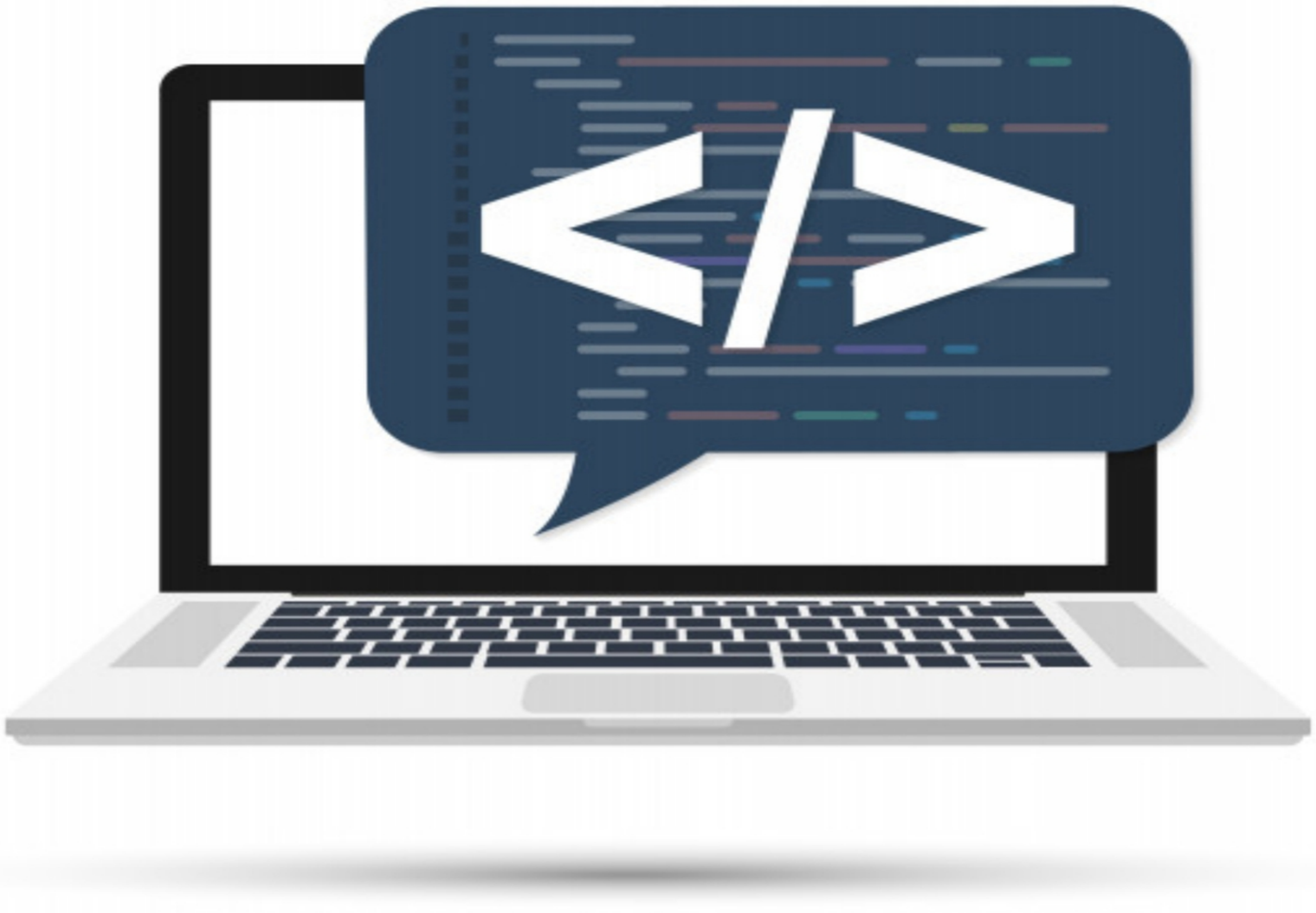
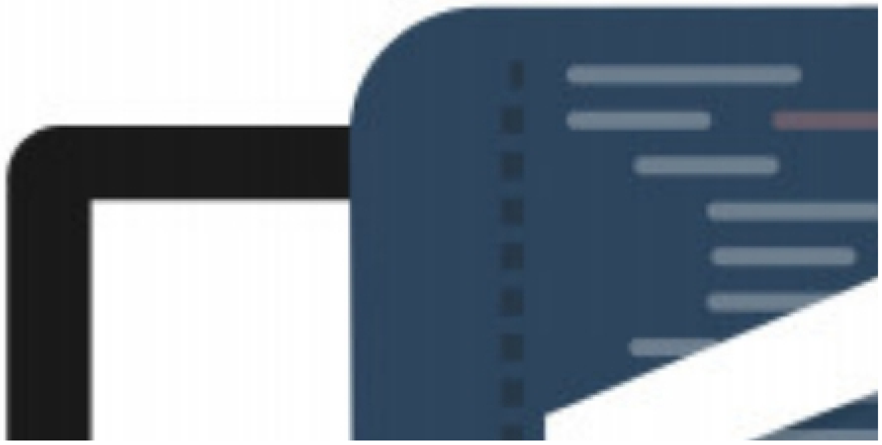
Programming in HTML



Completed In 1 DAY

Progra
in HTML



Programming in
H T M L

Contents

Introducing HTML 1

This chapter provides an introduction to using HTML to create simple Web pages suitable for interfacing with PHP applications. Several examples show how to modify the appearance of a document by using HTML tags and their attributes.

1. Introducing the Tools 1. What Is an HTML Document?

HTML is an acronym for HyperText Markup Language. HTML documents, the foundation of all content appearing on the World Wide Web (WWW), consist of two essential parts: information content and a set of instructions that tells your computer how to display that content. These instructions—the “markup,” in editorial jargon—comprise the HTML “language.” It is not a programming language in the traditional sense, but rather a set of instructions about how to display content in a Web browser. Ideally, online content should look the same regardless of the browser being used or the operating system on which the browser resides. This goal of complete platform independence is achieved only approximately in practice.

Every HTML document should contain a minimum of four elements:

<html> …< / h t m l > <head>…</head> < t i t l e >…< / t i t l e >
<body>…</body>

These elements de fine the essential parts of an HTML document: the document itself, a heading section, a title section, and a body. All four elements should be included even if they don’t enclose any content. Every HTML element is defined by one or two tags—usually a start tag and an end tag. Tags are always enclosed in angle brackets: <…>. End tags start with a slash (/). A few HTML elements have only one tag.

The four basic elements are organized as follows within an HTML document:

```
<html>
<head>

< t i t l e > …< / t i t l e > </head>
<body>

…
</body>
< / h t m l >
```

The h tm l tag encloses all other tags and defines the boundaries of the HTML document. We will return to the other tags later. The indenting used to set off pairs of tags is optional, but it makes documents easier to create, read, and edit. This style is part of good programming practice in all languages.

HTML documents are usually used as to distribute information for access on the Web. However, for the purposes of this book, HTML documents will be used along with the PHP programming language to create an environment for solving a wide range of computing problems.

Good programming technique often involves separating the input/output (I/O) interface from the underlying calculations that do the work of a program. The HTML/PHP programming environment provides a conceptually elegant means of implementing this strategy. An HTML document provides the I/O interface and PHP handles the calculations. An advantage of HTML is that it provides a wealth of interface possibilities that far surpass those of older text-based languages.

1.1.2 How Do You Create HTML Documents?

Because HTML documents are just text documents, they can be created with any text editor. Even Windows' very basic Notepad application is a workable choice for simple tasks.[1] Once HTMLfiles have been created, you can open them in your computer's browser, hopefully without regard to which browser you are using. As long as you give such documents an .htm or . h t m lfile name extension, they should automatically open in your browser when you double-click on thefile name. Although Windows documents are no longer restricted to three-letter extensions, a convention dating back to the preWindows days of MS-DOS operating systems, the three-letter .htm extension is often used on Windows systems. The four-letter . h t m l extension is commonly used on UNIX systems. However, either extension is perfectly acceptable.

[1] When you save afile in Notepad, the default extension is . t x t , so you will have to specify .htm or . h t m l as afile extension.

There is one other consequence of using Windows computers for creating all the code examples in this text: Windowsfile names are caseinsensitive, while on UNIX systems, all spellings, includingfile names and commands, are case-sensitive. In Windows, you can name a document newDocument.htm. Later, you can spell it newdocument.htm, NEWDOCUMENT.HTM, or any other combination of uppercase and lowercase letters and it won't matter. On a UNIX system, thatfile will be recognized only with the originalspelling.

Although you can create text (and, therefore, HTML) documents with a full-featured word processor such as Microsoft Word, this is not recommended. When you save a word processor document it no longer contains just the text you have typed, but also all the layout and formatting information that goes with that document. You can choose to save a document as just text with an .htm extension, but it is easy to forget to do this.

Microsoft Word and other modern word-processing applications can also format any document as an HTML document. However, this is also not recommended. These converted documents may include a huge quantity of extraneous information and HTML instructions that make the resultingfile much larger and more complex than it needs to be. (To see this for yourself, save a Word document as an HTML document and then look at the result in a text editor such as Notepad!)

RTF (“rich text format”) documents are also unacceptable, as they still retain some formatting information that is inappropriate for an HTML document. Any document that contains“smart quotes” rather than "straight quotes" can also cause problems, because smart quotes may not be displayed properly by browsers. (This is much less of a problem on current browsers than it used to be.)

There are commercial Web development tools that allow you to create Web pages without actually knowing anything about HTML. These applications are not suitable for use with this book. The obvious reason is that the primary purpose of the book is to show you how to write your own HTML documents to be used as an interface to a PHPfile. Also, these applications may create HTMLfiles that are much larger and more complex than they need to be. Such applications are better suited for Web development projects that involve a lot of the other“bells and whistles” that make commercial Web pages attractive.

Creating an HTML document that does what you need it do inevitably involves switching back and forth between a text editor and a browser—making changes and observing the effects of those changes. A good editor should allow you to move back and forth quickly and easily between the source document and its display in a browser when you make changes. It is certainly possible, but not particularly convenient, to do this with a simple text editor such asNotepad.

There are many commercial software tools whose purpose is to facilitate writing and editing HTML documents by integrating document creation, editing, and viewing. As noted above, some of them are intended for large and complicated projects and may be“overkill” for use with this book. For many years, I have used Visicom Media’s freeware AceHTML. This software is no longer available, but there are other freeware alternatives which provide automatic color-based text formatting, coding tools, and an integrated browser that makes it easy to create and edit HTML and PHP documents.

Although it *shouldn ’t* make any difference which browser you use, it is worth noting that all the HTML documents displayed in this text come from either AceHTML’s internal browser or Mozilla’s Firefox, which is the default browser on the author’s Windows computers.

1.1.3 Some Typographic Conventions Used in This Book

HTML tags and other code are printed in a monospaced (C o u r i e r) f o n t in document examples and whenever they are referred to in the text. Thus, document is interpreted as a reference to an HTML object, as opposed to its general use as a term identifying a body of text. Within descriptions of HTML and PHP document features, user-supplied text is denoted by *{italicized text in braces (curly brackets)}* ; the curly brackets are usually *not* meant to be included in the user-supplied text.

AceHTML and other editors typically apply some combination of color coding, bold fonts, and italicized fonts to various language elements. When HTML and PHP code is copied from the editor and inserted into this book, bold and italic fonts are retained but the color coding is not.

The renderings of HTML documents and other output as displayed in a browser window have been captured and edited on a Windows computer by pressing the PrtScn (Print Screen) key and copying the resulting screen image into an image editing program.[2] Pressing Alt-PrtScn copies just the currently active window instead of the entire screen.

[2] For many years, I have used the freeware IrfanView program.

Because of the small format of this book, line breaks in HTML document examples, and later in PHP scripts, are often necessary and may sometimes cause problems. Although every effort has been made to use line breaks in a way that does not cause problems, it will sometimes be necessary to remove breaks and“rejoin” some lines when you reproduce these documents for your own use.

1.1.4 Finding More Information About HTML

It should be clear that this book is in no way intended as a reference source for HTML. Any attempt to provide complete coverage of HTML would thoroughly confound the purpose of the book and is far beyond the author’s capabilities! You can easilyfind support online for those portions of HTML required for its use as an interface for PHP applications. Here are two standard reference sources for HTML programmers.

Thomas Powell, *HTML: The Complete Reference, Third Edition* , 2001, Osborne/McGraw-Hill, Berkeley, CA. ISBN0-07-212951-4.

Thomas Powell and Dan Whitworth, *HTML Programmer 's Reference, Second Edition* , 2001, Osborne/McGraw-Hill, Berkeley, CA. ISBN , 2001, Osborne/McGraw-Hill, Berkeley, CA. ISBN 213232-9.

These exhaustive treatments will tell more than you will ever need to know about using HTML!
1.2 Your First HTML Document

A typical first goal in learning any programming language is to display a simple message. With HTML, this is trivially simple: Just type the message in the body of the document, as shown in Document 1.1. (Appendix 1 contains an index to all documents in the text.) Save thefile with the name shown.

Document 1.1 (HelloWorldHTML.htm)
```
>jvon@
>jgcf@

>vkvng@ First HTML Document >1vkvng@
>1jgcf@
>dqf{@
Hello, world!Hello ,world >1dqf{@

>1jvon@
```

Many examples presented in this text will include a browser 's rendering of the screen output produced by the document. When a border appears around the output, as it does for the output from Document 1.1, the purpose is to distinguish the output from the rest of the text—the document doesn't generate that border. In the text, renderings are always in black and white or grayscale. Some documents will produce colored output, but you will have to try the code yourself to see these results.

Document 1.1 is certainly not very exciting. The point is that an HTML document simply displays the static content you provide. As you will learn in Chap. 2 , HTML provides many facilities for changing the *appearance* of this content, but not the content itself.

HTML syntax is case-insensitive, which means that <html> is equivalent to <HTML> or even <hTmL>. Some HTML document authors favor uppercase spellings for tags because they stand out from the text content. However, XHTML (extensible HTML), the apparent successor to HTML, requires tags to be in lowercase letters.[3] Hence, this text will always use lowercase letters for tag names. Note that, despite previous warnings thatfile names and commands are case-sensitive in some systems, browsers should not be case-sensitive in their interpretation of HTML tags.

[3] Although this book adopts some XHTML style rules, the documents are written in HTML and are not intended to be fully XHTML-compliant.

1.3 Accessing HTML Documents on the Web

Documents intended for access by others on the World Wide Web are posted on a Web server, a computer system connected to the Internet. Colleges and universities typically provide Web servers for use by their faculty and students. Individuals not affiliated with an institution may have to purchase space on a commercial Web server, or they can set up their own server. In any case, access to Web pages is universal in the sense that any computer with an Internet connection and a browser can request to be connected to a website through its Internet address—its Uniform Resource Locator (URL).

Not all HTML documents have to be publicly accessible on the Web. They can be protected with logon identifications and passwords, or they can be available only locally through an intranet (as opposed to the Internet). The Internet is a global network of interconnected computers, whereas an intranet is a local network that may or may not also provide connections to the Internet. For example, a company can provide an intranet with no external access, exclusively for internal use by its own employees.

Note that when you view HTML documents in the browser on your local computer, they are not available on the Internet unless you have specifically set up a server, assigned it a URL, and placed HTML documents in a folder associated with that server. If you are associated with a university or other institution, you may be able to put internet content on its server. Typically, commercial providers of online services for individuals do not allow you to set up servers with afixed URL. In that case, you have to purchase a domain name and set up a URL with an Internet Service Provider (ISP) which specializes in in Web hosting.

A university Internet address might look something like this: http://www.m yUni v ersi ty.edu/m yNam e/i ndex.htm
The URL for the author’s organization looks like this:
h t t p : / / w w w . i n s t e s r e . o r g /

URLs usually start with the h t t p :// prefix, to indicate that the Hypertext Transfer Protocol (HTTP) is being used. There are some variations, such as h t t p s, indicating that the address that follows resides on a secure server, as required forfinancial transactions, for example. The rest of the address identifies a Web server and then a folder or directory on a computer system. The .edu extension identifies this site as belonging to an educational institution, in the same way as .gov, .com , and . o r g identify government, commercial, and organization sites. Sometimes names in URLs are case-sensitive, depending on the operating system installed on the computer system containing the Web page. Users of Windows computers should note the use of forward slashes rather than backslashes to separate folders (or directories).

