XML

October 24, Unit 6

What is XML?

- Stands for *eXtensible Markup Language*
- It is a markup language, like HTML
- But,
  - XML is designed to markup data
  - HTML is designed to display data
  - XML is concerned with what data is
  - HTML is concerned with how data looks
- Both use tags, attributes, entities, etc.

XML vs. HTML

- In HTML
  - Pre-defined tags
  - Pre-defined attributes
- XML
  - Define our own tags
  - Define our own attributes
  - Can have anything we want
  - Must have a root element

General XML

- Recommended by the W3C
- Why?
  - It is designed to easily transfer data from one system to another
  - Especially across the Internet
- With XML we can categorize, structure, organize, and display data
  - We can use XSL (like CSS) to describe how the data should look
    - More on that later

XML Syntax

- Again, XML looks a lot like HTML
  - We have tags, attributes, etc.
- XML documents require a *root element*
  - Must “wrap” around the entire document
  - <html> is an example
- We have to **create** all the tags, attributes, etc. that we need

Recipe Example

- Recipes have two main parts:
  - Ingredient list
  - Steps for preparation
- So what tags do we need?
  - Root element: <recipe>
  - Maybe a title? : <title>
  - Ingredients List: <ingredients>
  - Steps to make the recipe: <steps>
  - Way to list individual items: <item>

Pizza Dough Recipe

<recipe>
<title> Pizza Dough</title>
<ingredients>
<item> 3 c flour</item>
<item> 1 c warm water</item>
<item> 1 package yeast</item>
<item> 2 tbs olive oil</item>
</ingredients>
<steps>
<item> Mix water and yeast</item>
<item> Combine flour, water and yeast mixture, and oil in a bowl until smooth</item>
<item> Cover and let rise until doubled, about 1 hour</item>
<item> Divide into 2 portions and use as needed</item>
</steps>
</recipe>

Adding More Structure

- Could describe the recipe in more detail
- What about adding a quantity tag?

  <item><quantity> 3 cups**</quantity>** flour**</item>**
- Could also add a units tag

  <item><quantity> 3**</quantity> <unit>** cup**</unit>** flour**</item>**
- Flour is a food item, we could also give it a special tag

  <item><quantity> 3**</quantity> <unit>** cup**</unit> <ingred>** flour**</ingred></item>**

Student List Example

```
<studentlist>
<title> CMPT165 Student List</title>
<student>
    <name> Mary Washburn</name>
    <number> 200127772</number>
</student>
<student>
    <name> Sam Smith</name>
    <number> 200128888</number>
</student>
<student>
    <name> Mike Jones</name>
    <number> 2000129999</number>
</student>
</studentlist>
```

- Root element: **<studentlist>**
- **<title>**
- **<student>**
  - **<name>**
  - **<number>**

Student List Example, Cont.

```
<studentlist>
<title> CMPT165 Student List</title>
<student>
    <firstname> Mary</firstname>
    <lastname> Washburn</lastname>
    <number> 200127772</number>
</student>
<student>
    <firstname> Sam</firstname>
    <lastname> Smith</lastname>
    <number> 200128888</number>
</student>
<student>
    <firstname> Mike</firstname>
    <lastname> Jones</lastname>
    <number> 2000129999</number>
</student>
</studentlist>
```

- Root Element: **<studentlist>**
- **<title>**
- **<student>**
  - **<firstname>**
  - **<lastname>**
  - **<number>**

Using Structure

- How much structure?
  - Better to have too much than too little
  - We can always ignore tags
    - If we don't specify the way they should look, they'll be ignored
    - Programs can be written which also ignore the tags we don't want to use
- Structure should be logical
  - Not logical to have a <letter> tag which goes around each letter in our lists

    <item><letter>M</letter><letter>i</letter><letter>x</letter>….</item>
  - We use tags to describe the data
    - Letter is not the data we are interested in
- Structure can be used to manipulate the data
  - For example, if we wanted to double the recipe, it would be easy with the highly structured version and a recipe program
    - Some websites use this

XML Schemas

- XML is not very useful if we have to define a new set of tags for every single document we write
- Other people have written sets of tags and attributes for many purposes
  - These sets of XML tags and attributes are called *schemas*
- Common XML schema include:
  - XHTML
  - SVG
  - MathML
  - OpenOffice

XHTML

- Defined by an XML *schema*
- In the schema it has the tags and attributes we're used to
  - Rules relating to those tags
    - must contain at least one
- Browsers use the definition to display web content
- The schema being used is part of the very first part of your XHTML pages
  - (big long text no one can remember)

SVG

- Scalable Vector Graphics
- XML language
- Represents vector graphics on the web
- Created by the W3C
- You wouldn’t create or edit an SVG image file in a text editor
  - But! If there is a definition for an image file available, then any browser, or image editing program could open the file
  - Not all browsers support SVG yet, some have limited support
- Example in your course pack, page 112 Figure 6.3 of a smiley face created using SVG

MathML

- Markup language for mathematical expressions
- Why?
  - Mathematical expressions aren't well supported by most programs
    - And if they are, the expressions are meaningless
    - But most writing type programs support HTML
  - Difficult because a lot of computers don't have the required fonts to display the expression
  - Ever try to add an expression in MS Word?
    - Not trivial
- Not well-supported by browsers
  - Mozilla supports this (and I assume the Mozilla family of browsers, Firefox and Netscape)
- Can include mathematical expression in your web page

OpenOffice

- Free office-suite of products
  - Word processor, spreadsheet, etc.
- Sponsored by Sun Microsystems
- Open source project
  - Anyone can look at the code and contribute to it
- Cross-platform
  - Windows, Mac, and Linux!
- The file formats for OpenOffice are XML-based
  - Why does it matter?
    - Programs with access to the definitions for the file format can all open the documents you create!
- OpenOffice is almost identical to MS Office in terms of user interface
  - Good, cheap (read free) alternative to purchasing MSWord

Why XML?

- So it's used for lots of things, by why bother?
  - In general, using XML as a file format is both human and computer readable
    - Not like viewing a .doc file in a text editor
  - Supports Unicode (can contain pretty much anything)
  - Strict syntax makes it easy to write programs that can read the xml files
  - Stored as plain text files
  - Cross-platform
  - We can validate it! (I know…your favorite, right?)

Questions?