

Cascading Style Sheets (CSS)

Content

Basic CSS

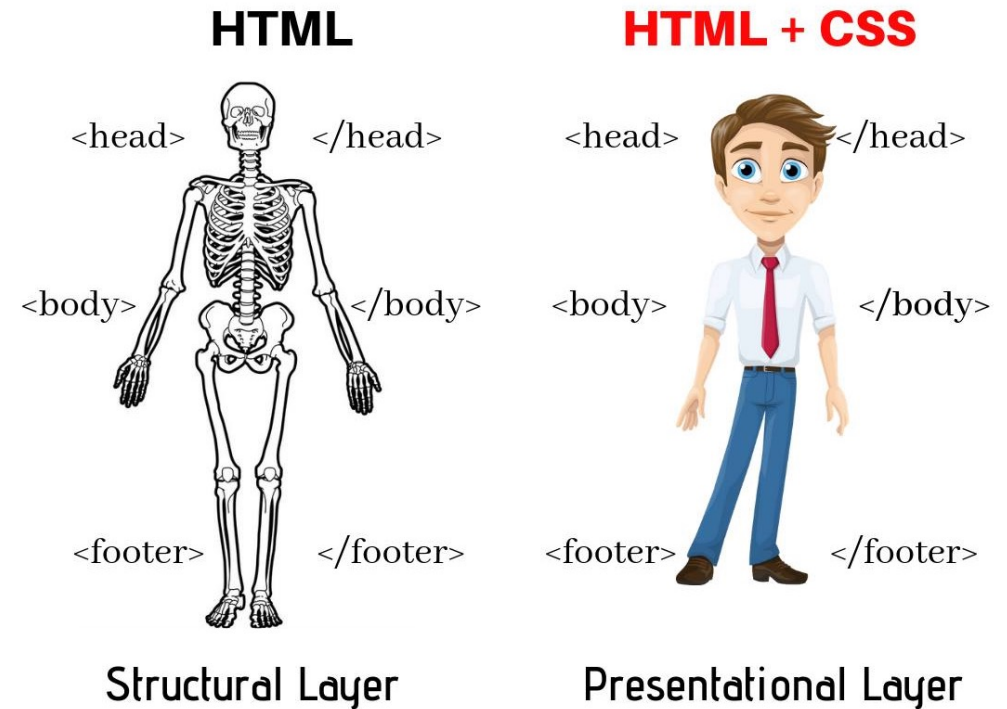
Advanced CSS

Basic CSS

Content vs. Presentation

- Most HTML tags define content type, independence of presentation.
 - exceptions?
 - e.g. `` `` for bold text and `<i>` `</i>` for italicized text
- Style sheets associate presentation formats with HTML elements.
 - CSS1: published in 1996
 - CSS2: published in 1998
 - CSS3: published in 1999

HTML Vs CSS



Content vs. Presentation (cont.)

- Style sheets can be used to specify how tables should be rendered, how lists should be presented, what colors should be used on the webpage, what fonts should be used and how big/small they are, etc.
- HTML style sheets are known as *Cascading Style Sheets*, since can be defined at three different levels
 1. *inline* style sheets apply to the content of a single HTML element
 2. *document* style sheets apply to the whole BODY of a document
 3. *external* style sheets can be linked and applied to numerous documents, might also specify how things should be presented on screen or in print lower-level style sheets can override higher-level style sheets
- User-defined style sheets can also be used to override the specifications of the webpage designer. These might be used, say, to make text larger (e.g. for visually-impaired users).

Inline Style Sheets

- Using the **style** attribute, you can specify presentation style for a *single* HTML element
- within tag, list sequence of **property:value** pairs separated by semi-colons

font-family:Courier,monospace

font-style:italic

font-weight:bold

font-size:12pt font-size:large font-size:larger

color:red color:#000080

background-color:white

text-decoration:underline

text-decoration:none

text-align:left text-align:center

text-align:right text-align:justify

vertical-align:top vertical-align:middle

vertical-align:bottom

text-indent:5em text-indent:0.2in

```
<html>
<!-- CS443 page17.html 17.10.14 -->

<head>
  <title>Inline Style Sheets</title>
</head>

<body>
  <p style="font-family:Arial,sans-serif;
           text-align:right">This is a
right-justified paragraph in a sans serif
font (preferably Arial), with some
<span style="color:green">green text</span>.
</p>

  <p>And <a style="color:red;
                text-decoration:none;
                font-size:larger;"
            href="page01.html">here</a>
    is a formatted link.
  </p>
</body>
</html>
```

view page

Inline Style Sheets (cont.)

