CSS, XML, Ajax

Week 4
LBSC 690
Information Technology
“Hello World” HTML

This is the header

This is the actual content of the HTML document
Rendering

- Different devices have different capabilities
  - Desktop
  - PDA
- Rendering maps logical tags to physical layout
  - Controls line wrap, size, font…
    - Place the title in the page border
    - Render <h1> as 24pt Times
    - Render <strong> as bold
- Somewhat browser-dependent
  - Internet Explorer and Netscape make different choices
HTML Document Structure

• “Tags” mark structure
  – <html>a document</html>
  – <ol>an ordered list</ol>
  – <i>something in italics</i>

• Tag name in angle brackets <>
  – Not case sensitive

• Open/Close pairs
  – Close tag is sometimes optional (if unambiguous)
Logical Structure Tags

• Head
  – Title

• Body
  – Headers: <h1> <h2> <h3> <h4> <h5>
  – Lists: <ol>, <ul> (can be nested)
  – Paragraphs:<p>
  – Definitions: <dt><dd>
  – Tables: <table> <tr> <td> </td> </tr> </table>
  – Role: <cite>, <address>, <strong>, …
Physical Structure Tags

- Font
  - Typeface: `<font face="Arial"></font>`
  - Size: `<font size="+1"></font>`
  - Color: `<font color="990000"></font>`
  - Bold: `<b></b>`
  - Italics: `<i></i>`
(Hyper)Links

index.html

```html
<html>
<head>
<title>Hello World!</title>
</head>
<body>
<p>Hello world! This is my first webpage!</p>
<p>Click <a href="test.html">here</a> for another page.</p>
</body>
</html>
```

test.html

```html
<html>
<head>
<title>Another page</title>
</head>
<body>
<p>This is another page.</p>
</body>
</html>
```
Hypertext “Anchors”

• Internal anchors: somewhere on the same page
  – `<a href="#students"> Students</a>
  • Links to: `<a name="students">Student Information</a>

• External anchors: to another page
  – `<a href="http://www.clis.umd.edu">CLIS</a>
  – `<a href="http://www.clis.umd.edu#students">CLIS students</a>

• URL may be complete, or relative to current page
  – `<a href="video/week2.rm">2</a>

• File name part of URL is case sensitive (on Unix servers)
  – Protocol and domain name are not case sensitive
Images

• `<img src="URL"> or `<img src="path/file">`
  – `<img src="http://www.clis.umd.edu/IMAGES/head.gif">
  – SRC: can be url or path/file
  – ALT: a text string
  – ALIGN: position of the image
  – WIDTH and HEIGHT: size of the image

• Can use as anchor:
  – `<a href=URL> <img src=URL2> </a>`
<table>
<thead>
<tr>
<th></th>
<th>eenie</th>
<th>mennie</th>
<th>miney</th>
</tr>
</thead>
<tbody>
<tr>
<td>mo</td>
<td>catch</td>
<td>a tiger</td>
<td></td>
</tr>
<tr>
<td>by</td>
<td>the</td>
<td>toe</td>
<td></td>
</tr>
</tbody>
</table>

code: `<table>
<tr><td>eenie</td><td>mennie</td><td>miney</td></tr>
<tr><td>mo</td><td>catch</td><td>a tiger</td></tr>
<tr><td>by</td><td>the</td><td>toe</td></tr>
</table>`
Table Example

<table align="center">
  <caption align="right">The caption</caption>
  <tr align="LEFT">
    <th>Header1</th>
    <th>Header2</th>
  </tr>
  <tr>
    <td>first row, first item</td>
    <td>first row, second item</td>
  </tr>
  <tr>
    <td>second row, first item</td>
    <td>second row, second item</td>
  </tr>
</table>
Frames

• Divide browser pages into separate sections
  – Useful when you want to scroll separately

• Each section can display an HTML page

• Example 1: menu frame on the left side of a page
  <frameset cols="10%,90%" >
    <frame src="template.html">
    <frame src="images.html">
  </frameset>

• Example 2:
  – http://www.hq.nasa.gov/alsj/frame.html
Cascading Style Sheets (CSS)

• Specify appearance, based on structure

• Style rules “cascade” from broad to narrow:
  – Browser’s default behavior
  – External style sheet
  – Internal style sheet
  – Inline style
Some Ways of Using CSS

