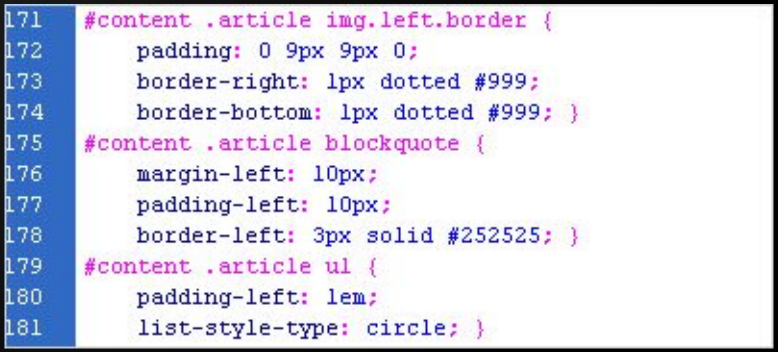
telerik

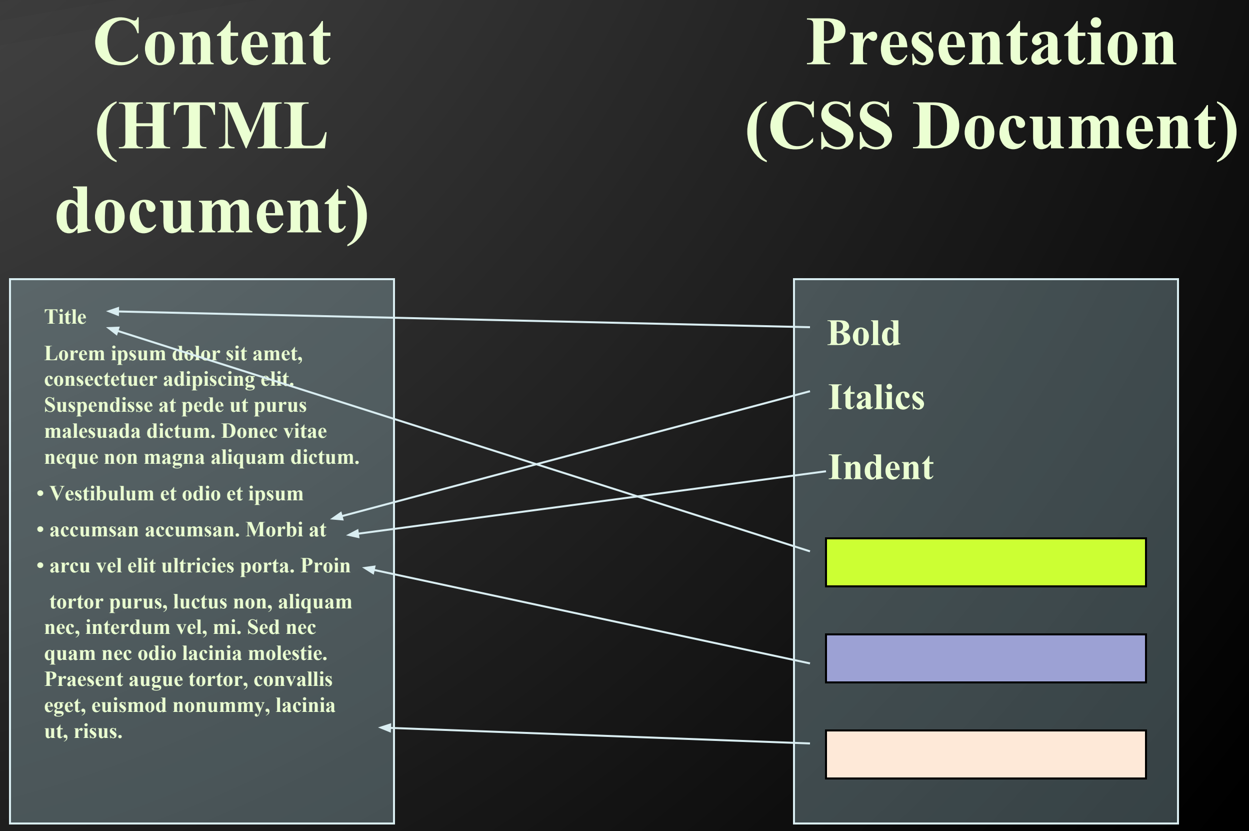
Cascading Style Sheets (CSS)



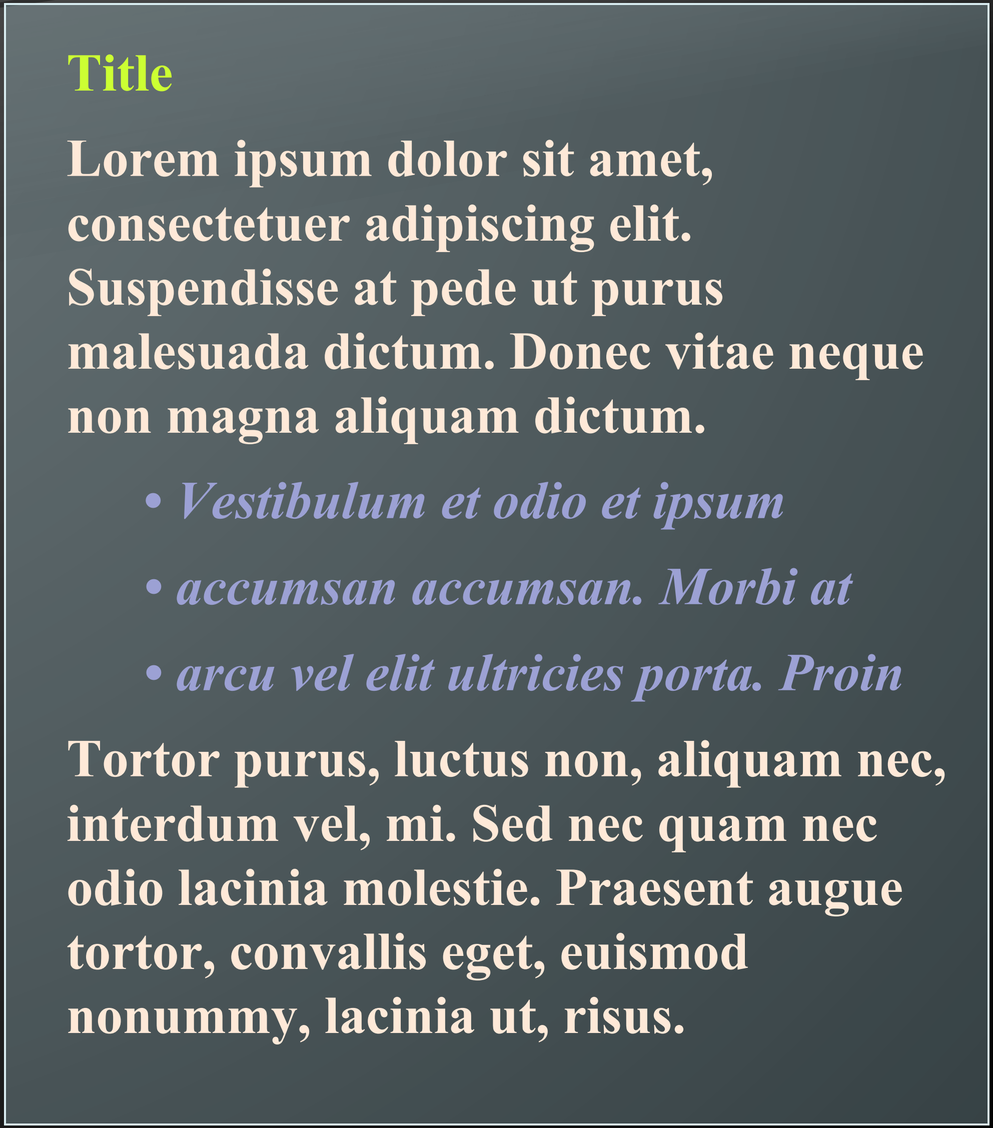
```
#content .article img.left.border {
    padding: 0 9px 9px 0;
    border-right: 1px dotted #999;
    border-bottom: 1px dotted #999; }
#content .article blockquote {
    margin-left: 10px;
    padding-left: 10px;
    border-left: 3px solid #252525; }
#content .article ul {
    padding-left: 1em;
    list-style-type: circle; }
```

