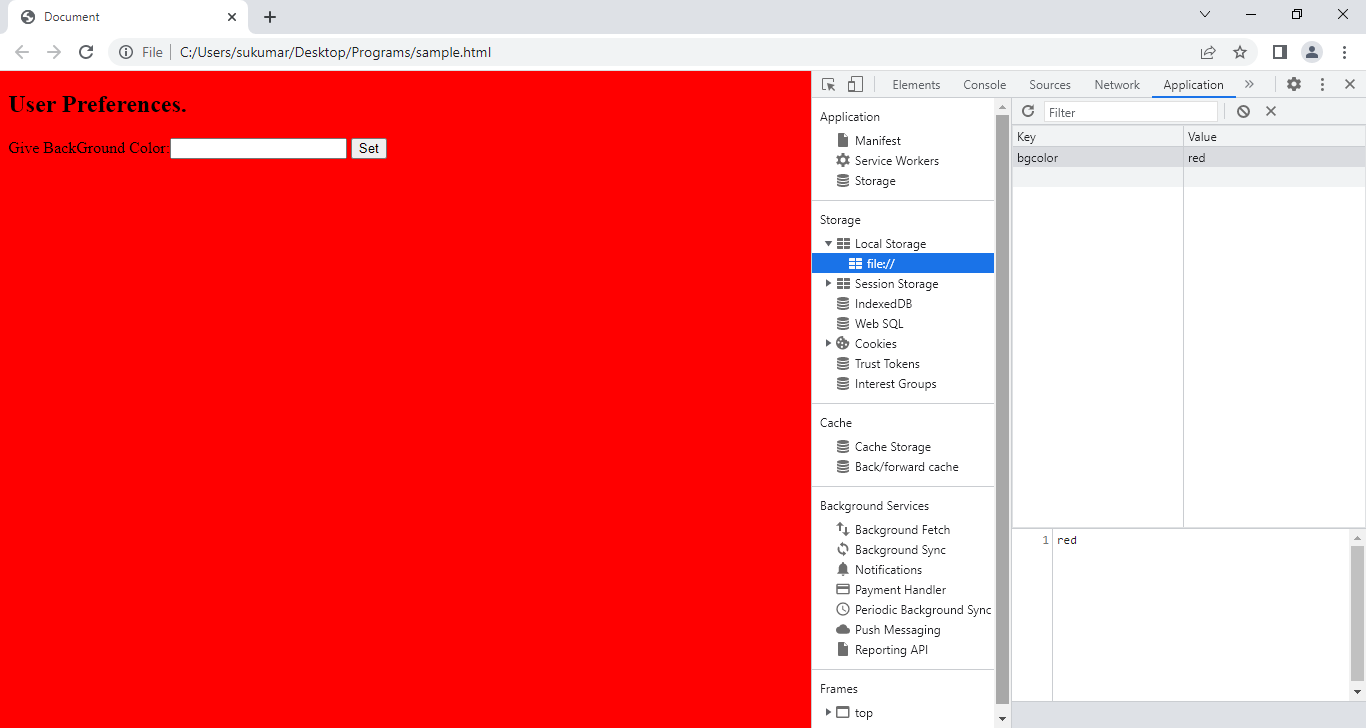
 </script>

</body>

</html>



I entered red in text box . I pressed entered button and I closed the browser. When I reopened the same webpage, It’s background will be in red color.



**5.CANVAS**:- The canvas is region/place on web page. In which , we draw graphics on fly(dynamically/statically)via JS. By default, It does not has border and content.

Syntax:

<canvas height=’ ‘ width=’ ‘> content<canvas>

5.1.Canvas API:- This API has several properties and methods.

a) **Line related Methods**:-

Path consists of single point or multiple points.

i.moveTo(x,y): This method creates **subpath** with given point.(or) It specifies the starting point of line.

ii.lineTo(x,y): This methos add given point in subpath .(or) It specifies the enidng point of line.

iii)stroke(): This method strokes the subpath with current **stroke style**.

iv)beginPath(): This method reset the current path.(i.e) considers top most left corner of canvas.

v)closePath(): this method marks the current path as closed.

vi)fill(): this method fills the subpath inside area with current **fontstyle**.

Example:1 Draw a Line

 <bodyonload = "rock()">

      <canvaswidth = "400"height = "400"style="border:1px solid green"></canvas>

      <scripttype = "text/javascript">

         functionrock(){

         tcs1=document.querySelector('canvas');

         tcs=tcs1.getContext('2d');

         tcs.beginPath();

         tcs.moveTo(200,200);

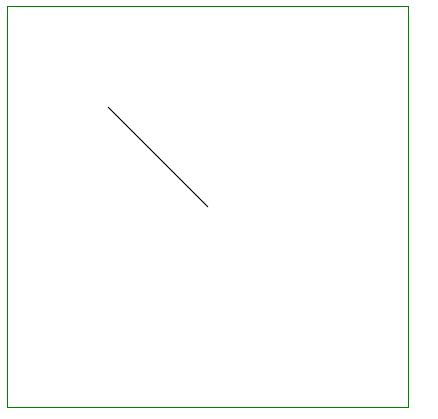
         tcs.lineTo(100,100);

         tcs.stroke();

         }

      </script>

    </body>



Example:2 Draw a Triangle

<bodyonload = "rock()">

      <canvaswidth = "400"height = "400"style="border:1px solid green"></canvas>

      <scripttype = "text/javascript">

         functionrock(){

         tcs1=document.querySelector('canvas');

         tcs=tcs1.getContext('2d');

         tcs.strokeStyle='red';

         tcs.beginPath();

         tcs.moveTo(200,200);

         tcs.lineTo(100,100);

         tcs.lineTo(100,200);

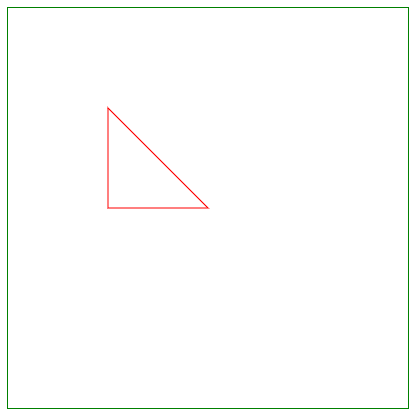
         tcs.closePath();

         tcs.stroke();

         }

      </script>

    </body>



Example:3 Draw Triangle and fill inside it with green color.

<bodyonload = "rock()">

      <canvaswidth = "400"height = "400"style="border:1px solid green"></canvas>

      <scripttype = "text/javascript">

         functionrock(){

         tcs1=document.querySelector('canvas');

         tcs=tcs1.getContext('2d');

         tcs.strokeStyle='red';

         tcs.fillStyle='blue';

         tcs.beginPath();

         tcs.moveTo(200,200);

         tcs.lineTo(100,100);

         tcs.lineTo(100,200);

         tcs.closePath();

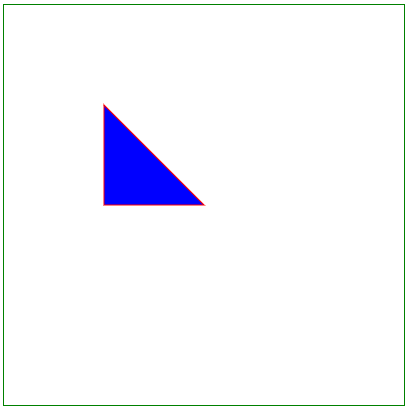
         tcs.fill();

         tcs.stroke();

         }

      </script>

    </body>



**b.Rectangle Related methods**:-

i)strokeRect(x,y,width,height): It draws border rectangle.

ii)fillRect(x,y,width,height): It draws rectangle. The rectangle inside area is filled with **fillStyle** .

iii)clearRect(x,y,width,height): It draws rectangle. Its inside area is fully transparent.

Example:1

<bodyonload = "rock()">

      <canvaswidth = "400"height = "400"style="border:1px solid green"></canvas>

      <scripttype = "text/javascript">

         functionrock(){

         tcs1=document.querySelector('canvas');

         tcs=tcs1.getContext('2d');

         tcs.strokeStyle='red';

         tcs.fillStyle='blue';

         tcs.beginPath();

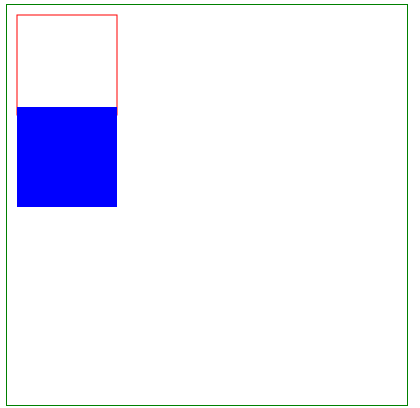
         tcs.strokeRect(10,10,100,100);

         tcs.fillRect(10,102,100,100);

         }

      </script>

    </body>



**c)Text & Font Related propertes and methods:**

**c.1)Properties& Methods:**

i. font:- It is used to set the fontsize.

Font=’italic/bold unitlength color’;

ii.textAlign: It is used to specify the text alignment. The values are left,right,center,start or end.

iii.fillText(text,x,y[,maxwidth]): It displays solid text on canvas from x,ycordinate position.

iv.strokeText(text,x,y[,maxwidth]): It displays line text on canvas from x,ycordinate position.

Example:

<bodyonload = "rock()">

      <canvaswidth = "400"height = "300"style="border:1px solid green"></canvas>

      <scripttype = "text/javascript">

         functionrock(){

         tcs1=document.querySelector('canvas');

         tcs=tcs1.getContext('2d');

         tcs.strokeStyle='red';

         tcs.fillStyle='blue';

         tcs.beginPath();

         tcs.font='italic 30px blue';

         tcs.textAlign='center';

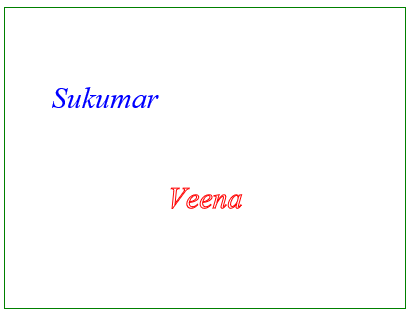
         tcs.fillText("Sukumar",100,100);

         tcs.strokeText("Veena",200,200);

         }

      </script>

    </body>



**d)FillStyle&StrokeStyle:**

The fillStyle property respresents color to use inside a shapes.

Example: fillStyle=’red’

The StrokeStyle represents color to user for lines around the shapes.

Example:strokeStyle=’blue’.

**e)pattern:**

**syntax:**

createPattern(image,repitition);

This method will use image to create the pattern. The second argument could be a string with one of the following values: repeat, repeat-x, repeaty, andno-repeat. If the empty string or null is specified, repeat will. be assumed.

