

Objectives

Session 1.1

- Review the history of the Internet, the Web, and HTML
- Describe different HTML standards and specifications
- Learn about the basic syntax of HTML code

Session 1.2

- Mark elements using two-sided and one-sided tags
- Insert an element attribute
- Create comments
- Describe block-level elements and inline elements
- Specify an element's appearance with inline styles
- Create and format different types of lists
- Create boldfaced and italicized text
- Describe logical and physical elements

Session 1.3

- Define empty elements
- Insert an inline image into a Web page
- Insert a horizontal line into a Web page
- Store meta information in a Web document
- Display special characters and symbols

Developing a Basic Web Page

Creating a Web Page for Stephen Dubé's Chemistry Classes

Case

Stephen Dubé's Chemistry Classes

Stephen Dubé teaches chemistry at Robert Service High School in Edmonton, Alberta (Canada). In previous years, he has provided course information to students and parents with handouts. This year, he wants to put that information on the World Wide Web, where anyone can access it easily. Eventually, he hopes to post homework assignments, practice tests, and even grades on the Web site. Stephen is new to this technology and has asked you to help him create a Web page for his class.

Student Data Files

You can find a complete listing of the associated files in the comment section of each HTML file.

▼ Tutorial.01

▼ Tutorial folder
dube.jpg

▼ Review folder
chemtxt.htm
logo.jpg
flask.jpg

▼ Case1 folder
childtxt.htm
newborn.jpg

▼ Case2 folder
euler.jpg
eulertxt.htm
pi.jpg

▼ Case3 folder
flakes.jpg
frostdtxt.htm
runner.jpg

▼ Case 4 folder
logo.jpg
smith.jpg
smith.txt

Session 1.1

Introducing the World Wide Web

Before you start creating a Web page for Stephen, it's helpful to first look at the history of the Web and how the HTML language was developed. To understand this history, we need to first become familiar with some of the basic features of networks.

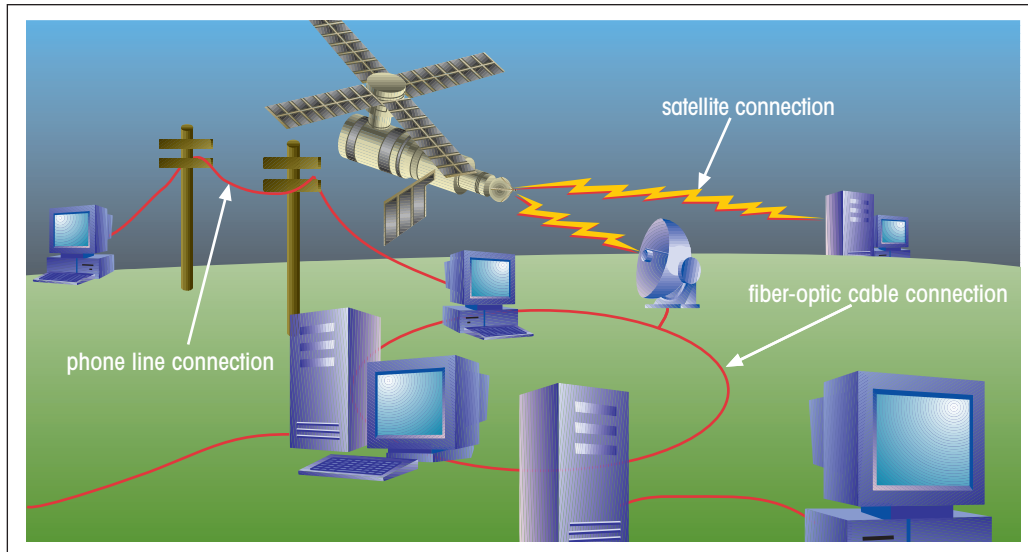
A **network** is a structure linking computers together for the purpose of sharing resources such as printers and files. Users typically access a network through a computer called a **host** or **node**. A computer that makes a resource available to a network is called a **server**. A computer or other device that requests services from a server is called a **client**. Networks can be structured in many different ways. One of the most common structures is the **client-server network**, which is made up of several clients accessing information provided by one or more servers. You may be using such a network to access your data files for this tutorial from a network file server.

We can also classify networks based on their ranges. If the computers that make up a network are close together—for example, within a single department or building—then the network is referred to as a **local area network** or **LAN**. A network that covers a wider area, such as several buildings or cities, is called a **wide area network** or **WAN**. Wide area networks are typically built from two or more local area networks. The largest WAN in existence is the **Internet**.

In its early days in the late 1960's, the Internet was called the **ARPANET** and consisted of two network nodes located at UCLA and Stanford connected by a phone line. Today, the Internet has grown to include hundreds of millions of interconnected computers, cell phones, PDAs, televisions, and networks. The physical structure of the Internet uses fiber-optic cables, satellites, phone lines, and other telecommunications media, enabling a worldwide community to communicate and share information (see Figure 1-1).

Figure 1-1

Structure of the Internet



Most of the early Internet tools required users to master a bewildering array of terms, acronyms, and commands. Even navigating the network required users to be well-versed in both computers and network technology. Before the Internet could be accessible to the general public, it needed a simpler interface. This interface proved to be the World Wide Web.

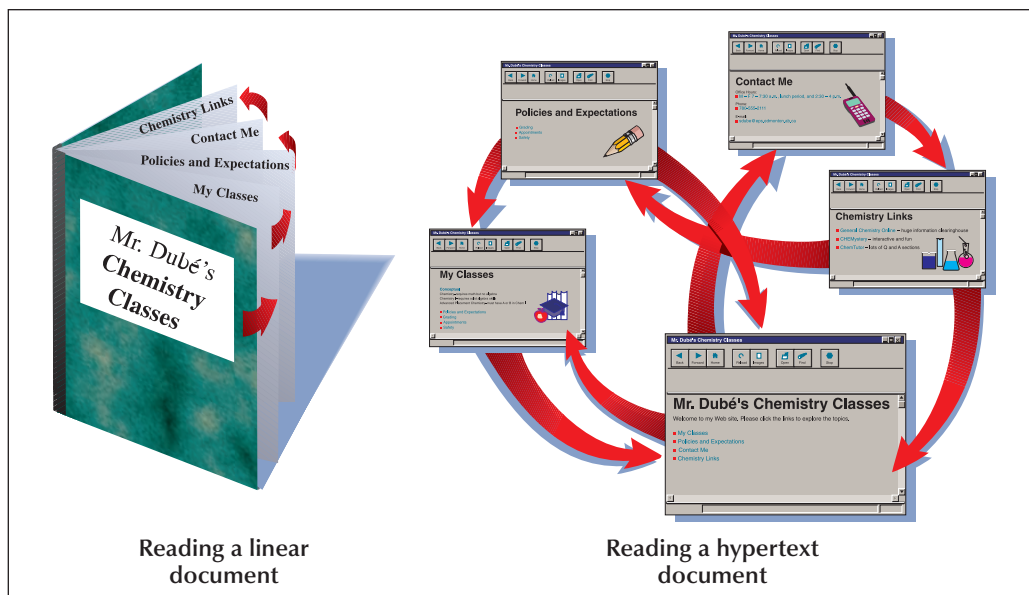
The Development of the World Wide Web

The foundations for the **World Wide Web**, or the **Web** for short, were laid in 1989 by Timothy Berners-Lee and other researchers at the CERN nuclear research facility near Geneva, Switzerland. They needed an information system that would make it easy for their researchers to locate and share data, and which would require minimal training and support. To meet this need, they developed a system of interconnected hypertext documents that allowed their users to easily navigate from one topic to another. **Hypertext** is a method of organizing information that gives the reader control over the order in which the information is presented.

Properly used, hypertext provides quicker and simpler access to diverse pieces of information than traditional methods could. For example, when you read a book, you follow a linear progression, reading one page after another. With hypertext, you progress through those pages in whatever order best suits you and your objectives. Figure 1-2 illustrates the relationships of topics in linear and hypertext documents.

Linear versus hypertext documents

Figure 1-2



The key to hypertext is the use of **hyperlinks**, or **links**, which are the elements in a hypertext document that allow you to jump from one topic to another, often with a mouse click. A link may point to another section of the same document, or to another document entirely. A link can open a document on your computer or, through the Internet, a document on a computer anywhere in the world.

The hypertext approach was just what was needed to make the Internet accessible to the general public. The end user didn't need to know exactly where a document was stored, just how to get it. Since getting it was often no more difficult than clicking a mouse, access to any document anywhere was literally at every user's fingertips. The fact that the Internet and the World Wide Web are synonymous in many users' minds is a testament to the success of this approach.

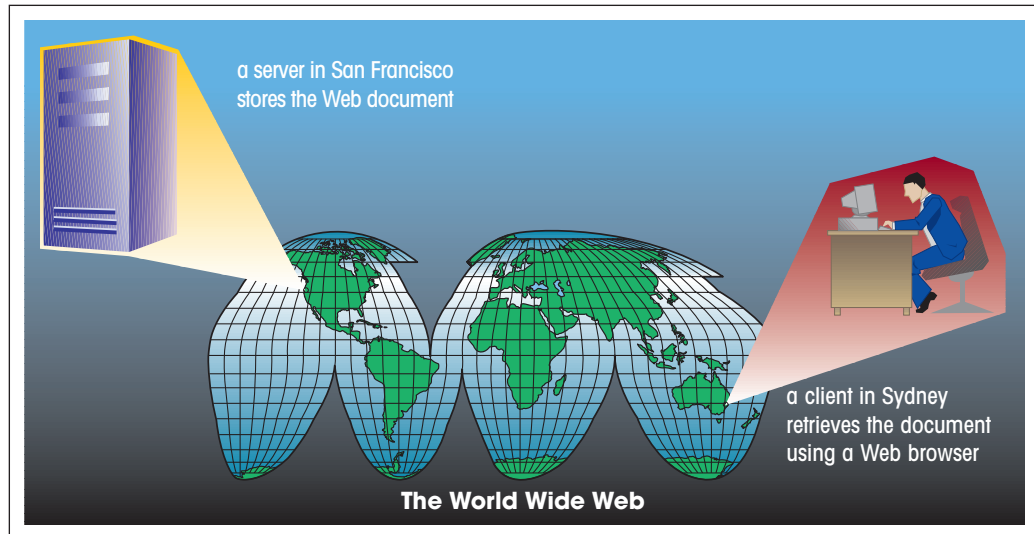
An entire collection of linked documents is referred to as a **Web site**. The hypertext documents within a Web site are known as **Web pages**. Web sites have evolved from simple collections of text documents into complex sites where users can make purchases online, discuss a variety of topics, or access real-time stock market quotes. Individual pages can contain text, audio, video, and even programs that users can run remotely. While Web pages are primarily sources of information, they are increasingly becoming works of art in their own right.

Web Servers and Web Browsers

A Web page is stored on a **Web server**, which in turn makes it available to the network. To view a Web page, a client runs a software program called a **Web browser**, which retrieves the page from the server and displays it (see Figure 1-3).

Figure 1-3

Using a browser to view a Web document from a Web server



The earliest browsers, known as **text-based browsers**, were incapable of displaying images. Today most computers support **graphical browsers**, which are capable of displaying not only images, but also video, sound, animations, and a variety of graphical features. Cell phones can also connect to the Web to display sports scores or stock market tickers. Browsers can run on teletype machines, PDAs (personal digital assistants), Braille machines, and even information devices within a car. How does a Web page work with so many combinations of browsers and clients and devices? To understand, let's look at how Web pages are created.

HTML: The Language of the Web

A Web page is simply a text file written in a language called **Hypertext Markup Language** or **HTML**. We've already discussed hypertext, but what is a markup language? A **markup language** is a language that describes a document's structure and content. For example, if this tutorial were created using a markup language, that language would mark heading text, paragraph text, figure captions, and so forth.

Even though you can incorporate program code into an HTML file, HTML is not a programming language. HTML is also not a formatting language like those used by some desktop publishing programs. HTML does not necessarily tell you how a browser will display a document. If you want to format your document, the preferred method is to use styles. **Styles** are format descriptions written in a separate language from HTML that tell browsers how to render each element. We'll explore some of the basic styles as we create our first Web pages.

The History of HTML

Because the way HTML evolved impacts how you use it today, it's a good idea to review the language's history before going further into its details. The first version of HTML

was created using the **Standard Generalized Markup Language (SGML)**. Introduced in the 1980's, SGML is a strong and highly flexible **metalanguage**, or a language used to create other languages. SGML is device- and system-independent, meaning that it can be applied to almost any type of document stored in almost any format. While powerful, SGML is also quite complex; for this reason, SGML is limited to those organizations that can afford the cost and overhead of maintaining complex SGML environments. However, SGML can also be used to create markup languages like HTML, which are tailored to specific tasks and are simpler to use and maintain.

In the early years after HTML was created, no one organization was responsible for the language. Web developers were free to define and modify HTML in whatever ways they thought best. While many rules were common, competing browsers, seeking to dominate the market, introduced some differences in the language. Such changes to the language were called **extensions**. The two major browsers during the 1990's, Netscape Navigator and Microsoft Internet Explorer, added the most extensions to HTML. These extensions were providing Web page authors with more options, but at the expense of complicating Web page development.

Web designers faced the challenge of determining which browser or browser version supported a particular extension, and creating a workaround for browsers that did not. By adding this layer of complexity to Web design, extensions, while often useful, diminished the promise of simplicity that made HTML so attractive in the first place.

Ultimately, a group of Web developers, programmers, and authors called the **World Wide Web Consortium**, or the **W3C**, created a set of **standards** or **specifications** that all browser manufacturers were to follow. The W3C has no enforcement power, but since a uniform approach to Web page creation is in the best interests of everyone, the W3C's recommendations are usually followed, though not always right away. The W3C also provides online tutorials, documentation, and quizzes that you can use to test your knowledge of HTML and other languages. For more information on the W3C and the services they offer, see their Web site at <http://www.w3c.org>.

Figure 1-4 summarizes the various versions of HTML that the W3C has released over the past decade. While you may not grasp all of the details of these versions yet, the important thing to understand is that HTML doesn't come in only one flavor.

