Website Development

Basic HTML

- hypertext
- tags & elements
- text formatting
- hyperlinks
- images
- tables
- frames

Hypertext & HTML

HyperText Markup Language (HTML) is the language for specifying the *static* content of Web pages

- *hypertext* refers to the fact that Web pages are more than just text

  can contain multimedia, provide links for jumping within & without

- *markup* refers to the fact that it works by augmenting text with special symbols (tags) that identify structure and content type

Web development tools

many high-level tools exist for creating Web pages

e.g., Microsoft FrontPage, Netscape Composer, Adobe PageMill,
Macromedia DreamWeaver, HotDog, …

also, many applications have "save to HTML" options (e.g., Word)

for most users who want to develop basic, static Web pages, these are fine

assembly language vs. high-level language analogy

so, why are we learning low-level HTML using a basic text editor?

- may want low-level control
- may care about size/readability of pages
- may want to "steal" page components and integrate into existing pages
- may want dynamic features such as scripts or applets

Tags vs. elements

HTML specifies a set of *tags* that identify structure and content type

- tags are enclosed in < >

  `<img src="image.gif" />` specifies an image

- most tags come in pairs, marking a beginning and ending

  `<title>` and `</title>` enclose the title of a page

an HTML *element* is an object enclosed by a pair of tags

`<title>My Home Page</title>` is a TITLE element

`<b>This text appears bold.</b>` is a BOLD element

`<p>Part of this text is <b>bold</b>.</p>`
is a PARAGRAPH element that contains a BOLD element

HTML document is a collection of elements (text/media with context)

Structural elements

an HTML document has two main structural elements

- HEAD contains setup information for the browser & the Web page
  e.g., the title for the browser window, style definitions, JavaScript code, …
- BODY contains the actual content to be displayed in the Web page

```
<html>
<!-- page01.html     -->
<!-- Demo web page   -->

<head>
  <title>Title for Page</title>
</head>

<body>
  Text that appears in the page
</body>

</html>
```

view page in browser

HTML documents begin and end with <html> and </html> tags

Comments appear between `<!--` and `-->`

HEAD section enclosed between `<head>` and `</head>`

BODY section enclosed between `<body>` and `</body>`

Text layout

```
<html>
<!-- page02.html     -->
<!-- Demo web page   -->

<head>
  <title>Title for Page</title>
</head>

<body>
  This is a whole lot of text that
  goes on   and   on   and
  on
  and

  on
  .
  .
  .
</body>

</html>
```

view page in browser

the BODY can contain multiple lines of text

- text layout and spacing is pretty much ignored by the browser
- every sequence of whitespace is interpreted as a single space
- browser automatically wraps the text to fit the window size

➔ can layout text in an HTML document for readability, will not affect how it is viewed

Overriding default layouts

```
<html>
<!-- page03.html     -->
<!-- Demo web page   -->

<head>
  <title>Title for Page</title>
</HEAD>

<body>
  <p>
  This is a paragraph of text<br/>
  made up of two lines.
  </p>

  <p>
  This is another paragraph with a
  &nbsp; GAP &nbsp; between
  some of the words.
  </p>

  <p>
  &nbsp;&nbsp; This paragraph is<br/>
  indented on the first line<br/>
  but not on subsequent lines.
  </p>
</body>

</html>
```

for the most part, layout of the text must be left to the browser

WHY?

can override some text layout

- can cause a line break using the `<br/>` tag (no closing tag)
- can specify a new paragraph (starts on a new line, preceded by a blank line) using `<p>`…`</p>`
- can force a space character using the symbol for a non-breaking space: `&nbsp;`

view page in browser

Separating blocks of text

```
<html>
<!-- page04.html     -->
<!-- Demo web page   -->

<head>
  <title>Title for Page</title>
</head>

<body>
  <h1>Major heading 1</h1>
  <p>
  Here is some text.
  </p>

  <h2>Subheading</h2>
  <p>
  Here is some subtext.
  </p>

  <hr/>

  <h1>Major heading 2</h1>
  <p>
  Here is some more text.
  </p>
</body>

</html>
```

can specify headings for paragraphs or blocks of text

- `<h1>…</h1>` tags produce a large, bold heading
- `<h2>…</h2>` tags produce a slightly smaller heading

