HTML is the *lingua franca* of the internet's World Wide Web. HTML, an acronym for HyperText oMarkup Language, provides text, images, sounds, and hypertext links within a formatted context to the web browser program on your desktop computer.

HTML is defined by specifications adopted by the World Wide Web Consortium (W3C), an organization founded in 1994 by Tim Berners-Lee, the inventor of the Web. Since then W3C has developed more than 35 technical specifications for the Web's infrastructure. HTML 4 is an SGML (Standard Generalized Markup Language) application. SGML is a system for defining markup languages. A markup language is a system of codes which are embedded in text for the purpose of controlling its presentation.

The current HTML version is 4.01 published December 24, 1999.

Unfortunately, HTML as a specification has not been implemented completely and uniformly by the several web browsers. Consequently HTML that looks one way on one web browser will likely look differently on another web browser. This disparity is behind the complexity of web development. Proficiency in web development requires both knowledge of the HTML specification and how each browser implements it.

Contents:

Elements of HTML

An HTML file is a text file composed of markup tags, text, and references to images and sounds. Markup tags are composed of descriptive codes surrounded by < and > characters, e.g., <P>.

Document character set: Accurate presentation of text characters by browsers depends on (1) the characters within the HTML document, (2) the character encoding used by the web server to transform the document character stream into a byte stream, and (3) the ability of the browser to transform the byte stream into a document character stream. Documents can declare their character

set. Servers may or may not adhere to it when preparing the byte stream. Browsers may or may not properly encode bytes to characters. Consequently, the use of certain characters by HTML authors may not be presented as intended.

The standard[1] document character set is ASCII (American Standard Code for Information Interchange, adopted in 1963) extended to the 8-bit graphic character set known as ISO 8859 or Latin-1 (designed in the mid-1980s by the European Computer Manufacturer's Association).

Sample HTML File

```
1.  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
2.  <HTML>
3.  <HEAD>
4.  <META HTTP-EQUIV="Content-Type" CONTENT="text/html;
    charset=ISO-8859-1">
5.  <TITLE>This text appears in the browser's title bar</title>
6.  </HEAD>
7.  <BODY>
8.  <P>This is a paragraph.</P>
9.  </BODY>
10. </HTML>
```

The line numbers in the above HTML were added to facilitate an explanation; they are **not** a part of the actual HTML.

Line 1 declares the SGML document type and HTML specification.
Line 2 declares the beginning of the actual HTML; line 10 declares its end.
Line 3 declares the beginning of the head section; line 6 declares its end.
Line 4 declares the document character set.
Line 5 is the document title.
Line 7 declares the beginning of the body section; line 9 declares its end.
Line 8 is a paragraph of text.

Structure of HTML

The basic structure of an HTML file is illustrated in the previous example. There are two sections: head and body. The *head* section contains the document-wide elements. The *body* section contains the "content" of the document: text, images, sounds, and hypertext links. This structure is always expressed in markup tags as follows:

```
<HTML>
    <HEAD>
    . . .
    </HEAD>
    <BODY>
    . . .
    </BODY>
```

---

1 The HTML 4.01 specification prescribes the Universal Character Set (UCS), known as ISO 10646, which is a UNICODE equivalent. The HTTP protocol specifies ISO-8859-1 as a default character set; implementation varies.

```
</HTML>
```

Comments can be placed anyplace within HTML as follows:

```
<!-- comment in one line -->
<!-- comment
   in two lines -->
```

HTML tags can include one or more *attributes*, e.g., <HR NOSHADE>. Multiple attributes are separated by white space. Attributes may require a value of a specific *data type*, e.g., <HR WIDTH=10>. Tags and attributes are case insensitive, hence <p> and <P> are functionally identical. This document presents both tags and attributes in capital letters for ease of reading.

Most HTML tags are used as a pair, e.g., <HTML> and </HTML>, where the first tag indicates the start of the tagged text and the second tag, which always has a "/" preceding the tag's code, indicates the end of the tagged text.

Paired tags may be nested but may not overlap (overlapping is tolerated by current browsers, but is illegal in SGML). Examples:
```
<P>This <B>nested tag</B> is allowed.</P>
<P>This <B>overlapping tag is not allowed.</P></B>
```

White space and indentation in HTML have no effect on the presentation of the contents. Its use is exclusively one of facilitating the reading of the HTML.

The <u>continuation</u> of text and HTML tag attributes is controlled by their data type. The CDATA data type applies to the CONTENT attribute (of META tags) and to document text (commonly associated with P tags). Continuation of this text is handled:
- line feeds are ignored
- carriage returns are replaced by a single space
- trailing blanks are ignored

This means that you can continue a META tag as follows:
```
<META NAME="description" CONTENT="Shows up in a search results
list. You might want to make this a complete paragraph.
And because of the ability to continue it to several lines,
it is easy to do so." LANG="en">
```

Element Identifiers

There are two methods for identifying HTML elements: ID and CLASS, both of which are attributes to most HTML tags. The ID attribute assigns a unique identifier to one element. The CLASS attribute assigns one or more class names to an element; the element may be said to belong to these classes.

The ID attribute has several roles in HTML:
- As a style sheet selector.
- As a target anchor for hypertext links. This use is preferred over the NAME attribute of the <A> tag.
- As a means to reference a particular element from a script.
- As the name of a declared OBJECT element.

- For general purpose processing by browsers (e.g., for identifying fields when extracting data from HTML pages into a database, translating HTML documents into other formats, etc.).

The CLASS attribute has one primary role in HTML:
- As a style sheet selector (when an author wishes to assign style information to a set of elements).
Examples:
```
<P ID="para1">This is a uniquely named paragraph.</P>
<P ID="para2">This is also a uniquely named paragraph.</P>
<P><SPAN ID="msg1" CLASS="info">This portion</SPAN> of the paragraph has both
id and class.</P>
<STYLE>  .info { color: green } </STYLE>
<P><A HREF="#para1">This links to a location</A> in the document with the named
id.</P>
```

- An HTML element can be assigned more than one class
```
P.one     { . . . }   applies to any <P> with CLASS = "one"
P.two     { . . . }   applies to any <P> with CLASS = "two"
P.one.two { . . . }   applies to any <P> with CLASS = "one" and "two"
 . . .
<P CLASS="one two three">blah blah blah.</P>
```
This allows a single CSS rule to apply to any HTML element assigned one or more of the corresponding CLASS values.

Styles

The word *style* refers to cascading style sheets (CSS). Styles specify appearance elements so that they are separate from HTML. Some HTML tags have appearance-controlling attributes that are deprecated—obsolete—meaning that the effect is best achieved with CSS.

Styles can be assigned to HTML tags in three different ways.
(1) The style is defined for a standard HTML tag in a <STYLE> tag. Example:
```
<STYLE> p {font-family: arial} </STYLE>
. . .
<P>This text is presented in arial.</P>
```
(2) The style is defined for an element identifier in a <STYLE> tag, and the actual text is tagged with the element identifier. Example:
```
<STYLE> p.special {color: red} </STYLE>
. . .
<P CLASS=special>This text is presented in red.</P>
```
(3) A style can also be embedded within an HTML tag with the STYLE attribute; the value of the attribute is the style declaration. Example:
```
<P STYLE="color: red; font-weight: 600">this paragraph gets a special style
applied.</P>
```

How styles are defined and how those definitions are provided to the HTML are discussed in their own primer.

HTML elements differ in the way styles are inherited. Usually inline elements inherit from block elements.

HTML permits any number of STYLE tags in the HEAD section.

The STYLE tag has attributes TYPE and MEDIA. The TYPE attribute specifies the style sheet language as a content type; typically this is "text/css." The MEDIA attribute specifies the intended destination medium as a single descriptor or a comma-separated list. Examples: "screen" and "screen, print." The default value is "screen."

Text

The basic presentation of text is as lines and paragraphs. Browsers effect automatic word wrapping based on the width of the browser window and the length of the text line. If the continuing box has a fixed width, then when the width of the browser window is such that the box is wider, the text on the right side disappears and a horizontal scroll bar appears to provide access to it.

The paragraph tag is <P>. It has a companion </P> tag that, because many browsers assume its existence, is often omitted by authors. The use of Cascading Style Sheets (CSS) to specify the formatting of paragraphs makes the omission of the </P> tag a bad idea. Paragraphs are typically presented by browsers as a left-justified block of text followed by a blank line.

A line break is caused by the <BR> tag. It has no end tag. A line break immediately following a start tag or immediately preceding an end tag is ignored. If you really want the empty line, use 2 line breaks. The <BR> tag has an attribute CLEAR which has been deprecated. CLEAR specifies where the next line should appear relative to floating elements.

Semantic Markup

The designers of the World Wide Web intended that the markup language, HTML, be used to represent the semantic quality of the content. While you could use the <P> tag for all text and use CSS to apply different formats to different texts, that practice would conceal the semantic quality of the content.

The best practice is to tag text elements to reflect their semantic meaning:

Type of Text	Semantic Tag
heading	<Hx>
paragraph	<P>
line item	<LI>
emphasis	<EM>
strong emphasis	<STRONG>

Special Characters and Punctuation

When text characters are not in the standard character set, cannot be entered directly by keyboards, or if they are reserved for HTML itself, they must be specified as character references. Character references may be expressed in either of two forms: numeric character references or character entity references. *Numeric character references* specify the code position of a character in the document character set. They can take the form &#n; where n is a decimal number in the ISO 10646 decimal character set. Character entity references use symbolic names and have the format &x; where x is the symbolic name.

