CSS3 Modules

CSS3 has been split into "modules". It contains the "old CSS specification" (which has been split into smaller pieces). In addition, new modules are added.

Some of the most important CSS3 modules are:

* Selectors
* Box Model
* Backgrounds and Borders
* Image Values and Replaced Content
* Text Effects
* 2D/3D Transformations
* Animations
* Multiple Column Layout
* User Interface

## CSS3 border-radius Property

#rcorners1 {  
    border-radius: 25px;  
    background: #73AD21;  
    padding: 20px;   
    width: 200px;  
    height: 150px;   
}  
  
#rcorners2 {  
    border-radius: 25px;  
    border: 2px solid #73AD21;  
    padding: 20px;   
    width: 200px;  
    height: 150px;   
}  
  
#rcorners3 {  
    border-radius: 25px;  
    background: url(paper.gif);  
    background-position: left top;  
    background-repeat: repeat;  
    padding: 20px;   
    width: 200px;  
    height: 150px;   
}

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

#rcorners4 {  
    border-radius: 15px 50px 30px 5px;  
    background: #73AD21;  
    padding: 20px;   
    width: 200px;  
    height: 150px;   
}  
  
#rcorners5 {  
    border-radius: 15px 50px 30px;  
    background: #73AD21;  
    padding: 20px;   
    width: 200px;  
    height: 150px;   
}  
  
#rcorners6 {  
    border-radius: 15px 50px;  
    background: #73AD21;  
    padding: 20px;   
    width: 200px;  
    height: 150px;   
}

You could also create elliptical corners:

#rcorners7 {  
    border-radius: 50px/15px;  
    background: #73AD21;  
    padding: 20px;   
    width: 200px;  
    height: 150px;   
}  
  
#rcorners8 {  
    border-radius: 15px/50px;  
    background: #73AD21;  
    padding: 20px;   
    width: 200px;  
    height: 150px;   
}  
  
#rcorners9 {  
    border-radius: 50%;  
    background: #73AD21;  
    padding: 20px;   
    width: 200px;  
    height: 150px;  
}

## CSS3 Rounded Corners Properties

|  |  |
| --- | --- |
| **Property** | **Description** |
| [border-radius](http://www.w3schools.com/cssref/css3_pr_border-radius.asp) | A shorthand property for setting all the four border-\*-\*-radius properties |
| [border-top-left-radius](http://www.w3schools.com/cssref/css3_pr_border-top-left-radius.asp) | Defines the shape of the border of the top-left corner |
| [border-top-right-radius](http://www.w3schools.com/cssref/css3_pr_border-top-right-radius.asp) | Defines the shape of the border of the top-right corner |
| [border-bottom-right-radius](http://www.w3schools.com/cssref/css3_pr_border-bottom-right-radius.asp) | Defines the shape of the border of the bottom-right corner |
| [border-bottom-left-radius](http://www.w3schools.com/cssref/css3_pr_border-bottom-left-radius.asp) | Defines the shape of the border of the bottom-left corner |

# **CSS3 Border Images**

## CSS3 border-image Property

The CSS3 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

#borderimg {  
    border: 10px solid transparent;  
    padding: 15px;  
    -webkit-border-image: url(border.png) 30 round; /\* Safari 3.1-5 \*/  
    -o-border-image: url(border.png) 30 round; /\* Opera 11-12.1 \*/  
    border-image: url(border.png) 30 round;  
}

#borderimg {  
    border: 10px solid transparent;  
    padding: 15px;  
    -webkit-border-image: url(border.png) 30 stretch; /\* Safari 3.1-5 \*/  
    -o-border-image: url(border.png) 30 stretch; /\* Opera 11-12.1 \*/  
    border-image: url(border.png) 30 stretch;  
}

#borderimg1 {  
    border: 10px solid transparent;  
    padding: 15px;  
    -webkit-border-image: url(border.png) 50 round; /\* Safari 3.1-5 \*/  
    -o-border-image: url(border.png) 50 round; /\* Opera 11-12.1 \*/  
    border-image: url(border.png) 50 round;  
}  
  
