HTML, CSS, JavaScript

In a couple of weeks when you make a web page and put your creativity programs on there, some people will want to show what they did to the class?

HW 10: Another chance to be creative: Last two weeks of term. Get a pair programming partner again!

Encoding Information: There’s more!

- Bits and bytes encode the information, but that’s not all
- Tags encode format and some structure in word processors
- Tags encode format and some structure in HTML
- Tags are one form of meta-data
- Meta-data is information about information

HTML and the Web

- The Web uses http:// protocol
- Its asking for a Web page, which usually means a page expressed in hyper-text markup language, or HTML
  - Hyper-text refers to text containing LINKS that allow you to leave the linear stream of text, see something else, and return to the place you left
  - Markup language is a notation to describe how a published document is supposed to look: what kinds of fonts, text color, headings, images, etc.
There are great resources out there

- http://www.w3schools.com/html/default.asp

Let's do it.

Example

- Write HTML in text editor: notepad++ or TextWrangler
- The file extension is .html; show it in Firefox or your browser

Example: myfirst.html

- To put in an image (.gif, .jpg, .png), use 1 tag
  `<img src="cooking-ewan-isabel.jpg" alt="Kids Cooking"/>

- To put in a link, use 2 tags
  `<a href="http://users.soe.ucsc.edu/~charlie/Charlie's page.html">

Basics of HTML #2

- Rule 4: An HTML file has this structure:
  `<html>
    <head><title>Name of Page</title></head>
    <body>
      ... Actual HTML page description goes here...
    </body>
  </html>

- Rule 5: Tags must be properly nested
- Rule 6: White space is mostly ignored
- Rule 7: Attributes (`style="color:red"`) preceded by space, name not quoted, value quoted

Basics of HTML #3

- To put in an image (.gif, .jpg, .png), use 1 tag
  `<img src="cooking-ewan-isabel.jpg" alt="Kids Cooking"/>

- To put in a link, use 2 tags
  `<a href="http://users.soe.ucsc.edu/~charlie/Charlie's page.html">

- More on HTML (including good tutorials) at
  http://www.w3schools.com/html/default.asp
**HTML Cheat Sheet:** In Resources on Course page

- `<html>`: Creates an HTML document.
- `<head>`: Sets off the title and other information that isn’t displayed on the web page itself.
- `<body>`: Sets off the visible portion of the document.

**BodyTags**

- `<head>`
- `<body` style=
- “background-color:pink”
- “color:black”

**TextTags**

- `<h1>`: Creates the largest headline.
- `<h6>`: Creates the smallest headline.
- `<b>`: Creates bold text.
- `<i>`: Creates italic text.
- `<tt>`: Creates teletype, or typewriter-style text.
- `<em>`: Emphasizes a word (with italic or bold).
- `<strong>`: Emphasizes a word (with italic or bold).

**Links**

- `<a href=“URL”>`: Creates a hyperlink; anchor between tags.
- `<a href=“URL”>`<img src=“URL”>

**Formatting**

- `<p>`: Creates a new paragraph.
- `<p style=text-align:left>`: Aligns a paragraph to the left (default), right, or center.
- `<br/>`: Inserts a line break.
- `<blockquote>`: Indents text from both sides.
- `<hr>`: Inserts a horizontal rule.
- `<hr size=“3”>`: Sets size (height) of rule.
- `<hr width=“80%”>`: Sets width of rule, in percentage or absolute value.

**Lists**

- `<dl>`: Creates a definition list.
- `<dt>`: Identifies each definition term.
- `<dd>`: Identifies each definition.
- `<ol>`: Creates a numbered list.
- `<ul>`: Creates a bulleted list.
- `<li>`: Encloses each list item, and adds a number or symbol depending upon the type of list selected.

**Images**

- `<img src=“name” alt=“description”>`: Places an image.
- `<img src=“name” alt=“description” style=float:right>`: Aligns an image: right.

**Tables**

- `<table>`: Creates a table.
- `<tr>`: Sets off each row in a table.
- `<td>`: Sets off each cell in a row.
- `<th>`: Sets off the table header (a normal cell with bold, centered text).
- `<table border=“1”>`: Sets width of border around table cells.
- `<table width=“500”>`: Sets width of table, in pixels.
- `<td colspan=“2”>`: Sets number of columns a cell should span (default=1).
- `<td rowspan=“4”>`: Sets number of rows a cell should span (default=1).

---

**Which does not apply to HTML?**

A. It is the language used for the web browser and the web server to communicate over the Internet.
B. It is the language used to express how a document should be displayed.
C. It is a language that allows for “documents” to be created that are not linear. (A book with chapters is linear – you normally read from start to end in order.)
D. All of A-C apply to HTML.

