In this chapter, you will learn how to:

- Employ the basic methods for displaying elements of multimedia on a web page, including using HTML, CSS, and nibbling
- Manipulate the appearance of text on the Web
- Determine which graphics formats are best suited for different types of images and how they can be manipulated
- Play audio on a web page by embedding the sound within the site
- Include animation on a web page
- Include video on a web page with and without the use of plug-ins

Launched in 1989, the World Wide Web was not originally designed with multimedia in mind, but rather as a simple method for delivering text documents formatted in HTML, with occasional inline graphic illustrations and figures. By 1995, because it was operational, essentially free, and good enough to support traffic (see “Vaughan’s Law of Multimedia Minimums” in Chapter 4), the Web had become a full-bore information highway of words and pictures with tens of millions of users cruising along it. The Doppler back-draft of passing travelers has exposed the gristle of an overwhelming number of disappointing audio and visual experiences on the Web: “This is my home page; here is a list of my favorite places; this is me with my dog . . .” To fill this vacuum of content and presentation, inventive multimedia solutions and enhancements now compete for mind share, stretching the capabilities of HTML, web browsers, PCs, and the very fabric of the Internet in order to bring multimedia power to this environment. Plain text and pictures are no longer enough for this highway!

WARNING

Powerful multimedia tools can be used to create totally vacuous web pages.

Developing for the Web

This chapter investigates and illustrates some methods for developing and presenting the basic elements of multimedia within the constraints of HTML, Cascading Style Sheets (CSS), and the World Wide Web. This chapter is not intended to substitute for a more complete library of HTML, CSS, web design, and Internet how-to texts, but to present basic examples that will get you started. In 2001, there were more than 2,000 published books with the word “Internet” in their title. In 2003 there were more than 6,000. In 2006, there were more than 10,000. In 2010, a search at Amazon.com showed 43,196 books with the word “Internet” in their title!
This chapter introduces you to basic ways you can put the elements of multimedia onto a web page. Learn how to use HTML tags with CSS styles rather than rely entirely on web page builders and WYSIWYG editors that never expect you to look under the hood.

All modern browsers allow you to examine the HTML code. Look for a menu item such as “View Source.” Use this feature to dig around in the source HTML code of web sites to see how the page is laid out. As you explore, you will discover that some code is neat and clear, some has plenty of embedded descriptive comments, and some is a mess of what programmers call “spaghetti” code.

**HTML Is a Markup Language**

You should have a basic understanding of HTML and CSS before you begin developing multimedia for the Web. HTML-coded documents, which are the fundamental vehicles for all types of information delivered on the World Wide Web, are explained in Chapter 12, but for this chapter you need to understand the basics of how HTML works.

HTML stands for Hypertext Markup Language. The “Markup Language” part of the name means that tags are used to do such things as format text and embed media. The tags are enclosed by angled brackets: `<>`.

Some tags are bounding tags, requiring both an opening tag and a closing tag. The closing tag is indicated by a leading forward slash inside the angled brackets. This example for bolded text illustrates the use of the two tags:

```
<strong>This text is emphasized</strong>
```

Other tags, such as the tag for inserting an inline image, stand by themselves:

```
<IMG src="grey_ball.gif">
```

Note that the tags may be written in either upper- or lowercase; some HTML text-editing programs have a switch allowing you to select the case in which you want the tags written in your document.

Tags listed in Table 13-1 are used by **HTML5**. These simple tags along with CSS elements are used to build web pages. **Deprecated** tags such as `<FONT>` and `<CENTER>` are no longer supported in the HTML standard, yet their use continues to be supported by most browsers.
<table>
<thead>
<tr>
<th>HTML Tag</th>
<th>Description</th>
<th>HTML Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;!--</td>
<td>Defines a comment</td>
<td>&lt;footer&gt;*</td>
<td>Defines a footer for a section or page</td>
</tr>
<tr>
<td>&lt;!DOCTYPE&gt;</td>
<td>Defines the document type</td>
<td>&lt;form&gt;</td>
<td>Defines a form</td>
</tr>
<tr>
<td>&lt;a&gt;</td>
<td>Defines a hyperlink</td>
<td>&lt;h1&gt; to &lt;h6&gt;</td>
<td>Defines header 1 to header 6</td>
</tr>
<tr>
<td>&lt;abbr&gt;</td>
<td>Defines an abbreviation</td>
<td>&lt;head&gt;</td>
<td>Defines information about the document</td>
</tr>
<tr>
<td>&lt;address&gt;</td>
<td>Defines an address element</td>
<td>&lt;header&gt;*</td>
<td>Defines a header for a section or page</td>
</tr>
<tr>
<td>&lt;area&gt;</td>
<td>Defines an area inside an image map</td>
<td>&lt;hgroup&gt;*</td>
<td>Defines information about a section in a document</td>
</tr>
<tr>
<td>&lt;article&gt;*</td>
<td>Defines an article</td>
<td>&lt;hr&gt;</td>
<td>Defines a horizontal rule</td>
</tr>
<tr>
<td>&lt;aside&gt;*</td>
<td>Defines content aside from the page content</td>
<td>&lt;i&gt;</td>
<td>Defines italic text</td>
</tr>
<tr>
<td>&lt;audio&gt;*</td>
<td>Defines sound content</td>
<td>&lt;ins&gt;</td>
<td>Defines inserted text</td>
</tr>
<tr>
<td>&lt;b&gt;</td>
<td>Defines bold text</td>
<td>&lt;keygen&gt;*</td>
<td>Defines a generated key in a form</td>
</tr>
<tr>
<td>&lt;br&gt;</td>
<td>Inserts a single line break</td>
<td>&lt;kbd&gt;</td>
<td>Defines keyboard text</td>
</tr>
<tr>
<td>&lt;button&gt;</td>
<td>Defines a push button</td>
<td>&lt;label&gt;</td>
<td>Defines a label for a form control</td>
</tr>
<tr>
<td>&lt;canvas&gt;*</td>
<td>Defines graphics</td>
<td>&lt;legend&gt;</td>
<td>Defines a title in a fieldset</td>
</tr>
<tr>
<td>&lt;caption&gt;</td>
<td>Defines a table caption</td>
<td>&lt;li&gt;</td>
<td>Defines a list item</td>
</tr>
<tr>
<td>&lt;cite&gt;</td>
<td>Defines a citation</td>
<td>&lt;link&gt;</td>
<td>Defines a resource reference</td>
</tr>
<tr>
<td>&lt;code&gt;</td>
<td>Defines computer code text</td>
<td>&lt;map&gt;</td>
<td>Defines an image map</td>
</tr>
<tr>
<td>&lt;col&gt;</td>
<td>Defines attributes for table columns</td>
<td>&lt;mark&gt;*</td>
<td>Defines marked text</td>
</tr>
<tr>
<td>&lt;colgroup&gt;</td>
<td>Defines groups of table columns</td>
<td>&lt;menu&gt;</td>
<td>Defines a menu list</td>
</tr>
<tr>
<td>&lt;command&gt;*</td>
<td>Defines a command button</td>
<td>&lt;meter&gt;*</td>
<td>Defines measurement within a predefined range</td>
</tr>
<tr>
<td>&lt;datalist&gt;*</td>
<td>Defines a drop-down list</td>
<td>&lt;nav&gt;*</td>
<td>Defines navigation links</td>
</tr>
<tr>
<td>&lt;dd&gt;</td>
<td>Defines a definition description</td>
<td>&lt;noscript&gt;</td>
<td>Defines a noscript section</td>
</tr>
<tr>
<td>&lt;del&gt;</td>
<td>Defines deleted text</td>
<td>&lt;object&gt;</td>
<td>Defines an embedded object</td>
</tr>
<tr>
<td>&lt;details&gt;*</td>
<td>Defines details of an element</td>
<td>&lt;option&gt;</td>
<td>Defines an option group</td>
</tr>
<tr>
<td>&lt;dfn&gt;</td>
<td>Defines a definition term</td>
<td>&lt;ol&gt;</td>
<td>Defines an ordered list</td>
</tr>
<tr>
<td>&lt;div&gt;</td>
<td>Defines a section in a document</td>
<td>&lt;optgroup&gt;</td>
<td>Defines an option group</td>
</tr>
<tr>
<td>&lt;dt&gt;</td>
<td>Defines a definition list</td>
<td>&lt;option&gt;</td>
<td>Defines an option in a drop-down list</td>
</tr>
<tr>
<td>&lt;em&gt;</td>
<td>Defines emphasized text</td>
<td>&lt;output&gt;*</td>
<td>Defines some types of output</td>
</tr>
<tr>
<td>&lt;embed&gt;*</td>
<td>Defines external interactive content or plug-in</td>
<td>&lt;p&gt;</td>
<td>Defines a paragraph</td>
</tr>
<tr>
<td>&lt;fieldset&gt;</td>
<td>Defines a fieldset</td>
<td>&lt;pre&gt;</td>
<td>Defines preformatted text</td>
</tr>
<tr>
<td>&lt;figcaption&gt;*</td>
<td>Defines the caption of a figure element</td>
<td>&lt;progress&gt;*</td>
<td>Defines progress of a task of any kind</td>
</tr>
<tr>
<td>&lt;figure&gt;*</td>
<td>Defines a group of media content, and their caption</td>
<td>&lt;q&gt;</td>
<td>Defines a short quotation</td>
</tr>
</tbody>
</table>