Versions of HTML and XHTML

Figure 1-4

Version	Date	Description
HTML 1.0	1989–1994	The first public version of HTML which included browser support for inline images and text controls.
HTML 2.0	1995	The first version supported by all graphical browsers. It introduced interactive form elements such as option buttons and text boxes. A document written to the HTML 2.0 specification is compatible with almost all browsers on the World Wide Web.
HTML 3.0	1996	A proposed replacement for HTML 2.0 that was never widely adopted.
HTML 3.2	1997	This version included additional support for creating and formatting tables and expanded the options for interactive form elements. It also supported limited programming using scripts.
HTML 4.01	1999	This version added support for style sheets to give Web designers greater control over page layout. It added new features to tables and forms and provided support for international features. This version also expanded HTML's scripting capability and added increased support for multimedia elements.
XHTML 1.0	2001	This version is a reformulation of HTML 4.01 in XML and combines the strength of HTML 4.0 with the power of XML. XHTML brings the rigor of XML to Web pages and provides standards for more robust Web content on a wide range of browser platforms.
XHTML 1.1	2002	A minor update to XHTML 1.0 that allows for modularity and simplifies writing extensions to the language.
XHTML 2.0	2004–	The latest version, designed to remove most of the presentational features left in HTML.

When you create your Web pages you'll have to keep in mind not only what the W3C has recommended, but also what browsers currently in use actually support. This may mean dealing with a collection of approaches: some are new and meet the latest specifications, while some are older but still widely supported. Older features of HTML are often **deprecated**, or phased out, by the W3C. While deprecated features might not be supported in current or future browsers, that doesn't mean that you can't continue to use them—indeed, if you are supporting older browsers, you may *need* to use them. Because it's hard to predict how quickly a deprecated feature will disappear from the Web, it's crucial to be familiar with these features.

Future Web development is focusing increasingly on two other languages. **XML (Extensible Markup Language)** is a metalanguage like SGML, but without SGML's complexity and overhead. Using XML, document developers can create documents that obey specific rules for their content and structure. This is in contrast with a language like HTML, which included a wide variety of rules without a built-in mechanism for enforcing them. Indeed, one of the markup languages created with XML is **XHTML (Extensible Hypertext Markup Language)**, a stricter version of HTML. XHTML is designed to confront some of the problems associated with the different and competing versions of HTML, and to better integrate HTML with XML.

Even though XHTML shows great promise for the Web, HTML will not become obsolete anytime soon. HTML and XHTML overlap considerably, and the World Wide Web is still full of old HTML documents. In addition, we need to support those Web users who are still using older versions of Web browsers. In this book, we'll discuss the syntax of HTML 4.01 and XHTML 1.1, but we'll also bring in deprecated features and browser-supported extensions where appropriate.

Where does all of this leave you as a potential Web page author? A few guidelines are helpful:

- **Become well-versed in the history of HTML.** Unlike other languages, HTML history impacts how you write your code.
- **Know your market.** Do you have to support older browsers, or have your clients standardized on a particular browser or browser version? The answer affects how you write the code for your Web pages. Become familiar with what different browsers can and can't do.
- **Test.** If you have to support several types of browsers and several types of devices, get them and use them to view your documents. Don't assume that if your page works in one browser it will work in an older version of that same browser. In addition, a given browser version might even perform differently under different operating systems.

Tools for Creating HTML Documents

Because HTML documents are simple text files, you can create them with nothing more than a basic text editor such as Windows Notepad. However, specialized HTML authoring programs, known as HTML converters and HTML editors, are available to perform some of the rote work of document creation.

An **HTML converter** converts formatted text into HTML code. You can create the source document with a word processor such as Microsoft Word, and then use the converter to save the document as an HTML file. Converters free you from the laborious task of typing HTML code, and because the conversion is automated, you do not have to worry about introducing coding errors in your document. However, HTML code created using a converter is often longer and more complicated than it needs to be, resulting in larger-than-necessary files. Also, it is more difficult to edit HTML code directly in a file created by a converter.

An **HTML editor** helps you create an HTML file by inserting HTML codes for you as you work. HTML editors can save you a lot of time and can help you work more efficiently. Their advantages and limitations are similar to those of HTML converters. In addition,

while HTML editors allow you to set up a Web page quickly, you will usually still have to work directly with the HTML code to create a finished document.

In the next session, you'll start creating your first HTML document using a simple text editor.

Session 1.1 Quick Check

1. What is a hypertext document?
2. What is a Web server? A Web browser? Describe how they work together.
3. What is HTML?
4. How do HTML documents differ from documents created with a word processor such as Word or WordPerfect?
5. What is a deprecated feature?
6. What are HTML extensions? What are some advantages and disadvantages of using extensions?
7. What software program do you need to create an HTML document?

Review

Session 1.2


Creating an HTML Document

It's always a good idea to plan out a Web page before you start coding. You can do this by drawing a planning sketch or by creating a sample document using a word processor. The preparatory work can weed out errors or point to potential problems. In this case, the chemistry teacher, Stephen Dubé, has already drawn up a handout that he's used for many years with his students and their parents. The handout lists his classes and describes his class policies regarding homework, exams, and behavior. Figure 1-5 shows the current handout that Stephen is using.

Figure 1-5

The chemistry class handout

heading → **Mr. Dubé's Chemistry Classes**
at Robert Service High School

image → 

horizontal line →

paragraph → Welcome to the Robert Service High School Chemistry Web page. Here you'll learn more about our chemistry classes and our policies.

list → **Chemistry Classes**

- Conceptual Chemistry: An introductory course, requiring basic math but no algebra
- Chemistry I: An introductory course, requiring solid algebra skills
- Advanced Placement Chemistry: An advanced course requiring a grade of A or B in Chemistry I

bold and italicized text → **Class Policies**

Grading

Homework: Homework is worth 5 to 10 points and will be given daily. A quiz consisting of 1 or 2 homework problems from the previous week may be given in place of homework.

Tests and Quizzes: Quizzes are worth 10 to 25 points and will be given at least once a month. Tests are worth up to 100 points and will be given three times each quarter.

Labs: Labs are worth 10 to 30 points and will be graded on safety, participation, and write-up. Reports should be neatly written or typed. *Research projects* will also be assigned throughout the year.

Appointments

You can meet with Mr. Dubé before or after school or during most lunch hours in room H113. Do not hesitate to ask for help! It can be very hard to catch up if you fall behind.

Safety

Labs are completed weekly. Because of the potential danger of any lab exercise, you will follow the highest standards of conduct. Misbehavior can result in dismissal from the lab.

When we begin the planning process with a sample document, it can be helpful to identify the document's different elements. An **element** is a distinct object in the document, like a paragraph, a heading, or the page's title. Even the whole document can be considered an element. Stephen's handout includes several elements. A heading prominently displays his name, and beneath the heading, the document contains a photo and a horizontal line. The handout includes a brief introductory paragraph followed by two main sections: Chemistry Classes and Class Policies. The Chemistry Classes section lists the three classes he teaches. In the Class Policies section, three smaller headings list and describe his policies on Grading, Appointments, and Safety. We also want to take note of formatting features, such as text displayed in **boldfaced** font, and *italicized* text. As you recreate Stephen's handout as a Web page, you should periodically refer to Figure 1-5.

Marking Elements with Tags

The core building block of HTML is the **tag**, which marks each element in a document. Tags can be either two-sided or one-sided. A **two-sided tag** is a tag that contains some document content. The general syntax for a two-sided tag is:

```
<element>content</element>
```

where *element* is the name of the HTML element and *content* is any content it contains. For example, the following code marks the text, “Robert Service High School”, with the `<p>` tag:

```
<p>Robert Service High School</p>
```

As you’ll learn later, this indicates that a browser should treat “Robert Service High School” as a paragraph element.

In this book, the term “element” refers to an object in a Web page, and “tag” refers to the HTML code that creates the object. Thus we would say that we can create a `p` element in a Web page by inserting a `<p>` tag into the HTML file.

A two-sided tag’s **opening tag** (`<p>`) and **closing tag** (`</p>`) should completely enclose its content. Earlier versions of HTML allowed designers to omit a closing tag if the surrounding code clearly indicated the tag’s content, but this practice is no longer recommended. XHTML requires both an opening and closing tag.

HTML allows you to enter element names in either uppercase or lowercase letters. Thus you can type either `<p>` or `<P>` to indicate a paragraph. However, since XHTML strictly requires lowercase tag names, we will follow that convention here, and strongly recommend that you do likewise so that your Web pages will be consistent with current and future standards.

Unlike a two-sided tag, a **one-sided tag** contains no content. The general syntax for a one-sided tag is.

```
<element />
```

where *element* is once again the element name. HTML also allows you to enter one-sided tags using the syntax `<element>` (omitting the space and closing slash). However, XHTML does not support this form, so we once again strongly recommend that you include the space and the closing slash at all times. Elements that employ one-sided tags are called **empty elements** since they contain no content. One example of an empty element is a line break, which forces the browser to display the next set of characters on a new line. To create a line break you would use the one-sided tag:

```
<br />
```

Reference Window

Unless a Reference Window item is labeled “Deprecated,” it is compliant with XHTML 1.0 standards.

Inserting Two-Sided and One-Sided Tags

- To create a two-sided tag, use the syntax:
`<element>content</element>`
 where *element* is the name of the HTML and *content* is any content it contains. Element names should be lowercase.
- To create a one-sided tag, use the syntax:
`<element />`

Deprecated

- Many browsers also accept one-sided tags written as:
`<element>`
 This syntax is not recommended because it is not supported by XHTML and will probably not be supported by future browsers.
- Some browsers allow uppercase element names. This technique is not recommended because it is not supported by XHTML and probably not by future browsers.

A third type of tag is the **comment tag**, which you can use to add notes to your HTML code. While comments are not required and are not displayed or used by the Web browser, they are useful in documenting your HTML code for yourself and others. The syntax of the comment tag is:

```
<!-- comment -->
```

where *comment* is the text of your note. The following is an example of a comment tag that could describe the page you'll be creating for Stephen Dubé:

```
<!-- Chemistry page created for Robert Service High School -->
```

A comment can also be spread over several lines as follows:

```
<!-- Chemistry Class Web Page
      Created for Robert Service High School
-->
```

Reference Window

Inserting a Comment

- To insert a comment anywhere within your HTML file, enter:
`<!-- comment -->`
 where *comment* is the text of your comment. Comments can extend over several lines.

White Space and HTML

The ability to extend a comment over several lines is not unique to the comment tag. You can do this with any tag. As simple text files, HTML documents are composed of text characters and **white space**—the blank spaces, tabs and line breaks within the file. HTML treats each occurrence of white space as a single blank space. Thus, as far as HTML is concerned, there is no difference between a blank space, a tab, or a line break. When a

browser encounters consecutive occurrences of white space, it collapses them into a single occurrence. The following code samples are equivalent as far as HTML is concerned:

```
<p>This is an example of White Space</p>
<p>This is an example of   White   Space</p>
<p>This is an example
  of   White
    Space</p>
```

Even though browsers ignore extra white space, you can use it to make your HTML documents more readable—for example, by indenting lines or by separating blocks of code from one another.

Element Attributes

Many tags contain **attributes** that control the behavior, and in some cases the appearance, of elements in the page. You insert attributes within the tag brackets using the syntax

```
<element attribute1="value1" attribute2="value2" .../>
```

for one-sided tags, and the syntax

```
<element attribute1="value1" attribute2="value2" ...>content</element>
```

for two-sided tags, where *attribute1*, *attribute2*, and so forth are the names of the attributes, and *value1*, *value2*, etc. are the values associated with those attributes. For example, you can identify an individual element using the *id* attribute. The following code assigns the *id* value of “title” to the paragraph “Robert Service High School”, distinguishing it from other paragraphs in the document.

```
<p id="title">Robert Service High School</p>
```

You’ll learn more about the *id* attribute in the next tutorial.

You can list attributes in any order, but you must separate them from one another with white space. As with element names, you should enter attribute names in lowercase letters. In addition, you should enclose attribute values within quotation marks. While many browsers still accept attribute values without quotation marks, you can ensure maximum compatibility by always including them. XHTML requires quotation marks for all attribute values.

Inserting Attributes

- To add attributes to an element, insert the following into the element’s opening tag:
`attribute1="value1" attribute2="value2"`
where *attribute1*, *attribute2*, and so forth are the names of the attributes, and *value1*, *value2*, etc. are the values associated with each attribute.

Deprecated

- Some browsers accept attributes without quotation marks, as well as attribute names in uppercase. This syntax is not recommended because it is not supported by XHTML or will probably not be supported by future browsers.

Reference Window

The Structure of an HTML File

Now that we've studied the general syntax of HTML tags, we'll create our first HTML document. The most fundamental element is the HTML document itself. We mark this element using the two-sided `<html>` tag as follows:

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

The opening `<html>` tag marks the start of an HTML document, and the closing `</html>` tag tells a browser when it has reached the end of that HTML document. Anything between these two tags makes up the content of the document, including all other elements, text, and comments.

An HTML document is divided into two sections: the head and the body. The **head element** contains information about the document—for example, the document's title, or keywords that a search engine on the Web might use to identify this document for other users. The content of the head element is not displayed within the Web page, but Web browsers may use it in other ways; for example, Web browsers usually display a document's title in the title bar.

The **body element** contains all of the content to be displayed in the Web page. It can also contain code that tells the browser how to render that content.

To mark the head and body elements, you use the `<head>` and `<body>` tags as follows:

```
<html>

<head>
</head>

<body>
</body>

</html>
```

Note that the body element is placed after the head element.

The first thing we'll add to the document's head is the page title, also known as the **title element**. A given document can include only one title element. You create a title by inserting the two-sided `<title>` tag within the document's head. Since Stephen wants to give his page the title, "Mr. Dube's Chemistry Classes", our HTML code now looks like this:

```
<html>

<head>
<title>Mr. Dube's Chemistry Classes</title>
</head>

<body>
</body>

</html>
```

The technique of placing one element within another is called **nesting**. When one element contains another, you must close the inside element before closing the outside element, as shown in the code above. It would *not* be correct to close the outside element before closing the inside one, as in the following code sample:

```
<head><title>Mr. Dube's Chemistry Classes</head></title>
```

Now that you've seen how to insert HTML tags, let's start creating the chemistry Web page. In addition to the above code, we'll also add a comment that specifies the page's purpose and author, as well as the current date.

To create an HTML file:

1. Ensure that you can access your data files from your file server, CD-ROM, or floppy disk drive.
Trouble? If you don't have access to your data files, talk to your instructor. See the Read This Before You Begin page at the beginning of the tutorials for further instructions.

2. Create a new document with a text editor.

If you don't know how to locate, start, or use the text editor on your system, ask your instructor or technical support person for help.

