CSS3 CURRICULUM

1. CSS3 INTRODUCTION
   - WHAT IS CSS3?
   - CSS3 MODULES
2. CSS3 BACKGROUNDS
   - MULTIPLE BACKGROUND
   - BACKGROUND-SIZE
   - BACKGROUND-ORIGINE
   - BACKGROND-CLIP
3. CSS3 TEXT
   - TEXT-OVERFLOW
   - WORD-WRAP
   - WORD-BREAK
4. CSS3 FONTS
   - FONT-FAMILY
   - FONT-STRETCH
   - FONT-WEIGHT
   - FONT-STYLE
5. CSS3 COLORS
   - RGBA COLORS
   - HSL COLORS
   - HSLA COLORS
   - OPACITY
6. CSS3 ROUNDED CORNERS
   - CSS3 BORDER-RADIUS
   - CSS3 BORDER-RADIUS –SPECIFY EACH CORNER
7. CSS3 BUTTONS
   - BASIC BUTTON STYLING
   - BUTTON COLORS
   - BUTTON SIZES
   - ROUNDED BUTTONS
   - COLORED BUTTON BORDERS
   - HOVERABLE BUTTONS
   - SHADOW BUTTONS
   - DISABLED BUTTONS
   - BUTTONS WIDTH
   - BUTTON GROUPS
   - BORDERED BUTTON GROUPS
   - ANIMATED BUTTONS
8. CSS3 BORDER IMAGES
   - CSS3 BORDER IMAGES-PROPERTY
   - CSS3 BORDER IMAGES-DIFFERENT SLICE VALUES
9. CSS3 GRADIENTS
   - LINEAR GRADIENT
   - USING ANGLES
   - USING MULTIPLE COLOR STOPS
   - USING TRANSPARENCY
   - REPEATING A LINEAR-GRADIENT
   - CSS3 RADIAL GRADIENTS
   - SET SHAPE
   - USE OF DIFFERENT SIZE KEYWORDS
   - REPEATING A RADIAL-GRADIENT
10. CSS3 SHADOWS
    - TEXT SHADOW EFFECT
    - BOX-SHADOW PROPERTY
11. CSS3 2D TRANSFORMS
    - THE TRANSLATE() METHOD
    - THE ROTATE() METHOD
    - THE SCALE() METHOD
    - THE SKEW X() METHOD
    - THE SKEW Y() METHOD
    - THE SKEW() METHOD
    - THE MATRIX() METHOD
12. CSS3 3D TRANSFORMS
    - THE ROTATEX() METHOD
    - THE ROTATEY() METHOD
    - THE ROTATEZ() METHOD
13. CSS3 TRANSITIONS
    - HOW TO USE CSS3 TRANSITIONS?
    - CHANGE SEVERAL PROPERTY VALUES
    - SPECIFY THE SPEED CURVE OF THE TRANSITION
    - DELAY THE TRANSITION EFFECT
    - TRANSITION + TRANSFORMATION
14. CSS3 ANIMATIONS
    - WHAT ARE CSS3 ANIMATIONS?
    - THE @KEYFRAMES RULE

- DELAY AN ANIMATION
- SET HOW MANY TIMES AN ANIMATION SHOULD RUN
- RUN ANIMATION IN REVERSE DIRECTION OR ALTERNATE CYCLES
- SPECIFY THE SPEED CURVE OF THE ANIMATION
- ANIMATION SHORTHAND PROPERTY

15. CSS3 MULTIPLE COLUMNS
    - CSS3 MULTI-COLUMN
    - CSS3 CREATE MULTIPLE COLUMNS
    - CSS3 SPECIFY THE GAP BETWEEN COLUMNS
    - CSS3 COLUMN RULES
18. CSS3 FLEXBOX
    - CSS3 FLEXBOX
    - CSS3 FLEXBOX CONCEPTS
    - FLEX DIRECTION
    - THE JUSTIFY-CONTENT PROPERTY
    - THE ALIGN-ITEMS PROPERTY
    - THE FLEX-WRAP PROPERTY
    - THE ALIGN-CONTENT PROPERTY
    - FLEX ITEM PROPERTIES
19. CSS3 FILTERS
    - CSS3 FILTERS
    - SYNTAX
    - FILTER FUNCTIONS
    - BRIGHTNESS(%)
    - CONTRAST(%)
    - DROP SHADOW(H-SHADOW V-SHADOW BLUR COLOR)
    - SPECIFY HOW MANY COLUMNS AN ELEMENT SHOULD SPAN
    - SPECIFY THE COLUMN WIDTH
16. CSS3 USER INTERFACE
    - CSS3 USER INTERFACE
    - CSS3 RESIZING
    - CSS3 OUTLINE OFFSET
17. CSS3 BOX SIZING
    - CSS3 BOX SIZING
    - WITHOUT THE CSS3 BOX-SIZING PROPERTY
    - WITH THE CSS3 BOX-SIZING PROPERTY
    - GRAYSCALE(%)
    - HUE ROTATION(DEG)
    - INVERT(%)
    - OPACITY(%)
    - SATURATE(%)
    - SEPIA(%)
    - COMBINING FILTERS / MULTIPLE FILTERS
    - ALL FILTERS

    sa
20. CSS3 MEDIA QUERIES
    - CSS3 INTRODUCES MEDIA QUERIES
    - MEDIA QUERY SYNTAX
    - CSS3 MEDIA TYPES
    - MEDIA QUERIES SIMPLE EXAMPLES

Chapter 1 CSS3 INTRODUCTION

WHAT IS CSS3?

CSS3 is a cascading piece of paper that specifies concerning the data with a joined hypertext markup language document display. it's considerably additional options than previous CSS versions. additionally to further graphics functions, CSS3 permits, to pick out additional hypertext markup language tags and outline however they're displayed on an online browser.

The standard structure of CSS3 permits a gradual unharness of recent options, and lets browsers update piecemeal to support the most recent definitions.

CSS3 is completely backwards compatible, so you will not have to change existing designs. Browsers will always support CSS2. CSS3 is split up into "modules". The old specification has been split into smaller pieces, and new ones have been also added.

CSS3 MODULES

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

Chapter 2 CSS3 BACKGROUNDS

A CSS3 Backgrounds is an affords, which is used to resizing of the background properties. It is also used for multiple background implementation. Before developing of CSS3 it was unable to resize the background, but with the help of CSS3 We can implement these affords also.

There are mainly two properties of background, which can be used for background fixing, background size, multiple background image and etc.

Below are properties that are used for background properties:

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

MULTIPLE BACKGROUND

CSS3 Multi background property is used to add one or more images at a time without HTML code, We can add images as per our requirement.

A sample syntax of multi background images is as follows –

```
#multibackground {

        background-image: url(images/css3 img.jpg),
                          url(images/toplogo.png);
        background-position: right bottom, left top;
        background-repeat: no-repeat, repeat;
        padding: 15px;

}
```

Example :

Following is the example which demonstrate the multi background images:

```
<html>
  <head>
  <style>
    #multibackground {
        background-image: url(images/ css3 img.jpg),
                          url(images/toplogo.png);
        background-position: right bottom, left top;
        background-repeat: no-repeat, repeat;
        padding: 15px;
       }
```

```
</style>
    </head>
  <body>
  <div id="multibackground">
      <h2> futurevisioncomputers</h2>
      <p> Future Vision Computer Institute was founded in 2006 with the mission of
        providing best quality Computer education to all classes of people at a very
        reasonable fee structure.</p>
    </div>
  </body>
</html>
```

BACKGROUND-SIZE

Multi background property is accepted to add different sizes for different images.

A sample syntax is as shown below:

```
#multibackground1 {

   border: 1px solid black;

   background:   url(images/toplogo.png);

   background-repeat: no-repeat;

   padding:15px;

}

#multibackground2 {

border: 1px solid black;

background: url(images/toplogo.png);

background-size: 100px 80px;

background-repeat: no-repeat;

padding:15px;

}
```

Example :

Following is the example which demonstrate the multi background images:

```
<html>
<head>
<style>
 #multibackground1 {
                border: 1px solid black;
                background:   url(images/toplogo.png);
                background-repeat: no-repeat;
                padding:15px;
               }
        #multibackground2 {
                  border: 1px solid black;
                   background: url(images/toplogo.png);
                   background-size: 100px 80px;
                   background-repeat: no-repeat;
                   padding:15px;
                 }
</style>
     </head>
  <body>
<div id="multibackground1">
      <h2>futurevisioncomputers</h2>
      <p> Future Vision Computer Institute was founded in 2006 with the mission of providing
        best quality Computer education to all classes of people at a very reasonable fee
        structure.</p>
    </div>
  <div id="multibackground2">
      <h2> futurevisioncomputers </h2>
      <p> Future Vision Computer Institute was founded in 2006 with the mission of providing
        best quality Computer education to all classes of people at a very reasonable fee
        structure.</p>
    </div>
</body>
</html>
```

DEFINE SIZES OF MULTIPLE BACKGROUND

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

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

```
#multibackground {

        background: url(images/toplogo.png) left top no-repeat,
                    url(images/toplogo.png) right bottom  no-repeat,
                   url(images/toplogo.png)left top repeat;
                   background-size:50px,130px,auto;
                 }
```

Example :

Following is the example which demonstrate the multi background images:

```
<html>
<head>
<style>
  #multibackground {
          background: url(images/toplogo.png) left top no-repeat,
                      url(images/toplogo.png) right bottom  no-repeat,
                      url(images/toplogo.png)left top repeat;
        background-size:50px,130px,auto;
          }
</style>
</head>
 <body>
  <div id="multibackground">
     <h2> futurevisioncomputers</h2>
      <p> Future Vision Computer Institute was founded in 2006 with the mission of providing best quality Computer education to all classes of people at a very reasonable fee structure.</p>
    </div>
  </body>
</html>
```

FULL SIZE BACKGROUND

The requirements are as follows:

- ✓ Fill the entire page with the image (no white space)
- ✓ Scale image as needed
- ✓ Center image on page

- ✓ Do not cause scrollbars

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

Example :

Following is the example which demonstrate the multi background images:

```
<html>
<head>
<style>
 mybackground{

background:url(img_flower.jpg)no-repeat,center,center,fixed;
                background-size:cover;
          }
 body {   color: white;}
</style>
 </head>
 <body>
  <div class=" mybackground ">
    <h2> futurevisioncomputers</h2>
<p> Future Vision Computer Institute was founded in 2006 with the mission of providing
        best quality Computer education to all classes of people at a very reasonable fee
        structure.</p>
    </div>
  </body>
</html>
```

BACKGROUND-ORIGINE

CSS3 background-origin property specifies where the background image is positioned.

The property takes three different values:

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

```
#multibackground1 {

    border: 10px solid black;
     padding:35px;
     background:url(img_flwr.gif);
```

```
    background-repeat: no-repeat;
    background-origin: content-box;

}
```

Example :

Following is the example which demonstrate the multi background images:

```
<!DOCTYPE html>
<html>
<head>
<style>
#multibackground1 {
                        border:10px solid black;
                        padding:35px;
                        background:url(img_flwr.gif);
                        background-repeat: no-repeat;
                    }
#multibackground2 {
                        border: 10px solid black;
                        padding:35px;
                        background:url(img_flwr.gif);
                        background-repeat: no-repeat;
                        background-origin: border-box;

                    }
#multibackground3 {
                        border: 10px solid black;
                        padding:35px;
                         background:url(img_flwr.gif);
                         background-repeat: no-repeat;
                         background-origin: content-box;

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

 <p>No background-origin (padding-box is default):</p>
 <div id="multibackground1">
 <h2> futurevisioncomputers</h2>
```

```
<p> Future Vision Computer Institute was founded in 2006 with the mission of providing
        best quality Computer education to all classes of people at a very reasonable fee
        structure.</p>
 </div>
<p>background-origin: border-box:</p>
 <div id="multibackground2">
 <h2> futurevisioncomputers</h2>
<p> Future Vision Computer Institute was founded in 2006 with the mission of providing
        best quality Computer education to all classes of people at a very reasonable fee
        structure.</p>
 </div>
 <p>background-origin: content-box:</p>
 <div id="multibackground3">
 <h2> futurevisioncomputers</h2>
<p> Future Vision Computer Institute was founded in 2006 with the mission of providing
        best quality Computer education to all classes of people at a very reasonable fee
        structure.</p>
 </div>
</body>
</html>
```

BACKGROND-CLIP

CSS3 background-clip property specifies where the background image is positioned.

The property takes three different values:

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

```
#multibackground1 {

        border: 10px dotted black;
        padding:35px;
        background: yellow;
        background-clip: content-box;

 }
```

Example :

Following is the example which demonstrate the multi background images:

```
<!DOCTYPE html>
<html>
<head>
<style>
#multibackground1 {
```

```
                border: 10px dotted black;
        padding:35px;
        background: yellow;
    }
#multibackground2 {
                border: 10px dotted black;
                padding:35px;
                background: yellow;
                background-clip: padding-box;
            }
#multibackground1 {

                        border: 10px dotted black;
                        padding:35px;
                        background: yellow;
                        background-clip: content-box;

        }
</style>
</head>
<body>
 <p>No background-clip (border-box is default)::</p>
 <div id="multibackground1">
 <h2> futurevisioncomputers</h2>
<p> Future Vision Computer Institute was founded in 2006 with the mission of providing best
        quality Computer education to all classes of people at a very reasonable fee
        structure.</p>
 </div>
 <p>background-clip: padding-box:</p>
 <div id="multibackground2">
 <h2> futurevisioncomputers</h2>
<p> Future Vision Computer Institute was founded in 2006 with the mission of providing best
        quality Computer education to all classes of people at a very reasonable fee
        structure.</p>
 </div>
 <p>background-clip: content-box:</p>
 <div id="multibackground3">
 <h2> futurevisioncomputers</h2>
<p> Future Vision Computer Institute was founded in 2006 with the mission of providing best
        quality Computer education to all classes of people at a very reasonable fee
        structure.</p>
 </div>
 </body>
</html>
```

Chapter 3 CSS3 TEXT

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

- ✓ Text-overflow
- ✓ Word-wrap
- ✓ Word-break

TEXT-OVERFLOW

The CSS3 text-overflow property specifies how overflowed content that is not displayed should be signaled to the user.

The CSS code is as follows:

```
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;
}
```

Example :

Following is the example which demonstrate the multi background images:

```
<!DOCTYPE html>
<html>
<head>
<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>
```

```
</body>

</html>
```

The following example shows how you can display the overflowed content when hovering over the element:

```
div.test:hover {
  text-overflow: inherit;
   overflow: visible;
}
```

Example :

Following is the example which demonstrate the multi background images:

```
<!DOCTYPE html>

<html>

<head>

<style>

div.test {

  white-space: nowrap;

  width: 200px;

  overflow: hidden;

  border: 1px solid #000000;

}

div.test:hover {

  text-overflow: inherit;

  overflow: visible;

}

</style>

</head>

<body>
```

```
<p>Hover over the two divs below, to see the entire text.</p>

<div class="test" style="text-overflow:ellipsis;">This is some long text that will not fit in the box</div>

<br>

<div class="test" style="text-overflow:clip;">This is some long text that will not fit in the box</div>

</body>

</html>
```

WORD WRAPPING

Allow long words to be able to be broken and wrap onto the next line:

```
p {
  word-wrap: break-word;
}
```

Example :

Following is the example which demonstrate long words to be able to be broken and wrap:

```
<!DOCTYPE html>

<html>

<head>

<style>

p.test {

  width: 11em;

  border: 1px solid #000000;

  word-wrap: break-word;

}

</style>

</head>
```

```
<body>

<p class="test> Future Vision Computer Institute was founded in 2006 with the mission of providing best quality Computer education. The long word will break and wrap to the next line.</p>

</body>

</html>
```

WORD BREAKING

The CSS code is as follows:

```
p.test1 {
  word-break: keep-all;
}

p.test2 {
  word-break: break-all;
}
```

Example :

Following is the example which demonstrate long words to be able to be broken and wrap:

```
<!DOCTYPE html>

<html>

<head>

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

Chapter 4 CSS3 FONTS

CSS3 Web Fonts - The @font-face Rule

Web fonts allow Web designers to use fonts that are not installed on the user's computer.

When you have found/bought the font you wish to use, just include the font file on your web server, and it will be automatically downloaded to the user when needed.

Your "own" fonts are defined within the CSS3 @font-face rule.

FONT-FAMILY

The CSS code is as follows:

```
@font-face {
  font-family: myFirstFont;
  src: url(sansation_light.woff);
}

div {
  font-family: myFirstFont;
}
```

Example :

Following is the example which demonstrate how to set the font family of an element. Possible value could be any font family name.:

```
<!DOCTYPE html>

<html>

<head>

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

FONT-STRETCH

The following example demonstrates how to set the font stretch of an element. This property relies on the user's computer to have an expanded or condensed version of the font being used.

