



# CSS Overview

## Cascading Style Sheets

CSS

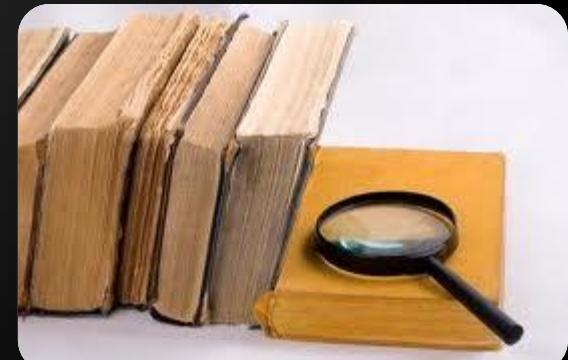
Telerik Software Academy  
<http://academy.telerik.com>

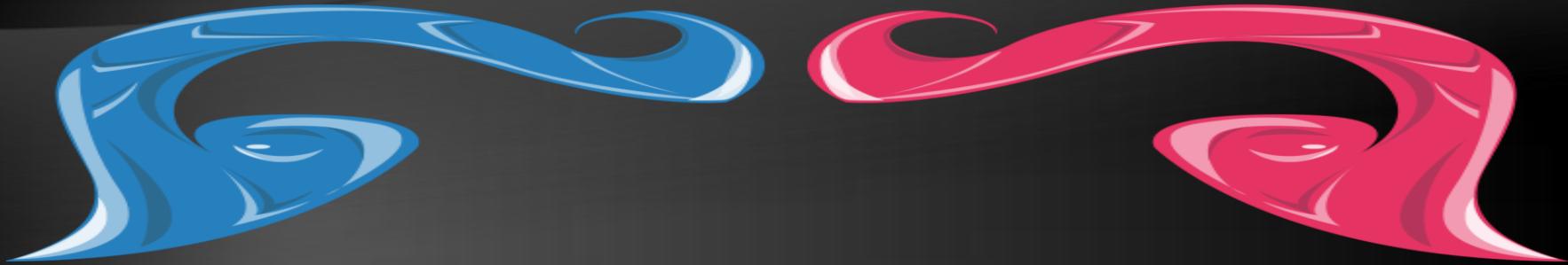
```
171 #content .article img.left.border {  
172     padding: 0 9px 9px 0;  
173     border-right: 1px dotted #999;  
174     border-bottom: 1px dotted #999; }  
175 #content .article blockquote {  
176     margin-left: 10px;  
177     padding-left: 10px;  
178     border-left: 3px solid #252525; }  
179 #content .article ul {  
180     padding-left: 1em;  
181     list-style-type: circle; }
```

**CONTENTS!**

# Table of Contents

- ◆ **What is CSS?**
- ◆ **Styling with Cascading Style Sheets (CSS)**
- ◆ **CSS Selectors**
  - ◆ **Select by element name, id or class**
  - ◆ **Nested Selectors**
- ◆ **Importing CSS into HTML**
- ◆ **Selectors**
  - ◆ **Attribute selectors**
  - ◆ **Pseudo Selectors**





# Cascading Style Sheets

## Separating Content from Presentation

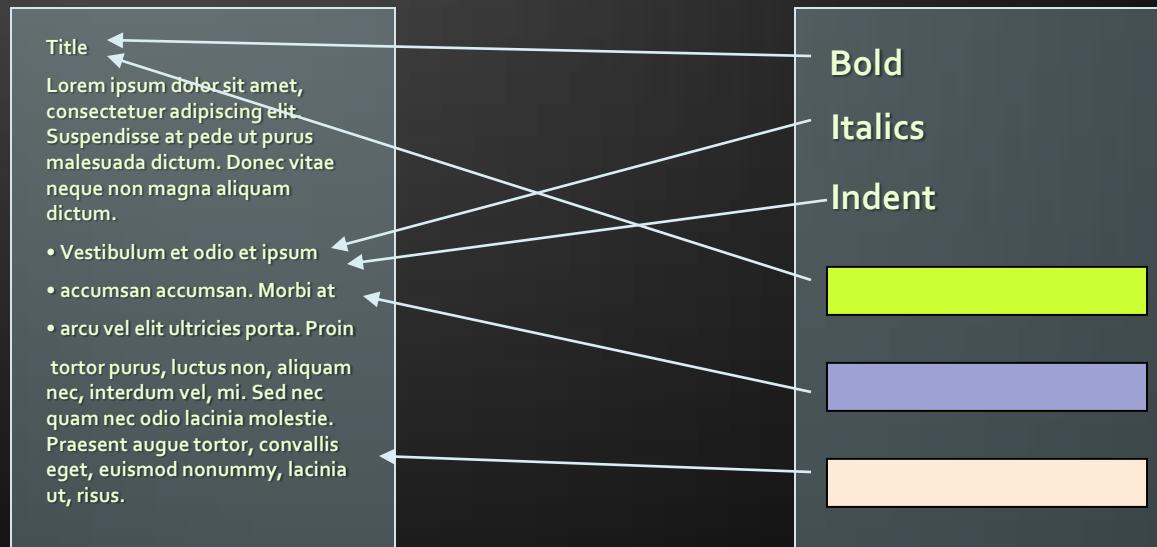


# CSS: A New Philosophy

- ◆ Separate content from presentation!

