7.1 Another Look at XHTML Hyperlinks

Hyperlinks make the Web a "web" of interconnected information. In this section you'll revisit the topic of hyperlinks and explore coding relative links, using the target attribute to open Web pages in a new browser window, and coding hyperlinks that are internal to a Web page.

More on Relative Linking

As indicated earlier in Chapter 2, a relative link is used to link to Web pages within your site. You've been coding relative links to display Web pages within the same folder. There are times when you need to link to files in other folders on your Web site. Let's consider the example of a Web site for a dog groomer that highlights services and products. The Web developer for this site created separate folders called services and products in order to organize the site. See the folder and file listing shown in Figure 7.1.

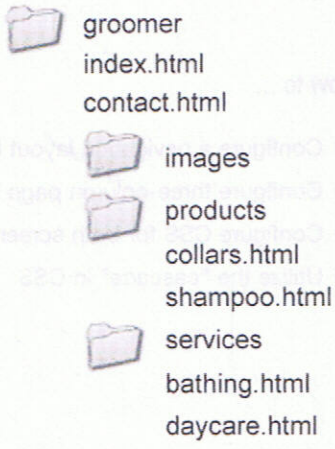


Figure 7.1
The dog groomer site contains the images, products, and services folders

Relative Link Examples

- To review, when linking to a file in the same folder or directory, the value of the `href` is the name of the file. For example, to link to the contact.html page from the home page (index.html), code the anchor element as follows:

```
<a href="contact.html">Contact</a>
```

- When linking to a folder located within the current directory, use both the folder name and the file name in the relative link. For example, to link to the collars.html page in the products folder from the home page (index.html), code the anchor element as follows:

```
<a href="products/collars.html">Collars</a>
```

- In Figure 7.1 the collars.html page is located in a subfolder of the groomer folder. The home page for the site, index.html is located in the groomer folder. When linking to a file that is up one directory level from the current page use `"../"` notation. To link to the home page for the site from the collars.html page, code the anchor element as follows:

```
<a href="../index.html">Home</a>
```

Figure 7.2
Notice how the anchor tags are used

```
<?xml version="1.0" encoding="UTF-8"?>
<!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>Favorite Sites</title>
</head>
<body bgcolor="#FFFFFF">
<a id="top" name="top"></a><h1 align="center">Favorite Sites</h1>
<dl>
  <dt>Running</dt>
     <dd><a href="http://www.running.com">running.com</a></dd>
  <dt>Cooking</dt>
     <dd><a href="http://www.cooking.com">cooking.com</a></dd>
  <dt>Internet</dt>
     <dd><a href="http://www.internet.com">internet.com</a></dd>
</dl>
<p><a href="#top">Back to Top</a></p>
</body>
</html>
```

This anchor tag creates the named fragment for the top of the page.

This anchor tag indicates the link to the top of the page.

2. **Reference Target.** At the point of the page where you want to place a link to the top, type another anchor element. Use the `href` attribute and place a `#` (sometimes called a hash mark) before the name of the bookmark. The XHTML for a hyperlink to the named anchor `"top"` is

```
<a href="#top">Top of Page</a>
```

The hash mark indicates that the browser should search for an anchor tag on the same page. If you forget to type the hash mark, the browser will not look on the same Web page; it will look for an external file. A bookmark or named anchor does not have to be at the top of a page; it can be just about anywhere.

If you are coding only for an XHTML-compliant browser such as Internet Explorer 5 (or later), Mozilla Firefox, or Netscape 6 (or later), you can use the `id` attribute with any container tag, such as a `<p>` or a `<h1>`, to create a named fragment or bookmark. The top of page example uses the anchor element to provide for backward compatibility with Netscape 4.

HANDS-ON PRACTICE 7.1

You will work with internal links in this Hands-On Practice. Locate the Chapter7/starter1.html file in the student files. Figure 7.3 shows a partial screenshot of this Web page.