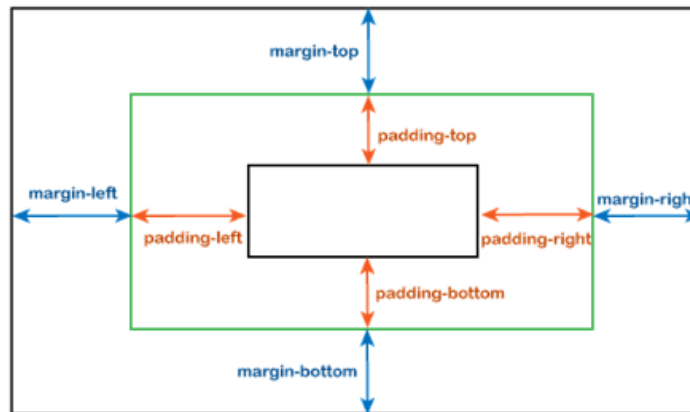
•more style properties & values

margin-left:0.1in margin-right:5%
margin:3em
padding-top:0.1in padding-bottom:5%
padding:3em

border-width:thin border-width:thick
border-width:5
border-color:red
border-style:dashed border-style:dotted
border-style:double border-style:none

whitespace:pre

list-style-type:square
list-style-type:decimal
list-style-type:lower-alpha
list-style-type:upper-roman



```
<html>
<!-- CS443 page18.html 17.09.09 -->

<head>
  <title>Inline Style Sheets</title>
</head>

<body>
  <p>Here is an image
    

    embedded in text.
  </p>

  <ol style="list-style-type:upper-alpha">
    <li> one thing</li>
    <li> or another</li>
    <ul style="list-style-type:square;
              whitespace:pre">
      <li> with this</li>
      <li> or that</li>
    </ul>
  </ol>
</body>
</html>
```

view page

Inline Style Sheets (cont.)

- style sheets can be applied to tables for interesting effects

```
<html>
<!-- CS443 page19.html 17.10.14 -->

<head>
  <title> Inline Style Sheets </title>
</head>

<body>
  <table style="font-family:Arial,sans-serif">
    <caption style="color:red;
                  font-style:italic;
                  text-decoration:underline">
      Student data. </caption>
    <tr style="background-color:red">
      <th> name </th> <th> age </th>
    </tr>
    <tr>
      <td> Chris Smith </td> <td> 19 </td>
    </tr>
    <tr>
      <td> Pat Jones </td> <td> 20 </td>
    </tr>
    <tr>
      <td> Doogie Howser </td> <td> 9 </td>
    </tr>
  </table>
</body>
</html>
```

[view page](#)

Document Style Sheets

- Inline style sheets apply to individual elements in the page.
 - using inline style directives can lead to inconsistencies, as similar elements are formatted differently
 - e.g., we might like for all `<h1>` elements to be centered
 - inline definitions mix content & presentation
 - violates the general philosophy of HTML
- As a general rule, inline style sheet directives should be used as sparingly as possible
- Alternatively, document style sheets allow for a cleaner separation of content and presentation.
 - style definitions are placed in the `<head>` of the page (within `STYLE` tags)
 - can apply to all elements, or a subclass of elements, throughout the page.

Document Style Sheets

- Document style sheets ensure that similar elements are formatted similarly
 - can even define **subclasses** of elements and specify formatting
 - p.indented** defines subclass of paragraphs
 - inherits all defaults of <p>
 - adds new features

to specify this newly defined class, place class="**ID**" attribute in tag

```
<html>
<!-- CS443 page20.html 17.10.14 -->

<head>
  <title>Document Style Sheets</title>
  <style type="text/css">
    h1 {color:blue;
        text-align:center}
    p.indented {text-indent:0.2in}
  </style>
</head>

<body>
  <h1> Centered Title </h1>

  <p class="indented">This paragraph will have
the first line indented, but subsequent lines
will be flush. </p>

  <p>This paragraph will not be indented.
</p>

  <h1> The End </h1>

</body>
</html>
```

[view page](#)

Document Style Sheets (cont.)

