XML Parsing: XPath

Laboratory of Service Design and Engineering

2011/2012

Outline

- XPath Overview
- Java & XPath
- Exercises

Introduction to XPath

- XPath is a query language designed for querying XML documents
- XPath uses path expressions to navigate in XML documents
- XPath contains a library of standard functions
- XPath describes paths to elements in a similar way an operating system describes paths to files
- XPath is a W3C recommendation
  - http://www.w3.org/TR/xpath20/

XML Nodes & Relationships

- library is the **parent of book; book is** the **parent of the two chapters**
- The two chapters are the **children of** book, and the section is the **child of** the second chapter
- The two chapters of the book are **siblings (they have the same parent)**
- Library, book, and the second chapter are the **ancestors of the section**
- The two chapters, the section, and the two paragraphs are the **descendents of the book**

```
<library>
  <book>

    <chapter>
    </chapter>

    <chapter>
      <section>
        <paragraph/>
        <paragraph/>
      </section>
    </chapter>

  </book>
</library>
```

XML Document

```
<bookstore>
    <book year="2000">
        <title lang="eng">Snow Crash</title>
        <author>Neal Stephenson</author>
        <publisher>Spectra</publisher>
        <isbn>0553380958</isbn>
        <price>14.95</price>
    </book>

    <book year="2005">
        <title>Burning Tower</title>
        <author>Larry Niven</author>
        <author>Jerry Pournelle</author>
        <publisher>Pocket</publisher>
        <isbn>0743416910</isbn>
        <price>5.99</price>
    </book>

    <book year="1995">
        <title>Zodiac</title>
        <author>Neal Stephenson</author>
        <publisher>Spectra</publisher>
        <isbn>0553573862</isbn>
        <price>7.50</price>
    </book>
</bookstore>
```

Node Selection

- A path that begins with a / represents an absolute path, starting from the top of the document **/bookstore/book/title**
- Note: an absolute path can select more than one element
- A / by itself means "*the whole document*"
- A path that does not begin with a / represents a path *starting from the current element*
- **book/title** Selects all title elements that are children of book

Node Selection

- A path that begins with // can start from anywhere in the document.
- //title Selects every element title, no matter where it is.
- bookstore//title Selects all title elements that are descendant of the bookstore element, no matter where they are under the bookstore element.

Predicates

- Predicates are used to find a specific node or a node that contains a specific value.
- **/bookstore/book[1]** Selects the first book element that is the child of the bookstore element.
- **/bookstore/book[last()]** Selects the last book element that is the child of the bookstore element.
- **/bookstore/book[position()<3]** Selects the
- first two book elements that are children of the bookstore element.
- **/bookstore/book[price>3]** Selects all the book elements of the bookstore element that have a price element with a value greater than 3.

Attributes

- You can select attributes by themselves, or elements that have certain attributes.
- To choose the attribute itself, prefix the name with **@**
- //**@lang** Selects all attributes that are named lang
- To choose elements that have a given attribute, put the attribute name in square brackets
- //**title**[**@lang**='**eng**'] Selects all the title elements that have an attribute named lang with a value of 'eng'

Wildcards

- * Matches all element node at this level
- /bookstore/* Selects all the children nodes of the bookstore element
- @* Matches all attribute node
- //title[@*] Selects all title elements which have any attribute
- node() Matches any node of any kind

Selecting Several Paths

- By using the | operator in an XPath expression you can select several paths.
- //book/title | //book/price
  - Selects all the title and price elements of all book elements
- /bookstore/book/title | //price
  - Selects all the title elements of the book element of the bookstore element and all the price elements in the document

Axes

- An axis is a set of nodes relative to a given node
  - X::Y means “choose Y from the X axis”
- child::book Selects all book nodes that are children of the current node
- child::text() Selects all text child nodes of the current node
- child::node() Selects all child nodes of the current node
- ancestor::book Selects all book ancestors of the current node
- ……

Arithmetic Operators

- + add
- - subtract
- * multiply
- div (not /) divide
- mod modulo (remainder)

Boolean Operators

- = equals (Notice it's not ==)
- != not equals
- value = node-set will be true if the node-set contains any node with a value that matches value
- value != node-set will be true if the node-set contains any node with a value that does not match value
- Hence, value = node-set and value != node-set may both be true at the same time!

Boolean Operators

- And
- Or
- not()
- The following are used for numerical comparisons only:
  - <, <=, >, >=

XPath and Java

javax.xml.xpath

Class/Interface	Description
XpathFactory	Used to create an XPath object.
XPath	Provides access to the XPath evaluation environment. Provides the **evaluate** methods to evaluate XPath expressions in an DOM tree.
XPathExpression	Provides the **evaluate** methods to evaluate **compiled** XPath expressions in an XML document.

Java XPath Example

```
public class XPathTest {

       public static void main(String[] args)
        throws ParserConfigurationException, SAXException,
               IOException, XPathExpressionException {

         DocumentBuilderFactory domFactory = DocumentBuilderFactory.newInstance();
         domFactory.setNamespaceAware(true);
         DocumentBuilder builder = domFactory.newDocumentBuilder();
         Document doc = builder.parse("books.xml");

         XPathFactory factory = XPathFactory.newInstance();
         XPath xpath = factory.newXPath();
         //Compile an XPath expression for later evaluation
         XPathExpression expr = xpath.compile("/bookstore/book/title/text()");

         NodeList nodes = (NodeList) expr.evaluate(doc, XPathConstants.NODESET);
         for (int i = 0; i < nodes.getLength(); i++) {
             System.out.println(nodes.item(i).getNodeValue());
         }

       }
}
```

Java and XPath Types

- XPath and Java language do not have identical type systems
- XPath 1.0 has only four basic data types:
  - node-set
  - Number
  - Boolean
  - String
- The evaluate() method may return
  - org.w3c.dom.NodeList
  - java.lang.Double
  - java.lang.Boolean
  - java.lang.String
  - org.w3c.dom.Node
- When you **evaluate an XPath expression in Java,** the second argument specifies the return type you expect.

Exercise

- Download the code and run on your machines
- Use “Employee.xml” file from previous lab
  - Make a function which prints all employees list with detail
  - A function which accepts name as parameter and print that particular employee
  - A function which accepts salary and a operator (=, > , <) as parameters and prints employee that fulfill that condition.