

CSY1018

Web Development

Week 7

CSS3 Properties

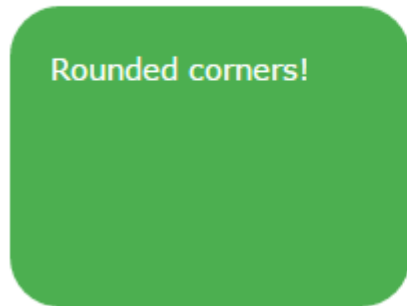
CSS3 Introduction

- Latest standard for CSS
- completely backwards-compatible with earlier versions of CSS
- Most of the new CSS3 properties are implemented in modern browsers
- 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

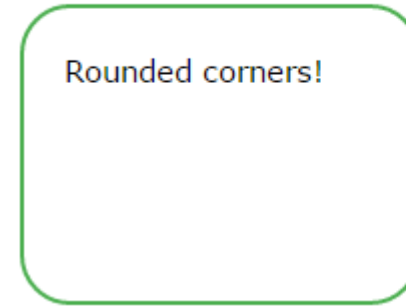
CSS3 Rounded Corners

- you can give any element "rounded corners"
- Border-radius property

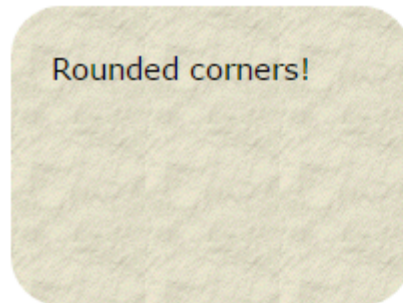
1. Rounded corners for an element with a specified background color:



2. Rounded corners for an element with a border:



3. Rounded corners for an element with a background image:



CSS3 Rounded Corners

1. Four values - `border-radius: 15px 50px 30px 5px;`



2. Three values - `border-radius: 15px 50px 30px;`

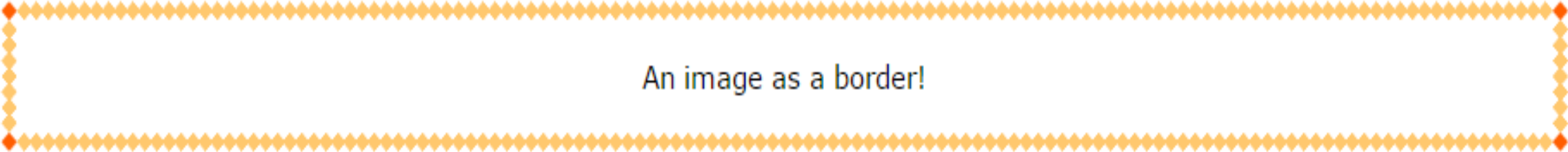


3. Two values - `border-radius: 15px 50px;`



CSS3 Border Images

- With the CSS3 border-image property, you can set an image to be used as the border around an element
- allows you to specify an image to be used instead of the normal border around an element
- The property has three parts:
 - The image to use as the border
 - Where to slice the image
 - Define whether the middle sections should be repeated or stretched

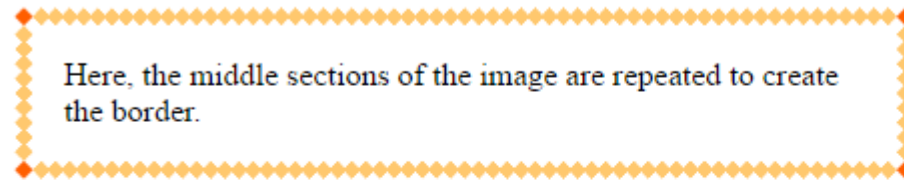


An image as a border!

