FULL STACK DEVELOPMENT

UNIT I

----

1. HTML5 New Elements:

In HTML5, there are lots of new elements are added which provides some extra functionality to create an attractive and dynamic website. With the help of these elements, you can make your code easy and quick.

Following is the complete list of the newly added elements with their descriptions.

1)Semantic Tags

2)Form Tags

3)Graphics Tags

4)Media Tags

5)input tags

Tag	Description
Structural or Semantic Tags	
<article>	It defines the independent or self-contained content of a webpage.
<aside>	It defines the content which provide information about the main content.
<bdi>	It is used to isolate the part of text which might be formatted in another direction.
<details>	It defines additional information which only visible as per user demand.
<dialog>	It represents a dialog box or other interactive components.
<figcaption>	It defines caption for the <figure> element.
<figure>	It defines a self-contained content, and referenced as a single unit.
<footer>	It represents the footer section of the webpage.
<header>	It defines the introductory or navigational content of the webpage.
<main>	It specifies the main content of the HTML document.
<mark>	It represent the text which is highlighted or marked for reference or notation purposes.
<meter>	It represents a scalar value within a known range.
<nav>	It represents the section which contains navigation links.
<progress>	It defines a progress bar which shows completions progress of a task.
<rp>	It defines alternative content for the browser which do not support ruby annotations.
<rt>	It defines explanations and pronunciations of characters in ruby annotations.
<ruby>	It defines ruby annotations (Specifically for Asian language).
<section>	It defines a generic section within an HTML document.
<summary>	It defines summary or caption for a <details> element which can be clicked to change the state of <details> element.
<time>	It defines data/time within an HTML document.

<wbr>	It specifies a line break opportunity. (Where line break possible)
HTML5 Form Tags	
<datalist>	It represent predefined list for input <option> element.
<output>	It is used a container element to represent the output of a calculation or outcome of user action.
Graphics Tags	
<canvas>	It allows drawing graphics and animations via scripting.
<svg>	It is used to draw scalable vector graphics.
HTML5 Media Tags	
<audio>	It defines sound content.
<embed>	It defines a container for external files/application/media.
<source>	It defines multiple media resources for the media elements.
<track>	It defines text tracks for <audio> and <video> files
<video>	It defines video content within HTML document.

HTML5 New <input> types

Type	**Description**
color	It represents an input field which defines a color selector.
date	It represents an input field to define a date selector.
datetime	It defines full date and time display with time zone information.
datetime-local	It defines date and time without time zone information.
email	It defines an input field with email pattern Validation.
month	It defines the input field to enter month for the particular year
number	It defines field which selects a numeric value only.

range	It defines a numeric value selector with a given range of 1 to 100.
search	It is used to define a search field.
tel	It represents a control to enter a telephone number.
time	It represents a control to enter time value with no time zone.
url	It represents an input field to enter a URL
week	It defines a selector for week value for the particular year.

2 HTML5 VIDEO and AUDIO: [10 marks]

HTML 5 supports <video> tag also. The HTML video tag is used for streaming video files such as a movie clip, song clip on the web page.

Currently, there are three video formats supported for HTML video tag:

1. mp4
2. webM
3. ogg

Attributes of HTML Video Tag

Let's see the list of HTML 5 video tag attributes.

Attribute	**Description**
controls	It defines the video controls which is displayed with play/pause buttons.
height	It is used to set the height of the video player.
width	It is used to set the width of the video player.
poster	It specifies the image which is displayed on the screen when the video is not played.
autoplay	It specifies that the video will start playing as soon as it is ready.
loop	It specifies that the video file will start over again, every time when it is completed.
muted	It is used to mute the video output.
preload	It specifies the author view to upload video file when the page loads.
src	It specifies the source URL of the video file.

HTML audio tag is used to define sounds such as music and other audio clips. Currently there are three supported file format for HTML 5 audio tag.

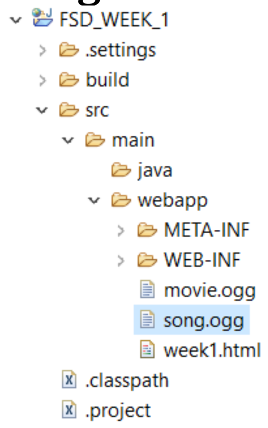
1. mp3
2. wav
3. ogg

Attributes of HTML Audio Tag

There is given a list of HTML audio tag.

Attribute	**Description**
controls	It defines the audio controls which is displayed with play/pause buttons.
autoplay	It specifies that the audio will start playing as soon as it is ready.
loop	It specifies that the audio file will start over again, every time when it is completed.
muted	It is used to mute the audio output.
preload	It specifies the author view to upload audio file when the page loads.
src	It specifies the source URL of the audio file.

Program structure



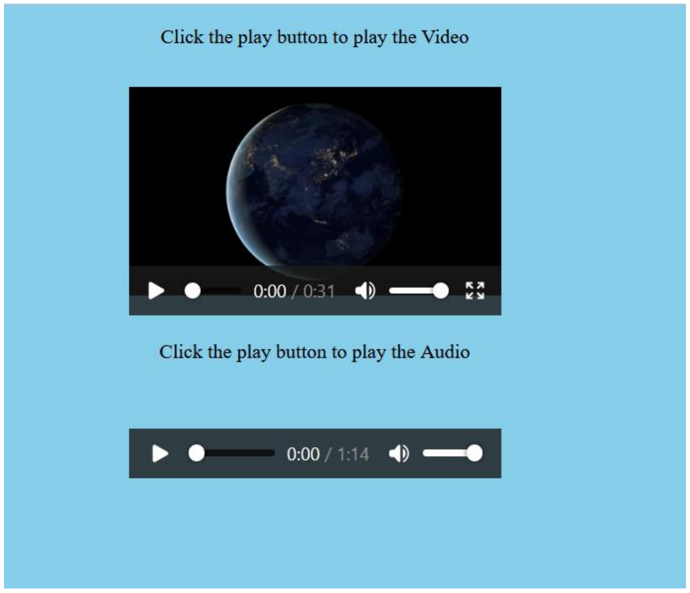
```
FSD_WEEK_1
  .settings
  build
  src
    main
      java
      webapp
        META-INF
        WEB-INF
        movie.ogg
        song.ogg
        week1.html
  .classpath
  .project
```

Program:

```
<!DOCTYPE HTML>
<html>
  <body bgcolor="skyblue">
  <center>
    <P> Click the play button to play the Video</P>
    <video  width = "300" height = "200" controls autoplay>
           <source src="movie.ogg" type="video/ogg">
    </video>
```

```
        <P> Click the play button to play the Audio</P><br><br>
     <audio controls>
       <source src="song.ogg" type="audio/ogg">
   </audio>
  </body>
  </center>
</html>
```

Out Put:



Click the play button to play the Video
0:00 / 0:31
Click the play button to play the Audio
0:00 / 1:14

3.HTML5 Canvas [10 Marks]

HTML Canvas Tag

The **HTML canvas element** provides HTML a bitmapped surface to work with. It is used to draw graphics on the web page.

The **HTML 5 <canvas> tag** is used to draw graphics using scripting language like JavaScript.

The <canvas> element is only a container for graphics, you must need a scripting language to draw the graphics. The <canvas> element allows for dynamic and scriptable rendering of 2D shapes and bitmap images.

It is a low level, procedural model that updates a bitmap and does not have a built-in scene. There are several methods in canvas to draw paths, boxes, circles, text and add images.

How to create a HTML canvas?

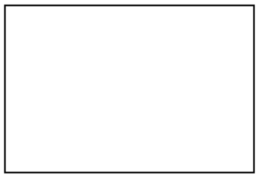
A canvas is a rectangle like area on an HTML page. It is specified with canvas element. By default, the <canvas> element has no border and no content, it is like a container.

```
<canvas id = "mycanvas" width ="200" height ="100"> </canvas>
```

HTML 5 Canvas Tag Example

```
<canvas id="myCanvas1" width="300" height="100" style="border:2px solid;">
Your browser does not support the HTML5 canvas tag.
</canvas>
```

Output:



> ***Note: It is always necessary to specify the id attribute and the height & width attribute to define the size of the canvas. You can have multiple canvas elements on one HTML page.***

HTML Canvas Tag with JavaScript

A canvas is a two dimensional grid.

Coordinates (0,0) defines the upper left corner of the canvas. The parameters (0,0,200,100) is used for fillRect() method. This parameter will fill the rectangle start with the upper-left corner (0,0) and draw a 200 * 100 rectangle.

```
<canvas id="myCanvas" width="250" height="150" style="border:1px solid #c3c3c3;">

Your browser does not support the HTML5 canvas tag.
</canvas>
<script>
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.fillStyle = "#FF0000";
ctx.fillRect(0,0,200,100);
</script>
```

Output:

Drawing Line on Canvas

If you want to draw a straight line on the canvas, you can use the following two methods.

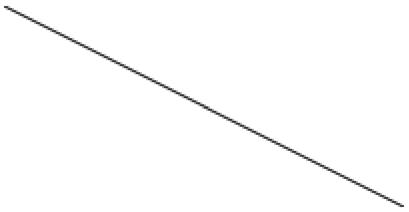
moveTo(x,y): It is used to define the starting point of the line.

lineTo(x,y): It is used to define the ending point of the line.

If you draw a line which starting point is (0,0) and the end point is (200,100), use the stroke method to draw the line.

```
<canvas id="myCanvasLine" width="200" height="100" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>
<script>
var c = document.getElementById("myCanvasLine");
var ctx = c.getContext("2d");
ctx.moveTo(0,0);
ctx.lineTo(200,100);
ctx.stroke();
</script>
```

Output:



Drawing Circle on Canvas

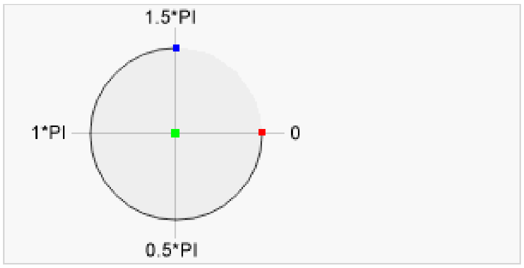
If you want to draw a circle on the canvas, you can use the arc() method:

arc(x, y, r, start, stop)

The arc() method creates an arc/curve (used to create circles, or parts of circles).

Tip: To create a circle with arc(): Set start angle to 0 and end angle to 2*Math.PI.

Tip: Use the stroke() or the fill() method to actually draw the arc on the canvas.



Center

arc(**100,75**,50,0*Math.PI,1.5*Math.PI)

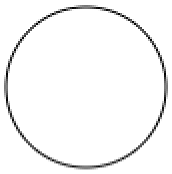
Start angle arc(100,75,50,**0**,1.5*Math.PI)

End angle arc(100,75,50,0*Math.PI,**1.5*Math.PI**)

To sketch circle on HTML canvas, use one of the ink() methods, like stroke() or fill().

```
<canvas id="myCanvasCircle" width="200" height="100" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>
<script>
var c = document.getElementById("myCanvasCircle");
var cctx = c.getContext("2d");
ctx.beginPath();
ctx.arc(95,50,40,0,2*Math.PI);
ctx.stroke();
</script>
```

Output:



Drawing text on canvas

There are property and methods used for drawing text on the canvas.

font property: It is used to define the font property for the text.

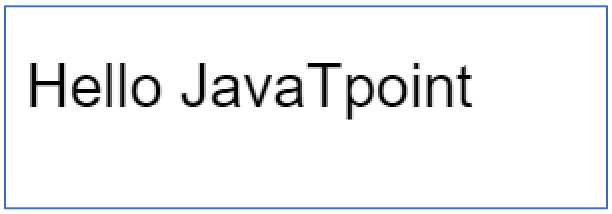
fillText(text,x,y) method: It is used to draw filled text on the canvas. It looks like bold font.

strokeText(text,x,y) method: It is also used to draw text on the canvas, but the text is unfilled.

Let's see **fillText()** method example.

```
<canvas id="myCanvasText1" width="300" height="100" style="border:1px solid #d3d3d3;">
Sorry! Your browser does not support the HTML5 canvas tag.</canvas>
<script>
var c = document.getElementById("myCanvasText1");
var cctx = c.getContext("2d");
ctx.font = "30px Arial";
ctx.fillText("Hello JavaTpoint",10,50);
</script>
```

Output:

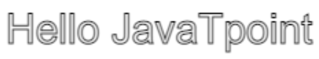


```
Hello JavaTpoint
```

Let's see **strokeText()** method example.

```
<canvas id="myCanvasText2" width="300" height="100" style="border:1px solid #d3d3d3;">
 Sorry!Upgrade your browser. It does not support the HTML5 canvas tag.</canvas>
<script>
var c = document.getElementById("myCanvasText2");
var cctx = c.getContext("2d");
ctx.font = "30px Arial";
ctx.strokeText("Hello JavaTpoint",10,50);
</script>
```

Output:

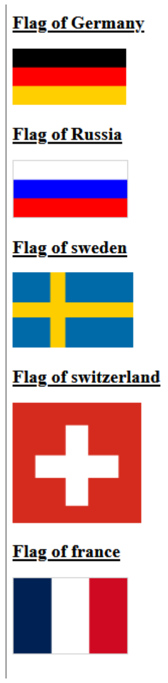


```
Hello JavaTpoint
```

Examples

```
<!DOCTYPE html>
<html>
<head>
  <title>Different countries flag Drawing on Canvas using HTML5 canvas property </title>
</head>
<body>
  <h2><u>Flag of Germany</u></h2>
  <canvas id="flag-germany" width="150px" height="75px"></canvas>

  <h2><u>Flag of Russia</u></h2>
  <canvas id="flag-Russia" width="150px" height="75px" style="border: 1px solid
#D3D3D3"></canvas>

  <h2><u>Flag of sweden</u></h2>
  <canvas id="flag-sweden" width="160px" height="100px"></canvas>

  <h2><u>Flag of switzerland</u></h2>
  <canvas id="flag-switzerland" width="170px" height="160px" style="background:
#D52B1E"></canvas>

  <h2><u>Flag of france</u></h2>
  <canvas id="flag-france" width="150px" height="100px" style="border: 1px solid
#D3D3D3"></canvas>
  <script>
   var getId1 = document.getElementById('flag-germany');
   var getIdContext1  = getId1.getContext('2d');
   getIdContext1.fillStyle = 'black';
   getIdContext1.fillRect(0,0,150,25);
   getIdContext1.fillStyle = 'red';
   getIdContext1.fillRect(0,25,150,25);
   getIdContext1.fillStyle = '#FFCF00';
   getIdContext1.fillRect(0,50,150,25);

   var getId2 = document.getElementById('flag-Russia');
   var getIdContext2  = getId2.getContext('2d');
   getIdContext2.fillStyle = 'white';
   getIdContext2.fillRect(0,0,150,25);
   getIdContext2.fillStyle = 'blue';
   getIdContext2.fillRect(0,25,150,25);
   getIdContext2.fillStyle = 'red';
   getIdContext2.fillRect(0,50,150,25);

   var getId3 = document.getElementById('flag-sweden');
   var getIdContext3  = getId3.getContext('2d');
   getIdContext3.fillStyle = '#006AA9';
   getIdContext3.fillRect(0,0,50,40);
   getIdContext3.fillStyle = '#FECD00';
   getIdContext3.fillRect(50,0,20,40);
  getIdContext3.fillStyle = '#006AA9';
  getIdContext3.fillRect(70,0,90,40);

  getIdContext3.fillStyle = '#FECD00';
  getIdContext3.fillRect(0,40,160,20);

  getIdContext3.fillStyle = '#006AA9';
  getIdContext3.fillRect(0,60,50,40);
  getIdContext3.fillStyle = '#FECD00';
  getIdContext3.fillRect(50,60,20,40);
  getIdContext3.fillStyle = '#006AA9';
  getIdContext3.fillRect(70,60,90,40);

  var getId4 = document.getElementById('flag-switzerland');
  var getIdContext4  = getId4.getContext('2d');
  getIdContext4.fillStyle = 'white';
  getIdContext4.fillRect(70,30,30,107);
  getIdContext4.fillStyle = 'white';
  getIdContext4.fillRect(30,70,110,30);

  var getId5 = document.getElementById('flag-france');
  var getIdContext5  = getId5.getContext('2d');
  getIdContext5.fillStyle = '#002153';
  getIdContext5.fillRect(0,0,50,100);
  getIdContext5.fillStyle = '#FFFFFF';
  getIdContext5.fillRect(50,0,50,100);
  getIdContext5.fillStyle = '#CF0921';
  getIdContext5.fillRect(100,0,50,100);
 </script>
</body>
</html>
```

Output:


Flag of Germany

Flag of Russia

Flag of sweden

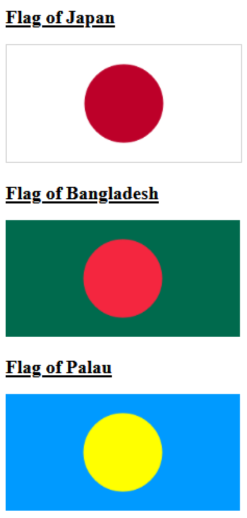
Flag of switzerland

Flag of france

Example-2

```
<!DOCTYPE html>
<html>
<head>
  <title>Draw Circulr flags on Canvas </title>
</head>
<body>
  <h2><u>Flag of Japan</u></h2>
  <canvas id="flag-japan" width="300px" height="150px" style="border: 0.5px solid #D3D3D3;"></canvas>

  <h2><u>Flag of Bangladesh</u></h2>
  <canvas id="flag-bangladesh" width="300px" height="150px" style="background:#006A4D"></canvas>

  <h2><u>Flag of Palau</u></h2>
  <canvas id="flag-palau" width="300px" height="150px" style="background:#009AFF;"></canvas>

  <script>
   var getId1 = document.getElementById('flag-japan');
   var getIdContext1  = getId1.getContext('2d');
   getIdContext1.beginPath();
   getIdContext1.arc(150,75,50,0,2 * Math.PI);
   getIdContext1.fillStyle = "#BD0029";
   getIdContext1.fill();
   getIdContext1.strokeStyle = "#BD0029";
   getIdContext1.stroke();

   var getId2 = document.getElementById('flag-bangladesh');
   var getIdContext2  = getId2.getContext('2d');
   getIdContext2.beginPath();
   getIdContext2.arc(150,75,50,0,2 * Math.PI);
   getIdContext2.fillStyle = "#F4263F";
   getIdContext2.fill();
   getIdContext2.strokeStyle = "#F4263F";
   getIdContext2.stroke();

   var getId3 = document.getElementById('flag-palau');
   var getIdContext3  = getId3.getContext('2d');
   getIdContext3.beginPath();
   getIdContext3.arc(150,75,50,0,2 * Math.PI);
   getIdContext3.fillStyle = "#FFFF00";
   getIdContext3.fill();
   getIdContext3.strokeStyle = "#FFFF00";
   getIdContext3.stroke();
  </script>
</body>
</html>
```

Output:



Flag of Japan

Flag of Bangladesh

Flag of Palau

4. Vector Graphics or (scalable vector graphics)

The Scalable Vector Graphics (SVG) is an XML-based image format that is used to define two-dimensional vector based graphics for the web. Unlike raster image (e.g. .jpg, .gif, .png, etc.), a vector image can be scaled up or down to any extent without losing the image quality.
An SVG image is drawn out using a series of statements that follow the XML schema — that means SVG images can be created and edited with any text editor, such as Notepad. There are several other advantages of using SVG over other image formats like JPEG, GIF, PNG, etc.

