

XML DOM and SLT

Qiang Yang
Simon Fraser University

2/1/01 topic 1

XML: tasks

- Parse:
 - to see if document is well formed
 - validates against a DTD or schema
- Traverse
 - Extract nodes and perform tree operations
- Transformation
 - Generates HTML file for viewing
- Querying
 - "Select -statements" wrapped in XML format
 - XML-QL
 - SQLX

2/1/01 topic 2

Document Object Model

- An important element of XML
- Allows XML database elements to be represented as objects
- Has a large collection of built-in attributes and methods
 - childNodes – list of children nodes
 - documentElement – root of the hierarchy
 - nextSibling – next object for traversal
- XSL: supplies method for data processing

2/1/01 topic 3

An example XML File: [message.xml](#)

```

<?xml version="1.0"?>
<!-- This is our first XML file -->

<MESSAGE>
<TO>Student</TO>
<FROM> Author </FROM>
<SUBJECT> Introduction to XML </SUBJECT>
<BODY>
<welcome>Welcome to XML! </welcome>
<mainbody>I would like to make an introduction!</mainbody>
</BODY>
</MESSAGE>

```

2/1/01 topic 4

Object Model

```

graph TD
  MESSAGE[MESSAGE] --- TO[TO]
  MESSAGE --- FROM[FROM]
  MESSAGE --- SUBJECT[SUBJECT]
  MESSAGE --- BODY[BODY]
  BODY --- welcome[welcome]
  BODY --- mainbody[mainbody]

```

2/1/01 topic 5

Using XML DOM for traversal

- Using Microsoft XMLDOM


```

<script language = "JavaScript">
  var RootElement1;
  var xmlDoc1 = new ActiveXObject("microsoft.xmlDOM");
  xmlDoc1.load("message.xml");

```
- Accessing the elements


```

RootElement1=xmlDoc1.documentElement;
toData.innerText = RootElement1.childNodes.item(0).text;
fromData.innerText = RootElement1.childNodes(1).text;
bodyData1.innerText = RootElement1.childNodes(3).childNodes(0).text;
bodyData2.innerText = RootElement1.childNodes(3).childNodes(1).text;

```

2/1/01 topic 6

Displaying Results

- Embed in HTML document [display.html](#)

```
body bgcolor="ffffff" onLoad="StartUp()">  
  
To: <span id=toData> </span><br>  
From: <span id=fromData> </span><br>  
Body Welcome: <span id=bodyData1></span><br>  
Body Main Part: <font color = "green"><span id=bodyData2> </span>  
                </font><br>  
  
</body>
```

2/1/01

topic

7

Traversal

```
function StartUp()  
{ if (xmlDoc1.readyState=="4")  
  { Traverse(xmlDoc1); }  
  else  
  { alert("Process could not start"); }  
}
```

```
function Traverse (node)  
{  
  alert("Tree Node is: " + node.xml);  
  alert("Tree Node Length is: " + node.childNodes.length);  
  alert("Tree Node first Child Length is " + node.firstChild.childNodes.length);  
  Traverse(node.firstChild);  
}
```

2/1/01

topic

8

XSL Transformation



- XSL Processors
 - www.alphaworks.ibm.com
 - IE5 has built in XSL processors
 - www.w3.org/Style/XSL
- Using IBM Lotus API for transformation
 - Java Transform <xmlfile.xml> <xslfile.xsl> <outputfile.html>

2/1/01

topic

9

XSL, SAX and DOM

- XSL: easy to program
 - Static program for transformation
- SAX and DOM: more flexible
 - Parsing and transformation
 - Dynamic
 - SAX: lightweight for memory
- DOM
 - Can generate new XML document
 - Memory limitation for large XML documents

2/1/01

topic

10

XSL: XML Stylesheet

- Two functions
 - Query
 - Transformation to HTML
- XML Namespace
 - Defines a context
 - Example

```
<?xml:namespace ns=file:reference" prefix="message">  
<message:body>Hi</message:body>
```
- XML has a predefined namespace

```
<xsl:stylesheet xmlns:xsl="uri:xsl">
```

