hansen media

HTML Basic Tutorial

Overview

HTML documents are plain-text (also known as ASCII) files that can be created using any text editor, such as Emacs or vi on a Unix system, SimpleText on a Macintosh, or Notepad on an IBM-compatible machine. You can also use word-processing software if you save the document as “text only.” In order to make HTML documents available to anyone with a web browser on the Internet, you will need to have them posted on a server. A webmaster at an educational institution, a corporation, or a commercial Internet Service Provider (ISP) will be able to assist you.

There are many WYSIWYG editors available to automate HTML development and new tools for creating and managing entire sites. You may wish to try them after you learn some of the basics of HTML tagging. It is useful to know enough HTML to code a document before you become reliant on a WYSIWYG editor.

HTML Tags

An “element” is a fundamental component of the structure of a text document. Some examples of elements are heads, tables, paragraphs, and lists. HTML “tags” are used to mark the elements of a file for your browser. Elements may contain plain text, other elements, or both.

Tags identify the various elements in an HTML document. HTML tags consist of a left angle bracket (<), a tag name, and a right-angle bracket (>). Tags are usually paired (e.g., <H1> and </H1>) to start and end the tag instruction. The end tag looks like the start tag except a slash (/) precedes the text within the brackets. Not all tags are supported by all web browsers. A browser usually ignores a tag that it does not support.

Some elements may include an “attribute”, which is additional information included inside the start tag. For example, you can specify the alignment of images (top, middle, or bottom) by including the appropriate attribute with the image source HTML code. Tags that have optional attributes are noted below.

Note: HTML is not case sensitive. <title> is equivalent to <TITLE> or <TiTlE>. There are a few exceptions noted in the Escape Sequences section below.

A Minimal HTML Document

Every HTML document should contain certain standard HTML tags. Each document consists of head and body text. The head contains the title, and the body contains the actual text that is made up of paragraphs, lists, and other elements. Browsers expect specific information because they are programmed according to HTML and SGML specifications.

Required elements are shown in this sample document:

```
<html>
head>
<TITLE>A Simple HTML Example</TITLE>
</head>
<body> <H1>HTML is Easy To Learn</H1>
 <P>Welcome to the world of HTML.
This is the first paragraph. While short, it is still a paragraph!</P>
<P>And this is the second paragraph.</P>
</body>
</html>
```

The required elements are the <HTML>, <HEAD>, <TITLE>, and <BODY> tags, along with their corresponding end tags. Because you should include these tags in each file, you might want to create a template file with them. Some browsers will format your HTML file correctly even if these tags are not included, while other browsers will not.

To view a copy of the file that your browser reads to generate the information in your current window, select View Source (or the equivalent) from the browser menu. The file contents, along with all the HTML tags, are displayed in a new window.

This is an excellent way to discover how HTML is used and to learn tips and constructs. Of course, the HTML that you find might not be technically correct. Once you become familiar with HTML and check the many online and hard-copy references on the subject, you will learn to distinguish between “good” and “bad” HTML. You can save a source file with the HTML codes and use it as a template for one of your web pages, or you can modify the format to suit your purposes.

Markup Tags

HTML

This element tells your browser that the file contains HTML-coded information. The file extension .html also indicates that it is an HTML document. If you are restricted to 8.3 file names, use only .htm for your extension.

Head

The head element identifies the first part of your HTML-coded document. It contains the title, which is shown as part of your browser’s window.

Title

The title element contains your document title and identifies its content in a global context. The title is displayed somewhere on the browser window (usually at the top), but not within the text area. The title is also displayed on someone’s hot list or bookmark list, so choose something descriptive, unique, and relatively short. A title is also used during a WAIS search of a server.

For example, you might include a shortened title of a book with the chapter contents; for example, “NCSA Mosaic Guide (Windows): Installation.” This tells the software name, the platform, and the chapter contents, which is more useful than calling the document Installation. Generally, titles should be restricted to 64 characters or fewer.

Body

The second part of your HTML document is the body, which contains the content of your document (displayed within the text area of your browser window). The tags explained below are used in the body of your HTML document.

Headings

HTML has six levels of headings, numbered 1 through 6, with 1 being the most prominent. Web browsers display headings in larger and/or bolder fonts than normal body text. The first heading in each document should be tagged <H1>.

The syntax of the heading element is <H*n*>Text of heading</H*n*> where *n* is a number between 1 and 6 specifying the level of the heading.

Do not skip levels of headings in your document. For example, don’t start with a level-one heading (<H1>) and follow it with a level-three (<H3>) heading.