Example:

<bodyonload = "rock()">

      <canvaswidth = "400"height = "300"style="border:1px solid green"></canvas>

      <scripttype = "text/javascript">

         functionrock(){

         tcs1=document.querySelector('canvas');

         tcs=tcs1.getContext('2d');

         vari=newImage();

         i.src='hrbottle.png';

         i.onload=function(){

         letp=tcs.createPattern(i,'repeat');

         tcs.fillStyle=p;

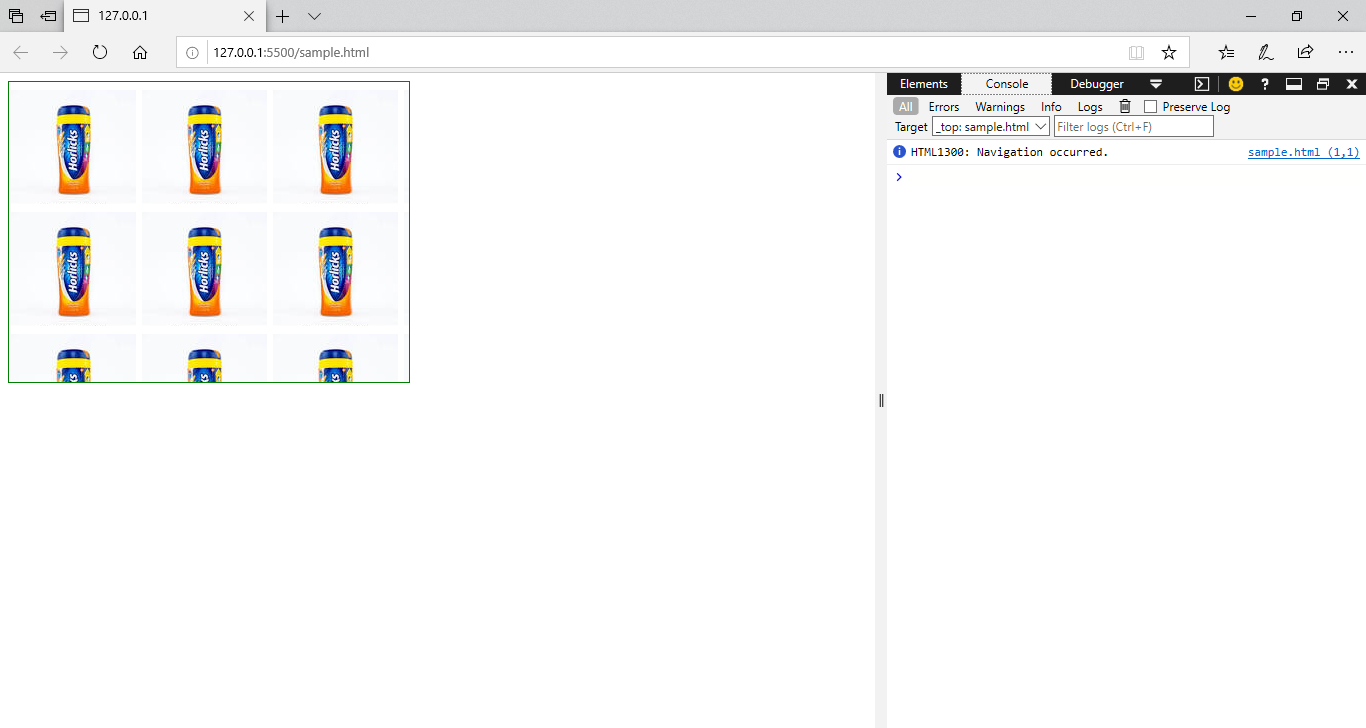
         tcs.fillRect(0,0,400,300);

         }

         }

      </script>

    </body>



1. **Shadow Properties:**

a.shadowColor.

b.shadowOffsetX

c.shadowOffsetY

d.shadowBlur

Syntax:

Propertyname=value;

Example:

<bodyonload = "rock()">

      <canvaswidth = "400"height = "300"style="border:1px solid green"></canvas>

      <scripttype = "text/javascript">

         functionrock(){

         tcs1=document.querySelector('canvas');

         tcs=tcs1.getContext('2d');

         tcs.font='bold 30px Times new Roman';

         tcs.shadowOffsetX=3;

         tcs.shadowOffsetY=3;

         tcs.shadowColor='red';

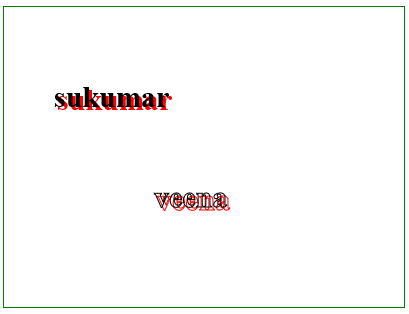
         tcs.fillText('sukumar',50,100);

         tcs.strokeText('veena',150,200);

         }

      </script>

    </body>

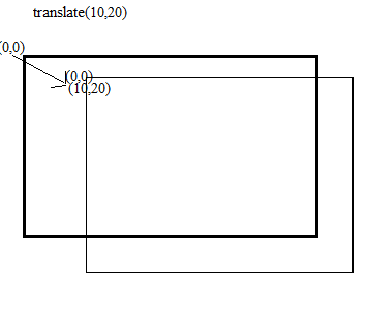


1. Translate: This method remaps the (0,0) position to specified position on canvas.

Syntax:

Translate(x,y)

x and y are new origion position coordinates.



1. Rotate:- This method rotates the currently drawn graphics.

Syntax:

Rotate(degreen in radian)

To calculate from degrees to radians: radians\*Math.PI/180

Example: to rotate 5 degrees : 5\*Math.PI/180.

1. Scale:- This method scales the currently drawn graphics either bigger or smaller.

Syntax:

Scale(x,y)

Where x and y values are

1=100%; 2=200%,0.5=50%,…etc.

Example:

<bodyonload = "rock()">

      <canvaswidth = "400"height = "300"style="border:1px solid green"></canvas>

      <scripttype = "text/javascript">

         functionrock(){

         tcs1=document.querySelector('canvas');

         tcs=tcs1.getContext('2d');

         tcs.translate(5,5);

         tcs.fillRect(50,50,100,100);

         tcs.rotate(5\*Math.PI/180);

         tcs.font="bold 25px Times New Roman";

         tcs.fillText("sukumar",200,200);

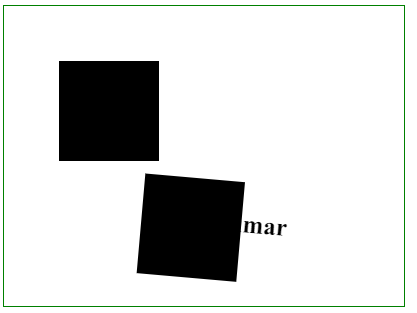
         tcs.scale(2,2);

         tcs.fillRect(75,75,50,50)

         }

      </script>

    </body>



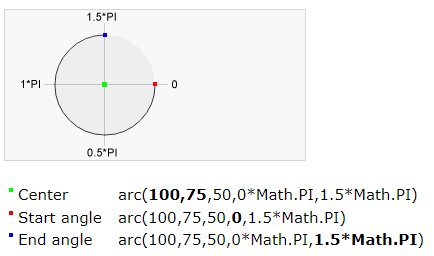
j.arc:- It is used to display arc/complete circle on canvas.

Syntax:

arc(x, y, radius, startAngle, endAngle, anticlockwise)

The angle is specified in **radians**.

X,y are co-ordinates of center point.



Example:

 <bodyonload = "rock()">

      <canvaswidth = "400"height = "300"style="border:1px solid green"></canvas>

      <scripttype = "text/javascript">

         functionrock(){

         tcs1=document.querySelector('canvas');

         tcs=tcs1.getContext('2d');

         tcs.strokeStyle='red';

         tcs.arc(100,100,50,0,Math.PI\*2);

         tcs.stroke();

         tcs.closePath();

         tcs.beginPath();

         tcs.strokeStyle='blue';

         tcs.arc(200,200,30,0,Math.PI\*1.5);

         tcs.stroke();

         tcs.closePath();

         tcs.beginPath();

         tcs.strokeStyle='blue';

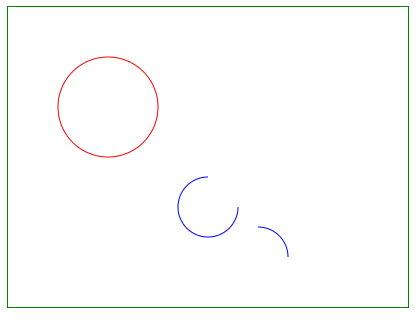
         tcs.arc(250,250,30,0,Math.PI\*1.5,true);

         tcs.stroke();

         }

      </script>

</body>



**RealTime Example on canvas Tag.**

<!DOCTYPEHTML>

<html>

   <head>

      <style>

         div{

            width:500px;

            height:500px;

            border:0.5pxsolidblack;

            margin:100px50px;

            float:left;

         }

         ul{

            list-style-type: none;

         }

         ul>li{

            margin:15px5px;

         }

         canvas{

            height:100%;

            width:100%;

         }

      </style>

   </head>

   <body>

      <div><h4style="text-align:center">StudentForm</h4>

         <hr>

         <ul>

            <li>StudentName:<inputtype="text"></li>

            <li>Branch     :<inputtype="text"></li>

            <li>Telugu     :<inputtype="number"max=100></li>

            <li>maths      :<inputtype="number"max=100></li>

            <li>Science    :<inputtype="number"max=100></li>

            <li>Social     :<inputtype="number"max=100></li>

            <li><buttononclick="drawGraphic()">Submit</button></li>

         </ul>

         <hr>

      </div>

      <div>

         <canvasstyle="border:0.5px solid black;"></canvas>

      </div>

      <script>

         functiondrawGraphic()

         {

           variEle=document.querySelectorAll('input');

           varc1=document.querySelector('canvas');

           varc=c1.getContext('2d');

           c.lineWidth=0.5;

           c.strokeStyle='blue';

           c.beginPath();

           c.strokeText('BarChart',100,10);

           c.closePath();

           c.stroke();

           c.beginPath();

           c.moveTo(50,20);

           c.lineTo(50,100 );

           c.lineTo(250,100);

           c.stroke();

           c.closePath();

           c.beginPath();

           c.font="italic 10px Times new Roman";

           c.fillText("100",30,23);

           c.fillText("050",30,43);

           c.fillText("025",30,70);

           c.fillText("000",30,100);

           c.stroke();

           c.closePath();

           c.beginPath();

           c.font="italic 7px Times new Roman";

           c.fillText("Telugu",55,115);

           c.fillText("Maths",100,115);

           c.fillText("Social",150,115);

           c.fillText("Sciece",200,115);

           c.closePath();

           c.beginPath();

           c.fillStyle='#eb9743';

           c.fillRect(55,100,20,-iEle[2].value);

           c.fillStyle='#bccd7a';

           c.fillRect(100,100,20,-iEle[3].value);

           c.fillStyle='#67b6c7';

           c.fillRect(150,100,20,-iEle[4].value);

           c.fillStyle="#a55ca5";

           c.fillRect(200,100,20,-iEle[5].value);

           c.stroke();

           c.closePath();

           for(iofiEle){

            i.value="";