• Inline style:
  – Causes only the tag to have desired properties
    
    ```html
    <p style="font-family:arial; color:blue">…</p>
    ```

• Internal stylesheet:
  – Causes *all* tags to have the desired properties
    
    ```html
    ... 
    <head>...
    <style type="text/css">
    p { font-family:arial; color:blue}
    </style>
    </head>
    <body>
    <p>...</p>
    ... 
    ```
Customizing Classes

• Define a custom style for standard HTML tag

... 
<head>...
<style type="text/css">
p.style1 { font-family:arial; color:blue}
p.style2 { font-family:serif; color:red}
</style>
</head>
<body>
<p class="style1">…</p>
<p class="style2">…</p>
...</
External Style Sheets

• Store formatting metadata in a separate file

```html
<head>
<link rel="stylesheet" href="mystyle.css" type="text/css" />
</head>
<body>
<p class="style1">…</p>
<p class="style2">…</p>
...
```

```
@charset "utf-8";

p.style1 { font-family:arial; color:blue}
p.style2 { font-family:serif; color:red}
```

General Structure for CSS

• Basic syntax:

```
selector {property: value}
```

- HTML tag you want to modify…
- The property you want to change…
- The value you want the property to take

• Example:

```
p { text-align: center;
   color: black;
   font-family: arial }
```

Causes

- Font to be center-aligned
- Font to be Arial and black
Designing Web Pages

• Key design issues:
  – Content: What do you want to publish?
  – Style: How do you want to present it?
  – Syntax: How can you achieve that presentation?

• Sources of information
  – Online tutorials (there are lots of these)
  – Technical materials (e.g., the W3C HTML spec)
Some Style Guidelines

• Design for generic browsers
  – And test on every version you wish to support
• Provide appropriate “access points”
  – User needs and navigation strategies differ
• Design useful navigational aids
  – A Web search may lead to the middle of a site
• Include some indication of currency
  – Date of last update, “new” icons, etc.
• Indicate who is responsible for the content
  – Helps readers assess authority
Some Accessibility Guidelines

• Design for device independence
• Maintain compatibility with earlier browsers
  – Provide alternative pages if necessary
• Provide alternatives to aural and visual content
  – Alt tags for images, transcripts for audio
• Make it is easy for assistive devices to work
  – Use structural (rather than layout-oriented) markup
  – Give a title to each frame
  – Only use HTML tables for table data (not content layout)
  – Use markup to indicate language switching
A text equivalent for every non-text element shall be provided.
Equivalent alternatives for any multimedia presentation shall be synchronized with the presentation.
Web pages shall be designed so that all information conveyed with color is also available without color.
Documents shall be organized so they are readable without requiring an associated style sheet.
Redundant text links shall be provided for each active region of a server-side image map.
Client-side image maps shall be provided instead of server-side image maps except where the regions cannot be defined with an available geometric shape.
Row and column headers shall be identified for data tables.
Markup shall be used to associate data cells and header cells for data tables that have two or more logical levels of row or column headers.
Frames shall be titled with text that facilitates frame identification and navigation.
Pages shall be designed to avoid causing the screen to flicker with a frequency >2 Hz and <55 Hz.
A text-only page, with equivalent information or functionality, shall be provided when compliance cannot be accomplished in any other way. The content shall be updated when the primary page changes.
When pages use scripting languages to display content or to create interface elements, the information provided by the script shall be identified with functional text that can be read by assistive technology.
When a web page requires that an applet, plug-in or other application be present on the client system to interpret page content, the page must provide a link to a plug-in or applet that complies with the above.
When electronic forms are designed to be completed on-line, the form shall allow people using assistive technology to access the information, field elements, and functionality required.
A method shall be provided that permits users to skip repetitive navigation links.
When a timed response is required, the user shall be alerted and given sufficient time to indicate more time is required.
HTML Editors

• Goal is to create Web pages, not learn HTML!
• Several are available
  – Macromedia Dreamweaver available commercially
  – Microsoft Word (Page->Edit with Word in IE 7
  – Many more options available on www.tucows.com
• Tend to use physical layout tags extensively
  – Detailed control can make hand-editing difficult
• You may still need to edit the HTML file
  – Some editors use browser-specific features
  – Some HTML features may be unavailable
  – File names may be butchered when you upload
HTML Validators

• Syntax checking: cross-browser compatibility
  – http://validator.w3.org
  – Try it on http://www.umd.edu 😞

• Style checking: Section 508 compliance
  – Try it on http://terpconnect.umd.edu 😞
What’s Wrong with the Web?

• HTML
  – Confounds structure and appearance (XML)

• HTTP
  – Can’t recognize related transactions (Cookies)

• URL
  – Links breaks when you move a file (PURL)
What’s a Document?