- Document style sheets are especially useful in formatting tables
- Effectively separates content from presentation
- what if you wanted to right-justify the column of numbers?
- what if you changed your mind?

```
<html>
<!-- CS443 page21.html 17.10.14 -->

<head>
  <title> Inline Style Sheets </title>
  <style type="text/css">
    table {font-family:Arial,sans-serif}
    caption {color:red;
              font-style:italic;
              text-decoration:underline}
    th {background-color:red}
  </style>
</head>

<body>
  <table>
    <caption> Student data. </caption>
    <tr><th> name </th>          <th> age</th></tr>
    <tr><td> Chris Smith </td>    <td> 19 </td></tr>
    <tr><td> Pat Jones </td>      <td> 20 </td></tr>
    <tr><td> Doogie Howser </td> <td> 9 </td></tr>
  </table>
</body>
</html>
```

[view page](#)

Pseudo-Class

- Pseudo-class is used to define a special state of an element.
 - Style an element when users mouses over it
 - Style visited and unvisited links differently
 - Style an element when it gets focus

```
<html>
<!-- CS443 page23.html 17.10.14 -->

<head>
  <title>Title for Page</title>
  <style type="text/css">
    a {color : red;
      text-decoration : none;
      font-size : larger}
    a:visited {color : black}
    a:active {color : orange}
    a:hover {color : blue}
    p::first-letter {font-size : large;
                     color : white;
                     background-color : darkblue}

  </style>
</head>

<body>
  <p> Welcome to my Web page.  I am so
happy you are here.
</p>
  <p> Be sure to visit
  <a href="http://www.cnn.com">CNN</a>
for late-breaking news.
</p>
</body>
</html>
```

view page

Pseudo-Element

- Pseudo-element is used to style specified parts of an element.
 - Style the first letter, or line, of an element
 - Insert content before, or after, the content of an element

```
<html>
<!-- CS443 page23.html 17.10.14 -->

<head>
  <title>Title for Page</title>
  <style type="text/css">
    a {color : red;
      text-decoration : none;
      font-size : larger}
    a:visited {color : black}
    a:active {color : orange}
    a:hover {color : blue}
    p::first-letter {font-size : large;
                     color : white;
                     background-color : darkblue}
  </style>
</head>

<body>
  <p> Welcome to my Web page.  I am so
happy you are here.
</p>
  <p> Be sure to visit
  <a href="http://www.cnn.com">CNN</a>
for late-breaking news.
</p>
</body>
</html>
```

[view page](#)

External Style Sheets

- modularity is key to the development and reuse of software
 - design/implement/test useful routines and classes
 - package and make available for reuse
 - saves in development cost & time
 - central libraries make it possible to make a single change and propagate the changes
- external style sheets place the style definitions in separate files
 - multiple pages can link to the same style sheet, consistent look across a site
 - possible to make a single change and propagate automatically
 - represents the ultimate in content/representation separation

Modularity & Style Sheets

- Ideally, the developer(s) of a Web site would place all formatting options in an external style sheet.
- All Web pages link to that same style sheet for a uniform look.
 - simplifies Web pages since only need to specify structure/content tags
 - Note: no <style> tags are used in the external style sheet

```
/* myStyle.css  CS443 02.09.05 */
```

```
h1 {color : blue; text-align : center}  
p.indented {text-indent:0.2in}
```

```
<html>  
<!-- CS443 page26.html 17.10.14 -->  
  
<head>  
  <title>Title for Page</title>  
  <link rel="stylesheet"  
        type="text/css"  
        href="myStyle.css"  
        title="myStyle" />  
</head>  
  
<body>  
  <h1>Centered Title</h1>  
  
  <p class="indented">This paragraph will  
have the first line indented, but subsequent  
lines will be flush.</p>  
  
  <p>This paragraph will not be indented.  
</p>  
  
  <h1>The End</h1>  
  
</body>  
</html>
```

<div> and Tags

- Problem: font properties apply to whole elements, which are often too large
 - Solution: a new tag to define an element in the content of a larger element - ****
 - The default meaning of **** is to leave the content as it is (i.e. unchanged)

```
<p> Now is the <span> best time </span> ever! </p>
```

- Use **** to apply a document style sheet definition to its content

```
<style type = "text/css">  
  .bigred {font-size: 24pt;  
    font-family: Ariel; color: red}  
</style>  
...  
<p> Now is the <span class="bigred">  
  best time </span> ever!  
</p>
```

