

CASCADING STYLE SHEETS:

Style sheets represent the World Wide Web consortium's effort to improve on the tag and attribute based style of formatting. Style sheets provide a way of customizing whole pages all at once and in much richer detail than the simple use of tags and attributes. The format of style sheet will be:

```
<style type="text/css">
    selector{property:value;property:value;}
    selector{property:value;property:value;}
</style>
```

Every line in <style> tag is called as a „Rule“ and a style rule has two parts:

- a. Selector.
- b. Set of declarations.

A selector is used to create a link between the rule and the HTML tag. The declaration has two parts

again:

- a. Property.
- b. Value.

A property specifies additional information and value specifies property value. For example:

```
<style type="text/css">
body {background-color: #d0e4fe;}
h1 {
    color: orange;
    text-align: center;
}
p {
```

```

font-family: "Times New Roman";
font-size: 20px;
}
</style>

```

If we add above code in the <head> element of web page, entire web page will be displayed in various styles given in style element.

Style sheets are implemented with cascading style sheets specification. Conventionally styles are cascaded i.e., we don't have to use just a single set of styles inside a document, but we can import as many styles as we like. There are three mechanisms by which we can apply styles to our HTML documents:

1. Inline Style sheets.
2. Embedded Style sheets.
3. External Style sheets.

Inline Style Sheets:

Inline style sheets mix content with presentation. To use inline styles we use style attribute in the relevant tag.

Example:

```

<html>
  <head>
    <title>HTML Tables</table>
  </head>
  <body bgcolor="pink">
    <center>
      <h1>Creating HTML Tables</h1><br>
      <table border="2" cellpadding="4" cellspacing="4">
        <tr>
          <th colspan="2" style="background-color:red"> WebSites</th>

```

```
</tr>
<tr>
  <th style="background-color:blue">MailSites</th>
  <th style="background-color:green">JobSites</th>
</tr>
<tr>
  <td style="background-color:grey">Gmail</td>
  <td style="background-color:aqua">Naukri</td>
</tr>
<tr>
  <td style="background-color:yellow">Yahoo</td>
  <td style="background-color:purple">JobStreet</td>
</tr>
</table>
</center>
</body>
</html>
```

Output:



Embedded Style sheets:

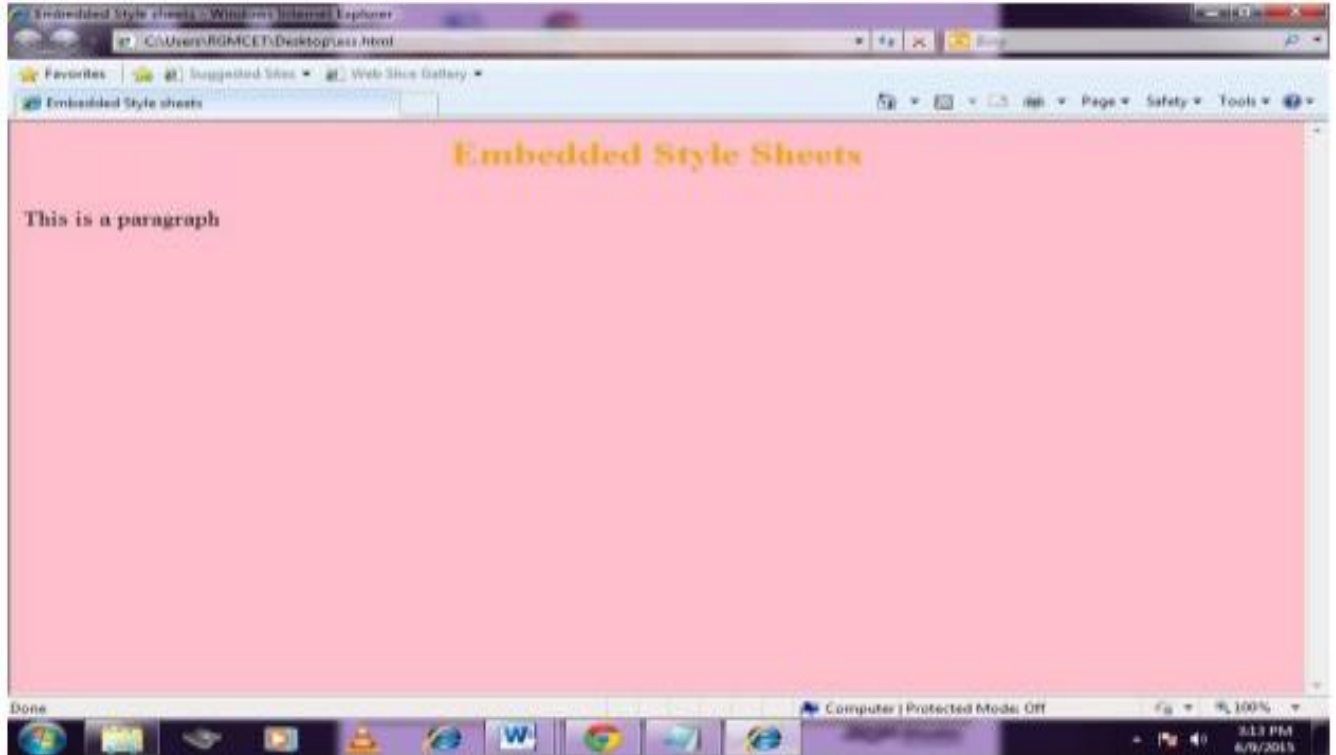
An embedded style sheet is used when a single document has a unique style. We define internal styles in the head section of a HTML page by using „<style>“ tag. The styles defined using embedded style sheets are applied throughout the page and we put the styles into one place.