Common characters are:

Special Character	Numeric Character Reference	Character Entity Reference
nonbreaking space		
— em dash	—	—[1]
– en dash	–	–[*]
' apostrophe	’	
soft hyphen	­	­
single prime[2]	'	
double prime	"	
opening quotation mark	“	
closing quotation mark	”	
opening single quote	‘	
closing single quote	’	
< greater than	<	<
> less than	>	>
% percent	%	
& ampersand	&	&
« left guillemet	«	(often separated from enclosed text with thin space)
» right guillemet	»	
thin space		
hair space	 	
¶ paragraph sign	¶	
☎ telephone icon	☎	
° degree sign	°	
{ left curly brace	{	
} right curly brace	}	
× multiply sign	×	
÷ divide sign	÷	
… ellipsis	…	
é lower case, acute accent	é	
É upper case, acute accent	É	

---

1 This reference is an example of one that is not rendered correctly by all browsers; use the numeric reference instead.
2 Strictly speaking, the prime characters should be slanted. The keyboard characters " and ' appear the same as the special prime characters.

Special Character	Numeric Character Reference	Character Entity Reference
è lower case, grave accent	è	
È upper case, grave accent	È	

Note: the nonbreaking space character is invaluable for inserting more than one blank space between letters. If you type more than one blank space within text in an HTML file, all spaces beyond the first one are ignored by the browser.

Some commonly found fonts like Symbol and Wingdings have characters that may be used like images or bullets. Example: ■, ▢, ▲, ▼, ◄, ►. These are typed with the corresponding keyboard key and the font family, color and size applied with tags. (Use <u>Control Panel, Fonts</u> to browse character fonts.)

The thinspace character has two problems:
- it is not thin enough
- it allows line breaks

These problems can be avoided by (1) NOT using the thinspace and (2) using CSS left and right padding (with the SPAN tag).

Word Wrapping

The HTML 4.1 specification says "text should only be wrapped at white space." IE6 and other browsers break lines at inter-word spaces and after some punctuation marks like the hyphen and slash (/). You can control line breaking somewhat:

- You can prevent line breaking at a particular inter-word space by replacing the space character with " ".
- You can specify where the browser can break a word with hyphenation by inserting the soft hyphen "­" inside the word. If the line is broken at the point of hyphenation, the hyphen is displayed, otherwise no hyphen is displayed.
- A line break can be forced with "
."
- A non-breaking hyphen may not be possible.
- CSS can suppress line breaks on whitespace and hyphens et al:
  ```
  <SPAN STYLE="white-space: nowrap">mm-dd-yy</SPAN>
  ```
- The <PRE> tag does not allow line breaking, instead a horizontal scroll bar appears when there are lines longer than the window width.
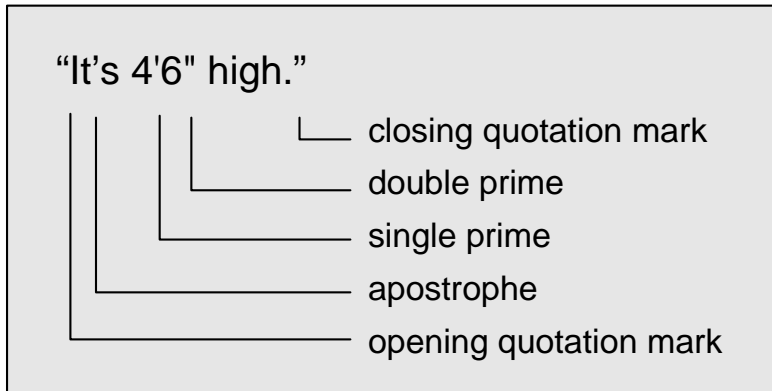
If you want content to line break at particular places while allowing word wrapping elsewhere, you can use the <BR> tag to force the line breaks. Now it matters where you put the <BR> tag. While it can be easier to read HTML when the tag is at the left of the line, this can cause a problem: When the rendered line just fits in the container element, you may get a line break before the <BR> tag. This is not desirable as it will insert a blank line between two lines of text that you want to be contiguous. The solution to this problem is to place the <BR> tag at the end of each line.

Punctuation

Proper punctuation may require the use of special characters. The following example illustrates the correct use of quotation marks, apostrophes, and primes:

"It's 4'6" high."   This incorrectly uses the keyboard characters ' and ".

"It's 4'6" high."
             └── closing quotation mark
         └── double prime
      └── single prime
   └── apostrophe
└── opening quotation mark

Other common punctuation errors involve the dash and hyphen, particularly the misuse of a hyphen for a dash. The probable origin of the errors is the typewriter keyboard: there is only one character, the hyphen (-). Word processing programs can make the distinction and so can HTML.

A hyphen is used to (1) divide a word at the end of a line, (2) to join words into compound words, and (3) for some prefixes. There are two lengths of dashes: An em[1] dash (—) is used to set off an appositional phrase or parenthetical expression whenever a comma might be misread or is insufficient to signify the import, as in "the sun—bright and yellow—just appeared." An en dash (–) is used to represent "to" between figures or words, e.g., Berlin–Baghdad Railway, 1940–1955. Space on either side of a dash should be equal but very small (not as wide as the space between words).

Typography

Typography addresses the style, arrangement, and appearance of typeset matter—text. The goals are legible, beautiful, and expressive type. "Good typography demands that the designer perceive very small differences in the way letters and spaces relate to each other."[2] At its simplest, typeface, "type size, line width, and interlinear space all affect readability." These elements are best handled with CSS.

Indention is the blank space left in setting a line in from the margin. This is commonly done to mark the beginning of a paragraph. It is also used in outlines and lists. The indention of paragraphs should be in proportion to the width of the page. In printing, an em space is considered enough for a page less than 4.5 inches wide while an em and a half or 2 ems look better on wider pages. The problem for web pages whose width can be changed by the reader is to choose a space that works with the most common page width. Indention is best handled with CSS.

Browsers select fonts on the reader's computer based on CSS specifications of font family and font properties. The selection process is detailed in the W3C's CSS2 Specification.

---

1 The *em* is a unit of measure equal to the square of the body size of the type. Typically a 10-point typeface has an em that is 10 points.

2 This quote is excerpted from "Better Type" by Betty Binns. I also refer to "Words Into Type."

Text Tags

There are six heading tags which can be used to present text differently than paragraphs: <H1>, <H2>, <H3> ,<H4>, <H5>, <H6>. These are paired. Browsers commonly present H1 text in a larger, bold font. Lesser headings are presented in increasingly smaller font sizes.

<BLOCKQUOTE>	for long quotations; generally rendered as an indented block; its use solely for indention is deprecated in favor of CSS
<ADDRESS>	

The following paired tags establish inline content:

<EM>	emphasis
<STRONG>	stronger emphasis
<I>	italic
<B>	bold
<U>	underline
<SUB>	subscript
<SUP>	superscript
<Q>	short quotation; the text is rendered with delimiting quotation marks (the text itself should not have quotation marks); quotes can be nested [NICE IDEA BUT IE ignores it]

The following phrase elements (inline) have particular significance only in technical documents.

<CODE>	fragment of computer code
<SAMP>	sample output
<KBD>	text to be entered by the user with keyboard
<VAR>	instance of variable or program argument
<ABBR>	abbreviation; the TITLE attribute can be used to provide the full or expanded form of the phrase
<ACRONYM>	acronym; the TITLE attribute can be used to provide the full or expanded form of the phrase
<PRE>	preformatted text; some things are optional to the browser like leaving white space intact, rendering text with a fixed-pitched font, and disabling word wrap

When text is presented in HTML, especially text extracted from a database, problems may arise if text includes characters that are special to HTML, such as "&", "<", and ">". This can be resolved by four tags:
- The inline paired PRE tag has several drawbacks in IE6:
  - It is rendered in a fixed-pitch font.
  - The font cannot be changed with CSS.
  - Automatic word wrap is disabled.
- The inline paired CODE and SAMP tags solve the word wrap problem but not the font problem.
- The block paired BLOCKQUOTE tag solves both the word wrap and font problems. The default indentation can be removed by CSS margin: 0px. It may handle paragraph breaks.

Format and Layout

With HTML individual pieces of content are placed relative to previous pieces. This is similar to a simplistic word processor. Cascading style sheets (CSS) were developed as a means of effecting more sophisticated format and layout techniques. As CSS was developed, its abilities caused some HTML tags and attributes to be deprecated (obsoleted); the W3C—and many web designers—recommend that CSS be used whenever possible. CSS is discussed in a separate primer.

Typography and layout—paragraph, line, word, and letter—is best handled with CSS.

Document Body

The document body contains the contents of the document. The paired <BODY> tag defines the document body. It has attributes, the first six of which are deprecated: BACKGROUND, TEXT, LINK, VLINK, ALINK, BGCOLOR, ID, CLASS, STYLE. Attribute BACKGROUND specifies an image file to be used as a background. Attribute TEXT specifies the color of the text. Attributes LINK, VLINK, and ALINK specify the color of hyperlinked text that is unvisited, visited, or selected respectively.