• Content

• Structure

• Appearance

• Behavior
History of Structured Documents

• Early standards were “typesetting languages”
  – NROFF, TeX, LaTeX, SGML

• HTML was developed for the Web
  – Too specialized for other uses

• Specialized standards met other needs
  – Change tracking in Word, annotating manuscripts, …

• XML seeks to unify these threads
  – One standard format for printing, viewing, processing
eXtensible Markup Language (XML)

- SGML was too complex
- HTML was too simple
- Goals for XML
  - Easily adapted to specific tasks
    - Rendering Web pages
    - Encoding metadata
    - “Semantic Web”
  - Easily created
  - Easily processed
  - Easily read
  - Concise
Some XML Applications

• Text Encoding Initiative
  – For adding annotation to historical manuscripts
  – http://www.tei-c.org/

• Encoded Archival Description
  – To enhance automated processing of finding aids
  – http://www.loc.gov/ead/

• Metadata Encoding and Transmission Standard
  – Bundles descriptive and administrative metadata
  – http://www.loc.gov/standards/mets/
Really Simple Syndication (RSS)

```xml
<?xml version="1.0"?>
<rss version="2.0">
  <channel>
    <title>Lift Off News</title>
    <link>http://liftoff.msfc.nasa.gov/</link>
    <description>Liftoff to Space Exploration.</description>
    <language>en-us</language>
    <pubDate>Tue, 10 Jun 2003 04:00:00 GMT</pubDate>
    <lastBuildDate>Tue, 10 Jun 2003 09:41:01 GMT</lastBuildDate>
    <docs>http://blogs.law.harvard.edu/tech/rss</docs>
    <generator>Weblog Editor 2.0</generator>
    <managingEditor>editor@example.com</managingEditor>
    <webMaster>webmaster@example.com</webMaster>
    <ttl>5</ttl>
    <item>
      <title>Star City</title>
      <description>How do Americans get ready to work with Russians aboard the International Space Station? They take a crash course in culture, language and protocol at Russia's Star City.</description>
      <pubDate>Tue, 03 Jun 2003 09:39:21 GMT</pubDate>
      <guid>http://liftoff.msfc.nasa.gov/2003/06/03.html#item573</guid>
    </item>
  </channel>
</rss>

See example at http://www.nytimes.com/services/xml/rss/
Atom Feeds

<?xml version="1.0" encoding="utf-8"?>
<feed xmlns="http://www.w3.org/2005/Atom">
  <title>Example Feed</title>
  <subtitle>A subtitle.</subtitle>
  <link href="http://example.org/feed/" rel="self"/>
  <link href="http://example.org/"/>
  <updated>2003-12-13T18:30:02Z</updated>
  <author>
    <name>John Doe</name>
    <email>johndoe@example.com</email>
  </author>
  <id>urn:uuid:60a76c80-d399-11d9-b91c-0003939e0af6</id>
  <entry>
    <title>Atom-Powered Robots Run Amok</title>
    <link href="http://example.org/2003/12/13/atom03"/>
    <id>urn:uuid:1225c695-cfb8-4ebb-aaaa-80da344efa6a</id>
    <updated>2003-12-13T18:30:02Z</updated>
    <summary>Some text.</summary>
  </entry>
</feed>
XML: A Family of Standards

• Definition: DTD
  – Known types of entities with “labels”
  – Defines part-whole and is-a relationships

• Markup: XML
  – “Tags” regions of text with labels

• Markup: XLink
  – Defines “hypertext” (and other) link relationships

• Presentation: XSL
  – Specifies how each type of entity should be “rendered”
XHTML Example

- View “The Song of the Wandering Aengus”

- Built from three files
  - yeats01.xml
  - poem01.dtd
  - poem01.xsl
XML Example

<?xml version="1.0"?>
<!DOCTYPE POEM SYSTEM "poem01.dtd">
<?xml-stylesheet type="text/xsl" href="poem01.xsl"?>

<POEM>
  <TITLE>The Song of Wandering Aengus</TITLE>
  <AUTHOR>  
    <FIRSTNAME>W.B.</FIRSTNAME>
    <LASTNAME>Yeats</LASTNAME>
  </AUTHOR>
  <STANZA>
    <LINE>I went on to the hazel wood,</LINE>
    <LINEIN>Because a fire was in my head,</LINEIN>
    <LINE>And cut and peeled a hazel wand,</LINE>
  </STANZA>
</POEM>
Document Type Definition (DTD)

<!ELEMENT poem ( (title, author, stanza)* )>
<!ELEMENT title (#PCDATA) >
<!ELEMENT author (firstname, lastname) >
<!ELEMENT firstname (#PCDATA) >
<!ELEMENT lastname (#PCDATA) >
<!ELEMENT stanza (line+ | linein+) >
<!ELEMENT line (#PCDATA) >
<!ELEMENT linein (#PCDATA) >

#PCDATA  span of text
a,b       a followed by b
a|b       either a or b
a*       0 or more a’s
a+       1 or more a’s
Specifying Appearance: XSL