#borderimg2 {  
    border: 10px solid transparent;  
    padding: 15px;  
    -webkit-border-image: url(border.png) 20% round; /\* Safari 3.1-5 \*/  
    -o-border-image: url(border.png) 20% round; /\* Opera 11-12.1 \*/  
    border-image: url(border.png) 20% round;  
}  
  
#borderimg3 {  
    border: 10px solid transparent;  
    padding: 15px;  
    -webkit-border-image: url(border.png) 30% round; /\* Safari 3.1-5 \*/  
    -o-border-image: url(border.png) 30% round; /\* Opera 11-12.1 \*/  
    border-image: url(border.png) 30% round;  
}

# **CSS3 Backgrounds**

## CSS3 Multiple Backgrounds

#example1 {  
    background-image: url(img\_flwr.gif), url(paper.gif);  
    background-position: right bottom, left top;  
    background-repeat: no-repeat, repeat;  
}

#example1 {  
    background: url(img\_flwr.gif) right bottom no-repeat, url(paper.gif) left top repeat;  
}

## CSS3 Background Size

#div1 {  
    background: url(img\_flower.jpg);  
    background-size: 100px 80px;  
    background-repeat: no-repeat;  
}

#div1 {  
    background: url(img\_flower.jpg);  
    background-size: contain;  
    background-repeat: no-repeat;  
}  
#div2 {  
    background: url(img\_flower.jpg);  
    background-size: cover;  
    background-repeat: no-repeat;  
}

## Define Sizes of Multiple Background Images

#example1 {  
    background: url(img\_flwr.gif) left top no-repeat, url(img\_flwr.gif) right bottom no-repeat, url(paper.gif) left top repeat;  
    background-size: 50px, 130px, auto;  
}

## Full Size Background Image

html {  
    background: url(img\_flower.jpg) no-repeat center fixed;   
    background-size: cover;  
}

## CSS3 background-clip Property

#example1 {

border: 10px dotted black;

padding:35px;

background: yellow;

}

#example2 {

border: 10px dotted black;

padding:35px;

background: yellow;

background-clip: padding-box;

}

#example3 {

border: 10px dotted black;

padding:35px;

background: yellow;

background-clip: content-box;

}

CSS3 Colors

CSS supports color names, hexadecimal and RGB colors.

In addition, CSS3 also introduces:

* RGBA colors
* HSL colors
* HSLA colors
* opacity

#p1 {background-color: rgba(255, 0, 0, 0.3);}  /\* red with opacity \*/  
#p2 {background-color: rgba(0, 255, 0, 0.3);}  /\* green with opacity \*/  
#p3 {background-color: rgba(0, 0, 255, 0.3);}  /\* blue with opacity \*/

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.

#p1 {background-color: hsl(120, 100%, 50%);}  /\* green \*/  
#p2 {background-color: hsl(120, 100%, 75%);}  /\* light green \*/  
#p3 {background-color: hsl(120, 100%, 25%);}  /\* dark green \*/  
#p4 {background-color: hsl(120, 60%, 70%);}   /\* pastel green \*/

## 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).

#p1 {background-color: hsla(120, 100%, 50%, 0.3);}  /\* green with opacity \*/  
#p2 {background-color: hsla(120, 100%, 75%, 0.3);}  /\* light green with opacity \*/  
#p3 {background-color: hsla(120, 100%, 25%, 0.3);}  /\* dark green with opacity \*/  
#p4 {background-color: hsla(120, 60%, 70%, 0.3);}   /\* pastel green with opacity \*/

## 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).

#p1 {background-color:rgb(255,0,0);opacity:0.6;}  /\* red with opacity \*/  
#p2 {background-color:rgb(0,255,0);opacity:0.6;}  /\* green with opacity \*/  
#p3 {background-color:rgb(0,0,255);opacity:0.6;}  /\* blue with opacity \*/

# **CSS3 Gradients**

CSS3 gradients let you display smooth transitions between two or more specified colors.