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.
Bold
Italics
Indent

The Resulting Page

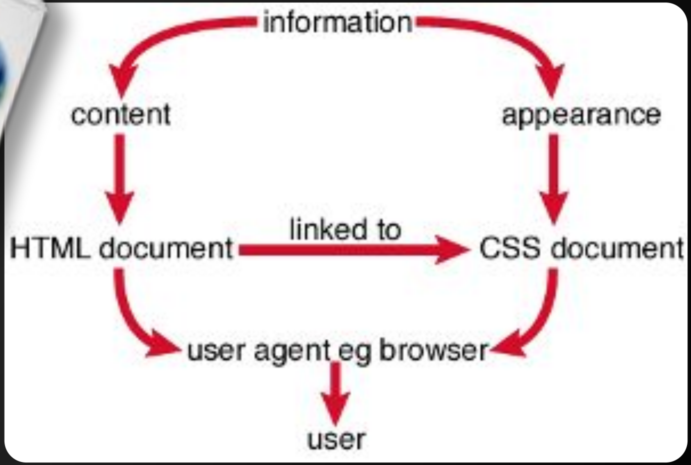


Title

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

CSS Introduction

- 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.

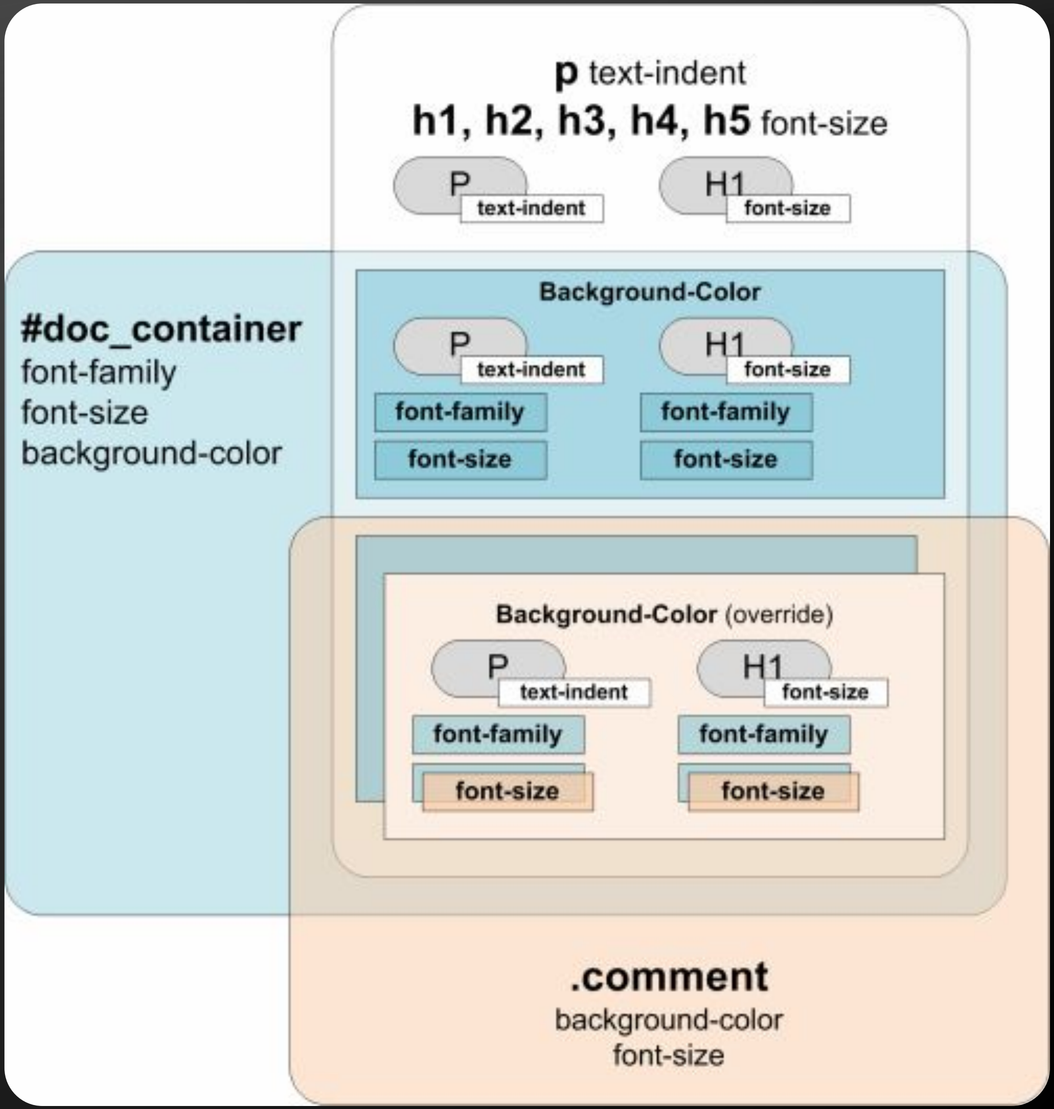
CSS Introduction (2)

- 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)

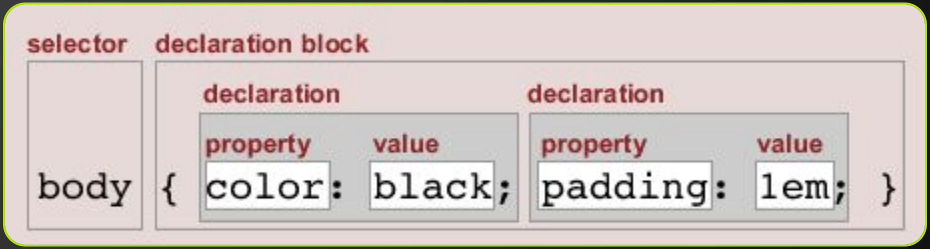


Why "Cascading"? (3)

- Some CSS styles are inherited and some 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



```
body { color: black; padding: 1em; }
```

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; }
```

Selectors

- Selectors determine which element the rule applies to:
  - All elements of specific type (tag)
  - Those that mach 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 }
```

Selectors (2)

- 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`,

telerik

Selectors (3)

- 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: "«"; }
```

Selectors (4)

- 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

Selectors (5)

- `>` selector – matches direct child nodes:

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

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

- `[ ]` – matches tag attributes by regular expression:

```
img[alt~=logo] {border: none}
```

  This will match all `<img>` tags with `alt` attribute containing the word `logo`

- `.class1.class2` (no space) - matches elements with

Values in the CSS Rules

- Colors are set in RGB format (decimal or hex):
  - Example: `#a0a6aa = rgb(160, 166, 170)`
  - Predefined color aliases exist: `black`, `blue`, etc.
- Numeric values are specified in:
  - Pixels, ems, e.g. `12px` , `1.4em`
  - Points, inches, centimeters, millimeters
    - E.g. `10pt` , `1in`, `1cm`, `1mm`
  - Percentages, e.g. `50%`
    - Percentage of what?...
  - Zero can be used with no unit: `border: 0;`

Default Browser Styles

- Browsers have default 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 and usually developers reset them

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

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

Linking HTML and CSS

- HTML (content) and CSS (presentation) can be linked 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 or `@import` directive in embedded CSS block

Linking HTML and CSS (2)

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

Inline Styles: Example

inline-styles.html

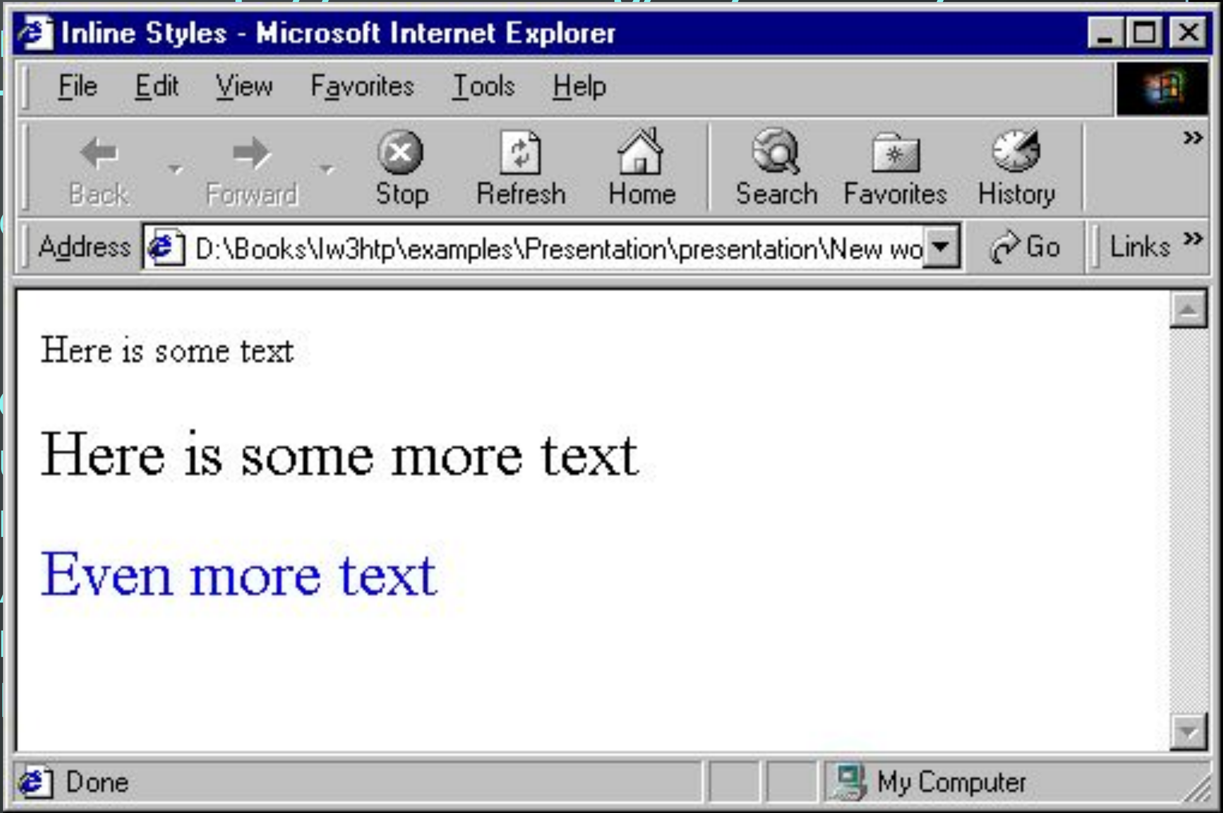
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/
DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Inline Styles</title>
</head>
<body>
  <p>Here is some text</p>
<!--Separate multiple styles with a semicolon-->
  <p style="font-size: 20pt">Here is some
    more text</p>
  <p style="font-size: 20pt;color:
    #0000FF" >Even more text</p>
</body>
</html>
```

telerik

Inline Styles: Example

inline-styles.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/
DTD/xhtml1-tra
<html xmlns="h
<head>
  <title>Inlin
</head>
<body>
  <p>Here is s
<!--Separate m
  <p style="fo
    more text<
  <p style="fo
    #0000FF" >
</body>
</html>
```


Inline Styles - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History
Address D:\Books\Iw3htp\examples\Presentation\presentation\New wo
Go Links
Done
My Computer

Here is some text

Here is some more text

Even more text

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

- CSS specificity is used to determine the precedence of CSS style declarations with the same origin. Selectors are what matters
  - 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/

Embedded Styles

- 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` …
- Used for document-specific styles

Embedded Styles: Example

embedded-stylesheets.html

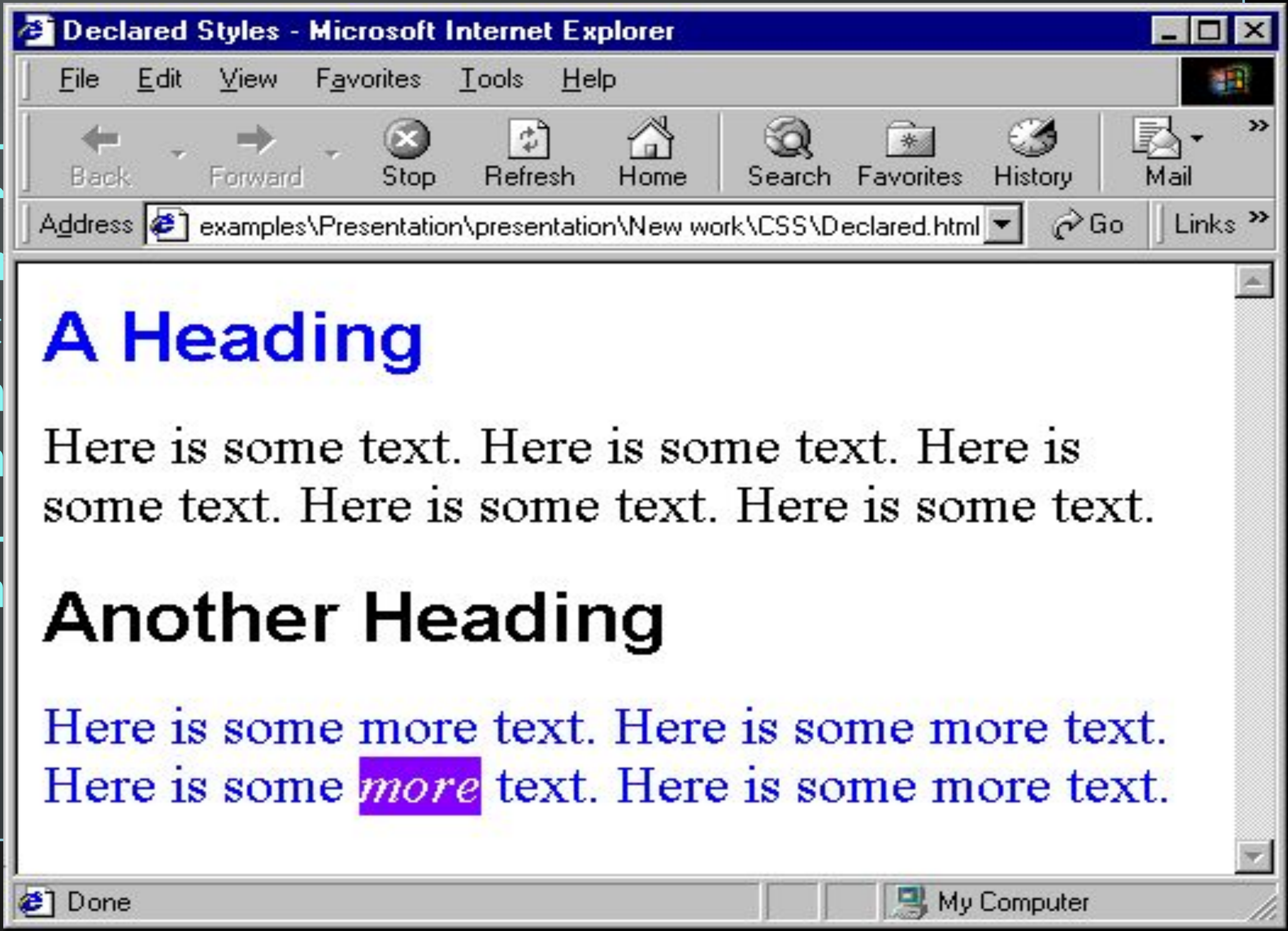
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitio
nal.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Style Sheets</title>
  <style type="text/css">
    em {background-color:#8000FF; color:white}
    h1 {font-family:Arial, sans-serif}
    p  {font-size:18pt}
    .blue {color:blue}
  </style>
<head>
```