Paragraphs

Unlike documents in most word processors, carriage returns in HTML files aren’t significant. It doesn’t matter how long your lines of text are, but it is best to restrict them to fewer than 72 characters in length. Word wrapping can occur at any point in your source file, and multiple spaces are collapsed into a single space by your browser.

In the example shown in the “Minimal HTML Document” section, the first paragraph is coded as

```
<P>Welcome to the world of HTML.
this is the first paragraph. While short, it is
still a paragraph!</P>
```

In the source file there is a line break between the sentences. A web browser ignores this line break and starts a new paragraph only when it encounters another <P> tag.

Important: You must indicate paragraphs with <P> elements. A browser ignores any indentations or blank lines in the source text. Without <P> elements, the document becomes one large paragraph. (One exception is text tagged as "preformatted," which is explained below.) For example, the following would produce identical output to the "Minimal HTML" example:

```
<H1>Level-one heading</H1> <P>Welcome to the world of HTML. This is the first
paragraph. While short, it is still a paragraph! </P>

<P>And this is the second paragraph.</P>
```

To preserve readability in HTML files, put headings on separate lines, use a blank line or two where it helps identify the start of a new section, and separate paragraphs with blank lines in addition to the <P> tags. These extra spaces help you when you edit your files, but your browser will ignore the extra spaces because it has its own set of rules on spacing that do not depend on the spaces you put in your source file.

Note: The </P> closing tag can be omitted because browsers understand that when they encounter a <P> tag, it implies that there is an end to the previous paragraph.

Using the <P> and </P> as a paragraph container means that you can center a paragraph by including the ALIGN=alignment attribute in your source file. The formatted version is below the tagged version.

```
<P ALIGN=CENTER>
This is a centered paragraph.</P>
```

This is a centered paragraph.

Lists

HTML supports unnumbered, numbered, and definition lists. You can nest lists but use this feature sparingly because too many nested items can make a document difficult to follow.

Unnumbered Lists

To make an unnumbered, bulleted list:

1. Start with an opening list <UL> (for unnumbered list) tag.
2. Enter the <LI> (list item) tag followed by the individual item; no closing </LI> tag is needed.
3. End the entire list with a closing list </UL> tag.

Below is a sample three-item list:

```
<UL>
<LI> apples
<LI> bananas
<LI> grapefruit </UL>
```

The output is:

- apples
- bananas

- grapefruit

The <LI> items can contain multiple paragraphs. Indicate the paragraphs with the <P> paragraph tags.

Numbered Lists

A numbered list (also called an ordered list, from which the tag name derives) is identical to an unnumbered list, except it uses <OL> instead of <UL>. The items are tagged using the same <LI> tag. The following HTML code:

```
<OL>
<LI> oranges
<LI> peaches
<LI> grapes </OL>
```

produces this formatted output:

1. oranges
2. peaches
3. grapes

Definition Lists

A definition list (coded as <DL>) usually consists of alternating a definition term (coded as <DT>) and a definition (coded as <DD>). Web browsers generally format the definition on a new line.

The following is an example of a definition list:

```
<DL>
<DT> NCSA
<DD> NCSA, the National Center for Supercomputing
Applications, is located on the campus of the University
of Illinois at Urbana-Champaign.
<DT> Cornell Theory Center
<DD> CTC is located on the campus of Cornell University
in Ithaca, New York.
</DL>
```

The output looks like:

NCSA

> NCSA, the National Center for Supercomputing Applications, is located on the campus of the University of Illinois at Urbana-Champaign.

Cornell Theory Center

> CTC is located on the campus of Cornell University in Ithaca, New York.

The <DT> and <DD> entries can contain multiple paragraphs (indicated by <P> paragraph tags), lists, or other definition information.

The COMPACT attribute can be used routinely in case your definition terms are short. If, for example, you are showing some computer options, the options may fit on the same line as the start of the definition.

```
<DL COMPACT>
<DT> -i
<DD>invokes NCSA Mosaic for Microsoft Windows using the initialization
file defined in the path
<DT> -k
<DD>invokes NCSA Mosaic for Microsoft Windows in kiosk mode
</DL>
```

The output looks like:

-i   invokes NCSA Mosaic for Microsoft Windows using the initialization file
defined in the path –

k   invokes NCSA Mosaic for Microsoft Windows in kiosk mode

Nested Lists

Lists can be nested. You can also have a number of paragraphs, each containing a nested list, in a single list item. Here is a sample nested list:

```
<UL>
<LI> A few New England states:
<UL>
Vermont
New Hampshire
Maine
</UL>
<LI> Two Midwestern states:
<UL>
Michigan
Indiana
</UL>
</UL>
```

The nested list is displayed as:

- A few New England states:
    - Vermont
    - New Hampshire
    - Maine
- Two Midwestern states:
    - Michigan
    - Indiana