3. Type the following lines of code in your document. Press the **Enter** key after each line. Press the **Enter** key twice for a blank line between lines of code. Insert your name in place of the text *your name* and the current date in place of *the date*.

```
<html>

<head>
<!-- Chemistry Classes Web Page
      Author: your name
      Date:   the date
-->
<title>Mr. Dube's Chemistry Classes</title>
</head>

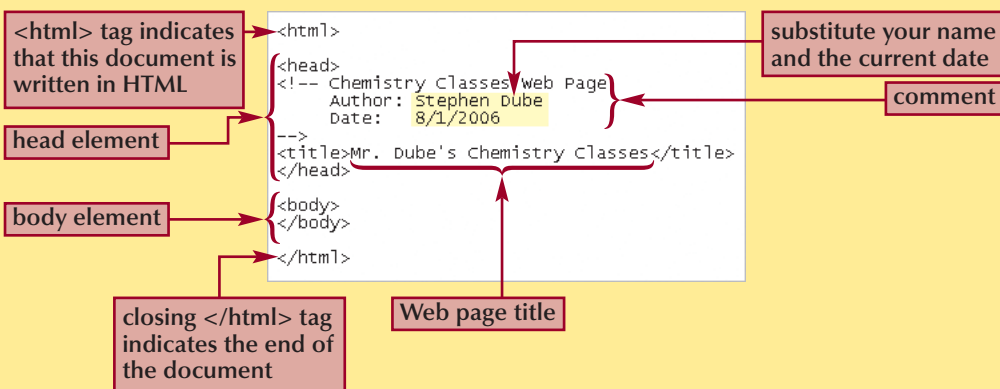
<body>
</body>

</html>
```

4. Using your text editor, save the file as **chem.htm** in the tutorial.01/tutorial folder where your Data Files are stored, but do not close your text editor. The text you typed should look similar to the text displayed in Figure 1-6.

Initial HTML code in chem.htm

Figure 1-6



Trouble? If you don't know where to save your Data Files, ask your instructor or technical support person for assistance.

Trouble? Don't worry if your screen doesn't look exactly like Figure 1-6. The figures show the Windows Notepad text editor. Your text may look different. Take the time to ensure that you entered the text correctly.

Trouble? If you are using the Windows Notepad text editor to create your HTML file, make sure you don't save the file with the extension .txt, which is the default for Notepad. Instead, make sure you save the file with the file extension .htm or .html. Using an invalid

file extension may make the file unreadable to Web browsers, which require .htm or .html as the file extension.

Trouble? If you are using Microsoft Word as your text editor, be sure to save your files as Web page files and not as Word documents.

Note that the extra space before and after the <body> tags is also not required, but it makes your file easier to read, especially as you add more code to the file.

Displaying an HTML File

As you continue adding to Stephen's Web page, you should occasionally view the page with your Web browser to verify that the file contains no syntax errors or other problems. You may even want to view the results using different browsers to check for compatibility. The steps and figures that follow use the Internet Explorer browser to display Stephen's page as you develop it. If you are using a different browser, ask your instructor how to view local files (those stored on your own computer rather than on the Web).

To view Stephen's Web page:

1. Start your browser. You do not need to be connected to the Internet to view local files stored on your computer.

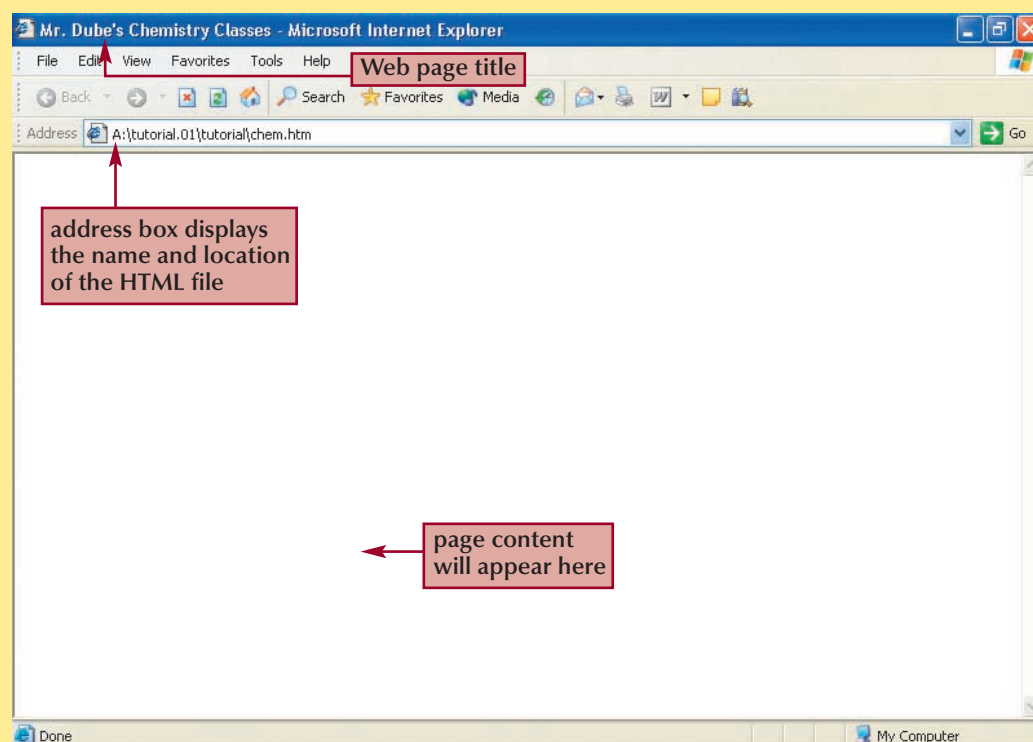
Trouble? If you try to start your browser and are not connected to the Internet, you might get a warning message. Click OK to ignore the message and continue.

2. After your browser loads its home page, open the **chem.htm** file that you saved in the tutorial.01/tutorial folder.

Your browser displays the chemistry Web page, as shown in Figure 1-7. Note that the page title appears in the browser's title bar and not on the page itself.

Figure 1-7

Initial Web page viewed in Internet Explorer



Trouble? To open a file in most browsers, click File on the menu bar, click Open, and click the Browse button to locate the file.

Trouble? Depending on the browser you're using, you may have to use a different command to open the file from your Data Files. Talk to your instructor or technical support person to find out how to open the file.

Trouble? If your browser displays something different, compare the code in your file to the code shown in Figure 1-6 and correct any errors. Save your changes, then return to your browser and click the Refresh or Reload button to view the new version of your Web page. So far, you have only entered a title for the Web page, which is why the main content area of the Web page is blank.

Working with Block-Level Elements

Now that you have created the basic structure of your document, you can start inserting the page's content. In a Web page, most content is marked as either a block-level element or an inline element. A **block-level element** contains content displayed in a separate section within the page, setting it off from other blocks. Paragraphs and headings are examples of block-level elements. An **inline element** is part of the same block as its surrounding content—for example, individual words or phrases within a paragraph.

Stephen's Web page includes several block-level elements. You will start adding page content by inserting the headings. You need to create a heading for the entire page and headings for each of two sections: Chemistry Classes and Class Policies. The Class Policies section includes three additional subheadings: Grading, Appointments, and Safety. You can mark all of these headings with HTML heading tags.

Creating Headings

HTML supports six heading elements, numbered h1 through h6. The h1 heading is reserved for the largest and most prominent headings, while the h6 element indicates a minor heading. The syntax for inserting a heading element is

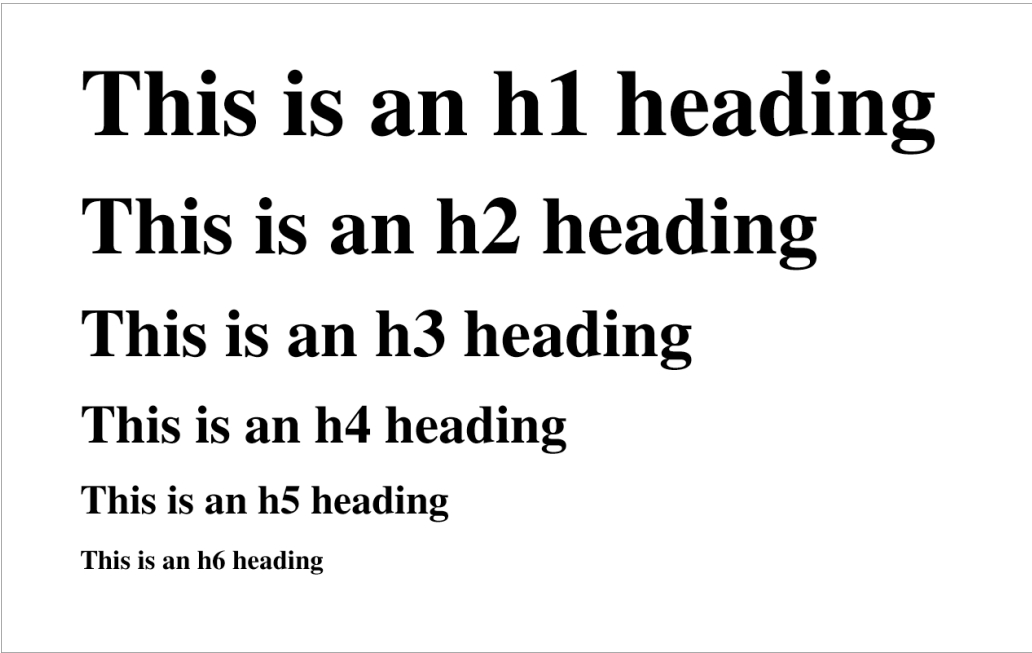
```
<hy>content</hy>
```

where *y* is a heading number 1 through 6 and *content* is the content of the heading.

Figure 1-8 illustrates the general appearance of the six heading elements. Your browser might use slightly different fonts and sizes.

Figure 1-8

HTML headings



Reference Window

Inserting a Heading

- To define a heading, use the syntax
`<hy>content</hy>`
where *y* is a heading number 1 through 6, and *content* is the content of the heading.

Inserting an Inline Style

By default, the contents of a heading are aligned with the left margin of the page. You notice that some of the headings in Stephen’s handout are centered. To use styles to control the appearance of an element, such as text alignment, you use the style attribute. The syntax for inserting the style attribute into an HTML tag is

```
<element style="style1: value1; style2: value2; style3: value3; ...">
```

where *element* is the element’s name, *style1*, *style2*, *style3* and so forth are the names of styles, and *value1*, *value2*, *value3* and so on are the values associated with those styles. Styles specified as attributes in a tag are also referred to as **inline styles**. As you proceed in your study of HTML, you’ll learn more about different styles and the many ways to apply them. For now, we’ll focus on the text-align style.

Reference Window

Inserting an Inline Style

- To add an inline style to an element, insert the following attribute into the element’s tag:
`style="style1: value1; style2: value2; style3: value3; ..."`
where *style1*, *style2*, *style3*, and so forth are the names of the styles, and *value1*, *value2*, *value3*, etc. are the values associated with those styles.

Applying the Text-Align Style

The text-align style tells the browser how to horizontally align the contents of an element. The style has four values: left, right, center, and justify; the value "justify" tells a browser to spread the content to touch both the left and right margins of the element. To display the text "Chemistry Class" as a centered h1 heading, you would use the following code:

```
<h1 style="text-align: center">Chemistry Class</h1>
```

Most browsers also support the align attribute. Thus, you could also write

```
<h1 align="center">Chemistry Class</h1>
```

However, because the align attribute is a deprecated feature of HTML, you should probably not use it unless you need to provide backward-compatibility with older browsers. HTML attributes such as the align attribute are known as **presentational attributes**, meaning that they specify exactly how the browser should render an element. Remember that one of the goals of HTML is to separate content from design. HTML should inform the browser about the content of the document, but you should use styles to inform the browser how to render that content. For this reason, almost all presentational attributes have been deprecated in favor of styles.

Aligning the Contents of an Element

- To horizontally align the contents of an element, use the style:
`text-align: align`
 where *align* is left, right, center, or justify.

Deprecated

- You can also align the contents of an element by adding the following attribute to the element's tag:
`align="align"`
 where *align* is left, right, center, or justify. Not all elements support the align attribute. It is often used with paragraphs and headings.

Reference Window

To add headings to the chemistry file:

- Using your text editor, open **chem.htm**, if it is not currently open.
- Place the insertion point after the <body> tag, press the **Enter** key to move to the next line, and then type the following lines of code:

```
<h1 style="text-align: center">Mr. Dube's Chemistry Classes</h1>
<h2 style="text-align: center">at Robert Service High School</h2>
<h2>Chemistry Classes</h2>
<h2>Class Policies</h2>
<h3>Grading</h3>
<h3>Appointments</h3>
<h3>Safety</h3>
```

Figure 1-9 displays the revised code. To make it easier for you to follow the changes to the HTML file, new and modified text in the figures is highlighted in red. This will not be the case in your own text files.

Figure 1-9

Entering heading elements

```

<html>
<head>
<!-- Chemistry Classes web Page
      Author: Stephen Dube
      Date: 8/1/2006
-->
<title>Mr. Dube's Chemistry Classes</title>
</head>

<body>
<h1 style="text-align:center">Mr. Dube's Chemistry Classes</h1>
<h2 style="text-align:center">at Robert Service High School</h2>
<h2>Chemistry Classes</h2>
<h2>Class Policies</h2>
<h3>Grading</h3>
<h3>Appointments</h3>
<h3>Safety</h3>
</body>
</html>

```

centered headings

3. Save your changes to **chem.htm**. You can leave your text editor open.

Now view the revised page in your Web browser.