Example:

```
<html>

  <head>
  <title>Embedded Style sheets</title>
  <style type="text/css">
    body{background-color: pink;}
  h1 {
  color:orange;
  text-align:
  center;
  }
  p {
```

```
font-family: "Times New Roman"; font-size: 20px;
}
</style>
</head>
<body>
<h1>Embedded Style Sheets</h1><br>
<p>This is a paragraph
</body>
</html>
```

Output:**External Style Sheets:**

External style sheets are just that the style sheets are stored separately from our web page. These are useful especially if we are setting the styles for an entire website. When we change the styles in external style sheet we change the styles of all pages. We use „<link>” element to access the style sheet file defined into our web page. The format of <link> element is:

```
<link rel="stylesheet" type="text/css" href="extstylesheet.css">
```

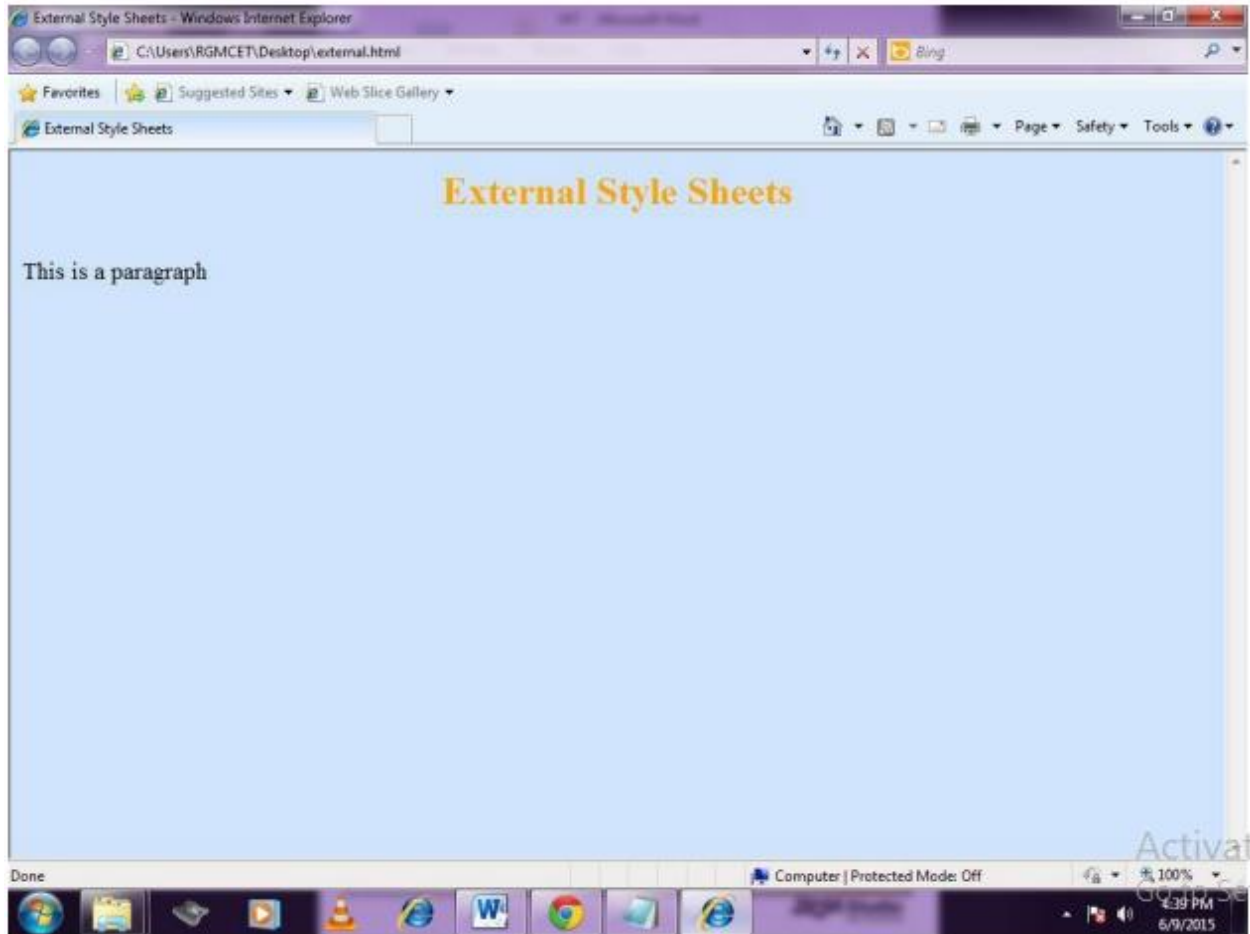
Example:**extern.css:**

```
body {background-color: #d0e4fe;}
    h1 {
        color: orange; text-align: center;
    }
    p {
        font-family: "Times New Roman"; font-size: 20px;
    }
```