The i n dex .htm (or i n d ex . h tm l)file contains the home page for a web site. By default, the i n dex .htm file is automatically opened, if it exists, whenever this URL is accessed. That is, the address

http://www.m yUni v ersi ty.edu/m yNam e/
is equivalent to the address that includes the i n d e x .h tmfile name.

As they were being developed, the HTML documents discussed in this book resided neither on the Internet nor on an intranet. Instead, they were stored in a folder on a local computer and accessed simply by doubleclicking on them.

You should create a separate folder on your computer as you work through the examples in this book and write your own documents. You *could* make documents you create yourself accessible on the Internet or an intranet by placing them on a Web server. For example, if you are taking a course based on this book, your instructor may require you to post homework assignments on a Web site.

1.4 Another Example

This example shows how to include an image in an HTML document.
Document 1.2 (house.htm)

```
< h t m l >
< h e a d >
< t i t l e > O u r New H o u s e < / t i t l e > < s c r i p t
t y p e = " t e x t / j a v a s c r i p t " > c o l o r = ' g r e e n ' > T h i s
document "+ d o c u m e n t . l a s t M o d ified + " < / f o n t > " ) ; < / s c r i
p t >
< / h e a d >
<body>
<h1>Our New House</h1>
<p>
l a n g u a g e = " j a v a s c r i p t "

d o c u m e n t . w r i t e ( " < f o n t was l a s t modified on
```

```
H e r e ' s t h e s t a t u s o f o u r new house. (We know y o u ' r e f a s c i n a
t e d ! ) < / p >
```
<!-Link t o your image goes h e r e . -- >
```
<i m g s r c = " h o u s e . j p g " a l i g n = " l e f t " / > < b r /> < / b o d y >
< / h t m l >
```

Although this book doesn’t deal with the JavaScript language, Document 1.2 does include one short but very useful JavaScript script:

```
< s c r i p t l a n g u a g e = " j a v a s c r i p t "
t y p e = " t e x t / j a v a s c r i p t " > d o c u m e n t . w r i t e ( " < f o n t c o
l o r = ' g r e e n ' > T h i s document was l a s t modified on "+ d o c u m e n
t . l a s t M o d ified + " < / f o n t > " ) ;

< / s c r i p t >
```

This script displays the date on which the document was last modi fied. It is always a good idea to provide users of your documents (including yourself!) with some information about how recent the document is. As you have probably noticed, many websites do not include this information, so it is impossible to tell whether information on that site is current or not.

As mentioned previously, long lines of code are sometimes broken to fit in the pages of this text. The docum ent .wri te statement extending over two lines is one such case. If you copy this text, as is, into an editor and save the document like that, the message won’t be displayed. To make it work again, you have to reassemble the entire statement on just one long line.

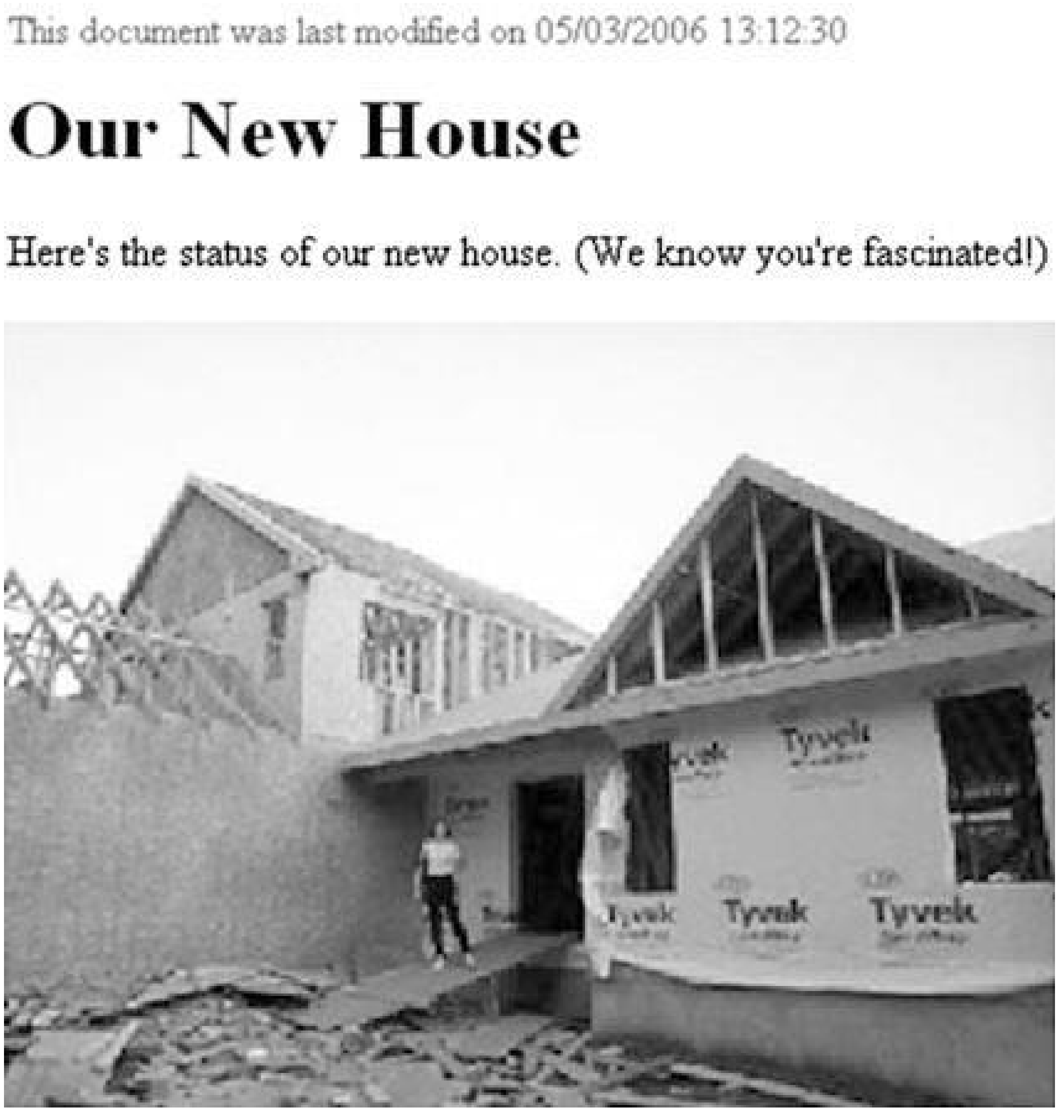
There are several image formats that are widely used in HTML documents, including image bitmaps (.bmp), Graphics Interchange Format (. g i f), and Joint Photographic Experts Group(.j pg).

The original .jpgfile used in Document 1.2 has been compressed, and this process can result in jagged edges where edges should be straight. This effect is visible in the house framing and roof lines.

Within the img element, h e i g h t andwi dth attributes allow you to control the size of the image display (in pixels). However, this is not necessarily a good idea for photos like this because it is not equivalent to actually“resizing” the image, as is possible with image-editing software.[4] Hence, it is important to use images that initially are sized appropriately. The house.jpg image was resized to 300 pixels high by 400 pixels wide, which retained the height-to-width ratio of the original (cropped) photo. If a very large highresolution imagefile is displayed as a very small image, using thehei ght and wi dth attributes, the original largefile must still be transmitted to the client computer. In view of the fact that high-resolution images can produce very large files (>10 Mb), it is still important to consider appropriate resolution and sizing for images included in HTML documents, even in an age of high-speed broadband Internet connections and large amounts of online storage space. (The size of the compressed grayscalehouse.j pg image printed here is about 93 Kb.)

Document 1.2 could be made into a default home page simply by changing its name toi ndex .htm.
Here is afinal admonition which hopefully does not sound too preachy: Intellectual honesty and fairness in the use of other people’s material is important, no matter what the setting. The image displayed by Document 1.2

was taken by this book’s author, of his own house under construction. In other words, the author“owns” this image. Whenever you post images or other material online, please be careful to respect intellectual property rights. Your default approach should be that online materials are copyrighted and cannot be used freely without permission. If you are in doubt about whether you have permission to use an image or other material, don’t!



This document was last modified on 05/03/2006 13:12:30

Our New House

Here's the status of our new house. (We know you're fascinated!)

4 IrfanView (www.irfanview.com) has been used for all image processing in this book.

HTML Document Basics 2

This chapter describes the characteristics of an HTML document, including some of the basic HTML elements and their attributes. The list of attributes is not complete, but is restricted to a subset larger than will usually be needed for working with PHP. The chapter includes a description of how to set colors in documents and a brief introduction to cascading style sheets.

1. Documents, Elements, Attributes, and Values 1. Documents and Their Essential Elements

As noted in Chap. 1 , a basic HTML document consists of four sections defined by four sets of element tags, arranged as follows:

```
<html>
<head>
<title>… < / t i t l e > …
</head>
<body>
…
</body>
< / h t m l >
```

Each of these four required elements has a start tag and an end tag. The < t i t l e >…< / t i t l e > element is nested inside the <head>…</head> tag. This element, which search engines use tofind documents on the Web, is required even if you don't include text for a title. Tags are always enclosed in angle brackets <…>and the end tag always includes a forward slash before the element name. The body element[1] supports attributes that can be used to control the overall appearance of an HTML document. Documents, elements, attributes, and values are organized in a specific hierarchy: HTML document ! elements ! attributes ! values

[1] Often, this book will use just the name of the element, like body , to refer to the element with its tags: <body>…</body> .

Elements exist within a document. Elements can have attributes and attributes (usually) have values. All elements are nested inside the <html>…</html> element. Some of the elements, such as < t i t l e >…< / t i t l e >, are nested inside other elements. Some elements have only a single tag, in which case a forward slash precedes the closing angle bracket. For example, the <br /> tag is used to start a new line.

Following is a brief description of the four elements that will be part of every HTML document. Attributes, if any, are listed for each element. Note, however, that not all possible attributes are listed. Thus, a listing of“none” may mean that there are attributes for this element, but that they are not used in this book. Consult an HTML reference manual for a complete list of attributes.

`<body> … </body>`

The `body` element contains the HTML document content, along with whatever elements are required to format, access, and manipulate the content.
Attributes : `background` , `bgcolor` ,`text`

`<head> … </head>`

The `head` element contains information about the document. The `head` element must contain a`title` element and under XHTML rules, the`title` must be the first element after`head` .
Attributes : none

`<html> … </html>`

The `html` element surrounds the entire document. All other HTML elements are nested within this element.
Attributes : none

`<title> … </title>`

The `title` element contains the text that will be displayed in the browser’s title bar. Every HTML document should have a title, included as the first element inside the`head` element.
Attributes : none

2.1.2 Some Other Important Elements

The four basic elements discussed above constitute no more than a blank template for an HTML document. Other elements are needed to display and control the appearance of content within the document. Here are some important elements that you will use over and over again in your HTML documents. They are listed in alphabetical order. The list of attributes is not necessarily complete, but includes only those which will be used in this book. Because several elements can share common attributes, attributes and their values are listed separately, following the list of elements.

`<a> … </a>`

The `a` (for “anchor”) element provides links to an external resource or to an internal link within a document.
Attributes: `href` ,`name`

`<b> … </b>`
The`b` element forces the included text to be displayed in a bold font.
This is a“physical element” in the sense that it is associated specifically with displaying text in a bold font.
Attributes : none

`<br />`

The `br` element inserts a break (line feed) in the text. Multiple breaks can be used to insert multiple blank lines between sections of text. The break element has no end tag because it encloses no content. Under XHTML rules, a closing slash (after a space) must be included:`<br />` . The slash is rarely seen in older HTML documents, so its use will be encouraged but not required.
Attributes : none

`<center> … </center>`

The `center` element causes displayed text to be centered on the computer screen.
Attributes : none

`<font> … </font>`

The `font` element controls the appearance of text. The two most commonly used attributes control the size and color of the text. *Attributes* : `size` , `color` ,`face`

`<hr />`

The horizontal rule element draws a shaded horizontal line across the screen. It does not have an end tag. A closing slash (after a space) is required in XHTML. A `noshade` attribute displays the rule as a solid color, rather than shaded.
Attributes : `align` , `color` , `noshade` , `size` ,`width`

`<h`*n*`>` … `</h`*n*`>`
Up to six levels of headings (for *n* ranging from 1 to 6) can be defined, with decreasing font sizes as *n* increases from 1 to 6. *Attributes* :`align`

`<i>` … `</i>`
`i` is a“physical element” that forces the included text to be displayed
in italics. The actual appearance may depend on the browser and computer used.
Attributes : none

`<img />`

The `img` element provides a link to an image to be displayed within a document. The image is stored in a separate file, perhaps even at another Web address, the location of which is provided by the`src` attribute.
Attributes : `align` , `border` , `height` , `src` , `vspace` ,`width`

`<p>` … `</p>`

The `p` element marks the beginning and end of a paragraph of text content. Note that HTML does not automatically indent paragraphs. Rather, it separates paragraphs with an empty line, with all the text aligned left. It is common to see only the start tag used in HTML documents, without the corresponding end tag. However, the use of the end tag is enforced by XHTML and this is the style that should be followed. *Attributes* : none

`<pre> … </pre>`

The default behavior of HTML is to collapse multiple spaces, line feeds, and tabs to a single space. This destroys some of the text formatting that you may wish to preserve in a document, such as tabs at the beginning of paragraphs.

The `pre` element forces HTML to recognize multiple spaces, line feeds, and tabs embedded in text. The default action for`pre` is to use a monospaced font such as`Courier` . This may not always be appropriate. But, because line feeds and other text placement conventions are recognized, `pre` is very useful for embedding programming code examples within an HTML document.
Attributes : none

Note that most of the elements described here require both start and end tags. The general rule is that any element enclosing content requires both a start and end tag. The <br /> and < hr /> elements, for example, do not enclose content, so no end tag is needed.

Description of attributes:

These descriptions may not include all possible values. For a complete listing, consult an HTML reference manual. Values of attributes should be enclosed in single or double straight quotes, although HTML doesn’t enforce this requirement.

`align = "…"`
Values : `"left"` , `"right"` , or`"center"`
Aligns text horizontally.
`background = "…"`
Value : the URL of a gif- or jpeg-format graphics file

Setting the background attribute displays the specified image as the background, behind a displayed HTML document page. Depending on the image size (in pixels), background images may automatically be“tiled,” resulting in a repeating image that can be visually distracting. It is never required to use background images, and they should be used with care.

`bgcolor = "…"`
Values : Background colors can be set either by name or by specifying the intensity of red, green, and blue color components. This topic is addressed in section **2.4 Selecting and Using Colors** .

`border="…"`
Value : The width, in pixels, of a border surrounding an image

`color = "…"`
Values : Text colors can be set either by name or by directly specifying the intensity of red, green, and blue color components. See section **2.4 Selecting and Using Colors.**

`face = "…"`
Values : Font typefaces can be set either `monospace` , `sans-serif` , or `serif` , or supported by theuser’s computer. The generic names should always produce something that looks reasonable on any computer, but specific font names that are not available on theuser’s computer may produce unexpected results.

`height = "…"`
Value : The displayed height of an image in pixels (`width="80"` , for example) or, when followed by a`%` sign (`width="80%"` , for example), as a percent of total screen height. The displayed height overrides the actual height of the image file—the number of rows in the image.

`href = "…"`
Value: The URL of an external or internal Web resource, or the name of an internal document reference.
generically, with `cursive` ,

with specific font names

`hspace = "…"`
Value: The horizontal space, in pixels, between an image and the surrounding text.
`name = "…"`

Value: The name assigned to an internal document reference through an "`a` " element.

`size = "…"`
Values : An unsigned integer from 1 to 7 or a signed number from +1 to +6 or -1 to -6.

An unsigned integer is an absolute font size, which may be systemdependent. The default value is 3. A signed integer is a font size relative to the current font size, larger for positive values and smaller for negative values.

For the`hr` element,`size` is the vertical height of the horizontal rule, in pixels.