Embedded Styles: Example (2)

```
…
<body>
  <h1 class="blue">A Heading</h1>
  <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>
</body>
</html>
```

Embedded Styles: Example (3)

```
…
<body>
  <h1 cl
  <p>Her
  is som
  text.<
  <h1>An
  <p cla
  Here i
  <p cla
  text.
</body>
</html>
```


Declared Styles - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail
Address examples\Presentation\presentation\New work\CSS\Declared.html
Go Links
Done
My Computer

A Heading

Here is some text. Here is some text. Here is some text. Here is some text. Here is some text.

Another Heading

Here is some more text. Here is some more text. Here is some *more* text. Here is some more text.

External CSS Styles

- 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 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">
```

telerik

External CSS Styles (2)

@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 32 CSS file limit

External Styles: Example

styles.css

```
/* 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)

external-styles.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
  Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitio
nal.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<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>
```

…

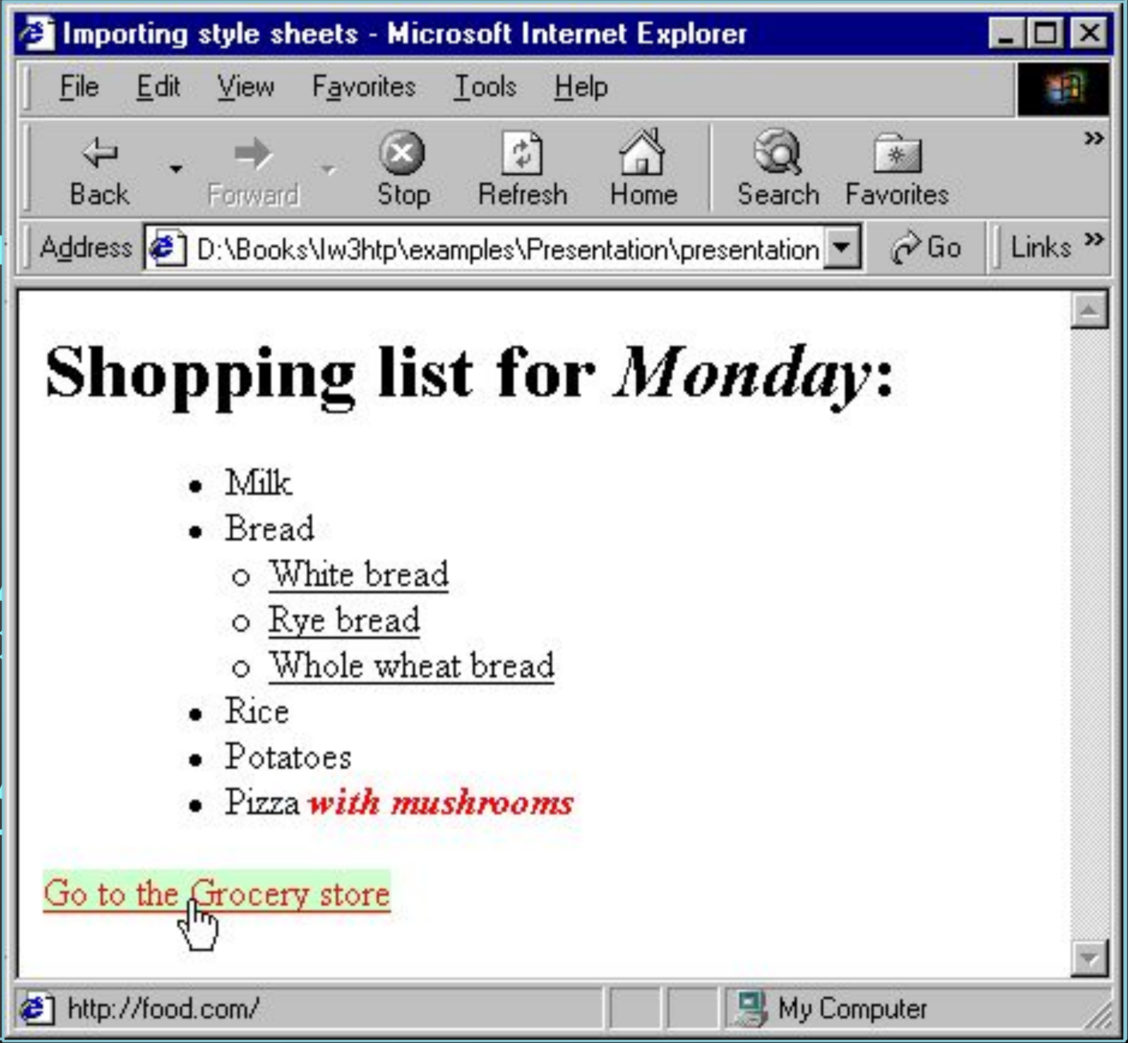
telerik

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 br
      <li>Whole
    </ul>
  </li>
  <li>Rice</li>
  <li>Potatoes</
  <li>Pizza <em
</ul>
<a href="http://
  store">Go to t
</body>
</html>
```



Importing style sheets - Microsoft Internet Explorer

File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address D:\Books\Iw3htp\examples\Presentation\presentation
Go Links

Shopping list for *Monday*:

- Milk
- Bread
  - White bread
  - Rye bread
  - Whole wheat bread
- Rice
- Potatoes
- Pizza ***with mushrooms***

Go to the Grocery store

http://food.com/
My Computer

Text-related CSS Properties

- `color` – specifies the color of the text
- `font-size` – size of font: `xx-small`, `x-small`, `small`, `medium`, `large`, `x-large`, `xx-large`, `smaller`, `larger` or numeric value
- `font-family` – comma separated font names
  - Example: `verdana`, `sans-serif`, etc.
  - The browser loads the first one that is available
  - There should always be at least one generic font
- `font-weight` can be `normal`, `bold`, `bolder`, `lighter` or a number in range [`100` … `900`]

CSS Rules for Fonts (2)

- `font-style` – styles the font
  - Values: `normal`, `italic`, `oblique`
- `text-decoration` – decorates the text
  - Values: `none`, `underline`, `line-trough`, `overline`, `blink`
- `text-align` – defines the alignment of text or other content
  - Values: `left`, `right`, `center`, `justify`

Shorthand Font Property

- font
  - Shorthand rule for setting multiple font properties at the same time

```
font:italic normal bold 12px/16px verdana
```

is equal to writing this:

```
font-style: italic;
font-variant: normal;
font-weight: bold;
font-size: 12px;
line-height: 16px;
font-family: verdana;
```

Backgrounds

- background-image
  - URL of image to be used as background, e.g.:

```
background-image:url("back.gif");
```

- background-color
  - Using color and image and the same time
- background-repeat
  - repeat-x, repeat-y, repeat, no-repeat
- background-attachment

Backgrounds (2)

- `background-position`: specifies vertical and horizontal position of the background image
  - Vertical position: `top`, `center`, `bottom`
  - Horizontal position: `left`, `center`, `right`
  - Both can be specified in percentage or other numerical values
  - Examples:

```
background-position: top left;
```

```
background-position: -5px 50%;
```

Background Shorthand Property

- background: shorthand rule for setting background properties at the same time:

```
background: #FFF0C0 url("back.gif") no-repeat
fixed top;
```

is equal to writing:

```
background-color: #FFF0C0;
background-image: url("back.gif");
background-repeat: no-repeat;
background-attachment: fixed;
background-position: top;
```

- Some browsers will not apply BOTH color and image for background if using shorthand rule

Background-image or `<img>`?