extern.html:

```
<html>
    <head>
        <title>External Style Sheets</title>
        <link rel="stylesheet" type="text/css" href="extern.css">
    </head>
    <body>
        <h1>External Style Sheets</h1><br>
        <p>This is a paragraph
    </body>
</html>
```

Output:



CSS3

CSS3: CSS3 stands for Cascading Style Sheet level 3, which is the advanced version of CSS. It is used for structuring, styling, and formatting web pages. Several new features have been added to CSS3 and it is supported by all modern web browsers. The most important feature of CSS3 is the splitting of CSS standards into separate modules that are simpler to learn and use.

CSS3 BACKGROUND

This chapter teaches you how to set backgrounds of various HTML elements. You can set the following background properties of an element –

- The **background-color** property is used to set the background color of an element.
- The **background-image** property is used to set the background image of an element.

- The **background-repeat** property is used to control the repetition of an image in the background.
- The **background-position** property is used to control the position of an image in the background.
- The **background-attachment** property is used to control the scrolling of an image in the background.
- The **background** property is used as a shorthand to specify a number of other background properties.

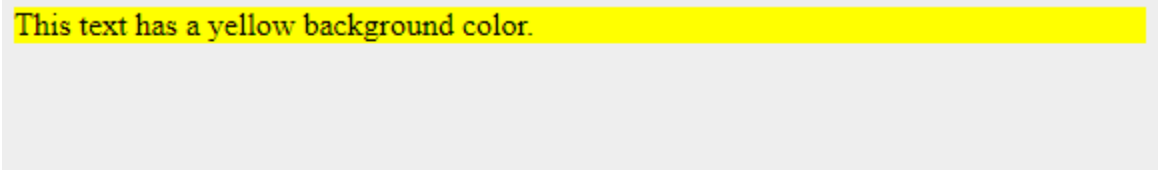
Set the Background Color

Following is the example which demonstrates how to set the background color for an element.

```
<html>
  <head>
  </head>

  <body>
    <p style = "background-color:yellow;">
      This text has a yellow background color.
    </p>
  </body>
</html>
```

This will produce following result –



This text has a yellow background color.

Set the Background Image

We can set the background image by calling local stored images as shown below –


```

<html>
<head>
  <style>
    body {
      background-image: url("/css/images/css.jpg");
      background-color: #cccccc;
    }
  </style>
</head>

<body>
  <h1>Hello World!</h1>
</body>
</html>

```

It will produce the following result –



Repeat the Background Image

The following example demonstrates how to repeat the background image if an image is small. You can use *no-repeat* value for *background-repeat* property if you don't want to repeat an image, in this case image will display only once.