- SVG images can be searched, indexed, scripted, and compressed.
- SVG images can be created and modified using JavaScript in real time.
- SVG images can be printed with high quality at any resolution.
- SVG content can be animated using the built-in animation elements.
- SVG images can contain hyperlinks to other documents.

Embedding SVG into HTML Pages

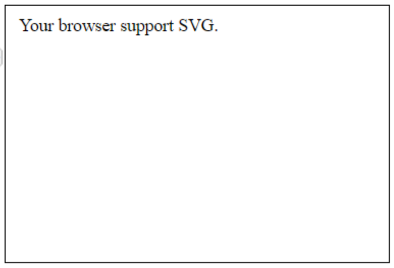
You can embed SVG graphics directly into your document using the

HTML5 **<svg>** element.

Let's take a look at the following example to understand how it basically works:

```
<!DOCTYPE html>
<html>
<style>
   svg {
      border: 1px solid black;
   }
</style>
<body>
   <svg width="300" height="200">
      <text x="10" y="20" style="font-size:14px;">
         Your browser support SVG.
      </text>
      Sorry, your browser does not support SVG.
   </svg>
</body>
</html>
```

Output:

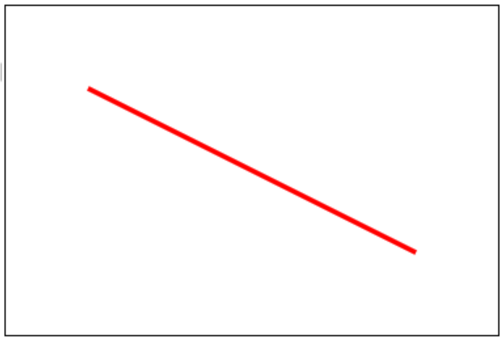

Your browser support SVG.

Drawing a Line

The most basic path you can draw with SVG is a straight line. The following example will show you how to create a straight line using the SVG `<line>` element:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Create a Line with HTML5 SVG</title>
<style>
   svg {
       border: 1px solid black;
   }
</style>
</head>
<body>
   <svg width="300" height="200">
       <line x1="50" y1="50" x2="250" y2="150" style="stroke:red; stroke-width:3;" />
   </svg>
</body>
</html>
```

Output:



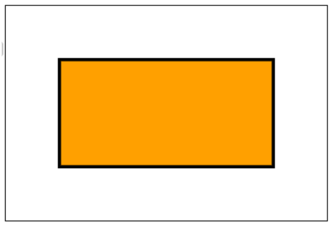
The attributes `x1`, `x2`, `y1` and `y2` of the `<line>` element draw a line from `(x1,y1)` to `(x2,y2)`.

Drawing a Rectangle

You can create simple rectangle and square shapes using the SVG **<rect>** element. The following example will show you how to create and style a rectangular shape with SVG:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Create a Rectangle with HTML5 SVG</title>
<style>
    svg {
        border: 1px solid black;
    }
</style>
</head>
<body>
    <svg width="300" height="200">
        <rect x="50" y="50" width="200" height="100" style="fill:orange; stroke:black; stroke-
width:3;" />
    </svg>
</body>
</html>
```

Output:



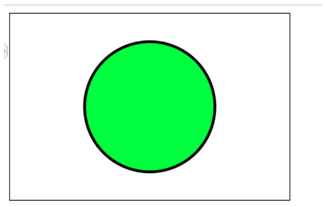
The attributes x and y of <rect> element defines the co-ordinates of the top-left corner of the rectangle. The attributes width and height specifies the width and height of the shape.

Drawing a Circle

You can also create the circle shapes using the SVG <circle> element. The following example will show you how to create and style a circular shape with SVG:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Create a Circle with HTML5 SVG</title>
<style>
    svg {
        border: 1px solid black;
    }
</style>
</head>
<body>
    <svg width="300" height="200">
        <circle cx="150" cy="100" r="70" style="fill:lime; stroke:black; stroke-width:3;" />
    </svg>
</body>
</html>
```

Output



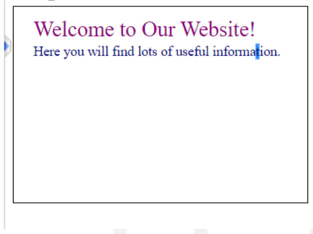
The attributes cx and cy of the <circle> element defines the co-ordinates of the center of the circle and the attribute r specifies the radius of the circle. However, if the attributes cx and cy are omitted or not specified, the center of the circle is set to (0,0).

Drawing Text with SVG

You can also draw text on the web pages with SVG. The text in SVG is rendered as a graphic so you can apply all the graphic transformation to it but it is still acts like text — that means it can be selected and copied as text by the user. Let's try an example to see how this works:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Render Text with HTML5 SVG</title>
<style>
    svg {
        border: 1px solid black;
    }
</style>
</head>
<body>
    <svg width="300" height="200">
        <text x="20" y="30" style="fill:purple; font-size:22px;">
            Welcome to Our Website!
        </text>
        <text x="20" y="30" dx="0" dy="20" style="fill:navy; font-size:14px;">
          Here you will find lots of useful information.
        </text>
    </svg>
</body>
</html>
```

Output:



Welcome to Our Website!
Here you will find lots of useful information.

The attributes x and y of the <text> element defines the location of the top-left corner in absolute terms whereas the attributes dx and dy specifies the relative location.

Text direction:

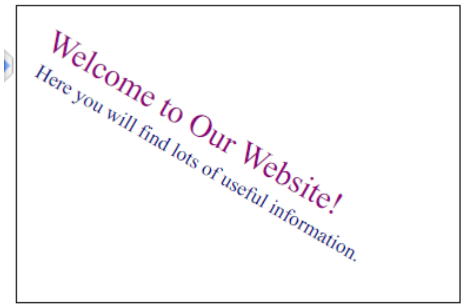
You can use the <tspan> element to reformat or reposition the span of text contained within a <text> element. Text contained in separate tspans, but inside the same text element can all be selected at once — when you click and drag to select the text. However, the text in separate text elements cannot be selected at the same time.

Let's check out an example:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Rotate and Render Text with HTML5 SVG</title>
<style>
    svg {
        border: 1px solid black;
    }
</style>
</head>
<body>
    <svg width="300" height="200">
        <text x="30" y="15" style="fill:purple; font-size:22px; transform:rotate(30deg);">
            <tspan style="fill:purple; font-size:22px;">
                Welcome to Our Website!
            </tspan>
            <tspan dx="-230" dy="20" style="fill:navy; font-size:14px;">
                Here you will find lots of useful information.
            </tspan>
        </text>
    </svg>
```

```
</body>
</html>
```

Output:



Differences between SVG and Canvas[5 Marks]

The HTML5 introduced the two new graphical elements `<canvas>` and `<svg>` for creating rich graphics on the web, but they are fundamentally different.

SVG	**Canvas**
Vector based (composed of shapes)	Raster based (composed of pixel)
Multiple graphical elements, which become the part of the page's DOM tree	Single element similar to `<img>` in behavior. Canvas diagram can be saved to PNG or JPG format
Modified through script and CSS	Modified through script only
Good text rendering capabilities	Poor text rendering capabilities
Give better performance with smaller number of objects or larger surface, or both	Give better performance with larger number of objects or smaller surface, or both
Better scalability. Can be printed with high quality at any resolution. Pixelation does not occur	Poor scalability. Not suitable for printing on higher resolution. Pixelation may occur

5. Web Storage [10 Marks]

HTML Web Storage:

The Web Storage is one of the great features of HTML5. With the Web Storage feature, *web applications can locally store data within the browser on the client side*. It stores data in the form of key/value pair on the browser. Web Storage sometimes also known as DOM storage.

Storing data with the help of web storage is similar to cookies, but it is better and faster than cookies storage.

In compared to cookies Web Storage has Following Advantages:

- Web Storage can use storage space upto 5MB per domain. (The browser software may prompt the user if the space limit is reached).
- It will not send data to the server side, hence it is faster than cookies storage.
- The data stored by local Storage never expires, but cookies data expires after some time or session.
- Web Storage is more secure than cookies.

Types of Web Storage:

There are two types of web storage with different scope and lifetime.

- **Local Storage:** Local Storages uses Windows.localStaorage object which stores data and available for every page. But data persist even if the browser is closed and reopened (Stores data with no Expiration).
- **Session Storage:** Session Storage uses Windows.sessionStorage object which stores data for one session and data will be lost if the window or browser tab will be closed.

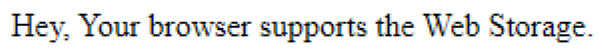
Note: For both storage type, web storage data will not be available for different browsers, and Storage size may vary from browser to browser.

Browser support for Web Storage:

Before learning for web Storage we must check whether our browser is supporting the web Storage or not. So you can check by executing the following code:

```
<!DOCTYPE html>
<html>
<body>
 <div id="result"></div>
 <script>
  if(typeof(Storage)!=="undefined") {
   document.getElementById("result").innerHTML = "Hey, Your browser supports the Web Storage.";
  }
  else{
document.getElementById("result").innerHTML = "Sorry, your browser does not support Web Storage";
 }
</script>
</body>
</html>
```

Output:



```
Hey, Your browser supports the Web Storage.
```

The localStorage Object

The localStorage object stores data locally within the browser. The data stored by localStroage object does not have any expiration date. Hence the stored data will not be deleted if the browser is closed or reopened.

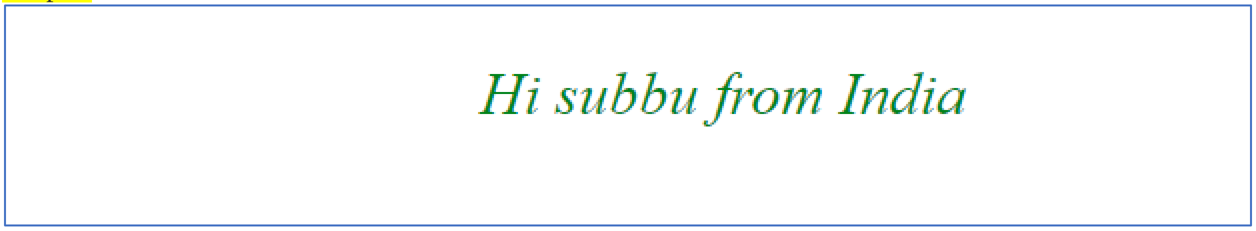
Each piece of data is stored in simple key-value pairs. The key/values are always stored as String, and can be accessed with localStorage.getItem() and localStorage.setItem() methods.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Web Storage API</title>
  <style>
    body{
      color: green;
      text-align: center;
      font-size: 30px;
      margin-top: 30px;
      font-style: italic;
    }
  </style>
</head>
<body>
<script>
 if(typeof(Storage)!=="undefined") {
  localStorage.setItem("name","subbu");
  localStorage.setItem("Country", "India");
   document.write("Hi"+" "+localStorage.name+" "+"from" +" "+ localStorage.Country);
}
 else{
  alert("Sorry! your browser is not supporting the browser")
 }
</script>
</body>
```

```
</html>
```

Output:



```
Hi subbu from India
```

Example Explanation:

- In the above example, we have used **typeof(Storage)!=="undefined"** to check browser support.
- **localStorage.setItem("name","Harshita")** is used to set the key and value data where "name" is key and "Harshita" is value.
- The **localStorage.name** is used to retrieve the values using key. You can also use another method: **localStorage.getItem** to retrieve the value.

Remove Web Storage:

As we have seen the session storage data will automatically be deleted, when you close the browser but the data saved by local storage will remain in the browser even if you close it.

Hence to delete the local storage data, you need to call two methods:

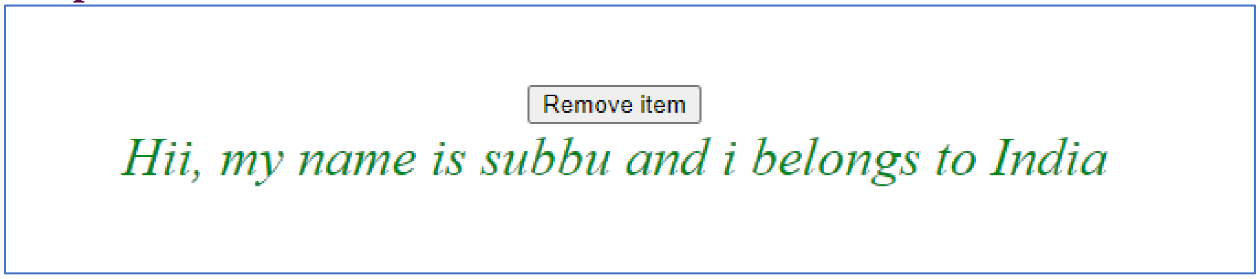
- **localStorage.removeItem('key'):** If you want to delete the value on a particular key, then you can use the "key," and that value will be deleted.
- **localStorage.clear():** If you want to delete or clear all settings with key/value pair, then you can call this method.

Example

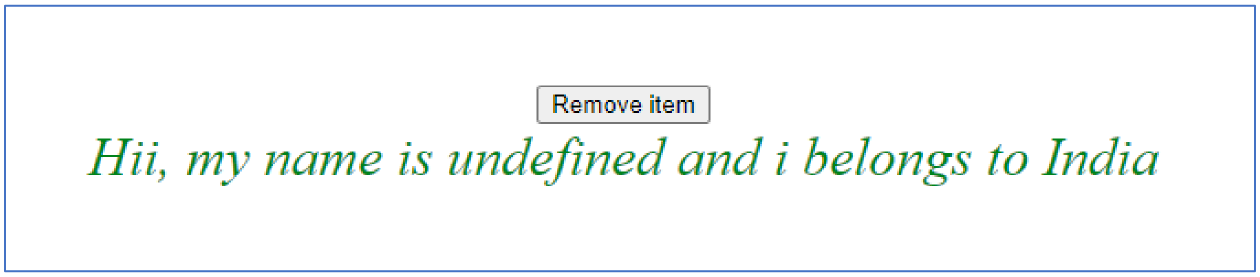
```
<!DOCTYPE html>
<html>
<head>
  <title>Web Storage API</title>
  <style>
     body{
       color: green;
       text-align: center;
       font-size: 30px;
       margin-top: 30px;
       font-style: italic;
       }
  </style>
</head>
<body>
<button onclick="remove();">Remove item</button>
<div id="output"></div>

```

```
<script>
 if(typeof(Storage)!=="undefined") {
  localStorage.setItem("name","Harshita");
  localStorage.setItem("Country", "India");
  document.getElementById('output').innerHTML= "Hii, my name is"+ " "+ localStorage.
name +" "+"and i belongs to"+" "+localStorage.Country;
   }
  else{
  alert("Sorry! your browser is not supporting the browser")
 }
  function remove() {
 localStorage.removeItem("name");
   document.getElementById('output').innerHTML= "Hii, my name is"+ " "+ localStorage.name +
" "+"and i belongs to"+" "+localStorage.Country;
}
</script>
</body>
</html>
```

Output:



Hii, my name is subbu and i belongs to India



Hii, my name is undefined and i belongs to India

Example Explanation:

In the above example we have used **localStorage.removeItem("name");** Which will delete the value for the key "name".

You can remove id for a particular key, or you can also remove all data using **localStorage.clear()** method.

6. Drag & Drop

HTML Drag and Drop (DnD) is a feature of HTML5. It is a powerful user interface concept which *is used to copy, reorder and delete items with the help of mouse*. You can hold the mouse button down over an element and drag it to another location. If you want to drop the element there, just release the mouse button.

If you want to achieve the Drag and Drop functionality in traditional HTML4, you must either have to use complex JavaScript programming or other JavaScript frameworks like jQuery etc.

Events for Drag and Drop feature

Event	**Description**
Drag	It fires every time when the mouse is moved while the object is being dragged.
Dragstart	It is a very initial stage. It fires when the user starts dragging object.
Dragenter	It fires when the user moves his/her mouse cursur over the target element.
Dragover	This event is fired when the mouse moves over an element.
Dragleave	This event is fired when the mouse leaves an element.
Drop	Drop It fires at the end of the drag operation.
Dragend	It fires when user releases the mouse button to complete the drag operation.

HTML5 Drag and Drop Example

Let's see an example of HTML 5 drag and drop feature.

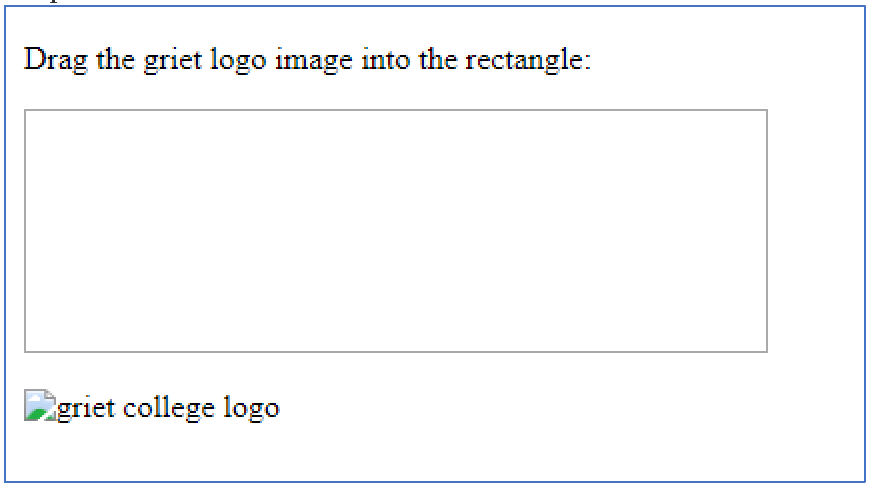
To understand this example, you must have the knowledge of JavaScript.

```
<script>
function allowDrop(ev) {ev.preventDefault();}
function drag(ev) {ev.dataTransfer.setData("text/html", ev.target.id);}
function drop(ev) {
ev.preventDefault();
var data = ev.dataTransfer.getData("text/html");
ev.target.appendChild(document.getElementById(data));
}
</script>
<p>Drag the griet logo image into the rectangle:</p>
```

```
<div id="div1" style="width:350px;height:100px;padding:10px;border:1px solid #aaaaaa;"
ondrop="drop(event)" ondragover="allowDrop(event)"></div>
<br>
<img id="drag1" src="grietlogo.png" alt="griet college logo"
draggable="true" ondragstart="drag(event)"/>
```

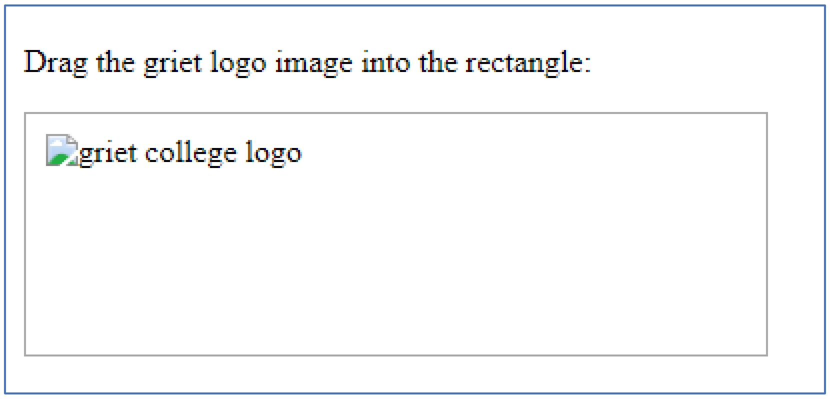
In the above example, we have used **ondrop and ondragover events on div** element, and **ondragstart event on img** tag.