Table 13-1  Showing All HTML Tags Used for Building Web Pages. An Asterisk Marks Tags That Are New for HTML5 (see www .w3schools.com/html5/html5_reference.asp)
### HTML and Multimedia

HTML provides tags for inserting media into HTML documents: the `<IMG>` tag for inline images; the `<AUDIO>` and `<VIDEO>` tags for multimedia; and the `<EMBED>` and `<OBJECT>` tags for compound document embedding used to insert a “nonstandard” item such as a Java applet or Flash animation into an HTML document.

But it is not as simple as it seems. There is a difference between the way various versions of browsers handle multimedia elements and the plug-ins that play them. Some browsers that understand the `<OBJECT>` tag ignore the `<EMBED>` tag, and some browsers that cannot read the `<OBJECT>` tag need the `<EMBED>` tag. The **Object/Embed method** places an `<EMBED>` tag within the `<OBJECT>` tag to ensure that multimedia elements will play in all browsers. Thus the HTML code to play a flash animation might look something like this:

```html
<object classid="clsid:D27CDB6E-AB6D-11cf-96B8-444553540000" width="320" height="240" id="player1" name="player1">
  <param name="flashvars" value="file=playlist1.xml">
  <param name="movie" value="player.swf">
  <param name="allowfullscreen" value="true">
  <param name="allowscriptaccess" value="always">
  <embed id="player1"
    flashvars="file=playlist1.xml"
    name="player1"
    ...
</object>
```

---

**Table 13-1**  Showing All HTML Tags Used for Building Web Pages. An Asterisk Marks Tags That Are New for HTML5 (see www.w3schools.com/html5/html5_reference.asp)  (Continued)
If you develop multimedia for the Internet, budget time and effort for keeping current in this rapidly changing environment—staying at the leading edge takes effort. It will be some years before multimedia delivery tools and techniques for the Web stabilize.

The Desktop Workspace

Make your web pages look good on a 1024 × 768 display in true color (millions). Working at this resolution, you will satisfy more than 95 percent of all desktop viewers. Depending upon the browser and preferences set by the user, however, the area of the screen available for your web page, called the viewport, will always be less than the full display, and it is not controllable by the designer. Browser “chrome” (toolbars and other shiny stuff around the edges of your page’s viewport) can be either hidden or shown by the user. If you want to maximize the browser active window size, in Internet Explorer press f11 (function key 11) and go back to regular mode by clicking the mouse; other browsers offer Full Screen toggle switches in the “View” options. So design your web page for a 1,024-pixel-wide display by using tables and images that do not exceed about 1,000 pixels across the page, and you will have room for browser scroll bars. Many designers choose a viewport workspace 960 pixels wide; a number divisible by 3, 4, 5, 6, 8, 10, 12, 15, and 16. This makes many logical “grid systems” of columns possible.

The Small-Device Workspace

Under the hood of many browsers is a layout engine for rendering pages. Versions of those browser engines have been customized to run on small devices such as tablets, e-readers, netbooks, PDAs, and smartphones, and they follow known rules when laying out web pages for smaller viewports. Microsoft Internet Explorer uses the Trident engine; Firefox uses the Gecko engine; Opera uses the Presto engine; and Apple’s Safari and Google’s Chrome use the WebKit engine.