CSS3 Border Images: Example1

```
#borderimg {  
  width:400px;  
  border: 10px solid;  
  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;  
}
```

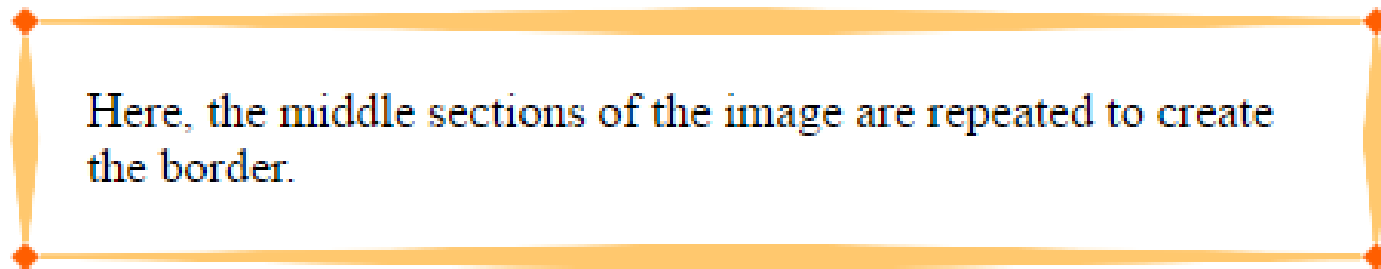
The border-image property specifies an image to be used as the border around an element:



Here is the original image:



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



CSS3 Backgrounds

- CSS3 contains a few new background properties, which allow greater control of the background element
- You will also learn about the following new CSS3 properties
 - background-size
 - background-origin
 - background-clip
- to add multiple background images for an element
- 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

CSS3 Backgrounds: Example1

```
#example1 {  
    width:400px;  
    background-image: url(img_flwr.gif), url(paper.gif);  
    background-position: right bottom, left top;  
    background-repeat: no-repeat, repeat;  
    padding: 15px;  
}
```

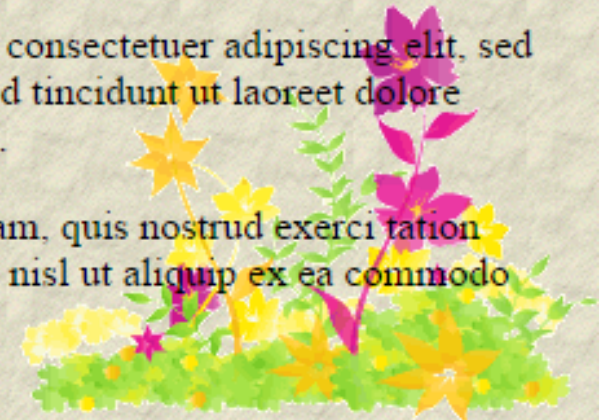
Note: Shorthand

**background: url(img_flwr.gif) right bottom no-repeat,
url(paper.gif) left top repeat;**

Lorem Ipsum Dolor

Lorem ipsum dolor sit amet, consectetur 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.



CSS3 Backgrounds: Example2:background-size

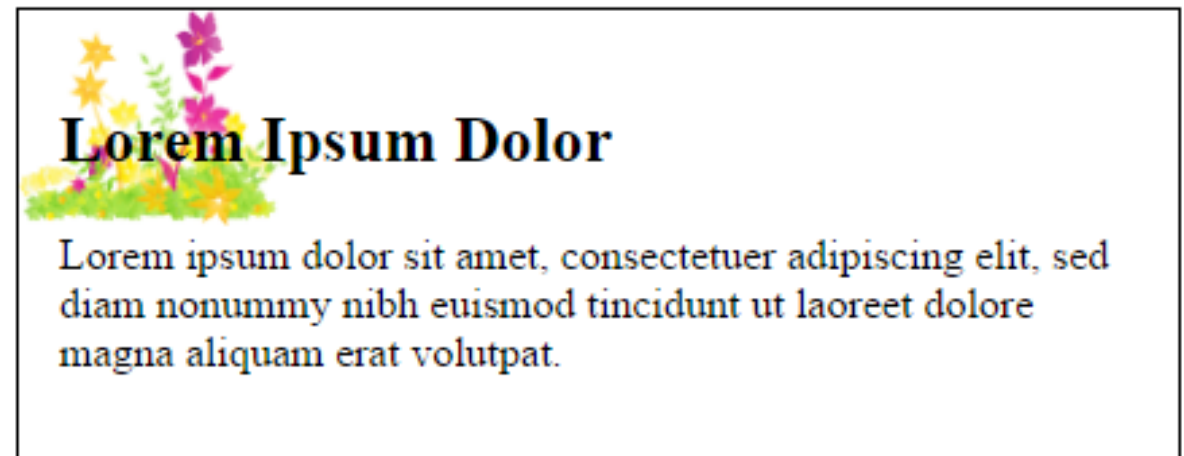
```
#example1 {  
  border: 1px solid black;  
  background:url(img_flwr.gif);  
  background-repeat: no-repeat;  
  padding:15px;  
}
```

```
#example2 {  
  border: 1px solid black;  
  background:url(img_flwr.gif);  
  background-size: 100px 80px;  
  background-repeat: no-repeat;  
  padding:15px;  
}
```

