WAD – Lecture 2 - HTML

Basic HTML

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

Hypertext & HTML

- HyperText Markup Language (HTML) is the language for specifying the *static* content of Web pages (based on SGML, the Standard Generalized Markup Language)
  - *hypertext* refers to the fact that Web pages are more than just text
    can contain multimedia, provide links for jumping within the same document & to other documents
  - *markup* refers to the fact that it works by augmenting text with special symbols (tags) that identify the document structure and content type

Hypertext & HTML (cont.)

- HTML is an evolving standard (as new technology/tools are added)
  - HTML 1 (Berners-Lee, 1989): very basic, limited integration of multimedia
    in 1993, Mosaic added many new features (e.g., integrated images)
  - HTML 2.0 (IETF, 1994): tried to standardize these & other features, but late
    in 1994-96, Netscape & IE added many new, divergent features
  - HTML 3.2 (W3C, 1996): attempted to unify into a single standard
    but didn't address newer technologies like Java applets & streaming video
  - HTML 4.0 (W3C, 1997): current standard (but moving towards XHTML)
    attempted to map out future directions for HTML, not just react to vendors
  - XHTML 1.0 (W3C, 2000): HTML 4.01 modified to conform to XML standards
  - XHTML 1.1 (W3C, 2001): “Modularization” of XHTML 1.0
  - HTML 5 (Web Hypertext Application Technology Working Group, W3C, 2006): New version of HTML4, XHTML 1.0, and DOM 2 (still a work in progress), no longer based on SGML, but “backward compatible” with parsing of older versions of HTML. HTML 5 is referred to as a "living language".

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)

  Don't use these tools!!

  for most users who want to develop basic, static Web pages, these are fine (but many of these programs produce very poorly structured HTML code)

- 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 "borrow" components from other pages and integrate into existing pages
  - may want dynamic features such as scripts or applets
  - remote editing of web pages may only be possible using a basic text editor
  - **sticking to (internationally and industrially) agreed upon standards will help ensure your web documents are rendered as you intend them to look and operate as you desire**

Tags and Elements

- HTML specifies a set of *tags* that identify structure of the document and the 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 (in most cases) 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

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

Structural Elements

- a standard 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>
 <!-- Version information --
 -- File: page01.html --
 -- Author: RC --
 -- Creation: 22.09.14 --
 -- Description: introductory page --
 -- Copyright: UPT --
 -->
 <head>
  <title>My first HTML document</title>
 </head>
  <body>
    <p> Hello world! </p>
  </body>
</html>
```

view page

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

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

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

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

** Find more info on HTML docs!*

<head> and <body> elements

- The <head> element is where you include a <title> element (that appears in the title bar of the browser).
- You can also include lots of other type of information in the <head> element.
  - Cascading Style sheet information, or a link to an external style sheet (or several)
  - "Meta" data, such as who authored the page, the type of content, and clues that search engines may (or may not) use to help categorize your page
  - JavaScript code

- The <body> element contains the main bulk of the material to be displayed on the webpage.
  - Paragraphs
  - Tables and lists
  - Images
  - JavaScript code
  - PHP code can be included here too (if passed through a PHP parser before being served to the client's browser)
  - Other embedded objects (videos, etc)

Text Layout

```
<html>
<!-- RC page02.html 30.08.14 -->
<head>
  <title>Text Layout</title>
</head>

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

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

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

</html>
```

view page

for the most part, layout of the text is left to the browser

- (almost) every sequence of whitespace is interpreted as a single space
- browser automatically wraps the text to fit the window size

can override some text layout

- can specify a new paragraph (starts on a new line, preceded by a blank line) using `<p>...</p>`
- can cause a line break using the `<br/>` tag ("self-closing" tag)
- can force a space character using the symbol for a "non-breaking space": ` `

Separating Blocks of Text

```
<html>
<!-- RC page03.html  15/08/14 -->
<head>
  <title>Blocks of Text</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>
```

view page

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/> (or use <hr>) draws line across window

The Basic Web page – A Worked Example