Launch Notepad and open the starter1.html file. Save the file as favorites.html. Examine the source code and notice that the top portion of the page contains an unordered list with categories of interest (such as Hobbies, XHTML, CSS, and Professional Organizations) that correspond to the text displayed in the `<h2>` elements below. After each `<h2>` element is a definition list of topics and URLs related to that category. It might be helpful to Web page visitors if they can click a category item and

7.2 CSS Pseudo-Classes and Links

Have you ever visited a Web site and found that the text hyperlinks changed color when you moved the mouse pointer over them? Often, this is accomplished using a special technique in CSS called a pseudo-class. The four pseudo-classes that can be applied to the anchor tag are shown in Table 7.1. The **link pseudo-class** configures the appearance of the hyperlink before it is clicked. The **visited pseudo-class** configures the appearance

Table 7.1 Commonly used CSS pseudo-classes

Pseudo-class	When Applied
link	Default state for a link that has not been visited
visited	Default state for a visited link
hover	Triggered when the mouse moves over the link
active	Triggered when the link is actually clicked

of the hyperlink after it is clicked. The **hover pseudo-class** configures the hyperlink as the mouse is held or "hovered" over it. The **active pseudo-class** configures the appearance of the hyperlink while it is being clicked. Notice the order in which the pseudo-classes are listed in Table 7.1. Anchor element pseudo-classes must be coded in this order (although it's okay to omit one or more of those listed). If you code the pseudo-classes in a different order, the styles will not be reliably applied. Some students find the order easier to remember if they think of the mnemonic device "lovehate" – link, visited, hover, active.

The syntax of pseudo-classes uses a colon (:) to apply the **pseudo-class** to the anchor tag. The following code sample will configure text hyperlinks to be red initially. The sample also uses the hover pseudo-class, `a:hover`, to configure the links to change their appearance when the visitor places the mouse pointer over them so that the underline disappears and the color changes.

```
<style type="text/css">
a:link {   color: #ff0000;
}
a:hover {  text-decoration: none;
           color: #000066;
}
</style>
```

Figure 7.4 shows part of a Web page that uses this technique. Note the position of the mouse pointer over the Print this Page link—the link color has changed and has no underline.

While some Web design experts, such as Jakob Nielsen, recommend that Web developers not change the default look of text links, this technique is often used. Most modern browsers (since Internet Explorer 4+ and Netscape 6+) support CSS pseudo-classes. Netscape 4.x does not support the hover pseudo-class, but the technique degrades gracefully and the hyperlink is still usable.

```
a:link {   background-color: #ffffff;
           color: #ff0000;
}
a:visited {  background-color: #ffffff;
             color: #00ff00;
}
a:hover {  background-color: #ffffff;
           color: #000066;
           text-decoration: none;
}
</style>
</head>
<body>
<div align="center">
  <h2>Navigation Links</h2>
  <p><a href="http://yahoo.com">Yahoo!</a>
  <a href="http://google.com">Google</a></p>
</div>
</body>
</html>
```

Save your file as link1.html. Test your page in a browser and compare it with Figure 7.5. The student files contain a sample solution at Chapter7/link1.html. The browser applies the CSS pseudo-class rules to every link on the page. In this example, the CSS was coded using embedded styles, but an external style sheet also could have been used.

Part 2

Now you will create a page that uses CSS and pseudo-classes to configure navigation links that look like buttons. These can be used in place of image links to save on the bandwidth used by graphics. See the sample in Figure 7.6. When the mouse hovers over a navigation button, the text color and border change.

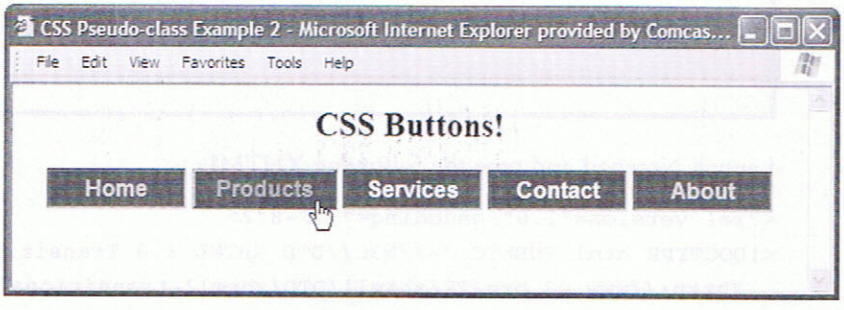


Figure 7.6 The hyperlink's appearance changes when the mouse hovers

You will use the following CSS properties to configure the buttons: `width`, `border`, and `padding`. Let's review these properties. The `width` property configures the amount of horizontal space used by the element in the browser window. The `border` property configures the width (`border-width`), style (`border-style`), and color (`border-color`) of the border around an element. The `padding` property configures the amount of padding—the blank space between the element and its border.

attribute in the XHTML anchor tags connects the link on the page with the CSS style rules in the header.

Part 3

It is often the case that the design of the Web page requires the main navigation links to look different from the links within the content of the pages. You have already created a page, link2.html, with specially configured navigation links. You used the class called `button` to configure these links. In this part of the Hands-On Practice, you will add a line of text containing a hyperlink to the page to verify that the hyperlink retains the default browser appearance and behavior. Figure 7.7 shows a sample page.