Original background-image:



Resized background-image:



CSS3 Colors

- CSS supports color names, hexadecimal and RGB colors
- In addition, CSS3 also introduces
 - RGBA colors (Red, Green, Blue, Alpha value)
 - HSL colors (Hue, Saturation, Lightness)
 - HSLA colors (Hue, Saturation, Lightness, Alpha:opacity)
 - opacity

CSS3 Colors: RGBA

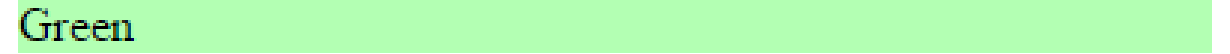
```
#p1 {background-color:rgba(255,0,0,0.3);}
#p2 {background-color:rgba(0,255,0,0.3);}
#p3 {background-color:rgba(0,0,255,0.3);}
#p4 {background-color:rgba(192,192,192,0.3);}
#p5 {background-color:rgba(255,255,0,0.3);}
#p6 {background-color:rgba(255,0,255,0.3);}
```

RGBA colors:

Red



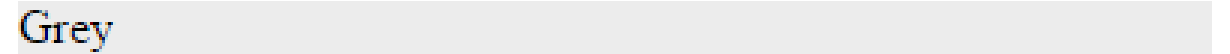
Green



Blue



Grey



Yellow



Cerise



CSS3 Colors: Opacity

- sets the opacity for the whole element (both background color and text will be opaque/transparent)
- value must be a number between 0.0 (fully transparent) and 1.0 (fully opaque)

```
#p1 {background-color:rgb(255,0,0);opacity:1;}  
#p2 {background-color:rgb(255,0,0);opacity:0.8;}  
#p3 {background-color:rgb(255,0,0);opacity:0.6;}  
#p4 {background-color:rgb(255,0,0);opacity:0.4;}  
#p5 {background-color:rgb(255,0,0);opacity:0.2;}  
#p6 {background-color:rgb(255,0,0);opacity:0;}
```

Elements with opacity:

Red

Green

Blue

Grey

Yellow

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 Gradients: Linear Gradient

```
#grad1 {  
  width:460px;  
  height: 200px;  
  background: red;  
  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);  
}
```

Linear Gradient - Top to Bottom

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



Note: Internet Explorer 9 and earlier versions do not support gradients.

CSS3 Gradients: Linear Gradient(left to right)

```
#grad1 {  
    height: 200px;  
    background: red;  
    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 (must be last) */  
}
```

Linear Gradient - Left to Right

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



Note: Internet Explorer 9 and earlier versions do not support gradients.

CSS3 Shadow Effects

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

- Text-shadow

Text shadow effect!

- Box-shadow

This is a yellow <div> element with
a black box-shadow

CSS3 Text Shadow:example1

- Text-shadow: horizontal-shadow vertical shadow

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

Text-shadow effect!

Note: Internet Explorer 9 and earlier versions, do not support the text-shadow property.

CSS3 Text Shadow:example2(with color)

- Text-shadow: horizontal-shadow vertical shadow

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

Text-shadow effect!

CSS3 Text Shadow:example2(with blur)

- Text-shadow: horizontal-shadow vertical shadow

```
h1 {  
  text-shadow: 2px 2px 8px red;  
}
```

Text-shadow effect!

CSS3 Text Shadow:example2(multiple shadows)

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

Text-shadow effect!