Output:



Drag the griet logo image into the rectangle:

griet college logo



Drag the griet logo image into the rectangle:

griet college logo

Note: MouseEvent is not fired during drag operation.

Stages during Drag and Drop operations

1) Make an element draggable

If you want to make an element draggable, set the draggable attribute to "true" on the element. For example:

```
<img draggable = "true">
```

2) What to drag:

Use ondragstart and setData () methods.

Specify what should happen when the element is dragged.

3) Where to Drop:

Use ondragover event.

4) Do the Drop:

Use ondrop event.

7. HTML5 Geolocation[10M]

The Geolocation is one of the best HTML5 API which is used to identify the user's geographic location for the web application.

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 send to the server which can show your current location on the website

Most of the 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.

Tips: To completely understand the concept of Geolocation API you must have some knowledge of JavaScript.

User privacy:

The user's location is the privacy concern, 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.

Note: Your browser must support the geolocation to use it for the web application. Although most of the browsers and mobile devices support the Geolocation API, and this API is only available for HTTPS request.

Geolocation object

The Geolocation API is work with the navigation.geolocation object. Its read-only property returns a Geolocation object which identifies the location of the user and can generate a customized result based on user location.

Syntax:

```
geo=navigator. geolocation;
```

If this object is present, then you can get the geolocation services.

Geolocation Methods

The Geolocation API uses three methods of Geolocation interface which are given following:

Methods	Description
getCurrentPosition()	It identifies the device or the user's current location and returns a position object with data.
watchPosition()	Return a value whenever the device location changes.
clearWatch()	It cancels the previous watchPosition() call

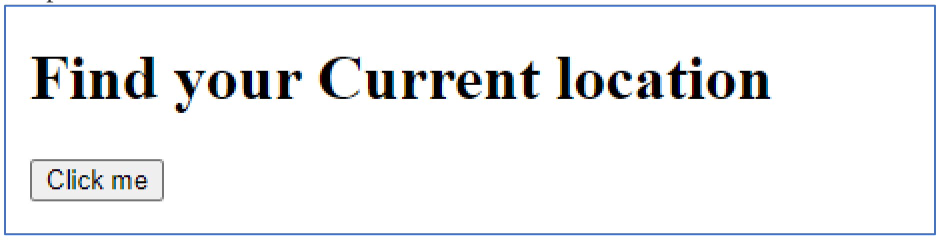
Checking for browser support:

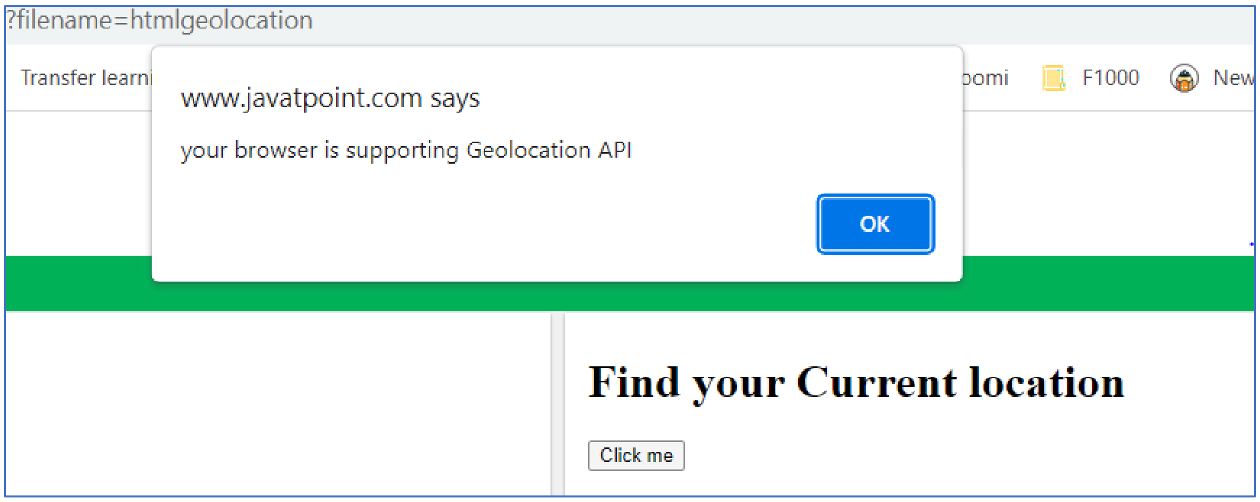
The geolocation property of navigator.geolcation object helps to determine the browser support for the Geolocation API.

```
 <!DOCTYPE html>
<html>
<head>
   <title>Geolocation API</title>
</head>
<body>
  <h1>Find your Current location</h1>
<button onclick="getlocation()">Click me</button>
<div id="location"></div>
<script>
   var x= document.getElementById("location");


   function getlocation() {
      if(navigator.geolocation){
          alert("your browser is supporting Geolocation API")
      }
       else
      {
          alert("Sorry! your browser is not supporting")
      }
   }
```

```
</script>
</body>
</html>
```

Output:





www.javatpoint.com says

your browser is supporting Geolocation API

Getting the User's current position:

To get the user's current location, getCurrentPosition() method of the navigator.geolocation object is used. This method accepts three parameters:

- **success:** A success callback function to get the location of the user
- **error:** An error callback function which takes "Position Error" object as input.
- **options:** It defines various options for getting the location.

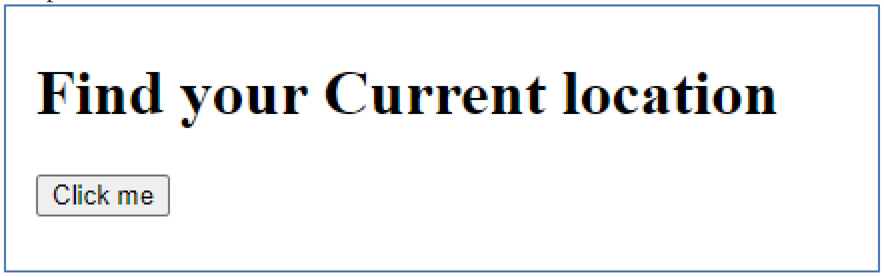
The below example will return the longitude and latitude of the visitor's current location.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Geolocation API</title>
</head>
<body>
  <h1>Find your Current location</h1>
<button onclick="getlocation()">Click me</button>
<div id="location"></div>
```

```
<script>
  var x= document.getElementById("location");
  function getlocation() {
      if(navigator.geolocation){
        navigator.geolocation.getCurrentPosition(showPosition)
        }
      else
      {
         alert("Sorry! your browser is not supporting")
      } }

   function showPosition(position){
    var x = "Your current location is (" + "Latitude: " + position.coords.latitude + ", " + "Longitude: " +    position.coords.longitude + ")";
            document.getElementById("location").innerHTML = x;
   }
</script>
</body>
</html>
```

Output:


Find your Current location

Explanation:

- First checking the browser support
- Getting current position with getCurrentPosition()
- Getting latitude and longitude values with showPosition() method which is call back method of getCurrentPosition().

Handling Errors and Rejections: Using an Error callback function

The second parameter of getCurrentPosition is an error Callback function. It is an optional parameter and used to handle errors and user rejection while getting the user's location.

Following are the possible options for invoking the error call back function:

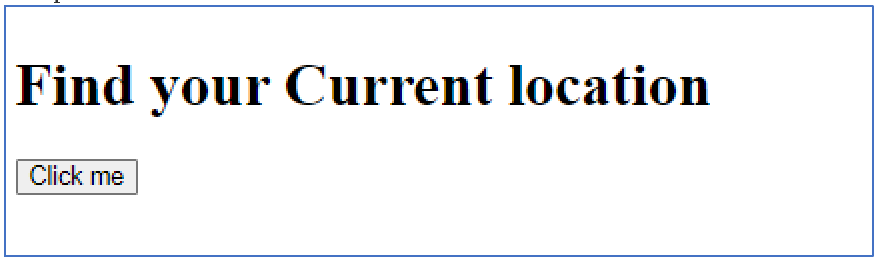
- Unknown random error Occurred
- If the user has denied for sharing location

- Location information is not available
- Request for location is timed-out.

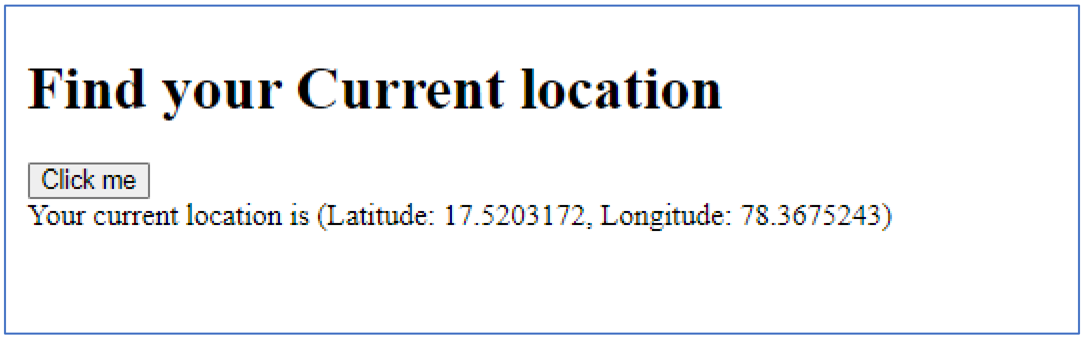
Example

```
function showError(error) {
        switch(error.code){
            case error.PERMISSION_DENIED:
           alert("User denied the request for Geolocation API.");
            break;
        case error.POSITION_UNAVAILABLE:
            alert("USer location information is unavailable.");
           break;
         case error.TIMEOUT:
           alert("The request to get user location timed out.");
            break;
        case error.UNKNOWN_ERROR:
            alert("An unknown error occurred.");
           break;
    }
        }
```

Output:



Find your Current location



Find your Current location

Your current location is (Latitude: 17.5203172, Longitude: 78.3675243)

Displaying location on Google Map

Till now, we have seen how to show your location using latitude and longitude values, but it is not sufficient. Hence we can also show the exact location on Google map with this API.

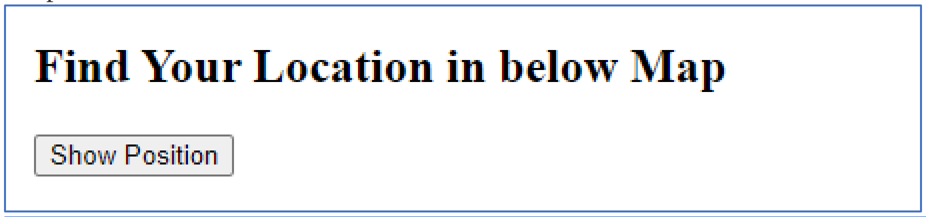
Following example showing the location using Google Map.

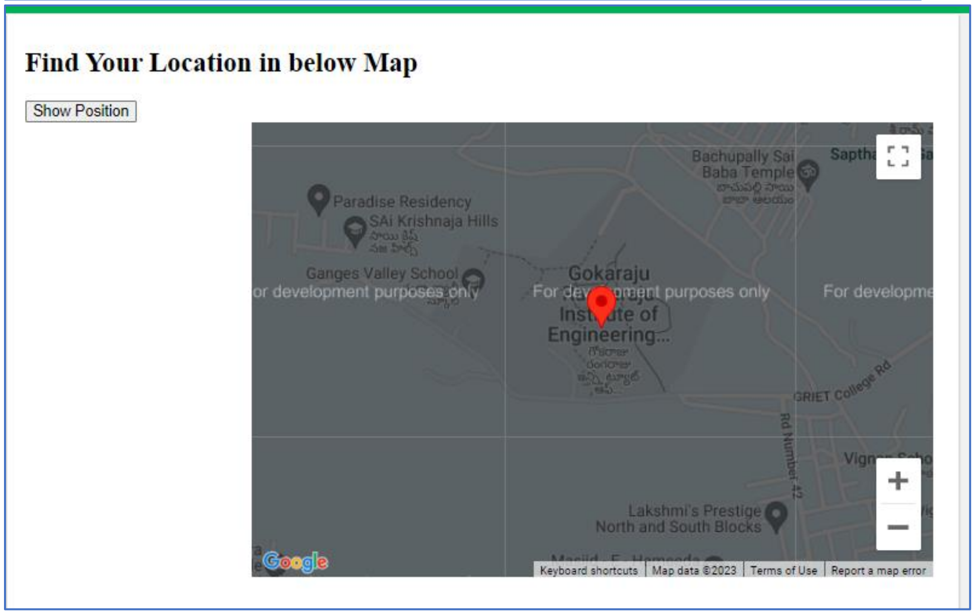
Example

```
<!DOCTYPE html>
```

```
<html>
   <head>
    <title>Geolocation API</title>
   </head>
   <body>
     <h2>Find Your Location in below Map</h2>
      <button onclick="getlocation();"> Show Position</button>
      <div id="demo" style="width: 600px; height: 400px; margin-left: 200px;"></div>

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

      <script type="text/javascript">
      function getlocation(){
         if(navigator.geolocation){
            navigator.geolocation.getCurrentPosition(showPos, showErr);
         }
         else{
            alert("Sorry! your Browser does not support Geolocation API")
         }
      }
      //Showing Current Poistion on Google Map
      function showPos(position){
         latt = position.coords.latitude;
         long = position.coords.longitude;
         var lattlong = new google.maps.LatLng(latt, long);
         var myOptions = {
            center: lattlong,
            zoom: 15,
            mapTypeControl: true,
            navigationControlOptions: {style:google.maps.NavigationControlStyle.SMALL}
         }
         var maps = new google.maps.Map(document.getElementById("demo"), myOptions);
         var markers =
         new google.maps.Marker({position:lattlong, map:maps, title:"You are here!"});
      }

      //Handling Error and Rejection
          function showErr(error) {
           switch(error.code){
           case error.PERMISSION_DENIED:
          alert("User denied the request for Geolocation API.");
           break;
          case error.POSITION_UNAVAILABLE:
          alert("USer location information is unavailable.");
         break;
         case error.TIMEOUT:
         alert("The request to get user location timed out.");
         break;
        case error.UNKNOWN_ERROR:
         alert("An unknown error occurred.");
         break;
        }
      }       </script>
   </body>
</html>
```

Output:





Location properties

The getCurrentPosition() method of Geolocation API returns callback methods which retrieve the user location information. This callback method returns a Position Object which contains all location information and specifies different properties. It always returns latitude and longitude properties, but the following table describes some other properties of Position object.

Properties	**Description**
coords.latitude	It returns latitude of user location as a decimal number.
coords.longitude	It returns longitude of user location as a decimal number.
coords.altitude	It returns altitude in meters above the sea level (Only if available).
coords.accuracy	It returns the accuracy of the user's position.
coords.altitudeAccuracy	It returns the altitude accuracy of user location. (If available)
coords.heading	It returns headings as degree clockwise from North. (If available)
coords.speed	It returns the speed in meter per seconds. (If available).

timestamp	It returns data or time of response. (If available).

Watching the current location:

If we want to know the user location while he is moving and want accurate location at every changed position, then it can be achieved by using watchPosition() callback function.

This function has all three parameters which getCurrentPosition() contains.

Syntax:

```
1.   var id = navigator.geolocation.watchPosition(success[, error[, options]])
```

The watchPosition() method returns an ID that can be used to uniquely identifying the user?s position, and this ID can also be used with clearWatch() method to stop watching the location.

Syntax:

```
navigator.geolocation.clearWatch(id);
```

Chapter II CSS3:

1. Basic Styling
2. Positioning & Border Image
3. Pseudo Classes
4. Colors
5. Backgrounds & Gradients
6. Text & Box Shadows
7. Transitions & Animation
8. Columns & Flexbox

1. Basic Styling:

CSS (Cascading Style Sheets) is the code that styles web content. *CSS basics* walks through what you need to get started. We'll answer questions like: How do I make text red? How do I make content display at a certain location in the (webpage) layout? How do I decorate my webpage with background images and colors?

Like HTML, CSS is not a programming language. It's not a markup language either. **CSS is a style sheet language.** CSS is what you use to selectively style HTML elements. For example, this CSS selects paragraph text, setting the color to red:

```
p {
color: red;
}
```

Let's try it out! Using a text editor, paste the three lines of CSS (above) into a new file. Save the file as style.css in a directory named styles.

To make the code work, we still need to apply this CSS (above) to your HTML document. Otherwise, the styling won't change the appearance of the HTML. (If you haven't been following our project, pause here to read Dealing with files and HTML basics.)

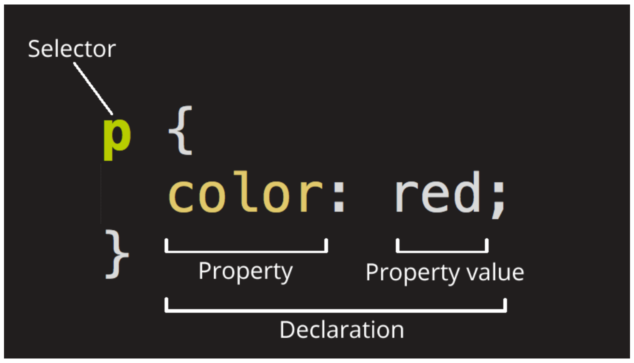
1. Open your index.html file. Paste the following line in the head (between the <head> and </head> tags):
2. <link href="styles/style.css" rel="stylesheet" />

   Copy to Clipboard

3. Save index.html and load it in your browser. You should see something like this:

Anatomy of a CSS ruleset

Let's dissect the CSS code for red paragraph text to understand how it works:



The whole structure is called a **ruleset**. (The term *ruleset* is often referred to as just *rule*.) Note the names of the individual parts:

Selector

This is the HTML element name at the start of the ruleset. It defines the element(s) to be styled (in this example, <p> elements). To style a different element, change the selector.

Declaration

This is a single rule like color: red;. It specifies which of the element's **properties** you want to style.

Properties

These are ways in which you can style an HTML element. (In this example, color is a property of the <p> elements.) In CSS, you choose which properties you want to affect in the rule.

Property value

To the right of the property—after the colon—there is the **property value**. This chooses one out of many possible appearances for a given property. (For example, there are many color values in addition to red.)

Note the other important parts of the syntax:

- Apart from the selector, each ruleset must be wrapped in curly braces. ({})
- Within each declaration, you must use a colon (:) to separate the property from its value or values.
- Within each ruleset, you must use a semicolon (;) to separate each declaration from the next one.

To modify multiple property values in one ruleset, write them separated by semicolons, like this:

```
p {
  color: red;
  width: 500px;
  border: 1px solid black;
}
```

Selecting multiple elements

You can also select multiple elements and apply a single ruleset to all of them. Separate multiple selectors by commas. For example:

```
p,
li,
h1 {
  color: red;
}
```

Copy to Clipboard

Different types of selectors

There are many different types of selectors. The examples above use **element selectors**, which select all elements of a given type. But we can make more specific selections as well. Here are some of the more common types of selectors:

Selector name	**What does it select**	**Example**
Element selector (sometimes called a tag or type selector)	All HTML elements of the specified type.	p selects <p>
ID selector	The element on the page with the specified ID. On a given HTML page, each id value should be unique.	#my-id selects <p id="my-id"> or <a id="my-id">
Class selector	The element(s) on the page with the specified class. Multiple instances of the same class can appear on a page.	.my-class selects <p class="my-class"> and <a class="my-class">
Attribute selector	The element(s) on the page with the specified attribute.	img[src] selects <img src="myimage.png"> but not <img>
Pseudo-class selector	The specified element(s), but only when in the specified state. (For example, when a cursor hovers over a link.)	a:hover selects <a>, but only when the mouse pointer is hovering over the link.

2. Positioning and border image

2.1 Positioning:

The **CSS position property** is used *to set position for an element*. it is also used to place an element behind another and also useful for scripted animation effect.

You can position an element using the top, bottom, left and right properties. These properties can be used only after position property is set first. A position element's computed position property is relative, absolute, fixed or sticky.

Let's have a look at following CSS positioning:

1. CSS Static Positioning
2. CSS Fixed Positioning
3. CSS Relative Positioning
4. CSS Absolute Positioning

---

1) CSS Static Positioning

This is a by default position for HTML elements. It always positions an element according to the normal flow of the page. It is not affected by the top, bottom, left and right properties.

2) CSS Fixed Positioning

The fixed positioning property helps to put the text fixed on the browser. This fixed test is positioned relative to the browser window, and doesn't move even you scroll the window.

