telerik

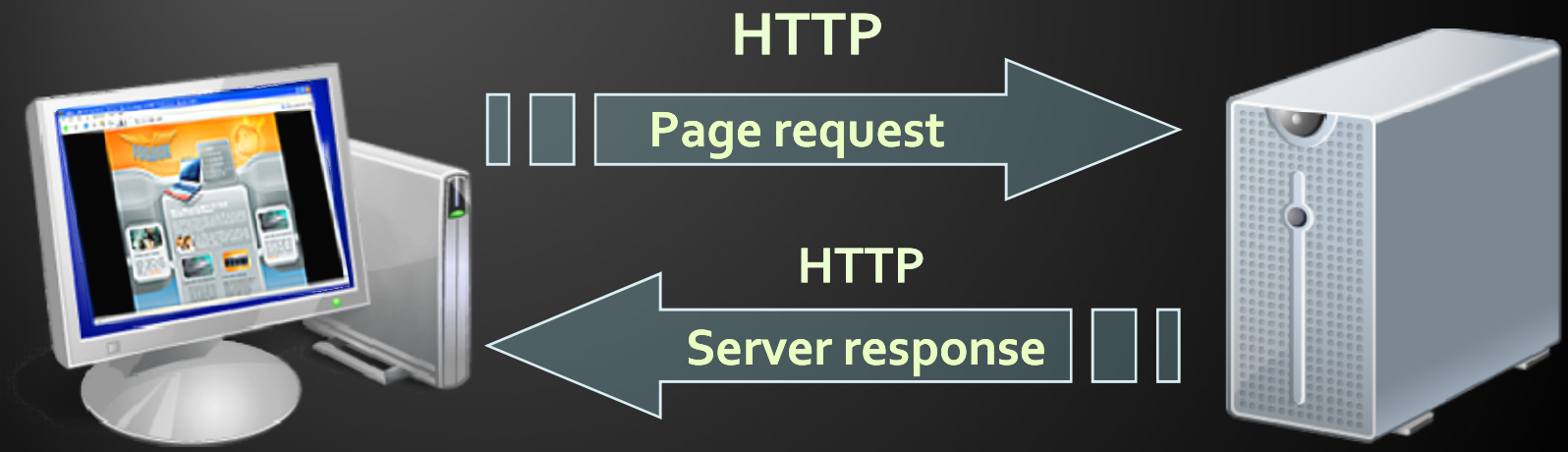
HTML .. CSS

Mr.Jestin James M
Asst.Professor, Department of Computer Science
L.F College Guruvayoor

telerik

How the Web Works?

- WWW use classical client / server architecture
  - HTTP is text-based request-response protocol



Client running a Web Browser

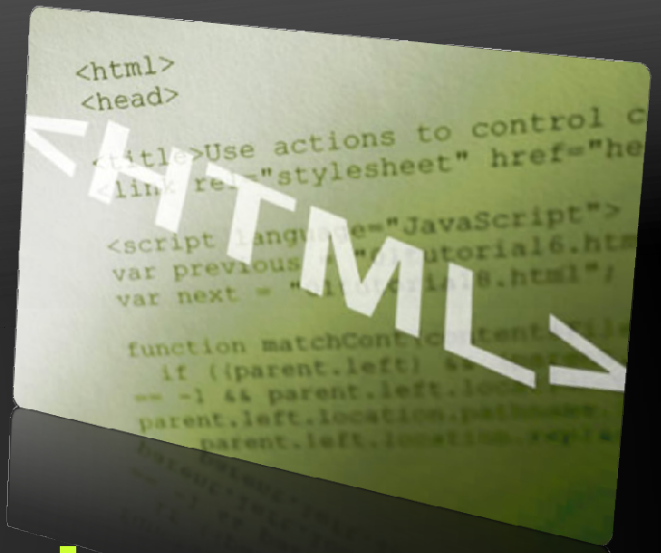
Server running Web Server Software (IIS, Apache, etc.)

telerik

What is a Web Page?

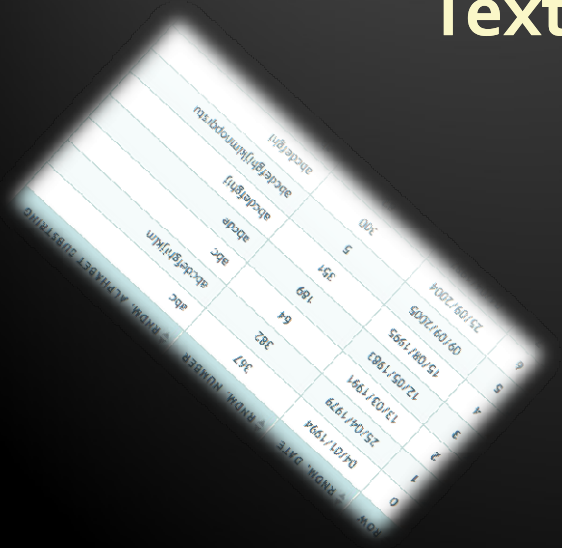
- Web pages are text files containing HTML
- HTML – Hyper Text Markup Language
- The markup tags provide information about the page content structure
- An HTML file must have an `.htm` or `.html` file extension
- HTML files can be created with text editors:
  - NotePad, NotePad ++, PSPad

telerik



HTML Basics

Text, Images, Tables, Forms



telerik

HTML Structure

- HTML is comprised of "elements" and "tags"
  - Begins with <html> and ends with </html>
- Elements (tags) are nested one inside another:

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

- Tags have attributes:

```
<img src="logo.jpg" alt="logo" />
```

- HTML describes structure using two main sections: <head> and <body>

telerik

HTML Code Formatting

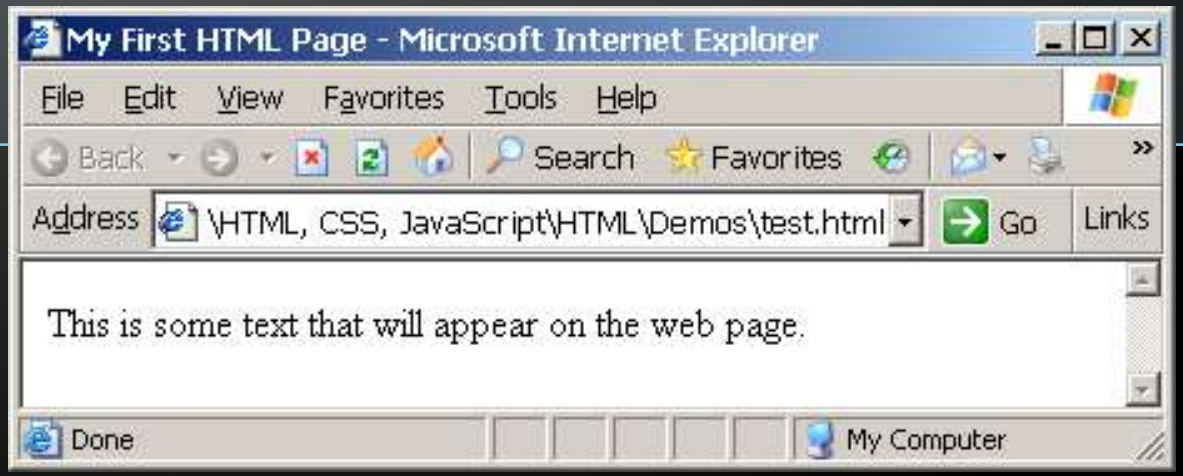
- The HTML source code should be formatted to increase readability and facilitate debugging.
  - Every block element should start on a new line.
  - Every nested (block) element should be indented.
  - Browsers ignore multiple whitespaces in the page source, so formatting is harmless.

telerik

First HTML Page

test.html

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>My First HTML Page</title>
  </head>
  <body>
     <p>This is some text...</p>
  </body>
</html>
```



telerik

First HTML Page: Tags

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>My First HTML Page</title>
  </head>
  <body>
    <p>This is some text...</p>
  </body>
</html>
```

Opening tag

Closing tag

An HTML element consists of an opening tag, a closing tag and the content inside.

telerik

First HTML Page: Header

HTML header

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>My First HTML Page</title>
  </head>
  <body>
     <p>This is some text...</p>
  </body>
</html>
```

telerik

First HTML Page: Body

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>My First HTML Page</title>
  </head>
  <body>
    <p>This is some text...</p>
  </body>
</html>
```

HTML body

telerik

Some Simple Tags

- Hyperlink Tags

```
<a href="http://www.telerik.com/"
  title="Telerik">Link to Telerik Web site</a>
```

- Image Tags

```
<img src="logo.gif" alt="logo" />
```

- Text formatting tags

```
This text is <em>emphasized.</em>
<br />new line<br />
This one is <strong>more emphasized.</strong>
```

telerik

Some Simple Tags – Example

some-tags.html

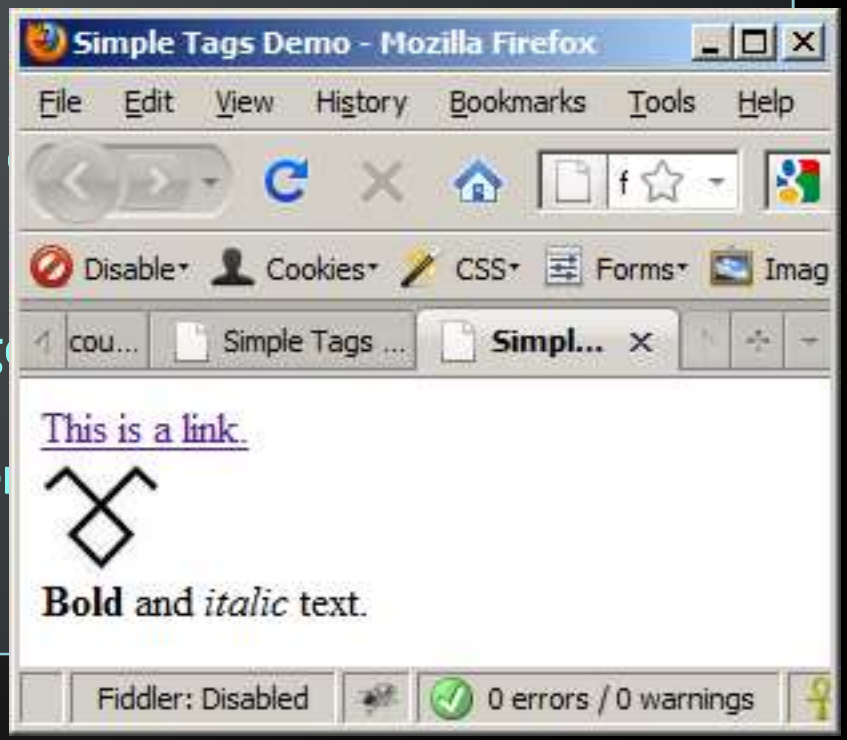
```
<!DOCTYPE HTML>
<html>
<head>
  <title>Simple Tags Demo</title>
</head>
<body>
<a href="http://www.telerik.com/" title=
    "Telerik site">This is a link.</a>
<br />
<img src="logo.gif" alt="logo" />
<br />
<strong>Bold</strong> and <em>italic</em> text.
</body>
</html>
```

telerik

Some Simple Tags – Example (2)

some-tags.html

```
<!DOCTYPE HTML>
<html>
<head>
  <title>Simple Tags Demo</title>
</head>
<body>
<a href="http://www.telerik.
    "Telerik site">This is a
<br />
<img src="logo.gif" alt="log
<br />
<strong>Bold</strong> and <e
</body>
</html>
```



telerik

Tags Attributes

- Tags can have attributes
  - Attributes specify properties and behavior
  - Example:

Attribute `alt` with value "`logo`"

```
<img src="logo.gif" alt="logo" />
```

  - Few attributes can apply to every element:
    - `id`, `style`, `class`, `title`
    - The `id` is unique in the document

telerik

Headings and Paragraphs

- Heading Tags (h1 – h6)

```
<h1>Heading 1</h1>
<h2>Sub heading 2</h2>
<h3>Sub heading 3</h3>
```

- Paragraph Tags

```
<p>This is my first paragraph</p>
<p>This is my second paragraph</p>
```

- Sections: div and span

```
<div style="background: skyblue;">
   This is a div</div>
```

telerik

Headings and Paragraphs – Example

headings.html

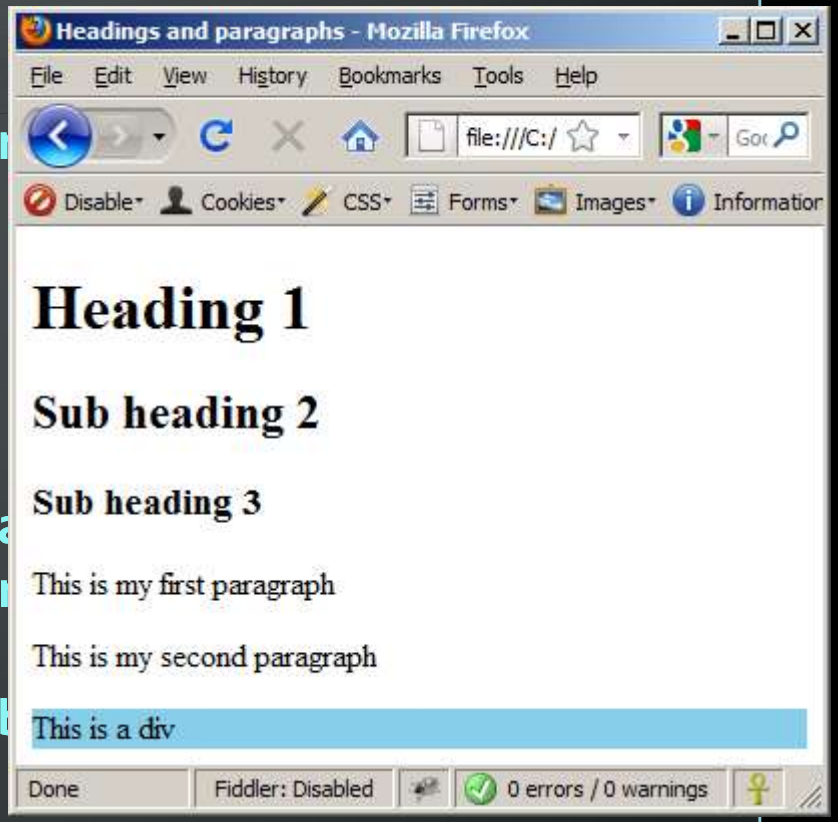
```
<!DOCTYPE HTML>
<html>
  <head><title>Headings and paragraphs</title></head>
  <body>
    <h1>Heading 1</h1>
    <h2>Sub heading 2</h2>
    <h3>Sub heading 3</h3>

    <p>This is my first paragraph</p>
    <p>This is my second paragraph</p>

    <div style="background:skyblue">
      This is a div</div>
  </body>
</html>
```

telerik

Headings and Paragraphs – Example (2)

headings.html

```
<!DOCTYPE HTML>
<html>
  <head><title>Headings and par
  <body>
    <h1>Heading 1</h1>
    <h2>Sub heading 2</h2>
    <h3>Sub heading 3</h3>

    <p>This is my first paragra
    <p>This is my second paragr

    <div style="background:skyb
      This is a div</div>
  </body>
</html>
```



telerik

The <!DOCTYPE> Declaration

- HTML documents must start with a document type definition (DTD)
  - It tells web browsers what type is the served code
  - Possible versions: HTML 4.01, XHTML 1.0 (Transitional or Strict), XHTML 1.1, HTML 5
- Example:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

telerik

HTML vs. XHTML

- XHTML is more strict than HTML
  - Tags and attribute names must be in lowercase
  - All tags must be closed (`<br/>`, `<img/>`) while HTML allows `<br>` and `<img>` and implies missing closing tags (`<p>par1 <p>par2`)
  - XHTML allows only one root `<html>` element (HTML allows more than one)

telerik

XHTML vs. HTML (2)

- Many element attributes are deprecated in XHTML, most are moved to CSS
- Attribute minimization is forbidden, e.g.

```
<input type="checkbox" checked>
```

↓

```
<input type="checkbox" checked="checked" />
```

- Note: Web browsers load XHTML faster than HTML and valid code faster than invalid!

telerik

The <head> Section

- Contains information that doesn't show directly on the viewable page
- Starts after the <!doctype> declaration
- Begins with <head> and ends with </head>
- Contains mandatory single <title> tag
- Can contain some other tags, e.g.
  - <meta>
  - <script>
  - <style>
  - <!-- comments -->

telerik

<head> Section: <title> tag

- Title should be placed between <head> and </head> tags

```
<title>Telerik Academy – Winter Season 2009/2010
</title>
```

- Used to specify a title in the window title bar
- Search engines and people rely on titles

telerik

<head> Section: <meta>

- Meta tags additionally describe the content contained within the page

```
<meta name="description" content="HTML
tutorial" />

<meta name="keywords" content="html, web
design, styles" />

<meta name="author" content="Chris Brewer" />

<meta http-equiv="refresh" content="5;
url=http://www.telerik.com" />
```

telerik

<head> Section: <script>

- The <script> element is used to embed scripts into an HTML document
  - Script are executed in the client's Web browser
  - Scripts can live in the <head> and in the <body> sections
- Supported client-side scripting languages:
  - JavaScript (it is not Java!)
  - VBScript
  - JScript

telerik

The <script> Tag – Example

scripts-example.html

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>JavaScript Example</title>
    <script type="text/javascript">
      function sayHello() {
        document.write("<p>Hello World!<\/p>");
      }
    </script>
  </head>
  <body>
    <script type=
      "text/javascript">
      sayHello();
    </script>
  </body>
</html>
```

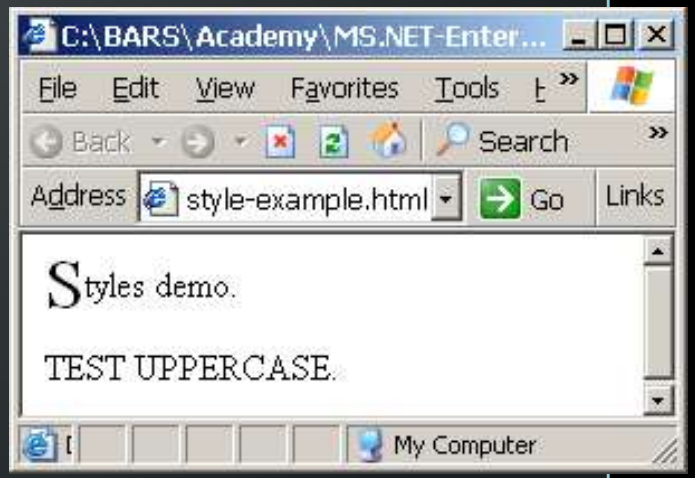
telerik

<head> Section: <style>

- The <style> element embeds formatting information (CSS styles) into an HTML page

style-example.html

```
<html>
  <head>
    <style type="text/css">
      p { font-size: 12pt; line-height: 12pt; }
      p:first-letter { font-size: 200%; }
      span { text-transform: uppercase; }
    </style>
  </head>
  <body>
    <p>Styles demo.<br />
       <span>Test uppercase</span>.
    </p>
  </body>
</html>
```



telerik

Comments: <!-- --> Tag

- Comments can exist anywhere between the <html></html> tags
- Comments start with <!-- and end with -->

```
<!-- Telerik Logo (a JPG file) -->
<img src="logo.jpg" alt="Telerik Logo">
<!-- Hyperlink to the web site -->
<a href="http://telerik.com/">Telerik</a>
<!-- Show the news table -->
<table class="newstable">
...
```

telerik

<body> Section: Introduction

- The <body> section describes the viewable portion of the page
- Starts after the <head> </head> section
- Begins with <body> and ends with </body>

```
<html>
  <head><title>Test page</title></head>
  <body>
    <!-- This is the Web page body -->
  </body>
</html>
```

telerik

Text Formatting

- Text formatting tags modify the text between the opening tag and the closing tag
  - Ex. `<b>Hello</b>` makes "Hello" bold

`<b></b>`	**bold**
`<i></i>`	*italicized*
`<u></u>`	underlined
`<sup></sup>`	Sample^{superscript}
`<sub></sub>`	Sample$_{subscript}$
`<strong></strong>`	**strong**
`<em></em>`	*emphasized*
`<pre></pre>`	`Preformatted text`
`<blockquote></blockquote>`	`Quoted text block`
`<del></del>`	Deleted text – ~~strike through~~

telerik

Text Formatting – Example

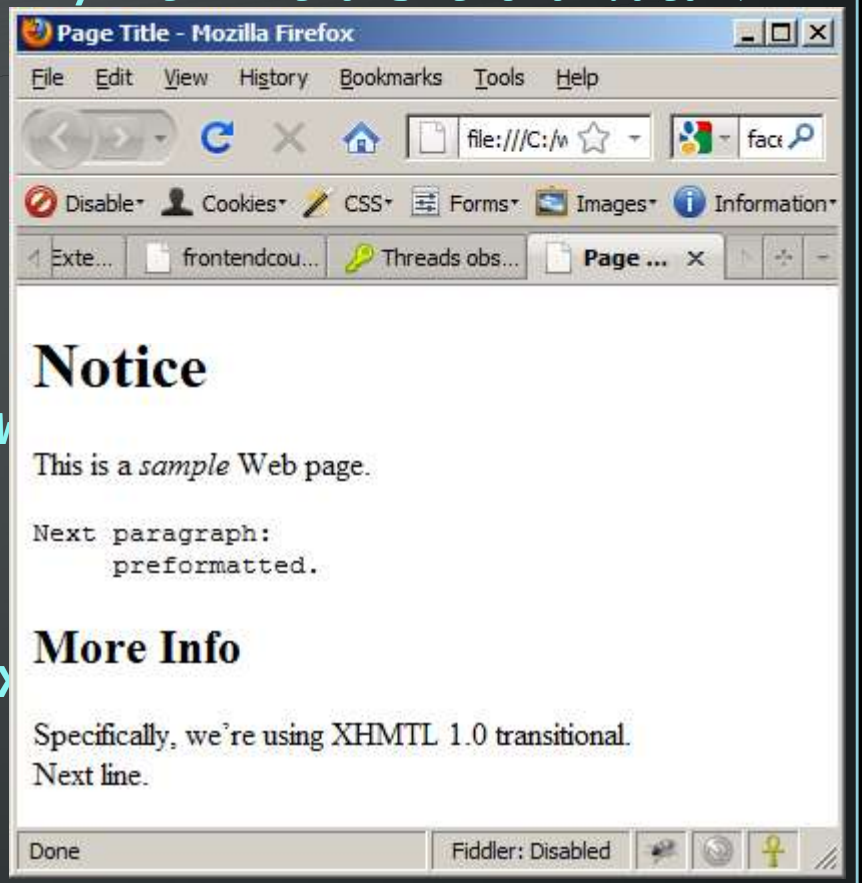
text-formatting.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>
  <h1>Notice</h1>
  <p>This is a <em>sample</em> Web page.</p>
  <p><pre>Next paragraph:
     preformatted.</pre></p>
  <h2>More Info</h2>
  <p>Specifically, we're using XHMTL 1.0 transitional.<br />
    Next line.</p>
  </body>
</html>
```

