1.264 Lecture 14

XML
What is XML?

• Extensible Markup Language (XML) is:
  – a World Wide Web Consortium (W3C) standard for
  – a file format to
  – easily and cheaply distribute electronic documents on the World Wide Web
  – extensible, not frozen like HTML
  – supporting rich structure, like objects or hierarchies or relationships
  – supporting validation and well-formed properties
  – avoiding applets, scripts, plug-ins, etc.
  – separating form (how it looks) from content (what it is)
Markup languages

• Many markup languages exist
  MS Word uses Rich Text Format (RTF), a proprietary markup language (next version of MS Office will use XML)
  WordPerfect
  HTML
  XML (Interleaf word processors have used XML for many years)
  SGML
• These store tags (markup) in addition to raw text
• XML and HTML are subsets of SGML
  XML provides 80% of SGML function with 20% of its complexity
  SGML spec is 155 pages; XML spec is 35 pages
  Removed all optional features of SGML
XML Concepts

- XML is self-describing and can be validated:
  XML document contains the business rules to which its data must conform
  Rules can be reused in other documents: documents can be more specialized types (inheritance) of a base type
- XML applications
  Data interchange format between computers
    Using Web server as data channel between databases
    Automated processing of documents exchanged
  Common format for Web, electronic, paper documents, ...
    XML as a general markup language
    XML used for manuals, CDs, help and other text documents
    Handled by standard browsers (IE, Mozilla, Netscape...)
  Remote procedure call/invocation protocol
    Executes Web services or processes on other computers
XML Tools

• XHTML displays data
  – CSS formats XHTML pages
• XML describes data
  – XSLT formats XML pages (different handling of tags)
• XML has other associated languages
  – DTD: Document type definition: business rules
    Not XML itself (oddly), being replaced by XML Schema
  – XML Schema
  – XSLT: extensible stylesheet language/transformation
    Can also do ‘simple’ transformations of tags
  – XPath sublanguage for dynamic hyperlinks, queries
  – Others: RDF, XForms, … see www.w3c.org
XML Document Structure

- **HTML:**
  - Head
  - Body
    Tags are predefined in HTML (or XHTML) standard

- **XML:**
  - Prolog
    XML declaration (defines version)
    Document type declaration: DTD (defines tags)
    Stylesheet declaration, etc.
  - Body
    Tags describe data elements
    Tags are defined in DTD or XSchema document, which anyone can create
XML Document Type Declarations

• Well formed document
  – Follows XML syntax but may not be valid
  – Used by browsers to accept XML documents that have already been validated by a server
    No need to download the DTD and revalidate

• Valid document
  – Follows all rules:
    e.g. #REQUIRED for an element
Example 1: XML file only

```xml
<?xml version="1.0" encoding="utf-8"?>
<MEMO>
  <TO>darris@mit.edu</TO>
  <FROM>george@mit.edu</FROM>
  <CC>loai@mit.edu</CC>
  <SUBJECT>XML example</SUBJECT>
  <BODY>This is an example of an XML document. We use email since it's familiar; we could use a purchase order, catalog item, bus route, etc. Note that we can redefine the body tag, unlike HTML. In fact, we can define any set of tags we wish. These tags define the meaning, not the display appearance of the document.
  </BODY>
</MEMO>
```

Memo.xml
XML exercise

In Dreamweaver, File->New: Basic, XML
  Type a short memo in XML format: follow last slide example
  File-> Check Page -> Validate as XML
    Make a deliberate error to see what happens

File-> New: Basic, XSLT entire page
  Select XML file you just created
  Look at Bindings panel to see the XML schema

Drag each tag into document window in a separate paragraph

Hit F12 to preview in XML and XSL files in browser
  Use Internet Explorer only (examples not generalized for all browsers)

Examine Code view briefly

XSLT styles and transforms XML pages:
  For browser display
  For translation to and from databases
Example 2: XML and DTD

```xml
<?xml version="1.0"?>
<!DOCTYPE MEMO SYSTEM "Memo.dtd">

<MEMO LANGUAGE="Western" ENCRYPTED="128"
      PRIORITY="HIGH">
  <TO>loai@mit.edu</TO>
  <FROM>administration@mit.edu</FROM>
  <CC>darris@mit.edu</CC>
  <BCC>george@mit.edu</BCC>
  <SUBJECT>Sample Document with External DTD</SUBJECT>

  <BODY>
    This is the monthly &MITREMINDER;.
  </BODY>
</MEMO>
```