To display the revised version of the chemistry page:

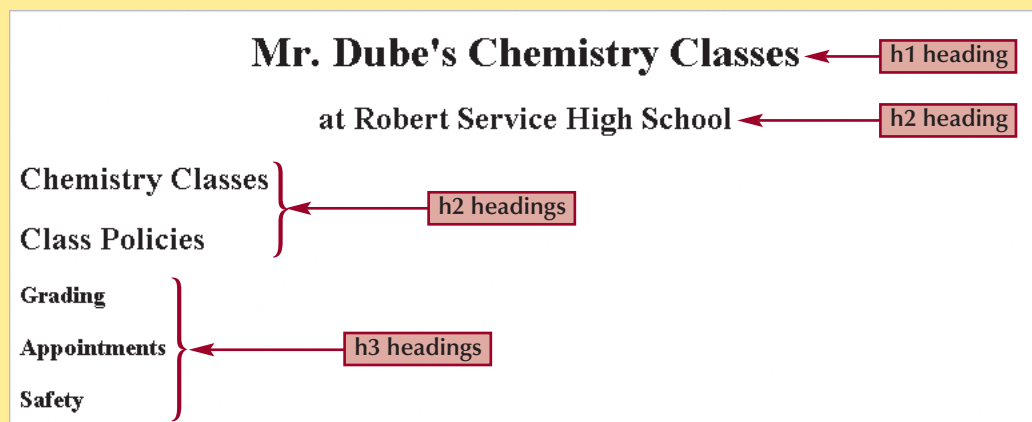
1. Return to your Web browser. Note that the previous version of chem.htm probably appears in the browser window.
2. To view the revised page, click **View** on the menu bar, and then click **Refresh**. If you are using a Netscape browser, you will need to click **View** and then click **Reload**.

Trouble? If you closed the browser or the file in the last set of steps, reopen your browser and the chem.htm file.

The updated Web page looks like Figure 1-10.

Figure 1-10

Headings as they appear in the browser



Creating Paragraphs

The next step is to enter text information for each section. As you saw earlier, you can insert a paragraph element using the `<p>` tag as follows:

```
<p>content</p>
```

where *content* is the content of the paragraph. When a browser encounters the opening `<p>` tag, it starts a new line with a blank space above it, separating the new paragraph from the preceding element. In earlier versions of HTML when standards were not firmly fixed, Web authors would often include only the opening `<p>` tag, omitting the closing tag entirely. While many browsers still allow this, your Web pages display more reliably if you consistently use the closing tag. Additionally, if you wish to write XHTML-compliant code then you must include the closing tag.

Creating a Paragraph

- To create a paragraph, use the syntax:

```
<p>content</p>
```

 where *content* is the content of the paragraph.

To enter paragraph text:

- Return to **chem.htm** in your text editor.
- Place the insertion point at the end of the line that creates the h2 heading, "at Robert Service High School", and press the **Enter** key to create a blank line.
- Type the following text:

```
<p>Welcome to the Robert Service High School Chemistry Web page.  
Here you'll learn more about our chemistry classes and our policies.</p>
```

- Press the **Enter** key to insert a blank line below the paragraph.

Note that a blank line is not required for the text to display correctly in your browser. However, adding this space makes it easier for you to read the code by separating the first paragraph from the heading that follows. See Figure 1-11.

Inserting the first paragraph

Figure 1-11

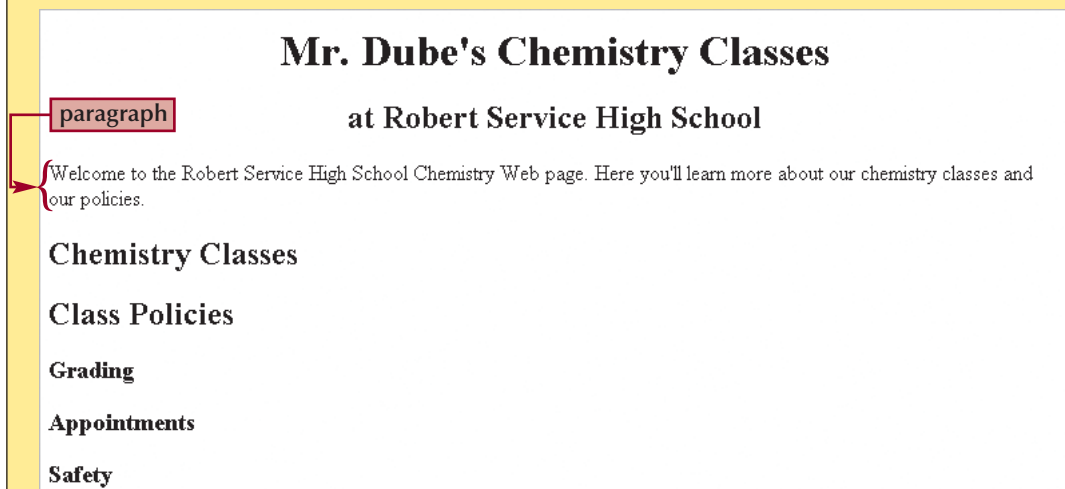
```
<body>  
<h1 style="text-align: center">Mr. Dube's Chemistry Classes</h1>  
<h2 style="text-align: center">at Robert Service High School</h2>  
<p>Welcome to the Robert Service High School Chemistry Web page.  
Here you'll learn more about our chemistry classes and our  
policies.</p>  
  
<h2>Chemistry Classes</h2>  
<h2>Class Policies</h2>  
<h3>Grading</h3>  
<h3>Appointments</h3>  
<h3>Safety</h3>  
</body>
```

paragraph

- Save your changes to **chem.htm**.
- Using your Web browser, refresh or reload **chem.htm** to view the new paragraph. See Figure 1-12.

Figure 1-12

First paragraph in the browser



Next you need to add the three paragraphs under the Grading heading and the single paragraphs under both Appointments and Safety.

To enter the remaining paragraphs:

1. Return to the **chem.htm** file in your text editor.
2. Below the h3 heading "Grading," insert the following three paragraphs:

```
<p>Homework: Homework is worth 5 to 10 points and will be given
daily. A quiz consisting of 1 or 2 homework problems from the
previous week may be given in place of homework.</p>
```

```
<p>Tests and Quizzes: Quizzes are worth 10 to 25 points and will be
given at least once a month. Tests are worth up to 100 points and
will be given three times each quarter.</p>
```

```
<p>Labs: Labs are worth 10 to 30 points and will be graded on
safety, participation, and write-up. Reports should be neatly
written or typed. Research projects will also be assigned throughout
the year.</p>
```

Note that because of how HTML handles white space, you can insert additional blank lines between the paragraphs or line breaks within paragraphs to make your code easier to read. This does not affect how a browser renders the paragraphs.

3. Below the h3 heading "Appointments," insert the following paragraph:

```
<p>You can meet with Mr. Dube before or after school or during most
lunch hours in room H113. Do not hesitate to ask for help! It can be
very hard to catch up if you fall behind.</p>
```

4. Below the h3 heading "Safety," insert the following paragraph:

```
<p>Labs are completed weekly. Because of the potential danger of any
lab exercise, you will follow the highest standards of conduct.
Misbehavior can result in dismissal from the lab.</p>
```

Figure 1-13 shows the new code in the chem.htm file.

Inserting additional paragraphs

Figure 1-13

```

<h2>Chemistry Classes</h2>
<h2>Class Policies</h2>
<h3>Grading</h3>
<p>Homework: Homework is worth 5 to 10 points and will be given
daily. A quiz consisting of 1 or 2 homework problems from the
previous week may be given in place of homework.</p>

<p>Tests and Quizzes: Quizzes are worth 10 to 25 points and will be
given at least once a month. Tests are worth up to 100 points and
will be given three times each quarter.</p>

<p>Labs: Labs are worth 10 to 30 points and will be graded on safety,
participation, and write-up. Reports should be neatly written or
typed. Research projects will also be assigned throughout the
year.</p>

<h3>Appointments</h3>
<p>You can meet with Mr. Dube before or after school or during most
lunch hours in room H113. Do not hesitate to ask for help! It can be
very hard to catch up if you fall behind.</p>

<h3>Safety</h3>
<p>Labs are completed weekly. Because of the potential danger of any
lab exercise, you will follow the highest standards of conduct.
Misbehavior can result in dismissal from the lab.</p>

</body>

```

5. Save your changes to the file.

6. Return to your Web browser and refresh or reload **chem.htm** to view the new paragraphs. Figure 1-14 displays the revised version.

New paragraphs in the chemistry page

Figure 1-14

Mr. Dube's Chemistry Classes

at Robert Service High School

Welcome to the Robert Service High School Chemistry Web page. Here you'll learn more about our chemistry classes and our policies.

Chemistry Classes

Class Policies

Grading

Homework: Homework is worth 5 to 10 points and will be given daily. A quiz consisting of 1 or 2 homework problems from the previous week may be given in place of homework.

Tests and Quizzes: Quizzes are worth 10 to 25 points and will be given at least once a month. Tests are worth up to 100 points and will be given three times each quarter.

Labs: Labs are worth 10 to 30 points and will be graded on safety, participation, and write-up. Reports should be neatly written or typed. Research projects will also be assigned throughout the year.

Appointments

You can meet with Mr. Dube before or after school or during most lunch hours in room H113. Do not hesitate to ask for help! It can be very hard to catch up if you fall behind.

Safety

Labs are completed weekly. Because of the potential danger of any lab exercise, you will follow the highest standards of conduct. Misbehavior can result in dismissal from the lab.

Creating Lists

You still need to describe the three chemistry courses that the school offers. Rather than entering these in paragraph form, you'll use a list. HTML supports three kinds of lists: ordered, unordered, and definition.

Creating an Ordered List

You use an **ordered list** for items that must appear in a particular sequential order. You create an ordered list using the `ol` element in the following form:

```
<ol>
  <li>item1</li>
  <li>item2</li>
  ...
</ol>
```

where *item1*, *item2*, etc, are items in the list. Each `` tag marks the content for a single list item. For example, if Stephen wants to list the three chemistry classes from the least difficult to the most difficult, the HTML code could look as follows:

```
<ol>
  <li>Conceptual Chemistry</li>
  <li>Chemistry I</li>
  <li>Advanced Placement Chemistry</li>
</ol>
```

By default, browsers display ordered lists as a series of sequentially numbered items. Based on the preceding HTML code, Stephen's list would appear in the following form:

1. Conceptual Chemistry
2. Chemistry I
3. Advanced Placement Chemistry

Reference Window

Creating an Ordered List

- To create an ordered list, use the syntax:


```
<ol>
  <li>item1</li>
  <li>item2</li>
  ...
</ol>
```

 where *item1*, *item2*, etc. are items in the list.

Creating an Unordered List

To display a list in which the items do not need to occur in any special order, you would create an **unordered list**. The structures of ordered and unordered lists are the same, except that the contents of an unordered list are contained within a `` tag:

```
<ul>
  <li>item1</li>
  <li>item2</li>
  ...
</ul>
```

By default, the contents of an unordered list are displayed as bulleted items. Thus, the code

```
<ul>
  <li>Introductory course</li>
  <li>No algebra required</li>
</ul>
```

would be displayed by a browser as

- Introductory course
- No algebra required

Creating an Unordered List

- To create an unordered list, use the syntax:

```
<ul>
  <li>item1</li>
  <li>item2</li>
  ...
</ul>
```

where *item1*, *item2*, etc. are items in the list.

Creating a Nested List

One list can contain another. For example, it can sometimes be useful to combine two different types of lists, as in the following example:

1 Conceptual Chemistry

- Introductory course
- No algebra required

2 Chemistry I

- Introductory course
- Algebra required

3 Advanced Placement Chemistry

- Advanced course
- Requires an A or B in Chemistry I

You could create the preceding combination of ordered and unordered lists using the following HTML code:

```
<ol>
  <li>Conceptual Chemistry
    <ul>
      <li>Introductory course</li>
      <li>No algebra required</li>
    </ul>
  </li>
  <li>Chemistry I
    <ul>
      <li>Introductory course</li>
      <li>Algebra required</li>
    </ul>
  </li>
```

```

<li>Advanced Placement Chemistry
  <ul>
    <li>Advanced course</li>
    <li>Requires an A or B in Chemistry I</li>
  </ul>
</li>
</ol>

```

Note that some of the list items in this code contain lists themselves.

Applying a Style to a List

If you don't want your list items marked with either numbers or bullets, you can specify a different marker by applying the following style to either the ordered or unordered list:

`list-style-type: type`

where *type* is one of the markers listed in Figure 1-15.

Figure 1-15

List style types

List-Style-Type	Marker (s)
disc	•
circle	○
square	■
decimal	1, 2, 3, 4, ...
decimal-leading-zero	01, 02, 03, 04, ...
lower-roman	i, ii, iii, iv, ...
upper-roman	I, II, III, IV, ...
lower-alpha	a, b, c, d, ...
upper-alpha	A, B, C, D, ...
none	<i>no marker displayed</i>

For example, to create the following list:

- a. Conceptual Chemistry
- b. Chemistry I
- c. Advanced Placement Chemistry

you would enter the HTML code:

```

<ol style="list-style-type: lower-alpha">
  <li>Conceptual Chemistry</li>
  <li>Chemistry I</li>
  <li>Advanced Placement Chemistry</li>
</ol>

```

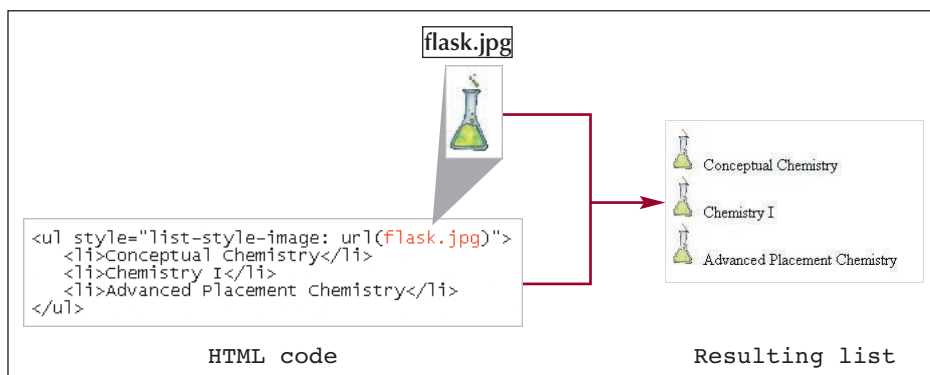
You can also substitute a graphic image for a list marker by using the style:

`list-style-image: url(file)`

where *file* is the name of an image file containing the marker. Figure 1-16 demonstrates how to use a graphic image named "flask.jpg" as a marker in a list.

Using a graphic list marker

Figure 1-16



In this style, URL stands for Uniform Resource Locator. A URL is the standard method for specifying the location of a document or resource on the Internet. You'll learn about URLs in the next tutorial.

