CSS3 : Advanced CSS 2.0

- Border-radius
- resize
- Transitions
- Animations
- Transformations
- Gradients
- font-face
- filters

CSS3 resize Property

The resize property specifies whether or not an element is resizable by the user.
Note: The resize property applies to elements whose computed overflow value is something other than "visible".

resize: none|both|horizontal|vertical|initial|inherit;

none
Default value.   The user cannot resize the element
both
The user can adjust both the height and the width of the element
horizontal
The user can adjust the width of the element
vertical
The user can adjust the height of the element
initial
Sets this property to its default value.
inherit
Inherits this property from its parent element.

Example :

```
<!DOCTYPE html>
<html>  <head>
<title>CSS 3 - Resize</title>
          <style>
               body
```

```
            {
                font-family: 'Segoe UI';
                font-size: 23px;
            }
            .root
            {
                resize: horizontal;
            }
            div
            {
                border: 2px solid;
                padding: 10px 40px;
                width: 500px;
                font-family: 'Segoe UI';

                resize: inherit;

                /*Options: none | both | horizontal | vertical */
                overflow: auto;
            }
        </style>
    </head>
    <body>


<div class="root">
        <div>Lorem Ipsum is simply dummy text of the printing
 and typesetting industry. Lorem Ipsum has been the industry's
 standard dummy text ever since the 1500s, when an unknown
 printer took a galley of type and scrambled it to make a type
 specimen book. It has survived not only five centuries, but also the
 leap into electronic typesetting, remaining essentially unchanged. It
 was popularised in the 1960s with the release of Letraset sheets
 containing Lorem Ipsum passages, and more recently with desktop
 publishing software like Aldus PageMaker including versions of
 Lorem Ipsum.</div>

        Lorem Ipsum is simply dummy text of the printing and
 typesetting industry. Lorem Ipsum has been the industry's standard
 dummy text ever since the 1500s, when an unknown printer took a
 galley of type and scrambled it to make a type specimen book. It
```

```
has survived not only five centuries, but also the leap into
electronic typesetting, remaining essentially unchanged. It was
popularised in the 1960s with the release of Letraset sheets
containing Lorem Ipsum passages, and more recently with desktop
publishing software like Aldus PageMaker including versions of
Lorem Ipsum.

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

CSS3 word-wrap Property
The word-wrap property allows long words to be able to be broken and wrap onto the next line.

Default value:
normal

word-wrap: normal|break-word|initial|inherit;

normal	Break words only at allowed break points
break-word	Allows unbreakable words to be broken
initial	Sets this property to its default value.
inherit	Inherits this property from its parent element.

```
<!DOCTYPE html>
<html>
      <head>
        <title>CSS 3 - Word Wrap</title>

        <style>
                  body
                  {
                        font-family: 'Segoe UI';
                        font-size: 23px;
                  }
            p.test1
            {
        width: 270px;
        border: 1px solid #000000;
    }
            p.test2
            {
        width: 270px;
        border: 1px solid #000000;
        word-wrap: break-word;
    }
  </style>
 </head>

 <body>
        <p class="test1">This paragraph contains a very long
word: thisisaveryveryveryveryveryverylongword. The long word will
break and wrap to the next line.</p>
        <p class="test2">This paragraph contains a very long
word: thisisaveryveryveryveryveryverylongword. The long word will
break and wrap to the next line.</p>

 </body>
</html>
```

CSS3 Colors

CSS supports color names, hexadecimal and RGB colors.
In addition, CSS3 also introduces:

- RGBA colors
- HSL colors
- HSLA colors
- opacity

RGBA Colors

RGBA color values are an extension of RGB color values with an alpha channel - which specifies the opacity for a color.
An RGBA color value is specified with: rgba(red, green, blue, alpha). The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (fully opaque).
rgba(255, 0, 0, 0.2);

```
rgba(255, 0, 0, 0.4);
rgba(255, 0, 0, 0.6);
rgba(255, 0, 0, 0.8);

<!DOCTYPE html>
<html>
      <head>
        <title>CSS 3 - RGBA</title>
      </head>
      <body>
            <div>Hello how are you!!</div>