`src = "…"`
Value : As an attribute for an`img` tag, the URL of a graphics file. For local use, images and their HTML document are usually stored in the same folder.

`text = "…"`
Values : The`text` attribute, used with the`body` element, selects the color of text in a document, which prevails unless overridden by a `font` attribute.

`vspace = "…"`
Value: The vertical space, in pixels, between an image and the surrounding text.

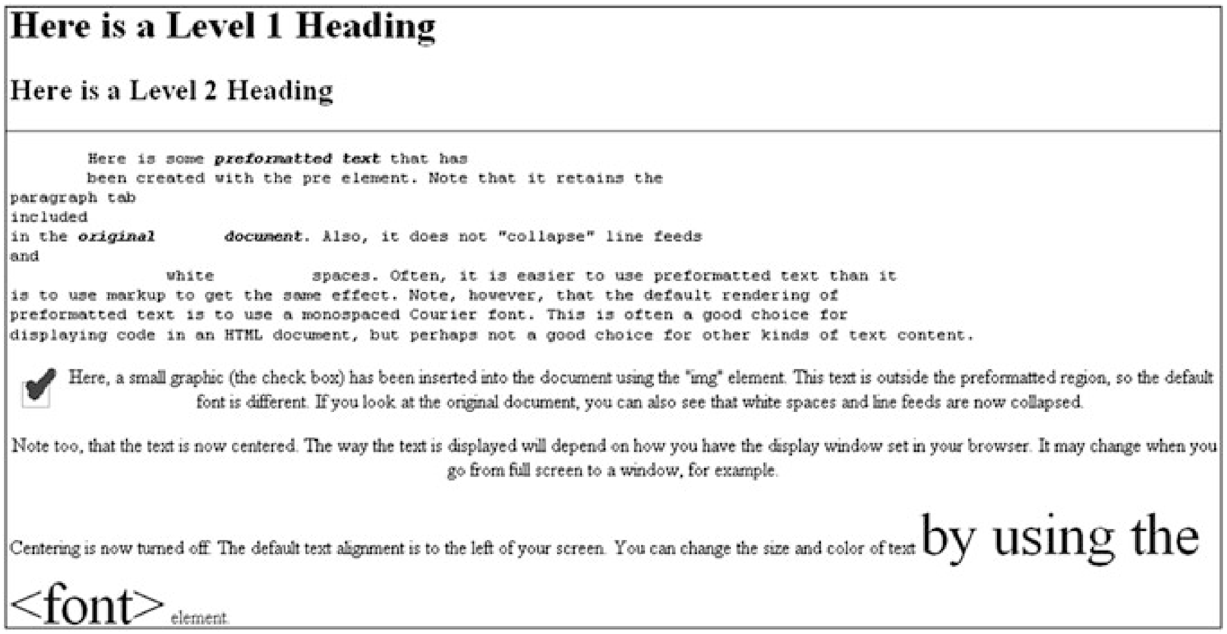
`width = "…"`
Values : The width of an image or horizontal rule, in pixels or as a percent of total screen width, in percent. For example,`width="80"` is interpreted as a width of 80 pixels, but`width="80%"` is a width equal to 80 percent of the total screen width. The displayed width overrides the actual width of the image file—the number of columns in the image.

Document 2.1 shows how to use some of these elements. Document 2.1 (tagExamples.htm)

```
<html>
<head>
<title>Tag Examples</title>
</head>
<body bgcolor="white">
<h1>Here is a Level 1 Heading</h1>
<h2>Here is a Level 2 Heading</h2>
<hr />
<pre>
    Here is some <b><i>preformatted text</i></b>
that has
   been created with the pre element. Note that it retains
the
paragraph tab
included
in the <b><i>original       document</b></i>. Also, it does
not "collapse" line feeds
and
white           spaces. Often, it is easier to use
preformatted text than it
is to use markup to get the same effect. Note, however, that
the default
rendering of
preformatted text is to use a monospaced Courier font. This
is often a good choice for
displaying code in an HTML document, but perhaps not a good
choice for other kinds of text content.
```

```
</pre><p><center>
<img src="checkmark.gif"
align="left" />Here, a small
graphic (the check box) has been inserted into
the document using the "img" element. This text is outside
the preformatted
region, so the default font is different. If you look at the
original document, you can also see that
white         spaces and line   feeds are now collapsed.
</p><p>
Note too, that the text is now centered. The way the text is
displayed will depend on how you
have the display window set in your browser. It may change
when you go from full screen to a window, for example.
</center></p><p>
Centering is now turned off. The default text alignment is
to the left of your screen.
You can change the size and color of text <font
color="blue"> by using the &lt;font&gt;</font>
<font color="purple">element.</font>
</body>
</html>
"7"
```

The small checkbox graphic has been created with Windows ’ Paint program. The actual text displayed in your browser is larger than this, but the output image has been reduced in size (perhaps to the extent of not being readable) tofit on the page. Also, because of the line feeds imposed on the text of this code example by the page width, the output looks a little different from what you might expect. So, you need to try this document on your own browser.


Here is a Level 1 Heading

Here is a Level 2 Heading

```
        Here is some preformatted text that has
        been created with the pre element. Note that it retains the
paragraph tab
included
in the original       document. Also, it does not "collapse" line feeds
and
                white         spaces. Often, it is easier to use preformatted text than it
is to use markup to get the same effect. Note, however, that the default rendering of
preformatted text is to use a monospaced Courier font. This is often a good choice for
displaying code in an HTML document, but perhaps not a good choice for other kinds of text content.
```

Here, a small graphic (the check box) has been inserted into the document using the "img" element. This text is outside the preformatted region, so the default font is different. If you look at the original document, you can also see that white spaces and line feeds are now collapsed.

Note too, that the text is now centered. The way the text is displayed will depend on how you have the display window set in your browser. It may change when you go from full screen to a window, for example.

Centering is now turned off. The default text alignment is to the left of your screen. You can change the size and color of text by using the <font> element.

Document 2.1 answers an interesting question: How can HTML display characters that already have a special meaning in the HTML language, or which do not appear on the keyboard? The angle brackets (< and >) are two such characters because they are part of HTML tags. They can be displayed with the & l t ; and & g t ; escape sequences (for the“less than” and“greater than” symbols from mathematics). There are many standardized escape sequences for special symbols. A list of some of them is given in Appendix 2.

2.2 HTML Syntax and Style

A general characteristic of programming languages is that they have very strict syntax rules. HTML is different in that regard, as it is not highly standardized. The positive spin on this situation is to call HTML an“open standard,” which means that self-described bearers of the standard can treat the language as they seefit, subject only to usefulness and market acceptance. HTML has an established syntax, but it is very forgiving about how that syntax is used. For example, when a browser encounters HTML code that it does not understand, typically it just ignores it rather than crashing, as a“real” program would do.

Fortunately, market forces—the desire to have as many people as possible accept your browser’s interpretation of HTML documents—have forced uniformity on a large subset of HTML. This text will adopt some HTML style conventions and syntax that will be as platform-independent as possible. Although these“rules” might seem troublesome if you are not used to writing stylistically consistent HTML documents, they should actually help beginners by providing a more stable and predictable working environment. The only thing worse than having syntax and style rules is having rules that nobody follows.

Here are some style rules that will be used in this text. Under the circumstances of HTML, they are more accurately referred to as“guidelines.” Some of them will make more sense later on, as you create more complicated documents.

1. Spell the names of HTML elements in lowercase letters.

Unlike some other languages, the HTML language is not sensitive to case. Thus, <htm l >, <HTML>, and <hTmL> are equivalent. However, the XHTML standard requires element names to be spelled with lowercase letters. In the earlier days of HTML, many programmers adopted the style of using uppercase letters for element names because they stood out in a document. You will often still see this style in Web documents. Nonetheless, this book will consistently use lowercase letters for element names.

2.Use the pre element to enforce text layout whenever it is reasonable to use a monospaced font (such as Couri er).

HTML always collapses multiple “white space” characters—spaces, tabs, and line breaks—into a single space when text is displayed. The easiest way to retain white space characters is to use the pre element. Other approaches may be needed if proportional fonts are required. Also, tabbed text may still not line up, as different browsers have different default settings for tabs.

3. Nest elements properly.

Improperly nested elements can cause interpretation problems for your browser. Even when browsers do not complain about improperly nested elements, HTML is easier to learn, read, and edit when these guidelines are followed.

Recall this markup in Document 2.1: Here i s some <b><i >pref ormatted t e x t < / i > < / b >
If you write thisas: Here is some
<b>

< i > *…{text}*
< / i >
</b>

it is easy to see that the i element is properly nested inside the b element. If this is changed to
<b><i>*…{text}* < / b > < / i >
with the order of the </b> and < / i > tags reversed, your browser probably won't complain, but it is not good programming style.

4. Enclose the values of attributes in single or double quotes. In Document 2.1, b g c o l o r = " wh i t e " is an attribute of <body>.
Browsers generally will accept bgcol or= whi te, but the XHTML standard enforces the use of quoted attribute values. This text will be consistent about using double quotes unless attribute values appear inside a string that is already surrounded with double quotes. Then attribute values will be singlequoted.

5. Use comments to explain what you are doing.

Good programmers always provide comments in their code for their own benefit as well as for the benefit of others who might use their code. This applies to HTML documents as well as to"real" programs. The syntax for HTML comments, which can appear anywhere in a document, including across multiple lines, is:

<!— *{Insert text here …}* – –>
2.3 Creating and Organizing a Website

Creating, organizing, and maintaining a website is a major topic, a thorough investigation of which would go far beyond the reach of this text. There is an entire industry devoted to hosting and creating websites, including helping a user obtain a domain name, providing storage space, developing content, and tracking access. For the purposes of this book, the goal is extremely simple: create web pages to solve some computational problems.

The first step toward creating a website is establishing its location. In an academic environment, a college, university, or department computer may provide space for web pages. A URL might look something like this:

h t t p : / / www. m y u n i v e r si t y . e d u / * u se r n a m e

where the “~” symbol indicates a directory where web pages are stored. Together with a user name, this URL directs a browser to the home Web directory for that user. As noted in Chap. 1 , HTML documents are not automatically Internet-accessible, and for the purposes of this book your web pages may be accessible only locally on your own computer.

In this home directory there should be at least one file, called i n dex .htm (or i n dex .htm l). UNIX systems favor the . h t m l extension, but Windows users may prefer the three-character .htm extension because it is more consistent with Windowsfile extension conventions. This is thefile that will be opened automatically in response to entering the above URL. That is, the i nd ex .htmfile is the“home page” for the Website. This home pagefile could be named something different, but then its name would have to be added to the URL:

http://www.m yuni v ersi ty.edu/ *usernam e/Hom ePage.htm

An i nd ex .htmfile can contain both its own content as well as links to other content (hyperlinks), including other pages on the user’s Website and to external URLs. Here are four important kinds of links:

1. Links to other sites on the World Wide Web.
This is the essential tool for globally linking web pages. Syntax:<a href="
{URL of web page} ">
{description of linked web page} </a>

The URL may refer to a completely different Website, or it may be a link to local documents in the current folder or a subfolder within that folder. 2. Links to images.
The img element is used to load images for display or to use as a page background.
Syntax:<img src="*{URL plus image name}* " align="…"
height="…" width="…" />

The image may exist locally or it may be at a different Website. The a l i g n , hei g ht, and wi d t h attributes, which can be used to position and size an image, are optional. However, for high-resolution images, it is almost always necessary to specify the height and width as a percentage of the full page or as a number of pixels in order to reduce the image to a manageable size in the context of the rest of the page. Actually resizing the image with a photo editing program will solve this problem.

You can also make a“clickable image” to direct the user to anotherlink:
Syntax:<a href="*{URL of web page}* ">
<img src="*{URL plus image name}* " al i gn="…"
hei ght="…" width="…" / > < / a>
3. Links to email addresses.
An email link is an essential feature that allows users to communicate with the author of a web page.
Syntax:<a href ="m ai l to:*{ e mai l address}* ">
{description of recipient} </a>

The *{description of recipient}* is usually the email address or the recipient’s name. The actual sending of an email will be handled by the default mailer on the sender’s computer.

4. Internal links within a document.
Within a large document, it is often convenient to be able to move from place to place within the document, using internal links.

Syntax: <a href="#*{internal link name}* ">
{description of target position} </a>
…
<a name="*{internal link name}* ">*{target text}* </a>

The “#” symbol is required when specifying the value of the h r e f attribute, to differentiate this internal link from a link to another (external) document.

The careless speci fication of linked documents can make websites very difficult to maintain and modify. As noted above, every website should have a“home” directory containing an i ndex .htmfile. In order to make a site easy to transport from one computer to another, all other content should be contained either in the home directory or in folders created within that directory. References to folders that are not related in this way should be avoided, as they will typically need to be renamed if the site is moved to a different computer. Although it is allowed as a matter of syntax to give a complete (absolute) URL for a local web page, this should be avoided in favor of a reference relative to the current folder.
This matter is important enough to warrant a complete example. Document 2.2a–c shows a simple website with a home folder on a Windows desktop called home and two subfolders within the home folder named homework and personal. Each subfolder contains a single HTML document, homework.htm in homework and resume.htm in personal.

Document 2.2a (i ndex.htm)

```
< h t m l >
< h e a d >
< t i t l e >My P a g e < / t i t l e >
< / h e a d >
<body>
<! –– These absolute lin k s are a bad id e a! – –>
Here are l i n k s to
< a h r e f = " C : / D o c u m e n t s and S e t t i n g s / D a v i d / d e s k t o p
/ /Book/hom ework.htm "> hom ework</a> and
< a h r e f = " C : / D o c u m e n t s and

S e t t i n g s /
D a v i d / d e s k t o p / B o o k / r e s u m e . h t
m " >

p e r so n a l docum ents.</a>
< / b o d y >
< / h t m l >
```

Document 2.2b (resume.htm)

```
< h t m l >
< h e a d >
< t i t l e > R e s u m e < / t i t l e >
< / h e a d >
<body>
Here i s myr&eacute;sum &eacute;. < / b o d y >
< / h t m l >
```

Document 2.2c (homework.htm)

```
< h t m l > <h e a d >
< t i t l e > H o m e w o r k < / t i t l e > < / h e a d >
<body>
Here are myhomework probl em s. < / b o d y >
< / h t m l >
```

Note that Document 2.2a uses forward slashes to separate the directories andfile names. This is consistent with UNIX syntax, but Windows/DOS systems use backward slashes. Forward slashes are the HTML standard, and they should always be used even though backward slashes may also work. Another point of interest is that UNIX directory paths andfilenames are case-sensitive, but Windows paths andfilenames are not. This could cause problems if you develop a web page on a Windows computer and then move it to a UNIX-based system. As a matter of style, you should be consistent about case in directory andfile names even when it appears not to matter. Note how the“é”s in résumé, which are not keyboard characters, are produced by using the escape sequence &eacute;—see Appendix 2.

In Document 2.2a, the absolute references to a folder on a particular Windows computer desktop are a bad idea because this reference will need to be changed if the i nd ex .htmfile is moved to a different place on the same computer, or to a different computer—for example, to a University department computer with a different directory/folder structure. Document 2.2d shows the preferred solution. Now the paths to homework.htm and resume.htm are given relative to the home folder, wherever the i n d e x 2 . htmfile resides. (Remember that thisfile, no longer named i ndex.htm, will not be recognized as a default home page, but you can rename it.) This document assumes that folders homework and p e r so nal exist in the home folder. This relative URL should work without modification when the website is moved to a different computer. If the website is moved, only a single reference, to the i ndex 2.htmfile, needs to be changed.

Document 2.2d (index2.htm, a new version of i ndex .htm)

```
< h t m l >
< h e a d >
< t i t l e >My P a g e < / t i t l e >
< / h e a d >
<body>
<! — Use these r e l a t i v e lin k s instead! —>
Here are l i n k s to
<a
h r e f = " h o m e wo r k / h o m e wo r k . h t m " >
homework</a> and
<a h r e f = " p e r s o n a l / r e s u m e . h t m " >
p e r so n a l docum ents.</a>
< / b o d y >
< / h t m l >
```

Paying proper attention to using relative URLs from the very beginning when designing a website will save a lot of time in the future!