Smartphones use various operating systems: Android, iPhone OS, Linux, Maemo, Palm WebOS, RIM’s BlackBerry, Symbian OS, and Windows CE, with the most widely used being Symbian OS on handsets and devices manufactured by BenQ, Fujitsu, LG, Mitsubishi, Motorola, Nokia, Samsung, Sharp, and Sony Ericsson.
To deal with the multiplicity of viewport sizes in the small-device world (320 × 480, 240 × 320, 240 × 400, 854 × 480, etc.), the Android OS allows programmers to write one application that flexibly covers all display sizes by using virtual **density-independent pixels (dips)**:

The density-independent pixel is equivalent to one physical pixel on a 160 dpi screen, the baseline density assumed by the platform....

At run time, the platform transparently handles any scaling of the dip units needed, based on the actual density of the screen in use.

The conversion of dip units to screen pixels is simple: \[ \text{pixels} = \text{dips} \times \frac{\text{density}}{160} \]. For example, on a 240 dpi screen, 1 dip would equal 1.5 physical pixels. Using dip units to define your application's UI is highly recommended, as a way of ensuring proper display of your UI on different screens.


While HTML and CSS do not provide for device-independence, if you expect that your project will be widely viewed on small devices, consider designing at 960 pixels to allow the device’s browser the most flexibility. More, you should keep in mind that input events on small devices are different from the clicks and drags of a computer with mouse or touchpad: a **Double Tap** makes the browser zoom in and center on a document; a **Touch and Hold** will display an information bubble; a **Drag** will move the viewport or pan; a **Flick** will scroll up or down; and a **Pinch Open** or **Pinch Closed** will zoom in or out. There are no mouseOver events without a mouse.

**Nibbling**

A principle you must always keep in mind when designing and making multimedia elements for the Web and particularly for handheld devices should be called “nibbling.” At a serious metal-working supply store you can buy a power tool called a nibbler—it devours the edges of sheet metal in an ear-damaging staccato of rapid tiny bites. You must apply this concept, for example, to the elegant bitmapped logo you created in Photoshop when you trim it from 24- to 8- to 4-bit color depth and resize it from 96 pixels square to 64 pixels square and create a transparent .png file. Nibble the audio clip of your client’s theme song from 44.1 kHz to 11 kHz, and see if it’s acceptable at an 8-bit sample size. Text as HTML is cheap: nibble your page design and throw away the pretty shadowed GIF graphic headers and image maps—re-create your text in HTML headers or emphasized text, and try coloring it. Put on your protective headgear—this compromising work is painful for you as the creator—and start nibbling, while...
constantly seeking a balance between quality and the patience of a user who is downloading your material at 56 Kbps from home. Every choice you make should be tempered by bandwidth worry.

**WARNING** For every image file referenced in an HTML document, a separate Internet HTTP connection must be made between your computer and that image’s server before the image itself is downloaded; so using many different tiny images (such as various graphic images for bullets) may not be efficient. After a user has downloaded a file once, however, it should load more quickly from the user’s local hard disk, where the browser stores them in a cache.

---

**Text for the Web**

In addition to variations in the size of the viewport, viewers of your web site may not be displaying the same “preferred” font that you used to design your page because user preferences in the browser may alter the way text in your document looks and flows. To make the best of this uncertainty, many developers design their documents in Times Roman for the proportional serifed font, Verdana for proportional sans serif, and Courier as the monospaced font. These fonts readily move across platforms and are the default fonts users typically see if they do not set their own preferences. Although you can specify a font, and even alternate fonts, using CSS, browsers can only use a specified font if that font is installed on the end user’s computer. Figure 2-3 in Chapter 2 shows a list of the most commonly installed fonts on Windows, Macintosh, and Linux computers.

**NOTE** As with projects built for CD-ROM or DVD distribution using a multimedia authoring tool, if you wish to absolutely control the look of text on your web page, you must use a graphic bitmap rather than text in your HTML document. Adding images in place of text, however, increases the amount of time necessary to download your page. Embedding graphics into HTML documents is explained later in the chapter.

You can tag text so that it is displayed as a header, strong, emphasized, or sub- or superscripted. Using CSS, you can specify your “preference” for font face and many text attributes (see Chapter 2), but the viewer’s browser ultimately determines if and how these styles are displayed.

**Making Columns of Text**

The most powerful feature of HTML may be found in the `<TABLE>` tag. Study this tag and its attributes! To the right, you’ll see how to organize your text into two columns, so it displays more like a newspaper or a magazine, using a table (see Figure 13-1).
In those days there were mandatory school meetings where Headmaster Wilkie sat in his Eames chair next to the fireplace, usually with a guest of honor or the Dean sitting in a matching chair at the other side of the fire. The entire student body then sat at their August feet on thick oriental carpets (the same ones that were rolled up for the tea dances). When the meeting was about fully assembled, we drifted across the front of the room and carefully unloaded the odious vials onto the stone hearth between the two honorables by tipping them secretly from our blazer pockets.

Crystals were sprinkled unseen like nuclear fallout onto Nevada rock and lay there dormant as we took places in the crowd.

When Mr. Wilkie arose to introduce our guest, who was Walter Lord, the noted author and maritime disaster expert, he stepped onto the loaded hearth. It was a smokeless sound like tearing statically charged nylon from rubber, but there were occasional louder pops and crackles as the event continued for a long second or two. With magnificent aplomb after his startled, and in a terrible vacuum of silence which included none of the outbursts of laughter or glees which we had so expected from the student body, the Headmaster flatly continued his introduction.
Flowing Text Around Images

As you can see in Figure 13-2, it is possible (and easy) to “flow” text around an image using the ALIGN attribute of the `<IMG>` tag. This is a quick and simple method for mixing text and images in a pleasing layout. Add a `<BR CLEAR="left">` tag at the end of your text paragraph, so that if there is not enough text to fill the entire vertical height of the image, your next paragraph will begin on a new line, left-justified, and below the image. To add space around your image so it doesn’t butt right up against the text, use the Horizontal Space (HSPACE) and Vertical Space (VSPACE) attributes of the `<IMG>` tag.

