Birkbeck
UNIVERSITY OF LONDON

XML

Session 6

Introduction to XPath

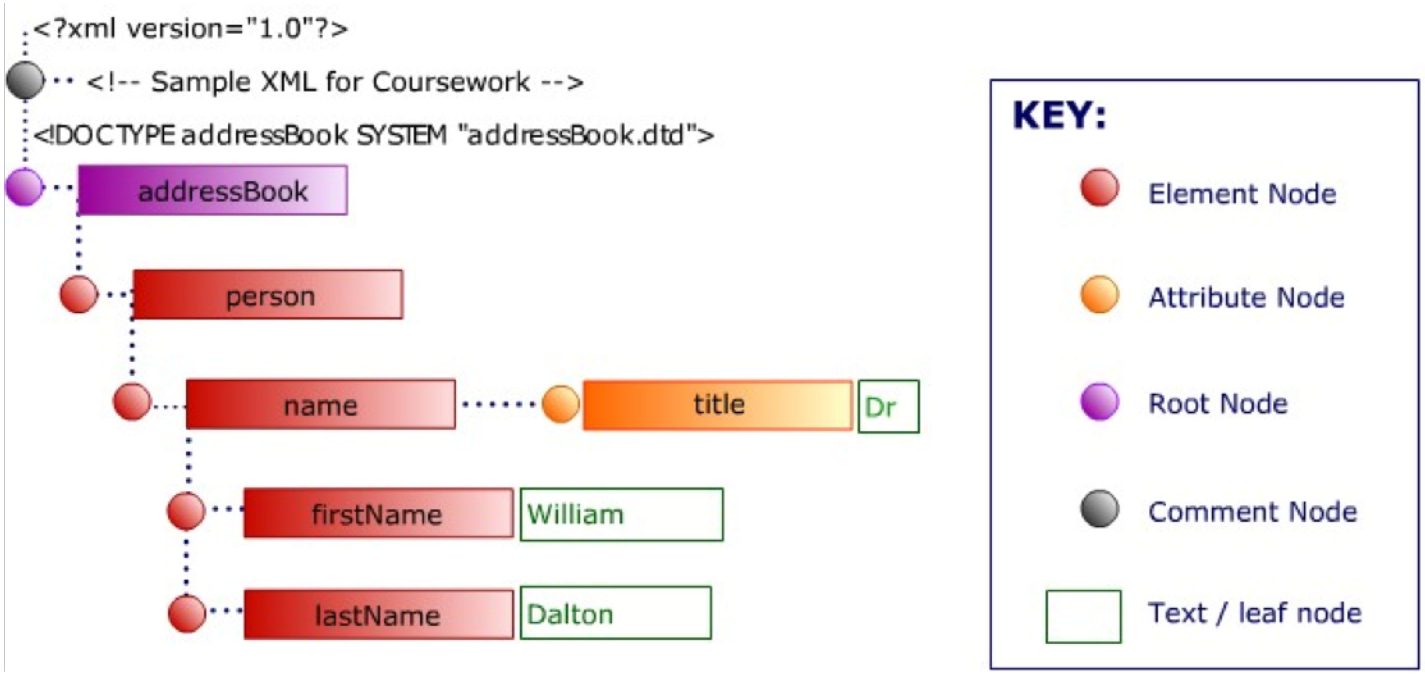
Purpose of XPath

- Syntax for identifying nodes and groups of nodes in an XML document
- Use it for navigating an XML document
- Use it to query and test content
- XPath is a major component of XSLT
- XPath is a W3C standard

XPath – Definition of nodes

- Nodes are the components of an XML document
- There are 7 types of nodes but the most common are:
  - Elements
  - Attributes
  - Text
  - Document node (root node)
- Used in XSLT to identify content and values
- Forms a tree structure with the root node at the base (or top)

An XML tree


<?xml version="1.0"?>
<!-- Sample XML for Coursework -->
<!DOCTYPE addressBook SYSTEM "addressBook.dtd">
addressBook
person
name
title
Dr
firstName
William
lastName
Dalton
KEY:
Element Node
Attribute Node
Root Node
Comment Node
Text / leaf node