Memo2.xml
Example 2: XML and DTD

```xml
<?xml version="1.0"?>

<!ELEMENT MEMO (TO+, FROM, CC*, BCC*, SUBJECT?, BODY?)>
<!ATTLIST MEMO
  LANGUAGE (Western|Greek|Latin|Universal) "Western"
  ENCRYPTED CDATA #IMPLIED
  PRIORITY (NORMAL|LOW|HIGH) "NORMAL">

<!ELEMENT TO (#PCDATA)>
<!ELEMENT FROM (#PCDATA)>
<!ELEMENT CC (#PCDATA)>

<!ELEMENT BCC (#PCDATA)>
<!ATTLIST BCC
  HIDDEN CDATA #FIXED "TRUE">

<!ELEMENT SUBJECT (#PCDATA)>
<!ELEMENT BODY (#PCDATA)>

<!ENTITY MITREMINDER "reminder to turn in all timesheets">
```

Memo.dtd
Document Type Definition

- **DOCTYPE**: class (type) of document
  Placed in XML file, refers to DTD file to be used to validate

- **ELEMENT**: object in document
  Either all valid values are given in a list in (), or
  The element is defined later in the DTD file
  Symbols: +: 1 or more, *: 0 or more, ?:0 or 1, none: exactly 1

- **ATTLIST**: valid attribute list for element
  #CDATA: character data
  #PCDATA: parsed character data (can’t have < > &…)
  #REQUIRED: element must be present
  #IMPLIED: element optional, no default value
  #FIXED: attribute value is fixed

- **ENTITY**: a constant value

- | means OR
Example 3: XHTML file with XML, DTD

- Examine ChemicalProduct.xml in Dreamweaver
  Contains two chemicals
- Examine ChemicalProduct.dtd in Dreamweaver
  Contains tag definitions, validation (very simple)
- Open ChemicalProduct.htm in Dreamweaver
  Preview in browser
  `<xml>` tag is Microsoft extension, used for simplicity here
- We can validate the .xml file against the .dtd file
  There are many XML validators available, and your supply chain applications, etc. have them
  Run ValidateXML.htm (preview in browser) and check Memo.xml and ChemProduct.xml
    Make a deliberate mistake in a tag and see what happens
Example 3: ChemProduct.xml

```xml
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE CATALOG SYSTEM "ChemProduct.dtd">

<CATALOG>
  <CHEMICAL>
    <UNNBR>2796</UNNBR>
    <CHEMICALNAMES>Battery fluid acid</CHEMICALNAMES>
    <QTYLIMIT>30L</QTYLIMIT>
    <HAZCLASS>8</HAZCLASS>
  </CHEMICAL>

  <CHEMICAL>
    <UNNBR>1738</UNNBR>
    <CHEMICALNAMES>Chloride</CHEMICALNAMES>
    <QTYLIMIT>30L</QTYLIMIT>
    <HAZCLASS>6.1</HAZCLASS>
  </CHEMICAL>

</CATALOG>
```
Example 3: ChemProduct.dtd

<?xml version="1.0"?>

<!ELEMENT CATALOG (CHEMICAL+)>  
<!ELEMENT CHEMICAL (UNNBR, CHEMICALNAMES, QTYLIMIT, HAZCLASS)>  

<!ELEMENT UNNBR (#PCDATA)>  
<!ELEMENT CHEMICALNAMES (#PCDATA)>  
<!ELEMENT QTYLIMIT (#PCDATA)>  
<!ELEMENT HAZCLASS (#PCDATA)>
Example 3: ChemProduct.htm

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Untitled Document</title>
</head>
<body>

<xml
src="ChemProduct.xml"
id="xmlldso"
async="false">
</xml>

<table
datasrc="#xmlldso"
width="100%"
border="1">

... Continued on next page
Example 3: ChemProduct.htm, part 2

```html
<thead>
  <th>UN Number</th>
  <th>Chemical Names</th>
  <th>Quantity Limit</th>
  <th>Hazardous Material Class</th>
</thead>

<tr align="left">
  <td><span datafld="UNNBR"></td>
  <td><span datafld="CHEMICALNAMES"></td>
  <td><span datafld="QTYLIMIT"></td>
  <td><span datafld="HAZCLASS"></td>
</tr>
</table>

</body>
</html>
```
Example 4: XSchema

• XSchema is an XML language that replaces DTDs, which are not XML
• XSchema defines the business rules for an XML document in a database-oriented way, and allows for validation
  – Validation may be done on the server sending the XML document, the client receiving the document, or both
• Open Memo.xsd, the XSchema file
  – Its business rules are identical to those in Memo.dtd
<?xml version="1.0" encoding="UTF-8" ?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="BCC">
    <xs:complexType mixed="true">
      <xs:attribute name="HIDDEN" type="xs:string" use="required" fixed="TRUE" />
    </xs:complexType>
  </xs:element>