![Image of text flowing around a ship]

**Figure 13-2** You can flow text around an image by using the ALIGN attribute of the `<IMG>` tag.

```html
<HTML>
<HEAD>
<TITLE>Sailing</TITLE>
</HEAD>
<BODY>
<IMG SRC="gbsky.gif" ALIGN="left" HSPACE="15" VSPACE="5">
<H2>Departure</H2>
... text goes here ...
<BR CLEAR="left">
<hr>
</BODY>
</HTML>
```
The following HTML and CSS code sets up a more complicated screen with flowing text (see Figure 13-3). It also includes a background image, a portrait image, and an image map that is used for navigation. (Background images and image maps are described later in this chapter.) This document also contains the foreign language special character ä, which is called out in the document using HTML’s escape sequence for special characters, in this case, “&auml;”. An escape sequence begins with an ampersand and ends with a semicolon. Also, note the link to a separate style sheet file holding the CSS code. A MIDI file is embedded in this page to provide background music.

```html
<html>
<head>
<title>Annan Lapsuus</title>
<link rel="stylesheet" type="text/css" href="anna.css">
</head>
<body>
<table><tr>
<td width="20"></td> <!-- blank spacer column for indent -->
<td width="180" align="center" valign="top"> <!-- Column to hold image -->
<img src="anna.jpg" align="left">
</td>
<td width="400" align="left" valign="top"> <!-- Column to hold text and nav button -->
<h1>Annan Lapsuus</h1>
<p class="annaText">
<br><br>
<br><br>
”Tästä lähtien minun huoneeni on aina hyvässä järjestyksessä”, sanoin isille.
<br><br>
Isi hymyili.</p>
</td></tr></table>
</body>
</html>
```

Figure 13-3 Images, text, and sound can be mixed in an HTML document. Note the use of escape sequences for special characters and an image map for navigation.
Images for the Web

Theoretically, the Web can support any graphics format the client and server have in common. Practically, even though web standards do not specify a graphics format you must use, browsers recognize four image formats—GIF, PNG, JPEG, and SVG—without resorting to special plug-ins. These formats use built-in compression algorithms to reduce file size. (Graphic image formats are described in detail in Chapter 3.) For other graphics formats, such as CGM, CMX, DXF, and fractal- and wavelet-compressed images, special proprietary creation software and browser plug-ins may be required.

GIF and PNG Images

GIF images (Graphic Interchange File, also discussed in Chapter 3) are limited to 8 bits of color depth (256 colors). This is a commercial image format developed by CompuServe Information Services, an online
company once owned by Unisys and currently folded into America Online. In late 1994, Unisys announced a patent fee charge to all software developers who use the GIF format. In an angry, industry-wide response, **PNG** (for Portable Network Graphics Specification) was developed as a new “open” format (not requiring fees) to replace GIF. By allowing transparency by single pixel or by alpha channel mask and a 24-bit indexed palette, the PNG format is an improvement on the GIF format it was intended to replace. But it does not support animation. And because it only uses the RGB color model (not CMYK), PNG images may not print well.

![First Person]

A few years ago somebody told me about an interesting web survey: how does the world pronounce GIF? The results turned out about 50/50 on the hard/soft question, my colleague claimed. Then I spent considerable time using that word (softly) in Europe before realizing everybody was being smirkingly polite about my outlandish pronunciation. In the San Francisco Bay Area, a world center for multimedia development, GIF has the soft “g” of “ginger,” “gin,” and “gybe.” In New York, where little is soft, and in Europe, GIF has a more cutting, hard pronunciation, as in “giggling,” “gingham,” “girdled,” “guilty,” or “girls.” The real question is whether the written word requires a prefixed dot.

### JPEG Images

**JPEG** (Joint Photographic Experts Group) images may contain 24 bits of color depth (millions of colors). JPEG uses a powerful but **lossy** compression method that produces files as much as ten times more compressed than GIF. Lossy means that information in the original image is lost in the compression process and cannot be retrieved. A **lossless** compression method does not irretrievably discard the original data.

**WARNING**  
Do not edit and reedit files that are in JPEG format. Every time you open a JPEG image and edit it, then recompress and save it as a compressed JPEG, the image degrades. After a few editing/saving cycles, you will be very disappointed. Edit and archive your images in a 24-bit lossless graphic format (such as TIFF, BMP, or PSD), then convert to JPEG (if you need to).

The JPEG compression scheme compresses about 20:1 before visible image degradation occurs. Test the amount of compression acceptable for your JPEG image; stay inside the “threshold of visible error.” To compress an image with JPEG, the image is divided into $8 \times 8$-pixel blocks, and the resulting 64 pixels (called a “search range”) are mathematically described
relative to the characteristic of the pixel in the upper-left corner. The binary description of this relationship requires far less than 64 pixels, so more information can be squeezed into less space. JPEG compresses slowly, about one to three seconds for a 1MB image, depending upon computer speed, but JPEG can compress images as much as 75:1, with loss.

**GIF or JPEG?**

Use JPEG for photo-realistic images containing many colors, and avoid using it for images already forced into a 256-color palette or for line drawings or 1-bit black-and-white images. GIF compresses drawings and cartoons that have only a few colors in them much better than JPEG, which may introduce visible defects—sharp edges and lines that blur—especially with small-size text. Figures 13-4 and 13-5 show the “blocky” and “lossy” nature of compressed JPEG images. For the Web, use the JPEG format for photo-realistic images that are busy with color; use the GIF format for line art and drawings where there are large areas of the same color.

---

**Figure 13-4** Both images at the top were saved in the JPEG format, which compresses image data and trades image quality for small file size. The resulting compressed images at the bottom show the “lossy” and “blocky” nature of compressed JPEGs. The photo at top left is 71K in size when saved as a GIF (not shown) and only 27K saved as a JPEG (bottom left). The drawing at top right is 17K when saved as a GIF (not shown) and 46K as a JPEG (bottom right).

**Figure 13-5** Lossy compression schemes save disk space but can also degrade an image. For the Web, line art is often better saved in Gif, PNG, or SVG format than in JPEG.
Using Photoshop

Adobe’s Photoshop is the “tool of choice” for most graphic artists, so it is worth taking some time to provide a few suggestions for creating images for use on the World Wide Web. If you use a different image-editing application, follow the same logic and use the commands appropriate for that application. Always work in native Photoshop format using PSD files—these images are typically in RGB mode and use the maximum color depth. They are larger, but they contain more information that can be usefully processed when resizing and dithering, and you will get better final results. PSD files also contain layers, a very useful application feature. When creating images for display on a web page, use 72 pixels per inch resolution, which is the resolution of most monitors. When you convert a 24-bit RGB image to an 8-bit indexed image (change its mode), you lose huge amounts of color information that cannot be retrieved, meaning that the fine data is gone forever. So you should follow two practices in order to protect your original image. One is to save the original image in a 24-bit lossless image format (such as PSD, TIFF, or BMP). The other is to do all of your image manipulation (such as resizing, sharpening, and hue adjustments) in RGB mode. Next, save this source image in RGB mode as a PSD file, before reducing the color palette by saving it as a GIF or using a lossy compression like JPEG. By saving the high-quality original and saving the manipulated image in the program’s native format, you can return to them if you need to make changes later.