Earlier, you had to use images for these effects. However, by using CSS3 gradients you can reduce download time and bandwidth usage. In addition, elements with gradients look better when zoomed, because the gradient is generated by the browser.

CSS3 defines two types of gradients:

* **Linear Gradients (goes down/up/left/right/diagonally)**
* **Radial Gradients (defined by their center)**

## CSS3 Linear Gradients

To create a linear gradient you must define at least two 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 with the gradient effect.

#grad {  
    background: red; /\* For browsers that do not support gradients \*/  
    background: -webkit-linear-gradient(red, yellow); /\* For Safari 5.1 to 6.0 \*/  
    background: -o-linear-gradient(red, yellow); /\* For Opera 11.1 to 12.0 \*/  
    background: -moz-linear-gradient(red, yellow); /\* For Firefox 3.6 to 15 \*/  
    background: linear-gradient(red, yellow); /\* Standard syntax \*/  
}

**Linear Gradient - Left to Right**

#grad {  
  background: red; /\* For browsers that do not support gradients \*/  
  background: -webkit-linear-gradient(left, red , yellow); /\* For Safari 5.1 to 6.0 \*/  
  background: -o-linear-gradient(right, red, yellow); /\* For Opera 11.1 to 12.0 \*/  
  background: -moz-linear-gradient(right, red, yellow); /\* For Firefox 3.6 to 15 \*/  
  background: linear-gradient(to right, red , yellow); /\* Standard syntax \*/  
}

**Linear Gradient - Diagonal**

You can make a gradient diagonally by specifying both the horizontal and vertical starting positions.

#grad {  
  background: red; /\* For browsers that do not support gradients \*/  
  background: -webkit-linear-gradient(left top, red, yellow); /\* For Safari 5.1 to 6.0 \*/  
  background: -o-linear-gradient(bottom right, red, yellow); /\* For Opera 11.1 to 12.0 \*/  
  background: -moz-linear-gradient(bottom right, red, yellow); /\* For Firefox 3.6 to 15 \*/  
  background: linear-gradient(to bottom right, red, yellow); /\* Standard syntax \*/  
}

## Using Angles

If you want more control over the direction of the gradient, you can define an angle, instead of the predefined directions (to bottom, to top, to right, to left, to bottom right, etc.).

#grad {  
  background: red; /\* For browsers that do not support gradients \*/  
  background: -webkit-linear-gradient(-90deg, red, yellow); /\* For Safari 5.1 to 6.0 \*/  
  background: -o-linear-gradient(-90deg, red, yellow); /\* For Opera 11.1 to 12.0 \*/  
  background: -moz-linear-gradient(-90deg, red, yellow); /\* For Firefox 3.6 to 15 \*/  
  background: linear-gradient(-90deg, red, yellow); /\* Standard syntax \*/  
}

## Using Multiple Color Stops

#grad {  
  background: red; /\* For browsers that do not support gradients \*/  
  background: -webkit-linear-gradient(red, yellow, green); /\* For Safari 5.1 to 6.0 \*/  
  background: -o-linear-gradient(red, yellow, green); /\* For Opera 11.1 to 12.0 \*/  
  background: -moz-linear-gradient(red, yellow, green); /\* For Firefox 3.6 to 15 \*/  
  background: linear-gradient(red, yellow, green); /\* Standard syntax \*/  
}

#grad {  
  background: red; /\* For browsers that do not support gradients \*/  
  /\* For Safari 5.1 to 6.0 \*/  
  background: -webkit-linear-gradient(left,red,orange,yellow,green,blue,indigo,violet);  
  /\* For Opera 11.1 to 12.0 \*/  
  background: -o-linear-gradient(left,red,orange,yellow,green,blue,indigo,violet);  
  /\* For Fx 3.6 to 15 \*/  
  background: -moz-linear-gradient(left,red,orange,yellow,green,blue,indigo,violet);  
  /\* Standard syntax \*/  
  background: linear-gradient(to right, red,orange,yellow,green,blue,indigo,violet);   
}

## Using Transparency

CSS3 gradients also support transparency, which can be used to create fading effects.