Text Formatting – Example (2)

text-formatting.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>
  <h1>Notice</h1>
  <p>This is a <em>sample</em> W
  <p><pre>Next paragraph:
     preformatted.</pre></p>
  <h2>More Info</h2>
  <p>Specifically, we're using X
    Next line.</p>
  </body>
</html>
```



telerik

Hyperlinks: <a> Tag

- Link to a document called `form.html` on the same server in the same directory:

```
<a href="form.html">Fill Our Form</a>
```

- Link to a document called `parent.html` on the same server in the parent directory:

```
<a href="../parent.html">Parent</a>
```

- Link to a document called `cat.html` on the same server in the subdirectory `stuff`:

```
<a href="stuff/cat.html">Catalog</a>
```

telerik

Hyperlinks: <a> Tag (2)

- Link to an external Web site:

```
<a href="http://www.devbg.org" target="_blank">BASD</a>
```

  - Always use a full URL, including "http://", not just "www.somesite.com"
  - Using the target="_blank" attribute opens the link in a new window

- Link to an e-mail address:

```
<a href="mailto:bugs@example.com?subject=Bug+Report">
Please report bugs here (by e-mail only)</a>
```

telerik

Hyperlinks: <a> Tag (3)

- Link to a document called `apply-now.html`
  - On the same server, in same directory
  - Using an image as a link button:

```
<a href="apply-now.html"><img
  src="apply-now-button.jpg" /></a>
```

- Link to a document called `index.html`
  - On the same server, in the subdirectory `english` of the parent directory:

```
<a href="../english/index.html">Switch to
English version</a>
```

telerik

Hyperlinks and Sections

- Link to another location in the same document:

```
<a href="#section1">Go to Introduction</a>
...
<h2 id="section1">Introduction</h2>
```

- Link to a specific location in another document:

```
<a href="chapter3.html#section3.1.1">Go to Section
3.1.1</a>

<!-- In chapter3.html -->
...
<div id="section3.1.1">
  <h3>3.1.1. Technical Background</h3>
</div>
```

telerik

Hyperlinks – Example

hyperlinks.html

```
<a href="form.html">Fill Our Form</a> <br />
<a href="../parent.html">Parent</a> <br />
<a href="stuff/cat.html">Catalog</a> <br />
<a href="http://www.devbg.org" target="_blank">BASD</a>
<br />
<a href="mailto:bugs@example.com?subject=Bug
Report">Please report bugs here (by e-mail only)</a>
<br />
<a href="apply-now.html"><img src="apply-now-button.jpg”
/></a> <br />
<a href="../english/index.html">Switch to English
version</a> <br />
```

telerik

Hyperlinks – Example (2)

hyperlinks.html

```
<a href="form.html">Fill Our Form</a> <br />
<a href="../parent.html">Parent</a> <br />
<a href="stuff/cat.html">Catalog</a> <br />
<a href="http://www.devbg.org" target="_blank">BA
<br />
<a href="mailto:bugs@example.com?subject=Bug
Report">Please report bugs here (by e-mail only)<
<br />
<a href="apply-now.html"><img src="apply-now-butt
/></a> <br />
<a href="../english/index.html">Switch to Englis
version</a> <br />
```

telerik

Links to the Same Document – Example

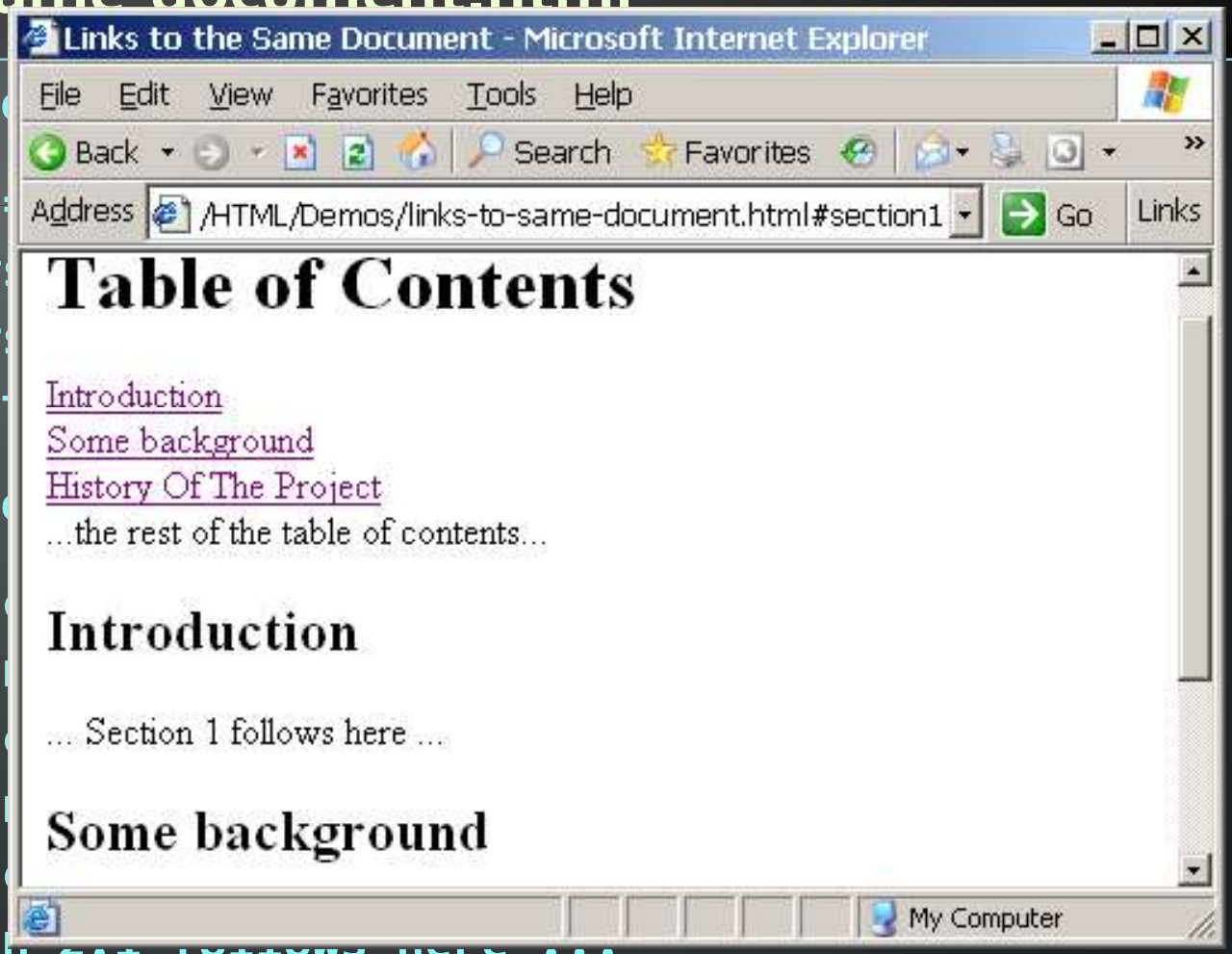
links-to-same-document.html

```
<h1>Table of Contents</h1>

<p><a href="#section1">Introduction</a><br />
<a href="#section2">Some background</A><br />
<a href="#section2.1">Project History</a><br />
...the rest of the table of contents...

<!-- The document text follows here -->

<h2 id="section1">Introduction</h2>
... Section 1 follows here ...
<h2 id="section2">Some background</h2>
... Section 2 follows here ...
<h3 id="section2.1">Project History</h3>
... Section 2.1 follows here ...
```

Links to the Same Document – Example (2)

links-to-same-document.html

```
<h1>Table
<p><a href=
<a href="#
<a href="#
...the res
<!-- The d
<h2 id="se
... Sectio
<h2 id="se
... Sectio
<h3 id="se
... Sectio
```



telerik

Images: <img> tag

- Inserting an image with <img> tag:

```
<img src="/img/basd-logo.png">
```

- Image attributes:

src	Location of image file (relative or absolute)
alt	Substitute text for display (e.g. in text mode)
height	Number of pixels of the height
width	Number of pixels of the width
border	Size of border, 0 for no border

- Example:

```
<img src="./php.png" alt="PHP Logo" />
```

telerik

Miscellaneous Tags

- <hr />: Draws a horizontal rule (line):

```
<hr size="5" width="70%" />
```

- : Deprecated!

```
<center>Hello World!</center>
```

- : Deprecated!

```
<font size="3" color="blue">Font3</font>
<font size="+4" color="blue">Font+4</font>
```

Miscellaneous Tags – Example

misc.html

```
<html>
  <head>
    <title>Miscellaneous Tags
  </head>
  <body>
    <hr size="5" width="70%" />
    <center>Hello World!</center>
    <font size="3" color="blue">Font3</font>
    <font size="+4" color="blue">Font+4</font>
  </body>
</html>
```

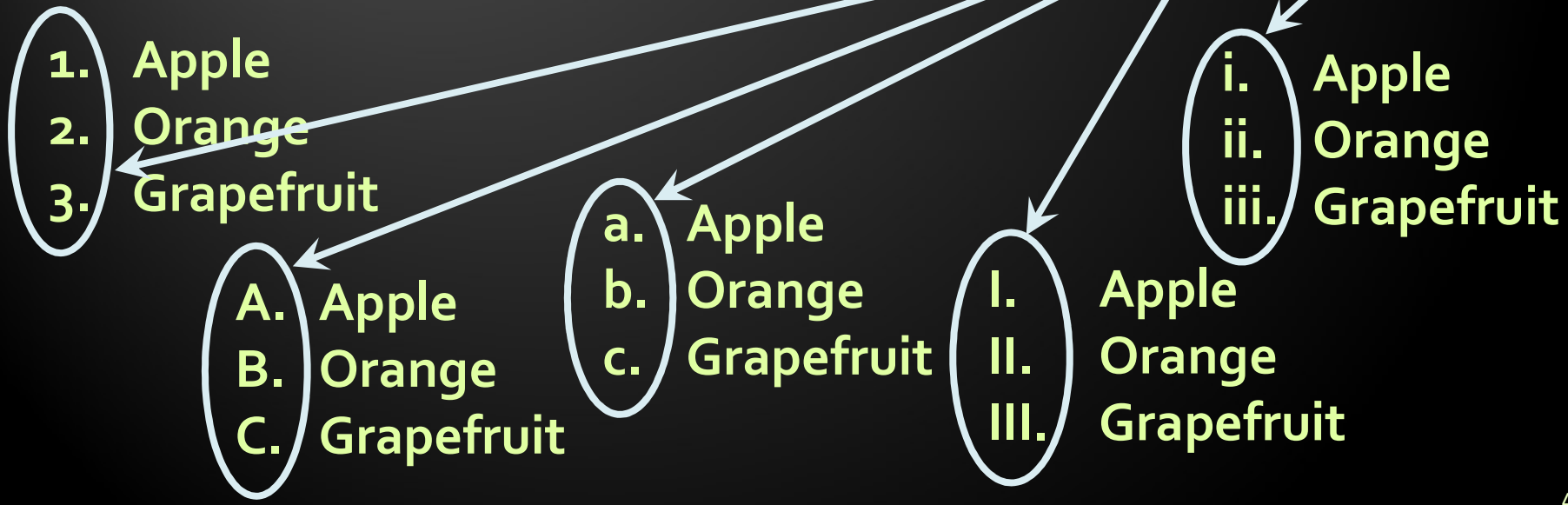
telerik

Ordered Lists: <ol> Tag

- Create an Ordered List using :

```
<ol type="1">
  <li>Apple</li>
  <li>Orange</li>
  <li>Grapefruit</li>
</ol>
```

- Attribute values for type are 1, A, a, I, or i



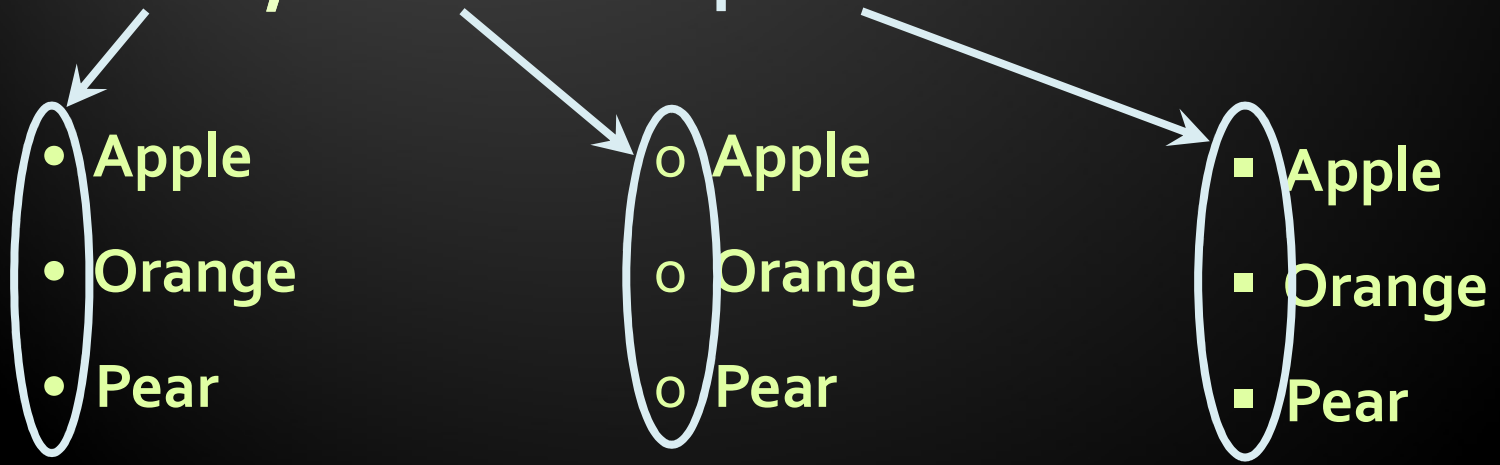
telerik

Unordered Lists: <ul> Tag

- Create an Unordered List using :

```
<ul type="disk">
  <li>Apple</li>
  <li>Orange</li>
  <li>Grapefruit</li>
</ul>
```

- Attribute values for type are:
  - disc, circle or square



telerik

Definition lists: <dl> tag

- Create definition lists using <dl>
  - Pairs of text and associated definition; text is in <dt> tag, definition in <dd> tag

```
<dl>
   <dt>HTML</dt>
   <dd>A markup language …</dd>
   <dt>CSS</dt>
   <dd>Language used to …</dd>
</dl>
```

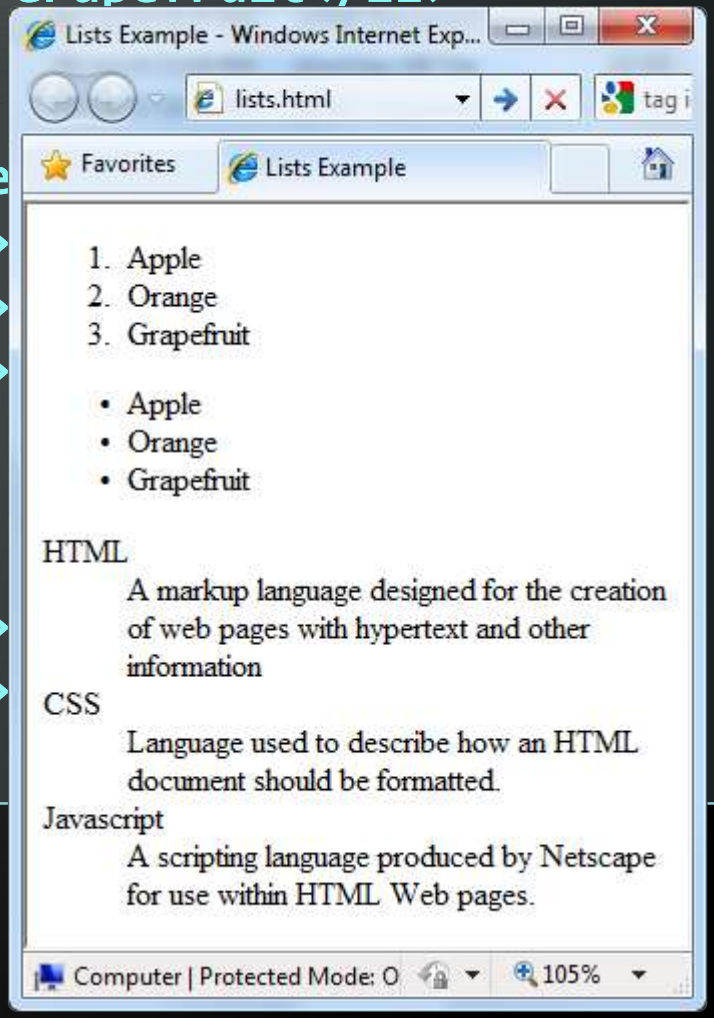
  - Renders without bullets
  - Definition is indented

telerik

Lists – Example

```
<ol type="1">
    <li>Apple</li>
    <li>Orange</li>
    <li>Grapefruit</li>
</ol>

<ul type
    <li>
    <li>
    <li>
</ul>

<dl>
    <dt>
    <dd>
</dl>
```

lists.html



telerik

HTML Special Characters

Symbol Name	HTML Entity	Symbol
Copyright Sign	©	©
Registered Trademark Sign	®	®
Trademark Sign	™	™
Less Than	<	<
Greater Than	>	>
Ampersand	&	&
Non-breaking Space		
Em Dash	—	—
Quotation Mark	"	"
Euro	€	€
British Pound	£	£
Japanese Yen	¥	¥

telerik

Special Characters – Example

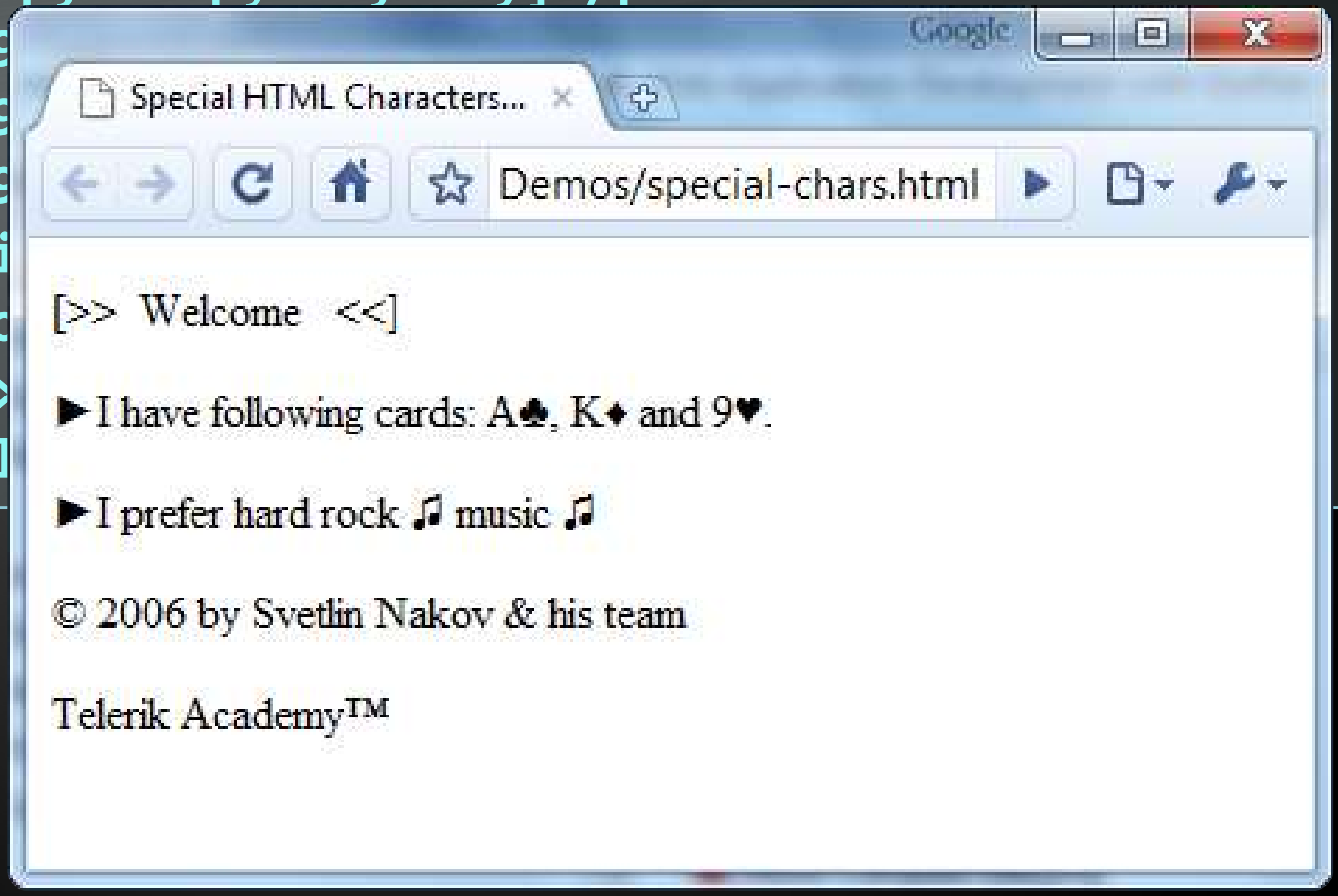
special-chars.html