2.4 Selecting and Using Colors

As previously noted, several attributes, such as bgcol or, are used to set colors of text or backgrounds. Colors may be identified by name or by a sixcharacter hexadecimal numeric code that, historically, specified the strength of the signal emitted from the red, green, and blue electron“guns” that excited the corresponding phosphors on a cathode ray tube color screen. This convention has been retained even though other monitor display technologies are used now. The hex code[2] is in the format #*RRGGBB* where each color value can range from 00 (turned off) to FF (maximum intensity).

There are many color names in use on the Web, but only 16 are completely standardized, representing the 16 colors recognized by the Windows VGA color palette. These colors are listed in Table 2.1 . The problem with additional color names is that there is no enforced standard for how browsers should interpret them. Two examples: magenta probably should be, but doesn’t have to be, the same as fuchsia; ivory is a nonstandard color that should be rendered as a yellowish off-white. The colors in Table 2.1 are standardized in the sense that all browsers should associate these 16 names with the same hexadecimal code. Of course, variations can still occur because monitors themselves will respond somewhat differently to the same name or hex code; blue on my computer monitor may look somewhat different than blue on your monitor.

Note that the standardized colors use a limited range of hex codes. With the exception of silver (nothing more than a lighter gray), the RGB gun colors are either off (00), on (FF), or halfway on (80).

What should you do about choosing colors? Favor standardized colors, and if you wish to make an exception, try it in as many browser environments as possible. Be careful to choose background and text colors so that text will always be visible against its background. The safest approach for

Table 2.1 16 standard HTML color names and hex codes

Color Name	Hexadecimal Code
aqua	#00FFFF
black	#000000
blue	#0000FF
fuchsia	#FF00FF
gray	#808080
green	#008000
lime	#00FF00
maroon	#800000
navy	#000080
olive	#808000
purple	#800080
red	#FF0000
silver	#C0C0C0
teal	#008080
white	#FFFFFF

yellow

#FFFF00

[2] Hex code = hexadecimal code, a base-16 numbering system using integers 0–9 and letters A–F to represent values ten tofifteen.

setting colors in the body element is to specify both background and text colors. This will ensure that default colors set in a user’s browser will not result in unreadable text.

If you ’re not sure whether a color name is supported and what it looks like on your monitor, you have nothing to lose by trying it. If you set b g c o l o r = " l i g h t b l u e ", you will either like the result or not. If a color name isn’t recognized by your browser, the result will be unpredictable, but not catastrophic. There are (of course!) numerous websites that will help you work with colors, including getting the desired result with hex codes.

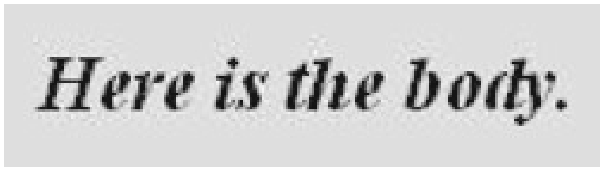
2.5 Using Cascading Style Sheets

As you create more web pages, you may wish to impose a consistent look for all your pages, or for groups of related pages. It is tedious to insert elements for all the characteristics you may wish to replicate—font size, font color, background color, etc. Style sheets make it much easier to replicate layout information in multiple sheets. A complete discussion of style sheets is far beyond the scope of this book, as there are many different kinds of style sheets, many ways to make use of them, and many browser-specific nuances. This book will use cascading style sheets (CSS), which are widely accepted as a default kind of style sheet, but will present only a *small* subset of all the possibilities! By way of introduction, Document 2.3 shows how to use a s t y l e element to establish the default appearance of the body of an HTML document.

Document 2.3 (st y l e 1.h tm)

```
<html>
<head>
<title> Style Sheets</title>
<style title= David s default type= text/css >

body.bright {background: red; font: 16 pt serif; color: blue; font-
style: italic; font-weight: bold} </style>
</head>
<body class= bright >
Here is the body.
```


Here is the body.

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

The s t y l e element has an optional t i t l e attribute and a t y pe attribute set equal to " t e x t / c s s ", where the css stands for cascading style sheet. Inside the s t y l e element, dot notation is used to assign a class name, b r i g h t, to the body element: body . b r i g h t. Inside curly brackets attributes are assigned values, with each attribute and its value being separated by a semicolon. Then, the <body> tag assigns the class name b r i g h t as the value of the c l a s s attribute. As a result, the document background color is red, with the font set to a blue, bold, italicized 16-point serif font.

Any HTML tag that encloses content can be assigned a class value defined in a style element. For this simple example, with styles applying only to a single body element, the class name is optional. With no class name and no c l a s s attribute in <body>, the style rules will automatically be applied to the entire HTML document.

In general, several different style rules can apply to the same HTML element. For example, several style rules could be established for paragraphs (<p>…</p>), each of which would have its own class name.

In summary, style specifications follow a hierarchy:
s t y l e element ! other HTML elements*{.class name}* ! properties ! value(s)
where the *{.class name}* is optional.

How did CSS get that name? Because the properties set for an element cascade down, or are“inherited,” by other elements contained within that element unless those elements are assigned their own style properties. So, for example, properties set for the body element are inherited by the p and h1 elements, because these are contained within the body element. Properties set for the head element are inherited by content appearing in the t i t l e element.

CSS can be used to modify the appearance of any HTML element that encloses content. Here are some properties that can be specified in style sheets.

Background properties
background-col or

When used in a body element, backgroun d-c ol or sets the background color for an entire document. It can also be used to highlight a paragraph, for example, when used with a p element.

background-image

This property is used with a URL to select an image file (gif or jpeg) that will appear as a background. Typically, this is used with a body element, but it can also be used with other elements, such as p. For other background properties that can be used to control the appearance of a background image, consult an HTML reference text.

background
This allows you to set all background properties in a single rule.

Color property
The c o l o r property sets the default color for text, using the descrip tions discussed in Sect. 2.4 .

Font properties
f o n t - f a m i l y

Font support is not completely standardized. However, browsers that support style sheets should support at least the generic font families given in Table 2.2 .

Example: f o n t - f a m i l y : A r i a l , s a n s - s e r i f ;
f o n t - s i z e

This property allows you to set the actual or relative size of text. You can use relative values, such as l a r g e , s m a l l , l a r g e r , sm a l l e r (relative to a default size); a percentage, such as 200%of the default size; or an actual point size such as16pt. Some sources advise against using absolute point sizes because a point size that is perfectly readable on one system might be uncomfortably small on another. But, for our purposes, specifying the point size is a reasonable choice.

Example: f o n t - s i z e : 2 4 p t ;
f o n t - s t y l e

This property allows you to specify n o r m a l , i t a l i c , or o b l i q u e fonts. Example: f o n t - s t y l e : i t a l i c ;
f o n t - w e i g h t

This propertyallows you to select the font weight. You can use values in the range from 100 (extra light) to 900 (extra bold), or words: e x t r a - l i g h t , l i g h t , d e m i - l i g h t , medium, d em i - b o l d, bol d, and e x t r a - b o l d. Some choices may not have a noticeable effect on some fonts in some browsers.

Example: f o n t - w e i g h t : 900;
f o n t
This property allows you to set all font properties with one style rule.
Table 2.2 Some font families **Generic Name**

`monospace` sans-serif serif

Example
Script MT Bold
Courier
Arial

Times New Roman

Example: f o n t : i t a l i c 18pt H e l v e t i c a , s a n s - s e r i f ;

How will your browser interpret a font name? For the generic name s e r i f, it will pick the primary serif font it supports—probably Times or Times Roman. Browsers will probably also recognize specific font names such as Times or Helvetica (a sans-seriffont). If you specify a font name not supported by your browser, it will simply ignore your choice and use its default font for text. It is possible to list several fonts, in which case your browser will select thefirst one it supports. For example, consider this style specification:

f o n t - f a m i l y : A r i a l , H e l v e t i c a , s a n s - s e r i f ;

Your browser will use an Arial font if it supports that, Helvetica if it doesn ’t support Arial but does support Helvetica, or,finally, whatever sans-serif font it does support by default. By giving your browser choices, with the generic serif or non-serif name as the last choice, you can be reasonably sure that text will be displayed approximately as you wish it to appear.

Text properties
Of the many text properties, here are just three that may be useful.

t e x t -align
This is used in block elements such as p. It is similar in effect to the HTML a l i g n attribute. The choices are l e f t , r i g h t , center, and j u s t i f y. With large font sizes, j u s t i f y may produce odd-looking results.

Example: t e x t - a l i g n : c e n t e r ;
t e x t -indent

Recall that paragraphs created with the p element do not indent the first word in the paragraph. (HTML inserts a blank line and left-justifies the text.) This property allows you to set indentation using typesetting notation or actual measurements. An actual English or metric measurement—inches inches (in), millimeters (mm), or centimeters (cm)—may be easiest and will always give predictable results.

Example: t e x t - i n d e n t : 0 . 5 i n ;
whi te-space

The value of this property is that you can prevent spaces from being ignored. (Remember that the default HTML behavior is to collapse multiple spaces and other non-printable characters into a single blank space.) Some older browsers may not support this property. You can use the HTML pre element by itself, instead, but this causes text to be displayed in a monospaced font such as Courier. The example given here retains white space regardless of the typeface being used.

Example: wh i t e - sp a c e : p r e ;

Styles aren ’t restricted just to the body element. For example, paragraphs (<p>…</p>) and headings (<h*n* >…</h*n* >) can also have styles associated with them. You can also set styles in selected portions of text, using the span element, and in blocks of text using the d i v element.

`<div> … </div>`
Attributes:`align` ,`style`

`<span> … </span>`
Attributes : `align` ,`style`
Values for align : `"left"` (default),`"right"` , `"center"`

You can create style sheets as separate files and then use them whenever you wish to use a particular style on a web page. This makes it easy to impose a uniform appearance on multiple web pages. Document 2.4a and 2.4b show a simple example.

Document 2.4a (body.css)

body { b a c k g r o u n d : s i l v e r ; c o l o r : w h i t e ; f o n t : 2 4 p t Times} h1 { c o l o r : r e d ; f o n t : 1 8 p t I m p a c t ; }
h2 { c o l o r : b l u e ; f o n t : 1 6 p t C o u r i e r ; }

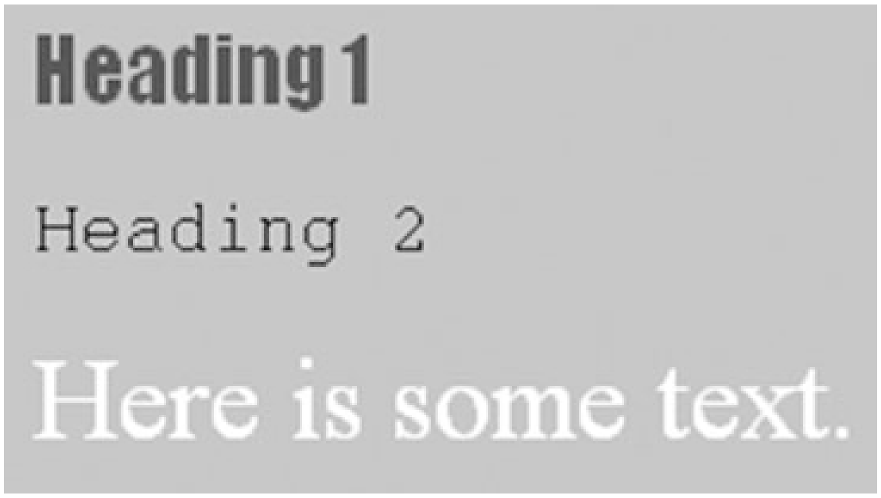
Document 2.4b (st y l e 2.h tm)

< h t m l >
< h e a d >
< t i t l e > S t y l e Sheet E x a m p l e < / t i t l e > < l i n k h r e f = " b o d y . c s s "

r e l = " s t y l e s h e e t " t y p e = " t e x t / c s s " />
< / h e a d >
<body>

```
< h 1 > Heading 1 < / h 1 >
< h 2 > Heading 2 < / h 2 >

Here i s some t e x t .
< / b o d y >
< / h t m l >
```



This example shows how to create a file, body.css, containing style elements that can be applied to any document by using the l i n k element, as in Document 2.4b. The . c s s extension is standard, but not required. (You could use . t x t, for example, although there is no reason to confuse matters by doing that.) This example is very simple, but the concept is powerful because it makes it easy to create a standard style for all your documents which can be invoked with the l i n k element. The Impact font chosen for h1 headings may not be supported by all browsers. If not, a default font will be used in its place.

The attributes of l i n k include href, which contains the URL of the style sheet file, the r e l = " s t y l e s h e e t " (relationship) attribute, which describes how to use thefile (as a style sheet), and the type, which should be " t e x t / c s s ", just as it would be defined if you created a s t y l e element directly in the head element. In this example, body.css is in the same folder as st yl e 2. h tm. If you keep your style sheets in a separate folder, you will of course need to reference that folder.

This discussion has barely scratched the surface of style sheets. Style sheets can make your web pages more visually appealing and can greatly simplify your work on large Web projects. Some developers advocate replacing *all* individual formatting elements, such as f o n t and its attributes, with style sheet specifications. Despite the fact that the use of individual formatting elements is“deprecated” in favor of using CSS, there is little likelihood that support for individual elements will disappear from browsers in the foreseeable future. For the kinds of applications discussed in this book, cascading style sheets may sometimes be convenient, but they are not required.

2.6 Another Example

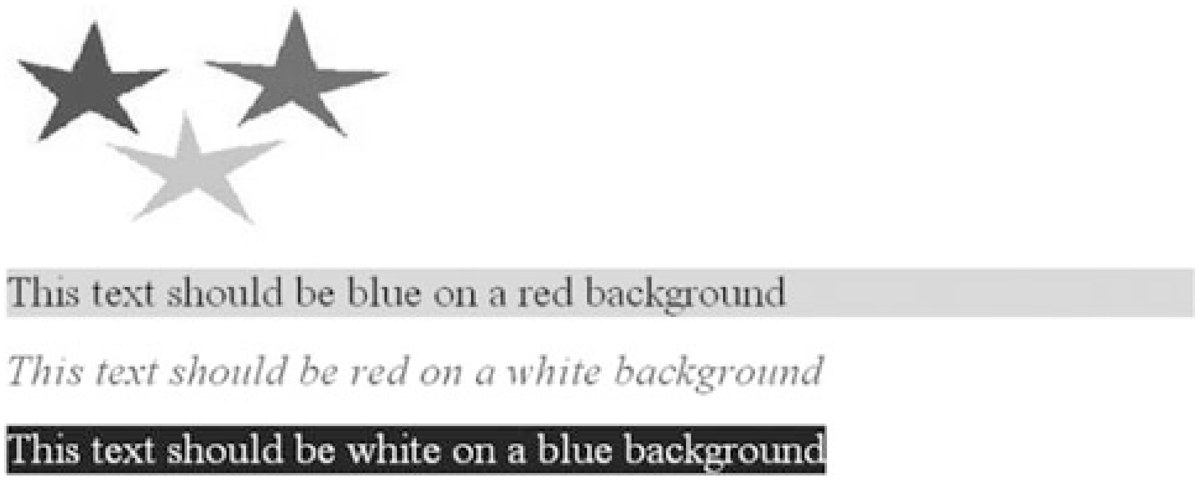
Documents 2.5a and 2.5b show how to use a style sheetfile to specify different background and text colors for different sections of text. Document 2.5a (rwb.css)

```
p . r e d { b a c k g r o u n d : r e d ; c o l o r : b l u e ; f o n t : 2 0 p t Times}
d i v . w h i t e { b a c k g r o u n d : w h i t e ; c o l o r : r e d ; f o n t : 2 0 p t
Times} s p a n . b l u e
{ b a c k g r o u n d : b l u e ; c o l o r : w h i t e ; f o n t : 2 0pt Times}
```

Document 2.5b (rwb.htm)

```
< h t m l >
< h e a d >
< t i t l e >A Red, W h i t e, and Bl ue D o c u m e n t < / t i t l e >
< l i n k h r e f = " r w b . c s s " r e l = " s t y l e s h e e t " t y p e = " t e x t /
c s s " />
< / h e a d >
<body>
<i m g s r c = " s t a r s . j p g " h e i g h t = " 1 5 0 " wi d t h = " 2 5 0 " /> <p
c l a s s = " r e d " >
T h i s t e x t shoul d be b l u e on a red background.
< / p > < p > < d i v c l a s s = " w h i t e " s t y l e = " f o n t - s t y l e : i t a l
i c ; " > T h i s t e x t shoul d be red on a wh i t e background. < / d i v > < / p
>
< p > < sp a n c l a s s = " b l u e " > T h i s t e x t shoul d be wh i t e on a b l
u e b a c k g r o u n d . </sp a n >
```

```
< / p >
< / b o d y >
< / h t m l >
```



This text should be blue on a red background

This text should be red on a white background

This text should be white on a blue background

The stars, which are supposed to be red, silver, and blue, have been drawn using Windows’ Paint program.