  . . .
- `<h6>…</h6>` tags produce a tiny heading

can insert a horizontal rule to divide sections

- `<hr/>` draws line across window
- `<hr width="50%" />` sets width
- `<hr size=10 />` sets thickness

view page in browser

Aligning text

```
<html>
<!-- page05.html   -->
<!-- Demo web page -->

<head>
  <title>Title for Page</title>
</head>

<body>
  <h1 style="text-align:center">Centered Heading</h1>
  <p>
  Here is some left-justified text
  (which is the default in HTML).
  </p>

  <p style="text-align:center">
  Here is some centered text.
  </p>

  <div style="text-align:right">
  <h2>Right-justified Heading</h2>
  <p>Here is some right-justified text.</p>
  </div>
</body>

</html>
```

can specify how elements should be aligned (default is left-justified)

- utilize STYLE attribute of tag

to justify more than one element as a group, use DIV tags

- ell elements enclosed in DIV are formatted similarly

view page in browser

Text styles

```
<html>
<!-- page06.html     -->
<!-- Demo web page   -->

<head>
  <title>Title for Page</title>
</head>

<body>
  <p>
  Text can be emphasized using
  <b>bold</b>, <i>italics</i>, or even
  <big>resizing</big>.  <br/>
  <u>Underlining</u> text is not
  generally recommended since it looks
  too much a like a hyperlink.  <br/>
  The typewriter font is good for
  displaying code:
  <small><tt>sum = sum + i;</tt></small>
  </p>
</body>

</html>
```

view page in browser

can specify styles for fonts

- `<b>`… `</b>` specify bold
- `<i>`… `</i>` specify italics
- `<u>`… `</u>` specify underlined
- `<tt>`… `</tt>` specify typewriter-like (fixed-width) font
- `<big>`… `</big>` increase the size of the font
- `<small>`… `</small>` decrease the size of the font

Note: if elements are nested, the order of opening/closing is important!
(must be LIFO)

More text styles

```
<html>
<!-- page07.html      -->
<!-- Demo web page    -->

<head>
  <title>Title for Page</title>
</head>

<body>
  <p>
  <span style="color:red">Subscripts</span>
  (e.g., x<sub>1</sub>) and
  <span style="color:blue">superscripts
  </span> (e.g., 2<sup>10</sup>)
  can be embedded directly in text.
  </p>

  <p>
  In order to avoid affecting line
  spacing, usually it should be made
  smaller (e.g.,
  <small>2<sup>10</sup></small>).
  </p>
</body>

</html>
```

- `<sub>… </sub>` specify a subscript
- `<sup>… </sup>` specify a superscript
- `<p style="color:red"> …</p>` for paragraphs
- `<span style="color:blue"> …</span>` for inline text

view page in browser

More text grouping

```
<html>
<!-- page08.html  -->

<head>
  <title>Title for Page</title>
</head>

<body>
  <p>
  <tt><pre>
    for (i = 0; i < 10; i++) {
        sum = sum + i;
    }
  </pre></tt>
  </p>

  <p>
  Eagleson's Law states that:
  <blockquote>
  Any code of your own that you haven't
  looked at for six or more months
  might as well have been written by
  someone else.
  </blockquote>
  </p>
</body>

</html>
```

- `<pre>…</pre>` specify text that is to be displayed as is (line breaks and spacing are preserved)

  useful for code or whenever you want text to fit a specific layout

- `<blockquote>…</blockquote>` specify text that is to be indented on both margins

  useful for quotations or for indenting text in subsections

view page in browser

Lists

```
<html>
<!-- page09.html   -->

<head>
  <title>Title for Page</title>
</head>

<body>
  <p>
  <ol>
    <li>First thing.
    <li>Second thing.
    <li>Third thing.
  </ol>
  </p>

  <p>
  <dl>
    <dt>HTML
    <dd>HyperText Markup Language
    <dt>HTTP
    <dd>HyperText Transfer Protocol
  </dl>
  </p>
</body>

</html>
```

there are 3 different types of list elements

- `<ol>…</ol>` specifies an ordered list (using numbers or letters to label each list item)
  `<li>` identifies each list item

  can set type of ordering, start index

- `<ul>…</ul>` specifies unordered list (using a bullet for each)
  `<li>` identifies each list item