```
<p>[&gt;&gt;  Welcome
      &lt;&lt;]</p>
  <p>&#9658;I have following cards:
    A&#9827;, K&#9830; and 9&#9829;.</p>
  <p>&#9658;I prefer hard rock &#9835;
    music &#9835;</p>
  <p>&copy; 2006 by Svetlin Nakov &amp; his
team</p>
  <p>Telerik Academy™</p>
```

Special Chars – Example (2)

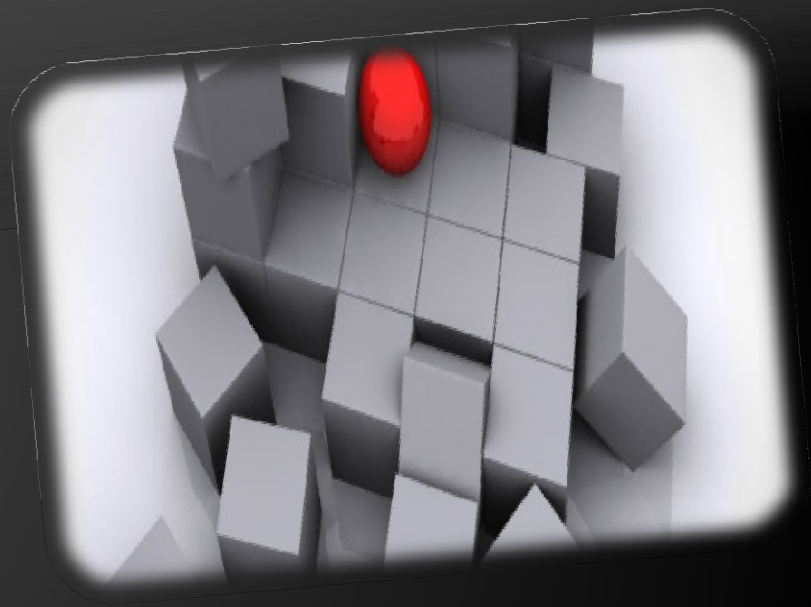
special-chars.html

```
<p>[&gt;&gt;  Welcome
      &lt;&lt;]</p>
  <p>&#9
    A&#9
  <p>&#9
    musi
  <p>&cc
team</p>
  <p>Tel
```



telerik

```
<head>
<meta http-equiv="Content-Type"
<title>Home</title>
<link rel="stylesheet" href="sty
<style type="text/css">
.style1 {
    color: #FF0000;
}
</style>
</head>

<span class="style1">You will have to p
separate license to use the OpenCube
```



Using <DIV> and <SPAN> Block and Inline Elements

telerik

Block and Inline Elements

- Block elements add a line break before and after them
  - `<div>` is a block element
  - Other block elements are `<table>`, `<hr>`, headings, lists, `<p>` and etc.
- Inline elements don't break the text before and after them
  - `<span>` is an inline element
  - Most HTML elements are inline, e.g. `<a>`

telerik

The Tag

- creates logical divisions within a page
- Block style element
- Used with CSS
- Example:

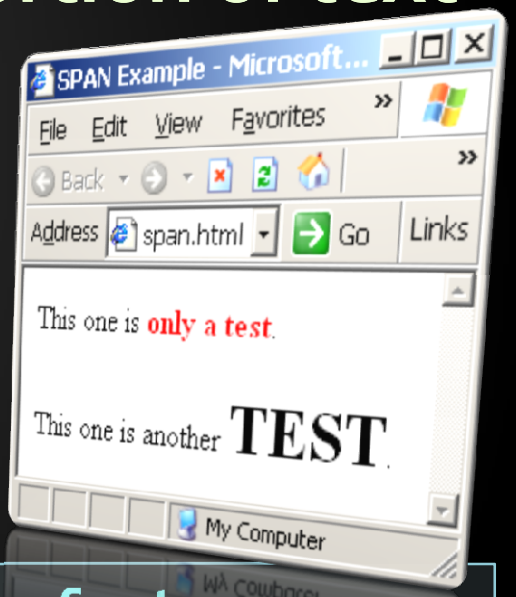
div-and-span.html

```
<div style="font-size:24px; color:red">DIV
example</div>

<p>This one is <span style="color:red; font-
weight:bold">only a test</span>.</p>
```

telerik

The Tag

- Inline style element
- Useful for modifying a specific portion of text
  - Don't create a separate area (paragraph) in the document
- Very useful with CSS

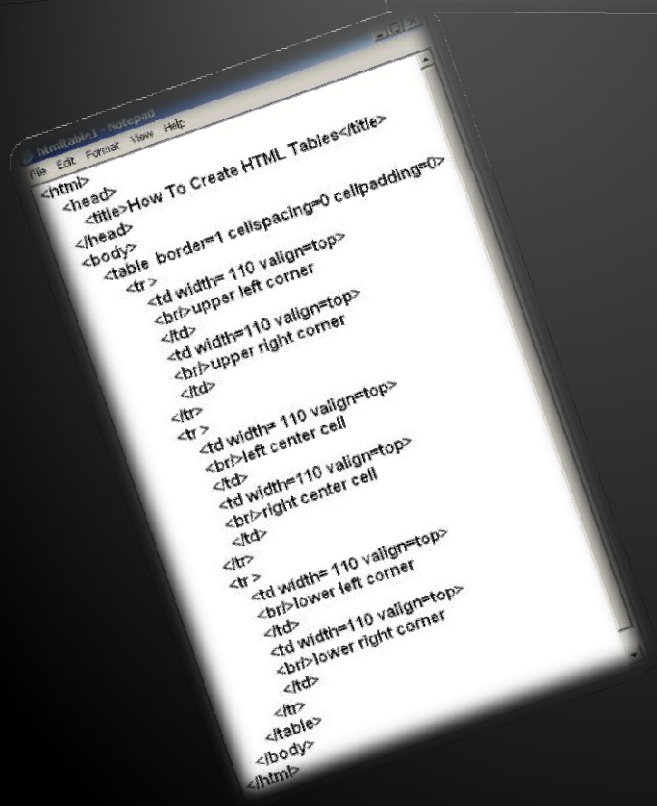


span.html

```
<p>This one is <span style="color:red; font-
weight:bold">only a test</span>.</p>

<p>This one is another <span style="font-size:32px;
font-weight:bold">TEST</span>.</p>
```

telerik

US time	European date (D/M/Y) & time	Y-M-D date & time	Dollar	Chinese money	IP addresses	Names	Numbers
	29/10/1965	83-03-24		YMB 4	98.176.35.80		26.32 E +03
Fri Mar 22 21:48:49 UTC+0200 1957		1967-08-22 06:07:16 PM		YMB -81.38	162.117.253.34	dyse chidi	
Thu, 14 Feb 2002 04:24:20 UTC	06/07/99 06:45:01 AM	81-02-04 09:09:54 AM		YMB -108.83	122.205.50.6	bochai dychai	-191.45E-05
Monday, May 30, 1994 4:47:31 PM	06/09/05 05:11:16 AM			YMB 33.16		dydy balie	-131.20E+01
09/28/2000	24/11/1957		$-38.77	YMB 112.42	15.192.151.209		
		97-08-13 00:01:33 AM	$14.5	YMB -1.75	99.93.147.150	dychai tonchai	-187.28E-05
Mon, 29 Oct 1979 00:44:03 UTC		87-10-16	$14.66	YMB 61.14		chite malie	- 125.19 E -03
Sat, 9 Jan 1982 05:45:06 UTC	04/06/68	74-10-20	$20.47		121.169.225.22	dyma bama	138.11E+02
04/05/75		2000-03-20	$68.84	YMB 88.19	239.133.227.68	made liete	195.44 E +03
Monday, July 15, 2002 1:05:02 AM	01/02/1961 09:40:16 AM		$97.9	YMB 44.28	223.66.228.116	mava sete	-107
this is footer	row	number	ONE!	adsf	adsf	adsf	adsf



HTML Tables

Title	Title	Title	Title	Title	Title
Data	Data	Data	Data	Data	Data
Data	Data	Data	Data	Data	Data
Data	Data	Data	Data	Data	Data
Data	Data	Data	Data	Data	Data
Data	Data	Data	Data	Data	Data

telerik

HTML Tables

- Tables represent tabular data
  - A table consists of one or several rows
  - Each row has one or more columns
- Tables comprised of several core tags:
  `<table></table>`: begin / end the table
  `<tr></tr>`: create a table row
  `<td></td>`: create tabular data (cell)
- Tables should not be used for layout. Use CSS floats and positioning styles instead

telerik

HTML Tables (2)

- Start and end of a table

```
<table> ... </table>
```

- Start and end of a row

```
<tr> ... </tr>
```

- Start and end of a cell in a row

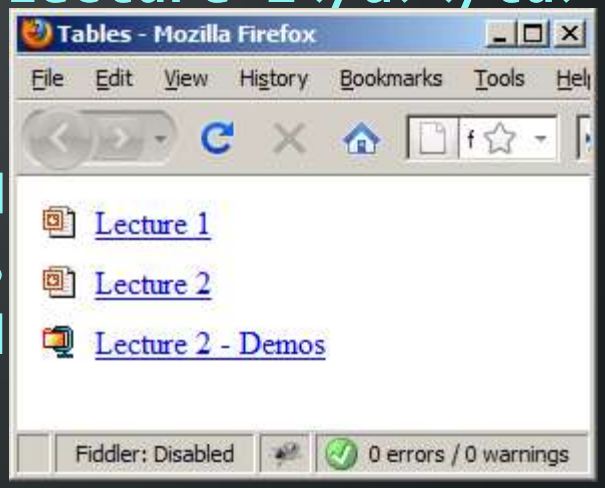
```
<td> ... </td>
```

Simple HTML Tables – Example

```
<table cellspacing="0" cellpadding="5">
  <tr>
    <td><img src="ppt.gif"></td>
    <td><a href="lecture1.ppt">Lecture 1</a></td>
  </tr>
  <tr>
    <td><img src="ppt.gif"></td>
    <td><a href="lecture2.ppt">Lecture 2</a></td>
  </tr>
  <tr>
    <td><img src="zip.gif"></td>
    <td><a href="lecture2-demos.zip">
      Lecture 2 - Demos</a></td>
  </tr>
</table>
```

Simple HTML Tables – Example (2)

```
<table cellspacing="0" cellpadding="5">
  <tr>
    <td><img src="ppt.gif"></td>
    <td><a href="lecture1.ppt">Lecture 1</a></td>
  </tr>
  <tr>
    <td><img src="ppt.gif"></td>
    <td><a href="lecture2.ppt">Lecture 2</a></td>
  </tr>
  <tr>
    <td><img src="zip.gif"></td
    <td><a href="lecture2-demos
      Lecture 2 - Demos</a></td
  </tr>
</table>
```



telerik

Complete HTML Tables

- Table rows split into three semantic sections: header, body and footer
  - `<thead>` denotes table header and contains `<th>` elements, instead of `<td>` elements
  - `<tbody>` denotes collection of table rows that contain the very data
  - `<tfoot>` denotes table footer but comes BEFORE the `<tbody>` tag
  - `<colgroup>` and `<col>` define columns (most often used to set column widths)

Complete HTML Table: Example

```
<table>
<colgroup>
  <col style="width:100px" /><col />
</colgroup>
<thead>
  <tr><th>Column 1</th><th>Column 2</th></tr>
</thead>
<tfoot>
  <tr><td>Footer 1</td><td>Footer 2</td></tr>
</tfoot>
<tbody>
  <tr><td>Cell 1.1</td><td>Cell 1.2</td></tr>
  <tr><td>Cell 2.1</td><td>Cell 2.2</td></tr>
</tbody>
</table>
```

columns

header

th

footer

Last comes the body (data)

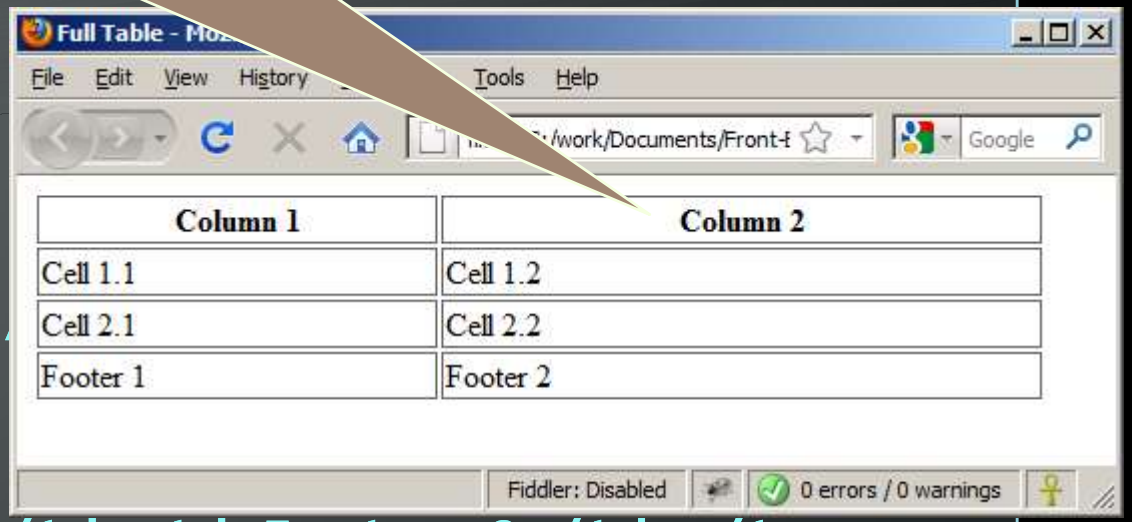
Complete HTML Table: Example (2)

telerik

table-full.html

```
<table>
<colgroup>
  <col style="width
</colgroup>
<thead>
  <tr><th>Column 1</
</thead>
<tfoot>
  <tr><td>Footer 1</td><td>Footer 2</td></tr>
</tfoot>
<tbody>
  <tr><td>Ce                                    d></tr>
  <tr><td>Ce                                    d></tr>
</tbody>
</table>
```

By default, header text is bold and centered.



Although the footer is before the data in the code, it is displayed last

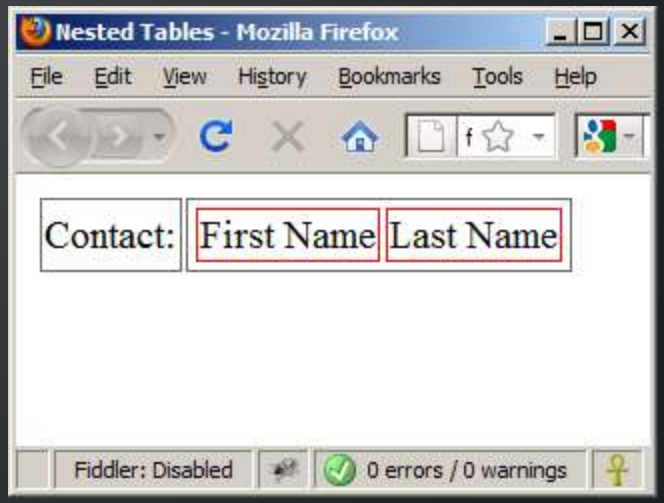
telerik

Nested Tables

- Table data "cells" (<td>) can contain nested tables (tables within tables):

nested-tables.html

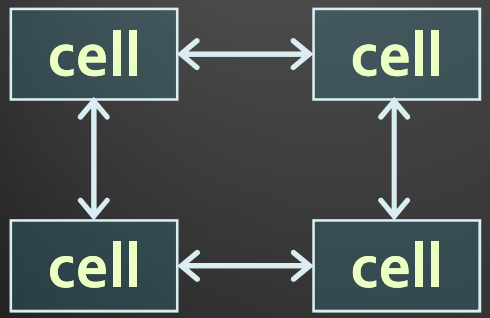
```
<table>
  <tr>
    <td>Contact:</td>
    <td>
      <table>
        <tr>
          <td>First Name</td>
          <td>Last Name</td>
        </tr>
      </table>
    </td>
  </tr>
</table>
```



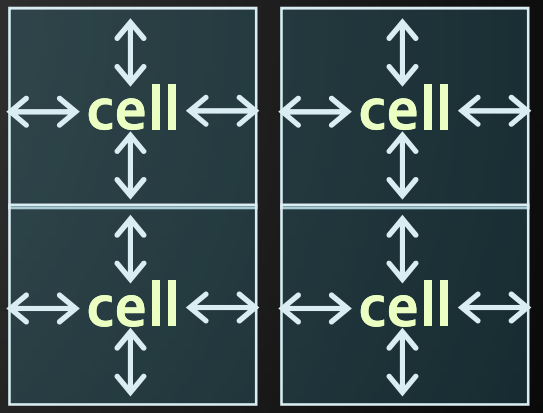
telerik

Cell Spacing and Padding

- Tables have two important attributes:
  - `cellspacing`
    - Defines the empty space between cells



  - `cellpadding`
    - Defines the empty space around the cell content



telerik

Cell Spacing and Padding – Example

table-cells.html

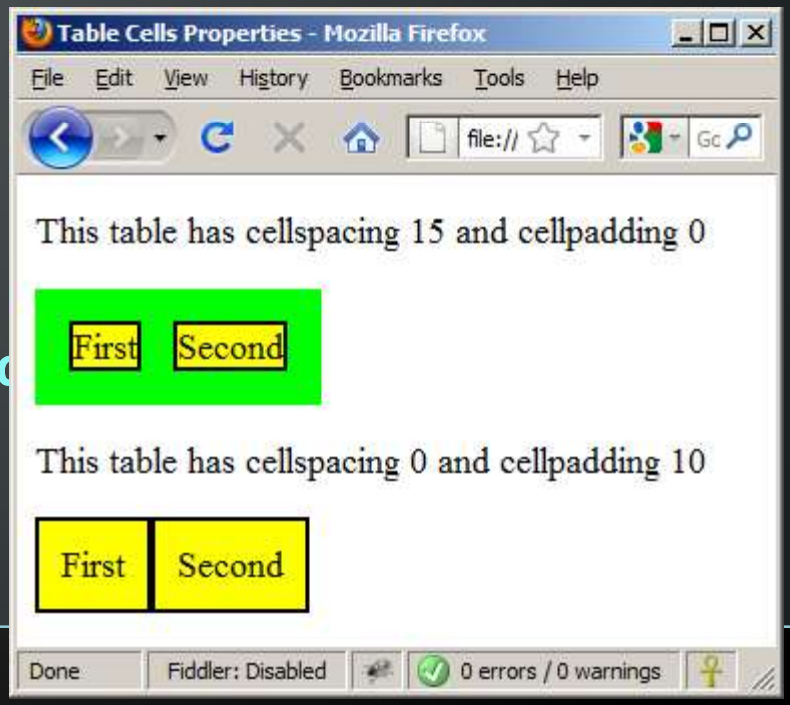
```
<html>
  <head><title>Table Cells</title></head>
  <body>
    <table cellspacing="15" cellpadding="0">
      <tr><td>First</td>
      <td>Second</td></tr>
    </table>
    <br/>
    <table cellspacing="0" cellpadding="10">
      <tr><td>First</td><td>Second</td></tr>
    </table>
  </body>
</html>
```

telerik

Cell Spacing and Padding – Example (2)

table-cells.html

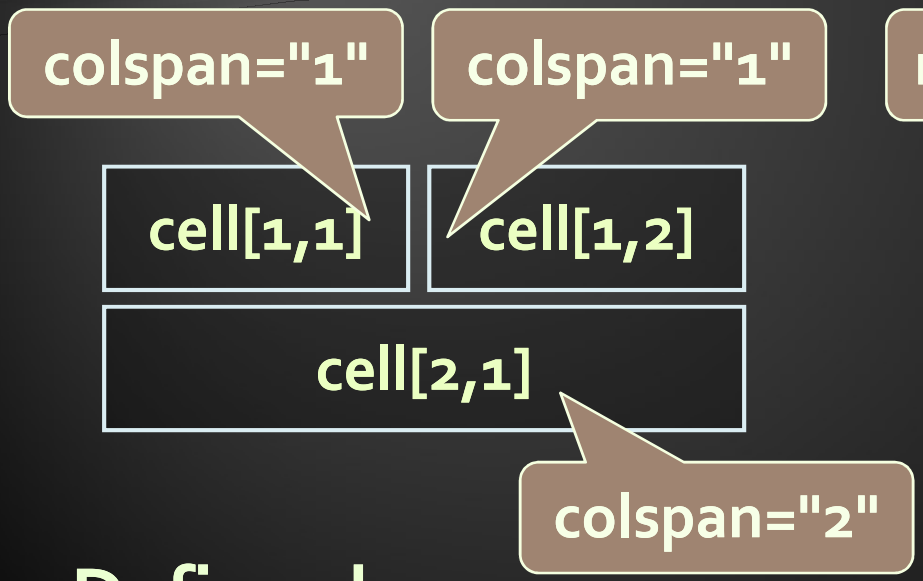
```
<html>
  <head><title>Table Cells</title></head>
  <body>
    <table cellspacing="15" cellpadding="0">
      <tr><td>First</td>
      <td>Second</td></tr>
    </table>
    <br/>
    <table cellspacing="0"
      <tr><td>First</td><td
    </table>
  </body>
</html>
```

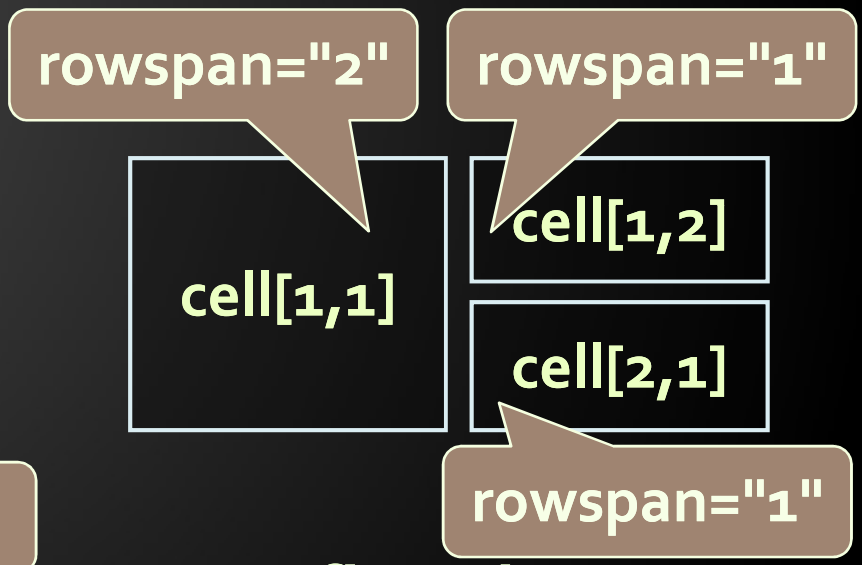


telerik

Column and Row Span

- Table cells have two important attributes:
  - colspan
  - rowspan





- Defines how many columns the cell occupies
- Defines how many rows the cell occupies

telerik

Column and Row Span – Example

table-colspan-rowspan.html

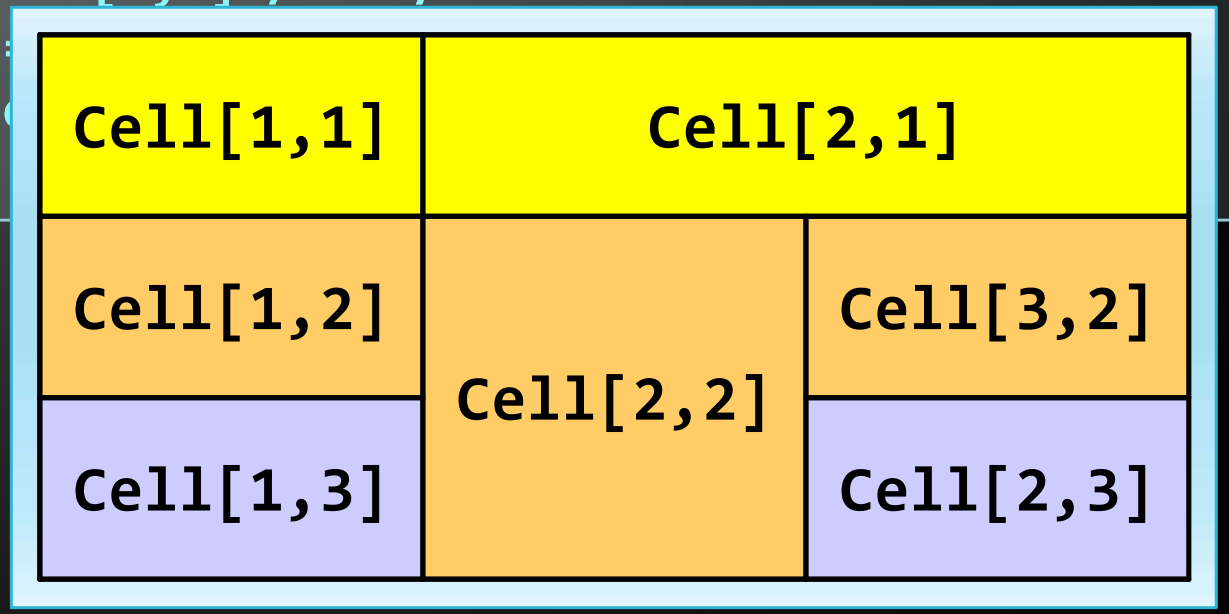
```
<table cellspacing="0">
    <tr class="1"><td>Cell[1,1]</td>
        <td colspan="2">Cell[2,1]</td></tr>
    <tr class=“2"><td>Cell[1,2]</td>
        <td rowspan="2">Cell[2,2]</td>
        <td>Cell[3,2]</td></tr>
    <tr class=“3"><td>Cell[1,3]</td>
        <td>Cell[2,3]</td></tr>
</table>
```

telerik

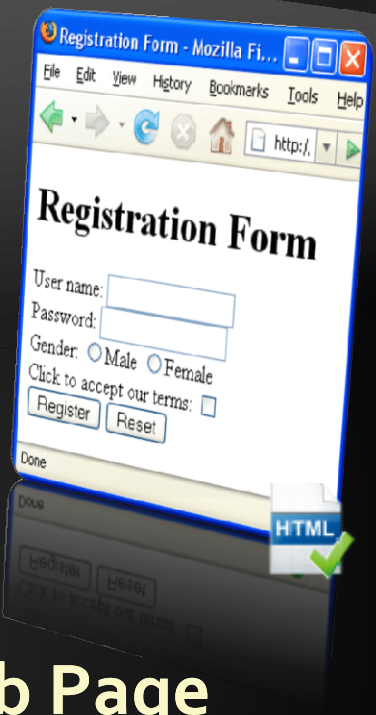
Column and Row Span – Example (2)

table-colspan-rowspan.html