HTML Tables, Forms, Lists, and Frames 3

This chapter shows how to use HTML tables, forms, lists, and frames to organize documents and provide a framework for interfacing with PHP applications.

1. The t a b l e Element 1. Table Formatting

HTML table and form elements are the two most important ways to organize the content of a web page. Forms are critical because they provide the interface with PHP. Sometimes it is helpful to organize information in a form through the use of one or more tables. With that approach in mind,first consider tables.

Because HTML ignores text formatting, such as white space and line feeds (the Enter key), it can be difficult to control the placement of content on a web page. The addition of images only compounds this problem. An easy way to gain some control is to create a table, using the table element. Then the relative locations of text and graphics can be established by entering them into cells of the table. Within the start and end tags, <ta bl e >…< / tabl e>, rows and cells are defined with thetr (“table row”) and td (“table data”) elements. These elements are nested as follows:

```
< t a b l e >
< t r >
<td>... </td> {as many columns as you need …} …

< / t r >
{as many rows as you need …}
…

< / t a b l e >
```

The < t r >…< / t r > tags define the rows and the <td>…</td> tags define cells in columns within those rows. You can define as many rows and columns as you need to organize information in a familiar spreadsheet-like row-and-column format. Document 3.1 shows how to use a table to organize and display some results from residential radon testing.

Document 3.1 (radonTable.htm)

```
< h t m l >
<head>
< t i t l e> Radon T a b l e< / t i t l e >
< / h ea d >
<body>
<h1> Resul ts o f radon t e s t i n g< / h 1 >
<p>
The t a b l e below shows some radon l e v e l s measured i n r e si d e n c e
s.< b r / > For v a l u es g r e a t e r than o r equal t o 4 p C i / L , a c t i o n
shoul d be t a k e n< b r / > t o reduce t h e c o n c e n t r a t i o n o f radon g a
s. For v al ues g r e a t e r than or< b r / > equal to 3 p C i / L , r e t e s t i n g is
recommended.
< / p >
< t a b l e >

< t r b g c o l o r ="s i l v e r " >
< t d > L o c a t i o n< / t d >< t d > V a l u e , p C i / L< / t d >
```

```
<t d> Com m ents</ td></tr >
< t r >
<t d> DB's house, b a se m e nt< / t d> <t d > 1 5.6</ t d>
< t d b g c o l o r ="p i n k " > A c t i o n shoul d be t a k e n ! < / t d > < / t r >
< t r >
< t d > I D ' s house, 2nd f l o o r b e d r o o m</ t d ><t d > 3 .7< / t d> < t d b g c o l
o r=" y el l ow " > Sho ul d be r e t e s t e d .< / t d > < / t r >
< t r >
< t d > F J ' s house, 1 s t f l o o r l i v i n g room</ t d><t d> 0 . 9< / t d >
< t d b g c o l o r ="l i g h t g r een " > No a c t i o n r e q u i r e d .< / t d > < / t r >
< t r >
< t d > MB's house, 2nd f l o o r b e d r o om< / t d> <t d > 2 .9< / t d> < t d b g c o l
o r ="l i g h t g r een " > No a c t i o n r e q u i r e d .< / t d > < / t r > < / t a b l e >
< / b o d y >
< / h t m l >
```

In a color rendition of this page, thefirst value will have a pink background, the second a yellow background, and the others a light green background.

Results of radon testing

The table below shows some radon levels measured in residences. For values greater than or equal to 4 pCi/L, action should be taken to reduce the concentration of radon gas. For values greater than or equal to 3 pCi/L, retesting is recommended.

Location Value, pCi/L Comments

DB's house, basement 15.6 ID' s house, 2nd floor bedroom 3.7 FJ's house, 1st floor living room 0.9 MB's house, 2nd floor bedroom 2.9 Action should be taken! Should be retested. No action required. No action required.

The syntax for tables includes several possibilities in addition to tr and td for customizing appearance. These include the c a p t i o n element, which associates a caption with the table, and the th element, which is used to create a“header” row in a table by automatically displaying text in bold font. (Theth element can be used anywhere in a table in place of td.) The c a pti o n, td, th, andtr elements are used only inside the start and end tags of a t a b l e element: <tabl e>…</table>. With these elements, a more comprehensive table layout looks like this:

```
< t a b l e >
<capti on>…< / c a p t i o n >
```

```
< t r >
<!— Use o f t h i n p l a c e o f t d i s o p t i o n a l . —>

<th> …< / t h >
…
< / t r >
< t r >
<td> …< / t d >

…
< / t r >
…
< / t a b l e >
```

Here is a summary of some table-related elements and their attributes. All the elements except t a b l e itself should appear only inside a t a b l e element.

`<caption> … </caption>`

Displays the specified text as a caption for a table. Earlier versions of HTML support only`"top"` (the default value) or`"bottom"` for the value of the`align` attribute. Some browsers may allow`"center"` as a value

for`align` , which might often be the alignment of choice for a table caption.
Attributes :`align`

`<table> … </table>`
Contains table-related and other elements.
Attributes : `border` , `bordercolor` , `cellpadding` , `cellspacing` , `width`

`<tbody> … </tbody>`

Groups rows within the body of a table so each group can be given different attributes and styles.
Attributes:`align` , `char` , `charoff` ,`valign`

`<td> … </td>`

Defines data cells in the rows of a table. Does not contain other tablerelated elements.
Attributes : `align` , `bgcolor` , `char` , `charoff` , `colspan` , `nowrap` , `rowspan` ,`width`

`<th> … </th>`

The `th` element works just like the`td` element except it automatically displays text in bold font, serving as headings for table columns. Does not contain other elements.
Attributes : `align` , `bgcolor` , `char` , `charoff` , `colspan` , `nowrap` , `rowspan` , `valign` ,`width`

`<tr> … </tr>`
Defines rows in a table. Contains `td` or`th` elements.
Attributes : `align` , `bgcolor` ,`valign`

Description of attributes:
`align = "…"`
Values : `"left"` , `"right"` , or`"center"`

Aligns text horizontally. When `align` is specified in a`tr` element, its value will be overridden if it is specified again within a`td` element in that row.

`bgcolor = "…"`
Values : color names or hexadecimal values `"#` *`RRGGBB`* `"`

Sets the background color for a cell or row. When `bgcolor` is specified in a`tr` element, its value will be overridden if it is specified again within a `td` element in that row.

`border = "…"`
Values : an integer number of pixels

Adds a border to the table and its cells. A value is optional. If it is included, a colored (or gray, by default) border is added around the outer boundary of the table.

`bordercolor = "…"`
Values : color names or hexadecimal values`"#` *`RRGGBB`* `"`
Sets the color of a table border.
`cellpadding = "…"`
Values : an integer number of pixels
Defines vertical spacing between cells in a table.
`cellspacing = "…"`
Values : an integer number of pixels
Defines horizontal spacing between cells in a table.
`colspan = "…"`
Values : an integer
Defines how many columns a cell will span.
`nowrap`
Prevents text from being automatically wrapped within a cell. It does not have a value.
`rowspan = "…"`
Values : an integer
Defines how many rows a cell will span.
`valign = "…"`
Values : `"top"` , `"middle"` , or`"bottom"`

Aligns text vertically. When `valign` is specified in a`tr` element, its value will be overridden if it is specified again within a`td` element in that row.

`width = "…"`
Values : a number or a percentage
Specifies table or cell width in pixels (`width="140"`) or as a percentage of the window or table header width (`width="80%"`).

The attributes associated with these tags all have default values or are just ignored if you don't use them. You can create a table without using any attributes and then add attributes as needed. When creating tables, it is often useful to include the b or de r attribute, which can be removed when you are happy with how the table looks. In Document 3.1, the only specified attribute is the background color in some cells. An easy way to familiarize yourself with the effects of specifying table attributes and their values is to experiment with Document 3.1.

Document 3.1 is not very useful except for showing how to create tables. The radon values and the colors associated with them are all“hard coded” values. There is no provision for changing the radon values, and even if you could do that, you couldn’t use HTML to change the background colors in response to the radon values. However, if the radon values and perhaps the site names are entered into i n p u t elements, PHP can be used to generate a table with background colors calculated based on the radon values entered. (See Document 4.1.)

3.1.2 Subdividing Tables into Sections

The tbody element allows a table to be divided into two or more groups of rows. Each group of rows enclosed by a <tbody>…</tbody> tag can have its own attributes and can have different pre-defined c l a s s attribute values. Document 3.2 shows a simple example in which rows in a table are grouped by background color.

Document 3.2 (tbody.htm)

```
< h t m l >
< h e a d >
< t i t l e > U s i n g t h e tbody e l e m e n t < / t i t l e >
< s t y l e >
t h { b a c k g r o u n d - c o l o r : b l a c k ;
c o l o r : w h i t e ; } t b o d y . c o l d { t e x t
a l i g n : c e n t e r ;
f o n t - w e i g h t : b o l d ; b a c k g r o u n d - c o l o r : g r a y ; } t b o d y .
c o o l { t e x t - a l i g n : c e n t e r ;
f o n t - w e i g h t : b o l d ; b a c k g r o u n d
c o l o r : s i l v e r ; } t b o d y . h o t { t e x t
a l i g n : c e n t e r ;
f o n t - w e i g h t :bold; b a c k g r o u n d - c o l o r : i v o r y ; } < / s t y l e
>
< / h e a d >
<body>
```

```
< t a b l e border>
< t r > < t h > M o n th </ th > <t h > Av e r a g e < b r />Temperature < b r
/>& d e g ; F < / t d > < / t r >
< t b o d y c l a s s = " c o l d " >
< t r > < t d
> J a n u a r y < / t d > < t d > 3 0 . 4 < / t d > < / t r >
< t r > < t d > F e b r u a r y < / t d > < t d > 3 3 . 0 < / t d > < / t r
>
< t r > < t d > M a r c h < / t d > < t d > 4 2 . 4 < / t d > < / t r >
< / t b o d y >
< t b o d y c l a s s = " c o o l " >
< t r > < t d > A p r i l < / t d > < t d > 5 2 . 4 < / t d > < / t r >
< t r > < t d > M a y < / t d > < t d > 6 2 . 9 < / t d > < / t r >
< / t b o d y >
< t b o d y c l a s s = " h o t " >
< t r > < t d > J u n e < / t d > < t d > 7 1 . 8 < / t d > < / t r >
< t r > < t d > J u l y < / t d > < t d > 7 6 . 7 < / t d > < / t r >
< t r > < t d > A u g u s t < / t d > < t d > 7 5 . 5 < / t d > < / t r >
< / t b o d y >
< t b o d y c l a s s = " c o o l " >
< t r > < t d > S e p t e m b e r < / t d > < t d > 6 8 . 2 < / t d > < / t
r>
< t r > < t d > O c t o b e r < / t d > < t d > 5 6 . 4 < / t d > < / t r >
< / t b o d y >
< t b o d y c l a s s = " c o l d " >
< t r > < t d > N o v e m b e r < / t d > < t d > 4 6 . 4 < / t d > < / t r
>
< t r > < t d > D e c e m b e r < / t d > < t d > 3 5 . 8 < / t d > < / t r
>
< / b o d y >
< / h t m l >
```

Average Month Temperature °F **January 30.4**

January –March and November–Decem
ber use the “cold” class, April–May and
September–October use “cool,” and June–
August use“hot.” Each class has a different

background color. (For this grayscale ren
dering of the output, gray, silver, and ivory
have been chosen instead of something more
colorful.)

Document 3.2 is another example of an
HTML document which displays content, but
which offers no way to interpret or manipu
late that content.

February 33.0
March 42.4
April 52.4
May 62.9
June 71.8
July 76.7
August 75.5
September 68.2
October 56.4
November 46.4
December 35.8

3.1.3 Merging Cells Across Rows and Columns

If you are familiar with creating tables in a word processing application, you know that it is easy to create more complicated table layouts by merging cells across rows and columns. You can also do this with HTML tables, using the col span and rowspan attributes. Document 3.3 shows a table that displays cloud names, altitudes, and whether they produce precipitation or not.

Document 3.3 (cloudType.htm)

```
< h t m l >
<head>
< t i t l e > C l o u d Type C h a r t< / t i t l e >
< / h ea d >
<body>
< t a b l e border="2">
< c a p t i o n > C l o u d Type C h a r t< / c a p t i o n >
< t r >
< t h a l i g n =" c e n t e r " > A l t i t u d e< / t h >
< t h col span="2"> Cl oud Nam e</ t h></ t r >

< t r > < t d a l i g n =" c e n t e r " rowspan="3"> Hi gh</ t d> < t d c o l s p a n =" 2 " >
C i r r u s< / t d > < / t r >
< t r > < t d c o l sp a n =" 2 " > C i r r o c u m u l u s< / t d> < / t r > < t r > < t d c o l s
p a n =" 2 " > C i r r o s t r a t u s< / t d > < / t r > < / t r >

< t r > < t d a l i g n =" c e n t e r " rowspan="2"> Mi ddl e</ t d> < t d c o l sp a n =" 2 "
> A l t o c u m u l us< / t d > < / t r > < t r > < t d c o l s p a n =" 2 " > A l t o s t r a t u
s< / t d > < / t r > < / t r >

< t r > < t d a l i g n =" c e n t e r " rowspan="5"> Low</td> <t d> Cum ul us</ td>
< t d > n o n p r e c i p i t a t i n g< / t d > < / t r >

< t r > < t d > A l t o c u m u l u s< / t d >
< t d > n o n p r e c i p i t a t i n g< / t d > < / t r >
< t r > < t d > S t r a t o c u m u l u s< / t d >
< t d > n o n p r e c i p i t a t i n g< / t d > < / t r >
<t r ><t d> Cumulonimbus</ t d>
< t d a l i g n =" c e n t e r "
b g c o lor=" s i l v e r " >prec i p i t a t i ng< / t d > < / t r
>

< t r > < t d > N i m b o st r a t u s< / t d > < t d
a l i g n =" c e n t e r "
b g c o l o r =" s i l v e r " > p r e c i p i t a t i n g< / t d > < / t r > < / t r > < / t a b l e >
< / b o d y > < / h t m l >
```

It is more tedious to merge cells across rows in columns in an HTML table than it is in a word processor. You need to plan your table in advance, and even then pared for editing!
you should be presome trial-and-error

Altitude
High
Middle
Low

Cloud Type Chart **Cloud Name** Cirrus
Cirrocumulus
Cirrostratus
Altocumulus
Altostratus
Cumulus
Altocumulus
Stratocumulus
Cumulonimbus
Nimbostratus
nonprecipitating nonprecipitating nonprecipitating

precipitating precipitating

3.2 The f orm and i n p u t Elements

One of the most important applications of HTML documents is to provide the web page equivalent of a paper data input form. In some cases, a form just helps to organize user input to a web page. For this book, the purpose of a form is to facilitate interaction between an HTML document and PHP code for processing user input.

HTML forms are defined by the f orm element, using start and end tags: <form>…</f orm > tags. The attributes of the f orm element are:

`action = "…"`
Value : a programmer-supplied URL that identifies a processing script, PHP file name, or`mailto:` followed by an email address. For example, `action="mailto:my_mail@my_univ.edu"` .

`enctype="…"`
Value : `enctype="text/plain"` is the usual value, but it is not needed when`method="post"` is used with PHP applications.