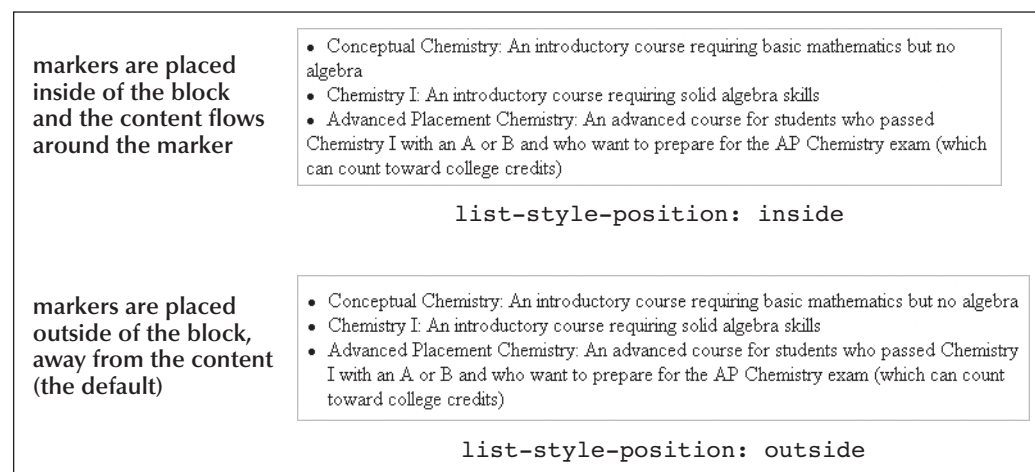
Each list item is itself a block-level element. By default, most browsers place each list marker outside of its corresponding block; however you can change this by using the following style:

`list-style-position: position`

where *position* is either "outside" (the default) or "inside". Placing the marker inside of the block causes the content of the list to flow around the marker (see Figure 1-17).

Defining the marker position

Figure 1-17



The three previous styles can be combined in the following single style:

`list-style: type url(file) position`

where *type* is one of the marker types, *file* is the location of a graphic file that can be used for a marker, and *position* is either "inside" or "outside". Text-based browsers use the *type* value, while graphical browsers use the graphic file. For example the tag

`<ul style='list-style: square url(flask.jpg) inside';>`

would create an unordered list with a square marker for text-based browsers and the flask.jpg image marker for graphical browsers. Whichever marker a browser uses appears inside of each list item.

For older browsers that don't support inline styles, you can use one of the presentational attributes that HTML provides for ordered and unordered lists. See the accompanying reference window for details.

Reference Window

Formatting a List

- To change the list marker, use the style:
`list-style-type: type`
where *type* is disc, circle, square, decimal, decimal-leading-zero, lower-roman, upper-roman, lower-alpha, upper-alpha, or none.
- To use a graphic image in place of a marker, use the style:
`list-style-image: url(file)`
where *file* is the name of the image file.
- To specify the location of the list marker, use the style:
`list-style-position: position`
where *position* is either inside or outside.

Deprecated

- You can also change the list marker by adding the following attribute to the `` or `` tag:
`type="type"`
For unordered lists, the *type* value can be circle, square, or disc. For ordered lists, *type* values are a (for lower-alpha), A (for upper-alpha), i (for lower-roman), I (for upper-roman), and 1 (for numeric).
- For ordered lists, you can specify the starting number of the first item in the list using the attribute:
`start="number"`
where *number* is the starting value. A start number of "2" starts the list with the second marker. For a numeric marker, this is the value '2', while for an alphabetical list, this is the letter "b" or "B".

Creating a Definition List

HTML supports a third list element, the **definition list**, which contains a list of definition terms, each followed by a definition description. The syntax for creating a definition list is:

```
<dl>
  <dt>term1</dt>
  <dd>definition1</dd>
  <dt>term2</dt>
  <dd>definition2</dd>
  ...
</dl>
```

where *term1*, *term2*, etc. are the terms in the list, and *definition1*, *definition2*, etc. are the definitions of the terms.

If Stephen wanted to create a list of his classes and briefly describe each one, he could use a definition list. The code might look as follows:

```
<dl>
  <dt>Conceptual Chemistry</dt>
  <dd>An introductory course requiring basic mathematics but no
    algebra</dd>
  <dt>Chemistry I</dt>
  <dd>An introductory course requiring solid algebra skills</dd>
  <dt>Advanced Placement Chemistry</dt>
  <dd>An advanced course for students who passed Chemistry I with an A
    or B and who want to prepare for the AP Chemistry exam (which can
    count toward college credits)</dd>
</dl>
```

Web browsers typically display the definition description below the definition term and slightly indented. Most browsers would display the definition list code shown above as:

```
Conceptual Chemistry
  An introductory course requiring basic mathematics but no algebra
Chemistry I
  An introductory course requiring solid algebra skills
Advanced Placement Chemistry
  An advanced course for students who passed Chemistry I with an A or B and who
  want to prepare for the AP Chemistry exam (which can count toward college credits)
```

Creating a Definition List

Reference Window

- To create a definition list, use the syntax:

```
<dl>
  <dt>term1</dt>
  <dd>definition1</dd>
  <dt>term2</dt>
  <dd>definition2</dd>
  ...
</dl>
```

where *term1*, *term2*, etc. are the terms in the list, and *definition1*, *definition2*, etc. are the definitions of the terms.

Now that you've seen how you can use HTML to create different kinds of lists, you'll add an unordered list of classes to the chemistry Web page. You decide to use a square marker for each item. By default, the marker is placed outside of the block.

To add an unordered list to the chemistry page:

- Return to the **chem.htm** file in your text editor.
- Below the line "`<h2>Chemistry Classes</h2>`" insert the following code, as shown in Figure 1-18.

```
<ul style="list-style-type: square">
  <li>Conceptual Chemistry: An introductory course, requiring basic
    math but no algebra</li>
  <li>Chemistry I: An introductory course, requiring solid algebra
    skills</li>
```

```
<li>Advanced Placement Chemistry: An advanced course requiring a
    grade of A or B in Chemistry I</li>
</ul>
```

You can indent the lines using either tabs or blank spaces. Remember that indenting has no effect on the appearance of the list in a browser.

Figure 1-18

Inserting an unordered list

```
<body>
<h1 style="text-align: center">Mr. Dube's Chemistry Classes</h1>
<h2 style="text-align: center">at Robert Service High School</h2>
<p>Welcome to the Robert Service High School Chemistry Web page.
    Here you'll learn more about our chemistry classes and our
    policies.</p>

<h2>Chemistry Classes</h2>
<ul style="list-style-type: square">
  <li>Conceptual Chemistry: An introductory course, requiring basic
    math but no algebra</li>
  <li>Chemistry I: An introductory course, requiring solid algebra
    skills</li>
  <li>Advanced Placement Chemistry: An advanced course requiring a
    grade of A or B in Chemistry I</li>
</ul>
```

3. Save your changes to the file.

4. Using your Web browser, refresh or reload **chem.htm**. Figure 1-19 shows the latest version of the page.

Figure 1-19

An unordered list in the browser

Mr. Dube's Chemistry Classes

at Robert Service High School

Welcome to the Robert Service High School Chemistry Web page. Here you'll learn more about our chemistry classes and our policies.

Chemistry Classes

- Conceptual Chemistry: An introductory course, requiring basic math but no algebra
- Chemistry I: An introductory course, requiring solid algebra skills
- Advanced Placement Chemistry: An advanced course requiring a grade of A or B in Chemistry I

Using Other Block-Level Elements

HTML supports several other block-level elements that you may find useful in your Web pages. For example HTML supports the address element to indicate contact information. Most browsers display an address element in an italicized font. You can indicate long quoted passages by applying the blockquote element. A browser encountering this element typically indents the quoted text. Figure 1-20 describes additional block-level elements and shows how they look in most browsers.

Block-level elements

Figure 1-20

Block Level Element	Description	Visual Appearance
<address> ... </address>	Identifies contact information	<i>Italicized text</i>
<blockquote> ... </blockquote>	Identifies a long quotation	Plain text indented from the left and right
<center> ... </center>	Centers content horizontally within a block. Deprecated	Plain text, centered
<dd> ... </dd>	Identifies a definition description	Plain text
<div> ... </div>	Identifies a multicolumn directory list; superseded by the ul element. Deprecated	Plain text
<dl> ... </dl>	Identifies a generic block-level element	Plain text
<dt> ... </dt>	Identifies a definition list	Plain text
<dt> ... </dt>	Identifies a definition term	Plain text
<h1> ... </h1>	Identifies a heading, where y is a value from 1 to 6	Boldfaced text of various font sizes
 ... 	Identifies a list item in an ordered or unordered list	Bulleted or numbered text
<menu> ... </menu>	Identifies a single column menu list; superseded by the ul element. Deprecated	Plain text
 ... 	Identifies an ordered list	Plain text
<p> ... </p>	Identifies a paragraph	Plain text
<pre> ... </pre>	Retains all white space and special characters in preformatted text	<code>Fixed width text</code>
 ... 	Identifies an unordered list	Plain text

You'll have a chance to apply some of these other block-level elements in the case problems at the end of the tutorial.

Working with Inline Elements

As you compare your Web page with Figure 1-5, you notice that Stephen's original hand-out contains several words formatted in boldface or italics. In order to apply this formatting to the chemistry Web page, we need to use a set of HTML's inline elements known as **character formatting elements**, which allow us to format text characters. Figure 1-21 describes some character formatting elements that HTML supports.

Figure 1-21

Inline elements

Inline Element	Identifies	Visual Appearance
<abbr> ... </abbr>	an abbreviation	Plain text
<acronym> .. </acronym>	an acronym	Plain text
 ... 	boldfaced text	Boldfaced text
<big> ... </big>	big text	Larger text
<cite> ... </cite>	a citation	<i>Italicized text</i>
<code> ... </code>	program code text	Fixed width text
 ... 	deleted text	Strikethrough text
<dfn> ... </dfn>	a definition term	<i>Italicized text</i>
 ... 	emphasized content	<i>Italicized text</i>
<i> ... </i>	italicized text	<i>Italicized text</i>
<ins> ... </ins>	inserted text	<u>Underlined text</u>
<kbd> ... </kbd>	keyboard-style text	Fixed width text
<q> ... </q>	quoted text	"Quoted text"
<s> ... </s>	strikethrough text. Deprecated	Strikethrough text
<samp> ... </samp>	sample computer code text	Fixed width text
<small> ... </small>	small text	Smaller text
 ... 	a generic inline element	Plain text
<strike> ... </strike>	strikethrough text. Deprecated	Strikethrough text
 ... 	strongly emphasized content	Boldfaced text
_{...}	subscripted text	Subscripted text
^{...}	superscripted	Superscripted text
<tt> ... </tt>	teletype text	Fixed width text
<u> ... </u>	underlined text. Deprecated	<u>Underlined text</u>
<var> ... </var>	programming variables	<i>Italicized text</i>

For example, if you wanted to mark a section of boldfaced text within a paragraph, you could enter the following HTML code:

```
<p>Welcome to our <b>Chemistry Classes</b></p>
```

resulting in the following paragraph in the Web page:

Welcome to our **Chemistry Classes**

To mark those same words as italicized text, you would use

```
<p>Welcome to our <i>Chemistry Classes</i></p>
```

If you want the phrase "Chemistry Classes" to be marked as both boldface and italics, you could use the code

```
<p>Welcome to our <b><i>Chemistry Classes</i></b></p>
```

which would be displayed as

Welcome to our ***Chemistry Classes***

Stephen's handout requires the use of character tags only in the Grading section, where he wants to highlight the name of each grading topic. You decide to use a combination of the and <i> tags to display the key words in bold and italics.

To add character tags to the chemistry file:

1. Return to the **chem.htm** file in your text editor.
2. Type the `<i>` and `` tags around the key words in the Grading section of the handout as follows:

```
<p><i><b>Homework</b></i>: Homework is worth ...
<p><i><b>Tests and Quizzes</b></i>: Quizzes are worth ...
<p><i><b>Labs</b></i>: Labs are worth ...
<i><b>Research projects</b></i> will also be assigned ...
```

See Figure 1-22.

Inserting boldfaced and italicized text

Figure 1-22

```
<h2>Class Policies</h2>
<h3>Grading</h3>
<p><i><b>Homework</b></i>: Homework is worth 5 to 10 points and will be given
daily. A quiz consisting of 1 or 2 homework problems from the
previous week may be given in place of homework.</p>
<p><i><b>Tests and Quizzes</b></i>: Quizzes are worth 10 to 25 points and will be
given at least once a month. Tests are worth up to 100 points and
will be given three times each quarter.</p>
<p><i><b>Labs</b></i>: Labs are worth 10 to 30 points and will be graded on safety,
participation, and write-up. Reports should be neatly written or
typed. <i><b>Research projects</b></i> will also be assigned throughout the
year.</p>
```

3. Save your changes to the file.
4. Using your Web browser, refresh or reload **chem.htm**. The updated Grading section of your page should look like Figure 1-23.

Displaying boldfaced and italicized text

Figure 1-23

Class Policies

Grading

Homework: Homework is worth 5 to 10 points and will be given daily. A quiz consisting of 1 or 2 homework problems from the previous week may be given in place of homework.

Tests and Quizzes: Quizzes are worth 10 to 25 points and will be given at least once a month. Tests are worth up to 100 points and will be given three times each quarter.

Labs: Labs are worth 10 to 30 points and will be graded on safety, participation, and write-up. Reports should be neatly written or typed. **Research projects** will also be assigned throughout the year.

5. If you are continuing to Session 1.3, leave your text editor and browser open. Otherwise you can close them at this time.

Understanding Logical and Physical Elements

As you examine the tag list in Figure 1-21, you may notice some overlap in the way the content appears in the browser. For example, if you want to display italicized text you could use the `<dfn>`, ``, `<i>`, or `<var>` tags, or if you want to italicize an entire block of text, you could use the `<address>` tag. Why does HTML support so many different ways of formatting text?

While HTML can control the way text appears, the language's main purpose is to create a structure for a document's contents. While some browsers render different elements

in the same way, it's important to distinguish between how a browser displays an element and the element's purpose in the document. For this reason, page elements are therefore often organized into two types: logical elements and physical elements. A **logical element**, which might be created with tags like `<cite>` or `<code>`, describes the nature of the enclosed content, but not necessarily how that content should appear. A **physical element**, on the other hand, which you might create with tags like `` or `<i>`, describes how content should appear but doesn't indicate the content's nature.