```
<table cellspacing="0">
    <tr class="1"><td>Cell[1,1]</td>
        <td colspan="2">Cell[2,1]</td></tr>
    <tr class="2"><td>Cell[1,2]</td>
        <td rowspan="2">Cell[2,2]</td>
        <td>Cell[3,2]</td></tr>
    <tr class=
        <td>C
</table>
```

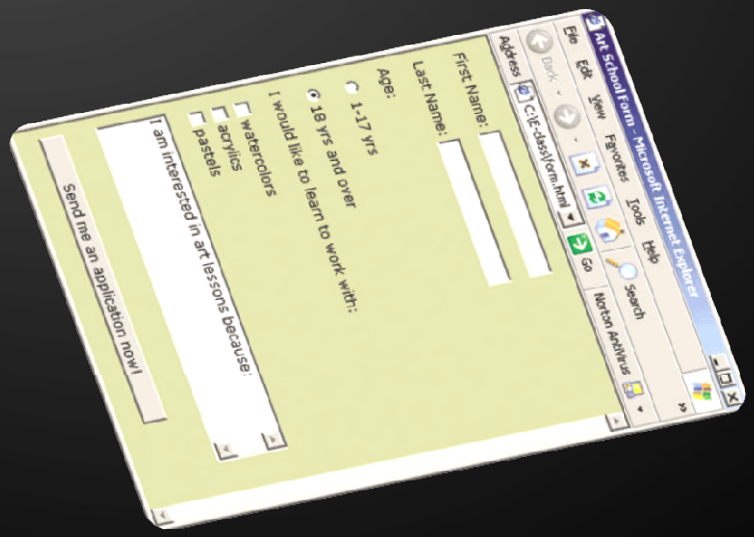


telerik



HTML Forms

Entering User Data from a Web Page



telerik

HTML Forms

- Forms are the primary method for gathering data from site visitors
- Create a form block with

```
<form></form>
```

The "method" attribute tells how the form data should be sent – via GET or POST request

- Example:

```
<form name="myForm" method="post"
action="path/to/some-script.php">
    ...
</form>
```

The "action" attribute tells where the form data should be sent

telerik

Form Fields

- Single-line text input fields:

```
<input type="text" name="FirstName" value="This
is a text field" />
```

- Multi-line textarea fields:

```
<textarea name="Comments">This is a multi-line
text field</textarea>
```

- Hidden fields contain data not shown to the user:

```
<input type="hidden" name="Account" value="This
is a hidden text field" />
```

  - Often used by JavaScript code

telerik

Fieldsets

- Fieldsets are used to enclose a group of related form fields:

```
<form method="post" action="form.aspx">
    <fieldset>
        <legend>Client Details</legend>
        <input type="text" id="Name" />
        <input type="text" id="Phone" />
    </fieldset>
    <fieldset>
        <legend>Order Details</legend>
        <input type="text" id="Quantity" />
        <textarea cols="40" rows="10"
            id="Remarks"></textarea>
    </fieldset>
</form>
```

- The <legend> is the fieldset's title.

telerik

Form Input Controls

- Checkboxes:

```
<input type="checkbox" name="fruit"
value="apple" />
```

- Radio buttons:

```
<input type="radio" name="title" value="Mr." />
```

- Radio buttons can be grouped, allowing only one to be selected from a group:

```
<input type="radio" name="city" value="Lom" />
<input type="radio" name="city" value="Ruse" />
```

telerik

Other Form Controls

- Dropdown menus:

```
<select name="gender">
  <option value="Value 1"
    selected="selected">Male</option>
  <option value="Value 2">Female</option>
  <option value="Value 3">Other</option>
</select>
```

- Submit button:

```
<input type="submit" name="submitBtn"
value="Apply Now" />
```

telerik

Other Form Controls (2)

- Reset button – brings the form to its initial state

```
<input type="reset" name="resetBtn"
value="Reset the form" />
```

- Image button – acts like submit but image is displayed and click coordinates are sent

```
<input type="image" src="submit.gif"
name="submitBtn" alt="Submit" />
```

- Ordinary button – used for Javascript, no default action

```
<input type="button" value="click me" />
```

telerik

Other Form Controls (3)

- Password input – a text field which masks the entered text with * signs

```
<input type="password" name="pass" />
```

- Multiple select field – displays the list of items in multiple lines, instead of one

```
<select name="products" multiple="multiple">
  <option value="Value 1"
    selected="selected">keyboard</option>
  <option value="Value 2">mouse</option>
  <option value="Value 3">speakers</option>
</select>
```

telerik

Other Form Controls (4)

- File input – a field used for uploading files

```
<input type="file" name="photo" />
```

  - When used, it requires the form element to have a specific attribute:

```
<form enctype="multipart/form-data">
...
  <input type="file" name="photo" />
...
</form>
```

telerik

Labels

- Form labels are used to associate an explanatory text to a form field using the field's ID.

```
<label for="fn">First Name</label>
<input type="text" id="fn" />
```

- Clicking on a label focuses its associated field (checkboxes are toggled, radio buttons are checked)
- Labels are both a usability and accessibility feature and are required in order to pass accessibility validation.

telerik

HTML Forms – Example

form.html

```
<form method="post" action="apply-now.php">
  <input name="subject" type="hidden" value="Class" />
  <fieldset><legend>Academic information</legend>
    <label for="degree">Degree</label>
    <select name="degree" id="degree">
      <option value="BA">Bachelor of Art</option>
      <option value="BS">Bachelor of Science</option>
      <option value="MBA" selected="selected">Master of
        Business Administration</option>
    </select>
    <br />
    <label for="studentid">Student ID</label>
    <input type="password" name="studentid" />
  </fieldset>
  <fieldset><legend>Personal Details</legend>
    <label for="fname">First Name</label>
    <input type="text" name="fname" id="fname" />
    <br />
    <label for="lname">Last Name</label>
    <input type="text" name="lname" id="lname" />
```

telerik

HTML Forms – Example (2)

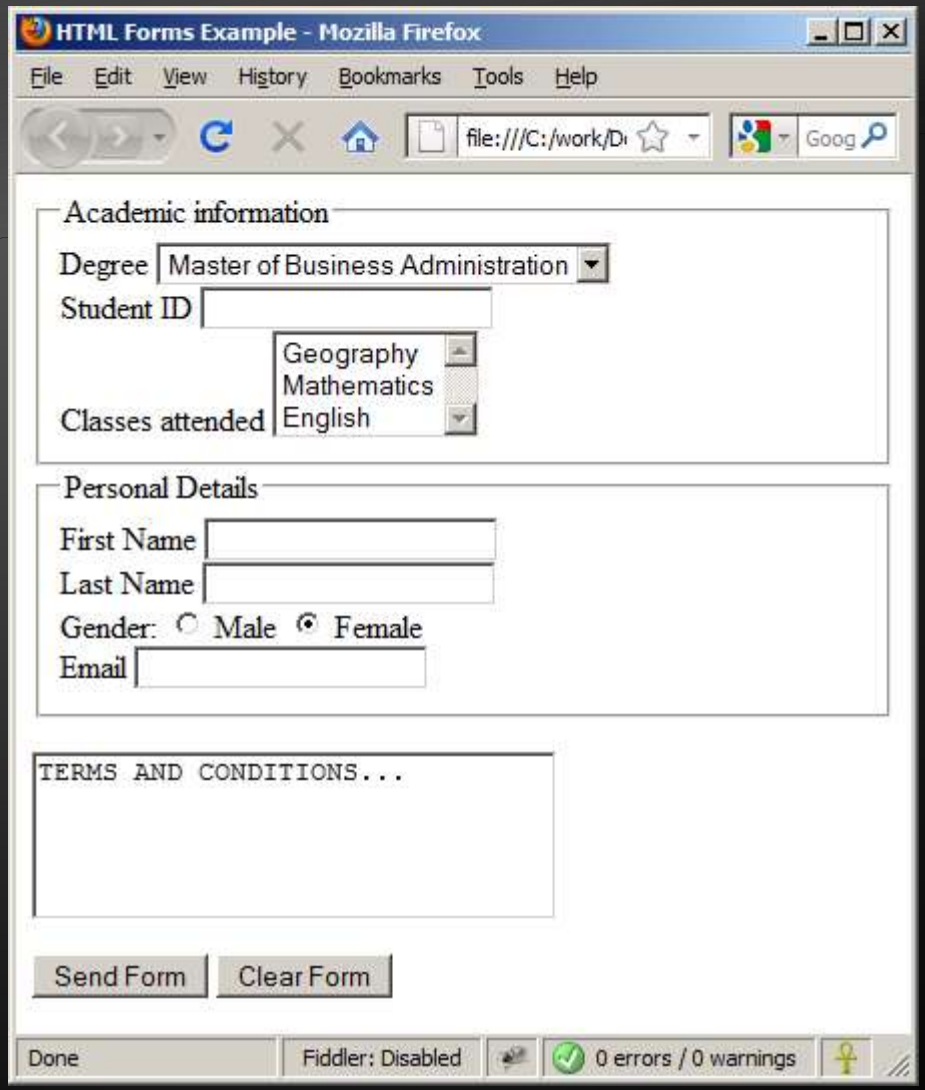
form.html (continued)

```
    <br />
      Gender:
      <input name="gender" type="radio" id="gm" value="m" />
      <label for="gm">Male</label>
      <input name="gender" type="radio" id="gf" value="f" />
      <label for="gf">Female</label>
    <br />
     <label for="email">Email</label>
     <input type="text" name="email" id="email" />
 </fieldset>
    <p>
     <textarea name="terms" cols="30" rows="4"
       readonly="readonly">TERMS AND CONDITIONS...</textarea>
  </p>
  <p>
     <input type="submit" name="submit" value="Send Form" />
     <input type="reset" value="Clear Form" />
  </p>
</form>
```

telerik

HTML Forms – Example (3)

form.html (continued)



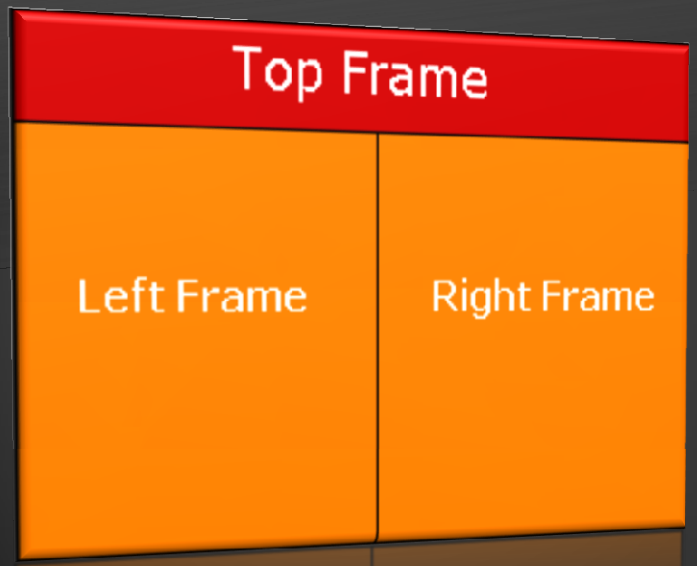
telerik

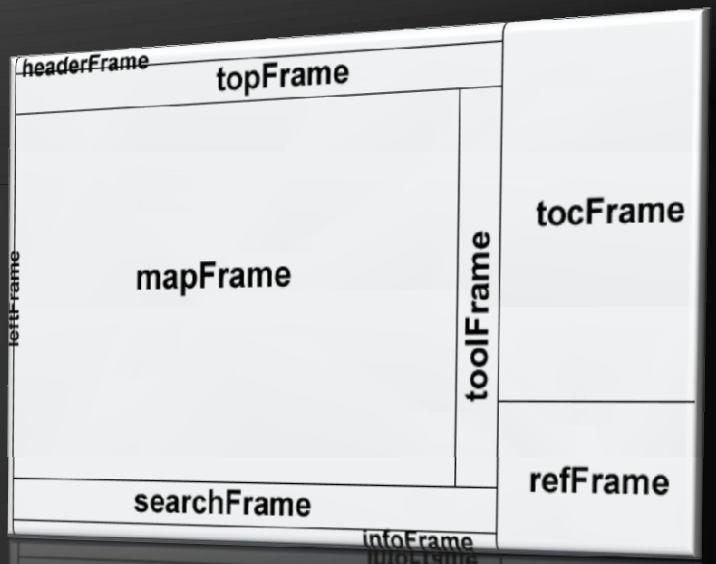
TabIndex

- The tabindex HTML attribute controls the order in which form fields and hyperlinks are focused when repeatedly pressing the TAB key
  - tabindex="0" (zero) - "natural" order
  - If X > Y, then elements with tabindex="X" are iterated before elements with tabindex="Y"
  - Elements with negative tabindex are skipped, however, this is not defined in the standard

```
<input type="text" tabindex="10" />
```

telerik





HTML Frames

<frameset>, <frame> and <iframe>

telerik

HTML Frames

- Frames provide a way to show multiple HTML documents in a single Web page
- The page can be split into separate views (frames) horizontally and vertically
- Frames were popular in the early ages of HTML development, but now their usage is rejected
- Frames are not supported by all user agents (browsers, search engines, etc.)
  - A <noframes> element is used to provide content for non-compatible agents.

telerik

HTML Frames – Demo

frames.html

```
<html>
  <head><title>Frames Example</title></head>
  <frameset cols="180px,*,150px">
    <frame src="left.html" />
    <frame src="middle.html" />
    <frame src="right.html" />
  </frameset>
</html>
```

- Note the target attribute applied to the <a> elements in the left frame.

telerik

Inline Frames: <iframe>

- Inline frames provide a way to show one website inside another website:

iframe-demo.html

```
<iframe name="iframeGoogle" width="600" height="400"
src="http://www.google.com" frameborder="yes"
scrolling="yes"></iframe>
```

telerik

Cascading Style Sheets (CSS)

