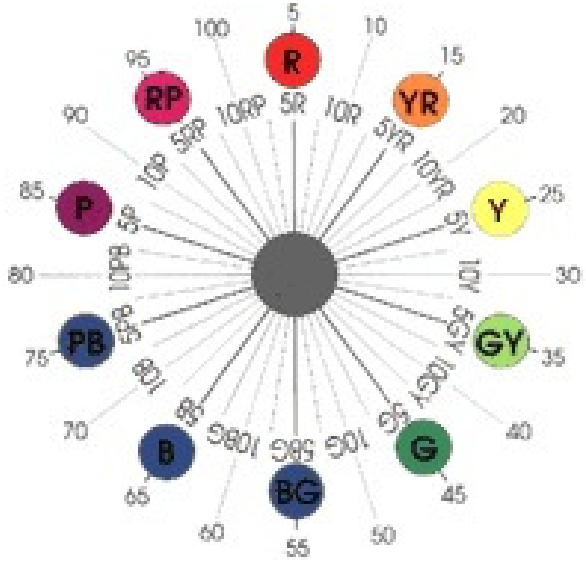
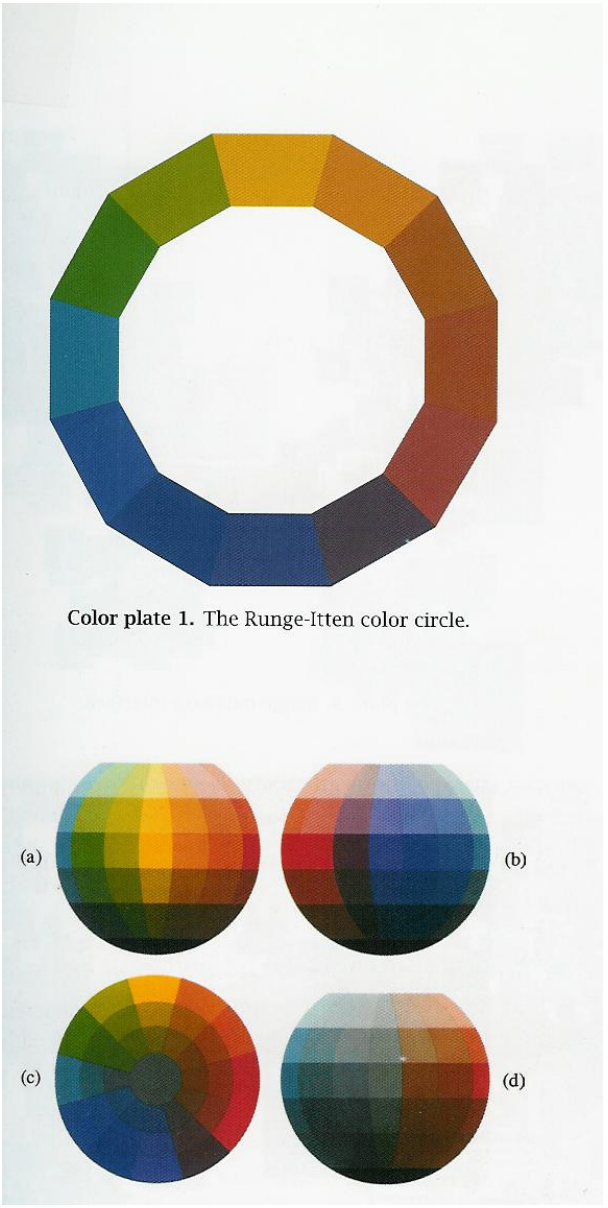
Chapter 3
Introduction to HTML5: Part 2



Munsell Hue Circle		Primary Hue Circle	
Hue	**Symbol**	**Hue**	**Symbol**
Red	R	Red	R
Yellow-Red	YR	Yellow-Red	YR
Yellow	Y	Yellow	Y
Green-Yellow	GY	Green-Yellow	GY
Green	G	Green	G
Blue-Green	C	Cyan	C
Blue	BG	Blue-Cyan	BC
Purple-Blue	PB	Blue	B
Purple	P	Magenta -Blue	MB
Red-Purple	RP	Magenta	M



Color plate 1. The Runge-Itten color circle.

Munsell Chroma Scale

1810 Runge
1810 Goethe
1961 Itten

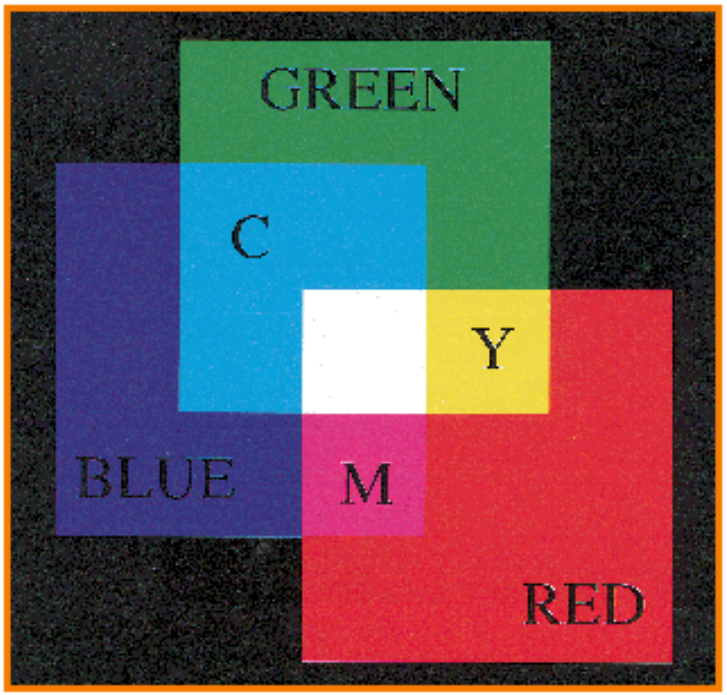
Color

- Black is the absorption of all colors. It is not considered as a color.
- Black is considered the absence of color, and therefore is not allowed to join its color family.

COSMIC RAYS	GAMMA RAYS	ULTRAVIOLET RAYS	VISIBLE RAYS	INFRARED RAYS	MICRO-WAVES RAYS	RADIO RAYS
0.001 nm -	1nm -	10 nm -	400 nm -	700 nm -	1000 nm -	1000 m

VIOLET	INDIGO	BLUE	GREEN	YELLOW	ORANGE	RED
400 nm -	450 nm -	480 nm -	550 nm -	580 nm -	610 nm -	700 nm

RGB



Colors are additive

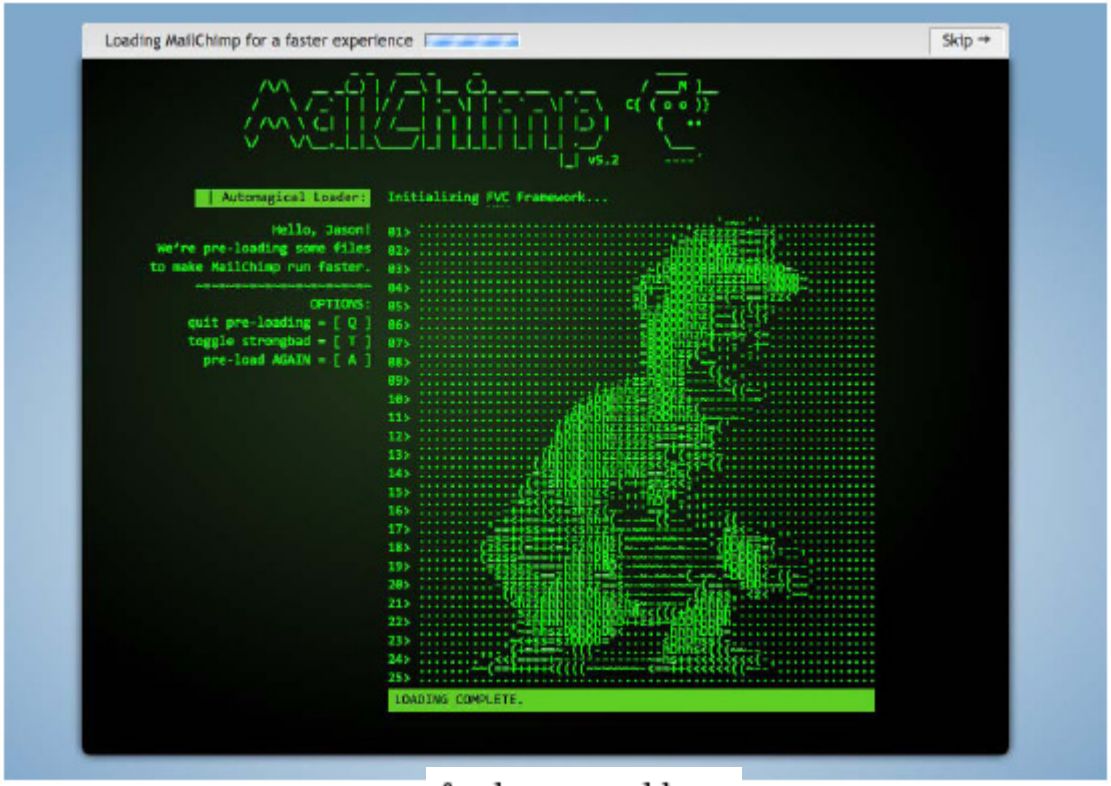
R	G	B	Color
0.0	0.0	0.0	Black
1.0	0.0	0.0	Red
0.0	1.0	0.0	Green
0.0	0.0	1.0	Blue
1.0	1.0	0.0	Yellow
1.0	0.0	1.0	Magenta
0.0	1.0	1.0	Cyan
1.0	1.0	1.0	White
0.5	0.0	0.0	?
1.0	0.5	0.5	?
1.0	0.5	0.0	?
0.5	0.3	0.1	?

Yellow, the color of smileys

passion

Red, the color of affection

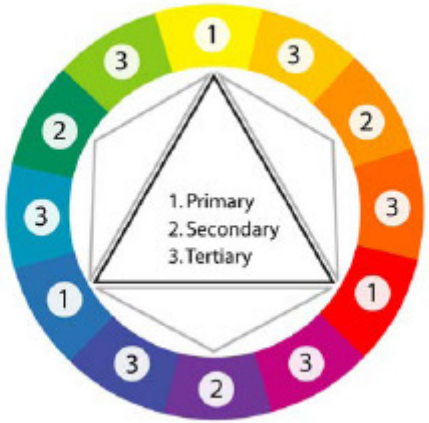
Calming stones, sky, and sea



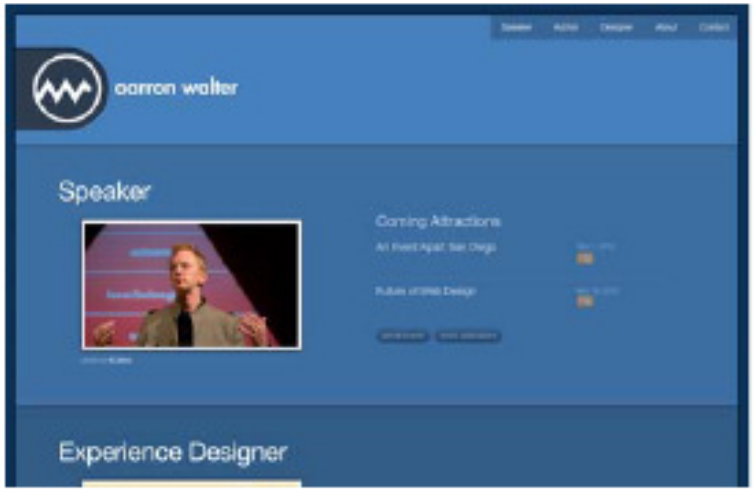
freshness, and hope.

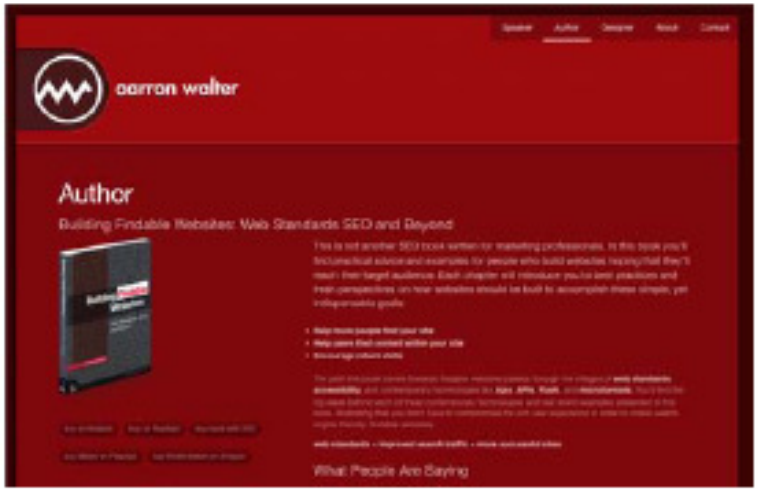
Warm Colors

Cool Colors



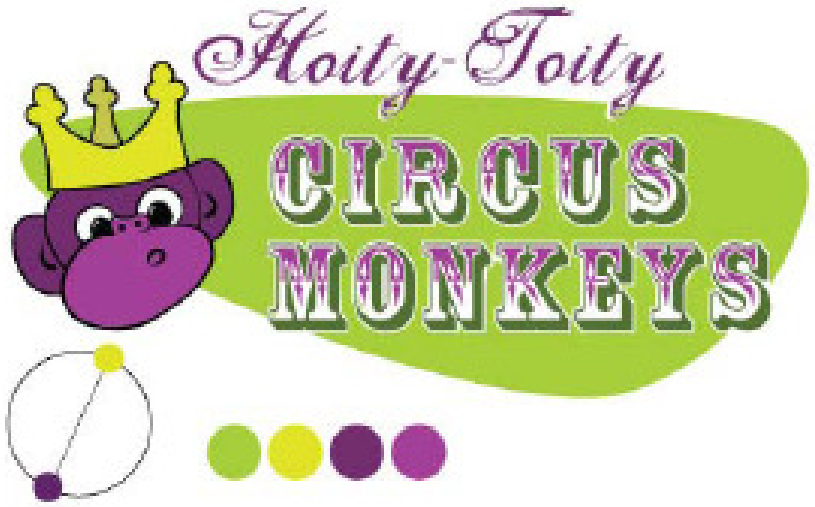
A Monochromatic Color Scheme


aaron walter
Speaker
Coming Attractions
Experience Designer


aaron walter
Author
Building Findable Websites: Web Standards SEO and Beyond
What People Are Saying

. Art in My Coffee—featuring colors directly drawn from the site's subject matter


Omihi Waipara
New Zealand Wine
Estate
Black
Top
Vineyard
Wines
Stockists
Join the Black List
Contact
Region
Black Estate is a remarkable place for wine.
A family owned vineyard on the warm, sunny slopes of the Waipara Valley, where the vines grow through clay and limestone soils to produce intense wines completely expressive of this site.
Our wines are hand crafted using artisan techniques and a dedication to creating exceptional Waipara Pinot Noir, Chardonnay and Riesling.
Chardonnay
2009
Estate
Black
Latest News



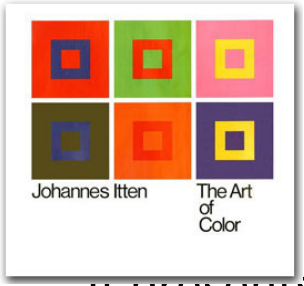
A funky complementary monkey

Complementary colors

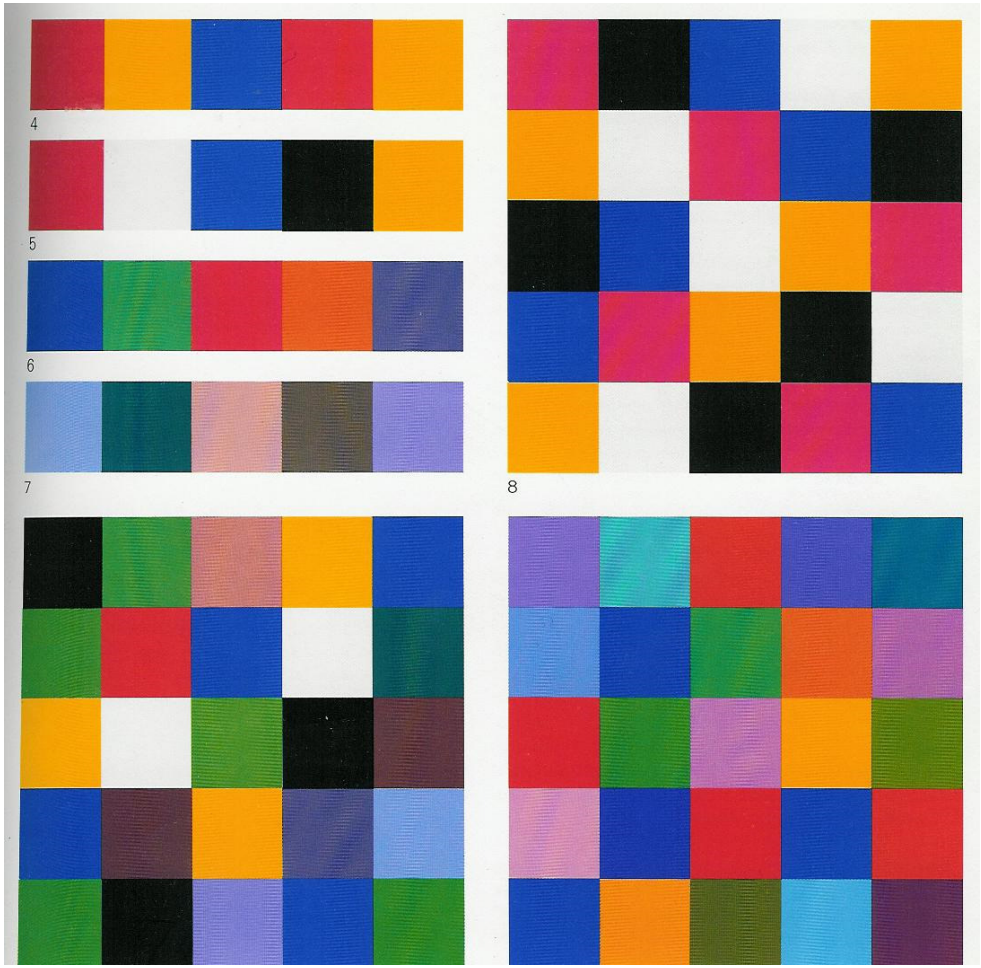
Syntax

- Colors
  - RGB
    - 0 0 0 is black
      - 00 00 00
      - #000000
    - 255 255 255 is white
      - FF FF FF
      - #FFFFFF
  - Background color
    - <body bgcolor="#FFFFFF">
    - <body bgcolor=green>

High level color characteristics. Contrast of hue.

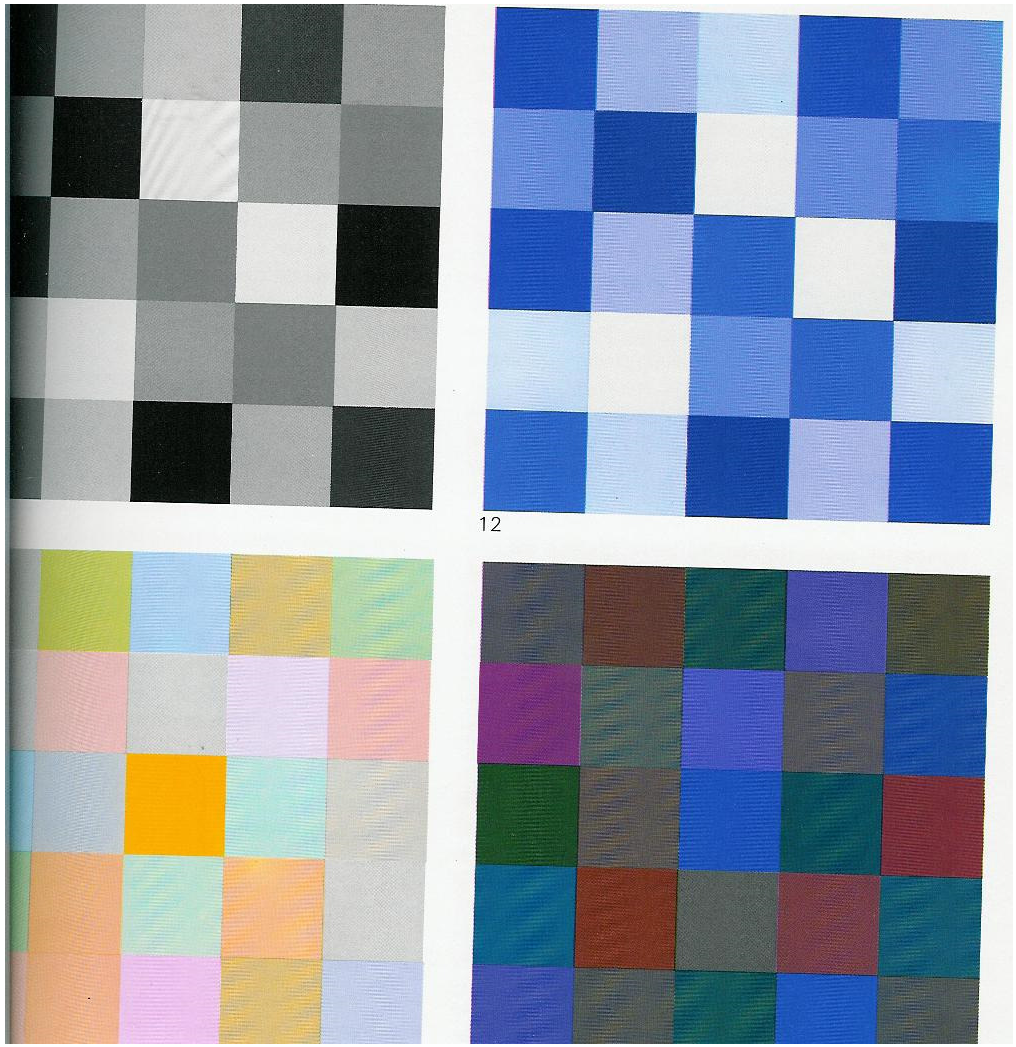


It presents undiluted colors in their most intense luminosity. Some color combinations are: yellow/red/blue, red/blue/green, blue/yellow/violet, yellow/green/violet/red/, violet/green/blue/orange/black.



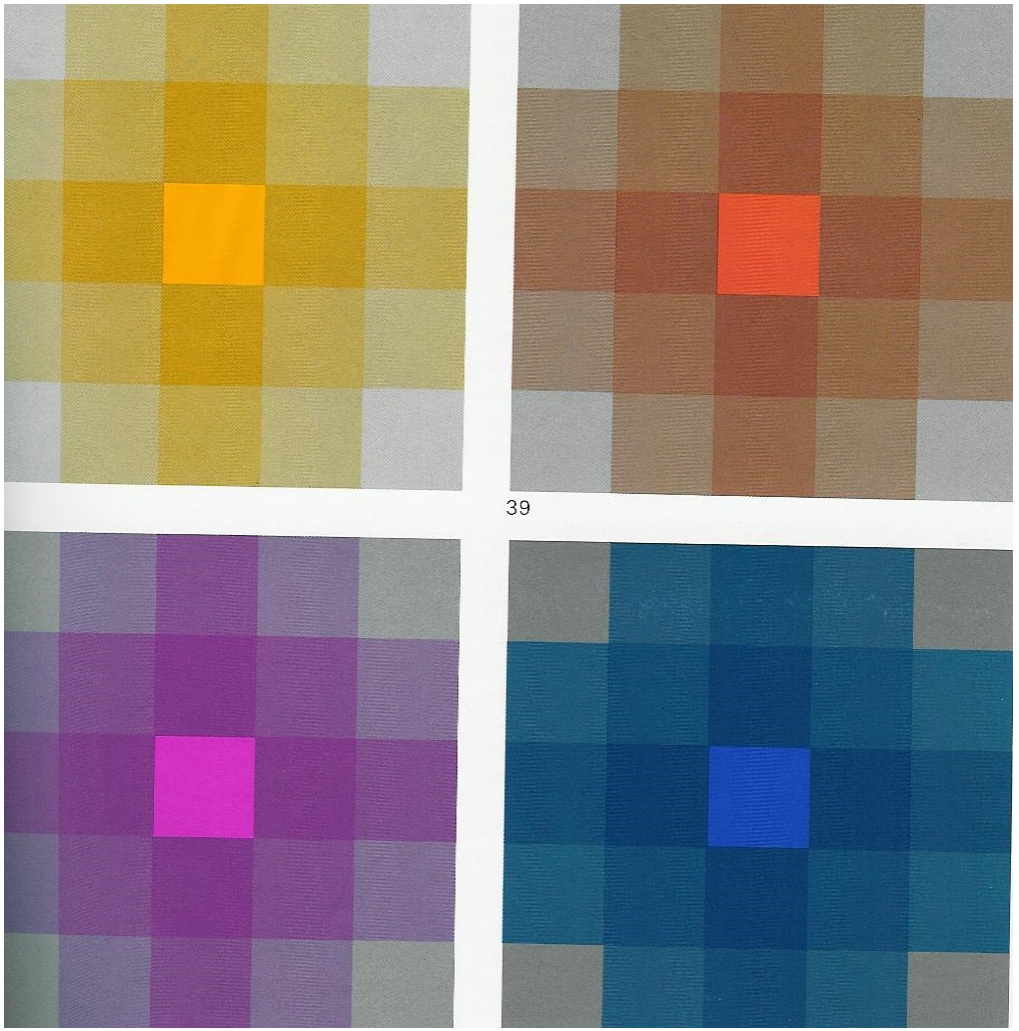
High level color characteristics. Complementary contract

Two colors are called complementary if their pigments mixed together yield in neutral grey. Examples are: yellow-violet, blue-orange, red-green. This contrast gives the effect of a stability fixed image.



High level color characteristics. Saturation contract

Saturation relates to the degree of purity of the color.



Contrasts

 **The contrast of saturation**. Juxtaposition of light and dark values and their relative saturation.

The contrast of light and dark. Juxtaposition of light and dark values. This could be a monochromatic composition. 

 **The contrast of extension.** Assigning proportional field sizes in relation to the visual weight of a color.

The contrast of complements. Juxtaposition of color wheel or perceptual opposites. 

 **Simultaneous contrast.** The boundaries between colors perceptually vibrate.

The contrast of hue. Juxtaposition of different hues. 