To add transparency, we use the rgba() function to define the color stops. The last parameter in the rgba() function can be a value from 0 to 1, and it defines the transparency of the color: 0 indicates full transparency, 1 indicates full color (no transparency).

#grad {  
  background: red; /\* For browsers that do not support gradients \*/  
  background: -webkit-linear-gradient(left,rgba(255,0,0,0),rgba(255,0,0,1)); /\*Safari 5.1-6\*/  
  background: -o-linear-gradient(right,rgba(255,0,0,0),rgba(255,0,0,1)); /\*Opera 11.1-12\*/  
  background: -moz-linear-gradient(right,rgba(255,0,0,0),rgba(255,0,0,1)); /\*Fx 3.6-15\*/  
  background: linear-gradient(to right, rgba(255,0,0,0), rgba(255,0,0,1));/\*Standard\*/  
}

## Repeating a linear-gradient

#grad {  
  background: red; /\* For browsers that do not support gradients \*/  
  /\* Safari 5.1 to 6.0 \*/  
  background: -webkit-repeating-linear-gradient(red, yellow 10%, green 20%);  
  /\* Opera 11.1 to 12.0 \*/  
  background: -o-repeating-linear-gradient(red, yellow 10%, green 20%);  
  /\* Firefox 3.6 to 15 \*/  
  background: -moz-repeating-linear-gradient(red, yellow 10%, green 20%);  
  /\* Standard syntax \*/  
  background: repeating-linear-gradient(red, yellow 10%, green 20%);  
}

## CSS3 Radial Gradients

#grad {  
  background: red; /\* For browsers that do not support gradients \*/  
  background: -webkit-radial-gradient(red, yellow, green); /\* Safari 5.1 to 6.0 \*/  
  background: -o-radial-gradient(red, yellow, green); /\* For Opera 11.6 to 12.0 \*/  
  background: -moz-radial-gradient(red, yellow, green); /\* For Firefox 3.6 to 15 \*/  
  background: radial-gradient(red, yellow, green); /\* Standard syntax \*/  
}

#grad {  
  background: red; /\* For browsers that do not support gradients \*/  
  background: -webkit-radial-gradient(red 5%, yellow 15%, green 60%); /\* Safari 5.1-6.0 \*/  
  background: -o-radial-gradient(red 5%, yellow 15%, green 60%); /\* For Opera 11.6-12.0 \*/  
  background: -moz-radial-gradient(red 5%, yellow 15%, green 60%); /\* For Firefox 3.6-15 \*/  
  background: radial-gradient(red 5%, yellow 15%, green 60%); /\* Standard syntax \*/  
}

## Set Shape

#grad {  
  background: red; /\* For browsers that do not support gradients \*/  
  background: -webkit-radial-gradient(circle, red, yellow, green); /\* Safari \*/  
  background: -o-radial-gradient(circle, red, yellow, green); /\* Opera 11.6 to 12.0 \*/  
  background: -moz-radial-gradient(circle, red, yellow, green); /\* Firefox 3.6 to 15 \*/  
  background: radial-gradient(circle, red, yellow, green); /\* Standard syntax \*/  
}

## Use of Different Size Keywords

The size parameter defines the size of the gradient. It can take four values:

* **closest-side**
* **farthest-side**
* **closest-corner**
* **farthest-corner**

#grad1 {

height: 150px;

width: 150px;

background: -webkit-radial-gradient(60% 55%, closest-side, red, yellow, black); /\* Safari 5.1 to 6.0 \*/

background: -o-radial-gradient(60% 55%, closest-side, red, yellow, black); /\* For Opera 11.6 to 12.0 \*/

background: -moz-radial-gradient(60% 55%, closest-side, red, yellow, black); /\* For Firefox 3.6 to 15 \*/

background: radial-gradient(closest-side at 60% 55%, red, yellow, black); /\* Standard syntax (must be last) \*/