The **** tag is similar to other HTML tags, they can be nested and they have id and class attributes

view page

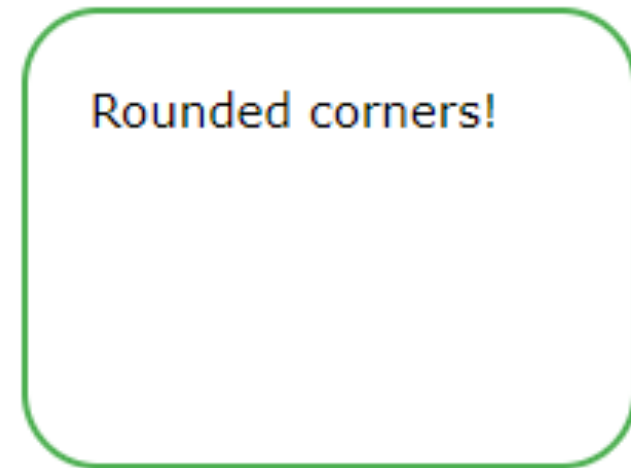
Another tag that is useful for style specifications: **<div>**
Used to create document sections (or divisions) for which style can be specified
e.g., a section of five paragraphs for which you want some particular style

Advanced CSS

Rounded Corners

- With the CSS border-radius property, you can give any element “rounded corners”.
 - Rounded corners for an element with a border:

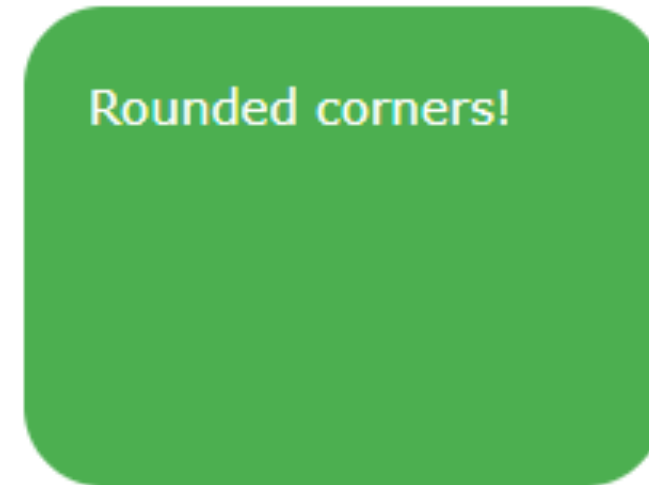
```
#rcorners2 {  
  border-radius: 25px;  
  border: 2px solid #73AD21;  
  padding: 20px;  
  width: 200px;  
  height: 150px;  
}
```



Rounded Corners

- With the CSS border-radius property, you can give any element “rounded corners”.
 - Rounded corners for an element with a specified background color:

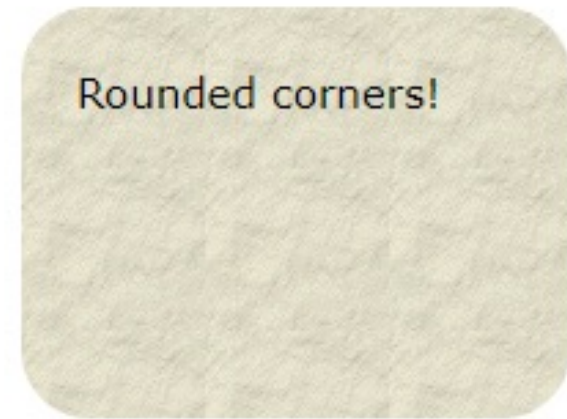
```
#rcorners1 {  
  border-radius: 25px;  
  background: #73AD21;  
  padding: 20px;  
  width: 200px;  
  height: 150px;  
}
```



Rounded Corners

- With the CSS border-radius property, you can give any element “rounded corners”.
 - If you only specify one keyword, the other value will be “center”

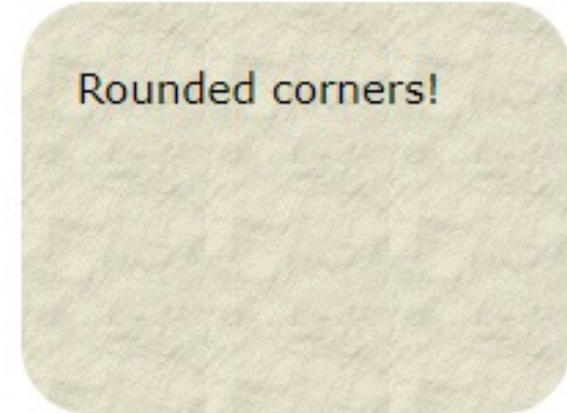
```
#rcorners3 {  
  border-radius: 25px;  
  background: url(paper.gif);  
  background-position: left top;  
  background-repeat: repeat;  
  padding: 20px;  
  width: 200px;  
  height: 150px;  
}
```