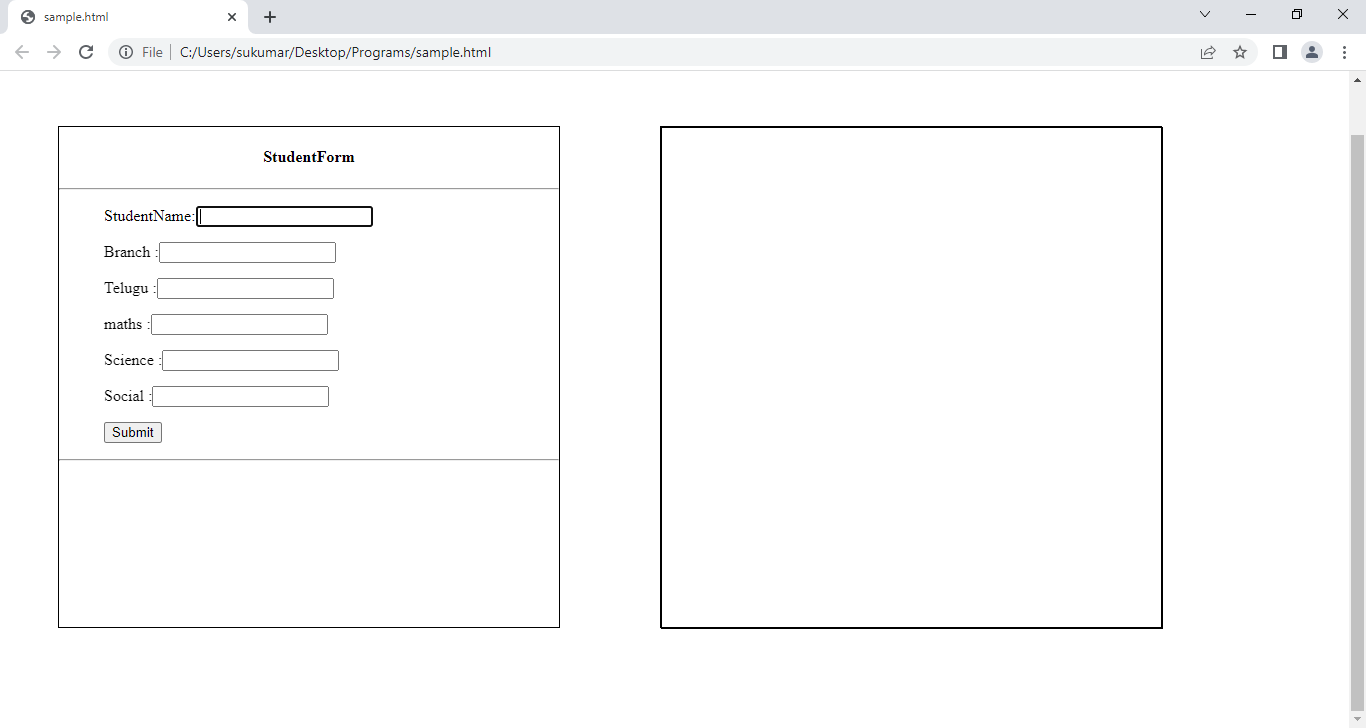
           }

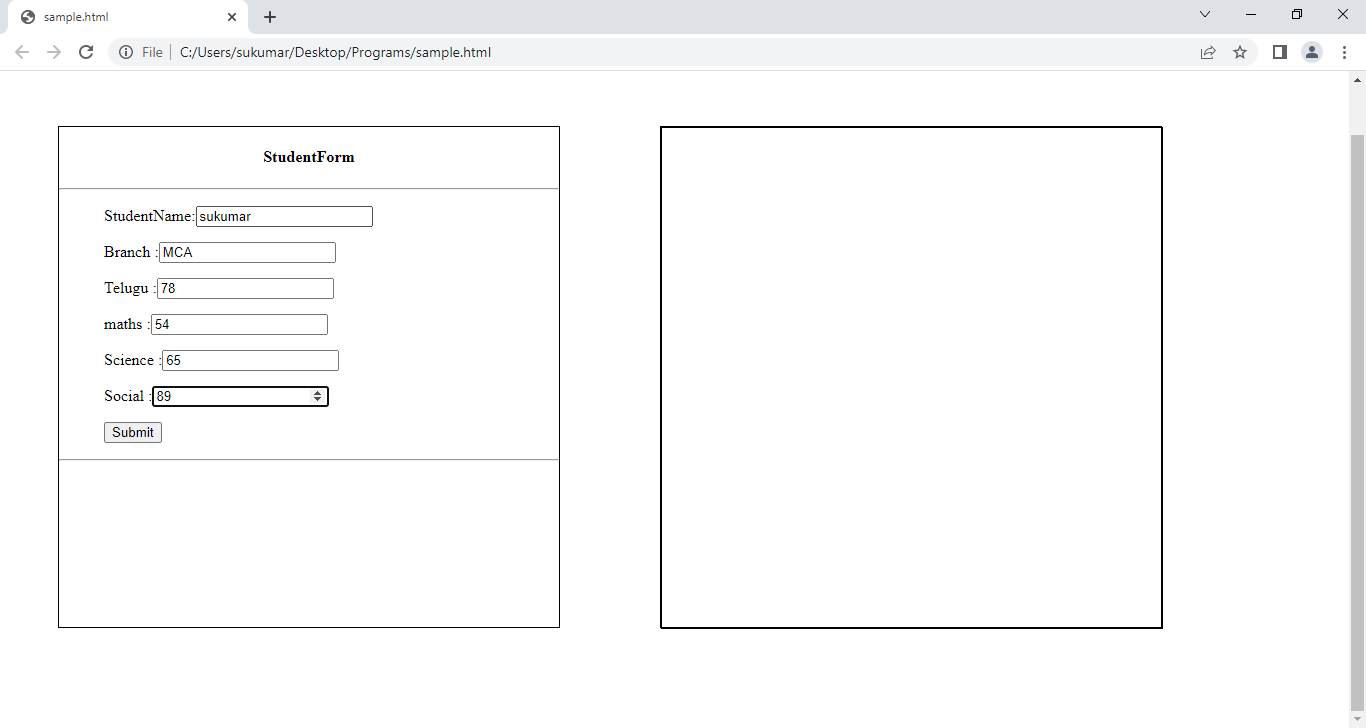
         }

      </script>

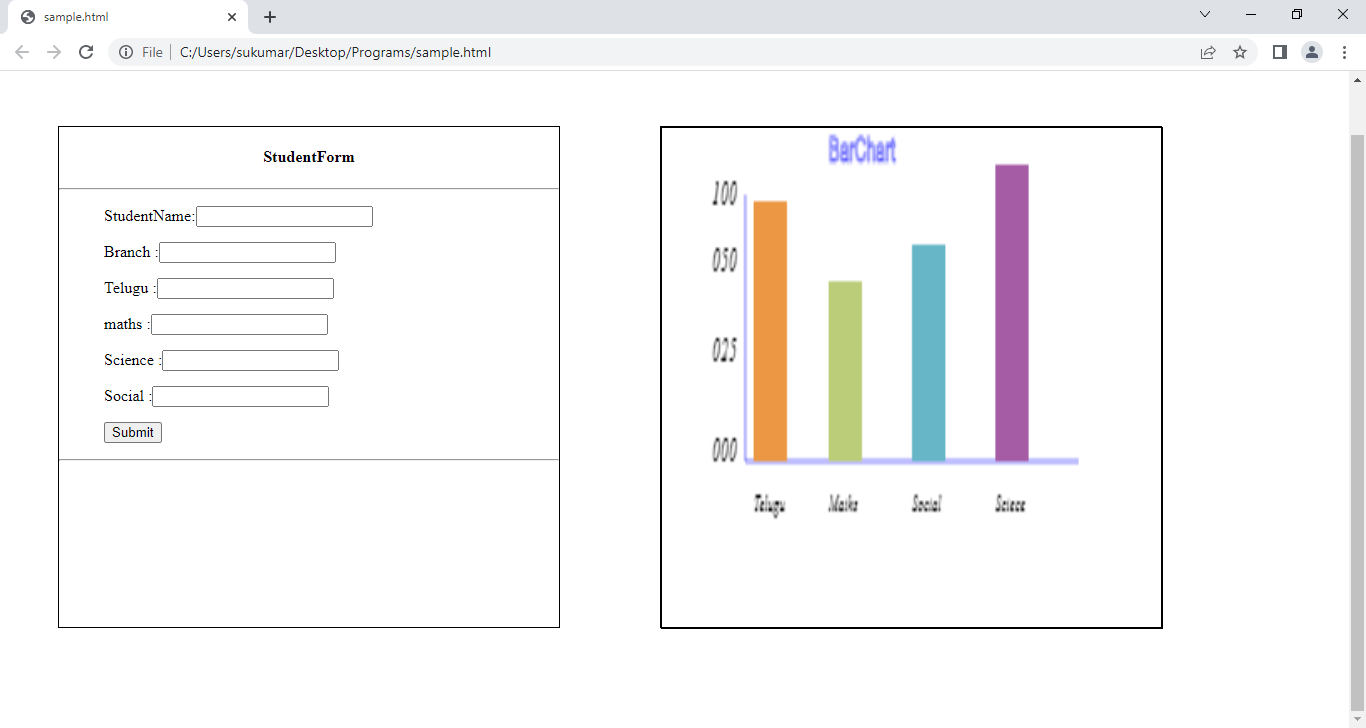
    </body>

 </html>

****

****

Press the Submit button to see the barchart



Note: The programmer has to written more code to display bar graphs, line graphs, pie charts …etc. The programmer can display them with out writing huge code using external APIS like **(chart.Js, D3.js,Googlecharts …etc).**

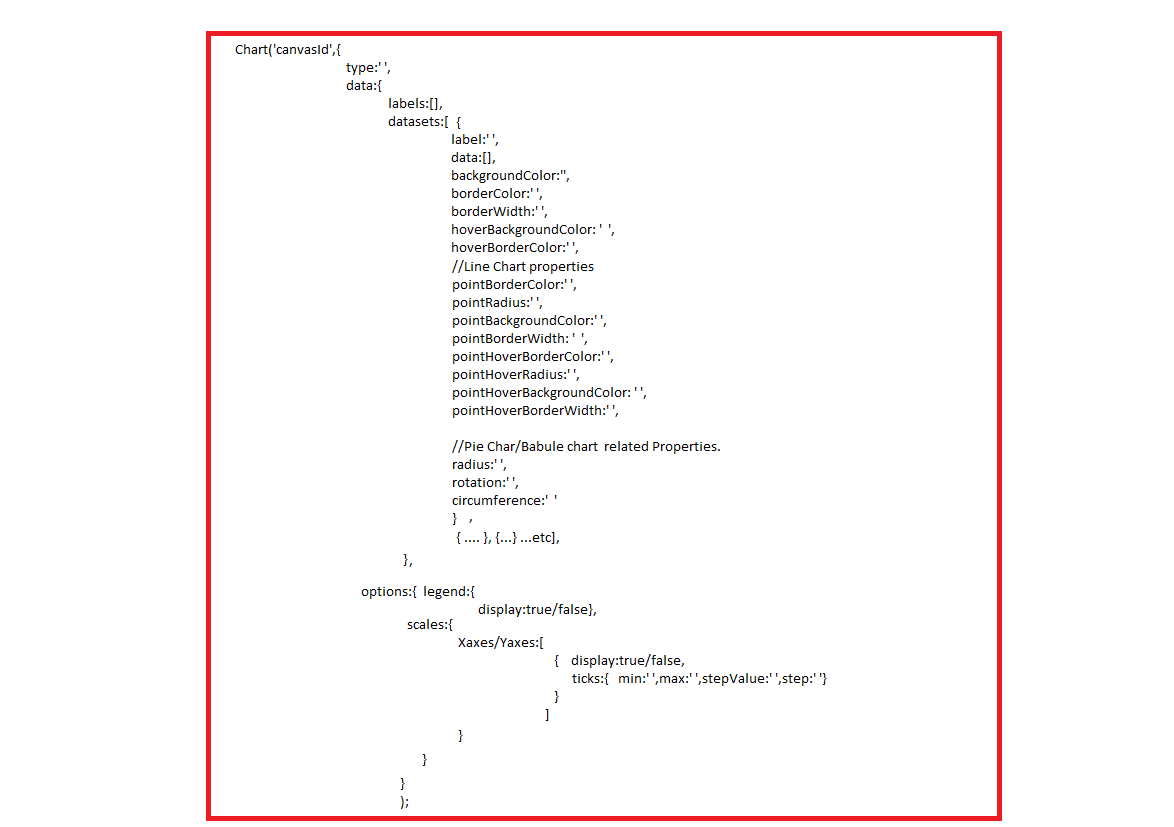
**5.1. Chart JS:-**It is free Java Script Library for making the HTML charts.

**Step1.**Link our program to CDN(content Delivery Network) of Chart.js

**Stpe2.**create a Canvas in which charts will be displayed.

**Step3.**Create a Chart() object in JS code.

**Syntax:**



a.type: It represents type of chart.

Type: bar|bubble|pie|doughnut|line|plotArea|radar|scatter.

b.labels: Its value is array. This array contains the labels of x-axis. The provided labels can be type of string or number.

c.dataSets:

c.1. label:- This label will appear in top of the chart and tooltip.

Label:”..”

c.2. hidden:- If it is true, data set is hidden from rendering in chart.

hidden:true|false.

d.options:- Its value is object. It has following properties.

d.1. layout:- It is object. This object has following properties.