```
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; }
```

telerik

Table of Contents

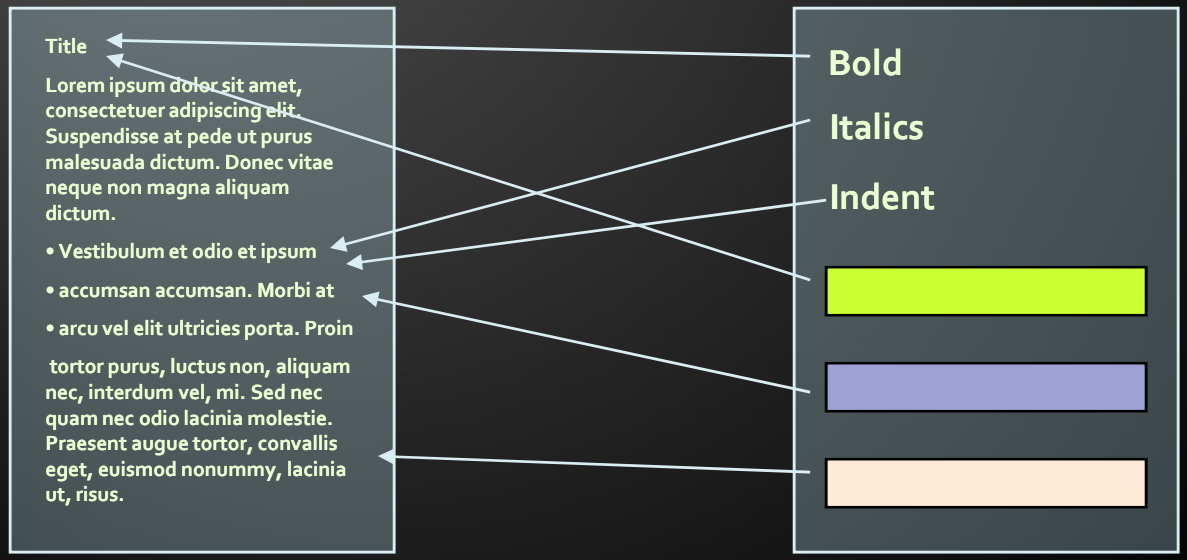
telerik

CSS: A New Philosophy

- Separate content from presentation!

Content (HTML document)

Presentation (CSS Document)



telerik

The Resulting Page

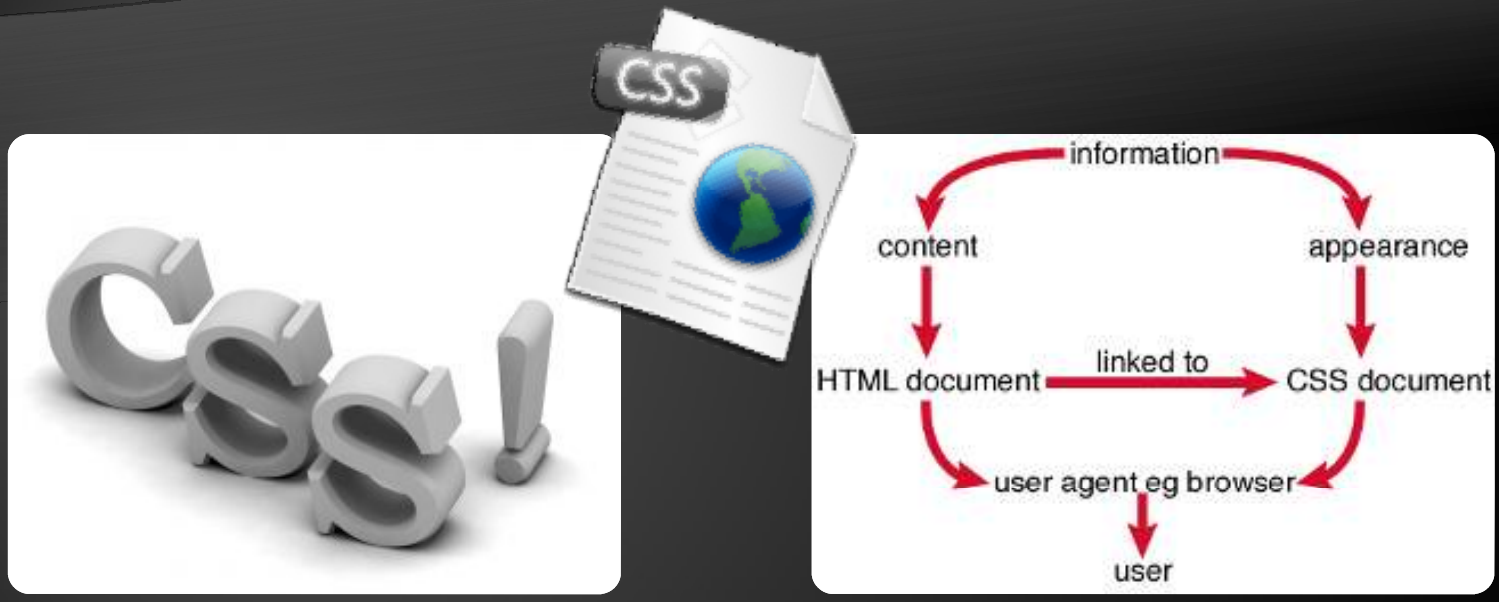
Title

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.

telerik



CSS Intro

Styling with Cascading Stylesheets

telerik

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.

telerik

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

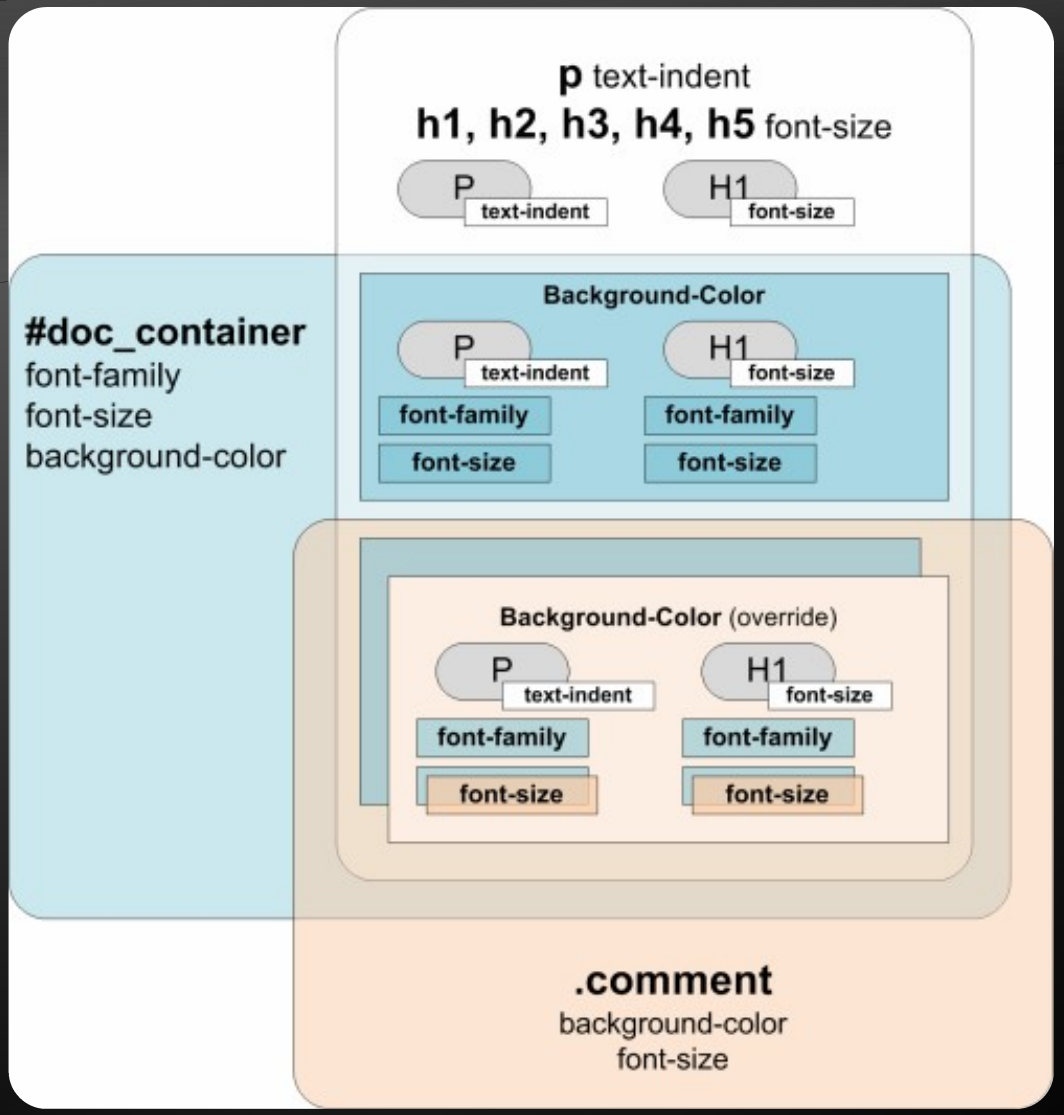
telerik

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

telerik

Why "Cascading"? (2)



telerik

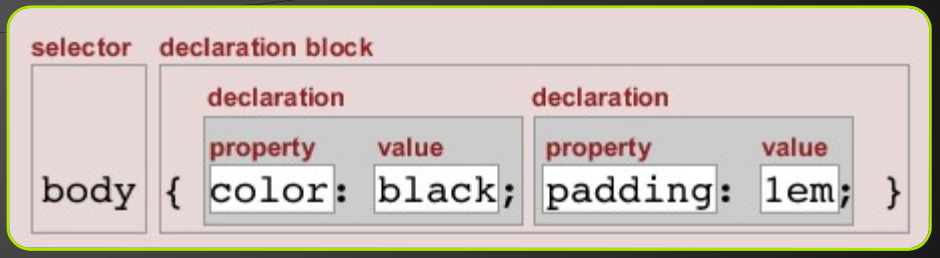
Why "Cascading"? (3)

- 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

telerik

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; }
```

telerik

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 }
```

telerik

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`, and element with id `top-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: "«"; }
```

telerik

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

telerik

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 both (all) classes applied at the same time

telerik

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;`

telerik

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; }
```

telerik

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

telerik

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

telerik

Inline Styles: Example

inline-styles.html

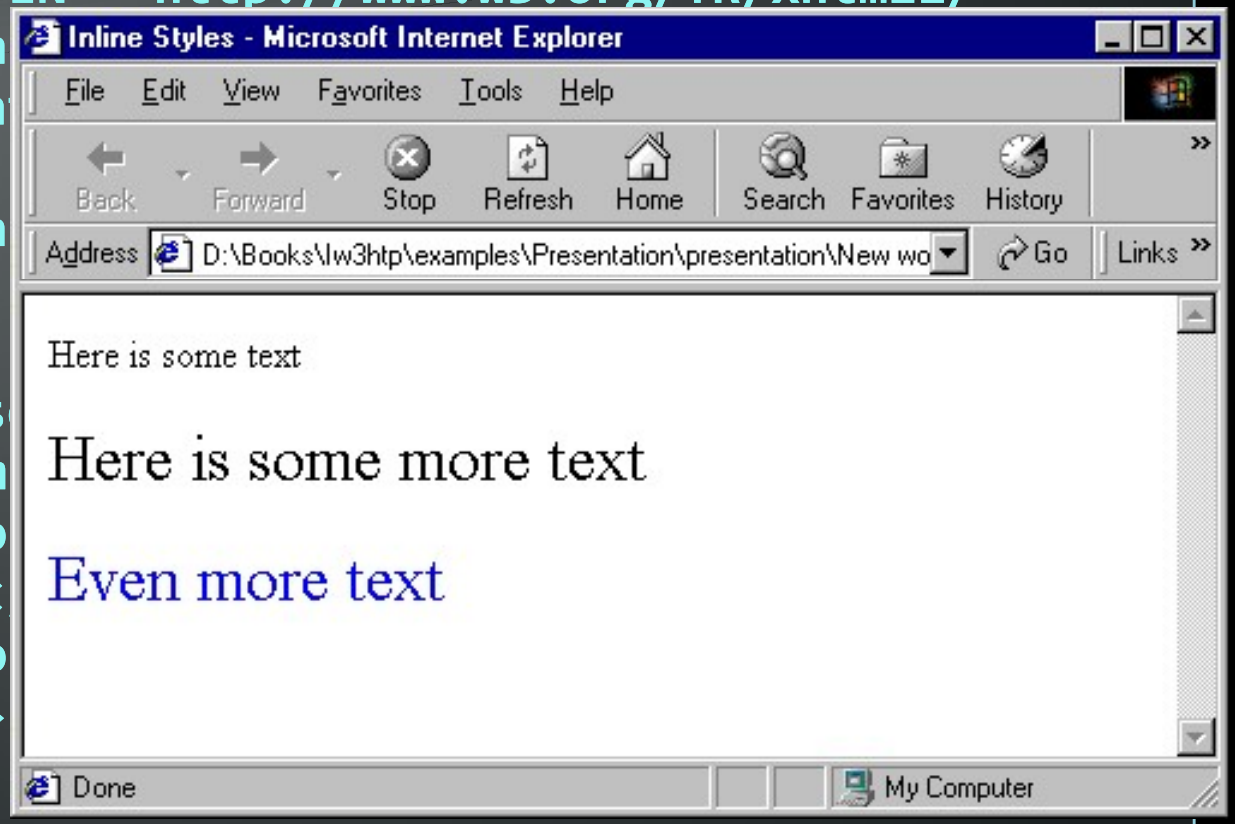
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/
DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Inline Styles</title>
</head>
<body>
  <p>Here is some text</p>
<!--Separate multiple styles with a semicolon-->
  <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>
```



telerik

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

telerik

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/
  - http://css.maxdesign.com.au/selectutorial/advanced_conflict.htm

telerik

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

telerik

Embedded Styles: Example

embedded-stylesheets.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>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>
```

telerik

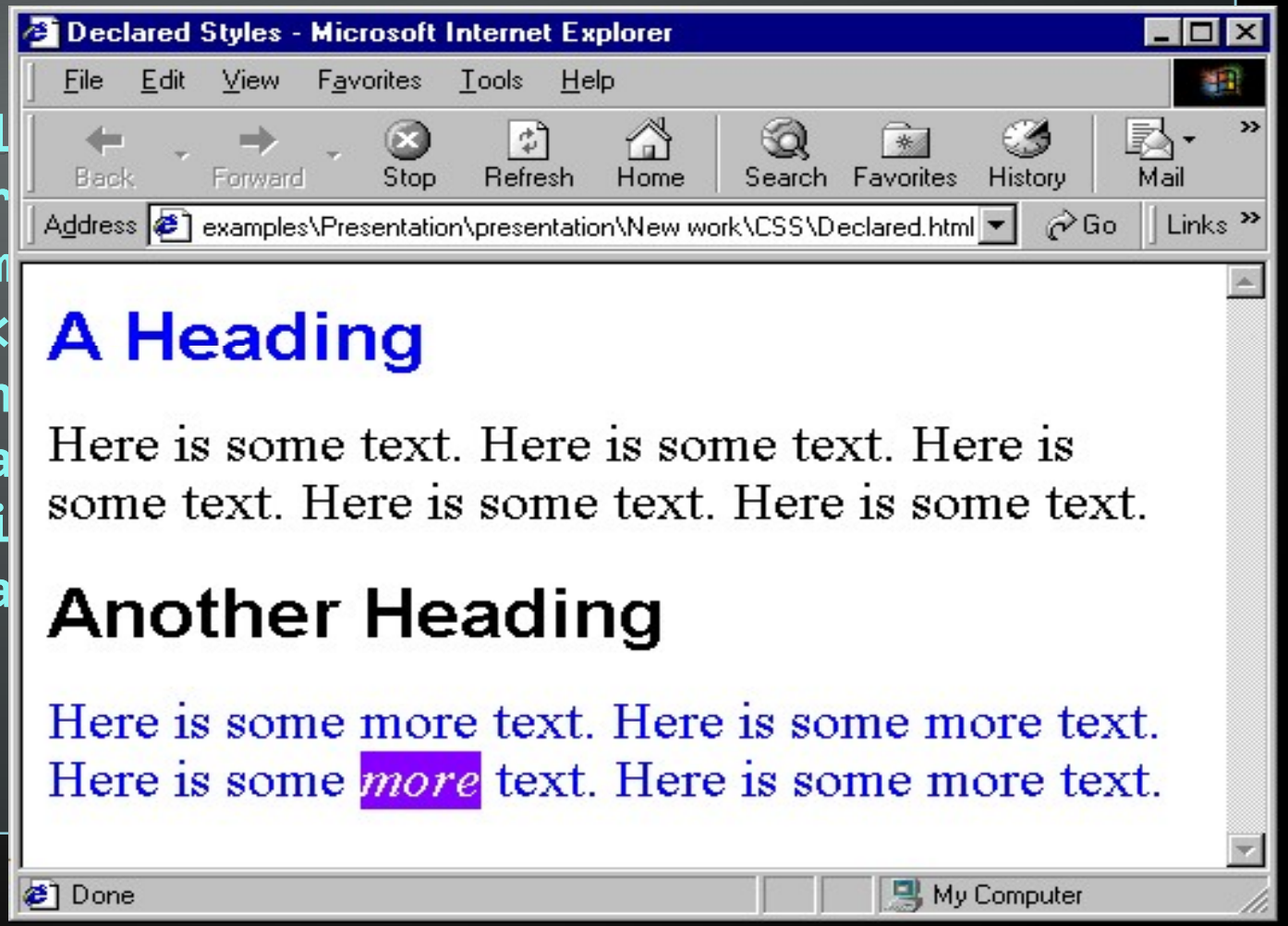
Embedded Styles: Example (2)

```
…
<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>
```

telerik

Embedded Styles: Example (3)

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



telerik

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">
```

  - `link` elements should be in the `<head>`

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

telerik

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 }
```

telerik

External Styles: Example (2)

external-styles.html

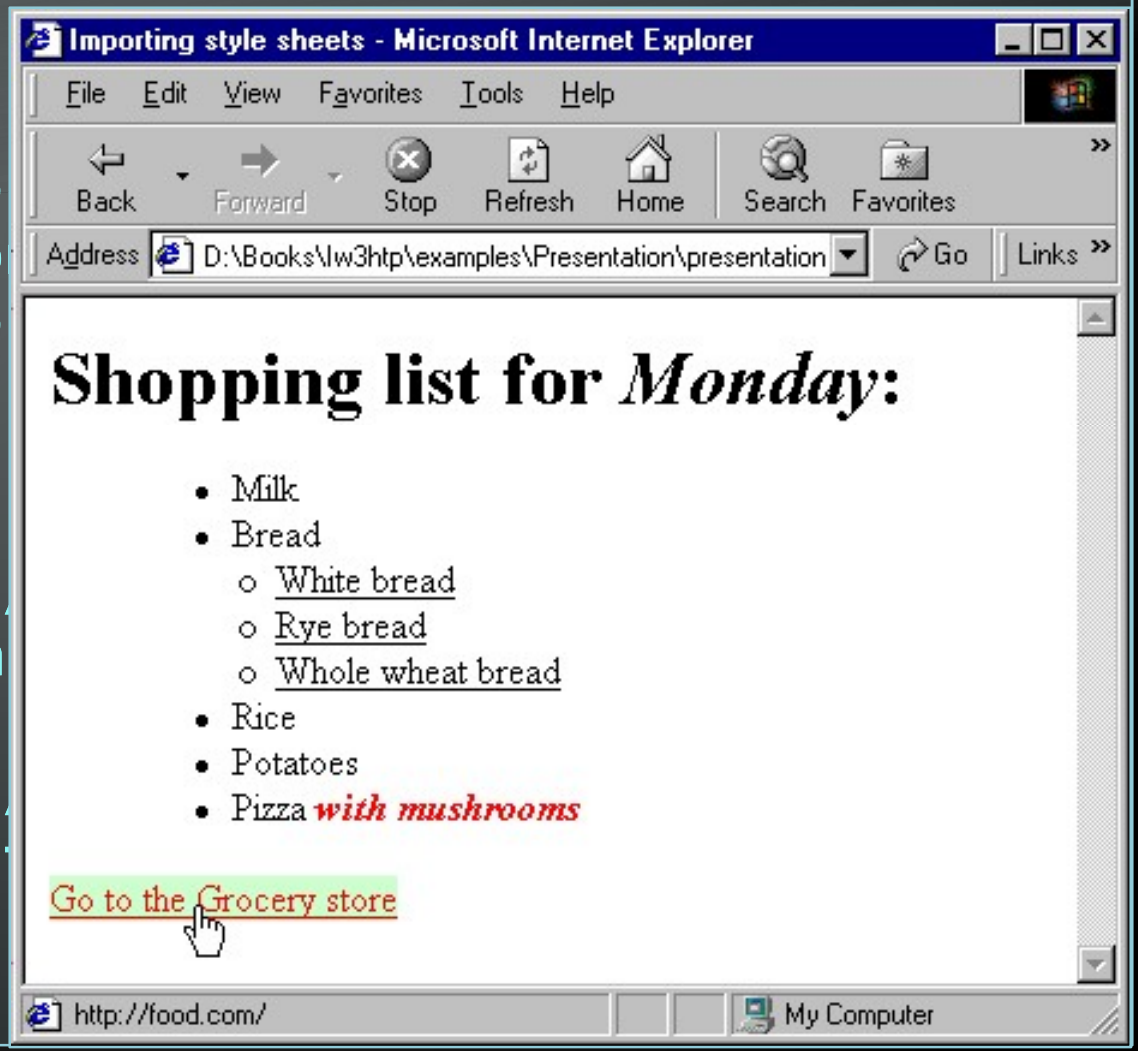
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
  Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Importing style sheets</title>
  <link type="text/css" rel="stylesheet"
    href="styles.css"  />
</head>
<body>
  <h1>Shopping list for <em>Monday</em>:</h1>
  <li>Milk</li>
  …
```

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>
```



telerik

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`]

telerik

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`

telerik

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;
```