The most useful attributes are those that initiate a script: ONLOAD and ONUNLOAD. Attribute ONLOAD runs the named script when the page is loaded (and refreshed). Attribute ONUNLOAD runs the named script when the page is unloaded (removed). Example:

```
<BODY ONLOAD="maxwin()">
```

Blocks and Inline Content

A unit of content is classified as either block or inline. This distinction usually affects the inheritance of styles.

An example of a block is one or more paragraphs. An example of inline is one or more words in a sentence; tags that apply include <I>, <SUB>, and <A>. Content can be grouped to facilitate group-wide formatting and layout. The paired <DIV> tag defines a block group. The paired <SPAN> tag defines an inline group. The only attribute these tags have is ALIGN. Their main use is with CSS, as they can have a style class or id.

Example:
```
<P>This sentence illustrates the use of span to change the <SPAN
CLASS="wacky">appearance</SPAN> of some of the words.</P>
```

Some tags are empty, i.e., they have no content; examples are <BR> and <IMG>. Some elements can be either block or online; examples are <LI>, <TH>, <TD>.
 has attribute CLEAR which specifies how following lines are to reflect an existing float. Values are: none, left, right, all. This was deprecated in favor of CSS.

Vertical Spacing

There are two methods for creating vertical space. (1) Use as many <BR> tags as necessary. (2) Use CSS margin-top. Do not use empty <P> tags.

Horizontal Alignment

Use the ALIGN attribute with <P>, <TABLE>, <TR>, <TD>, <DIV>, and <SPAN> tags. Alignment can be specified as left, center, right, or justify. This has been deprecated by CSS.

Font

The font in which text is presented can be specified as the typeface, size, bold, italic, and underline. The first two are effected with the paired tags <BASEFONT> and <FONT> which have attributes FAMILY and SIZE; both tags have been deprecated by CSS. Bold is effected by the paired <B> tag. Italic is effected by the paired <I> tag. Underline is effected by the paired <U> tag; this tag has been deprecated by CSS. All font modifiers can be effected by CSS (and should).

Color

Color can be specified for page and table backgrounds and for text. Colors are specified by 16 names or over 1 million six-digit numbers. The numbers are specified as a set of three hexadecimal numbers for Red, Green, and Blue (RGB), where the values of each range from 00 to FF. An example: #0066CC means Red = 00, Green = 66, and Blue = CC.

Background color is effected by the BGCOLOR attribute of the tags <BODY>, <TABLE>, <TR>, <TH>, and <TD>. Text color is effected by the COLOR attribute of the <P> tag.

The color of text flagged for a hypertext link is commonly controlled by the browser's user preferences. Those colors can be overridden by BODY tag attributes LINK (for unvisited links), VLINK (for visited links), and ALINK (for selected links). Example: `<BODY LINK="red" VLINK="#3060ff" ALINK="green">`

All color references in HTML have been deprecated. Use CSS instead.

Sample colors:

white	R255	G255	B255	#FFFFFF
black	R0	G0	B0	#000000
red	R255	G0	B0	#FF0000
yellow	R255	G255	B0	#FFFF00
blue	R0	G0	B255	#0000FF
green	R0	G255	B0	#00FF00
cerise	R255	G0	B255	#FF00FF
bright blue green	R0	G255	B255	#00FFFF
pink	R255	G187	B187	#FFBBBB

Tables

As with word processors, the primary use of tables is to present a grid of values, such as distances between two cities. On web pages tables have also been used to make parallel columns of text and images, to indent groups of paragraphs, and to limit the width of paragraphs. CSS provides an alternate solution for all these uses, especially for all but the first.

Tables are composed of sets of paired tags.
- The table itself is defined with paired <TABLE> tags.

- Each row is defined with paired <TR> tags; the end tag, </TR> may be omitted.
- Each cell within a row is defined with paired <TD> tags; the end tag, </TD> may be omitted.

There are additional tags which can be used to make visual and/or semantic distinctions:
- The number and width of columns may be defined with one or more paired <COLGROUP> tags; each exception is defined with the <COL> tag. Use of these is optional.
- Horizontal groups of rows can be defined with tags <THEAD>, <TBODY>, and <TFOOT>; their end tags are optional. This is necessary only when you want to have grid lines limited to groups of rows; see discussion of attribute RULES below.
- Cells may be specified as column headings so that they are represented differently by the browser; the tag <TH> is used.

Simple example:
```
<TABLE>
<TR>
<TD>This is the first paragraph in the first cell.
<TD>This is the first paragraph in the second cell.
<TR>
<TD>More.
<TD>More.
</TABLE>
```

Simple example with column width:
```
<TABLE>
<TR>
<TD WIDTH="*">This is the first paragraph in the first cell.
<TD WIDTH="200">This is the first paragraph in the second cell.
<TR>
<TD>More.
<TD>More.
</TABLE>
```

Example with one COLGROUP:
```
<TABLE BORDER=1 FRAME=box RULES=all CELLSPACING=2 CELLPADDING=3>
<COLGROUP SPAN=3 WIDTH=150>
</COLGROUP>
<TR>
<TD>This is the first paragraph in the first cell.
</TABLE>
```

Example with two COLGROUPS:
```
<TABLE BORDER=1 RULES=groups CELLSPACING=2 CELLPADDING=3>
<COLGROUP SPAN=2>
<COL WIDTH=100>
<COL WIDTH=150>
</COLGROUP>
<COLGROUP SPAN=1 WIDTH=60%>
</COLGROUP>
<TR>
<TD>This is the first paragraph in the first cell. Its width is 100 pixels.
<TD>second column. Its width is 150 pixels.
<TD>third column. It's width is 60% of the viewport minus 250 pixels.
</TABLE>
```

The two previous examples illustrate different ways of using COLGROUP. In the second example, there are three columns, the last one will fill the window (you have to use trial and error to arrive at a percentage width that works properly, i.e., not reduce the width of the fixed width columns), the rules are drawn between columns 2 and 3.

The effect of most attributes can be achieved with CSS 2; some attributes are deprecated.

Attribute BORDER specifies the width in pixels of the table border. No border is effected by BORDER=0.

Attribute FRAME specifies which sides of the frame surrounding the table are to be visible; value choices are void (no sides), above (top side only), below (bottom side only), hsides (top and bottom sides only), vsides (right and left sides only), lhs (left-hand side only), rhs (right-hand side only), box (all four sides), border (all four sides).

Attribute RULES specifies which rules within the table (between the cells) are to be visible; value choices are: none, groups (between row groups and column groups only), rows (between rows only), cols (between columns only), all (between all rows and all columns).

Attribute CELLSPACING specifies the width in pixels around the outside of the table and between the cells. The table's background color is visible in the cellspacing. This is the way to color a table's border and rules.

Attribute CELLPADDING specifies the width in pixels of the internal cell margins—the space between the cell edge and the contents.

Attribute SPAN specifies the number of columns that comprise the column group.

When width is not specified, the default rendering is to fill the containing block with the table and adjust each column width to best fit its contents. This can be a very satisfactory effect. When the table width is set in CSS as `width: 100%` and the content is less than the space available, each column width allows the text to have the same side margins (padding in CSS jargon). When the content exceeds the space available, the column widths vary so the grid pattern is balanced.

Attribute WIDTH specifies width of the object it modifies. It applies to tags <TABLE>, <COLGROUP>, <COL>, <TH>, and <TD>. Width can be specified as a fixed number of pixels, a percentage of the width available to the table, or—for columns only—as a proportion of the available width divided by the number of columns that specify proportional width. Examples: WIDTH="500"; WIDTH="50%"; WIDTH="*" (means use all available space within the container exclusive of any preceding column width[1]); WIDTH="0*" (means the minimum necessary to hold the column's contents); WIDTH="2*" (means the column is twice as wide as the unit width available for proportional space). See the HTML 4.01 specification for a thorough discussion of proportional widths.

---

1 This form of the WIDTH attribute is very helpful when you want a table to fill up the window regardless of its actual width. You can set one or more columns to a fixed width and let one column expand to the remaining space.

Some browsers do not handle the WIDTH attribute reliably. Authors compensate by using a transparent single-pixel gif image to "fill" the width of a cell, especially an empty cell. See discussion of images.

Attribute HEIGHT specifies cell height as a fixed number of pixels or a percentage. This attribute is deprecated.

Attribute ALIGN specifies horizontal alignment of the contents of rows, cells, columns, and column groups. Value choices are: center, left, right, justify, char (alignment is on the character specified by the CHAR attribute).

Attribute CHAR identifies the character on which horizontal alignment is to be done. Example:
```
<COL ALIGN="char" CHAR=".">
```

Attribute VALIGN specifies vertical alignment of the contents of rows, cells, columns, and column groups. Value choices are: top, middle, bottom, baseline.

When using a table for layout purposes, you must work out how many columns and rows you need. After that, it is possible to merge cells across rows or columns. For instance, in the following table, merged cells are indicated by a grey background.