`method = "…"`
Values : `"get"` ,`"post"`

Controls how data from a form is sent to the URL, PHP file, or email address identified in the`action` attribute. In this book, the`"post"` value is used because it is the easiest way to transmit form data in an easily usable format.

`name = "…"`
Value : a programmer-selected name that is used to identify the form. The`name` attribute is needed only if a document contains more than one form.

Forms contain one or more input fields identified by < i n p ut /> tags. Because the i n p u t element does not enclose content, it has no end tag, so it requires a closing slash for XHTML compliance. The most important attribute of i n p u t is its type. There are severalfield types that have welldefined default behaviors in HTML. The possible values are listed in Table 3.1 .

Table 3.1 Values for thei n p u t element’s t ype attribute
Field type
t ype = " b u t t o n "
Description

Provides a programmer-defined action to be associated with the field through the use of an event handler such as on c l i c k t ype = "c hec k box" Allows selection of one or more values from a set of possible

values
t ype = "hidd e n " Allows the definition of textfields that can be accessed by a PHP
script but are not displayed in a document
t ype = "passw or d"
t ype = " r a d i o "
Allows entry of character data but displays only asterisks Allows selection of one and only one value from a set of possible
values
t ype = " r e s e t "
t ype = " s u b m i t "
t ype = " t e x t "
Used to reset all formfields to their default values Processes form contents according to method and ac t i on Allows entry of character data

There is no field type specifically for numerical values. In some circumstances, this can be an issue, but as will be seen later, it presents no problem for sending values from an i n p u t element to a PHP application.

Here is a list of attributes for the input element. `checked`

Value : none

Applies to`type="radio"` and`type="checkbox"` only.

`maxlength="…"`

Value : Maximum number of characters that can be entered in the field. This value can be greater than the value given for the`size` attribute.

`name="…"`

Value : A programmer-supplied name for the field. The name should follow the variable-naming conventions for PHP in order to facilitate its use in PHP scripts.

`readonly`

Value : none

Prevents field values in`type="text"` or`text="password"` from being changed.

`size="…"`

Value : width of the displayed field, in characters.

`type="…"`
Values : See Table 3.1.

`value="…"`
Value : a programmer-supplied default value that will be displayed in the field. This value can be overridden by user input unless the`readonly` attribute is also specified.

The form element typically contains a combination of text and input fields. The text can be used to explain to the user of the form what kind of input is expected. Document 3.4 gives a simple example that uses several inputfield types:

Document 3.4 (l o c a t i o n . h tm)

```
< h t m l >
< h e a d >
< t i t l e > D a t a R e p o r ti n g S i t e I n f o r m a t i o n < / t i t l e > < / h
e a d >
<body>
< f o rm >

Please e n t e r your l a s t name:

< i n p u t t y p e =" t e x t " nam e="l ast _nam e"
si z e =" 2 0 " m ax length="20" / > < b r />
Please e n t e r your l a t i t u d e :
< i n p u t t y p e = " t e x t " n a m e = " l at" v a l u e = "4 0 "
s i z e = " 7 " m ax l ength="7" />
N < i n p u t t y p e = " r a d i o " name="NS" v al ue= "N" checked /> o r S
< i n p u t t y p e = " r a d i o " name="NS" v a l ue = "S " / > < b r />
Please e n t e r your l o n g i t u d e :
< i n p u t t y p e = " t e x t " nam e="lon" v a l u e = "7 5 "
s i z e = " 8 " m ax l ength="8" />
E < i n p u t t y p e = " r a d i o " name="EW" v a l u e=" E" / > or W
< i n p u t t y p e = " r a d i o " name="EW" v alue="W" checked / > < b r
/>
Please e n t e r your e l e v a t i o n :
```

```
< i n p u t t y p e = " t e x t " n a m e = "e l ev a ti o n " si z e = " 8 "
maxlength="8"
/ > m eters< br/>
Please i n d i c a t e t h e seasons d u r i n g which your s i t e r e p o r t s d a t
a : < b r />
W i n te r : < i n p u t t y p e = " ch ec k bo x "
nam e= "seasons" v a l ue = "W i n t e r " />
S p r i n g : < i n p u t t y p e = " c he c kb ox "
nam e= "seasons" v a l u e = " S p r i n g " />
Summer: < i n p u t t y p e = " ch e ck b ox "
nam e= "seasons" v alue="Summer" />
F a l l : < i n p u t t y p e = "c he c kb ox "
nam e= "seasons" v a l u e = " F a l l " />
< / f o r m >
< / b o d y >

< / h t m l >
```

Please enter your last name:
Please enter your latitude: 40 N
◉ or S
● Please enter your longitude: 75 E
● or W
◉ Please enter your elevation: meters
Please indicate the seasons during which your site reports data: Winter:
Spring: Summer: Fall:

Note that some of the text fields are blank because no default v al ue attribute has been specified. These require user input, and there is no way to establish ahead of time what this input might be. However, it may still be worthwhile in some cases to provide a default value if that would help the user to understand what is required. When the allowed input choices can be limited ahead of time by the creator of the document, it is appropriate to use

radio buttons and checkboxes. You can create as many different combinations of these kinds offield as your application needs.

Each group of r a d i o and checkbox buttons has its own uniquefield name and, within each group, each button should have its own value. In Document 3.4, there are two r a d i o button groups, named NS and EW. It is important to specify a value for each button, because the value of the checked button will be captured when the contents of the form are submitted to a PHP application. Default values for the radiofield can be specified by using the checked attribute. When you access the document, the button with the checked attribute will be“on.” You can change it by clicking on another of the buttons in the group.

The same basic rules apply to checkboxfields. You can have more than one group of checkboxes, each with its unique name. The only difference is that you can select as many boxes as you like within each group, rather than just one value with r a d i o fields.

3. Creating Pull-Down Lists

A common feature on web pages that use forms is a pull-down list. The s e l e c t and o p t i o n tags provide another way to limit the input choices a user can make on a form. The implementation described here is similar to a group of radio buttons in the sense that only one item can be selected from a list. This can simplify a document interface and eliminate the need for some input checking that might otherwise need to be done if a user is free to type whatever she/he likes in an inputfield. For example, creating a pull-down list of the months of the year eliminates the need for a user to type (and perhaps to mistype) the name of a month, as shown in Document 3.5.

Document 3.5 (s e l e c t . htm)

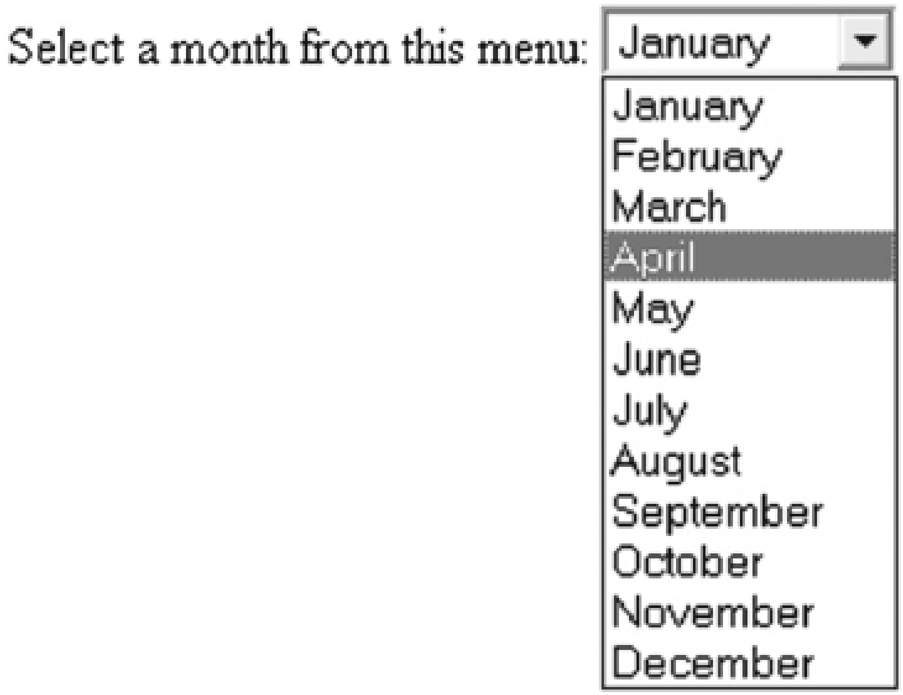
```
< h t m l >
< h e a d >
< t i t l e > P u l l - D o w n L i s t < / t i t l e >
< / h e a d >
<body><f orm >
S e l e c t a month f rom t h i s menu:

< s e l e c t n a m e = " t e st i n g " >
< o p t i o n v a l u e = "1 " se l e c t e d > J a n u a r y< / o p t i o n > < o p t i
```

```
o n v a l u e = " 2 " > F e b r u a r y < / o p t i o n > < o p t i o n v a l u e = "3
" > M a r c h </ o p ti o n >
< o p t i o n v a l u e = " 4 " > A p r i l < / o p t i o n >
< o p t i o n v a l u e = " 5 "> M a y< /o p t i o n >
< o p t i o n v a l u e = " 6 " > J u n e < / o p t i o n >
< o p t i o n v a l u e = " 7 " > J u l y < / o p t i o n >
< o p t i o n v a l u e = " 8 " > A u g u st < / o p t i o n >
< o p t i o n v a l u e= "9 "> S ep tem b e r </o pt i on >
< o p t i o n v a l u e = " 1 0 " > O c to b e r < / o p t i o n >
< o p t i o n v al ue= "11"> Nov em ber</opti on>
< o p t i o n v al ue= "12"> Decem ber</opti on>

< / s e l e c t >
< / f o r m > < / b o d y >
< / h t m l >
```

Including si z e = "12 " ensures that all the months will be visible. Without this attribute, HTML can add a scroll bar to the list.
For the output shown, the user has chosen the month of April, which is now highlighted. The values of thev al ue attribute can be, but do not have to be, the same as the text displayed for each option. In this case, the month values are numbers between 1 and 12, rather than the names of the months.
Assigning the se l e c t e d attribute to the first option means that “January” will be highlighted when the pull-down box isfirst displayed. For longer lists, the default action is for HTML to include a scroll bar alongside the list.
Adding the m u l t i p l e attribute to the < se l ect> tag allows more than one selection to be made by holding down the Control key on Windows computers or the Command key on Mac Computers.
Although it is easy to create pull-down lists as well as groups of radio buttons and checkboxes, it is not yet obvious how to make use of the selections a user makes. As will be shown in subsequent chapters, PHP provides the required capabilities.



3.4 Combining Tables and Forms

In terms of organizing an HTML document which will be linked to a PHP application, it is often helpful to create one or more tables in which the cell contents arefields in a form. Document 3.6 gives an example.

Document 3.6 (si t e D efin i t i o n .h t m)

```
< h t m l >
< h e a d >
< t i t l e > O b s e r v a t i o n S i t e D e s c r i p t i o n s < / t i t l e > < / h e a
d >
<body>
< f o rm >
< t a b l e b o r d e r = " 2 " c e l l p a d d i n g = " 5 "

c e l l s p a c i n g = " 2 " a l i g n = " c e n t e r " >
< c a p t i o n > < f o n t si z e = " +2"> Ob se rv ati o n S i t e D e s c r i t i o
n s < / f o n t > < / c a p t i o n >
< t r b g c o l o r = " l i g h t b l u e " >

< t h > S i t e # < / t h > < t h > S i t e
N a m e < / t h > < t h > L a t i t u d e < / th >
```

```
< t h > L o n g i t u d e < / t d > < t h > E l e v a t i o n < / t h > < / t r >
< t r b g c o l o r = " p a l e g r e e n " >

< t d > S i t e 1 < / t d >
< t d > < i n p u t t y p e = " t e x t " name="Name1"
s i z e = " 1 0 " m ax length="10" v al ue= "Name1" / > < / t d >
< t d > < i n p u t t y p e = " t e x t " n a m e = " L a t i t u d e 1 "
s i z e = " 1 0 "
m ax l ength="10"
v a l u e = " L a t i t u d e 1 "
/ > < / t d >
< t d > < i n p u t t y p e = " t e x t " n a m e = "L on gi tu de 1" s i z e = " 1 0
" m ax length= "10" v a l u e = " L o n g i t u d e 1 " / > < / t d >
< t d > < i n p u t t y p e = " t e x t " n a m e = " E l e v a t i o n 1 "
s i z e = " 1 0 "
m ax l ength="10" v a l u e = " E l e v a t i o n 1 " / > < / t d > < / t r >
< t r b g c o l o r = " i v o r y " >
< t d > S i t e 2 < / t d >
< t d > < i n p u t t y p e = " t e x t " name="Name2"
s i z e = " 1 0 " m ax length="10" v al ue= "Name2" / > < / t d >
< t d > < i n p u t t y p e = " t e x t " n a m e = " L a t i t u d e 2 "
s i z e = " 1 0 "
m ax l ength="10" v a l u e = " L a t i t u d e 2 " / > < / t d >
< t d > < i n p u t t y p e = " t e x t " n a m e = "L on gi tu de 2" s i z e = " 1 0
" m ax length= "10" v a l u e = " L o n g i t u d e 2 " / > < / t d >
< t d > < i n p u t t y p e = " t e x t " n a m e = " E l e v a t i o n 2 "
s i z e = " 1 0 "
m ax l ength="10" v a l u e = " E l e v a t i o n 2 " / > < / t d > < / t r >
< t r b g c o l o r = " p a l e g r e e n " >
< t d > S i t e 3 < / t d >
m ax l ength="10" v a l u e = " L a t i t u d e 3 " / > < / t d > < t d > < i n p u t
t y p e = " t e x t " n a m e ="L on gi tu de 3" s i z e = " 1 0 " m ax length=
"10" v a l u e = " L o n g i t u d e 3 " / > < / t d >
< t d > < i n p u t t y p e = " t e x t " n a m e = " E l e v a t i o n 3 " s i z e = "
1 0 " m ax l ength="10" v a l u e = " E l e v a t i o n 3 " / > < / t d > < / t r >
< t r b g c o l o r = " i v o r y " >
< t d > S i t e 4 < / t d >
< t d > < i n p u t t y p e = " t e x t " name="Name4"
```

```
s i z e = " 1 0 " m ax length="10" v al ue= "Name4" / > < / t d >
< t d > < i n p u t t y p e = " t e x t " n a m e = " L a t i t u d e 4 " s i z e = " 1
0 " m ax l ength="10" v a l u e = " L a t i t u d e 4 " / > < / t d > < t d > < i n
p u t t y p e = " t e x t " n a m e ="L on gi tu de 4" s i z e = " 1 0 " m ax
length= "10" v a l u e = " L o n g i t u d e 4 " / > < / t d >
< t d > < i n p u t t y p e = " t e x t " n a m e = " E l e v a t i o n 4 " s i z e = "
1 0 " m ax l ength="10" v a l u e = " E l e v a t i o n 4 " / > < / t d >
< / t r >
< t r b g c o l o r = " p a l e g r e e n " >
< t d > S i t e 5 < / t d >
< t d > < i n p u t t y p e = " t e x t " name="Name5"
s i z e = " 1 0 " m ax length="10" v al ue= "Name5" / > < / t d >
< t d > < i n p u t t y p e = " t e x t " n a m e = " L a t i t u d e 5 " si z e = " 1 0
" m ax l ength="10" v a l u e = " L a t i t u d e 5 " / > < / t d > < t d > < i n p
u t t y p e = " t e x t " n a m e ="L on gi tu de 5" s i z e = " 1 0 " m ax length=
"10" v a l u e = " L o n g i t u d e 5 " / > < / t d >
< t d > < i n p u t t y p e = " t e x t " n a m e = " E l e v a t i o n 5 " s i z e = "
1 0 " m ax l ength="10" v a l u e = " E l e v a t i o n 5 " / > < / t d >
< / t r >
< / t a b l e
>

< / f o r m > Site # Site Name Latitude < / b o d y >
< / h t m l > Site 1 Name1 Latitude1
```

Longitude Elevation

Longitude1 Elevation1

Site 2 Name2 Latitude2 Longitude2 Elevation2

Site 3

Name3

Observation Site Descritions

Latitude3 Longitude3

Elevation3

Longitude4 Site 4 Name4 Latitude4 Elevation4

Longitude5 Site 5 Name5 Latitude5 Elevation5

The output is shown with the original default field names, before a user starts to add new values.
Although it may seem like a lot of work to create Document 3.6, the task is

greatly simplified by copying and pasting information for the rows. When you access this page, the Tab key moves fromfield tofield but skips thefirst column, which is justfixed text. The user of the page can change the default values of all the input text boxes.