---

**CSS – separating style from content**

© Lawrence Snyder 2004

From http://www.w3schools.com/css/css_syntax.asp

---

**Adding CSS to your html file**

```html
<!DOCTYPE html>
<html>
<head>
<style>
    p {color:red; text-align:center;}
    body {background-image:url("images/ComputerSlug.gif");}
</style>
</head>
<body>
<p>Hello World!</p>
<p>This paragraph is styled with CSS.</p>
</body>
</html>
```
Using an external CSS

```html
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
<body>
<p>Hello World!</p>
<p>This paragraph is styled with CSS.</p>
</body>
</html>
```

mystyle.css

```css
p {color:red;text-align:center;}
body {background-image:url("images/ComputerSlug.gif");}
```

Using inline style annotations

```html
<!DOCTYPE html>
<html>
<head>
</head>
<body>
<p>Hello World!</p>
<p style="color:red;text-align:center;">This paragraph is styled with CSS.</p>
</body>
</html>
```

A. css allows you to separate the specific formatting information from the main body of the document.
B. css allows you to change how many html documents will be displayed by changing just a single file
C. A & B
D. neither A nor B

Tagging and Meta Data

OED Entry For Byte – Meta-data

byte (baɪt). Computer. (Etymology: prob. influenced by ba’t, ba’t). 1. A group of eight consecutive bits operated on as a unit in a computer. IBM (Blauw & Brooks in IBM Systems Jnl. III. 122 for 6-bit unit of information is fundamental to most of the formats of the System 360). A consecutive group of n such units constitutes a field of length n. Fixed-length fields of length one, two, four, and eight are termed bytes, halfwords, words, and double words respectively. 1964 IBM Jnl. Res. & Developm. VII. 971 When a byte of data appears from an I/O device, the CPU is halted, dumped, used and restored. 1967 P. A. Stark Digital Computer Programming no. 351 The normal operations in fixed point are done on four bytes at a time. 1968 Dataweek 24 Jan. 111 Tape reading and writing is at from 34,160 to 192,000 bytes per second.
Meta-Data Describes Data

- Meta-data is data about data ... a description of what the data is
- Knowing what the data is, as in the OED, allows us to process it better for users
- Here's an example: Search OED for def of "binary"
  - Without meta-data, get 8,311 hits ... which one is the definition?
  - With meta-data, get each definition in order ... how?

Meta-data Separation

- Improving on the meta-data of HTML
- Meta-data describes what the data is, but because the tags can be distinguished from the content, it separates itself from the content – that's good
- But HTML combines "identifying content" with "saying how to process it", i.e. display it
- Big new idea (not part of HTML)

The Advantage of Separating

- Separating the content (data) from the processing
  - You can do many different things with the content
    - Display it in multiple ways
    - Do calculations on it
  - You can maximize expertise
    - The content expert (you) puts the data together
    - The processing expert (some programmer) writes the processing code based on the tags

The World of XML

- The Extensible Markup Language (XML) is a mark up language in which YOU think up the tags ... it is a self-defining language!
- The usual rules for tags apply
  - Enclose in < and >
  - Start tag <mynewtag> and End tag </mynewtag>
  - Tags must always be matched or self-terminated
  - Tags can have attributes (think those up, too) of form attributename="valueInQuotes"
- Use .xml as the file extension
- Always start with "standard text" (shown later)
Example of XML

- Suppose I want to record information about my cd collection using XML, I might write:

```xml
<catalog>
  <cd>
    <title>Maggie May</title>
    <artist>Rod Stewart</artist>
    <country>UK</country>
    <company>Pickwick</company>
    <price>8.50</price>
    <year>1990</year>
  </cd>
  <cd>
    <title>When a man loves a woman</title>
    <artist>Percy Sledge</artist>
    <country>USA</country>
    <company>Atlantic</company>
    <price>8.70</price>
    <year>1987</year>
  </cd>
  ...
</catalog>
```

I invent the tags; they make sense to me, and I can write a program to process such descriptions.

Learning XML

- Since we think up the tags ourselves, it’s the easiest language in the world to learn, right?
- Tags can serve in three roles ...
  - Identity – tag it so you know what it is
    ```xml
    <name>George Washington</name>
    ```
  - Affinity – all properties of a thing should be collected together
    ```xml
    <personal>
      <name>George Washington</name>
      <height>6' 2"</height>
      <teeth>Wooden</teeth>
      <home>Mount Vernon</home>
    </personal>
    ```
  - Collection – enclose a group of items of the same type in a collective tag
    ```xml
    <presidents>
      <prez num="1"><personal><name>George ...</prez>
      <prez num="2"><personal><name>John ...</prez>
      <prez num="3"><personal><name>Thomas ...</prez>
      ...
      <prez num="44"><personal><name>Barack ...
    </presidents>
    ```
- These uses become intuitive quickly

Ways To Use Tags

- Identity – tag it so you know what it is
  ```xml
  <name>George Washington</name>
  ```
- Affinity – all properties of a thing should be collected together
  ```xml
  <personal>
    <name>George Washington</name>
    <height>6' 2"</height>
    <teeth>Wooden</teeth>
    <home>Mount Vernon</home>
  </personal>
  ```