## Content (HTML document)

## Presentation (CSS Document)

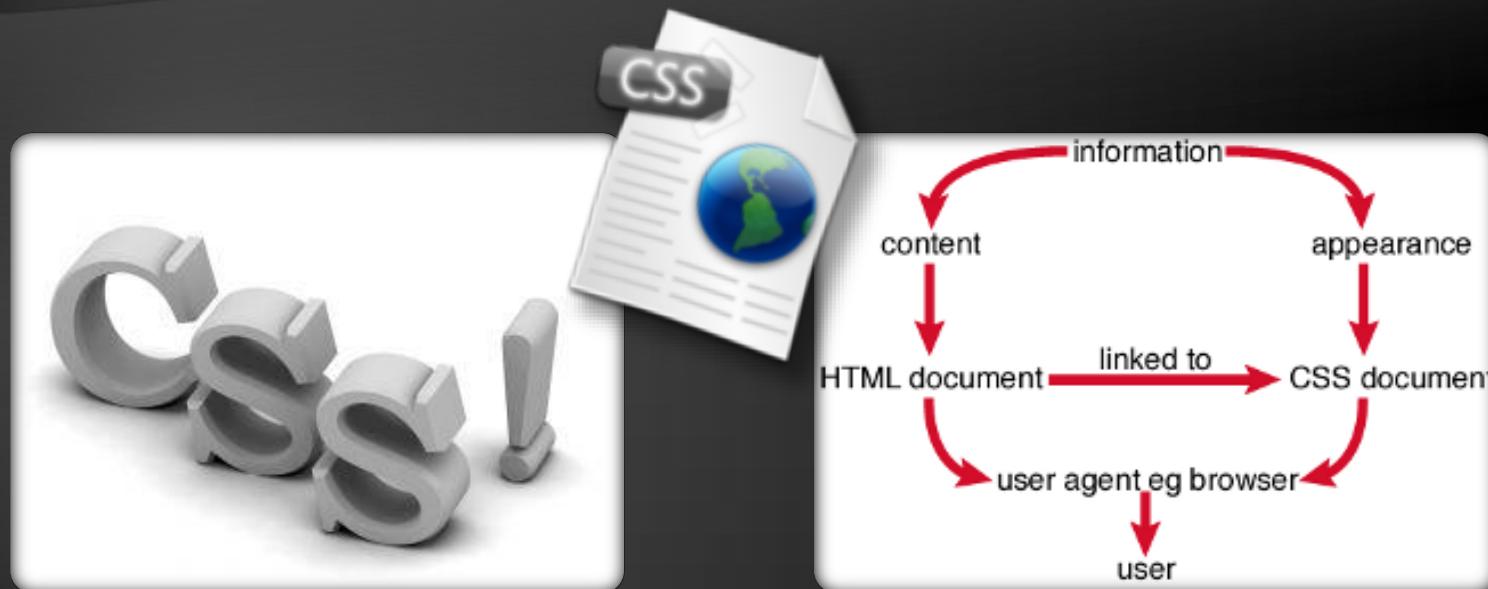


## Title

**Lorem ipsum dolor sit amet,  
  consectetuer adipiscing elit.  
  Suspendisse at pede ut purus  
  malesuada dictum. Donec vitae neque  
  non magna aliquam dictum.**

- *Vestibulum et odio et ipsum*
- *accumsan accumsan. Morbi at*
- *arcu vel elit ultricies porta. Proin*

**Tortor purus, luctus non, aliquam nec,  
interdum vel, mi. Sed nec quam nec  
odio lacinia molestie. Praesent augue  
tortor, convallis eget, euismod  
nonummy, lacinia ut, risus.**



# CSS Intro

## Styling with Cascading Stylesheets

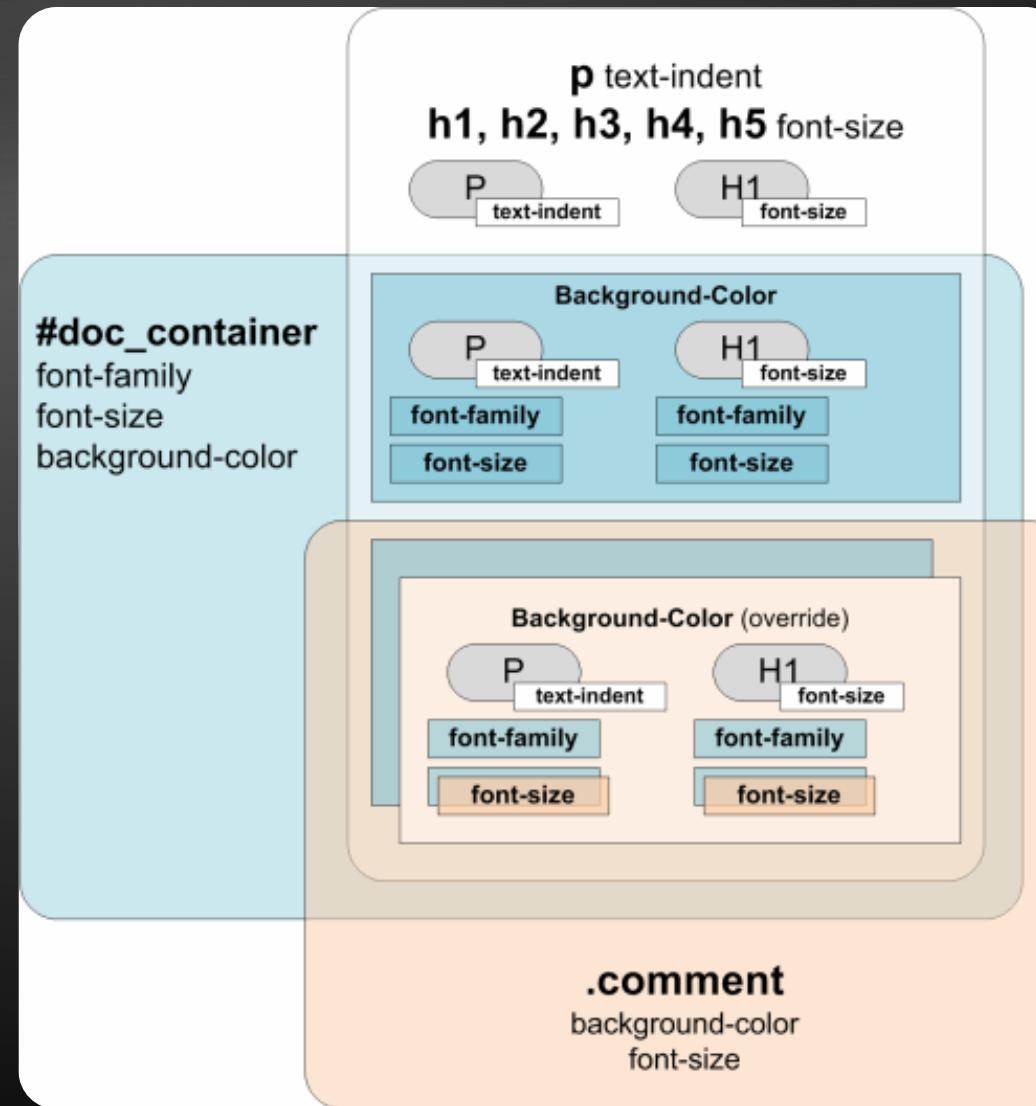
- ◆ Cascading Style Sheets (CSS)
  - ◆ Used to describe the presentation of documents
  - ◆ Define sizes, spacing, fonts, colors, layout, etc.
  - ◆ Improve content accessibility
  - ◆ Improve flexibility
- ◆ Designed to separate presentation from content
- ◆ Due to CSS, all HTML presentation tags and attributes are deprecated, e.g. font, center, etc.