```
<!DOCTYPE html>
<html>
<head>
<style>
p.pos_fixed {
    position: fixed;
    top: 50px;
    right: 5px;
    color: blue;
}
</style>
</head>
<body>

<p>Some text...</p><p>Some text...</p><p>Some text...</p><p>........</p><p>.... ...</p
><p>........</p><p>........</p><p>........</p><p>........</p>
<p>........ </p><p>........</p><p>........</p><p>........</p><p>........</p>
<p>........</p><p>........</p><p>Some text...</p><p>Some text...</p><p>Some text...</p>
<p class="pos_fixed">This is the fix positioned text.</p>
</body>
</html>
```

3) CSS Relative Positioning

The relative positioning property is used to set the element relative to its normal position.

```
<!DOCTYPE html>
<html>
<head>
<style>
h2.pos_left {
    position: relative;
    left: -30px;
}
h2.pos_right {
    position: relative;
    left: 30px;
}
</style>
</head>
<body>
<h2>This is a heading with no position</h2>
<h2 class="pos_left">This heading is positioned left according to its normal position</h2
>
<h2 class="pos_right">This heading is positioned right according to its normal position</h2>
<p>The style "left:-
30px" subtracts 30 pixels from the element's original left position.</p>
<p>The style "left:30px" adds 30 pixels to the element's original left position.</p>
</body>
</html>
```

4) CSS Absolute Positioning

The absolute positioning is used to position an element relative to the first parent element that has a position other than static. If no such element is found, the containing block is HTML.

With the absolute positioning, you can place an element anywhere on a page.

```
<!DOCTYPE html>
<html>
<head>
<style>
h2 {
    position: absolute;
    left: 150px;
    top: 250px;
}
</style>
</head>
<body>
<h2>This heading has an absolute position</h2>
<p> The heading below is placed 150px from the left and 250px from the top of the page.</p>
</body>
</html>
```

All CSS Position Properties

No.	property	description	values
1)	bottom	It is used to set the bottom margin edge for a positioned box.	auto, length, %, inherit
2)	clip	It is used to clip an absolutely positioned element.	shape, auto, inherit
3)	cursor	It is used to specify the type of cursors to be displayed.	url, auto, crosshair, default, pointer, move, e-resize, ne-resize, nw-resize, n-resize, se-resize, sw-resize, s-resize, w-resize, text, wait, help
4)	left	It sets a left margin edge for a positioned box.	auto, length, %, inherit
5)	overflow	This property is used to define what happens if content overflow an element's box.	auto, hidden, scroll, visible, inherit
6)	position	It is used to specify the type of positioning for an element.	absolute, fixed, relative, static, inherit
7)	right	It is used to set a right margin edge for a positioned box.	auto, length, %, inherit
8)	top	It is used to set a top margin edge for a positioned box.	auto, length, %, inherit
9)	z-index	It is used to set stack order of an element.	number, auto, inherit

2.2 CSS border image:

With the CSS border-image property, you can set an image to be used as the border around an element.

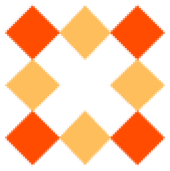
CSS border-image Property

The CSS border-image property allows you to specify an image to be used instead of the normal border around an element.

The property has three parts:

1. The image to use as the border
2. Where to slice the image
3. Define whether the middle sections should be repeated or stretched

We will use the following image (called "border.png"):



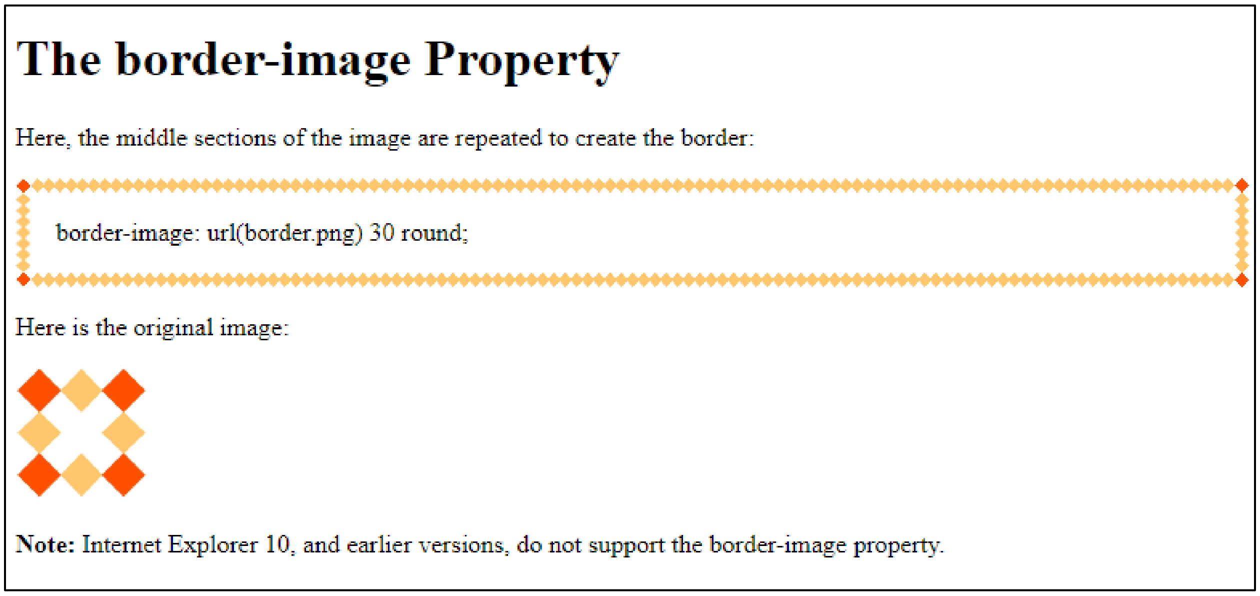
The border-image property takes the image and slices it into nine sections, like a tic-tac-toe board. It then places the corners at the corners, and the middle sections are repeated or stretched as you specify.

Note: For border-image to work, the element also needs the border property set!

Example

```
#borderimg {
  border: 10px solid transparent;
  padding: 15px;
  border-image: url(border.png) 30 round;
}
```

Output:



The border-image Property

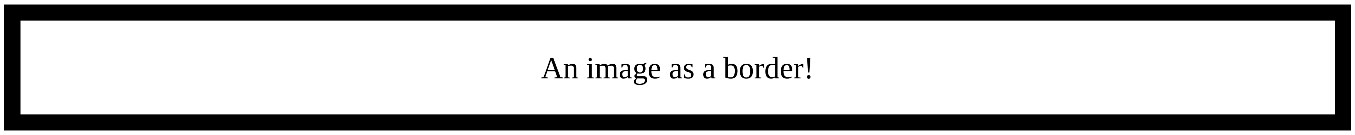
Here, the middle sections of the image are repeated to create the border:

border-image: url(border.png) 30 round;

Here is the original image:

Note: Internet Explorer 10, and earlier versions, do not support the border-image property.

Here, the middle sections of the image are stretched to create the border:



An image as a border!

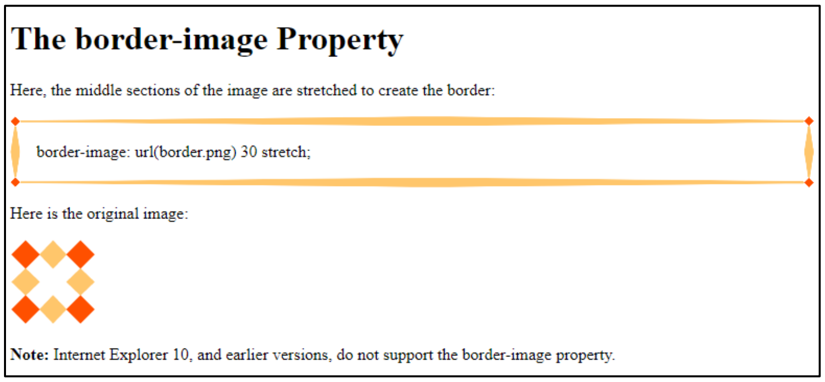
Here is the code:

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
#borderimg {
  border: 10px solid transparent;
  padding: 15px;
  border-image: url(border.png) 30 stretch;
}
</style>
</head>
<body>

<h1>The border-image Property</h1>

<p>Here, the middle sections of the image are stretched to create the border:</p>
<p id="borderimg">border-image: url(border.png) 30 stretch;</p>
```

```
<p>Here is the original image:</p><img src="border.png">
<p><strong>Note:</strong> Internet Explorer 10, and earlier versions, do not support the border-
image property.</p>

</body>
</html>
```

Output:



The border-image Property

Here, the middle sections of the image are stretched to create the border:

border-image: url(border.png) 30 stretch;

Here is the original image:

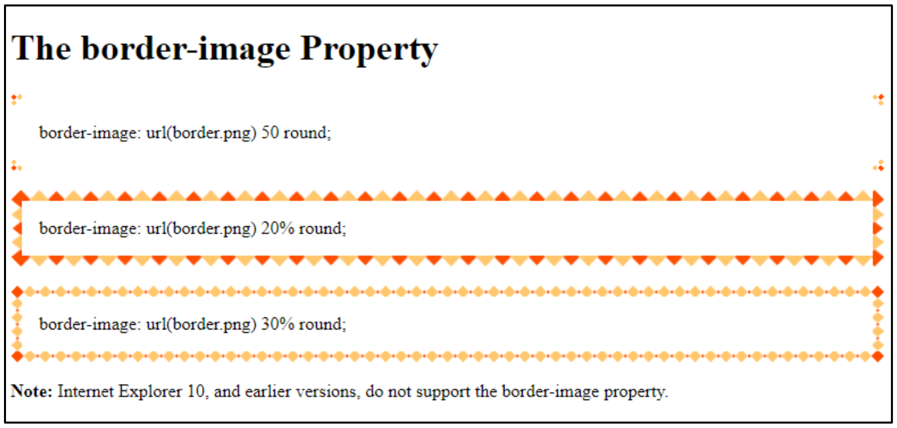
Note: Internet Explorer 10, and earlier versions, do not support the border-image property.

Tip: The border-image property is actually a shorthand property for the border-image-source, border-image-slice, border-image-width, border-image-outset and border-image-repeat properties.

CSS border-image - Different Slice Values

```
<!DOCTYPE html>
<html>
<head>
<style>
#borderimg1 {
  border: 10px solid transparent;
  padding: 15px;
  border-image: url(border.png) 50 round;
}

#borderimg2 {
  border: 10px solid transparent;
  padding: 15px;
  border-image: url(border.png) 20% round;
}

#borderimg3 {
  border: 10px solid transparent;
  padding: 15px;
  border-image: url(border.png) 30% round;
}
</style>
</head>
<body>

<h1>The border-image Property</h1>

<p id="borderimg1">border-image: url(border.png) 50 round;</p>
<p id="borderimg2">border-image: url(border.png) 20% round;</p>
<p id="borderimg3">border-image: url(border.png) 30% round;</p>

<p><strong>Note:</strong> Internet Explorer 10, and earlier versions, do not support the
border-image property.</p>

</body>
</html>
```

Output:



The border-image Property

border-image: url(border.png) 50 round;

border-image: url(border.png) 20% round;

border-image: url(border.png) 30% round;

Note: Internet Explorer 10, and earlier versions, do not support the border-image property.

CSS Border Image Properties

Property	Description
border-image	A shorthand property for setting all the border-image-* properties
border-image-source	Specifies the path to the image to be used as a border
border-image-slice	Specifies how to slice the border image

border-image-width	Specifies the widths of the border image
border-image-outset	Specifies the amount by which the border image area extends beyond the border box
border-image-repeat	Specifies whether the border image should be repeated, rounded or stretched

3. Pseudo classes

A pseudo-class can be defined as a keyword which is combined to a selector that defines the special state of the selected elements. It is added to the selector for adding an effect to the existing elements based on their states. For example, The **":hover"** is used for adding special effects to an element when the user moves the cursor over the element.

The names of the pseudo-class are not case-sensitive.

Syntax

A pseudo-class starts with a colon **(:).** Let's see its syntax.

```
selector: pseudo-class {
  property: value;
}
```

Although there are various CSS pseudo-classes, here we are going to discuss some of the most commonly used pseudo-classes. The widely used CSS classes are tabulated as follows:

pseudo-class	Description
:active	IIt is used to add style to an active element.
:hover	IIt adds special effects to an element when the user moves the mouse pointer over the element.
:link	IIt adds style to the unvisited link.

:visited	IIt adds style to a visited link.
:lang	IIt is used to define a language to use in a specified element.
:focus	IIt selects the element which is focused by the user currently.
:first-child	It adds special effects to an element, which is the first child of another element.

Let's discuss the above pseudo-classes, along with an example.

:hover pseudo-class

This pseudo-class adds a special style to an element when the user moves the cursor over it. If you want to make it effective, it must be applied after the **":link"** and **":visited"** pseudo-classes.

Example

```
<html>
  <head>
    <style>
      body{
    text-align:center;
    }
    h1:hover{
    color:red;
    }
    h2:hover{
    color:blue;
    }
    </style>
  </head>

  <body>
    <h1>Hello world </h1>
    <h2>This is an example of :hover pseudo class</h2>
  </body>
</html>
```

:active pseudo-class

It applies when the elements are clicked or activated. It selects the activated element.

We can understand it with the example given below.

Example

```
<!DOCTYPE html>
<html>
<head>
   <style>
    body{
   text-align:center;
    }
   a:active{
       color: yellow;
   }
   h1, h2, h3{
      color:red; ;
   }
   </style>
</head>

<body>
   <h1>Hello World</h1>
   <h2>The :active pseudo-class</h2>
   <h3>Click the following link to see the effect</h3>
   <a href="#">Click the link</a>
</body>
</html>
```

:visited pseudo-class

It selects the visited links and adds special styles to them. Its possible values can be any color name in a valid format.

Example

```
<!DOCTYPE html>
<html>
<head>
   <style>
    body{
   text-align:center;
    }
   a:visited{
       color: red;
   }

   </style>
</head>

<body>
```

```
    <h1>Hello World</h1>
    <h2>The :visited pseudo-class</h2>
    <h3>Click the following link to see the effect</h3>
    <a href="#">Click the link</a>
</body>
</html>
```

:lang pseudo-class

It is helpful in documents that require multiple languages. It allows us to define special rules for different languages.

Example

In this example, we specify **p:lang(fr)** which selects the elements having attribute **lang="fr"**.

```
<!DOCTYPE HTML>
 <html>
  <head>
    <style>
    body{
    text-align:center;
    }
       p:lang(fr)
    {
       font-family:Verdana;
        color:blue;

    }
   </style>
 </head>
<body>

  <p>Without :lang pseudo class</p>
  <p lang="fr">With :lang pseudo class with the value fr</p>
</body>
</html>
```

:focus pseudo-class

It selects the elements that are currently focused on by the user. It is generally used in input elements of the forms and triggers when the user clicks on it.

Example

```
<!DOCTYPE HTML>
<html>
 <style>
 form{
 text-align:center;
 }
 input:focus{
        border:5px solid lightblue;
         box-shadow:10px 10px 10px black;
        color: blue;
         width:300px;
        }
  </style>
 <body>
 <form>
   <h1>Name: <input type="text" value="Enter your name"></h1>
  </form>
</body>
</html>
```

:first-child pseudo-class

It matches a particular element, which is the first child of another element and adds a special effect to the corresponding element.

Note: We have to declare a <!DOCTYPE> at the top of the document to make ":first-child" pseudo-class to work in IE8 and its earlier versions.

The following illustration explains it more clearly.