Preformatted Text

Use the <PRE> tag (which stands for preformatted) to generate text in a fixed-width font. This tag also makes spaces, new lines, and tabs significant (multiple spaces are displayed as multiple spaces, and lines break in the same locations as in the source HTML file). This is useful for program listings, among other things. For example, the following lines:

```
<PRE>
      #!/bin/csh
      cd $SCR
      cfs get mysrc.f:mycfsdir/mysrc.f
      cfs get myinfile:mycfsdir/myinfile
      fc -02 -o mya.out mysrc.f
      mya.out
      cfs save myoutfile:mycfsdir/myoutfile
      rm *
</PRE>
```

display as:

```
#!/bin/csh
cd $SCR
cfs get mysrc.f:mycfsdir/mysrc.f
cfs get myinfile:mycfsdir/myinfile
fc -02 -o mya.out mysrc.f
mya.out
cfs save myoutfile:mycfsdir/myoutfile
rm *
```

The <PRE> tag can be used with an optional WIDTH attribute that specifies the maximum number of characters for a line. WIDTH also signals your browser to choose an appropriate font and indentation for the text. Hyperlinks can be used within <PRE> sections. You should avoid using other HTML tags within <PRE> sections, however.

Note that because <, >, and & have special meanings in HTML, you must use their escape sequences (&lt;, &gt;, and &amp;, respectively) to enter these characters. See the section on escape sequences for more information.

Extended Quotations

Use the <BLOCKQUOTE> tag to include lengthy quotations in a separate block on the screen. Most browsers change the margins for the quotation to separate it from surrounding text. In the example:

```
<BLOCKQUOTE> <P>Omit needless words.</P> <P>Vigorous writing is
concise. A sentence should contain no unnecessary words, a
paragraph no unnecessary sentences, for the same reason that a
drawing should have no unnecessary lines and a machine no
unnecessary parts.</P> -William Strunk, Jr., 1918 </BLOCKQUOTE>
```

the result is:

> Omit needless words.
>
> Vigorous writing is concise. A sentence should contain no unnecessary words, a paragraph no unnecessary sentences, for the same reason that a drawing should have no unnecessary lines and a machine no unnecessary parts.
>
> -William Strunk, Jr., 1918

Addresses

The <ADDRESS> tag is generally used to specify the author of a document, a way to contact the author (e.g., an email address), and a revision date. It is usually the last item in a file.

For example, the last line of the online version of this guide is:

```
<ADDRESS> A Beginner’s Guide to HTML / NCSA /pubs@ncsa.uiuc.edu / revised
April 96 </ADDRESS>
```

The result is:

> *A Beginner’s Guide to HTML / NCSA / pubs@ncsa.uiuc.edu / revised April 96*

Note: <ADDRESS> is not used for postal addresses. See “Forced Line Breaks” below to learn how to format postal addresses.

Forced Line Breaks/Postal Addresses

The <BR> tag forces a line break with no extra (white) space between lines. Using <P> elements for short lines of text such as postal addresses results in unwanted additional white space. For example, with <BR>:

```
National Center for Supercomputing Applications<BR>
605 East Springfield Avenue<BR>
Champaign, Illinois 61820-5518<BR>
```

The output is:

> National Center for Supercomputing Applications
> 605 East Springfield Avenue
> Champaign, Illinois 61820-5518

Horizontal Rules

The <HR> tag produces a horizontal line the width of the browser window. A horizontal rule is useful when you are separating sections of your document. For example, many people add a rule at the end of their text and before the <ADDRESS> information.

You can vary a rule size (thickness) and width (the percentage of the window covered by the rule). Experiment with the settings until you are satisfied with the presentation. For example, to create a four-pixel-thick centered line half as wide as the window, use the tag <HR SIZE=4 WIDTH="50%">.

Character Formatting

HTML has two styles for individual words or sentences: logical and physical. Logical styles tag text according to its meaning, while physical styles indicate the specific appearance of a section. For example, in the preceding sentence, the words “logical styles” might be tagged as a “definition.” The same effect (formatting those words in italics) could be achieved with a different tag that tells your browser to “put these words in italics.”

Note: Some browsers do not attach any style to the <DFN> tag, so you cannot be assured that all users would see the tagged phrase in italics.

Logical Versus Physical Styles

In the ideal SGML universe, content is divorced from presentation. Thus, SGML tags a level-one heading as a level-one heading, but it does not specify that the level-one heading should be, for instance, 24-point bold Times centered. The advantage of this approach (it’s similar in concept to style sheets in many word processors) is that if you decide to change level-one headings to 20-point left-justified Helvetica, all you have to do is change the definition of the level-one heading in your web browser. Indeed, many browsers today let you define how you want the various HTML tags rendered onscreen.

