Spinning the Web: A Hands-On Introduction to Building Mosaic and WWW Documents

by James Powell
Scholarly Communications Project
(http://scholar.lib.vt.edu/)
University Libraries
(http://vatech.lib.vt.edu/)
Virginia Polytechnic Institute and State University
(http://www.vt.edu/)
Introduction to HTML

The HyperText Markup Language (HTML) is an SGML application for marking up documents for inclusion in the World Wide Web. It, along with the World Wide Web were invented by Tim Berners-Lee of the CERN High Energy Particle Physics Laboratory in Geneva, Switzerland. HTML allows you to:

- Publish documents to the Internet in a platform independent format
- Create links to related works from your document
- Include graphics and multimedia data with your document
- Link to non-World Wide Web information resources on the Internet

What is Markup?

To use HTML, you must understand the concept of markup:

- **Markup** is the act of inserting additional text into a document which is not usually visible to the reader, and is not part of the content, but enhances the document in some way, such as capturing document structure or adding hypertext capability.
- **Markup** also refers to the additional text, also known as tags, which are inserted in the document.

An example of markup:

```
Grocery list
<UL>
<LI>Apples
<LI>Oranges
</UL>
```
HTML Markup

HTML tags are used to markup the structure of a document, as well as some formatting information. HTML has two types of markup: tags and character entities.

**Tags** are constructed of brackets between which the tag is placed. Tags are placed around segments of text, so there is usually a companion end tag which is identical to the start tag except it includes a forward slash. Here are start and end tags for a title:

<TITLE>Introduction to HTML</TITLE>

HTML also includes markup called **character entities**. These are used to include international characters as well as characters usually included in tags as markup. Here is a character entity for an ampersand:

`&` &

Components of HTML Markup

Here is a second example of HTML markup:

<A HREF="Virginia.gif">More about Virginia</A>

- `<A` is the anchor tag (tags are also referred to as **elements**).
- `HREF="Virginia"` is an attribute of the anchor tag
- `>` closes the anchor tag
- The phrase **More about Virginia** is the tag’s contents
- `</A>` is an end tag for the anchor tag.

Tags can have elements, which are only allowed between them. For example, all HTML tags are elements of the `<HTML>` tag.

Some tags lack contents, attributes or end tags.

HTML Levels

There are currently three levels of HTML conformance. Each encompasses a set of tags and higher levels include tags from all those below it.
Level 0
The minimum tags which constitute an HTML document (most tags currently in use). Level 0 tags are usually rendered consistently from browser to browser.

Level 1
Level 0 tags plus tags for highlighting (also called Logical Tags) and images

Level 2
Level 0 and Level 1 tags plus form tags

Most browsers support all Level 0 tags covered in this presentation. Some level 1 tags such as `<EM>` for emphasis are not widely supported. Only very recent releases of browsers such as 2.0 Mosaic support Level 2 form tags.

Document-wide tags

**HTML, HEAD, BODY**

Each HTML document is contained within the `<HTML>` tag. This tag is not required by all browsers but using it is good practice and may be required in the future.

Each HTML document also includes a header section indicated by the `<HEAD>` tag which contains information about the document. It should always be present and at least contain the `<TITLE>` with the document title.

The remainder of the HTML document should be enclosed by the `<BODY>` tag.

The minimal HTML document:

```
<HTML>
<HEAD> <TITLE> Minimal </TITLE> </HEAD>
<BODY>
</BODY> </HTML>
```

**HEAD Tags**

The following tags are elements of the `<HEAD>` tag and cannot appear outside the header section of the document:

`<TITLE>`
This should be a unique and descriptive title for the document. While it is not
displayed with the document, many browsers display the title elsewhere and use it when constructing hotlist entries.

<TITLE>Introduction to HTML</TITLE>

<BASE>
Sets the URL for relative links within the current document. This tag is not required but can be useful when a document includes many local hypertext links.

<Base href="/reports/annual-1994/">

<ISINDEX>
Indicates a document is keyword searchable - this is usually an unnecessary tag as documents are considered searchable by default.

<ISINDEX>

HEAD Tags

More tags used within the document header

<NEXTID>
This is usually generated by automated markup systems as a portion of anchors within this particular document. There is no need to insert this tag manually.

<NEXTID X=A10>

<LINK>
A Level 1 HEAD element that can be used to indicate relationships between documents such as next and previous documents, glossaries, related indexes, document author. Multiple <LINK> tags can be used to include multiple relationships:

<Link mailto:jpowell@vt.edu>
<Link glossary.html>

<!-- --> (comment tags)
Comment tags may actually occur anywhere in an HTML document but are most commonly used in the document header to encode miscellaneous information about a document, such as version, creation date, last update, etc.

<!-- Version 2.1, last updated August 22, 1994 -->
Document Body tags

Headings

HTML provides six levels of heading tags (<H1>-<H6>). All levels are rendered larger than surrounding text, with the lowest level rendered at the largest size. Heading tags also insert vertical space between the text marked and the surrounding text.

You should avoid skipping levels to achieve a certain visual effect as there is no guarantee how the headings will be rendered on any given system. They are intended to indicate document structure.

Here is a markup example along with tagged examples for all six levels:

<H1>Level 1</H1>

Level 1

Level 2

Level 3

Level 4

Level 5

Level 6

Information Type Elements

Information type elements are used to markup information content in a document. There are tags for highlighting sections of text, for definitions and citations, and computer oriented data such as program source code and output. Since these are level 1 conformant tags, appearance is not guaranteed to be the same from browser to browser.

Highlighting Elements

• <EM> indicates this portion of text should be emphasized (usually italicized)
• <STRONG> indicates stronger emphasis than <EM> (usually bold)

**General Text Elements**
• <ADDRESS> is used to record information that can be used to contact the document author.
• <DFN> is used to markup a definition *(proposed tag - not yet supported)*
• <CITATION> is used to markup a citation from another document
• <STRIKE> indicates that the selected text should be struck out of the document *(proposed - not widely supported)*

**The following tags are intended for computer related information:**
• <CODE> is used to markup sections of program code
• <SAMPLE> indicates the data is output from a computer application
• <KBD> represents the key a user should enter when responding to a computer prompt
• <VAR> marks a variable

**Information Type Examples**

Each example includes segments of text marked up by the tag listed:

<EM>Emphasized text</EM>:

*Emphasized text*

<STRONG>Strong emphasis</STRONG>:

*Strong emphasis*

<CITATION> HTML 2.0 Specification</CITE>:

*HTML 2.0 Specification*

<DFN> HTML stands for HyperText Markup Language</DFN>:

*HTML stands for HyperText Markup Language*

<STRIKE>completely unacceptable</STRIKE>:

*completely unacceptable*

<CODE>for (i=0; i<10; i++) { printf("Count = %d","i); } </CODE>:

for (i=0; i<10; i++) { printf("Count = %d","i); }

<SAMPLE>Count = 1 Count = 2</SAMPLE>:

Count = 1 Count = 2
Enter <KBD>Q</KBD> to quit :
Enter Q to quit

VAR>Percent</VAR> :
Percent

Most browsers do not render all of these tags differently than surrounding text, although the specification indicates most should do so. It is a good idea to use these tags as needed regardless of whether they are displayed differently, since they enhance the accessibility of the document content.