- Background images allow you to save many image tags from the HTML
  - Leads to less code
  - More content-oriented approach
- All images that are not part of the page content (and are used only for "beautification") should be moved to the CSS

Borders

- `border-width`: `thin`, `medium`, `thick` or numerical value (e.g. `10px`)
- `border-color`: color alias or RGB value
- `border-style`: `none`, `hidden`, `dotted`, `dashed`, `solid`, `double`, `groove`, `ridge`, `inset`, `outset`
- Each property can be defined separately for left, top, bottom and right
  - `border-top-style`, `border-left-color`, …

Border Shorthand Property

- `border`: shorthand rule for setting border properties at once:

```
border: 1px solid red
```

is equal to writing:

```
border-width:1px;
border-color:red;
border-style:solid;
```

- Specify different borders for the sides via shorthand rules: `border-top`, `border-left`, `border-right`, `border-bottom`

Width and Height

- `width` – defines numerical value for the width of element, e.g. `200px`
- `height` – defines numerical value for the height of element, e.g. `100px`
  - By default the height of an element is defined by its content
  - Inline elements do not apply height, unless you change their `display` style.

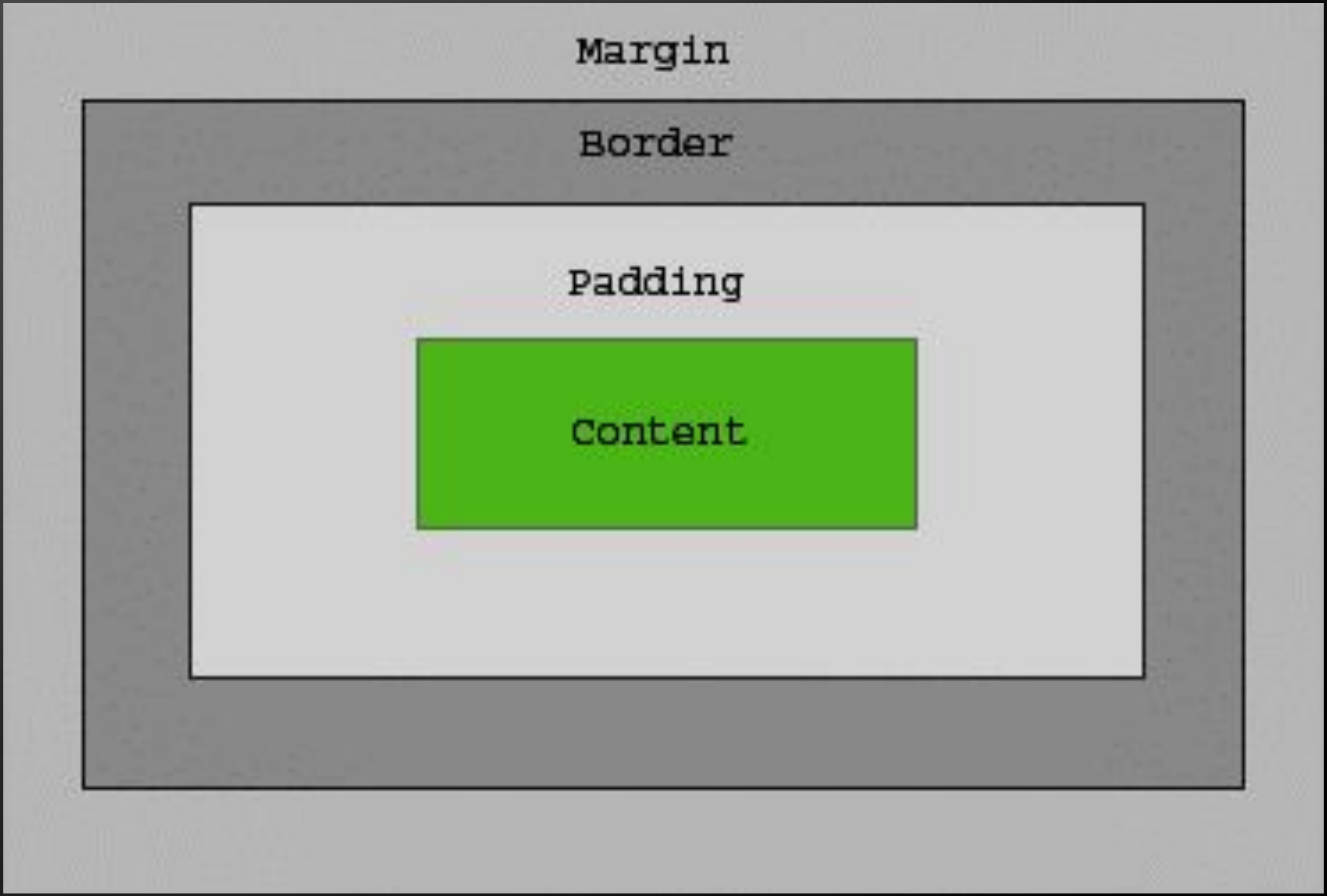
Margin and Padding

- `margin` and `padding` define the spacing around the element
  - Numerical value, e.g. `10px` or `-5px`
  - Can be defined for each of the four sides separately - `margin-top`, `padding-left`, …
  - `margin` is the spacing outside of the border
  - `padding` is the spacing between the border and the content
  - What are collapsing margins?

Margin and Padding: Short Rules

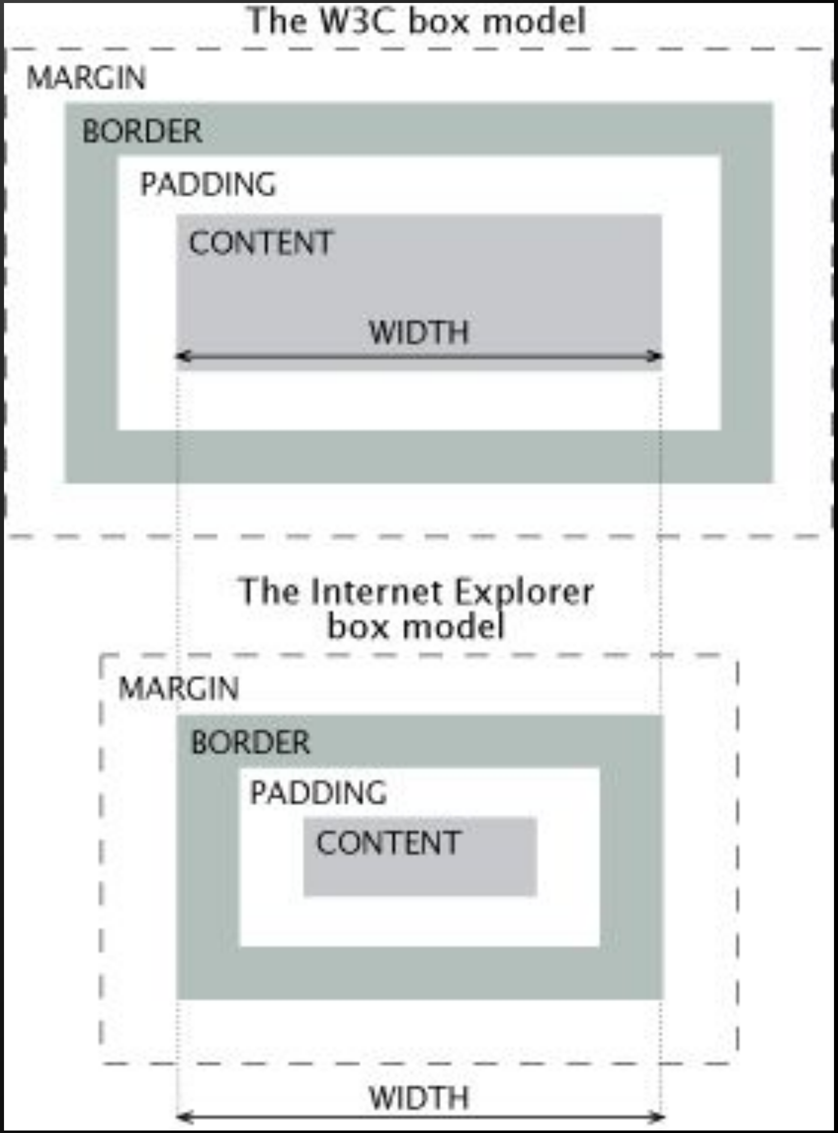
- `margin: 5px;`
  - Sets all four sides to have margin of 5 px;
- `margin: 10px 20px;`
  - top and bottom to 10px, left and right to 20px;
- `margin: 5px 3px 8px;`
  - top 5px, left/right 3px, bottom 8px
- `margin: 1px 3px 5px 7px;`
  - top, right, bottom, left (clockwise from top)
- Same for `padding`