- `<dl>…</dl>` specifies a definition list
  `<dt>` identifies each term
  `<dd>` identifies its definition

view page in browser

Hyperlinks

```
<html>
<!-- page10.html  -->

<head>
  <title>Title for Page</title>
</head>

<body>
  <p>
  <a href="http://www.csuhayward.edu">
  Cal State Hayward</a>
  <br>
  <a href="page09.html" target="_blank">
  Open page09 in a new window</a>
  </p>
</body>

</html>
```

view page in browser

perhaps the most important HTML element is the hyperlink, or ANCHOR

- `<a href="URL">…</a>`

  where URL is the Web address of the page to be displayed when the user clicks on the link

  if the page is accessed over the Web, must start with `http://`

  if not there, the browser will assume it is the name of a local file

- `<a href="URL" target="_blank">…</a>`

  causes the page to be loaded in a new window

Hyperlinks (cont.)

```
<html>
<!-- page11.html  -->

<head>
  <title>Title for Page</title>
</head>

<body>
  <p align="center">
  [ <a href="#HTML">HTML</a> |
    <a href="#HTTP">HTTP</a> |
    <a href="#IP">IP</a> |
    <a href="#TCP">TCP</a> ]
  </p>
  <p>
  Computer acronyms:
  <dl>
    <a name="HTML"></a><dt>HTML
    <dd>HyperText Markup Language
    <a name="HTTP"></a><dt>HTTP
    <dd>HyperText Transfer Protocol
    <a name="IP"></a><dt>IP
    <dd>Internet Protocol
    <a name="TCP"></a><dt>TCP
    <dd>Transfer Control Protocol
  </p>
</body>
</html>
```

for long documents, you can even have links to other locations in that document

- `<a name="`*`ident`*`">…</a>`

  where ident is a variable for identifying this location

- `<a href="#`*`ident`*`">…</a>`

  will then jump to that location within the file

- `<a href="URL#`*`ident`*`">…</a>`

  can jump into the middle of another file just as easily

view page in browser

Images

can include images using `IMG`

- by default, browsers can display GIF and JPEG files
- other image formats may require plug-in applications for display

```
<img src="filename" alt="alternate text" />
```

again, if file is to be accessed over the Web, must start with http:// (if not, will assume local file)

```
<html>
<!-- page12.html -->

<head>
  <title>Title for Page</title>
</head>

<body>
  <div style="text-align:center"
  <p><img border="0" src="heckerb.jpg" width="362"
height="539"></p>
  <p>Barbara Hecker</p>
  </div>
</body>
</html>
```

view page in browser

Tables

tables are common tools for arranging complex layout on a Web page

- a table divides contents into rows and columns
- by default, column entries are left-justified, so provide for alignment

```
<html>
<!-- page13.html  -->

<head>
  <title>Title for Page</title>
</head>

<body>
  <table>
    <tr>
      <td>foo</td> <td>bar</td>
    </tr>
    <tr>
      <td>bizbaz</td> <td>booboo</td>
    </tr>
  </table>
</body>
</html>
```

`<table>…</table>` specify a table element

`<tr>…</tr>` specify a row in the table

`<td>…</td>` specify table data (i.e., each column entry in the table)

view page in browser

Layout in a table

```
<html>
<!-- page14.html  -->

<head>
  <title>Title for Page</title>
</head>

<body>
  <table border=1>
    <tr align="center">
      <td>foo<br>foo</td>
      <td valign="top">bar</td>
    </tr>
    <tr>
      <td>bizbaz</td>
      <td>booboo</td>
    </tr>
  </table>
</body>
</html>
```

view page in browser

can have a border on tables using the BORDER attribute

```
<table border=1>
```

increasing the number makes the border thicker

can control the horizontal & vertical layout within cells

```
<td align="center">
<td align="right">

<td valign="top">
<td valign="bottom">
```

can apply layout to an entire row

```
<tr align="center">
<tr valign="top">
```

Table width

```
<html>
<!-- page15.html  -->

<head>
  <title>Title for Page</title>
</head>

<body>
  <table width="100%">
    <tr>
      <td>left-most
      <td align="right">right-most</td>
    </tr>
</table>
</body>
</html>
```

view page in browser

by default, the table is sized to fit the data

can override & specify the width of a table relative to the page

```
<table width="60%">
```

useful for page footer –
- *set table width to 100%*
- *1st column: left-justified*
- *2nd column: right-justified*