By default *background-repeat* property will have *repeat* value.

```

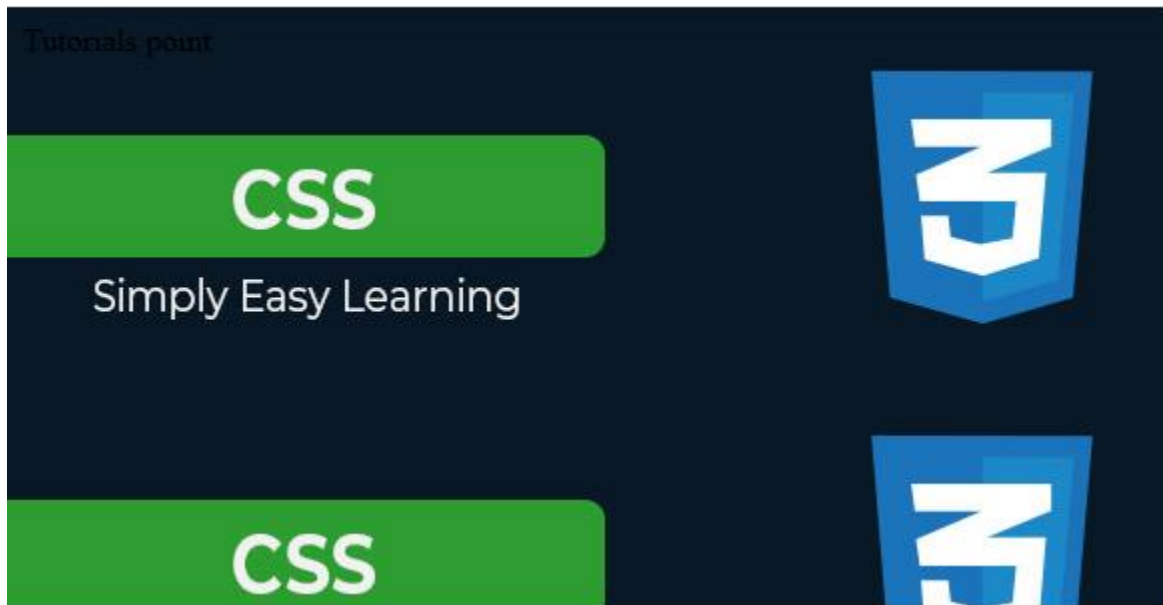
<html>
<head>
  <style>
    body {
      background-image: url("/css/images/css.jpg");
      background-repeat: repeat;
    }
  </style>
</head>
</html>

```

```
</style>
</head>

<body>
  <p>Tutorials point</p>
</body>
</html>
```

It will produce the following result –



CSS3 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 Hue 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;}
```

The following example shows rgba color property.

```
<html>
<head>
```

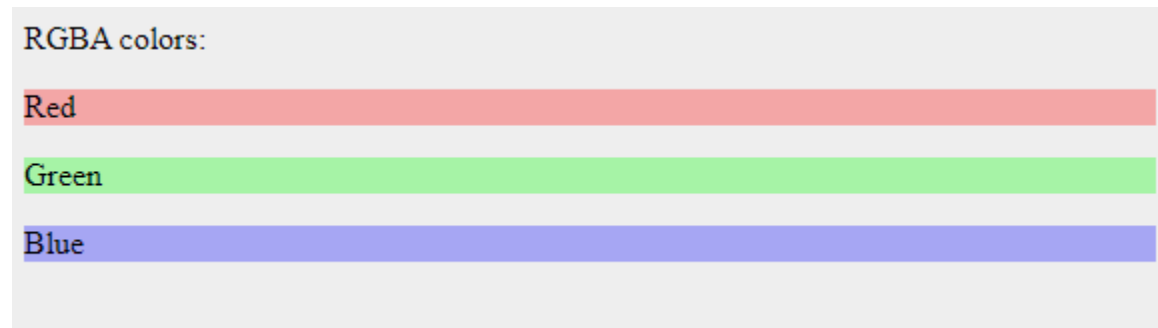
```

<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>

```

It will produce the following result –



CSS3 SHADOWS

CSS3 supported to add shadow to text or elements. Shadow property has divided as follows –

- Text shadow
- Box Shadow

Text shadow

CSS3 supported to add shadow effects to text. Following is the example to add shadow effects to text –