Layout{

Padding:value// It is applied to all sides.

}

Or

Layout{

Padding:{ top:v1,left:v2,bottom:v3,right:v4}

}

d.2. font:- Its value is object. It contains following properties.

font:{

size: v1,

family:v2,

style:v3,

width;v4

}

d.3. legend: Its value is object.It has following property.

Display:true|false.

When you click on the label of chart, chart may appear or disappear.

d.4. scales: Its value is object. This object contains the following properties.

d.4.1. xAxis.

d.4.2.yAxis.

These properties holds Array. The array contains object. The object has following properties.

1. Display: true|false. //axis labels will or will not be displayed.
2. Ticks: Its value is object. This object has following properties.

b.1.beginAtZero:true|false.

b.2.min:v1.

b.3.max:v2.

b.4.steps:v3.

Example:1 Bar Chart

<!DOCTYPEhtml>

<html>

<scriptsrc="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.9.4/Chart.js"></script>

<body>

  <br>

  <br>

  <br>

<canvasid="myChart"style="width:100%;max-width:700px"></canvas>

<divstyle="user-select:text">

    This is sukumar.I love india.

</div>

<script>

newChart("myChart", {

  type:"bar",

  data: {

    labels:['Telugu','Hindi','English','Maths','Science','Social'],

    datasets: [{

      label:"Student Marks",

      backgroundColor:["#3e95cd", "#8e5ea2","#3cba9f","#e8c3b9","#c45850","#c78786"],

      borderColor:["#c78786","#c78786","#e8c3b9","#3cba9f","#8e5ea2","#3e95cd"],

      data:[25,54,65,79,89,91],

      barThickness:[50,50,50,50,50,50],

      hidden:false,

      borderWidth:2,

      borderRadius:5,

      hoverBackgroundColor:["#3cba9f","#8e5ea2","#3e95cd","#3e95cd","#e8c3b9","#e8c3b9"]

    }]

  },

  options:{

    legend: {display:true},

    scales:{

      yAxes: [{

                            display:true,//false

                            ticks: {

                                //beginAtZero: true,

                                min:10,

                                steps:10,

                                stepValue:5,

                                max:100

                            }

                        }],

    }

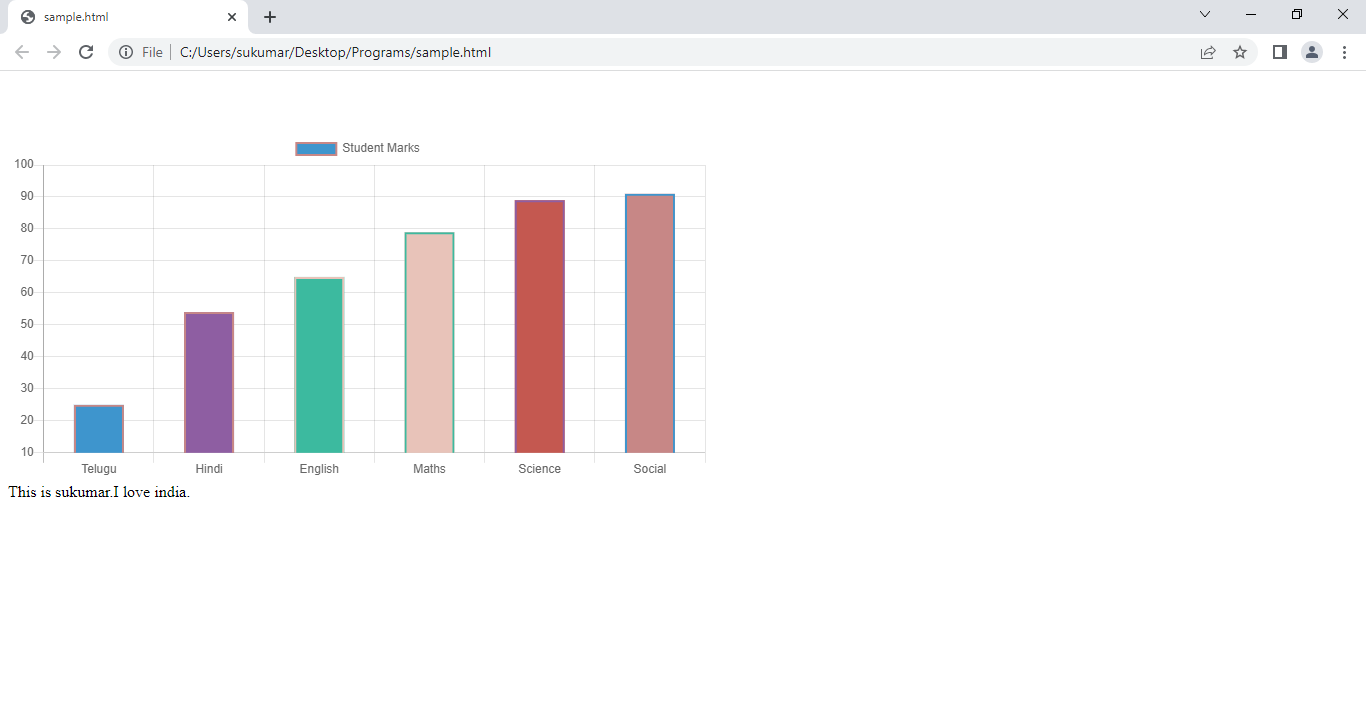
  }

});

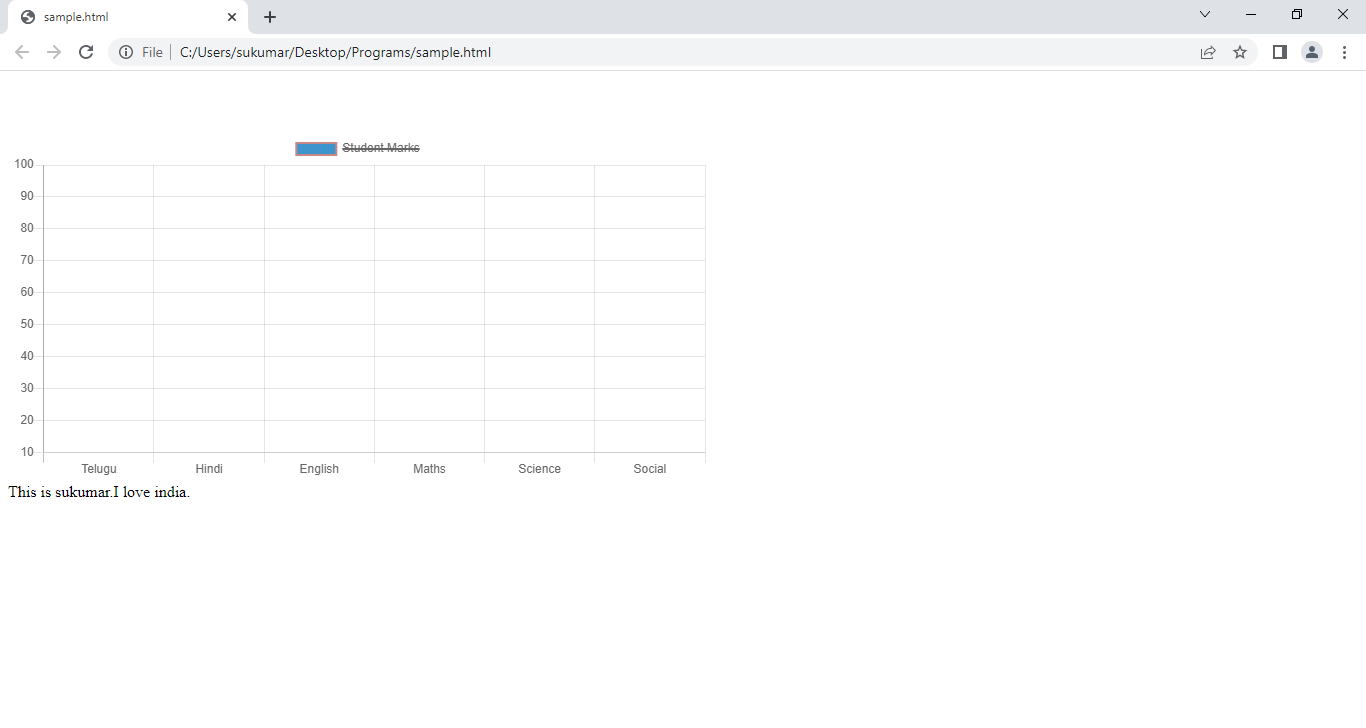
</script>

</body>

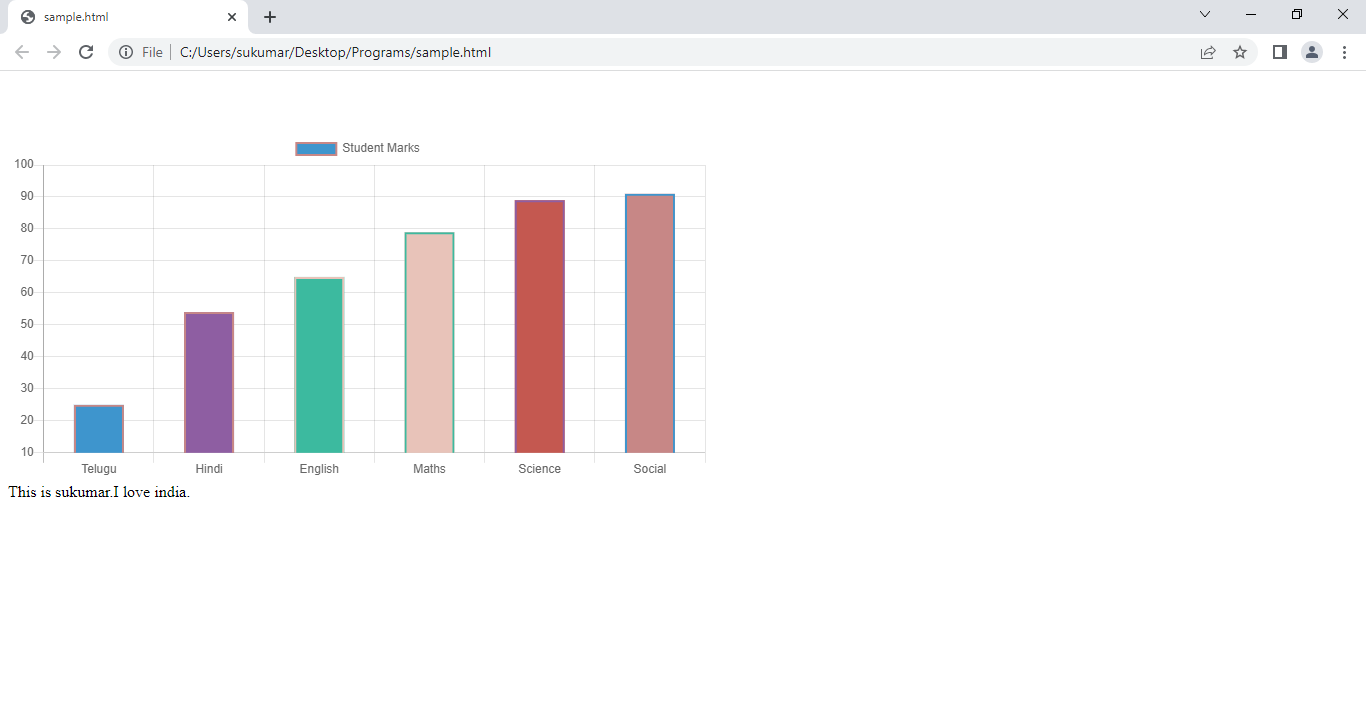
</html>



When I click label of chart(studentMarks), the chart is like below.



When I reclick the label(studentMarks), the chart is like below.



Example:2 Line Charts.

