Introduction to XML
Shortcourse Handout

August 17, 2004
Technology Support Shortcourses
Texas Tech University
Copyright © 2004

Introduction
This introductory course provides an overview of XML (Extensible Markup language) technology. XML is fast becoming the standard in cross-platform data exchange. It is used to describe data and format it for exchange. Topics covered in this shortcourse are: what XML is and what it looks like (syntax), XML components (elements, attribute, etc.) and use, XPATH, XML as a tree, DTD and schemas, and building HTML from XML. A basic knowledge of the WWW and HTML would be helpful for understanding XML.

Course Objectives
After completing the Introduction to XML shortcourse, you should be able to:
• Explain what XML is and what it is used for.
• Identify XML and understand basic syntax.
• Draw an XML document in tree format.
• Understand XPath conceptually and how it is used to identify node locations in XML.
• Create a well-formed XML document.
• Understand what DTDs and XML schemas are used for.
• Understand what XSLT is used for.

What is XML?
■ XML is eXtensible Markup Language.
■ XML is a way to format data into a simple but structural text format:
  <XMLclass>
    Eric'sXMLpresentation.ppt
  </XMLclass>
■ XML defines a universal standard for electronic data exchange.
■ Interpretable across languages and environments.
■ XML can be read by any platform capable of reading simple text documents.

Example 1
<?xml version="1.0"?>
<authors>
  <name>
    <first>Jim</first>
    <middle>Bob</middle>
    <last>Jackson</last>
  </name>
  <name>
    <first>Kathleen</first>
    <middle>Marie</middle>
    <last>Smith</last>
  </name>
</authors>
Components of XML
- The first line must always be (as of Fall 2002):
  - `<?xml version="1.0"?>`
- XML uses `<` and `>` symbols to create ‘tags’ similar to HTML; however in XML, the
tags can be defined explicitly by the programmer.
- Tags begins like this `<tag>` and end with `</tag>`.
- An XML document can be viewed as a tree structure, in which each location on the
  tree is a node.

XML Use
- XML is used to transfer data between two or more groups that might not
  necessarily use otherwise compatible technology.
- XML is a way to structure data in certain predefined formats.
- XML uses ‘tags’ to contain and structure data.

XML Advantages
- Platform independence
- Language independence
- Easily readable by text editors
- Ability to structure data in a way agreeable by two or more parties
- XML simplifies application integration

XML over the web
- XML can be sent easily over the web, just as any other text document
  - Via FTP
  - Via SMTP (Simple Mail Transfer Protocol)
  - Via HTTP

XML Example 2
`<?xml version="1.0"?>`
<cars>
  <car type="Sportscar">
    <make>Chevrolet</make>
    <model>Corvette</model>
    <color>White</color>
    <year>1999</year>
  </car>
  <car type="Truck">
    <make>Dodge</make>
    <model>Ram</model>
    <color>Red</color>
    <year>2000</year>
  </car>
</cars>`
XML document structure as a tree

```
<author>
  <name>
    <first>Jim</first>
    <middle>Bob</middle>
    <last>Jackson</last>
  </name>
  <publisher>
    <name>Westinghouse</name>
    <address>
      <street>4567 8th Street</street>
      <city>Boston</city>
      <state>Massachusetts</state>
      <zip>23490</zip>
    </address>
  </publisher>
</author>
```

- Each circle in the tree is called a node with a node-set being a node and its descendants (like a tree-branch).
- `author` is the PARENT of `name` and `publisher`.
- Similarly, `address` is a CHILD of `publisher`.
- `street` and `city` are DESCENDANTS of `author` and `publisher` (grand-____ is not a correct designation).
**Back to Example 2**

Let’s draw out the tree structure ourselves.  
(This is the interactive part.)

```xml
<?xml version="1.0"?>
<cars>
  <car type="Sportscar">
    <make>Chevrolet</make>
    <model>Corvette</model>
    <color>White</color>
    <year>1999</year>
  </car>
  <car type="Truck">
    <make>Dodge</make>
    <model>Ram</model>
    <color>Red</color>
    <year>2000</year>
  </car>
</cars>
```

**Intro To XPath**

- XPath is a way of identifying exactly where you are in the XML tree.
- It allows you to pinpoint the exact location of data in XML.

- Samples:
  - "author/publisher/name"
  - "author/publisher/address/street"

- XPath differentiates between
  - "author/name" and
  - "author/publisher/name"
**Node Types in XML**

- **Root node**
  - Top level above all else (`/`)
  - Not actually seen in the XML (implied)
  - Ex: contains `<cars>`

- **Element node**

- **Attribute node**

- **Text node**

- **Comment node**

- **Namespace node (not shown)**

```xml
<cars>
  <!-- this is a description of cars -->
  <car type="Sportscar">
    <make>Chevrolet</make>
    <model>Corvette</model>
    <color>White</color>
    <year>1999</year>
  </car>
  <car type="Truck">
    <make>Dodge</make>
    <model>Ram</model>
    <color>Red</color>
    <year>2000</year>
  </car>
</cars>
```

**Well-formed XML**

- ALL tags must be closed.
  