}

#grad2 {

height: 150px;

width: 150px;

background: -webkit-radial-gradient(60% 55%, farthest-side, red, yellow, black); /\* Safari 5.1 to 6.0 \*/

background: -o-radial-gradient(60% 55%, farthest-side, red, yellow, black); /\* For Opera 11.6 to 12.0 \*/

background: -moz-radial-gradient(60% 55%, farthest-side, red, yellow, black); /\* For Firefox 3.6 to 15 \*/

background: radial-gradient(farthest-side at 60% 55%, red, yellow, black); /\* Standard syntax (must be last) \*/

}

#grad3 {

height: 150px;

width: 150px;

background: -webkit-radial-gradient(60% 55%, closest-corner, red, yellow, black); /\* Safari 5.1 to 6.0 \*/

background: -o-radial-gradient(60% 55%, closest-corner, red, yellow, black); /\* For Opera 11.6 to 12.0 \*/

background: -moz-radial-gradient(60% 55%, closest-corner, red, yellow, black); /\* For Firefox 3.6 to 15 \*/

background: radial-gradient(closest-corner at 60% 55%, red, yellow, black); /\* Standard syntax (must be last) \*/

}

#grad4 {

height: 150px;

width: 150px;

background: -webkit-radial-gradient(60% 55%, farthest-corner, red, yellow, black); /\* Safari 5.1 to 6.0 \*/

background: -o-radial-gradient(60% 55%, farthest-corner, red, yellow, black); /\* For Opera 11.6 to 12.0 \*/

background: -moz-radial-gradient(60% 55%, farthest-corner, red, yellow, black); /\* For Firefox 3.6 to 15 \*/

background: radial-gradient(farthest-corner at 60% 55%, red, yellow, black); /\* Standard syntax (must be last) \*/

}

# **CSS3 Shadow Effects**

## CSS3 Shadow Effects

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

In this chapter you will learn about the following properties:

* text-shadow
* box-shadow

## CSS3 Text Shadow

h1 {  
    text-shadow: 2px 2px;  
}

h1 {  
    text-shadow: 2px 2px red;  
}

blur effect to the shadow

h1 {  
    text-shadow: 2px 2px 5px red;  
}

h1 {  
    color: white;  
    text-shadow: 2px 2px 4px #000000;  
}

h1 {  
    text-shadow: 0 0 3px #FF0000;  
}

## Multiple Shadows

h1 {  
    text-shadow: 0 0 3px #FF0000, 0 0 5px #0000FF;  
}

shows a white text with black, blue, and darkblue shadow:

h1 {  
    color: white;  
    text-shadow: 1px 1px 2px black, 0 0 25px blue, 0 0 5px darkblue;  
}

## CSS3 box-shadow Property

div {  
    box-shadow: 10px 10px;  
}

add a color to the shadow

div {  
    box-shadow: 10px 10px grey;  
}

add a blur effect to the shadow

div {  
    box-shadow: 10px 10px 5px grey;  
}

## Cards

using the box-shadow property to create paper-like cards

<style>

div.polaroid {

width: 250px;

box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);

text-align: center;

}

div.container {

padding: 10px;

}

</style>

</head>

<body>

<h2>Polaroid Images / Cards</h2>

<p>The box-shadow property can be used to create paper-like cards:</p>

<div class="polaroid">

<img src="rock600x400.jpg" alt="Norway" style="width:100%">

<div class="container">

<p>Hardanger, Norway</p>

</div>

</div>

<style>

div.card {

width: 250px;

box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);

text-align: center;

}

div.header {

background-color: #4CAF50;

color: white;

padding: 10px;

font-size: 40px;

}

div.container {

padding: 10px;

}

</style>

</head>

<body>

<h2>Cards</h2>

<p>The box-shadow property can be used to create paper-like cards:</p>

<div class="card">

<div class="header">

<h1>1</h1>

</div>

<div class="container">

<p>January 1, 2016</p>

</div>

</div>

## CSS3 Text

CSS3 contains several new text features.