CSS3 Box Shadow:example

```
div {  
  width: 300px;  
  height: 100px;  
  padding: 15px;  
  background-color: yellow;  
  box-shadow: 10px 10px;  
}
```



This is a div element with a box-shadow

CSS3 Box Shadow:example(with color)

```
div {  
  width: 300px;  
  height: 100px;  
  padding: 15px;  
  background-color: yellow;  
  box-shadow: 10px 10px;  
}
```

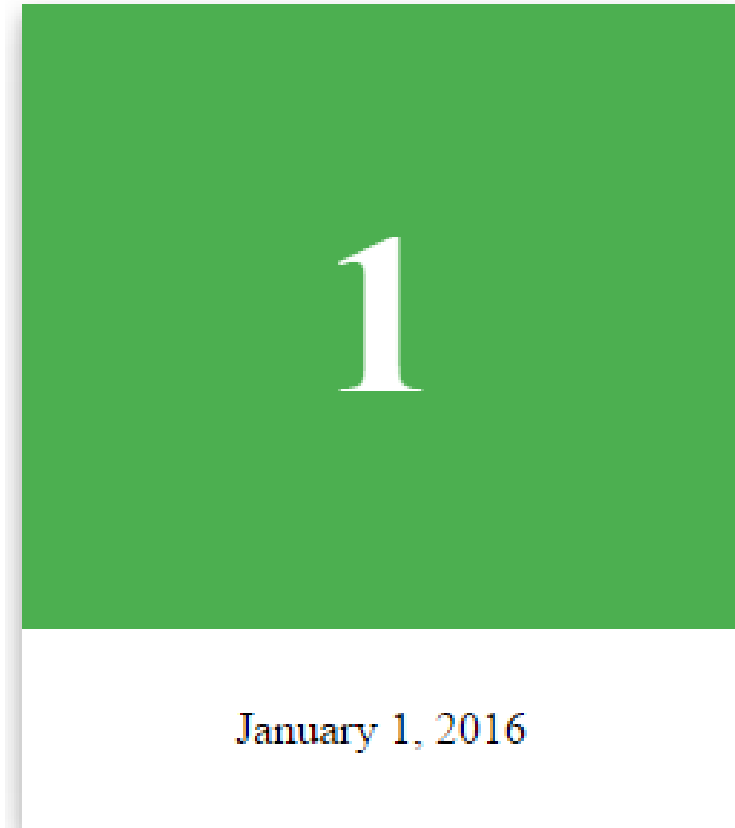


This is a div element with a box-shadow

CSS3 Box Shadow:example

```
<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>

<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>
```

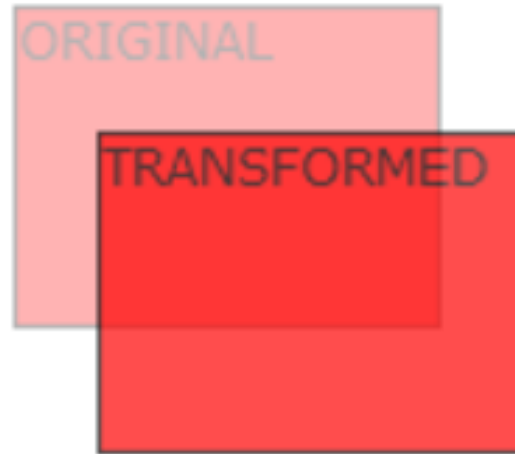


CSS3 2D Transforms

- allow you to translate, rotate, scale, and skew elements.
- is an effect that lets an element change shape, size and position.
- CSS3 supports 2D and 3D transformations
- following 2D transformation methods
 - Translate()
 - Rotate()
 - scale()
 - skewX()
 - skewY()
 - Matrix()

CSS3 2D Transforms(translate)

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



CSS3 2D Transforms(translate)

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

```
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 */  
}
```

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.

CSS3 2D Transforms(rotate)

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



CSS3 2D Transforms(rotate clockwise)

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

```
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 */  
}
```

This a normal div element.

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

CSS3 2D Transforms(rotate anti-clockwise)

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

```
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 */  
}
```

This a normal div element.

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

CSS3 2D Transforms(scale)

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



CSS3 2D Transforms(scale increase)

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

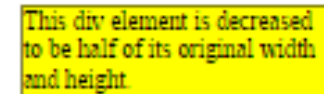
```
div {  
  margin: 150px;  
  width: 200px;  
  height: 100px;  
  background-color: yellow;  
  border: 1px solid black;  
  border: 1px solid black;  
  -ms-transform: scale(2,3); /* IE 9 */  
  -webkit-transform: scale(2,3); /* Safari */  
  transform: scale(2,3); /* Standard syntax */  
}
```

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

CSS3 2D Transforms(scale decrease)

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

```
div {  
  margin: 150px;  
  width: 200px;  
  height: 100px;  
  background-color: yellow;  
  border: 1px solid black;  
  border: 1px solid black;  
  -ms-transform: scale(0.5,0.5); /* IE 9 */  
  -webkit-transform: scale(0.5,0.5); /* Safari */  
  transform: scale(0.5,0.5); /* Standard syntax */  
}
```



This div element is decreased to be half of its original width and height.