Another advantage of logical tags is that they help enforce consistency in your documents. It’s easier to tag something as <H1> than to remember that level-one headings are 24-point bold Times centered, for example. Consider the <STRONG> tag, which most browsers render in bold text. However, it is possible for a reader to prefer these sections to be displayed in red. Logical styles offer this flexibility.

Of course, if you want something to be displayed in italics, for example, and do not want a browser’s setting to display it differently, use physical styles. Physical styles, therefore, offer consistency because something you tag a certain way will always be displayed that way for readers of your document.

Try to be consistent about which type of style you use. If you tag with physical styles, do so throughout a document. If you use logical styles, stick with them within a document. The version 4.0 release of HTML strongly advocates the use of Cascading Style Sheets (CSS) to apply styles in HTML documents.

Logical Styles

<DFN> for a word being defined. Typically displayed in italics.
(*NCSA Mosaic is a World Wide Web browser.*)

<EM> for emphasis. Typically displayed in italics.
(*Consultants cannot reset your password unless you call the help line.*)

<CITE> for titles of books, films, etc. Typically displayed in italics.
(*For Whom the Bell Tolls)*

<CODE> for computer code. Displayed in a fixed-width font.
(`The <stdio.h> header file`)

<KBD> for user keyboard entry. Typically displayed in plain fixed-width font.
(`Enter passwd to change your password.)`

<SAMP> for a sequence of literal characters. Displayed in a fixed-width font.
(`Segmentation fault: Core dumped.)`

<STRONG> for strong emphasis. Typically displayed in bold.
(**Note: Always check your links to see how text is displayed.***)*

<VAR> for a variable, where you will replace the variable with specific information. Typically displayed in italics.
*(*rm *filename* deletes the file.*)*

Physical Styles

<B> bold text
<I> italic text
<TT> typewriter text (`fixed-width font`)

Escape Sequences (a.k.a. Character Entities)

Character entities have two functions:

1. Escaping special characters.
2. Displaying other characters not available in the plain ASCII character set (primarily characters with diacritical marks).

Three ASCII characters—the left angle bracket (<), the right angle bracket (>), and the ampersand (&)—have special meanings in HTML and therefore cannot be used “as is” in text. (The angle brackets are used to indicate the beginning and end of HTML tags, and the ampersand is used to indicate the beginning of an escape sequence.) Double quote marks may be used as-is, but a character entity may also be used (&quot;).

To use one of the three characters in an HTML document, you must enter its escape sequence instead:

`&lt;`	escape sequence for <

`&gt;`	escape sequence for >
`&amp;`	escape sequence for &

Additional escape sequences support accented characters, such as:

`&ouml;`	escape sequence for a lowercase o with an umlaut: ö
`&ntilde;`	escape sequence for a lowercase n with a tilde: ñ
`&Egrave;`	escape sequence for an uppercase E with a grave accent: È

You can substitute other letters for the ö, ñ, and È shown above. Refer to the "HTML Coded Characters" table under the HTML entry for a complete list of special characters.

Note: Unlike the rest of HTML, the escape sequences are case sensitive. You cannot, for instance, use &LT; instead of &lt;.

Linking

The chief power of HTML comes from its ability to link text and/or an image to another document or section of a document. A browser highlights the identified text or image with color and/ or underlines to indicate that it is a hypertext link (often shortened to hyperlink or link).

HTML's single hypertext-related tag is <A>, which stands for anchor. To include an anchor in your document:

1. Start the anchor with <A (include a space after the A).
2. Specify the document you're linking to by entering the parameter HREF="filename" followed by a closing right angle bracket (>).
3. Enter the text that will serve as the hypertext link in the current document.
4. Enter the ending anchor tag </A>. (No space is needed before the end anchor tag.)

Here is a sample hypertext reference in a file called US.html:

```
<A HREF="MaineStats.html">Maine</A>
```

This entry makes the word *Maine* the hyperlink to the document *MaineStats.html*, which is in the same directory as the first document.

Relative Path Names vs. Absolute Path Names

You can link to documents in other directories by specifying the relative path from the current document to the linked document. For example, a link to a file NYStats.html located in the subdirectory AtlanticStates would be:

```
<A HREF= "AtlanticStates/NYStats.html">New York</A>
```

These are called relative links because you specify the path to the linked file relative to the location of the current file. You can also use the absolute path name (the complete URL) of the file, but relative links are more efficient in accessing a server.

Path names use the standard Unix syntax. The Unix syntax for the parent directory (the directory that contains the current directory) is "..". (For more information consult a beginning Unix reference text.)