        <style>
           body
                 {
               background-image: url(sample.png);
           }
           div
                 {
               border: 2px solid #a1a1a1;
               padding: 10px 40px;
               width: 300px;
               font-family: Candara;
               font-size: 30px;

                        background-color: rgba(50, 180, 110,0.2);
                /*Syntax: rgba(red, green, blue, opacity) */
           }
        </style>
      </body>
</html>
```

HSL Colors

HSL stands for Hue, Saturation and Lightness.
An HSL color value is specified with: hsl(hue, saturation, lightness).

1. Hue is a degree on the color wheel (from 0 to 360):
   - 0 (or 360) is red
   - 120 is green
   - 240 is blue

2. Saturation is a percentage value: 100% is the full color.
3. Lightness is also a percentage; 0% is dark (black) and 100% is white.

```
hsl(0, 100%, 30%);
hsl(0, 100%, 50%);
hsl(0, 100%, 70%);
hsl(0, 100%, 90%);
```

```
<!DOCTYPE html>
<html>
<head>
<title>CSS HSL Color Values</title>
<!-- HSL model (hue-saturation-lightness) using the hsl() functional
notation.

    Hue is represented as an angle (from 0 to 360) of the color
wheel or circle (i.e. the rainbow represented in a circle). -->

<style type="text/css">
   h1 {
      color: hsl(390,70%,60%);
   }
   p {
      background-color: hsl(480,50%,80%);
   }
</style>
</head>
<body>
   <h1>This is a heading</h1>
   <p>This is a paragraph.</p>
</body>
</html>
```

HSLA Colors

HSLA color values are an extension of HSL color values with an alpha channel - which specifies the opacity for a color.
An HSLA color value is specified with: hsla(hue, saturation, lightness, alpha), where the alpha parameter defines the opacity.

The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (fully opaque).

```
hsla(0, 100%, 30%, 0.3);
hsla(0, 100%, 50%, 0.3);
hsla(0, 100%, 70%, 0.3);
hsla(0, 100%, 90%, 0.3);
```

```
<!DOCTYPE html>
<html>
<head>
<title>CSS HSLA Color Values</title>

<!-- HSLA model (hue-saturation-lightness-alpha) using the hsla()
 functional notation.

Hue is represented as an angle (from 0 to 360) of the color wheel
 or circle (i.e. the rainbow represented in a circle).

The alpha parameter accepts a value from 0.0 (fully transparent) to
 1.0 (fully opaque)

-->
<style type="text/css">
    h1 {
        color: hsla(360,80%,50%,0.5);
    }
    p {
        background-color: hsla(480,60%,30%,0.3);
    }
</style>
</head>
<body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
</body>
</html>
```

Opacity

The CSS3 opacity property sets the opacity for a specified RGB value.

The opacity property value must be a number between 0.0 (fully transparent) and 1.0 (fully opaque).

rgb(255, 0, 0);opacity:0.2;
rgb(255, 0, 0);opacity:0.4;

CSS3 border-radius Property

With CSS3, you can give any element "rounded corners", by using the border-radius property.
CSS3 border-radius - Specify Each Corner
If you specify only one value for the border-radius property, this radius will be applied to all 4 corners.
However, you can specify each corner separately if you wish.

Here are the rules:

- Four values: first value applies to top-left, second value applies to top-right, third value applies to bottom-right, and fourth value applies to bottom-left corner
- Three values: first value applies to top-left, second value applies to top-right and bottom-left, and third value applies to bottom-right
- Two values: first value applies to top-left and bottom-right corner, and the second value applies to top-right and bottom-left corner
- One value: all four corners are rounded equally

Example -1 :

```
<!DOCTYPE html>
<html>
    <head>
      <title>CSS 3 - Border Radius</title>
    </head>
    <body>

        <div>The border-radius property allows you to add
rounded corners to elements.</div>
        <style>
            body
            {
```

```
                                font-family: 'Segoe UI';
                                font-size: 23px;
                        }
                        div
                        {
                                border: 2px solid #a1a1a1;
                                padding: 10px 40px;
                                background-color: pink;
                                width: 300px;