**Tip**  When you scan an image, the scanner will often default to print resolution of 300 dpi. When displayed on a 72 dpi resolution monitor, the picture will be displayed more than three times bigger than the original. Never fix this problem by changing the height and width attributes of the IMG tag. Even though this will display the image at the size you want, you still have a huge image file that will slow down the downloading and display of your page. Instead, use Photoshop or another image-editing program to resample the image at a 72 dpi resolution, and use that new image on your page.

When you are satisfied with your image and ready to save it as a GIF, PNG, or JPEG file, archive it as described earlier. If you make any mistakes while converting modes or saving, you will still have the original, complete with any layers you might have used. To be very safe, duplicate the original file and open the copy before saving to other formats.

**Saving as JPEG Files**

To save your image as a JPEG file, you do not need to change Photoshop’s mode from RGB, but if you are using layers, you will need to “flatten” the image, merging all layers into a single bitmap. Once an image is flattened and you have edited or saved it, its layers cannot be remade without a great
deal of difficult cutting and pasting—so again, archive your original file! You must name your file with the extension .jpg or .jpeg if you will use it on the Web. Then click Save, and choose Maximum, High, Medium, or Low-quality compression in the dialog box that appears. Your file is ready for the Web.

**Saving as GIF Files**

To save a GIF file using Photoshop, you must first set the mode of your image to Indexed Color, converting it to the best 8-bit palette (256 colors) that will represent the image and be displayed well by web browsers. Note that the option of saving a Photoshop 24-bit RGB file in GIF format will not be available in Windows, and it will be grayed out on the Macintosh menu until you have converted your image to 8-bit mode: GIF is only for 8-bit images.

**TIP** Use GIF files for line art and images that contain large areas of the same color (that can be easily compressed). Use JPEG for photo-realistic images.

**Palettes** When you change the mode to Indexed Color, you must specify the color depth of the converted image, the color palette to be used, and whether the colors of your image should be dithered (Diffusion or Pattern) or not (None). Figure 13-6 shows the mode changing dialog box from Photoshop, where the custom Netscape Navigator palette for Windows has been selected.

![Figure 13-6](image)

*Figure 13-6*  
In Photoshop, changing the mode of your image from RGB Color to Indexed Color changes the color depth of your image.
Interlaced and Progressive Scans  
Both GIF and JPEG images can be saved so that when your browser displays the image as it is being downloaded, you can immediately see a chunky approximation of the final image, with resolution improving as more and more data comes in. While in baseline, or normal configuration, image data is stored as a single top-to-bottom scan; in interlaced GIF and progressive JPEG files, the data is organized in a different sequence within the file. An interlaced GIF file, for example, is arranged into a series of four passes:

- **Pass 1**: Every 8th row, starting with row 0
- **Pass 2**: Every 8th row, starting with row 4
- **Pass 3**: Every 4th row, starting with row 2
- **Pass 4**: Every 2nd row, starting with row 1

Figure 13-7 shows Photoshop’s Save for Web & Devices dialog for saving an image as interlaced, and four increasingly resolved images.

![Figure 13-7](image)

Interlacing settings when exporting a GIF89a file from Photoshop. With interlacing, the image incrementally improves its resolution as it downloads.

Transparency  
The GIF89a and PNG specifications allow for transparency: you can save your file with instructions to a browser to use a specific color or palette of colors (with PNG) as your selected
transparency color. In many cases, such as for company logos and inline illustrations, it is attractive to let an image float on top of the browser’s background.

Images on web pages are displayed as rectangles. The area outside of the circle in Figure 13-8 is filled with a wash of color and would (without transparency) be displayed as a rectangle showing those colors to its edges. To make the part surrounding the circle transparent so that the circle floats on your web page, fill the area outside the circle with a single color, and then save the file, selecting that fill color to be transparent. While white is often used as the transparency color, in this example it would not work because there are white pixels inside of the circle that would also become transparent. Use a fill color not in the area you wish to show; in this case red works. Most image-editing tools provide a palette from which you can select the transparency index color. You cannot make a JPEG file transparent.

![Figure 13-8](image)

**Figure 13-8** Use a transparent GIF or PNG to float a circle or other image on a web page: select the area outside of the circle (upper left), fill it with a single indexed color (red, lower left), choose that color to be transparent (lower right), and save the image as a GIF file. The circle will float on your page (upper right).
Backgrounds

Most browsers allow you to specify an image or color to place in the background of your page or into table cells. Text and images will float on top of this layer.

Background Coloring

You can choose colors for backgrounds, text, and anchors to URL links. Color controls for the entire page are attributes of the `<BODY>` tag and are set using CSS:

```css
body {background-color: #0000FF;}
```

where “#0000FF” is a hexadecimal red-green-blue triplet used to specify the background color, in this case, blue. See Chapter 3 for an explanation of red-green-blue triplets.

Once you have chosen a background color, you will then want to set the color of your text and establish proper contrasts. Red on green shimmers, black on black is invisible. By setting styles in the `<BODY>` tag, you set default styles for the entire document. For white text on a blue background, the CSS code would be:

```css
body {color: #FFFFFF;}
```

Background Images

Background images are by default tiled, or repeated, across and down the page until the page or page element is filled, so a randomly distributed “sandy” background image (see Figure 13-9) can easily be made from a very small source image.

Load a background image into a document by specifying its URL (if it is available somewhere on the Web) or its relative file path (if it is on the same server as the page) in the CSS attributes for the `<BODY>` tag, for example:

```css
body {background-image: url('paper.gif');}
```

**Tip**  It is a good idea to specify a background color similar to the prevailing color of the image being used for a background. If the user viewing your page has Image Loading turned off, or if your background image cannot be found for some reason, the page may still look close to the way you designed it. If the image you specify as a background has transparent areas, the background color will show through.
Figure 13-9  A simplified navigation map not only provides an overview of a multimedia project but also contains active links to documents using “hot” areas of the graphic. This is the navmap page from Navigare Necesse Est, a student-built love story.
**Sidebars**

In the navigation map shown in Figure 13-9, a commonly seen graphic layout was used: a vertical bar containing the word “navmap” is displayed at the left of the screen and in the background. When users scroll up or down, this bar remains stationary. Make the graphic bar at the left as wide as you wish (say 75 pixels); then set the full width of your image to 1,000 pixels. Fill the space to the right of your bar with plain color or a texture. When this background image repeats itself (tiles), it will repeat to the right only if the user widens the viewing window to more than 1,000 pixels; but the image will tile vertically in increments of its height until it reaches the bottom of the window. With CSS you can force the browser to repeat only vertically, only horizontally, or not at all. In this example, adding `background-repeat: repeat-y;` to your CSS code will allow repeats only vertically, even when the window is made wider than 1,000 pixels.