In this chapter you will learn about the following text properties:

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

## CSS3 Text Overflow

<style>

p.test1 {

white-space: nowrap;

width: 200px;

border: 1px solid #000000;

overflow: hidden;

text-overflow: clip;

}

p.test2 {

white-space: nowrap;

width: 200px;

border: 1px solid #000000;

overflow: hidden;

text-overflow: ellipsis;

}

</style>

</head>

<body>

<p>The following two paragraphs contains a long text that will not fit in the box.</p>

<p>text-overflow: clip:</p>

<p class="test1">This is some long text that will not fit in the box</p>

<p>text-overflow: ellipsis:</p>

<p class="test2">This is some long text that will not fit in the box</p>

## CSS3 Word Wrapping

<style>

p.test {

width: 11em;

border: 1px solid #000000;

word-wrap: break-word;

}

</style>

</head>

<body>

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

## CSS3 Word Breaking

<style>

p.test1 {

width: 140px;

border: 1px solid #000000;

word-break: keep-all;

}

p.test2 {

width: 140px;

border: 1px solid #000000;

word-break: break-all;

}

</style>

</head>

<body>

<p class="test1">This paragraph contains some text. This line will-break-at-hyphens.</p>

<p class="test2">This paragraph contains some text. The lines will break at any character.</p>

<p><b>Note:</b> The word-break property is not supported in Opera 12 and earlier versions.</p>

# **CSS3 Web Fonts**

## Different Font Formats

**TrueType Fonts (TTF)**

TrueType is a font standard developed in the late 1980s, by Apple and Microsoft. TrueType is the most common font format for both the Mac OS and Microsoft Windows operating systems.

**OpenType Fonts (OTF)**

OpenType is a format for scalable computer fonts. It was built on TrueType, and is a registered trademark of Microsoft. OpenType fonts are used commonly today on the major computer platforms.

**The Web Open Font Format (WOFF)**

WOFF is a font format for use in web pages. It was developed in 2009, and is now a W3C Recommendation. WOFF is essentially OpenType or TrueType with compression and additional metadata. The goal is to support font distribution from a server to a client over a network with bandwidth constraints.

**The Web Open Font Format (WOFF 2.0)**

TrueType/OpenType font that provides better compression than WOFF 1.0.

**SVG Fonts/Shapes**

SVG fonts allow SVG to be used as glyphs when displaying text. The SVG 1.1 specification define a font module that allows the creation of fonts within an SVG document. You can also apply CSS to SVG documents, and the @font-face rule can be applied to text in SVG documents.

**Embedded OpenType Fonts (EOT)**

EOT fonts are a compact form of OpenType fonts designed by Microsoft for use as embedded fonts on web pages.

<style>

@font-face {

font-family: myFirstFont;

src: url(sansation\_light.woff);

}

div {

font-family: myFirstFont;

}

</style>

</head>

<body>

<div>

With CSS3, websites can finally use fonts other than the pre-selected "web-safe" fonts.

</div>

<p><b>Note:</b> Internet Explorer 8 and earlier, do not support the @font-face rule.</p>

## Using Bold Text

@font-face {  
    font-family: myFirstFont;  
    src: url(sansation\_bold.woff);  
    font-weight: bold;  
}

# **CSS3 2D Transforms**

## 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.

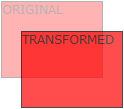
CSS3 supports 2D and 3D transformations.

## CSS3 2D Transforms

In this chapter you will learn about the following 2D transformation methods:

* 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).

<style>

div {

width: 300px;

height: 100px;

background-color: yellow;

border: 1px solid black;

-ms-transform: translate(50px,100px); /\* IE 9 \*/

-webkit-transform: translate(50px,100px); /\* Safari \*/

transform: translate(50px,100px); /\* Standard syntax \*/

}

</style>

</head>

<body>

<div>

The translate() method moves an element from its current position. This div element is moved 50 pixels to the right, and 100 pixels down from its current position.