Other table options

```
<html>
<!-- page16.html  -->

<head>
  <title>Title for Page</title>
</head>

<body>
  <table border=1
          cellspacing=4 cellpadding=8>
    <tr>
      <th>HEAD1</th> <th>HEAD2</th> <th>HEAD3</th>
    </tr>
    <tr>
      <td>one</td> <td>two</td> <td>three</td>
    </tr>
    <tr>
      <td rowspan=2 align="center"> four </td>
      <td colspan=2 align="center"> five </td>
    </tr>
    <tr>
      <td> six </td> <td> seven </td>
    </tr>
</table>
</body>
</html>
```

can control the space between cells & margins within cells

`<table cellspacing=5>`

`<table cellpadding=5>`

can add headings

`<th>` is similar to `<td>` but displays heading centered in bold

can have data that spans more than one column

`<td colspan=2>`

similarly, can span more than one row

`<td rowspan=2>`

view page in browser

Frames

frames provide the ability to split the screen into independent pages

- must define a FRAMESET that specifies the layout of the pages
- actual pages to be displayed must be in separate files

```
<html>
<!-- page17.html -->

<frameset cols="*,*">
  <frame src="page01.html">
  <frame src="page02.html">
</frameset>

</html>
```

view page in browser

can divide vertically

```
<frameset cols="50%,50%">
```

or, can divide horizontally

```
<frameset rows="30%,*,*">
```

** causes the browser to divide the remaining space evenly among pages*

Frame defaults

by default, each frame is an independent page, scrollable

- the relative size of the frames can be changed by dragging the border in between

```
<html>
<!-- page18.html  -->

<frameset rows="35%,*">
   <frameset cols="*,*" frameborder=0 >
    <frame src="page01.html">
    <frame src="page02.html">
  </frameset>
  <frame src="page03.html">
</frameset>

</html>
```

view page in browser

can specify whether you want a border

`frameborder=1`	*(default)*
`frameborder=0`	*no border*

can even nest frames

Frame controversy

frames are probably the most controversial HTML feature

- some people love them, some people hate them

2 reasonable uses for frames

- as a navigational aid:

  can divide the screen into a static menu frame and the main frame for navigating a site

- as a means of separating program input from output:

  can divide the screen into a static man input form frame and the main frame for displaying output

Menu frame

to create a menu, need to be able to direct links to the main frame

- name the frames in the FRAMESET
- specify the frame name as TARGET in the link
- specify `_top` as target to return to top level of browser

```
<html>
<!-- page19.html -->

<head>
  <title>Demo Browser</title>
</head>

<frameset cols="30%,*">
    <frame src="menu19.html" name="menu">
    <frame src="page01.html" name="main">
</frameset>

</html>
```

view page in browser

```
<html>
<!-- menu19.html -->

<head>
  <title>Menu of Demos</title>
</head>

<body>
Links to demo pages

<p>
<a href="page01.html"
   target="main">Demo 1</a><br/>
<a href="page02.html"
   target ="main">Demo 2</a><br/>
<a href="page03.html"
   target ="main"> Demo 3</a><br/>
<a href="page04.html"
   target ="main"> Demo 4</a><br/>
<a href="page05.html"
   target ="main"> Demo 5</a><br/>
<a href="page06.html"
   target ="main"> Demo 6</a><br/>
<a href="http://www.creighton.edu"
   target="_top" >Creighton</a>
</p>
</body>
</html>
```

Web rules of thumb

HTML provides for lots of neat features,
but just because you can add a feature doesn't mean you should!

don't add features that distract from the content of the page

- use color & fonts sparingly and be careful how elements fit together
  e.g, no purple text on a pink background, no weird fonts
- use images only where appropriate
  e.g., bright background images can make text hard to read
  e.g., the use of clickable images instead of buttons or links can slow access
- don't rely on window or font size for layout
  e.g., font size may be adjusted by viewer, window constrained
- don’t be annoying
  e.g., no pop-up windows, excessive advertising, silly music
- break large document into smaller or provide a menu (either internal or frame)
- stick to standard features and test using both IE and Netscape

End of Lecture