CSS3 2D Transforms (skewX)

- skews an element along the X-axis by the given angle

```
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 */  
}
```

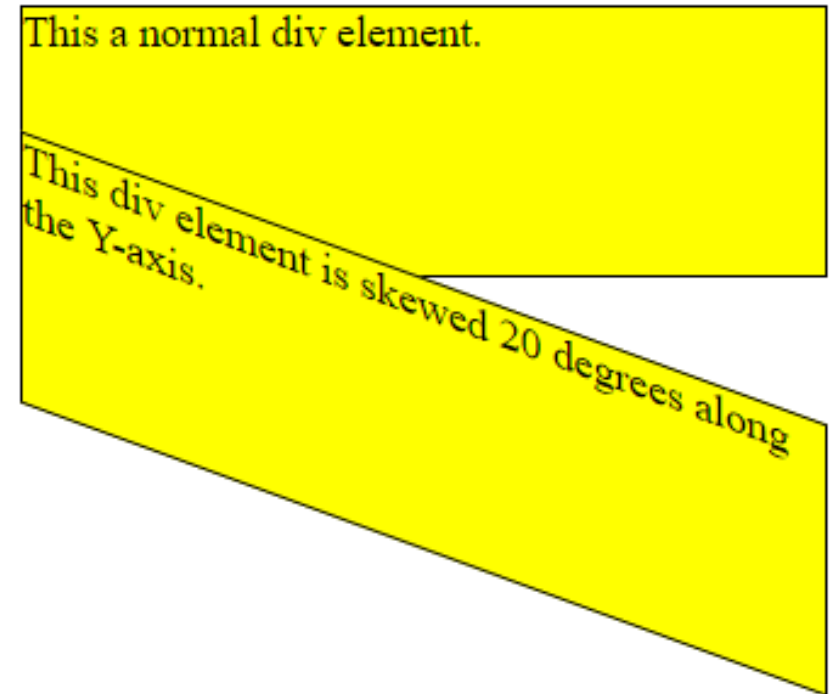
This a normal div element.

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

CSS3 2D Transforms (skewY)

- skews an element along the Y-axis by the given angle

```
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 */  
}
```



CSS3 2D Transforms (skew)

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

```
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 */  
}
```

This a normal div element.

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

CSS3 2D Transforms (matrix)

- combines all the 2D transform methods into one
- takes 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(),rotate())`:

CSS3 2D Transforms (matrix)

```
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 */  
}
```

This a normal div element.

Using the matrix() method.

Another use of the matrix() method.

CSS3 3D Transforms

- CSS3 allows you to format your elements using 3D transformations
- the following 3D transformation methods
 - rotateX()
 - rotateY()
 - rotateX()

CSS3 3D Transforms (rotateX)

- rotates an element around its X-axis at a given degree

```
div {  
  width: 300px;  
  height: 100px;  
  background-color: yellow;  
  border: 1px solid black;  
}
```

```
div#myDiv {  
  -webkit-transform: rotateX(150deg); /* Safari */  
  transform: rotateX(150deg); /* Standard syntax */  
}
```

This a normal div element.

element is rotated 120 degrees.
around its X-axis at a given degree. This div
The rotateX() method rotates an element

CSS3 3D Transforms (rotateY)

- rotates an element around its Y-axis at a given degree

```
div {  
  width: 300px;  
  height: 100px;  
  background-color: yellow;  
  border: 1px solid black;  
}  
  
div#myDiv {  
  -webkit-transform: rotateY(150deg); /* Safari */  
  transform: rotateY(150deg); /* Standard syntax */  
}
```

This a normal div element.