Physical Style Elements

HTML has a few tags for controlling the appearance of text. There are three text markup elements for controlling the font of a text segment (a fourth is proposed), and three tags for controlling the flow of text.

Font Tags:
These tags are placed around segments of text for the desired effect. A few browsers do not support the <TT> tag and none currently support the proposed <U> tag for underlining text.

<B> will display text in BOLD
<I> will display text in Italics
<TT> will display text in a fixed typewriter-like font
<U> is proposed for displaying text underlined

Text Flow Tags:
Text flow tags should be inserted between sections of text to be combined or divided.
Text flow tags do not have end tags.

<BR> forces a line break
this text is on a new line
<NOBSP> will force a long line to stay together rather than wrap
<HR> will display a rule line between segments of text, like this one:
Text Elements

Text elements, like Information Type Elements are inserted around segments of text as structural elements. There are three HTML tags for text elements:

<P> is intended to mark paragraphs. The paragraph end tag is optional, although it is good practice to insert it.

<PRE> is the preformatted text tag. There are a variety of uses for the preformatted text tag. It can be used to retain formatted ASCII text such as newsletters, calendars, spreadsheets or statistical data in columns (there is not yet an HTML tag for tables). Here is a brief example:

<table>
<thead>
<tr>
<th>World Wide Web Stats</th>
<th>January</th>
<th>February</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document 1</td>
<td>752</td>
<td>697</td>
</tr>
<tr>
<td>Document 2</td>
<td>134</td>
<td>232</td>
</tr>
</tbody>
</table>

<BLOCKQUOTE> has as its content sections of text included from other sources:
The World Wide Web Initiative (W3) links information throughout the world.

Lists - Unordered, Ordered

HTML supports five types of lists. A list is first marked with the start and end list tag and then each list item is indicated with a list item tag <LI> (unless it is a definition list). List item tags have end tags, but they are optional since a new list item tag implies the end of the previous item. Lists may be nested and if they are nested lists are indented farther than their parent list when displayed.

Unordered List: <UL>
A list of items which may appear in any particular order. It is usually displayed as a bulleted list of items.

<UL>
  <LI>Apples • Apples
  <LI>Oranges • Oranges
</UL>

Ordered List: <OL>
A list of items to be displayed in a particular order. These are usually numbered when displayed.

<OL>
Lists - Definition List

A definition list `<DL>` is a list of terms `<DT>` and their definitions `<DD>`. Each definition is usually displayed indented slightly in relation to the term. Each term should have a corresponding definition.

```html
<DL>
  <DT>HTML</DT>
  <DD>HyperText Markup Language</DD>
  <DT>SGML</DT>
  <DD>Standard Generalized Markup Language</DD>
</DL>
```

HTML

  HyperText Markup Language

SGML

  Standard Generalized Markup Language

Lists - Menu, Directory

A directory list `<DIR>` is a list of brief items, less than twenty characters long each.

```html
<DIR>
  <LI>jte-v1n1</LI>
  <LI>jte-v1n2</LI>
  <LI>jte-v2n1</LI>
  <LI>jte-v2n2</LI>
</DIR>
```

jte-v1n1
jte-v1n2
jte-v2n1
jte-v2n2

A menu list `<MENU>` is a compact list of items, usually one per line.

```html
<MENU>
  <LI>Press 1 for help</LI>
</MENU>
```
Review: Structural Elements

Document-wide

HTML - all tags are elements of the <HTML> tag:
Level 0    HEAD (TITLE, BASE, ISINDEX, NEXUID)
            <!-- -->
            BODY
Level 1    HEAD (LINK)

Document-body

BODY - all tags not elements of <HEAD> are elements of the <BODY> tag:
Level 0    H1, H2, H3, H4, H5, H6
            B, I, TT, U, BR, NOBSP, HR
            P, PRE, BLOCKQUOTE
            UL, OL, MENU, DIR (LI)
            DL (DT, DD)
            ADDRESS
Level 1    EM, STRONG, CITATION, STRIKE, CODE,
            SAMPLE, KBD, VAR
Adding Hypertext...  
HTML Anchor Tag

Anchors are what make HTML a hypertext language. The anchor tag consists of a start tag `<A` plus one or more attributes naming or describing the anchor plus `>` then content which becomes the link, followed by an end tag `</A>`:

```
<A HREF="slide16.html">HTML Anchor Tag</A>
```

There are basically two types of anchors: **start** and **destination**. Start anchors are selectable segments of text such as **HTML Anchor Tag** above, while destination anchors are portions of text that mark an available destination. Here is an example of a start and a destination in which the start example is pointed to by the destination example:

```
Start:          <A NAME="SGML">Standard Generalized Markup Language</A>
Destination:    <A HREF="#SGML">SGML</A>
```

Building Links to HTML Files

Links are built using the **HREF** attribute with the anchor tag. **HREF** must be assigned some value, a target value for a destination. The target can be within the same document, another document on the same server (filename), a document on a different server (URL), or a portion of text in another document.

```
<A HREF="#SGML">More information about SGML</A>
```

provides a link to a target named SGML in the current document.

```
<A HREF="http://scholar.lib.vt.edu/html-intro.html">More information about HTML</A>
```


```
<A HREF="http://scholar.lib.vt.edu/html-intro.html#SGML">More information about SGML</A>
```

links to a target named SGML in a file called html-intro.html on the World Wide Web.

Finally, an anchor can be both a link and a target:

```
<A NAME="SGML2" HREF="#SGML">if you are still lost</A>
```

can be pointed to with the name **SGML2** and points to a target called **SGML**.

*Any value assigned to an attribute must be enclosed by double quotes.*
Using Uniform Resource Locators

Anchors can link to remote data when a Uniform Resource Locator (URL) is used with the HREF attribute. Any type of data on almost any type of information server on the Internet can be accessed using a URL. The URL has three main components:

- **Server/Resource Type**
  - file (File Transfer Protocol)
  - gopher
  - http (World Wide Web)
  - news (Usenet News)
  - telnet

- **Internet Name** (and port if required)

- **Filename and path**

Most URLs include the characters :// to divide the server type from the internet address, except for Usenet news URLs. Filename can be truncated to a forward slash / which tells the server to send a default document or directory listing. Telnet does not require a filename or path.

Here are example URLs for several common resource types:

file://scholar.lib.vt.edu/pub/next/HTML-Editor.FAT.compressed
gopher://vatech.lib.vt.edu/
http://scholar.lib.vt.edu/library.html
news:comp.infosystems.www.providers
telnet://vtls.vt.edu

Summary of Anchor Attributes

**HREF**

HREF makes an anchor a hypertext link. It is used with an anchor name, filename, or Uniform Resource Locator.