While it can be tempting to use logical and physical elements interchangeably, your Web pages benefit in several ways when you respect the distinction. For one, different browsers can and do display logical elements differently. For example, both Netscape's browser and Internet Explorer display text created with the `<cite>` tag in italics, but the text-based browser Lynx displays citation text using a fixed width font. Some browsers, like those that display Braille or convert HTML code into speech, don't even display formatted text. For example, an aural browser might increase the volume when it encounters emphasized text. In addition, Web programmers can also use logical elements to extract a page's content. For example, a program could automatically generate a bibliography from all of the citations listed within a Web site.

In general, you should use a logical element that accurately describes the enclosed content whenever possible, and use physical elements only for general content.

You have finished inserting the text of the chemistry Web page. In the next session, you will add additional elements to the page, including an image and a graphical line.

Review

Session 1.2 Quick Check

1. What are the two main sections of an HTML file?
2. What are empty elements?
3. What is the syntax for creating a centered heading 1 of the text, "Chemistry Classes"?
4. What is the difference between a block-level element and an inline element?
5. If you want to create an extra blank line between paragraphs, why can't you simply add a blank line in the HTML file?
6. What are presentational attributes? When should you use them?
7. What attribute would you add to the `` tag to display uppercase Roman numerals?
8. What attribute would you add to the `` tag to display the ball.gif image mark on the inside of the item block?
9. What is the difference between a logical element and a physical element? Which would probably be more appropriate for a non-visual browser, such as a Braille browser?

Session 1.3

Working with Empty Elements

In the last session, you worked exclusively with two-sided tags to create content for the chemistry Web page. Stephen also wants to add images and horizontal lines to the page. To create these objects, you use empty elements. We'll start by inserting a graphic into the Web page.

Inserting a Graphic

To display a graphic, you insert an inline image into the page. An **inline image**, which is another example of an inline element, displays a graphic image located in a separate file within the contents of a block-level element. While a variety of file formats are available

for image files, inline images are most widely viewable in two formats: GIF (Graphics Interchange Format) or JPEG (Joint Photographic Experts Group). You can use an image editing application such as Adobe Photoshop to convert images to either the GIF or JPEG file format. To create an inline image, you use the `img` element as follows:

```

```

where *file* is the name of the image file and *text* is an alternative text string that browsers can use in place of an image. It's important to include an `alt` attribute in all of your inline images. Some users run browsers that do not display images, meaning that you need to duplicate with text any information that an image conveys. HTML does not require you to use an `alt` attribute with your inline images, but XHTML does.

Inserting an inline image

- To insert an inline image, use the tag:

```

```

 where *file* is the name of the image file and *text* is alternative text that browsers can display in place of the image.

If the image file is located in the same folder as the HTML file, you do not need to include any file location path information along with the filename. However, if the image file is located in another folder or on another computer, you need to include the full location path along with the filename in the `src` attribute. The next tutorial discusses folder paths and filenames in more detail. For now, you can assume that Stephen's image file is located in the same folder that contains the HTML file.

The image file that Stephen wants you to use in place of the page's main heading is named **dube.jpg** and is located in the tutorial.01/tutorial folder on your data disk (see Figure 1-24).

Image for the top of the chemistry page

Figure 1-24



Stephen wants you to center the image on the page. Since the `img` element is an inline element, it does not support an alignment attribute. In order to center it, we need to place it within a block-level element like a paragraph. We can then center the contents of the paragraph, which in this case consists only of the image.

To add Stephen's image to the Web page:

1. If necessary, use your text editor to reopen **chem.htm**.
2. Near the top of the file, select the two lines of code just below the `<body>` tag (from the `<h1>` opening tag to the `</h2>` closing tag), and then press the **Delete** key. This removes the first two headings from the document.

3. Insert the following code directly below the <body> tag (see Figure 1-25):

```
<p style="text-align: center">
  
</p>
```

Figure 1-25

Inserting an inline image

source of the
inline image

text displayed for
non-graphical browsers

```
<body>
<p style="text-align: center">
  
</p>

<p>welcome to the Robert Service High School Chemistry web page.
  Here you'll learn more about our chemistry classes and our
  policies.</p>
```

4. Save your changes to the file.

5. Open or refresh **chem.htm** in your Web browser. Figure 1-26 shows the placement of the image in the page.

Figure 1-26

Displaying an inline image

inline
image



Welcome to the Robert Service High School Chemistry Web page. Here you'll learn more about our chemistry classes and our policies.

Inserting Horizontal Lines

Stephen is pleased with the image's placement on the page. He would like you to add a horizontal line below the image, separating it from the page's text. To create a horizontal line, you use the one-sided tag

```
<hr />
```

To modify the line's size, you can use the styles

```
width: value; height: value
```

where *value* is a size measurement in pixels. A **pixel** is a dot on your computer screen that measures about 1/72" square. You can alternately specify a width value as a percentage of the page's width. The default width is 100% (the width of the Web page) and the default height is 2 pixels.

You can set a line's color using either of the two following styles:

```
color: color; background-color: color
```


where *color* is either the name of a color or an RGB color value. We'll study the issue of color in greater detail in a later tutorial. For now, know that browsers understand basic color names like red or blue or green.

While some browsers use the color style to assign a color to a horizontal line, other browsers use the background-color style. Therefore, if setting a line's color is an important aspect of your page's design, it's best to include both the color and background-color styles.

Inserting a Horizontal Line

Reference Window

- To insert a horizontal line, use the tag `<hr />`
- To change the color of the line, use the style `color: color; background-color: color` where *color* is either a recognized color name or an RGB color value.
- To change the width and height of the line, use the style `width: value; height: value` where *value* is the width or height of the line in pixels. You can also express the width value as a percentage of the page width. The default width is 100%, which is equal to the width of the Web page. The default height is usually 2 pixels.

Deprecated

- You can also format the appearance of your horizontal lines by adding the following attributes to the `<hr />` tag:
`align="align" color="color" size="value" width="value"`
 The align attribute specifies the alignment of the line on the page and can have the values left, right, or center (the default). The color attribute specifies the color of the line (Internet Explorer only). The size attribute specifies the height of the line in pixels. The width attribute specifies the width of the line in pixels.

For example, if Stephen wants to create a red horizontal line that is half the width of the page and 5 pixels high, he would enter the following tag into his HTML document:

```
<hr style="color: red; background-color: red; width: 50%; height: 5" />
```

The default rendering of a horizontal line is not standard across browsers. Typically the line extends across the complete width of the page at a height of 2 pixels. Some graphical browsers display the line in a solid black color, while others apply a chiseled or embossed effect. Text-based browsers display the line using dashes or underscores.

For the chemistry page, Stephen simply wants a red horizontal line, 2 pixels high. He'll let the line extend across the width of the page.

To add a horizontal line to the chemistry file:

1. Return to the **chem.htm** file in your text editor.
2. Below the paragraph containing the dube.jpg inline image, insert the following tag (see Figure 1-27):

```
<hr style="color: red; background-color: red; height: 2; width: 100%" />
```

Figure 1-27

Inserting a horizontal line

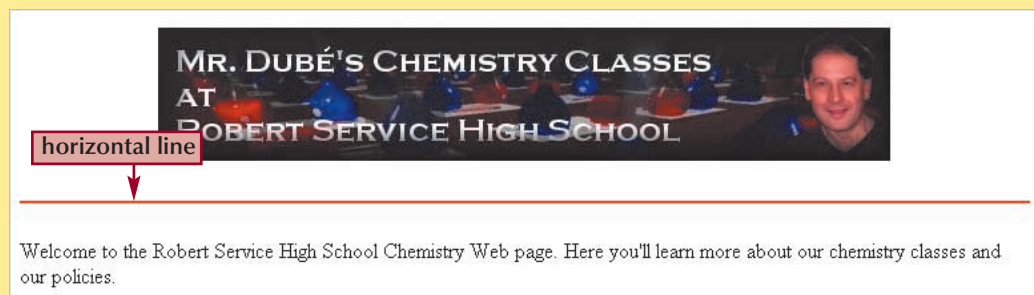
```
<body>
<p style="text-align:center">
  
</p>
<hr style="color:red;background-color:red;height:2;width:100%" />
<p>welcome to the Robert Service High School Chemistry Web page.
  Here you'll learn more about our chemistry classes and our
  policies.</p>
```

3. Save your changes to the file.

4. Using your Web browser, refresh or reload the **chem.htm** file. Figure 1-28 shows the new horizontal line.

Figure 1-28

Displaying a horizontal line



Most browsers still support several deprecated presentational attributes, which you can use in place of styles for your horizontal lines. See the “Inserting a Horizontal Line” reference window for more details.

Other Empty Elements

Other empty elements you may wish to use in your Web page include line breaks and meta elements. The `
` tag creates a line break, which starts a new line within a paragraph. For example, the following code

```
<br />
<br />
<br />
```

creates three consecutive line breaks. You can use the `
` tag to control the spacing of the different sections in your document.

Meta elements are placed in the document's head and contain information about the document that may be of use to programs that run on Web servers. You create a meta element using the one-sided `<meta />` tag as follows:

```
<meta name="text" content="text" scheme="text" http-equiv="text" />
```

where the name attribute specifies the name of a property for the page, the content attribute provides a property value, the scheme attribute provides the format of the property value, and the http-equiv attribute takes the place of the name attribute for some Web servers. For example the following `<meta />` tag stores the name of the Web page's author.

```
<meta name="author" content="Stephen Dube" />
```

Some Web sites, like Google, use search engines to create lists of Web pages devoted to particular topics. You can give extra weight to your Web page by including a description of the page and a list of keywords in `<meta />` tags at the top of the document.

```
<meta name="description" content="Chemistry Class Web page" />
<meta name="keywords" content="chemistry, school, Edmonton, science" />
```

Note that a document's head can contain several meta elements.

Working with Special Characters

Occasionally, you will want to include special characters in your Web page that do not appear on your keyboard. For example, a page might require mathematical symbols such as β or μ , or you might need to include the copyright symbol © to show that an image or text is copyrighted.

Stephen's last name uses an accented letter, "é". His name appears three times in his Web page: in the title at the top of the page, again in the alt text for the inline image, and finally in a paragraph on making an appointment.

HTML supports the use of character symbols that are identified by a code number or name. The syntax for creating a special character is:

`&code;`

where *code* is either a code name or a code number. Code numbers are preceded by a pound symbol (#). Figure 1-29 shows some HTML symbols and the corresponding code numbers or names. The appendices include a more complete list of special characters. Some older browsers support only code numbers, not code names.

Inserting a Special Character

- To insert a special character, enter:
`&code;`
 where *code* is either a code name or code number. Code numbers are preceded by a pound symbol (#).

Reference Window

Special characters and codes

Figure 1-29

Symbol	Code	Name	Description
©	©	©	Copyright symbol
®	®	®	Registered trademark
•	·	·	Middle dot (bullet)
°	°	°	Degree symbol
	 	 	Nonbreaking space, used to insert consecutive blank spaces
<	<	<	Less than symbol
>	>	>	Greater than symbol
&	&	&	Ampersand

To add a character code to the chemistry page:

1. Return to the **chem.htm** file in your text editor.
2. Replace the **e** in Mr. Dubé's name in the page title, the **img** element, and the appointments paragraph with the code, **é**, as shown in Figure 1-30.

Figure 1-30**Inserting a special character**

character code
for the é character

```
<h3>Appointments</h3>
<p>You can meet with Mr. Dub&#233; before or after school or during most
lunch hours in room H113. Do not hesitate to ask for help! It can be
very hard to catch up if you fall behind.</p>

<h3>Safety</h3>
<p>Labs are completed weekly. Because of the potential danger of any
lab exercise, you will follow the highest standards of conduct.
Misbehavior can result in dismissal from the lab.</p>

</body>
```

3. Save your changes to the file.
4. Using your Web browser, refresh or reload **chem.htm**. Figure 1-31 shows Stephen's page with the accented é in his last name; you should also see the é in the page title in the browser's title bar.

Figure 1-31**Displaying a special character****Appointments**

You can meet with Mr. Dubé before or after school or during most lunch hours in room H113. Do not hesitate to ask for help! It can be very hard to catch up if you fall behind.

Safety

Labs are completed weekly. Because of the potential danger of any lab exercise, you will follow the highest standards of conduct. Misbehavior can result in dismissal from the lab.

special character é

Now that you've completed the Web page for Stephen, you decide to print both the text file and the Web page as it appears in the browser for his review.

To print the text file and Web page:

1. Using your browser, carefully compare your Web page to Figure 1-32, which shows the entire page. If you see any errors, return to your text editor to fix them. When the page is error-free, use your browser to print the page.

Final Web page

Figure 1-32



Welcome to the Robert Service High School Chemistry Web page. Here you'll learn more about our chemistry classes and our policies.

Chemistry Classes

- **Conceptual Chemistry:** An introductory course, requiring basic math but no algebra
- **Chemistry I:** An introductory course, requiring solid algebra skills
- **Advanced Placement Chemistry:** An advanced course requiring a grade of A or B in Chemistry I

Class Policies

Grading

Homework: Homework is worth 5 to 10 points and will be given daily. A quiz consisting of 1 or 2 homework problems from the previous week may be given in place of homework.

Tests and Quizzes: Quizzes are worth 10 to 25 points and will be given at least once a month. Tests are worth up to 100 points and will be given three times each quarter.

Labs: Labs are worth 10 to 30 points and will be graded on safety, participation, and write-up. Reports should be neatly written or typed. **Research projects** will also be assigned throughout the year.

Appointments

You can meet with Mr. Dubé before or after school or during most lunch hours in room H113. Do not hesitate to ask for help! It can be very hard to catch up if you fall behind.

Safety

Labs are completed weekly. Because of the potential danger of any lab exercise, you will follow the highest standards of conduct. Misbehavior can result in dismissal from the lab.

2. Using your text editor, print chem.htm, and compare it to the complete code shown in Figure 1-33. When you are finished, you can close your text editor and browser unless you are continuing on to the assignments.