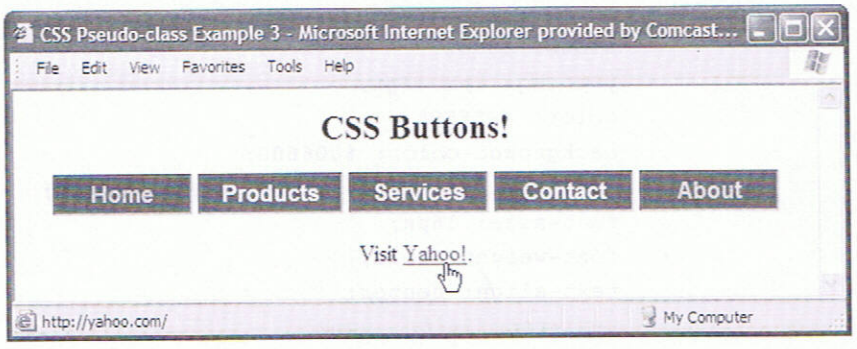


Figure 7.7
The link to Yahoo! retains the default hyperlink properties

Launch Notepad and open your link2.html file. Save the file as link3.html. Modify the title to be "CSS Pseudo-class Example 3" and add the following paragraph under the navigation links:

```
<p>Visit <a href="http://yahoo.com">Yahoo!</a>.</p>
```

Save your file, test your page in a browser, and compare it with the one shown in Figure 7.7. The student files contain a sample solution at Chapter7/link3.html. Because the new link is not part of the defined class `button`, it retains the default hyperlink characteristics. If you needed yet another set of characteristics for links in another section of the page such as the footer, you could define a new class with a unique name and configure pseudo-classes, as was done in Part 2 of this Hands-On Practice.

As you can see, pseudo-classes—along with careful configuration of classes, can be a powerful tool for a Web developer.

7.3 CSS Navigation Layout Using Lists

One of the advantages of using CSS for page layout involves the use of semantically correct code. Writing semantically correct code means using the markup tag that most accurately reflects the purpose of the content. Using the various levels of heading tags for content headings and subheadings, or placing paragraphs of text within paragraph tags (rather than using line breaks) are examples of writing semantically correct code. This type of coding is a step in the direction to support the Semantic Web. Leading Web

You may be wondering how to apply this technique to a horizontal navigation menu such as the one coded on the page1.html page used in Hands-On Practice 6.5. The answer is CSS! List items are block elements. They need to be configured as inline elements to display in a single line. The `display:inline` property is used to accomplish this. Figure 7.11 displays a new version of the page using this technique. The page looks about the same as the original (Figure 6.16) when displayed in a browser even though the XHTML and CSS are configured to use a list.

Figure 7.11 Horizontal navigation using an unordered list configured with CSS

The XHTML code snippet is the same as the one used for the vertical navigation menus shown at the beginning of this section. For the horizontal list to display properly, you must add a CSS configuration for the `<li>` element within the `nav` class as follows:

```
.nav li {  display: inline;
           list-style-type: none;
}
```

Focus on Accessibility

View the home0.html and wildflower0.css files in the Chapter7 folder in the student files to experiment with this technique. See Chapter7/skipnav.html for a version of this page that includes a transparent image configured as an internal link to the named fragment maincontent. This "skip navigation" method allows visitors using screen readers to easily skip repetitive navigation links.

7.4 Three-Column CSS Page Layout

Often a Web page layout will consist of a header across the top of the page with three columns below: navigation, content, and sidebar. If you are thinking about this layout as a series of boxes—you're thinking correctly for configuring pages using CSS! Figure 7.12 shows a wireframe sketch of this page layout design. Figure 7.13 (shown also in the color insert section) shows a Web page configured using this design. You will create this page in the next Hands-On Practice.

HANDS-ON PRACTICE 7.3

In this Hands-On Practice you will develop your first three-column Web page using CSS. The same techniques that you used to configure the two-column page will apply here—think of the page as a series of elements or boxes. Assign ids or classes to the elements as you code the XHTML. Configure the CSS to correspond to the ids and classes. Recall that a key technique in creating a two-column page with left column navigation was to design the left column to float to the left. A key technique in our three-column page is to code the left column with `float:left` and the right column with `float:right`. The center column occupies the middle of the browser window. Refer to Figures 7.12 and 7.13 as you complete this Hands-On Practice.