Ways To Use Tags (continued)

- Collection – enclose a group of items of the same type in a collective tag
  ```xml
  <presidents>
    <prez num="1"><personal><name>George ...</prez>
    <prez num="2"><personal><name>John ...</prez>
    <prez num="3"><personal><name>Thomas ...</prez>
    ...
    <prez num="44"><personal><name>Barack ...
  </presidents>
  ```

Example: Classify The Uses

- Identity
- Affinity
- Collection
Display XML with Browser

- We can see the structure of our XML (and check that it is well formed) by displaying it in Firefox.
- Introduce an error in a tag and see the error message when you browse it.

```
This page contains the following errors:
error on line 3 in column 6 opening and ending tag mismatch: sight, line 8 and sight
Below is a rendering of the page up to the first error.
```

A Diary Of Travels

- One XML database could have entries for trips
- Kinds of info:
  - Where you went
  - Year
  - Pictures
- Also need a title

Example: Building The XML File

```
<travels>
  <article>
    <sight>Orangutan Reserve, Borneo</sight>
    I got invited to give a talk at a workshop in Singapore, and we all went and popped over to Borneo before the workshop and went to an Orangutan Reserve. On the last morning we saw a mom and baby in the wild. You can kind of see it here. Borneo is the most interesting place I’ve ever been.
    <year>2006</year>
  </article>
  <article>
    <sight>Kyoto Japan</sight>
    I got invited to give the keynote at SIGGRAPH 2010 in Tokyo and we all went and visited Kyoto and outlying areas like the holy mountai
    <year>2010</year>
  </article>
  <article>
    <sight>Roatan Honduras</sight>
    I got invited to give the keynote at SIGGRAPH 2010 in Tokyo and we all went and visited Kyoto and outlying areas like the holy mountai
    <year>2010</year>
```
Example: Building The XML File

```xml
<travel>
  <visit>
    <sight>Orangutan Reserve, Borneo</sight>
  </visit>
</travel>
```

Processing The File

- For this application, “processing” the file means displaying it on the screen.
- We use a browser to display XML tagged content.
- Here’s how:
  - For each tag, we say what HTML we want, using “XSL”
  - Put those definitions into a file (they’re called templates)
  - Put directions at the top of the XML definition telling it where to find the definitions file
  - “Run” the XML in Firefox or other browser … it does all the rest!

XSL: Extensible Stylesheet Language

- XSL is “processing” markup language for XML … and (of course) it’s written in XML.
- Let’s take a look.
- This is the top of the file.

The Rest of XSL Definition

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:mytravel="http://www.w3.org/1999/XSL/Transform">

  <!-- Standard header text … must be first in XSL file
  Here’s how to say what XML tag this definition is for:HTML in here
  Important tag, to be explained next-->

  <xsl:template match="mytravel:visit">
    <!-- HTML in here-->
  </xsl:template>
</xsl:stylesheet>
```

XSL: Extensible Markup Language

- Just looking at XSL, it seems very complicated because it mixes its own tags with HTML tags.
- XSL is “processing” markup language for XML and (of course) it’s written in XML.
- Let’s take a look.
- This is the top of the file.

More kinds of meta data all the time

- The semantic web
- An “ontology” i.e. “taxonomy” of the kinds of things there are in the world
  - People, place, thing, animal, organization, country
- New tags within web pages
- Could be put there by people
- Could be programmatically identified by text processing algorithms (like what Watson Jeopardy uses)
What are you supposed to learn?

- HTML lets you programmatically indicate how a particular content should be displayed.
- It can be served up by any HTTP server anywhere in the world.
- Typically uses UTF-8 encoding to guarantee being able to be shown.
- CSS lets you partially separate content from presentation.
- BUT XML entirely separates the DATA from the PROCESSING of that data.
- This is a powerful idea.
- Processing => search it, display it etc.

When an ____ “program” runs it takes in ____ as input and outputs ____

A. html, xml, xsl
B. xml, xsl, html
C. xsl, html, xml
D. xsl, xml, html
E. xml, html, xsl

True or false, every proper html file is also a proper xml file.

JavaScript

```html
<!DOCTYPE html>
<html>
<head>
<script>
function myFunction() {
  document.getElementById("demo").innerHTML="Goodbye";
}
</script>
</head>
<body>
<h1>My Web Page</h1>
<p id="demo">Hello!</p>
<button type="button" onclick="myFunction()">Click Me!</button>
</body>
</html>
```
**goodbye.js**

```javascript
function myFunction() {
    document.getElementById("demo").innerHTML="Goodbye!";
}
```

---

**What are you supposed to learn?**

- HTML lets you programmatically indicate how a particular content should be displayed.
- It can be served up by any HTTP server anywhere in the world.
- CSS lets you partially separate content from presentation
- BUT XML entirely separates the DATA from the PROCESSING of that data
- This is a powerful idea
- Processing => search it, display it etc
- JavaScript puts full power of computing in a web page