**NAME**

NAME assigns an identifier to a piece of text which can then be linked to by other anchors.

**TITLE**

TITLE is an optional attribute which is supported by some browsers. It specifies that the contents of the <TITLE> tag in the HEAD of a target document can be displayed before a user loads the entire document.
Other attributes less commonly used include **URN** (Uniform Resource Number), **METHODS**, and the proposed **REL** and **REV**. Uniform Resource Numbers will become more common as the URN standard is finalized, and may replace URL in the future.

**Images**

Images can be included with HTML documents using the `<IMG>` tag. Images can be icons, small images of characters HTML cannot support, or photographs. The linked image must be in one of two graphics formats:

- Xbitmap (**XBM**)
- Compuserve’s Graphics Interchange Format (**GIF**)

Many tools are available for Macintosh and PC platforms for converting from other graphics formats to those supported by WWW browsers.

**IMG** has four attributes:

- **SRC** is a required attribute that is assigned the filename or URL of the image to be linked.

  `<IMG SRC="icon.gif">`

- **ALIGN** can be set to top, middle or bottom and indicates how text following a graphic should be aligned with the image.

  `<IMG SRC="newman.gif" ALIGN="middle">Newman Library`

- **ALT** specifies text data to be displayed instead of the graphic if the image cannot be displayed.

  `<IMG SRC="warning.gif" ALT="Warning!">`

- **ISMAP** is used to make an image a graphical navigation tool. More about this soon.

**Image Examples**

Here is an icon:

`<IMG SRC="index.xbm">Search archives`

`Search archives`
Here is an image of non-supported characters:

Mathematical symbols \( \sum_{i=1}^{n} x_i = \int_0^1 \) are not supported by HTML

Here is a photograph:

<IMG SRC="newman.gif" ALT="Newman Library">

Images can also be hypertext links:

<A HREF="http://vatech.lib.vt.edu/"> <IMG SRC="newman.gif"> </A>

Special Characters

HTML supports the **ISO-Latin-1** character set, a larger character set than ASCII. ISO-Latin 1 characters are represented as character entities in HTML. These are prefixed by an ampersand and followed by a semicolon. Here is an example for the less than sign:

Capital A with acute accent (á) is represented by &Aacute;

Characters which are used to construct tags and entities must also be represented as entities:
A complete list of ISO-Latin-1 characters and entities is included in your manual.

**Review: Anchors, Images**

**Character Entities**

**Document-body**

*BODY* - all tags not elements of *<HEAD>* are elements of the *<BODY>* tag:

*Level 0*  
A [HREF, NAME, TITLE, URN, METHODS, REL, REV]  
&amp; &lt; &gt; &quot; ISO-Latin-1 character entities

*Level 1*  
IMG [SRC, ALIGN, ALT, ISMAP]
Special Applications of HTML: Forms

Form support is a recent addition to HTML. Forms can be constructed from five level 2 HTML tags:

- FORM
- INPUT
- OPTION
- SELECT
- TEXTAREA

They provide a user with the ability to enter information which can then be processed on the server as survey information, search information for a database, information request, etc.

Forms by themselves only allow data entry. They require software commonly referred to as gateways, which receive the data, process it, and return a response to the WWW client. Gateways are custom applications written in an available programming language on the server (such as Perl or C). Some gateways are available on the Internet for Z39.50, WAIS, and mail forwarding.

Form Tags - FORM

The `<FORM>` tag is placed around a section of an HTML document which includes FORM elements. Other BODY tags can occur in a form, and multiple forms can occur in a document, but forms cannot be nested.

There are two attributes essential to forms:
- **ACTION** indicates the URL of the processing gateway. This URL will point to a program rather than a document. This program will receive the contents of the form in one of two ways depending on what value is specified for the **METHOD** attribute.
- **METHOD** can be assigned one of two values: GET or POST. Gateways can accept data directly when METHOD is GET or look for it in a special variable if POST is used. If you are using an existing gateway, refer to its documentation for the correct METHOD.

**Example:**

```
<FORM METHOD=GET ACTION="http://nebula.lib.vt.edu:8001/cgi-bin/marian-gate">
Sends the contents of a form directly to a gateway called marian-gate.
```

*Form elements should not occur outside `<FORM>` start and end tags.*
Form Tags - INPUT

The `<INPUT>` tag, as its name implies, provides some type of data entry point in the form depending on the value of its `TYPE` attribute:

CHECKBOX and RADIO specify selectable options:

```html
<INPUT TYPE="CHECKBOX">  <INPUT TYPE="RADIO">
```

RESET and SUBMIT make the INPUT field a button to clear the form or send its contents to the gateway:

```html
<INPUT TYPE="RESET">  <INPUT TYPE="SUBMIT">
```

HIDDEN is used to conceal a field that has a preset value that will never change but is unknown to the gateway.

TEXT specifies that a data input field be displayed:

```html
<INPUT TYPE="TEXT">
```

IMAGE displays an image and when the user selects a spot, the coordinates are passed to the gateway.

*INPUT is an empty tag, like IMG, with no end tag.*

Form Tags - More INPUT Attributes

Each INPUT field must have a NAME attribute which is associated with the input value. For example, if you were building an OPAC searching form and had two fields:

```html
Enter a Title:  
<INPUT TYPE=TEXT NAME="Title">

Enter an Author:  
<INPUT TYPE=TEXT NAME="Author">
```

would pair whatever the user enters in the text field with the name Author, and the Title data with the name title, the gateway could construct an author-title search from the
input.

The \texttt{VALUE} attribute will set a default value:
\begin{verbatim}
<INPUT TYPE="TEXT" VALUE="Blacksburg, VA">
\end{verbatim}

\section*{Form Tags - Still More INPUT Attributes...}

Other \texttt{INPUT} attributes include:

\texttt{ALIGN} and \texttt{SRC} used in conjunction with the \texttt{IMAGE} attribute.

\texttt{MAXLENGTH} for limiting the number of characters that can be entered in a \texttt{TEXT} field and \texttt{SIZE} for controlling the displayed length of a \texttt{TEXT} field:
\begin{verbatim}
<INPUT TYPE="TEXT" MAXLENGTH=5 SIZE=5>
\end{verbatim}

\texttt{CHECKED} to select a default for a checkbox or set of radio buttons:
\begin{verbatim}
<INPUT TYPE="CHECKBOX" CHECKED>
<INPUT TYPE="RADIO" CHECKED>
\end{verbatim}

\textit{Despite its complexity, INPUT’s role is strictly data acquisition}

\section*{Form Tags - SELECT, OPTION}

\texttt{SELECT} is the form equivalent of a list tag. It is used in conjunction with the \texttt{OPTION} tag to build a pull down list of options:
\begin{verbatim}
<SELECT NAME="Catalog">
<OPTION>Virginia Tech
<OPTION>University of Virginia
<OPTION>Radford University
</SELECT>
\end{verbatim}

\texttt{SELECT} has a \texttt{MULTIPLE} attribute when users are allowed to choose more than one item.
OPTION has two attributes: SELECTED to configure a default item which is initially selected, and VALUE to specify a value to be returned when a certain option is chosen (otherwise the item label is returned).