Possible values could be normal, wider, narrower, ultra-condensed, extra-condensed, condensed, semi-condensed, semi-expanded, expanded, extra-expanded, ultra-expanded.

```
p{
font-stretch:ultra-expanded;
}
```

Example :

Following is the example which demonstrate

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

```
p{
font-stretch:ultra-expanded;
}
</style>
 </head>
 <body>
    <p >
       If this doesn't appear to work, it is likely that your computer doesn't have a
       condensed or expanded version of the font being used.
    </p>
  </body>
</html>
```

FONT-WEIGHT

The CSS code is as follows:

```
@font-face {
  font-family: myFirstFont;
  src: url(sansation_bold.woff);
  font-weight: bold;
}
```

Example :

Following is the example which demonstrate

```
<!DOCTYPE html>
<html>
```

```
<head>
<style>
@font-face {
  font-family: myFirstFont;
  src: url(sansation_light.woff);
}

@font-face {
  font-family: myFirstFont;
  src: url(sansation_bold.woff);
  font-weight: bold;
}
div {
  font-family: myFirstFont;
}
</style>
</head>
<body>
<div>
With CSS3, websites can <b>finally</b> use fonts other than the pre-selected "web-safe" fonts.
</div>
</body>
</html>
```

FONT-STYLE

The font-style property is mostly used to specify italic text.

This property has three values:

- ✓ normal - The text is shown normally
- ✓ italic - The text is shown in italics
- ✓ oblique - The text is "leaning" (oblique is very similar to italic, but less supported)

```
p.normal {
  font-style: normal;
}

p.italic {
  font-style: italic;
}

p.oblique {
  font-style: oblique;
}
```

Example :

Following is the example which demonstrate

```
<!DOCTYPE html>
<html>
<head>
<style>
p.normal {
  font-style: normal;
}
p.italic {
  font-style: italic;
```

```
}
p.oblique {
   font-style: oblique;
}
</style>
</head>
<body>
<p class="normal">This is a paragraph in normal style.</p>
<p class="italic">This is a paragraph in italic style.</p>
<p class="oblique">This is a paragraph in oblique style.</p>
</body>
</html>
```

Chapter 5 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);

The following example defines different RGBA colors:

```
#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 */
```

```
<!DOCTYPE html>

<html>

<head>

<style>

#p1 {background-color:rgba(255,0,0,0.3);}

#p2 {background-color:rgba(0,255,0,0.3);}

#p3 {background-color:rgba(0,0,255,0.3);}

#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);}
```

```
</style>

</head>

<body>


<p>RGBA colors:</p>

<p id="p1">Red</p>

<p id="p2">Green</p>

<p id="p3">Blue</p>

<p id="p4">Grey</p>

<p id="p5">Yellow</p>

<p id="p6">Cerise</p>


</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%);

The following example defines different HSL colors:

```
#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 */
```

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
#p1 {background-color:hsl(120,100%,50%);}
#p2 {background-color:hsl(120,100%,75%);}
#p3 {background-color:hsl(120,100%,25%);}
#p4 {background-color:hsl(120,60%,70%);}
#p5 {background-color:hsl(290,100%,50%);}
#p6 {background-color:hsl(290,60%,70%);}
</style>
</head>
<body>

<p>HSL colors:</p>
<p id="p1">Green</p>
<p id="p2">Light green</p>
<p id="p3">Dark green</p>
<p id="p4">Pastel green</p>
```

```
<p id="p5">Violet</p>

<p id="p6">Pastel violet</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);

The following example defines different HSLA colors:

Example

```
#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 */
```

```
<!DOCTYPE html>

<html>

<head>

<style>

#p1 {background-color:hsla(120,100%,50%,0.3);}

#p2 {background-color:hsla(120,100%,75%,0.3);}

#p3 {background-color:hsla(120,100%,25%,0.3);}
```

```
#p4 {background-color:hsla(120,60%,70%,0.3);}

#p5 {background-color:hsla(290,100%,50%,0.3);}

#p6 {background-color:hsla(290,60%,70%,0.3);}

</style>

</head>

<body>

<p>HSLA colors:</p>

<p id="p1">Green</p>

<p id="p2">Light green</p>

<p id="p3">Dark green</p>

<p id="p4">Pastel green</p>

<p id="p5">Violet</p>

<p id="p6">Pastel violet</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;
rgb(255, 0, 0);opacity:0.6;
rgb(255, 0, 0);opacity:0.8;
```

Notice that the text above will also be opaque.

The following example shows different RGB values with opacity:

```
#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 */
```

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
#p1 {background-color:rgb(255,0,0);opacity:0.6;}
#p2 {background-color:rgb(0,255,0);opacity:0.6;}
#p3 {background-color:rgb(0,0,255);opacity:0.6;}
#p4 {background-color:rgb(192,192,192);opacity:0.6;}
#p5 {background-color:rgb(255,255,0);opacity:0.6;}
#p6 {background-color:rgb(255,0,255);opacity:0.6;}
</style>
</head>
<body>
<p>RGB colors with opacity:</p>
<p id="p1">Red</p>
<p id="p2">Green</p>
<p id="p3">Blue</p>
<p id="p4">Grey</p>
<p id="p5">Yellow</p>
<p id="p6">Cerise</p>
</body>
```

```
</html>
```

Chapter 6 CSS3 ROUNDED CORNERS

CSS3 BORDER-RADIUS

With the CSS3 border-radius property, you can give any element "rounded corners".

Here is the code:

Example

```
#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;
}
```

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
#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;
}
</style>
</head>
<body>

<p>The border-radius property allows you to add rounded corners to elements.</p>
<p>Rounded corners for an element with a specified background color:</p>
<p id="rcorners1">Rounded corners!</p>
<p>Rounded corners for an element with a border:</p>
<p id="rcorners2">Rounded corners!</p>
<p>Rounded corners for an element with a background image:</p>
<p id="rcorners3">Rounded corners!</p>

</body>
</html>
```

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

```
#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;
}
```

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
#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;
}
</style>
</head>
<body>

<p>Four values - border-radius: 15px 50px 30px 5px:</p>
<p id="rcorners4"></p>

<p>Three values - border-radius: 15px 50px 30px:</p>
<p id="rcorners5"></p>

<p>Two values - border-radius: 15px 50px:</p>
<p id="rcorners6"></p>

</body>
</html>
```

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;
}
```

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
#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;
}
</style>
</head>
<body>
<p>Elliptical border - border-radius: 50px/15px:</p>
```

```
<p id="rcorners7"></p>

<p>Elliptical border - border-radius: 15px/50px:</p>

<p id="rcorners8"></p>

<p>Ellipse border - border-radius: 50%:</p>

<p id="rcorners9"></p>

</body>

</html>
```

Chapter 7 CSS3 BUTTONS

BASIC BUTTON STYLING

```
.button {
  background-color: #4CAF50; /* Green */
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
.button {
  background-color: #4CAF50;
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  margin: 4px 2px;
  cursor: pointer;
```

```
}

</style>

</head>

<body>


<h2>CSS Buttons</h2>

<button>Default Button</button>

<a href="#" class="button">Link Button</a>

<button class="button">Button</button>

<input type="button" class="button" value="Input Button">


</body>

</html>
```

BUTTON COLORS

Use the background-color property to change the background color of a button:

```
.button1 {background-color: #4CAF50;} /* Green */
.button2 {background-color: #008CBA;} /* Blue */
.button3 {background-color: #f44336;} /* Red */
.button4 {background-color: #e7e7e7; color: black;} /* Gray */
.button5 {background-color: #555555;} /* Black */
```

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
.button {
  background-color: #4CAF50; /* Green */
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  margin: 4px 2px;
  cursor: pointer;
}
.button2 {background-color: #008CBA;} /* Blue */
.button3 {background-color: #f44336;} /* Red */
.button4 {background-color: #e7e7e7; color: black;} /* Gray */
.button5 {background-color: #555555;} /* Black */
</style>
</head>
<body>
```

```
<h2>Button Colors</h2>

<p>Change the background color of a button with the background-color property:</p>


<button class="button">Green</button>

<button class="button button2">Blue</button>

<button class="button button3">Red</button>

<button class="button button4">Gray</button>

<button class="button button5">Black</button>


</body>

</html>
```

BUTTON SIZES

Use the font-size property to change the size of a button:

Example

```
.button1 {font-size: 10px;}
.button2 {font-size: 12px;}
.button3 {font-size: 16px;}
.button4 {font-size: 20px;}
.button5 {font-size: 24px;}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

.button {

  background-color: #4CAF50; /* Green */
```

```
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  margin: 4px 2px;
  cursor: pointer;
}

.button1 {font-size: 10px;}
.button2 {font-size: 12px;}
.button3 {font-size: 16px;}
.button4 {font-size: 20px;}
.button5 {font-size: 24px;}
</style>
</head>
<body>

<h2>Button Sizes</h2>
<p>Change the size of a button with the font-size property:</p>

<button class="button button1">10px</button>
<button class="button button2">12px</button>
```

```
<button class="button button3">16px</button>

<button class="button button4">20px</button>

<button class="button button5">24px</button>


</body>

</html>
```

ROUNDED BUTTONS

Use the border-radius property to add rounded corners to a button:

Example

```
.button1 {border-radius: 2px;}
.button2 {border-radius: 4px;}
.button3 {border-radius: 8px;}
.button4 {border-radius: 12px;}
.button5 {border-radius: 50%;}
```

Example:

```
<!DOCTYPE html>

<html>

<head>

<style>

.button {
  background-color: #4CAF50; /* Green */
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
```

```
  font-size: 16px;
  margin: 4px 2px;
  cursor: pointer;
}

.button1 {border-radius: 2px;}
.button2 {border-radius: 4px;}
.button3 {border-radius: 8px;}
.button4 {border-radius: 12px;}
.button5 {border-radius: 50%;}
</style>
</head>
<body>

<h2>Rounded Buttons</h2>
<p>Add rounded corners to a button with the border-radius property:</p>

<button class="button button1">2px</button>
<button class="button button2">4px</button>
<button class="button button3">8px</button>
<button class="button button4">12px</button>
<button class="button button5">50%</button>

</body>
</html>
```

COLORED BUTTON BORDERS

Use the border property to add a colored border to a button:

Example

```
.button1 {
  background-color: white;
  color: black;
  border: 2px solid #4CAF50; /* Green */
}
```

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
.button {
  background-color: #4CAF50; /* Green */
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  margin: 4px 2px;
  cursor: pointer;
}

.button1 {
```

```
    background-color: white;
    color: black;
    border: 2px solid #4CAF50;
}

.button2 {
    background-color: white;
    color: black;
    border: 2px solid #008CBA;
}

.button3 {
    background-color: white;
    color: black;
    border: 2px solid #f44336;
}

.button4 {
    background-color: white;
    color: black;
    border: 2px solid #e7e7e7;
}

.button5 {
    background-color: white;
```

```
  color: black;
  border: 2px solid #555555;
}
</style>
</head>
<body>

<h2>Colored Button Borders</h2>
<p>Use the border property to add a border to the button:</p>

<button class="button button1">Green</button>
<button class="button button2">Blue</button>
<button class="button button3">Red</button>
<button class="button button4">Gray</button>
<button class="button button5">Black</button>

</body>
</html>
```

HOVERABLE BUTTONS

Use the :hover selector to change the style of a button when you move the mouse over it.

Tip: Use the transition-duration property to determine the speed of the "hover" effect:

```
.button {
  -webkit-transition-duration: 0.4s; /* Safari */
  transition-duration: 0.4s;
}
```

```
.button:hover {
  background-color: #4CAF50; /* Green */
  color: white;
}
```

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
.button {
  background-color: #4CAF50; /* Green */
  border: none;
  color: white;
  padding: 16px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  margin: 4px 2px;
  -webkit-transition-duration: 0.4s; /* Safari */
  transition-duration: 0.4s;
  cursor: pointer;
}

.button1 {
```

```
    background-color: white;
    color: black;
    border: 2px solid #4CAF50;
}

.button1:hover {
    background-color: #4CAF50;
    color: white;
}

.button2 {
    background-color: white;
    color: black;
    border: 2px solid #008CBA;
}

.button2:hover {
    background-color: #008CBA;
    color: white;
}
.button3 {
    background-color: white;
    color: black;
    border: 2px solid #f44336;
}
```

```
.button3:hover {
   background-color: #f44336;
   color: white;
}

.button4 {
   background-color: white;
   color: black;
   border: 2px solid #e7e7e7;
}

.button4:hover {
background-color: #e7e7e7;
}
.button5 {
   background-color: white;
   color: black;
   border: 2px solid #555555;
}

.button5:hover {
   background-color: #555555;
   color: white;
}
</style>
```

```
</head>

<body>


<h2>Hoverable Buttons</h2>

<p>Use the :hover selector to change the style of the button when you move the mouse over it.</p>

<p><strong>Tip:</strong> Use the transition-duration property to determine the speed of the "hover" effect:</p>


<button class="button button1">Green</button>

<button class="button button2">Blue</button>

<button class="button button3">Red</button>

<button class="button button4">Gray</button>

<button class="button button5">Black</button>


</body>

</html>
```

SHADOW BUTTONS

Use the box-shadow property to add shadows to a button:

Example

```
.button1 {
  box-shadow: 0 8px 16px 0 rgba(0,0,0,0.2), 0 6px 20px 0 rgba(0,0,0,0.19);
}

.button2:hover {
  box-shadow: 0 12px 16px 0 rgba(0,0,0,0.24), 0 17px 50px 0 rgba(0,0,0,0.19);
}
```

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
.button {
  background-color: #4CAF50; /* Green */
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  margin: 4px 2px;
  cursor: pointer;
  -webkit-transition-duration: 0.4s; /* Safari */
  transition-duration: 0.4s;
}

.button1 {
  box-shadow: 0 8px 16px 0 rgba(0,0,0,0.2), 0 6px 20px 0 rgba(0,0,0,0.19);
}

.button2:hover {
  box-shadow: 0 12px 16px 0 rgba(0,0,0,0.24),0 17px 50px 0 rgba(0,0,0,0.19);
```

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

<h2>Shadow Buttons</h2>
<p>Use the box-shadow property to add shadows to the button:</p>

<button class="button button1">Shadow Button</button>
<button class="button button2">Shadow on Hover</button>

</body>
</html>
```

DISABLED BUTTONS

Use the opacity property to add transparency to a button (creates a "disabled" look).

Tip: You can also add the cursor property with a value of "not-allowed", which will display a "no parking sign" when you mouse over the button:

Example

```
.disabled {
  opacity: 0.6;
  cursor: not-allowed;
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

.button {

  background-color: #4CAF50; /* Green */

  border: none;

  color: white;

  padding: 15px 32px;

  text-align: center;

  text-decoration: none;

  display: inline-block;

  font-size: 16px;

  margin: 4px 2px;

  cursor: pointer;

}

.disabled {

  opacity: 0.6;

  cursor: not-allowed;

}
```

```
</style>

</head>

<body>


<h2>Disabled Buttons</h2>

<p>Use the opacity property to add some transparency to the button (make it look disabled):</p>


<button class="button">Normal Button</button>

<button class="button disabled">Disabled Button</button>


</body>

</html>
```

BUTTON WIDTH

By default, the size of the button is determined by its text content (as wide as its content). Use the width property to change the width of a button:

Example

```
.button1 {width: 250px;}
.button2 {width: 50%;}
.button3 {width: 100%;}

<!DOCTYPE html>

<html>

<head>

<style>

.button {

  background-color: #4CAF50; /* Green */
```

```
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  margin: 4px 2px;
  cursor: pointer;
}

.button1 {width: 250px;}
.button2 {width: 50%;}
.button3 {
  padding-left: 0;
  padding-right: 0;
  width: 100%;
}
</style>
</head>
<body>
<h2>Button Width</h2>
<p>Use the width property to change the width of the button:</p>
<p><strong>Tip:</strong> Use pixels if you want to set a fixed width and use percent for responsive buttons (e.g. 50% of its parent element).</p>
```

```
<button class="button button1">250px</button><br>
<button class="button button2">50%</button><br>
<button class="button button3">100%</button>

</body>
</html>
```

Button Groups

Remove margins and add float:left to each button to create a button group:

```
.button {
  float: left;
}
```

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
.button {
  background-color: #4CAF50; /* Green */
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
```

```
  display: inline-block;
  font-size: 16px;
  cursor: pointer;
  float: left;
}

.button:hover {
  background-color: #3e8e41;
}
</style>
</head>
<body>

<h2>Button Groups</h2>
<p>Remove margins and float the buttons to create a button group:</p>

<button class="button">Button</button>
<button class="button">Button</button>
<button class="button">Button</button>
<button class="button">Button</button>

<p style="clear:both"><br>Remember to clear floats after, or else will this p element also float next to the buttons.</p>

</body>
```

```
</html>
```

BORDERED BUTTON GROUPS

Use the border property to create a bordered button group:

```
.button {
  float: left;
  border: 1px solid green
}
```

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
.button {
  background-color: #4CAF50; /* Green */
  border: 1px solid green;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  cursor: pointer;
  float: left;
}
```

```
.button:hover {
  background-color: #3e8e41;
}
</style>
</head>
<body>

<h2>Bordered Button Group</h2>
<p>Add borders to create a bordered button group:</p>

<button class="button">Button</button>
<button class="button">Button</button>
<button class="button">Button</button>
<button class="button">Button</button>

<p style="clear:both"><br>Remember to clear floats after, or else will this p element also float next to the buttons.</p>

</body>
</html>
```

ANIMATED BUTTONS

Example:

Add an arrow on hover:

```
<!DOCTYPE html>
<html>
```

```
<head>
<style>
.button {
  display: inline-block;
  border-radius: 4px;
  background-color: #f4511e;
  border: none;
  color: #FFFFFF;
  text-align: center;
  font-size: 28px;
  padding: 20px;
  width: 200px;
  transition: all 0.5s;
  cursor: pointer;
  margin: 5px;
}