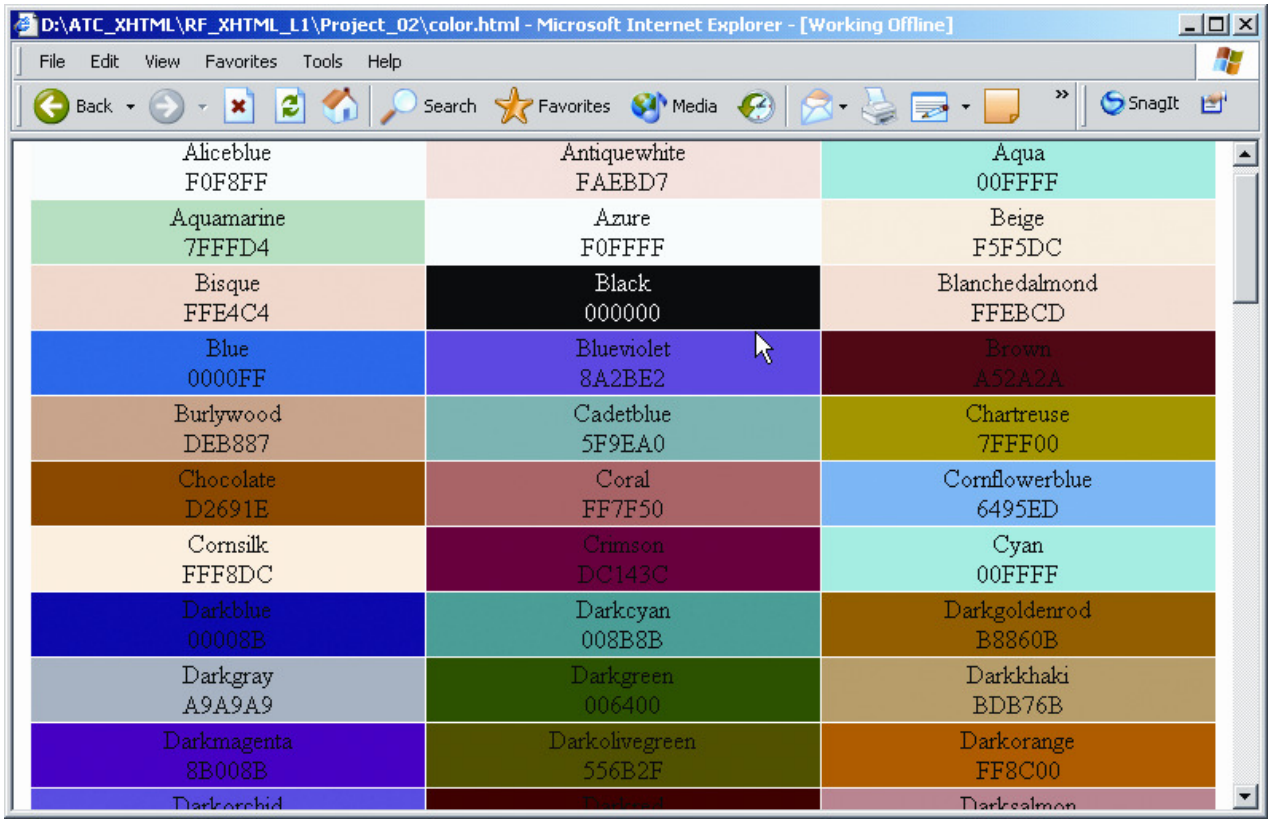
Color Names

There are 16 color names supported in XHTML

COLOR NAME:	HEX NUMBER:
BLACK	
SILVER	#C0C0C0
GRAY	#808080
WHITE	#FFFFFF
MAROON	#800000
RED	#FF0000
PURPLE	#800080
FUCHSIA	#FF00FF
GREEN	#008000
LIME	#00FF00
OLIVE	#808000
YELLOW	#FFFF00
NAVY	#000080
BLUE	#0000FF
TEAL	#008080
AQUA	#00FFFF

Each color name has an associated Hexadecimal code

Some RGB Color Codes



Aliceblue F0F8FF	Antiquewhite FAEBD7	Aqua 00FFFF
Aquamarine 7FFFD4	Azure F0FFFF	Beige F5F5DC
Bisque FFE4C4	Black 000000	Blanchedalmond FFEBCD
Blue 0000FF	Blueviolet 8A2BE2	Brown A52A2A
Burlywood DEB887	Cadetblue 5F9EA0	Chartreuse 7FFF00
Chocolate D2691E	Coral FF7F50	Cornflowerblue 6495ED
Cornsilk FFF8DC	Crimson DC143C	Cyan 00FFFF
Darkblue 00008B	Darkcyan 008B8B	Darkgoldenrod B8860B
Darkgray A9A9A9	Darkgreen 006400	Darkkhaki BDB76B
Darkmagenta 8B008B	Darkolivegreen 556B2F	Darkorange FF8C00
Darkorchid		Darksalmon

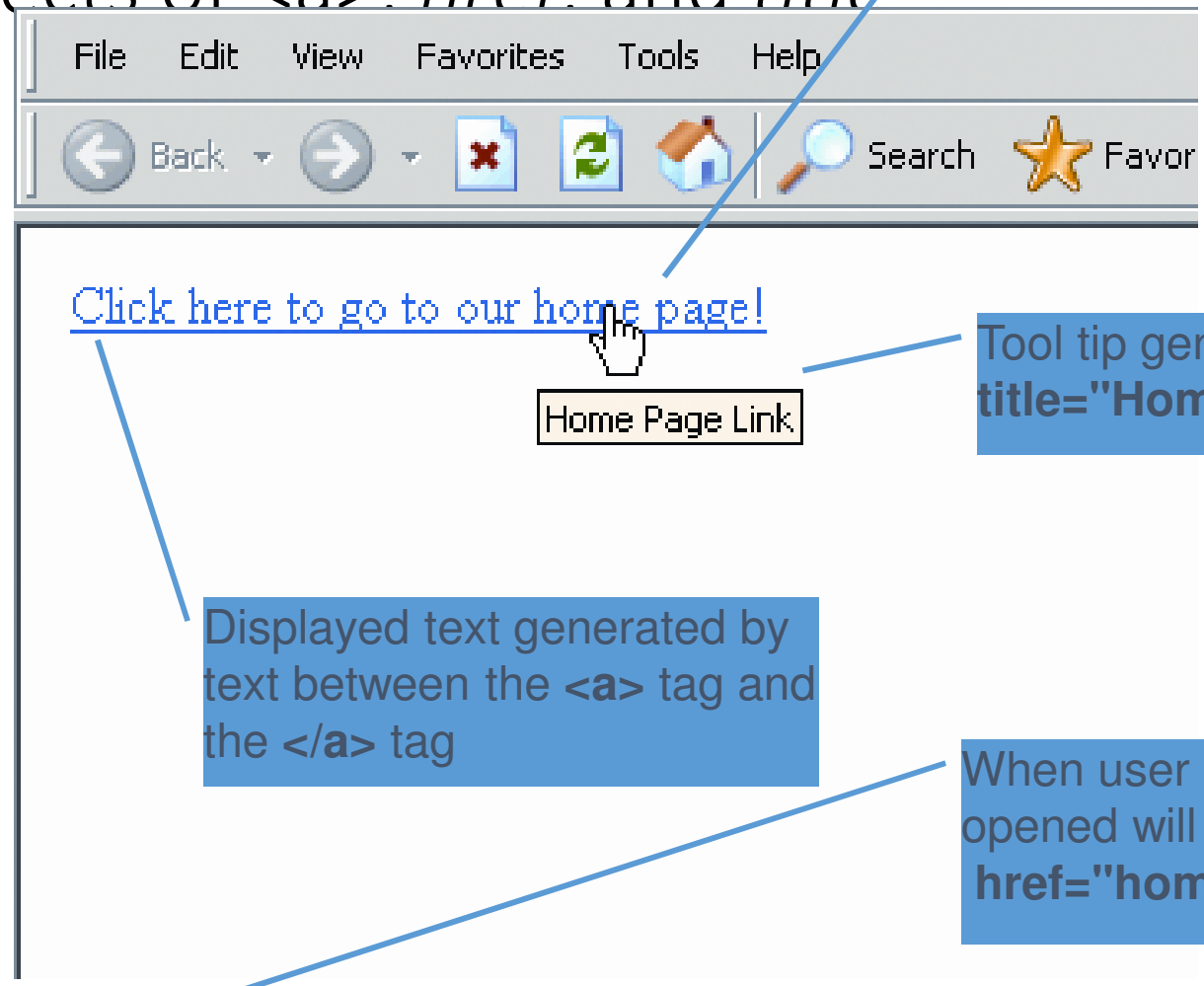
Syntax

- Colors
  - Other attributes
    - text
      - Color of page's body text that isn't a link
    - link
      - Color of unfollowed links
      - Blue by default
    - vlink
      - Color of visited links
      - Purple or red by default

Syntax

- Colors
  - Other attributes
    - Local font color changes
      - <font color="#FF0000">
    - Using an image as a background
      - <body background="tiles.gif">

Effects of <a>, *href*, and *title*



Mouse pointer changes when hovered over the link

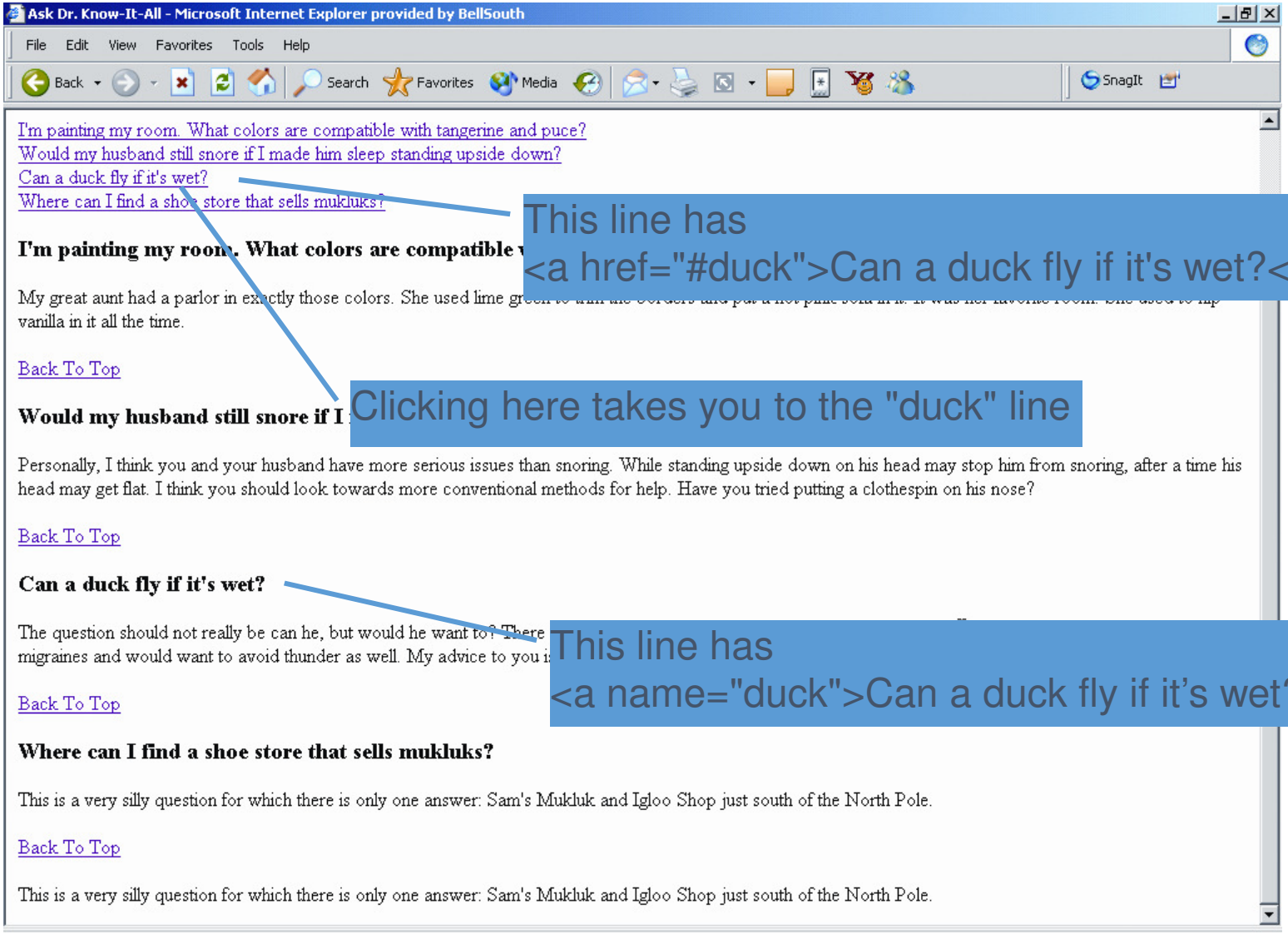
Tool tip generated by **title="Home Page Link"**

Displayed text generated by text between the **<a>** tag and the **** tag

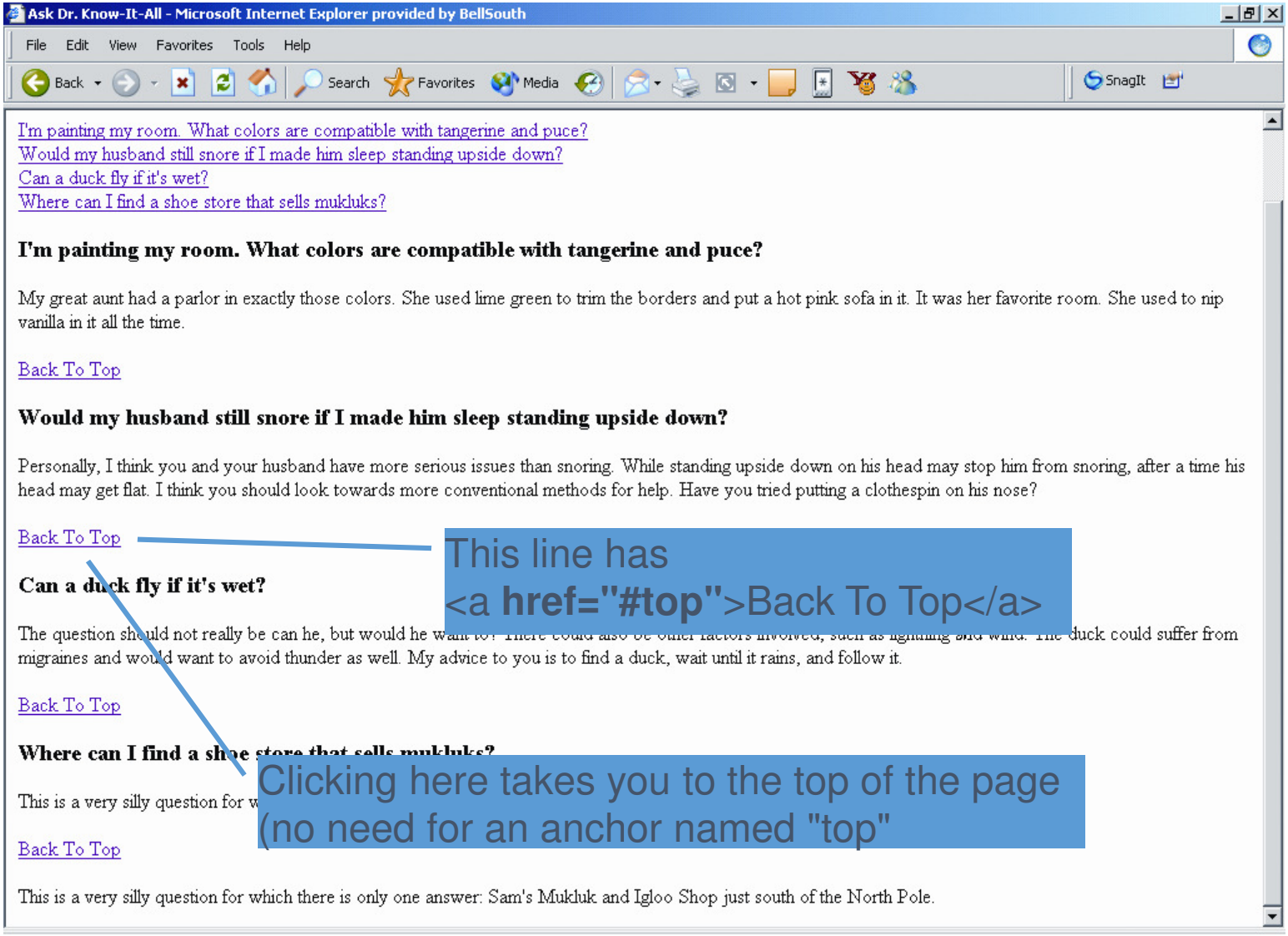
When user clicks, the page opened will be based on **href="home.htm"**

```
<a href="home.htm" title="Home Page Link">Click here to go to our home page!</a>
```