- ◆ CSS can be applied to any XML document
  - ◆ Not just to HTML / XHTML
- ◆ CSS can specify different styles for different media
  - ◆ On-screen
  - ◆ In print
  - ◆ Handheld, projection, etc.
  - ◆ ... even by voice or Braille-based reader

# Why “Cascading”?

- ◆ Priority scheme determining which style rules apply to element
  - ◆ Cascade priorities or specificity (weight) are calculated and assigned to the rules
  - ◆ Child elements in the HTML DOM tree inherit styles from their parent
    - ◆ Can override them
    - ◆ Control via !important rule

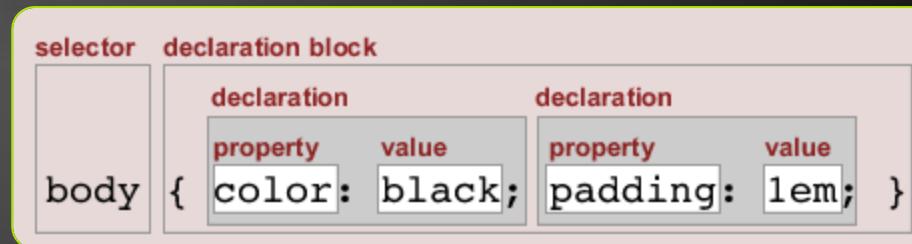
# Why "Cascading"? (2)



- ◆ Some CSS styles are inherited and some are not
  - ◆ Text-related and list-related properties are inherited: `color`, `font-size`, `font-family`, `line-height`, `text-align`, `list-style`, etc.
  - ◆ Box-related and positioning styles are not inherited: `width`, `height`, `border`, `margin`, `padding`, `position`, `float`, etc
  - ◆ `<a>` elements do not inherit color and text-decoration

# Style Sheets Syntax

- ◆ Stylesheets consist of rules, selectors, declarations, properties and values



<http://css.maxdesign.com.au>

- ◆ Selectors are separated by commas
- ◆ Declarations are separated by semicolons
- ◆ Properties and values are separated by colons

```
h1,h2,h3 { color: green; font-weight: bold; }
```

# Common Selectors

Select the Elements to Apply a Style

A hand holding a dark fountain pen is positioned above a teal-colored oval. Inside the oval, the words "CSS Selectors" are written in a white, sans-serif font.

CSS Selectors

- ◆ Selectors determine which element the rules apply to:
  - All elements of specific type (tag)
  - Those that match a specific attribute (id, class)
  - Elements may be matched depending on how they are nested in the document tree (HTML)
- ◆ Examples:

```
.header a { color: green }
```

```
#menu>li { padding-top: 8px }
```

- ◆ Three primary kinds of selectors:

- By tag (type selector):

```
h1 { font-family: verdana,sans-serif; }
```

- By element id:

```
#element_id { color: #ff0000; }
```

- By element class name (only for HTML):

```
.myClass {border: 1px solid red}
```

- ◆ Selectors can be combined with commas:

```
h1, .link, #top-link {font-weight: bold}
```

This will match `<h1>` tags, elements with class link, and the element with id top-link

- ◆ Match relative to element placement:

```
p a {text-decoration: underline}
```

This will match all `<a>` tags that are inside of `<p>`

- ◆ \* – universal selector (avoid or use with care!):

```
p * {color: black}
```

This will match all descendants of `<p>` element

- ◆ + selector – used to match “next sibling”:

```
img + .link {float:right}
```

This will match all siblings with class name link that appear immediately after `<img>` tag

# Nested Selectors (2)

- ◆ > selector – matches direct child nodes:

```
p > .error {font-size: 8px}
```

This will match all elements with class error, direct children of <p> tag

- ◆ .class1.class2 (no space!)
  - ◆ Matches elements with both (all) classes applied at the same time

```
p.post-text.special {font-weight: bold}
```

# Common Selectors

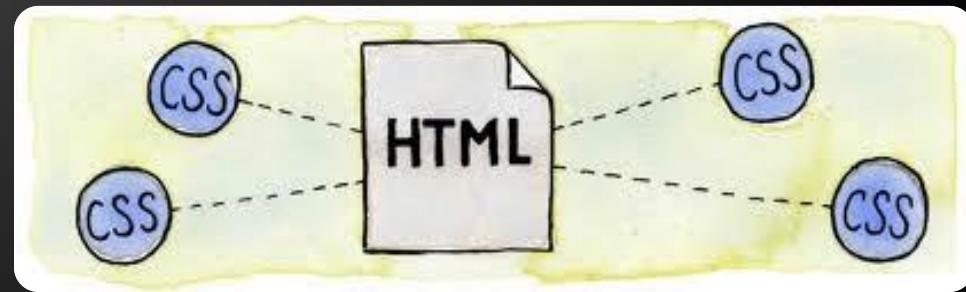
## Live Demo





# Importing CSS Into HTML

How to Use CSS with HTML?