<!DOCTYPEhtml>

<html>

<scriptsrc="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.9.4/Chart.js"></script>

<body>

  <br>

  <br>

  <br>

<canvasid="myChart"style="width:100%;max-width:700px"></canvas>

<divstyle="user-select:text">

    This is sukumar.I love india.

</div>

<script>

newChart("myChart", {

  type:"line",

  data: {

    labels:['Telugu','Hindi','English','Maths','Science','Social'],

    datasets: [{

      label:"Student Marks",

      //backgroundColor:["#3e95cd", "#8e5ea2","#3cba9f","#e8c3b9","#c45850","#c78786"],

      fill:false,

      borderColor:["#e8c3b9","#c78786","#c78786","#3cba9f","#8e5ea2","#3e95cd"],

      pointRadius:5,

      data:[25,54,65,79,89,91],

      barThickness:[50,50,50,50,50,50],

      hidden:false,

      borderWidth:2,

      borderRadius:5,

      hoverBackgroundColor:["#3cba9f","#8e5ea2","#3e95cd","#3e95cd","#e8c3b9","#e8c3b9"],

    }]

  },

  options: {

    legend: {display:true},

    scales:{

      yAxes: [{

                            display:true,//false

                            ticks: {

                                //beginAtZero: true,

                                min:10,

                                steps:10,

                                stepValue:5,

                                max:100

                            }

                        }],

    }

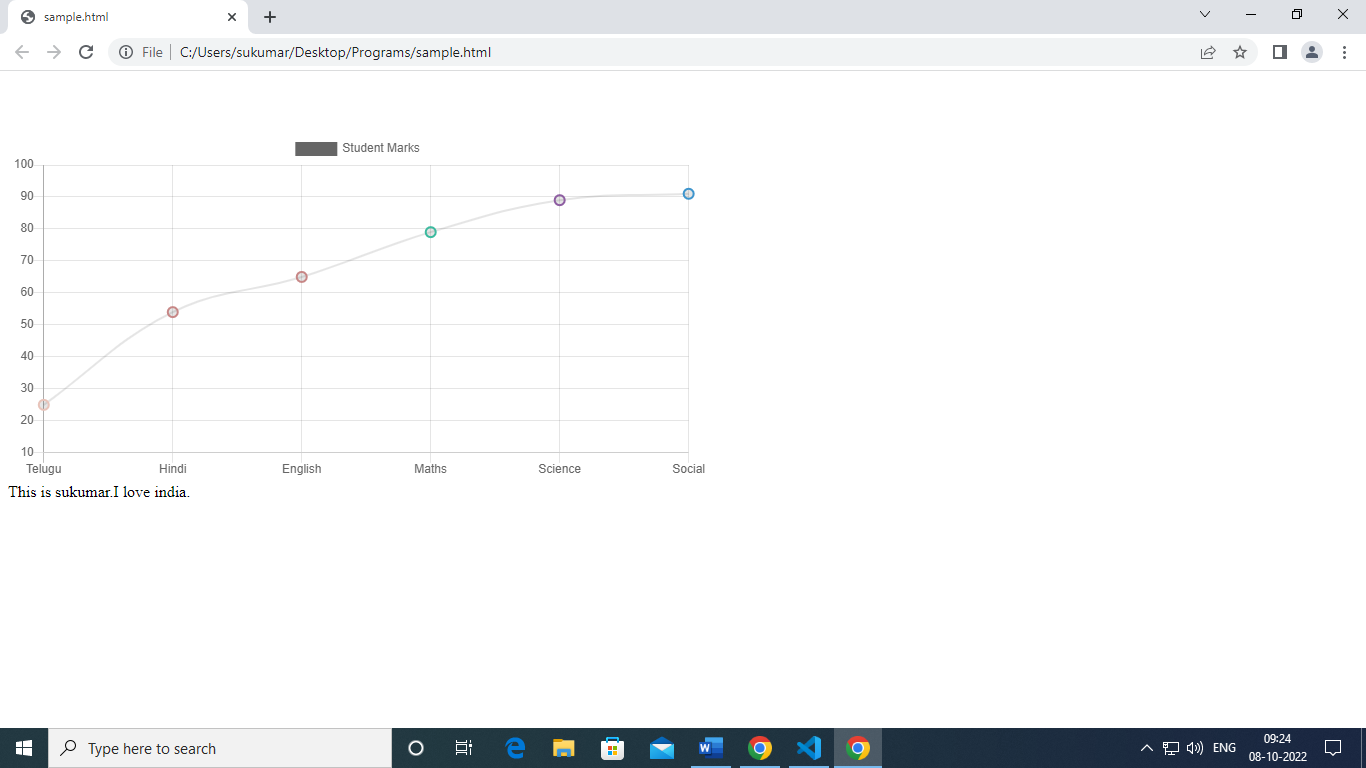
  }

});

</script>

</body>

</html>



Example:3 Pie Chart

<!DOCTYPEhtml>

<html>

<scriptsrc="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.9.4/Chart.js"></script>

<body>

  <br>

  <br>

  <br>

<canvasid="myChart"style="width:100%;max-width:700px"></canvas>

<divstyle="user-select:text">

    This is sukumar.I love india.

</div>

<script>

newChart("myChart", {

  type:"pie",

  data: {

    labels:['Telugu','Hindi','English','Maths','Science','Social'],

    datasets: [{

      label:"Student Marks",

      backgroundColor:["#e8c3b9","#3e95cd", "#8e5ea2","#3cba9f","#e8c3b9","#c45850","#c78786"],

      fill:false,//fill:true,

      borderColor:["#e8c3b9","#c78786","#c78786","#3cba9f","#8e5ea2","#3e95cd"],

      pointRadius:5,

      data:[25,54,65,79,89,91],

      barThickness:[50,50,50,50,50,50],

      hidden:false,

      borderWidth:2,

      borderRadius:5,

      hoverBackgroundColor:["#3cba9f","#8e5ea2","#3e95cd","#3e95cd","#e8c3b9","#e8c3b9"],

    }]

  },

  options: {

    legend: {display:true},

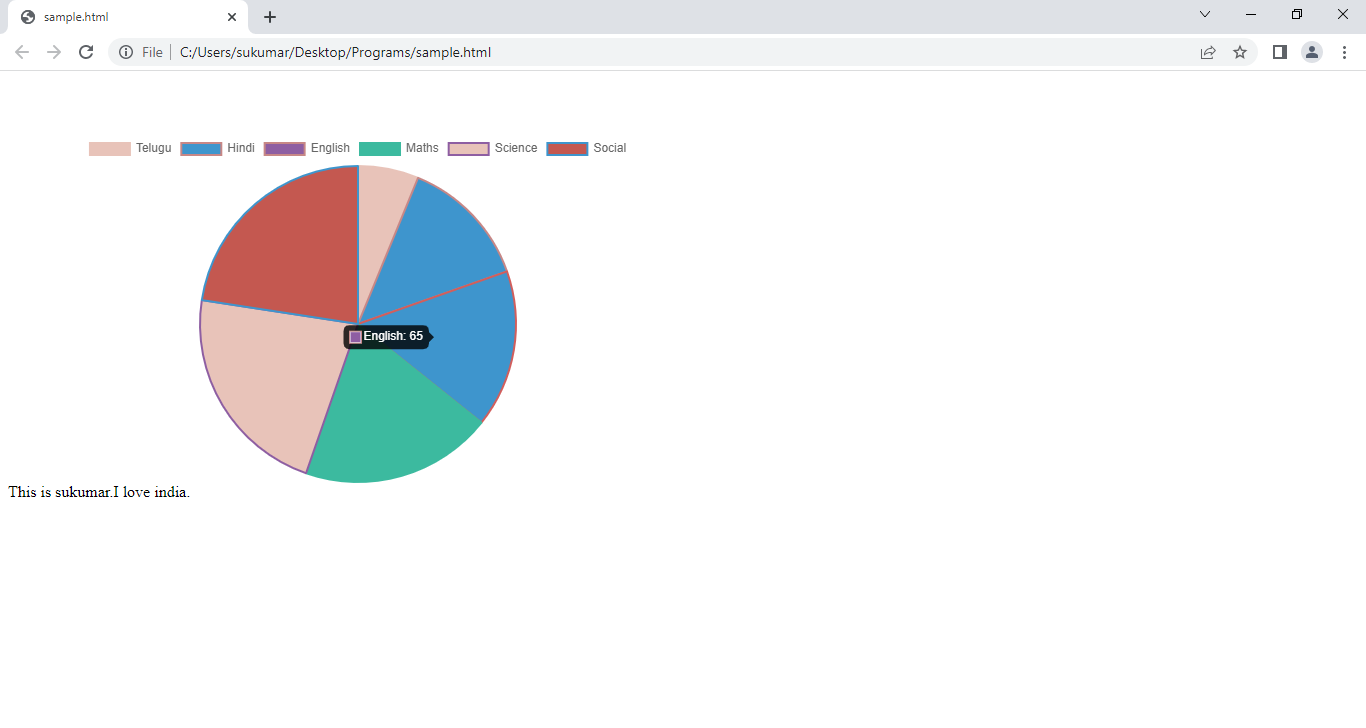
  }

});

</script>

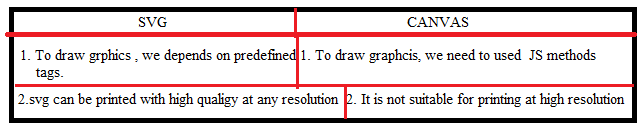
</body>

</html>



SVG: The SVG(scalar vector graphics) is area on webpage. On SVG, 2D graphics are drawn.

Diff between svg and canvas.



**6.Geo location API:-**HTML5 API which is used to identify the website visited user's geographic location.Geolocation services use Network routing addresses such as IP addresses, RFID, WIFI and MAC addresses or internal GPS devices to identify the user's location.This new feature of HTML5 allows you to navigate the latitude and longitude coordinates of the current website's visitor. These coordinates can be captured by JavaScript and based on co-ordinates your current location is shown on the website.

so geolocation API protects the user's privacy by taking the user's permission before getting the location. Geolocation API sends a notification prompt box which user can allow or deny, and if the user allows then only his location will be identified.

6.1.Getting geo location Object:- The program get geoLocation object by following statement.

Var var-name=navigator.geolocation;

6.2. Geolocation object methods:-

i.getCurrentPosition:- This method identifies a device or user current location and returns **position** object. The position object contains location data.

Syntax:

getCurrentPosition(success,error,options);

1.success:- It is callback function which receives position object as argument.

2.error:-It is callback function which receives positionError object as argument. By following reasons, program can’t get location data.

* If the user has denied for sharing location
* Location information is not available
* Request for location is timed-out.

The positionError object has following properties:

a. PERMISSION\_DENIED.

b.POSITION\_UNAVAILABLE.

c.TIME\_OUT.

d.UNKNOWN\_ERROR.

ii.watchPosition: Returns the current position of user and continues to returns the updated position as user moves(like GPS in car).

Syntax:

watchPosition(success,error,options)

iii.clearWatch(): It stops the clear watch.

**6.3.position**:- This is object which is passed as argument from getcurrentposition method to its success callback function.

1.coords.latitude.

2.coords.longitude.

3.coords.altitude.

4.coords.accuracy

5.coords.altitudeAccuracy

6.coords.speed:speed in meters persecond.

7.timestamp: The date and time of response.

**7.GoogleAPI:**

Link the Google API to current page by placing following one of following lines in <head> section .

<script src = "https://maps.googleapis.com/maps/api/js"></script>

<script src="https://maps.google.com/maps/api/js?sensor=false"></script>

**7.1.** create a map:- we should create a map inside a html container.

Syntax:

Var var-name1=new google.map.Map(htmlElement,option);

Option is object. It contains following properties.

1.zoomControl:true/false

2.scaleControl:true/false .

3.zoom:value.

4.Center:object of LatLng,

5.Time:milliseconds,// wait for specified time for permission from user.