**Clickable Buttons**

To make a graphic image “clickable” so that it links to another document, simply include the image tag inside the bounding tags of an HTML anchor that points to that document’s URL:

```html
<a href="documentToGoTo.html">
  <img src="greenButton.gif" border="0">
</a>
```

You can also use the `<A>` tag to provide a link to a larger graphic or even to a video clip from a small, thumbnail-sized image:

```html
<a href="bigPicture.jpg"><img src="thumbnail.gif" border="0"></a>
```

Be sure to include the `BORDER attribute (border="0")` in the `<IMG>` tag if you wish to avoid showing a border around the button image (sometimes an ugly two blue pixels wide).

**Client-Side Image Maps**

Image maps are pictures with defined hot spots that link to other documents when a user clicks on them. Browsers support client-side image maps so that mouse coordinates and their associated document URLs can be included in an HTML document. This is managed by the `<MAP>` tag and the `USEMAP attribute` of the `<IMG>` tag.

To make a client-side image map with `USEMAP`, you need three things: an image, a list of coordinates designating hot spots on the image, and the document URL associated with each hot spot. To program the image map into your HTML document, you use the `USEMAP attribute`
of the \texttt{<IMG>} tag. Here is the HTML code for the navigation button in Figure 13-3 and detailed in Figure 13-10:

\begin{verbatim}
<img src="compas.gif" hspace="5" vspace="50" border="0" usemap="#compass">
<MAP name="compass">
  <area shape="circle" coords="60,60,10" href="help.htm">
  <area shape="polygon" coords="60,60,0,0,120,0" href="back.htm">
  <area shape="polygon" coords="60,60,0,120,120,120" href="forward.htm">
  <area shape="polygon" coords="60,60,0,0,0,120" href="navmap.htm">
</MAP>
\end{verbatim}

Compas.gif is the transparent image, the \texttt{hspace="5"} and \texttt{vspace="50"} attributes provide space between the image and the text around it, and the \texttt{border="0"} attribute makes the image borderless. The \texttt{usemap="#compass"} attribute points to the \texttt{<map>} extension tag that contains the coordinates and URLs. (The pound sign means the \texttt{<MAP>} tag is located in this same document.) A \texttt{<MAP>} segment may be placed anywhere in the body of the HTML document and is related to the correct image by the \texttt{name="xxxxxxx"} attribute of the \texttt{<MAP>} tag. You can have more than one image map in an HTML document, but they must have different names.

Within the \texttt{<MAP>} tag, the \texttt{<AREA>} tag defines the shape of the hot spot (as a circle, polygon, or rectangle) and anchors or links it to a URL. Areas are defined by \texttt{x,y} coordinates of the pixels in your bitmap: a circle by the \texttt{x,y} coordinates of its center location and radius (60,60,10), a polygon by a sequence of sets of \texttt{x,y} locations that close automatically (60,60,0,0,120,0 defines a triangle), or a rectangle (two \texttt{x,y} locations defining top left and bottom right).

\begin{figure}
\centering
\includegraphics[width=\textwidth]{figure13-10.png}
\caption{This enlarged image illustrates the coordinates used to define hot spots for image maps (the ruler is marked in pixels).}
\end{figure}
Sound for the Web

In the beginning, when the Internet was primarily a collection of Unix machines, sound files were sent from machine to machine in **AU format** and, when downloaded, were played back using a sound application. As the Web has developed, sound has become more important, and most browsers allow embedding of sounds into documents using the `<AUDIO>` tag. Inside this tag, the `autoplay` attribute, if present, starts the audio playing as soon as it is ready. If `controls` is present, a play/pause and other controls will be displayed. When `preload` is present, the audio will load when the page does and be ready to run. Text can be included in the tag that will be ignored unless the user’s browser cannot understand the `<AUDIO>` tag:

```html
<audio src="LizLaugh.aiff" preload autoplay controls loop>
  Sorry, your browser does not support the HTML audio element.
</audio>
```

The `<A>` anchor tag and `<EMBED>` tag can also be used to play sound files:

```html
Click <a href="LizLaugh.aiff">here</a> to play sound file.
<embed src="Mozart.mid" autostart="true" loop="false" width="120" height="50" hidden/>
```

Chapter 4 describes designing and making MIDI and digitized sound files in detail.

**TIP**  Making sound for the Web requires the basic tools and techniques described in Chapter 4. Always nibble at your sound elements and reduce them to the lowest file sizes that will play acceptably. Remember, they will move across the Internet and may be downloaded or played on machines with low-bandwidth connections.

Animation for the Web

HTML makes no provision for animation, by itself delivering only a static page of text and graphics. Boring, many people said, and programmers went to work devising methods to liven up the view. JavaScript can dynamically change a web page without needing to reload it. JavaScript with XML features, combined into **Ajax** (Asynchronous JavaScript and XML), is used for powerful interactive applications such as Google’s “Office.” The Flash plug-in for browsers offers animation and interaction.

GIF89a

Browsers implement a little-known animation feature in the final 1989 revision “a” of the GIF file format specification. It is possible to make
simple animations by putting multiple images, or frames, into a single GIF89a file and display them with programmable delays (in 100ths of a second) between them.

When you use the `<IMG>` tag to embed a GIF89a multiframe image, the browser downloads the file and stores it in the cache folder of your local hard disk. Once fully downloaded, the browser plays each frame of the image quickly and smoothly. Limit animated GIFs to small images, and use a more capable plug-in like Flash for animations over larger areas.

Read Chapter 5 to learn the basics of animation. Pick a tool or method and start creating. Lokki, the Shockwaved seagull, was created by a beginner and was flying in just a few hours:

Animation software includes Swish (www.swishzone.com), Flash, Director, After Effects, DHTML, and animated GIF files built using shareware and freeware. Designers must be careful how they use animation though: too much motion and too many flashy colors can cheapen a web site. Subtle animation, however, enhances a site’s content and messages.

For animation styles and tips

**Video for the Web**

In the past (and still today), playing video on a web page requires special plug-ins like Flash, QuickTime, RealVideo, Windows Media Player, or other proprietary software. These plug-ins use many different codecs (compressor/decompressors) and many different streaming and storage container file formats, each with its own interface and custom options, as shown here.