# Importing CSS Into HTML

- ◆ CSS (presentation) can be imported in HTML (content) in three ways:
  - ◆ Inline: the CSS rules in the `style` attribute
    - ◆ No selectors are needed
  - ◆ Embedded: in the `<head>` in a `<style>` tag
  - ◆ External: CSS rules in separate file (best)
    - ◆ Usually a file with `.css` extension
    - ◆ Linked via `<link rel="stylesheet" href="...">` tag
    - ◆ Via `@import` directive in embedded CSS block

# Linking HTML and CSS (2)

- ◆ Using external CSS files is highly recommended
  - Simplifies the HTML document
  - Improves page load speed (CSS file is cached)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//>
<http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
      lang="en">
    <head>
        <title>Fullsize Test</title>
        <script type="text/javascript" src="fullsize.js">
        <script src="jquery.js" type="text/javascript">
        <script src="jquery.fullsize.js" type="text/javascript">
        <script src="fullsize/jquery.zoomit.js" type="text/javascript">
        </script>
    </head>
    <body>
        <h1>Fullsize Test</h1>
        <h2>The standard Lorem ipsum passage, used since the 1500s</h2>
        <p>
            <img alt="Test Image" href="7798-lqg.jpg" title="1 like
This image" style="border:none;right:10px;" />
              
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed et eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
        <h2>The standard Lorem ipsum passage, used since the 1500s</h2>
        <p>
            <img alt="Test Again" href="7798-lqg.jpg" style="float:left; margin-right:10px; margin-top:-20px;" />
        </p>
        <br/>
        <div class="button">Button</div>
    </body>
</html>
```

**HTML**

HTML links to  
external CSS file

```
.fullsize-icon {
    position: absolute;
    right: 0;
    padding: 0;
    width: 30px;
    height: 30px;
    background: transparent url(fullsize-icon.png) no-repeat left top;
    z-index: 999;
    cursor: url(fullsize.cur), auto;
}

.fullsize-loading, .fullsize-wrapper {
    position: absolute;
    margin: 0;
    padding: 0;
    z-index: 999;
}

.fullsize-loading {
    height: 30px;
    width: 30px;
    background: transparent url(fullsize-loading-lqg.png) no-repeat left top;
}

.fullsize-loading-inner {
    height: 100%;
    width: 100%;
    background: transparent url(fullsize-loading-spinner.gif) no-repeat center center;
}

.fullsize-image {
    display: block;
}

.fullsize-title {
    position: relative;
    width: 100%;
    margin: 0;
}
```

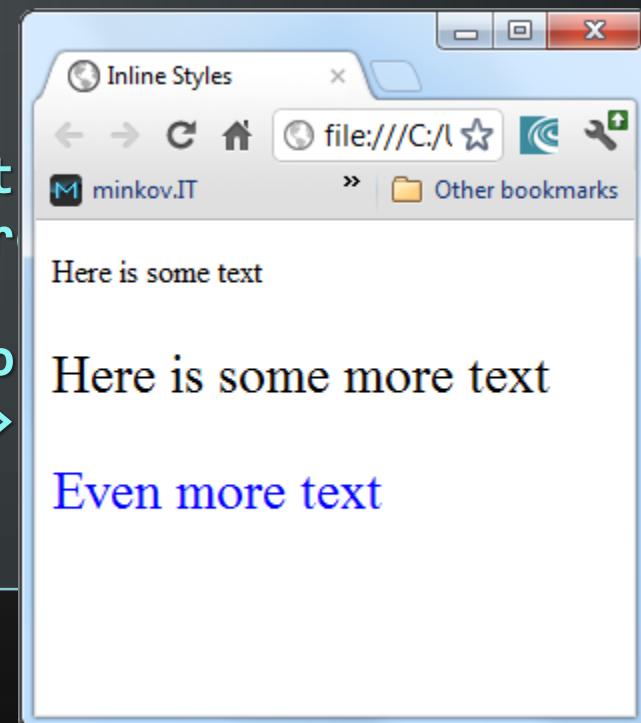
**CSS**

# Inline Styles: Example

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Inline Styles</title>
</head>
<body>
    <p>Here is some text</p>
<!--Separate multiple styles with a semicolon-->
    <p style="font-size: 20pt">Here is some
        more text</p>
    <p style="font-size: 20pt; color:
        #0000FF" >Even more text</p>
</body>
</html>
```

# Inline Styles: Example

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Inline Styles</title>
</head>
<body>
    <p>Here is some text</p>
    <!--Separate multiple styles with a comment-->
    <p style="font-size: 20pt">Her
        more text</p>
    <p style="font-size: 20pt; color: #0000FF">Even more text</p>
</body>
</html>
```



- ◆ Embedded in the HTML in the `<style>` tag:

```
<style type="text/css">
```

- The `<style>` tag is placed in the `<head>` section of the document
- type attribute specifies the MIME type
  - MIME describes the format of the content
  - Other MIME types include `text/html`, `image/gif`, `text/javascript` ...
  - Not required in HTML5
- Used for document-specific styles

# Embedded Styles: Example

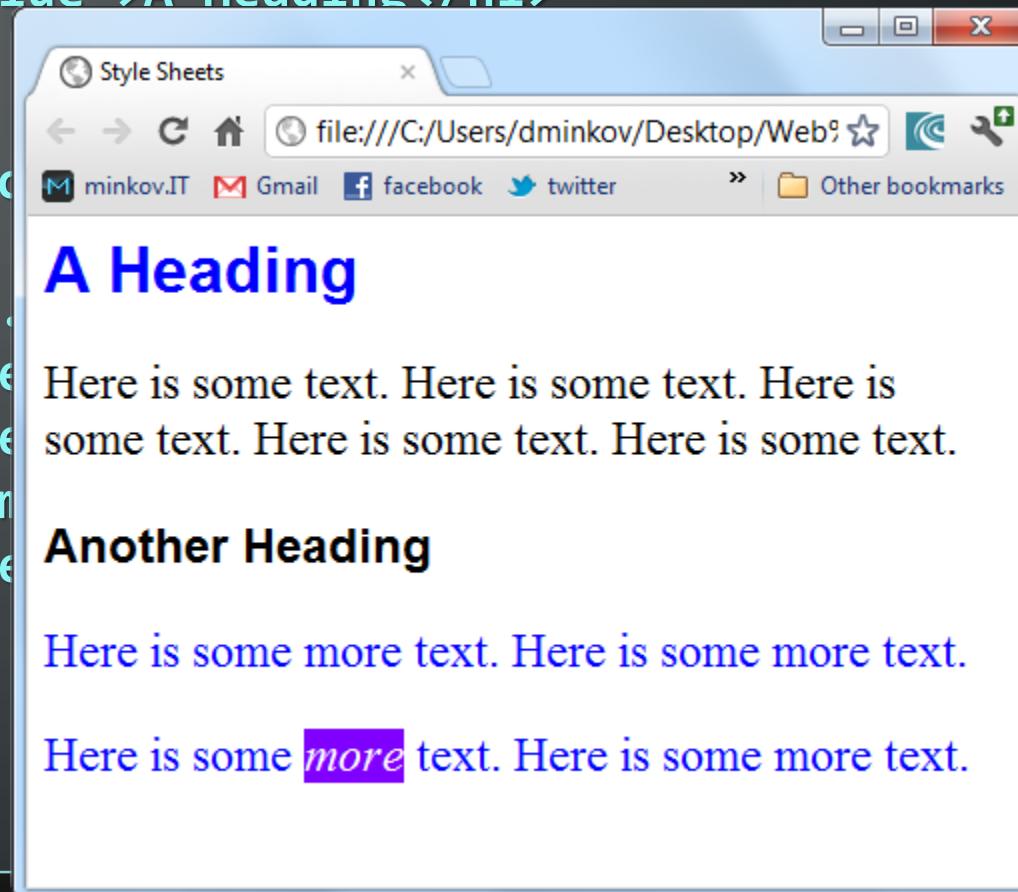
```
<!DOCTYPE html>
<html>
<head>
    <title>Style Sheets</title>
    <style type="text/css">
        em {background-color:#8000FF; color:white}
        h1 {font-family:Arial, sans-serif}
        p {font-size:18pt}
        .blue {color:blue}
    </style>
</head>
```

# Embedded Styles: Example (2)