**Form Tags - TEXTAREA**

TEXTAREA allows the user to enter multiple lines of text. It has three attributes:

NAME to associate the field data with some identifier, and ROWS and COLS to specify the size of the text area:

```html
<TEXTAREA NAME="Address" ROWS=4 COLS=60>
University Libraries
Blacksburg, VA
24060
</TEXTAREA>
```

The data between the start and end TEXTAREA tag is optional, but the end tag is required even if no default value is specified.

**HTML Form Example**

This form from our library home page serves as a Z39.50 client for searching several locally mounted databases:

```html
<HTML><HEAD>
<TITLE>INFOSHARE Page </TITLE>
</HEAD>
<BODY>
<H1>INFOSHARE Database Query Page</H1>
<IMG SRC="index.xbm"> Search University Libraries INFOSHARE databases using Z39.50.
<P>
<HR>
<FORM METHOD=POST ACTION="http://nebula.lib.vt.edu:8001/cgi-bin/infogate">
<H3>NOTE: Words <B>must</B> be filled in sequential order.</H3>
Database: <SELECT NAME="dbname">
```
<OPTION SELECTed>WHUM Humanities Index
<OPTION>WGSI General Science
<OPTION>WSSI Social Science
<OPTION>WBAI Biology and Agriculture
<OPTION>WAST Applied Science and Technology
<OPTION>WRGA Reader’s Guide Abstracts
<OPTION>WWBA Wilson Business Abstracts
<OPTION>XWIL Test Database
<OPTION>CCON Current Contents
</SELECT>
Records to View: <INPUT NAME="maxrecords" SIZE=4 VALUE=100>
Full or Brief displays:
<SELECT NAME="esn">
<OPTION SELECTED>B
<OPTION>F
</SELECT>
Word 1: <INPUT NAME="term_term_1">
<SELECT NAME="term_use_1">
<OPTION SELECTED>Author
<OPTION>Title
<OPTION>Subject
<OPTION>Word
</SELECT>
Word 2: <INPUT NAME="term_term_2">
<SELECT NAME="term_use_2">
<OPTION SELECTED>Subject
<OPTION>Word
<OPTION>Title
<OPTION>Author
</SELECT>
Word 3: <INPUT NAME="term_term_3">
<SELECT NAME="term_use_3">
<OPTION SELECTED>Title
<OPTION>Subject
<OPTION>Word
<OPTION>Author
</SELECT>
</SELECT>
WORD<INPUT TYPE="radio" NAME="operator" VALUE="and"> AND
<INPUT TYPE="radio" NAME="operator" VALUE="or"> OR
<INPUT TYPE="radio" NAME="operator" VALUE="not"> NOT
</SELECT>
</FORM>
</BODY>
</HTML>
Special Applications of HTML: Image Mappings

The ISMAP attribute of the IMG tag is a highly specialized anchor for graphical navigation. Any GIF image can serve as a graphical navigator. Coordinates for each selectable portion of the image are recorded using a graphics application. The two pair represent the upper left and lower right section of the rectangle (other shapes are possible). A map file is then created with a URL associated with each set of coordinates. You can also specify a default URL for any click detected within the image but outside the coordinate pairs.

Here is an ISMAP example:

```
<A HREF="http://vatech.lib.vt.edu/cgi-bin/imagemap/library-2"><IMG SRC="floor2.gif" ISMAP></A>
```

Example uses include selectable floor maps for libraries, weather maps allowing the user to click on their region, shelf of manuals with selectable spines, and learning exercises such as an image map for anatomy.

ISMAP Map File Example

Here is an image map file and an example HTML document which uses it:

default gopher://scholar.lib.vt.edu/

# Link to Spectrum search
rect gopher://scholar.lib.vt.edu:70/0/about 7,23,512,109

# Link to VPI home page
rect http://www.vt.edu/ 517,47,598,88

# Link to first article - page 1
rect gopher://scholar.lib.vt.edu:70/0/spectrum/sp940512/spt1a.051294.txt 11,125,597,351

# Link to second article - lower left page 1
rect gopher://scholar.lib.vt.edu:70/0/spectrum/sp940512/spt1b.051294.txt 13,354,149,776

# Link to third article - middle bottom page 1
rect gopher://scholar.lib.vt.edu:70/0/spectrum/sp940512/spt1c.051294.txt 157,547,457,775
HTML Tag Summary

Document-wide

HTML - all tags are elements of the <HTML> tag:
Level 0
    HEAD (TITLE, BASE, ISINDEX, NEXTID)
<!-- -->
    BODY
Level 1
    HEAD (LINK)

Document-body

BODY - all tags not elements of <HEAD> are elements of the <BODY> tag:
Level 0
    H1, H2, H3, H4, H5, H6
    B, I, TT, U, BR, NOBSP, HR
    P, PRE, BLOCKQUOTE
    UL, OL, MENU, DIR (LI)
    DL (DT, DD)
    ADDRESS
    A [HREF, NAME, TITLE, URN, METHODS, REL, REV]
    &amp; &lt; &gt; ISO-Latin-1 character entities

Level 1
    EM, STRONG, CITATION, STRIKE, CODE,
    SAMPLE, KBD, VAR
    IMG [SRC, ALIGN, ALT, ISMAP]

Level 2
    FORM [ACTION, METHOD]
    INPUT [TYPE, NAME, VALUE, ALIGN, SRC, MAXLENGTH
            SIZE, CHECKED]
    SELECT [NAME, MULTIPLE] (OPTION [SELECTED, VALUE])
    TEXTAREA [NAME, ROWS, COLS]
References

A Beginner’s Guide to HTML
by NCSA (pubs@ncsa.uiuc.edu)
http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimer.html

A Beginner’s Guide to URLs
by Marc Andraesson
http://www.ncsa.uiuc.edu/demoweb/url-primer.html

Crash Course on writing documents for the Web
by Eamonn Sullivan
http://www.ziff.com/~eamonn/crash_course.html

Composing Good HTML
by James "Eric" Tilton

HTML Quick Reference
by Michael Grobe
http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLQuickRef.html

HTML Specification
by Daniel W. Connolly
http://www.hal.com/users/connolly/html-spec/

HTML+ Discussion Document
by David Raggett
http://info.cern.ch/hypertext/WWW/MarkUp/HTMLPlus/htmlplus_1.html

ISO Latin 1 character entities
derived from ISO 8879:1986/ENTITIES Added Latin 1//EN
http://info.cern.ch/hypertext/WWW/MarkUp/ISOlat1.html