Figure 1-33

Final HTML code

```

<html>

<head>
<!-- Chemistry Classes Web Page
      Author: Stephen Dube
      Date: 8/1/2006
-->
<title>Mr. Dube's Chemistry Classes</title>
</head>

<body>
<p style="text-align: center;">
  
</p>
<hr style="color: red; background-color: red; height: 2; width: 100%" />
<p>welcome to the Robert Service High School Chemistry web page.
  Here you'll learn more about our chemistry classes and our
  policies.</p>

<h2>Chemistry Classes</h2>
<ul style="list-style-type: square">
  <li>Conceptual Chemistry: An introductory course, requiring basic
    math but no algebra</li>
  <li>Chemistry I: An introductory course, requiring solid algebra
    skills</li>
  <li>Advanced Placement Chemistry: An advanced course requiring a
    grade of A or B in Chemistry I</li>
</ul>

<h2>Class Policies</h2>
<h3>Grading</h3>
<p><i><b>Homework</b></i>: Homework is worth 5 to 10 points and will be given
  daily. A quiz consisting of 1 or 2 homework problems from the
  previous week may be given in place of homework.</p>

<p><i><b>Tests and Quizzes</b></i>: Quizzes are worth 10 to 25 points and will be
  given at least once a month. Tests are worth up to 100 points and
  will be given three times each quarter.</p>

<p><i><b>Labs</b></i>: Labs are worth 10 to 30 points and will be graded on safety,
  participation, and write-up. Reports should be neatly written or
  typed. <i><b>Research projects</b></i> will also be assigned throughout the
  year.</p>

<h3>Appointments</h3>
<p>You can meet with Mr. Dube before or after school or during most
  lunch hours in room H113. Do not hesitate to ask for help! It can be
  very hard to catch up if you fall behind.</p>

<h3>Safety</h3>
<p>Labs are completed weekly. Because of the potential danger of any
  lab exercise, you will follow the highest standards of conduct.
  Misbehavior can result in dismissal from the lab.</p>

</body>

</html>

```

Stephen is pleased with your work on his Web site and feels that it effectively captures the content of the original handout. You explain to him that the next step is to add hyperlinks to his Web page so that you can add contact information and create links to the interesting chemistry Web sites you've discovered. You'll do this in the next tutorial.

Tips for Good HTML Code

- Use line breaks and indented text to make your HTML file easier to read.
- Insert comments into your HTML file to document your work.
- Enter all tag and attribute names in lowercase.
- Place all attribute values in quotes.
- Close all two-sided tags.
- Make sure that nested elements do not cross.
- Use styles in place of presentational attributes whenever possible.
- Use logical elements to describe an element's content. Use physical elements to describe the element's appearance.

- Include the alt attribute for any inline image to specify alternative text for non-graphical browsers.
- Know your market and the types of browsers that your audience will use to view your Web page.
- Test your Web page on all relevant browsers and devices.

Review

Session 1.3 Quick Check

1. What is an inline image?
2. Why is it important to always include the alt attribute when inserting an inline image?
3. What code would you enter to display the inline image, logo.jpg, into your Web page? Assume that the alternate text for this image is “Chemistry Web Page”.
4. What code would you enter to insert a blue horizontal line that is 200 pixels wide?
5. How does a text-based browser display a horizontal line?
6. How do you insert a line break into a Web page?
7. What tag would you add to your document to insert the meta information that the page author’s name is “Diane Chou”?
8. What code would you use to insert the copyright symbol © into your page?

Review

Tutorial Summary

This tutorial introduced you to the basics of HTML. You learned about the history of the Internet, the Web, and HTML. You also studied the philosophy of HTML and learned how the language’s standards and specifications were developed, and how they are maintained. You created your first HTML file through the use of two-sided and one-sided tags and learned how to use these tags to mark the various elements of your page, such as headings, paragraphs, and lists. You also learned how to use inline styles to provide formatting instructions for your browser. This tutorial also covered how to insert inline images and horizontal lines into a Web page. Finally, you learned how to use HTML to insert special characters and symbols.

Key Terms

ARPANET	Extensible Markup	inline style
block-level element	Language	Internet
body element	extension	LAN
character formatting	graphical browser	link
elements	head element	local area network
client	host	logical element
client-server network	HTML	markup language
closing tag	HTML converter	metalinguage
comment tag	HTML editor	nesting
definition list	hyperlink	network
deprecated	hypertext	node
element	Hypertext Markup	one-sided tag
empty element	Language	opening tag
Extensible Hypertext	inline element	ordered list
Markup Language	inline image	physical element

pixel	text-based browser	Web server
presentational attribute	title element	Web site
server	two-sided tag	white space
SGML	unordered list	wide area network
specification	W3C	World Wide Web
standard	WAN	World Wide Web
Standard Generalized	Web	Consortium
Markup Language	Web browser	XHTML
style	Web page	XML
tag		

Practice


Practice the skills you learned in the tutorial using the same case scenario.

Review Assignments

Data files needed for this Review Assignment: [chemtxt.htm](#), [flask.jpg](#), [logo.jpg](#)





Stephen has some time to study the Web page you created for him and has asked your help to make some additional revisions. In the Chemistry Classes section, he wants you to add a new class that he'll be offering next semester, and he would like the chemistry class names displayed in boldface. He also wants the list marker changed to a graphic image of a flask. He would like you to indent the paragraphs on grading, office hours, and safety. He also wants you to add a numbered list in the Safety section listing his five main safety rules. He wants you to insert horizontal lines dividing the main sections of the page. Finally, he wants to add a whimsical sentence at the bottom of the site to let his students know that though he is serious about learning and safety, he wants his classes to be fun. He'd like the line to read: "Chemistry with Dubé is like medicine with a spoonful of $C_{12}H_{22}O_{11}$!" ($C_{12}H_{22}O_{11}$ is the formula for sugar.) Figure 1-34 shows a preview of the page you'll create.

Figure 1-34



Welcome to the Robert Service High School Chemistry Web page. Read below to learn about our classes and policies.

Chemistry Classes

-  **Conceptual Chemistry:** An introductory course, requiring basic math but no algebra
-  **Chemistry I:** An introductory course, requiring solid algebra skills
-  **Applied Chemistry:** An introductory course requiring solid algebra skills and an interest in using critical thinking to solve real-world chemistry-related problems
-  **Advanced Placement Chemistry:** An advanced course requiring a grade of A or B in Chemistry I

Class Policies

Assignments and Grading

Homework: Homework is assigned after each class and is worth 5 to 10 points. A periodic quiz consisting of 1 or 2 homework problems may be given in place of homework.

Tests and Quizzes: Tests and quizzes are essential to check your understanding of the material. Quizzes are worth 10 to 25 points and will be given at least once a month. Tests are worth up to 100 points and will be given 3 times each semester.

Labs: Labs are worth 10 to 30 points and will be graded on safety, participation, and write-up. Your reports should be neatly written or typed.

Research projects: Research projects give you a chance to expand your knowledge beyond the classroom. There will be several research projects throughout the year.

Office Hours

You can meet with Mr. Dubé before or after school or during most lunch hours in room H113. Do not hesitate to ask for help! It is very hard to catch up if you fall behind.

Safety Rules

Because of the potential danger of any lab exercise, you will be held to the highest standards of behavior, and will be removed from the class if you pose a threat to yourself or other students.

1. Follow the instructor's written and oral directions carefully and immediately.
2. Never perform any procedure that the instructor does not specifically direct you to do.
3. No playful behavior is permitted in the lab.
4. You must wear safety equipment as directed at all times, even if you find it uncomfortable or unbecoming.
5. No food, drinks, or loose clothing are permitted in the lab.

Chemistry with Dubé is like medicine with a spoonful of $C_{12}H_{22}O_{11}$!

To complete this task:

1. Using your text editor, open **chemtxt.htm** located in the tutorial.01/review folder. Save the file as **chem2.htm** to the same folder.
2. Within the head element of the document, insert the Web page title, “Robert Service High School Chemistry”. Also insert a comment that includes your name and the date.
3. Directly above each of the h2 headings (Chemistry Classes and Class Policies), insert a blue horizontal line that is 3 pixels high with a width equal to the width of the page.
4. In the unordered list section, after the line describing the Chemistry I class, add the following new list item:
Applied Chemistry: An introductory course requiring solid algebra skills and an interest in using critical thinking to solve real-world chemistry-related problems
5. Display each of the four class names in the list in a boldfaced font. Change the markers to the graphic image found in the flask.jpg file in the tutorial.01/review folder.
6. Enclose the four paragraphs below the h3 heading, “Assignments and Grading” within a blockquote element (this will cause the text to appear indented in most browsers.) Also enclose the paragraph below the Office Hours heading and the paragraph below the Safety Rules heading in separate blockquote elements.
7. After the block quote describing Stephen’s safety rules, create a numbered list with the following five list items:
 1. Follow the instructor’s written and oral directions carefully and immediately.
 2. Never perform any procedure that the instructor does not specifically direct you to do.
 3. No playful behavior is permitted in the lab.
 4. You must wear safety equipment as directed at all times, even if you find it uncomfortable or unbecoming.
 5. No food, drinks, or loose clothes are permitted in the lab.
8. Enclose the numbered list you just created within a blockquote element.
9. Below the numbered list insert another blue horizontal line 3 pixels high and extending the width of the page.
10. Below the horizontal line, insert a centered paragraph containing the following text. (*Hint: Use the <sub> tag to mark the numbers as subscripts and use a character code to display the character, é.*)
“Chemistry with Dubé is like medicine with a spoonful of C₁₂H₂₂O₁₁!”
11. Save your changes to chem2.htm.
12. Open your page in your Web browser and verify that it matches the page shown in Figure 1-34.
13. Submit your completed assignment to your instructor.

Apply

Use the skills you have learned to create a Web page for a childcare agency.

Case Problem 1

Data files needed for this Case Problem: [childtxt.htm](#), [newborn.jpg](#)

ChildLink, Inc. You are an employee of ChildLink, Inc., a small, nonprofit agency in Las Cruces, New Mexico. ChildLink provides financial and emotional support for families with children who have newly discovered physical or mental disabilities. The agency received significantly more donations in the last year than expected, and it has decided to offer qualifying clients temporary help with housing and medical costs. The assistant director, Sandra Pauls, has asked you to post the eligibility requirements and application process on the Web. Figure 1-35 shows a preview of the page you’ll create for ChildLink, Inc.

Figure 1-35

ChildLink of Las Cruces

A Loving Connection between Children with Disabilities and the Resources They Need



Temporary Financial Assistance Available

To be eligible for this program, you must meet the following criteria:

- Your child must have been diagnosed with a physical or mental disability within the last 6 months (the diagnosis can be prenatal or at any age)
- Your family must be at or below the State of New Mexico's poverty line

To apply, please complete the following steps:

1. Pick up an application from ChildLink (address below)
2. Assemble the following documents:
 - a. Your completed application
 - b. Doctor's record of your child's diagnosis
 - c. Tax records or New Mexico Social Services certificate of your income level
 - d. Your lease, mortgage, or medical bills, depending on which you need help with
3. Make an appointment with a ChildLink volunteer, available at the following times:
 - a. Ida: MW 10:30 a.m. to 3:30 p.m.
 - b. Juan: TR 9:00 a.m. to noon
 - c. Chris: F 10:30 a.m. to 3:30 p.m.

ChildLink
1443 Cortnic Drive
Las Cruces, NM 88001
505-555-2371

To complete this task:

1. Use your text editor to open the file **childtxt.htm** from the tutorial.01/case1 folder, and save the file as **child.htm** to the same folder.
2. Within the head element, insert a comment containing your name and the date and insert the following Web page title: ChildLink Temporary Financial Assistance.
3. Within the body element, create an h1 heading containing the text "ChildLink of Las Cruces", and center the heading on the page.

4. Below the h1 heading, create an h3 heading containing the text “A Loving Connection between Children with Disabilities and the Resources They Need”, and center the heading on the page.
5. Below the h3 heading, create an h2 heading containing the text “Temporary Financial Assistance Available”, and center the heading.
6. Below the h2 heading, create an h4 heading containing the text “To be eligible for this program, you must meet the following criteria:” Leave this heading left-aligned.
7. Below the h4 heading, create a bulleted list with square bullets. Include the two list items shown in Figure 1-35.
8. Below the bulleted list, insert an h4 heading containing the text “To apply, please complete the following steps:” Leave this heading left-aligned.
9. Below the heading, insert an ordered list containing the three numbered items shown in Figure 1-35.
10. Within the “Assemble the following documents:” list item, create an ordered list containing the four items shown in Figure 1-35. Use the lower-alpha style to display a letter rather than a number next to each item.
11. Within the “Make an appointment” list item, create another ordered list containing the three volunteer names and times as shown in Figure 1-35; as in the previous step, display the items using letters rather than numbers.
12. Insert a horizontal line below the main numbered list.
13. Below the horizontal line insert the contact information shown in Figure 1-35 as an address element. Insert line breaks within the address, and display the word, “ChildLink” in a boldfaced font. Align the address with the right margin of the page.
14. After the h3 heading (“A Loving Connection...”) near the top, insert a centered paragraph containing the inline image **newborn.jpg**. For text-based or non-visual browsers, display the alternative text, “We provide support for newborns”.
15. Below this image, insert a horizontal line.
16. Save the file, view it in your browser, compare it to Figure 1-35, and then make any corrections necessary in your text editor. Submit your completed assignment to your instructor.

Explore

Broaden your knowledge and challenge yourself by exploring how to create a Web page for a mathematics department.

Case Problem 2

Data files needed for this Case Problem: [euler.jpg](#), [eulertxt.htm](#), [pi.jpg](#)

Mathematics Department, Coastal University Professor Laureen Coe of the Mathematics Department at Coastal University in Beachside, Connecticut is preparing material for her course on the history of mathematics. As part of the course, she has written short profiles of famous mathematicians. Laureen would like you to use content she’s already written to create several Web pages that students can access on Coastal University’s Web server. You’ll create the first one in this exercise. Figure 1-36 previews this page, which profiles the mathematician Leonhard Euler.

Figure 1-36



Euler, Leonhard

(1707-1783)

The greatest mathematician of the eighteenth century, **Leonhard Euler** was born in Basel, Switzerland. There, he studied under another giant of mathematics, **Jean Bernoulli**. In 1731 Euler became a professor of physics and mathematics at St. Petersburg Academy of Sciences. Euler was the most prolific mathematician of all time, publishing over *800 different books and papers*. His influence was felt in physics and astronomy as well. Euler's work on mathematical analysis, *Introductio in analysin infinitorum* (1748) remained a standard textbook for well over a century. For the princess of Anhalt-Dessau he wrote *Lettres à une princesse d'Allemagne* (1768-1772), giving a clear non-technical outline of the main physical theories of the time.

One can hardly write mathematical equations without copying Euler. Notations still in use today, such as e and π , were developed by Euler. He is perhaps best known for his research into mathematical analysis. Euler's formula:

$$\cos(x) + i\sin(x) = e^{(ix)}$$

demonstrates the relationship between analysis, trigonometry and imaginary numbers, in one beautiful and elegant equation.