Merging cells is effected by two attributes: ROWSPAN and COLSPAN. These apply to the tags <TH> and <TD>. These attributes specify the number of cells which are to be merged. Example of above table:
```
<TABLE BORDER=1 FRAME=box RULES=all>
<COLGROUP SPAN=3 WIDTH="33%"></COLGROUP>
<TR>
<TD COLSPAN=2 BGCOLOR=gray>  
<TD>  
<TR>
<TD>  
<TD>  
<TD>  
<TR>
<TD>  
<TD ROWSPAN=2 BGCOLOR=gray>  
<TD>  
<TR>
<TD>  
<TD>  
<TR>
<TD>  
<TD>  
<TD>  
</TABLE>
```

Text can be entered in table cells with all text tags: paragraph, line break, list, bold, italic, etc. Using the <P> tag is especially helpful for using styles within a table, which is otherwise a problematic situation.

Lists

Lists can be nested.

Ordered and unordered lists

Lists, like with word processors, have either bullets or numbers and have a hanging indent that keeps the bullet/number in a columnar area to the left of the text. A list is typically presented with a blank line before and after; sometimes each list item is separated by a blank line. A list is established with the paired <UL> or <OL> tags, the former is an unordered list with bullets, the latter is an ordered list with numbers. Within the pair of tags are individual list item tags, <LI> whose end tag </LI> is optional.

The <UL> tag has attributes, all of which are deprecated: TYPE indicates the type of bullet as either disc, square, or circle; the size of the bullet depends on browser's setting for text size. COMPACT indicates that less space be presented between line items; this is interpreted differently by different browsers.

The <OL> tag has attributes, all of which are deprecated: TYPE indicates the numbering style as 1 (arabic number), a (lower case letter), A (upper case letter), i (lower case roman), or I (upper case roman). START is the number of the first list item. COMPACT is as for the <UL> tag.

The <LI> tag in an ordered list has an attribute, which is deprecated: VALUE is its number, intended to override the "normal" numbering sequence. Subsequent list items are numbered sequentially starting with the number after this.

Example:
```
<UL TYPE=disc>
<LI>First bulleted list item.
<LI>Second bulleted list item.
</UL>
```

An unordered list can be created with bullets that are characters in a special font. This is nice because you can control both the size and the color of the bullets. This technique is effected with paragraph and line break tags; one drawback is that there is no hanging indent. If you allow a blank line between the list items, then you can use a paragraph tag for each list item and use CSS to effect a hanging indent.
```
<P>
<SPAN STYLE="font-family: Wingdings; color: #E17557">n</SPAN> Data Analysis<BR>
<SPAN STYLE="font-family: Wingdings; color: #E17557">n</SPAN> Data Maintenance
Processes<BR>
<SPAN STYLE="font-family: Wingdings; color: #E17557">n</SPAN> Editor's
Guide</P>

<P STYLE="margin-left: 17px; text-indent: -15px"><SPAN STYLE="font-family:
Wingdings; color: red">n</SPAN> test of hanging indent in paragraph, the actual
spacing will require trial-and-error testing.</P>
```

CSS can be used to define number/bullet. This approach has more options.

Definition list

A definition list has a different appearance. It consists of a series of term-definition pairs where the term is presented in bold text and the definition is presented below it and indented. A definition list is established with the paired <DL> tag. Within the pair of tags are individual list items as a pair of <DT> and <DD> tags, for definition term and definition; these have no end tag.

Example:
```
<DL>
<DT>First term
<DD>First definition.
<DT>Second term
<DD>Second definition.
</DL>
```

This example is presented as:
First term
    First definition.
Second term
    Second definition.

It is possible to have more than one DD tag for one DT tag. This enables a two-level list.
```
<DT>Term x
<DD>First discussion.
<DD>Second discussion.
```

It is possible to insert paragraphs in definitions:
```
<DT>Term x
<DD>Beginning of discussion.
<P>Second paragraph of discussion.
```

But, do not put ULs in DD.

Hypertext Links

Hypertext links are the greatest feature of the web. They provide for a "seamless" navigation from one page to another: when a link is clicked, the linked-to page opens in a browser window. A link is created with the paired <A> tag, referred to as an anchor. The text and/or image within the tag pair are typically presented in a different color and are underlined as a visual clue that the link exists. A hypertext link can be to a different location within the same document, to a different document within the same web site, or to a document in a different web site.

The <A> tag has several attributes:
- NAME provides a name to the current location so that it can be used as a destination. Usually text tagged as a destination is not rendered differently than the surrounding text.
- HREF identifies the destination document.
- TARGET identifies the frame or browser window in which the destination document is to be opened. The default is the same window as the current document. There are good reasons for opening a linked document in a different window.
- TITLE provides text commonly presented as a tool tip, that is in small sized letters within a rectangle immediately below the link and only when the cursor is moved over the link.

However, the tool tip is so slow to appear it may not be seen; a better approach is to use JavaScript.
- ACCESSKEY provides a shortcut key that, when pressed, activates the link.
- HIDEFOCUS provides a way to hide the focus outline, may be specific to IE.

The NAME attribute can be combined with the HREF attribute when the same text can be both an anchor and a destination:
```
<P><A NAME="here" HREF="help.html">Help is available</A> for this
transaction.</P>
```

Destination documents are specified differently depending on where they are relative to the current document:
```
<A HREF="#here">this links to a location within the same document</A>
<A HREF="docB.html">this links to a document in the same web site and in the
same directory</A>
<A HREF="docA.html#section3">this links to a location within a document in the
same web site and in the same directory</A>
<A HREF="http://www.pge.com/index.html">this links to a document in a different
web site</A>
```

The TARGET attribute can specify any frame or window name, e.g., "content." In this way, you can specify in which frame of a frameset the linked file should open. In addition, there are several special values:
use target="_ " to force link from frame to open in new window without frame
use target="_top" to force link from frame to open in same window without frame
use target="_blank" to open doc in new empty un-named window
use target="_parent" to open doc in the window in which the parent document was opened
use target="_search" to open doc in browser's search pane
use target="_self" to open doc in the same frame/window (the active window)

The HIDEFOCUS attribute can hide the focus outline:
```
<A HREF=". . . " HIDEFOCUS="true">this links</A>
```

The <BASE> tag can be used to specify the target once and dispense with the TARGET attribute in each <A> tag. You can rely on the <BASE> TARGET attribute for some links and use the <A> TARGET attribute for other links (because the <A> TARGET attribute has precedence over the <BASE> TARGET attribute). The <BASE> tag has attributes:
- HREF provides a base URI for resolving relative URLs.
- TARGET provides the frame/window name,
Locate <BASE> in the HEAD section.

A standard feature of many long web pages is a "go to top" link. The easiest way to do this is with JavaScript:
```
<a href="javascript: self.scrollTo(0,0)">go to top</a>
```

Data can be passed to another web page as a suffix of the URL and separated from it by a question mark. Example:
```
<A HREF="sub.html?test">
```
The receiving web page has to use a script (program) to parse the URL string and extract the data. Suffixing a URL with "?" by itself has no effect on the receiving page. Data is commonly passed from

---

a data entry form page to a form processing page, but could also be used to control conditional presentation of content.

URLs have several schemes:

ftp   file transfer protocol; used to download file, e.g., ftp://www.some.com/filename.exe
http   hypertext transfer protocol
mailto   electronic mail address
telnet   reference to interactive session
file   host-specific file name

URL for file on file server, where GO251\CRTSYS is server name:
```
<A HREF="file:\\GO251\CRTSYS/SJDA/Web/pagename.html">
<A HREF="file://c:\a\b\filename.doc">
```

A hypertext link can also initialize an email message:
```
<A HREF="mailto:yourname@somewhere.com?subject=More information from Webmaster
Tools&bcc=wishiwas@play.org&cc=someone@home.com,you@work.net&body=Please
provide some information on...">Email the Webmaster</a>
```

Link to relative URL:
examples when current URL is http://a/b/c/file.html

relative URL   yields
```
../x.html                http://a/b/x.html
../../y.html             http://a/y.html
../g/z.html              http://a/b/g/z.html
v.html                   http://a/b/c/v.html
/w.html                  http://a/w.html
//w.html                 http://w.html
./w.html                 http://a/b/c/w.html
```

Link to non-HTML file
Links to non-HTML files typically cause the files to be opened within the browser by a "helper" application. This is especially true for Internet Explorer when the files are Microsoft Office files and Adobe Acrobat files. A problem exists for Excel files: not all Excel tools, especially the data tools, are available through IE. So while you can open an Excel file in the IE browser, you may not have all the Excel tools available to use on it. The solution is to (1) open the directory from which (2) the user opens the desired file—which opens in its native application (and with all tools available).

To open a directory in IE a la File Manager:
```
<A HREF="file://c:\data\genealogy" TARGET="_blank">link text</A>
```

If you link to an EXE file, the viewer will be treated to the "File Download" box saying "some files can harm your computer" and prompt you to open or save:
```
<A HREF="file://c:\data\my.exe" TARGET="_blank">link text</A>
```

Accesskey example:
```
<A HREF="a.html" ACCESSKEY="h">Link here</A>
```
In Windows, [Alt+h] is the shortcut key.
In Mac, [Ctrl+h] is the shortcut key.

JavaScript message in status bar:
When mouse is moved over linked text, a message can be displayed on the status bar:
```
<A HREF="…" ONMOUSEOVER="window.status='my message'; return true">
```

This message can be cleared when the mouse moves on:

```
<A HREF="…" ONMOUSEOUT="window.status=''; return true">
```

The ONMOUSEOVER and ONMOUSEOUT can be combined in the same <A> tag.

Pros and cons of opening a file in a new window:

Cons: In the new window there is no Back button; the user may not realize they are seeing a new window; the Windows taskbar will have one more item.