                                border-radius: 40px; /* IE 9+ and Firefox 4+ */
                                -moz-border-radius: 40px; /* Firefox 1, 2, 3*/
                                -webkit-border-radius: 40px; /* Chrome and
Safari */
                        }
                </style>
        </body>
</html>
```

Example -2 :

```
<!DOCTYPE html>
<html>
        <head>
          <title>CSS 3 - Border Radius</title>
        </head>
        <body>

                <input type="text">
                <input type="text">

                <style>
                        body, input
                        {
                                font-family: 'Segoe UI';
                                font-size: 23px;
                        }
                        input
                        {
                                border: 2px solid #a1a1a1;
                                padding: 5px;
```

```
                    background-color: green;
                    color: white;

                    border-radius: 25px 15px;
                    /* IE 9+ and Firefox 4+ */
                    -moz-border-radius: 25px 15px; /* Firefox 1, 2, 3
*/
                    -webkit-border-radius: 25px 15px; /* Chrome and
Safari */
               }
          </style>
     </body>
</html>
```

Example -3 :

```
<!DOCTYPE html>
<html>
     <head>
       <title>CSS 3 - Border Radius</title>
     </head>
     <body>

          <p>
               The border-radius property allows you to add rounded
corners to elements.
          </p>

          <style>
               body
               {
                    font-family: 'Segoe UI';
                    font-size: 23px;
               }
               p
               {
                    border: 2px solid #a1a1a1;
                    padding-left: 25px;
                    background-color: green;
                    color: white;
                    width: 300px;
                    /* Syntax: border-radius: topLeft topRight
bottomRight bottomLeft */
                    border-radius: 5px 45px 5px 45px; /* IE 9 and
New Firefox */
                    -moz-border-radius: 5px 45px 5px 45px; /* old
Firefox */
                    -webkit-border-radius: 5px 45px 5px 45px; /*
Chrome and Safari */
               }
          </style>
     </body>
</html>
```

CSS3 Shadow Effects

With CSS3 you can add shadow to text and to elements.

- text-shadow
- box-shadow

CSS3 Text Shadow

The CSS3 text-shadow property applies shadow to text.
In its simplest use, you only specify the horizontal shadow (2px) and the vertical shadow (2px):

CSS3 box-shadow Property

The CSS3 box-shadow property applies shadow to elements.

```
<!DOCTYPE html>
<html>
     <head>
       <title>CSS 3 - Shadow</title>
     </head>
     <body>

          <div>CSS 3 shadow example</div>
```

```
        <style>
                 body
                 {
                       font-family: 'Segoe UI';
                       font-size: 25px;
                 }
              div
                 {
              width: 300px;
              height: 100px;
              background-color: yellow;

                       /* Syntax: box-shadow: horizontalPosition
 verticalPosition blurRadius spread shadowColor */
               box-shadow: 20px 10px 10px 5px green; /* IE 9+ and
 Firefox 4+ */
              -moz-box-shadow: 20px 10px 10px 5px green; /*
 Firefox 1 to 3 */
              -webkit-box-shadow: 20px 10px 10px 5px green; /*
 Chrome and Safari */
                       /* Syntax: text-shadow: horizontalPosition
 verticalPosition blurRadius shadowColor */
                       text-shadow: 5px 5px 5px #FF0000;
            }
        </style>
      </body>
</html>


<!--
Syntax of box-shadow:
    Horizontal position: 35px
    Vertical position: 20px
    Blur Radius: 15px
    Spread: 7px
    Shadow color: rgba(r, g, b, a)
-->

<!--
Syntax of text-shadow:
    Horizontal Length: 35px
    Vertical Length: 20px
    Blur Radius: 15px
    Shadow color: rgba(r, g, b, a)
-->
```

Box Shadow:

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Example of CSS3 Box Shadow Effect</title>
<!--
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;
-->

<style type="text/css">
    .box{
            width: 200px;
            height: 150px;
            background: #ccc;
            box-shadow: 5px 5px 10px #999;
      }

      .box1{
    width: 200px;
    height: 150px;
    background: #000;
    box-shadow: 5px 5px 10px red, 10px 10px 20px yellow,15px
 15px 30px blue;
}
</style>
</head>
<body>
    <div class="box"></div></br>
    <div class="box1"></div>
```