If you were in the NYStats.html file and were referring to the original document US.html, your link would look like this:

```
<A HREF="../US.html">United States</A>
```

Generally, relative links should be used for the following reasons:

1. It is easier to move a group of documents to another location (because the relative path names will still be valid).
2. It is more efficient connecting to the server.
3. There is less to type.

However, use absolute path names when linking to documents that are not directly related. For example, consider a group of documents that comprise a user manual. Links within this group should be relative links. Links to other documents (perhaps a reference to related software) should use full path names. This way if you move the user manual to a different directory, none of the links would have to be updated.

URLs

The World Wide Web uses uniform resource locators (URLs) to specify the location of files on other servers. A URL includes the type of resource being accessed (e.g., web, gopher, WAIS), the address of the server, and the location of the file. The syntax is *scheme://host.domain [:port]/path/ filename* where *scheme* is one of the following types:

file	A file on your local system.
ftp	A file on an anonymous FTP server.
http	A file on a World Wide Web server.
gopher	A file on a gopher server.
WAIS	A file on a WAIS server.
news	A Usenet newsgroup.
telnet	A connection to a Telnet-based service.

The *port* number can generally be omitted, unless it is specifically requested. For example, to include a link in your document to the NCSA primer (upon which this appendix is based), enter:

```
<A HREF=”http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimer.html”>
NCSA’s Beginner’s Guide to HTML</A>
```

This entry makes the text *NCSA's Beginner's Guide to HTML* a hyperlink to your document.

Links to Specific Sections

Anchors can also be used to move a reader to a particular section in a document (either the same or a different document) rather than to the top, which is the default. This type of an anchor is commonly called a "named anchor" because to create the links, you insert HTML names within the document.

Internal hyperlinks are used to create a "table of contents" at the top of a long document. These hyperlinks move you from one location to another in the same document. You can also link to a specific section in another document.

Links Between Sections of Different Documents

Suppose you want to set a link from document A (documentA.html) to a specific section in another document (MaineStats.html).

Enter the HTML coding for a link to a named anchor:

```
documentA.html:

In addition to the many state parks, Maine is also home to <a
href="MaineStats.html#ANP">Acadia National Park</a>.
```

Think of the characters after the hash mark (#) as a tab within the MaineStats.html file. This tab tells your browser what should be displayed at the top of the window when the link is activated. In other words, the first line in your browser window should be the Acadia National Park heading.

Next, create the named anchor (in this example "ANP") in MaineStats.html:

```
<H2><A NAME="ANP">Acadia National Park</a></H2>
```

With both of these elements in place, you can bring a reader directly to the Acadia reference in MaineStats.html.

Note: You cannot make links to specific sections within a different document unless you have permission to edit the coded source of that document or that document already contains in-document named anchors. For example, you could include named anchors to the NCSA online HTML guide in a document you are writing because there are named anchors in this guide. But if this document did not have named anchors, you could not make a link to a specific section because you cannot edit the original file on NCSA's server.

Links to Specific Sections Within the Current Document

The technique is the same except the file name is omitted. For example, to link to the ANP anchor from within MaineStats, enter:

```
...More information about <A HREF="#ANP">Acadia National Park</A> is
available elsewhere in this document.
```

Be sure to include the <A NAME=> tag where you want the link to jump to (<H2> <A NAME="ANP">Acadia National Park</A></H2>).

Named anchors are useful when you think readers will print an entire document or when you have a lot of short information you want to place online in one file.

Mailto

You can make it easy for a reader to send email to a specific person or mail alias by including the mailto attribute in a hyperlink. The format is:

```
<A HREF="mailto:emailinfo@host">Name</a>
```

For example, enter:

```
<A HREF="mailto:brad@hansenmedia.com">Brad Hansen</a>
```

This creates a mail window configured to send email. You may put any valid Internet address in your mailto window.

Inline Images

Inline Images

Most web browsers can display inline images (images next to text) that are in X BitMap (XBM), GIF, or JPEG format. Another image format recognized by current web browsers is the Portable Network Graphics (PNG) format. Each image takes time to process and slows down the initial display of a document. Carefully select your images and the number of images in a document.

To include an inline image, enter:

```
<IMG SRC=ImageName>
```

where *ImageName* is the URL of the image file.

The syntax for <IMG SRC> URLs is identical to that used in an anchor HREF. If the image file is a GIF file, then the file name part of *ImageName* must end with .gif. File names of X BitMap images must end with .xbm; JPEG image files must end with .jpg or .jpeg; and Portable Network Graphic files must end with .png.

Image Size Attributes

Include two other attributes on <IMG> tags to tell your browser the size of the images it is downloading with the text. The HEIGHT and WIDTH attributes let the browser set aside the appropriate space (in pixels) for the images as it downloads the rest of the file. (Get the pixel size from your image-processing software, such as Adobe Photoshop.)