Getting Started

Locate the showybg.jpg, plsthumb.jpg, and trillium.jpg files in the Chapter 7 folder in the student files. Create a new folder called wildflowers3. Copy the files to the folder.

Part 1—Code the XHTML

Review Figures 7.12 and 7.13. Notice the page elements: a logo area with both a logo and a background image that repeats; a left column with a navigation area and an image; a center column with paragraphs of text, a heading, and an image that floats to the right; a right column with two news items; and a footer. These will all be coded to use ids and classes corresponding to CSS, which configures a number of properties including the `float`, `margin`, `border`, `font-family`, and so on. The navigation menu links will be configured using an unordered list. As you code the XHTML document, you will place the elements on the page and assign `id` and `class` values that correspond to the areas in the sketch in Figure 7.12. Launch Notepad and type in the following XHTML:

```
<<?xml version="1.0" encoding="utf-8"?>
<!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>Door County Wildflowers</title>
</head>
<body>
<div id="container">
  <div id="logo">
    Door County Wildflowers
  </div>
  <div id="left">
    <ul class="navBar">
      <li><a class="nav" href="home.html">Home</a></li>
      <li><a class="nav" href="spring.html">Spring</a></li>
      <li><a class="nav" href="summer.html">Summer</a></li>
      <li><a class="nav" href="fall.html">Fall</a></li>
      <li><a class="nav" href="winter.html">Winter</a></li>
    </ul>
```

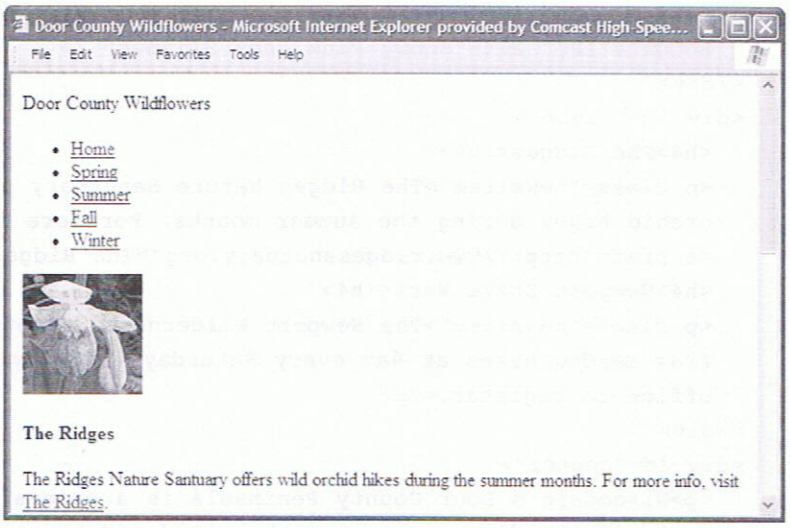


Figure 7.14
The three-column page before CSS is applied

the page and the page begins right at the browser margin. Launch Notepad and open your threecolumn.html file. In the header section of your Web page document, add a tag to begin the embedded styles:

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

Now let's consider the CSS configuration. Type the CSS in your document as it is discussed as follows:

1. **Body Selector.** Set the margin to 0 pixels. Configure the background color to `#ffffff`.

```
body {  margin:0;
        background-color: #ffffff;
}
```

2. **Container.** Configure this area with background (`#eeeeee`) and text (`#006600`) colors, a minimum width of 700 pixels, and with font family of Verdana, Arial or sans-serif.

```
#container {   background-color: #eeeeee;
               color: #006600;
               min-width: 700px;
               font-family: Verdana, Arial, sans-serif;
}
```

3. **Logo.** Code this area so that the image showybg.jpg will repeat using `background-image:url(showybg.jpg)`. The text should be set to 2.5em font size and bold. The height of the logo area is 100 pixels—this corresponds to the height of the background image. Although it will most likely never display, configure the background color to `#eeeeee`. The text color should be `#cc66cc`. Set the left padding to 20 pixels. Configure a 2 pixel solid black border across the bottom of this area as follows:

```
#logo {  color: #cc66cc;
         background-color: #eeeeee;
         font-size: 2.5em;
```