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

Text Shadow:

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Example of CSS3 Text Shadow Effect</title>
<!--
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.

text-shadow: offset-x offset-y blur-radius color;

-->

<style type="text/css">
   h1 {
          text-shadow: 5px 5px 10px #666;
     }
     h2 {
          text-shadow: 5px 5px 10px red, 10px 10px 20px yellow;
     }
</style>
</head>
<body>
   <h1>This is heading 1</h1>
   <h2>This is heading 2</h2>
</body>
</html>
```

CSS3 Multi-column Layout

CSS3 Multi-column Properties

- column-count
- column-gap
- column-rule-style
- column-rule-width

- column-rule-color
- column-rule
- column-span
- column-width

CSS3 Create Multiple Columns
The column-count property specifies the number of columns an element should be divided into.

CSS3 Specify the Gap Between Columns
The column-gap property specifies the gap between the columns.

CSS3 Column Rules
The column-rule-style property specifies the style of the rule between columns:
The column-rule-width property specifies the width of the rule between columns:
The column-rule-color property specifies the color of the rule between columns:

The column-rule property is a shorthand property for setting all the column-rule-* properties above.

Multiple Columns:

```
<!DOCTYPE html>
<html>
      <head>
        <title>CSS 3 - Multiple Columns</title>
      </head>
      <body>

            <div class="newspaper1">Contrary to popular belief,
Lorem Ipsum is not simply random text. It has roots in a piece of
classical Latin literature from 45 BC, making it over 2000 years old.
Richard McClintock, a Latin professor at Hampden-Sydney College
in Virginia, looked up one of the more obscure Latin words,
consectetur, from a Lorem Ipsum passage, and going through the
cites of the word in classical literature, discovered the undoubtable
source. </div>
            <br>
        <div class="newspaper2">Contrary to popular belief, Lorem
Ipsum is not simply random text. It has roots in a piece of classical
Latin literature from 45 BC, making it over 2000 years old. Richard
McClintock, a Latin professor at Hampden-Sydney College in
Virginia, looked up one of the more obscure Latin words,
consectetur, from a Lorem Ipsum passage, and going through the
cites of the word in classical literature, discovered the undoubtable
source. </div>

      <style>
            body
            {
                font-family: 'Segoe UI';
                font-size: 23px;
            }
        .newspaper1
            {
                text-align: justify;
                border: 1px solid gray;
        }
            .newspaper2
            {
                text-align: justify;
                border: 1px solid gray;

          column-count: 4;
          /* IE9+ */
          -moz-column-count: 4; /* Firefox */
          -webkit-column-count: 4; /* Safari and Chrome */

          column-gap: 60px; /* IE9+ */
          -moz-column-gap:  60px; /* Firefox */
          -webkit-column-gap:  60px; /* Safari and Chrome */
        }
      </style>
     </body>
</html>
```

Column -count ::

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Example of CSS3 Multi-column Layout</title>
<style type="text/css">
      p {
            -webkit-column-count: 3; /* Chrome, Safari, Opera */
               -moz-column-count: 3; /* Firefox */
                         column-count: 3; /* Standard syntax */
      }
</style>
</head>
<body>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut
bibendum nisi egestas suscipit gravida. Sed velit nisl, tristique sed
dui mollis, porta tempus ligula. Phasellus vel orci vel arcu
pellentesque fermentum. Duis bibendum metus arcu. Aliquam eget
tortor vulputate, sollicitudin felis a, mollis libero. Aliquam vitae
consequat sapien, id blandit lectus. Integer ac nibh ac nulla
tincidunt accumsan sit amet sit amet risus. Integer id nisl urna. In
a enim elementum, auctor justo quis, tincidunt augue. Donec nibh
dui, congue non neque quis, semper aliquam felis. Praesent
efficitur massa vel convallis euismod. Quisque metus lectus,
consectetur sit amet justo in, venenatis faucibus nunc. Aenean
faucibus, enim id egestas convallis, felis magna mattis est, in
ultricies est urna ac nisl.

    Cras placerat quis tortor quis molestie. Nullam imperdiet gravida
velit eget sollicitudin. Nunc dictum pretium justo vel congue.
Praesent auctor leo maximus leo aliquam, eget vehicula tortor
tincidunt. Vestibulum finibus venenatis dui, nec lobortis mauris
convallis id. Maecenas porttitor erat tellus, at vulputate eros
euismod a. In aliquam, dolor et bibendum consequat, eros felis
ultricies lorem, ac fermentum arcu metus in magna. Integer auctor
sapien a massa porta, et suscipit sapien sollicitudin. Maecenas vel
hendrerit nibh. Curabitur convallis interdum ornare. Curabitur justo
nibh, pretium ac convallis vitae, consectetur sit amet orci. Nunc
non enim non ligula efficitur venenatis et at metus. Duis turpis
velit, lacinia interdum purus ac, venenatis semper lacus. Aenean
mattis fermentum odio ut suscipit.
```