```
<html>
```

```
<head>
  <style>
    h1 {
      text-shadow: 2px 2px;
    }
    h2 {
      text-shadow: 2px 2px red;
    }
    h3 {
      text-shadow: 2px 2px 5px red;
    }
    h4 {
      color: white;
      text-shadow: 2px 2px 4px #000000;
    }
    h5 {
      text-shadow: 0 0 3px #FF0000;
    }
    h6 {
      text-shadow: 0 0 3px #FF0000, 0 0 5px #0000FF;
    }
    p {
      color: white;
      text-shadow: 1px 1px 2px black, 0 0 25px blue, 0 0 5px darkblue;
    }
  </style>
</head>

<body>
  <h1>Tutorialspoint.com</h1>
  <h2>Tutorialspoint.com</h2>
```

```

<h3>Tutorialspoint.com</h3>
<h4>Tutorialspoint.com</h4>
<h5>Tutorialspoint.com</h5>
<h6>Tutorialspoint.com</h6>
<p>Tutorialspoint.com</p>
</body>
</html>

```

It will produce the following result –



CSS3 TEXT

CSS3 contained several extra features, which is added later on.

- text-overflow
- word-wrap
- word-break

There are following most commonly used property in CSS3 –

Sr.No.	Value & Description
1	text-align-last Used to align the last line of the text
2	text-emphasis Used to emphasis text and color
3	text-overflow used to determines how overflowed content that is not displayed is signaled to users
4	word-break Used to break the line based on word
5	word-wrap Used to break the line and wrap onto next line

Text-overflow

The text-overflow property determines how overflowed content that is not displayed is signaled to users. the sample example of text overflow is shown as follows –

```
<html>
<head>
  <style>
    p.text1 {
      white-space: nowrap;
      width: 500px;
      border: 1px solid #000000;
      overflow: hidden;
      text-overflow: clip;
    }
  </style>
</head>
</html>
```

```
p.text2 {
  white-space: nowrap;
  width: 500px;
  border: 1px solid #000000;
  overflow: hidden;
  text-overflow: ellipsis;
}
</style>
</head>

<body>

<b>Original Text:</b>

<p>
  Tutorials Point originated from the idea that there exists a class of
  readers who respond better to online content and prefer to learn new
  skills at their own pace from the comforts of their drawing rooms.
</p>

<b>Text overflow:clip:</b>

<p class = "text1">
  Tutorials Point originated from the idea that there exists
  a class of readers who respond better to online content and prefer
  to learn new skills at their own pace from the comforts of their
  drawing rooms.
</p>

<b>Text overflow:ellipsis</b>
```



```
<p class = "text2">
```

Tutorials Point originated from the idea that there exists a class of readers who respond better to online content and prefer to learn new skills at their own pace from the comforts of their drawing rooms.

```
</p>
```

```
</body>
```

```
</html>
```

It will produce the following result –

Original Text:

Tutorials Point originated from the idea that there exists a class of readers who respond better to online content and prefer to learn new skills at their own pace from the comforts of their drawing rooms.

Text overflow:clip

Tutorials Point originated from the idea that there exists a class of readers who

Text overflow:ellipsis

Tutorials Point originated from the idea that there exists a class of readers ...

TRANSFORMATIONS

2D TRANSFORM

2D transforms are used to re-change the element structure as translate, rotate, scale, and skew.

The following table has contained common values which are used in 2D transforms –

Sr.No.	Value & Description
1	matrix(n,n,n,n,n,n) Used to defines matrix transforms with six values
2	translate(x,y) Used to transforms the element along with x-axis and y-axis
3	translateX(n) Used to transforms the element along with x-axis
4	translateY(n) Used to transforms the element along with y-axis
5	scale(x,y) Used to change the width and height of element
6	scaleX(n) Used to change the width of element
7	scaleY(n) Used to change the height of element
8	rotate(angle) Used to rotate the element based on an angle
9	skewX(angle) Used to defines skew transforms along with x axis
10	skewY(angle) Used to defines skew transforms along with y axis

The following examples are shown the sample of all above properties.

Rotate 20 degrees

Box rotation with 20 degrees angle as shown below –