Leonhard Euler died in 1783, leaving behind a legacy perhaps unmatched, and certainly unsurpassed, in the annals of mathematics.

Math 895: The History of Mathematics

To complete this task:

1. Using your text editor, open the file **eulertext.htm** located in the tutorial.01/case2 folder, and save it as **euler.htm**.
2. Add the opening and closing `<html>`, `<head>`, and `<body>` tags to the file in the appropriate locations.
3. Within the head element, enter "Leonhard Euler" as the page title.
4. Also within the head element, enter two meta elements. The first should contain an author property set to your name. The second should contain a date property set to the current date.
5. Insert the inline image **euler.jpg** (located in the tutorial.01/case2 folder on your Data Disk) at the top of the body of the document, within a paragraph element. Provide the alternate text, "Image of Leonhard Euler" for non-graphical browsers.
6. Format the first line of the page's body, "Euler, Leonhard", as an h1 heading and format the second line of the page's body, "(1707-1783)" as an h3 heading.
7. Define the next two blocks of text as paragraphs.
8. Within the first paragraph, display the names "Leonhard Euler" and "Jean Bernoulli" in boldface. Identify the phrase "800 different books and papers" as emphasized text, and identify the phrase, "Introductio in analysin infinitorum" as a citation.

Explore

Explore

9. In the phrase “Lettres a une princesse d’Allemagne,” use the character code à to replace the one-letter word “a” with an à, and identify the name of the publication as a citation.

Explore

10. In the second paragraph, italicize the notation “e” and replace the word “pi” with the inline image **pi.jpg**, located in the case2 folder on your Data Disk. Provide the alternate text “pi” for non-graphical browsers.

Explore

11. Place the equation in a centered paragraph element and italicize each occurrence of the letters “x”, “i”, and “e” in the equation. Display the term “(ix)” as a superscript.
12. Format the next two blocks of text as paragraphs.
13. Define the name of the course at the bottom of the page as an address element.
14. Add horizontal lines before and after the biographical information.
15. Save your changes to the euler.htm file. Submit your completed assignment to your instructor.

Explore

Go beyond what you’ve learned in the tutorial by exploring how to use color and background images in a racing results Web page.

Case Problem 3

Data files needed for this Case Problem: flakes.jpg, frosttxt.htm, runner.jpg

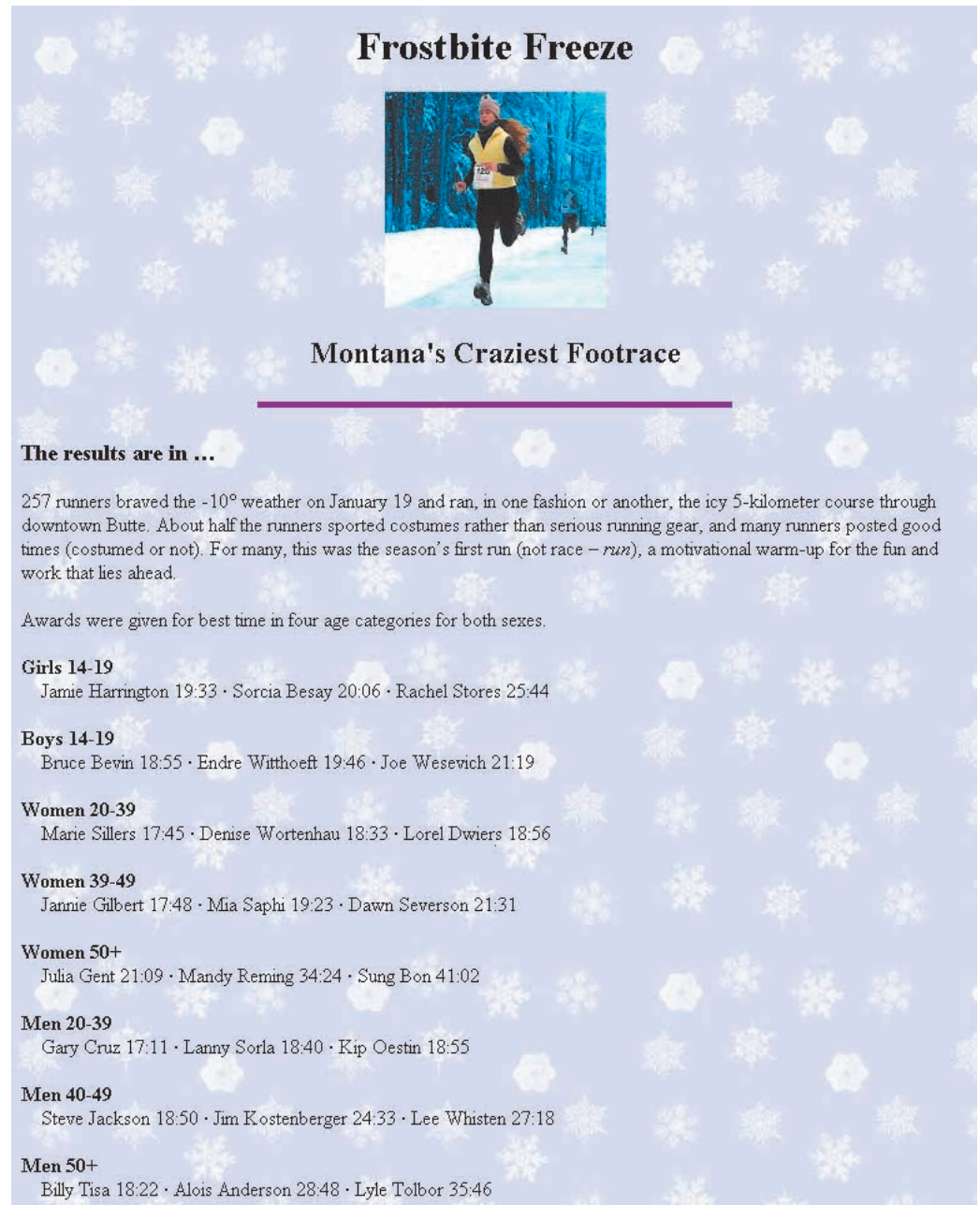
Frostbite Freeze You are on the organizing committee for the Frostbite Freeze, a fun but competitive event held each January in Butte, Montana. You’ve volunteered to publish the results for Montana’s craziest running race on the Web. You talk about the format of the page with Matt Turner, the chairman of the committee.

Matt wants you to include a snowflake background behind the text, which you can do using a graphic image. Such backgrounds are called tiled-image backgrounds because the browser repeats, or tiles, the image to cover the background of the entire page. You can create a tiled-image background with an image in either GIF or JPEG file format. To add a background image to an element, you apply the following style:


```
background-image: url(file)
```

where *file* is the name of the image file. Matt has given you two JPEG files to use for this Web page: **flakes.jpg**, which contains a snowflake pattern, and **runner.jpg**, which shows a Frostbite Freeze racer. Figure 1-37 shows a preview of the page you’ll create.

Figure 1-37



Frostbite Freeze



Montana's Craziest Footrace

The results are in ...

257 runners braved the -10° weather on January 19 and ran, in one fashion or another, the icy 5-kilometer course through downtown Butte. About half the runners sported costumes rather than serious running gear, and many runners posted good times (costumed or not). For many, this was the season's first run (not race – *run*), a motivational warm-up for the fun and work that lies ahead.

Awards were given for best time in four age categories for both sexes.

Girls 14-19
 Jamie Harrington 19:33 · Sorcia Besay 20:06 · Rachel Stores 25:44

Boys 14-19
 Bruce Bevin 18:55 · Endre Witthoeft 19:46 · Joe Wesewich 21:19

Women 20-39
 Marie Sillers 17:45 · Denise Wortenhau 18:33 · Lorel Dwiers 18:56

Women 39-49
 Jannie Gilbert 17:48 · Mia Saphi 19:23 · Dawn Severson 21:31

Women 50+
 Julia Gent 21:09 · Mandy Reming 34:24 · Sung Bon 41:02

Men 20-39
 Gary Cruz 17:11 · Lanny Sorla 18:40 · Kip Oestin 18:55

Men 40-49
 Steve Jackson 18:50 · Jim Kostenberger 24:33 · Lee Whisten 27:18

Men 50+
 Billy Tisa 18:22 · Alois Anderson 28:48 · Lyle Tolbor 35:46

To complete this task:

1. Using your text editor, open **frosttxt.htm** from the tutorial.01/case3 folder, and then save it as **frost.htm**.
2. Insert the <html>, <head>, and <body> tags in the appropriate locations.
3. Insert the Web page title "Frostbite Freeze Results" in the head element in the document and add a comment containing your name and the date.
4. Apply a style to the body element to add flakes.jpg as the background image.
5. Mark the text "Frostbite Freeze" with as an h1 heading and center it on the Web page.
6. Mark the text "Montana's Craziest Footrace" as an h2 heading and center it on the page.

Explore**Explore****Explore**

7. Insert a purple horizontal line below the h2 heading that is 50% of the width of the screen and 5 pixels high.
8. Mark the text “The results are in” as an h3 heading tag, leaving the text left aligned.
9. Add an ellipsis (...) after the text “The results are in” so it reads, “The results are in...” Use the character code for the ellipsis symbol, which you can find in appendices.
10. Add a degree symbol after “-10” in the first line of the first paragraph. (Use the character code for the degree symbol from the appendices.)
11. Identify the first two text blocks as paragraphs (one starts with “257 runners” and the other starts with “Awards were given”).
12. Near the end of the first paragraph, display the word “run” in italics (see Figure 1-37).
13. Mark each of the eight age-sex categories (for example, “Girls 14-19”) and the winners as eight individual paragraphs. Insert a line break between the category name and the winners.
14. Display the names of the eight age-sex categories in a boldfaced font.
15. Insert a middle dot symbol between each of the three names in each of the eight age-sex categories. Insert three nonbreaking spaces at the start of each line containing the winners to make the line appear indented on the page.
16. Insert the inline image **runner.jpg** (located in the case3 folder of the tutorial.01 folder on your Data Disk) between the top two headings, as shown in Figure 1-37. For non-graphical browsers, display the alternate text, “Race Results Page”. Center the image within a paragraph.
17. Save your changes and view the completed page in your Web browser. Submit your assignment to your instructor.

Create

Test your knowledge of HTML by creating a product page for Body Systems.

Case Problem 4

Data files needed for this Case Problem: logo.jpg, smith.jpg, smith.txt

Body Systems Body Systems is one of the leading manufacturers of home gyms. The company recently hired you to assist in developing their Web site. Your first task is to create a Web page for the LSM400, a popular weight machine sold by the company. You’ve been given a text file, smith.txt, describing the features of the LSM400. You’ve also received two image files: logo.jpg, displaying the company’s logo and smith.jpg, an image of the LSM400. You are free to supplement these files with any other resources available to you. You are responsible for the page’s content and appearance.

To complete this task:

1. Create an HTML file named **smith.htm**, and save it in the tutorial.01/case4 folder.
2. In the head element, include an appropriate page title, along with a comment describing the purpose of the page, your name, and the date.
3. Include at least one example of each of the following in the document:
 - a heading
 - a paragraph
 - an ordered or unordered list
 - a character formatting element
 - an inline image
 - a horizontal line
 - a special character
 - a block-level element that is not a heading, paragraph, list, or horizontal line

4. Demonstrate your understanding of inline styles by including at least two different examples of an inline style.
5. Use proper HTML syntax at all times. Close all two-sided tags. Properly nest all tags. Use lowercase element and attribute names. Enclose attribute values in quotes. Include alternate text for non-graphical browsers with inline images.
6. Write your code so that it will be easy for your supervisor to read and understand.
7. Save your HTML file, and then view the resulting Web page in a browser.
8. Submit your completed assignment to your instructor.

Review

Quick Check Answers

Session 1.1

1. A hypertext document is an electronic file containing elements that users can select, usually by clicking a mouse, to open other documents.
2. Web pages are stored on Web servers, which then makes those pages available to clients. To view a Web page, the client runs a software program called a Web browser, which retrieves the page and displays it.
3. HTML (Hypertext Markup Language) is the language in which Web pages are written.
4. HTML documents do not exactly specify the appearance of a document; rather they describe the purpose of different elements in the document and leave it to the Web browser to determine the final appearance. A word processor like Word exactly specifies the appearance of each document element.
5. Deprecated features and those features that are being phased out by the W3C, and which might not be supported by future browsers.
6. Extensions are special formats supported by a particular browser, but not generally accepted by all browsers. The advantage is that people who use that browser have a wider range of document elements to work with. The disadvantage is that the document will not work for users who do not have that particular browser, thus complicating the development process.
7. Because HTML documents are simple text files, you can create them with nothing more than a basic text editor such as Windows Notepad. However, specialized HTML authoring programs, known as HTML converters and HTML editors, are available to do some of the rote work of creating an HTML document.

Session 1.2

1. The head element which contains information about the document or instructions to the browser, and the body element which contains the content that the browser should render in the page.
2. Empty elements are elements that have no content. They are created using one-sided HTML tags.
3. `<h1 style="text-align: center">Chemistry Classes</h1>`
4. Block-level elements contain content that is displayed in a separate section within the page, such as a paragraph or a heading. An inline element is part of the same block as its surrounding content—for example, individual words or phrases within a paragraph.
5. HTML treats all white space (tabs, line breaks, blank spaces) as a single blank space and collapses consecutive occurrences of white space into a single occurrence. Thus, adding an extra blank line to your HTML file does not create an extra blank line in the rendered page.

6. Presentational attributes are HTML attributes that exactly specify how to the browser should render an HTML element. Most presentational attributes have been deprecated, replaced by styles. You should use presentational attributes when you need to support older browsers.
7. `style="list-style-type: upper-roman"`
8. `style="list-style-image: url(ball.gif); list-style-position: inside"`
or
`style="list-style: url(ball.gif) inside"`
9. Logical elements describe the nature of their enclosed content, but do not necessarily indicate how that content should appear. Physical elements describe how an element should appear, but provide little or no information about the nature of its content. Logical elements are more appropriate for a non-visual browser.

Session 1.3

1. An inline image is an inline element, used to display graphical images within the contents of a block-level element.
2. The alt attribute provides a text alternative to the image for non-graphical browsers.
3. ``
4. `<hr style="color: blue; background-color: blue; width: 200" />`
5. Text based browsers display horizontal lines using dashes or underscores.
6. Use the `
` tag.
7. `<meta name="author" content="Diane Chou" />`
8. `©` or `©`