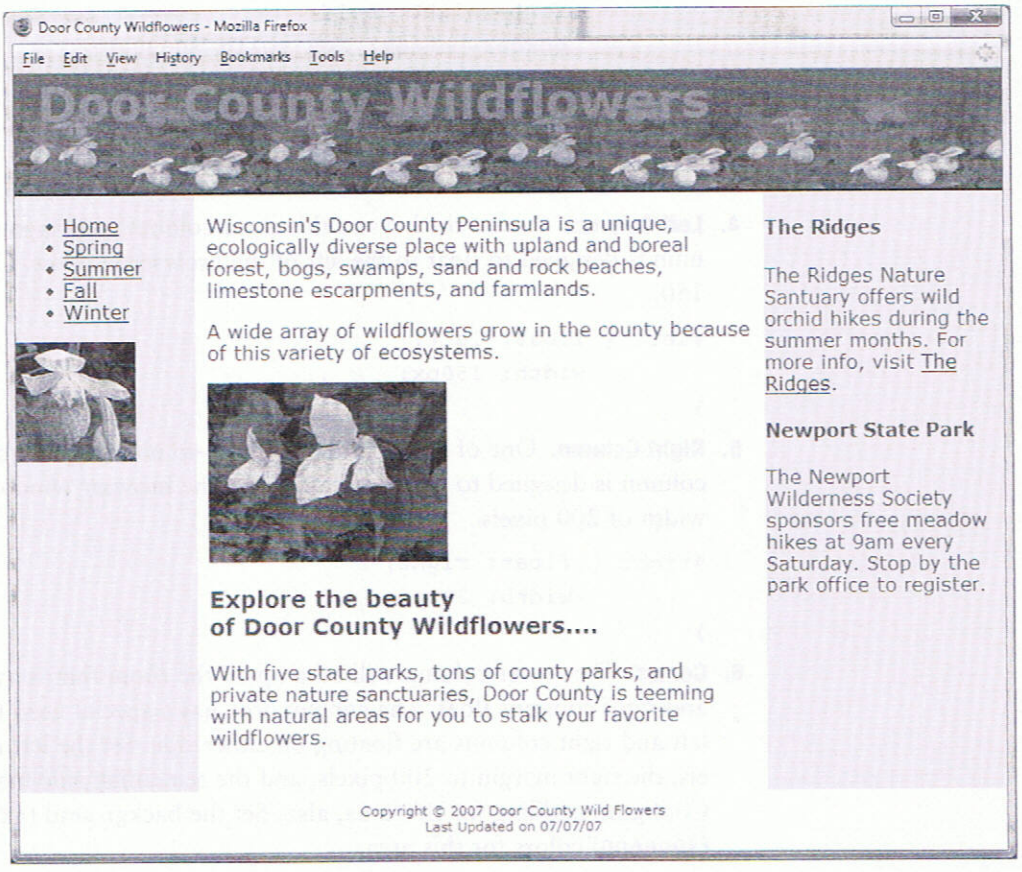


Figure 7.15
The CSS for the basic elements of the three-column layout is complete

Part 3—Continue Coding CSS

Now you are ready to continue with your styles. Open the threecolumn.html page in Notepad and position your cursor on a blank line above the closing style tag. First we will configure the components in the left column as follows:

1. **Navigation Menu.** Configure the unordered list to provide for a 20 pixel top margin and not to display any bullets.

```
.navBar { margin-top: 20px;
          list-display-type: none;
}
```

Configure the navigation links to have no underline (`text-decoration:none`). Configure the font size to 1.2em. Pseudo-classes should be configured for link, visited, and hover with different text colors as follows:

```
.nav {  text-decoration: none;
        font-size: 1.2em;
}
a.nav:link {  color:#006600;}
              background-color: #eeeeee; }
a.nav:visited {  color: #003300;}
                 background-color: #eeeeee; }
a.nav:hover {  color: #cc66cc;}
               background-color: #eeeeee; }
```

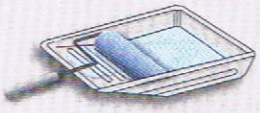
FAQ

How do I create a custom-color scroll bar?

It can be fun to color-coordinate the scroll bar with your Web site! Keep in mind that not all your Web visitors will see your handiwork. While this effect is supported by Internet Explorer, it is not supported by all browsers. To configure a scroll bar with colors that you choose, add the following styles to the body tag: `scrollbar-face-color`, `scrollbar-arrow-color`, and `scrollbar-track-color`. For example:

```
body {  scrollbar-face-color:#cc66cc;
        scrollbar-arrow-color:#006600;
        scrollbar-track-color:#cccccc;
}
```

Note: Your CSS will not pass W3C validation tests if you use these Internet Explorer only properties.



CHECKPOINT 7.1

1. Describe a reason to organize the files in a Web site using folders and subfolders.
2. State a reason to use an unordered list to configure navigation links.
3. You are using CSS pseudo-classes on a Web page to configure the navigation links to look like buttons. You want the "regular" links in the Web page content to be configured as they normally would (not look like a button). Describe how you could configure the styles and XHTML to accomplish this.