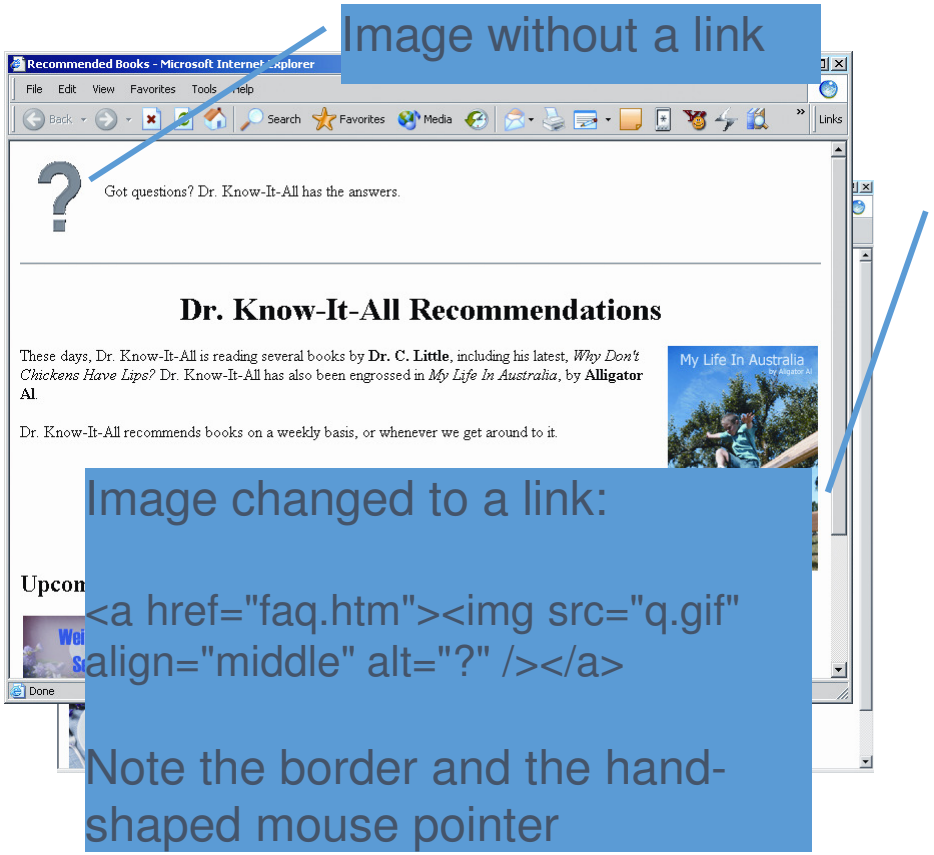
Effects of Within-Page Linking



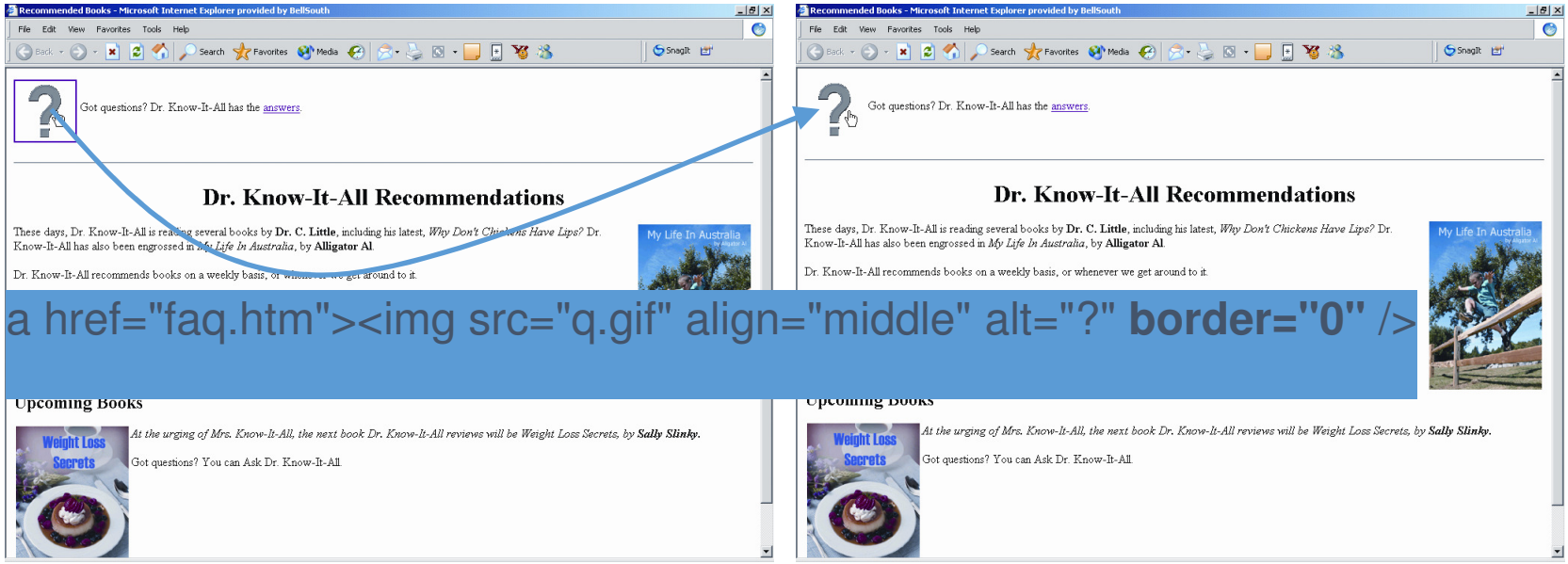
Using a Special Value *#top*



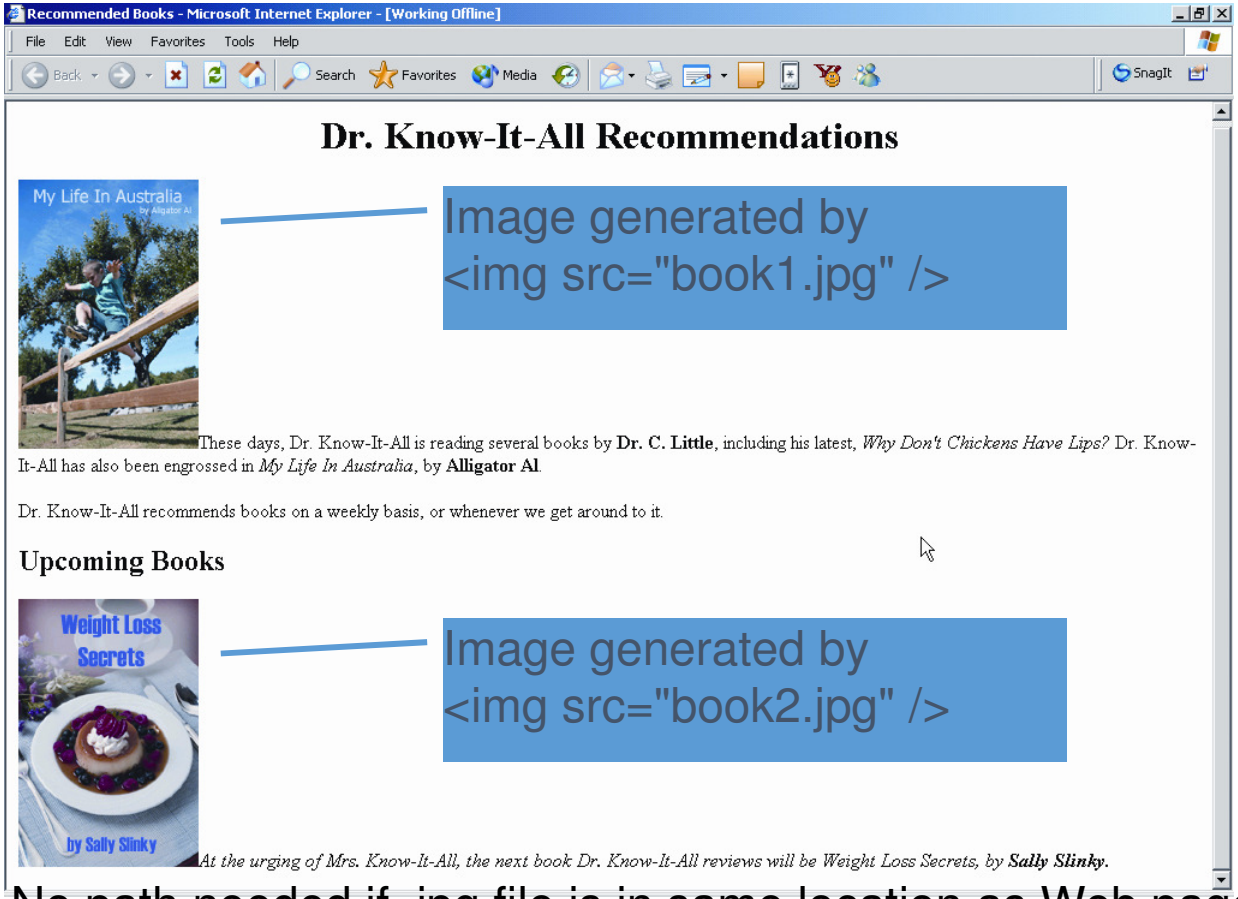
Effect of Image Hyperlinks



Effect of Border Removal

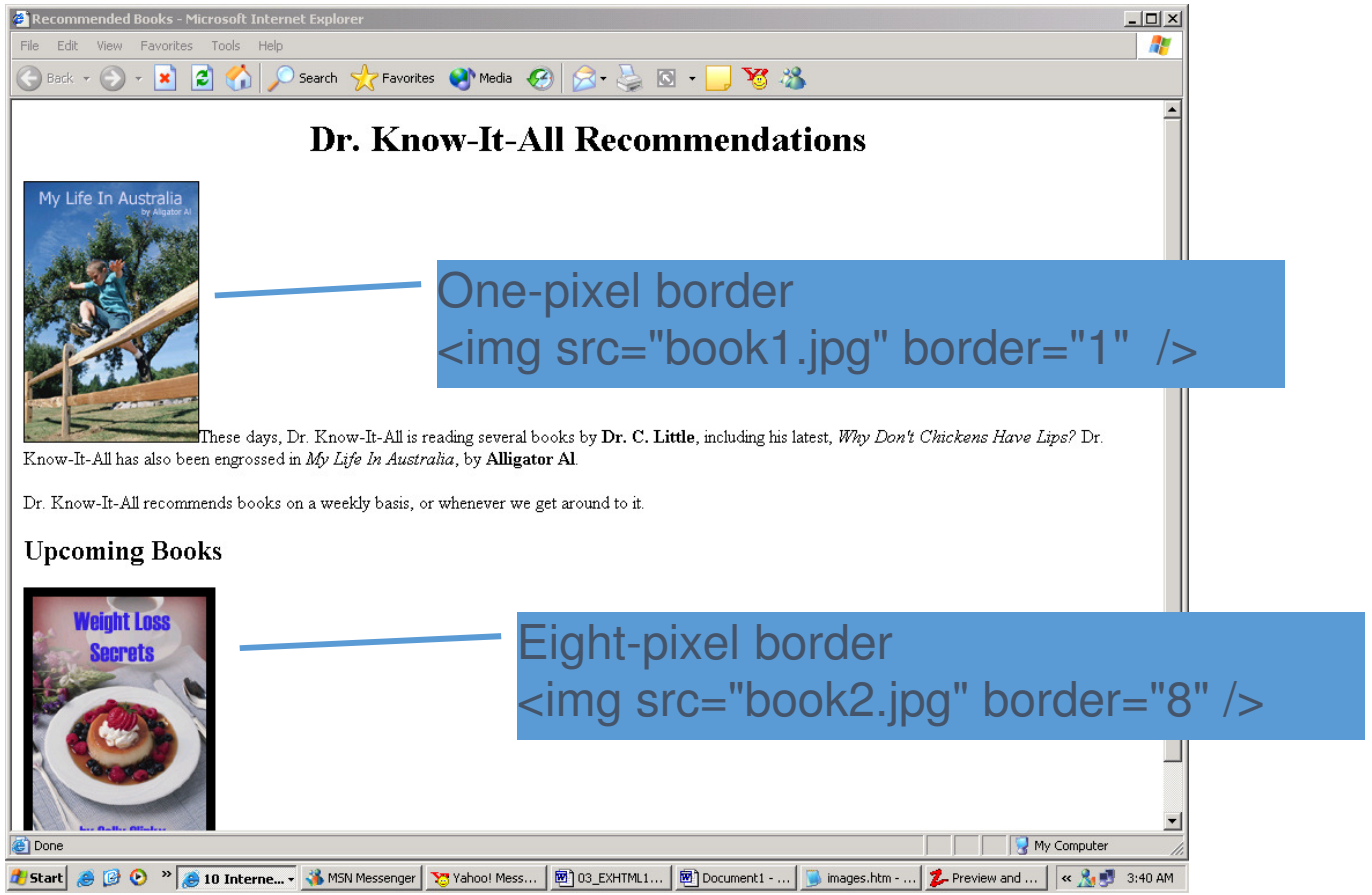


Effects of <img> and *src*

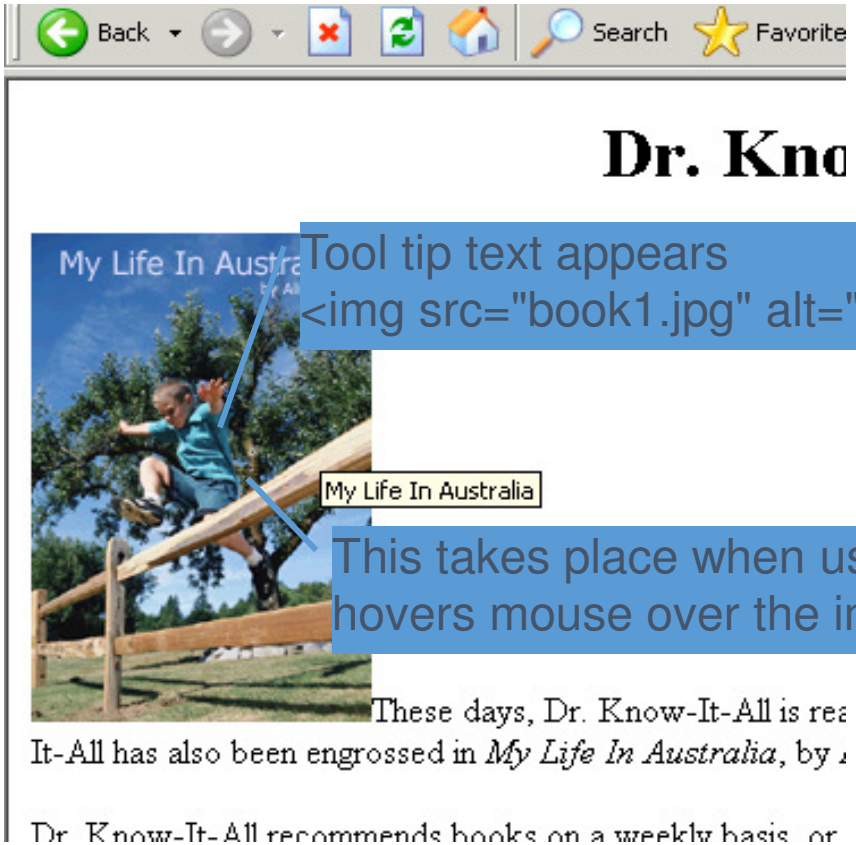


No path needed if .jpg file is in same location as Web page

Effects of *border*



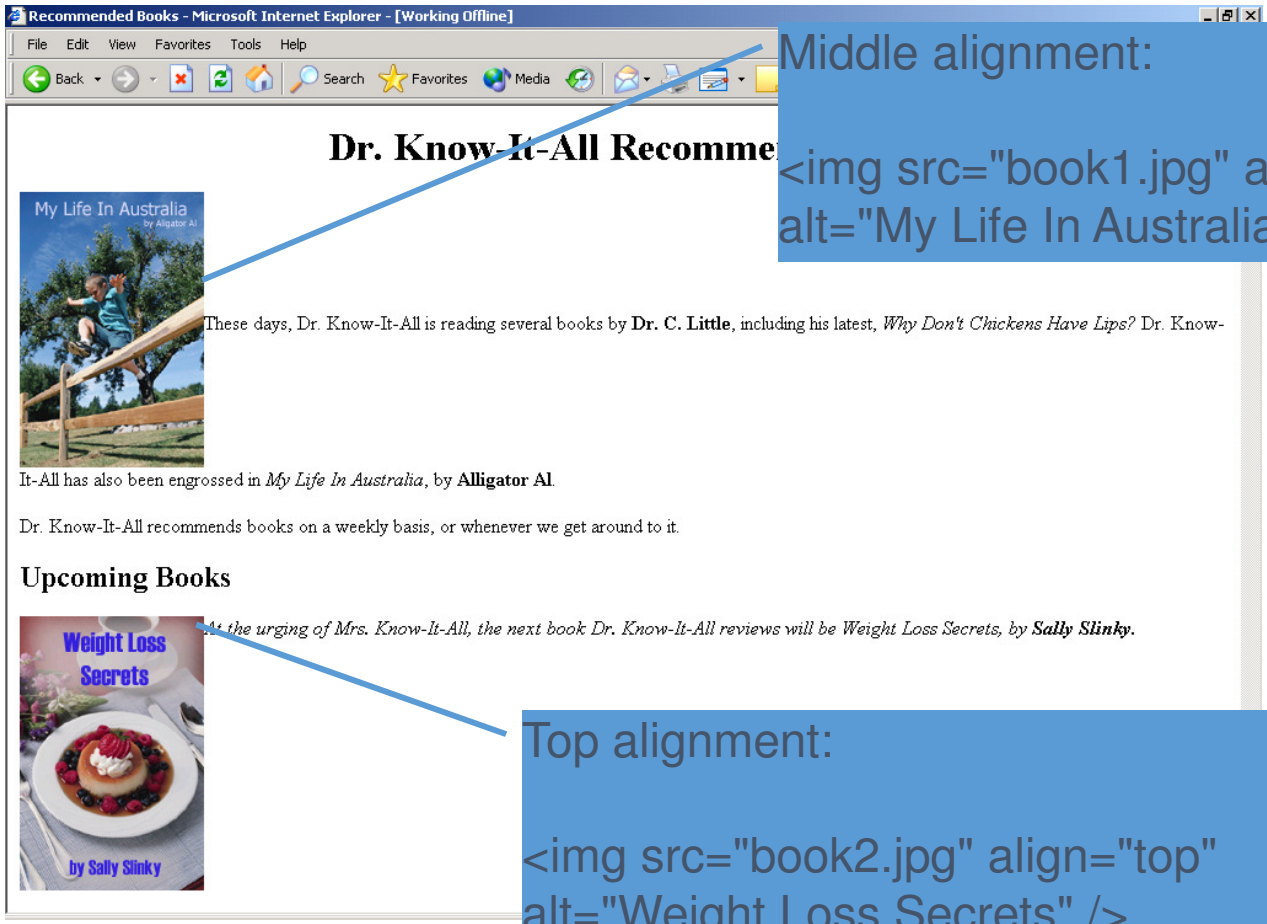
Effects of Setting Alternate Text



Tool tip text appears

This takes place when user hovers mouse over the image

Effect of Image Alignment



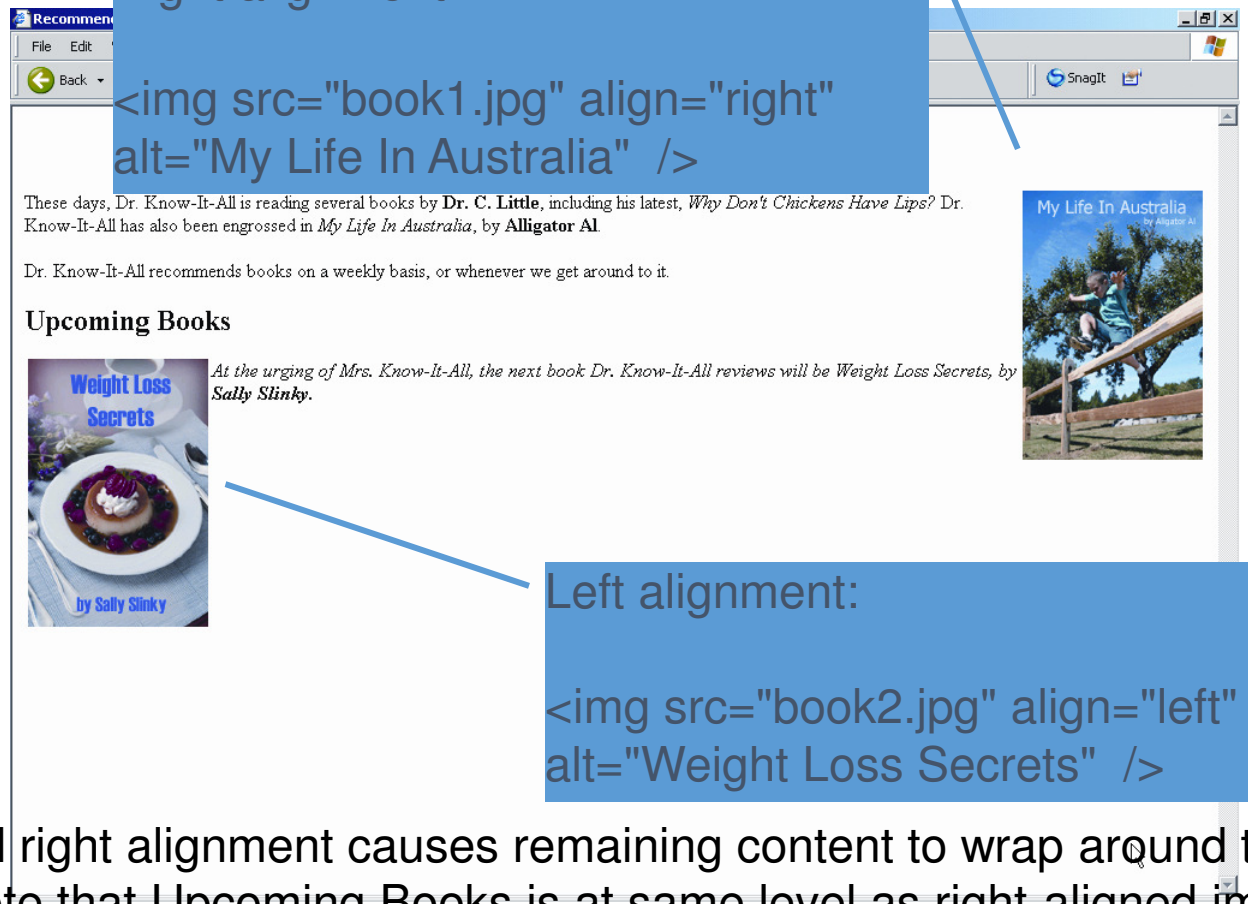
Middle alignment:

```
<img src="book1.jpg" align="middle"
alt="My Life In Australia" />
```

Top alignment:

```
<img src="book2.jpg" align="top"
alt="Weight Loss Secrets" />
```

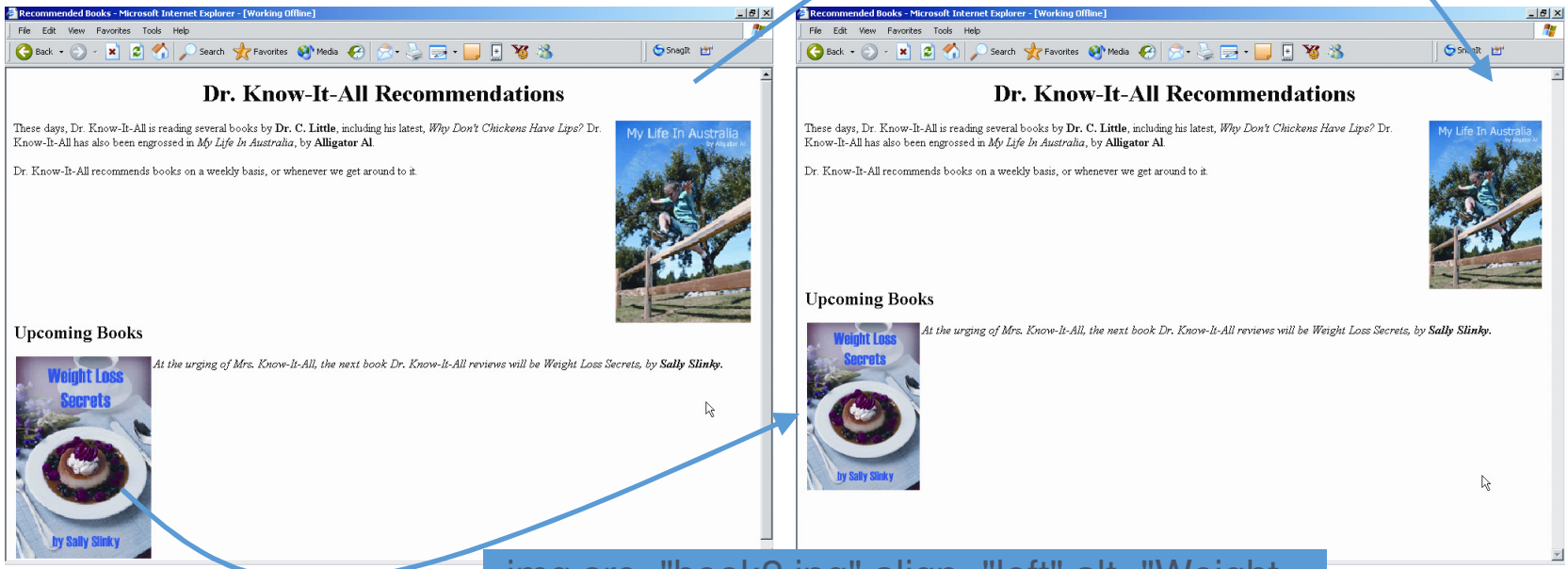
Effect of Image Alignment (continued)



Left and right alignment causes remaining content to wrap around the image. Note that Upcoming Books is at same level as right-aligned image.

Effects of Image Sizing

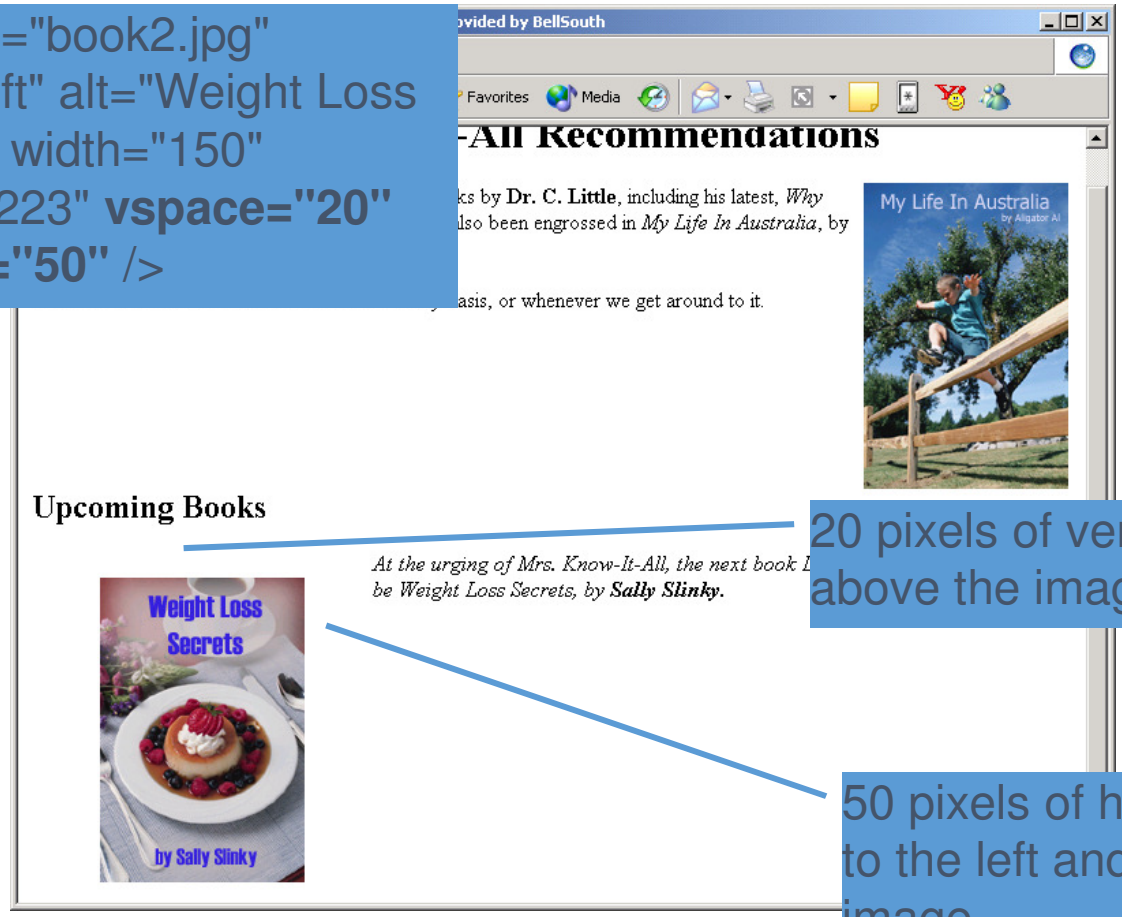
```
<img src="book1.jpg" align="right" alt="My Life In Australia" width="150" height="224" />
```




```
<img src="book2.jpg" align="left" alt="Weight Loss Secrets" width="150" height="223" />
```

Effect of Vertical and Horizontal Spacing

```
<img src="book2.jpg"
align="left" alt="Weight Loss
Secrets" width="150"
height="223" vspace="20"
hspace="50" />
```

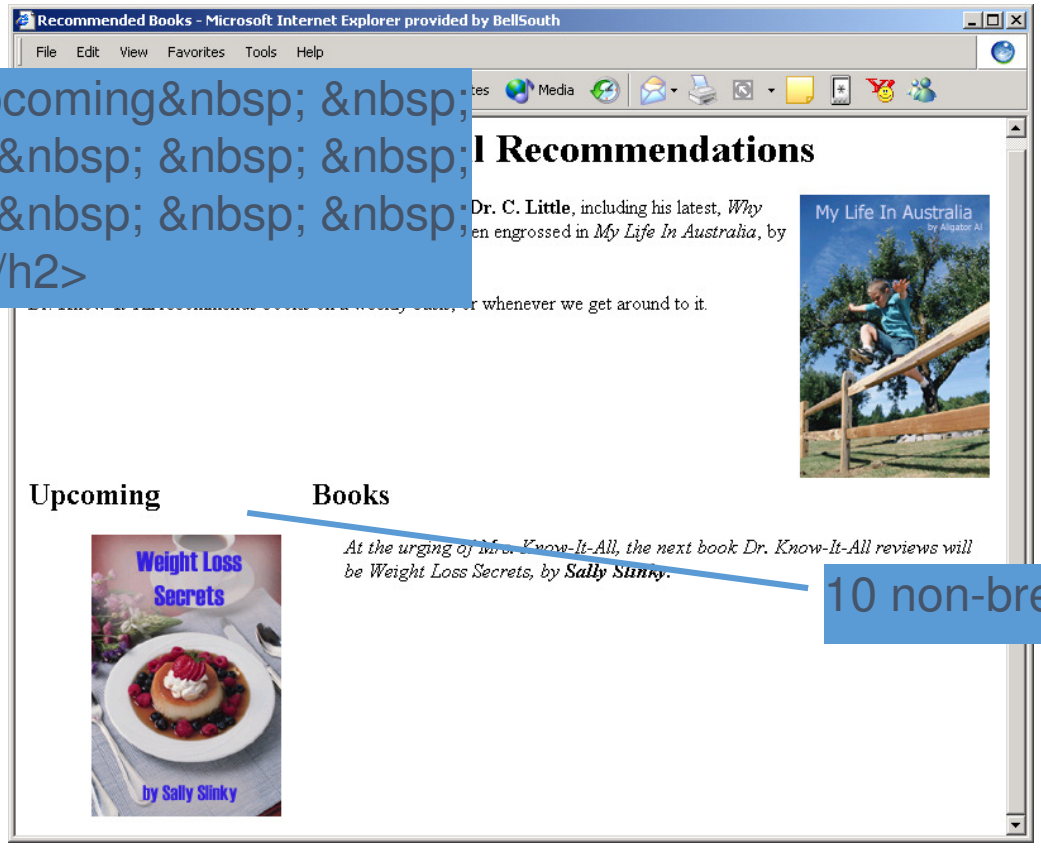


20 pixels of vertical space above the image

50 pixels of horizontal space to the left and right of the image

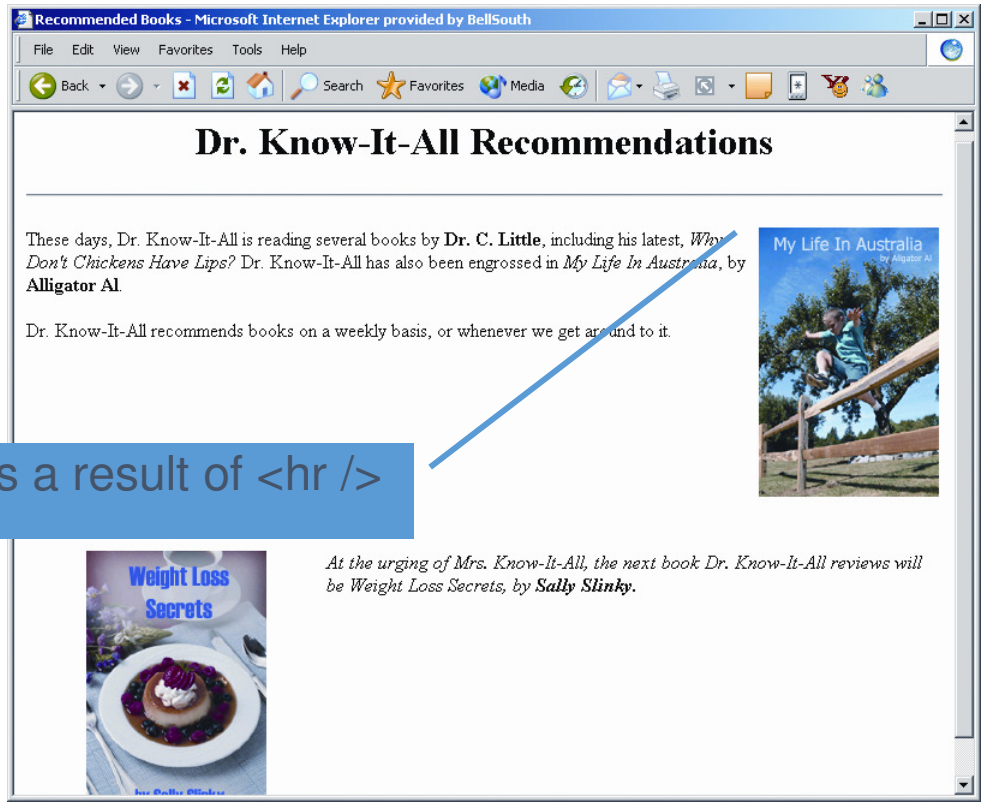
Effect of Non-Breaking Space

<h2>Upcoming   Books</h2>



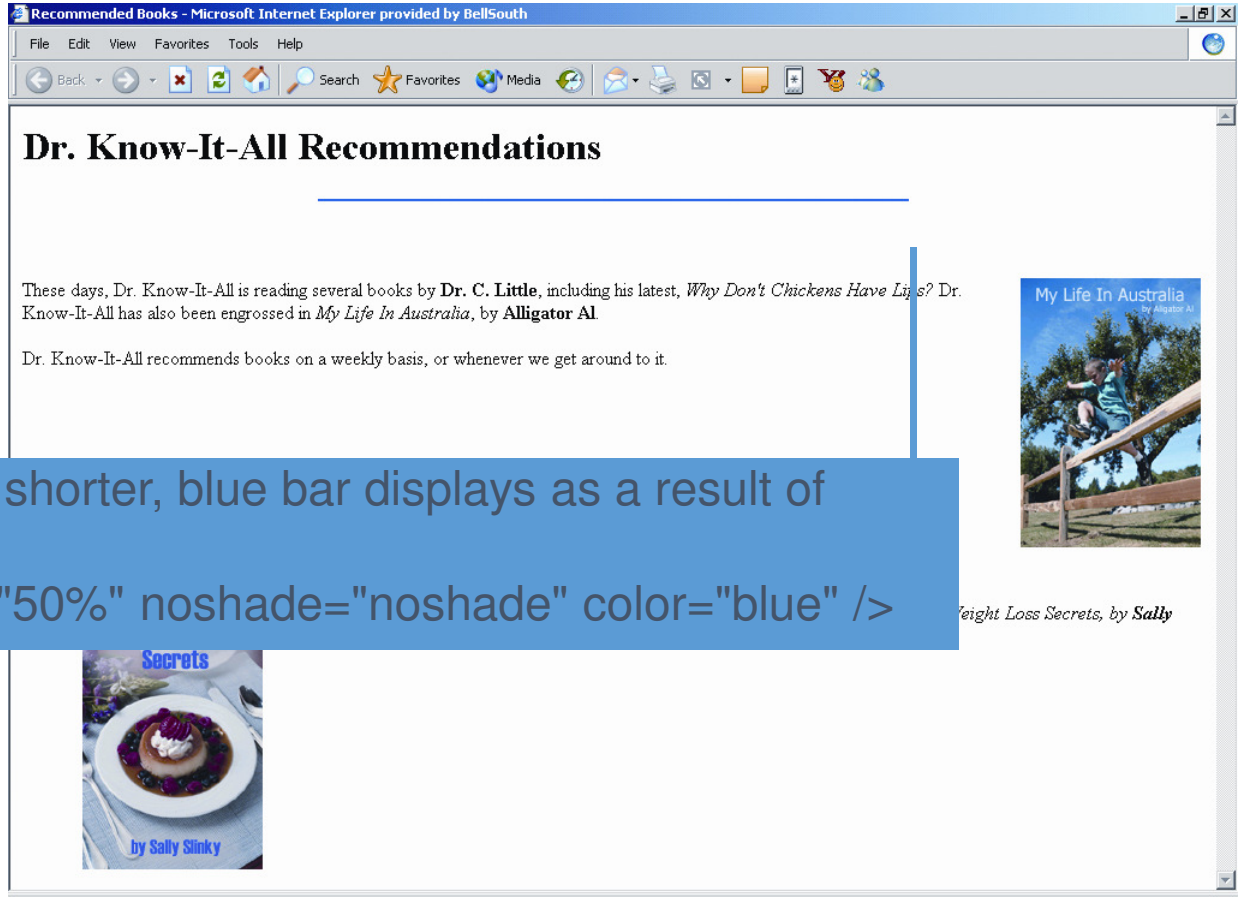
10 non-breaking spaces

Effects of Horizontal Rule



Grey bar displays as a result of <hr />

Effects of Horizontal Rule Attributes

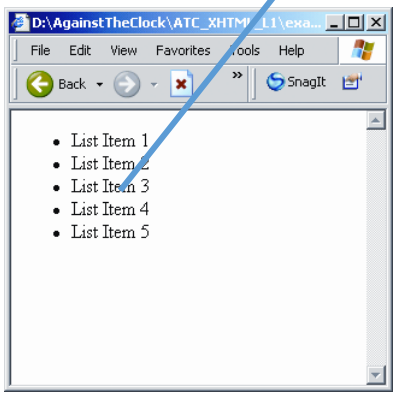


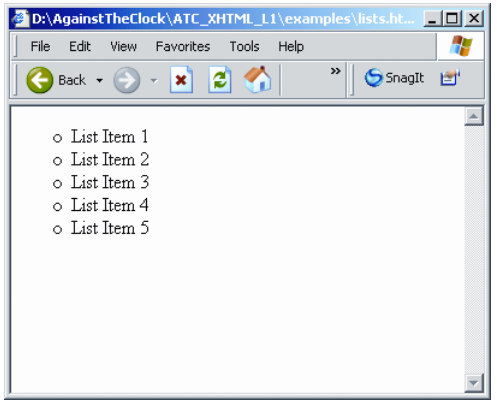
Unshaded, shorter, blue bar displays as a result of

```
<hr  width="50%" noshade="noshade" color="blue" />
```

Effects of Unordered Lists

```
<ul>
<li> List Item 1</li>
<li> List Item 2</li>
<li> List Item 3</li>
<li> List Item 4</li>
<li> List Item 5</li>
</ul>
```

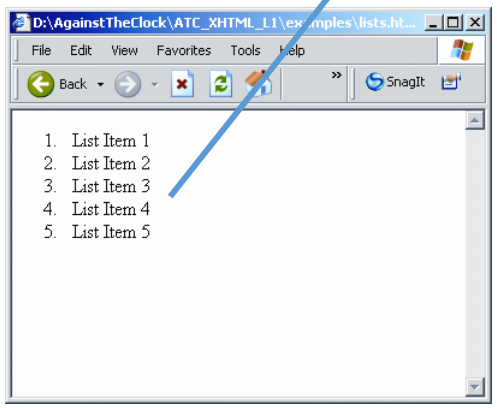


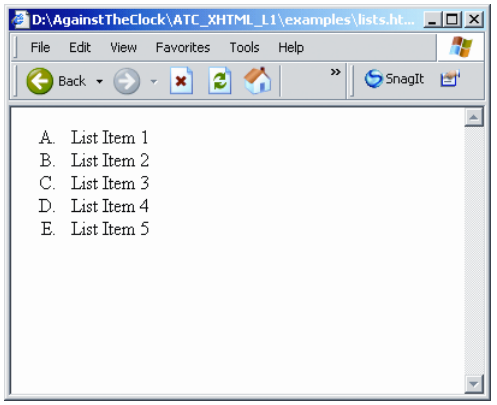


<ul type="circle"> affects the type of bullet

Effects of Ordered Lists

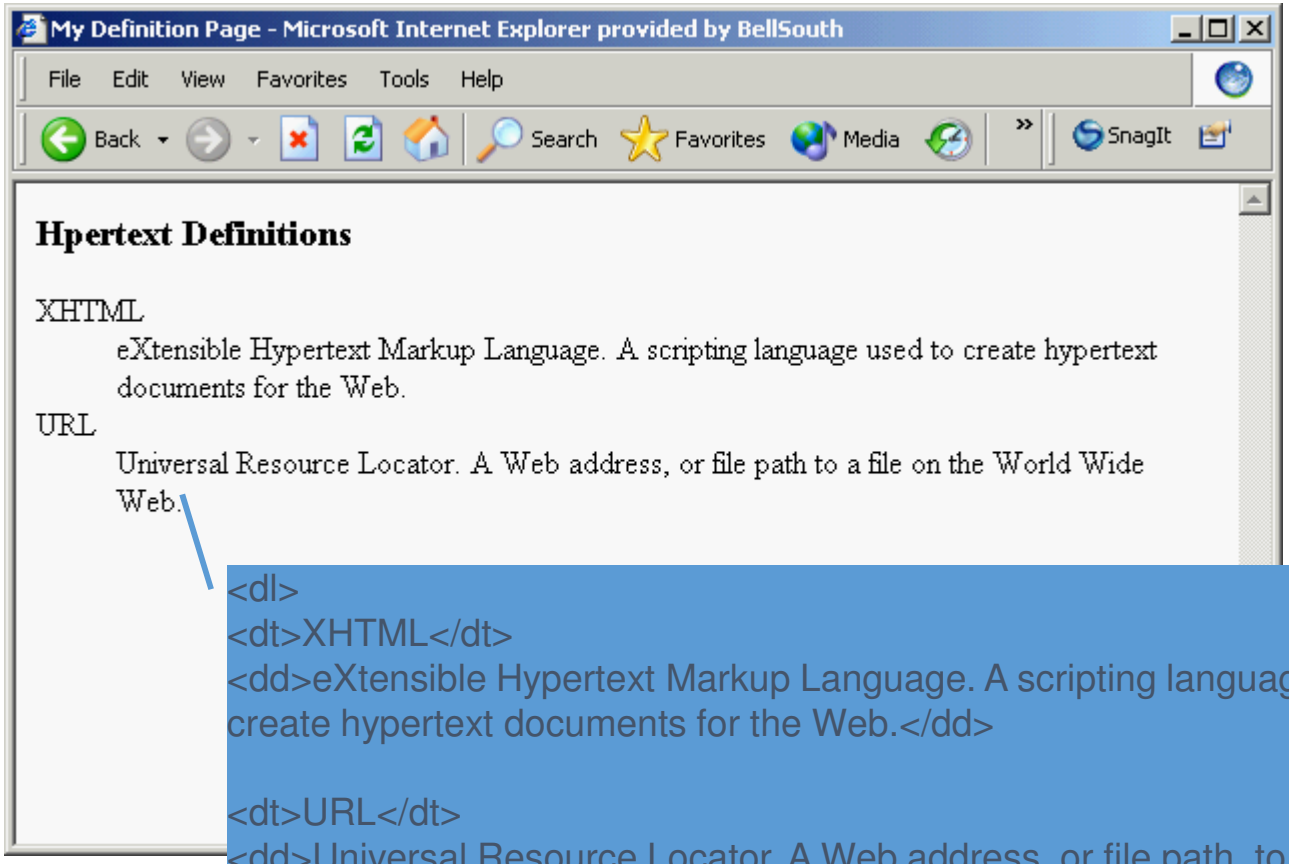
```
<ol>
<li> List Item 1</li>
<li> List Item 2</li>
<li> List Item 3</li>
<li> List Item 4</li>
<li> List Item 5</li>
</ol>
```