</div>

The rotate() Method



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

<style>

div {

width: 300px;

height: 100px;

background-color: yellow;

border: 1px solid black;

}

div#myDiv {

-ms-transform: rotate(20deg); /\* IE 9 \*/

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

transform: rotate(20deg); /\* Standard syntax \*/

}

</style>

</head>

<body>

<div>

This a normal div element.

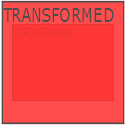
</div>

<div id="myDiv">

The rotate() method rotates an element clockwise or counter-clockwise. This div element is rotated clockwise 20 degrees.

</div>

The scale() Method



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

<style>

div {

margin: 150px;

width: 200px;

height: 100px;

background-color: yellow;

border: 1px solid black;

border: 1px solid black;

-ms-transform: scale(2,3); /\* IE 9 \*/

-webkit-transform: scale(2,3); /\* Safari \*/

transform: scale(2,3); /\* Standard syntax \*/

}

</style>

</head>

<body>

<p>The scale() method increases or decreases the size of an element.</p>

<div>

This div element is two times of its original width, and three times of its original height.

</div>

## 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.

<style>

div {

width: 300px;

height: 100px;

background-color: yellow;

border: 1px solid black;

}

div#myDiv {

-ms-transform: skewX(20deg); /\* IE 9 \*/

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

transform: skewX(20deg); /\* Standard syntax \*/

}

</style>

</head>

<body>

<p>The skewX() method skews an element along the X-axis by the given angle.</p>

<div>

This a normal div element.

</div>

<div id="myDiv">

This div element is skewed 20 degrees along the X-axis.

</div>

<style>

div {

width: 300px;

height: 100px;

background-color: yellow;

border: 1px solid black;

}

div#myDiv {

-ms-transform: skewY(20deg); /\* IE 9 \*/

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

transform: skewY(20deg); /\* Standard syntax \*/

}

</style>

</head>

<body>

<p>The skewY() method skews an element along the Y-axis by the given angle.</p>

<div>

This a normal div element.

</div>

<div id="myDiv">

This div element is skewed 20 degrees along the Y-axis.

</div>

## The skew() Method

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

<style>

div {

width: 300px;

height: 100px;

background-color: yellow;

border: 1px solid black;

}

div#myDiv {

-ms-transform: skew(20deg,10deg); /\* IE 9 \*/

-webkit-transform: skew(20deg,10deg); /\* Safari \*/

transform: skew(20deg,10deg); /\* Standard syntax \*/

}

</style>

</head>

<body>

<p>The skew() method skews an element into a given angle.</p>

<div>

This a normal div element.

</div>

<div id="myDiv">

This div element is skewed 20 degrees along the X-axis, and 10 degrees along the Y-axis.

</div>

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.

<style>

div {

width: 300px;

height: 100px;

background-color: yellow;

border: 1px solid black;

}

div#myDiv1 {

-ms-transform: matrix(1, -0.3, 0, 1, 0, 0); /\* IE 9 \*/

-webkit-transform: matrix(1, -0.3, 0, 1, 0, 0); /\* Safari \*/

transform: matrix(1, -0.3, 0, 1, 0, 0); /\* Standard syntax \*/

}

div#myDiv2 {

-ms-transform: matrix(1, 0, 0.5, 1, 150, 0); /\* IE 9 \*/

-webkit-transform: matrix(1, 0, 0.5, 1, 150, 0); /\* Safari \*/

transform: matrix(1, 0, 0.5, 1, 150, 0); /\* Standard syntax \*/

}

</style>

</head>

<body>

<p>The matrix() method combines all the 2D transform methods into one.</p>

<div>

This a normal div element.

</div>

<div id="myDiv1">

Using the matrix() method.

</div>

<div id="myDiv2">

Another use of the matrix() method.