Rounded Corners

- With the CSS border-radius property, you can give any element “rounded corners”.
 - Rounded corners for an element with a background image:

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

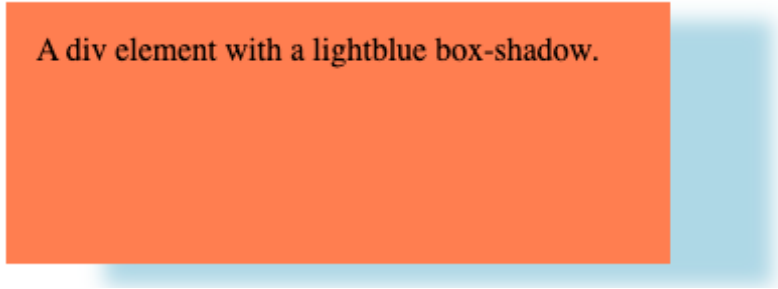


Shadows

- With CSS you can add shadow to text and to elements.
- Box Shadows: applies shadow to elements.

```
box-shadow: 50px 10px 10px 1px lightblue;
```

A div element with a lightblue box-shadow.



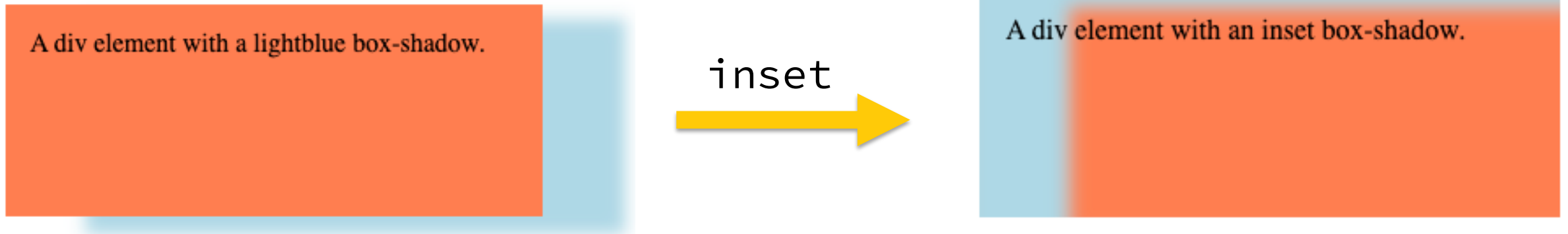
- The first value is **the horizontal offset** — how far the shadow is nudged to the right (or left if it's negative)
- The second value is **the vertical offset** — how far the shadow is nudged downwards (or upwards if it's negative)
- The third value is **the blur radius** — the higher the value the less sharp the shadow. ("0" being absolutely sharp). This is optional — omitting it is equivalent of setting "0".
- The fourth value is **the spread distance** — the higher the value, the larger the shadow ("0" being the inherited size of the box). This is also optional — omitting it is equivalent of setting "0".
- The fifth value is a **color**. That's optional, too.

Shadows

- Box Shadows: applies shadow to elements.

```
box-shadow: 50px 20px 10px 1px lightblue inset;
```

- apply shadows to the inside of a box by adding “inset” to the list



Shadows

- Text Shadows: applies shadow to text.

```
text-shadow: -5px 19px 2px red;
```

Text-shadow effect!

- The first value is the **horizontal offset**
- The second value is the **vertical offset**
- The third value is the **blur radius** (optional)
- The fourth value is the **color** (optional, although omitting this will make the shadow the same color as the text itself)

Universal, Child, and Adjacent Selectors

- Universal selectors: set global styles for a page, or as a descendant of a selector to set styles of everything within something.

```
* {  
    margin: 0;  
    padding: 0;  
}  
#contact * {  
    display: block;  
}
```

Example: set the margin and padding on everything in a page to zero and everything within an element with the ID “contact” to be displayed as a block

Universal, Child, and Adjacent Selectors

- Child selectors: A greater-than symbol (“>”) can be used to specify something that is a child of something else, that is, something immediately nested within something.

```
#genus_examples > li { border: 1px solid red }
```

Example: set the border for all child of element has id=“genus_examples”

Universal, Child, and Adjacent Selectors

- Adjacent selectors: A plus sign (“+”) is used to target an adjacent sibling of an element, essentially, something immediately following something.

```
<h1>Clouded leopards</h1>  
<p>Clouded leopards are cats that belong  
to the genus Neofelis.</p>  
<p>There are two extant species: Neofelis  
nebulosa and Neofelis diardi.</p>
```

```
h1 + p { font-weight: bold }
```

Only the first paragraph, that following the heading, will be made bold.