telerik

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
  - fixed / scroll

telerik

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%;
```

telerik

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

telerik

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

telerik

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`, …

telerik

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
- When to avoid border:0

telerik

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.

telerik

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`

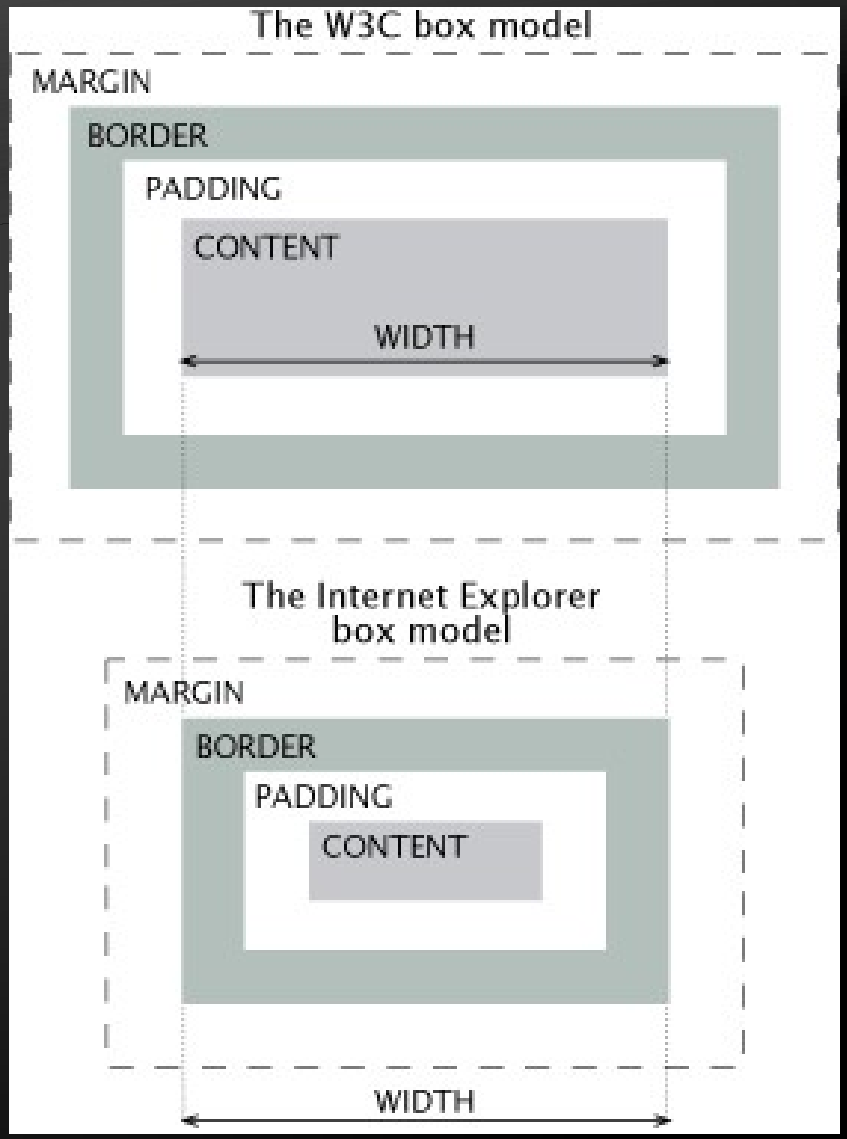
telerik

The Box Model

telerik

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



telerik

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

telerik

Positioning (2)

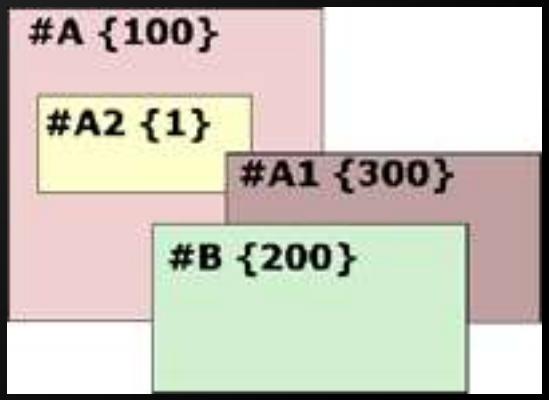
- Margin VS relative positioning
- Fixed and absolutely positioned elements do not influence the page normal flow and usually stay on top of other elements
  - Their position and size is ignored when calculating the size of parent element or position of surrounding elements
  - Overlaid according to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

telerik

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.



telerik

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)

telerik

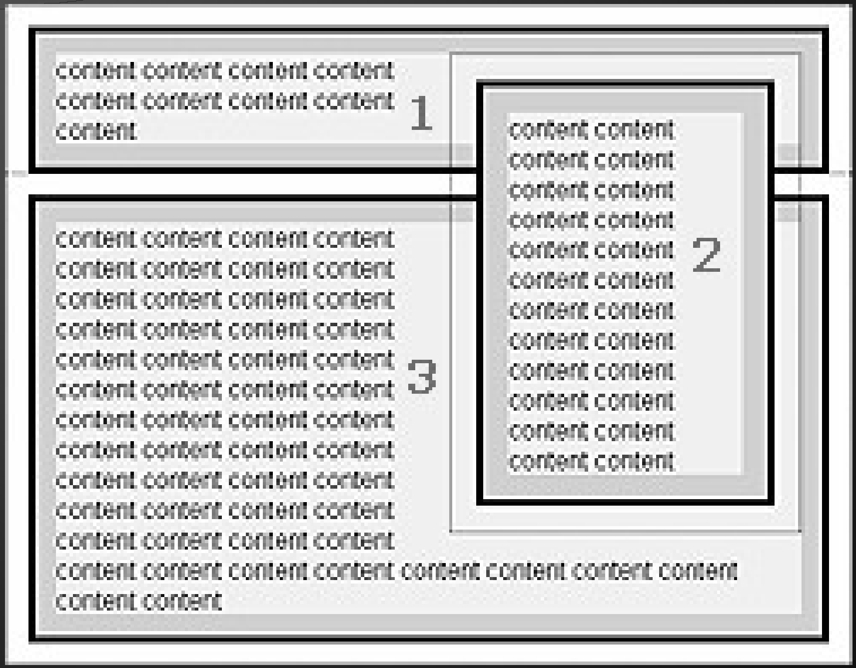
Float

- `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

telerik

Float (2)

- How floated elements are positioned



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

telerik

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;

telerik

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`

telerik

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

telerik

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)

telerik

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`

telerik

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

telerik

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

telerik

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

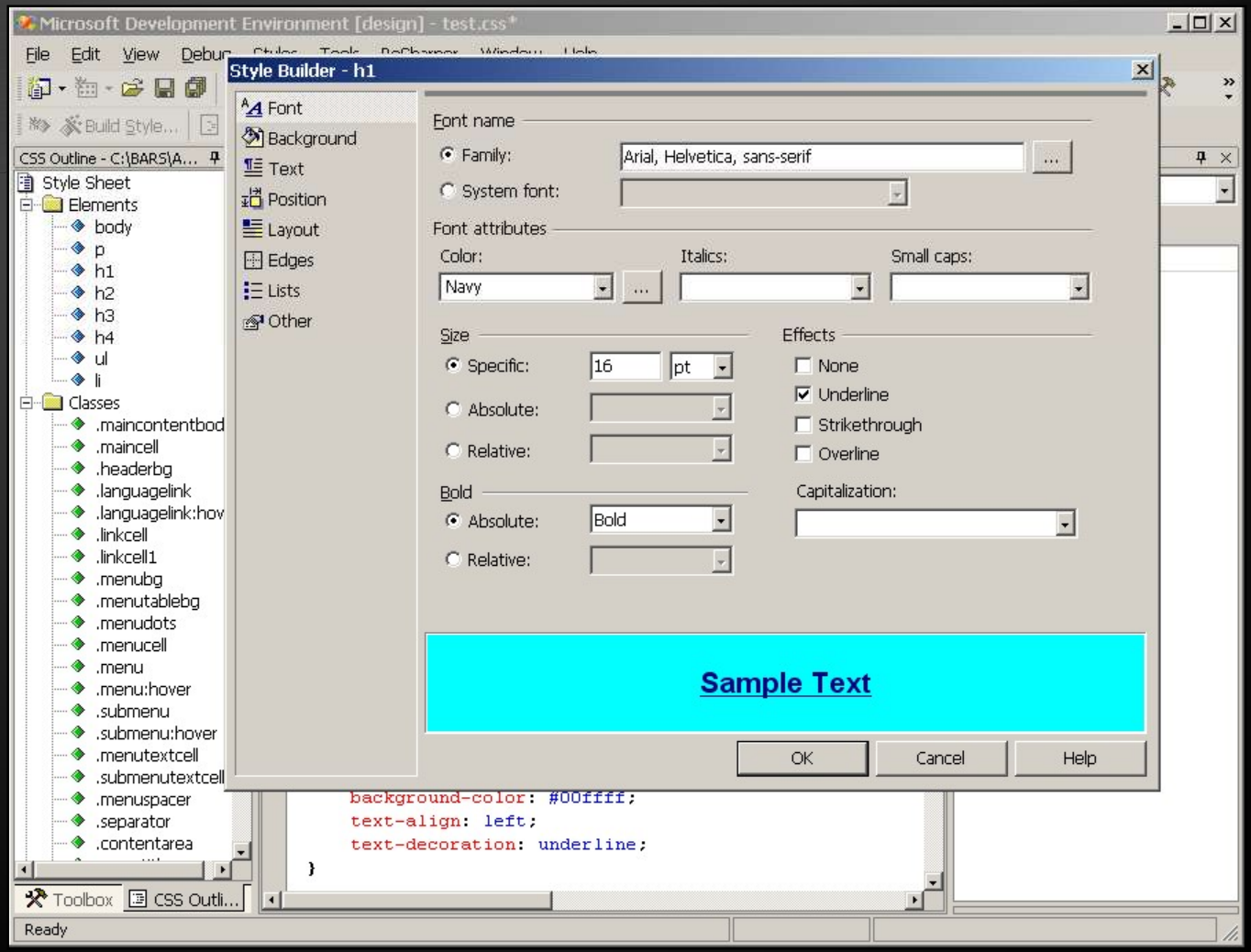
telerik

Maintenance Example

telerik

CSS Development Tools

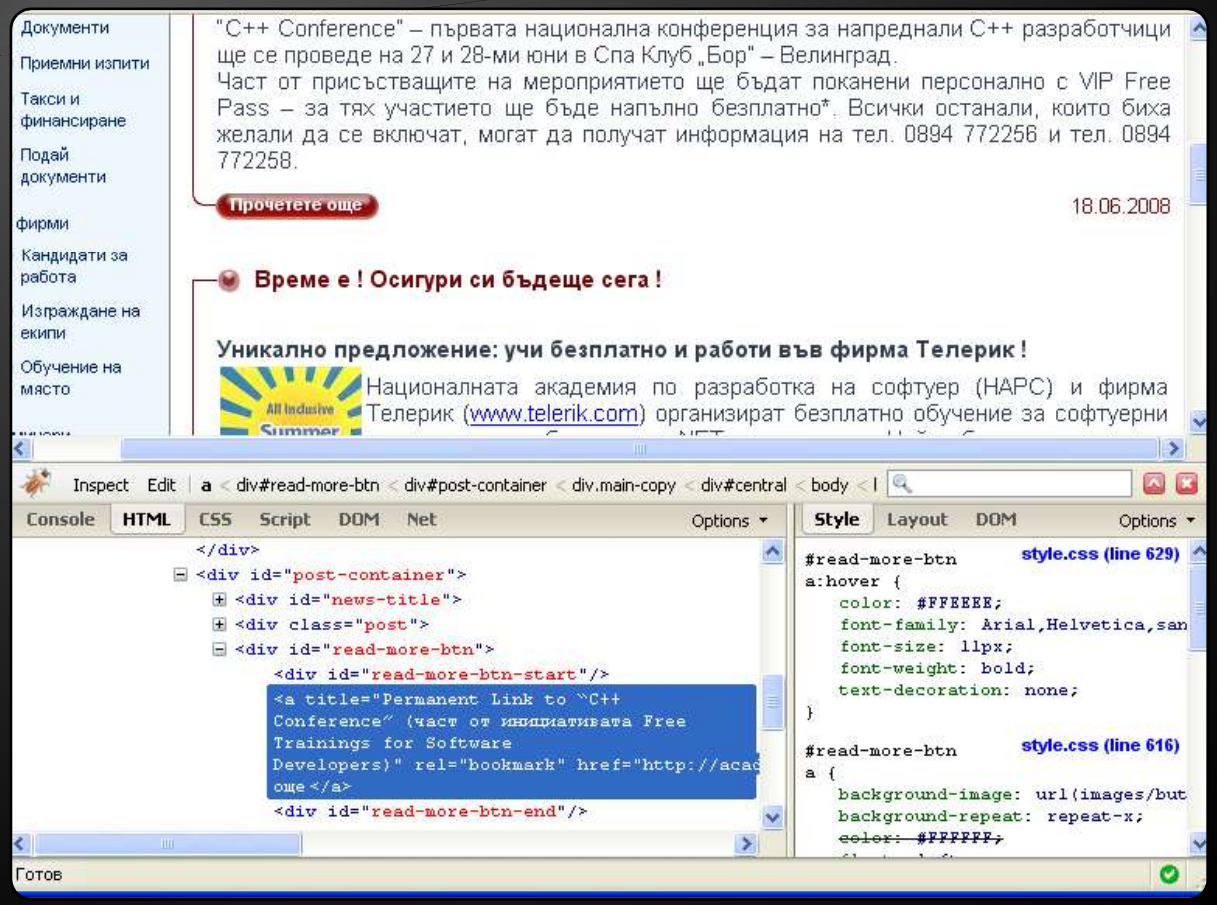
- Visual Studio – CSS Editor



telerik

CSS Development Tools (3)

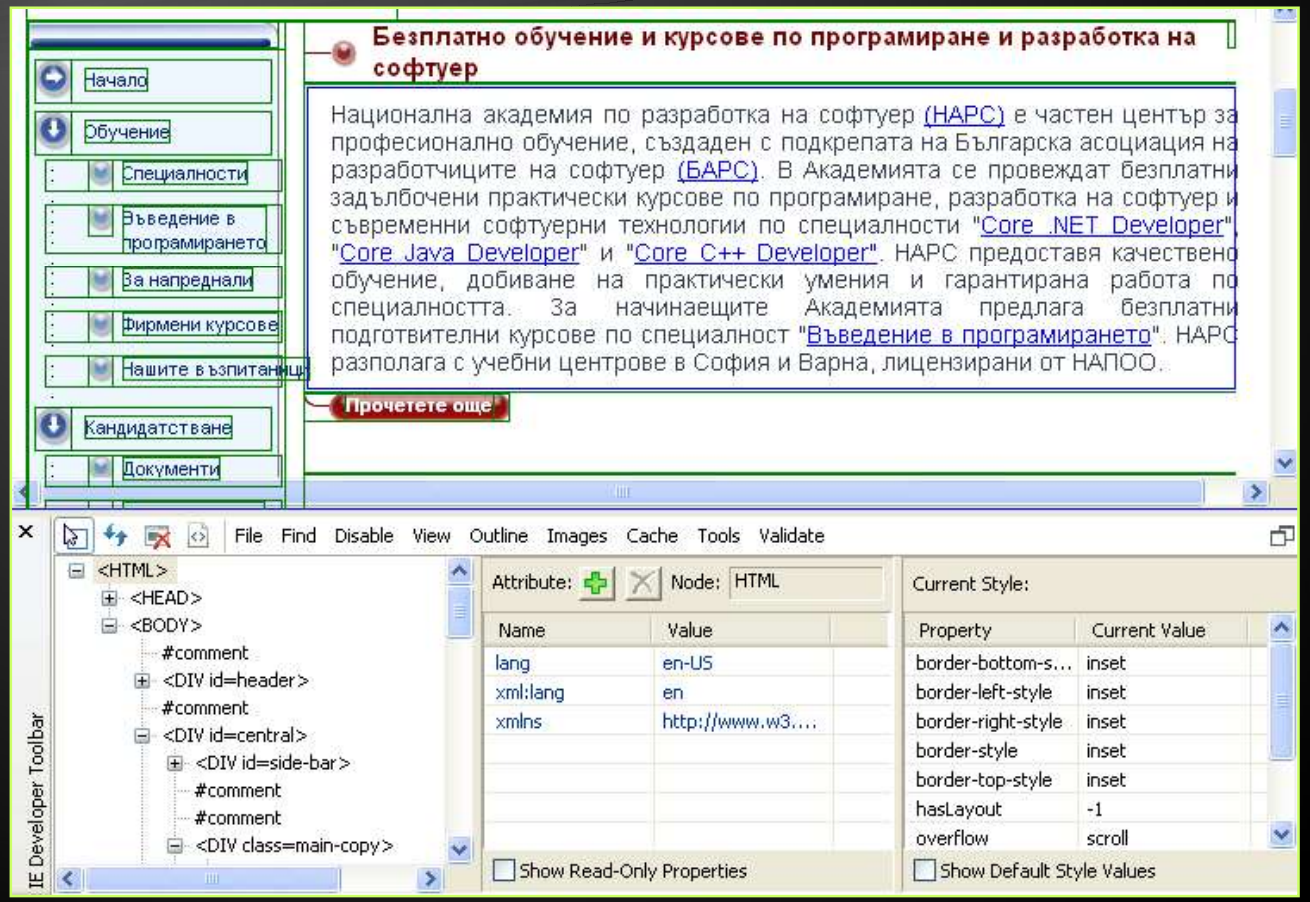
- Firebug – add-on to Firefox used to examine and adjust CSS and HTML



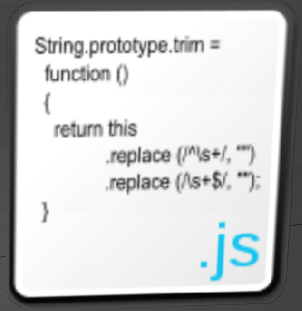
telerik

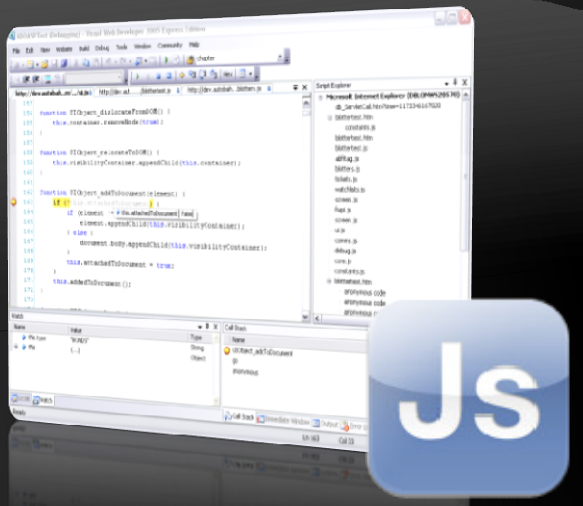
CSS Development Tools (4)

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

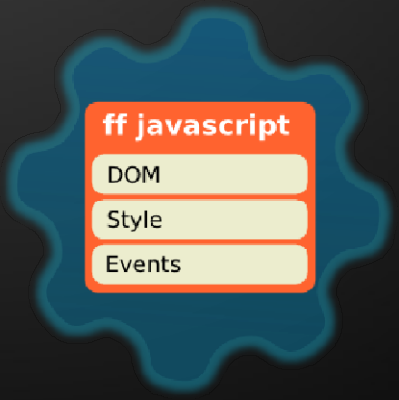


telerik





Introduction to JavaScript



telerik

Table of Contents

telerik

Table of Contents (2)

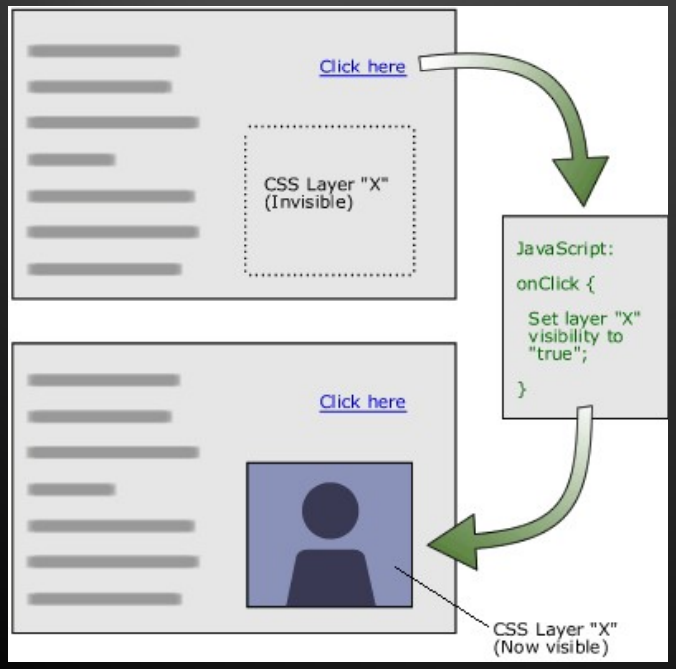
telerik

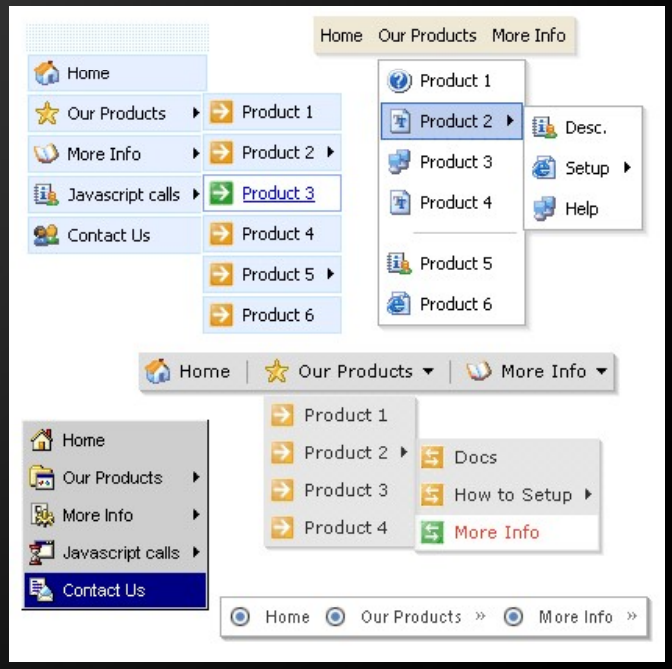
Table of Contents (3)

telerik

DHTML

Dynamic Behavior at the Client Side

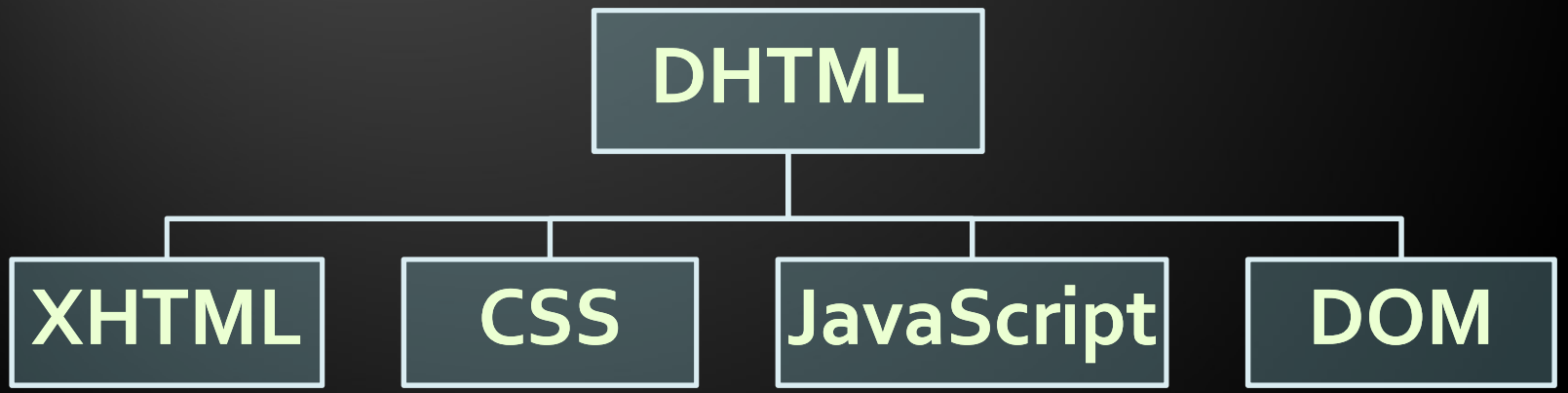




telerik

What is DHTML?

- Dynamic HTML (DHTML)
  - Makes possible a Web page to react and change in response to the user's actions
- DHTML = HTML + CSS + JavaScript



telerik

DTHML = HTML + CSS + JavaScript

- HTML defines Web sites content through semantic tags (headings, paragraphs, lists, …)
- CSS defines 'rules' or 'styles' for presenting every aspect of an HTML document
  - Font (family, size, color, weight, etc.)
  - Background (color, image, position, repeat)
  - Position and layout (of any object on the page)
- JavaScript defines dynamic behavior
  - Programming logic for interaction with the user, to handle events, etc.

telerik

JavaScript

Dynamic Behavior in a Web Page

telerik

JavaScript

- JavaScript is a front-end scripting language developed by Netscape for dynamic content
  - Lightweight, but with limited capabilities
  - Can be used as object-oriented language
- Client-side technology
  - Embedded in your HTML page
  - Interpreted by the Web browser
- Simple and flexible
- Powerful to manipulate the DOM

telerik

JavaScript Advantages

- JavaScript allows interactivity such as:
  - Implementing form validation
  - React to user actions, e.g. handle keys
  - Changing an image on moving mouse over it
  - Sections of a page appearing and disappearing
  - Content loading and changing dynamically
  - Performing complex calculations
  - Custom HTML controls, e.g. scrollable table
  - Implementing AJAX functionality

telerik

What Can JavaScript Do?

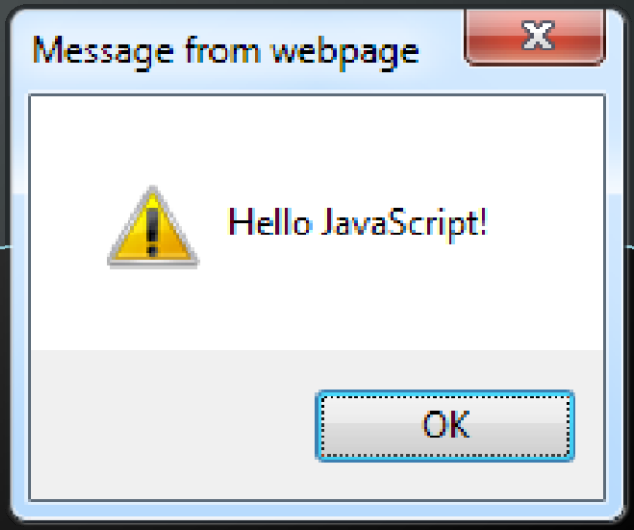
- Can handle events
- Can read and write HTML elements and modify the DOM tree
- Can validate form data
- Can access / modify browser cookies
- Can detect the user's browser and OS
- Can be used as object-oriented language
- Can handle exceptions
- Can perform asynchronous server calls (AJAX)

telerik

The First Script

first-script.html