.button span {
  cursor: pointer;
  display: inline-block;
  position: relative;
  transition: 0.5s;
}
.button span:after {
  content: '»';
```

```
  position: absolute;
  opacity: 0;
  top: 0;
  right: -20px;
  transition: 0.5s;
}
.button:hover span {
  padding-right: 25px;
}
.button:hover span:after {
  opacity: 1;
  right: 0;
}
</style>
</head>
<body>
<h2>Animated Button</h2>
<button class="button" style="vertical-align:middle"><span>Hover </span></button>
</body></html>
```

Example

Add a "ripple" effect on click:

```
<!DOCTYPE html>
<html>
<head>
<style>
```

```
.button {
    position: relative;
    background-color: #4CAF50;
    border: none;
    font-size: 28px;
    color: #FFFFFF;
    padding: 20px;
    width: 200px;
    text-align: center;
    -webkit-transition-duration: 0.4s; /* Safari */
    transition-duration: 0.4s;
    text-decoration: none;
    overflow: hidden;
    cursor: pointer;
}
.button:after {
    content: "";
    background: #90EE90;
    display: block;
    position: absolute;
    padding-top: 300%;
    padding-left: 350%;
    margin-left: -20px!important;
    margin-top: -120%;
    opacity: 0;
```

```
  transition: all 0.8s
}
.button:active:after {
  padding: 0;
  margin: 0;
  opacity: 1;
  transition: 0s
}
</style>
</head>
<body>
<h2>Animated Button - Ripple Effect</h2>
<button class="button">Click Me</button>
</body>
</html>
```

Example

Add a "pressed" effect on click:

```
<!DOCTYPE html>
<html>
<head>
<style>
.button {
 display: inline-block;
 padding: 15px 25px;
```

```
  font-size: 24px;
  cursor: pointer;
  text-align: center;
  text-decoration: none;
  outline: none;
  color: #fff;
  background-color: #4CAF50;
  border: none;
  border-radius: 15px;
  box-shadow: 0 9px #999;
}
.button:hover {background-color: #3e8e41}
.button:active {
  background-color: #3e8e41;
  box-shadow: 0 5px #666;
  transform: translateY(4px);
}
</style>
</head>
<body>
<h2>Animated Buttons - "Pressed Effect"</h2>
<button class="button">Click Me</button>
</body>
</html>
```

Chapter 8 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:
- ✓ The image to use as the border
- ✓ Where to slice the image
- ✓ Define whether the middle sections should be repeated or stretched

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

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

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

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

Here is the code:

Example

```
#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;
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

#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;
}
</style>
</head>
<body>

<p>The border-image property specifies an image to be used as the border around an element:</p>
<p id="borderimg">Here, the middle sections of the image are repeated to create the border.</p>

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

</body>
</html>
```

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

An image as a border!

Here is the code:

Example

```
#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;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
#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;
}
</style>
</head>
<body>

<p>The border-image property specifies an image to be used as the border around an element:</p>

<p id="borderimg">Here, the middle sections of the image are stretched to create the border.</p>

<p>Here is the original image:</p><img src="border.png">

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

```
</body>

</html>
```

CSS3 BORDER-IMAGE - DIFFERENT SLICE VALUES

Different slice values completely changes the look of the border:

Example 1:

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

Example 2:

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

Example 3:

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

Here is the code:

```
#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;
}
```

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
#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;
}
</style>
</head>
<body>

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

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

</body>
</html>
```

Chapter 9 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.

Syntax

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

Linear Gradient - Top to Bottom (this is default)

The following example shows a linear gradient that starts at the top. It starts red, transitioning to yellow:

Example:

```
#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 */
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {
   height: 200px;
   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 (must be last) */
}
</style>
</head>
<body>
<h3>Linear Gradient - Top to Bottom</h3>
<p>This linear gradient starts at the top. It starts red, transitioning to yellow:</p>
<div id="grad1"></div>
<p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support gradients.</p>
</body>
</html>
```

Linear Gradient - Left to Right

The following example shows a linear gradient that starts from the left. It starts red, transitioning to yellow:

Example

```
#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 */
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

#grad1 {

   height: 200px;

   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 (must be last)
*/
}
</style>

</head>

<body>
```

```
<h3>Linear Gradient - Left to Right</h3>

<p>This linear gradient starts at the left. It starts red, transitioning to yellow:</p>

<div id="grad1"></div>

<p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support gradients.</p>

</body>

</html>
```

LINEAR GRADIENT - DIAGONAL

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

The following example shows a linear gradient that starts at top left (and goes to bottom right). It starts red, transitioning to yellow:

Example

```
#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 */
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

#grad1 {

   height: 200px;

   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 (must
be last) */

}

</style>

</head>

<body>

<h3>Linear Gradient - Diagonal</h3>

<p>This linear gradient starts at top left. It starts red, transitioning to yellow:</p>

<div id="grad1"></div>

<p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support
gradients.</p>

</body>

</html>
```

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.).

Syntax

Background: linear-gradient(*angle*, *color-stop1*, *color-stop2*);

The angle is specified as an angle between a horizontal line and the gradient line.

The following example shows how to use angles on linear gradients:

Example

```
#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 */
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {
  height: 100px;
  background: red; /* For browsers that do not support gradients */
  background: -webkit-linear-gradient(0deg, red, yellow); /* For Safari 5.1 to 6.0 */
  background: -o-linear-gradient(0deg, red, yellow); /* For Opera 11.1 to 12.0 */
  background: -moz-linear-gradient(0deg, red, yellow); /* For Firefox 3.6 to 15 */
  background: linear-gradient(0deg, red, yellow); /* Standard syntax (must be last) */
}

#grad2 {
  height: 100px;
  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 (must be last) */
```

```
}
#grad3 {
  height: 100px;
  background: red; /* For browsers that do not support gradients */
  background: -webkit-linear-gradient(180deg, red, yellow); /* For Safari 5.1 to 6.0 */
  background: -o-linear-gradient(180deg, red, yellow); /* For Opera 11.1 to 12.0 */
  background: -moz-linear-gradient(180deg, red, yellow); /* For Firefox 3.6 to 15 */
  background: linear-gradient(180deg, red, yellow); /* Standard syntax (must be last) */
}
#grad4 {
  height: 100px;
  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 (must be last) */
}
</style>
</head>
<body>

<h3>Linear Gradients - Using Different Angles</h3>
<div id="grad1" style="color:white;text-align:center;">0deg</div><br>
<div id="grad2" style="color:white;text-align:center;">90deg</div><br>
<div id="grad3" style="color:white;text-align:center;">180deg</div><br>
```

```
<div id="grad4" style="color:white;text-align:center;">-90deg</div>

<p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support gradients.</p>

</body>

</html>
```

USING MULTIPLE COLOR STOPS

The following example shows a linear gradient (from top to bottom) with multiple color stops:

Example

```
#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 */
}

<!DOCTYPE html>

<html>

<head>

<style>

#grad1 {

   height: 200px;

   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 (must be last) */

}
```

```
#grad2 {
  height: 200px;
  background: -webkit-linear-gradient(red, orange, yellow, green, blue, indigo, violet); /* For Safari 5.1 to 6.0 */
  background: -o-linear-gradient(red, orange, yellow, green, blue, indigo, violet); /* For Opera 11.1 to 12.0 */
  background: -moz-linear-gradient(red, orange, yellow, green, blue, indigo, violet); /* For Firefox 3.6 to 15 */
  background: linear-gradient(red, orange, yellow, green, blue, indigo, violet); /* Standard syntax (must be last) */
}
#grad3 {
  height: 200px;
  background: -webkit-linear-gradient(red 10%, green 85%, blue 90%); /* For Safari 5.1 to 6.0 */
  background: -o-linear-gradient(red 10%, green 85%, blue 90%); /* For Opera 11.1 to 12.0 */
  background: -moz-linear-gradient(red 10%, green 85%, blue 90%); /* For Firefox 3.6 to 15 */
  background: linear-gradient(red 10%, green 85%, blue 90%); /* Standard syntax (must be last) */
}
</style>
</head>
<body>

<h3>3 Color Stops (evenly spaced)</h3>
<div id="grad1"></div>
```

```
<h3>7 Color Stops (evenly spaced)</h3>

<div id="grad2"></div>

<h3>3 Color Stops (not evenly spaced)</h3>

<div id="grad3"></div>

<p><strong>Note:</strong> Color stops are automatically spaced evenly when no percents are specified.</p>

<p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support gradients.</p>

</body>

</html>
```

The following example shows how to create a linear gradient (from left to right) with the color of the rainbow and some text:

Example

```
#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);
}

<!DOCTYPE html>

<html>

<head>

<style>
```

```
#grad1 {
  height: 55px;
  background: -webkit-linear-gradient(left, red, orange, yellow, green, blue, indigo, violet); /* For Safari 5.1 to 6.0 */
  background: -o-linear-gradient(left, red, orange, yellow, green, blue, indigo, violet); /* For Opera 11.1 to 12.0 */
  background: -moz-linear-gradient(left, red, orange, yellow, green, blue, indigo, violet); /* For Fx 3.6 to 15 */
  background: linear-gradient(to right, red, orange, yellow, green, blue, indigo, violet); /* Standard syntax (must be last) */
}
</style>
</head>
<body>
<div id="grad1" style="text-align:center;margin:auto;color:#888888;font-size:40px;font-weight:bold">
Gradient Background
</div>
<p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support gradients.</p>
</body>
</html>
```

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).

The following example shows a linear gradient that starts from the left. It starts fully transparent, transitioning to full color red:

Example

```
#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*/
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

#grad1 {

   height: 200px;

   background: -webkit-linear-gradient(left, rgba(255,0,0,0), rgba(255,0,0,1)); /* For
Safari 5.1 to 6.0 */

   background: -o-linear-gradient(right, rgba(255,0,0,0), rgba(255,0,0,1)); /* For Opera
11.1 to 12.0 */

   background: -moz-linear-gradient(right, rgba(255,0,0,0), rgba(255,0,0,1)); /* For
Firefox 3.6 to 15 */

   background: linear-gradient(to right, rgba(255,0,0,0), rgba(255,0,0,1)); /* Standard
syntax (must be last) */

}

</style>

</head>
```

```
<body>

<h3>Linear Gradient - Transparency</h3>

<p>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).</p>

<div id="grad1"></div>

<p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support gradients.</p>

</body>

</html>
```

REPEATING A LINEAR-GRADIENT

The repeating-linear-gradient() function is used to repeat linear gradients:

Example

A repeating 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%);
}

<!DOCTYPE html>

<html>

<head>

<style>

#grad1 {
```

```
  height: 200px;
  background: -webkit-repeating-linear-gradient(red, yellow 10%, green 20%); /* For Safari 5.1 to 6.0 */
  background: -o-repeating-linear-gradient(red, yellow 10%, green 20%); /* For Opera 11.1 to 12.0 */
  background: -moz-repeating-linear-gradient(red, yellow 10%, green 20%); /* For Firefox 3.6 to 15 */
  background: repeating-linear-gradient(red, yellow 10%, green 20%); /* Standard syntax (must be last) */
}

#grad2 {
  height: 200px;
  background: -webkit-repeating-linear-gradient(45deg,red,yellow 7%,green 10%); /* For Safari 5.1 to 6.0 */
  background: -o-repeating-linear-gradient(45deg,red,yellow 7%,green 10%); /* For Opera 11.1 to 12.0 */
  background: -moz-repeating-linear-gradient(45deg,red,yellow 7%,green 10%); /* For Firefox 3.6 to 15 */
  background: repeating-linear-gradient(45deg,red,yellow 7%,green 10%); /* Standard syntax (must be last) */
}

#grad3 {
  height: 200px;
  background: -webkit-repeating-linear-gradient(190deg,red,yellow 7%,green 10%); /* For Safari 5.1 to 6.0 */
```

```
background: -o-repeating-linear-gradient(190deg,red,yellow 7%,green 10%); /* For Opera 11.1 to 12.0 */

  background: -moz-repeating-linear-gradient(190deg,red,yellow 7%,green 10%); /* For Firefox 3.6 to 15 */

  background: repeating-linear-gradient(190deg,red,yellow 7%,green 10%); /* Standard syntax (must be last) */

}


#grad4 {

  height: 200px;

  background: -webkit-repeating-linear-gradient(90deg,red,yellow 7%,green 10%); /* For Safari 5.1 to 6.0 */

  background: -o-repeating-linear-gradient(); /* For Opera 11.1 to 12.0 */

  background: -moz-repeating-linear-gradient(90deg,red,yellow 7%,green 10%); /* For Firefox 3.6 to 15 */

  background: repeating-linear-gradient(90deg,red,yellow 7%,green 10%); /* Standard syntax (must be last) */

}

</style>

</head>

<body>

<h3>Repeating Linear Gradient</h3>

<div id="grad1"></div>

<p>A repeating gradient on 45deg axe starting red and finishing green:</p>

<div id="grad2"></div>

<p>A repeating gradient on 190deg axe starting red and finishing green:</p>

<div id="grad3"></div>
```

```
<p>A repeating gradient on 90deg axe starting red and finishing green:</p>

<div id="grad4"></div>

<p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support gradients.</p>

</body>

</html>
```

RADIAL GRADIENTS

A radial gradient is defined by its center.

To create a radial gradient you must also define at least two color stops.

Syntax

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

By default, shape is ellipse, size is farthest-corner, and position is center.

Radial Gradient - Evenly Spaced Color Stops (this is default)

The following example shows a radial gradient with evenly spaced color stops:

Example:

```
#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 */
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

#grad1 {

   height: 150px;
```

```
    width: 200px;
    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 (must be last) */
  }
  </style>
  </head>
  <body>
  <h3>Radial Gradient - Evenly Spaced Color Stops</h3>
  <div id="grad1"></div>
  <p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support gradients.</p>
  </body>
  </html>
```

RADIAL GRADIENT - DIFFERENTLY SPACED COLOR STOPS

The following example shows a radial gradient with differently spaced color stops:

Example:

```
#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
```

```
*/
}

<!DOCTYPE html>

<html>

<head>

<style>

#grad1 {

   height: 150px;

   width: 200px;

   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 to
12.0 */

   background: -moz-radial-gradient(red 5%, yellow 15%, green 60%); /* For Firefox 3.6
to 15 */

   background: radial-gradient(red 5%, yellow 15%, green 60%); /* Standard syntax
(must be last) */

}

</style>

</head>

<body>

<h3>Radial Gradient - Differently Spaced Color Stops</h3>

<div id="grad1"></div>

<p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support
gradients.</p>

</body>
```

```
</html>
```

SET SHAPE

The shape parameter defines the shape. It can take the value circle or ellipse. The default value is ellipse.

The following example shows a radial gradient with the shape of a circle:

Example

```
#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 */
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

#grad1 {

   height: 150px;

   width: 200px;

   background: -webkit-radial-gradient(red, yellow, green); /* For 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 Fx 3.6 to 15 */

   background: radial-gradient(red, yellow, green); /* Standard syntax (must be last) */

}
```

```
#grad2 {
  height: 150px;
  width: 200px;
  background: -webkit-radial-gradient(circle, red, yellow, green); /* For Safari 5.1 to 6.0 */
  background: -o-radial-gradient(circle, red, yellow, green); /* For Opera 11.6 to 12.0 */
  background: -moz-radial-gradient(circle, red, yellow, green); /* For Fx 3.6 to 15 */
  background: radial-gradient(circle, red, yellow, green); /* Standard syntax (must be last) */
}
</style>
</head>
<body>
<h3>Radial Gradient - Shapes</h3>
<p><strong>Ellipse (this is default):</strong></p>
<div id="grad1"></div>
<p><strong>Circle:</strong></p>
<div id="grad2"></div>
<p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support gradients.</p>
</body>
</html>
<!DOCTYPE html>
<html>
<head>
<style>
```

```
#grad1 {
  height: 150px;
  width: 200px;
  background: -webkit-radial-gradient(red, yellow, green); /* For 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 Fx 3.6 to 15 */
  background: radial-gradient(red, yellow, green); /* Standard syntax (must be last) */
}
#grad2 {
  height: 150px;
  width: 200px;
  background: -webkit-radial-gradient(circle, red, yellow, green); /* For Safari 5.1 to 6.0
*/
  background: -o-radial-gradient(circle, red, yellow, green); /* For Opera 11.6 to 12.0 */
  background: -moz-radial-gradient(circle, red, yellow, green); /* For Fx 3.6 to 15 */
  background: radial-gradient(circle, red, yellow, green); /* Standard syntax (must be
last) */
}
</style>
</head>
<body>
<h3>Radial Gradient - Shapes</h3>
<p><strong>Ellipse (this is default):</strong></p>
<div id="grad1"></div>
<p><strong>Circle:</strong></p>
<div id="grad2"></div>
```

```
<p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support gradients.</p>


</body>

</html>
```

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

Example:

A radial gradient with different size keywords:

```
#grad1 {
  background: red; /* For browsers that do not support gradients */
  /* Safari 5.1 to 6.0 */
  background: -webkit-radial-gradient(60% 55%, closest-side, red, yellow, black);
  /* For Opera 11.6 to 12.0 */
  background: -o-radial-gradient(60% 55%, closest-side, red, yellow, black);
  /* For Firefox 3.6 to 15 */
  background: -moz-radial-gradient(60% 55%, closest-side, red, yellow, black);
  /* Standard syntax */
  background: radial-gradient(closest-side at 60% 55%, red, yellow, black);
}

#grad2 {
  /* Safari 5.1 to 6.0 */
  background: -webkit-radial-gradient(60% 55%, farthest-side, red, yellow, black);
  /* Opera 11.6 to 12.0 */
  background: -o-radial-gradient(60% 55%, farthest-side, red, yellow, black);
  /* For Firefox 3.6 to 15 */
  background: -moz-radial-gradient(60% 55%, farthest-side, red, yellow, black);
  /* Standard syntax */
  background: radial-gradient(farthest-side at 60% 55%, red, yellow, black);
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
#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) */
}
</style>
</head>
<body>
<h3>Radial Gradients - Use of different size keywords</h3>
```

```
<p><strong>closest-side:</strong></p>

<div id="grad1"></div>

<p><strong>farthest-side:</strong></p>

<div id="grad2"></div>

<p><strong>closest-corner:</strong></p>

<div id="grad3"></div>

<p><strong>farthest-corner (this is default):</strong></p>

<div id="grad4"></div>

<p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support gradients.</p>

</body>

</html>
```

REPEATING A RADIAL-GRADIENT

The repeating-radial-gradient() function is used to repeat radial gradients:

Example:

A repeating radial gradient:

```
#grad {
  background: red; /* For browsers that do not support gradients */
  /* For Safari 5.1 to 6.0 */
  background: -webkit-repeating-radial-gradient(red, yellow 10%, green 15%);
  /* For Opera 11.6 to 12.0 */
  background: -o-repeating-radial-gradient(red, yellow 10%, green 15%);
  /* For Firefox 3.6 to 15 */
  background: -moz-repeating-radial-gradient(red, yellow 10%, green 15%);
  /* Standard syntax */
  background: repeating-radial-gradient(red, yellow 10%, green 15%);
}
```

```
<!DOCTYPE html>

<html>
```

```
<head>
<style>
#grad1 {
  height: 150px;
  width: 200px;
  background: -webkit-repeating-radial-gradient(red, yellow 10%, green 15%); /* For Safari 5.1 to 6.0 */
  background: -o-repeating-radial-gradient(red, yellow 10%, green 15%); /* For Opera 11.6 to 12.0 */
  background: -moz-repeating-radial-gradient(red, yellow 10%, green 15%); /* For Firefox 3.6 to 15 */
  background: repeating-radial-gradient(red, yellow 10%, green 15%); /* Standard syntax (must be last) */
}
</style>
</head>
<body>
<h3>Repeating Radial Gradient</h3>
<div id="grad1"></div>
<p><strong>Note:</strong> Internet Explorer 9 and earlier versions do not support gradients.</p>
</body>
</html>
```

Chapter 10 CSS3 SHADOWS

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

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):

TEXT SHADOW EFFECT!

Example

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

```
<!DOCTYPE html>

<html>

<head>

<style>

h1 {

  text-shadow: 2px 2px;

}

</style>

</head>

<body>

<h1>Text-shadow effect!</h1>
```

```
<p><b>Note:</b> Internet Explorer 9 and earlier versions, do not support the text-shadow property.</p>

</body>

</html>
```

TEXT SHADOW EFFECT!

Example:

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

<!DOCTYPE html>

<html>

<head>

<style>

h1 {

  text-shadow: 2px 2px red;

}

</style>

</head>

<body>

<h1>Text-shadow effect!</h1>

<p><b>Note:</b> Internet Explorer 9 and earlier versions, do not support the text-shadow property.</p>

</body></html>
```

TEXT SHADOW EFFECT!

Example:

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

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  text-shadow: 2px 2px 5px red;
}
</style>
</head>
<body>
<h1>Text-shadow effect!</h1>
<p><b>Note:</b> Internet Explorer 9 and earlier versions, do not support the text-shadow property.</p>
</body>
</html>
```

TEXT SHADOW EFFECT!

Example

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

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  color: white;
  text-shadow: 2px 2px 4px #000000;
}
</style>
</head>
<body>
<h1>Text-shadow effect!</h1>
<p><b>Note:</b> Internet Explorer 9 and earlier versions, do not support the text-shadow property.</p>
</body>
</html>
```

TEXT SHADOW EFFECT!

Example:

```
h1 {
  text-shadow: 0 0 3px #FF0000;
}
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
```

```
  text-shadow: 0 0 3px #FF0000;
}
</style>
</head>
<body>
<h1>Text-shadow effect!</h1>
<p><b>Note:</b> Internet Explorer 9 and earlier versions, do not support the text-shadow property.</p>
</body>
</html>
```

MULTIPLE SHADOWS

To add more than one shadow to the text, you can add a comma-separated list of shadows.

The following example shows a red and blue neon glow shadow:

TEXT SHADOW EFFECT!

Example:

```
h1 {
  text-shadow: 0 0 3px #FF0000, 0 0 5px #0000FF;
}
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  text-shadow: 0 0 3px #FF0000, 0 0 5px #0000FF;
}
</style>
```

```
</head>

<body>

<h1>Text-shadow effect!</h1>

<p><b>Note:</b> Internet Explorer 9 and earlier versions, do not support the text-shadow property.</p>

</body>

</html>
```

TEXT SHADOW EFFECT!

Example:

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

<!DOCTYPE html>

<html>

<head>

<style>

    h1 {

      color: white;

      text-shadow: 1px 1px 2px black, 0 0 25px blue, 0 0 5px darkblue;

    }

</style>

</head>

<body>

<h1>Text-shadow effect!</h1>

<p><b>Note:</b> Internet Explorer 9 and earlier versions, do not support the text-shadow property.</p>
```

```
</body>

</html>
```

CSS3 BOX-SHADOW PROPERTY

The CSS3 box-shadow property applies shadow to elements.

In its simplest use, you only specify the horizontal shadow and the vertical shadow:

This is a yellow element with a black box-shadow

Example:

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

<!DOCTYPE html>

<html>

<head>

<style>

div {

  width: 300px;

  height: 100px;

  padding: 15px;

  background-color: yellow;

  box-shadow: 10px 10px;

}

</style>

</head>

<body>

<div>This is a div element with a box-shadow</div>

</body>

</html>
```

Next, add a color to the shadow:

This is a yellow element with a grey box-shadow

Example:

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

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 300px;
  height: 100px;
  padding: 15px;
  background-color: yellow;
  box-shadow: 10px 10px grey;
}
</style>
</head>
<body>
<div>This is a div element with a box-shadow</div>
</body>
</html>
```

Next, add a blur effect to the shadow:

This is a yellow element with a blurred, grey box-shadow

Example:

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

```
<!DOCTYPE html>

<html>

<head>

<style>

div {

  width: 300px;

  height: 100px;

  padding: 15px;

  background-color: yellow;

  box-shadow: 10px 10px 5px grey;

}

</style>

</head>

<body>


<div>This is a div element with a box-shadow</div>

</body>

</html>
```

Chapter 11 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.

Mouse over the elements below to see the difference between a 2D and a 3D transformation:

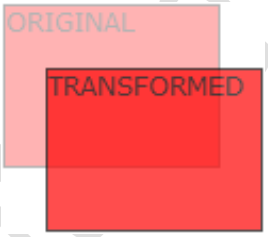
- ✓ 2D rotate
- ✓ 3D rotate

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).

The following example moves the element 50 pixels to the right, and 100 pixels down from its current position:

Example

```
div {
  -ms-transform: translate(50px,100px); /* IE 9 */
  -webkit-transform: translate(50px,100px); /* Safari */
  transform: translate(50px,100px);
}
```

```
<!DOCTYPE html>
<html>
<head>
<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>
</body>
```

```
</html>
```

THE ROTATE() METHOD

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

The following example rotates the element clockwise with 20 degrees:

Example

```
div {
  -ms-transform: rotate(20deg); /* IE 9 */
  -webkit-transform: rotate(20deg); /* Safari */
  transform: rotate(20deg);
}

<!DOCTYPE html>

<html>

<head>

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

Using negative values will rotate the element counter-clockwise.

The following example rotates the element counter-clockwise with 20 degrees:

Example:

```
div {
  -ms-transform: rotate(-20deg); /* IE 9 */
  -webkit-transform: rotate(-20deg); /* Safari */
  transform: rotate(-20deg);
}
<!DOCTYPE html>
<html>
<head>
<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 counter-clockwise with 20 degrees.
</div>
</body>
</html>
```

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 following example increases the element to be two times of its original width, and three times of its original height:

Example

```
div {
  -ms-transform: scale(2,3); /* IE 9 */
  -webkit-transform: scale(2,3); /* Safari */
  transform: scale(2,3);
}
```

```
<!DOCTYPE html>

<html>

<head>

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

The following example decreases the element to be half of its original width and height:

Example:

```
div {
  -ms-transform: scale(0.5,0.5); /* IE 9 */
  -webkit-transform: scale(0.5,0.5); /* Safari */
  transform: scale(0.5,0.5);
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
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 */
}
</style>
</head>
<body>
<p>The scale() method increases or decreases the size of an element.</p>
<div>
```

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

</div>

</body>

</html>
```

THE SKEWX() METHOD

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

The following example skews the element 20 degrees along the X-axis:

Example:

```
div {
  -ms-transform: skewX(20deg); /* IE 9 */
  -webkit-transform: skewX(20deg); /* Safari */
  transform: skewX(20deg);
}

<!DOCTYPE html>

<html>

<head>

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

THE SKEWY() METHOD

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

The following example skews the element 20 degrees along the Y-axis:

Example:

```
div {
  -ms-transform: skewY(20deg); /* IE 9 */
  -webkit-transform: skewY(20deg); /* Safari */
  transform: skewY(20deg);
}
<!DOCTYPE html>
<html>
<head>
<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>
</body>
</html>
```

THE SKEW() METHOD

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

The following example skews the element 20 degrees along the X-axis, and 10 degrees along the Y-axis:

Example:

```
div {
  -ms-transform: skew(20deg, 10deg); /* IE 9 */
  -webkit-transform: skew(20deg, 10deg); /* Safari */
  transform: skew(20deg, 10deg);
}
```

```
<!DOCTYPE html>

<html>

<head>

<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>

</body>

</html>
```

If the second parameter is not specified, it has a zero value. So, the following example skews the element 20 degrees along the X-axis:

Example:

```
div {
  -ms-transform: skew(20deg); /* IE 9 */
  -webkit-transform: skew(20deg); /* Safari */
  transform: skew(20deg);
}

<!DOCTYPE html>

<html>

<head>

<style>

div {

  width: 300px;

  height: 100px;

  background-color: yellow;

  border: 1px solid black;

}
```

```
div#myDiv {
  -ms-transform: skew(20deg); /* IE 9 */
  -webkit-transform: skew(20deg); /* Safari */
  transform: skew(20deg); /* 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.
</div>
</body>
</html>
```

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:

Example:

```
div {
  -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);
}
```

```
<!DOCTYPE html>

<html>

<head>

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

Chapter 12 CSS3 3D TRANSFORMS

CSS3 allows you to format your elements using 3D transformations.

Mouse over the elements below to see the difference between a 2D and a 3D transformation:

- ✓ 2D rotate
- ✓ 3D rotate

CSS3 3D TRANSFORMS

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

- ✓ rotateX()
- ✓ rotateY()
- ✓ rotateZ()

THE ROTATEX() METHOD

The rotateX() method rotates an element around its X-axis at a given degree:

Example

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

<!DOCTYPE html>

<html>

<head>

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

THE ROTATEY() METHOD

The rotateY() method rotates an element around its Y-axis at a given degree:

Example

```
div {
  -webkit-transform: rotateY(130deg); /* Safari */
  transform: rotateY(130deg);
}
```

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

</style>
</head>
<body>
<div>
This a normal div element.
</div>
```

```
<div id="myDiv">

The rotateY() method rotates an element around its Y-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 rotateY() method.</p>

</body>

</html>
```

THE ROTATEZ() METHOD

The rotateZ() method rotates an element around its Z-axis at a given degree:

Example

```
div {
  -webkit-transform: rotateZ(90deg); /* Safari */
  transform: rotateZ(90deg);
}

<!DOCTYPE html>

<html>

<head>

<style>

div {

  width: 300px;

  height: 100px;

  background-color: yellow;

  border: 1px solid black;

}

div#myDiv {
```

```
  -webkit-transform: rotateZ(90deg); /* Safari */
  transform: rotateZ(90deg); /* Standard syntax */
}
</style>
</head>
<body>
<div>
This a normal div element.
</div>
<div id="myDiv">
The rotateZ() method rotates an element around its Z-axis at a given degree. This div element is rotated 90 degrees.
</div>
<p><b>Note:</b> Internet Explorer 9 (and earlier versions) does not support the rotateZ() method.</p>
</body></html>
```

Chapter 13 CSS3 TRANSITIONS

CSS3 TRANSITIONS

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

Example:

Mouse over the element below to see a CSS3 transition effect:

CSS3

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.

The following example shows a 100px * 100px red element. The element has also specified a transition effect for the width property, with a duration of 2 seconds:

Example

```
div {
  width: 100px;
  height: 100px;
  background: red;
  -webkit-transition: width 2s; /* Safari */
  transition: width 2s;
}
```

The transition effect will start when the specified CSS property (width) changes value.

Now, let us specify a new value for the width property when a user mouses over the element:

Example

```
div:hover{
  width:300px;
}
```

```
<!DOCTYPE html>
```

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

CHANGE SEVERAL PROPERTY VALUES

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:

Example

```
div {
  -webkit-transition: width 2s, height 4s; /* Safari */
  transition: width 2s, height 4s;
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

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;

}

</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>

</body>

</html>
```

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

The following example shows the some of the different speed curves that can be used:

Example

```
#div1 {transition-timing-function: linear;}
#div2 {transition-timing-function: ease;}
#div3 {transition-timing-function: ease-in;}
#div4 {transition-timing-function: ease-out;}
#div5 {transition-timing-function: ease-in-out;}

<!DOCTYPE html>

<html>

<head>

<style>

div {

  width: 100px;
```

```
  height: 100px;
  background: red;
  -webkit-transition: width 2s; /* Safari */
  transition: width 2s;
}
/* For Safari 3.1 to 6.0 */
#div1 {-webkit-transition-timing-function: linear;}
#div2 {-webkit-transition-timing-function: ease;}
#div3 {-webkit-transition-timing-function: ease-in;}
#div4 {-webkit-transition-timing-function: ease-out;}
#div5 {-webkit-transition-timing-function: ease-in-out;}
/* Standard syntax */
#div1 {transition-timing-function: linear;}
#div2 {transition-timing-function: ease;}
#div3 {transition-timing-function: ease-in;}
#div4 {transition-timing-function: ease-out;}
#div5 {transition-timing-function: ease-in-out;}
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 id="div1">linear</div><br>

<div id="div2">ease</div><br>

<div id="div3">ease-in</div><br>

<div id="div4">ease-out</div><br>

<div id="div5">ease-in-out</div><br>

<p>Hover over the div elements above, to see the different speed curves.</p>

</body>

</html>
```

DELAY THE TRANSITION EFFECT

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

The following example has a 1 second delay before starting:

Example

```
div {
  -webkit-transition-delay: 1s; /* Safari */
  transition-delay: 1s;
}

<!DOCTYPE html>

<html>

<head>

<style>

div {

  width: 100px;

  height: 100px;

  background: red;

  -webkit-transition: width 3s; /* Safari */

  -webkit-transition-delay: 1s; /* Safari */

  transition: width 3s;
```

```
  transition-delay: 1s;
}
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>
<p><b>Note:</b> The transition effect has a 1 second delay before starting.</p>
</body>
</html>
```

TRANSITION + TRANSFORMATION

The following example also adds a transformation to the transition effect:

Example

```
div {
  -webkit-transition: width 2s, height 2s, -webkit-transform 2s; /* Safari */
  transition: width 2s, height 2s, transform 2s;
}
<!DOCTYPE html>
<html>
<head>
<style>
div {
```

```
  width: 100px;
  height: 100px;
  background: red;
  -webkit-transition: width 2s, height 2s, -webkit-transform 2s; /* Safari */
  transition: width 2s, height 2s, transform 2s;
}

div:hover {
  width: 300px;
  height: 300px;
  -webkit-transform: rotate(180deg); /* Safari */
  transform: rotate(180deg);
}
</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>
</body>
</html>
```

MORE TRANSITION EXAMPLES

The CSS3 transition properties can be specified one by one, like this:

Example

```
div {
  transition-property: width;
  transition-duration: 2s;
  transition-timing-function: linear;
  transition-delay: 1s;
}

<!DOCTYPE html>

<html>

<head>

<style>

div {

  width: 100px;

  height: 100px;

  background: red;

  /* For Safari 3.1 to 6.0 */

  -webkit-transition-property: width;

  -webkit-transition-duration: 2s;

  -webkit-transition-timing-function: linear;

  -webkit-transition-delay: 1s;

  /* Standard syntax */

  transition-property: width;

  transition-duration: 2s;

  transition-timing-function: linear;

  transition-delay: 1s;

}

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

or by using the shorthand property transition:

Example:

```
div {
  transition: width 2s linear 1s;
}
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background: red;
  -webkit-transition: width 2s linear 1s; /* For Safari 3.1 to 6.0 */
```

```
  transition: width 2s linear 1s;
}

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

Chapter 14 CSS3 ANIMATIONS

CSS3 ANIMATIONS

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

CSS3

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.