3.5 HTML List Elements

As shown earlier in this chapter, the t a b l e and f orm elements are used as tools for organizing content in an HTML document. List elements provide another way to impose formatting on related content. Table 3.2 gives a brief summary of three kinds of lists.

Table 3.2 HTML list elements Description

De finition
(or glossary) Ordered
Unordered

List item
Glossary
HTML tags
<dl>…< / d l > Use
For a list that includes names and extensive descriptions

<ol> …< / ol > <ul >…< / u l > < l i > …< / l i > <dt>…< / d t > When a list of things needs to be numbered For a list of“bulleted” items

Create list entry for <ul> or <ol>

Create glossary heading for <dl>
head

Glossary term <dd>…</dd> Create glossary term description for <dl>

Document 3.7 shows how to use these list tags.
Document 3.7 (l i s t s . h t m)

```
< h t m l >
< h e a d >

< t i t l e > U s i n g HTML L i s t s < / t i t l e >
< / h e a d >
<body>
T h i s page dem onstrates t h e use o f unordered, o r d e r e d , and defin i t i
o n l i s t s .
< u l >

< l i > Use unordered l i s t s f o r " b u l l e t e d " i t e m s . < / l i > < l i >
Use ordered l i s t s f o r numbered i t e m s . < / l i > < l i > Use defin i t i o
n l i s t s f o r l i s t s o f i t e m s t o be defined. < / l i >
< / u l >
Here are t h r e e ways t o o r g a n i z e c o n t e n t in an HTML document:
< o l >

< l i > U se a t a b l e . < / l i >
< l i > U se a l i s t . < / l i >
< l i > U se < f o n t f a c e = " c o u r i e r " >& l t; p r e& g t; …

& l t; / p r e& g t; < / f o n t > t a g s . < / l i >
< / o l >
T h i s is a way to produce a n e a t l y f o r m a t te d g l o ssa r y l i s t . < d l
>

< d t > < b >de fin i t i o n l i s t < / b >
( < f o n t f a c e = " c o u r i e r " >& l t ; dl& g t ; < / f o n t > ) < / d t >
<dd>Use t h i s to d i s p l a y a l i s t of g l o ssa r y i t e m s and t h e i r
defin i t i o n s . < / d d >
<dt><b> ordered l i s t < / b >
( < f o n t f a c e = " c o u r i e r " >& l t ; ol& g t ; < / f o n t > ) < / d t >
<dd>Use t h i s to d i s p l a y a numbered l i s t . < / d d >
<dt><b> unordered l i s t < / b >
( < f o n t f a c e = " c o u r i e r " >& l t ; ul& g t ; < / f o n t > ) < / d t >
<dd>Use t h i s to d i s p l a y a l i s t of b u l l e t e d i t e m s . < / d d > < / d
```

l >
< / b o d y >
< / h t m l >

The use of these tags imposes a preset format for displaying list items. Blank lines are inserted before and after the list, with no <br /> or <p>… <p> tags required to separate the lists from other text in the document. For ordered and unordered lists, the list items themselves are indented. For the definition list, the items are not indented, but the“definitions” are. The contents of a list item can include text formatting elements. For example, in Document 3.7, the items in the definition list use the b element to display the item name in a bold font. A list item can be an image, <img src="…" />, or a URL reference, <a href ="…">.

Note the use of & l t; and &gt; to display the < and > characters in the document. (If you just type these characters, they will not be displayed on the screen because HTML will try to associate them with tags and will simply ignore them when it can’tfigure out what you meant.)

This page demonstrates the use of unordered, ordered, and definition lists.

Use unordered lists for “bulleted” items. Use ordered lists for numbered items.
Use definition lists for lists of items to be defined.

Here are three ways to organize content in an HTML document:

1. Use a table.
2. Use a list.
3. Use`<pre> ... </pre>` tags.

This is a way to produce a neatly formatted glossary list.
definition list (`<dl>`)

Use this to display a list of glossary items and their definitions. **ordered list** (`<ol>`)
Use this to display a numbered list.

unordered list (`<ul>`)
Use this to display a list of bulleted items.

There are some attributes associated with list elements that provide a little more control over the appearance of lists.

`start="` *n* `"`
Value : The integer *n* specifies the starting value of an ordered list. The default value is`start="1"` .

`type = "…"`
Values : For unordered lists: `"disc"` (the default value), `"square"` , `"circle"`

For ordered lists: `"A"` (uppercase letters),`"a"` (lowercase letters),`"I"` (uppercase Roman letters,`"i"` (lowercase Roman letters),`"1"` (numbers, the default value)

`value = "` *n* `"`
Value : The integer *n* specifies a numerical value for an item in an ordered list which overrides the default value. Subsequent list items will be renumbered starting at this value.

Finally, it is possible to combine list types to create more complicated list structures. Document 3.8 shows how list tags can be used to create the table of contents for a book.

Document 3.8 ((bookContents.htm)

```
< h t m l >
< t i t l e > T a b l e o f Contents f o r My B o o k< / t i t l e > <body>
<h2> Table o f Contents f o r My Book</h2> < o l >
<b><l i > Chapter O n e< / b > </ l i >

< o l t y p e ="I " >
< l i > S e c t i o n 1 . 1< / l i >
```

```
<ol type="i">
<li> First Topic</li>
<li> Second Topic</li>

<ul type="circle">
<li><i> subtopic 1</i></li><li><i> subtopic 2</i></li>

</ul>
</ol>
<li> Section 1.2</li>
<li> Section 1.3</li>

</ol>
<b><li> Chapter Two</b></li>
<ol type="I">
<li> Section 2.1</li>
<ol type="i">
<li> First Topic</li>
<li> Second Topic</li>
<ul type="circle">
<li><i> subtopic 1</i></li>
<li><i> subtopic 2</i></li>
</ul>
</ol>
<li> Section 2.2</li>
<li> Section 2.3</li>
</ol>
<b><li> Chapter Three</b></li>
<ol type="I">
<li> Section 3.1</li>
<ol type="i">
<li> First Topic</li>
<li> Second Topic</li>
<ul type="circle">
<li><i> subtopic 1</i></li><li><i> subtopic 2</i></li><li><i
> subtopic 3</i></li></ul>
</ol>
<li> Section 3.2</li>
```

```
<li> Section 3.3</li>
<ol type="i">
<li> First Topic</li>
<li> Second Topic</li>
</ol>
<li> Section 3.4 </li>
</ol>
</ol>
</body>
</html>
```

Note that if this list were used for an online book, for example, each list item could include a link to a URL or a hypertext link to another location within the same document.

Table of Contents for My Book

1. Chapter One

I. Section 1.1
i. First Topic
ii. Second Topic

subtopic 1

subtopic 2

II. Section 1.2
III. Section 1.3
1. Chapter Two
I. Section 2.1
i. First Topic
ii. Second Topic
subtopic 1
subtopic 2
II. Section 2.2
III. Section 2.3
1. Chapter Three
I. Section 3.1
i. First Topic
ii. Second Topic
subtopic 1
subtopic 2
subtopic 3
II. Section 3.2
III. Section 3.3
i. First Topic
ii. Second Topic
IV. Section 3.4

3.6 HTML Frames

Another way of organizing content in HTML documents is through the use of frames to divide a window into several separately addressable blocks of content. Frames are built using two elements, frame and f ram eset.

`<frame > … </frame>`
Attributes : `bordercolor` , `frameborder` , `marginheight` , `marginwidth` , `name` , `scrolling` (`yes` , `no` , or`auto`),`src`

Provides a nameable window region, as defined by the `frameset` element, with a link to the content of that region. A value for the`src` attribute must be given, but the other attributes are optional. The default value for the`scrolling` attribute is`auto` , which automatically provides a scroll bar if needed to display all of awindow’s content.

`<frameset> … </frameset>`
Attributes : `border` , `bordercolor` , `cols` , `frameborder` , `framespacing` ,`rows`

Provides specifications for dividing a webpage window into two or more separately linkable sub-windows. All attributes are optional except `cols` and`rows` , which must have values of *n* pixels, *n* % of the available window, or`*` to fill the remaining window space.

Consider the following screen display. It is divided into three sections. The upper left-hand corner contains a clickable image. The lower left-hand corner contains links to other HTML documents. The right-hand column will be used to display those documents. When this page isfirst accessed, a “home page” document should be displayed.

Document 3.9a shows thefirst step in creating this page. Document 3.9a (frameMain.htm)

< h t m l >
< h e a d >
< t i t l e >A si m p l e f ram eset d o c u m e n t < / t i t l e >
< / h e a d >
< f r a m e s e t c o l s= " 3 0 % , 70%" f ram eborder= "1">

< f r a m e s e t rows= "60%, 40%">
< f r a m e s r c = " f r a m e 1 . h t m " s c r o l l i n g = " n o " /> < f r a m e s r c = " f r a m e 2 . h t m " />

< / f r a m e s e t >
< f r a m e name="homeFrame" sr c = " h o m e F r a m e .h tm " /> < / f r a m e s e t > < / h t m l >

The f ram eset element is used to define the frames. In this case, the window is divided into two columns. The left-hand column occupies 30% of the page and the right-hand column occupies the remaining 70%. (In the graphic displayed below, the proportions look different because the screen display has been cropped to save space.) The line

< f r a m e s e t c o l s= " 3 0 % , 70%" f ram eborder= "1"> could also be written
< f r a m e s e t c o l s= " 3 0 % , *" f ram eborder= "1">

where the asterisk is interpreted as “fill the remaining portion of the screen with the right-hand column.” If the frame size is given as a number without the %sign, it is interpreted as pixels rather than a percentage of the full window. Setting this frame size to c o l s= " 2 0 0,* " will produce a left-side frame that is always 200 pixels wide, regardless of the screen width and resolution.

The left-hand column is further divided into two sub-windows. The top window occupies the top 60% and the bottom window occupies the remaining 40%. Each window is associated with a separate HTML document, frame1.htm and frame2.htm. These windows *could* be given names, but they don’t have to have names. The right-hand column is associated with another HTML document, homeFrame.htm. This“home frame” will be the destination for content that will be linked from the frame in the lower left-hand corner. This frame needs a name to serve as a“target” for the other documents that will be displayed here. The name can be anything, but homeFrame is a self-explanatory and therefore reasonable choice.

Documents 3.9b through 3.9d show the HTML code for each of the three frames.
Document 3.9b (homeFrame.htm)

```
< h t m l >
< h e a d >
< t i t l e > M y Home F r a m e < / t i t l e >
< / h e a d >
<body b g c o l o r = " l i g h t g r e e n " >
< h 1 > < b l i n k > < f o n t col or= "maroon"><b>< i>Home page d i s p l a
y goes h e r e . < / i > < / b > < / f o n t > < / b l i n k > < / h 1 > < / b o d y >
< / h t m l >
```

Document 3.9c (frame1.htm)

```
< h t m l >
< h e a d >
< t i t l e > T i t l e F r a m e < / t i t l e >
< / h e a d >
<body b g c o l o r = " p i n k " >
< f o n t si z e = " + 6 " c o l o r = " n a v y " > < c e n t e r > < b> < i >F r a
m e s < b r />Demo<br/>
< a h r e f = " f r a m e D e s c r i p t i o n . h t m " / > < i m g

s r c = " f r a m e . g i f "
b o r d e r = " 2 " > < / i > < / b > < / c e n t e r > < / a >
< / f o n t >
< / b o d y >
< / h t m l >
```

Document 3.9d (frame2.htm)

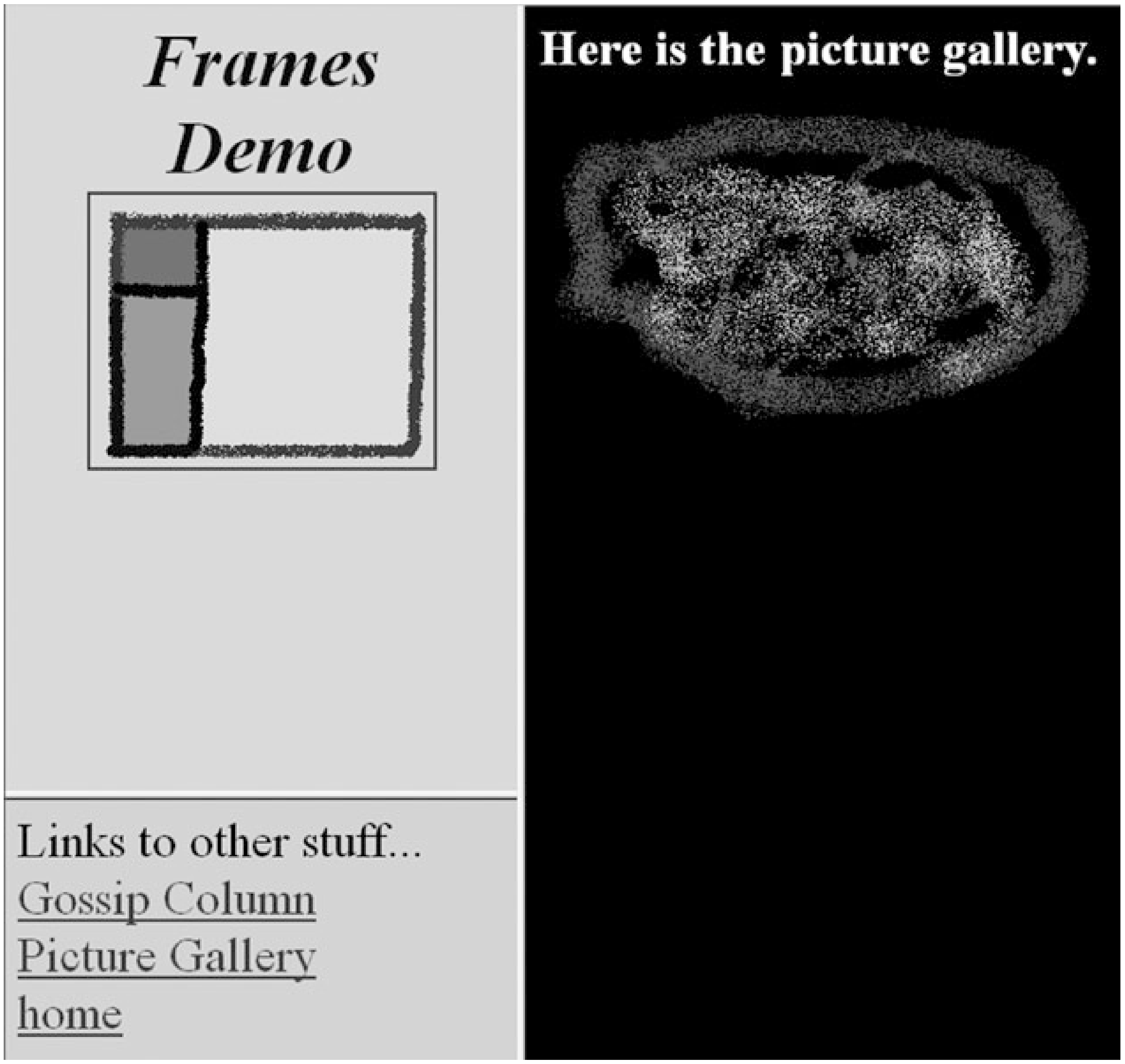
```
< h t m l >
< h e a d >
< t i t l e > G o s s i p C o l u m n < / t i t l e >
< / h e a d >
<body b g c o l o r = " l i g h t b l u e " >
< f o n t si z e = " + 3 " >
```

```
L i n k s to o t h e r s t u f f . . . < b r />
<a h r e f =" g o s s i p . h t m " target= "hom eFram e" / > Gossip Column</a>
< b r />
<a h r e f = " p h o t o G a l l e r y . h t m " target= "hom eFram e" />

P i c t u r e G a l l e r y < / a > < b r />
<a href = "hom eFram e.htm " target= "hom eFram e" />home</a><br />
< / f o n t >
< / b o d y>
< / h t m l >
```

Document 3.9e is the HTML document referenced in Document 3.9c.
Document 3.9e (f ram eDescri pti on.htm)

```
< h t m l >
< h e a d >
< t i t l e > H o w t h i s image was c r e a t e d . < / t i t l e >
< / h e a d >
<body>
T h i s image was c r e a t e d i n Windows' P a i n t program . <a h r e f = " f
r a m e 1 . h t m " /> C l i c k here to r e t u r n . < / a > < / b o d y >
< / h t m l >
```



Document 3.9d, for the lower left-hand corner, contains links to several other documents, each of which can be displayed in the right-hand window. This is done by using the t a r g e t attribute, which links to homeFrame, the namevalue given in Document 3.9a:

```
<a h r e f = " g o s s i p . h t m " ta rg et=" hom e Fr am e"> Go ssi p
Column</a>
```

It is up to you to provide the gossi p.htm and p h o toG al l e ry .htm documents. Document 3.9d also includes a link back to the home page document. The image shown here is the result of clicking on the“Picture

Gallery” link to a document on the author’s computer; the page image has been cropped to save space.