<ol type="A"> causes uppercase alphabetic listing

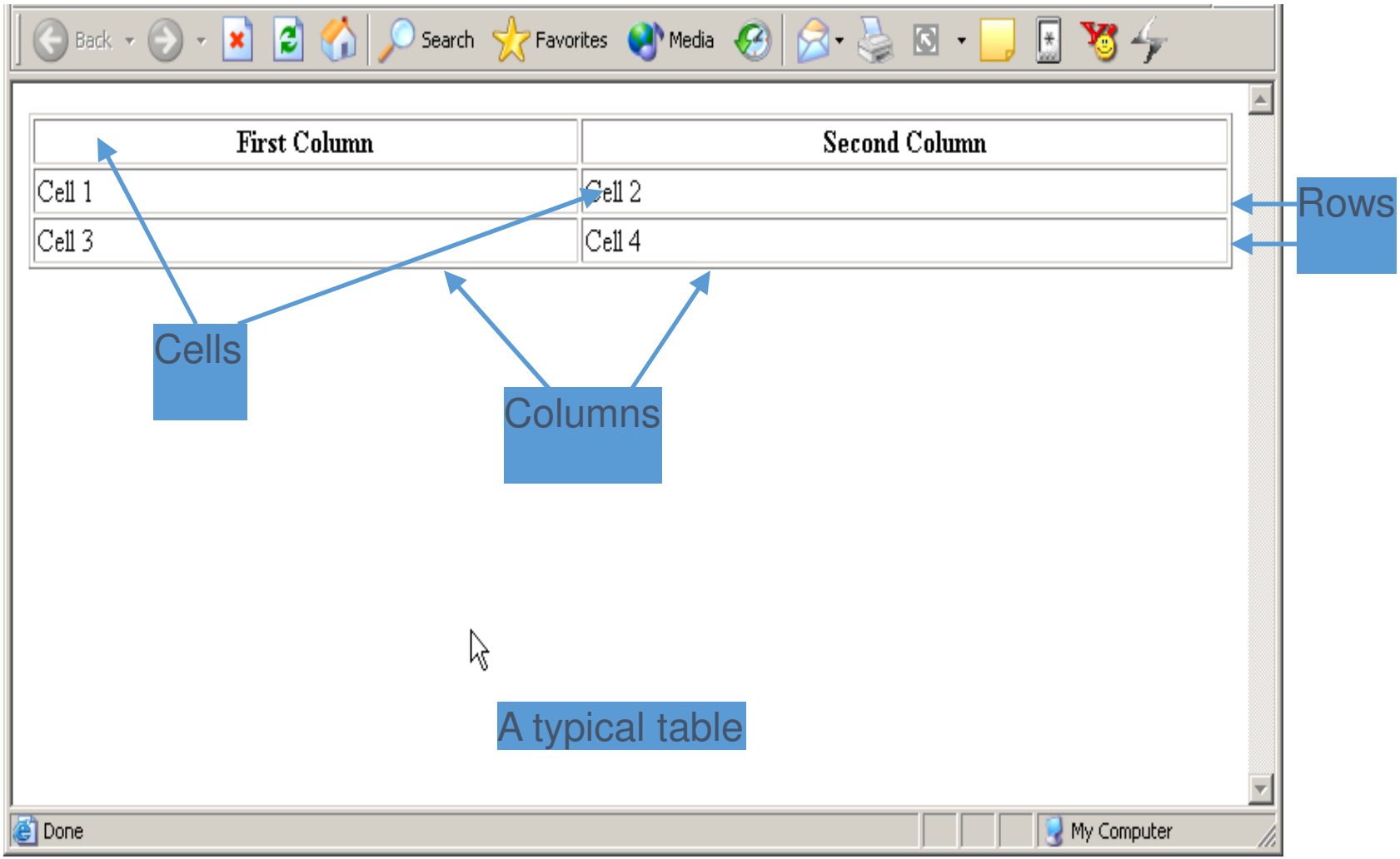
Effects of Definition Lists



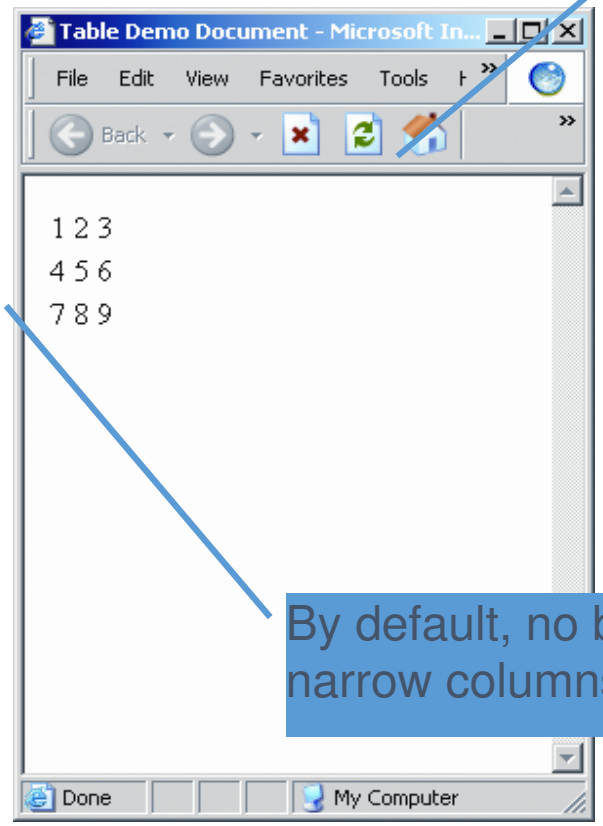

```
<dl>
<dt>XHTML</dt>
<dd>eXtensible Hypertext Markup Language. A scripting language used to create hypertext documents for the Web.</dd>

<dt>URL</dt>
<dd>Universal Resource Locator. A Web address, or file path  to a file on the World Wide Web.</dd>
</dl>
```

Visual Summary



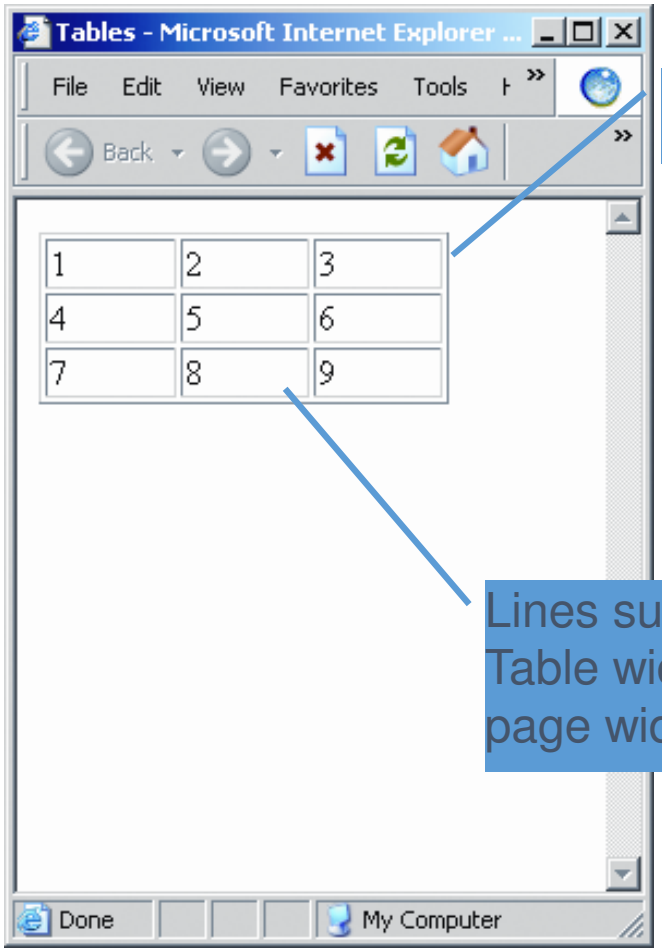
Effects of <table>, <tr>, and <td> Tags




```
<html>
<head><title>Tables</title></head>
<body>
<table>
   <tr> <td>1</td> <td>2</td>
<td>3</td> </tr>
   <tr> <td>4</td> <td>5</td>
<td>6</td> </tr>
   <tr> <td>7</td> <td>8</td>
<td>9</td> </tr>
</table>
</body>
</html>
```

By default, no border and narrow columns

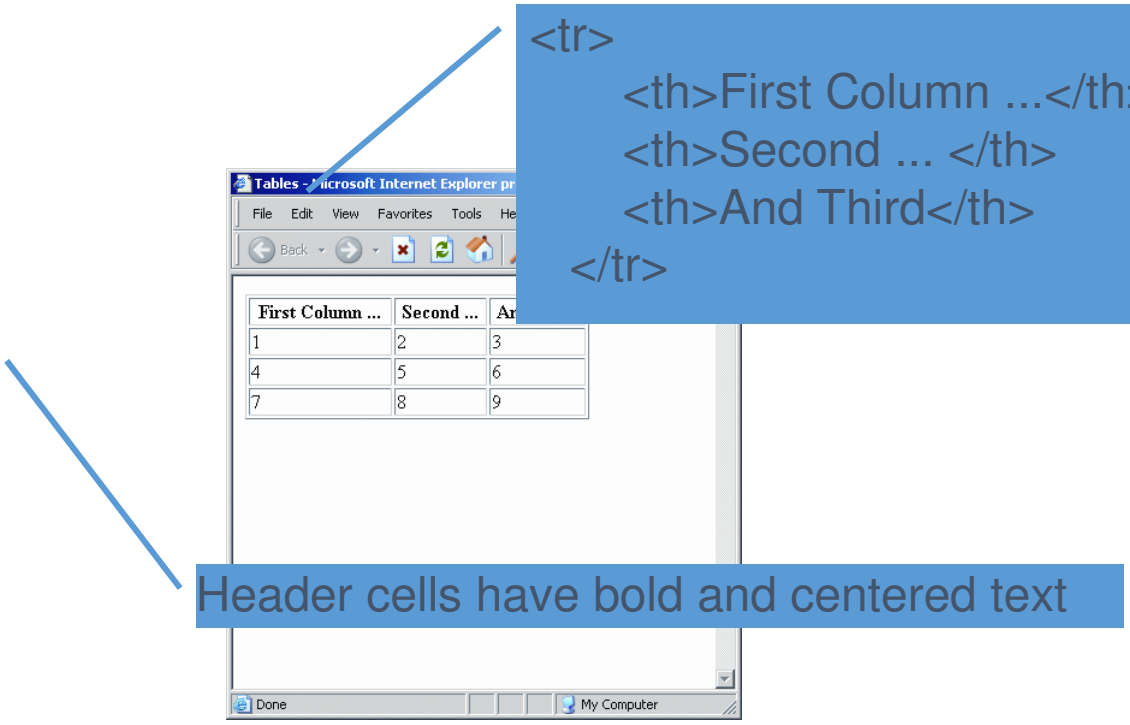
Effects of *border* and *width* attributes



<table **border="1" width="75%"**>

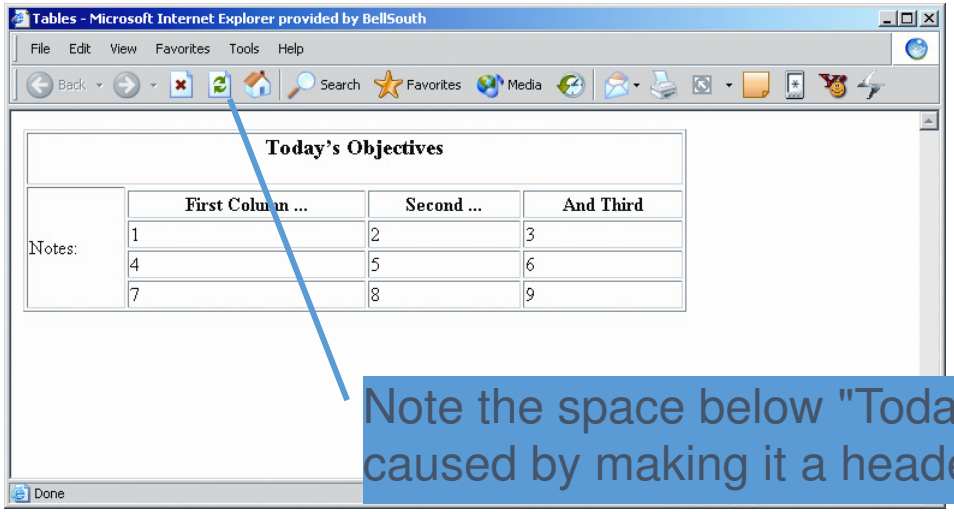
Lines surround cells
Table width takes up ¾ of page width

Effects of *border* and *width* attributes



Effects of Column and Row Spanning

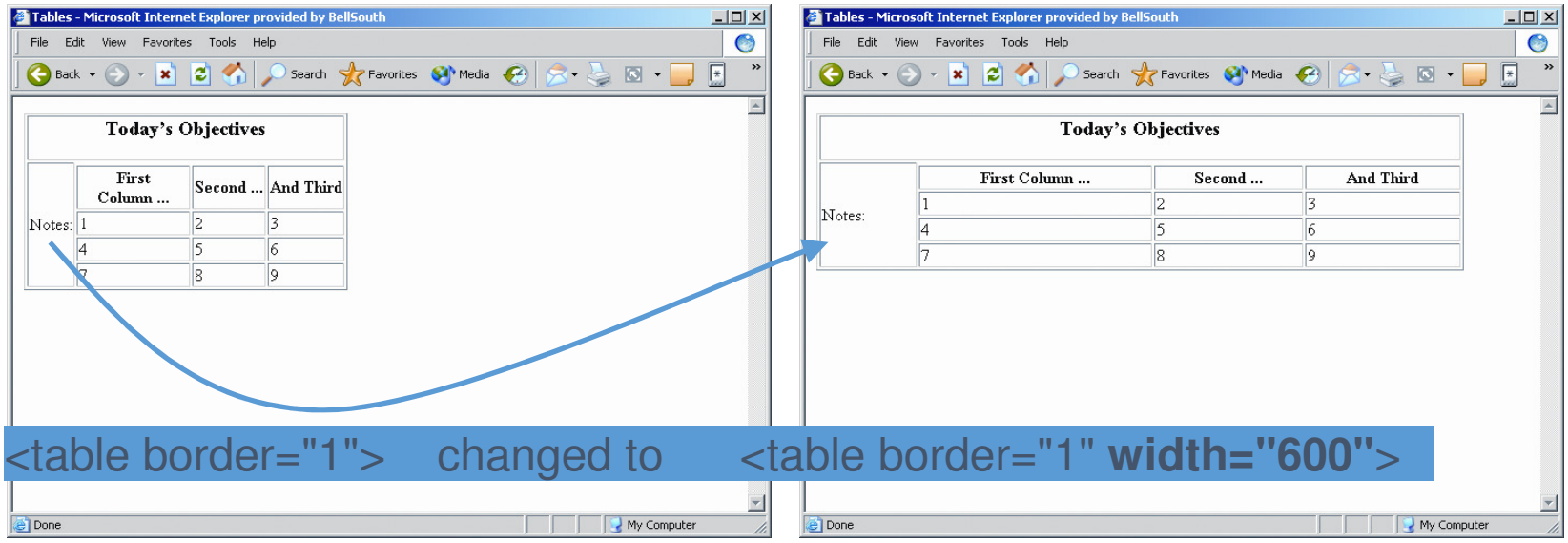
Column spanning:
<td **colspan="4"**><h3 align="center">Today's Objectives</h3></td>



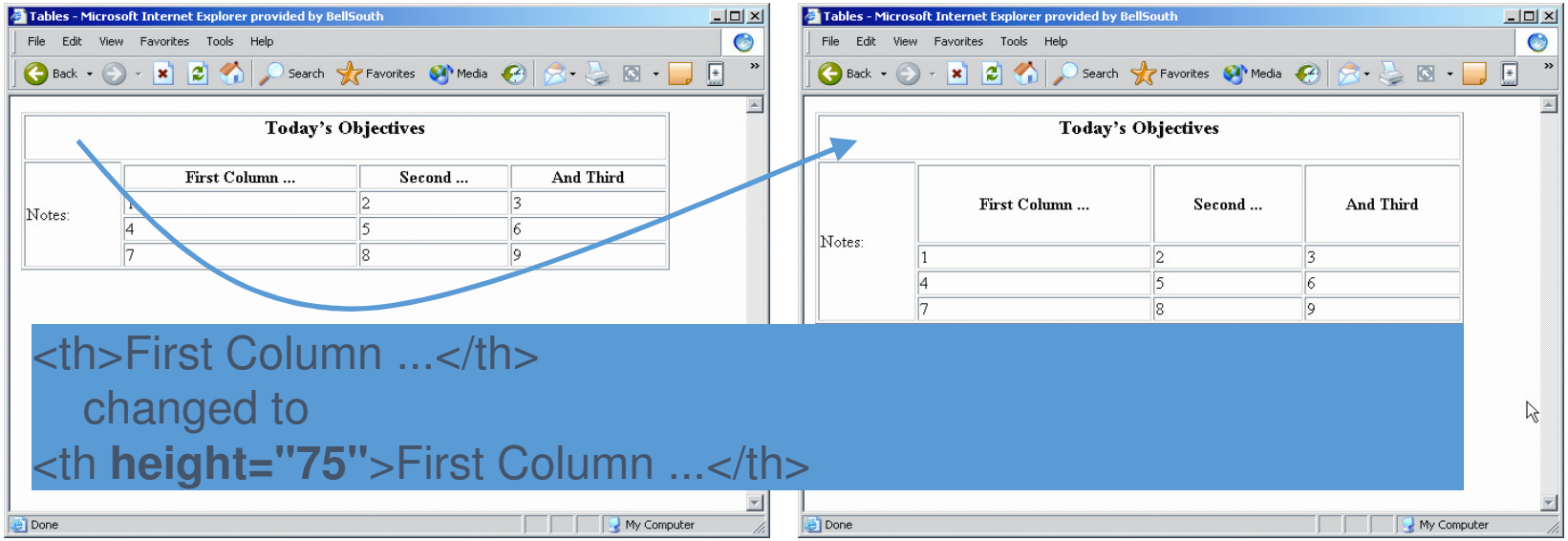
Note the space below "Today's Objectives", caused by making it a header (h3)

Row spanning:
<td rowspan="5">Notes:</td>

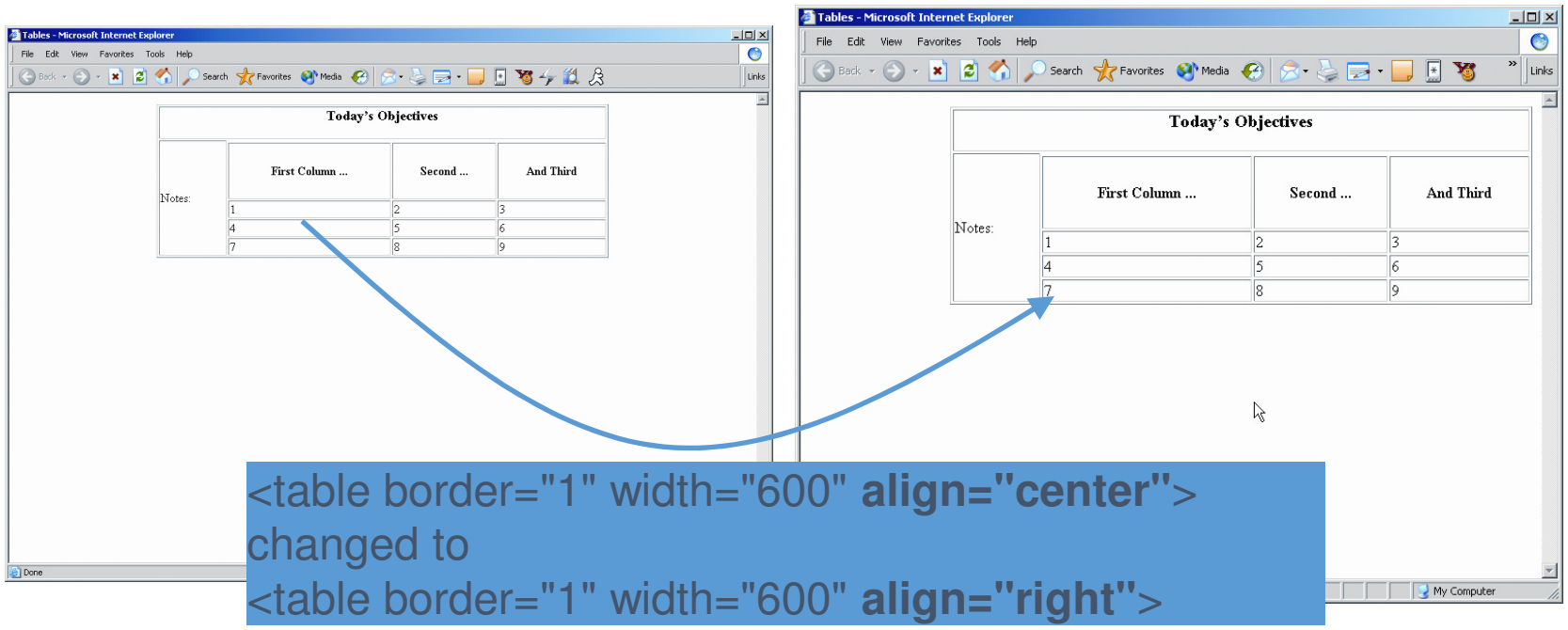
Effect of Changing Table Size



Effect of Changing Cell Size

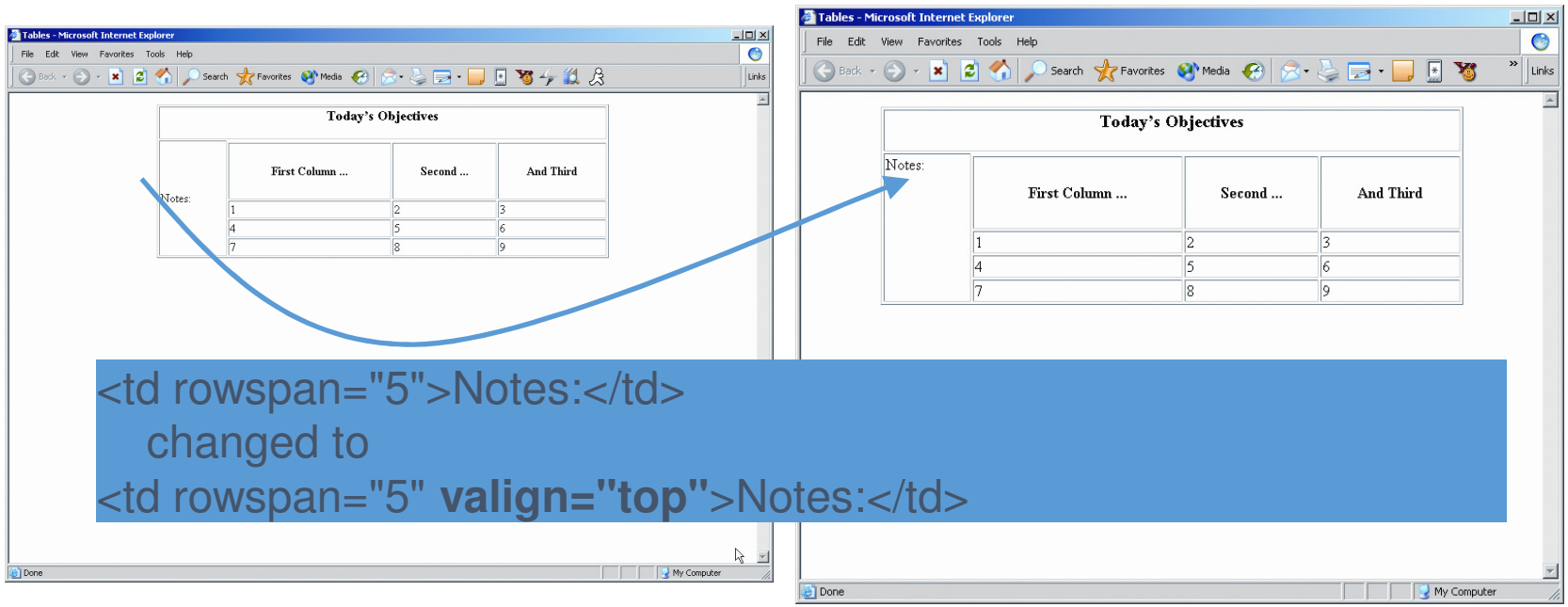


Effect of Table Alignment



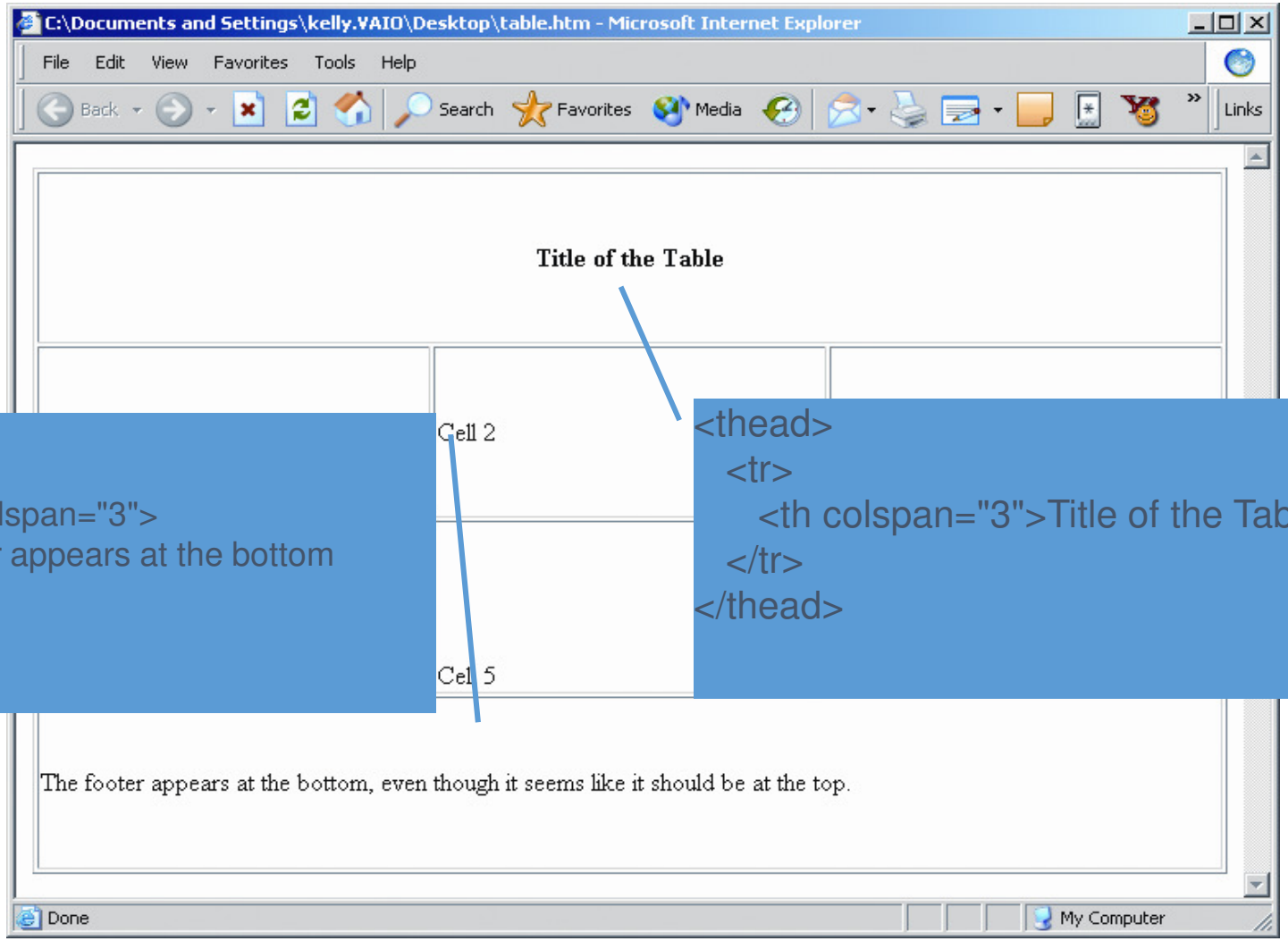
<table border="1" width="600" **align="center"**>
changed to
<table border="1" width="600" **align="right"**>

Effect of Cell Data Alignment




```
<td rowspan="5">Notes:</td>
   changed to
<td rowspan="5" valign="top">Notes:</td>
```