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

Column-width:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Example of Setting CSS3 Column Count or Width</title>
<style type="text/css">
    p {
        -webkit-column-width: 150px; /* Chrome, Safari, Opera */
          -moz-column-width: 150px; /* Firefox */
                column-width: 150px; /* Standard syntax */
    }
</style>
</head>
<body>
  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut
bibendum nisi egestas suscipit gravida. Sed velit nisl, tristique sed
dui mollis, porta tempus ligula. Phasellus vel orci vel arcu
pellentesque fermentum. Duis bibendum metus arcu. Aliquam eget
tortor vulputate, sollicitudin felis a, mollis libero. Aliquam vitae
consequat sapien, id blandit lectus. Integer ac nibh ac nulla
tincidunt accumsan sit amet sit amet risus. Integer id nisl urna. In
a enim elementum, auctor justo quis, tincidunt augue. Donec nibh
dui, congue non neque quis, semper aliquam felis. Praesent
efficitur massa vel convallis euismod. Quisque metus lectus,
consectetur sit amet justo in, venenatis faucibus nunc. Aenean
faucibus, enim id egestas convallis, felis magna mattis est, in
ultricies est urna ac nisl.

  Cras placerat quis tortor quis molestie. Nullam imperdiet gravida
velit eget sollicitudin. Nunc dictum pretium justo vel congue.
Praesent auctor leo maximus leo aliquam, eget vehicula tortor
tincidunt. Vestibulum finibus venenatis dui, nec lobortis mauris
convallis id. Maecenas porttitor erat tellus, at vulputate eros
euismod a. In aliquam, dolor et bibendum consequat, eros felis
ultricies lorem, ac fermentum arcu metus in magna. Integer auctor
```

```
sapien a massa porta, et suscipit sapien sollicitudin. Maecenas vel
hendrerit nibh. Curabitur convallis interdum ornare. Curabitur justo
nibh, pretium ac convallis vitae, consectetur sit amet orci. Nunc
non enim non ligula efficitur venenatis et at metus. Duis turpis
velit, lacinia interdum purus ac, venenatis semper lacus. Aenean
mattis fermentum odio ut suscipit.
  </p>
</body>
</html>
```

Column-gap

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Example of Setting CSS3 Column Gap</title>
<style type="text/css">
    p {
        /* Chrome, Safari, Opera */
        -webkit-column-count: 3;
        -webkit-column-gap: 100px;
        /* Firefox */
        -moz-column-count: 3;
        -moz-column-gap: 100px;
        /* Standard syntax */
        column-count: 3;
        column-gap: 100px;
    }
</style>
</head>
<body>
  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut
bibendum nisi egestas suscipit gravida. Sed velit nisl, tristique sed
dui mollis, porta tempus ligula. Phasellus vel orci vel arcu
pellentesque fermentum. Duis bibendum metus arcu. Aliquam eget
tortor vulputate, sollicitudin felis a, mollis libero. Aliquam vitae
consequat sapien, id blandit lectus. Integer ac nibh ac nulla
tincidunt accumsan sit amet sit amet risus. Integer id nisl urna. In
a enim elementum, auctor justo quis, tincidunt augue. Donec nibh
dui, congue non neque quis, semper aliquam felis. Praesent
```

```
efficitur massa vel convallis euismod. Quisque metus lectus,
consectetur sit amet justo in, venenatis faucibus nunc. Aenean
faucibus, enim id egestas convallis, felis magna mattis est, in
ultricies est urna ac nisl.

   Cras placerat quis tortor quis molestie. Nullam imperdiet gravida
velit eget sollicitudin. Nunc dictum pretium justo vel congue.
Praesent auctor leo maximus leo aliquam, eget vehicula tortor
tincidunt. Vestibulum finibus venenatis dui, nec lobortis mauris
convallis id. Maecenas porttitor erat tellus, at vulputate eros
euismod a. In aliquam, dolor et bibendum consequat, eros felis
ultricies lorem, ac fermentum arcu metus in magna. Integer auctor
sapien a massa porta, et suscipit sapien sollicitudin. Maecenas vel
hendrerit nibh. Curabitur convallis interdum ornare. Curabitur justo
nibh, pretium ac convallis vitae, consectetur sit amet orci. Nunc
non enim non ligula efficitur venenatis et at metus. Duis turpis
velit, lacinia interdum purus ac, venenatis semper lacus. Aenean
mattis fermentum odio ut suscipit.
  </p>
</body>
</html>
```