```
<html>

<body>
  <script type="text/javascript">
     alert('Hello JavaScript!');
  </script>
</body>

</html>
```

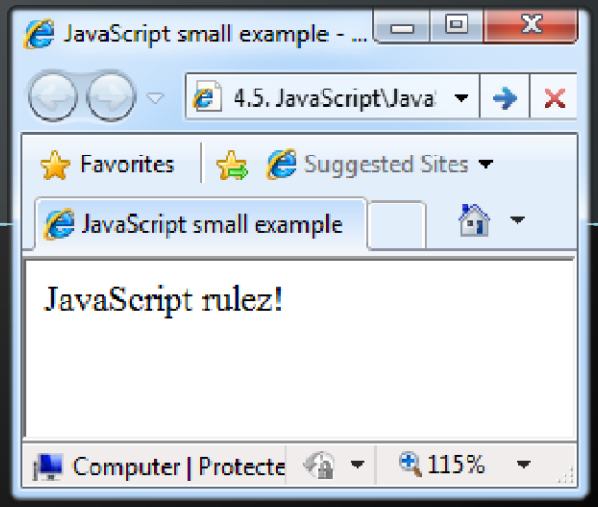


telerik

Another Small Example

small-example.html

```
<html>

<body>
  <script type="text/javascript">
     document.write('JavaScript rulez!');
  </script>
</body>

</html>
```



telerik

Using JavaScript Code

- The JavaScript code can be placed in:
  - `<script>` tag in the head
  - `<script>` tag in the body – not recommended
  - External files, linked via `<script>` tag the head
    - Files usually have `.js` extension

```
<script src="scripts.js" type="text/javscript">
<!- code placed here will not be executed! -->
</script>
```

    - Highly recommended
    - The `.js` files get cached by the browser

JavaScript – When is Executed?

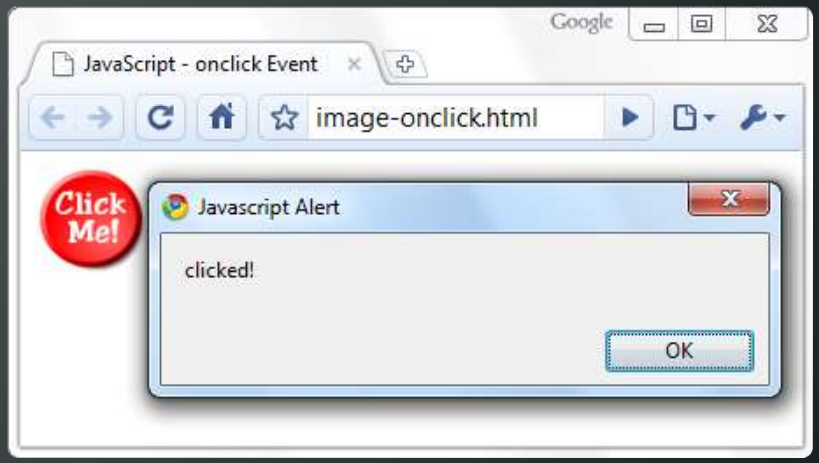
- JavaScript code is executed during the page loading or when the browser fires an event
  - All statements are executed at page loading
  - Some statements just define functions that can be called later
- Function calls or code can be attached as "event handlers" via tag attributes
  - Executed when the event is fired by the browser

```
<img src="logo.gif" onclick="alert('clicked!')" />
```

Calling a JavaScript Function from Event Handler – Example

image-onclick.html

```
<html>
<head>
<script type="text/javascript">
  function test (message) {
    alert(message);
  }
</script>
</head>

<body>
  <img src="logo.gif"
    onclick="test('clicked!')" />
</body>
</html>
```



telerik

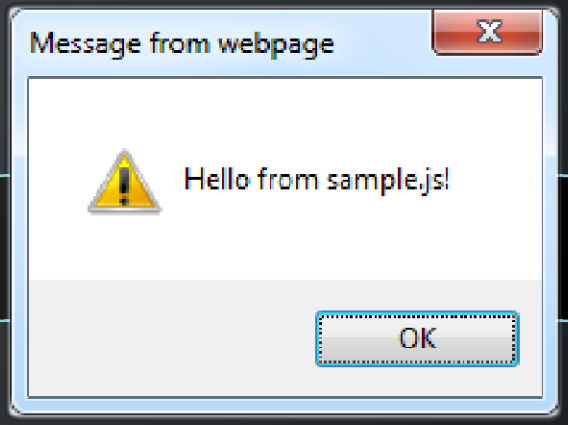
Using External Script Files

- Using external script files:

external-JavaScript.html

```
<html>
<head>
  <script src="sample.js" type="text/javascript">
  </script>
</head>
<body>
  <button onclick="sample()" value="Call JavaScript
    function from sample.js" />
</body>
</html>
```

The <script> tag is always empty.



- External JavaScript file:

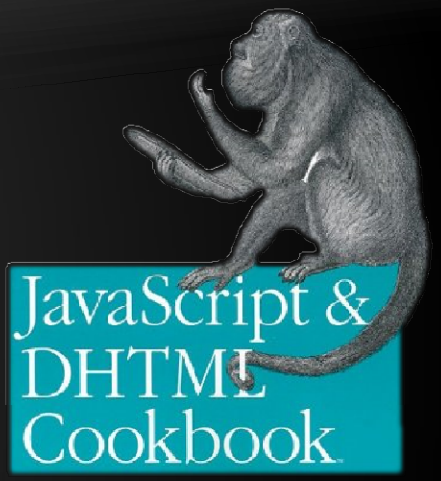
```
function sample() {
  alert('Hello from sample.js!')
}
```

sample.js

telerik

The JavaScript Syntax

```
if (pop < 10)
{
    map.graphics.add(features[i].setSymbol(onePopSymbol));
}
else if (pop >= 10 && pop < 95)
{
    map.graphics.add(features[i].setSymbol(twoPopSymbol));
}
else if (pop >= 95 && pop < 365)
{
    map.graphics.add(features[i].setSymbol(threePopSymbol));
}
else if (pop >= 365 && pop < 1100)
{
    map.graphics.add(features[i].setSymbol(fourPopSymbol));
}
else
{
    map.graphics.add(features[i].setSymbol(fivePopSymbol));
}
```



telerik

JavaScript Syntax

- The JavaScript syntax is similar to C# and Java
  - Operators (+, *, =, !=, &&, ++, …)
  - Variables (typeless)
  - Conditional statements (`if`, `else`)
  - Loops (`for`, `while`)
  - Arrays (`my_array[]`) and associative arrays (`my_array['abc']`)
  - Functions (can return value)
  - Function variables (like the C# delegates)

telerik

Data Types

- JavaScript data types:
  - Numbers (integer, floating-point)
  - Boolean (true / false)
- String type – string of characters

```
var myName = "You can use both single or double
quotes for strings";
```

- Arrays

```
var my_array = [1, 5.3, "aaa"];
```

- Associative arrays (hash tables)

```
var my_hash = {a:2, b:3, c:"text"};
```

telerik

Everything is Object

- Every variable can be considered as object
  - For example strings and arrays have member functions:

objects.html

```
var test = "some string";
alert(test[7]); // shows letter 'r'
alert(test.charAt(5)); // shows letter 's'
alert("test".charAt(1)); //shows letter 'e'
alert("test".substring(1,3)); //shows 'es'
```

```
var arr = [1,3,4];
alert (arr.length); // shows 3
arr.push(7); // appends 7 to end of array
alert (arr[3]); // shows 7
```

telerik

String Operations

- The + operator joins strings

```
string1 = "fat ";
string2 = "cats";
alert(string1 + string2);  // fat cats
```

- What is "9" + 9?

```
alert("9" + 9);  // 99
```

- Converting string to number:

```
alert(parseInt("9") + 9);  // 18
```

telerik

Arrays Operations and Properties

- Declaring new empty array:

```
var arr = new Array();
```

- Declaring an array holding few elements:

```
var arr = [1, 2, 3, 4, 5];
```

- Appending an element / getting the last element:

```
arr.push(3);
var element = arr.pop();
```

- Reading the number of elements (array length):

```
arr.length;
```

- Finding element's index in the array:

```
arr.indexOf(1);
```

telerik

Standard Popup Boxes

- Alert box with text and [OK] button
  - Just a message shown in a dialog box:

```
alert("Some text here");
```

- Confirmation box
  - Contains text, [OK] button and [Cancel] button:

```
confirm("Are you sure?");
```

- Prompt box
  - Contains text, input field with default value:

```
prompt ("enter amount", 10);
```

telerik

Sum of Numbers – Example

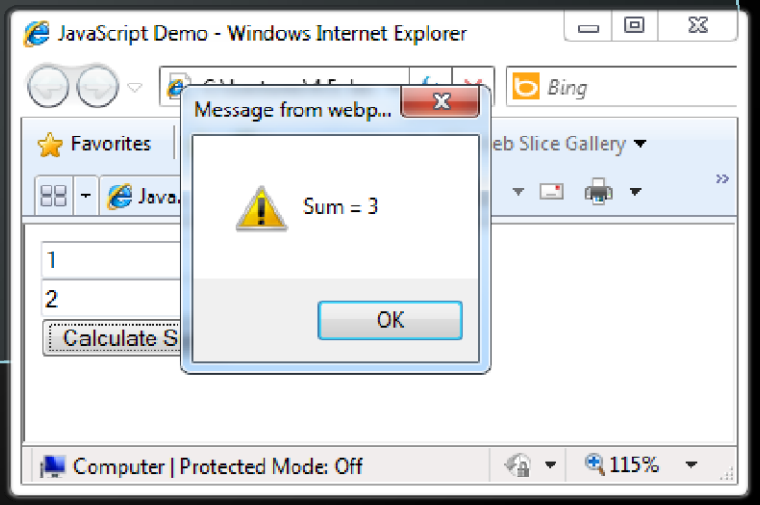
sum-of-numbers.html

```
<html>

<head>
  <title>JavaScript Demo</title>
  <script type="text/javascript">
    function calcSum() {
      value1 =
        parseInt(document.mainForm.textBox1.value);
      value2 =
        parseInt(document.mainForm.textBox2.value);
      sum = value1 + value2;
      document.mainForm.textBoxSum.value = sum;
    }
  </script>
</head>
```

Sum of Numbers – Example (2)

sum-of-numbers.html (cont.)

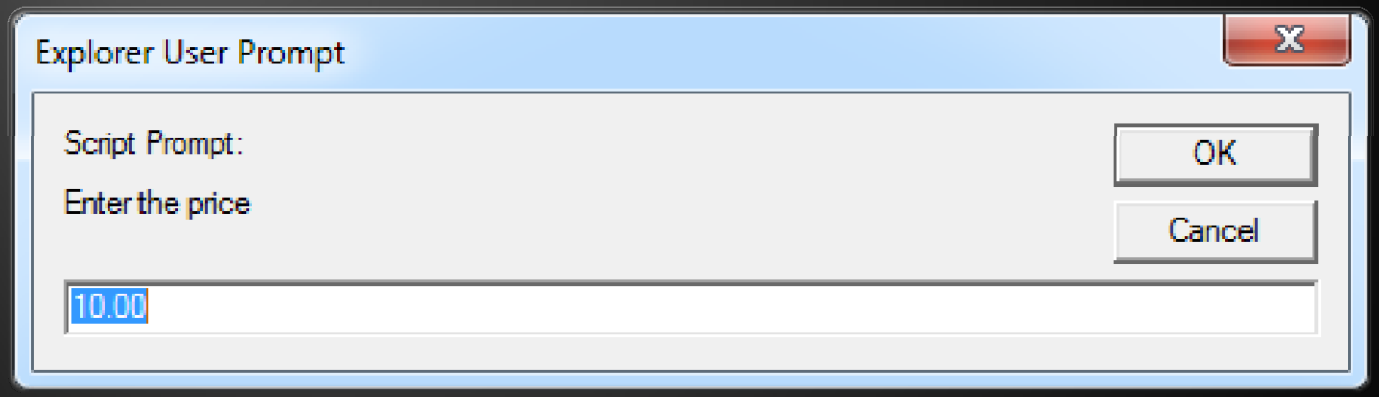
```
<body>
  <form name="mainForm">
    <input type="text" name="textBox1" /> <br/>
    <input type="text" name="textBox2" /> <br/>
    <input type="button" value="Process"
      onclick="javascript: calcSum()" />
    <input type="text" name="textBoxSum"
      readonly="readonly"/>
  </form>
</body>

</html>
```



JavaScript Prompt – Example

prompt.html

```
price = prompt("Enter the price", "10.00");
alert('Price + VAT = ' + price * 1.2);
```



telerik

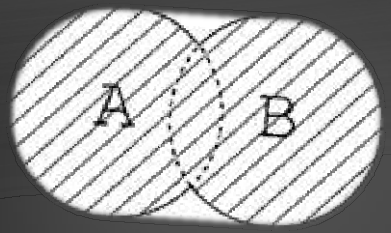
Conditional Statement (if)

```
unitPrice = 1.30;
if (quantity > 100) {
  unitPrice = 1.20;
}
```

Symbol	Meaning
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
==	Equal
!=	Not equal

Conditional Statement (if) (2)

- The condition may be of Boolean or integer type:



conditional-statements.html

```
var a = 0;
var b = true;
if (typeof(a)=="undefined" || typeof(b)=="undefined") {
  document.write("Variable a or b is undefined.");
}
else if (!a && b) {
  document.write("a==0; b==true;");
} else {
  document.write("a==" + a + "; b==" + b + ";");
}
```

telerik

Switch Statement

- The `switch` statement works like in C#:

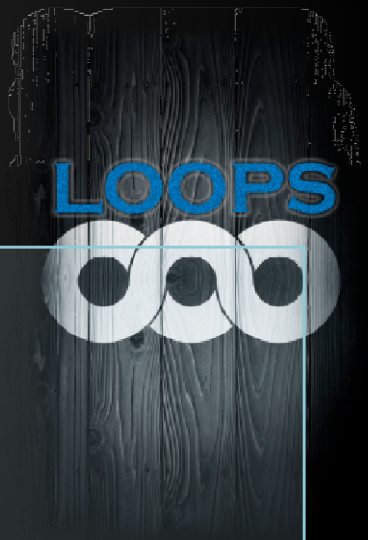
switch-statements.html

```
switch (variable) {
  case 1:
    // do something
    break;
  case 'a':
    // do something else
    break;
  case 3.14:
    // another code
    break;
  default:
    // something completely different
}
```

telerik

Loops

- Like in C#
  - for loop
  - while loop
  - do … while loop




```
var counter;
for (counter=0; counter<4; counter++) {
  alert(counter);
}
while (counter < 5) {
  alert(++counter);
}
```

loops.html

telerik

Functions

- Code structure – splitting code into parts
- Data comes in, processed, result returned

```
function average(a, b, c)
{
    var total;
    total = a+b+c;
    return total/3;
}
```

Parameters come in here.

Declaring variables is optional. Type is never declared.

Value returned here.

telerik

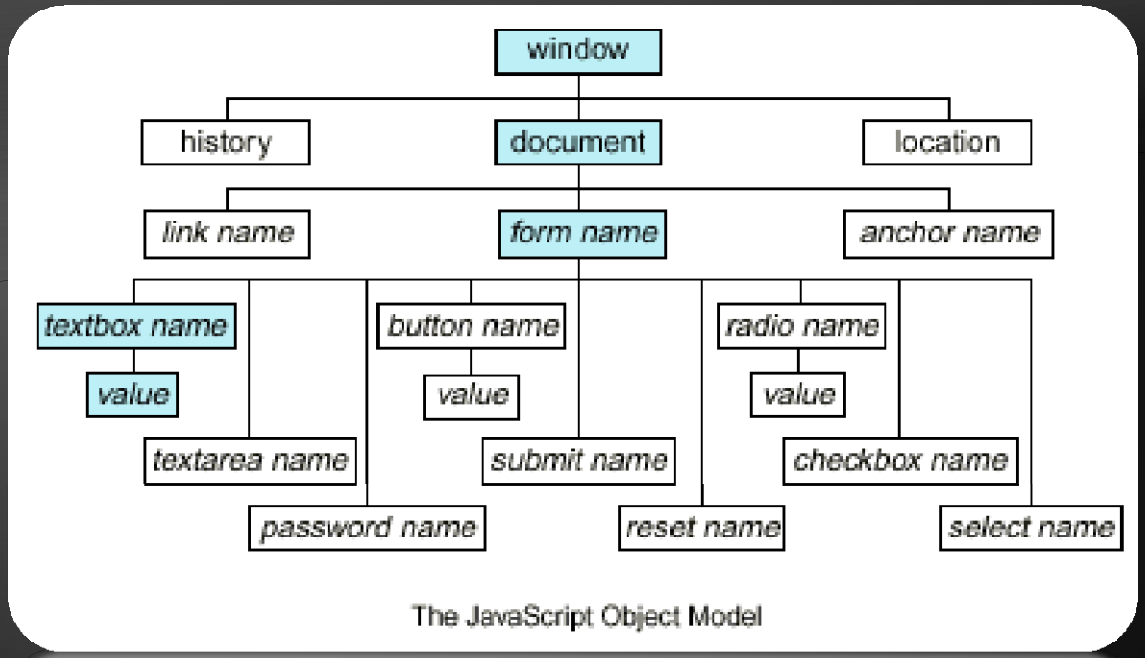
Function Arguments and Return Value

- Functions are not required to return a value
- When calling function it is not obligatory to specify all of its arguments
  - The function has access to all the arguments passed via `arguments` array

```
function sum() {
  var sum = 0;
  for (var i = 0; i < arguments.length; i ++)
    sum += parseInt(arguments[i]);
  return sum;
}
alert(sum(1, 2, 4));
```

functions-demo.html

telerik



Document Object Model (DOM)

telerik

Document Object Model (DOM)

- Every HTML element is accessible via the JavaScript DOM API
- Most DOM objects can be manipulated by the programmer
- The event model lets a document to react when the user does something on the page
- Advantages
  - Create interactive pages
  - Updates the objects of a page without reloading it

telerik

Accessing Elements

- Access elements via their ID attribute

```
var elem = document.getElementById("some_id")
```

- Via the name attribute

```
var arr = document.getElementsByName("some_name")
```

- Via tag name

```
var imgTags = el.getElementsByTagName("img")
```

  - Returns array of descendant `<img>` elements of the element "el"

telerik

DOM Manipulation

- Once we access an element, we can read and write its attributes

DOM-manipulation.html