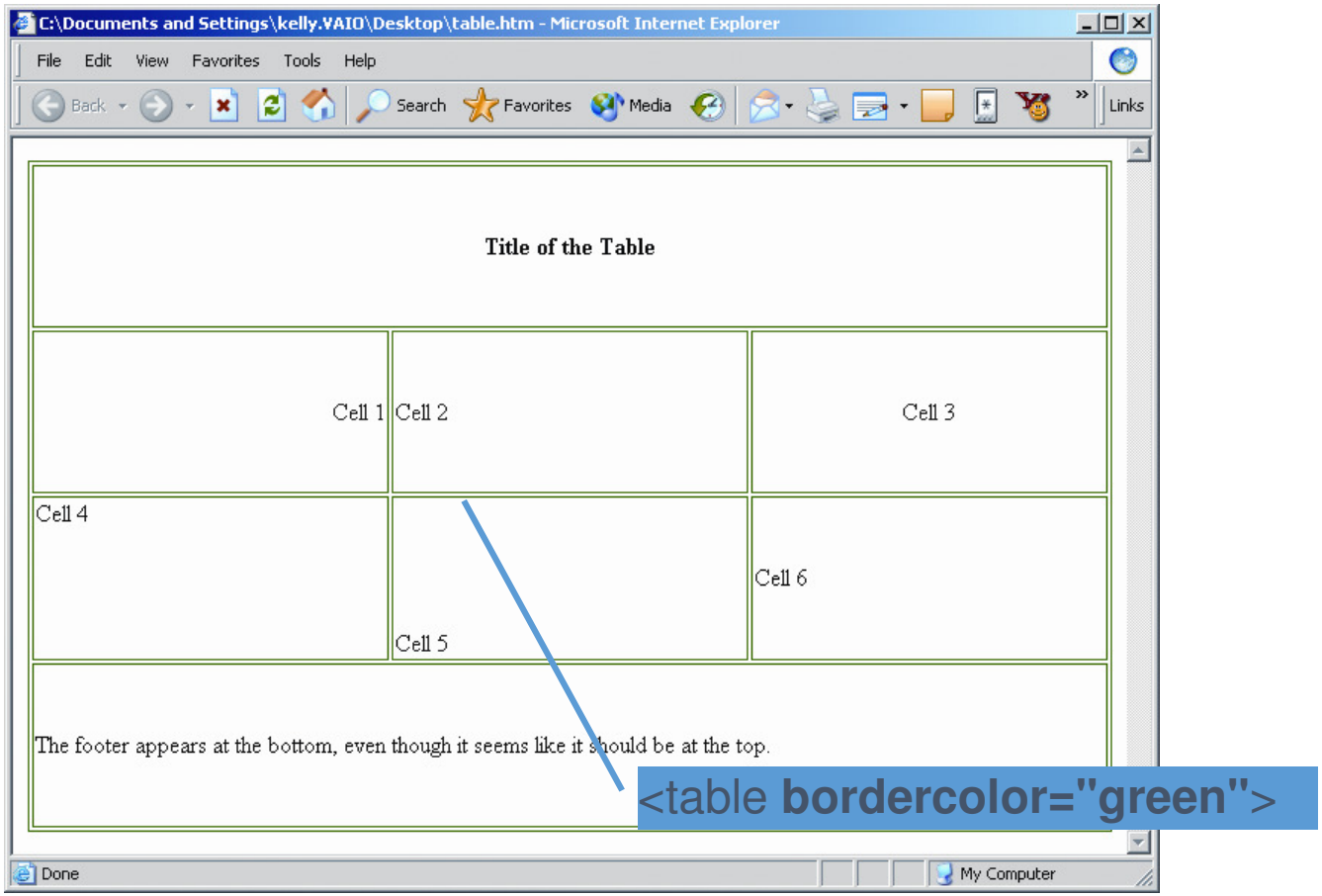
Effects of Headers and Footers



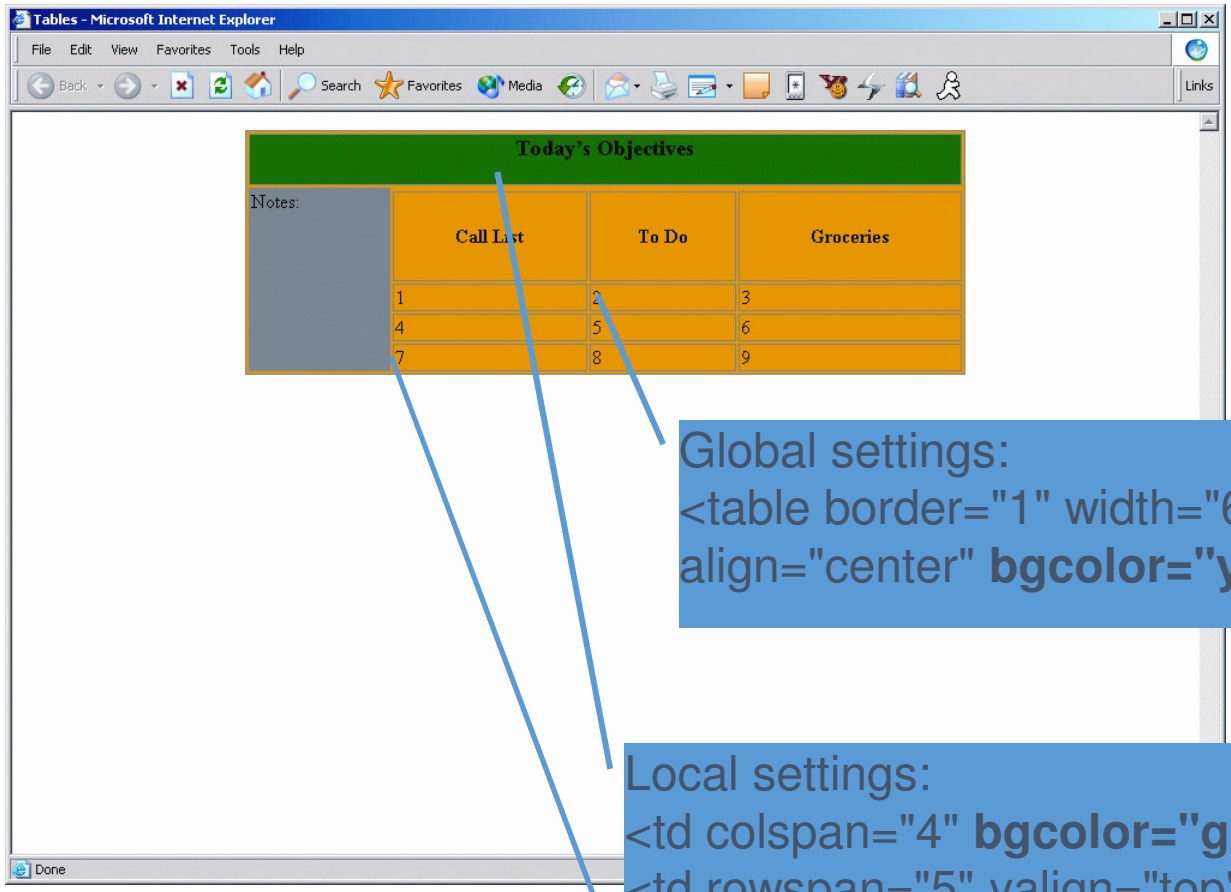

```
<tfoot>
  <tr>
    <td colspan="3">
The footer appears at the bottom
    </td>
  </tr>
</tfoot>
```

```
<thead>
  <tr>
    <th colspan="3">Title of the Table</th>
  </tr>
</thead>
```

Effects of Border Color Control



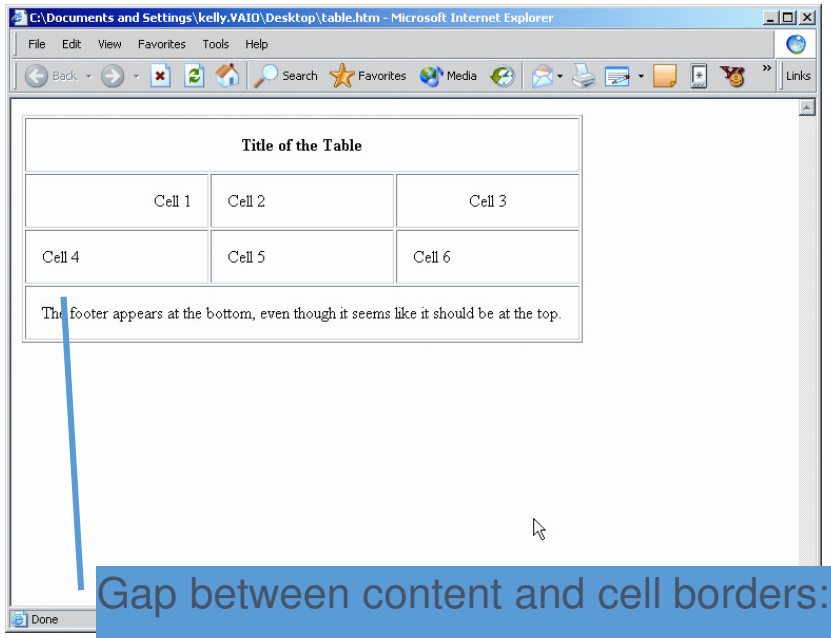
Effects of Background Color Control



Global settings:
<table border="1" width="600" align="center" **bgcolor="yellow"**>

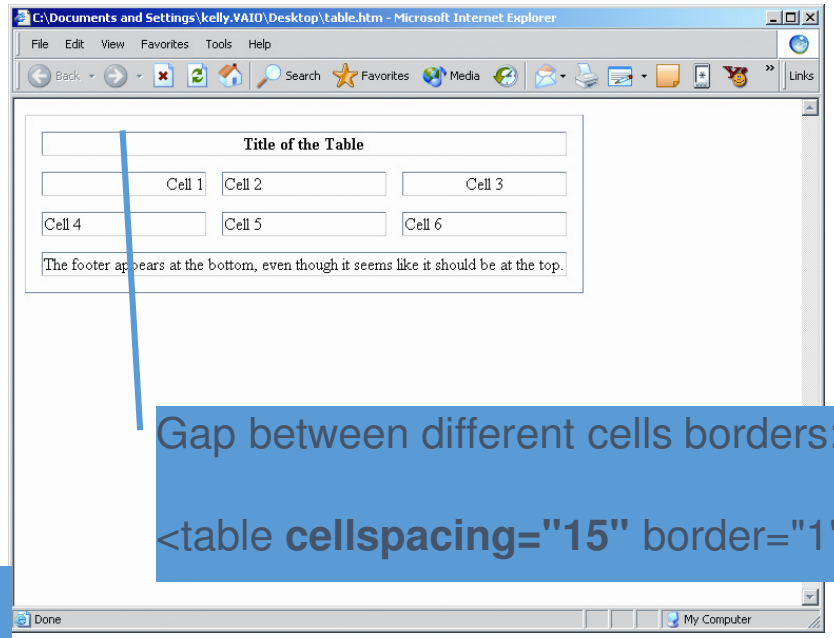
Local settings:
<td colspan="4" **bgcolor="green"**>
<td rowspan="5" valign="top" **bgcolor="gray"**>

Effect of Cellpadding and Spacing



Gap between content and cell borders:

<table **cellpadding="15"** border="1">

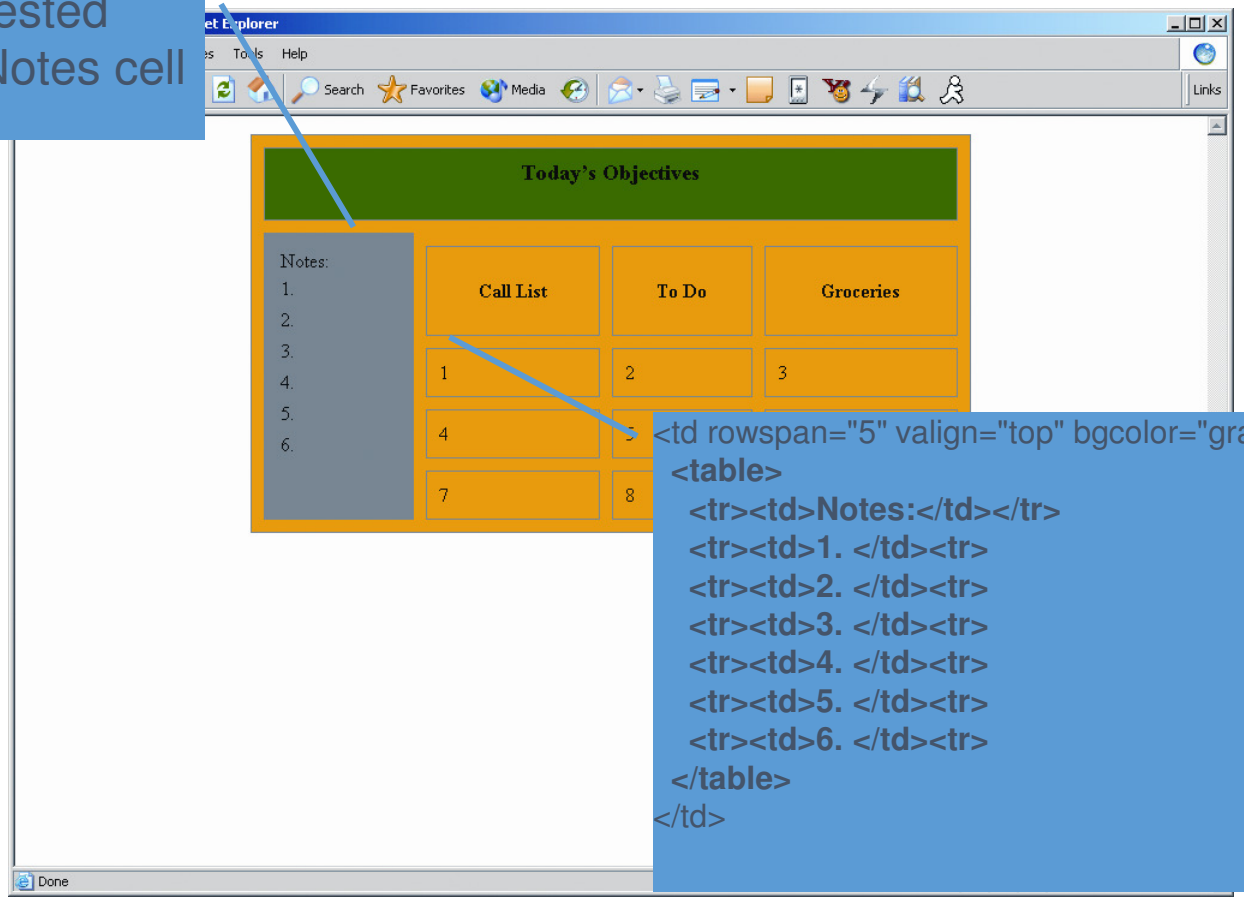


Gap between different cells borders:

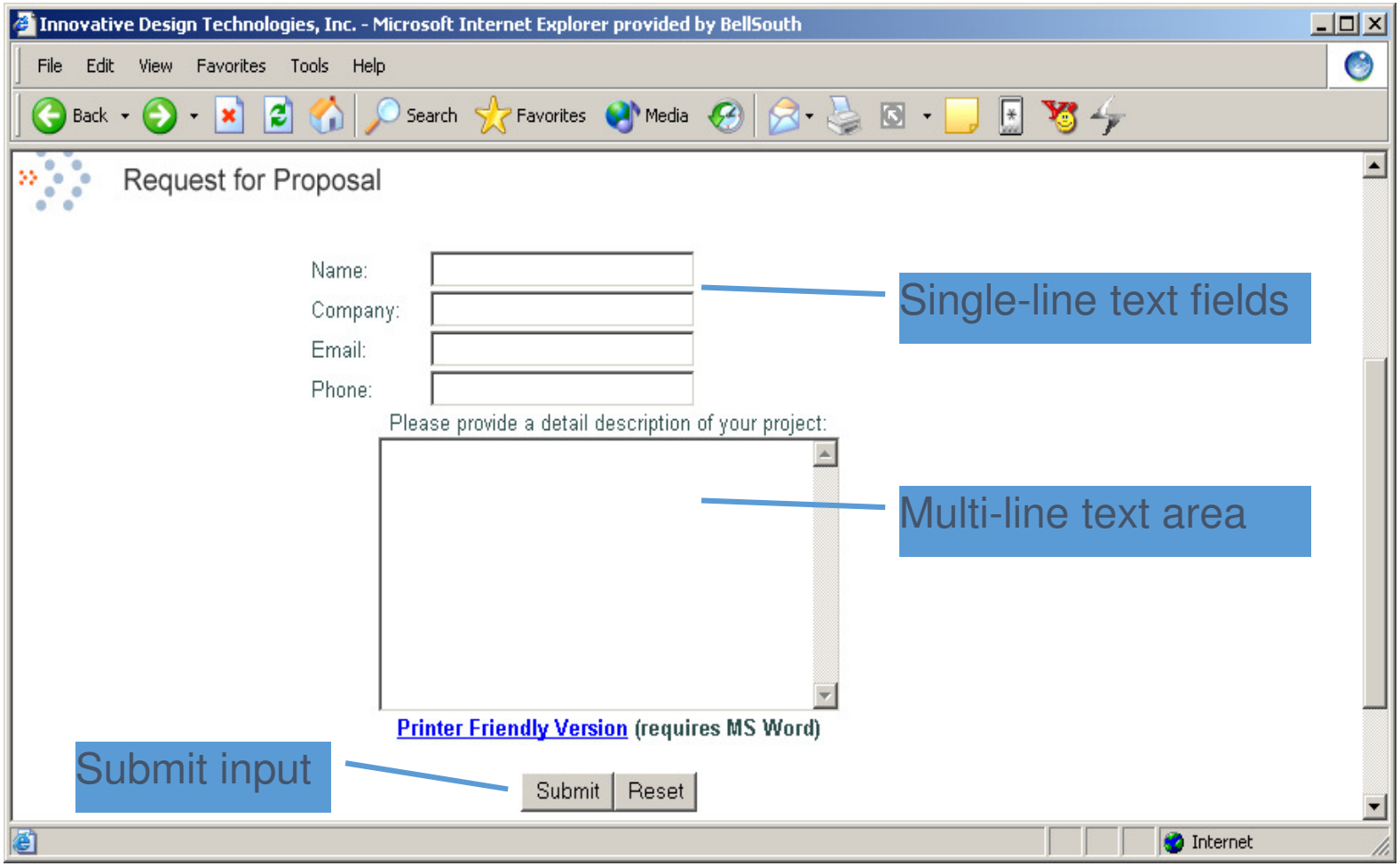
<table **cellspacing="15"** border="1">

Effects of Table Nesting

A table is nested inside the Notes cell

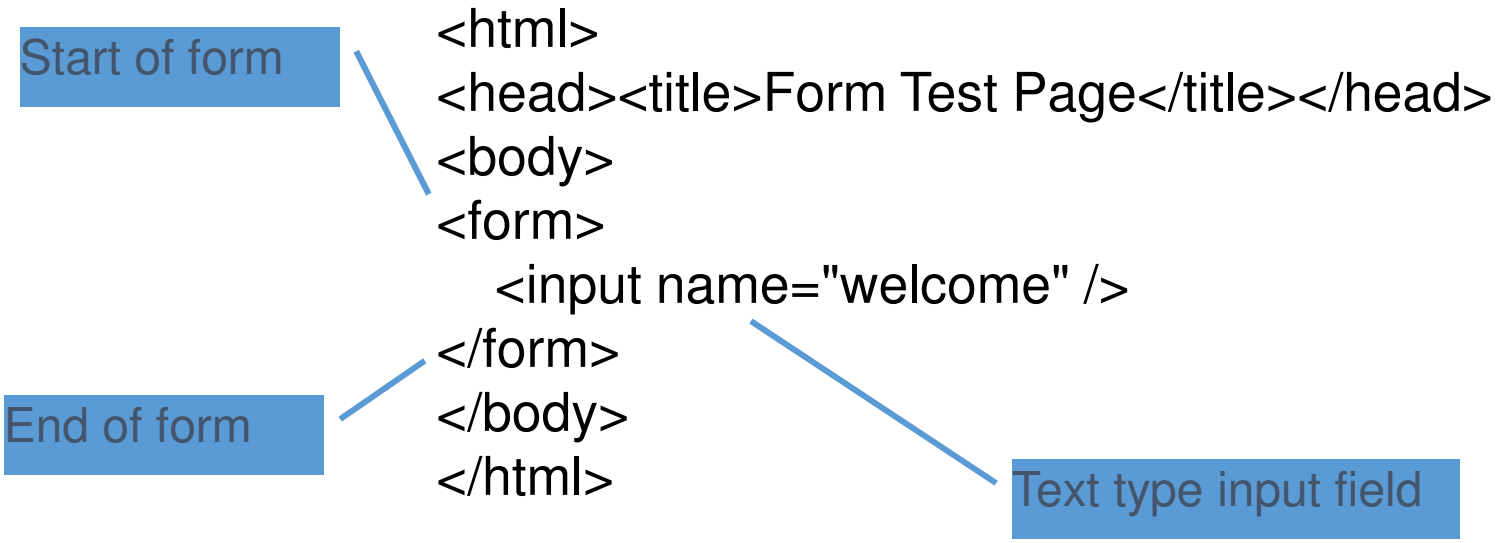



```
<td rowspan="5" valign="top" bgcolor="gray">
  <table>
    <tr><td>Notes:</td></tr>
    <tr><td>1. </td><tr>
    <tr><td>2. </td><tr>
    <tr><td>3. </td><tr>
    <tr><td>4. </td><tr>
    <tr><td>5. </td><tr>
    <tr><td>6. </td><tr>
  </table>
</td>
```

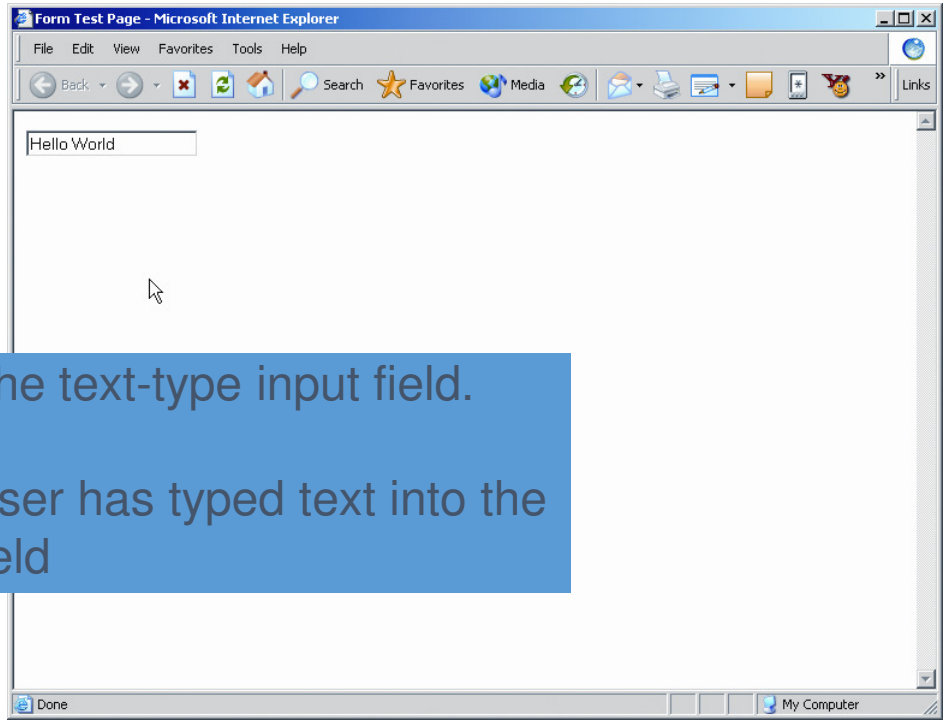


A typical form

Code for Form Page



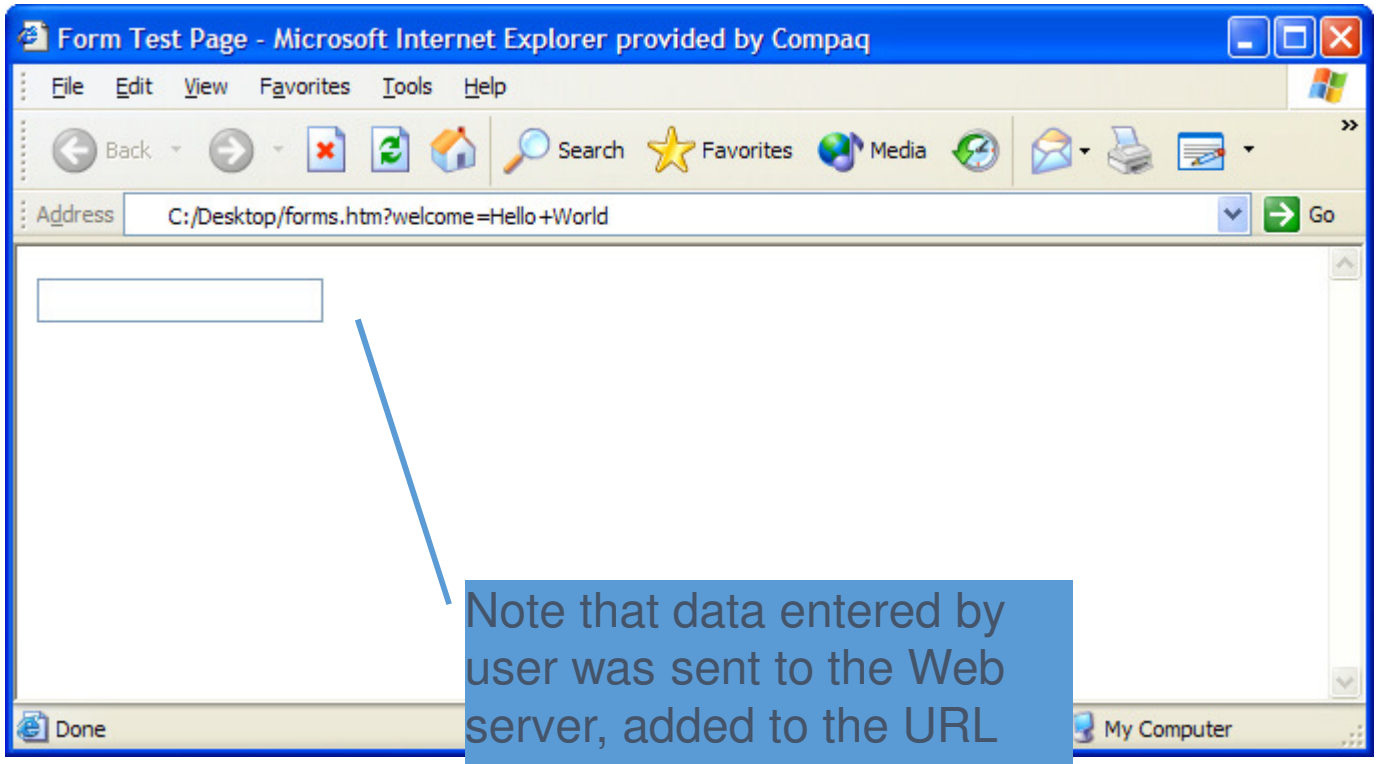
Effects of <form> and <input> (initial page)



The text-type input field.

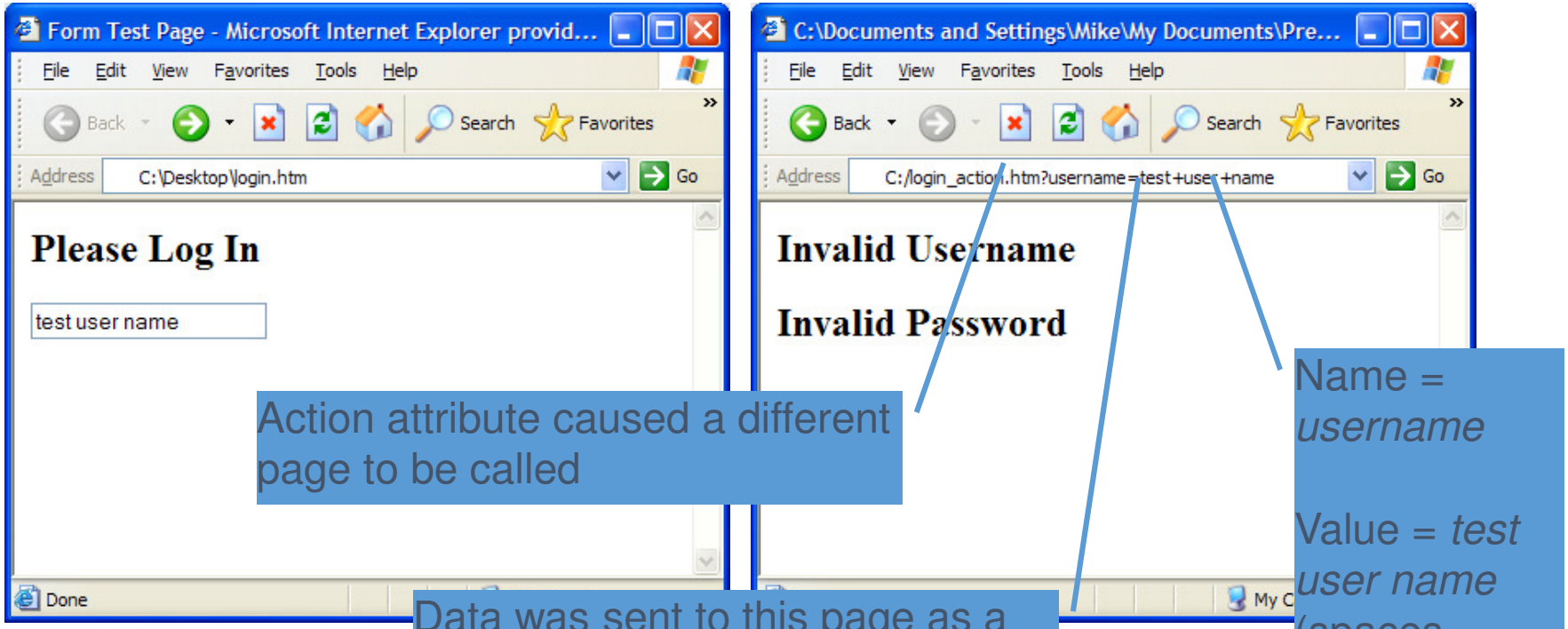
User has typed text into the field

Effects of <form> and <input> (after user types text and hits Enter)