Column-Rule

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Example of Setting CSS3 Column Rules</title>
<!--

column-countSpecifies the number of columns inside a multi-
 column element.
column-fill     Specifies how content is spread across columns.
column-gap   Specifies the gap between the columns.
column-rule   Specifies a straight line or rule, to be drawn between
 each column.
column-rule-color Specifies the color of the rule between columns.
column-rule-style Specifies the style of the rule between columns.
column-rule-widthSpecifies the width of the rule between columns.
column-span Specifies how many columns an element spans
```

```
 across.
column-width Specifies the optimal width of the columns.
columns A shorthand property for setting both the column-width
 and the column-count properties at the same time.

-->
<style type="text/css">
    p {
        /* Chrome, Safari, Opera */
        -webkit-column-count: 3;
        -webkit-column-gap: 100px;
        -webkit-column-rule: 2px solid red;
        /* Firefox */
        -moz-column-count: 3;
        -moz-column-gap: 100px;
        -moz-column-rule: 2px solid red;
        /* Standard syntax */
        column-count: 3;
        column-gap: 100px;
        column-rule: 2px solid red;
    }
</style>
</head>
<body>
   <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut
 bibendum nisi egestas suscipit gravida. Sed velit nisl, tristique sed
 dui mollis, porta tempus ligula. Phasellus vel orci vel arcu
 pellentesque fermentum. Duis bibendum metus arcu. Aliquam eget
 tortor vulputate, sollicitudin felis a, mollis libero. Aliquam vitae
 consequat sapien, id blandit lectus. Integer ac nibh ac nulla
 tincidunt accumsan sit amet sit amet risus. Integer id nisl urna. In
 a enim elementum, auctor justo quis, tincidunt augue. Donec nibh
 dui, congue non neque quis, semper aliquam felis. Praesent
 efficitur massa vel convallis euismod. Quisque metus lectus,
 consectetur sit amet justo in, venenatis faucibus nunc. Aenean
 faucibus, enim id egestas convallis, felis magna mattis est, in
 ultricies est urna ac nisl.