NCSA HTML Style Sheet
by NCSA (pubs@ncsa.uiuc.edu)
http://www.ncsa.uiuc.edu/Pubs/StyleSheet/NCSAStyleSheet.html
## Appendix A: ISO Latin 1 character entities

<table>
<thead>
<tr>
<th>Entity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Æ</td>
<td>capital AE diphthong (ligature)</td>
</tr>
<tr>
<td>Å</td>
<td>capital A, dieresis or umlaut mark</td>
</tr>
<tr>
<td>Æ</td>
<td>capital A, acute accent</td>
</tr>
<tr>
<td>Å</td>
<td>capital A, circumflex accent</td>
</tr>
<tr>
<td>Æ</td>
<td>capital A, grave accent</td>
</tr>
<tr>
<td>Å</td>
<td>capital A, ring</td>
</tr>
<tr>
<td>Æ</td>
<td>capital A, tilde</td>
</tr>
<tr>
<td>À</td>
<td>capital A, grave accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, dieresis or umlaut mark</td>
</tr>
<tr>
<td>À</td>
<td>capital A, acute accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, circumflex accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, grave accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, ring</td>
</tr>
<tr>
<td>À</td>
<td>capital A, tilde</td>
</tr>
<tr>
<td>À</td>
<td>capital A, dieresis or umlaut mark</td>
</tr>
<tr>
<td>À</td>
<td>capital A, acute accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, circumflex accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, grave accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, ring</td>
</tr>
<tr>
<td>À</td>
<td>capital A, tilde</td>
</tr>
<tr>
<td>À</td>
<td>capital A, dieresis or umlaut mark</td>
</tr>
<tr>
<td>À</td>
<td>capital A, acute accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, circumflex accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, grave accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, ring</td>
</tr>
<tr>
<td>À</td>
<td>capital A, tilde</td>
</tr>
<tr>
<td>À</td>
<td>capital A, dieresis or umlaut mark</td>
</tr>
<tr>
<td>À</td>
<td>capital A, acute accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, circumflex accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, grave accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, ring</td>
</tr>
<tr>
<td>À</td>
<td>capital A, tilde</td>
</tr>
<tr>
<td>À</td>
<td>capital A, dieresis or umlaut mark</td>
</tr>
<tr>
<td>À</td>
<td>capital A, acute accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, circumflex accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, grave accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, ring</td>
</tr>
<tr>
<td>À</td>
<td>capital A, tilde</td>
</tr>
<tr>
<td>À</td>
<td>capital A, dieresis or umlaut mark</td>
</tr>
<tr>
<td>À</td>
<td>capital A, acute accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, circumflex accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, grave accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, ring</td>
</tr>
<tr>
<td>À</td>
<td>capital A, tilde</td>
</tr>
<tr>
<td>À</td>
<td>capital A, dieresis or umlaut mark</td>
</tr>
<tr>
<td>À</td>
<td>capital A, acute accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, circumflex accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, grave accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, ring</td>
</tr>
<tr>
<td>À</td>
<td>capital A, tilde</td>
</tr>
<tr>
<td>À</td>
<td>capital A, dieresis or umlaut mark</td>
</tr>
<tr>
<td>À</td>
<td>capital A, acute accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, circumflex accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, grave accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, ring</td>
</tr>
<tr>
<td>À</td>
<td>capital A, tilde</td>
</tr>
<tr>
<td>À</td>
<td>capital A, dieresis or umlaut mark</td>
</tr>
<tr>
<td>À</td>
<td>capital A, acute accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, circumflex accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, grave accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, ring</td>
</tr>
<tr>
<td>À</td>
<td>capital A, tilde</td>
</tr>
<tr>
<td>À</td>
<td>capital A, dieresis or umlaut mark</td>
</tr>
<tr>
<td>À</td>
<td>capital A, acute accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, circumflex accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, grave accent</td>
</tr>
<tr>
<td>À</td>
<td>capital A, ring</td>
</tr>
<tr>
<td>À</td>
<td>capital A, tilde</td>
</tr>
</tbody>
</table>
Appendix B:  
A Beginner’s Guide to URLs  
Marc Andraesson (http://www.ncsa.uiuc.edu/demoweb/url-primer.html)

What’s a URL? A URL is a Uniform Resource Locator. Think of it as a networked extension of the standard filename concept: not only can you point to a file in a directory, but that file and that directory can exist on any machine on the network, can be served via any of several different methods, and might not even be something as simple as a file: URLs can also point to queries, documents stored deep within databases, the results of a finger or archie command, or whatever.

Since the URL concept really pretty simple ("if it’s out there, we can point at it"), this beginner’s guide is just a quick walk through some of the more common URL types and should allow you to be creating and understanding URLs in a variety of contexts very quickly.

File URLs

Suppose there is a document called "foobar.txt"; it sits on an anonymous ftp server called "ftp.yoyodyne.com" in directory "/pub/files". The URL for this file is then:

```plaintext
```

The toplevel directory of this FTP server is simply:

```plaintext
file://ftp.yoyodyne.com/
```

The "pub" directory of this FTP server is then:

```plaintext
file://ftp.yoyodyne.com/pub
```

That’s all there is to it.

Gopher URLs

Gopher URLs are a little more complicated than file URLs, since Gopher servers are a little tricker to deal with than FTP servers. To visit a particular gopher server (say, the gopher server on gopher.yoyodyne.com), use this URL:

```plaintext
gopher://gopher.yoyodyne.com/
```
Some gopher servers may reside on unusual network ports on their host machines. (The default gopher port number is 70.) If you know that the gopher server on the machine "gopher.banzai.edu" is on port 1234 instead of port 70, then the corresponding URL would be:

```
gopher://gopher.banzai.edu:1234/
g```

**News URLs**

To point to a Usenet newsgroup (say, "rec.gardening"), the URL is simply:

```
news:rec.gardening
```

Currently, network clients like NCSA Mosaic don’t allow you to specify a news server like you would normally expect (e.g., news://news.yoyodyne.com/rec.gardening ); this may be coming down the road but in the meantime you will have to specify your local news server via some other method. The most common method is to set the environment variable NNTPSERVER to the name of your news server before you start Mosaic.

**HTTP URLs**

HTTP stands for HyperText Transport Protocol. HTTP servers are commonly used for serving hypertext documents, as HTTP is an extremely low-overhead protocol that capitalizes on the fact that navigation information can be embedded in such documents directly and thus the protocol itself doesn’t have to support full navigation features like the FTP and Gopher protocols do.

A file called "foobar.html" on HTTP server "www.yoyodyne.com" in directory "/pub/files" corresponds to this URL:

```
```

The default HTTP network port is 80; if a HTTP server resides on a different network port (say, port 1234 on www.yoyodyne.com), then the URL becomes:

```
```

**Partial URLs**
Once you are viewing a document located somewhere on the network (say, the document http://www.yoyodyne.com/pub/afile.html ), you can use a partial, or relative, URL to point to another file in the same directory, on the same machine, being served by the same server software. For example, if another file exists in that same directory called "anotherfile.html", then anotherfile.html is a valid partial URL at that point.

This provides an easy way to build sets of hypertext documents. If a set of hypertext documents are sitting in a common directory, they can refer to one another (i.e., be hyperlinked) by just their filenames -- however a reader got to one of the documents, a jump can be made to any other document in the same directory by merely using the other document’s filename as the partial URL at that point. The additional information (access method, hostname, port number, directory name, etc.) will be assumed based on the URL used to reach the first document.

Other URLs

Many other URLs are possible, but we’ve covered the most common ones you might have to construct by hand. At the top of each Mosaic document viewing window is a text field called "Document URL"; if you watch the contents of that as you navigate through information on the network, you’ll get to observe how URLs are put together for many different types of information.

The current IETF URL spec is here; more information on URLs can be found here.

marca@ncsa.uiuc.edu
Appendix C:
Composing Good HTML
James Tilton (http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimer.html)

Note: This document is available as both a single document (suitable for printing) and a multi-part document (more appropriate to hypertext). There is also a postscript version available via FTP, at jupiter.willamette.edu, as /outgoing/jtilton/strict-html.ps. These multiple views are automatically generated with a Perl script called "multiview".

The current edition of this document is available online at http://www.willamette.edu/html-composition/strict-html.html.

Introduction

As the Web continues to explode in its own inimitable fashion, it is becoming more and more important to write HTML that conforms to certain guidelines. Specifically, with the current diversity of clients for the Web (and we can only expect to see more!), it’s become important to write HTML that will look good on any client, and not just on the specific client which the author may have access to.

To that end, there are a few solutions. One approach is this one -- documents which point out common errors one might make in the composition of HTML. The other approach is software based -- a "lint"-like program for catching semantic errors in HTML, and perhaps even correcting them.

The thing to bear in mind is that, if you follow these guidelines, your document may not look as best as it possibly can on a particular browser. However, it also will not look ugly on any browser, which is the risk you take by disregarding these recommendations and tweaking your HTML for, say, Mosaic. Unfortunately, Mosaic may render things differently from Lynx which may render things differently from TkWWW, etc, etc, etc. These guidelines, in essence, should ensure the best fit across the space of all possible browsers, if you get my drift.

This document does not purport to be a style guide, or a beginner’s manual to HTML. Fine documents already exists for these purposes.

(Note: This document is fairly stable, but still open to amendment. Please feel free to comment on that which is missing, wrong, right, or silly. Especially, please point out anywhere that I don’t follow my own guidelines -- I’ll slink back and fix it, I promise! Thanks to everyone who’s already done so!)

Contents of this Document

• Introduction
• Contents of This Document (Douglas R. Hofstadter, Please...)
Good Practices

Things contained in this section are good practices for the generation of any HTML document. Specifically, this would include anything which should routinely be done in the creation of documents for the benefit of both reader and author.

Signing Documents, and Time-Stamps

It is a good idea to sign and date all documents served on the Web, so that people viewing the documents can form some impression of the authority of the document (i.e. how recent it is, and how reliable the information provider is). For example, this document has been signed.

Also, when dating a document, try to avoid ambiguous formats. For example, both the month/day/year and day/month/year format are used on the web -- so is "4/2/94" April 2 or February 4? A solution to this is to use the name of the month (or an abbreviation).

Finally, the best way to sign a document is to include, in your HEAD element -- include a LINK of type "made" in the following fashion:

\[ <\text{LINK REV="made" HREF="mailto:author@some.site.org">} \]

For an example, look at the HTML source of this document.
Common Errors

This section details common errors in HTML composition, that may lead to documents which are not fully device-independent. The behaviors of these errors are undefined, so certain browsers may render them as intended but not all browsers are guaranteed of doing so. Therefore, these mistakes should be avoided, even if your browser of choice renders your documents correctly.

Contents

- Paragraph Break Errors
- Character and Entity Reference Errors
- URL Errors
- Directory Reference Errors
- Not Using Fully Qualified Domain Names
- Improper Use of Relative URLs
- Missing Quotes in Start Tags
- Missed End Tags

Paragraph Break Errors

This is probably the most prevalent kind of error, and is the number one culprit in cases of ugly HTML rendering. If you fix nothing else, fix these! Perhaps the biggest misconception about the <P> element is that it signals an end-of-paragraph, rather than a paragraph break. According to the specification, "<P> is used between two pieces of text which otherwise would be flowed together".

In most cases this is not important -- functionally, the <P> serves as an end-of-paragraph marker. However, in certain contexts, use of <P> should be avoided, such as directly before or after any other element which already implies a paragraph break. To wit, the <P> element should not be placed either before or after the headings, HR (can I get a ruling on this? people don’t handle HR consistently... X Mosaic has no white space before or after, and Lynx appears to put white space after), ADDRESS, BLOCKQUOTE, or PRE.

It should also not be placed immediately before or after a list element of any stripe. That is, a <P> should not be used to mark the end-of-text for <LI>, <DT> or <DD>. These elements already imply paragraph breaks.

Caveats

Some clarifications on the above might be in order. One is the difficulties of rendering appropriate white space by a browser. While it is true that all of the entities mentioned above imply a paragraph break, this only occasionally means that they also
imply white space between sections -- this depends on the browser. So, while you might feel inclined to add a <P> in order to fix white space problems, please think twice and avoid it if you can.

Also, when using the glossary list (DL), please try to avoid using multiple DD's (definitions of terms) in order to provide multiple entries for a term (DT). Instead, use a <P> marker between paragraphs in a definition. The use of a DD (definition) without a matching DT (term) is illegal, although a DT without a DD can be used without dire consequences.

All clear now?

**Character and Entity Reference Errors**

Simply put, a character reference and an entity reference are ways to represent information that might otherwise be interpreted as a markup tag. For instance, in order to represent <P> in this text, I had to use &lt;P&gt; in my raw HTML. There are currently five entities for this purpose in HTML, as well as several entities which allow encoding of the ISO Latin-1 Character Set.

The most common error in the use of references is to leave off the trailing semicolon.
Also, no additional spaces are needed before or after the entity/character reference.

**URL Errors**

Another misunderstood aspect of HTML is in the composition of URL’s.

**Directory Reference Errors**

One grey area involves references to directories. It is possible to request an index of a directory from an HTTP server. The typical response from the server is to either return a pregenerated index document (which is often the document "index.html" in the referenced directory), or to construct an HTML document on the fly which contains a listing of all files in the directory. However, when making such a directory reference, it is important to make sure to have a trailing slash on the URL. That is, if you were to request the index of the directory which this document resides in, you would want to refer to it as http://www.willamette.edu/html-composition/, not as http://www.willamette.edu/html-composition.

Some servers are able to catch these errors, and provide redirection to the proper URL, but it’s best to get the URL right in the first place -- notably because not all browsers support transparent redirection.

**Not Using Fully Qualified Domain Names**

Problems can arise when the hostnames in URLs aren’t fully qualified. In local networks, you can usually refer to your own machines simply by their names -- for instance, here
at Willamette we refer to our local WWW server as "www". However, the server's FQDN (fully qualified domain name) is "www.willamette.edu". The FQDN provides enough information that any host, anywhere on the Internet, can find this particular machine. (It's like trying to find all the Vermeers in New York :).

What happens is that an HTML might construct a link that looks like this:

```html
<A HREF="http://www/~jtilton/metanoia/">Metanoia -- A Change In Spirit</A>
```

which produces a link to Metanoia -- A Change In Spirit that will only work for people in the local network that that machine is on. A correct link would look like this, instead:

```html
<A HREF="http://www.willamette.edu/~jtilton/metanoia/">Metanoia</A>
```

which would allow all of you who are interested in Metanoia to actually follow the link.

This leads almost directly into:

**Improper Use of Relative URLs**

Finally, a brief section on relative URLs. It is possible to construct a "relative" URL, which gives you the following advantages:

- It's shorter.
- It makes a collection of documents which are linked together more portable (easier to move from directory to directory, or server to server).

However, relative URLs can also break things.

A relative URL is a URL which doesn't contain all the necessary parts of a "full" URL (scheme, host, path information). There's a large number of things which might fit this description! The browser will try to assume the parts that have been "left out" by using the information from the URL of the document which contains the link. However, not all browsers will make these assumptions in the same way. Here's a short list of what's "safe" and "unsafe" (based on experience, and not on a specification anywhere -- unfortunately).

**Safe:** Same directory relative URL’s

A reference to a document in the same logical directory (such as `<A HREF="strict-html-gp.html">Good Practices</A>`) is safe. This kind of reference, roughly speaking, contains no "/"s.

**Safe:** Same server relative URL’s

A reference to a document in the same server (such as `<A HREF="/~jtilton/">Eric’s Hyplan</A>`) is also safe. This kind of reference, roughly speaking, will begin with a "/". (It will also be semi-absolute, in that it starts at the top of that server’s directory structure...)

**Unclear:** Most other kinds of relative URLs
References such as <A HREF="~jtilton/euphonium.html"></A> can be dangerous -- sometimes browsers will interpret that as meaning "go up one directory level, find the directory '~jtilton', and then find 'euphonium.html' in it." And sometimes they won’t. Currently, I don’t understand this problem well enough to speak about it. I will try and get a canonical answer Real Soon Now, now that the semester has finally ended.

Unsafe: "file://localhost/..."

It’s also possible to have a reference to "file://localhost/some/file/pathname". What this does is references the file described on the local host of whoever is browsing the document. Which is why a reference to <A HREF="file://localhost/etc/motd"></A> will display the message of the day on your machine, not the message of the day on my machine. Unless you know what you are doing, these references will really mess up your documents.

(This sub-section isn’t written very well, I fear. If anyone has any better copy, I’ll gladly put it here instead. -et 4/7/94)

**Missing Quotes in Start Tags**

One common error that I used to make all the time (I use Marc Andreesen’s html-mode.el for Emacs these days -- I had to learn Emacs, but now it’s so much easier to write HTML!) was to leave off a quote in my start tags. For example, this reference to the euphonium, king of instruments should look like:

<A HREF="http://www.willamette.edu/~jtilton/euphonium.html">

but I would often use

<A HREF="http://www.willamette.edu/~jtilton/euphonium.html>

instead. I suppose by the end of that huge URL, I’d forgotten it was supposed to be quoted. The behaviour of browsers upon encountering this varies -- some display a proper link, but you can’t follow it, while others actually eat up huge portions of the following text, thinking it to be part of the URL.

**Missed End Tags**

Many of the HTML elements contain information within them. For example, <em>emphasized text</em> would be rendered as *emphasized text*. There is a start tag ( <EM> ), some content (which may include text, and in some cases, other nested elements), and an end tag ( </EM> , indicated by the </>. A common mistake is to miss the / in the end tag. All elements (except empty elements, see next paragraph) must be terminated by an end tag -- otherwise, undefined behavior may occur.

Some HTML elements may be empty, such as <P> and <HR> (CERN provides an extended discussion on element content). If this is the case, there is no need for an end
Things to Avoid

This section concentrates on mistakes in HTML authoring that are more problems of aesthetics then problems of device-independence.

Contents

- Mixing HEAD and BODY Elements
- Using White Space Around Element Tags
- Heading Usage
- Meaningless Link Text
- Physical vs. Logical Character Emphasis

Mixing HEAD and BODY Elements

The section on HTML elements in the HTML specification indicates that HTML documents should not mix those elements which belong in the HEAD of a document with those which belong in the BODY. The urgency of this suggestion is unclear, but it does make a certain amount of common sense for readability of HTML code, and for conformance with possible future browsers which may not support the mixing of these elements. Essentially, it lacks serious style points >=).

Using White Space Around Element Tags

In general, the use of white space around element tags should be avoided. If white space immediately follows a start tag, for example, the style changes implied by that element may be applied to the initial space, as well. For instance, <A HREF="http://www.willamette.edu/~jtilton/" CZeCh THIZ 0uT </A> would be rendered as CZeCh THIZ 0uT . On some browsers, there may be white space around the anchor, which adds unwanted unsightliness to the rendering, and may lessen the impact of the document. (This comment really applies to white space immediately following start tags, and immediately preceding end tags).

Heading Usage

The HTML specification points out that a heading should not be more then one level below the heading which preceded it. That is, <H3> should not follow <H1>, etc.

Also, it is pointed out that "a heading element implies all the font changes, paragraph breaks before and after, and white space (for example) necessary to render the heading". Extra highlighting elements are discouraged, therefore.
Meaningless Link Text

When creating documents, make sure that your links are meaningful -- that is, that they avoid online-specific references, and that they don't detract from readability. The text of your links should flow well in the context of the rest of your text (especially avoid the click here syndrome!) , and your text should also be able to stand alone as a printable document.

In other words, avoid using sentences like, "You can find out more information about cows by clicking here". (This is also bad because it refers to "clicking", which assumes that everyone is using a mouse with their browser!) A much better alternative is "More information about cows is available."

Physical vs. Logical Character Emphasis

Since HTML (and also SGML) is designed to be a device independent language for describing the rendering of documents, most of the elements within it aren’t intended to give direct control to the author over how the final page layout will look. The major exceptions to this are in the character highlighting elements.

There are two types of character highlighting elements -- physical and logical. The physical styles involve things like "italic font", "boldface", etc; while the logical styles are things like "emphasis", "citation", "strong", etc. It is strongly recommended that you employ the logical styles rather then the physical styles in your documents. Using <I></I> to render text in italics will only be effective on those browsers which are capable of displaying italics -- which all browsers are not guaranteed to do. It is far better to encode semantic content -- to describe things in terms of logical styles -- and then allow the browser to display that semantic structure as best it can, given its display capabilities.

So, instead of

<\/italics>

you might use <EM>emphasized</EM> , or a <CITE>citation</CITE> , and instead of

<B>bold</B>

you might use <STRONG>strong</STRONG> .

This also leaves the possibilities open in the future for more sophisticated uses of these semantic renderings, which have much more inherent meaning then font styles like bold or italic.

(Unfortunately, the jury is still out to lunch on this one. One argument against logical character styles is that it turns out to a bottomless pit, attempting to define logical styles for every possibility. Physical styles, combined with the context of the text in which they are placed, seem to provide a much richer set without a huge number of tags. Oh, well. Use logical styles when you can, though.)
Deprecated and Obsolete Elements

This section lists elements of HTML whose use should be avoided, whether because the element is now obsolete, or because the element is being deprecated (i.e. still supported, but its use is not recommended and the element may eventually become obsolete).

Contents

• Obsolete Elements
• Deprecated Elements (under const.)

Obsolete Elements

Several elements of HTML are obsolete, including PLAINTEXT, XMP, LISTING, HPx, and COMMENT. The first three should be replaced with PRE; HP (highlighted phrase) should be replaced with the character highlighting elements; and COMMENT should be replaced with <!-- blah blah blah -->, the SGML comment characters.

Deprecated Elements

(Give me some time to fill this in. Like until who knows when? >=)

For More Information

There already exist documents on the Web which address this same topic, and perhaps in more detail. For definitive reference information you may wish to check the HTML specification from CERN. For a more detailed discussion of HTML composition style, you should also check the Style Guide (especially the section on device-independent formatting), which is also from CERN.

If you’re looking for a good document for learning the basics of HTML, you will want to check out the Beginner’s Guide to HTML, from NCSA.

Acknowledgements

I’d like to thank all of you have visited this document and commented on it, suggesting fixes, clarification, and even new sections. You know who you are (even if I managed to lose your addresses in the flood of information)! It is, in some senses, still a work in progress and is always amenable to suggestion, modification, and repair. I appreciate your help!

James "Eric" Tilton, HTML Guru Wannabee, jtilton@willamette.edu
Appendix D:
NCSA HTML Style Sheet
(http://www.ncsa.uiuc.edu/Pubs/StyleSheet/NCSAStyleSheet.html)

Purpose

Insure some measure of consistency across NCSA’s online documents and to simplify
document maintenance as much as possible.

Formatting Styles

Taglines

These are the things at the bottom of the HTML documents. The generic format is