Example

```
<!DOCTYPE HTML>
<html>
  <head>
    <style>
         h1:first-child {
          text-indent: 200px;
  color:blue;
        }
     </style>
  </head>

  <body>

    <div>
```

```
        <h1>It is the first heading in div. It will be indented, and its color will be blue.</h1
>
        <h1>It is the Second heading in div, which will not be affected.</h1>
      </div>
   </body>
</html>
```

The simple tooltip hover

A tooltip specifies extra information about something when the user moves the cursor over the element. Let's create a tooltip by using the **":hover"** pseudo-class.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
body{
text-align:center;
}
h2{
  display: none;
  background-color: red;
  color:white;
  padding: 20px;
}
div{
font-size:40px;
}
div:hover h2 {
  display: block;
}
</style>
</head>
<body>
<h1>Move your mouse to the below text to see the effect</h1>
<div>Hello World
  <h2>Welcome to the javaTpoint</h2>
</div>

</body>
</html>
```

CSS classes and pseudo-classes

The classes in CSS can be combined with **pseudo-classes.** We can write it as-

Learn more

Syntax

```
selector.class: pseudo-class {
  property: value;
}
```

We can understand it with the following example.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
body{
text-align:center;
}
div.hello:hover {
  color: red;
  font-size:40px;
}
</style>
</head>
<body>
<h1>CSS Classes and pseudo-classes</h1>
<h2>Move your cursor to the below text</h2>
<div class="hello">Hello World</p>

</body>
</html>
```

4. CSS Colors

CSS3 has Supported additional color properties as follows −

- RGBA colors
- HSL colors
- HSLA colors
- Opacity

RGBA stands for **Red Green Blue Alpha**.It is an extension of CSS2,Alpha specifies the opacity of a color and parameter number is a numerical between 0.0 to 1.0. A Sample syntax of RGBA as shown below −

```
#d1 {background-color: rgba(255, 0, 0, 0.5);}
#d2 {background-color: rgba(0, 255, 0, 0.5);}
```

```
#d3 {background-color: rgba(0, 0, 255, 0.5);}
```

HSL stands for **hue, saturation, lightness**.Here Huge is a degree on the color wheel, saturation and lightness are percentage values between 0 to 100%. A Sample syntax of HSL as shown below −

```
#g1 {background-color: hsl(120, 100%, 50%);}
#g2 {background-color: hsl(120, 100%, 75%);}
#g3 {background-color: hsl(120, 100%, 25%);}
```

HSLA stands for **hue, saturation, lightness and alpha**. Alpha value specifies the opacity as shown RGBA. A Sample syntax of HSLA as shown below −

```
#g1 {background-color: hsla(120, 100%, 50%, 0.3);}
#g2 {background-color: hsla(120, 100%, 75%, 0.3);}
#g3 {background-color: hsla(120, 100%, 25%, 0.3);}
```

opacity is a thinner paints need black added to increase opacity. A sample syntax of opacity is as shown below −

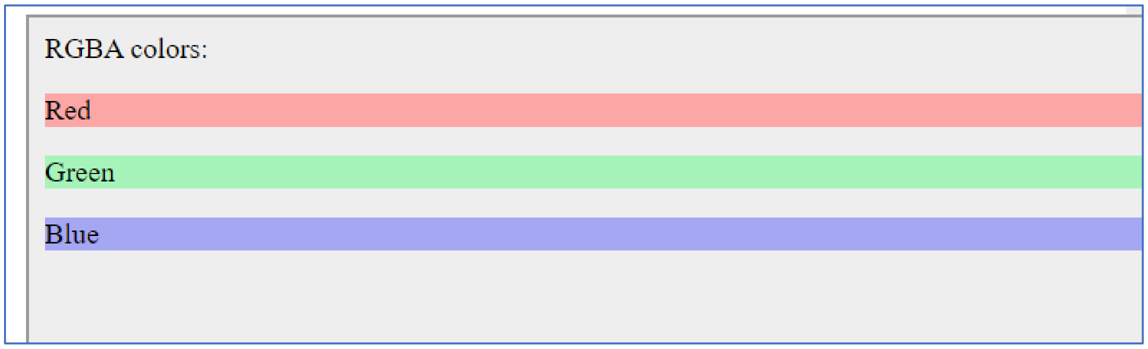
```
#g1 {background-color:rgb(255,0,0);opacity:0.6;}
#g2 {background-color:rgb(0,255,0);opacity:0.6;}
#g3 {background-color:rgb(0,0,255);opacity:0.6;}
```

Programs:

The following example shows rgba color property.

```
<html>
  <head>
    <style>
      #p1 {background-color:rgba(255,0,0,0.3);}
      #p2 {background-color:rgba(0,255,0,0.3);}
      #p3 {background-color:rgba(0,0,255,0.3);}
    </style>
  </head>

  <body>
    <p>RGBA colors:</p>
    <p id = "p1">Red</p>
    <p id = "p2">Green</p>
    <p id = "p3">Blue</p>
  </body>
</html>
```

Output:

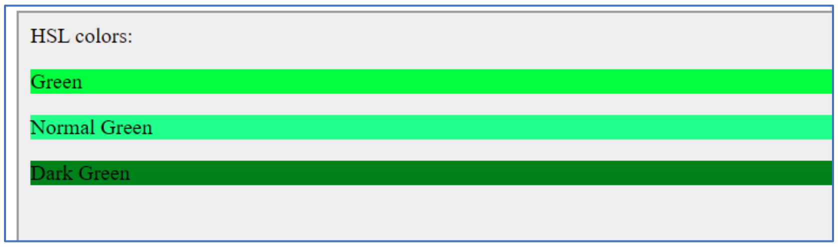


```
RGBA colors:

Red

Green

Blue
```

The following example shows HSL color property.

```
<html>
  <head>
    <style>
      #g1 {
                  background-color:hsl(120, 100%, 50%);
            }
      #g2 {background-color:hsl(120,100%,75%);}
      #g3 {background-color:hsl(120,100%,25%);}
    </style>
  </head>

  <body>
    <p>HSL colors:</p>
    <p id = "g1">Green</p>
    <p id = "g2">Normal Green</p>
    <p id = "g3">Dark Green</p>
  </body>
</html>
```

Output:

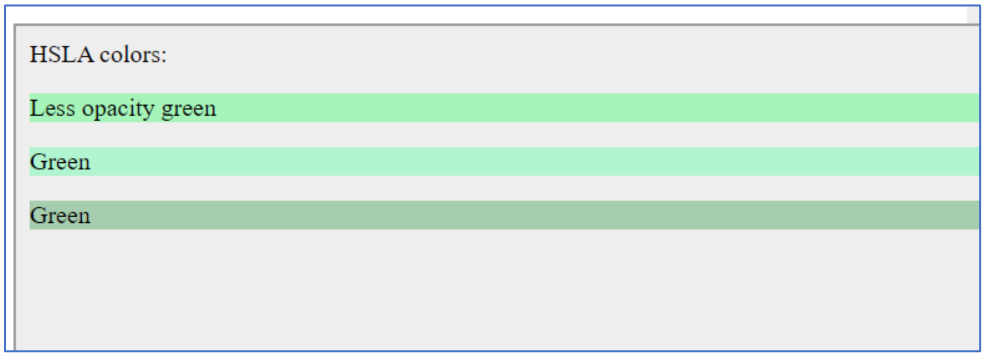


```
HSL colors:

Green

Normal Green

Dark Green
```

The following example shows HSLA color property.

```
<html>
  <head>
    <style>
      #d1 {background-color:hsla(120,100%,50%,0.3);}
      #d2 {background-color:hsla(120,100%,75%,0.3);}
      #d3 {background-color:hsla(120,100%,25%,0.3);}
    </style>
  </head>

  <body>
    <p>HSLA colors:</p>
    <p id = "d1">Less opacity green</p>
    <p id = "d2">Green</p>
    <p id = "d3">Green</p>
  </body>
</html>
```

Output:

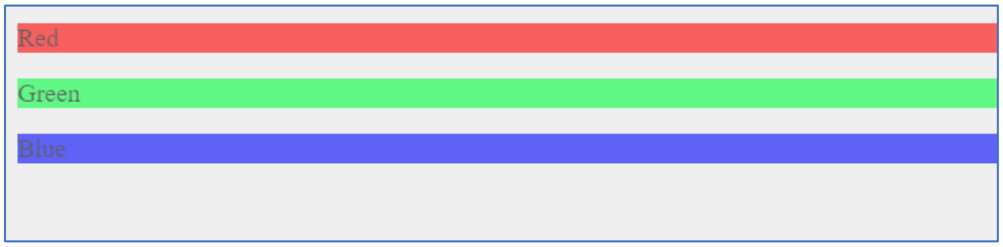


```
HSLA colors:

Less opacity green

Green

Green
```

The following example shows Opacity property.

```
<html>
  <head>
    <style>
      #m1 {background-color:rgb(255,0,0);opacity:0.6;}
      #m2 {background-color:rgb(0,255,0);opacity:0.6;}
      #m3 {background-color:rgb(0,0,255);opacity:0.6;}
    </style>
  </head>

  <body>
    <p>HSLA colors:</p>
    <p id = "m1">Red</p>
    <p id = "m2">Green</p>
    <p id = "m3">Blue</p>
  </body>
</html>
```

Output:



Red

Green

Blue

5. Backgrounds and gradient

5.1 background:

how to add multiple background images to one element.

You will also learn about the following properties:

- background-size
- background-origin
- background-clip

CSS allows you to add multiple background images for an element, through the background-image property.

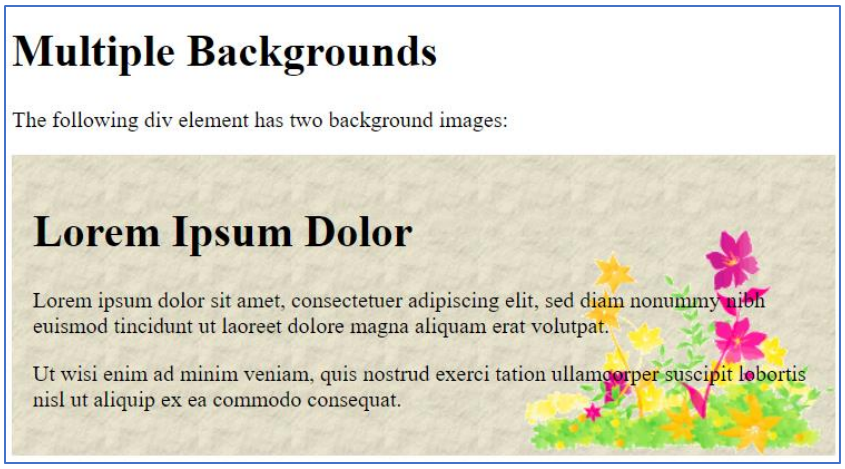
The different background images are separated by commas, and the images are stacked on top of each other, where the first image is closest to the viewer.

The following example has two background images, the first image is a flower (aligned to the bottom and right) and the second image is a paper background (aligned to the top-left corner):

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
#example1 {
  background-image: url(img_flwr.gif), url(paper.gif);
  background-position: right bottom, left top;
  background-repeat: no-repeat, repeat;
  padding: 15px;
}
</style>
</head>
<body>

<h1>Multiple Backgrounds</h1>
<p>The following div element has two background images:</p>

<div id="example1">
  <h1>Lorem Ipsum Dolor</h1>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut
aliquip ex ea commodo consequat.</p>
</div>

</body>
</html>
```

Output:



Multiple Backgrounds

The following div element has two background images:

Lorem Ipsum Dolor

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

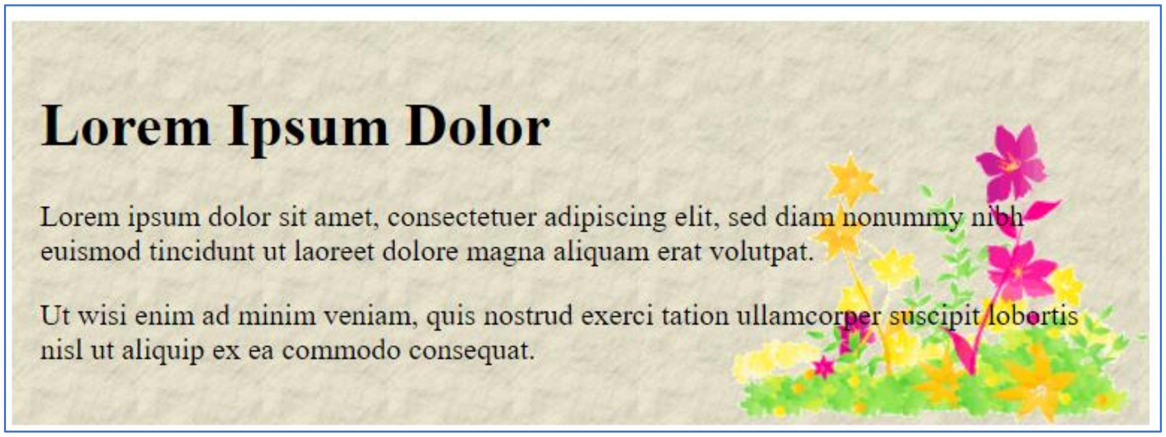
Multiple background images can be specified using either the individual background properties (as above) or the background shorthand property.

The following example uses the background shorthand property (same result as example above):

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
#example1 {
  background: url(img_flwr.gif) right bottom no-repeat, url(paper.gif) left top repeat;
  padding: 15px;
}
</style>
</head>
<body>

<div id="example1">
  <h1>Lorem Ipsum Dolor</h1>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl
ut aliquip ex ea commodo consequat.</p>
</div>

</body>
</html>
```

Output:



Lorem Ipsum Dolor

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

CSS Background Size

The CSS background-size property allows you to specify the size of background images.

The size can be specified in lengths, percentages, or by using one of the two keywords: contain or cover.

The following example resizes a background image to much smaller than the original image (using pixels):

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
#example1 {
  border: 1px solid black;
  background: url(img_flwr.gif);
  background-size: 100px 80px;
  background-repeat: no-repeat;
  padding: 15px;
}

#example2 {
  border: 1px solid black;
  background: url(img_flwr.gif);
  background-repeat: no-repeat;
  padding: 15px;
}
</style>
</head>
<body>

<h1>The background-size Property</h1>

<p>Resized background-image:</p>
<div id="example1">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut
aliquip ex ea commodo consequat.</p>
</div>

<p>Original size of the background-image:</p>
<div id="example2">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut
aliquip ex ea commodo consequat.</p>
```

```
</div>

</body>
</html>
```

The two other possible values for background-size are contain and cover.

The contain keyword scales the background image to be as large as possible (but both its width and its height must fit inside the content area). As such, depending on the proportions of the background image and the background positioning area, there may be some areas of the background which are not covered by the background image.

The cover keyword scales the background image so that the content area is completely covered by the background image (both its width and height are equal to or exceed the content area). As such, some parts of the background image may not be visible in the background positioning area.

The following example illustrates the use of contain and cover:

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
.div1 {
  border: 1px solid black;
  height: 120px;
  width: 150px;
  background: url(img_flwr.gif);
  background-repeat: no-repeat;
  background-size: contain;
}

.div2 {
  border: 1px solid black;
  height: 120px;
  width: 150px;
  background: url(img_flwr.gif);
  background-repeat: no-repeat;
  background-size: cover;
}

.div3 {
  border: 1px solid black;
  height: 120px;
  width: 150px;
  background: url(img_flwr.gif);
  background-repeat: no-repeat;
}
```

```
</style>
</head>
<body>

<h1>The background-size Property</h1>

<h2>background-size: contain:</h2>
<div class="div1">
<p>Lorem ipsum dolor sit amet.</p>
</div>

<h2>background-size: cover:</h2>
<div class="div2">
<p>Lorem ipsum dolor sit amet.</p>
</div>

<h2>No background-size defined:</h2>
<div class="div3">
<p>Lorem ipsum dolor sit amet.</p>
</div>

<p>Original image:</p>
<img src="img_flwr.gif" alt="Flowers" width="224" height="162">

</body>
</html>
```

Define Sizes of Multiple Background Images

The background-size property also accepts multiple values for background size (using a comma-separated list), when working with multiple backgrounds.

The following example has three background images specified, with different background-size value for each image:

```
<!DOCTYPE html>
<html>
<head>
<style>
#example1 {
  background: url(img_tree.gif) left top no-repeat, url(img_flwr.gif) right bottom no-repeat,
url(paper.gif) left top repeat;
  padding: 15px;
  background-size: 50px, 130px, auto;
}
</style>
</head>
```

```
<body>

<div id="example1">
  <h1>Lorem Ipsum Dolor</h1>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut
aliquip ex ea commodo consequat.</p>
</div>

</body>
</html>
```

Full Size Background Image

Now we want to have a background image on a website that covers the entire browser window at all times.

The requirements are as follows:

- Fill the entire page with the image (no white space)
- Scale image as needed
- Center image on page
- Do not cause scrollbars

The following example shows how to do it; Use the <html> element (the <html> element is always at least the height of the browser window). Then set a fixed and centered background on it. Then adjust its size with the background-size property:

Example

```
html {
  background: url(img_man.jpg) no-repeat center fixed;
  background-size: cover;
}
```

Example

```
.hero-image {
  background: url(img_man.jpg) no-repeat center;
  background-size: cover;
  height: 500px;
  position: relative;
}
```

CSS background-origin Property

The CSS background-origin property specifies where the background image is positioned.

The property takes three different values:

- border-box - the background image starts from the upper left corner of the border
- padding-box - (default) the background image starts from the upper left corner of the padding edge.
- content-box - the background image starts from the upper left corner of the content

The following example illustrates the background-origin property:

Example

```
#example1 {
  border: 10px solid black;
  padding: 35px;
  background: url(img_flwr.gif);
  background-repeat: no-repeat;
  background-origin: content-box;
}
```

CSS background-clip Property

The CSS background-clip property specifies the painting area of the background.

The property takes three different values:

- border-box - (default) the background is painted to the outside edge of the border
- padding-box - the background is painted to the outside edge of the padding
- content-box - the background is painted within the content box

The following example illustrates the background-clip property:

Example

```
#example1 {
  border: 10px dotted black;
  padding: 35px;
  background: yellow;
  background-clip: content-box;
}
```

5.2 CSS Gradient:

CSS gradient is used to display smooth transition within two or more specified colors.

Why CSS Gradient

These are the following reasons to use CSS gradient.

- You don't have to use images to display transition effects.
- The download time and bandwidth usage can also be reduced.
- It provides better look to the element when zoomed, because the gradient is generated by the browser.

There are two types of gradient in CSS3.