6.rotateControl: value,

7.panControl: true/false,

8.mapTypeControl: true/false,

9.streetViewControl: true/false,

10.disableDefaultUI:true/false,//It disable or enable all controls on map.

11.mapTypeId:google.maps.mapTypeId.HYBRID|ROADMAP|SATTELLITE|TERRAIN.

12.mapTypeControlOptions:{

style:google.maps.MapTypeControlStyle.HORIZANTAL\_BAR|DEFAULT|DROPDOWN\_MENU

}

//Horizantal\_bar: it displays one button for each maptype.

//Dropdown\_menu: it displays the maptypes in dropdown menu.

zoomControlOptions:{

style:google.maps.zoomControlStyle.SMALL|LARGE|DEFAULT.

}

….etc

7.2.OverLay:- The overlay is object on map that is bound at lattiude and longitude co-ordinates.Google Maps has following overlays.

i.marker

ii.polyline:series of straight lines between multiple locations on map.

iii.polygon: series of closed straight lines between multiple locations on map.

iv.circle and rectangle.

v.InfoWindows: displays content with in popup ballon on map.

vi.custom overlays.

7.2.1. Marker:- create marker object using following syntax.

Syntax:

Var var-name2=new google.maps.Marker({

Position: new google.maps.LatLng(v1,v2),

[ Animation:newgoogle.maps.Animation.Bounce,]

[icon: image file address]

})

The marker has to be added to googleMap.

Syntax:

Var-name2.setMap(mapObject);

7.2.2.Polyline:- create polyline object using following syntax.

Syntax:

Var var-name3=new google.maps.Polyline({

Path:[ new google.maps.LatLng(v1,v2),

New google.maps.LatLng(v1,v2),

…etc.

],

strokeColor: v1,

strokeOpacity:v2,

strokeWeight:v3,

editable:true|false, //It specifies weather line is editable or not.

})

To place polylines over map, we should invoke following method.

Var-name3.setMap(mapobject);

7.2.3. polygon:create polygon object using following syntax.

Syntax:

Var var-name3=new google.maps.Polygone({

Path:[ new google.maps.LatLng(v1,v2),

New google.maps.LatLng(v1,v2),

…etc.

],

strokeColor: v1,

strokeOpacity:v2,

strokeWeight:v3,

fillColor:v4,

fillOpacity:v5,

editable:true|false, //It specifies weather line is editable or not.

})

To place polygon over map, we should invoke following method.

Var-name3.setMap(mapobject);

7.2.4.circle: create circle object using following syntax.

Syntax:

Var var-name4=new google.maps.Cicle({

Center: new google.maps.LatLg(v1,v2),

Radius: v1,

strokeColor:v1,

strokeWeight:v2,

strokeOpacity:v3,

fillColor:v4,

fillOpacity:v5,

}).

7.2.5. InfoWindow : It shows some content with marker. CreateInfo Window object using following syntax.

Syntax:

Var var-name5=new google.maps.InfoWindow({

Content:’v2’});

To place both marker and infoWindow on map, we use invoke following method.

Var-name5.open(mapobject,markerobject).

7.3. mapcontrols:- The standard googlemap has following controls.

a. Zoom - displays a slider or "+/-" buttons to control the zoom level of the map

b. Pan - displays a pan control for panning the map

c. MapType - lets the user toggle between map types (roadmap and satellite)

d. Street View - displays a Pegman icon which can be dragged to the map to enable Street View.

e. Scale.

f. Rotate.

Note:- Goto w3schools.com for more information.

Example:1

<!DOCTYPEhtml>

<html>

    <head>

      <title>Geolocation API</title>

     </head>

    <body>

       <h2>Find Your Location in below Map</h2>

        <buttononclick="getlocation();"> Show Position</button>

        <divid="demo"style="width: 600px; height: 400px; margin-left: 200px;"></div>

        <scriptsrc="https://maps.google.com/maps/api/js?sensor=false"></script>

        <scripttype="text/javascript">

        functiongetlocation(){

            if(navigator.geolocation){

                navigator.geolocation.getCurrentPosition(showPos, showErr);

            }

            else{

                alert("Sorry! your Browser does not support Geolocation API")

            }

        }

        //Showing Current Poistion on Google Map

        functionshowPos(position){

            latt = position.coords.latitude;

            long = position.coords.longitude;

            varlattlong = newgoogle.maps.LatLng(latt, long);

            varmyOptions = {

                center:lattlong,

                zoom:15,

                mapTypeControl:true,

                navigationControlOptions: {style:google.maps.NavigationControlStyle.SMALL}

            }

            varmaps = newgoogle.maps.Map(document.getElementById("demo"), myOptions);

            varmarkers =

            newgoogle.maps.Marker({position:lattlong, map:maps, title:"You are here!"});

            varx=newgoogle.maps.InfoWindow({content:'Hai You are here'})  ;

            x.open(maps,markers);

        }

        //Handling Error and Rejection

             functionshowErr(error) {

              switch(error.code){

              caseerror.PERMISSION\_DENIED:

             alert("User denied the request for Geolocation API.");

              break;

             caseerror.POSITION\_UNAVAILABLE:

             alert("USer location information is unavailable.");

            break;

            caseerror.TIMEOUT:

            alert("The request to get user location timed out.");

            break;

           caseerror.UNKNOWN\_ERROR:

            alert("An unknown error occurred.");

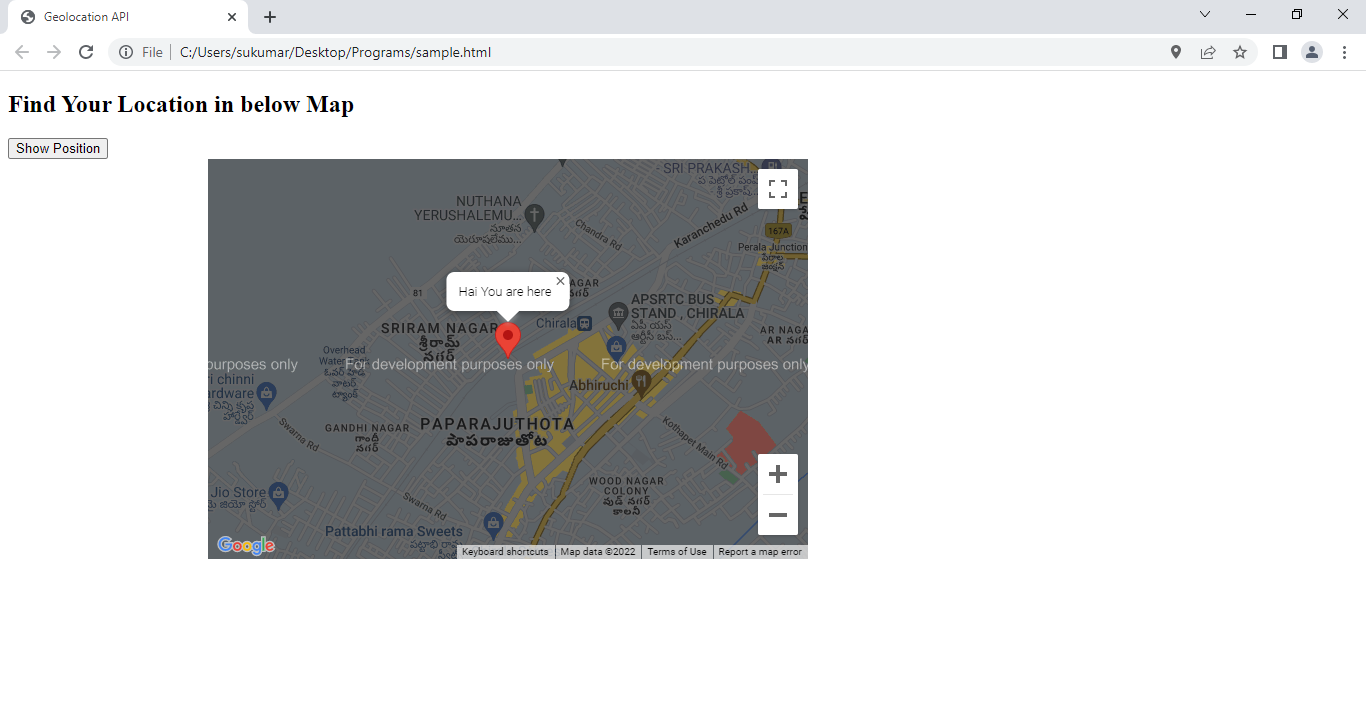
            break;

           }

        }        </script>

    </body>

</html>



Example:2

<!DOCTYPE html>

<html>

   <head>

      <script src = "https://maps.googleapis.com/maps/api/js"></script>

      <!-- <script src="https://maps.google.com/maps/api/js?sensor=false"></script>    -->

      <script>

          let lat;

          let long;

         function loadMap(){

            nav=window.navigator.geolocation;

            nav.getCurrentPosition(suc);

            function suc(p)

            {

            lat=p.coords.latitude;

            long=p.coords.longitude;

            console.log(lat);

            console.log(long);

            var mapProp = {

               center:new google.maps.LatLng(lat,long),

               zoom:15,

               mapTypeId:google.maps.MapTypeId.SATTILITE

            };

            var map = new google.maps.Map(document.getElementById("googleMap"),mapProp);

            var gmark= new google.maps.Marker({position:new google.maps.LatLng(lat,long),map:map,title:'this is sukumar house'});

            var tourplan = new google.maps.Polyline({

                path:[

                  new google.maps.LatLng(lat,long),

                  new google.maps.LatLng(15.783229805614647, 80.30283359229007),

                  new google.maps.LatLng(lat,long),

                  new google.maps.LatLng(15.82785187830909,  80.35353030578132),

                  new google.maps.LatLng(lat,long),

                  new google.maps.LatLng(15.829565332614576, 80.35803641663198),

                  new google.maps.LatLng(lat,long),

                   new google.maps.LatLng(15.776041335031353, 80.38271412501601),

                ],

               strokeColor:"red",

               strokeOpacity:0.6,

               strokeWeight:2

            });

            tourplan.setMap(map);

            let ginfo=new google.maps.InfoWindow({content:'This is my house'});

           // ginfo.open(map,gmark);

            Ginfo.open(map,gmark);

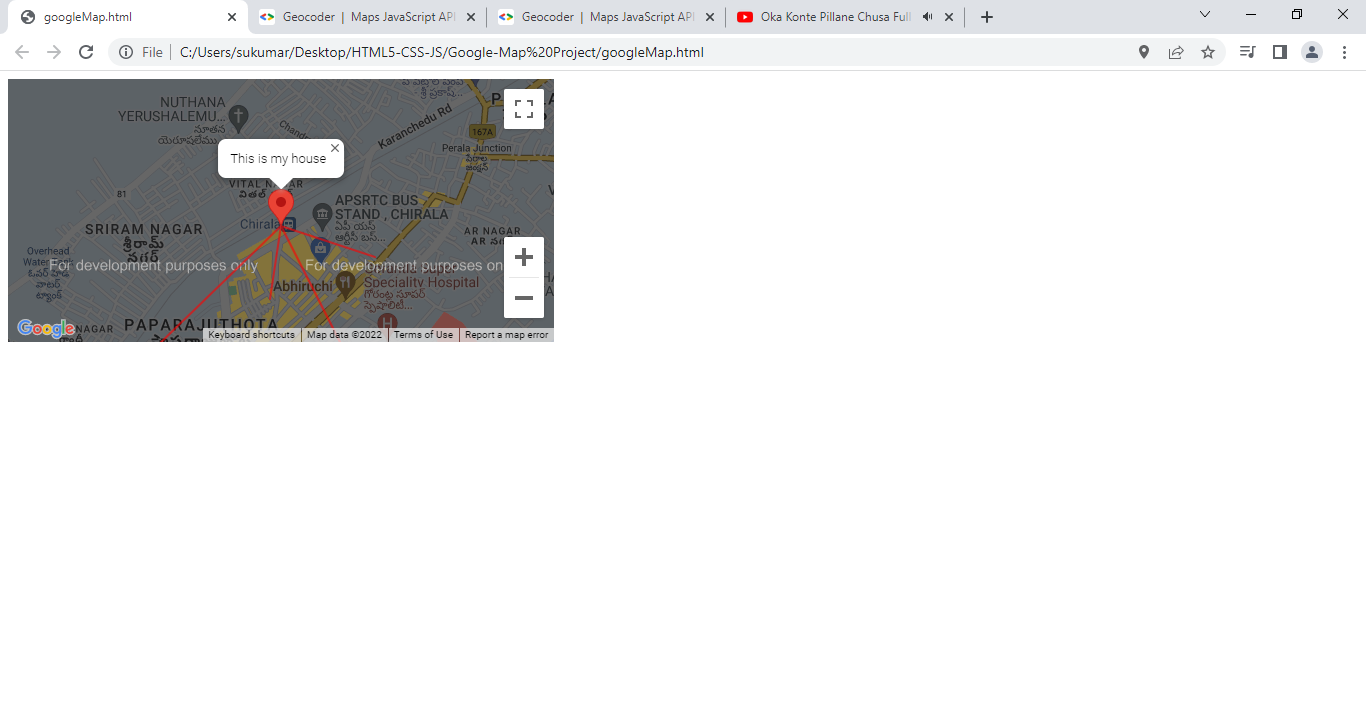
            //to remove plylines

          }  //tourplan.setmap(null);

         }

      </script>

   </head>



7.4.google.maps.geoCoder:- It is a class for converting address to lat and lang.