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

Example

```
/* The animation code */
@keyframes example {
  from {background-color: red;}
  to {background-color: yellow;}
}
```

```
/* The element to apply the animation to */
div {
   width: 100px;
   height: 100px;
   background-color: red;
  animation-name: example;
  animation-duration: 4s;
 }
```

```
<!DOCTYPE html>

<html>

<head>

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

Note: If the animation-duration property is not specified, the animation will have no effect, because the default value is 0.

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

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

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

Example

```
/* The animation code */
@keyframes example {
  0%   {background-color: red;}
  25%  {background-color: yellow;}
  50%  {background-color: blue;}
  100% {background-color: green;}
}

/* The element to apply the animation to */
div {
  width: 100px;
```

```
  height: 100px;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}
```

```
<!DOCTYPE html>
<html>
<head>
<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>
<div></div>
</body>
</html>
```

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

Example

```
/* The animation code */
@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;}
}

/* The element to apply the animation to */
div {
  width: 100px;
  height: 100px;
  position: relative;
  background-color: red;
  animation-name: example;
```

```
    animation-duration: 4s;
}

<!DOCTYPE html>

<html>

<head>

<style>

div {

    width: 100px;

    height: 100px;

    background-color: red;

    position: relative;

    -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; 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;}

}

/* 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;}
}
</style>
</head>
<body>
<p><b>Note:</b> This example does not work in Internet Explorer 9 and earlier versions.</p>
<div></div>
</body>
</html>
```

DELAY AN ANIMATION

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

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

Example

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

```
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
  position: relative;
  -webkit-animation-name: example; /* Chrome, Safari, Opera */
  -webkit-animation-duration: 4s; /* Chrome, Safari, Opera */
  -webkit-animation-delay: 2s; /* Chrome, Safari, Opera */
  animation-name: example;
  animation-duration: 4s;
  animation-delay: 2s;
}

/* Chrome, Safari, Opera */
@-webkit-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;}
}
/* 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;}
}
</style>
</head>
<body>
<p><b>Note:</b> This example does not work in Internet Explorer 9 and earlier versions.</p>
<div></div>
</body>
</html>
```

SET HOW MANY TIMES AN ANIMATION SHOULD RUN

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

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

Example

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

```
<!DOCTYPE html>

<html>

<head>

<style>

div {

  width: 100px;

  height: 100px;

  background-color: red;

  position: relative;

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

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

  -webkit-animation-iteration-count: 3; /* Chrome, Safari, Opera */

  animation-name: example;

  animation-duration: 4s;
```

```
    animation-iteration-count: 3;
}
/* Chrome, Safari, Opera */
@-webkit-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;}
}
/* 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;}
}
</style>
</head>
<body>
<p><b>Note:</b> This example does not work in Internet Explorer 9 and earlier versions.</p>
<div></div>
</body>
```

```
</html>
```

The following example uses the value "infinite" to make the animation continue for ever:

Example:

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

<!DOCTYPE html>

<html>

<head>

<style>

div {

  width: 100px;

  height: 100px;

  background-color: red;

  position: relative;

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

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

  -webkit-animation-iteration-count: infinite; /* Chrome, Safari, Opera */

  animation-name: example;

  animation-duration: 4s;

  animation-iteration-count: infinite;

}
```

```
/* Chrome, Safari, Opera */
@-webkit-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;}
}
/* 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;}
}
</style>
</head>
<body>
<p><b>Note:</b> This example does not work in Internet Explorer 9 and earlier versions.</p>

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

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.

The following example will run the animation in reverse direction:

Example

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

```
<!DOCTYPE html>

<html>

<head>

<style>

div {

  width: 100px;

  height: 100px;

  background-color: red;

  position: relative;

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

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

  -webkit-animation-iteration-count: 3; /* Chrome, Safari, Opera */

  -webkit-animation-direction: reverse; /* Chrome, Safari, Opera */
```

```
    animation-name: example;
    animation-duration: 4s;
    animation-iteration-count: 3;
    animation-direction: reverse;
}
/* Chrome, Safari, Opera */
@-webkit-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;}
}
/* 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;}
}
</style>
</head>
<body>
```

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

<div></div>

</body>

</html>
```

The following example uses the value "alternate" to make the animation first run forward, then backward, then forward:

Example

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

<!DOCTYPE html>

<html>

<head>

<style>

div {

  width: 100px;

  height: 100px;

  background-color: red;

  position: relative;

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

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

```
    -webkit-animation-iteration-count: 3; /* Chrome, Safari, Opera */
    -webkit-animation-direction: alternate; /* Chrome, Safari, Opera */
    animation-name: example;
    animation-duration: 4s;
    animation-iteration-count: 3;
    animation-direction: alternate;
}
/* Chrome, Safari, Opera */
@-webkit-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;}
}
/* 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;}
}
</style>
</head>
```

```
<body>

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

<div></div>

</body>

</html>
```

SPECIFY THE SPEED CURVE OF THE ANIMATION

The animation-timing-function property specifies the speed curve of the animation.

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

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

The following example shows the some of the different speed curves that can be used:

Example:

```
#div1 {animation-timing-function: linear;}
#div2 {animation-timing-function: ease;}
#div3 {animation-timing-function: ease-in;}
#div4 {animation-timing-function: ease-out;}
#div5 {animation-timing-function: ease-in-out;}

<!DOCTYPE html>

<html>

<head>

<style>

div {

  width: 100px;

  height: 50px;

  background-color: red;
```

```
  font-weight: bold;
  position: relative;
  -webkit-animation: mymove 5s infinite; /* Chrome, Safari, Opera */
  animation: mymove 5s infinite;
}

/* Chrome, Safari, Opera */
#div1 {-webkit-animation-timing-function: linear;}
#div2 {-webkit-animation-timing-function: ease;}
#div3 {-webkit-animation-timing-function: ease-in;}
#div4 {-webkit-animation-timing-function: ease-out;}
#div5 {-webkit-animation-timing-function: ease-in-out;}

/* Standard syntax */
#div1 {animation-timing-function: linear;}
#div2 {animation-timing-function: ease;}
#div3 {animation-timing-function: ease-in;}
#div4 {animation-timing-function: ease-out;}
#div5 {animation-timing-function: ease-in-out;}
/* Chrome, Safari, Opera */
@-webkit-keyframes mymove {
  from {left: 0px;}
  to {left: 300px;}
}
/* Standard syntax */
```

```
@keyframes mymove {
  from {left: 0px;}
  to {left: 300px;}
}
</style>
</head>
<body>

<p><strong>Note:</strong> The animation-timing-funtion property is not supported in Internet Explorer 9 and earlier versions.</p>
<div id="div1">linear</div>
<div id="div2">ease</div>
<div id="div3">ease-in</div>
<div id="div4">ease-out</div>
<div id="div5">ease-in-out</div>
</body>
</html>
```

ANIMATION SHORTHAND PROPERTY

The example below uses six of the animation properties:

Example:

```
div {
  animation-name: example;
  animation-duration: 5s;
  animation-timing-function: linear;
  animation-delay: 2s;
  animation-iteration-count: infinite;
  animation-direction: alternate;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
  position: relative;
  /* Chrome, Safari, Opera */
  -webkit-animation-name: example;
  -webkit-animation-duration: 5s;
  -webkit-animation-timing-function: linear;
  -webkit-animation-delay: 2s;
  -webkit-animation-iteration-count: infinite;
  -webkit-animation-direction: alternate;
  /* Standard syntax */
  animation-name: example;
  animation-duration: 5s;
  animation-timing-function: linear;
  animation-delay: 2s;
  animation-iteration-count: infinite;
  animation-direction: alternate;
}

/* Chrome, Safari, Opera */
```

```
@-webkit-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;}
}

/* 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;}
}
</style>
</head>
<body>
<p><b>Note:</b> This example does not work in Internet Explorer 9 and earlier versions.</p>
<div></div>
</body>
</html>
```

The same animation effect as above can be achieved by using the shorthand animation property:

Example:

```
div {
  animation: example 5s linear 2s infinite alternate;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
  position: relative;
  -webkit-animation: myfirst 5s linear 2s infinite alternate; /* Chrome, Safari, Opera */
  animation: myfirst 5s linear 2s infinite alternate;
}

/* Chrome, Safari, Opera */
@-webkit-keyframes myfirst {
  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;}
```

```
}

/* Standard syntax */
@keyframes myfirst {
   0%   {background-color:red; left:0px; top:0px;}
   25%  {background-color:yellow; left:200px; top:0px;}
   50%  {background-color:blue; left:200px; top:200px;}
   75%  {background-color:green; left:0px; top:200px;}
   100% {background-color:red; left:0px; top:0px;}
}
</style>
</head>
<body>
<p><b>Note:</b> This example does not work in Internet Explorer 9 and earlier versions.</p>
<div></div>
</body>
</html>
```

Chapter 15 CSS3 MULTIPLE COLUMNS

CSS3 MULTI-COLUMN LAYOUT

The CSS3 multi-column layout allows easy definition of multiple columns of text - just like in newspapers:

CSS3 MULTI-COLUMN PROPERTIES

In this chapter you will learn about the following 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.

The following example will divide the text in the element into 3 columns:

Example

```
div {
  -webkit-column-count: 3; /* Chrome, Safari, Opera */
  -moz-column-count: 3; /* Firefox */
  column-count: 3;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
.newspaper {
   -webkit-column-count: 3; /* Chrome, Safari, Opera */
   -moz-column-count: 3; /* Firefox */
   column-count: 3;
}
</style>
</head>
<body>
<p><b>Note:</b> Internet Explorer 9, and earlier versions, does not support the column-count property.</p>
<div class="newspaper">
</div>
</body>
</html>
```

CSS3 SPECIFY THE GAP BETWEEN COLUMNS

The column-gap property specifies the gap between the columns.

The following example specifies a 40 pixels gap between the columns:

Example:

```
div {
  -webkit-column-gap: 40px; /* Chrome, Safari, Opera */
  -moz-column-gap: 40px; /* Firefox */
  column-gap: 40px;
 }
```

```
<!DOCTYPE html>
<html>
<head>
<style>
.newspaper {
  -webkit-column-count: 3; /* Chrome, Safari, Opera */
  -moz-column-count: 3; /* Firefox */
  column-count: 3;
  -webkit-column-gap: 40px; /* Chrome, Safari, Opera */
  -moz-column-gap: 40px; /* Firefox */
  column-gap: 40px;
}
</style>
</head>
<body>

<p><b>Note:</b> Internet Explorer 9, and earlier versions, does not support the column-gap property.</p>
```

```
<div class="newspaper">

Future Vision Computer Institute was founded in 2006 with the mission of providing best quality Computer education to our students. Thousands of students have already got trained professionally and made their career successfully in the past.

</div>

</body>

</html>
```

CSS3 COLUMN RULES

The column-rule-style property specifies the style of the rule between columns:

Example:

```
div {
  -webkit-column-rule-style: solid; /* Chrome, Safari, Opera */
  -moz-column-rule-style: solid; /* Firefox */
  column-rule-style: solid;
}

<!DOCTYPE html>

<html>

<head>

<style>

.newspaper {

  -webkit-column-count: 3; /* Chrome, Safari, Opera */

  -moz-column-count: 3; /* Firefox */

  column-count: 3;

  -webkit-column-gap: 40px; /* Chrome, Safari, Opera */

  -moz-column-gap: 40px; /* Firefox */

  column-gap: 40px;

  -webkit-column-rule-style: solid; /* Chrome, Safari, Opera */
```

```
  -moz-column-rule-style: solid; /* Firefox */
  column-rule-style: solid;
}
</style>
</head>
<body>
<p><b>Note:</b> Internet Explorer 9, and earlier versions, does not support the column-rule-style property.</p>
<div class="newspaper">
Future Vision Computer Institute was founded in 2006 with the mission of providing best quality Computer education to our students. Thousands of students have already got trained professionally and made their career successfully in the past.
</div>
</body>
</html>
```

The column-rule-width property specifies the width of the rule between columns:

Example

```
div {
  -webkit-column-rule-width: 1px; /* Chrome, Safari, Opera */
  -moz-column-rule-width: 1px; /* Firefox */
  column-rule-width: 1px;
}
<!DOCTYPE html>
<html>
<head>
<style>
.newspaper {
```

```
  -webkit-column-count: 3; /* Chrome, Safari, Opera */
  -moz-column-count: 3; /* Firefox */
  column-count: 3;
  -webkit-column-gap: 40px; /* Chrome, Safari, Opera */
  -moz-column-gap: 40px; /* Firefox */
  column-gap: 40px;
  -webkit-column-rule-style: solid; /* Chrome, Safari, Opera */
  -moz-column-rule-style: solid; /* Firefox */
  column-rule-style: solid;
  -webkit-column-rule-width: 1px; /* Chrome, Safari, Opera */
  -moz-column-rule-width: 1px; /* Firefox */
  column-rule-width: 1px;
}
</style>
</head>
<body>

<p><b>Note:</b> Internet Explorer 9, and earlier versions, does not support the column-rule-width property.</p>
<div class="newspaper">
```

Future Vision Computer Institute was founded in 2006 with the mission of providing best quality Computer education to our students. Thousands of students have already got trained professionally and made their career successfully in the past.

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

```
</html>
```

The column-rule-color property specifies the color of the rule between columns:

Example

```
div {
  -webkit-column-rule-color: lightblue; /* Chrome, Safari, Opera */
  -moz-column-rule-color: lightblue; /* Firefox */
  column-rule-color: lightblue;
}

<!DOCTYPE html>

<html>

<head>

<style>

.newspaper {

  -webkit-column-count: 3; /* Chrome, Safari, Opera */

  -moz-column-count: 3; /* Firefox */

  column-count: 3;

  -webkit-column-gap: 40px; /* Chrome, Safari, Opera */

  -moz-column-gap: 40px; /* Firefox */

  column-gap: 40px;

  -webkit-column-rule-style: solid; /* Chrome, Safari, Opera */

  -moz-column-rule-style: solid; /* Firefox */

  column-rule-style: solid;

  -webkit-column-rule-width: 1px; /* Chrome, Safari, Opera */

  -moz-column-rule-width: 1px; /* Firefox */

  column-rule-width: 1px;

  -webkit-column-rule-color: lightblue; /* Chrome, Safari, Opera */

  -moz-column-rule-color: lightblue; /* Firefox */
```

```
  column-rule-color: lightblue;
}
</style>
</head>
<body>
<p><b>Note:</b> Internet Explorer 9, and earlier versions, does not support the column-rule-color property.</p>
<div class="newspaper">
Future Vision Computer Institute was founded in 2006 with the mission of providing best quality Computer education to our students. Thousands of students have already got trained professionally and made their career successfully in the past.
</div>
</body>
</html>
```

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

The following example sets the width, style, and color of the rule between columns:

Example

```
div {
  -webkit-column-rule: 1px solid lightblue; /* Chrome, Safari, Opera */
  -moz-column-rule: 1px solid lightblue; /* Firefox */
  column-rule: 1px solid lightblue;
}
<!DOCTYPE html>
<html>
<head>
<style>
.newspaper {
```

```
  -webkit-column-count: 3; /* Chrome, Safari, Opera */
  -moz-column-count: 3; /* Firefox */
  column-count: 3;
  -webkit-column-gap: 40px; /* Chrome, Safari, Opera */
  -moz-column-gap: 40px; /* Firefox */
  column-gap: 40px;
  -webkit-column-rule: 1px solid lightblue; /* Chrome, Safari, Opera */
  -moz-column-rule: 1px solid lightblue; /* Firefox */
  column-rule: 1px solid lightblue;
}
</style>
</head>
<body>

<p><b>Note:</b> Internet Explorer 9, and earlier versions, does not support the column-rule property.</p>
<div class="newspaper">
Future Vision Computer Institute was founded in 2006 with the mission of providing best quality Computer education to our students. Thousands of students have already got trained professionally and made their career successfully in the past.
</div>
</body>
</html>
```

SPECIFY HOW MANY COLUMNS AN ELEMENT SHOULD SPAN

The column-span property specifies how many columns an element should span across.

The following example specifies that the <h2> element should span across all columns:

Example

```
h2 {
  -webkit-column-span: all; /* Chrome, Safari, Opera */
  column-span: all;
}

<!DOCTYPE html>

<html>

<head>

<style>

.newspaper {

  -webkit-column-count: 3; /* Chrome, Safari, Opera */

  -moz-column-count: 3; /* Firefox */

  column-count: 3;

  -webkit-column-gap: 40px; /* Chrome, Safari, Opera */

  -moz-column-gap: 40px; /* Firefox */

  column-gap: 40px;

  -webkit-column-rule: 1px solid lightblue; /* Chrome, Safari, Opera */

  -moz-column-rule: 1px solid lightblue; /* Firefox */

  column-rule: 1px solid lightblue;

}


h2 {

  -webkit-column-span: all; /* Chrome, Safari, Opera */

  column-span: all;

}

</style>
```

```
</head>

<body>

<p><b>Note:</b> Firefox and Internet Explorer 9 (and earlier versions) do not support the column-span property.</p>

<div class="newspaper">

<h2>Lorem Ipsum Dolor Sit Amet</h2>Future Vision Computer Institute was founded in 2006 with the mission of providing best quality Computer education to our students. Thousands of students have already got trained professionally and made their career successfully in the past.

</div>

</body>

</html>
```

SPECIFY THE COLUMN WIDTH

The column-width property specifies a suggested, optimal width for the columns.