For example, to include a self-portrait image in a file along with the portrait’s dimensions, enter:

```
<IMG SRC=SelfPortrait.gif HEIGHT=100 WIDTH=65>
```

Note: Some browsers use the HEIGHT and WIDTH attributes to stretch or shrink an image to fit into the allotted space when the image does not exactly match the attribute numbers. Not all browser developers think stretching/shrinking is a good idea. Do not assume your readers will have access to this feature. Check your dimensions and use the correct ones.

Aligning Images

You have some flexibility when displaying images. You can have images separated from text and aligned to the left, right, or center, or you can have an image aligned with text. Try several possibilities to see how your information looks best.

Aligning Text with an Image

By default, the bottom of an image is aligned with the baseline of the text that follows it. You can align images to the top or center of a paragraph using the ALIGN= attributes TOP and CENTER.

To align text with the top of the image, use the tag:

```
(<IMG SRC = "MyImage.gif" ALIGN=TOP>)
```

Note that the browser aligns only one line and then jumps to the bottom of the image for the rest of the text.

For text that is centered on the image, use the tag:

```
(<IMG SRC = "MyImage.gif" ALIGN=CENTER>)
```

Only one line of text is centered; the rest is below the image.

Images Without Text

To display an image without any associated text (e.g., your organization's logo), make it a separate paragraph. Use the paragraph ALIGN= attribute to center the image or adjust it to the right side of the window as shown on the following page:

```
<p ALIGN=CENTER>
<IMG SRC = "MyImage.gif">
</p>
```

The image is centered, and the paragraph starts below it, left-justified.

Alternate Text for Images

Some World Wide Web browsers (primarily those that run on VT100 terminals) cannot display images. Some users turn off image loading if they have a slow connection, even if their software can display images. HTML provides a mechanism to tell readers what they are missing on your pages.

The ALT attribute lets you specify text to be displayed instead of an image. In the following example *UpArrow.gif* is the picture of an upward pointing arrow.

```
<IMG SRC="UpArrow.gif" ALT="Up">
```

With graphics-capable viewers that have image-loading turned on, the up arrow graphic is visible. With image-loading turned off, or with a VT100 browser, the word *Up* is shown in your window. It's a good idea to include alternate text for each image you use in your document as a courtesy for your readers.

Background Graphics

Newer versions of web browsers can load an image and use it as a background when displaying a page. Some people like background images, and some don't. In general, if you want to include a background, make sure your text can be read easily when displayed on top of the image. Background images can be a texture or an object, such as a logo. You create the background image as you do any image.

However, you have to create only a small piece of the image. Using a tiling feature, a browser takes the image and repeats it across and down to fill your browser window. Basically, you generate one image, and the browser replicates it enough times to fill your window. This action is automatic when you use the background tag.

Below is the tag to include a background image included in the <BODY> statement as an attribute:

```
<BODY BACKGROUND="filename.gif">
```

Background Color

By default, browsers display text in black on a gray background. However, you can change both elements if you want. Some HTML authors select a background color and coordinate it with a change in the color of the text.

Always preview changes like this to make sure your pages are readable. (For example, many people find red text on a black background difficult to read.)

You change the color of text, links, visited links, and active links using attributes of the <BODY> tag. For example, enter:

```
<BODY BGCOLOR="#000000" TEXT="#FFFFFF" LINK="#9690CC">
```

This creates a window with a black background (BGCOLOR), white text (TEXT), and silvery hyperlinks (LINK).

The six-digit number and letter combinations represent colors by giving their RGB (red, green, blue) value. The six digits are three two-digit numbers in sequence, representing the amount of red, green, or blue as a hexadecimal value in the range 00–FF. For example, 000000 is black (no color at all), FF0000 is bright red, and FFFFFF is white (fully saturated with all three colors). These number and letter combinations are cryptic. The six-digit color codes that map specific colors in HTML 3.2 and later are provided in the “HTML Color Table” under the HTML entry.

External Images, Sounds, and Animation

You may want to have an image open as a separate document when a user activates a link on either a word or a smaller inline version of the image included in your document. This is called an external image, and it is useful if you do not wish to slow down the loading of the main document with large inline images.

To include a reference to an external image, enter:

```
<A HREF="MyImage.gif">link anchor</A>
```

You can also use a smaller image as a link to a larger image. Enter:

```
<A HREF="LargerImage.gif"><IMG SRC="SmallImage.gif"></A>
```

The reader sees the SmallImage.gif image and clicks it to open the LargerImage.gif file.