Pros: Good for Word documents, PDF files, large images, and printable version of the current document.

Good practices: Warn the user that a link will open in a new document with one or more of (a) a tool tip "Opens in a new window" (accomplished with the Title attribute); (b) text in parentheses, e.g., (opens in new window); and (c) icon to the right of the link, e.g., ⬓.

Using CSS dynamic link pseudo-classes

You can use the A tag with no attributes in order to use the CSS dynamic link pseudo-classes. In this way you can restyle content when the pointing device hovers over it.

```
<A CLASS="special">content</A>
```

Link with ampersand in URL

Ampersands contained in URLs in the HREF attribute are assumed to begin an entity reference, are illegal, and can cause problems. The correct handling of an ampersand in an URL is to replace it with & as in the following example:

```
WRONG:      <A HREF="a.b.c/here&m=4&n=20">
RIGHT:      <A HREF="a.b.c/here&amp;m=4&amp;n=20">
```

Focus outline

When the user clicks on a link, a dotted line called a "focus outline" (some call it a "selection rectangle") appears around the link. The dotted outline remains as long as the focus is on the element, even when the user mouses off that link before letting go. When the focus is moved from element to element through tabbing, each focused element in turn gets the dotted outline. The outline persists after the user clicks to go to another page and then returns using the browser's BACK button. It also persists if link opens a document in a new window.

You may find this dotted outline anywhere from annoying to ugly. In particular, an image with a persistent dotted outline **is** ugly. For text links, having the outline appear when the link first gets the focus is a useful clue, especially for users who navigate to links with the [Tab] key, but it can be made extraneous by the use of pseudo-classes in CSS.

The dotted focus outline can be suppressed with JavaScript or CSS.

- You can suppress it completely with the *blur* method in the *focus* event handler:
```
<A HREF=". . ." ONFOCUS="blur()"> . . . </A>
```
The problem here is that you are disabling (removing) the focus event, not just hiding the outline. Consequently it prevents tabbing through the links, i.e., you will not be able to access the first blurred link with the [Tab] key and all succeeding links. This practice violates the W3C accessibility standards.

- You can suppress it after the initial click with the *blur* method in the *click* event handler; this has the same problem as with the *focus* event handler:
```
<A HREF=". . ." ONCLICK="blur()"> . . . </A>
```

- You can suppress it without blocking tab access in IE 6 and 7. Note that *hidefocus* is a Microsoft IE extension, not a W3C standard.

```
<A HREF=". . ." HIDEFOCUS="true"> . . . </A>
```

- You can suppress it completely with CSS:

```
A    { outline: none; }              /* does not work in IE 6 or 7 */
```

- You can selectively suppress it with CSS:

```
A.noline   { outline: none; }        /* does not work in IE 6 or 7 */
```

So, since I design my sites for IE and Firefox, I use a combination of *hidefocus="true"* and *outline:none*.

Horizontal Lines

A horizontal line is effected by the <HR> tag, which has no end tag. It has several attributes, all of which are deprecated: ALIGN has values left, center, or right. NOSHADE indicates the line is to be solid; the default is grooved. SIZE indicates the height of the line in pixels. WIDTH indicates the length of the line in pixels or percent of the available space.

Example:
```
<HR ALIGN="center" NOSHADE SIZE="5" WIDTH="50%">
```

Using CSS you have more formatting choices including color and border.

Images

Images can be placed in a document using either of two tags: IMG and OBJECT. The latter is preferred by W3C because it can also handle things like video clips and programs. The IMG tag has no end tag, while the OBJECT tag has a required end tag.

Images are placed relative to the text that precedes the IMG/OBJECT tag. When a image is placed within a block, the block text flows around the image. The visual alignment of the image with the surrounding text involves both the positioning of the image and the size of its external margins.

The IMG tag establishes an in-line element that is, by default, aligned vertically at the baseline of the containing block.

The IMG tag has attributes:
- SRC is the address (URI) of the image file
- ALT is the text of a short description which is presented like a tool tip when the cursor moves over the image. May not work in IE 6. REQUIRED.
- TITLE may be preferred to ALT.
- HEIGHT, in pixels, overrides the actual height of the image.
- WIDTH, in pixels, overrides the actual width of the image.
- ALIGN controls vertical or horizontal alignment vis-à-vis current baseline. There are five values:
  "bottom" aligns the bottom of the image with the baseline; it is the default.
  "middle" aligns the middle of the image with the baseline.
  "top" aligns the top of the image with the baseline.

"left" floats the image against the left margin.
"right" floats the image against the right margin.
This attribute is deprecated.
- BORDER, in pixels, is the width of the border around the image. This attribute is deprecated.
- HSPACE, in pixels, is the width of the left and right margins. This attribute is deprecated.
- VSPACE, in pixels, is the height of the top and bottom margins. This attribute is deprecated.

When the HEIGHT and/or WIDTH attribute values differ from the image's physical size, the image is scaled. Accordingly, the HEIGHT and WIDTH attributes can be used to extend a small image, especially a one-pixel GIF.
```
<IMG SRC="one-pixel.gif" HEIGHT=100 WIDTH=5>
```

Images can also be used as the background for the document:
```
<BODY BACKGROUND="logo.gif">
```
this has been deprecated
In CSS, a style can apply a background image—to more than just the document.

The whitespace surrounding an image is, by default, a small, non-zero distance.

A stylesheet should be used to specify external margins, alignment, and borders.

Video and Audio

Unfortunately there is no standard method for presenting video images with the various browsers. It varies by how you want the video to be presented and the type of video. "Embedded media" (animation, audio, video) are movies that play within a portion of a web page rather than in a separate application. You cannot get all videos to embed in all browsers. This presentation is by no means complete. You are advised to look on the internet for details about your particular needs.

It is easier to play a movie with a link to the movie file than to embed it:
```
<A HREF="MovieFile.mpg">Watch this movie</A>
```

Sounds are played by:
- helper application, e.g., <A HREF="sound.file">Play this</A>
- plug-in application

Embedding video is effected with an embedded player. Virtually all modern streaming video players for the Windows environment are built as ActiveX controls. Player vendors include Real (RealPlayer), Apple (QuickTime), and Microsoft (Windows Media Player).

Embedded video images can be placed by the IMG and OBJECT tags.

The IMG tag works with video but not audio media. The IMG tag is described in the previous section on static images. It works in IE6 for Windows. Additional attributes that apply to videos are:
- DYNSRC is the URL of the video source
- LOOP is the count of times to times to repeat and can be 0 (default), a positive number, or -1 (indefinitely)

The paired OBJECT tag was developed to handle generic objects, including videos, images, and applets. It has many attributes, among them:

- CLASSID is typically the GUID of the player control
- ID identifies the player control so that it can be referred to in code
- DATA is the (URL) address of the video file
- TYPE describes the file; e.g., "video/mpeg", "image/jpeg", "image/gif"
- CODEBASE is the base path used to resolve relative URIs in the CLASSID and DATA attributes
- HEIGHT
- WIDTH

It may include PARAM elements which define specific startup conditions; their syntax is <PARAM NAME="*ParameterName*" VALUE="*value*">.

An example:

```
<P>
Blondie does Hollywood
<OBJECT DATA="BlondieTheMovie.mpg" TYPE="video/mpeg">
</OBJECT>
</P>
```

Flash is played with an OBJECT tag:

```
<OBJECT CLASSID="clsid:D27CDB6E-AE6D-11CF-96B8-44455340000"
CODEBASE="http://download.macromedia.com/pub/schockwave/cabs/flash/swflash.cab#
version=6,0,40,0"
WIDTH="550" HEIGHT="400" ID="myMovieName">
</OBJECT>

<OBJECT CLASSID="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
    CODEBASE="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.c
ab#version=6,0,29,0" WIDTH="600" HEIGHT="400">
<PARAM NAME="movie" VALUE="ExxonToastsPlanet.swf">
<param name="quality" value="high">
<EMBED SRC="ExxonToastsPlanet.swf" QUALITY="high"
    PLUGINSPAGE="http://www.macromedia.com/go/getflashplayer"
    TYPE="application/x-shockwave-flash" WIDTH="600" HEIGHT="400">
</EMBED>
</OBJECT>
```

Microsoft has an ActiveX **ActiveMovie** control for Windows 95 and NT. It plays MPEG video and audio files, AVI video files, QuickTime video files, and WAV audio files. It is specified in the OBJECT tag with attributes:

```
<OBJECT ID="ActiveMovie1" WIDTH=267 HEIGHT=73
CLASSID="CLSID:05589FA1-C356-11CE-BF01-00AA0055595A">
<PARAM NAME="_ExtentX" VALUE="7038">
<PARAM NAME="_ExtentY" VALUE="1931">
<PARAM NAME="MovieWindowWidth" VALUE="352">
<PARAM NAME="MovieWindowHeight" VALUE="247">
<PARAM NAME="FileName" VALUE="cybervis.mpg">
</OBJECT>
```

To embed a streaming MPEG-4 video file on a web page with **QuickTime**, use the following code:

```
<OBJECT CLASSID="clsid:02BF25D5-8C17-4B23-BC80-D3488ABDDC6B"
CODEBASE="http://www.apple.com/qtactivex/qtplugin.cab"
WIDTH="320" HEIGHT="256" >
<PARAM NAME="src" VALUE="videofilename.mp4" >
<PARAM NAME="autoplay" VALUE="true" >
<EMBED SRC="QTMimeType.pntg" TYPE="image/x-macpaint"
```

```
PLUGINSPAGE="http://www.apple.com/quicktime/download" QTSRC="videofilename.mp4"
WIDTH="320" HEIGHT="256" AUTOPLAY="true">
</EMBED>
</OBJECT>
```

Windows Media Player can be used to play streaming media and creating a custom interface. See http://download.microsoft.com/download/6/f/7/6f75c69c-70c3-422c-9009-dc0364a7bff7/AdvertisingSolutions2.doc for details. The Player is used as an ActiveX control. The control version must depend on the browser version; the SDK includes code samples to identify the user's browser.