```
function change(state) {
  var lampImg = document.getElementById("lamp");
  lampImg.src = "lamp_" + state + ".png";
  var statusDiv =
    document.getElementById("statusDiv");
  statusDiv.innerHTML = "The lamp is " + state";
}
…
<img src="test_on.gif" onmouseover="change('off')"
  onmouseout="change('on')" />
```

telerik

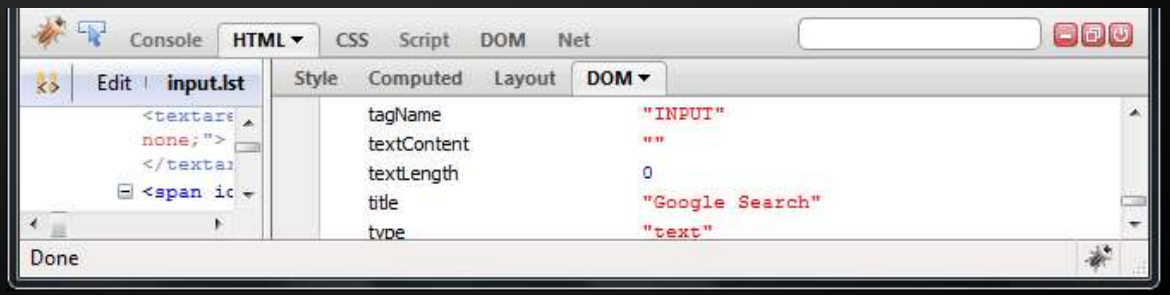
Common Element Properties

- Most of the properties are derived from the HTML attributes of the tag
  - E.g. `id`, `name`, `href`, `alt`, `title`, `src`, etc…
- `style` property – allows modifying the CSS styles of the element
  - Corresponds to the inline style of the element
    - Not the properties derived from embedded or external CSS rules
  - Example: `style.width`, `style.marginTop`, `style.backgroundImage`

telerik

Common Element Properties (2)

- className – the `class` attribute of the tag
- `innerHTML` – holds all the entire HTML code inside the element
- Read-only properties with information for the current element and its state
  - `tagName`, `offsetWidth`, `offsetHeight`, `scrollHeight`, `scrollTop`, `nodeType`, etc…



telerik

Accessing Elements through the DOM Tree Structure

- We can access elements in the DOM through some tree manipulation properties:
  - `element.childNodes`
  - `element.parentNode`
  - `element.nextSibling`
  - `element.previousSibling`
  - `element.firstChild`
  - `element.lastChild`

telerik

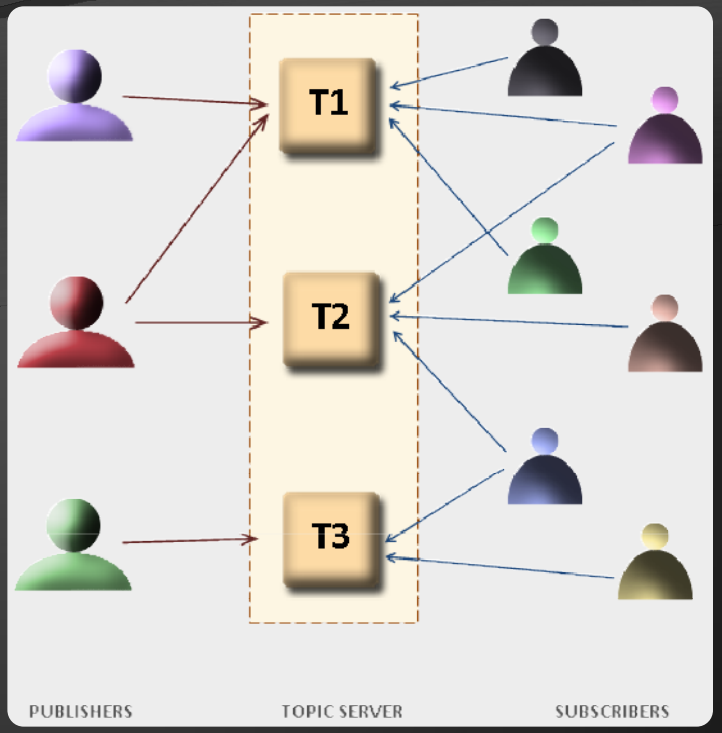
Accessing Elements through the DOM Tree – Example

```
var el = document.getElementById('div_tag');
alert (el.childNodes[0].value);
alert (el.childNodes[1].
  getElementsByTagName('span').id);
…
<div id="div_tag">
  <input type="text" value="test text" />
  <div>
    <span id="test">test span</span>
  </div>
</div>
```

accessing-elements-demo.html

- Warning: may not return what you expected due to Browser differences

telerik



The HTML DOM Event Model

telerik

The HTML DOM Event Model

- JavaScript can register event handlers
  - Events are fired by the Browser and are sent to the specified JavaScript event handler function
  - Can be set with HTML attributes:

```
<img src="test.gif" onclick="imageClicked()" />
```

  - Can be accessed through the DOM:

```
var img = document.getElementById("myImage");
img.onclick = imageClicked;
```

The HTML DOM Event Model (2)

- All event handlers receive one parameter
  - It brings information about the event
  - Contains the type of the event (mouse click, key press, etc.)
  - Data about the location where the event has been fired (e.g. mouse coordinates)
  - Holds a reference to the event sender
    - E.g. the button that was clicked

The HTML DOM Event Model (3)

- Holds information about the state of [Alt], [Ctrl] and [Shift] keys
- Some browsers do not send this object, but place it in the `document.event`
- Some of the names of the event's object properties are browser-specific

telerik

Common DOM Events

- Mouse events:
  - onclick, onmousedown, onmouseup
  - onmouseover, onmouseout, onmousemove
- Key events:
  - onkeypress, onkeydown, onkeyup
  - Only for input fields
- Interface events:
  - onblur, onfocus
  - onscroll

telerik

Common DOM Events (2)

- Form events
  - onchange – for input fields
  - onsubmit
    - Allows you to cancel a form submission
    - Useful for form validation
- Miscellaneous events
  - onload, onunload
    - Allowed only for the <body> element
    - Fires when all content on the page was loaded / unloaded

telerik

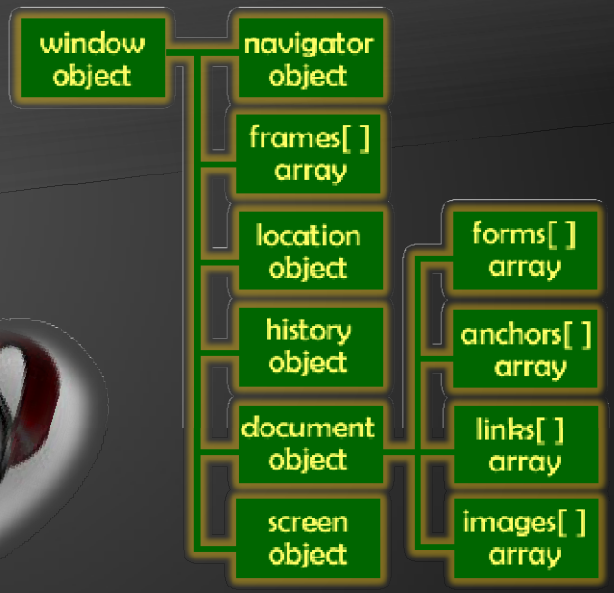
onload Event – Example

- onload event

onload.html

```
<html>
<head>
  <script type="text/javascript">
    function greet() {
      alert("Loaded.");
    }
  </script>
</head>
<body onload="greet()" >
</body>
</html>
```

telerik



The Built-In Browser Objects

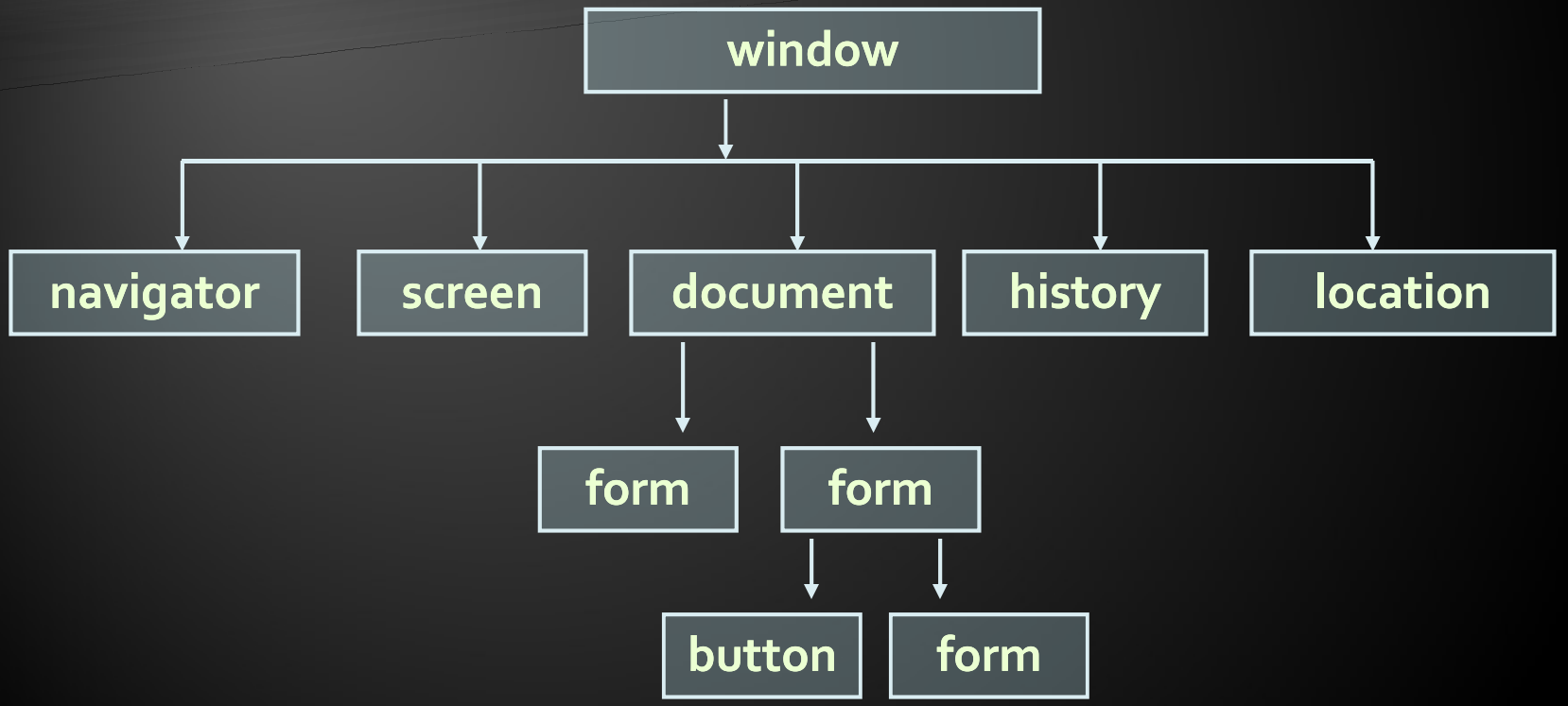
telerik

Built-in Browser Objects

- The browser provides some read-only data via:
  - `window`
    - The top node of the DOM tree
    - Represents the browser's window
  - `document`
    - holds information the current loaded document
  - `screen`
    - Holds the user's display properties
  - `browser`
    - Holds information about the browser

telerik

DOM Hierarchy – Example

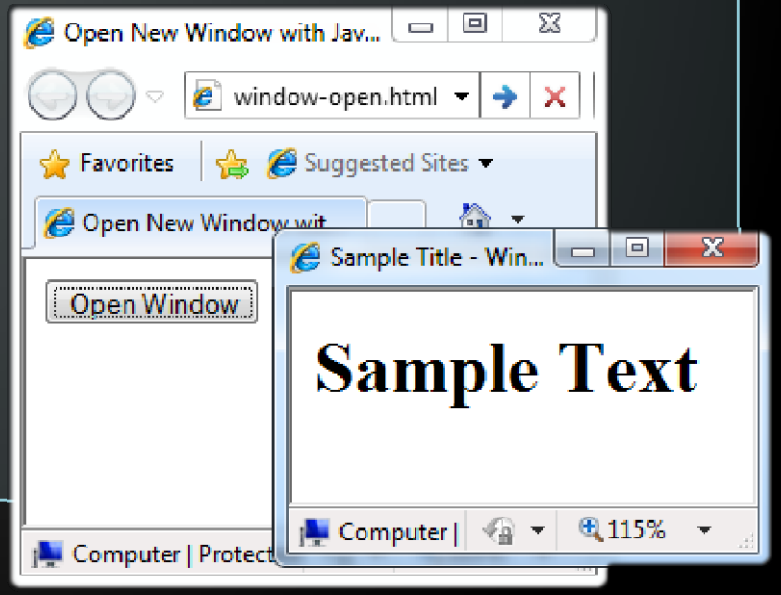


Opening New Window – Example

- window.open()

window-open.html

```
var newWindow = window.open("", "sampleWindow",
  "width=300, height=100, menubar=yes,
  status=yes, resizable=yes");

newWindow.document.write(
  "<html><head><title>
  Sample Title</title>
  </head><body><h1>Sample
  Text</h1></body>");
newWindow.status =
  "Hello folks";
```



telerik

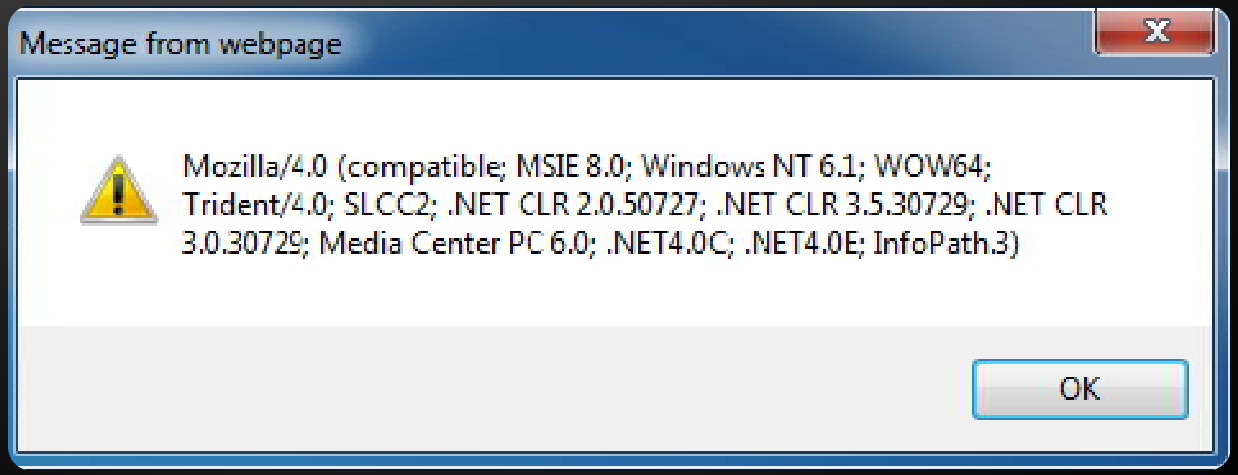
The Navigator Object

```
alert(window.navigator.userAgent);
```

The browser window

The navigator in the browser window

The userAgent (browser ID)



telerik

The Screen Object

- The screen object contains information about the display

```
window.moveTo(0, 0);
x = screen.availWidth;
y = screen.availHeight;
window.resizeTo(x, y);
```

telerik

Document and Location

- document object
  - Provides some built-in arrays of specific objects on the currently loaded Web page

```
document.links[0].href = "yahoo.com";
document.write(
  "This is some <b>bold text</b>");
```

- document.location
  - Used to access the currently open URL or redirect the browser

```
document.location = "http://www.yahoo.com/";
```

telerik

Form Validation – Example

form-validation.html

```
function checkForm()
{
  var valid = true;
  if (document.mainForm.firstName.value == "") {
    alert("Please type in your first name!");
    document.getElementById("firstNameError").
      style.display = "inline";
    valid = false;
  }
  return valid;
}
…
<form name="mainForm" onsubmit="return checkForm()">
  <input type="text" name="firstName" />
  …
</form>
```

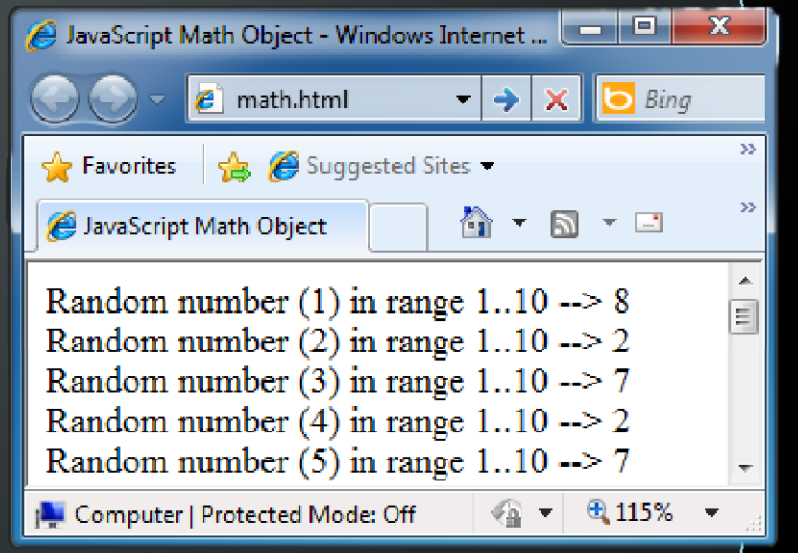
telerik

The Math Object

- The Math object provides some mathematical functions

math.html

```
for (i=1; i<=20; i++) {
  var x = Math.random();
  x = 10*x + 1;
  x = Math.floor(x);
  document.write(
    "Random number (" +
    i + ") in range " +
    "1..10 --> " + x +
    "<br/>");
}
```



telerik

The Date Object

- The Date object provides date / calendar functions

dates.html

```
var now = new Date();
var result = "It is now " + now;
document.getElementById("timeField")
   .innerText = result;
...
<p id="timeField"></p>
```

telerik

Timers: setTimeout()

- Make something happen (once) after a fixed delay

```
var timer = setTimeout('bang()', 5000);
```

5 seconds after this statement executes, this function is called

```
clearTimeout(timer);
```

Cancels the timer

telerik

Timers: setInterval()

- Make something happen repeatedly at fixed intervals

```
var timer = setInterval('clock()', 1000);
```

This function is called continuously per 1 second.

```
clearInterval(timer);
```

Stop the timer.

telerik

Timer – Example

timer-demo.html

```
<script type="text/javascript">
  function timerFunc() {
    var now = new Date();
    var hour = now.getHours();
    var min = now.getMinutes();
    var sec = now.getSeconds();
    document.getElementById("clock").value =
      "" + hour + ":" + min + ":" + sec;
  }

  setInterval('timerFunc()', 1000);
</script>

<input type="text" id="clock" />
```

telerik

Debugging JavaScript

telerik

Debugging JavaScript

- Modern browsers have JavaScript console where errors in scripts are reported
  - Errors may differ across browsers
- Several tools to debug JavaScript
  - Microsoft Script Editor
    - Add-on for Internet Explorer
    - Supports breakpoints, watches
    - JavaScript statement debugger; opens the script editor

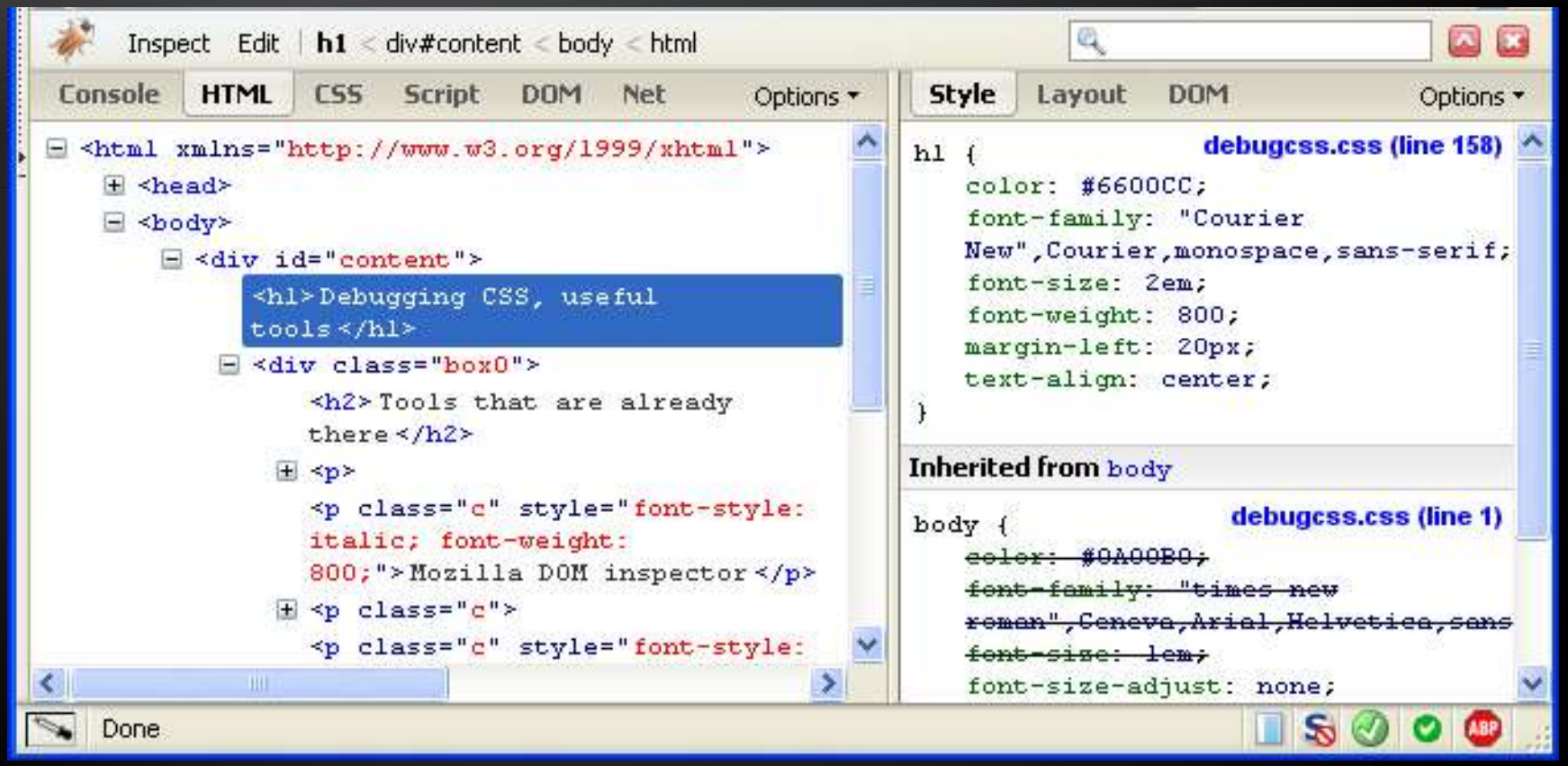
telerik

Firebug

- Firebug – Firefox add-on for debugging JavaScript, CSS, HTML
  - Supports breakpoints, watches, JavaScript console editor
  - Very useful for CSS and HTML too
    - You can edit all the document real-time: CSS, HTML, etc
    - Shows how CSS rules apply to element
  - Shows Ajax requests and responses
  - Firebug is written mostly in JavaScript

telerik

Firebug (2)



telerik

JavaScript Console Object

- The `console` object exists only if there is a debugging tool that supports it
  - Used to write log messages at runtime
- Methods of the `console` object:
  - `debug(message)`
  - `info(message)`
  - `log(message)`
  - `warn(message)`
  - `error(message)`

telerik

HTML, CSS and JavaScript Basics