Syntax:

Var var-name= new google.maps.Geocoder();

This method creates instance of Geocoder. It send geocode request to google servers.

Methods:

1. geocode:-

Syntax:

Geocode({},callback(p1,p2));

**First Parameter is object**. It contains the following properties.

1. Address: location|placeId|address.
2. Language:val /A language identifier for the language in which results should be returned.
3. Region: country code //It is used to search.
4. Location: LatLng object // It helps in searching.

The second parameter is callback function. This method invokes the callback and pass two arguments.

* The first argument is **array** of GeocoderResult object.
* The second argument is GeocoderStatus .

These two objects references are available in google.maps class.

7.5. google.maps.GeocoderStatus:- This property has one of following constants.

1. ERROR: There was a problem to contact with google server.
2. OK: The response contain geocode of given address.
3. INVALID\_REQUEST: The request is invalid.
4. ZERO\_RESULTS:- No Result was found for request.
5. UNKNOW\_ERROR.

7.6. google.maps.GeocodeResult:- This object has following properties.

1. Geometry :- This property has reference of **GeocoderGeometry** object.
2. placeId:- It hold id of location.
3. Formatted\_address: It contains the string of human readable address of location.

7.7.google.maps.GeocodeGeometry:- This object has following properties.

1. Location:- It contains the reference of **LatLng** object.

**8.Semantic Tags:**

<div>…</div>

Q. When we see the only tagname(div), can we say exactlywhat content It is holding?

A. No.

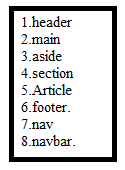
<div class=”header”>

Q.When we see the tagname and its classAttribute(div), can we say exactly what content It is holding?

A. yes . This tag will hold the heading information.

Therefore the <div> is not self descriptive tag because it can’t describe about its content by itself.

The HTML5 provided several semantic tags. These semantic tags are self descriptive tags. All semantic tags are block level elements. The semantic tags are



8.1. header:- The header is at top of web page. It contains logo,title of document, search box, social media icons and navigation bar.

The header tag can also be used inside a <section> and <article> tag.

8.2. footer:- The footer is at bottom of web page. It contains copy right notice, privacy policy, contact-Information, social media icons, Media gallery, awards certificates ..etc.

1. Privacy Polocy:- The privacy policy is a document which is required by law. This document contains following information.

* Methods to collect of information about user.
  + Ex: cookies,login,form,registration,signUp…etc.
* How Information will be used.
* How Users can block cookies and stop the advertaisements.

1. Copy right Notice: The copy right is form of protection. It is granted by law to owner of content. The copy right belongs to either entire website or part of web site.

The copy right info contains four elements.

* Copy right symbol or copy right word.
* Date(It has only year or range or year Ex: (2000-2002).
* Authorname.
* Statement of right. There are 3 types of rights.

i. All rights reserved.

ii. Some rights reserved.

iii. No rights Reserved.

Example: copyRightSymbol2017 sukumar india,inc. All rights reserved.

Note: I have developed website. I put t in public and allows people to access it. Some one may steal my website content(animation,photos,body of text ,..etc) . The copy right helps to prevent such situations.

1. Contact-Info: It contains following info.

i. Address to company.

ii.Map.

iii. Phone Numbers.

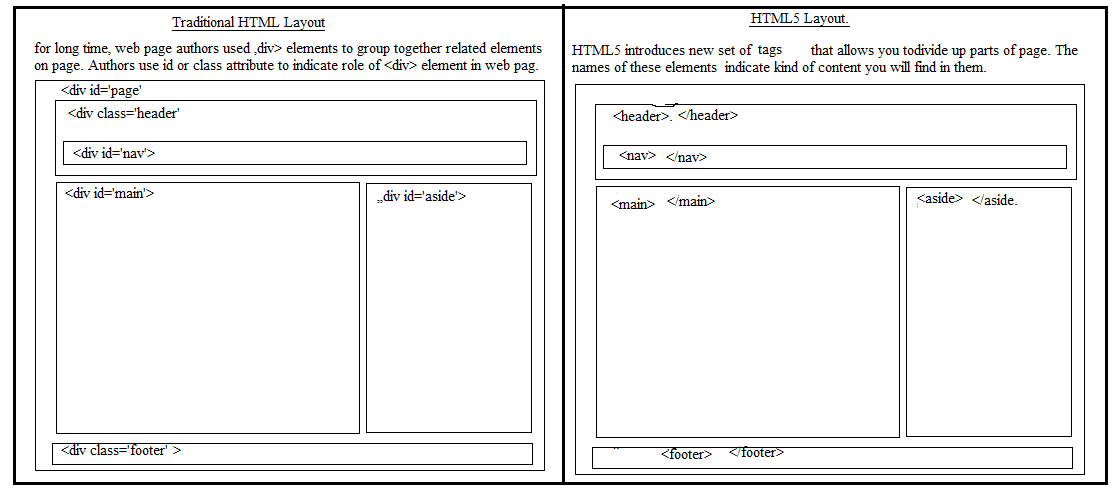
8.3.main:- This tag contains main content of web page. It should not be written inside a article ,header ,footer ,aside and section tags.

8.4.section:- It is used to make a division. We can write related content in that division. This section can have independent header ,sections and footer.

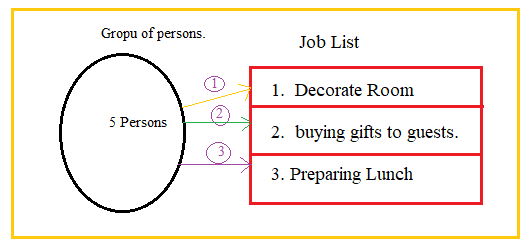
8.5.aside:- It indicates aside container. It is besides the main container usually.

8.6.article: it contains the blog posts ,forum posts ,news story.

8.7.nav:- This tag is used to make primary/secondary navigation bar of document.



**9.WebWorkers:**

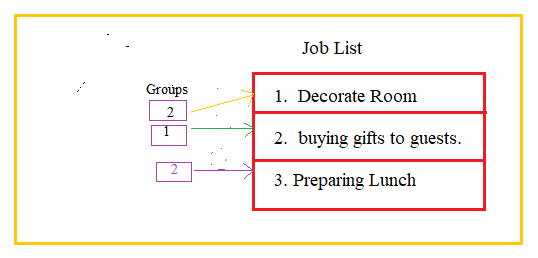
****

The 5 persons togethers does jobs sequentially. By doing sequentially, There are some problems.

1.shortest job has to wait for completion of longest job.

2. The performance of web page goes down.

Consider the following diagram



Among the 5 people, 2 people will do decoration ,2 people will prepare lunch and 1 person will buy the gifts.

Let us image, if 3 groups will do jobs simultaneously and independently then above problems will not happens.

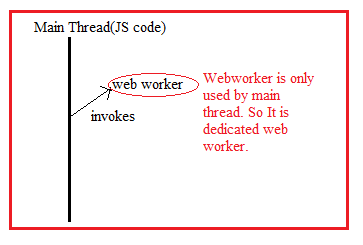
**9.1.Definition:**The Web Worker is the separate JavaScript code which runs in the background of the web page without affecting the user Interface/web page performance. The features of web worker are

1. Web worker runs code at client side.
2. Web worker is threaded java script.
3. Web workers can communicate with each other.

Limitation: The web-worker can’t communicate with DOM tree of webpage.

**9.2. Web worker Types:** In Html5 , There are 2 types of web workers.

a. Dedicated web workers:The dedicated worker can be accessed by only one script which has called it. The dedicated worker thread end as its parent thread ends. Dedicated workers are only used by one or single main thread.

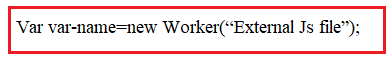


b. Shared Web workers:It can be shared by multiple scripts and can communicate using the port. Shared workers can be accessed by different windows, iframes or workers.

**9.3. Worker Object:**

The following syntax creates worker object.

Syntax:

****

This object have following methods.

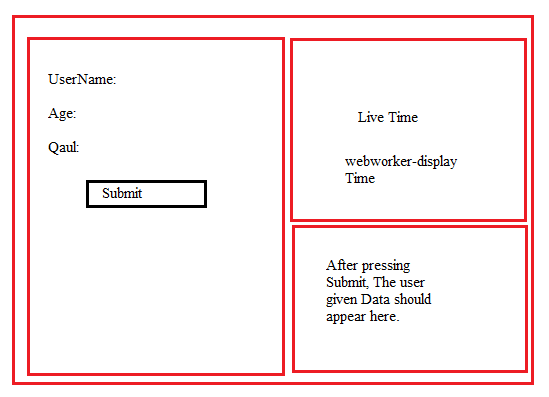
9.3.1. postMessage(“message”): This method is used to send result from web worker to main thread(JScode).

9.3.2.onMessage: It is property. It holds call back function. The callback function receives ‘event’ as argument.

The event.data property has info which has been sent by web worker.

9.3.3.terminate(): It stops the web-worker.

Example to Dedicated web workers.