Embed streaming **Windows Media** files: make the clip play automatically as soon as the web page loads.
```
<OBJECT
CLASSID="clsid:22D6F312-B0F6-11D0-94AB-0080C74C7E95">
<PARAM NAME="Filename" VALUE="http://www.webhost.com/yourname/rage.wma">
</OBJECT>
```

Alternatively,
```
<OBJECT
CLASSID="clsid:22D6F312-B0F6-11D0-94AB-0080C74C7E95"
HEIGHT="352" WIDTH="240"
ID="Player" >
```
The OBJECT part says to tell the web browser to insert the video wherever you paste the text. The green ID="Iplayer" part merely tells it to use a video control panel. The red HEIGHT and WIDTH let you to define how large the video will appear. This means even if you made a clip that was 50 pixels by 50 pixels, if you said the video screen should be 352 x 240 then that is what will be displayed. This is a useful feature because you can often make a video look more impressive by making a small resolution clip and then resizing it a third larger on the screen.

This player has useful parameters, which may vary by version (so check the specs). In the instances where the parameter has only two values, 0 is no/off, 1 is yes/on. The syntax is <PARAM NAME="*ParameterName*" VALUE="*n*">

Parameter	Use
AutoStart	When on starts playing the video automatically. If you have lots of clips on the same page then you will need this option, otherwise everything will play at the same time. When set off the video will only play when the visitor clicks on it.
AutoRewind	This tells the video to start from the beginning once it's done. It's not a loop to replay the video, it only resets it.
ShowStatusBar	This makes the little black bar we see at the bottom of Media Player, showing the play length and so on.
ShowTracker	This controls the Media Player slider bar. With most audio files you can move to parts of the song but this doesn't work too well with video files.
AutoSize	When on the video will automatically resize to the original size of the WMV file and not what you define.
AnimationAtStart	When on will show the Media Player logo before the video plays.

Parameter	Use
TransparentAtStart	Normally Media Player will give a black background before the video starts. When set on it will start with the same color as the web page.
ClickToPlay	When on requires the user to click on the video screen to start it playing. This is useful if you don't want any play buttons places at the bottom of the video.
ShowAudioControls	When on shows the volume control.
ShowDisplay	When on shows the author details.
uiMode	Controls the user interface in which the video is embedded: "invisible" means no visible user interface; "none" means no controls; "mini" means only status window, play/pause, stop, mute, and volume controls shown in addition to the video or visualization window; "full" (default) means status window, seek bar, play/pause, stop, mute, next, previous, fast forward, fast reverse, and volume controls in addition to the video or visualization window
Volume	The value is an integer ranging from 0 to 100. Zero specifies no volume and 100 specifies full volume. If no value is specified for this property, it defaults to the last volume setting established for the player.

Netscape has extension HTML element which can be used for video: EMBED tag. It works on Windows and Macs. It is good practice to handle both:

```
<OBJECT> . . .
<PARM> . . .
. . .
<EMBED . . .></EMBED>
</OBJECT>
```

You can handle Microsoft Media Player in Netscape browser with the EMBED tag.

```
<OBJECT> . . .
<PARM> . . .
. . .
<EMBED TYPE="application/x-mplayer2"
PLUGINSPAGE="http://www.microsoft.com/Windows/Downloads/Contents/Products/Media
Player/"
SRC="http://myserver/mypath/myfile.asf"
NAME="MediaPlayer"
WIDTH=320
HEIGHT=240>
</EMBED>
</OBJECT>
```

 Background sound is non-standard, effected in IE with Microsoft's extension BGSOUND tag. It has attributes LOOP, SRC, and VOLUME. See the MSDN website for details.

Frames

Frames allow you to present more than one document (web page) in the same browser window. You can arrange the documents to be side-by-side, top-and-bottom, or some combination. Frames seem to be a nifty solution to the problem of content management when some content is the same on several documents: the common content is relegated to its own document and presented in one frame with the other content documents are in a separate frame. However, there are a number of problems with frames and they have fallen into disfavor with some web developers, including me. Nevertheless, they are useful in some situations—just use them judiciously.

The central problem is that frames apply to the window—not the document. So the window is carved up into pieces and other HTML documents are fit into those pieces. You cannot scroll the entire page, only the piece.

The following example simulates a window divided into three frames. The left side contains document A, the center contains document B, the right contains document C.

document A	document B	document C

This framing is accomplished with the following HTML:
```
<FRAMESET COLS="33%, 33%, 33%" ROWS="100%">
<FRAME SRC="documentA.html">
<FRAME SRC="documentB.html">
<FRAME SRC="documentC.html">
</FRAMESET>
```

Another example:

document A	document B
document C	

This framing is accomplished with the following HTML:
```
<FRAMESET COLS="30%, 70%">
<FRAMESET ROWS="200, 100">
<FRAME SRC="documentA.html">
<FRAME SRC="documentC.html">
</FRAMESET>
<FRAME SRC="documentB.html">
</FRAMESET>
```

An HTML document that defines a frame layout does <u>not</u> have a BODY section. The FRAMESET tag and its subordinate tags are used instead. Example:
```
<HTML>
<HEAD>
<TITLE>Basic structure of frameset document</TITLE>
</HEAD>
<FRAMESET COLS="30%, 70%">
<FRAME SRC="documentA.html">
```

```
<FRAME SRC="documentB.html">
</FRAMESET>
</HTML>
```

The main tag is FRAMESET, which is paired. These can be nested, as in the second example above. Each frame within a frameset is defined by the FRAME tag, which is not paired. The FRAMESET tag has attributes ROWS and COLS. The FRAME tag has attributes SRC, NAME, MARGINWIDTH, MARGINHEIGHT, NORESIZE, and SCROLLING.

Alternate content can be provided for those browsers that do not support frames or are configured to not display frames with the paired NOFRAMES tag. This tag can also be used to support search engines: Some search engines are unable to follow links that are contained within framed documents. Some search engines use the first few lines of text within documents in their indices, this text is what is displayed in the search results list. A frameset by itself has no such descriptive text, but this can be remedied with the NOFRAMES tag. Normal text is placed within the tag:

```
<FRAMESET COLS="33%, 67%">
<NOFRAMES>
<H1>The Wonderful World of Stamp Collecting</H1>
<P>Many people consider stamp collection to be a hobby, but it can be more than
that. Stamp collecting can be a business, an art form, or even a topic of
conversation.</P>
<p><A HREF="document C.html">Contents</A></P>
</NOFRAMES>
<FRAME SRC="documentA.html">
<FRAME SRC="documentB.html">
</FRAMESET>
```

The NOFRAMES area can be located at the beginning (immediately after the first frameset tag) or the end of a frameset (just before the last frameset tag). You may include BODY tags within the NOFRAMES tag to help ensure that they exist for the browser or search engine that requires them.

The ROWS attribute specifies the layout of horizontal frames. It is a comma-separated list of pixels, percentages, and/or relative height. When there is only one row and it's height is the same as the browser window, the attribute may be omitted. Pixels specify a fixed height. A percentage specifies a height that is relative to the browser window's height; as the window changes height, so too does the frame.

The COLS attribute specifies the layout of vertical frames. It is a comma-separated list of pixels, percentages, and/or relative widths. When there is only one column and it's width is the same as the browser window, the attribute may be omitted. Pixels specify a fixed width. A percentage specifies a width that is relative to the browser window's width; as the window changes width, so too does the frame.

The SRC attribute identifies the address (URI) of the file that is initially contained in the frame.

The NAME attribute assigns a name to the frame. This name can be used as the target of subsequent links.

The NORESIZE attribute specifies that the size of the frame may not be changed by the reader. Its absence means that the frame may be resized.

The SCROLLING attribute specifies if scroll bars should be used in the frame. The value choices are: yes (always), no (never), auto (when necessary). The auto value is the default.

The FRAMEBORDER attribute describes the frame's border. The value choices are: 1 (yes, border exists), 0 (no, border does not exist). The 1 value is the default.

The MARGINWIDTH attribute specifies the inside left and right margins as a number of pixels. The value must be greater than zero. The default value depends on the browser.

The MARGINHEIGHT attribute specifies the inside top and bottom margins as a number of pixels. The value must be greater than zero. The default value depends on the browser.

When using frames, consider what choices are available to the reader who opens a document (meant to be framed) directly. This usually happens when the individual documents are indexed by a search engine. When a reader clicks on a document in a search results list, that document is opened without regard for the frameset for which it was designed. You can provide navigation for such eventualities with links in the document and/or use JavaScript to force the document to open within a frameset.