```
<html>
<head>
  <style>
    div {
      width: 300px;
      height: 100px;
      background-color: pink;
      border: 1px solid black;
    }
    div#myDiv {
      /* IE 9 */
      -ms-transform: rotate(20deg);

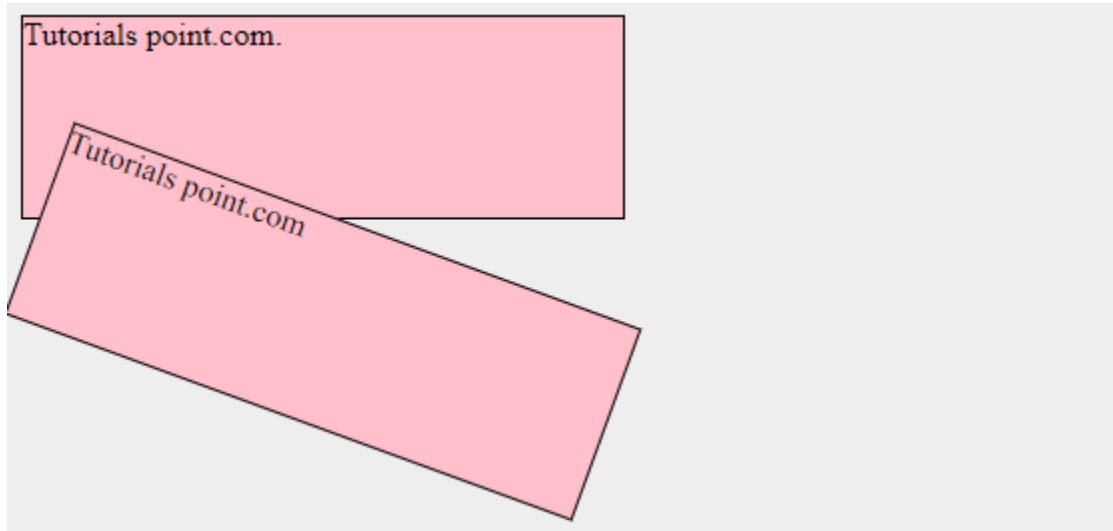
      /* Safari */
      -webkit-transform: rotate(20deg);

      /* Standard syntax */
      transform: rotate(20deg);
    }
  </style>
</head>

<body>
  <div>
    Tutorials point.com.
  </div>
  <div id = "myDiv">
    Tutorials point.com
  </div>
```

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

It will produce the following result –



CSS TRANSITIONS AND ANIMATION

CSS Transitions

A CSS transition allows you to change the property values of an element over a given duration that you set. To create a transition you must first identify which CSS property you want to add an effect to and then specify the duration of the effect. If no duration is set, the transition will not occur.

There are four transition properties:

transition-delay – specifies the delay, in seconds (s), you would like to assign your transition effect.

transition-duration – specifies the duration, in seconds (s) or milliseconds (ms), you would like to assign your transition effect.

transition-property – specifies the name of the CSS property your transition effect is meant for.

transition-timing-function – Specifies the speed curve of the transition effect. Meaning, the type of speed variation you want to select for your transition effect. There is no “fast” or “slow” options.

Instead there are speed curve options that go from one speed to another. Such as “ease” which tells your effect to start slow, then go fast, then end slowly.

To create a transition you only need to change one of these properties over the duration you choose. However, it is possible to change more than one property at the same time; resulting in more dramatic transitions.

CSS Animations

Where CSS transitions are all about altering element properties as they move from state to state, CSS animations are dependent on keyframes and animation properties.

keyframes – keyframes are used to define the styles an element will have at various times.

animation properties – animation properties are used to assign @keyframes to a specific element and determine how it is animated.

There are eight animation properties:

animation-delay – specifies a delay for the start of an animation.

animation-direction – specifies whether an animation should play in reverse direction or alternate cycles.

animation-duration – specifies how many seconds or milliseconds an animation takes to complete one cycle.

animation-fill-mode – specifies a style for the element when the animation is not playing. Such as when it is finished or when it has a delay.

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.

Example of key frames with left animation –

```
@keyframes animation {
```

```
from {background-color: pink;}  
to {background-color: green;}  
}  
div {  
width: 100px;  
height: 100px;  
background-color: red;  
animation-name: animation;  
animation-duration: 5s;  
}
```

The above example shows height, width, color, name and duration of animation with keyframes syntax.