```
<p>
______________________________________________________________
<ADDRESS>
National Center for Supercomputing Applications / pubs@ncsa.uiuc.edu
</ADDRESS>

(Replace “pubs@ncsa.uiuc.edu” with the appropriate e-mail contact.) The line consists
of 75 underscores (_). Be sure include the <P> paragraph marker before the line. Note
that there is a space before and after the ’/’.

For documents related to NCSA groups, the tagline is modified to include the group’s
name, e.g.

```

```
<ADDRESS>
Software Development Group / NCSA / softdev@ncsa.uiuc.edu
</ADDRESS>

For a publication (e.g., an issue of access, or an SDG manual), the tagline should
consist of the document title, the date of publication (if applicable), NCSA and an e-mail
contact:

```
```
The e-mail address used in the tagline should be generic, e.g. continue to exist long after you’ve left NCSA. Do not use your own e-mail address unless a) you want all the mail about the document sent to you directly and b) you plan to be at NCSA for some time.

**Headings**

For H1 and H2 headings, capitalize the first letter of each "important" word. (i.e. all words except prepositions and conjunctions of three or fewer letters). For example:

For H3-H4 headings, capitalize the first letter of the first word only.

**A Beginner’s Guide to HTML**

**Anchors**

In general, link the name of the document, and not a generic “here”, i.e.

“For more information see A Beginner’s Guide to HTML”

is preferred to

“For a beginner’s guide to HTML, click here.”

Why: *The eye quickly focuses on the highlighted text. If you highlight the actual document names, the user can quickly find the relevant link. If the only word highlighted is “here”, the the person has to backtrack from the word and read the entire sentence, which is slower.*

If you have a series of references, make it an unordered list:

“For more information, see

• A Beginner’s guide to HTML
• URL primer
• The HTML DTD”

Don’t leave a space between the HTML codes and the surrounding text:

```html
<A NAME = anchornamethe linked text</A>
^                ^ Don’t put a space in either of these places.
```

**Quotes**

When producing documents on Macintoshes, don’t use the curly quotes.
Don’t use typewriter double quote character (") unless you’re referring to inches (2”). Instead, use two back quotes (’) for an opening quote and two apostrophes (’) for a closing quote.

Why: *The typewriter double quote(') are just straight vertical lines.*

“Yo! I’m talking to you.”

*Looks better than*

"Yo! I’m talking to you."

**First vs. third person**

In general, write in third person. “We” is often ambiguous. Are you referring to yourself in a royal sense, your group, NCSA as a whole or even the entire University of Illinois? Be specific.

**Exclamation points**

Keep exclamation points to a minimum. That is, don’t use them to try to add excitement to an otherwise boring sentence. The sentence will still be boring, and now you’re shouting too.

**Source style**

**General**

Fit everything within 80 columns, if possible. (Sometimes this isn’t possible because of long anchor names.)

Insert a blank line between paragraphs.

**Lists**

The formatting of the HTML document should match the logical structure of the lists, i.e. if you have nested lists, nest the HTML source as well.

```html
<UL>
  <LI>First item
  <LI>Second item
    <UL>
      <LI>First subitem
      <LI>Second subitem
    </UL>
  <LI> Third item
</UL>
```
File names

Choose one of three conventions:

- Capitalize the first letter of each word (and all acronyms); no space between words, e.g. “NCSAHomPage.html”.
- All lowercase, separated by hyphens, e.g. “ncsa-home-page.html”.
- All lowercase, separated by underscores, e.g. “ncsa_home_page.html”.

The first two are preferred. Please do not use other variations or mix the conventions. **NO** “NCSA_Home_Page.html” or “NCSA-Home-Page.html”.

Use the same convention for all files in a given directory.

Use the first convention for names of directories.

Always include the file type extension (.ps, .html, .gif, etc.) as part of the file name. File name should be adequately descriptive by itself. Example: “MacTelnetContents.html” instead of “Contents.html”.

Miscellany

Copyrighted images

Do not put copyrighted images on the server unless you have received permission from the copyright holder. This includes magazine articles, animations, and images scanned from copyrighted publications.

Menus

Unlike printed works, put the explanatory text after a menu. For instance, on the NCSA Home Page, the one-sentence description of NCSA goes after the list of menu items. This way users who already know what NCSA is don’t have to stare at the same sentence every time they visit the home page. It also allows access to more items without scrolling.