1. Linear gradients
2. Radial gradients

---

1) CSS Linear Gradient

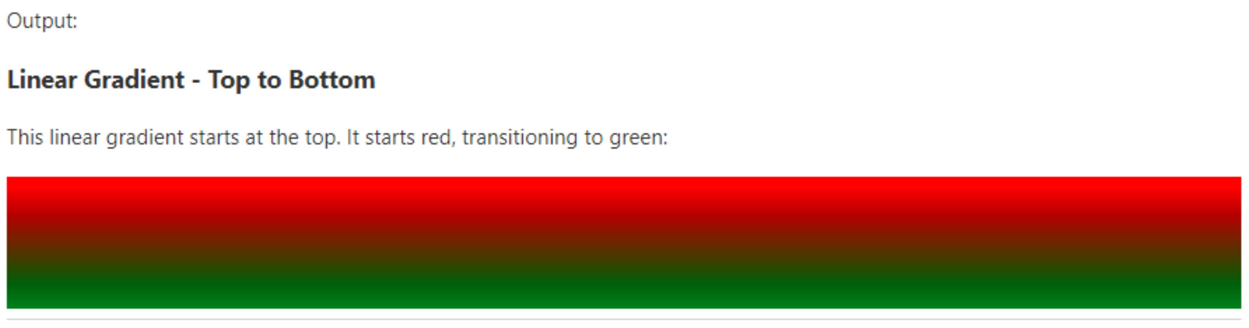
The CSS3 linear gradient goes up/down/left/right and diagonally. To create a CSS3 linear gradient, you must have to define two or more color stops. The color stops are the colors which are used to create a smooth transition. Starting point and direction can also be added along with the gradient effect.

```
background: linear-gradient (direction, color-stop1, color-stop2.....);
```

(i) CSS Linear Gradient: (Top to Bottom)

Top to Bottom Linear Gradient is the default linear gradient. Let's take an example of linear gradient that starts from top. It starts red and transitions to green.

```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {
    height: 100px;
    background: -webkit-linear-gradient(red, green); /* For Safari 5.1 to 6.0 */
    background: -o-linear-gradient(red, green); /* For Opera 11.1 to 12.0 */
    background: -moz-linear-gradient(red, green); /* For Firefox 3.6 to 15 */
    background: linear-gradient(red, green); /* Standard syntax (must be last) */
}
</style>
</head>
<body>
<h3>Linear Gradient - Top to Bottom</h3>
<p>This linear gradient starts at the top. It starts red, transitioning to green:</p>
<div id="grad1"></div>
</body>
</html>
```


Output:

Linear Gradient - Top to Bottom

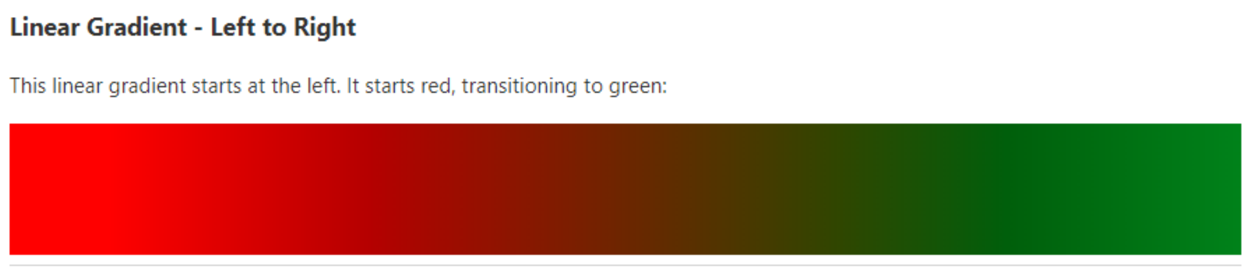
This linear gradient starts at the top. It starts red, transitioning to green:

(ii) CSS Linear Gradient: (Left to Right)

The following example shows the linear gradient that starts from left and goes to right. It starts red from left side and transitioning to green.

```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {
   height: 100px;
   background: -webkit-linear-gradient(left, red, green); /* For Safari 5.1 to 6.0 */
   background: -o-linear-gradient(right, red, green); /* For Opera 11.1 to 12.0 */
   background: -moz-linear-gradient(right, red, green); /* For Firefox 3.6 to 15 */
   background: linear-gradient(to right, red , green); /* Standard syntax (must be last) */
}
</style>
</head>
<body>
<h3>Linear Gradient - Left to Right</h3>
<p>This linear gradient starts at the left. It starts red, transitioning to green:</p>
<div id="grad1"></div>
</body>
</html>
```

Output:



Linear Gradient - Left to Right

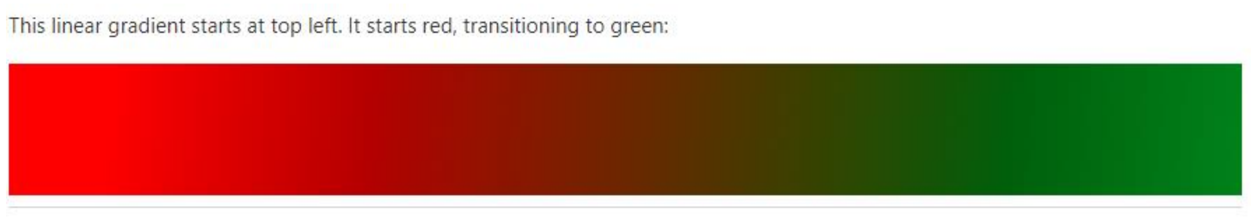
This linear gradient starts at the left. It starts red, transitioning to green:

(iii) CSS Linear Gradient: (Diagonal)

If you specify both the horizontal and vertical starting positions, you can make a linear gradient diagonal.

```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {
   height: 100px;
   background: -webkit-linear-gradient(left top, red , green); /* For Safari 5.1 to 6.0 */
   background: -o-linear-gradient(bottom right, red, green); /* For Opera 11.1 to 12.0 */
   background: -moz-linear-gradient(bottom right, red, green); /* For Firefox 3.6 to 15 */
   background: linear-gradient(to bottom right, red , green); /* Standard syntax (must be last) */
}
</style>
</head>
<body>
<h3>Linear Gradient - Diagonal</h3>
<p>This linear gradient starts at top left. It starts red, transitioning to green:</p>
<div id="grad1"></div>
</body>
</html>
```

Output:



This linear gradient starts at top left. It starts red, transitioning to green:

2) CSS Radial Gradient

You must have to define at least two color stops to create a radial gradient. It is defined by its center.

```
background: radial-gradient(shape size at position, start-color, ..., last-color);
```

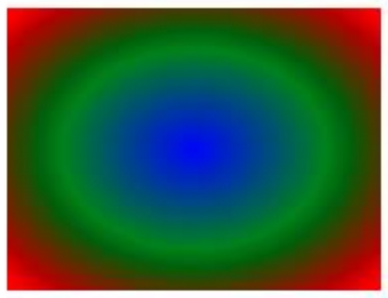
(i) CSS Radial Gradient: (Evenly Spaced Color Stops)

Evenly spaced color stops is a by default radial gradient. Its by default shape is eclipse, size is farthest- carner, and position is center.

```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {
    height: 150px;
    width: 200px;
    background: -webkit-radial-gradient(blue, green, red); /* Safari 5.1 to 6.0 */
    background: -o-radial-gradient(blue, green, red); /* For Opera 11.6 to 12.0 */
    background: -moz-radial-gradient(blue, green, red); /* For Firefox 3.6 to 15 */
    background: radial-gradient(blue, green, red); /* Standard syntax (must be last) */
}
</style>
</head>
<body>
<h3>Radial Gradient - Evenly Spaced Color Stops</h3>
<div id="grad1"></div>
</body>
</html>
```

Output:

Radial Gradient - Evenly Spaced Color Stops

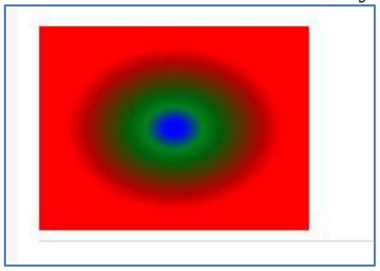


(ii) Radial Gradient: (Differently Spaced Color Stops)

```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {
    height: 150px;
    width: 200px;
    background: -webkit-radial-gradient(blue 5%, green 15%, red 60%); /* Safari 5.1 to 6.0 */
    background: -o-radial-
gradient(blue 5%, green 15%, red 60%); /* For Opera 11.6 to 12.0 */
    background: -moz-radial-gradient(blue 5%, green 15%, red 60%); /* For Firefox 3.6 to 15 */
    background: radial-
gradient(blue 5%, green 15%, red 60%); /* Standard syntax (must be last) */
}
</style>
</head>
<body>
<h3>Radial Gradient - Differently Spaced Color Stops</h3>
<div id="grad1"></div>
</body>
</html>
```

Output:

Radial Gradient - Differently Spaced Color Stops



6. Text shadow and box shadow

6.1 text shadow:

You can use the text-shadow property to apply the shadow effects on text. You can also apply multiple shadows to text using the same notation as box-shadow.

Syntax

text-shadow: h-shadow v-shadow blur-radius color| none | initial | inherit;

h-shadow: It is the required value. It specifies the position of the horizontal shadow and allows negative values.

v-shadow: It is also the required value that specifies the position of the vertical shadow. It does not allow negative values.

blur-radius: It is the blur-radius, which is an optional value. Its default value is 0.

color: It is the color of the shadow and also an optional value.

none: It is the default value, which means no shadow.

initial: It is used to set the property to its default value.

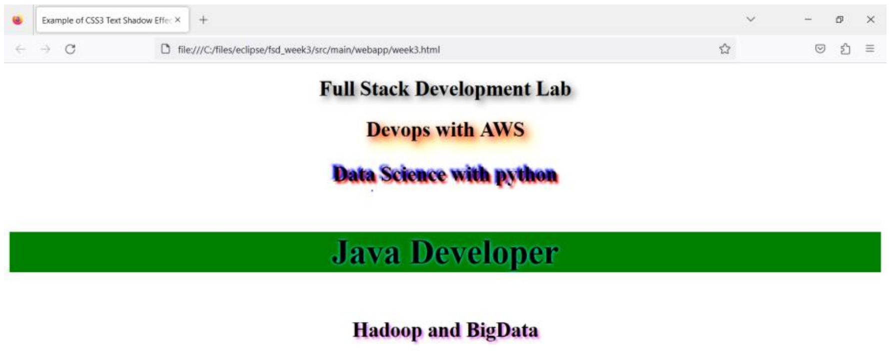
inherit: It simply inherits the property from its parent element.

Example

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Example of CSS3 Text Shadow Effects</title>
<style>
   h1 {
                    text-shadow: 5px 5px 10px #666;
                    font-size:30px;
                    text-align:center;
          }
          h2 {
                    text-shadow: 5px 5px 10px red, 10px 10px 20px yellow;
                    font-size:30px;
                    text-align:center;
          }
          h3{
      text-shadow: -3px -3px 3px blue, 3px 3px 3px red;
      font-size:30px;
```

```
        text-align:center;
      }
    h4{
        text-shadow: 0 0 .1em cyan;
        background-color: green;
        font-size:50px;
        text-align:center;
      }
    h5{
        text-shadow: 3px 3px 3px violet;
        font-size:30px;
        text-align:center;
      }
</style>
</head>
<body>
   <h1>Full Stack Development Lab</h1>
   <h2>Devops with AWS</h2>
   <h3>Data Science with python </h3>
   <h4>Java Developer</h4>
   <h5>Hadoop and BigData</h5>
</body>
</html>
```

Output:


Example of CSS3 Text Shadow Effe
file:///C:/files/eclipse/fsd_week3/src/main/webapp/week3.html

Full Stack Development Lab

Devops with AWS

Data Science with python

Java Developer

Hadoop and BigData

6.2 box-shadow

The box-shadow property can be used to add shadow to the element's boxes. You can even apply more than one shadow effects using a comma-separated list of shadows. The basic syntax of creating a box shadow can be given with:

box-shadow: offset-x offset-y blur-radius color.

The components of the box-shadow property have the following meaning:

- **offset-x** — Sets the horizontal offset of the shadow.

- **offset-y** — Sets the vertical offset of the shadow.
- **blur-radius** — Sets the blur radius. The larger the value, the bigger the blur and more the shadow's edge will be blurred. Negative values are not allowed.
- **color** — Sets the color of the shadow. If the color value is omitted or not specified, it takes the value of the color property.

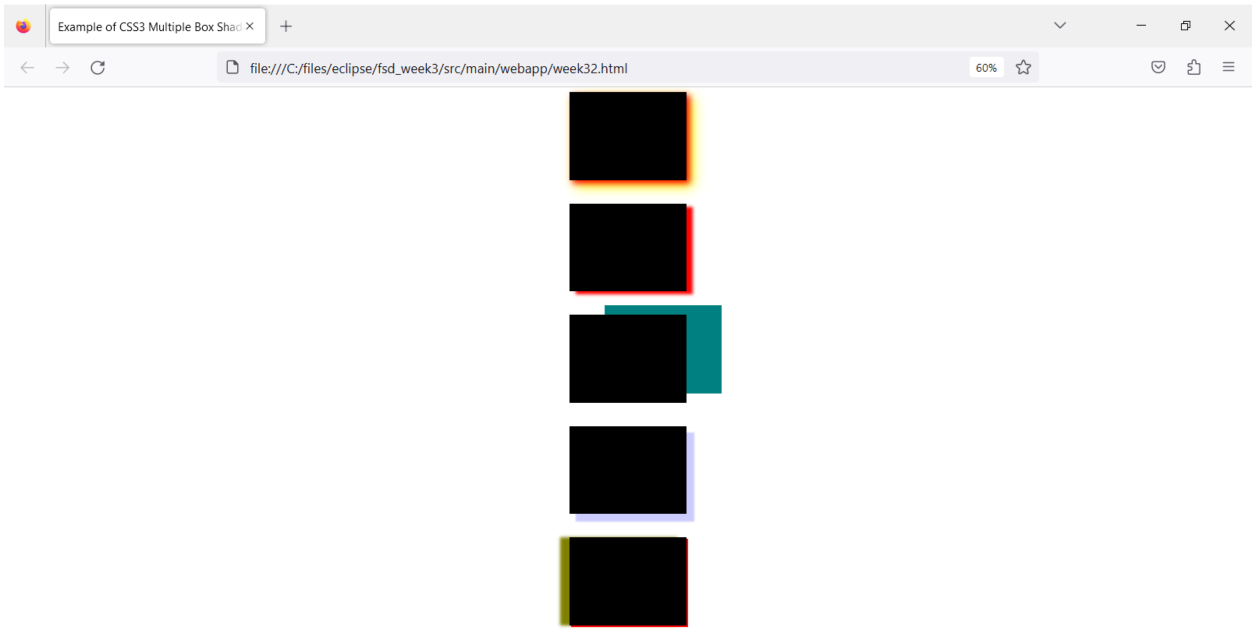
See the CSS3 box-shadow property to learn more about the other possible values.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Example of CSS3 Multiple Box Shadow Effects</title>
<style>
   .box{
                  width: 200px;
                  height: 150px;
                  background: #000;
                  box-shadow: 5px 5px 10px red, 10px 10px 20px yellow;

      }

   .box1{
                  width: 200px;
                  height: 150px;
                  background: #000;
                  box-shadow: 10px 5px 5px red;
         }
         .box2{
                  width: 200px;
                  height: 150px;
                  background: #000;
                  box-shadow:60px -16px teal;;
         }
         .box3{
                  width: 200px;
                  height: 150px;
                  background: #000;
                  box-shadow:12px 12px 2px 1px rgba(0, 0, 255, .2);
         }
         .box4{
                  width: 200px;
                  height: 150px;
                  background: #000;
```

```
                box-shadow:3px 3px red, -1em 0 .4em olive;
        }
</style>
</head>
<body>
<center>
   <div class="box"></div><br><br>
   <div class="box1"></div><br><br>
   <div class="box2"></div><br><br>
   <div class="box3"></div><br><br>
   <div class="box4"></div>
</body>
</center>
</html>
```

Output:



Example of CSS3 Multiple Box Shad
file:///C:/files/eclipse/fsd_week3/src/main/webapp/week32.html
60%

7. Transitions and animation

7.1 Transitions:

CSS transitions allows you to change property values smoothly, over a given duration.

In this week we will learn about the following properties:

- transition
- transition-delay
- transition-duration
- transition-property
- transition-timing-function

a) transition

The following example adds a transition effect for both the width and height property, with a duration of 2 seconds for the width and 4 seconds for the height:

Code:

```
<!DOCTYPE html>
<html>
<style>
div {
	width: 100px;
	height: 100px;
	background: red;
	transition: width 2s, height 4s;
}

div:hover {
	width: 300px;
	height: 300px;
}
</style>
<body>

	<h1>The transition Property</h1>

	<p>Hover over the div element below, to see the transition effect:</p>

	<div></div>

</body>
</html>
```

Output:

Practice in lab

b)transition-delay

The transition-delay property specifies a delay (in seconds) for the transition effect.The following example has a 1 second delay before starting:

Code:

```
<!DOCTYPE html>
<html>
<style>
div {
	width: 100px;
	height: 100px;
	background: red;
	transition: width 3s;
	transition-delay: 1s;
}

div:hover {
	width: 300px;
}
</style>
<body>

	<h1>The transition-delay Property</h1>

	<p>Hover over the div element below, to see the transition effect:</p>

	<div></div>

	<p>
		<b>Note:</b> The transition effect has a 1 second delay before
		starting.
	</p>

</body>
</html>
```

Output:
Practice in lab

c)

The transition-timing-function property specifies the speed curve of the transition effect.

The transition-timing-function property can have the following values:

- ease - specifies a transition effect with a slow start, then fast, then end slowly (this is default)
- linear - specifies a transition effect with the same speed from start to end
- ease-in - specifies a transition effect with a slow start
- ease-out - specifies a transition effect with a slow end
- ease-in-out - specifies a transition effect with a slow start and end
- cubic-bezier(n,n,n,n) - lets you define your own values in a cubic-bezier function

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
        width: 100px;
        height: 100px;
        background: red;
        transition-property: width;
        transition-duration: 2s;
        transition-timing-function: linear;
        transition-delay: 1s;
}

div:hover {
        width: 300px;
}
</style>
</head>
<body>

        <h1>The transition Properties Specified One by One</h1>

        <p>Hover over the div element below, to see the transition effect:</p>

        <div></div>

        <p>
                <b>Note:</b> The transition effect has a 1 second delay before
                starting.
        </p>

</body>
```

```
</html>
```

Output:
Practice in lab

7.2 Animations

An animation lets an element gradually change from one style to another.

You can change as many CSS properties you want, as many times as you want.

To use CSS animation, you must first specify some keyframes for the animation.

Keyframes hold what styles the element will have at certain times.

CSS Animation Properties

The following table lists the @keyframes rule and all the CSS animation properties:

Property	**Description**
@keyframes	Specifies the animation code
animation	A shorthand property for setting all the animation properties
animation-delay	Specifies a delay for the start of an animation
animation-direction	Specifies whether an animation should be played forwards, backwards or in alternate cycles
animation-duration	Specifies how long time an animation should take to complete one cycle
animation-fill-mode	Specifies a style for the element when the animation is not playing (before it starts, after it ends, or both)
animation-iteration-count	Specifies the number of times an animation should be played
animation-name	Specifies the name of the @keyframes animation
animation-play-state	Specifies whether the animation is running or paused

animation-timing-function	Specifies the speed curve of the animation

a) The @keyframes Rule

When you specify CSS styles inside the @keyframes rule, the animation will gradually change from the current style to the new style at certain times.

To get an animation to work, you must bind the animation to an element.

The following example binds the "example" animation to the <div> element. The animation will last for 4 seconds, and it will gradually change the background-color of the <div> element from "red" to "yellow":

Code:
```
<!DOCTYPE html>
<html>
<head>
<style>
div {
        width: 100px;
        height: 100px;
        background-color: red;
        animation-name: example;
        animation-duration: 4s;
}

@keyframes example {
        from {background-color: red;
}

to {
        background-color: yellow;
}
}
</style>
</head>
<body>

        <h1>CSS Animation</h1>

        <div></div>

        <p>
                <b>Note:</b> When an animation is finished, it goes back to its
```

```
                    original style.
          </p>

</body>
</html>
```

Note: The animation-duration property defines how long an animation should take to complete. If the animation-duration property is not specified, no animation will occur, because the default value is 0s (0 seconds).

b)

In the example above we have specified when the style will change by using the keywords "from" and "to" (which represents 0% (start) and 100% (complete)).

It is also possible to use percent. By using percent, you can add as many style changes as you like.

The following example will change the background-color of the <div> element when the animation is 25% complete, 50% complete, and again when the animation is 100% complete:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}

@keyframes example {
  0%   {background-color: red;}
  25%  {background-color: yellow;}
  50%  {background-color: blue;}
  100% {background-color: green;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>
```

```
<div></div>

<p><b>Note:</b> When an animation is finished, it goes back to its original style.</p>

</body>
</html>
```

c)

Delay an Animation

The `animation-delay` property specifies a delay for the start of an animation.

The following example has a 2 seconds delay before starting the animation:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
  position: relative;
  animation-name: example;
  animation-duration: 4s;
  animation-delay: 2s;
}

@keyframes example {
  0%   {background-color:red; left:0px; top:0px;}
  25%  {background-color:yellow; left:200px; top:0px;}
  50%  {background-color:blue; left:200px; top:200px;}
  75%  {background-color:green; left:0px; top:200px;}
  100% {background-color:red; left:0px; top:0px;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<p>The animation-delay property specifies a delay for the start of an animation. The following
example has a 2 seconds delay before starting the animation:</p>
```

```
<div></div>

</body>
</html>
```

d)

Set How Many Times an Animation Should Run

The animation-iteration-count property specifies the number of times an animation should run.

The following example will run the animation 3 times before it stops:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
  position: relative;
  animation-name: example;
  animation-duration: 4s;
  animation-iteration-count: 3;
}

@keyframes example {
  0%   {background-color:red; left:0px; top:0px;}
  25%  {background-color:yellow; left:200px; top:0px;}
  50%  {background-color:blue; left:200px; top:200px;}
  75%  {background-color:green; left:0px; top:200px;}
  100% {background-color:red; left:0px; top:0px;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<p>The animation-iteration-count property specifies the number of times an animation should
run. The following example will run the animation 3 times before it stops:</p>

<div></div>

</body>
```

```
</html>
```

e)

Run Animation in Reverse Direction or Alternate Cycles

The animation-direction property specifies whether an animation should be played forwards, backwards or in alternate cycles.