  <xs:element name="BODY">
    <xs:complexType mixed="true" />
  </xs:element>

  <xs:element name="CC">
    <xs:complexType mixed="true" />
  </xs:element>

</xs:schema>
<xs:element name="MEMO">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="TO" maxOccurs="unbounded" />
      <xs:element ref="FROM" />
      <xs:element ref="CC" minOccurs="0" maxOccurs="unbounded" />
      <xs:element ref="BCC" minOccurs="0" maxOccurs="unbounded" />
      <xs:element ref="SUBJECT" minOccurs="0" />  
      <xs:element ref="BODY" minOccurs="0" />
    </xs:sequence>
    <xs:attribute name="PRIORITY" use="optional" default="NORMAL">
      <xs:simpleType>
        <xs:restriction base="xs:NMTOKEN">
          <xs:enumeration value="NORMAL" />
          <xs:enumeration value="LOW" />
          <xs:enumeration value="HIGH" />
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="ENCRYPTED" type="xs:string" use="optional" />
  </xs:complexType>
</xs:element>
<xs:attribute name="LANGUAGE" use="optional"
  default="Western">
  <xs:simpleType>
    <xs:restriction base="xs:NMTOKEN">
      <xs:enumeration value="Western" />
      <xs:enumeration value="Greek" />
      <xs:enumeration value="Latin" />
      <xs:enumeration value="Universal" />
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>

<xs:element name="FROM">
  <xs:complexType mixed="true" />
</xs:element>
<xs:element name="SUBJECT">
  <xs:complexType mixed="true" />
</xs:element>
<xs:element name="TO">
  <xs:complexType mixed="true" />
</xs:element>
</xs:schema>
Example 5: AJAX (Asynchronous Javascript and XML)

• XML documents can be formatted by the XSLT language
  – It applies styles to an XML document for display to a human user
  – It queries (selects) values from an XML document for display or transmission to another system
  – It uses the XPath XML sublanguage for queries and flexible hyperlinks

• Dreamweaver generates XML and XSL combinations
  – Plants.xml and Plants.xsl are the next example
  – Combine this with some Javascript for a nice user interface
    Security problems still exist with Javascript
Example 5: Plants.xml

<CATALOG>
  <PLANT>
    <COMMON>Bloodroot</COMMON>
    <BOTANICAL>Sanguinaria canadensis</BOTANICAL>
    <ZONE>4</ZONE>
    <LIGHT>Mostly Shady</LIGHT>
    <PRICE>$2.44</PRICE>
    <AVAILABILITY>031507</AVAILABILITY>
  </PLANT>

  <PLANT>
    <COMMON>Columbine</COMMON>
    <BOTANICAL>Aquilegia canadensis</BOTANICAL>
    <ZONE>3</ZONE>
    <LIGHT>Mostly Shady</LIGHT>
    <PRICE>$9.37</PRICE>
    <AVAILABILITY>030607</AVAILABILITY>
  </PLANT>
</CATALOG>
Example 5: Customer.xml

<CUSTOMERS>
  <CUSTOMER>
    <CNAME></CNAME>
    <PNAME></PNAME>
    <QUANTITY></QUANTITY>
    <PRICE></PRICE>
    <DATE></DATE>
  </CUSTOMER>
</CUSTOMERS>
Example 5 concluded

- **PlantCatalog.htm**
  - Javascript to show plant catalog, allow selection, accumulate plants purchased
  - XHTML to format and display the page
    - Tables, buttons
  - XML DSO (Data Source Object) used to parse XML
    - Read Plant.xml
    - Write Customer.xml
**XML and databases**

- Databases can read and write XML
  - Tables and relationships can be expressed in DTDs or XSchema (xsd)
  - XML files that are database fragments can be exchanged between clients and servers
    - Domain tables can be sent as XML for validation
    - Data tables can be sent as XML for document transfer and transactions
  - XSLT can translate (the T in XSLT) from one set of XML tags to another
    - Allow integration between two supply chain partners who don’t have exactly the same document standards
  - Web services (next lecture) are a standard way of using XML and related standards between databases
Summary

• XML documents hold self-describing data
  Hierarchies, objects, database tables can be sent
  Extensible, flexible, decided by industry groups, partners
• XSL documents can format XML documents and transform (XSLT) tag names, data types, etc.
• DTDs can validate XML documents
  URL has the DTD that server or client can use
  DTDs are limited: can’t define data types, etc.
• XSchema can validate XML documents
  More structured, more extensive than DTDs
• Databases can read and write XML
  Web servers can send and receive XML as payload in HTTP, much like HTML pages
  Microsoft has made XML the markup language in Office
  Putting a disruptive technology in place to automate commerce