Advanced Colors

- We already know that colors can be defined by name, RGB, or hex values
- CSS 3 also allows you to paint away with HSL — hue, saturation, and lightness
- An HSL color value is specified with: `hsl(hue, saturation, lightness)`.
 - Hue is a degree on the color wheel (from 0 to 360):
 - 0 (or 360) is red
 - 120 is green
 - 240 is blue
 - Saturation is a percentage value: 100% is the full color.
 - 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%);`

CSS Transitions

- Transitions allow you to easily animate parts of your design without the need for the likes of JavaScript
 - transition-property: which property (or properties) will transition.
 - transition-duration: how long the transition takes.
 - transition-timing-function: if the transition takes place at a constant speed or if it accelerates and decelerates.
 - transition-delay: how long to wait until the transition takes place.

```
div {  
  width: 100px;  
  height: 100px;  
  background: red;  
  transition: width 2s;  
}
```

```
div:hover {  
  width: 500px;  
}
```



Backgrounds: Multiples, Size, and Origin

- Multiples background: CSS3 allows you to apply multiple background images to a single box by simply putting image locations in a comma-separated list

```
background-image: url(this.jpg), url(that.gif),  
url(theother.png);
```

```
#example1 {  
    background-image: url(img_flwr.gif), url(paper.gif);  
    background-repeat: repeat, repeat;  
}
```

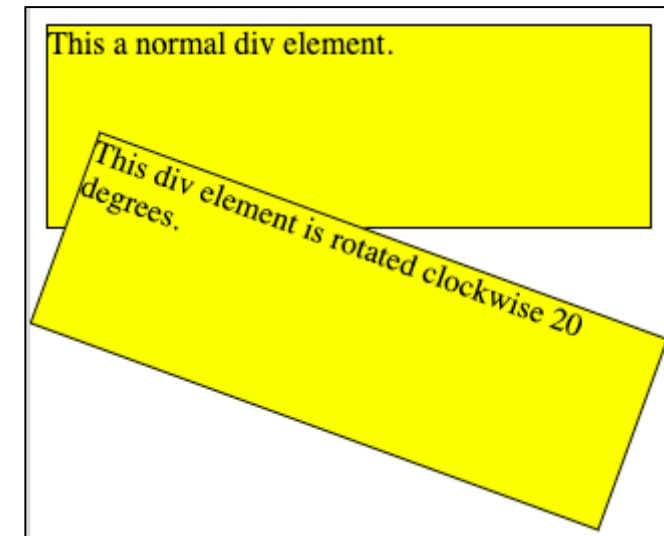


Transformations

- CSS transforms allow you to move, rotate, scale, and skew elements.
 - The **translate()** method moves an element from its current position (according to the parameters given for the X-axis and the Y-axis).
 - The **rotate()** method rotates an element clockwise or counter-clockwise according to a given degree.
 - The **scale()** method increases or decreases the size of an element (according to the parameters given for the width and height).
 - The **skew()** method skews an element along the X and Y-axis by the given angles.
 - The **matrix()** method combines all the 2D transform methods into one.

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

```
div#myDiv {  
  transform: rotate(20deg);  
} ...
```



Position

- specifies the type of positioning method used for an element
 - static: is not affected by the top, bottom, left, and right properties.
 - relative: is positioned relative to its normal position.
 - fixed: always stays in the same place even if the page is scrolled
 - absolute: is positioned relative to the nearest positioned ancestor (or document body if no ancestor exists)
 - sticky: is positioned based on the user's scroll position.

```
div.relative {  
  position: relative;  
  left: 20px;  
  width: 400px;  
  height: 200px;  
  border: 3px solid #73AD21;  
}
```

```
div.absolute {  
  position: absolute;  
  top: 80px;  
  right: 0;  
  width: 200px;  
  height: 100px;  
  border: 3px solid #73AD21;  
}
```

This div element has position: relative;

This div element has position:
absolute;