```
...
<body>
  <header>
    <h1 class="blue">A Heading</h1>
  </header>
  <article>
    <p>Here is some text. Here is some text.
      Here is some text. Here is some text. Here
      is some text.</p>
    <h1>Another Heading</h1>
    <p class="blue">Here is some more text.
      Here is some more text.</p>
    <p class="blue">Here is some <em>more</em>
      text. Here is some more text.</p>
  </article>
</body>
</html>
```

# Embedded Styles: Example (3)

```
...
<body>
  <header>
    <h1 class="blue">A Heading</h1>
  </header>
  <article>
    <p>Here is some text.
      Here is some text.
      is some text.
    <h1>Another Heading</h1>
    <p class="blue">
      Here is some more text.
    <p class="blue">
      text. Here is some more text.
    </article>
  </body>
</html>
```



- ◆ External linking
  - ◆ Separate pages can all use a shared style sheet
  - ◆ Only modify a single file to change the styles across your entire Web site (see [www.csszengarden.com](http://www.csszengarden.com))
- ◆ link tag (with a rel attribute)
  - ◆ Specifies a relationship between current document and another document

```
<link rel="stylesheet" type="text/css"  
      href="styles.css">
```

- ◆ link elements should be in the <head>

## @import

- Another way to link external CSS files
- Example:

```
<style type="text/css">
  @import url("styles.css");
  /* same as */
  @import "styles.css";
</style>
```

- Ancient browsers do not recognize `@import`
- Use `@import` in an external CSS file to workaround the IE CSS file limit of 31 files

# External Styles: Example

```
/* CSS Document */

a { text-decoration: none }

a:hover { text-decoration: underline;
           color: red;
           background-color: #CCFFCC }

li em { color: red;
         font-weight: bold }

ul { margin-left: 2cm }

ul ul { text-decoration: underline;
          margin-left: .5cm }
```

# External Styles: Example (2)

```
<!DOCTYPE html>
<html>
<head>
    <title>Importing style sheets</title>
    <link type="text/css" rel="stylesheet"
        href="styles.css"  />
</head>
<body>
    <h1>Shopping list for <em>Monday</em>:</h1>
    <li>Milk</li>
    ...

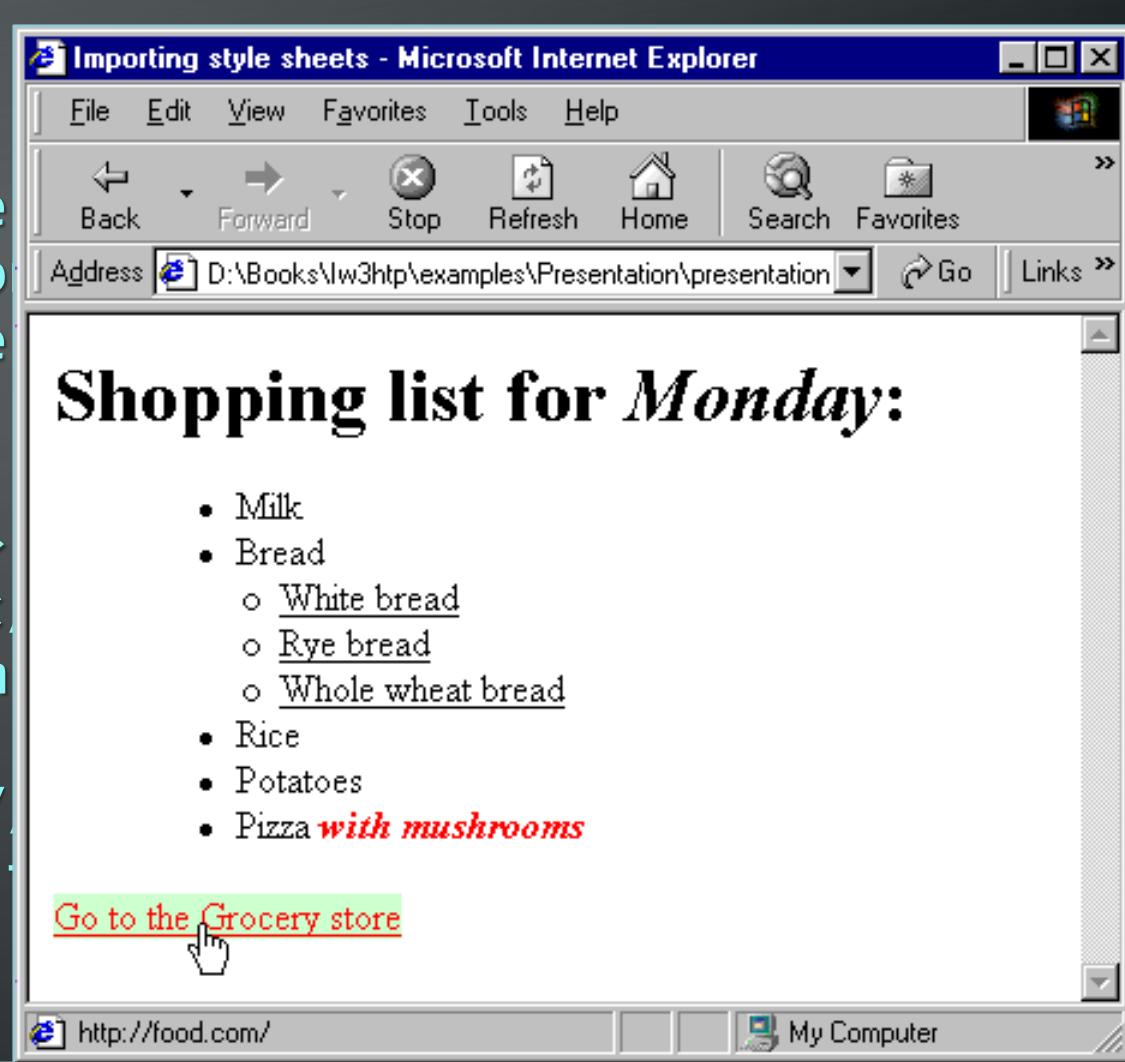
```

# External Styles: Example (3)