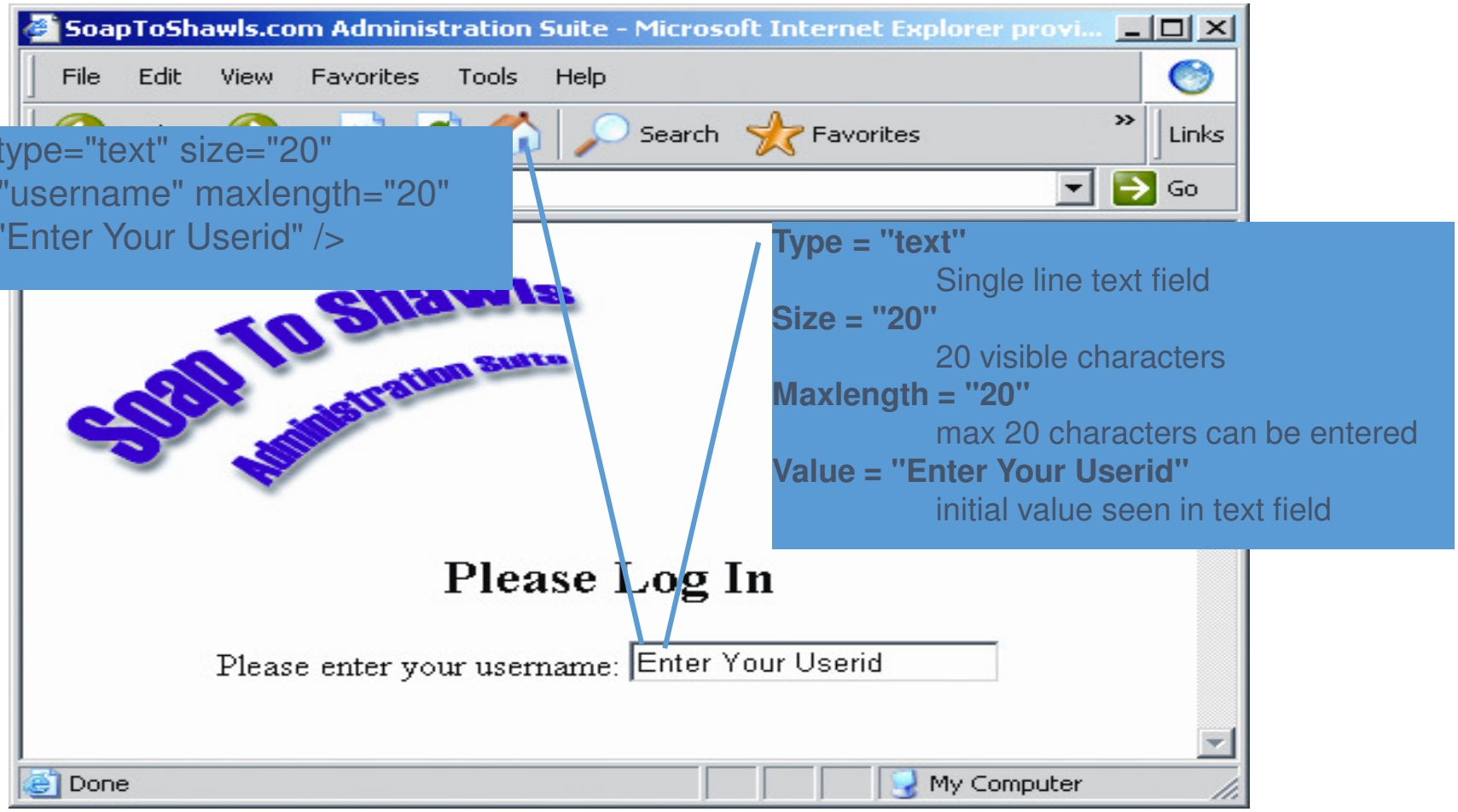


Effect of Using the *action* Attribute

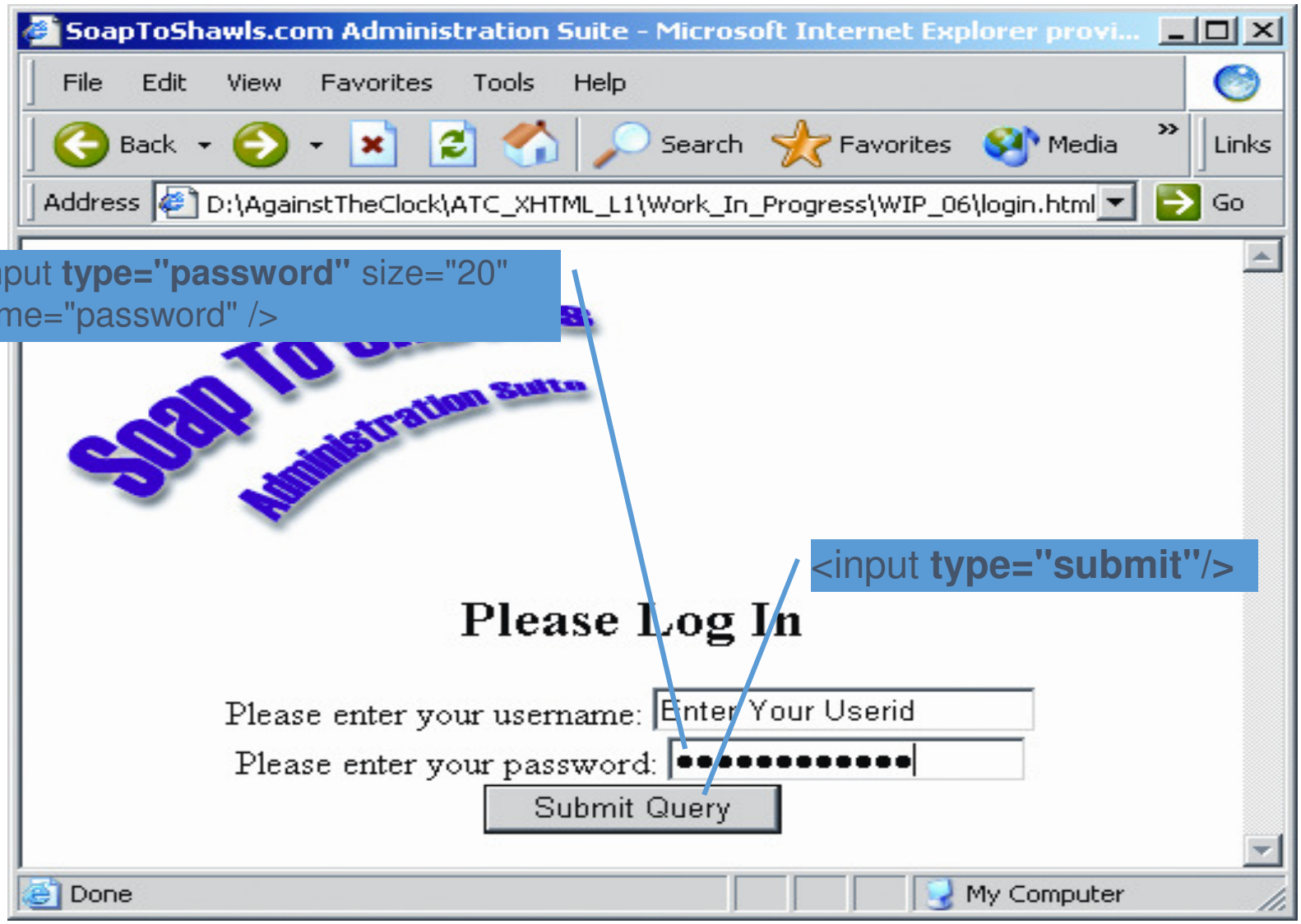
<form **action="login_action.htm"**>



Effect of Input Tag Attributes

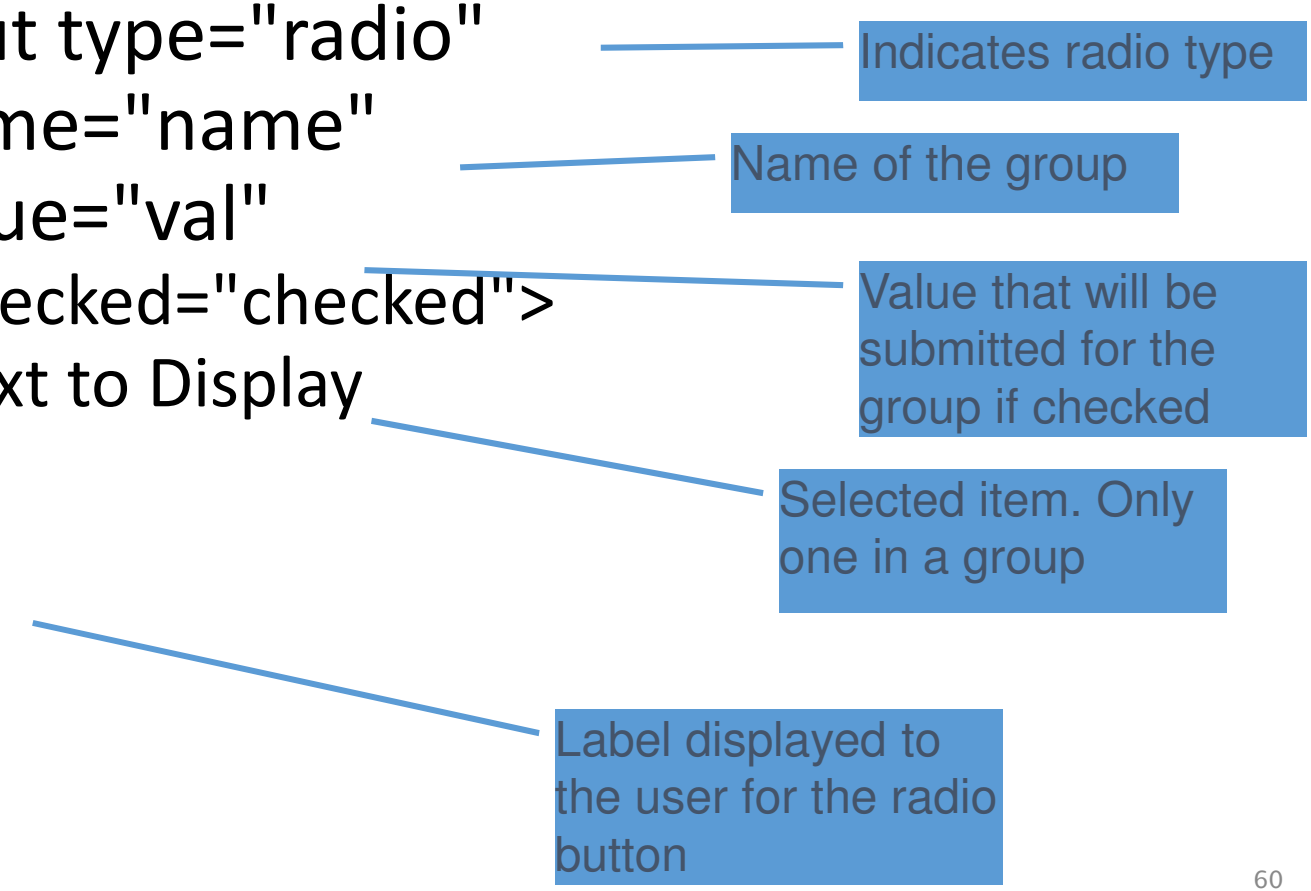


Effect of Input Password and Submit



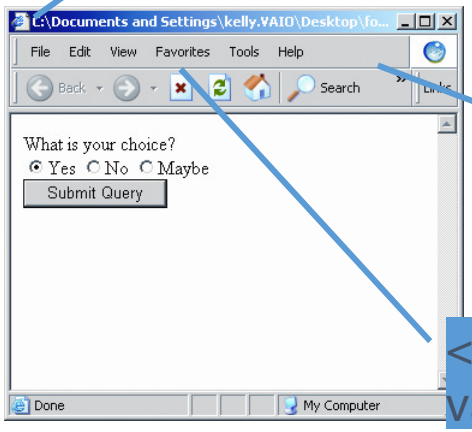
Radio Buttons Syntax

- <input type="radio"
  name="name"
  value="val"
  checked="checked">
  Text to Display



Effect of Radio Buttons

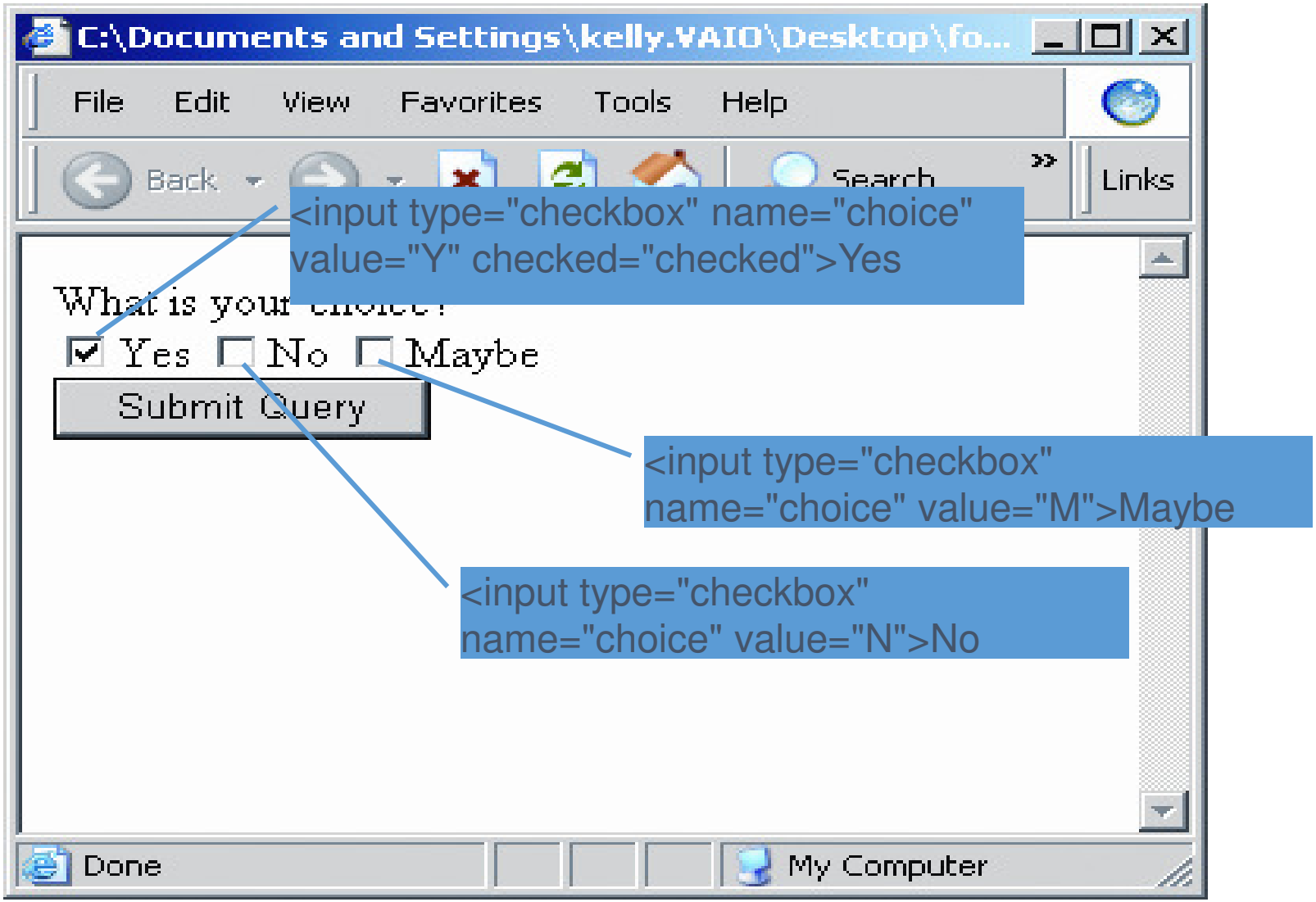
<input type="radio" name="choice" value="Y" checked="checked">Yes



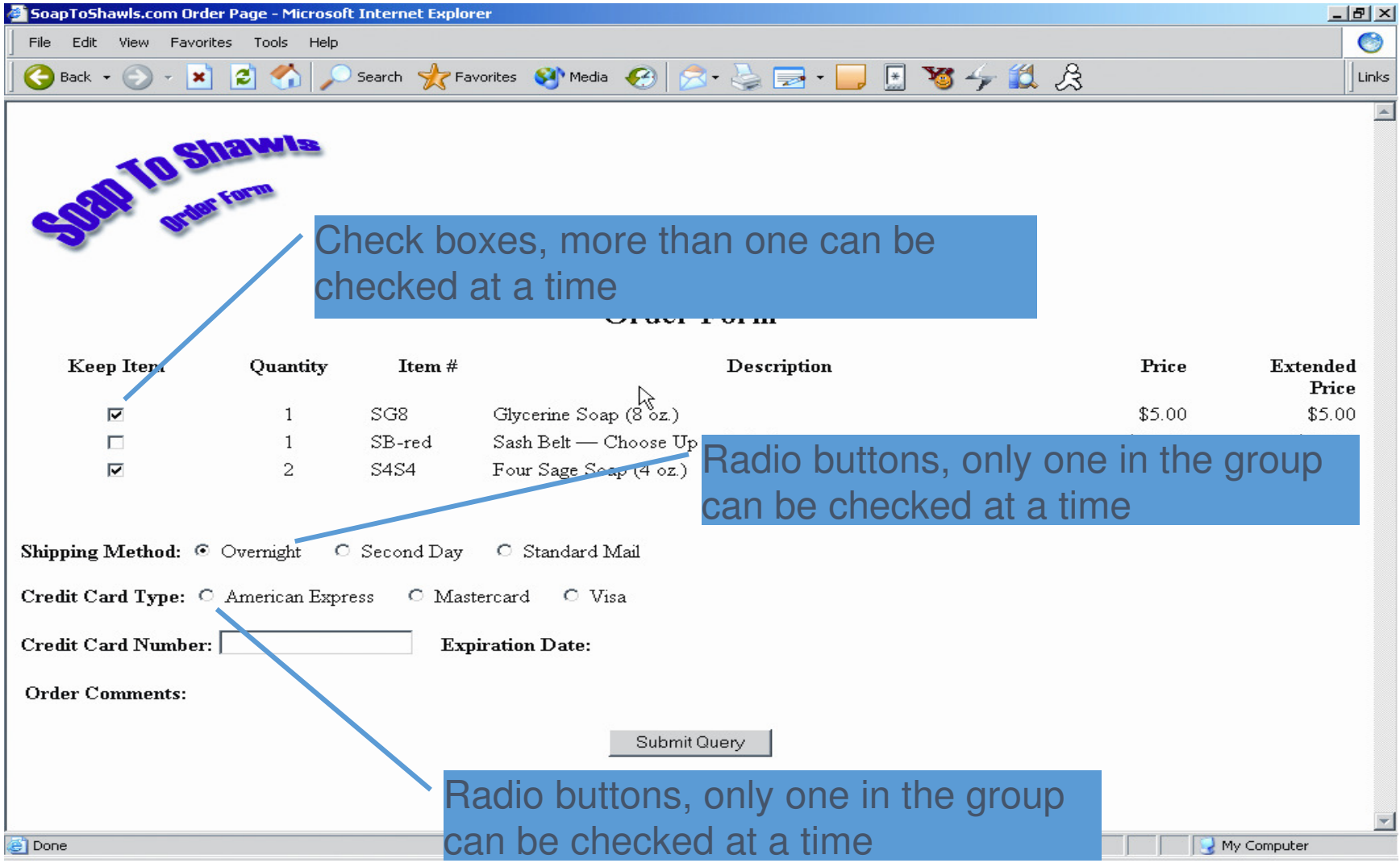
<input type="radio" name="choice" value="M">Maybe

<input type="radio" name="choice" value="N">No

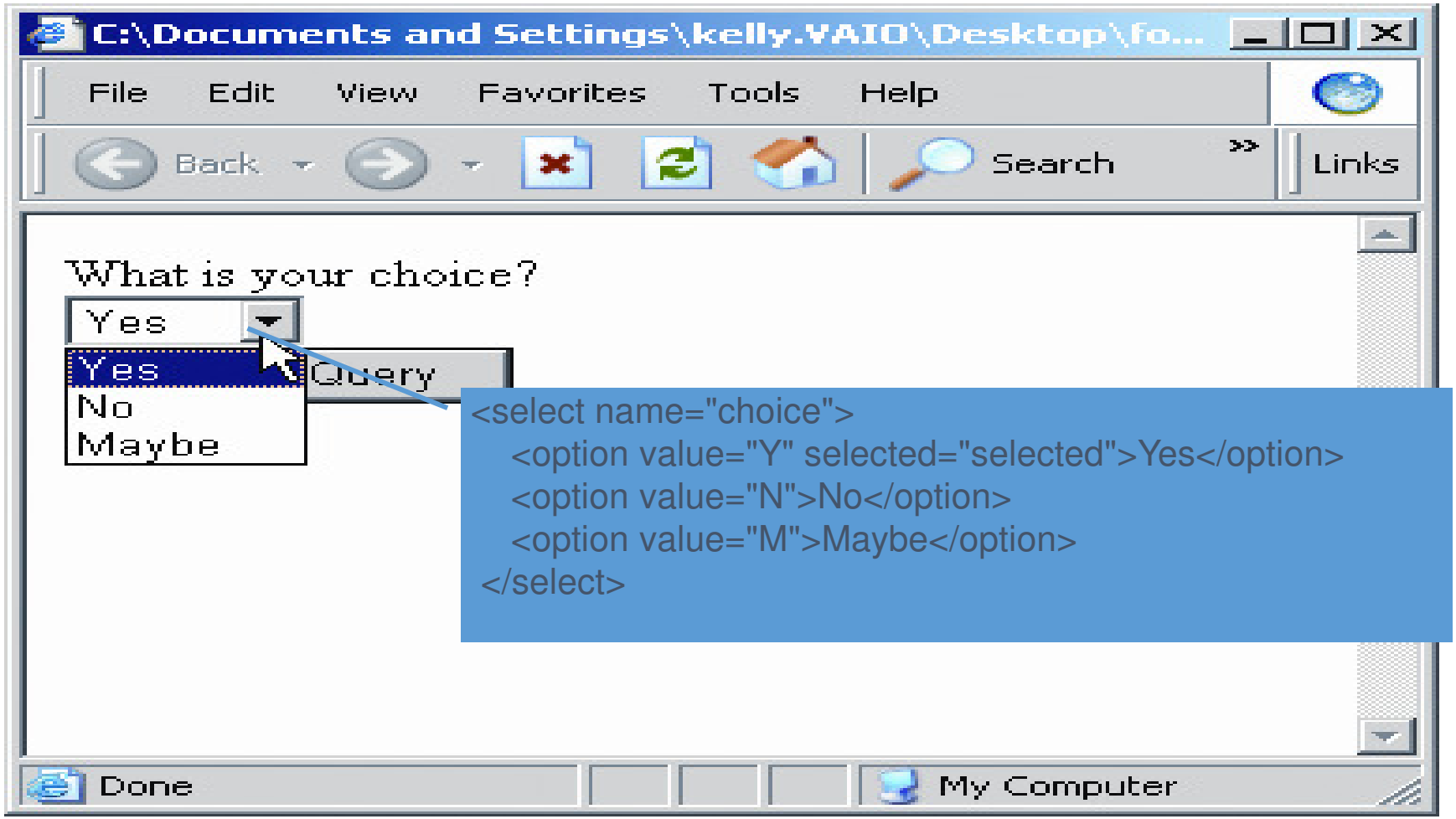
Effect of CheckBoxes



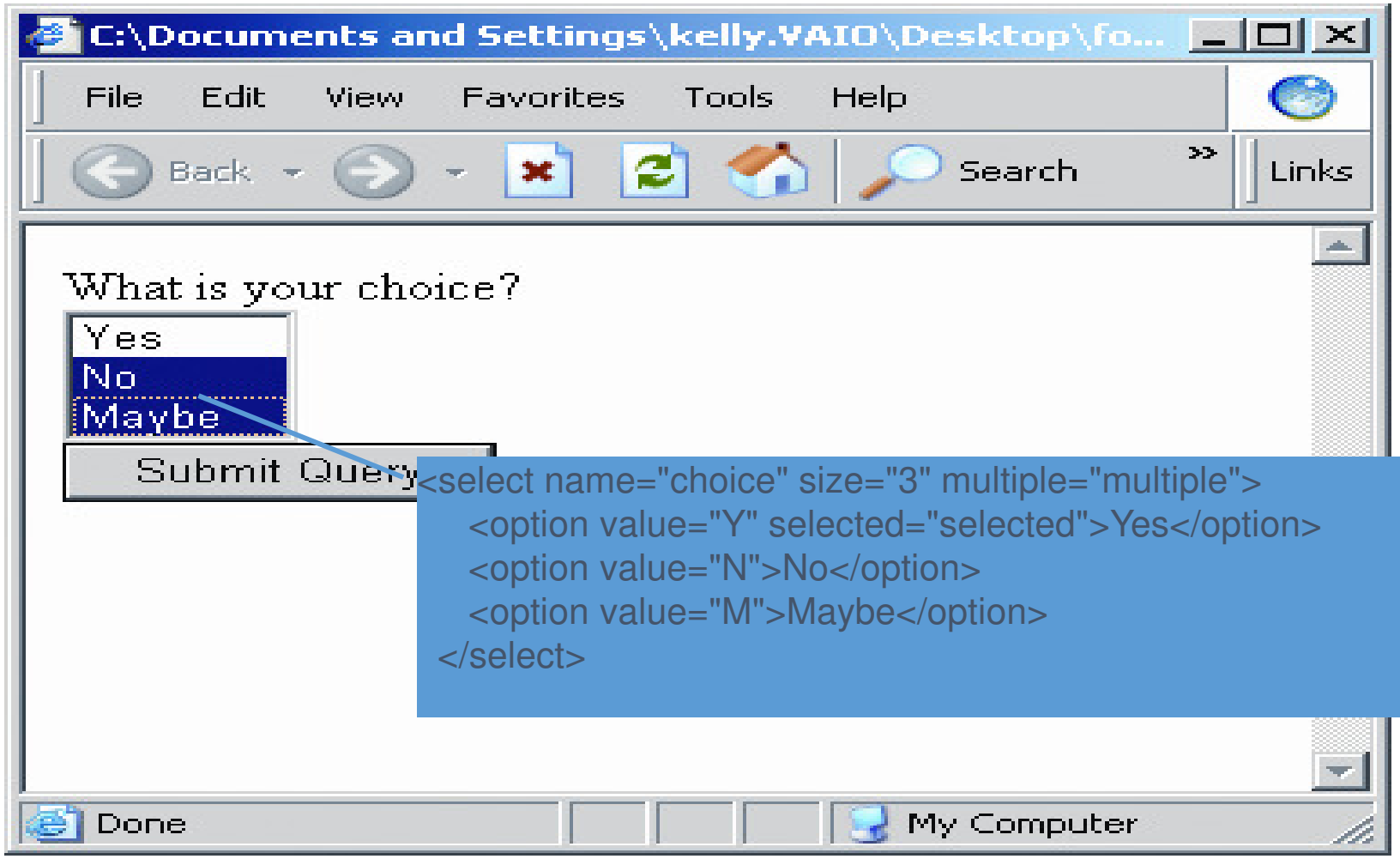
Creating an Order Form



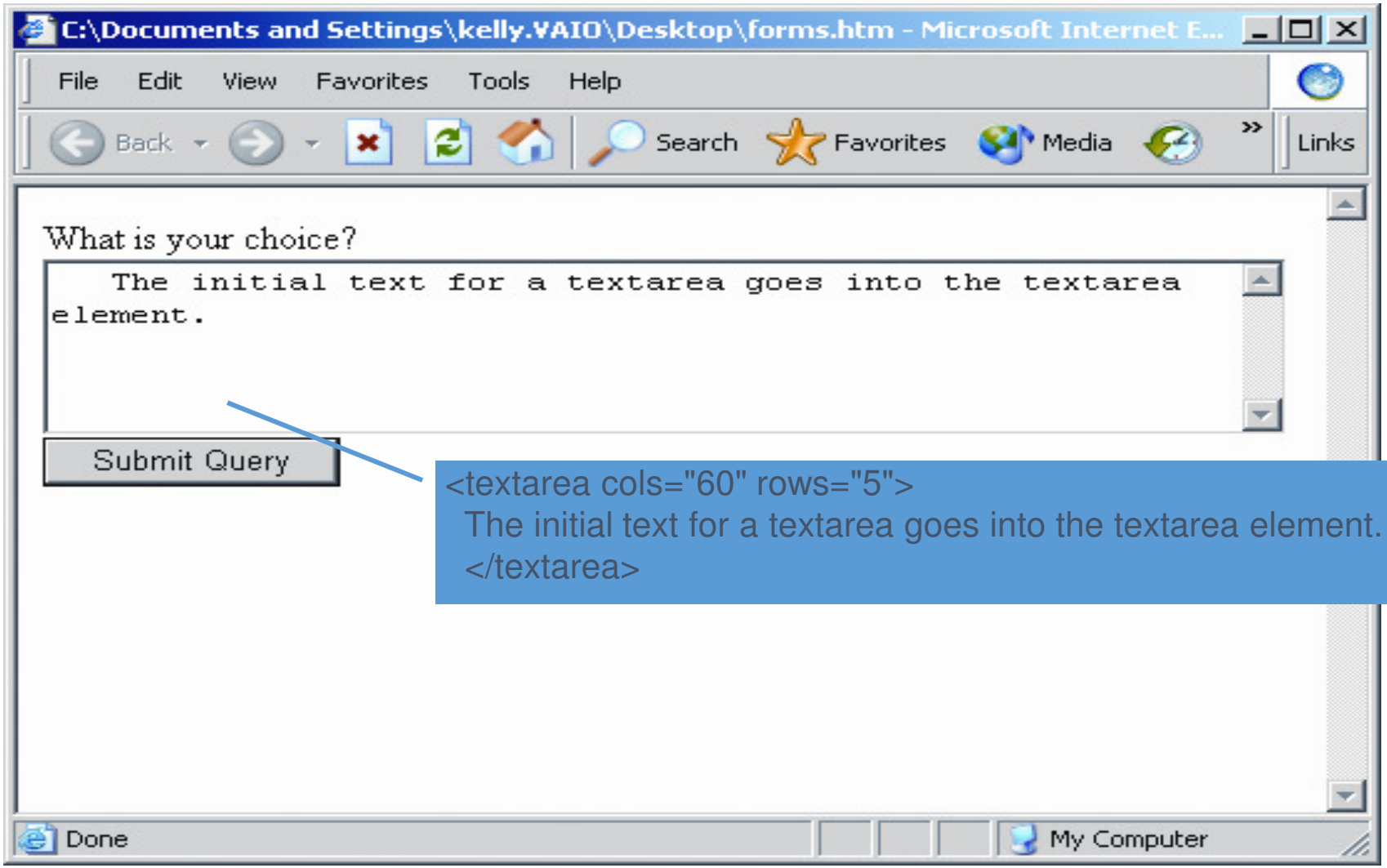
Single-Selection Dropdown List



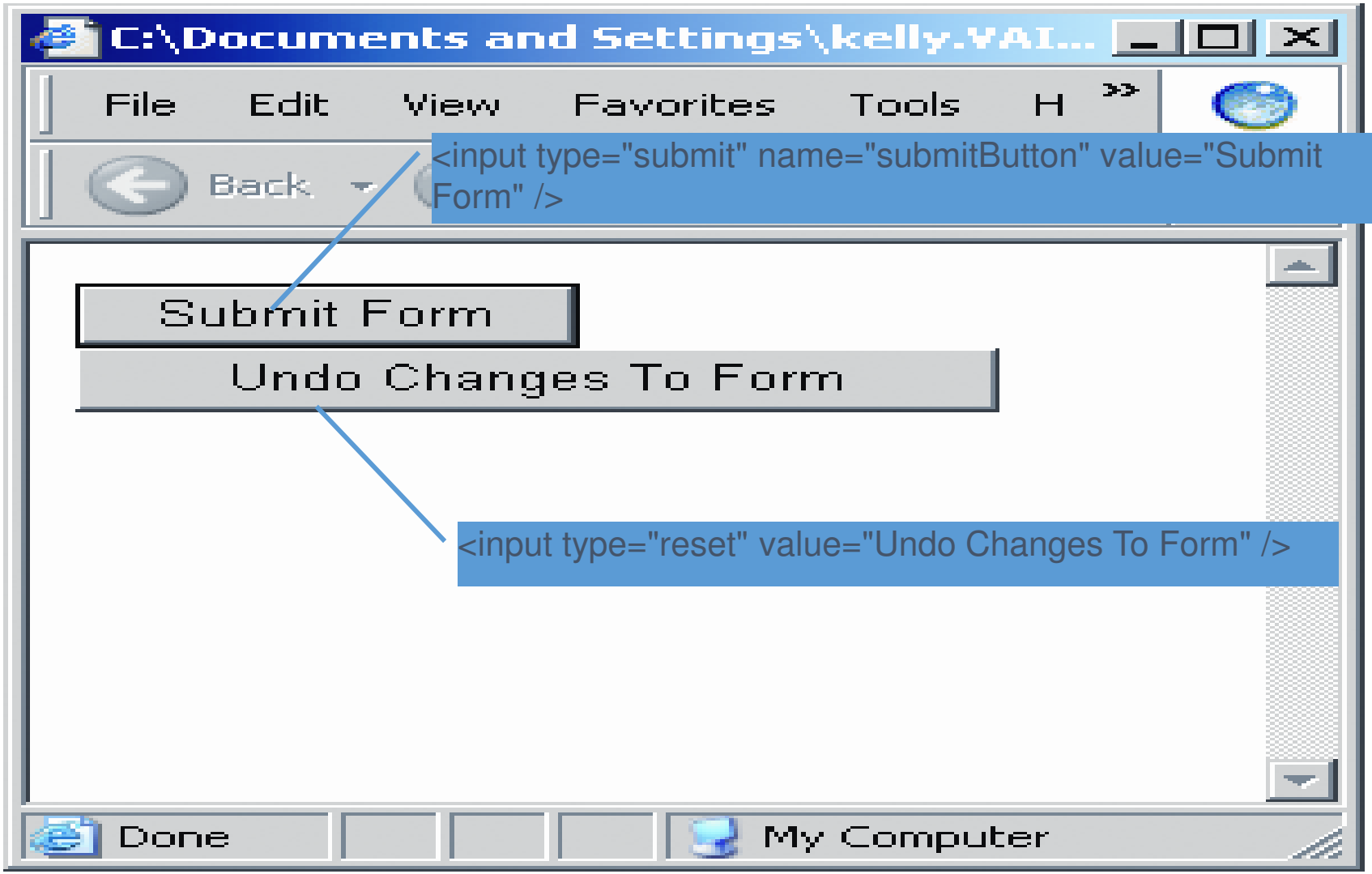
Multiple-Selection List



Effect of <textarea>



Button Tags


C:\Documents and Settings\kelly.VAI...
File
Edit
View
Favorites
Tools
H
Back
<input type="submit" name="submitButton" value="Submit Form" />
Submit Form
Undo Changes To Form
<input type="reset" value="Undo Changes To Form" />
Done
My Computer