  `<results>`some data`<results>`
  
  `<approved>`<yes>`<approved/>` must be `</yes>`

- All attributes must be contained in quotes
  
  `<purchaseOrder number="PO2056">`</purchaseOrder>

- All attributes must have values.
  
  `<car speedy/>` - not well-formed
  
  `<car speed="fast">` - well-formed

- Tag names cannot contain spaces.

- Certain characters not allowed.
  
  `&` (Bob & Jane) must be replaced with `&amp;`

- XML is case sensitive
  
  `<Bob/>` is not the same as `<boB/>`
**Now you guys do it!**
- Open the XML editor.
- Type in your XML, following the instructions in the handout.
- Keep in mind that there is no exact right or wrong way in this case if the XML is well-formed.

**DTD – Document Type Definition**
- Allows 2 or more parties to agree on a format for XML exchange between them.
- Validates XML documents to verify:
  - Proper nesting has occurred.
  - All required tags are present and accounted for.
  - Specific units of information are of the correct type and fall within the specified legal values.
- XML that passes the DTD test is **valid** (in compliance with the appropriate DTD) and **well-formed** (no XML syntax errors).

**XML Schemas**
- A more advanced version of DTDs with several advantages:
  - Provide support for namespaces
    - Helps resolve conflicts in tag names
  - Richer datatypes than DTDs
  - User-defined types called Archetypes
  - Allowance for attribute grouping
    - Many attributes often go together

**XML to HTML**
- XML is a cousin to HTML and can be formatted into HTML.
- XSLT ‘transforms’ XML into HTML by combining XML with a stylesheet or template.
- XSLT stands for eXtensible Stylesheet Language Transformations.

---

**Overview of XSL Transformations**
- XML + XSLT Stylesheet =
  - HTML
  - Or XML
  - Or WML
- HTML output is dynamic based on what data is contained in the XML string.
- XSLT stylesheets are XML documents and conform to all the properties of XML.
Example – TTU Colleges

Build an XML document from the following information.

TTU Colleges
Business Administration
Areas:
  Accounting
  Phone: 742-3181
  Fax: 742-3182
  
  Finance
  Area Coordinator: Dr. R. Stephen Sears
  Phone: 742-3196
  Fax: 742-2099
  
  ISQS
  Area Coordinator: Dr. Surya Yadav
  Phone: 742-2165
  
  Management
  Phone: 742-3176
  
  Marketing
  Area Coordinator: Dr. Robert Wilkes
  Phone: 742-3162
  Fax: 742-2199

A sample might look like...
<TexasTech>
  <college name="Business Administration">
    <areas>
      <area (fill in info starting here; don't forget to close your tags)...
### Advanced Example 1 – Sports Likes

Build an XML document from the following information.

<table>
<thead>
<tr>
<th>Sports Likes and Dislikes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>L = LIKES</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Person</th>
<th>Soccer</th>
<th>Basketball</th>
<th>Football</th>
<th>Baseball</th>
<th>Tennis</th>
<th>Badminton</th>
<th>Boxing</th>
<th>Cycling</th>
<th>Swimming</th>
<th>Running</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barry Parker</td>
<td>L</td>
<td>D</td>
<td>L</td>
<td>L</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>L</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td>Mary Beth</td>
<td>D</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>L</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td>Allen Parsons</td>
<td>L</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td>Steve Scott</td>
<td>L</td>
<td>D</td>
<td>L</td>
<td>D</td>
<td>L</td>
<td>D</td>
<td>L</td>
<td>D</td>
<td>L</td>
<td>D</td>
</tr>
<tr>
<td>Nancy Mack</td>
<td>D</td>
<td>L</td>
<td>D</td>
<td>L</td>
<td>D</td>
<td>L</td>
<td>D</td>
<td>L</td>
<td>D</td>
<td>L</td>
</tr>
<tr>
<td>Michael Phelps</td>
<td>D</td>
<td>L</td>
<td>L</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>L</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td>Peter Oster</td>
<td>L</td>
<td>D</td>
<td>D</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>D</td>
<td>D</td>
</tr>
</tbody>
</table>
**Advanced Example 2 – Budget Office**

Build an XML document from the following information.