The Box Model



IE Quirks Mode

- When using quirks mode (pages with no DOCTYPE or with a HTML 4 Transitional DOCTYPE), Internet Explorer violates the box model standard



Positioning

- `position`: defines the positioning of the element in the page content flow
- The value is one of:
  - `static` (default)
  - `relative` – relative position according to where the element would appear with static position
  - `absolute` – position according to the innermost positioned parent element
  - `fixed` – same as absolute, but ignores page scrolling

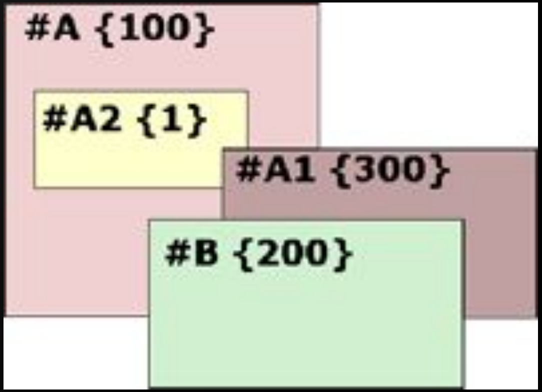
Positioning (2)

- Margin VS relative positioning
- Fixed and absolutely positioned elements do not influence the page normal flow and usually stay on top of other elements
  - Their position and size is ignored when calculating the size of parent element or position of surrounding elements
  - Overlaid according to their z-index
  - Inline fixed or absolutely positioned elements can apply height like block-level elements

Positioning (3)

- `top`, `left`, `bottom`, `right`: specifies offset of absolute/fixed/relative positioned element as numerical values
- `z-index` : specifies the stack level of positioned elements
  - Understanding stacking context

Each positioned element creates a stacking context.
Elements in different stacking contexts are overlapped according to the stacking order of their containers. For example, there is no way for #A1 and #A2 (children of #A) to be placed over #B without increasing the z-index of #A.



Inline element positioning

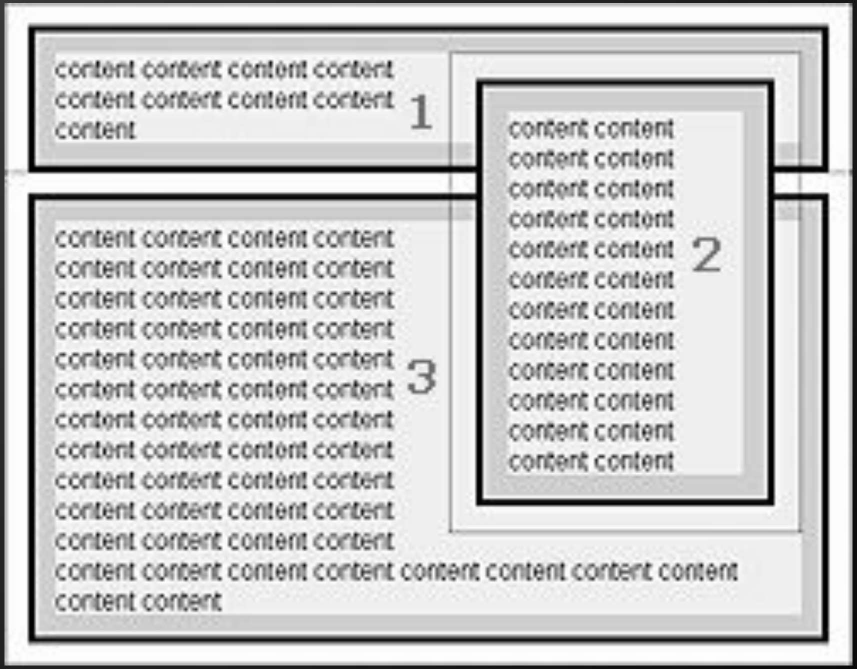
- `vertical-align`: sets the vertical-alignment of an inline element, according to the line height
  - Values: `baseline`, `sub`, `super`, `top`, `text-top`, `middle`, `bottom`, `text-bottom` or numeric
- Also used for content of table cells (which apply `middle` alignment by default)

Float

- `float`: the element "floats" to one side
  - `left`: places the element on the left and following content on the right
  - `right`: places the element on the right and following content on the left
  - floated elements should come before the content that will wrap around them in the code
  - margins of floated elements do not collapse
  - floated inline elements can apply height

Float (2)

- How floated elements are positioned



content content content content content content content content content
1
content content
2
content content
3

telerik

Clear

- `clear`
  - Sets the sides of the element where other floating elements are NOT allowed
  - Used to "drop" elements below floated ones or expand a container, which contains only floated children
  - Possible values: `left`, `right`, `both`
- Clearing floats
  - additional element (`<div>`) with a clear style

Clear (2)

- Clearing floats (continued)
  - `:after { content: ""; display: block; clear: both; height: 0; }`
  - Triggering hasLayout in IE expands a container of floated elements
    - display: inline-block;
    - zoom: 1;

Opacity

- `opacity`: specifies the opacity of the element
  - Floating point number from 0 to 1
  - For old Mozilla browsers use `–moz-opacity`
  - For IE use `filter:alpha(opacity=value)` where value is from 0 to 100; also, "binary and script behaviors" must be enabled and `hasLayout` must be triggered, e.g. with `zoom:1`

Visibility

- `visibility`
  - Determines whether the element is visible
  - `hidden`: element is not rendered, but still occupies place on the page (similar to `opacity:0`)
  - `visible`: element is rendered normally

Display

- `display`: controls the display of the element and the way it is rendered and if breaks should be placed before and after the element
  - `inline`: no breaks are placed before and after (`<span>` is an inline element)
  - `block`: breaks are placed before AND after the element (`<div>` is a block element)

Display (2)

- `display`: controls the display of the element and the way it is rendered and if breaks should be placed before and after the element
  - `none`: element is hidden and its dimensions are not used to calculate the surrounding elements rendering (differs from `visibility: hidden`!)
  - There are some more possible values, but not all browsers support them
    - Specific displays like `table-cell` and `table-row`

Overflow

- `overflow`: defines the behavior of element when content needs more space than you have specified by the size properties or for other reasons. Values:
  - `visible` (default) – content spills out of the element
  - `auto` - show scrollbars if needed
  - `scroll` – always show scrollbars
  - `hidden` – any content that cannot fit is clipped