The following example specifies that the suggested, optimal width for the columns should be 100px:

Example

```
div {
  -webkit-column-width: 100px; /* Chrome, Safari, Opera */
  column-width: 100px;
}

<!DOCTYPE html>

<html>

<head>

<style>

.newspaper {

  -webkit-column-count: 3; /* Chrome, Safari, Opera */

  -moz-column-count: 3; /* Firefox */

  column-count: 3;
```

```
  -webkit-column-width: 100px; /* Chrome, Safari, Opera */
  -moz-column-width: 100px; /* Firefox */
  column-width: 100px;
}
</style>
</head>
<body>
<p><b>Note:</b> Internet Explorer 9, and earlier versions, does not support the column-width property.</p>
<div class="newspaper">
Future Vision Computer Institute was founded in 2006 with the mission of providing best quality Computer education to our students. Thousands of students have already got trained professionally and made their career successfully in the past.
</div>
</body>
</html>
```

Chapter 16 CSS3 USER INTERFACE

CSS3 USER INTERFACE

CSS3 has new user interface features such as resizing elements, outlines, and box sizing.

In this chapter you will learn about the following user interface properties:

- ✓ resize
- ✓ outline-offset

CSS3 RESIZING

The resize property specifies whether or not an element should be resizable by the user.

This div element is resizable by the user (works in Chrome, Firefox, Safari and Opera).

The following example lets the user resize only the width of a element:

Example:

```
div {
  resize: horizontal;
  overflow: auto;
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

div {

  border: 2px solid;

  padding: 20px;

  width: 300px;

  resize: horizontal;

  overflow: auto;

}

</style>

</head>
```

```
<body>

<p><b>Note:</b> Internet Explorer does not support the resize property.</p>

<div>Let the user resize the width of this div element.</div>

</body>

</html>
```

The following example lets the user resize only the height of a element:

Example:

```
div {
  resize: vertical;
  overflow: auto;
}

<!DOCTYPE html>

<html>

<head>

<style>

div {

  border: 2px solid;

  padding: 20px;

  width: 300px;

  resize: vertical;

  overflow: auto;

}

</style>

</head>

<body>

<p><b>Note:</b> Internet Explorer does not support the resize property.</p>
```

```
<div>Let the user resize the height of this div element.</div>

</body>

</html>
```

The following example lets the user resize both the height and the width of a element:

Example

```
div {
  resize: both;
  overflow: auto;
}

<!DOCTYPE html>

<html>

<head>

<style>

div {

  border: 2px solid;

  padding: 20px;

  width: 300px;

  resize: both;

  overflow: auto;

}

</style>

</head>

<body>

<p><b>Note:</b> Internet Explorer does not support the resize property.</p>

<div>Let the user resize both the height and the width of this div element.</div>

</body>
```

```
</html>
```

CSS3 OUTLINE OFFSET

The outline-offset property adds space between an outline and the edge or border of an element.

Outlines differ from borders in two ways:

- ✓ An outline is a line drawn around elements, outside the border edge
- ✓ An outline do not take up space
- ✓ An outline may be non-rectangular

This div has an outline 15px outside the border edge.

The following example uses the outline-offset property to add a 15px space between the border and the outline:

Example

```
div {
  border: 1px solid black;
  outline: 1px solid red;
  outline-offset: 15px;
}

<!DOCTYPE html>

<html>

<head>

<style>

div {

  margin: 20px;

  padding: 10px;

  width: 300px;

  height: 100px;

  border: 1px solid black;

  outline: 1px solid red;
```

```
  /* Move the outline 15px away from the border */
  outline-offset: 15px;
}
</style>
</head>
<body>
<p><b>Note:</b> Internet Explorer does not support the outline-offset property.</p>
<div>This div has an outline 15px outside the border edge.</div>
</body>
</html>
```

Chapter 17 CSS3 BOX SIZING

CSS3 BOX SIZING

The CSS3 box-sizing property allows us to include the padding and border in an element's total width and height.

WITHOUT THE CSS3 BOX-SIZING PROPERTY

By default, the width and height of an element is calculated like this:

width+padding+border=actual width of an element
height + padding + border = actual height of an element

This means: When you set the width/height of an element, the element often appear bigger than you have set (because the element's border and padding are added to the element's specified width/height).

The following illustration shows two elements with the same specified width and height:

This div is smaller (width is 300px and height is 100px).

This div is bigger (width is also 300px and height is 100px).

The two elements above end up with different sizes in the result (because div2 has a padding specified):

Example

```
.div1 {
  width: 300px;
  height: 100px;
  border: 1px solid blue;
}

.div2 {
  width: 300px;
  height: 100px;
  padding: 50px;
  border: 1px solid red;
}

<!DOCTYPE html>

<html>

<head>
```

```
<style>
.div1 {
  width: 300px;
  height: 100px;
  border: 1px solid blue;
}

.div2 {
  width: 300px;
  height: 100px;
  padding: 50px;
  border: 1px solid red;
}
</style>
</head>
<body>
<div class="div1">This div is smaller (width is 300px and height is 100px).</div>
<br>
<div class="div2">This div is bigger (width is also 300px and height is 100px).</div>
</body>
</html>
```

So, for a long time web developers have specified a smaller width value than they wanted, because they had to subtract out the padding and borders.

With CSS3, the box-sizing property solves this problem.

WITH THE CSS3 BOX-SIZING PROPERTY

The CSS3 box-sizing property allows us to include the padding and border in an element's total width and height.

If you set box-sizing: border-box; on an element padding and border are included in the width and height:

Both divs are the same size now!

Hooray!

Here is the same example as above, with box-sizing: border-box; added to both elements:

Example:

```
.div1 {
  width: 300px;
  height: 100px;
  border: 1px solid blue;
  box-sizing: border-box;
}

.div2 {
  width: 300px;
  height: 100px;
  padding: 50px;
  border: 1px solid red;
  box-sizing: border-box;
}
<!DOCTYPE html>
<html>
<head>
<style>
.div1 {
  width: 300px;
  height: 100px;
```

```
    border: 1px solid blue;
    box-sizing: border-box;
}
.div2 {
    width: 300px;
    height: 100px;
    padding: 50px;
    border: 1px solid red;
    box-sizing: border-box;
}
</style>
</head>
<body>
<div class="div1">Both divs are the same size now!</div>
<br>
<div class="div2">Hooray!</div>
</body>
</html>
```

Since the result of using the box-sizing: border-box; is so much better, many developers want all elements on their pages to work this way.

The code below ensures that all elements are sized in this more intuitive way. Many browsers already use box-sizing: border-box; for many form elements (but not all - which is why inputs and textareas look different at width: 100%;).

Applying this to all elements is safe and wise:

Example:

```
* {
  box-sizing: border-box;
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

* {

  box-sizing: border-box;

}


input, textarea {

  width: 100%;

}

</style>

</head>

<body>

<form action="action_page.php">

First name:<br>

<input type="text" name="firstname" value="Mickey"><br>
```

```
Last name:<br>

<input type="text" name="lastname" value="Mouse"><br>

Comments:<br>

<textarea name="message" rows="5" cols="30">

</textarea>

<br><br>

<input type="submit" value="Submit">

</form>

<p><strong>Tip:</strong> Try to remove the box-sizing property from the style element
and look what happens.

Notice that the input, textarea, and submit button will no longer have equal widths.</p>

</body>

</html>
```

Chapter 18 CSS3 FLEXBOX

CSS3 FLEXBOX

Flexible boxes, or flexbox, is a new layout mode in CSS3.

Use of flexbox ensures that elements behave predictably when the page layout must accommodate different screen sizes and different display devices.

For many applications, the flexible box model provides an improvement over the block model in that it does not use floats, nor do the flex container's margins collapse with the margins of its contents.

CSS3 FLEXBOX CONCEPTS

Flexbox consists of flex containers and flex items.

A flex container is declared by setting the display property of an element to either flex (rendered as a block) or inline-flex (rendered as inline).

Inside a flex container there is one or more flex items.

Note: Everything outside a flex container and inside a flex item is rendered as usual. Flexbox defines how flex items are laid out inside a flex container.

Flex items are positioned inside a flex container along a flex line. By default there is only one flex line per flex container.

The following example shows three flex items. They are positioned by default: along the horizontal flex line, from left to right:

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
  display: flex;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}

.flex-item {
  background-color: cornflowerblue;
```

```
  width: 100px;
  height: 100px;
  margin: 10px;
}
</style>
</head>
<body>

<div class="flex-container">
 <div class="flex-item">flex item 1</div>
 <div class="flex-item">flex item 2</div>
 <div class="flex-item">flex item 3</div>
</div>

</body>
</html>
```

```
<!DOCTYPE html>

<html>

<head>

<style>

.flex-container {

  display: -webkit-flex;

  display: flex;

  width: 400px;

  height: 250px;

  background-color: lightgrey;

}


.flex-item {

  background-color: cornflowerblue;

  width: 100px;
```

```
    height: 100px;
    margin: 10px;
  }
</style>
</head>
<body>
<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

It is also possible to change the direction of the flex line.

If we set the direction property to rtl (right-to-left), the text is drawn right to left, and also the flex line changes direction, which will change the page layout:

Example

```
body {
  direction: rtl;
}

.flex-container {
  display: -webkit-flex;
  display: flex;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}

.flex-item {
```

```
    background-color: cornflowerblue;
    width: 100px;
    height: 100px;
    margin: 10px;
  }
```

FLEX DIRECTION

The flex-direction property specifies the direction of the flexible items inside the flex container. The default value of flex-direction is row (left-to-right, top-to-bottom).

The other values are as follows:

- row-reverse - If the writing-mode (direction) is left to right, the flex items will be laid out right to left
- column - If the writing system is horizontal, the flex items will be laid out vertically
- column-reverse - Same as column, but reversed

The following example shows the result of using the row-reverse value:

Example

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-flex-direction: row-reverse;
  flex-direction: row-reverse;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
```

```
    display: flex;
    -webkit-flex-direction: row-reverse;
    flex-direction: row-reverse;
    width: 400px;
    height: 250px;
    background-color: lightgrey;
}

.flex-item {
    background-color: cornflowerblue;
    width: 100px;
    height: 100px;
    margin: 10px;
}
</style>
</head>
<body>

<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>

</body>
```

```
</html>
```

The following example shows the result of using the column value:

Example:

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-flex-direction: column;
  flex-direction: column;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-flex-direction: column;
  flex-direction: column;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}

.flex-item {
  background-color: cornflowerblue;
  width: 100px;
  height: 100px;
  margin: 10px;
}
</style>
</head>
<body>
```

```
<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

The following example shows the result of using the column-reverse value:

Example:

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-flex-direction: column-reverse;
  flex-direction: column-reverse;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-flex-direction: column-reverse;
  flex-direction: column-reverse;
  width: 400px;
```

```
    height: 250px;
    background-color: lightgrey;
}

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

THE JUSTIFY-CONTENT PROPERTY

The justify-content property horizontally aligns the flexible container's items when the items do not use all available space on the main-axis.

The possible values are as follows:

- ✓ flex-start - Default value. Items are positioned at the beginning of the container
- ✓ flex-end - Items are positioned at the end of the container
- ✓ center - Items are positioned at the center of the container

- ✓ space-between - Items are positioned with space between the lines
- ✓ space-around - Items are positioned with space before, between, and after the lines

The following example shows the result of using the flex-end value:

Example:

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-justify-content: flex-end;
  justify-content: flex-end;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

.flex-container {

  display: -webkit-flex;

  display: flex;

  -webkit-justify-content: flex-end;

  justify-content: flex-end;

  width: 400px;

  height: 250px;

  background-color: lightgrey;

}


.flex-item {
```

```
    background-color: cornflowerblue;
    width: 100px;
    height: 100px;
    margin: 10px;
}
</style>
</head>
<body>
<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

The following example shows the result of using the center value:

Example:

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-justify-content: center;
  justify-content: center;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

.flex-container {

  display: -webkit-flex;

  display: flex;

  -webkit-justify-content: center;

  justify-content: center;

  width: 400px;

  height: 250px;

  background-color: lightgrey;

}

.flex-item {

  background-color: cornflowerblue;

  width: 100px;
```

```
    height: 100px;
    margin: 10px;
  }
</style>
</head>
<body>
<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

The following example shows the result of using the space-between value:

Example:

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-justify-content: space-between;
  justify-content: space-between;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-justify-content: space-between;
  justify-content: space-between;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}

.flex-item {
  background-color: cornflowerblue;
  width: 100px;
  height: 100px;
  margin: 10px;
}
</style>
</head>
<body>
<div class="flex-container">
 <div class="flex-item">flex item 1</div>
```

```
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

The following example shows the result of using the space-around value:

Example:

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-justify-content: space-around;
  justify-content: space-around;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}

<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-justify-content: space-around;
  justify-content: space-around;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
```

```
}

.flex-item {
  background-color: cornflowerblue;
  width: 100px;
  height: 100px;
  margin: 10px;
}
</style>
</head>
<body>

<div class="flex-container">
 <div class="flex-item">flex item 1</div>
 <div class="flex-item">flex item 2</div>
 <div class="flex-item">flex item 3</div>
</div>

</body>
</html>
```

THE ALIGN-ITEMS PROPERTY

The align-items property vertically aligns the flexible container's items when the items do not use all available space on the cross-axis.

The possible values are as follows:

- ✓ stretch - Default value. Items are stretched to fit the container
- ✓ flex-start - Items are positioned at the top of the container
- ✓ flex-end - Items are positioned at the bottom of the container
- ✓ center - Items are positioned at the center of the container (vertically)
- ✓ baseline - Items are positioned at the baseline of the container

The following example shows the result of using the stretch value (this is the default value):

Example:

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-align-items: stretch;
  align-items: stretch;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

.flex-container {

  display: -webkit-flex;

  display: flex;

  -webkit-align-items: stretch;

  align-items: stretch;

  width: 400px;

  height: 250px;

  background-color: lightgrey;
```

```
}

.flex-item {
   background-color: cornflowerblue;
   width: 100px;
   margin: 10px;
}
</style>
</head>
<body>
<div class="flex-container">
 <div class="flex-item">flex item 1</div>
 <div class="flex-item">flex item 2</div>
 <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

The following example shows the result of using the flex-start value:

Example

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-align-items: flex-start;
  align-items: flex-start;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-align-items: flex-start;
  align-items: flex-start;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}

.flex-item {
```

```
    background-color: cornflowerblue;
    width: 100px;
    margin: 10px;
  }
</style>
</head>
<body>
<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

The following example shows the result of using the flex-end value:

Example

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-align-items: flex-end;
  align-items: flex-end;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-align-items: flex-end;
  align-items: flex-end;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}

.flex-item {
  background-color: cornflowerblue;
  width: 100px;
  margin: 10px;
}
</style>
</head>
<body>

<div class="flex-container">
```

```
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

The following example shows the result of using the center value:

Example:

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-align-items: center;
  align-items: center;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-align-items: center;
  align-items: center;
  width: 400px;
  height: 250px;
```

```
    background-color: lightgrey;
}
.flex-item {
    background-color: cornflowerblue;
    width: 100px;
    margin: 10px;
}
</style>
</head>
<body>
<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

The following example shows the result of using the baseline value:

Example:

```
.flex-container {
  display: -webkit-flex;
  display: flex;
 -webkit-align-items: baseline;
  align-items: baseline;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

.flex-container {

  display: -webkit-flex;

  display: flex;

  -webkit-align-items: baseline;

  align-items: baseline;

  width: 400px;

  height: 250px;

  background-color: lightgrey;

}


.flex-item {
```

```
    background-color: cornflowerblue;
    width: 100px;
    margin: 10px;
  }
</style>
</head>
<body>

<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

THE FLEX-WRAP PROPERTY

The flex-wrap property specifies whether the flex items should wrap or not, if there is not enough room for them on one flex line.

The possible values are as follows:

- ✓ nowrap - Default value. The flexible items will not wrap
- ✓ wrap - The flexible items will wrap if necessary
- ✓ wrap-reverse - The flexible items will wrap, if necessary, in reverse order

The following example shows the result of using the nowrap value (this is the default value):

Example:

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-flex-wrap: nowrap;
  flex-wrap: nowrap;
  width: 300px;
  height: 250px;
  background-color: lightgrey;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-flex-wrap: nowrap;
  flex-wrap: nowrap;
  width: 300px;
  height: 250px;
  background-color: lightgrey;
}
.flex-item {
  background-color: cornflowerblue;
  width: 100px;
```

```
    height: 100px;
    margin: 10px;
  }
</style>
</head>
<body>
<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

The following example shows the result of using the wrap value:

Example:

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-flex-wrap: wrap;
  flex-wrap: wrap;
  width: 300px;
  height: 250px;
  background-color: lightgrey;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-flex-wrap: wrap;
  flex-wrap: wrap;
  width: 300px;
  height: 250px;
  background-color: lightgrey;
}
.flex-item {
  background-color: cornflowerblue;
  width: 100px;
  height: 100px;
  margin: 10px;
}
</style>
</head>
<body>

<div class="flex-container">
```

```
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

The following example shows the result of using the wrap-reverse value:

Example

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-flex-wrap: wrap-reverse;
  flex-wrap: wrap-reverse;
  width: 300px;
  height: 250px;
  background-color: lightgrey;
}
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-flex-wrap: wrap-reverse;
  flex-wrap: wrap-reverse;
  width: 300px;
  height: 250px;
```

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

THE ALIGN-CONTENT PROPERTY

The align-content property modifies the behavior of the flex-wrap property. It is similar to align-items, but instead of aligning flex items, it aligns flex lines.