Sample.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>

  <style>

    div:first-child{

      height:100vh;

      width: 48vw;

      border:2pxredsolid;

      float:left;

    }

    div:nth-child(2),div:last-child

    {

      height:48vh;

      width:48vw;

      margin-left: 1vw;

      border:2pxbluegroove;

      float:left;

      margin-top: 0.5vh;

    }

  </style>

</head>

<body>

    <div>

      UserName:<inputtype="text"><br><br><br>

      Age:<inputtype="number"><br><br><br>

      DOB:<inputtype="date"><br><br><br>

      <buttononclick="getData()">Submit</button>

    </div>

    <div>

      <h4>Time: <span></span></h4>

    </div>

    <div>

    </div>

    <script>

      variEle=document.querySelectorAll('input');

      vardEle=document.querySelectorAll('div');

      varww=newWorker("t.js");

      ww.onmessage=(event)=>{document.querySelector('span').innerHTML=event.data;};

      functiongetData(){

      dEle[2].innerHTML=iEle[0].value+''+iEle[1].value+' '+iEle[2].value;

      }

    </script>

</body>

</html>

Webworker:t.js

setInterval(()=>

{

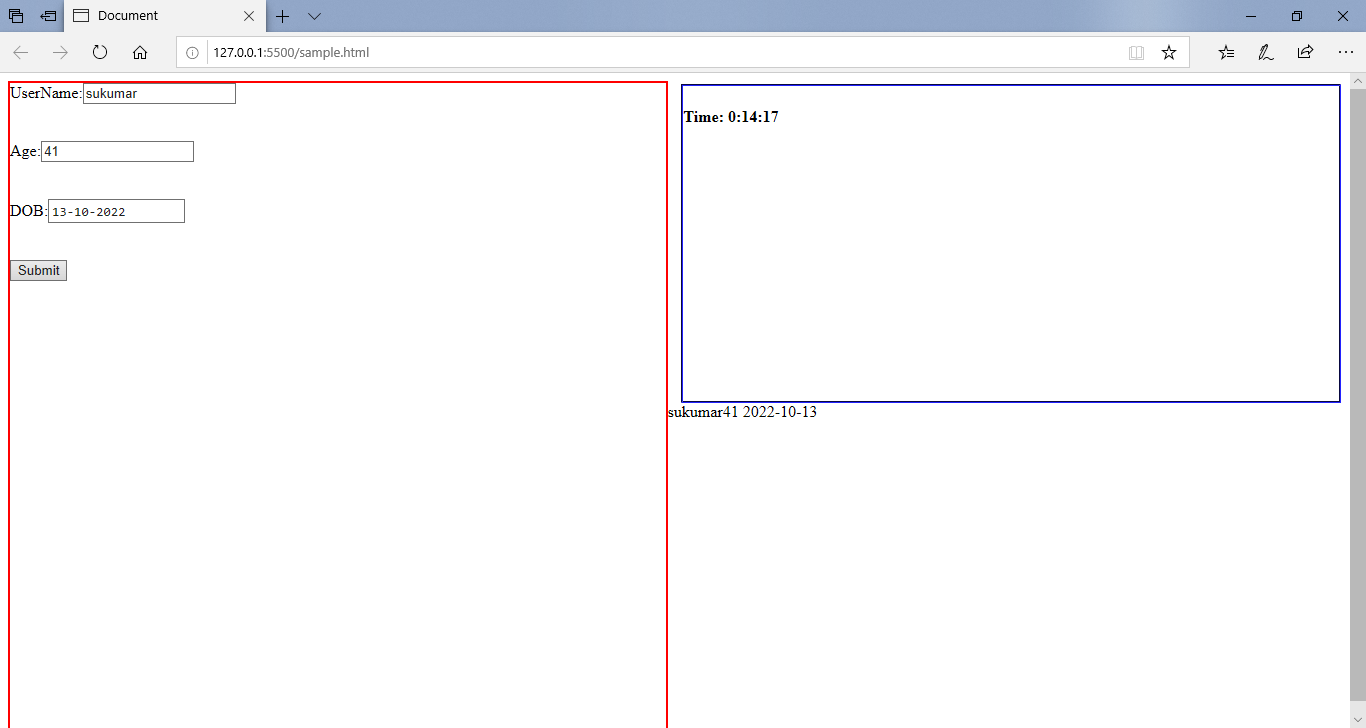
    vard=newDate();

    d1=d.getHours()+':'+d.getMinutes()+':'+d.getSeconds();

    postMessage(d1);

},1000);

Output:



**10.IndexDB:**

**11.Drag & Drop:** In html4, we have to write more JS code to achieve drag and drop functionality. HTML5 came up with drag and drop API. Therefore we can achieve it very easily.

There are no.of events which are fired during drag and drop action.

1.dragstart: it fires when user starts dragging the object.

2.drag: it fires everytime the mouse is being moved while object is being dragged.

3.drop: It fires when object is dropped in valid target.

4.dragover: it fires when object is being dragged over valid target.

5.dragenters: It fires when object enter the valid target.

When The drag & drop event occurred the event handler receives ‘event’ object. It has read only property called dataTransfer. This property holds reference of “dataTransfer” object. This object contains information of drag and drop operation. This object has following properties and methods.

1.dropEffect: It returns operation that is currently selected on object. The values are none , copy , link and move.

2.effectAllowed: It holds type of operation that is allowed on object.

effectAllowed=copy|link|move|none.

3.setData():It allows you to set the drag operation’s data to the specified format and data.

Syntax:

setData(format,data).

The format is either text/plain or text/uri-list.

4.getData():- It returns drag data stored by setData() method.

Syntax:

getData(format)

If there is nodata , it returns empty string.

5.clearData():- It clear the data stored by setData() method.

Syntax:

clearData([format])

It clear either specified formatted data or all the data,If argument is not passed.

6.files:- It returns the file list which is being dragged.

7.types:- it returns string list of formats that were set in the dragstart event.

Example:

<!DOCTYPEHTML>

<html>

<head>

<script>

functiondrag(ev) {

  ev.dataTransfer.setData("text", ev.target.id);

}

functionallowDrop(ev) {

  ev.preventDefault();

}

functiondrop(ev) {

  ev.preventDefault();

  vardata = ev.dataTransfer.getData("text");

  vardata1=document.getElementById(data);

  ev.target.appendChild(data1);

}

</script>

<style>

   #div1 {

     width: 350px;

     height: 370px;

     padding: 10px;

     border: 1pxsolid#aaaaaa;

   }

   </style>

</head>

<body>

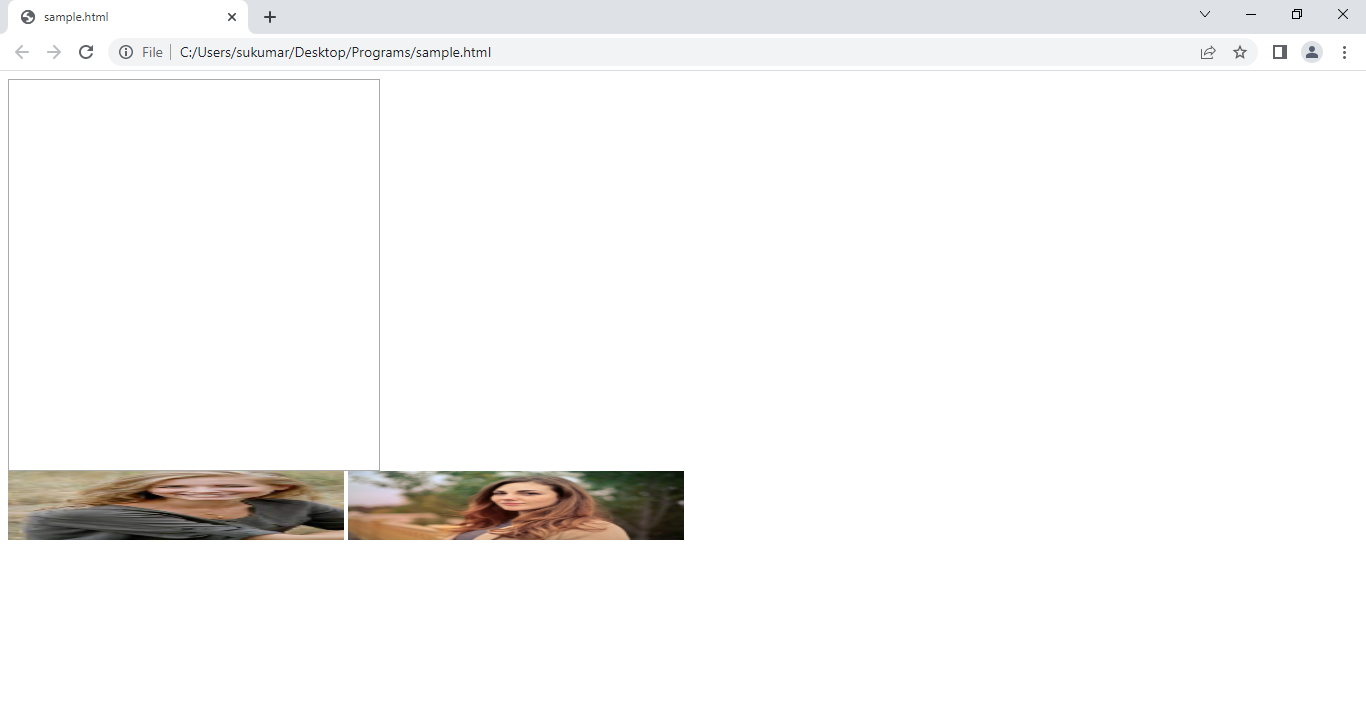
<divid="div1"ondrop="drop(event)"ondragover="allowDrop(event)"></div>

<imgid="drag1"src="front.jpg"draggable="true"ondragstart="drag(event)"width="336"height="69">

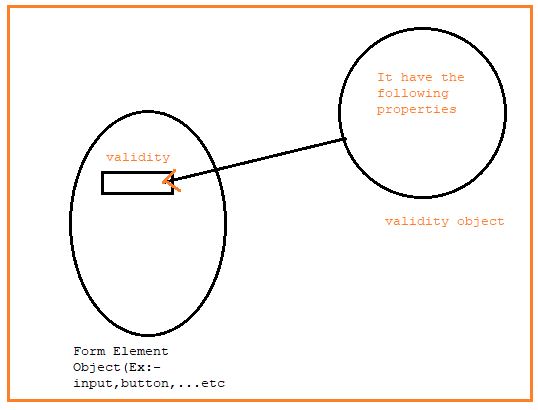
<imgid="drag2"src="back.jpg"draggable="true"ondragstart="drag(event)"width="336"height="69">

</body>

</html>



**12. WebForm API:**



1.Validity Object Properties:-

=====================

a.customError:- set to true, if custom validity message is set.

b. rangeOverflow:- set to true, if elements value is greater than its max attribute.

c.rangeUnderflow:- set to true, if element value is less than its min attribute.

d.tooLong:- set to true, if an elements value exceeds its maxLength attribute.

e. typeMismatch:- set to true, if an element’value is invalid per its type attribute.

f. valueMissing:- set to true, if an element (with required attribute) has no value.

g.valid:- set to true, if an element value is valid.

The Form “Input element” object has following methods and properties:

1. checkValidity() :- It returns true if an input element contains valid data.
2. setCustomValidity(): sets validation Message property of an input element.
3. Validity:It has the validity object reference.
4. validationMessage:- contains the message a browser will display when the validity is false.

Example:

<!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 bgcolor='green'>

<form action="">

    UserName:<input type="text" maxlength="5" id="uname"><br>

    password:<input type="number" min="1" max="10" id="pwd"><br>

    <button onclick="myFunction(event)"> submit</button>

</form>

<script>

    function myFunction(e){

        e.preventDefault();

        let a=document.getElementById("pwd");

        a.setCustomValidity("value is above range");

        if(a.validity.rangeOverflow){

            console.log(a.validationMessage);

        }

        if(a.validity.rangeUnderflow){

            console.log("value is under range");

        }

        if(a.checkValidity()){

            console.log("correct value has been given");

        }

        else{

            console.log("Wrong value has been given")

        }

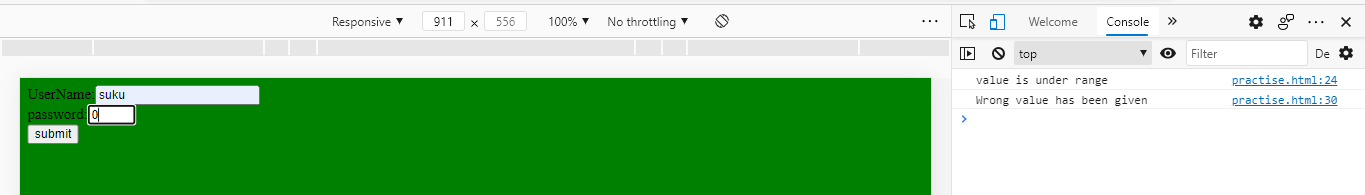
    }

</script>

</body>

</html>

Run1:-



Run2:-