Use the same syntax for links to external animation and sounds. The only difference is the file extension of the linked file. For example, <A HREF="AdamsRib.mov">link anchor</A> specifies a link to a QuickTime movie. Some common file types and their extensions are:

File Type	Extension
ASCII text	.txt
HTML document	.htm (or .html)
GIF image	.gif
TIFF image	.tif (or .tiff)
X BitMap image	.xbm
JPEG image	.jpg (or .jpeg)
PNG image	.png
PostScript file	.ps
AIFF sound file	.aif (or .aiff)
AU sound file	.au
WAV sound file	.wav
QuickTime movie	.mov
MPEG movie	.mpg (or .mpeg)

Keep in mind your intended audience's access to helper applications and plug-ins. For example, most Unix workstations cannot view QuickTime movies.

Tables

Before HTML tags for tables were finalized, authors had to format their tabular information carefully within <PRE> tags, counting spaces and previewing their output. Tables are useful for presenting tabular information, and they are a boon to creative HTML authors who use the table tags to present their regular web pages.

Think of your tabular information in light of the coding explained below. A table has heads where you explain what the columns/rows include, rows for information, and cells for each item. In the following table, the first column contains the header information, each row explains an HTML table tag, and each cell contains a paired tag or an explanation of the tag's function.

Table Elements

Element	Description
<TABLE>	Defines a table in HTML. If the BORDER attribute is present, your browser displays the table with a border.
<CAPTION>	Defines the caption for the title of the table. The default position of the title is centered at the top. The ALIGN=BOTTOM attribute can be used to position the caption below the table. *Note:* Any kind of markup tag can be used in the caption.

<TR>	Specifies a table row within a table. You may define default attributes for the entire row: ALIGN (LEFT, CENTER, RIGHT) and/or VALIGN (TOP, MIDDLE, BOTTOM). See “Table Attributes” at the end of this table for more information.
<TH>	Defines a table header cell. By default the text in this cell is bold and centered. Table header cells may contain other attributes to determine the characteristics of the cell and/or its contents. See “Table Attributes” at the end of this table for more information.
<TD>	Defines a table data cell. By default, the text in this cell is left-aligned and centered vertically. Table data cells may contain other attributes to determine the characteristics of the cell and/or its contents. See “Table Attributes” at the end of this table.

Table Attributes

Note: Attributes defined within <TH> ... </TH> or <TD> ... </TD> cells override the default alignment set in a <TR> ... </TR>. Attributes are listed below, along with their description.

`ALIGN (LEFT, CENTER, RIGHT)`
Horizontal alignment of a cell.

`VALIGN (TOP, MIDDLE, BOTTOM)`
Vertical alignment of a cell.

`COLSPAN=n`
The number (n) of columns a cell spans.

`ROWSPAN=n`
The number (n) of rows a cell spans.

`NOWRAP`
Turn off word wrapping within a cell.

General Table Format

The general format of a table looks like this:

<TABLE>	- start of table definition
<CAPTION> caption </CAPTION>	- caption definition
<TR> <TH> cell contents </TH>	- start of first row definition - first cell in row 1 (a head)

<TH> cell contents </TH> </TR>	- last cell in row 1 (a head) - end of first row definition
<TR> <TD> cell contents </TD>	- start of second row definition - first cell in row 2
<TD> cell contents </TD> </TR>	- last cell in row 2 - end of second row definition
<TR> <TD> cell contents </TD>	- start of last row definition - first cell in last row
<TD> cell contents </TD> </TR>	- last cell in last row - end of last row definition
</TABLE>	- end of table definition

The <TABLE> and </TABLE> tags must surround the entire table definition. The first item inside the table, CAPTION, is optional. Then you can have any number of rows defined by the <TR> and </TR> tags. Within a row you can have any number of cells defined by the <TD>...</TD> or <TH>...</TH> tags. Each row of a table is formatted independently of the rows above and below it. This lets you easily display tables like the one above with a single cell, such as *Table Attributes,* spanning columns of the table.

Tables for Nontabular Information

Some HTML authors use tables to present nontabular information. For example, because links can be included in table cells, some authors use a table with no borders to create "one" image from separate images. Browsers that can display tables properly show the various images seamlessly, making the created image seem like an image map (one image with hyperlinked quadrants). Using table borders with images can create an impressive display as well.

Fill-Out Forms

Web forms let you return information to a web server for some action. For example, suppose you collect names and email addresses so you can email information to people who request it. For each person who enters his or her name and address, you need some information to be sent and the respondent's particulars added to a database. This information is usually processed by a script or program written in Perl or another language that manipulates text, files, and information.