```xml
<xsl:template match="POEM">

<HTML>
<BODY BGCOLOR="#FFFFCC">
<xsl:apply-templates/>
</BODY>
</HTML>

</xsl:template>

<xsl:template match="TITLE">

<H1>
<FONT COLOR="Green">
<xsl:value-of/>
</FONT>
</H1>

</xsl:template>
```
XLink Example

......

<poem xmlns:xlink="http://www.w3.org/1999/xlink">
<author xlink:href="yeatsRDFS3.xml" xlink:type="simple">W. B. Yeats</author>
<poems>
</poems>
</poem>

........
XHTML: Writing HTML as XML

<?xml version="1.0" encoding="iso-8859-1"?>
<html xmlns="http://www.w3.org/TR/xhtml1">
<head>
  <title>Title of text XHTML Document</title>
</head>
<body>
  <div class="myDiv">
    <h1>Heading of Page</h1>
    <p>here is a paragraph of text. I will include inside this paragraph a bunch of wonky text so that it looks fancy.</p>
    <p>Here is another paragraph with <em>inline emphasized</em> text, and <b>absolutely no</b> sense of humor.</p>
    <p>And another paragraph, this one with an <img src="image.gif" alt="waste of time" /> image, and a <br /> line break.</p>
  </div>
</body></html>
Some Basic Rules for XML

• XML is case sensitive
• XML declaration is the first statement
  – <?xml version="1.0"?>
• An XML document is a “tree”
  – Must contain one root element
  – Other elements must be properly nested
• All start tags must have end tags
• Attribute values must have quotation marks
  – <item id="33905”>
• Certain characters are “reserved”
  – For example: &lt; is used to represent <
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rss="http://purl.org/rss/1.0/"
 xmlns:dc="http://purl.org/dc/elements/1.1/"

 <rss:channel rdf:about="http://www.xml.com/xml/news.rss">
  <rss:title>XML.com</rss:title>
  <rss:link>http://xml.com/pub</rss:link>
  <dc:description>
   XML.com features a rich mix of information and services for the XML community.
  </dc:description>
  <dc:subject>XML, RDF, metadata, information syndication services</dc:subject>
  <dc:identifier>http://www.xml.com</dc:identifier>
  <dc:publisher>O'Reilly & Associates, Inc.</dc:publisher>
  <dc:rights>Copyright 2000, O'Reilly & Associates, Inc.</dc:rights>
 </rss:channel>
</rdf:RDF>
Some Ajax Applications

• Google Maps
  – http://maps.google.com

• Google Suggest
  – http://www.google.com/webhp?complete=1&hl=en

• Sajax Tables

• Sajax
  – http://www.modernmethod.com/sajax/
classic web application model (synchronous)

**Client**
- User activity
- Data transmission
- System processing

**Server**
- Data transmission
- System processing

**Time**
Ajax web application model (asynchronous)
Database Server-side Programming

Client-side Programming

Web Browser

Client Hardware

Server Hardware

Business rules

Interface Design

Interaction Design

- Relational normalization
- Structured programming
- Software patterns
- Object-oriented design
- Functional decomposition

• (MySQL)
• (PHP)
• (HTML, XML)
• (JavaScript)
• (IE, Firefox)
• (PC)
• (PC, Unix)
Even More Uses of XML …

- CML – Chemical Markup Language
- CellML – biological models
- BSML – bioinformatic sequences
- MAGE-ML – MicroArray Gene Expression
- XSTAR – for archaeological research
- MARCXML – MARC in XML
- AML – astronomy markup language
- SportsML – for sharing sports data
Summary

• Learning to build simple Web pages is easy
  – Which is good news for the homework!

• All documents are structured documents
  – But some expose the structure better than others

• XML is a flexible markup language
  – Complete separation of structure and appearance
Before You Go

On a sheet of paper, answer the following (ungraded) question (no names, please):

What was the muddiest point in today’s class?