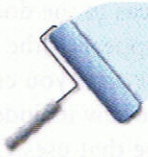
7.5 CSS Styling for Print

Even though the advent of the "paperless society" has been talked about for decades, the fact is that many people still love paper and you can expect your Web pages to be printed. CSS offers you some control over what gets printed and how the printouts are configured.

This is easy to do using external style sheets. Create one external style sheet with the configurations for browser display and a second external style sheet with the special printing configurations. Associate both of the external style sheets to the Web page using two `<link>` elements. The `<link>` elements will utilize a new attribute, called `media`. Configure the link element for your browser display with `media="screen"`. Configure the link element for your printout with `media="print"`. Modern browsers will use the correct style sheet depending on whether they are rendering a screen display or preparing to print a document. An example of the XHTML follows:

```
<link rel="stylesheet" href="wildflower.css" type="text/css"
  media="screen" />
<link rel="stylesheet" href="wildflowerprint.css" type="text/css"
  media="print" />
```

Often `display:none` is used in the print style sheet to prevent banner ads, navigation, or other extraneous areas from appearing on the printout. Another common practice is to configure the font sizes on the print style sheet to use `pt` sizes—this will better con-



HANDS-ON PRACTICE 7.4

In this Hands-On Practice you will code special styles to use when printing a Web page. We will use the page1.html and wildflower.css files that you created in Hands-On Practice 6.5 as a starting point. Figure 6.16 shows the browser display of the page1.html file. You will create a new version of the page1.html file and a new style sheet configured for printing. When printed, the logo will be configured using a 24 `pt` size and the navigation will not display.

Getting Started

Locate the pls.jpg, wildflower.css, and page1.html files in the student files, Chapter7 folder. Create a new folder called wildflowersPrint. Copy the files to the folder.

Part 1—Code the XHTML

Launch Notepad and open page1.html. This page is associated with an external style sheet called wildflower.css. The styles in wildflower.css should be used when the Web page is displayed on the screen. Add the **`media` attribute** with the value of `screen` to the link element for wildflower.css. Code a new link element to invoke an external style sheet called wildflowerprint.css for printing (`media="print"`). The XHTML follows:

```
<link rel="stylesheet" href="wildflower.css" type="text/css"
    media="screen" />
<link rel="stylesheet" href="wildflowerprint.css" type="text/css"
    media="print" />
```

Save the page1.html file in the wildflowersPrint folder.

Part 2—Code the New CSS

Launch Notepad and open wildflower.css. Since you want to keep most of the styles for printing, you will start by creating a new version of the external style sheet. Save wildflower.css with the name of wildflowerprint.css in the wildflowersPrint folder. You will modify three areas on this style sheet: the `contentlogo` id, the `content` id, and the `nav` class configuration.

1. Modify the `contentlogo` id so that the printer will use 24 point font size. The CSS follows:

```
#contentlogo { color: #000000;
               font-size: 24pt;
               padding: 10px;
}
```

2. Modify the `content` id so that the printer will use 12 point font size. The CSS follows:

```
.content {  font-family: Verdana,Arial,sans-serif;
            font-size: 12pt;
            margin: 10px;
}
```

The overall CSS cascade was described above. In addition to this general cascade of CSS types, the style rules themselves follow rules of precedence. Style rules applied to more local elements (such as a paragraph) take precedence over those applied to more global elements (such as a `<div>` which contains the paragraph).

Let's look at an example of the cascade. The CSS and XHTML code is shown below. The CSS has two style rules: a rule creating a class named `content` which configures text using the Arial (or generic sans-serif) font family, and a rule configuring all paragraphs to use the Times New Roman (or generic serif) font family. The CSS follows:

```
.content { font-family:Arial, sans-serif; }
p { font-family: "Times New Roman", serif;}
```

The XHTML on the page contains a `<div>` with multiple elements, such as headings and paragraphs. Partial code follows:

```
<div class="content">
  <h1>Main Heading</h1>
  <p>This is a paragraph. Notice how the paragraph is contained in
  the div.</p>
</div>
```

Here's how the browser would render the code:

1. The text contained in the heading is displayed with Arial font because it is part of the `<div>` assigned to the `content` class. It inherits the properties from its parent (`<div>`) class.
2. The text contained in the paragraph is displayed with Times New Roman font because the browser applied the styles associated with the most local element (the paragraph). Even though the paragraph was contained in (and is considered a child of) the `content` class, the local paragraph style rules took precedence and were applied by the browser.