The forms are not hard to code. They follow the same constructs as other HTML tags. What can be difficult is the program or script that takes the data submitted in a form and processes it. Because of the need for specialized scripts to handle the incoming form information, fill-out forms are not discussed in this primer.

Troubleshooting

Avoid Overlapping Tags

Consider this example of HTML:

```
<B>This is an example of <DFN>overlapping</B> HTML tags.</DFN>
```

The word *overlapping* is in both the <B> and <DFN> tags. A browser might be confused by this coding and might not display it the way you intend. The only way to know is to check each popular browser (a time-consuming proposition).

In general, avoid overlapping tags. Look at your tags and try pairing them up. Tags (with the obvious exceptions of elements whose end tags may be omitted, such as paragraphs) should be paired without an intervening tag in between. Look again at the example above. You cannot pair the bold tags without another tag in the middle (the first definition tag). Try matching your coding up like this to see if you have any problem areas that should be fixed before you release your files to a server.

Embed Only Anchors and Character Tags

HTML protocol allows you to embed links within other HTML tags:

```
<H1><A HREF="Destination.html">My heading</A></H1>
```

Do not embed HTML tags within an anchor:

```
<A HREF="Destination.html">
<H1>My heading</H1>
</A>
```

Although most browsers currently handle this second example, the official HTML specifications do not support this construct, and your file may not work with future browsers. Remember that browsers can be forgiving when displaying improperly coded files. However, that forgiveness may not continue in the next version of the software! When in doubt, code your files according to the HTML specification.

Character tags modify the appearance of the text within other elements:

```
<UL>
<LI><B>A bold list item</B>
<LI><I>An italic list item</I>
</UL>
```

Avoid embedding other types of HTML element tags. For example, you might be tempted to embed a heading within a list in order to make the font size larger:

```
<UL>
<LI><H1>A large heading</H1>
<LI><H2>Something slightly smaller</H2>
</UL>
```

Although some browsers handle this well, formatting of such coding is unpredictable because it is undefined. For compatibility with all browsers, avoid these kinds of constructs. Cascading Style Sheets (CSS) is the preferred method of defining style parameters in HTML documents.

What's the difference between embedding a <B> within a <LI> tag as opposed to embedding a <H1> within a <LI>? Within HTML the semantic meaning of <H1> is that it's the main heading of a document and that it should be followed by the content of the document. Therefore, it doesn't make sense to find a <H1> within a list.

Character formatting tags also are generally not additive. For example, you might expect that <B><I>sometext</I></B> would produce bold-italic text. On some browsers it does; other browsers interpret only the innermost tag.

Final Steps

Validate your code. When you put a document on a web server, check the formatting and each link (including named anchors). Ideally, you will have someone else read through and comment on your files before you consider a document finished.

You can run your coded files through an HTML validation service that will tell you if your code conforms to accepted HTML. If you are not sure your coding conforms to HTML specifications, this can be a useful teaching tool. Fortunately, the service lets you select the level of conformance you want for your files (i.e., strict, level 2, level 3). If you want to use some codes that are not officially part of the HTML specifications, this latitude is helpful.

Dummy Images

When an <IMG SRC> tag points to an image that does not exist, a dummy image is substituted by your browser software. When this happens during your final review of your files, make sure that the referenced image does in fact exist, that the hyperlink has the correct information in the URL, and that the file permission is set appropriately (world-readable). Then check online again.

Update Your Files

If the contents of a file are static (such as a biography of George Washington), no updating should be needed. Time-sensitive documents should be changed often.

Updating is important when the file contains information such as a weekly schedule or a deadline for a program funding announcement. Remove out-of-date files or note why something that seems dated is still on a server (e.g., the program requirements remain the same for the next cycle so the file is still available as an interim reference).

Browsers Differ

Web browsers display HTML elements differently. Remember that not all codes used in HTML files are interpreted by all browsers. Any code a browser does not understand is usually ignored.

You could spend a lot of time making your file look perfect using your current browser. If you check that file using another browser, it may display quite differently. Hence, these words of advice: code your files using correct HTML. Leave the interpreting to the browsers and hope for the best.

Commenting Your Files

You might want to include comments in your HTML files. Comments in HTML are like comments in a computer program—the text you enter is not used by the browser in any formatting and is not directly viewable by the reader, just as computer program comments are not used and are not viewable. The comments are accessible if a reader views the source file, however.

Comments such as the name of the person updating a file, the software and version used in creating a file, or the date that a minor edit was made are the norm.

To include a comment, enter:

```
<!-- your comments here -->
```

You must include the exclamation mark and the hyphens as shown.

This document is adapted from A Beginner's Guide to HTML, *which was created by the National Center for Supercomputing. It is printed with permission from the Board of Trustees of the University of Illinois. Source URL: http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimer.html*