```
...
<li>Bread
  <ul>
    <li>White bread</li>
    <li>Rye bread</li>
    <li>Whole wheat bread</li>
  </ul>
</li>
<li>Rice</li>
<li>Potatoes</li>
<li>Pizza <em>with mushrooms</em></li>
</ul>
<a href="http://food.com" title="grocery
  store">Go to the Grocery store</a>
</body>
</html>
```

# External Styles: Example (4)

```
...
<li>Bread
  <ul>
    <li>White
    <li>Rye b
    <li>Whole
  </ul>
</li>
<li>Rice</li>
<li>Potatoes<
<li>Pizza <em>
</ul>
<a href="http://
  store">Go to
</body>
</html>
```



# Attribute Selectors

Picking Elements with Certain Attributes



- ◆ [ ] selects elements based on attributes
  - Element with a given attribute

```
a[title] {color:black}
```
  - Elements with a concrete attribute value

```
input[type=text] { font-family:Consolas}
```
  - Selects <input> elements with type=text
  - Elements whose attribute values contain a word

```
a[title*=logo] {border: none}
```
  - Selects <a> elements whose title attribute value contains logo

# Attribute Selectors

Live Demo



[Flickr.com/l.Gouss](https://flickr.com/l.Gouss)

# Pseudo Selectors

Relative to Element Content or State

```
html, body, div { padding: 0; margin: 0; font-size: 100%; }

/* Start of "Micro clearfix" */

.cf { zoom: 1; }
.cf:before,
.cf:after { content: ""; display: table; }
.cf:after { clear: both; }
```

## :before and :after

```
</head>
<body>
  <h1>Micro clearfix demo</h1>
  <div class="container cf">
    <div class="section">Float</div>
    <div class="section">Float</div>
  </div>
```

```
</body>
<style>
  .cf::before, .cf::after { content: " "; display: block; overflow: hidden; width: 100%; }
  .cf::before, .cf::after { border-top: 1px solid black; }
  .cf::before { border-bottom: 1px solid black; }
  .cf::after { border-bottom: 1px solid black; }
  .cf { width: 100%; }
  .cf .section { float: left; width: 45%; }
  .cf .section { width: 48%; }
  .cf .section { width: 48%; }
  .cf .section { width: 48%; }
</style>
```

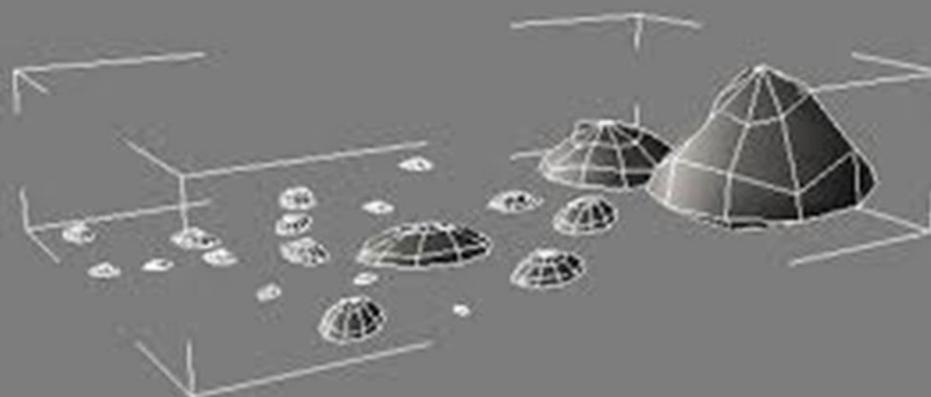
# Common Pseudo Selectors

- ◆ Pseudo-classes define state
  - ◆ **:hover, :visited, :active, :lang**
- ◆ Pseudo-elements define element "parts" or are used to generate content
  - ◆ **:first-line, :before, :after**