The animation-direction property can have the following values:

- normal - The animation is played as normal (forwards). This is default
- reverse - The animation is played in reverse direction (backwards)
- alternate - The animation is played forwards first, then backwards
- alternate-reverse - The animation is played backwards first, then forwards

The following example will run the animation in reverse direction (backwards):

Code:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
  position: relative;
  animation-name: example;
  animation-duration: 4s;
  animation-direction: reverse;
}

@keyframes example {
  0%   {background-color:red; left:0px; top:0px;}
  25%  {background-color:yellow; left:200px; top:0px;}
  50%  {background-color:blue; left:200px; top:200px;}
  75%  {background-color:green; left:0px; top:200px;}
  100% {background-color:red; left:0px; top:0px;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>
```

```
<p>The animation-direction property specifies whether an animation should be played forwards,
backwards or in alternate cycles. The following example will run the animation in reverse
direction (backwards):</p>

<div></div>

</body>
</html>
```

Output:
Practice in lab

8. Columns and Flexbox

8.1 CSS columns

The **columns** property in CSS sets the number and width of columns in a single declaration. It is a shorthand property that can take several values at a time.

It is used to set both **column-count** and **column-width** properties at the same time. Both of these properties are used to control how many columns will appear. The **column-count** property is used to set the number of columns, whereas the **column-width** property specifies the width of the columns.

Together using **column-count** and **column-width** properties creates a multi-column layout that breaks automatically into a single column at narrow browser widths without using the media queries. It is not always helpful to use both of them because it can restrict the responsiveness and flexibility of the layout.

If the column-count and width don't fit in the element's width, then the browser reduces the column count automatically to fit the specified column widths.

Syntax

columns: auto | column-width column-count| initial | inherit;

Values

The property values are tabulated as follows with their description.

Value	Description
auto	It is the default value which sets the values of **column-count** and **column-width** to the default browser values.
column-width	It is used to set the minimum width for columns. However, the actual width of the column may be narrower or wider based on the available space.
column-count	It specifies the maximum number of columns.
Initial	It is used to set the property to its default value.
Inherit	It inherits the property from its parent element.

If any of the value is omitted, then by default, the browser assumes the corresponding value to **auto**.

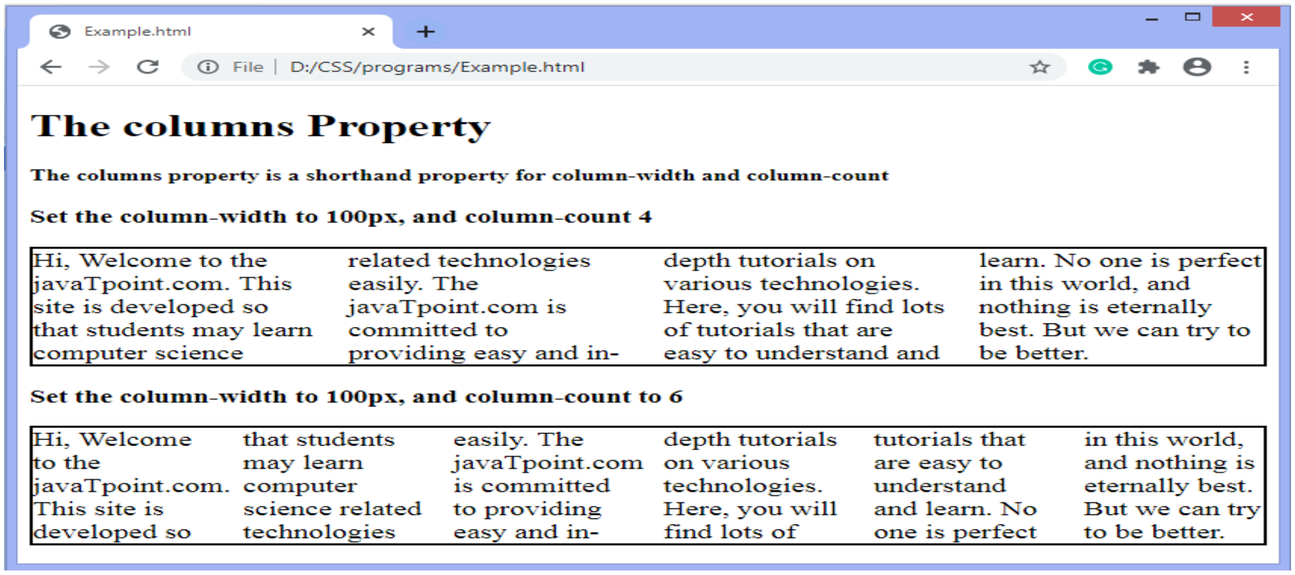
Example

In this example, we are defining two **<div>** elements, including text. For the first div element, the minimum width is 100px, and the maximum no. of columns can be four. Whereas for the second div element, the minimum width is 100px, and the maximum no. of columns can be six.

```
<!DOCTYPE html>
<html>
<head>
<style>
.div1 {
  columns: 100px 4;
  border: solid 2px black;
  font-size: 20px;
}

.div2 {
  columns: 100px 6;
  border: solid 2px black;
  font-size: 20px;
  }
</style>
</head>
<body>

```

```
<h1> The columns Property </h1>
<h4> The columns property is a shorthand property for column-width and column-count </h4>

<h3> Set the column-width to 100px, and column-count 4 </h3>
<div class="div1">
Hi, Welcome to the javaTpoint.com. This site is developed so that students may learn computer science related technologies easily. The javaTpoint.com is committed to providing easy and in-depth tutorials on various technologies. Here, you will find lots of tutorials that are easy to understand and learn.
No one is perfect in this world, and nothing is eternally best. But we can try to be better.
</div>

<h3> Set the column-width to 100px, and column-count to 6 </h3>
<div class="div2">
Hi, Welcome to the javaTpoint.com. This site is developed so that students may learn computer science related technologies easily. The javaTpoint.com is committed to providing easy and in-depth tutorials on various technologies. Here, you will find lots of tutorials that are easy to understand and learn.
No one is perfect in this world, and nothing is eternally best. But we can try to be better.
</div>

</body>
</html>
```

Output



Example.html

D:/CSS/programs/Example.html

```
The columns Property

The columns property is a shorthand property for column-width and column-count

Set the column-width to 100px, and column-count 4

Hi, Welcome to the          related technologies       depth tutorials on         learn. No one is perfect
javaTpoint.com. This        easily. The                various technologies.      in this world, and
site is developed so        javaTpoint.com is          Here, you will find lots   nothing is eternally
that students may learn     committed to               of tutorials that are      best. But we can try to
computer science            providing easy and in-     easy to understand and     be better.

Set the column-width to 100px, and column-count to 6

Hi, Welcome       that students     easily. The       depth tutorials   tutorials that    in this world,
to the            may learn         javaTpoint.com    on various        are easy to       and nothing is
javaTpoint.com.   computer          is committed      technologies.     understand        eternally best.
This site is      science related   to providing      Here, you will    and learn. No     But we can try
developed so      technologies      easy and in-      find lots of      one is perfect    to be better.
```

8.2 Flexbox:

CSS3 Flexible boxes also known as CSS Flexbox, is a new layout mode in CSS3.

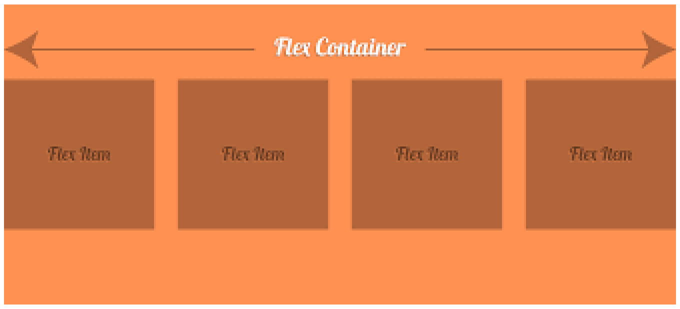
The CSS3 flexbox is used to make the elements behave predictably when they are used with different screen sizes and different display devices. It provides a more efficient way to layout, align and distribute space among items in the container.

It is mainly used to make CSS3 capable to change its item?s width and height to best fit for all available spaces. It is preferred over block model.

The CSS3 flexbox contains flex containers and flex items.

Flex container: The flex container specifies the properties of the parent. It is declared by setting the display property of an element to either flex or inline-flex.

Flex items: The flex items specify properties of the children. There may be one or more flex items inside a flex container.



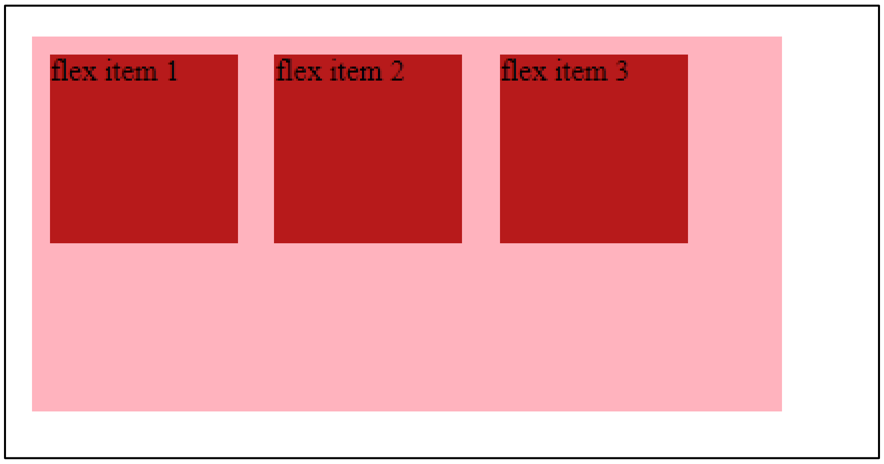
Note: Flexbox specifies how flex items are set inside a flex container. It sets the flex items inside a flex container along a flex line. By default, there is only one flex line per flex container. Everything outside a flex container and inside a flex item is considered as usual.

Let's take an example to show three flex items within a flex container. By default, they are set along the horizontal flex line, from left to right:

See this example:

```
<!DOCTYPE html>
<html>
<head><meta http-equiv="Content-Type" content="text/html; charset=windows-
1252">
<style>
.flex-container {
    display: -webkit-flex;
    display: flex;
    width: 400px;
    height: 200px;
    background-color: lightpink;
}
.flex-item {
    background-color: brown;
    width: 100px;
    height: 100px;
    margin: 10px;
}
</style>
</head>
<body>
<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

Output:



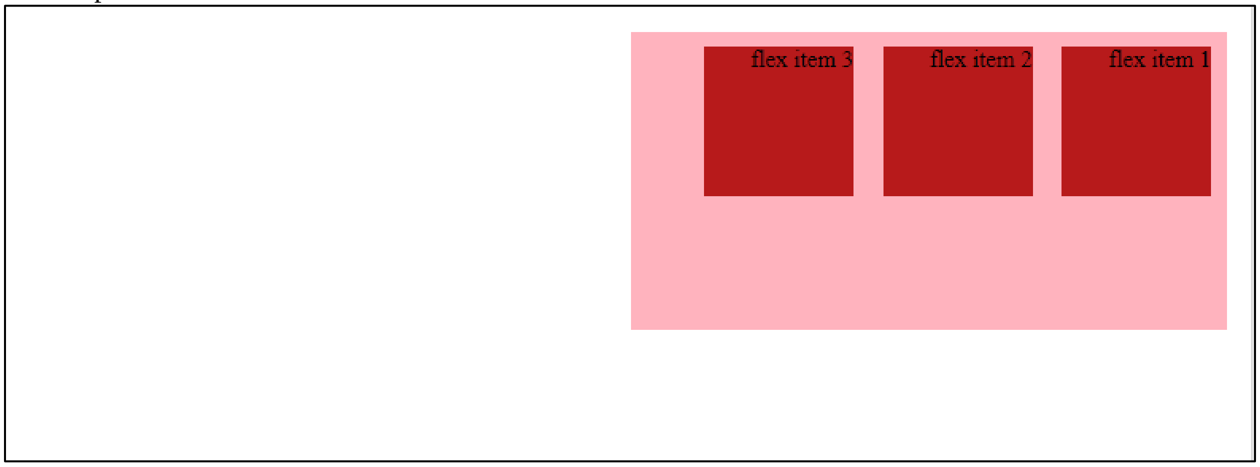
You can also change the direction of the flex line by using direction property. If you want to set the direction line right-to-left then set direction property to rtl.

See this example:

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
    direction: rtl;
}
```

```
.flex-container {
    display: -webkit-flex;
    display: flex;
    width: 400px;
    height: 200px;
    background-color: lightpink;
}
.flex-item {
    background-color: brown;
    width: 100px;
    height: 100px;
    margin: 10px;
}
</style>
</head>
<body>
<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

Output:



CSS3 Flexbox Properties

Following is a list of CSS3 Flexbox properties:

property	description
display	it is used to specify the type of box used for an html element.
flex-direction	it is used to specify the direction of the flexible items inside a flex container.
justify-content	it is used to align the flex items horizontally when the items do not use all available space on the main-axis.
align-items	it is used to align the flex items vertically when the items do not use all available space on the cross-axis.
flex-wrap	it specifies whether the flex items should wrap or not, if there is not enough room for them on one flex line.
align-content	it is used to modify the behavior of the flex-wrap property. it is similar to align-items, but instead of aligning flex items, it aligns flex lines.
flex-flow	it specifies a shorthand property for flex-direction and flex-wrap.
order	it specifies the order of a flexible item relative to the rest of the flex items inside the same container.
align-self	it is used on flex items. it overrides the container's align-items property.
flex	it specifies the length of a flex item, relative to the rest of the flex items inside the same container.

Html5 MCQ's

<canvas>

1) HTML5 element <canvas> gives you an easy and powerful way to draw graphics using JavaScript.

A) True
B) False
ANSWER: A) True

2) Which method is used to increase or decrease the units in a canvas grid.

A) Canvas Transform
B) Canvas Rotation
C) Canvas Scaling
D) Canvas Composition
ANSWER: C) Canvas Scaling

3) Which method is used to mask off certain areas or clear sections from the canvas.

A) Canvas Transform
B) Canvas Rotation
C) Canvas Scaling
D) Canvas Composition

ANSWER: D) Canvas Composition

4) Which version does not support for HTML5 canvas natively?

A) Firefox
B) Safari
C) Chrome
D) IE8

ANSWER: D) IE8

5) The latest versions of Firefox, Safari, Chrome and Opera all does not support for HTML5 Canvas.

A) True
B) False

ANSWER: B) False

6) The canvas element has a DOM method it is called.

A) getContext
B) getId
C) getElement
D) None of the above

ANSWER: A) getContext

7) HTML5 element <canvas> can be used to.

A) Draw graphics
B) Photo compositions
C) Animations
D) All the mentioned above

ANSWER: D) All the mentioned above

8) The HTML canvas is a

A) Three-dimensional grid
B) One-dimensional grid
C) Two-dimensional grid
D) None of the above

ANSWER: C) Two-dimensional grid

9) To draw text on a canvas, which property and method is used.

A) font
B) fillText(text,x,y)
C) stokeText(text,x,y)
D) Both A & B

ANSWER: C) stokeText(text,x,y)

Html5 geolocation:

1) From the following which methods are provided by geolocation?

A) getCurrentPosition()
B) watchPosition()
C) clearWatch()
D) All the mentioned above
ANSWER: D) All the mentioned above

2) Which method retrieves periodic updates about the current geographic location of the device.

A) getCurrentPosition()
B) watchPosition()
C) clearWatch()
D) All the mentioned above
ANSWER: B) watchPosition()

3) Which method cancels an ongoing watchPosition call?

A) getCurrentPosition()
B) watchPosition()
C) clearWatch()
D) None of the above
ANSWER: C) clearWatch()

4) Geolocation methods getCurrentPosition() and getPositionUsingMethodName() specify the callback method that retrieves the location information.

A) True
B) False
ANSWER: A) True

5) In location property what specifies the geographic location of the device. The location is expressed as a set of geographic coordinates together with information about heading and speed.

A) coords.accuracy
B) coords.longitude
C) coords
D) coords.altitudeAccuracy

ANSWER: C) coords

6) Which methods make use of an error handler callback method which gives PositionError object.

A) getCurrentPosition()
B) watchPosition()
C) clearWatch()
D) Both A & B
E) Both B & C
ANSWER: D) Both A & B

7) The location of the device could not be determined,the error code is 2 then what will be the constant?

A) UNKNOWN_ERROR
B) PERMISSION_DENIED
C) POSITION_UNAVAILABLE
D) TIMEOUT
ANSWER: C) POSITION_UNAVAILABLE

8) In the position options which property specifies whether the widget wants to receive

the most accurate location estimate possible. By default this is false.

A) maximumAge
B) timeout
C) enableHighAccuracy
D) None of the above
ANSWER: C) enableHighAccuracy

9) Geolocation is not complicated, and it is very much required to catch any error and handle it gracefully.

A) True
B) False

ANSWER: B) False

10) The following is a sample code to use for which of these methods?

```
function getLocation() {
var geolocation = navigator.geolocation;
geolocation.getCurrentPosition(showLocation,
errorHandler);
}
```

A) getCurrentPosition()
B) watchPosition()
C) clearWatch()
D) All the mentioned above

ANSWER: D) All the mentioned above

Drag and drop

1) Drag and Drop (DnD) is powerful User Interface concept which makes it easy to.

A) Copy
B) Deletion
C) Reorder
D) All the mentioned above
ANSWER: D) All the mentioned above

2)HTML 5 DnD is supported by all the major browsers like Chrome, Firefox 3.5 and Safari 4.

A) True
B) False
ANSWER: A) True

3) Drag and Drop (DnD) is powerful User Interface concept which makes it easy to copy, reorder and deletion of items with the help of.

A) Keyboard
B) Mouse
C) Both A & B
D) None of the above
ANSWER: B) Mouse

4) Which event will fire every time when the mouse is moved while the object is being dragged.

A) Drop
B) Drag
C) Dragend
D) Dragstart
ANSWER: B) Drag

5) Which event is fired as the mouse is moved over an element when a drag is occuring. Much of the time, the operation that occurs during a listener will be the same as the dragenter event.

A) Dragleave
B) Dragover
C) Dragstart
D) Dragenter
ANSWER: B) Dragover

6) Which data transfer attribute returns the specified data. If there is no such data, returns the empty string.