3.1 New HTML5 Form `input` Types

```
<!DOCTYPE html>

<!-- Fig. 3.1: newforminputtypes.html -->
<!-- New HTML5 form input types and attributes. -->
<html>
   <head>
      <meta charset="utf-8">
      <title>New HTML5 Input Types</title>
   </head>

   <body>
      <h1>New HTML5 Input Types Demo</h1>
      <p>This form demonstrates the new HTML5 input types
         and the placeholder, required and autofocus attributes.
      </p>

      <form method = "post" action = "http://www.deitel.com">
         <p>
            <label>Color:
               <input type = "color" autofocus />
                  (Hexadecimal code such as #ADD8E6)
            </label>
         </p>
```

Fig. 3.1 | New HTML5 form input types and attributes. (Part 1 of 5.)

```
24          <p>
25             <label>Date:
26                <input type = "date" />
27                   (yyyy-mm-dd)
28             </label>
29          </p>
30          <p>
31             <label>Datetime:
32                <input type = "datetime" />
33                   (yyyy-mm-ddThh:mm+ff:gg, such as 2012-01-27T03:15)
34             </label>
35          </p>
36          <p>
37             <label>Datetime-local:
38                <input type = "datetime-local" />
39                   (yyyy-mm-ddThh:mm, such as 2012-01-27T03:15)
40             </label>
41          </p>
42          <p>
43             <label>Email:
44                <input type = "email" placeholder = "name@domain.com"
45                   required /> (name@domain.com)
46             </label>
47          </p>
```

Fig. 3.1 | New HTML5 form input types and attributes. (Part 2 of 5.)

```
48          <p>
49             <label>Month:
50                <input type = "month" /> (yyyy-mm)
51             </label>
52          </p>
53          <p>
54             <label>Number:
55                <input type = "number"
56                   min = "0"
57                   max = "7"
58                   step = "1"
59                   value = "4" />
60             </label> (Enter a number between 0 and 7)
61          </p>
62          <p>
63             <label>Range:
64                0 <input type = "range"
65                   min = "0"
66                   max = "20"
67                   value = "10" /> 20
68             </label>
69          </p>
```

Fig. 3.1 | New HTML5 form input types and attributes. (Part 3 of 5.)

```
70          <p>
71             <label>Search:
72                <input type = "search" placeholder = "search query" />
73             </label> (Enter your search query here.)
74          </p>
75          <p>
76             <label>Tel:
77                <input type = "tel" placeholder = "(###) ###-####"
78                   pattern = "\(\d{3}\) +\d{3}-\d{4}" required />
79                   (###) ###-####
80             </label>
81          </p>
82          <p>
83             <label>Time:
84                <input type = "time" /> (hh:mm:ss.ff)
85             </label>
86          </p>
87          <p>
88             <label>URL:
89                <input type = "url"
90                   placeholder = "http://www.domainname.com" />
91                   (http://www.domainname.com)
92             </label>
93          </p>
```

Fig. 3.1 | New HTML5 form input types and attributes. (Part 4 of 5.)

```
94            <p>
95               <label>Week:
96                  <input type = "week" />
97                     (yyyy-Wnn, such as 2012-W01)
98               </label>
99            </p>
100           <p>
101              <input type = "submit" value = "Submit" />
102              <input type = "reset" value = "Clear" />
103           </p>
104        </form>
105     </body>
106  </html>
```

Fig. 3.1 | New HTML5 form input types and attributes. (Part 5 of 5.)

3.1.1 `input` Type `color`

- The color input type enables the user to enter a color.
- At the time of this writing, most browsers render the `color input` type as a text field in which the user can enter a hexadecamal code or a color name.
- In the future, when you click a `color input`, browsers will likely display a *color picker* similar to the Microsoft Windows color dialog shown in Fig. 3.2.

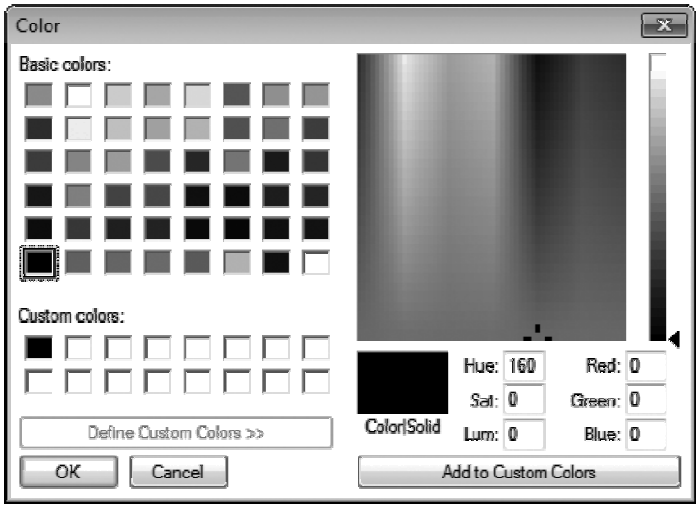


Fig. 3.2 | A dialog for choosing colors.

3.1.1 `input` Type `color`

`autofocus` Attribute

- The autofocus attribute—an optional attribute that can be used in only one `input` element on a form—automatically gives the focus to the `input` element, allowing the user to begin typing in that element immediately.

3.1.1 `input` Type `color` (cont.)

- Figure 3.3 shows `autofocus` on the `color` element—the first `input` element in our form—as rendered in Chrome. You do not need to include `autofocus` in your forms.

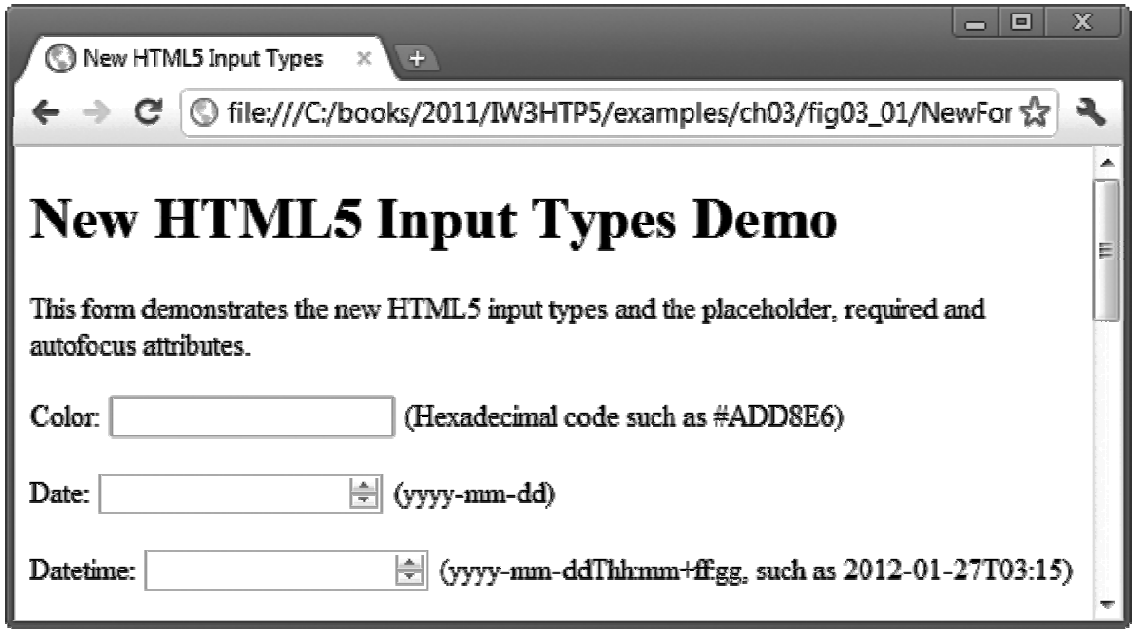


Fig. 3.3 | Autofocus in the `color input` element using Chrome.

3.1.1 `input` Type `color` (cont.)

Validation

- The new HTML 5 `input` types are *self validating* on the client side, eliminating the need to add complicated JavaScript code to your web pages to validate user input, reducing the amount of invalid data submitted and consequently reducing Internet traffic between the server and the client to correct invalid input.
- *The server should still validate all user input.*
- When a user enters data into a form then submits the form the browser immediately checks the self-validating elements to ensure that the data is correct (Fig. 3.4).

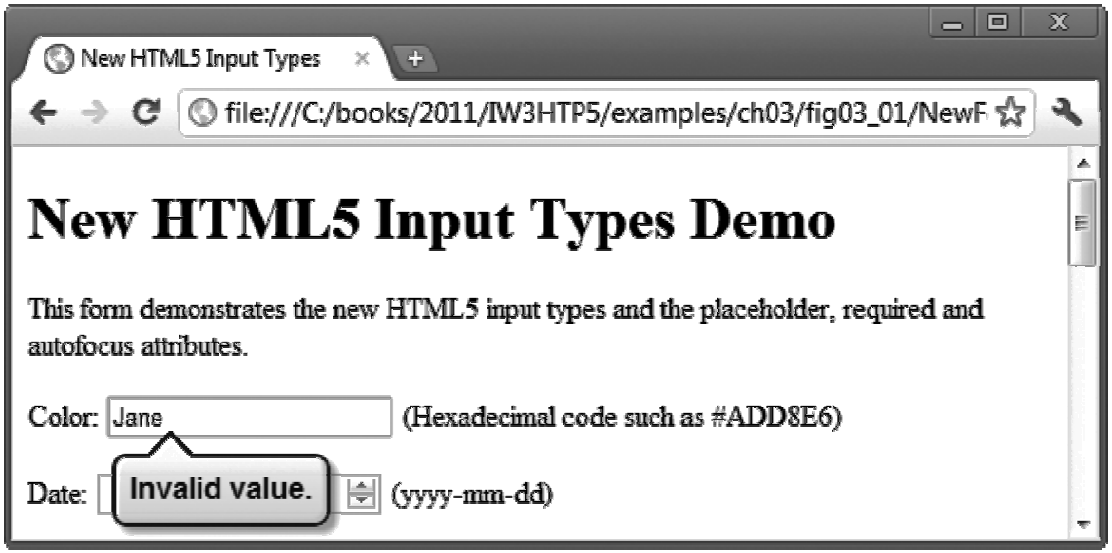


Fig. 3.4 | Validating a `color input` in Chrome.

3.1.1 `input` Type `color` (cont.)

- Figure 3.5 lists each of the new HTML5 `input` types and provides examples of the proper formats required for each type of data to be valid.

input type	Format
color	Hexadecimal code
date	yyyy-mm-dd
datetime	yyyy-mm-dd
datetime-local	yyyy-mm-ddThh:mm
month	yyyy-mm
number	Any numerical value
email	name@domain.com
url	http://www.domain-name.com
time	hh:mm
week	yyyy-Wnn

Fig. 3.5 | Self-validating input types.

3.1.2 `input` Type `date`

- The date input type enables the user to enter a date in the form `yyyy-mm-dd`.
- Firefox and Internet Explorer display a text field in which a user can enter a date such as `2012-01-27`.
- Chrome and Safari display a spinner control—a text field with an up-down arrow () on the right side—allowing the user to select a date by clicking the up or down arrow.
- The start date is the *current date*.
- Opera displays a calendar from which you can choose a date.
- In the future, when the user clicks a `date input`, browsers are likely to display a date control similar to the Microsoft Windows one shown in Fig. 3.6.

Fig. 3.6 | A date chooser control.

3.1.3 `input` Type `datetime`

- The datetime input type enables the user to enter a date (year, month, day), time (hour, minute, second, fraction of a second) and the time zone set to UTC (Coordinated Universal Time or Universal Time, Coordinated).
- Currently, most of the browsers render `datetime` as a text field; Chrome renders an up-down control and Opera renders a date and time control.

3.1.4 `input` Type `datetime-local`

- The datetime-local input type enables the user to enter the date and time in a *single* control.
- The data is entered as year, month, day, hour, minute, second and fraction of a second.
- Internet Explorer, Firefox and Safari all display a text field.
- Opera displays a date and time control.

3.1.5 `input` Type `email`

- The email input type enables the user to enter an e-mail address or a list of e-mail addresses separated by commas (if the `multiple` attribute is specified).
- Currently, all of the browsers display a text field.
- If the user enters an *invalid* e-mail address (i.e., the text entered is *not* in the proper format) and clicks the Submit button, a callout asking the user to enter an e-mail address is rendered pointing to the `input` element (Fig. 3.7).
- HTML5 does not check whether an e-mail address entered by the user actually exists—rather it just validates that the e-mail address is in the *proper format*.

Fig. 3.7 | Validating an e-mail address in Chrome.

3.1.5 `input` Type `email` (cont.)

`placeholder` Attribute

- The placeholder attribute allows you to place temporary text in a text field.
- Generally, `placeholder` text is *light gray* and provides an example of the text and/or text format the user should enter (Fig. 3.8).
- When the *focus* is placed in the text field (i.e., the cursor is in the text field), the `placeholder` text disappears—it's not "submitted" when the user clicks the Submit button (unless the user types the same text).

a) Text field with gray `placeholder` text

b) `placeholder` text disappears when the text field gets the focus

Fig. 3.8 | `placeholder` text disappears when the `input` element gets the focus.

3.1.5 `input` Type `email` (cont.)

- HTML5 supports `placeholder` text for only six `input` types—`text`, `search`, `url`, `tel`, `email` and `password`.

  `required` Attribute

- The required attribute forces the user to enter a value before submitting the form.
- You can add `required` to any of the `input` types.
- In this example, the user *must* enter an e-mail address and a telephone number to submit the form (Fig. 3.9).

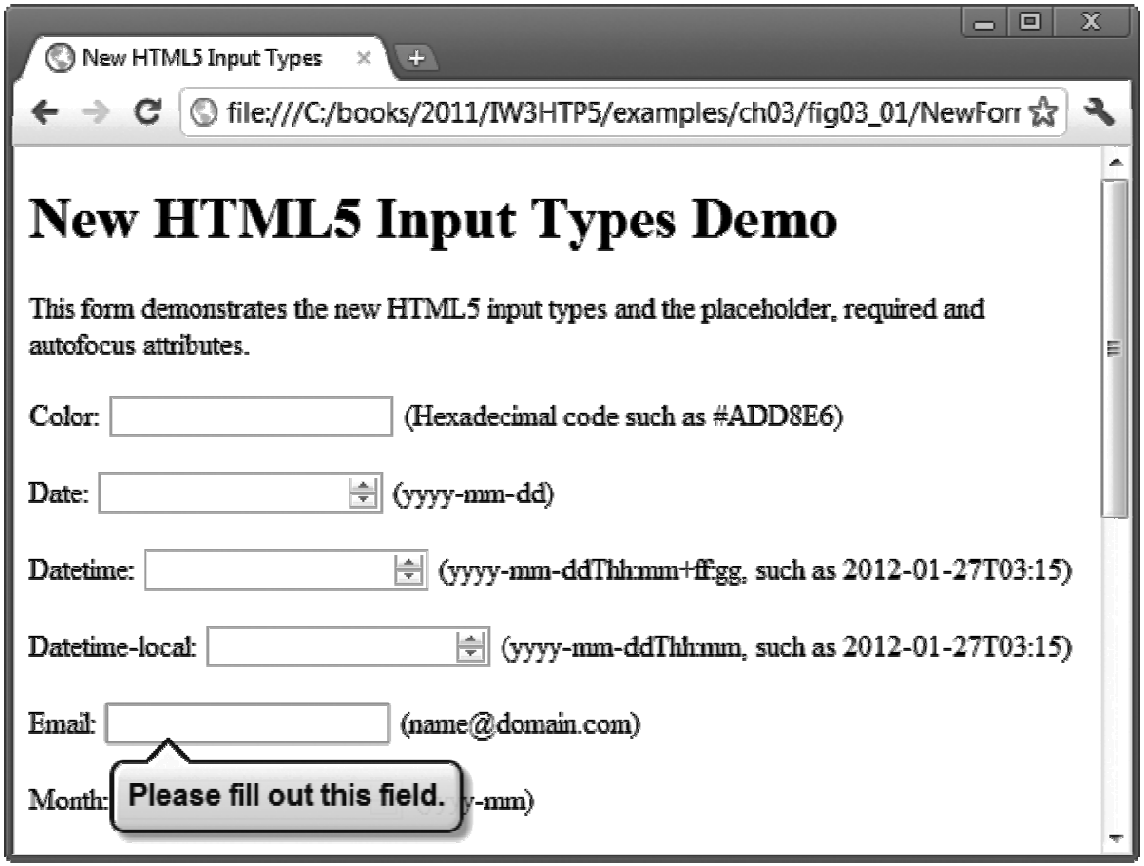


Fig. 3.9 | Demonstrating the `required` attribute in Chrome.

3.1.6 `input` Type `month`

- The month input type enables the user to enter a year and month in the format `yyyy-mm`, such as `2012-01`.
- If the user enters the data in an improper format (e.g., `January 2012`) and submits the form, a callout stating that an invalid value was entered appears.

3.1.7 `input` Type `number`

- The number input type enables the user to enter a numerical value—mobile browsers typically display a numeric keypad for this `input` type.
- Internet Explorer, Firefox and Safari display a text field in which the user can enter a number. Chrome and Opera render a spinner control for adjusting the number.
- The `min` attribute sets the minimum valid number.
- The `max` attribute sets the maximum valid number.
- The `step` attribute determines the increment in which the numbers increase.
- The `value` attribute sets the initial value displayed in the form (Fig. 3.10).
- The spinner control includes only the valid numbers.
- If the user attempts to enter an invalid value by typing in the text field, a callout pointing to the `number input` element will instruct the user to enter a valid value.

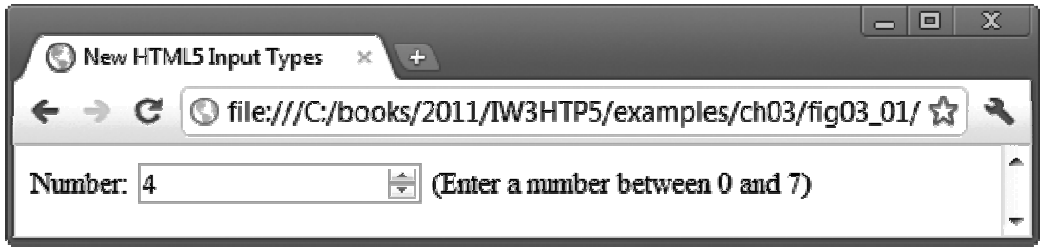


Fig. 3.10 | `input` type `number` with a `value` attribute of 4 as rendered in Chrome.

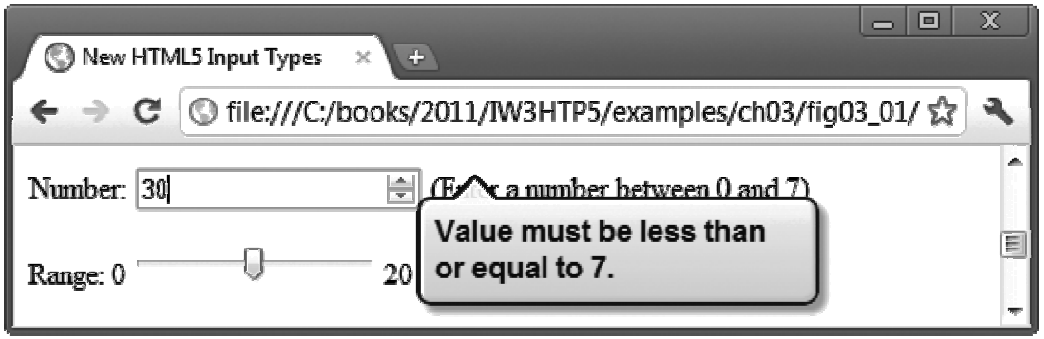


Fig. 3.11 | Chrome checking for a valid number.

3.1.8 `input` Type `range`

- The range input type appears as a *slider* control in Chrome, Safari and Opera (Fig. 3.12).
- You can set the minimum and maximum and specify a value.
- The `range input` type is *inherently self-validating* when it is rendered by the browser as a slider control, because *the user is unable to move the slider outside the bounds of the minimum or maximum value*.

Fig. 3.12 | `range` slider with a `value` attribute of 10 as rendered in Chrome.

3.1.9 `input` Type `search`

- The search input type provides a search field for entering a query.
- This `input` element is functionally equivalent to an `input` of type `text`.
- When the user begins to type in the search field, Chrome and Safari display an X that can be clicked to clear the field (Fig. 3.13).
-

Fig. 3.13 | Entering a search query in Chrome.

3.1.10 `input` Type `tel`

- The tel input type enables the user to enter a telephone number—mobile browsers typically display a keypad specific to entering phone numbers for this `input` type.
- At the time of this writing, the `tel input` type is rendered as a text field in all of the browsers.
- HTML5 does *not* self validate the `tel input` type.
- To ensure that the user enters a phone number in a proper format, we've added a `pattern` attribute that uses a *regular expression* to determine whether the number is in the format:
    - `(555) 555-5555`
- When the user enters a phone number in the wrong format, a callout appears requesting the proper format, pointing to the `tel input` element (Fig. 3.14).
-

Fig. 3.14 | Validating a phone number using the `pattern` attribute in the `tel` input type.

3.1.11 `input` Type `time`

- The time input type enables the user to enter an hour, minute, seconds and fraction of second (Fig. 3.15).
- The HTML5 specification indicates that a time must have two digits representing the hour, followed by a colon (:) and two digits representing the minute.
- Optionally, you can also include a colon followed by two digits representing the seconds and a period followed by one or more digits representing a fraction of a second (shown as `ff` in our sample text to the right of the time input element in Fig. 3.15.
-

Fig. 3.15 | `time input` as rendered in Chrome.

3.1.12 `input` Type `url`

- The url input type enables the user to enter a URL.
- The element is rendered as a text field, and the proper format is `http://www.deitel.com`.
- If the user enters an improperly formatted URL (e.g., `www.deitel.com` or `www.deitelcom`), the URL will *not* validate (Fig. 3.16).
- HTML5 does not check whether the URL entered is valid; rather it validates that the URL entered is in the proper format.

Fig. 3.16 | Validating a URL in Chrome.

3.1.13 `input` Type `week`

- The week input type enables the user to select a year and week number in the format `yyyy-Wnn`, where nn is 01–53—for example, `2012-W01` represents the first week of 2012. Internet Explorer, Firefox and Safari render a text field.
- Chrome renders an up-down control.
- Opera renders *week control* with a down arrow that, when clicked, brings up a calendar for the current month with the corresponding week numbers listed down the left side.

3.2 `input` and `datalist` Elements and `autocomplete` Attribute

```
<!DOCTYPE html>

<!-- Fig. 3.17: autocomplete.html -->
<!-- New HTML5 form autocomplete attribute and datalist element. -->
<html>
   <head>
      <meta charset="utf-8">
      <title>New HTML5 autocomplete Attribute and datalist Element</title>
   </head>

   <body>
      <h1>Autocomplete and Datalist Demo</h1>
      <p>This form demonstrates the new HTML5 autocomplete attribute
         and the datalist element.
      </p>

      <!-- turn autocomplete on -->
      <form method = "post" autocomplete = "on">
         <p><label>First Name:
            <input type = "text" id = "firstName"
               placeholder = "First name" /> (First name)
            </label></p>
```

Fig. 3.17 | New HTML5 form `autocomplete` attribute and `datalist` element. (Part 1 of 6.)

```
23          <p><label>Last Name:
24             <input type = "text" id = "lastName"
25                placeholder = "Last name" /> (Last name)
26             </label></p>
27          <p><label>Email:
28             <input type = "email" id = "email"
29                placeholder = "name@domain.com" /> (name@domain.com)
30             </label></p>
31          <p><label for = "txtList">Birth Month:
32             <input type = "text" id = "txtList"
33                placeholder = "Select a month" list = "months" />
34             <datalist id = "months">
35                <option value = "January">
36                <option value = "February">
37                <option value = "March">
38                <option value = "April">
39                <option value = "May">
40                <option value = "June">
41                <option value = "July">
42                <option value = "August">
43                <option value = "September">
44                <option value = "October">
45                <option value = "November">
```

Fig. 3.17 | New HTML5 form `autocomplete` attribute and `datalist` element. (Part 2 of 6.)

```
46                <option value = "December">
47             </datalist>
48          </label></p>
49          <p><input type = "submit" value = "Submit" />
50             <input type = "reset" value = "Clear" /></p>
51       </form>
52    </body>
53 </html>
```

Fig. 3.17 | New HTML5 form `autocomplete` attribute and `datalist` element. (Part 3 of 6.)

a) Form rendered in Firefox before the user interacts with it

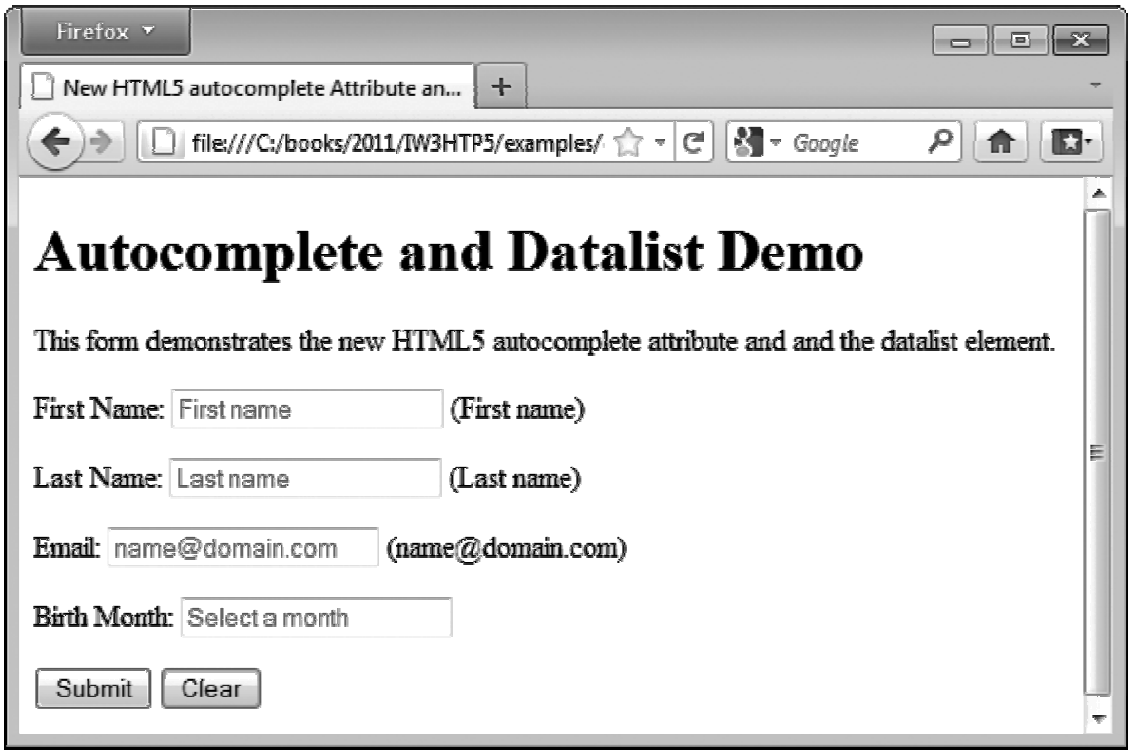


Fig. 3.17 | New HTML5 form `autocomplete` attribute and `datalist` element. (Part 4 of 6.)

b) `autocomplete` automatically fills in the data when the user returns to a form submitted previously and begins typing in the **First Name** `input` element; clicking `Jane` inserts that value in the `input`