```
<html>
<!-- RC page22.html  15.08.14 -->
   <head>
      <title> Bill Smiggins Inc. </title>
   </head>
   <body>
      <h1>Bill Smiggins Inc.</h1>
      <h2>About our Company...</h2>
      <p>This Web site provides clients, customers,
         interested parties and our staff with all of
         the information that they could want on
         our products, services, success and failures.
      </p>
      <hr/>
      <h3> Products </h3>
      <p> We are probably the largest
      supplier of custom widgets, thingummybobs, and bits
      and pieces in North America. </p>
      <hr/>
   </body>
</html>
```

view page

Text Appearance

```
<html>
 <!-- RC page25.html  15.08.14 -->
 <head>
  <title>Text Variations and Escape
Sequences</title>
 </head>
 <body>
  <h1>Text Variations</h1>
  <p>We can use <b>simple</b> tags to
     <i>change</i> the appearance of
     <strong>text</strong> within
     <tt>Web pages</tt>.
     Even super<sup>script</sup>
     and sub<sub>scripts</sub> are
      <em>supported</em>.</p>

   <h1>Text Escape Sequences</h1>
  <p>
    &amp; &lt; &gt; &quot; &copy;
  </p>
 <h1>Preformatted text</h1>
      <pre>
         University of Liverpool
         Department of Computer Science
         Ashton Building, Ashton Street
         Liverpool, L69 3BX, UK
      </pre>
 </body>
</html>
```

view page

can specify styles for fonts

- `<b>… </b>` specify bold
- `<i>… </i>` specify italics
- `<tt>… </tt>` specify typewriter-like (fixed-width) font

- `<small>… </small>` decreases the size of the font
- `<em>…</em>` puts emphasis
- `<strong>…</strong>` puts even more emphasis

- `<sub>… </sub>` specify a subscript
- `<sup>… </sup>` a superscript

- `<pre>…</pre>` include ready-formatted text

- & &al; > " © escape characters used in HTML control

- *Find more info on text tags!*

Lists

```
<html>
<!-- RC page07.html  23.09.14 -->
<head>  <title>(Sort of) Simple Lists</title>
  <style type="text/css">
  .my_li:before {  content: counter(list) ": ";
                   counter-increment: list;  }
  </style>  </head>
<body>

<ul style="list-style-type: square;">
 <li> ... first list item... </li>
 <li> ... second list item... ... </li>
</ul>
<dl>   <dt> Dweeb </dt>
   <dd> young excitable person who may
  mature into a <em>Nerd</em> </dd>
 <dt> Hacker </dt>
      <dd> a clever programmer </dd>
 <dt> Nerd </dt>  <dd> technically bright but
        socially inept person </dd>
</dl>
 <ol style="list-style-type: none;
            counter-reset: list 29;" >
  <li class="my_li">Makes first item number 30.</li>
  <li class="my_li">Next item continues to number
31.</li>
</ol>

</body>
</html>
```

view page

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

* We will learn more about the "style" attributes soon enough.

Hyperlinks

```
<html>
<!-- RC page08.html  15.08.14 -->

<head>
  <title>Hyperlinks</title>
</head>

<body>
  <p>
  <a href="http://www.cs.upt.ro">
  CS - UPT</a>
  <br/>
  <a href="page07.html" target="_blank">
   Open page07 in a new window</a>
  </p>
</body>

</html>
```

view page

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

** Find more info on attribute TARGET*

Hyperlinks (cont.)

```
<html>
<!-- RC page09.html  21.09.14 -->

<head>
  <title>Internal Links in a Page</title>
</head>

<body>
  <p>
  [ <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>
   <dt id="HTML">HTML</dt>
   <dd>HyperText Markup Language
   <dt id="HTTP">HTTP</dt>
   <dd>HyperText Transfer Protocol…</dd>
   <dt id="IP">IP</dt>
   <dd>Internet Protocol…</dd>
   <dt id="TCP">TCP</dt>
   <dd>Transfer Control Protocol…</dd>
  </dl>
  </p>

</body>
</html>
```

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

- `<xxxx id="ident">…</xxxx>`

  where ident is a variable for identifying this location, where "xxxx" can, in principle, be any HTML element

  (this is actually an HTML5 language specification, but seems to work in most browsers)

- `<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

Images

can include images using `img`

- by default, browsers can display GIF and JPEG files, more modern browsers can also typically support PNG files and SVG graphics (of course, use at your own risk)
- other image formats may require plug-in applications for display

