CmpE 473
Internet Programming

Pınar Yolum
pyolum@cmpe.boun.edu.tr

Department of
Computer Engineering
Boğaziçi University

Chapter 9
XSL Languages

Based largely on
Service-Oriented Computing: Semantics, Processes, Agents
– Munindar P. Singh and Michael N. Huhns, Wiley, 2004

Examples from www.w3schools.com

XML Query Languages

- XSL: e**X**tensible **S**tylesheet **L**anguage
  - XPath: Defining parts of XML documents
  - XSLT: For transforming XML documents
  - XSL-FO: For formatting XML documents

XPath

- Address parts of XML documents
- Uses the logical structure of XML documents
- Model XML documents as trees with nodes
  - Element nodes
  - Attributes nodes
  - Text nodes (PCDATA)
  - Comments
  - Root node: above root of document
  - Namespace nodes
  - Processing instruction nodes
  - Comment nodes

Data Model

- Parent in XPath is like parent as traditionally in computer science
- Child in XPath is confusing:
  - Element nodes, comment nodes, processing instruction nodes and text nodes are children
  - An attribute is not the child of its parent
  - Makes a difference for certain kinds of recursion (e.g., apply-templates discussed in XSLT)
- Our terminology is based on the traditional terminology:
  - e-children, a-children, t-children
  - Sets via et- or ta-, etc.

XPath Paths

- Path expressions to select nodes in XML document

Expression	Description
element	All child nodes of the node
/	From the root node
//	Nodes in the document from the current node that matches the selection
.	The current node
..	The parent of the current node
@	Attributes

XPath Paths

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<bookstore>
 <book>
<title lang="eng">Harry Potter</title>
<price>29.99</price>
</book>
 <book>
<title lang="eng">Learning XML</title>
 <price>39.95</price>
</book>
</bookstore>
```

Example: What happens if you select

- bookstore
- /bookstore
- bookstore/book
- //book
- bookstore//book
- //@lang

XPath Navigation

- Predicates embedded in square brackets
- Select children according to position, e.g., [j], where j could be 1 … last()
- //title[@lang]: All titles with attribute lang
- /bookstore/book[price>35.00]: All the book elements of bookstore with price>35
- Wildcard, *:
  - * matches any element node
  - @*: finds all attribute values
  - Node() matches any node of any kind
- AND
  - //book/title | //book/price: Title AND price elements of all book elements

XPath Queries

- Incorporate selection conditions in XPath
  - Attributes: //Song[@genre="jazz"]
  - Elements: //Song[starts-with(.//group, "Led")]
  - Existence of attribute: //Song[@genre]
  - Existence of subelement: //Song[group]
  - Boolean operators: and, not, or
  - Set operator: union (|); none others
  - Arithmetic operators: >, <, …
  - String functions: contains(), concat(), length(),
  - Aggregates: sum(), count()

XSLT

- A functional programming language
- Transform one XML into another format XML, HTML, XHTML
- Add (or remove) new elements
- Rearrange or sort elements
- XML Source Tree → XML Result tree
- Use XPath to define matching pattern

XSLT Stylesheets

- A stylesheet specifies transformations on a document

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
    href="URL-to-dot-xsl"?> <!– the sheet to use →
<main-tag>
…
</main-tag>
```

- Use the XSLT namespace, conventionally abbreviated as xsl
- Includes primitives:
  - Copy-of
  - <for-each select="…">
  - <if test="…">
  - <choose >

XSLT Templates: 1

- A pattern to specify where a given transform should apply
  - This match only works on the root:

```
<xsl:template match="/">
…
</xsl:template>
```

- Value-of Element
  - Select the value and add it to the transformation

XSLT Templates: 2

- <xsl:for-each>
  - Apply to every node of a specified element
  - <xsl:for-each select="catalog/cd">
- Filtering output
  - Add extra constraint
  - <xsl:for-each select="catalog/cd[artist='Bob Dylan']">
  - = (equal)
  - != (not equal)
  - < (less than)
  - > (greater than)

XSLT Templates: 3

- Sort the selected nodes
  - <xsl:sort select="artist"/>
- If test: Apply based on condition

```
<xsl:if test="price &gt; 10">
<tr>
    <td> <xsl:value-of select="title"/> </td>
    <td><xsl:value-of select="artist"/></td>
</tr>
</xsl:if>
```

XSLT Templates: 4

- Choose (if-then-else)
  - Used with <xsl:when> and <xsl:otherwise>

```
<xsl:choose>
    <xsl:when test="price &gt; 10">
        <td bgcolor="#ff00ff">
        <xsl:value-of select="artist"/></td>
    </xsl:when>
    <xsl:otherwise>
        <td><xsl:value-of select="artist"/></td>
    </xsl:otherwise>
</xsl:choose>
```


XSLT Templates: 5

- Can be applied recursively on the et-children via

  <xsl:apply-templates/>

- By default, if no other template matches, recursively apply to et-children of current node (ignores attributed) and to root:

```
<xsl:template match="*|/">
  <xsl:apply-templates/>
</xsl:template>
```

- Can over-apply; to override the default, may need an empty template:

  <xsl:template match="…"/> <!– e.g., match all text() →

Example XML

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<catalog>
<cd>
<title>Empire Burlesque</title>
 <artist>Bob Dylan</artist>
<country>USA</country>
<company>Columbia</company>
<price>10.90</price>
<year>1985</year> </cd>
</catalog>
```


Example XSL

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
      <html>
      <body>
            <h2>My CD Collection</h2>
            <table border="1">
            <tr bgcolor="#9acd32">
                  <th align="left">Title</th>
                  <th align="left">Artist</th> </tr>
            <xsl:for-each select="catalog/cd">
            <tr>
                  <td><xsl:value-of select="title"/></td>
                  <td><xsl:value-of select="artist"/></td> </tr>
            </xsl:for-each>
            </table>
     </body>
     </html>
</xsl:template>
</xsl:stylesheet>
```

Tools

- Many open and commercial tools
  - XMLSPY: XML and XML Schema edit & validate; XSL edit & transform
  - Xml.apache.org: Tools for integration with Java
  - Stylusstudio.com: XSLT editor
  - Eclipse: XML plug-ins