Internal Frames

A document can be inserted/embedded within a block of text with the paired IFRAME tag. Inserted content cannot be resized. Technically an iframe is an inline element, not a block-level element; it does not imply a line break before or after.

The content of an iframe can be:
▪ just text
▪ text-level markup
▪ block-level markup (including say the <P> tag), best to use <DIV>

Example:
```
<DIV>
Message:
<IFRAME ID="first" SRC="message.html" FRAMEBORDER="0" WIDTH=200
HEIGHT=50></IFRAME>
<BR>
Special conditions:
<IFRAME ID="second" SRC="spec.html" FRAMEBORDER="0" WIDTH=250
HEIGHT=80></IFRAME>
</DIV>
```

The IFRAME tag has attributes SRC, NAME, FRAMEBORDER, WIDTH, HEIGHT, MARGINWIDTH, MARGINHEIGHT, SCROLLING, ALIGN.

The SRC attribute identifies the address (URI) of the file that is initially contained in the frame.

The NAME attribute assigns a name to the frame. This name can be used as the target of subsequent links.

The FRAMEBORDER attribute specifies if a separator shold be used between the current frame and every adjoining frame. The value choices are: 0 (no separator), 1 (yes, use separator). The 0 value is the default.

The WIDTH attribute specifies the width of the frame in pixels.

The HEIGHT attribute specifies the height of the frame in pixels.

The MARGINWIDTH attribute specifies the inside left and right margins as a number of pixels. The value must be greater than zero. The default value depends on the browser.

The MARGINHEIGHT attribute specifies the inside top and bottom margins as a number of pixels. The value must be greater than zero. The default value depends on the browser.

The SCROLLING attribute specifies if scroll bars should be used in the frame. The value choices are: yes (always), no (never), auto (when necessary). The auto value is the default.

The ALIGN attribute specifies the horizontal alignment of the framed content with respect to its surrounding block. Values are: left (default), center, right, justify. This attribute is deprecated.

You may want the height of the IFRAME element to be dynamic, i.e., to reflect/fill available space. This can be done with CSS and/or JavaScript depending on the situation. In CSS it seems as if you could specify `height: 100%`, but this has not worked for me with IE7 or Firefox 3. JavaScript can be used to set the height to that of another element, you must determine any needed adjustment by trial-and-errror: include this code in the document presented within the IFRAME at its end, just before the </BODY> tag:

```
<script type="text/javascript">
parent.window.document.getElementById("<FRAME_NAME>").height =
document.body.offsetHeight
</script>
```

Caution: This technique many not work properly. Instead you may have to hard code the height. You can use the following code to learn the height of a related element, be sure to place it after that element has been rendered:

```
<script type="text/javascript">
alert(document.getElementById("projleft").offsetHeight);
</script>
```

It will display the height of the left cell, then I set the iframe height, in the HTML, to a smaller number. When done, remove the alert.

Forms

Information can be presented and/or captured on a form. A form has one or more different controls. There are eight types of controls which can be used: button, checkbox, radio button, menu, text input, file select, hidden, and object. A form requires at least two HTML files: (1) the document containing the form and (2) the document that processes the form data. Form processing is done with a programming language like JavaScript.

This is described in detail in a separate document.

Interaction

Interaction is the dynamic response of a web page to the user's actions. Interaction is generally done by an event handler, as JavaScript code. An event handler is established (registered) in HTML by the inclusion of an event attribute in an HTML element. It can also be registered as a JavaScript property; consult a JavaScript reference for information on this method. There are 20 events that can be used to initiate interaction, most of which have to do with specific mouse or keyboard actions. Interaction can also be done as a pseudo hypertext link.

CSS has some interaction capabilities which are often limited to the A element. While there are a number of CSS events, few currently enjoy cross-browser support.

HTML Events

HTML events are the primary basis for interaction. An event usually happens when the user does something, like move the pointer, press a key, resize the window. Most events go unnoticed, interaction happens when an event triggers an event handler. The event attribute identifies the event and the handler.

The following discussion applies to event registration by HTML attribute. A sample HTML statement:

```
<DIV onmouseover="pingMe()"> . . . </DIV>
```

In this case the event handler is a JavaScript function called "pingMe." It is executed when the DIV with the HTML event attribute is moused over.

Below are the event attributes that can be included in different HTML elements to define event actions. At this time (2011) these events are standard in the W3C and in the modern browsers.

Event Actor	HTML Event Attribute	The event occurs when...
body, frameset	onunload	The user exits the page and the document is removed
body, frameset, image	onload	A page or frame is finished loading
form	onblur	An element loses focus
form	onchange	The content of a field changes
form	onfocus	An element gets focus
form	onreset	The form is reset
form	onselect	Text is selected
form	onsubmit	The form is submitted
keyboard	onkeydown	A keyboard key is pressed
keyboard	onkeypress	A keyboard key is pressed or held down
keyboard	onkeyup	A keyboard key is released
mouse	onclick	Mouse clicks an object
mouse	ondblclick	Mouse double-clicks an object
mouse	onmousedown	A mouse button is pressed
mouse	onmousemove	The mouse is moved

Event Actor	HTML Event Attribute	The event occurs when...
mouse	onmouseout	The mouse is moved off an element
mouse	onmouseover	The mouse is moved over an element
mouse	onmouseup	A mouse button is released
window	onerror	An error occurs when loading a document or an image; not an HTML attribute, must be registered in JavaScript
window, frame	onresize	A window or frame is resized

Most of these attributes apply to most HTML elements. When in doubt consult the Index of HTML 4 Attributes at http://www.w3.org/TR/html4/index/attributes.html for the related elements. And consult QuirksMode for browser incompatabilites, http://www.quirksmode.org/js/events_properties.html

There is no right click event. Because of browser differences, the best event to use is mousedown or mouseup, each of which has a DOM property that indicates which button was clicked: button = 2 indicates the right button.

Other examples of an event handler in HTML:
```
<BODY onload="myFunction()" onresize="myOtherFunction()">
<LI onmouseover="myMouseFunction()">
<IMG onmouseout="xf()">
<P STYLE="cursor:pointer" ONCLICK="alert('hello');">Click me!</p>
```

where "myFunction," "myOtherFunction," "myMouseFunction," and "xf" are the names of a JavaScript function defined elsewhere. The last example shows the inclusion of JavaScript code. When the event is fired, the code is run.

A "rollover" is the effect you see when one image replaces another as your mouse moves over it. There are really two images: the image without the mouse over it and the image with the mouse over it. Javascript replaces one with the other when the mouse moves over (onMouseOver) and when it stops being over (onMouseOut).
```
<IMG src="image1.jpg" onmouseover="this.src='image2.jpg'"
onmouseout="this.src='image1.jpg'">
```

Pseudo Hypertext Link

JavaScript can be activated by a hypertext link, what I'm calling a pseudo link:
```
<A HREF="mypage.html"> . . . </A>
<A HREF="javascript:myfunction()"> . . . </A>
<A HREF="javascript:alert('hello')"> . . . </A>
```

When the user clicks the link, either the page named by the URL is loaded or the JavaScript code is run.

Keyboard Operation

It does not seem that the [Tab] key operation qualifies as interaction . . . BUT getting it to work in a particular way can improve the user's experience. Normally in a web page the [Tab] key moves the focus from one link or form control to the next. Both links and form fields can be accessed in the

order in which they appear in the source code. Their tab order can alternatively be specified in the TABINDEX attribute.

It is possible to block the focus, and prevent tabbing to a link, with the onfocus event:

```
<A . . . ONFOCUS="blur()">
```

It is possible to use JavaScript to control the behavior of the arrow keys, overriding their native behavior.

Tags for HEAD Section

Document type declaration:
Document type declaration specifies HTML version and document type definition (DTD). This **should be the first tag**, before the <HTML> tag.

There are three choices, one of which should be used:
<u>HTML 4.01 Strict DTD</u> excludes deprecated elements and attributes and frames:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
```

<u>HTML 4.01 Transitional DTD</u> includes deprecated elements and attributes and excludes frames:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
```

<u>HTML 4.01 Frameset DTD</u> includes deprecated elements and attributes and frames:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
    "http://www.w3.org/TR/html4/frameset.dtd">
```

IE6 uses the !DOCTYPE declaration to switch on its CSS standards-compliant mode.

Meta data:
Meta data is defined as data about data. It is used to declare certain information to browsers and search engines. There are several attributes: NAME identifies a property. CONTENT declares the property's value. HTTP-EQUIV is used instead of NAME and is used by HTTP servers to create HTTP response message headers. LANG declares the language of the value of the CONTENT tag.

Typical NAMES are <u>author</u>, <u>keywords</u>, and <u>description</u>. The last two properties are used primarily by search engines.

Examples:

```
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=ISO-8859-1">
```
   this specifies (1) the content is in simple text format and should be interpreted as HTML and (2) the default document character encoding which is consistent with HTML 4.1; it is best placed immediately following the HEAD tag

```
<META HTTP-EQUIV="Content-Script-Type" CONTENT="text/javascript">
<META HTTP-EQUIV="Content-Style-Type"  CONTENT="text/css">
```
   needed when CSS is inlined

```
<META NAME="description" CONTENT="shows up in search results list." LANG="en">
<META NAME="keywords" CONTENT="keyword a, keyword b, etc.">
```
   used by search engines to classify content