**TTU BUDGET OFFICE**

<table>
<thead>
<tr>
<th>Phone (806) 742-3228</th>
<th>Fax (806) 742-1309</th>
<th>Mailstop #1096</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nimfa Aguilar, Interim Asst. V.P. for Budget</td>
<td>Room 119A</td>
<td></td>
</tr>
<tr>
<td>Lisa Mills, Section Coordinator</td>
<td>Room 116</td>
<td></td>
</tr>
<tr>
<td>Jodi Coen, Administrative Assistant</td>
<td>Room 119</td>
<td></td>
</tr>
<tr>
<td>Vickie Craig, Senior Business Assistant</td>
<td>Room 119</td>
<td></td>
</tr>
<tr>
<td>Rachel Rodriguez, Senior Analyst</td>
<td>Room 121</td>
<td></td>
</tr>
<tr>
<td>Li Qin, Senior Analyst</td>
<td>Room 123</td>
<td></td>
</tr>
<tr>
<td>Linda Smekar, Senior Analyst</td>
<td>Room 127</td>
<td></td>
</tr>
<tr>
<td>Melissa Teinert, Senior Analyst</td>
<td>Room 125</td>
<td></td>
</tr>
<tr>
<td>Heather Crouch, Lead Account Processor</td>
<td>Room 127B</td>
<td></td>
</tr>
<tr>
<td>Claudius Garland, Lead Account Processor</td>
<td>Room 131</td>
<td></td>
</tr>
<tr>
<td>Peggy Dehay, Lead Account Processor</td>
<td>Room 131</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FUND #</th>
<th>FUND NAME</th>
<th>BUDGET ANALYST</th>
</tr>
</thead>
<tbody>
<tr>
<td>0180</td>
<td>Graduate Tuition</td>
<td>LINDA SMEKAR</td>
</tr>
<tr>
<td>0250-0299</td>
<td>Service</td>
<td></td>
</tr>
<tr>
<td>0300-0699</td>
<td>Designated Accounts</td>
<td></td>
</tr>
<tr>
<td>1101-1299</td>
<td>Auxiliary Accounts</td>
<td></td>
</tr>
<tr>
<td>1800-2599</td>
<td>Scholarships *</td>
<td></td>
</tr>
<tr>
<td>2600-2799</td>
<td>Loan Funds</td>
<td></td>
</tr>
<tr>
<td>2800-3599</td>
<td>Endowments *</td>
<td></td>
</tr>
<tr>
<td>3700-3999</td>
<td>Plant Funds (Including 3709 HEAF Bond Funds), Retirement of Indebtedness* and Awards*</td>
<td></td>
</tr>
<tr>
<td>4100-4999</td>
<td>Foundation and Agency Accounts</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>0070</th>
<th>Faculty Salaries (State)</th>
<th>RACHEL RODRIGUEZ</th>
</tr>
</thead>
<tbody>
<tr>
<td>0074-0096</td>
<td>E&amp;G (State) Accounts</td>
<td></td>
</tr>
<tr>
<td>5001-5099</td>
<td>TTUS E&amp;G (State) Accounts</td>
<td></td>
</tr>
<tr>
<td>5100-5499</td>
<td>TTUS Local Accounts</td>
<td></td>
</tr>
<tr>
<td>5900</td>
<td>TTUS Foundation</td>
<td></td>
</tr>
</tbody>
</table>

| 0128-0220 | Special Items (Exclude Adv. Tech. & Adv. Res.) & College Work Study (State) | LI QIN |
| 0213 | College Work Study (State) | |
| 0221-0223 | Tuition Revenue Bonds | |
| 0249 | Reserve for Adjustments (State) | |
| 1000-1099 | Designated Accounts | |
| 1300-1799 | Current Restricted | |

| 0020-0069 | E&G (State) Accounts | MELISSA TEINERT |
| 0102-0124 | E&G (State) Accounts | |
| 0700-0999 | Designated Accounts | |
| 1100 | Auxiliary Accounts | |

| 0200 | Texas Workforce (State) | LISA MILLS |
| 0202-0205 | Advanced Technology & Advanced Research (State) | |
| 0211-0212 | Advanced Technology & Research (State) | |
| 0224-0247 | HEAF Appropriated (State) | |

* ACCOUNTS—NO TECH E-MAIL ALLOWED FOR BUDGET REVISIONS
**Good XML Reference Books**
- Learning XML by Erik Ray  
  - ISBN: 0596000464
- Beginning XML by David Hunter (editor), Jeff Rafter, John Pinnock, Chris Dix, Kurt Cagle, Roger Kovack  
  - ISBN:1861005598
  by Aaron Skonnard, Martin Gudgin  
  - ISBN: 0201740958

**Good XSLT Reference Books**
  - ISBN:0130404462
- XSLT by Doug Tidwell (O'Reilly)  
  - ISBN:0596000537

**Questions**
- Any final questions?
- For anything that needs further clarification or explanation, please contact me.  
  Eric Burrell  
  742-4500 x237  
  eric.burrell@ttu.edu