```
<img src="URL (or filename)" height="n" width="n" alt="text"
title= "text" />
```

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

* Find more info on `<img />`

```
<html>
<!-- RC page10.html  18.09.14 -->
<head>
  <title>Image example</title>
</head>
<body>
<img
src="http://www.csc.liv.ac.uk/~martin/teaching/comp519/HTML/Cathedral.jpg"
title="Liverpool's Anglican cathedral"
alt="image of Liverpool's Anglican Cathedral" width="400" />

<p>The Anglican Cathedral of Liverpool</p> </body>

</html>
```

view page

Images (cont.)

- src - specifies the file name (and can include a URL)

- width and/or height - dimensions in pixels (often only need to specify one of them and the other is automatically scaled to match, where possible pictures should be resized using other programs to save on bandwidth and problems that some (older) browsers might have with resizing images)

- title - displayed when the mouse is “hovered” over the picture

- alt - text that is displayed when the image is missing, can’t be loaded (e.g. if file permissions aren’t set correctly), or if the client has disabled loading images in his/her browser

Tables

- tables are a common method for displaying data and other information
  - a table divides contents into rows and columns
  - by default, column entries are left-justified, so you must provide for your own alignment when needed (using Cascading Style Sheets, for example)

```
<html>
<!-- RC page11.html  15.08.14 -->
<head>
  <title>Tables</title>
</head>
<body>
<h2>A Simple Table</h2>
  <table>
   <tr>
     <td> Left Column </td>
     <td> Right Column </td>
   </tr>
   <tr>
     <td> Some data </td>
     <td> Some other data </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

Layout in a Table

```
<html>
<!- RC page12.html  15.08.14 -->

<head>
  <title>Table Layout</title>
</head>

<body>
  <table style="border: 1px solid;">
     <tr style="text-align: center;">
     <td style="border: 1px solid;">
            Left<br/>Column</td>
     <td style="border: 1px solid;
         vertical-align: top;">
            Right Column</td>
    </tr>
    <tr>
      <td style="border: 1px solid;">
           Some data</td>
      <td style="border: 1px solid;">
           Some data</td>
    </tr>
  </table>
</body>
</html>
```

view page

can have a border on tables using the “style” attribute

```
<table style= "border: 1px solid;">
```

increasing the number makes the border thicker

can control the horizontal & vertical layout within cells

```
<td style= "text-align:center">

<td style= "vertical-align:
  bottom">
```

can apply layout to an entire row

```
<tr style="text-align: center">

<tr style="vertical-align:
  top">
```

We will explore this more with Cascading Style Sheets (CSS).

Table Width

```
<html>
<!- RC page13.html  15.08.14 -->
<head>
  <title>Table Width</title>
</head>

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

view page

by default, the table is sized to fit the data

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

For example

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

Other Table Attributes

```
<html>
<!- RC page14.html  15.08.14 -->
<head>
  <title>Table Formatting</title>

 <style type="text/css" media="screen">
     table { border: 1px solid; padding: 1px;}
     th, td {  border: 1px solid; padding: 10px;
               text-align: center; }
  </style>
</head>

<body>
  <table>
    <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"> four </td>
      <td colspan="2"> five </td>
    </tr>
    <tr>
      <td> six </td> <td> seven </td>
    </tr>
</table>
</body>
</html>
```

view page

can control the space between cells & margins within cells

This is the "padding" attribute in the table and th,td style sheet declarations (more on this with Cascading Style Sheets).

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">
```

(This example uses CSS style sheet commands in the page <header>.)

Frames

- frames provide the ability to split the screen into independent parts

Frames are going out of fashion, partly because they interact poorly with web search engines (i.e. search engines cannot generally access the data stored in the inset frame objects).

Frames can also “break” the regular behaviour of browsers, most notably the “Back” button on the browser can behave in unexpected ways.

Because of these drawbacks to frames, I will not be discussing them in this course.

If you wish to design websites using frames (why would you??), I would encourage you to use the XHTML XFrames specifications (see the W3C website for more details), but this specification isn’t fully supported by all browsers at this time.

Frames are also not supported by the HTML 5 specification.