To reduce reliance on these plug-ins and offer a standard method for preparing and delivering video to the Web, the HTML5 specification provides a `<VIDEO>` tag, meaning that HTML5-capable browsers such as Internet Explorer
9, Firefox 3.5, Safari 4, Chrome, and Opera must contain within themselves the programming code required to recognize a video file, read and decompress both its audio and video components, and play that video on a screen—where and how you, as the designer, specify.

```html
<video src="myVideo.mpg" preload autoplay controls width="320" height="240">
  Sorry, your browser does not support the HTML video element.
</video>
```

There are more than 250 file formats that contain video elements (see www.fileinfo.com/filetypes/video), and there are more than 25 codecs from which to choose. And there are many methods and options to capture, compress, edit, store, and distribute video. Introduction of the HTML5 `<video>` tag is a push in the direction of a standardized few technologies and methodologies that will work for most everyone on the Internet.

The most commonly used codecs are H.264, Theora, and VP8 within MP4, Ogg, and WebM containers. Unfortunately, no one of these will necessarily play in every HTML5-compliant browser. To guarantee playability by all browsers, you may need to encode four separate versions of your video file, including a Flash .flv format as a fallback, and program your HTML `<video>` tag with all four. The browser will play the first file in the list that it can:

```html
<video width="160" height="120" controls autoplay>
  <source src="myVideo.mp4" type='video/mp4; codecs="avc1.42E01E, mp4a.40.2"'>
  <source src="myVideo.webm" type='video/webm; codecs="vp8, vorbis"'>
  <source src="myVideo.ogg" type='video/ogg; codecs="theora, vorbis"'>
  <object type="application/x-shockwave-flash" width="160" height="120" wmode="transparent" data="flvplayer.swf?file=myVideo.flv">
    <param name="movie" value="flvplayer.swf?file=myVideo.flv" />
    <param name="wmode" value="transparent" />
  </object>
</video>
```

For more about making and editing video files, codecs, and distribution methods, see Chapter 6.

**Plug-ins and Players**

Prior to Adobe’s acquisition, when Macromedia introduced Shockwave to allow the animation and interactivity of its flagship tool Director to be embedded into pages viewed, real animation and programmable power became available to web page developers. Later, they added Flash to their animation armory, which also uses Shockwave to create an .swf (Shockwave Flash) version of the native .fla file in order to make it displayable on a web page. **Players** and **plug-ins** became available for other multimedia tools with animation capabilities (for example, RunRev), and the
view came alive as long as the person viewing your page had installed the necessary plug-in on his or her machine. The QuickTime movie format includes the ability to create Virtual Reality (VR) files, also displayed on a web page via a player. Flash and proprietary viewers can be used to present panoramas. Figure 13-11 shows a real estate sales panorama—when you drag the mouse across this player’s window, the scene tracks and rotates in a 360-degree panorama. You can see adjacent rooms, too, by panning the image in a circle.
Chapter 13 Review

Chapter Summary

For your review, here’s a summary of the important concepts discussed in this chapter.

**Employ the basic methods for displaying elements of multimedia on a web page, including using HTML, CSS, and nibbling**

- The World Wide Web was designed as a simple method for delivering text and graphics.
- HTML provides tags for inserting media into HTML documents.
- Use the `<IMG>` tag for inline images, the `<AUDIO>`, `<VIDEO>`, `<OBJECT>`, and `<EMBED>` tags for multimedia objects.
- Specify a font, and even alternates for it, using Cascading Style Sheets (CSS), but remember: browsers can only use fonts already on their computer.
- Use Cascading Style Sheets (CSS) to set text styles across the pages of your web site.
- Use a graphic bitmap if you wish to absolutely control the look of text in your HTML document.
- A viewport 960 pixels wide may be the most flexible choice.
- The overriding principle in designing web pages is to “nibble” away at the content in order to keep the size of the data as small as possible.

**Manipulate the appearance of text on the Web**

- Choose your fonts carefully: viewers may not have your special fonts installed on their computers.
- Use HTML tables to create columns of text.
- Flow text around images using the ALIGN attribute.

**Determine which graphics formats are best suited for different types of images and how they can be manipulated**

- Image formats, GIF, PNG, and JPEG, use built-in compression algorithms to reduce file size.
- GIF images are limited to 8 bits of color depth, or a palette of 256 colors while JPEG and PNG images may contain 24 bits of color depth.
- Use GIF files for line art and images that contain large areas of the same color. Use JPEG for photo-realistic images.
- The GIF89a specification allows for a selected transparency color. You cannot make a JPEG file transparent.
- JPEG and PNG use a powerful but lossy compression method that produces files as much as ten times more compressed than GIF.
- Most browsers allow you to specify an image or color to place in the background of your page in the `<BODY>` tag. Text and images will float on top of this layer.
- CSS is used to control the color of text in a document.
- Background images are automatically tiled, or repeated, across and down the page unless told not to using CSS.
- Placing an image inside an HTML anchor tag makes the graphic image clickable.
- Image maps are pictures with defined hot spots that link to other documents when a user clicks on them.

**Play audio on a web page by embedding the sound within the site**

- Audio play is provided by the `<AUDIO>` tag.
- Always nibble at your sound elements and reduce them to the lowest file sizes that will play acceptably.

**Include animation on a web page**

- Limit animated GIFs to small images, and use a more capable plug-in for animations over larger areas.
- Flash provides animation on the Web.

**Include video on a web page with and without the use of plug-ins**

- Play video using the `<VIDEO>` tag.
### Key Terms

- `<A>` tag (411)
- Ajax (413)
- ALIGN attribute (400)
- `<AUDIO>` tag (395)
- AU format (413)
- autoplay (413)
- `<BODY>` tag (409)
- BORDER attribute (411)
- Cascading Style Sheets (CSS) (392)
- chrome (396)
- codec (414)
- compression algorithm (402)
- controls (413)
- density-independent pixels (DIPs) (397)
- deprecated (393)
- Double Tap (397)
- Drag (397)
- `<EMBED>` tag (395)
- escape sequence (401)
- Flick (397)
- GIF (402)
- HSPACE attribute (400)
- HTML5 (393)
- `<IMG>` tag (395)
- interlaced (407)
- JPEG (403)
- lossless (403)
- lossy (403)
- `<MAP>` tag (411)
- multiframe image (414)
- Object/Embed method (395)
- `<OBJECT>` tag (395)
- Pinch Open/Pinch Closed (397)
- player (415)
- plug-in (415)
- PNG (403)
- preload (413)
- progressive (407)
- `<TABLE>` tag (398)
- Touch and Hold (397)
- transparency (407)
- USEMAP attribute (411)
- `<VIDEO>` tag (395)
- viewport (396)
- VSPACE attribute (400)