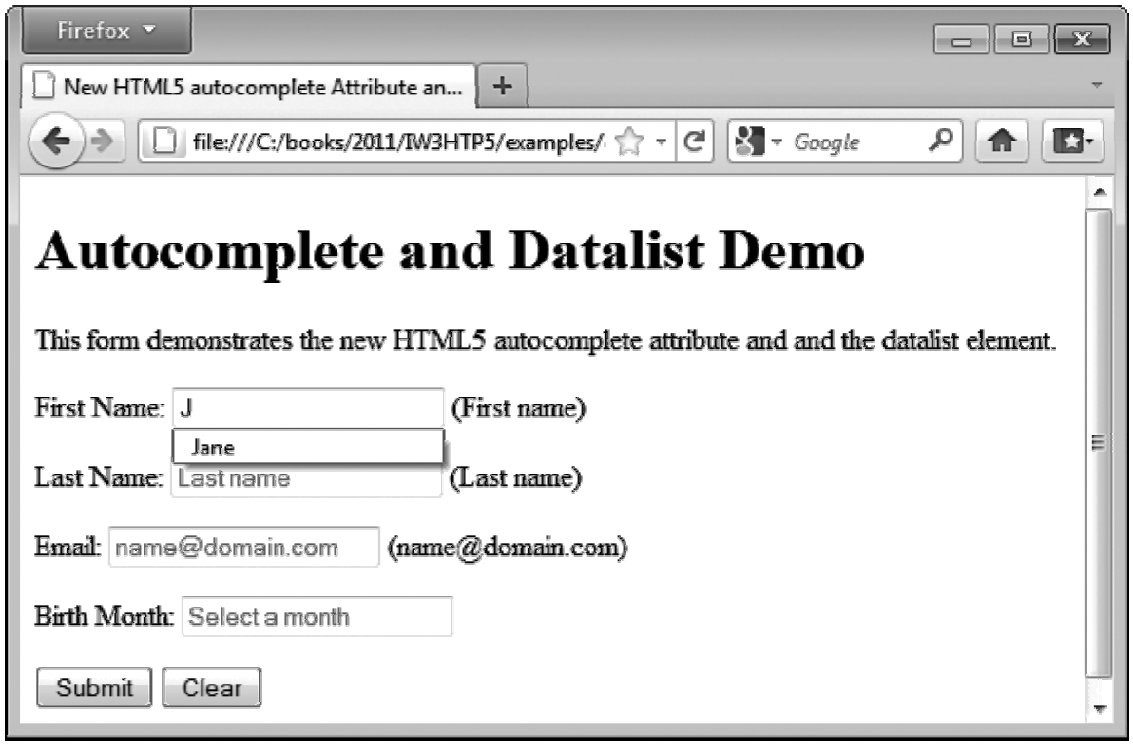


Fig. 3.17 | New HTML5 form `autocomplete` attribute and `datalist` element. (Part 5 of 6.)

c) `autocomplete` with a `datalist` showing the previously entered value (`June`) followed by all items that match what the user has typed so far; clicking an item in the `autocomplete` list inserts that value in the `input`

`datalist` values filtered by what's been typed so far

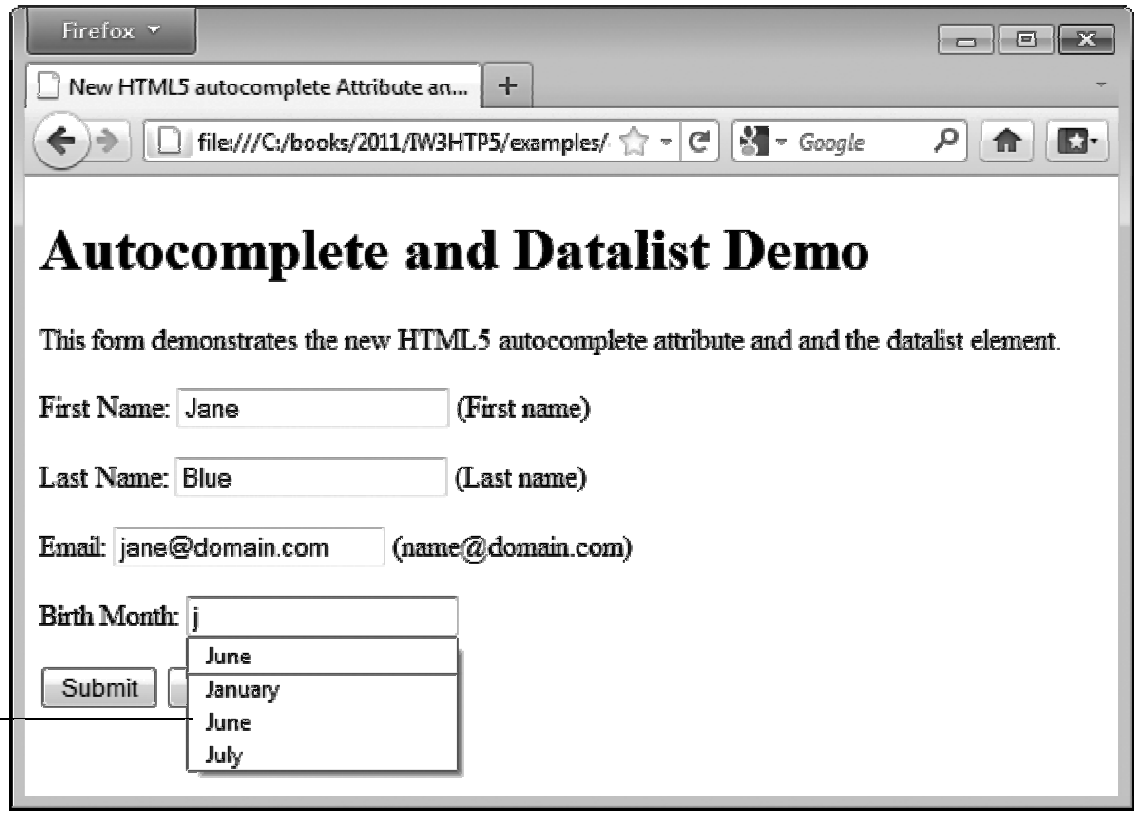


Fig. 3.17 | New HTML5 form `autocomplete` attribute and `datalist` element. (Part 6 of 6.)

3.2.1 `input` Element `autocomplete` Attribute

- The autocomplete attribute can be used on `input` types to automatically fill in the user's information based on previous input—such as name, address or e-mail.
- You can enable `autocomplete` for an entire form or just for specific elements.
- For example, an online order form might set `automcomplete = "on"` for the name and address `inputs` and set `autocomplete = "off"` for the credit card and password `inputs` for security purposes.

3.2.2 `datalist` Element

- The datalist element provides input options for a `text input` element.
- At the time of this writing, `datalist` support varies by browser.
- In this example, we use a `datalist` element to obtain the user's birth month.
- Using Opera, when the user clicks in the text field, a drop-down list of the months of the year appears. If the user types `"M"` in the text field, the list on months is narrowed to March and May.
- When using Firefox, the drop-down list of months appears only after the user begins typing in the text field. If the user types `"M"`, all months containing the letter `"M"` or `"m"` appear in the drop-down list—March, May, September, November and December.

3.3 Page-Structure Elements

- HTML5 introduces several new page-structure elements that meaningfully identify areas of the page as headers, footers, articles, navigation areas, asides, figures and more.
-

```
<!DOCTYPE html>

<!-- Fig. 3.18: sectionelements.html -->
<!-- New HTML5 section elements. -->
<html>
   <head>
      <meta charset="utf-8">
      <title>New HTML5 Section Elements</title>
   </head>

   <body>
      <header> <!-- header element creates a header for the page -->
         <img src = "deitellogo.png" alt = "Deitel logo" />
         <h1>Welcome to the Deitel Buzz Online<h1>

         <!-- time element inserts a date and/or time -->
         <time>2012-01-17</time>

      </header>

      <section id = "1"> <!-- Begin section 1 -->
         <nav> <!-- nav element groups navigation links  -->
            <h2> Recent Publications</h2>
```

Fig. 3.18 | New HTML5 section elements. (Part 1 of 13.)

```
24             <ul>
25                <li><a href = "http://www.deitel.com/books/iw3htp5">
26                   Internet & World Wide Web How to Program, 5/e</a></li>
27                <li><a href = "http://www.deitel.com/books/androidfp/">
28                   Android for Programmers: An App-Driven Approach</a>
29                </li>
30                <li><a href = "http://www.deitel.com/books/iphonefp">
31                   iPhone for Programmers: An App-Driven Approach</a></li>
32                <li><a href = "http://www.deitel.com/books/jhtp9/">
33                   Java How to Program, 9/e</a></li>
34                <li><a href = "http://www.deitel.com/books/cpphtp8/">
35                   C++ How to Program, 8/e</a></li>
36                <li>
37                   <a href = "http://www.deitel.com/books/vcsharp2010htp">
38                      Visual C# 2010 How to Program, 4/e</a></li>
39                <li><a href = "http://www.deitel.com/books/vb2010htp">
40                   Visual Basic 2010 How to Program</a></li>
41             </ul>
42          </nav>
43       </section>
44
45       <section id = "2"> <!-- Begin section 2 -->
46          <h2>How to Program Series Books</h2>
47          <h3><em>Java How to Program, 9/e</em></h3>
```

Fig. 3.18 | New HTML5 section elements. (Part 2 of 13.)

```
48
49          <figure> <!-- figure element describes the image -->
50             <img src = "jhtp.jpg" alt = "Java How to Program, 9/e" />
51
52             <!-- figurecaption element inserts a figure caption -->
53             <figcaption><em>Java How to Program, 9/e</em>
54                cover.</figcaption>
55          </figure>
56
57          <!--article element represents content from another source -->
58          <article>
59             <header>
60                <h5>From
61                   <em>
62                      <a href = "http://www.deitel.com/books/jhtp9/">
63                      Java How to program, 9/e: </a>
64                   </em>
65                </h5>
66             </header>
67
```

Fig. 3.18 | New HTML5 section elements. (Part 3 of 13.)

```
68          <p>Features include:
69             <ul>
70                <li>Rich coverage of fundamentals, including
71                   <!-- mark element highlights text -->
72                   <mark>two chapters on control statements.</mark></li>
73                <li>Focus on <mark>real-world examples.</mark></li>
74                <li><mark>Making a Difference exercises set.</mark></li>
75                <li>Early introduction to classes, objects,
76                   methods and strings.</li>
77                <li>Integrated exception handling.</li>
78                <li>Files, streams and object serialization.</li>
79                <li>Optional modular sections on language and
80                   library features of the new Java SE 7.</li>
81                <li>Other topics include: Recursion, searching,
82                   sorting, generic collections, generics, data
83                   structures, applets, multimedia,
84                   multithreading, databases/JDBC&trade;, web-app
85                   development, web services and an optional
86                   ATM Object-Oriented Design case study.</li>
87             </ul>
88
```

Fig. 3.18 | New HTML5 section elements. (Part 4 of 13.)

```
<!-- summary element represents a summary for the -->
<!-- content of the details element -->
<details>
   <summary>Recent Edition Testimonials</summary>
   <ul>
      <li>"Updated to reflect the state of the
         art in Java technologies; its deep and
         crystal clear explanations make it
         indispensable. The social-consciousness
         [Making a Difference] exercises are
         something really new and refreshing."
         <strong>&mdash;Jos&eacute; Antonio
         Gonz&aacute;lez Seco, Parliament of
         Andalusia</strong></li>
      <li>"Gives new programmers the benefit of the
         wisdom derived from many years of software
         development experience."<strong>
         &mdash;Edward F. Gehringer, North Carolina
         State University</strong></li>
      <li>"Introduces good design practices and
         methodologies right from the beginning.
         An excellent starting point for developing
         high-quality robust Java applications."
         <strong>&mdash;Simon Ritter,
         Oracle Corporation</strong></li>
```

Fig. 3.18 | New HTML5 section elements. (Part 5 of 13.)

```
                  <li>"An easy-to-read conversational style.
                     Clear code examples propel readers to
                     become proficient in Java."
                     <strong>&mdash;Patty Kraft, San Diego State
                     University</strong></li>
                  <li>"A great textbook with a myriad of examples
                     from various application domains&mdash;
                     excellent for a typical CS1 or CS2 course."
                     <strong>&mdash;William E. Duncan, Louisiana
                     State University</strong></li>
               </ul>
            </details>
         </p>
      </article>

      <!-- aside element represents content in a sidebar that's -->
      <!-- related to the content around the element -->
      <aside>
         The aside element is not formatted by the browsers.
      </aside>

```

Fig. 3.18 | New HTML5 section elements. (Part 6 of 13.)

```
135          <h2>Deitel Developer Series Books</h2>
136          <h3><em>Android for Programmers: An App-Driven Approach
137             </em></h3>
138             Click <a href = "http://www.deitel.com/books/androidfp/">
139             here</a> for more information or to order this book.
140
141          <h2>LiveLessons Videos</h2>
142          <h3><em>C# 2010 Fundamentals LiveLessons</em></h3>
143          Click <a href = "http://www.deitel.com/Books/LiveLessons/">
144          here</a> for more information about our LiveLessons videos.
145       </section>
146
147       <section id = "3"> <!-- Begin section 3 -->
148          <h2>Results from our Facebook Survey</h2>
149          <p>If you were a nonprogrammer about to learn Java for the first
150             time, would you prefer a course that taught Java in the
151             context of Android app development? Here are the results from
152             our survey:</p>
153
154             <!-- meter element represents a scale within a range -->
155             0 <meter min = "0"
156             max = "54"
157             value = "14"></meter> 54
```

Fig. 3.18 | New HTML5 section elements. (Part 7 of 13.)

```
158          <p>Of the 54 responders, 14 (green) would prefer to
159          learn Java in the context of Android app development.</p>
160       </section>
161
162       <!-- footer element represents a footer to a section or page, -->
163       <!-- usually containing information such as author name, -->
164       <!-- copyright, etc. -->
165       <footer>
166          <!-- wbr element indicates the appropriate place to break a -->
167          <!-- word when the text wraps -->
168          <h6>&copy; 1992-2012 by Deitel &amp; Associ<wbr>ates, Inc.
169             All Rights Reserved.<h6>
170          <!-- address element represents contact information for a -->
171          <!-- document or the nearest body element or article -->
172          <address>
173             Contact us at <a href = "mailto:deitel@deitel.com">
174             deitel@deitel.com</a>
175          </address>
176       </footer>
177    </body>
178 </html>
```

Fig. 3.18 | New HTML5 section elements. (Part 8 of 13.)

a) Chrome browser showing the **header** element and a **nav** element that contains an unordered list of links

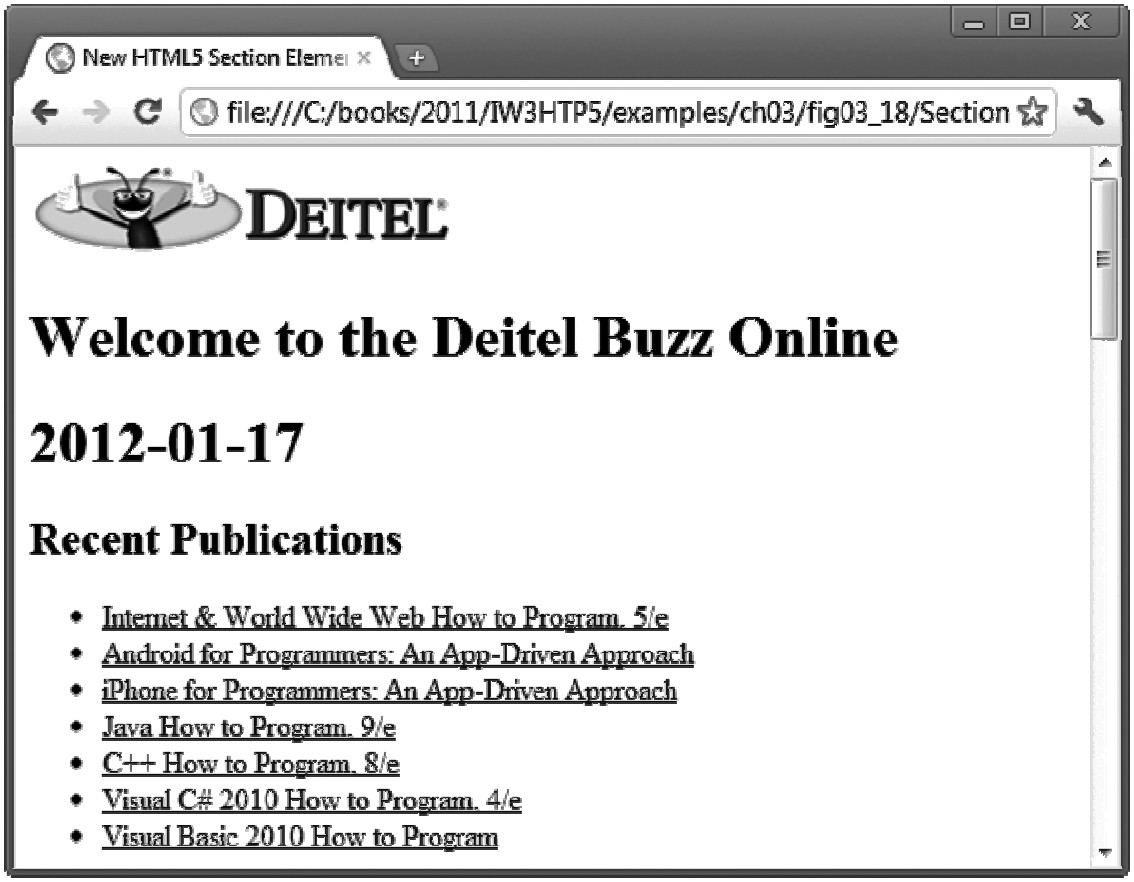


Fig. 3.18 | New HTML5 section elements. (Part 9 of 13.)

b) Chrome browser showing the beginning of a `section` containing a `figure` and a `figurecaption`

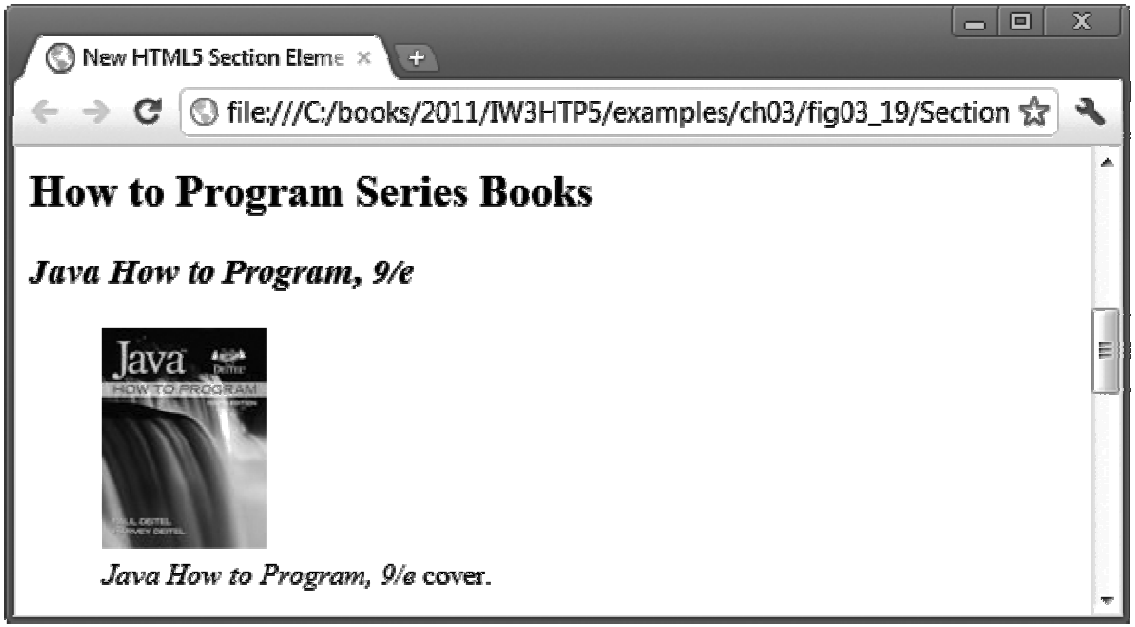


Fig. 3.18 | New HTML5 section elements. (Part 10 of 13.)

c) Chrome browser showing an `article` containing a `header`, some content and a collapsed `details` element, followed by an `aside` element

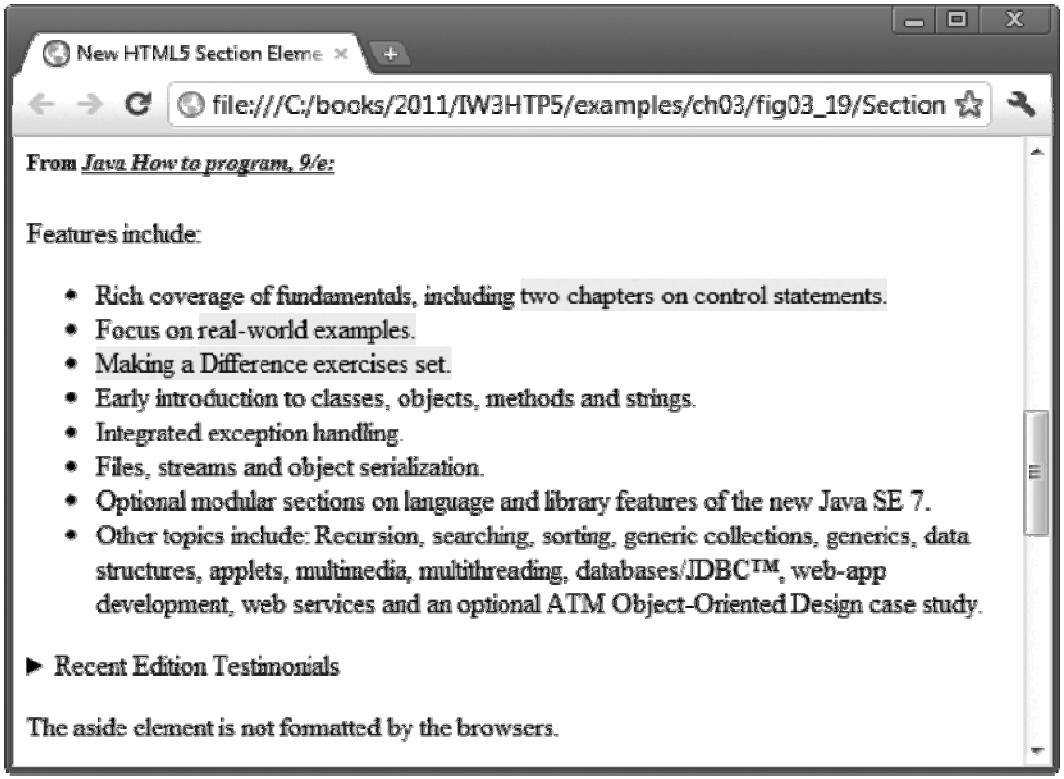


Fig. 3.18 | New HTML5 section elements. (Part 11 of 13.)

d) Chrome browser showing the end of the `section` that started in part (b)

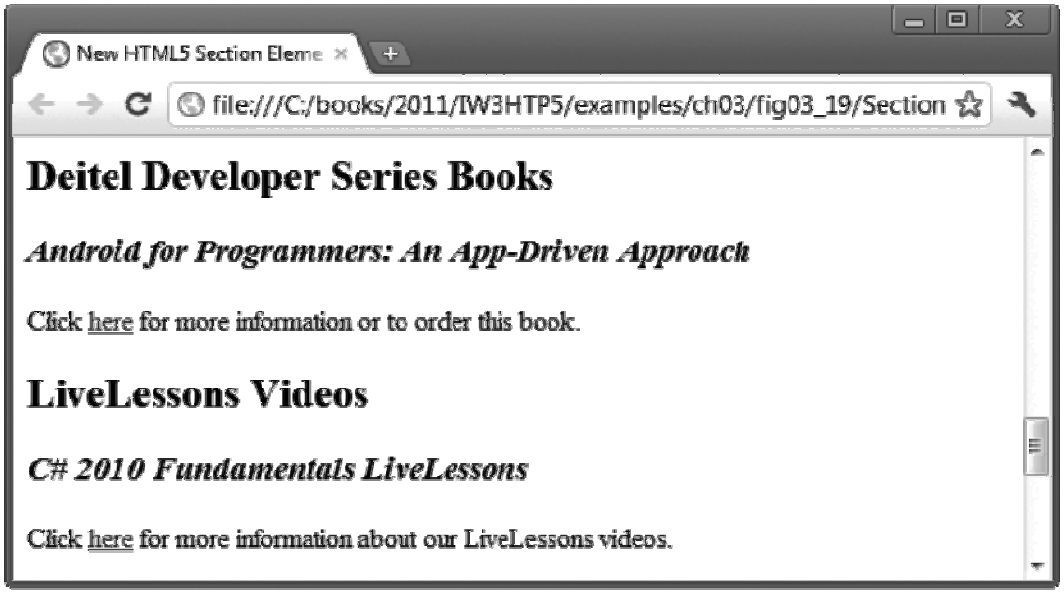


Fig. 3.18 | New HTML5 section elements. (Part 12 of 13.)

e) Chrome browser showing the last `section` containing a `meter` element, followed by a `footer` element

Fig. 3.18 | New HTML5 section elements. (Part 13 of 13.)

3.3.1 header Element

- The header element creates a header for this page that contains both text and graphics.
- The header element can be used multiple times on a page and can include HTML headings (<h1> through <h6>), navigation, images and logos and more.

time Element

- The time element, which does not need to be enclosed in a header, enables you to identify a date (as we do here), a time or both.

3.3.2 nav Element

- The nav element groups navigation links.
- In this example, we used the heading Recent Publications and created a **`ul`** element with seven **`li`** elements that link to the corresponding web pages for each book.

3.3.3 `figure` Element and `figcaption` Element

- The figure element describes a figure (such as an image, chart or table) in the document so that it could be moved to the side of the page or to another page.
- The figcaption element provides a caption for the image in the `figure` element.

3.3.4 `article` Element

- The **`article`** element describes standalone content that could potentially be used or distributed elsewhere, such as a news article, forum post or blog entry.
- You can nest `article` elements. For example, you might have reader comments about a magazine nested as an `article` within the magazine `article`.

3.3.5 `summary` Element and `details` Element

- The summary element displays a right-pointing arrow next to a summary or caption when the document is rendered in a browser (Fig. 3.19).
- When clicked, the arrow points downward and reveals the content in the details element.

Fig. 3.19 | Demonstrating the `summary` and `detail` elements. (Part 1 of 2.)

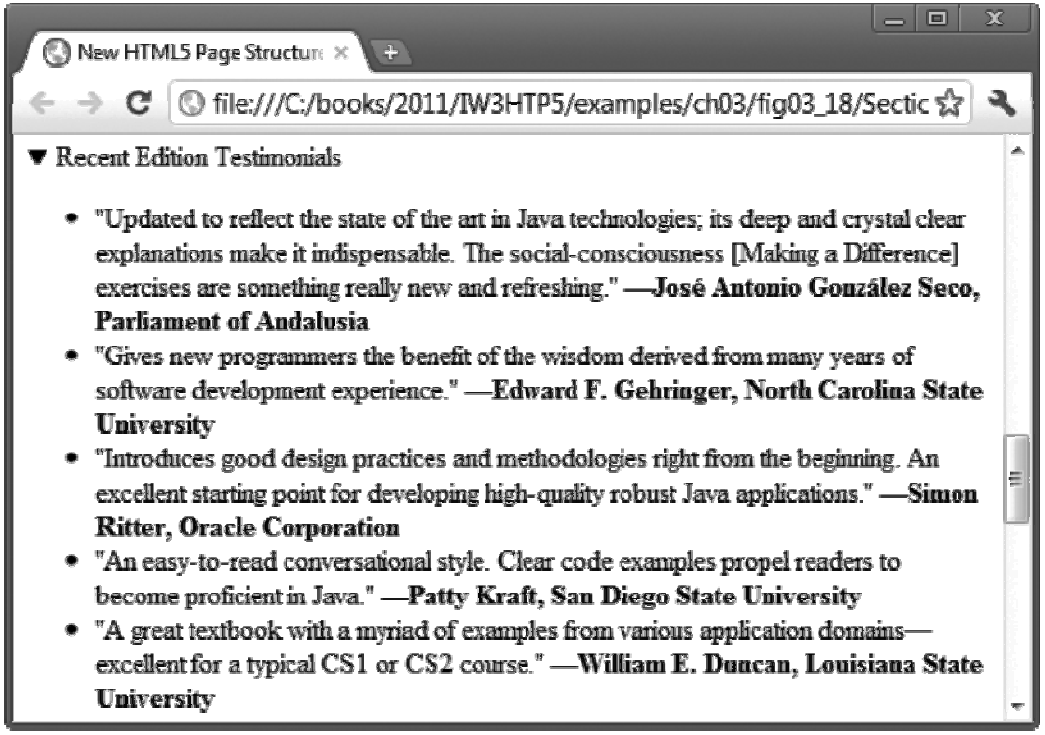


Fig. 3.19 | Demonstrating the `summary` and `detail` elements. (Part 2 of 2.)

3.3.6 `section` Element

- The section element describes a section of a document, usually with a heading for each section—these elements can be nested.
- In this example, we broke the document into three `section`s—the first is Recent Publications.
- The `section` element may also be nested in an article.

3.3.7 `aside` Element

- The aside element describes content that's related to the surrounding content (such as an `article`) but is somewhat separate from the flow of the text.
- For example, an `aside` in a news story might include some background history.

3.3.8 `meter` Element

- The meter element renders a visual representation of a measure within a range (Fig. 3.20).
- In this example, we show the results of a recent web survey we did.
- The `min` attribute is `"0"` and a `max` attribute is `"54"` — indicating the total number of responses to our survey.
- The `value` attribute is `"14"`, representing the total number of people who responded “yes” to our survey question.

Fig. 3.20 | Chrome rendering the `meter` element.

3.3.9 `footer` Element

- The footer element describes a *footer*—content that usually appears at the bottom of the content or `section` element.
- In this example, we use the `footer` to describe the copyright notice and contact information.

3.3.10 Text-Level Semantics: `mark` Element and `wbr` Element

- The mark element highlights the text that's enclosed in the element.
- The wbr element indicates the appropriate place to break a word when the text wraps to multiple lines.
- You might use `wbr` to prevent a word from breaking in an awkward place.