2/1/01

topic

11

Separating Data

```
<?xml version="1.0"?>  
<Catalog>  
  
<Book>  
<Title> Teach Yourself XML</Title>  
<Author>  
<Name>Charles A</Name>  
<Address>119 Northwood Drive </Address>  
<City> Hiawatha</City>  
<State> Iowa </State>  
...
```

2/1/01

topic

12

... from Style → Result

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="uri:xsl"*
<xsl:template match="/">
<html><body> <table border="1">
  <tr style = "font-weight:bold;color:blue">
    <td>Author Name </td>
    <td> Publisher Phone </td> ... other titles ...
  </tr>
  <xsl:for-each select="Catalog/Book" rder-by="-Author/Name">
    <tr>
      <xsl:apply-templates/>
    </tr>
  </xsl:for-each>
</table> </body></html>
</xsl:template>
```

2/1/01

topic

13

Using XSL Template for database querying

```
<xsl:for-each select="Catalog/Book">
  <tr>
    <xsl:apply-templates/>
    <td> <xsl:value-of select="ISBN"/><td>
    <td> <xsl:value-of select="PublisherBookID"/></td>
    ...
  </tr>
</xsl:for-each>
```

2/1/01

topic

14

XML Schemas

- More powerful than DTD's
- Defines concepts and relationships between pieces of data

```
<?xml version="1.0">
<!-- simple schema -->
<Schema name = "Catalog" xmlns="urn:schemas-microsoft-com:xml-data">
<!-- ... body of schema ... -->
</Schema>
```

2/1/01

topic

15

An example message data schema

```
<?xml version="1.0"?>
<Schema name="MessageDef" xmlns="urn:schemas-microsoft-com:xml-data">
  <!-- first the individual components -->
  <ElementType name="TO" content="textOnly"?>
  <ElementType name="BODY" content="textOnly"?>
  ...
  <!-- now the main message object -->
  <ElementType name="Message" content="eltOnly">
  <element type="TO"/>
  <element type="BODY"/>
  </ElementType>
</Schema>
```

2/1/01

topic

16

Importing Schema into Data

```
<?xml version="1.0"?>
<mm:Message xmlns:mm="x-schema:message-schema.xml">
<mm:TO> Hi </mm:TO>
<mm:BODY> XML Schema is used here in action...</mm:BODY>
</mm:MESSAGE>
```

2/1/01

topic

17

Java and DOM

- Defined in org.w3c.dom
- Also Sun package: com.sun.xml.tree
- New Document
XmlDocument doc = new XmlDocument ();
- Open an existing input source
XmlDocument doc = XmlDocument.createXmlDocument(InputStream is)
Or
XmlDocument doc = XmlDocument.createXmlDocument(String url)
- Traverse a tree
Node node=Doc.getDocumentElement().getFirstChild();
node.getNextSibling();
node.getNodeName().equals("key")

2/1/01

topic

18

Using XSL to transform to HTML

- Match all "Book" tags in an XML document
- Then, prints out all XML elements using the fonts specified

```
<xsl:template match = "Book" >
  <fo:block font-size = "10pt" space-before="12pt">
    <xsl:apply-templates/>
  </fo:block>
</xsl:template>
```

2/1/01

topic

19

Transform.java

```
import com.lotus.xml.*;

public class Transform {
    public static void main(String[] args) {

        // input to processors
        XSLProcessor myProcessor = new XSLProcessor();
        XSLTInputSource xmlSource = new XSLTInputSource(args[0]);
        XSLTInputSource xslStylesheet = new XSLTInputSource(args[1]);

        // output
        XSLResultTarget xmlOutput = new XSLResultTarget(args[2]);
        myProcessor.process(xmlSource, xslStylesheet, xmlOutput);
    }
    catch (java.net.MalformedURLException exc) {}
    catch (java.io.IOException exc) {}
    catch (org.xml.sax.SAXExceptions exc) {}
}
}
```

2/1/01

topic

20