</div>

# **CSS3 3D Transforms**

## CSS3 3D Transforms

In this chapter you will learn about the following 3D transformation methods:

* rotateX()
* rotateY()
* rotateZ()

<style>

div {

width: 300px;

height: 100px;

background-color: yellow;

border: 1px solid black;

}

div#myDiv {

-webkit-transform: rotateX(150deg); /\* Safari \*/

transform: rotateX(150deg); /\* Standard syntax \*/

}

</style>

</head>

<body>

<div>

This a normal div element.

</div>

<div id="myDiv">

The rotateX() method rotates an element around its X-axis at a given degree. This div element is rotated 150 degrees.

</div>

<p><b>Note:</b> Internet Explorer 9 (and earlier versions) does not support the rotateX() method.</p>

## CSS3 Transitions

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

<style>

div {

width: 100px;

height: 100px;

background: red;

-webkit-transition: width 2s; /\* For Safari 3.1 to 6.0 \*/

transition: width 2s;

}

div:hover {

width: 300px;

}

</style>

</head>

<body>

<p><b>Note:</b> This example does not work in Internet Explorer 9 and earlier versions.</p>

<div></div>

<p>Hover over the div element above, to see the transition effect.</p>

# **CSS3 Animations**

## 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.

<style>

div {

width: 100px;

height: 100px;

background-color: red;

-webkit-animation-name: example; /\* Chrome, Safari, Opera \*/

-webkit-animation-duration: 4s; /\* Chrome, Safari, Opera \*/

animation-name: example;

animation-duration: 4s;

}

/\* Chrome, Safari, Opera \*/

@-webkit-keyframes example {

from {background-color: red;}

to {background-color: yellow;}

}

/\* Standard syntax \*/

@keyframes example {

from {background-color: red;}

to {background-color: yellow;}

}

</style>

</head>

<body>

<p><b>Note:</b> This example does not work in Internet Explorer 9 and earlier versions.</p>

<div></div>

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

<style>

div {

width: 100px;

height: 100px;

background-color: red;

-webkit-animation-name: example; /\* Chrome, Safari, Opera \*/

-webkit-animation-duration: 4s; /\* Chrome, Safari, Opera \*/

animation-name: example;

animation-duration: 4s;

}

/\* Chrome, Safari, Opera \*/

@-webkit-keyframes example {

0% {background-color: red;}

25% {background-color: yellow;}

50% {background-color: blue;}

100% {background-color: green;}

}

/\* Standard syntax \*/

@keyframes example {

0% {background-color: red;}

25% {background-color: yellow;}

50% {background-color: blue;}

100% {background-color: green;}

}

</style>

</head>

<body>

<p><b>Note:</b> This example does not work in Internet Explorer 9 and earlier versions.</p>

## Delay an Animation

div {  
    width: 100px;  
    height: 100px;  
    position: relative;  
    background-color: red;  
    animation-name: example;  
    animation-duration: 4s;  
    animation-delay: 2s;  
}

## Set How Many Times an Animation Should Run

div {  
    width: 100px;  
    height: 100px;  
    position: relative;  
    background-color: red;  
    animation-name: example;  
    animation-duration: 4s;  
    animation-iteration-count: 3;  
}

## Run Animation in Reverse Direction or Alternate Cycles

div {  
    width: 100px;  
    height: 100px;  
    position: relative;  
    background-color: red;  
    animation-name: example;  
    animation-duration: 4s;  
    animation-iteration-count: 3;  
    animation-direction: reverse;  
}

# **CSS Images**

<style>

img {

border-radius: 8px;

}

</style>

</head>

<body>

<h2>Rounded Images</h2>

<p>Use the border-radius property to create rounded images:</p>

<img src="paris.jpg" alt="Paris" width="400" height="300">

<style>

img {

border-radius: 50%;

}

</style>

</head>

<body>

<h2>Rounded Images</h2>

<p>Use the border-radius property to create circled images:</p>

<img src="paris.jpg" alt="Paris" width="400" height="300">