### Key Term Quiz

1. The tag used for inline images is the _______________.
2. The attribute used to start a video playing when the page loads is _______________.
3. An image that loads with increasing detail is _______________.
4. Special characters on a web page may be shown using a(n) _______________.
5. Tags no longer included in the most current version of the HTML standard but still recognized by browsers have been _______________.
6. Perhaps the most powerful feature of HTML may be found in the _______________ tag.
7. A(n) _______________ is used to compress and decompress video files.
8. A new “open” format that was developed to replace GIF without requiring licensing fees is the _______________ format.
9. The tag that defines hot spots on an image and the links associated with the hot spots is the _______________ tag.
10. A graphic file format that allows for both transparency and animation is _______________.

418
1. The World Wide Web was originally designed to deliver:
   a. high-quality multimedia
   b. text documents with embedded graphics
   c. data in many formats, including file transfers, chat, and e-mail
   d. streaming media formats
   e. top-secret military information

2. The VGA standard of a $640 \times 480$–pixel monitor showing 256 colors is:
   a. the highest resolution and color depth currently available
   b. the standard used by most browsers
   c. still used by a small number of users
   d. not used by a significant number of users
   e. no longer considered a viable standard

3. Which of these is the only way to ensure that text appears exactly the same across platforms?
   a. Create a bitmap image of the text.
   b. Link to the font at a web site.
   c. Include the font as a download on the server.
   d. Specify the font using Cascading Style Sheets.
   e. Embed the font into the HTML code itself.

4. One of the most important tags, useful for creating columns of information, is:
   a. <BODY>
   b. <BACKGROUND>
   c. <EMBED>
   d. <HEAD>
   e. <TABLE>

5. Which of these statements about the GIF image specification is false?
   a. It can be used to embed animations.
   b. It can only include 256 colors.
   c. It is best used for drawings and cartoons that have only a few colors in them.
   d. It can be saved in an interlaced mode.
   e. It was developed by Microsoft in 1982.

6. The tag that sets text attributes for an entire web page is:
   a. <BODY>
   b. <BACKGROUND>
   c. <EMBED>
   d. <HEAD>
   e. <TABLE>

7. Which of these is not an attribute of the <VIDEO> tag?
   a. Controls
   b. Autoplay
   c. Height
   d. Preload
   e. Background Color

8. You link a graphic image to another document or image by including the image tag inside which tag?
   a. <A>
   b. <URL>
   c. <LINK>
   d. <CLICK>
   e. <GOTO>

9. What can the PNG image format not do?
   a. Provide interlacing
   b. Be transparent in some places
   c. Show millions of colors
   d. Be displayed by browsers
   e. Include advertising animations

10. Which of these is not an image format supported by most browsers?
    a. GIF
    b. JPEG
    c. PNG
    d. DXF
    e. All are supported by most browsers.

11. When you change an image’s mode to Indexed Color, which of these is not a specification to be selected?
    a. the color depth of the converted image
    b. the compression level to be used
    c. the color palette to be used
    d. whether the colors of your image should be dithered or not
    e. All of the above must be selected.
12. What is the most colors that can fit into a GIF palette?
   a. 16
   b. 40
   c. 216
   d. 254
   e. 256
   f. 512

13. When a background image is smaller than the viewport, by default the browser will:
   a. simply ignore the background
   b. display the background in the upper-left corner with the background color filling the rest of the window
   c. tile the background image to fill the window
   d. stretch the background image to fill the window
   e. crash

14. The <AREA> tag within a <MAP> tag defines the shape of a hot spot as a:
   a. circle
   b. polygon
   c. rectangle
   d. All of the above
   e. None of the above

15. When a browser downloads a file it cannot process itself, it can forward the file to an external application for processing. This external application is sometimes called a:
   a. plug-in
   b. helper application
   c. CGI script
   d. JavaScript
   e. Java applet

■ Essay Quiz

1. How is the World Wide Web used today? How do you use it? What types of sites do you visit? How has it changed in your personal experience?

2. Bandwidth limitations impose serious limitations on presenting multimedia over the Web. What tools and strategies does a multimedia developer have to deal with the limitations of bandwidth?

3. HTML was never designed to include multimedia. What are the original design limitations of HTML? How has it been stretched to accommodate other media? How has the browser architecture been adapted to allow other media? What tools and strategies does a multimedia developer have to deal with these limitations?

4. List and describe the most important tags for multimedia in HTML.

5. What are the most common graphics file formats in use on the Web today? Discuss what each is best suited for, its limitations, and its capabilities.

■ Lab Projects

■ Project 13.1

Select three different web pages, each from a different site and with differing layouts. Print out the web pages. Examine them carefully.

   Draw lines on the printout where you think the various cells of the tables were used to construct the pages.
Project 13.2

Again select three different web pages, each from a different site and with differing layouts. All browsers include the ability to view the source code of the web page. View the HTML source, and then print it out. Identify head, image, anchor, body, and table tags in the page by circling them.

(If you have access to a visual web page editor, import the source from the three pages in Lab Project 13.1 and note how the tables are laid out.)

Project 13.3

Select three different sites that present audio and three sites that present video, virtual reality, or animation. What data formats are used for the multimedia? Are different data rates offered? What plug-ins are required? Make a note of the URLs, media formats, data rates, and required plug-ins or helper applications.

Project 13.4

Again select three different web pages, each from a different site and with differing layouts. Reduce the size of the browser window. Does the layout contract past a certain point? Now widen the browser window. Does the layout expand beyond a certain point? Make notes of your findings, including the site’s URL.

Project 13.5

Locate a site that uses Cascading Style Sheets. This may be visible in the HTML code itself within <style> tags, or else they may be referenced externally in a <LINK> tag. As in Project 13.4, print out the style sheet if it is linked. Circle any fonts, colors, sizes, and other style information.

Project 13.6

Take a digital photograph of yourself. Determine the coordinates for a circle around your right eye and a polygon around your nose. Create a <MAP> tag with shape attributes and using HTML code place the image onto a web page so that when you click on your right eye the browser sends you to www.facebook.com; when you click on your nose, you are sent to www.recipes.com.