XPath – Node relationships

- Nodes are related to other nodes
- Relationship types:
  - Ancestors
  - Parent
  - Children
  - Descendents
  - Siblings

Examples

```
<?xml version="1.0" encoding="UTF-8"?>
<addressbook>
  <person>
    <name>
      <firstname>Alex</firstname>
      <lastname>Ukovic</lastname>
    </name>
  </person>
</addressbook>
```

- Ancestors
  - <addressbook>, <person>, <name> are ancestors of <firstname> & <lastname>

Examples

```
<?xml version="1.0" encoding="UTF-8"?>
<addressbook>
  <person>
    <name>
      <firstname>Alex</firstname>
      <lastname>Ukovic</lastname>
    </name>
  </person>
</addressbook>
```

- Parent
  - <name> is the parent of <firstname> & <lastname>

Examples

```
<?xml version="1.0" encoding="UTF-8"?>
<addressbook>
  <person>
    <name>
      <firstname>Alex</firstname>
      <lastname>Ukovic</lastname>
    </name>
  </person>
</addressbook>
```

- Children
  - <firstname> & <lastname> are children of <name>

Examples

```
<?xml version="1.0" encoding="UTF-8"?>
<addressbook>
  <person>
    <name>
      <firstname>Alex</firstname>
      <lastname>Ukovic</lastname>
    </name>
  </person>
</addressbook>
```

- Descendents
  - <name>, <firstname> & <lastname> are descendents of <person>

Examples

```
<?xml version="1.0" encoding="UTF-8"?>
<addressbook>
  <person>
    <name>
      <firstname>Alex</firstname>
      <lastname>Ukovic</lastname>
    </name>
  </person>
</addressbook>
```

- Siblings
  - <firstname> and <lastname> are siblings

Axes

- An axis defines a node-set relative to the current node:
  - ancestor: Selects all ancestors (parent, grandparent, etc.) of the current node
  - ancestor-or-self: Selects all ancestors (parent, grandparent, etc.) of the current node and the current node itself
  - attribute: Selects all attributes of the current node
  - child: Selects all children of the current node
  - descendant: Selects all descendants (children, grandchildren, etc.) of the current node
  - descendant-or-self: Selects all descendants (children, grandchildren, etc.) of the current node and the current node itself
  - following: Selects everything in the document after the closing tag of the current node
  - following-sibling: Selects all siblings after the current node
  - namespace: Selects all namespace nodes of the current node
  - parent: Selects the parent of the current node
  - preceding: Selects everything in the document that is before the start tag of the current node
  - preceding-sibling: Selects all siblings before the current node
  - self: Selects the current node

XPath – Expressions

- Locate or identify a node or node-set
- Compute a value
- Include functions
- Test conditions
- Predicates

XPath – Location Expressions

- Context node
  - Node currently processed
  - Represented by “.”
  - Result of XPath query.
- Path relative to context node
- The node you seek is the last in the path

XPath – Location Expressions

- Different levels of an XML tree are separated by "/"

  /addressbook/person/name/lastname
- Use "@" to select an attribute

  /person/address[1]/@type
- Parent node represented by ".."

  ../telephone
- "//" identifies any or all nodes that match from context node

  //person

  person//@type

XPath – Operators and Functions

- You can use mathematical operators for calculations on any node values that can be represented as numbers
  - salary[@week='1'] + salary[@week='2']
  - (totalrevenue * taxrate) / capital
- Functions can calculate or test certain values
  - contain() tests if a certain string contains a string
  - position() returns the context node's position in the node-set
- Use operators to test conditions
  - actualsalary != promisedsalary
  - lastname <= "F"
  - @today > @expiry

XPath – Predicates Expressions

- Part of an expression that identifies a specific node due to be tested
- Are enclosed in square brackets

  /person[name/lastname='Marsh']

  /product[@expiry < '20050704']/name
- If the predicate evaluates to false for a particular node, the node is removed from those returned by the current location step.

- Download the **XPath Checker addon** from the following URL: https://addons.mozilla.org/en-US/firefox/addon/xpath-checker/
- Then create the address book XML document and try out the XPath expressions just discussed.