```
a:hover { color: red; }  
p:first-line { text-transform: uppercase; }  
.title:before { content: "»"; }  
.title:after { content: "«"; }
```

# Common Pseudo Selectors

Live Demo



# Structural Pseudo-classes

- ◆ **:root**
  - ◆ The root of the document
- ◆ **E:nth-child(n)**
  - ◆ An E element, the n-th child of its parent
- ◆ **E:nth-last-child(n)**
  - ◆ An E element, the n-th child of its parent, counting from the last on
- ◆ **E:nth-of-type(n)**
  - ◆ An E element, the n-th sibling of its type

# Structural Pseudo-classes (2)

- ◆ E:nth-last-of-type(n)
  - An E element, the n-th sibling of its type, counting from the last one
- ◆ E:last-child
  - An E element, last child of its parent
- ◆ E:first-of-type
  - An E element, first sibling of its type
- ◆ E:last-of-type
  - An E element, last sibling of its type

# Structural Pseudo-classes (3)

- ◆ E:**only-child**
  - ◆ An E element, only child of its parent
- ◆ E:**only-of-type**
  - ◆ An E element, only sibling of its type
- ◆ E:**empty**
  - ◆ An E element that has no children (including text nodes)
- ◆ More detailed descriptions:

<http://www.w3.org/TR/css3-selectors/#structural-pseudos>



# Structural Selectors

Live Demo

# The UI Element States Pseudo-Classes

- ◆ E:enabled
  - A user interface element E which is enabled
- ◆ E:disabled
  - A user interface element E which is disabled
- ◆ E:checked
  - A user interface element E which is checked (for instance a radio-button or checkbox)
  - Currently supported only in Opera and IE10 !



# UI Selectors

Live Demo

# Other CSS 3 Selectors

- ◆ E:**target**
  - ◆ An E element being the target of the referring URI
- ◆ E:**not(s)**
  - ◆ An E element that does not match simple selector
- ◆ E ~ F
  - ◆ An F element preceded by an E element

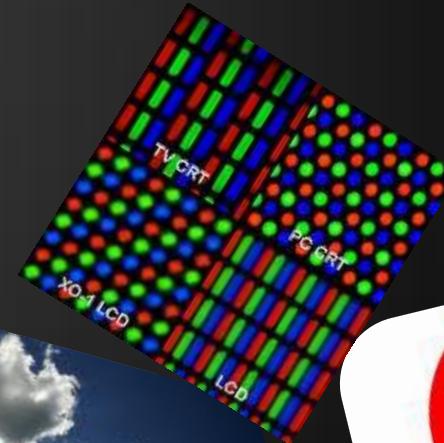
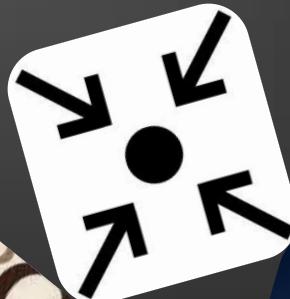


# Other CSS 3 Selectors

Live Demo

# CSS Values

## Types, Ranges, Units



- ◆ All values in CSS are strings
  - ◆ They can represent values that are not strings
  - ◆ I.e. 14px means size 14 pixels
- ◆ Colors are set in a red-green-blue format (RGB)
  - ◆ Both in hex and decimal

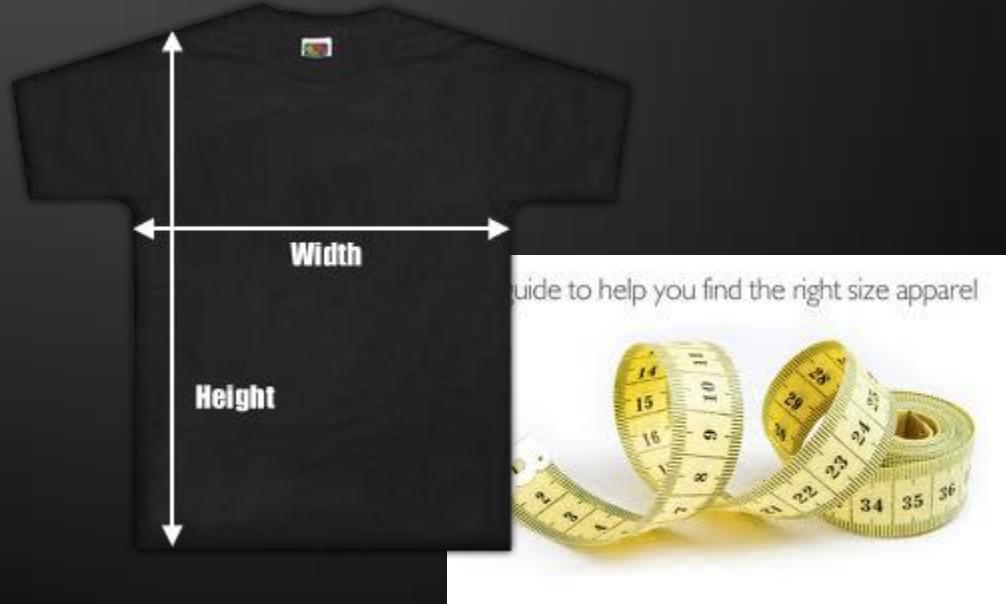
```
li.nav-item {  
    color: #44f1e1  
}
```

```
li.nav-item {  
    color: rgb(68, 241, 255)  
}
```

- ◆ When setting a size (width, height, font-size...) the values are given as numbers
  - ◆ Multiple formats / metrics may be used
  - ◆ Pixels, ems, e.g. `12px`, `1.4em`
  - ◆ Points, inches, centimeters, millimeters
    - ◆ E.g. `10pt`, `1in`, `1cm`, `1mm`
  - ◆ Percentages, e.g. `50%`
    - ◆ Of the size of the container/font size
  - ◆ Zero can be used with no unit: `border: 0;`

# Size Values

Live Demo



- ◆ Colors in CSS can be represented in few ways

- ◆ Using red-green-blue
  - ◆ Or red-green-blue-alpha

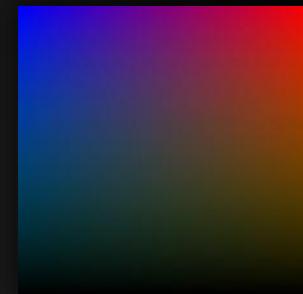
```
color: #f1a2ff  
color: rgb(241, 162, 255)  
color: rgba(241, 162, 255, 0.1)
```

The opacity values  
are from 0.0 to 1.0

- ◆ Using hue-saturation-light
  - ◆ Or hue-saturation-light-alpha