```
<META NAME="author" CONTENT="name of author">
<META NAME="robots" CONTENT="noindex,nofollow">
```
   tells robots not to index the contents of the page and not to scan it for links to follow

Robots can ignore the robots META tag, especially those looking for security vulnerabilities and email address harvesters.

Use the following to activate IE 8 mode, so CSS features new to IE 8 will work.
```
<META HTTP-EQUIV="X-UA-Compatible" CONTENT="IE=8">
```

Redirect:
One web document can redirect the browser to a second web document. This can be helpful when a website has moved. Use the syntax below. The number is how many seconds to wait before being redirected; if set to 5 or 10, readers have enough time to view the page that may be telling them why they are being redirected; if set to 0, may work almost instantaneously and be invisible to readers. There is a space following the semi-colon (;) then the text URL= and follow it with your own URL .
```
<META HTTP-EQUIV="refresh" CONTENT="5; URL=http://a.b.com/d/e/index.html">
```

Title:
The document title is displayed in the browser's title bar with the paired TITLE tag. Regular and special characters can be included in the text, but there is no font specification. Example:
```
<TITLE>This is the document title&#8212;but you knew that (: </TITLE>
```

The TITLE tag is the most important element that a search engine indexes. Every document should have one and it should reflect the topic and parentage. "Search Results" is insufficient!

Links to related files/documents:
The LINK tag defines a relationship between the current document and another resource. A primary use is to define a relationship with stylesheets in external files. Example:
```
<LINK REL="stylesheet" TYPE="text/css" HREF="mystyles.css"> for stylesheet
```

LINK has attributes:
- REL has values: stylesheet, alternatestylesheet, shortcut icon.
- TYPE for an external stylesheet specifies the style sheet language
- HREF is the address of the linked resource.
- TITLE can be used with only one REL="stylesheet", the title marks the stylesheet as preferred.
- MEDIA for an external stylesheet specifies the intended rendering medium or media; has values: screen (default), print, projection, aural, braille, tty, tv, alt. The ones I use are screen and print.

Icon in browser title bar:
You can place an icon in the browser title bar, address bar, page tab, and/or favorites entry with the "shortcut icon" link. The icon must be square of size 16×16, 32×32 or 64×64 pixels; it can have a color depth of either 8-bit or 24-bit. Wikipedia has a good explanation of this. Whereas the .ico format was registered with MIME, the use of icons in this way is not a part of the W3C standard, nevertheless, most browsers support them.
```
<link rel="SHORTCUT ICON" href="browser-icon.ico"
    type="image/vnd.microsoft.icon">
```

Alternatively, you can save the icon with the default file name of **favicon.ico** in the root directory of your domain—for example, **www.microsoft.com/favicon.ico**. The first time a user visits the web page, Internet Explorer automatically searches for this file and places the icon in the address bar, next to all favorites linking to your site, and on page tabs.

Inclusion of scripts:

Scripts are programs that accompany an HTML document and run on the reader's computer. They provide a means to make a normally static HTML document active and/or interactive.

Scripts are placed within a document by the paired SCRIPT tag. It has attributes: SRC identifies the location of an external script. TYPE identifies the scripting language. A script is either placed within the paired tags or a reference to it is placed there. When scripts are used, there must be a META tag that declares the use of scripts:

```
<META HTTP-EQUIV="Content-Script-Type" CONTENT="text/javascript">
```

Examples:
```
<SCRIPT TYPE="text/javascript" SRC="path/filename"
LANGUAGE="JavaScript"></SCRIPT>
<SCRIPT TYPE="text/javascript" > …some JavaScript code…</SCRIPT>
<SCRIPT TYPE="text/vbscript" > …some VBScript code…</SCRIPT>
```

JavaScript runs on all browsers while VBScript runs only on Internet Explorer.

Inclusion of styles:

Styles (CSS) provide a means to specify the format and layout of HTML elements in a way that is separate from the basic HTML tags and attributes. Styles can be included in an HTML document in three ways: (1) inline with the STYLE attribute (see page 4); (2) embedded with the paired STYLE tag, and (3) referenced as an external file with the LINK tag. The STYLE tag is placed in the HEAD section. Example:

```
<STYLE TYPE="text/css">
body  { font-family: verdana, arial; font-size: 10pt; font-color: blue }
</STYLE>
```

Favicon

A *favicon* (short for favorites icon) is a square image, 16×16 or 32×32 pixels, which is displayed in the browser's address bar when a particular website or webpage is displayed. Browsers may also display the favicon in the bookmark list and web page tab. This originated with Microsoft IE4. There are two methods of identifying the favicon to the browser:

- The icon is contained in a file called **favicon.ico** which is located in the web site's root directory.
- The W3C HTML specification now provides for an alternate method that it considers preferred:
  ```
  <link rel="icon" href="/images/myicon.ico">
  <link rel="shortcut icon" href="http://www.example.com/myicon.ico">
  <link rel="icon" type="image/gif" href="http://example.com/webicon.gif">
  <link rel="icon" type="image/vnd.microsoft.icon" href="favicon.ico">
  ```

There are three file types that may be used:
- ICO is the file format for Windows icon images.
- PNG can include transparent pixels.
- GIF can include transparent pixels.

As enticing as a transparent background may seem, it can make the actual image hard to see when the background for the icon file as a whole is not white. Given the size of the image, subtle colors may not be as successful as strong colors.

Miscellaneous

Protection from Spam

If you code a complete email address in a web page, the spambots will find it. And you will be deluged with spam forever. Avoid this nightmare by

- do not use "mailto:" code.
- using JavaScript to dynamically create the email address from snippets:
  ```
  function contact()
  {
  var adr = "sjd" + "orey@s" + "assy" + "me.net";
  alert("Email: " + adr + "\n\nTelephone: 123-456-7890");
  }
  ```
- encode each character in the address in its numerical equivalent.
  ```
  char  code
  a         &#97;
  b         &#98;
  c         &#99;
  1         &#49;
  2         &#50;
  3         &#51;
  ```

A freeware program makes this conversion for you at http://automaticlabs.com/products/encoder.

The following example is JavaScript code that puts my email address into a variable that is returned by the function:

```
<script language="javascript">
function hiveware_enkoder(){var i,j,x,y,x=
"x=\"783d227c4126253232452a3b3c6e3d3f27253232713e45642a3e3e29363649453f2e42" +
"6e3f40423c3f72426e2e3f434727414249337042334742727b71437e34403e754971433c" +
"47434375483e422936363a4543787140427a73413e25323274443e47753e3f37496f423e35" +
"3d3e293639273c4237494443237a3e42712141426f6d423f7c71704134336f3e31333e4237" +
"29363670423a213d42236e4542212532326f437e3440427538723f3e356d4335473f432b27" +
"4143642a433f4223453d422f2f7140432532323c43457d3f42766d3d437b6b414369783f42" +
"6d303d42253232317142436e4042777a724330713e4245384143437145424425323240423" +
"7441436d763f426f7c454270434342713371433313f3e25323572714245253232323f42366b" +
"7242706970417a4b6f3e776c3e3e6d493c437c30404271314543353c404243716f426e3041" +
"3f724470413b3a6f3e31723e4133456f3e413c3e3f4323714033453f425b7c72427a717143" +
"766f4042366e3d427a773f43754b403e70693c427a4b7042776c413f6d306f3e7231724227" +
"232a4372456d7e697430253232366b70697a497c30383131432532324525323236767b7d6a7b" +
"7c7a3039314323452f2f436e777a307145384371442532323674676d766f7c70437313345 3c31" +
"2532352333345253232367b7d6a7b7c7a3071343a3143276e777a3071453a43714425323236" +
"746d766f7c70437313345 3c31253235233334525323236 7b7d6a7b7c7a3071343a3143272345" +
"23367b7d6a7b7c7a30723143263f7d412b2b3f7c4179726977767674692c7c2d3f6a73762c" +
"6d41343f6d407c327069726b786c3f6d2f2f2d216e417c32676c65764773686945782c6d2d" +
"31383f6d6a2c6e4037362d6e2f413d383f7d2f4157787 66d726b326a767371476c6576 4773" +
"68692c6e2d237d223b793d27273b783d756e65736361706 52878293b666f7228693d303b69" +
"3c782e6c656e6774683b692b2b297b6a3d782e63686172436f646541742869292d343b6966" +
"286a3c3332296a2b3d39343b792b3d537472696e672e66726f6d43686172436f6465286a29" +
"7d79\";y='';for(i=0;i<x.length;i+=2){y+=unescape('%'+x.substr(i,2));}y";
while(x=eval(x));}hiveware_enkoder();
</script>
```

References and Resources

W3C	http://www.w3c.org
explanation of ASCII	http://czyborra.com/charsets/iso646.html
explanation of SGML	http://www.w3.org/MarkUp/SGML/
HTML 4.01 specification	http://www.w3.org/TR/html4/
special characters	http://hotwired.lycos.com/webmonkey/reference/special_cha racters/
support by browsers	http://developer.netscape.com/evangelism/docs/technotes/xre f/html-element/
support by browsers, Netscape Gecko vs. IE (Netscape commits to full compliance!)	http://wp.netscape.com/browsers/future/standards.html
support by browsers	http://www.westciv.com/style_master/academy/browser_sup port/index.html