The possible values are as follows:

- ✓ stretch - Default value. Lines stretch to take up the remaining space
- ✓ flex-start - Lines are packed toward the start of the flex container
- ✓ flex-end - Lines are packed toward the end of the flex container
- ✓ center - Lines are packed toward the center of the flex container
- ✓ space-between - Lines are evenly distributed in the flex container

- ✓ space-around - Lines are evenly distributed in the flex container, with half-size spaces on either end

The following example shows the result of using the center value:

Example:

```
.flex-container {
  display: -webkit-flex;
  display: flex;
  -webkit-flex-wrap: wrap;
  flex-wrap: wrap;
  -webkit-align-content: center;
  align-content: center;
  width: 300px;
  height: 300px;
  background-color: lightgrey;
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

.flex-container {

  display: -webkit-flex;

  display: flex;

  -webkit-flex-wrap: wrap;

  flex-wrap: wrap;

  -webkit-align-content: center;

  align-content: center;

  width: 300px;

  height: 300px;

  background-color: lightgrey;
```

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

FLEX ITEM PROPERTIES

ORDERING

The order property specifies the order of a flexible item relative to the rest of the flexible items inside the same container:

Example:

```
.flex-item {
  background-color: cornflowerblue;
  width: 100px;
  height: 100px;
  margin: 10px;
}

.first {
  -webkit-order: -1;
  order: -1;
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

.flex-container {

  display: -webkit-flex;

  display: flex;

  width: 400px;

  height: 250px;

  background-color: lightgrey;

}


.flex-item {
```

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

MARGIN

Setting margin: auto; will absorb extra space. It can be used to push flex items into different positions.

In the following example we set margin-right: auto; on the first flex item. This will cause all the extra space to be absorbed to the right of that element:

Example:

```
.flex-item {
    background-color: cornflowerblue;
```

```
  width: 75px;
  height: 75px;
  margin: 10px;
}

.flex-item:first-child {
  margin-right: auto;
}
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
  display: flex;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}

.flex-item {
  background-color: cornflowerblue;
  width: 75px;
  height: 75px;
  margin: 10px;
}
.flex-item:first-child {
```

```
  margin-right: auto;
}


</style>
</head>
<body>
<div class="flex-container">
 <div class="flex-item">flex item 1</div>
 <div class="flex-item">flex item 2</div>
 <div class="flex-item">flex item 3</div>
</div>
</body>
</html>
```

PERFECT CENTERING

In the following example we will solve an almost daily problem: perfect centering.

It is very easy with flexbox. Setting margin: auto; will make the item perfectly centered in both axis:

Example:

```
.flex-item {
  background-color: cornflowerblue;
  width: 75px;
  height: 75px;
  margin: auto;
}
<!DOCTYPE html>
<html>
<head>
<style>
```

```
.flex-container {
   display: -webkit-flex;
   display: flex;
   width: 400px;
   height: 250px;
   background-color: lightgrey;
}

.flex-item {
   background-color: cornflowerblue;
   width: 75px;
   height: 75px;
   margin: auto;
}
</style>
</head>
<body>

<div class="flex-container">
  <div class="flex-item">Perfect centering!</div>
</div>

</body>
</html>
```

ALIGN-SELF

The align-self property of flex items overrides the flex container's align-items property for that item. It has the same possible values as the align-items property.

The following example sets different align-self values to each flex item:

Example

```
.flex-item {
  background-color: cornflowerblue;
  width: 60px;
  min-height: 100px;
  margin: 10px;
}

.item1 {
  -webkit-align-self: flex-start;
  align-self: flex-start;
}
.item2 {
  -webkit-align-self: flex-end;
  align-self: flex-end;
}

.item3 {
  -webkit-align-self: center;
  align-self: center;
}

.item4 {
  -webkit-align-self: baseline;
  align-self: baseline;
}

.item5 {
  -webkit-align-self: stretch;
  align-self: stretch;
}

<!DOCTYPE html>
```

```
<html>
<head>
<style>
.flex-container {
  display: -webkit-flex;
  display: flex;
  width: 400px;
  height: 250px;
  background-color: lightgrey;
}

.flex-item {
  background-color: cornflowerblue;
  width: 60px;
  min-height: 100px;
  margin: 10px;
}

.item1 {
  -webkit-align-self: flex-start;
  align-self: flex-start;
}
.item2 {
  -webkit-align-self: flex-end;
  align-self: flex-end;
```

```
}

.item3 {
   -webkit-align-self: center;
   align-self: center;
}

.item4 {
   -webkit-align-self: baseline;
   align-self: baseline;
}

.item5 {
   -webkit-align-self: stretch;
   align-self: stretch;
}
</style>
</head>
<body>
<div class="flex-container">
 <div class="flex-item item1">flex-start</div>
 <div class="flex-item item2">flex-end</div>
 <div class="flex-item item3">center</div>
 <div class="flex-item item4">baseline</div>
 <div class="flex-item item5">stretch</div>
```

```
</div>

</body>

</html>
```

FLEX

The flex property specifies the length of the flex item, relative to the rest of the flex items inside the same container.

In the following example, the first flex item will consume 2/4 of the free space, and the other two flex items will consume 1/4 of the free space each:

Example

```
.flex-item {
  background-color: cornflowerblue;
  margin: 10px;
}

.item1 {
  -webkit-flex: 2;
  flex: 2;
}

.item2 {
  -webkit-flex: 1;
  flex: 1;
}

.item3 {
  -webkit-flex: 1;
  flex: 1;
}
<!DOCTYPE html>

<html>

<head>

<style>

.flex-container {
```

```
    display: -webkit-flex;
    display: flex;
    width: 400px;
    height: 250px;
    background-color: lightgrey;
}

.flex-item {
    background-color: cornflowerblue;
    margin: 10px;
}

.item1 {
    -webkit-flex: 2;
    flex: 2;
}

.item2 {
    -webkit-flex: 1;
    flex: 1;
}

.item3 {
    -webkit-flex: 1;
    flex: 1;
```

```
}
</style>
</head>
<body>
<div class="flex-container">
  <div class="flex-item item1">flex item 1</div>
  <div class="flex-item item2">flex item 2</div>
  <div class="flex-item item3">flex item 3</div>
</div>
</body>
</html>
```

MORE EXAMPLES

Create a responsive website with flexbox.
This example explains how to create a responsive website layout with flexbox.

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
    display: -webkit-flex;
    display: flex;
    -webkit-flex-flow: row wrap;
    flex-flow: row wrap;
    font-weight: bold;
    text-align: center;
}
```

```
.flex-container > * {
   padding: 10px;
   flex: 1 100%;
}
.main {
   text-align: left;
   background: cornflowerblue;
}
.header {background: coral;}
.footer {background: lightgreen;}
.aside1 {background: moccasin;}
.aside2 {background: violet;}
@media all and (min-width: 600px) {
   .aside { flex: 1 auto; }
}

@media all and (min-width: 800px) {
   .main    { flex: 3 0px; }
   .aside1 { order: 1; }
   .main    { order: 2; }
   .aside2 { order: 3; }
   .footer  { order: 4; }
}
</style>
```

```
</head>
<body>
<div class="flex-container">
 <header class="header">Header</header>
 <article class="main">
  <p>
```

Future Vision Computer Institute was founded in 2006 with the mission of providing best quality Computer education to our students. Thousands of students have already got trained professionally and made their career successfully in the past.

```
  </p>
 </article>
 <aside class="aside aside1">Aside 1</aside>
 <aside class="aside aside2">Aside 2</aside>
 <footer class="footer">Footer</footer>
</div>
</body>
</html>
```

Chapter 19 CSS3 Filters

The CSS filter property adds visual effects (like blur and saturation) to any HTML element - often <img> elements:

Example:

Change the color of all images to black and white (100% gray):

```
img {
  -webkit-filter: grayscale(100%); /* Chrome, Safari, Opera */
  filter: grayscale(100%);
}
```

Note: The filter property is not supported in Internet Explorer.

```
<!DOCTYPE html>

<html>

<head>

<style>

img {

  -webkit-filter: grayscale(100%); /* Chrome, Safari, Opera */

  filter: grayscale(100%);

}

</style>

</head>

<body>

<p>Convert the image to grayscale:</p>

<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">


<p><strong>Note:</strong> The filter property is not supported in Internet Explorer.</p>

</body>

</html>
```

FILTER FUNCTIONS

Note: The filters that use percentage values (i.e. 75%), also accept the value as decimal (i.e. 0.75).

Blur (*px*)

Applies a blur effect to the image. A larger value will create more blur. If no value is specified, 0 is used.

```
img {
  -webkit-filter: blur(5px); /* Chrome, Safari, Opera */
  filter: blur(5px);
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

img {

  -webkit-filter: blur(5px); /* Chrome, Safari, Opera */

  filter: blur(5px);

}

</style>

</head>

<body>

<p>Apply a blur effect to the image:</p>

<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<p><strong>Note:</strong> The filter property is not supported in Internet Explorer.</p>

</body></html>
```

BRIGHTNESS(%)

Adjusts the brightness of the image. 0% (0) will make the image completely black. 100% (1) is default and represents the original image. Values over 100% will provide brighter results.

```
img {
  -webkit-filter: brightness(200%); /* Chrome, Safari, Opera */
  filter: brightness(200%);
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
  -webkit-filter: brightness(200%); /* Chrome, Safari, Opera */
  filter: brightness(200%);
}
</style>
</head>
<body>

<p>Adjust the brightness of the image:</p>
<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">
<p><strong>Note:</strong> The filter property is not supported in Internet Explorer.</p>

</body>
</html>
```

CONTRAST(%)

Adjusts the contrast of the image. 0% (0) will make the image completely black. 100% (1) is default and represents the original image. Values over 100% will provide results with less contrast.

```
img {
  -webkit-filter: contrast(200%); /* Chrome, Safari, Opera */
  filter: contrast(200%);
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

img {

  -webkit-filter: contrast(200%); /* Chrome, Safari, Opera */

  filter: contrast(200%);

}

</style>

</head>

<body>

<p>Adjust the contrast of the image:</p>

<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<p><strong>Note:</strong> The filter property is not supported in Internet Explorer.</p>

</body>

</html>
```

DROP SHADOW(H-SHADOW V-SHADOW BLUR COLOR)

Possible values:

h-shadow - Required. Specifies a pixel value for the horizontal shadow. Negative values place the shadow to the left of the image.

v-shadow - Required. Specifies a pixel value for the vertical shadow. Negative values place the shadow above the image.

***blur* – Optional -** This is the third value, and must be in pixels. Adds a blur effect to the shadow. A larger value will create more blur (the shadow becomes bigger and lighter). Negative values are not allowed. If no value is specified, 0 is used (the shadow's edge is sharp).

Color - Optional. Adds a color to the shadow. If not specified, the color depends on the browser (often black).

Tip: This filter is similar to the box-shadow property.

```
img {
  -webkit-filter: drop-shadow(8px 8px 10px red); /* Chrome, Safari, Opera */
  filter: drop-shadow(8px 8px 10px red);
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
  -webkit-filter: drop-shadow(8px 8px 10px red); /* Chrome, Safari, Opera */
  filter: drop-shadow(8px 8px 10px red);
}
</style>
</head>
<body>
```

```
<p>Apply a drop shadow effect to the image:Apply a drop shadow effect to the image:</p>

<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<p><strong>Note:</strong> The filter property is not supported in Internet Explorer.</p>

</body>

</html>
```

GRAYSCALE(%)

Converts the image to grayscale. 0% (0) is default and represents the original image. 100% (1) will make the image completely gray (used for black and white images).

Negative values are not allowed.

```
img {
  -webkit-filter: grayscale(50%); /* Chrome, Safari, Opera */
  filter: grayscale(50%);
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

img {

  -webkit-filter: grayscale(50%); /* Chrome, Safari, Opera */

  filter: grayscale(50%);

}

</style>

</head>

<body>

<p>Convert the image to grayscale:</p>
```

```
<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<p><strong>Note:</strong> The filter property is not supported in Internet
Explorer.</p>

</body>

</html>
```

HUE ROTATION(*DEG*)

Applies a hue rotation on the image. The value defines the number of degrees around the color circle the image samples will be adjusted. 0deg is default, and represents the original image.

Maximum value is 360deg.

```
img {
  -webkit-filter: hue-rotate(90deg); /* Chrome, Safari, Opera */
  filter: hue-rotate(90deg);
}

<!DOCTYPE html>

<html>

<head>

<style>

img {

  -webkit-filter: hue-rotate(90deg); /* Chrome, Safari, Opera */

  filter: hue-rotate(90deg);

}

</style>

</head>

<body>

<p>Apply a hue rotation on the image:</p>

<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">
```

```
<p><strong>Note:</strong> The filter property is not supported in Internet
Explorer.</p>

</body>

</html>
```

INVERT(%)

Inverts the samples in the image. 0% (0) is default and represents the original image. 100% (1) will make the image completely inverted.

Negative values are not allowed.

```
img {
  -webkit-filter: invert(100%); /* Chrome, Safari, Opera */
  filter: invert(100%);
}

<!DOCTYPE html>

<html>

<head>

<style>

img {

  -webkit-filter: invert(100%); /* Chrome, Safari, Opera */

  filter: invert(100%);

}

</style>

</head>

<body>

<p>Invert the samples in the image:</p>

<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<p><strong>Note:</strong> The filter property is not supported in Internet Explorer.</p>
```

```
</body>

</html>
```

OPACITY(%)

Sets the opacity level for the image. The opacity-level describes the transparency-level, where: 0% (0) is completely transparent. 100% (1) is default and represents the original image (no transparency).

Negative values are not allowed.

```
img {
  -webkit-filter: opacity(30%); /* Chrome, Safari, Opera */
  filter: opacity(30%);
}
```

Example:

```
<!DOCTYPE html>

<html>

<head>

<style>

img {

  -webkit-filter: opacity(30%); /* Chrome, Safari, Opera */

  filter: opacity(30%);

}

</style>

</head>

<body>

<p>Change the opacity level for the image:</p>

<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<p><strong>Note:</strong> The filter property is not supported in Internet Explorer.</p>

</body></html>
```

SATURATE(%)

Saturates the image. 0% (0) will make the image completely un-saturated. 100% (1) is default and represents the original image. Values over 100% provides super-saturated results.

Negative values are not allowed.

```
img {
  -webkit-filter: saturate(800%); /* Chrome, Safari, Opera */
  filter: saturate(800%);
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

img {

  -webkit-filter: saturate(800%); /* Chrome, Safari, Opera */

  filter: saturate(800%);

}

</style>

</head>

<body>

<p>Saturate the image:</p>

<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<p><strong>Note:</strong> The filter property is not supported in Internet Explorer.</p>

</body>

</html>
```

SEPIA(%)

Converts the image to sepia. 0% (0) is default and represents the original image. 100% (1) will make the image completely sepia.

Negative values are not allowed.

```
img {
  -webkit-filter: sepia(100%); /* Chrome, Safari, Opera */
  filter: sepia(100%);
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

img {

  -webkit-filter: sepia(100%); /* Chrome, Safari, Opera */

  filter: sepia(100%);

}

</style>

</head>

<body>

<p>Convert the image to sepia:</p>

<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<p><strong>Note:</strong> The filter property is not supported in Internet Explorer.</p>

</body>

</html>
```

COMBINING FILTERS / MULTIPLE FILTERS

To use multiple filters, separate each filter with a space.

Note: Order is important (i.e. using grayscale() after sepia() will result in a completely gray image).