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

CSS3 3D Transforms (rotateZ)

- rotates an element around its Z-axis at a given degree

```
div {  
  width: 300px;  
  height: 100px;  
  background-color: yellow;  
  border: 1px solid black;  
}  
  
div#myDiv {  
  -webkit-transform: rotateZ(90deg); /* Safari */  
  transform: rotateZ(90deg); /* Standard syntax */  
}
```

This a normal d

The rotateZ() method rotates an element around its Z-axis at a given degree. This div element is rotated 90 degrees.

Note: Internet Explorer (earlier versions) d

CSS3 Transitions

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

CSS3 Transitions

```
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;  
}
```

CSS3 Transitions

- 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:

```
div {  
    width: 100px;  
    height: 100px;  
    background: red;  
    -webkit-transition: width 2s, height 4s; /* For Safari 3.1 to 6.0  
*/  
    transition: width 2s, height 4s;  
}  
  
div:hover {  
    width: 300px;  
    height: 300px;  
}
```

CSS3 Transitions

- Specify the Speed Curve of the Transition
- The transition-timing-function property specifies the speed curve of the transition effect
 - 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

CSS3 Transitions

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

```
div {  
    width: 100px;  
    height: 100px;  
    background: red;  
    -webkit-transition: width 3s; /* Safari */  
    -webkit-transition-delay: 1s; /* Safari */  
    transition: width 3s;  
    transition-delay: 1s;  
}
```

CSS3 Transitions

- Transition + Transformation
- The following example also adds a transformation to the transition effect

```
div {  
    width: 300px;  
    height: 300px;  
    background: red;  
    -webkit-transition: width 2s, height 2s, -webkit-transform 2s; /*  
Safari */  
    transition: width 2s, height 2s, transform 2s;  
}  
  
div:hover {  
    width: 350px;  
    height: 350px;  
    -webkit-transform: rotate(180deg); /* Safari */  
    transform: rotate(180deg);  
}
```

CSS3 Animations

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

CSS3 Animations

- 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"

```
div {  
    width: 100px;  
    height: 100px;  
    background-color: red;  
    -webkit-animation-name: example; /* Safari 4.0 - 8.0 */  
    -webkit-animation-duration: 4s; /* Safari 4.0 - 8.0 */  
    animation-name: example;  
    animation-duration: 4s;  
}  
  
/* Safari 4.0 - 8.0 */  
@-webkit-keyframes example {  
    from {background-color: red;}  
    to {background-color: yellow;}  
}  
  
/* Standard syntax */  
@keyframes example {  
    from {background-color: red;}  
    to {background-color: yellow;}  
}
```

CSS3 Animations

- 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:

```
div {
  width: 100px;
  height: 100px;
  background-color: red;
  -webkit-animation-name: example; /* Safari 4.0 - 8.0 */
  -webkit-animation-duration: 4s; /* Safari 4.0 - 8.0 */
  animation-name: example;
  animation-duration: 4s;
}

/* Safari 4.0 - 8.0 */
@-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;}
}
```

CSS3 Animations

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

```
div {
  width: 100px;
  height: 100px;
  background-color: red;
  position: relative;
  -webkit-animation-name: example; /* Safari 4.0 - 8.0 */
  -webkit-animation-duration: 4s; /* Safari 4.0 - 8.0 */
  animation-name: example;
  animation-duration: 4s;
}

/* Standard syntax */
@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;}
}
```

CSS3 Animations

- 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:

```
div {
  width: 100px;
  height: 100px;
  background-color: red;
  position: relative;
  -webkit-animation-name: example; /* Safari 4.0 - 8.0 */
  -webkit-animation-duration: 4s; /* Safari 4.0 - 8.0 */
  -webkit-animation-delay: 2s; /* Safari 4.0 - 8.0 */
  animation-name: example;
  animation-duration: 4s;
  animation-delay: 2s;
}
/* Standard syntax */
@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;}
}
```

CSS3 Animations

- 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:

```
div {
  width: 100px;
  height: 100px;
  background-color: red;
  position: relative;
  -webkit-animation-name: example; /* Safari 4.0 - 8.0 */
  -webkit-animation-duration: 4s; /* Safari 4.0 - 8.0 */
  -webkit-animation-iteration-count: 3; /* Safari 4.0 - 8.0 */
  animation-name: example;
  animation-duration: 4s;
  animation-iteration-count: 3;
}

/* Standard syntax */
@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;}
}
```