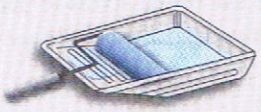
Don't worry if CSS and rules of precedence seem a bit overwhelming at this point. CSS definitely becomes easier with practice. You'll get a chance to practice with the "cascade" as you complete the next Hands-On Practice.

HANDS-ON PRACTICE 7.5

You will experiment with the "cascade" in this Hands-On Practice as you work with a Web page that uses external, embedded, and inline styles. Begin by creating an external style sheet called site.css that sets the `background-color` of the Web page to a shade of yellow (`#FFFF66`) and the `font-size` to `24px`. The code follows:

```
body {  background-color: #FFFF66;
        font-size: 24px;
}
```

Next, create a Web page called mypage1.html that is associated with the file site.css and has an embedded style that sets the text color to blue. The file mypage1.html will contain two paragraphs of text. The XHTML used to code the first paragraph will not use



CHECKPOINT 7.2

1. State an advantage of using CSS to style for print.
2. Describe how to choose whether to configure an XHTML tag, create a class, or create an id when working with CSS.
3. List the following terms in the order that the properties and attributes are applied when using CSS.

   Inline styles
   External styles
   XHTML attributes
   Embedded styles

7. Which of the following is the file extension for an external style sheet?

a. ess
b. css
c. html
d. no file extension is necessary

8. Which of the following elements is used to associate a Web page with an external style sheet?

a. `<target>`
b. `<a>`
c. `<include>`
d. `<link />`

9. Which of the following properties configures an image to use as a bullet point in an unordered list?

a. `bullet-image`
b. `image-style`
c. `list-style-image`
d. `bullet-style-image`

10. Which style rule below causes other page content to appear at the left of the element?

a. `position:left;`
b. `position:relative;`
c. `float:left;`
d. `float:right;`

Fill in the Blank

11. To indicate that an external style sheet is used to configure printing, code _____ on the `<link>` element.

12. The _____ is always transparent.

13. The _____ pseudo-class can be used to modify the display of a hyperlink when a mouse passes over it.

14. _____ is an attribute of the anchor element that can cause the new Web page to open in its own browser window.

15. The rules of _____ describe how Cascading Style Sheet rules, XHTML attributes, and browser defaults are applied.

Apply Your Knowledge

1. Predict the Result. Draw and write a brief description of the Web page that will be created with the following XHTML code:

```
<?xml version="1.0" encoding="utf-8"?>
<!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>Predict the Result</title>
<style type="text/css">
body { background-color: #000066;
       color: #CCCCCC;
       font-family: Arial,sans-serif;
}
h1 { background-color: #FFFFFF;
     color: #000066;
     padding: 20px;
}
.navBar { list-style-type: none;
          display: inline;
          padding: 20px;
}
```

```
#rightcolumn { "_": "_";
               background-color: #ffffff;
               color: #000000;
               padding: 20px;
}
</style>
</head>
<body>
<div id="leftcolumn">
  <ul>
    <li><a href="home.html">Home</a></li>
    <li><a href="spring.html">Spring</a></li>
    <li><a href="summer.html">Summer</a></li>
    <li><a href="fall.html">Fall</a></li>
    <li><a href="winter.html">Winter</a></li>
  </ul>
</div>
<div id="rightcolumn">
  <h1>Trillium Media Design</h1>
  <p>Our professional staff takes pride in its working
  relationship with our clients by offering personalized
  services that listen to their needs, develop their target
  areas, and incorporate these items into a well presented web
  site that works.
  </p>
  <div>
  Contact <a href="mailto:web@trilliumtechnologies.com">
  web@trilliumtechnologies.com</a>
  <br />Copyright &copy; 2008 Trillium Media Design
  </div>
</div>
</body>
</html>
```

3. Find the Error. The page below is intended for the navigation area to display on the right side of the browser window. What needs to be changed to make this happen?

```
<?xml version="1.0" encoding="utf-8"?>
<!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>Find the Error</title>
<style type="text/css">
body { background-color: #d5edb3;
       color: #000066;
       font-family: Verdana, Arial, sans-serif;
}
#rightcolumn { float: left;
               width: 120px;
}
```

mywildflower.css. (Modify pre-existing pages to use this style sheet). Rename threecolumn.html as appropriate. The following is some content to include on the new pages:

Spring Page (spring.html):

- Use the trillium.jpg image (see the Chapter7 folder in the student files).
- Trillium facts: 8–18 inches tall, perennial, native plant, grows in rich moist deciduous woodlands, white flowers turn pink with age, fruit is a single red berry, protected flower species.