A) data = dataTransfer.getData(format)
B) dataTransfer.setData(format, data)
C) dataTransfer.files
D) dataTransfer.setDragImage(element,x,y)
ANSWER: A) data = dataTransfer.getData(format)

7) These are the steps for,

a) The dragenter event, which is used to determine whether or not the drop target is to accept the drop. If the drop is to be accepted, then this event has to be canceled.
b) The dragover event, which is used to determine what feedback is to be shown to the user. If the event is canceled, then the feedback (typically the cursor) is updated based on the dropEffect attribute's value.

A) Dragging the object
B) Dropping the object
C) Both A & B
D) None of the above
ANSWER: B) Dropping the object

8) The following description is given for which data transfer attribute?

a) Returns the kind of operation that is currently selected.
b) This attribute can be set, to change the selected operation.
c) The possible values are none, copy, link, and move.

A) dataTransfer.effectAllowed [= value]
B) dataTransfer.dropEffect [= value]
C) dataTrnasfer.types
D) dataTransfer.clearData([format])
ANSWER: B) dataTransfer.dropEffect [= value]

9) The drop event fires when the user releases the mouse button while dragging an object.

A) True
B) False
ANSWER: B) False

10) HTML5 came up with a Drag and Drop (DnD) API that brings native DnD support to the browser making it much easier to code up.

A) Yes
B) No

ANSWER: A) Yes

HTML5 SVG

1) Abbreviate the term SVG.

A) Simple Velocity Graphics
B) Simple Vector Graph
C) Scalable Vector Graphics
D) System Vector Graphics

ANSWER: C) Scalable Vector Graphics

2) SVG is mostly useful for vector type diagrams like.

A) Pie charts
B) Two-dimensional graphs
C) Both A & B
D) None of the above
ANSWER: C) Both A & B

3) Vector graphics contain geometric objects such as lines and curves.

A) Yes
B) No
ANSWER: A) Yes

4) Which element is the most common method for using graphics in HTML pages.

A) object
B) img
C) applet
D) None of the above

ANSWER: B) img

5) Which is not suited for game applications?

A) Canvas
B) Svg
C) Both A & B
D) None of the above
ANSWER: B) Svg

6) SVG is resolution dependent.

A) True
B) False

ANSWER: B) False

7) Which is best suited for applications with large rendering areas eg: Google maps?

A) Svg
B) Canvas
C) Both A & B
D) None of the above

ANSWER: A) Svg

8) SVG graphics do NOT lose any quality if they are zoomed or re-sized.

A) True
B) False

ANSWER: A) True

9) SVG 1.1(first edition) became a W3C Recommendation on?

A) 4 September 2001
B) 14 January 2003
C) 16 August 2011
D) None of the above
ANSWER: B) 14 January 2003

10) Which relies on proprietary technology that is not open source.

A) SVG
B) Flash
C) Both A & B
D) None of the above
ANSWER: B) Flash

CSS MCQ's

1) CSS stands for -

a. Cascade style sheets
b. Color and style sheets
c. Cascading style sheets
d. None of the above

Answer: (c) Cascading style sheets

Explanation: CSS stands for Cascading Style Sheet. CSS is used to design HTML tags. CSS is a widely used language on the web. HTML, CSS and JavaScript are used for web designing. It helps the web designers to apply style on HTML tags.

---

2) Which of the following is the correct syntax for referring the external style sheet?

a. <style src = example.css>
b. <style src = "example.css" >
c. <stylesheet> example.css </stylesheet>
d. <link rel="stylesheet" type="text/css" href="example.css">

Answer: (d) <link rel="stylesheet" type="text/css" href="example.css">

Explanation: The external style sheet is generally used when you want to make changes on multiple pages. It uses the <link> tag on every pages and the <link> tag should be put inside the head section.

3) The property in CSS used to change the background color of an element is -

a. bgcolor
b. color
c. background-color
d. All of the above

Answer: (c) background-color

Explanation: The background-color property is used to specify the background color of an element. The background of an element covers the total size, including the padding and border and excluding margin.

4) The property in CSS used to change the text color of an element is –

a) bgcolor
b) color
c) background-color
d) All of the above

Answer: (b) color

Explanation: The color property in CSS is used to set the color of HTML elements. Typically, this property is used to set the font color of an element. In CSS, we use color values for specifying the color. We can also use this property for the border-color and other decorative effects.

5) The CSS property used to control the element's font-size is -

a. text-style
b. text-size
c. font-size
d. None of the above

Answer: (c) font-size

Explanation: The font-size property in CSS is used to specify the height and size of the font. It affects the size of the text of an element. Its default value is medium and can be applied to every element.

6) The HTML attribute used to define the inline styles is -

a. style
b. styles
c. class
d. None of the above

Answer: (a) style

Explanation: If you want to use inline CSS, you should use the style attribute to the relevant tag. The inline CSS is also a method to insert style sheets in HTML document. This method mitigates some advantages of style sheets so it is advised to use this method sparingly.

---

7) The HTML attribute used to define the internal stylesheet is -

a. <style>
b. style
c. <link>
d. <script>

Answer: (a) <style>

Explanation: The internal style sheet is used to add a unique style for a single document. It is defined in <head> section of the HTML page inside the <style> tag.

---

8) Which of the following CSS property is used to set the background image of an element?

a. background-attachment
b. background-image
c. background-color
d. None of the above

Answer: (b) background-image

Explanation: The background-image property is used to set an image as a background of an element. By default the image covers the entire element.

---

9) Which of the following is the correct syntax to make the background-color of all paragraph elements to yellow?

a. p {background-color : yellow;}
b. p {background-color : #yellow;}
c. all {background-color : yellow;}
d. all p {background-color : #yellow;}

Answer: (a) p {background-color : yellow;}

Explanation: The background-color property in CSS is used to change the background color of an element. The correct syntax to make the background color of all paragraph elements to yellow is: p {background-color : yellow;}.

10) Which of the following is the correct syntax to display the hyperlinks without any underline?

a. a {text-decoration : underline;}
b. a {decoration : no-underline;}
c. a {text-decoration : none;}

d. None of the above

Answer: (c) a {text-decoration : none;}

Explanation: The text-decoration property in CSS is used to decorate the content of the text. It adds lines under, above, and through the text. It sets the appearance of decorative lines on text. The correct syntax to display the hyperlinks without any underline is: a {text-decoration : none;}.

11) Which of the following property is used as the shorthand property for the padding properties?

a. padding-left
b. padding-right
c. padding
d. All of the above

Answer: (c) padding

Explanation: CSS padding property is used to define the space between the element content and the element border. Top, bottom, left and right padding can be changed independently using separate properties. By using the shorthand padding property we can also change all padding properties at once.

12) The CSS property used to make the text bold is -

a. font-weight : bold
b. weight: bold
c. font: bold
d. style: bold

Answer: (a) font-weight : bold

Explanation: The font-weight property is used for setting the thickness and boldness of the font. It is used to define the weight of the text. The available weight depends on the font-family, which is used by the browser.

13) Are the negative values allowed in padding property?

a. Yes
b. No
c. Can't say

d. May be

Answer: (b) No

Explanation: The negative values are not allowed when using the padding property.

14) Which of the following property is used as the shorthand property of margin properties?

a. margin-left
b. margin-right
c. margin
d. None of the above

Answer: (c) margin

Explanation: CSS Margin property is used to define the space around elements. The Top, bottom, left and right margin can be changed independently using separate properties. By using the shorthand margin property we can change all margin properties at once.

15) The CSS property used to specify the transparency of an element is -

a. opacity
b. filter
c. visibility
d. overlay

Answer: (a) opacity

Explanation: The CSS opacity property is used to specify the transparency of an element. In simple words, you can say that it specifies the clarity of the image.

16) Which of the following is used to specify the subscript of text using CSS?

a. vertical-align: sub
b. vertical-align: super
c. vertical-align: subscript
d. None of the above

Answer: (a) vertical-align : sub

Explanation: In CSS, the vertical-align property is used to make the text superscript or subscript. The "sub" of vertical-align property is used to make the text subscript in CSS. The subscript text appears in a smaller font and the half character below the normal line. It is generally used to write chemical equations or chemical formulas such as H2O, H2SO4, etc.

17) Which of the following CSS property is used to specify the space between every letter inside an element?

a. alpha-spacing
b. character-spacing
c. letter-spacing
d. alphabet-spacing

Answer: (c) letter-spacing

Explanation: The letter-spacing property in CSS is used to control the space between every letter inside an element or the block of text. It sets the spacing behavior between the characters of the text. Using this property, we can increase or decrease the space between the characters of the text.

18) The CSS property used to specify whether the text is written in the horizontal or vertical direction?

a. writing-mode
b. text-indent
c. word-break
d. None of the above

Answer: (a) writing-mode

Explanation: The writing-mode CSS property specifies whether the text is written in the vertical or horizontal direction. This property sets whether the lines of text are laid out vertically or horizontally. It specifies the direction in which the content flows on the page.

19) Which of the following syntax is correct in CSS to make each word of a sentence start with a capital letter?

a. text-style : capital;
b. transform : capitalize;
c. text-transform : capital;
d. text-transform : capitalize;

Answer: (d) text-transform : capitalize;

Explanation: The text-transform CSS property allows us to change the case of the text. It is used to control text capitalization. The "capitalize" value of the text-transform property transforms the first character of each word to uppercase. It will not capitalize the first letter after the number.

20) How to select the elements with the class name "example"?

a. example
b. #example
c. .example
d. Class example

Answer: (c) .example

Explanation: The class selector selects HTML elements with a specific class attribute. It is used with a period character . (full stop symbol) followed by the class name. A class name should not be started with a number.

21) Which of the following is the correct syntax to select all paragraph elements in a div element?

a. div p
b. p
c. div#p
d. div ~ p

Answer: (a) div p

Explanation: The CSS descendant selector is used to match the descendant elements of a particular element and represent it using a single space. The word descendant indicates nested anywhere in the DOM tree.

22) Which of the following is the correct syntax to select the p siblings of a div element?

a. p
b. div + p
c. div p
d. div ~ p

Answer: (d) div ~ p

Explanation: General sibling selector uses the tlide (~) sign as the separator between the elements. It can be used for selecting the group of elements that share the common parent element. The syntax div ~ p will select the paragraph elements that are the siblings of the div element.

23) The CSS property used to draw a line around the elements outside the border?

a. border
b. outline
c. padding
d. line

Answer: (b) outline

Explanation: The "outline" property in CSS facilitates you to draw an extra border around an element to get visual attention. It is similar to the CSS border property. It is as easy as to apply as a border.

24) Which of the following CSS property is used to add shadows to the text?

a. text-shadow
b. text-stroke
c. text-overflow
d. text-decoration

Answer: (a) text-shadow

Explanation: The CSS text-shadow property adds shadows to the text. It accepts the comma-separated list of shadows that applied to the text. It applies one or more than one text-shadow effect on the element's text content.

25) Which of the following is not a value of the font-variant property in CSS?

a. normal
b. small-caps
c. large-caps
d. inherit

Answer: (c) large-caps

Explanation: CSS font-variant property specifies how to set a font variant of an element. Its values may be normal and small-caps.

26) Which of the following CSS property is used to specify whether the table cells share the common or separate border?

a. border-collapse
b. border-radius
c. border-spacing
d. None of the above

Answer: (a) border-collapse

Explanation: The border-collapse CSS property is used to set the border of the table cells and specifies whether the table cells share a separate or common border. This property has two main values that are separate and collapse.

27) The CSS property used to make the rounded borders, or rounded corners around an element is -

a. border-collapse
b. border-radius
c. border-spacing
d. None of the above

Answer: (b) border-radius

Explanation: The border-radius CSS property sets the rounded borders and provides the rounded corners around an element, tags, or div. It defines the radius of the corners of an element.

28) The CSS property used to set the distance between the borders of the adjacent cells in the table is -

a. border-collapse
b. border-radius
c. border-spacing
d. None of the above

Answer: (c) border-spacing

Explanation: This CSS property is used to set the distance between the borders of the adjacent cells in the table. It applies only when the border-collapse property is set to separate.

29) Which of the following selector in CSS is used to select the elements that do not match the selectors?

a. :! selector
b. :not selector
c. :empty selector
d. None of the above

Answer: (b) :not selector

Explanation: The :not selector in CSS matches the elements that are not the specified element/selector.

30) Which of the following is not a type of combinator?

a. >
b. ~
c. +
d. *

Answer: (d) *

Explanation: CSS Combinators clarifies the relationship between two selectors. There are four types of combinators in CSS that are listed as follows:

- General sibling selector (~)
- Adjacent sibling selector (+)
- Child selector (>)
- Descendant selector (space)

31) Which of the following CSS property defines how an image or video fits into container with established height and width?

a. object-fit
b. object-position
c. position

d. None of the above

Answer: (a) object-fit

Explanation: The object-fit CSS property specifies how a video or an image is resized to fit its content box. It defines how an element fits into the container with an established width and height. It is generally applied to images or videos.

32) Which type of CSS is used in the below code?

```
1. <p style = "border:2px solid red;">
```

a. Inline CSS
b. Internal CSS
c. External CSS
d. None of the above

Answer: (a) Inline CSS

Explanation: If you want to use inline CSS, you should use the style attribute to the relevant tag. The inline CSS is also a method to insert style sheets in HTML document.

33) Which of the following CSS property specifies the origin of the background-image?

a. background-origin
b. background-attachment
c. background-size
d. None of the above

Answer: (a) background-origin

Explanation: The background-origin CSS property helps us to adjust the background image of the webpage. It specifies the background-position area, i.e., the origin of a background image. This CSS property will not work when the value of the background-attachment is set to be fixed. It is similar to the background-clip property, except that it resizes the background instead of clipping it.

34) The CSS property used to set the maximum width of the element's content box is -

a. max-width property
b. height property

c. max-height property
d. position property

Answer: (a) max-width property

Explanation: The max-width property in CSS is used to set the maximum width of the element's content box. It means that the width of the content box can be smaller than the max-width value but cannot be greater. It sets the upper bound on the element's width.

35) Which if the following CSS function allows us to perform calculations?

a. calc() function
b. calculator() function
c. calculate() function
d. cal() function

Answer: (a) calc() function

Explanation: The calc() is an inbuilt CSS function that allows us to perform the calculations. It can be used to calculate length, percentage, time, number, integer frequency, or angle. It uses the four simple arithmetic operators add (+), multiply (*), subtract (-), and divide (/).

36) The CSS property used to set the maximum height of the element's content box is -

a. max-width property
b. height property
c. max-height property
d. position property

Answer: (c) max-height property

Explanation: The max-height property in CSS sets the maximum height of the element's content box. It means that the height of the content box can be smaller than the max-height value but cannot be greater. It sets the upper bound on the element's height.

37) The CSS property used to set the minimum width of the element's content box is -

a. max-width property
b. min-width property
c. width property

d. All of the above

Answer: (b) min-width property

Explanation: The min-width property is used to set the minimum width of the element's content box. It means that the width of the content box can be greater than the min-width value but cannot be shorter. It sets the lower bound on the element's width.

38) Which of the following CSS property is used to represent the overflowed text which is not visible to the user?

a. text-shadow
b. text-stroke
c. text-overflow
d. text-decoration

Answer: (c) text-overflow

Explanation: The text-overflow property specifies the representation of overflowed text, which is not visible to the user. It signals the user about the content that is not visible. This property helps us to decide whether the text should be clipped, show some dots (ellipsis), or display a custom string.

39) The CSS property which is used to define the set the difference between two lines of your content is -

a. min-height property
b. max-height property
c. line-height property
d. None of the above

Answer: (c) line-height property

Explanation: The CSS line-height property is used to define the minimal height of line boxes within the element. It sets the differences between the two lines of your content. It defines the amount of space above and below inline elements.

40) The CSS property which is used to define the set the difference between two lines of your content is -

a. min-height property
b. max-height property
c. line-height property
d. None of the above

Answer: (c) line-height property

Explanation: The CSS line-height property is used to define the minimal height of line boxes within the element. It sets the differences between the two lines of your content. It defines the amount of space above and below inline elements.

41) Which of the following CSS property is used to add stroke to the text?

a. text-stroke property
b. text-transform property
c. text-decoration property
d. None of the above

Answer: (a) text-stroke property

Explanation: The text-stroke property in CSS is used to add a stroke to the text and also provides decoration options for them. It defines the color and width of strokes for text characters.

42) Which of the following CSS property is used to set the blend mode for each background layer of an element?

a. background-blend-mode property
b. background-collapse property
c. background-transform property
d. background-origin property

Answer: (a) background-blend-mode property

Explanation: The background-blend-mode property in CSS is used to set the blend mode for each background layer (image/color) of an element. It defines how the background image of an element blends with the background color of the element. We can blend the background images together or can blend them with background-color.

43) The CSS property used to specify the transparency of an element is -

a. Hover
b. opacity
c. clearfix
d. overlay

Answer: (b) opacity

Explanation: The CSS opacity property is used to specify the transparency of an element. In simple words, you can say that it specifies the clarity of the image.

---

44) Which of the following CSS property is used to set the horizontal alignment of a table-cell box or the block element?

a. text-align property
b. text-transform property
c. text-shadow property
d. text-decoration

Answer: (a) text-align property

Explanation: The text-align property in CSS is used to set the horizontal alignment of a table-cell box or the block element. It is similar to the vertical-align property but in the horizontal direction.

---

45) The CSS property which is used to set the text wider or narrower compare to the default width of the font is -

a. font-stretch property
b. font-weight property
c. text-transform property
d. font-variant property

Answer: (a) font-stretch property

Explanation: The font-stretch property in CSS allows us to select a normal, expanded, or condensed face from the font's family. This property sets the text wider or narrower to compare to the default width of the font. It will not work on any font but only works on the font-family that has a width-variant face.

---

46) Which of the following CSS property is used to specify the type of quotation mark?

a. quotes property
b. z-index property
c. hyphens property
d. None of the above

Answer: (a) quotes property

Explanation: The quotes property in CSS specifies the type of quotation mark for the quotation used in the sentence. It defines which quotation mark to be used when the quotation is inserted by using the open-quote and close-quote values of the content property.

---

47) The CSS property used to specify the order of flex item in the grid container is -

a. order property
b. float property
c. overflow property
d. None of the above

Answer: (a) order property

Explanation: This CSS property specifies the order of the flex item in the grid container or flex. It is basically used for ordering the flex items. It is to note that if the element isn't flexible, then this property will not work.

---

48) The CSS property used to set the indentation of the first line in a block of text is -

a. text-indent property
b. text-stroke property
c. text-decoration property

d. text-overflow property

Answer: (a) text-indent property

Explanation: This CSS property sets the indentation of the first line in a block of text. It specifies the amount of horizontal space that puts before the lines of text.

---

49) Which of the following CSS property creates a clipping region and specifies the visible area of the element?

a. visibility property
b. background-clip property
c. clip-path property
d. None of the above

Answer: (c) clip-path property

Explanation: The clip-path CSS property is used to create a clipping region and specifies the element's area that should be visible. The area inside the region will be visible, while the outside area is hidden. Anything outside the clipping will be clipped by browsers, including borders, text-shadows, and many more.

---

50) The correct syntax to give a line over text is -

a. text-decoration: line-through
b. text-decoration: none
c. text-decoration: overline
d. text-decoration: underline

Answer: (c) text-decoration: overline

Explanation: The text-decoration is a CSS property that decorates the content of the text. It sets the kind of text-decoration like overline, underline, or line-through.