Document 3.9b contains the code for the home frame that is displayed when the page isfirst accessed. (The b l i n k element, which causes text to blink on and off, will be ignored by some browsers.) Document 3.9c, for the upper left-hand frame, contains the clickable image, f r a m e .gif, with a border drawn around it. Clicking on the image opens a link to descriptive file, f r a m eD e sc r i p ti o n . htm (see Document 3.9e), to be provided by you. This document will be displayed in the“Frames Demo” window (not opened in a new window), and it should contain a link to return to frame1.htm:

<a h r e f = " f r a m e 1 .h t m "> C l i c k here to r e t u r n . < / a >

HTML frames provide a great deal of flexibility for displaying content, but there is one consequence that may not be immediately obvious. If you try these examples on your own computer, you will see that *only* the main frame document (frameMain.htm) is displayed as the URL link, regardless of which document is being displayed in the right-hand column. So, you cannot directly copy or bookmark the URL for a particular document. Accessing the“view source” option on your browser will display the HTML code only for frameMain.htm. If you wish to bookmark the“picture gallery” page, for example, you cannot do so directly. You can display the page separately by accessing the document separately:

h t t p :// ... / p h o t o G a l l e r y . h t m but doing that assumes you already know the name and location of this document.

This situation does not really hide all the code for these documents. You can look at the frameMain.htm HTML code and then access separatelythe homeFrame.htm, frame1.htm, and frame2.htm documents to examine their HTML code.

7. More Examples 1. Selecting Cloud Types from a List of Possibilities

Create a document that allows users to select observed cloud types from a list of possibilities. More than one cloud type can exist simultaneously. The categories are:

high altitude: Cirrus, Cirrocumulus, Cirrostratus mid altitude: Altostratus, Altocumulus
low altitude: Stratus, Stratocumulus, Cumulus precipitation-producing: Nimbostratus, Cumulonimbus A good way to organize this information is to use a table within a form.

The form fields should be of type checkbox rather than r a d i o because cause multiple selections are possible. Compare this problem with Document 3.3, in which a table was used to display just the cloud types.

Document 3.10 (cloud1.htm)

```
< h t m l >
<head>
< t i t l e > C l o u d O b s e r v a t i o n s< / t i t l e >
< / h ea d >
<body b g c o l o r =" # a a d d f f " >
<h1> Cloud Observ ati ons</ h1>
<b> Cloud Observ ati ons < / b >
( S e l e c t as many c l o u d t y p e s as o b se r v ed .)
< b r />
<f or m >
< t a b l e >

< t r >
<t d><b> Hi gh</ b> < / t d >

< t d >
< i n p u t t y p e =" c h ec k b o x"
name="high" v a l u e =" C i r r u s " />
C i r r u s< / t d >

< t d >
< i n p u t t y p e =" c h ec k b o x" name="high"
v a l u e =" C i r r o c u m u l u s " / > Ci rrocum ul us < / t d >

< t d >
< i n p u t t y p e = "c h ec k b o x" name="high"
```

```
value="Cirrostratus"/> Cirrostratus
</td></tr>

<tr>
<td colspan="4"><hr noshade color="black"/>

</td></tr>
<tr>
<td><b> Middle</b></td>
<td>

<input type="checkbox" name=" mid "
value="Altostratus"/> Altostratus
</td>

<td>
<input type="checkbox" name="mid"
value="Altocumulus"/>
Altocumulus</td></tr>

<tr>
<td colspan="4"><hr noshade color="black"/>

</td></tr>
<tr>
<td><b> Low</b></td>
<td>
<input type="checkbox" name="low" value="Stratus"/>

Stratus</td>
<td>

<input type="checkbox" name="low"
value="Stratocumulus"/>
Stratocumulus</td>

<td>
<input type="checkbox" name="low" value="Cumulus"/> Cumulus </td>
</tr>
```

```
< t r >
< t d c o l sp a n=" 4 " > <hr noshade c o l o r =" b l a c k " /> < / t d > < / t r >

< t r >
< t d > <b> Rai n-Produci ng < / b > < / t d >
< t d >

< i n p u t t y p e =" c h ec k b o x" nam e="r ai n"
v a l u e =" N i m b o s t r a t u s " />
N i m b o st r at us< / t d>

< t d >
< i n p u t t y p e =" c h ec k b o x" nam e="r ai n"
v al ue="Cum ul oni m bus " / > Cumulonimbus
< / t d > < / t r >

< / t a b l e >
< / f o r m >

< / b o d y >
In Document < / h t m l >
3.10, checkboxes
```

for the cloud types are organized into four groups, for high-, mid-, and low-altitude clouds, plus rain-producing clouds. Withineach group, each checkbox has a name associated with it.

Note that the names given to each checkbox in Document 3.10 are the same as the text entered in the corresponding cell. This is only because these names and text are reasonable descriptions of the cell contents. In general, the text in the cell does not need to be the same as, or even related to, the value of the name attribute of the checkbox.

In general, there ’s not much point in being able to make choices in checkboxes unless there is a way to respond to the choices. As will be shown later, it is easy to transfer these choices to a PHP application.

3.7.2 A “Split Window” Application

Cloud Observations

Cloud Observations (Select as many cloud types as observed.) **High**
☐ Cirrus ☐ Cirrocumulus ☐ Cirrostratus

Middle
☐ Altostratus ☐ Altocumulus

Low
☐ Stratus ☐ Stratocumulus ☐ Cumulus

Rain-Producing
☐ Nimbostratus ☐ Cumulonimbus

Create an application that maintains one or more “header lines” across the top of a web page window while scrolling through a long text document. Consider thisfile:

```
DRB Worcester PA
40.178 -75.3325
4030 5200
Mon day yr hr min sec EST PYR-1 PYR-2 T
7 1 2008 0 0 0 1 0.00031 0.00031 20.198
7 1 2008 0 1 0 1.000694444 0.00031 0.00031 20.174
7 1 2008 0 2 0 1.001388889 0.00031 0.00031 20.174 …
```

The file contains 1440 lines of data (24 hours times 60 minutes per hour for July 1, 2008) with the date and time, the day and time converted to a fractional Eastern Standard Time day (EST), data from two instruments, PYR-1 and PYR-2, and air temperature in degrees Celsius. The instruments measure incoming solar radiation.

For a file of this size, it might be convenient to be able to display these data under afixed header that identifies the columns, in the same way that spreadsheets allow creation of a“split window.” Documents 3.11a and 3.11b show a very simple solution to this problem, using HTML frames.

Document 3.11a (pyranometerMain.htm)

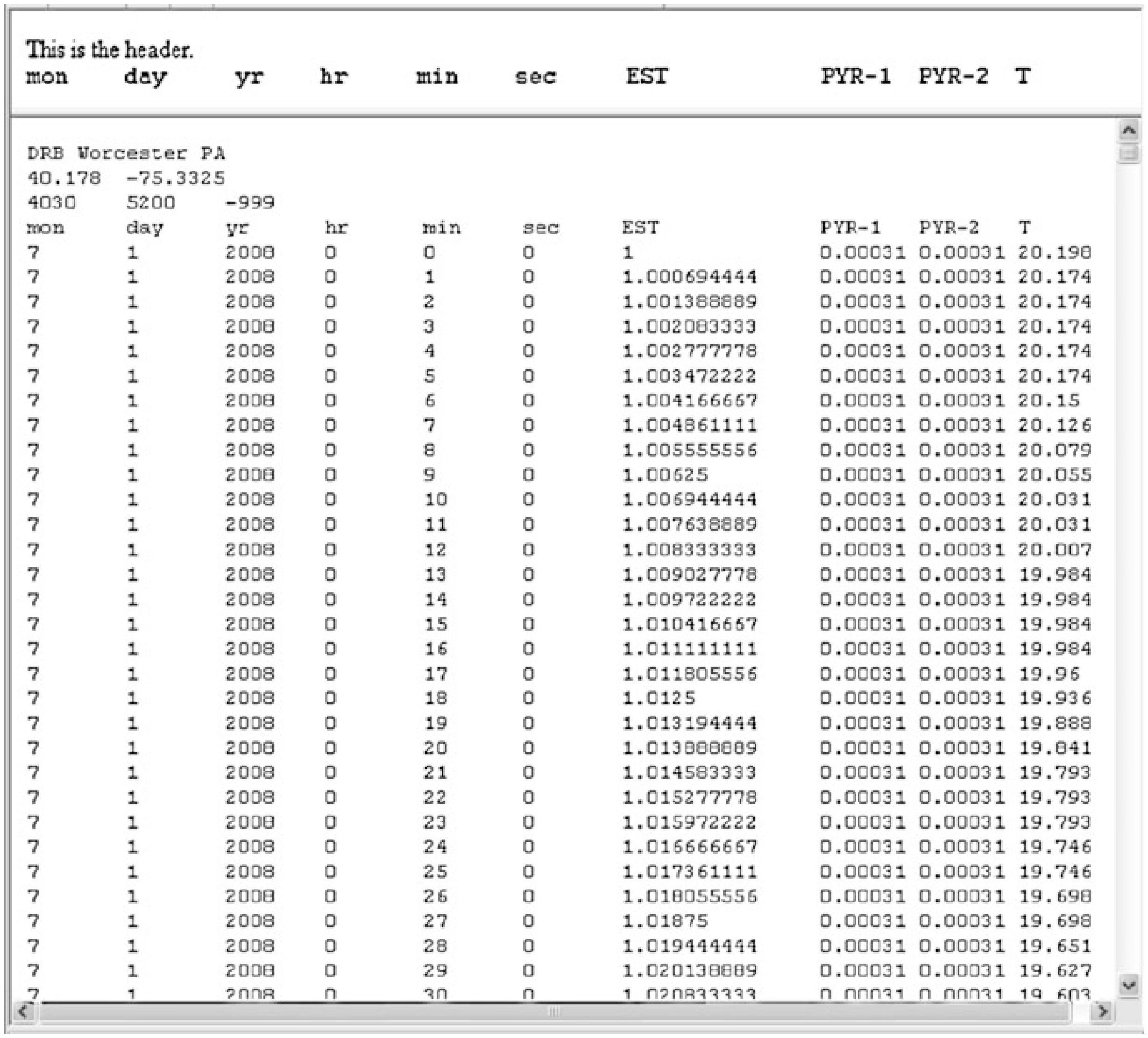
```
< h t m l >
< h e a d >
< t i t l e > D i s p l a y pyranometer d a t a < / t i t l e >
< / h e a d >
< f r a m e s e t rows="10%, * " >

< f r a m e s r c = " h e a d e r . h t m " s c r o l l i n g = " n o " /> < f r a m e s
r c = " p y r a n o m e t e r . d a t " />
< / f r a m e s e t >
< / h t m l >
```

Document 3.11b (header.htm)

```
< h t m l >
< h e a d >

< t i t l e > < / t i t l e >
< / h e a d >
<body>
< f o n t f a c e = " c o u r i e r " >
T h i s is t h e h e a d e r . < b r />
mon&nbsp; &nbsp; day &nbsp; &nbsp; y r &nbsp; &nbsp;hr &nbsp; &nbsp; min &nbsp; &nbsp;
sec &nbsp; &nbsp; EST &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; PYR-1 &nbsp; PYR-2 &nbsp;
T<br /> < / f o n t >
< / b o d y > < / h t m l >
```



```
This is the header.
mon     day     yr     hr     min     sec     EST            PYR-1  PYR-2  T
```

```
DRB Worcester PA
40.178  -75.3325
4030    5200    -999
mon     day     yr      hr      min     sec     EST             PYR-1   PYR-2   T
7       1       2008    0       0       0       1               0.00031 0.00031 20.198
7       1       2008    0       1       0       1.000694444     0.00031 0.00031 20.174
7       1       2008    0       2       0       1.001388889     0.00031 0.00031 20.174
7       1       2008    0       3       0       1.002083333     0.00031 0.00031 20.174
7       1       2008    0       4       0       1.002777778     0.00031 0.00031 20.174
7       1       2008    0       5       0       1.003472222     0.00031 0.00031 20.174
7       1       2008    0       6       0       1.004166667     0.00031 0.00031 20.15
7       1       2008    0       7       0       1.004861111     0.00031 0.00031 20.126
7       1       2008    0       8       0       1.005555556     0.00031 0.00031 20.079
7       1       2008    0       9       0       1.00625         0.00031 0.00031 20.055
7       1       2008    0       10      0       1.006944444     0.00031 0.00031 20.031
7       1       2008    0       11      0       1.007638889     0.00031 0.00031 20.031
7       1       2008    0       12      0       1.008333333     0.00031 0.00031 20.007
7       1       2008    0       13      0       1.009027778     0.00031 0.00031 19.984
7       1       2008    0       14      0       1.009722222     0.00031 0.00031 19.984
7       1       2008    0       15      0       1.010416667     0.00031 0.00031 19.984
7       1       2008    0       16      0       1.011111111     0.00031 0.00031 19.984
7       1       2008    0       17      0       1.011805556     0.00031 0.00031 19.96
7       1       2008    0       18      0       1.0125          0.00031 0.00031 19.936
7       1       2008    0       19      0       1.013194444     0.00031 0.00031 19.888
7       1       2008    0       20      0       1.013888889     0.00031 0.00031 19.841
7       1       2008    0       21      0       1.014583333     0.00031 0.00031 19.793
7       1       2008    0       22      0       1.015277778     0.00031 0.00031 19.793
7       1       2008    0       23      0       1.015972222     0.00031 0.00031 19.793
7       1       2008    0       24      0       1.016666667     0.00031 0.00031 19.746
7       1       2008    0       25      0       1.017361111     0.00031 0.00031 19.746
7       1       2008    0       26      0       1.018055556     0.00031 0.00031 19.698
7       1       2008    0       27      0       1.01875         0.00031 0.00031 19.698
7       1       2008    0       28      0       1.019444444     0.00031 0.00031 19.651
7       1       2008    0       29      0       1.020138889     0.00031 0.00031 19.627
7       1       2008    0       30      0       1.020833333     0.00031 0.00031 19.603
```

The &nbsp; escape sequence allows the insertion of“white space” between text to align headers. The f ram eset rows attribute allocates the top 10% of the page to the header and the outputfile, pyranom eter .dat, is displayed in the remainder of the page. For a display that is too long tofit in one window, HTML automatically creates a scroll bar down the righthand side of the window. A border has been retained under the top frame, just to make clear how the page is divided, but it is optional; to remove the border, set the f ram eset attributeborder= "0".

A simple modi fication of the f ram eset code in Document 3.11a would allow listing a number of differentfiles in a left-hand column, each of which could be displayed in the home page frame simply by clicking on thefile name. To do this, the direct link to pyranom eter .dat in Document 3.11a would be replaced with another name specified as the value of a t a r g e t attribute in the reference to each document to be displayed:

<ahref="…" target="…" …/>

The pyranom eter .datfile is just a tab-delimited textfile, not an HTML document. Although it is of some interest to be able to display a large textfile in HTML, it would be more useful to be able to use thatfile for something. HTML documents cannot process external datafiles and that is the purpose of interfacing HTML with PHP, which will be the topic of the rest of this book.

THE END