   Cras placerat quis tortor quis molestie. Nullam imperdiet gravida
 velit eget sollicitudin. Nunc dictum pretium justo vel congue.
 Praesent auctor leo maximus leo aliquam, eget vehicula tortor
```

tincidunt. Vestibulum finibus venenatis dui, nec lobortis mauris convallis id. Maecenas porttitor erat tellus, at vulputate eros euismod a. In aliquam, dolor et bibendum consequat, eros felis ultricies lorem, ac fermentum arcu metus in magna. Integer auctor sapien a massa porta, et suscipit sapien sollicitudin. Maecenas vel hendrerit nibh. Curabitur convallis interdum ornare. Curabitur justo nibh, pretium ac convallis vitae, consectetur sit amet orci. Nunc non enim non ligula efficitur venenatis et at metus. Duis turpis velit, lacinia interdum purus ac, venenatis semper lacus. Aenean mattis fermentum odio ut suscipit.

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

CSS3 Transitions

CSS3 transitions allows you to change property values smoothly (from one value to another), over a given duration.

How to Use CSS3 Transitions?

To create a transition effect, you must specify two things:

- the CSS property you want to add an effect to
- the duration of the effect

Note: If the duration part is not specified, the transition will have no effect, because the default value is 0.
Specify the Speed Curve of the Transition
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

Delay the Transition Effect

The transition-delay property specifies a delay (in seconds) for the transition effect.

CSS3 Animations

CSS3 animations allows animation of most HTML elements without using JavaScript or Flash!

What are CSS3 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 you want.

To use CSS3 animation, you must first specify some keyframes for the animation.

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

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.

Delay an Animation

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

Set How Many Times an Animation Should Run

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

Run Animation in Reverse Direction or Alternate Cycles

The animation-direction property is used to let an animation run in reverse direction or alternate cycles.

CSS3 Transforms

CSS3 transforms allow you to translate, rotate, scale, and skew elements.
A transformation is an effect that lets an element change shape, size and position.
CSS3 supports 2D and 3D transformations.

CSS3 2D Transforms

- translate()
- rotate()
- scale()
- skewX()
- skewY()
- matrix()

The translate() Method
The translate() method moves an element from its current position (according to the parameters given for the X-axis and the Y-axis)

The rotate() Method
The rotate() method rotates an element clockwise or counter-clockwise according to a given degree.

The scale() Method
The scale() method increases or decreases the size of an element (according to the parameters given for the width and height).

The skewX() Method
The skewX() method skews an element along the X-axis by the given angle.

The skewY() Method
The skewY() method skews an element along the Y-axis by the given angle.

The skew() Method
The skew() method skews an element along the X and Y-axis by the given angles.

The matrix() Method
The matrix() method combines all the 2D transform methods into one.

The matrix() method take six parameters, containing mathematic functions, which allows you to rotate, scale, move (translate), and skew elements.
The parameters are as follow:
matrix(scaleX(),skewY(),skewX(),scaleY(),translateX(),translateY()):

CSS3 Gradients

The CSS3 gradient feature allows you to create a gradient from one color to another without using any images.

Using CSS3 Gradients

The CSS3 gradient feature provides a flexible solution to generate smooth transitions between two or more colors. Earlier, to achieve such effect we had to use the images. Using CSS3 gradients you can reduce the download time and saves the bandwidth usages. The elements with gradients can be scaled up or down to any

extent without losing the quality, also the output will render much faster because it is generated by the browser.
Gradients are available in two styles: linear and radial.

Creating CSS3 Linear Gradients

To create a linear gradient you must define at least two color stops. However to create more complex gradient effects you can define more color stops. Color stops are the colors you want to render smooth transitions among. You can also set a starting point and a direction (or an angle) along which the gradient effect is applied. The basic syntax of creating the linear gradients using the keywords can be given with:
linear-gradient(direction, color-stop1, color-stop2, ...)

Linear Gradient - Top to Bottom
Linear Gradient - Left to Right
Linear Gradient - Diagonal

Setting Direction of Linear Gradients Using Angles

If you want more control over the direction of the gradient, you can set the angle, instead of the predefined keywords. The angle "0deg" creates a bottom to top gradient, and positive angles represent clockwise rotation, that means the angle "90deg" creates a left to right gradient. The basic syntax of creating the linear gradients using angle can be given with:
linear-gradient(angle, color-stop1, color-stop2, ...)

Creating Linear Gradients Using Multiple Color Stops

You can also create gradients for more than two colors. The following example will show you how to create a linear gradient using multiple color stops. All colors are evenly spaced.

Setting the Location Color Stops