Other CSS Properties

- `cursor`: specifies the look of the mouse cursor when placed over the element
  - Values: `crosshair`, `help`, `pointer`, `progress`, `move`, `hair`, `col-resize`, `row-resize`, `text`, `wait`, `copy`, `drop`, and others
- `white-space` – controls the line breaking of text. Value is one of:
  - `nowrap` – keeps the text on one line
  - `normal` (default) – browser decides whether to brake the lines if needed

Benefits of using CSS

- More powerful formatting than using presentation tags
- Your pages load faster, because browsers cache the `.css` files
- Increased accessibility, because rules can be defined according given media
- Pages are easier to maintain and update

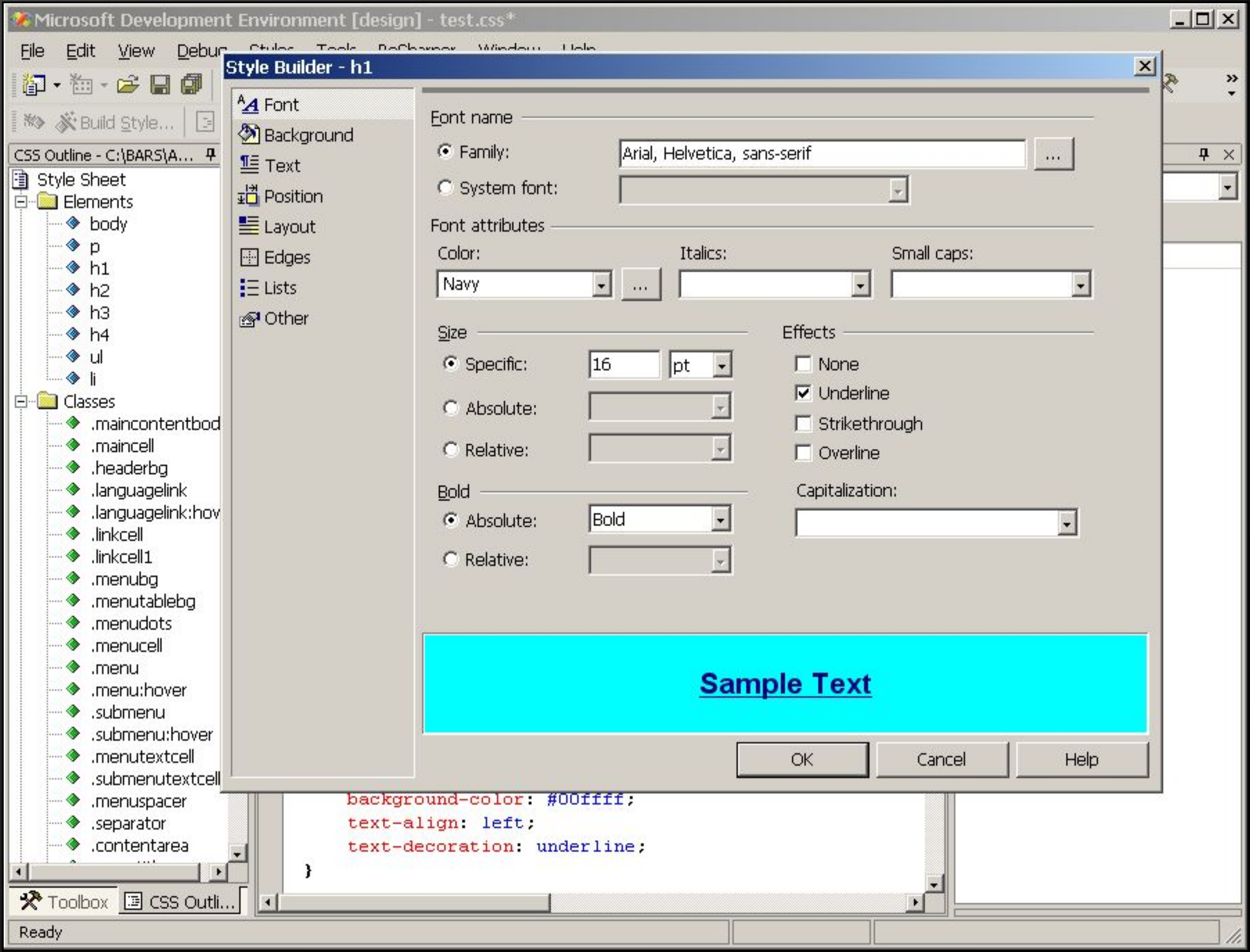
Maintenance Example

Title
Some random text here. You can't read it anyway! Har har har! Use Css.

CS
S
file

CSS Development Tools

- Visual Studio – CSS Editor



Font name
Family: Arial, Helvetica, sans-serif
System font:

Font attributes
Color: Navy
Italics:
Small caps:

Size
Specific: 16 pt
Absolute:
Relative:

Effects
[ ] None
[✓] Underline
[ ] Strikethrough
[ ] Overline

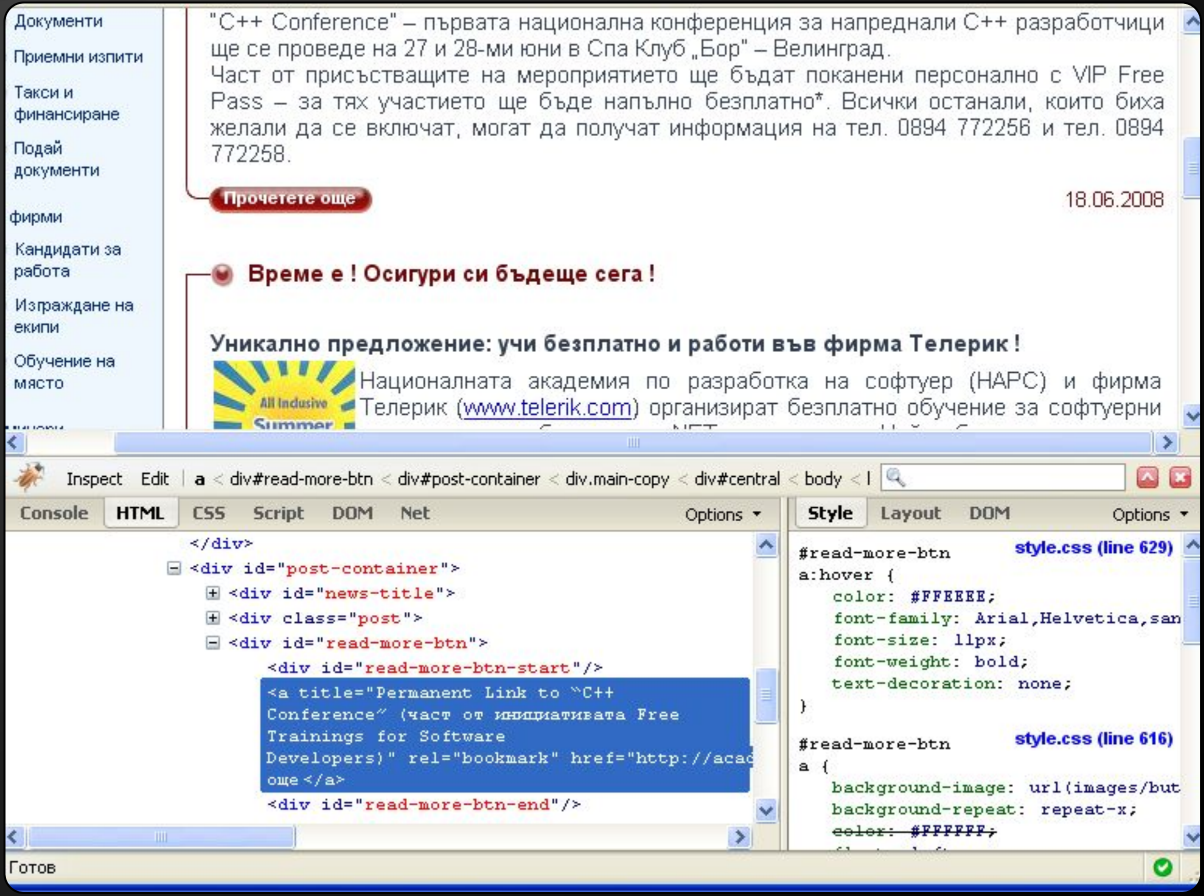
Bold
Absolute: Bold
Relative:

Capitalization:

```
    background-color: #00ffff;
    text-align: left;
    text-decoration: underline;
}
```

CSS Development Tools (3)

- Firebug – add-on to Firefox used to examine and adjust CSS and HTML



Документи
Приемни изпити
Такси и финансиране
Подай документи
фирми
Кандидати за работа
Изграждане на екипи
Обучение на място

"C++ Conference" – първата национална конференция за напреднали C++ разработчици ще се проведе на 27 и 28-ми юни в Спа Клуб „Бор" – Велинград.
Част от присъстващите на мероприятието ще бъдат поканени персонално с VIP Free Pass – за тях участието ще бъде напълно безплатно*. Всички останали, които биха желали да се включат, могат да получат информация на тел. 0894 772256 и тел. 0894 772258.

Прочетете още

18.06.2008

Време е ! Осигури си бъдеще сега !

Уникално предложение: учи безплатно и работи във фирма Телерик !

Националната академия по разработка на софтуер (НАРС) и фирма Телерик (www.telerik.com) организират безплатно обучение за софтуерни

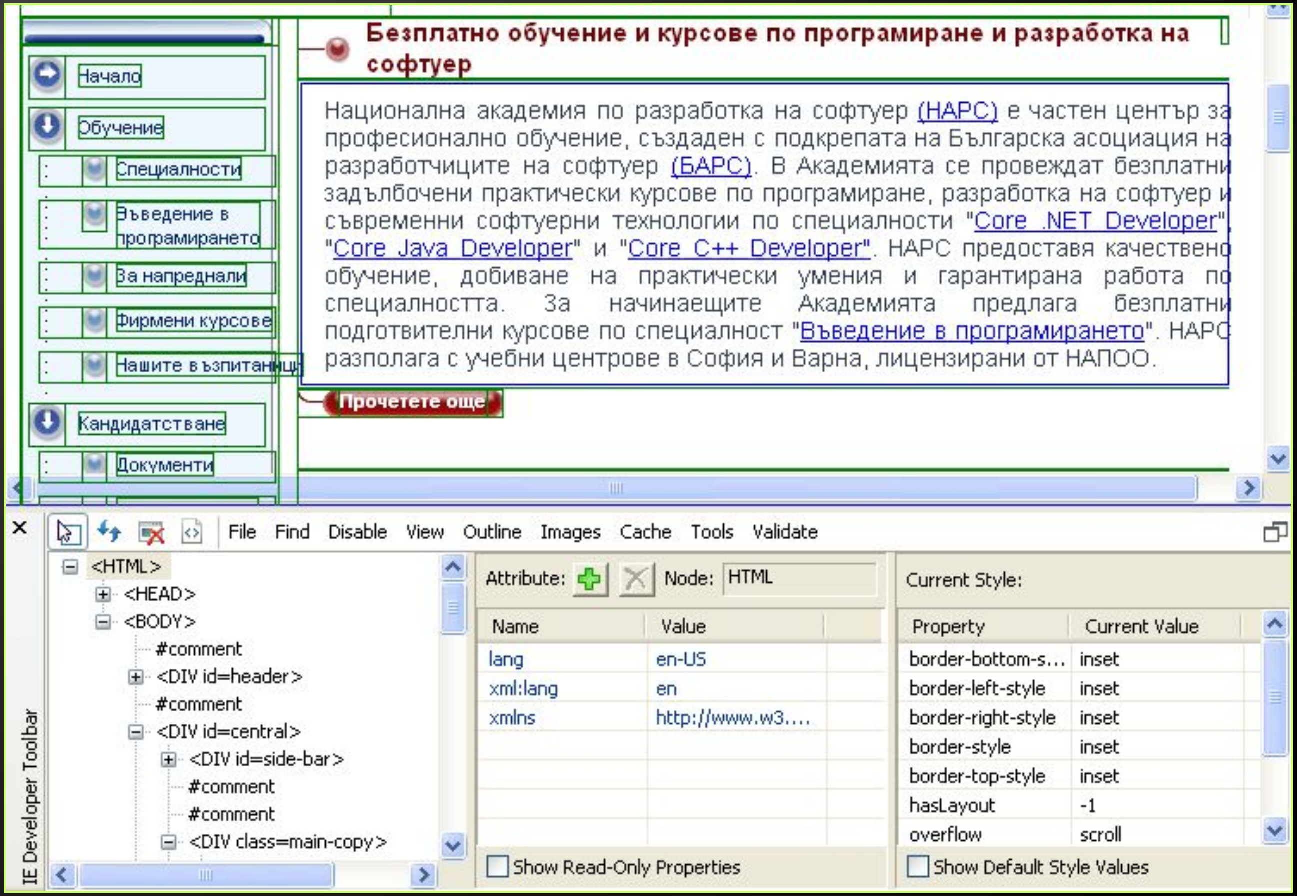
```
</div>
⊟ <div id="post-container">
   ⊞ <div id="news-title">
   ⊞ <div class="post">
   ⊟ <div id="read-more-btn">
        <div id="read-more-btn-start"/>
        <a title="Permanent Link to "C++
        Conference" (част от инициативата Free
        Trainings for Software
        Developers)" rel="bookmark" href="http://acad
        още </a>
        <div id="read-more-btn-end"/>
```

```
#read-more-btn          style.css (line 629)
a:hover {
   color: #FFEEEE;
   font-family: Arial,Helvetica,san
   font-size: 11px;
   font-weight: bold;
   text-decoration: none;
}

#read-more-btn          style.css (line 616)
a {
   background-image: url(images/but
   background-repeat: repeat-x;
   ~~color: #FFFFFF;~~
```

CSS Development Tools (4)

- **IE Developer Toolbar – add-on to IE used to examine CSS and HTML (press [F12])**



Безплатно обучение и курсове по програмиране и разработка на софтуер

Национална академия по разработка на софтуер (НАРС) е частен център за професионално обучение, създаден с подкрепата на Българска асоциация на разработчиците на софтуер (БАРС). В Академията се провеждат безплатни задълбочени практически курсове по програмиране, разработка на софтуер и съвременни софтуерни технологии по специалности "Core .NET Developer", "Core Java Developer" и "Core C++ Developer". НАРС предоставя качествено обучение, добиване на практически умения и гарантирана работа по специалността. За начинаещите Академията предлага безплатни подготвителни курсове по специалност "Въведение в програмирането". НАРС разполага с учебни центрове в София и Варна, лицензирани от НАПОО.

```
<HTML>
  <HEAD>
  <BODY>
    #comment
    <DIV id=header>
    #comment
    <DIV id=central>
      <DIV id=side-bar>
      #comment
      #comment
      <DIV class=main-copy>
```

Attribute: Node: HTML

Name	Value
lang	en-US
xml:lang	en
xmlns	http://www.w3....

Show Read-Only Properties

Current Style:

Property	Current Value
border-bottom-s...	inset
border-left-style	inset
border-right-style	inset
border-style	inset
border-top-style	inset
hasLayout	-1
overflow	scroll

Show Default Style Values