```
color: hsl(291, 85%, 89%);  
color: hsl(291, 85%, 89%, 0.1);
```

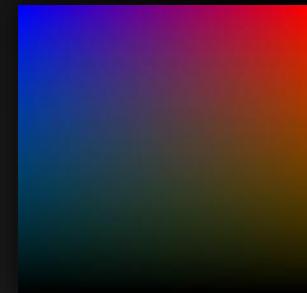
- ◆ RGB colors are defined with values for red, green and blue intensity
- ◆ Syntax:
  - ◆ #44fa36 – values are in hex
  - ◆ rgb(<red>, <green>, <blue>) – decimal values
- ◆ The range for red, green and blue is between integers 0 and 255



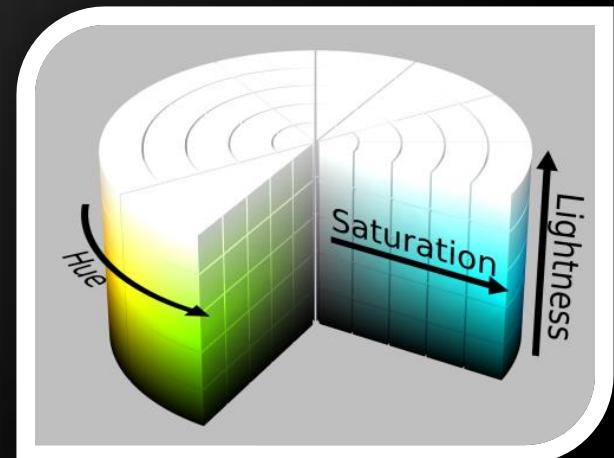
```
color: #07f2b3  
      <!- or -->  
color: rgb (7, 242, 179)
```



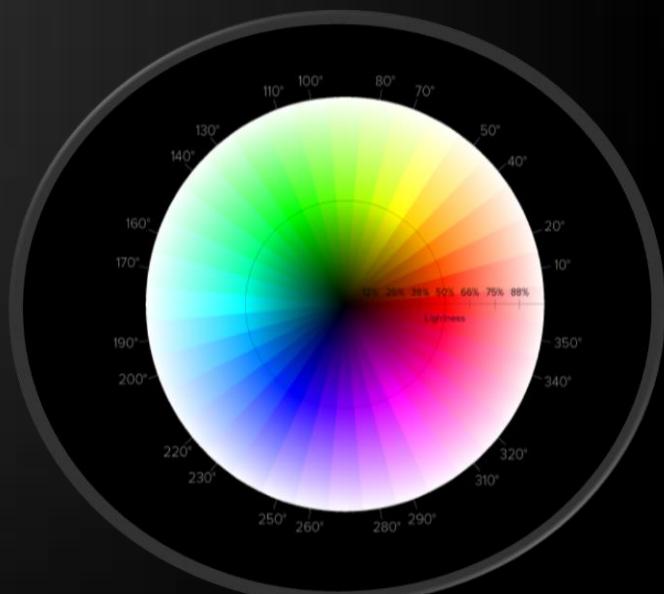
- ◆ Standard RGB colors with an opacity value for the color (alpha channel)
- ◆ Syntax: `rgba(<red>, <green>, <blue>, <alpha>)`
- ◆ The range for red, green and blue is between integers 0 and 255
- ◆ The range for the alpha channel is between 0.0 and 1.0
- ◆ Example: `rgba(255, 0, 0, 0.5)` ➔ 



- ◆ Hue is a degree on the color wheel
  - ◆ 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)
  - ◆ 100% is light (white)
  - ◆ 50% is the average



- ◆ HSLA allows a fourth value, which sets the Opacity (via the Alpha channel) of the element
- ◆ As RGBA is to RGB, HSLA is to HSL
- ◆ Supported in IE9+, Firefox 3+, Chrome, Safari, and in Opera 10+
- ◆ Example:
  - ◆ `hsla(0, 100%, 50%, 0.5)`
  - ◆ Result:



# Color Values

## Live Demo



# Default Browser Styles

Why Things Look Different on Different Browsers?



# Default Browser Styles

- ◆ **Browsers have predefined CSS styles**
  - ◆ Used when there is no CSS information or any other style information in the document
- ◆ **Caution: default styles differ in browsers**
  - ◆ E.g. margins, paddings and font sizes differ most often
  - ◆ Usually developers reset them

```
* { margin: 0; padding: 0; }
```

```
body, h1, p, ul, li { margin: 0; padding: 0; }
```

# CSS Cascade (Precedence)

- ◆ There are browser, user and author stylesheets with "normal" and "important" declarations
  - ◆ Browser styles (least priority)
  - ◆ Normal user styles
  - ◆ Normal author styles (external, in head, inline)
  - ◆ Important author styles
  - ◆ Important user styles (max priority)

```
a { color: red !important ; }
```

<http://www.slideshare.net/maxdesign/css-cascade-1658158>

- ◆ CSS specificity is used to determine the precedence (priority) of the CSS style declarations with the same origin
  - ◆ Simple calculation: #id = 100, .class = 10, :pseudo = 10, [attr] = 10, tag = 1, \* = 0
  - ◆ Same number of points? Order matters!
  - ◆ See also:
    - ◆ <http://www.smashingmagazine.com/2007/07/27/css-specificity-things-you-should-know/>
    - ◆ [http://css.maxdesign.com.au/selectutorial/advanced\\_conflict.htm](http://css.maxdesign.com.au/selectutorial/advanced_conflict.htm)

```
#homepage-image { background:transparent url(/images/kinetix-universal01.jpg) no-repeat center; }
#header-content { margin:30px 20px 30px 136px; width:726px; }
#header-inner { background:transparent url(/images/kinetixbeanslogo-black.gif) center; }
.kntx-page { color: #362819; }
h2.kntx-page { font-size: 20px; }
h3.kntx-page { font-size: 16px; }
.fineprint { font-size: 11px; color: #777; font-style: italic; }
```

# CSS Rules Precedence

Live Demo



- ◆ The CSS documentation at WebPlatform.org:
  - ◆ <http://docs.webplatform.org/wiki/css>
- ◆ CSS documentation at Mozilla
  - ◆ <https://developer.mozilla.org/en-US/docs/CSS>
- ◆ CSS3 tutorial
  - ◆ <http://www.w3schools.com/css3/>
- ◆ A list of all CSS 2.1 properties is available at  
<http://www.w3.org/TR/CSS2/propidx.html>

# Questions?

# Free Trainings @ Telerik Academy

- ◆ "Web Design with HTML 5, CSS 3 and JavaScript" course @ Telerik Academy



- ◆ [html5course.telerik.com](http://html5course.telerik.com)

- ◆ Telerik Software Academy

- ◆ [academy.telerik.com](http://academy.telerik.com)

Telerik Academy

- ◆ Telerik Academy @ Facebook

- ◆ [facebook.com/TelerikAcademy](https://www.facebook.com/TelerikAcademy)



- ◆ Telerik Software Academy Forums

- ◆ [forums.academy.telerik.com](http://forums.academy.telerik.com)