Summer Page (summer.html):

- Use the yls.jpg image (see the Chapter7 folder in the student files).
- Yellow Lady's Slipper facts: 4–24 inches tall, perennial, native plant, grows in wet shaded deciduous woods, swamps, and bogs, an orchid, official flower of Door County.

Hand in printouts of mywildflower.css, spring.html source code (print in Notepad), summer.html source code, the browser display of spring.html, and the browser display of summer.html to your instructor.

7. **Configure Printing for Hands-On Practice 7.3.** Configure special printing for the threecolumn.html file created in Hands-On Practice 7.3. Use the threecolumn.html file from Hands-On Practice 7.3 as a starting point. This file is in the Chapter7 folder in the student files. Save a copy of this file as threecolumnprint.html. Modify the file so that it links to an external style sheet called threecolumn.css instead of using embedded styles. Save and test your page. Create a new style sheet, called myprint.css, which will prevent the navigation from displaying when the page is printed. Modify the threecolumnprint.html page to link to this file. Review the use of the `media` attribute on the link element. Save all files and test your page. Select File, Print Preview to test the print styles. Hand in printouts of myprint.css, threecolumn.css, threecolumnprint.html source code (print in Notepad), and the browser display of threecolumprint.html to your instructor.

8. **Modify the Design of Hands-On Practice 7.3.** Locate the threecolumn.html page you created in Hands-On Practice 7.3. This file is in the Chapter7 folder in the student files. Recall from Chapter 5 that a Web page using jello design has content in the center of the Web page with blank margins on either side. Configure the style rules for threecolumn.html to display the page in this manner. Refer to Chapter 5 for CSS style rule suggestions. Hand in printouts of the source code (print in Notepad) and browser display for the Web page to your instructor.

9. **Practice Validating CSS.** Choose a CSS external style sheet file to validate—perhaps you have created one for your own Web site. Otherwise, use an external style sheet file that you worked with in this chapter. Use the W3C free CSS validator (http://jigsaw.w3.org/css-validator/). If your CSS does not immediately pass the validation test, modify it and test again. Repeat this process until the W3C validates your CSS code. Write a one or two paragraph summary about the validation process. Answer the following questions. Was it easy to use? Did anything surprise you? Did you encounter a number of errors or just a few? How easy was it to determine how to correct the CSS file? Would you recommend this to other students? Why or why not?

Hands-On Practice Case

1. Review Section 7.2 CSS Pseudo-classes and Links and Section 7.3 CSS Navigation Layout using Lists.
2. Modify the javajam.css file as needed to configure the main navigation links in an unordered list without "bullets". Also configure the main navigation links to change color when a mouse hovers over them.
3. Modify the index.html, menu.html, and music.html Web pages to display the main navigation links in an unordered list.
4. Launch a browser and test the pages in the javajamcss folder. Modify your java.css file as needed to configure your pages. Be sure to test in more than one browser.

Fish Creek Animal Hospital

See Chapter 2 for an introduction to the Fish Creek Animal Hospital Case Study. Figure 2.30 shows a site map for the Fish Creek Web site. The pages were created in earlier chapters. You will use the existing Web site in the fishcreekcss folder (unless your instructor specifies otherwise) as a start and create a new version that configures the main navigation using an unordered list.

Hands-On Practice Case

1. Review Section 7.3 CSS Navigation Layout using Lists.
2. Modify the fishcreek.css file as needed to configure the main navigation links in an unordered list without "bullets". *Hint*: To eliminate the extra space on the left side of the fish navigation links, use CSS to configure the unordered list to have 0 margin and padding on the left side.
3. Modify the index.html, services.html, and askvet.html Web pages to display the main navigation links in an unordered list.
4. Launch a browser and test the pages in the fishcreekcss folder. Modify your fishcreek.css file as needed to configure your pages. Be sure to test in more than one browser.

Pete the Painter

See Chapter 2 for an introduction to the Pete the Painter Case Study. Figure 2.34 shows a site map for the Pete the Painter Web site. The pages were created in earlier chapters. You will use the existing Web site in the paintercss folder (unless your instructor specifies otherwise) as a start and create a new version that configures the main navigation using an unordered list.

Hands-On Practice Case

1. Review Section 7.2 CSS Pseudo-classes and Links and Section 7.3 CSS Navigation Layout using Lists.
2. Modify the painter.css file as needed to configure the main navigation links in an unordered list without "bullets". Also configure the main navigation links to change color when a mouse hovers over them. *Hint*: To eliminate the extra space