```
img {
  -webkit-filter: contrast(200%) brightness(150%);  /* Chrome, Safari, Opera */
```

```
  filter: contrast(200%) brightness(150%);
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
  -webkit-filter: contrast(200%) brightness(150%);  /* Chrome, Safari, Opera */
  filter: contrast(200%) brightness(150%);
}
</style>
</head>
<body>
<p>Adjust the contrast and brightness of the image:</p>
<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">
<p><strong>Note:</strong> The filter property is not supported in Internet Explorer.</p>
</body>
</html>
```

ALL FILTERS

A demonstration of all filter functions:

```
.blur {
   -webkit-filter: blur(4px);
   filter: blur(4px);
}

.brightness {
   -webkit-filter: brightness(0.30);
   filter: brightness(0.30);
}

.contrast {
   -webkit-filter: contrast(180%);
   filter: contrast(180%);
}

.grayscale {
   -webkit-filter: grayscale(100%);
   filter: grayscale(100%);
}

.huerotate {
   -webkit-filter: hue-rotate(180deg);
   filter: hue-rotate(180deg);
}

.invert {
   -webkit-filter: invert(100%);
   filter: invert(100%);
}

.opacity {
   -webkit-filter: opacity(50%);
   filter: opacity(50%);
}
```

```
.saturate {
   -webkit-filter: saturate(7);
   filter: saturate(7);
}

.sepia {
   -webkit-filter: sepia(100%);
   filter: sepia(100%);
}

.shadow {
   -webkit-filter: drop-shadow(8px 8px 10px green);
   filter: drop-shadow(8px 8px 10px green);
}
<!DOCTYPE html>
<html>
<head>
<style>
img {
   width: 33%;
   height: auto;
   float: left;
}

.blur {-webkit-filter: blur(4px);filter: blur(4px);}
.brightness {-webkit-filter: brightness(0.30);filter: brightness(0.30);}
.contrast {-webkit-filter: contrast(180%);filter: contrast(180%);}
.grayscale {-webkit-filter: grayscale(100%);filter: grayscale(100%);}
.huerotate {-webkit-filter: hue-rotate(180deg);filter: hue-rotate(180deg);}
```

```
.invert {-webkit-filter: invert(100%);filter: invert(100%);}

.opacity {-webkit-filter: opacity(50%);filter: opacity(50%);}

.saturate {-webkit-filter: saturate(7); filter: saturate(7);}

.sepia {-webkit-filter: sepia(100%);filter: sepia(100%);}

.shadow {-webkit-filter: drop-shadow(8px 8px 10px green);filter: drop-shadow(8px 8px 10px green);}

</style>

</head>

<body>

<p>An example of all filter functions.</p>

<p><strong>Note:</strong> The filter property is not supported in Internet Explorer.</p>

<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="blur" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="brightness" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="contrast" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="grayscale" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="huerotate" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="invert" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="opacity" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="saturate" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="sepia" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="shadow" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

</body>

</html>
```

Chapter 20 CSS3 MEDIA QUERIES

CSS3 INTRODUCED MEDIA TYPES

The @media rule, introduced in CSS2, made it possible to define different style rules for different media types.

Examples: You could have one set of style rules for computer screens, one for printers, one for handheld devices, one for television-type devices, and so on.

Unfortunately these media types never got a lot of support by devices, other than the print media type.

CSS3 INTRODUCES MEDIA QUERIES

Media queries in CSS3 extend the CSS2 media types idea: Instead of looking for a type of device, they look at the capability of the device.

Media queries can be used to check many things, such as:

- ✓ width and height of the viewport
- ✓ width and height of the device
- ✓ orientation (is the tablet/phone in landscape or portrait mode?)
- ✓ resolution

Using media queries are a popular technique for delivering a tailored style sheet to tablets, iPhone, and Androids.

MEDIA QUERY SYNTAX

A media query consists of a media type and can contain one or more expressions, which resolve to either true or false.

```
@media not|only mediatype and (expressions) {
  CSS-Code;
}
```

The result of the query is true if the specified media type matches the type of device the document is being displayed on and all expressions in the media query are true. When a media query is true, the corresponding style sheet or style rules are applied, following the normal cascading rules.

Unless you use the not or only operators, the media type is optional and the all type will be implied.

You can also have different stylesheets for different media:

<link rel="stylesheet" media="*mediatype* and|not|only (*expressions*)" href="*print.css*">

CSS3 MEDIA TYPES

Value	Description
all	Used for all media type devices
print	Used for printers
screen	Used for computer screens, tablets, smart-phones etc.
speech	Used for screen readers that "reads" the page out loud

---

MEDIA QUERIES SIMPLE EXAMPLES

One way to use media queries is to have an alternate CSS section right inside your style sheet.

The following example changes the background-color to lightgreen if the viewport is 480 pixels wide or wider (if the viewport is less than 480 pixels, the background-color will be pink):

Example:

```
@media screen and (min-width: 480px) {
  body {
    background-color: lightgreen;
  }
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-color: pink;
}

@media screen and (min-width: 480px) {
  body {
    background-color: lightgreen;
  }
}
</style>
</head>
<body>
<h1>Resize the browser window to see the effect!</h1>
<p>The media query will only apply if the media type is screen and the viewport is 480px wide or wider.</p>
</body>
</html>
```

The following example shows a menu that will float to the left of the page if the viewport is 480 pixels wide or wider (if the viewport is less than 480 pixels, the menu will be on top of the content):

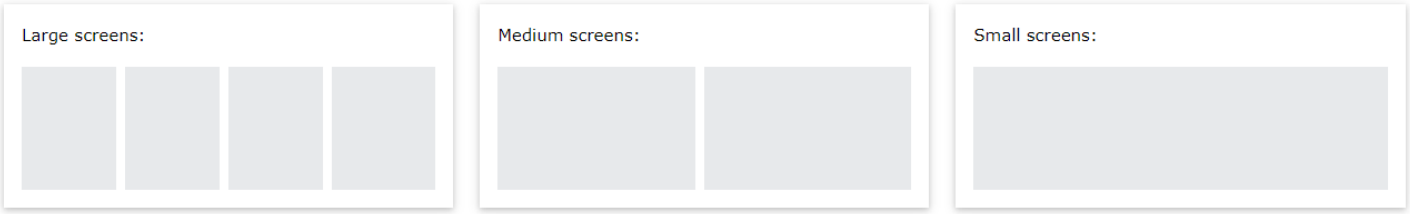
Example:

```
@media screen and (min-width: 480px) {
  #leftsidebar {width: 200px; float: left;}
  #main {margin-left:216px;}
}

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<style>

.wrapper {overflow:auto;}


#main {margin-left: 4px;}

#leftsidebar {float: none;width: auto;}

#menulist {margin:0;padding:0;}


.menuitem {
  background:#CDF0F6;
  border:1px solid #d4d4d4;
  border-radius:4px;
  list-style-type:none;
  margin:4px;
  padding:2px;
```

```
}
@media screen and (min-width: 480px) {
   #leftsidebar {width:200px;float:left;}
   #main {margin-left:216px;}
}
</style>
</head>
<body>
<div class="wrapper">
 <div id="leftsidebar">
   <ul id="menulist">
     <li class="menuitem">Menu-item 1</li>
     <li class="menuitem">Menu-item 2</li>
     <li class="menuitem">Menu-item 3</li>
     <li class="menuitem">Menu-item 4</li>
     <li class="menuitem">Menu-item 5</li>
  </ul>
 </div>
 <div id="main">
   <h1>Resize the browser window to see the effect!</h1>
   <p>This example shows a menu that will float to the left of the page if the viewport is
480 pixels wide or wider. If the viewport is less than 480 pixels, the menu will be on top
of the content.</p>
 </div>
</div>
</body></html>
```

CSS3 Media queries for columns

A common use of media queries, is to create a flexible layout. In this example, we create a layout that varies between four, two and full-width columns, depending on different screen sizes:

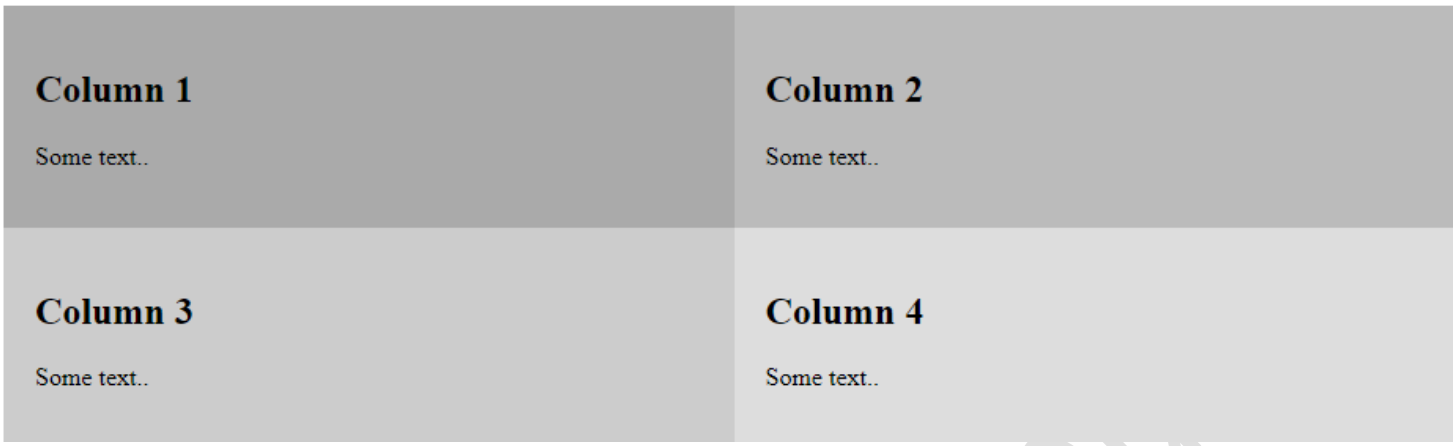



```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
* {
  box-sizing: border-box;
}

/* Create four equal columns that floats next to each other */
.column {
  float: left;
  width: 25%;
  padding: 20px;
}
```

```
/* Clear floats after the columns */
.row:after {
  content: "";
  display: table;
  clear: both;
}

/* On screens that are 992px wide or less, go from four columns to two columns */
@media screen and (max-width: 992px) {
  .column {
    width: 50%;
  }
}

/* On screens that are 600px wide or less, make the columns stack on top of each other instead of next to each other */
@media screen and (max-width: 600px) {
  .column {
    width: 100%;
  }
}
</style>
</head>
<body>
```

```
<h2>Responsive Four Column Layout</h2>

<p><strong>Resize the browser window to see the responsive effect.</strong> On screens that are 992px wide or less, the columns will resize from four columns to two columns. On screens that are 600px wide or less, the columns will stack on top of each other instead of next to eachother.</p>

<div class="row">

 <div class="column" style="background-color:#aaa;">

  <h2>Column 1</h2>

  <p>Some text..</p>

 </div>


 <div class="column" style="background-color:#bbb;">

  <h2>Column 2</h2>

  <p>Some text..</p>

 </div>


 <div class="column" style="background-color:#ccc;">

  <h2>Column 3</h2>

  <p>Some text..</p>

 </div>


 <div class="column" style="background-color:#ddd;">

  <h2>Column 4</h2>

  <p>Some text..</p>

 </div>

</div>

</body>

</html>
```

OUTPUT:

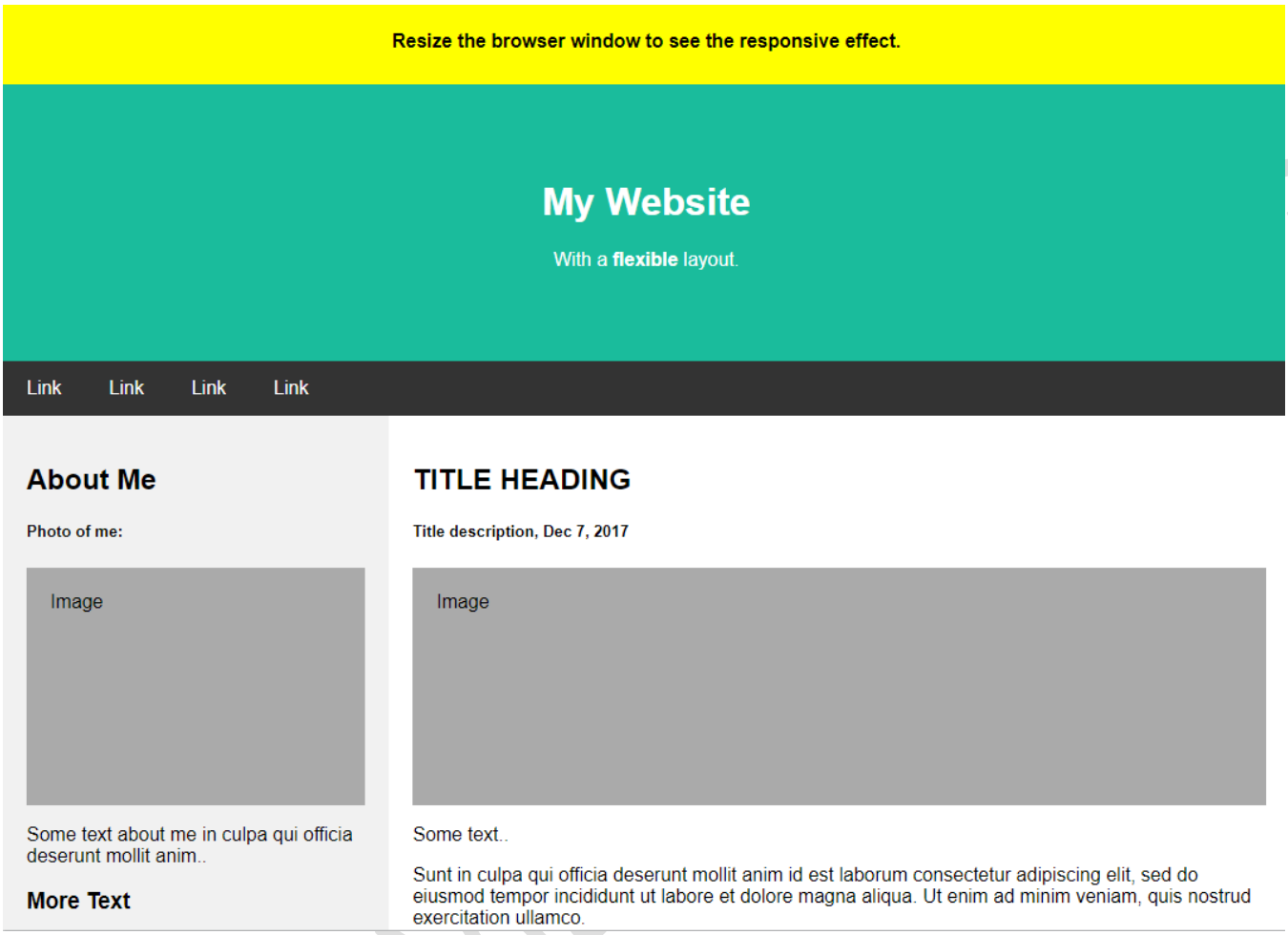


CSS3 Hide Elements with Media Queries

Another common use of media queries, is to hide elements on different screen sizes:

```
<!DOCTYPE html>
<html>
<head>
<style>
div.example {
  background-color: yellow;
  padding: 20px;
}

@media screen and (max-width: 600px) {
  div.example {
    display: none;
  }
}
</style>
</head>
<body>

<h2>Hide elements on different screen sizes</h2>

<div class="example">Example DIV.</div>

<p>When the browser's width is 600px wide or less, hide the div element. Resize the browser window to see the effect.</p>

</body>
</html>
```

Flexible Website




```
<!DOCTYPE html>

<html>

<head>

<title>Page Title</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<style>

* {

  box-sizing: border-box;

}
```

```
/* Style the top navigation bar */
.navbar {
  display: flex;
  background-color: #333;
}

/* Style the navigation bar links */
.navbar a {
  color: white;
  padding: 14px 20px;
  text-decoration: none;
  text-align: center;
}

/* Change color on hover */
.navbar a:hover {
  background-color: #ddd;
  color: black;
}

/* Column container */
.row {
  display: flex;
  flex-wrap: wrap;
}
```

```
/* Create two unequal columns that sits next to each other */

/* Sidebar/left column */

.side {

  flex: 30%;

  background-color: #f1f1f1;

  padding: 20px;

}


/* Main column */

.main {

  flex: 70%;

  background-color: white;

  padding: 20px;

}


/* Fake image, just for this example */

.fakeimg {

  background-color: #aaa;

  width: 100%;

  padding: 20px;

}


/* Footer */

.footer {

  padding: 20px;

  text-align: center;

  background: #ddd;

}
```

```
/* Responsive layout - when the screen is less than 700px wide, make the two columns stack on top of each other instead of next to each other */

@media screen and (max-width: 700px) {

  .row, .navbar {

    flex-direction: column;

  }

}

</style>

</head>

<body>


<!-- Note -->

<div style="background:yellow;padding:5px">

  <h4 style="text-align:center">Resize the browser window to see the responsive effect.</h4>

</div>


<!-- Header -->

<div class="header">

  <h1>My Website</h1>

  <p>With a <b>flexible</b> layout.</p>

</div>


<!-- Navigation Bar -->

<div class="navbar">

  <a href="#">Link</a>

  <a href="#">Link</a>

  <a href="#">Link</a>

  <a href="#">Link</a>

</div>
```

```
<!-- The flexible grid (content) -->

<div class="row">

  <div class="side">

    <h2>About Me</h2>

    <h5>Photo of me:</h5>

    <div class="fakeimg" style="height:200px;">Image</div>

    <p>Some text about me in culpa qui officia deserunt mollit anim..</p>

    <h3>More Text</h3>

    <p>Lorem ipsum dolor sit ame.</p>

    <div class="fakeimg" style="height:60px;">Image</div><br>

    <div class="fakeimg" style="height:60px;">Image</div><br>

    <div class="fakeimg" style="height:60px;">Image</div>

  </div>

  <div class="main">

    <h2>TITLE HEADING</h2>

    <h5>Title description, Dec 7, 2017</h5>

    <div class="fakeimg" style="height:200px;">Image</div>

    <p>Some text..</p>

    <p>Sunt in culpa qui officia deserunt mollit anim id est laborum consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco.</p>

    <br>
```

```
<h2>TITLE HEADING</h2>

  <h5>Title description, Sep 2, 2017</h5>

  <div class="fakeimg" style="height:200px;">Image</div>

  <p>Some text..</p>

  <p>Sunt in culpa qui officia deserunt mollit anim id est laborum consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco.</p>

 </div>

</div>


<!-- Footer -->

<div class="footer">

 <h2>Footer</h2>

</div>


</body>

</html>
```