Color stops are points along the gradient line that will have a specific color at that location. The location of a color stop can be specified either as a percentage, or as an absolute length. You may specify as many color stops as you like to achieve the desired effect.

Repeating the Linear Gradients

You can repeat linear gradients using the repeating-linear-gradient() function.

Creating CSS3 Radial Gradients

In a radial gradient color emerge from a single point and smoothly spread outward in a circular or elliptical shape rather than fading from one color to another in a single direction as with linear gradients. The basic syntax of creating a radial gradient can be given with:

radial-gradient(shape size at position, color-stop1, color-stop2, ...);

The arguments of the radial-gradient() function has the following meaning:

position — Specifies the starting point of the gradient, which can be specified in units (px, em, or percentages) or keyword (left, bottom, etc).

shape — Specifies the shape of the gradient's ending shape. It can be circle or ellipse.

size — Specifies the size of the gradient's ending shape. The default is farthest-side.

Setting the Shape of Radial Gradients

The shape argument of the radial-gradient() function is used to define the ending shape of the radial gradient. It can take the value circle or ellipse. Here's is an example:

Setting the Size of Radial Gradients

The size argument of the radial-gradient() function is used to define the size of the gradient's ending shape. Size can be set using units or the keywords: closest-side,farthest-side, closest-corner, farthest-corner.

Repeating the Radial Gradients

You can also repeat radial gradients using the repeating-radial-gradient() function.

CSS3 @font-face Rule

Definition and Usage

With the @font-face rule, web designers do no longer have to use one of the "web-safe" fonts.

In the new @font-face rule you must first define a name for the font (e.g. myFirstFont), and then point to the font file.

Tip: Use lowercase letters for the font URL. Uppercase letters can give unexpected results in IE!

Browser Code :

moz	=	mozilla firefox
webkit	=	chrome and safari
o	=	opera

CSS3 Media Queries

CSS media queries enable you to format your documents to be presented correctly on different size of output devices.

Media Queries and Responsive Web Design
Media queries allow you to customize the presentation of your web pages for a specific range of devices like mobile phones, tablets, desktops, etc. without any change in markups. A media query consists of a media type and zero or more expressions that match the type and conditions of a particular media features such as device width or screen resolution.

Since media query is a logical expression it can be resolve to either true or false. The result of the query will be true if the media type specified in the media query matches the type of device the document is being displayed on, as well as all expressions in the media query are satisfied. When a media query is true, the related style sheet or style rules are applied to the target device. Here's a simple example of the media query for standard devices.

Tip:Media queries are an excellent way to create responsive layouts. Using media queries you can customize your website differently for users browsing on devices like smart phones or tablets without changing the actual content of the page.

Changing Column Width Based on Screen Size
You can use the CSS media query for changing the web page width and related elements to offer the best viewing experience for the user on different devices.
The following style rules will automatically change the width of the container element based on the screen or viewport size. For example, if the viewport width is less than 768 pixels it will cover the 100% of the viewport width, if it is greater than the 768 pixels but less than the 1024 pixels it will be 750 pixels wide, and so on.

Note:You can use the CSS3 box-sizing property on the elements to create more intuitive and flexible layouts with much less effort.

Changing Layouts Based on Screen Size
You can also use the CSS media query for making your multi-

column website layout more adaptable and responsive for devices through little customization.

The following style rule will create a two column layout if the viewport size is greater than or equal to 768 pixels, but if less than that it'll be rendered as one column layout.

```
   /* Smartphones (portrait and landscape) ---------- */
   @media screen and (min-width: 320px) and (max-width: 480px)
{
      body{
               background: #7ce7e1;
           }
   }
   /* Smartphones (portrait) ---------- */
   @media screen and (max-width: 320px){
      body{
               background: #ffd280;
           }
   }
   /* Smartphones (landscape) ---------- */
   @media screen and (min-width: 321px){
      body{
               background: #9ddfbb;
           }
   }
   /* tablets, iPads (portrait and landscape) ---------- */
   @media screen and (min-width: 768px) and (max-width:
1024px){
      body{
               background: #ffb497;
           }
   }
```

```
/* tablets, iPads (portrait) ---------- */
@media screen and (min-width: 768px){
   body{
            background: #f0e68c;
       }
}
/* tablets, iPads (landscape) ---------- */
@media screen and (min-width: 1024px){
   body{
            background: #d6b3f4;
       }
}
/* Desktops and laptops ---------- */
@media screen and (min-width: 1224px){
   body{
            background: #d8ff9d;
       }
}
/* Large screens ---------- */
@media screen and (min-width: 1824px){
   body{
            background: #ffc0cb;
       }
```