Chapter 7: XPath

Chapter 7 Objectives

- Ways of looking at an XML document, including the XPath Data Model
- How to visualize XPath and how the component parts of XPath syntax fit together to allow you to navigate around the XPath data model
- The XPath axes—the "directions" that are available to navigate around the XPath data model
- XPath 1.0 functions
- XPath 2.0 new functions and features

Ways of Looking at an XML Document

Three Specifications:

- XPath *
- DOM *
- XML Information Set

XPath Data Model

- **Root node**
- **Element nodes**
- **Attributes nodes**
- **Comment nodes**
- **Processing instruction nodes**
- **Text nodes**
- **Namespace nodes**

The Document Object Model

- Nodes used in the DOM are different
- Hierarchical set of tree nodes
- Manipulation techniques are different

Visualizing XPath

- *All legal XPath code can be called an expression.*
- *An XPath expression that returns a node-set is called a location path.*

```
<Book>
 <Chapter number="1">This is the first chapter</Chapter>
 <Chapter number="2">This is the second chapter</Chapter>
 <Chapter number="3">This is the third chapter</Chapter>
 <Chapter number="4">This is the fourth chapter</Chapter>
 <Chapter number="5">This is the fifth chapter</Chapter>
</Book>
```

Absolute location

Example: /Book/Chapter
/Book/Chapter[@number=2]

Relative location (depends on the current context)

Example: Chapter
Chapter[@number=2]

Context

- Context: the location of the node where the processor is currently situated.
- Context contains
  - Context node
  - Context position
  - Context size

```
<Book>
 <Chapter number="1">This is the first chapter</Chapter>
 <Chapter number="2">This is the second chapter</Chapter>
 <Chapter number="3">This is the third chapter</Chapter>
 <Chapter number="4">This is the fourth chapter</Chapter>
 <Chapter number="5">This is the fifth chapter</Chapter>
</Book>
```

Let's try XPath on XML Copy Editor


```
<xsl:stylesheet version="1.0“ xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >
<xsl:template match="/">
 <html>
  <head>
   <title>This shows the context position and context size.</title>
  </head>
  <body>
   <h3>Context position and context size demo.</h3>
   <xsl:apply-templates select="/Book/Chapter" />
  </body>
 </html>
</xsl:template>
<xsl:template match="Chapter">
 <xsl:if test="position()=2">
 <p>When the context node is the second <b>Chapter</b> element node then</p>
 <p>the context position is <xsl:value-of select="position()" /></p>
 <p>and the context size is <xsl:value-of select="last()" />.</p>
 <p>The text the <b> Chapter</b> element node contains is
‘<xsl:value-of select="." />‘.</p>
 </xsl:if>
</xsl:template>
</xsl:stylesheet>
```

Transformation Results

```
<?xml version="1.0" encoding="UTF-8"?>
<html>
  <head>
    <title>This shows the context position and context size.</title>
  </head>
  <body>
    <h3>Context position and context size demo.</h3>
    <p>
       When the context node is the second <b>Chapter</b> element node then
     </p>
    <p>
       the context position is 2</p>
    <p>
       and the context size is 5.
     </p>
    <p>
       The text the <b> Chapter</b> element node contains is
       'This is the second chapter'.
     </p>
  </body>
</html>
```


What is a Node?

- XPath has seven types of nodes:
  - Root node
  - Element node
  - Attribute node
  - Text node
  - Namespace node
  - Comment node
  - Processing Instruction node

Root Node

- **Every XML document has only one root node**
- **Root element is a child of root node**
- **Not visible in serialization of document**

Element, Attribute, and Text Nodes

- Element nodes have a name (QName) and a string value.

- Attribute nodes have a name and a string value.
- Associated with its parent element
- Not a child of the parent element

- A text node does not have a name.
- The string value of a text node is its character data.

Namespace Node

```
<library:Book xmlns:library="http://www.XMML.com/booknamespace">
  <chapter:Chapter xmlns:chapter="http://www.XMML.com/chapter"
    number="1">
   Some text content.
  </chapter:Chapter>
  <chapter:Chapter xmlns:chapter="http://www.XMML.com/chapter"
    number="2">
   Some different text content.
  </chapter:Chapter>
</library:Book>
```

How many namespace nodes are there in the document?

The book (p. 257) says 5. Or is it __?

Comment and Processing Instruction Node

You will see them but they are not represented in the XPath data model

XPath 1.0 Types

- **Four Expression Types**
  - **Boolean**
  - **Node-set**
  - **Number**
  - **String**

Booleans

- Use true() or false()
- Not true or false
  - so that we could use

```
<true>
        ... some content
</true>
```

Node-Sets

- Set of XPath nodes
- XPath node-set is unordered
  - but *document order* is followed when used in XSLT
- Example of document order

```
<PurchaseOrder>
 <Date>2005-01-01</Date>
 <To>XMML.com</To>
 <ShippingAddress>
  <Street>123 Any Street</Street>
  <City>Anytown</City>
  <State>AZ</State>
 </ShippingAddress>
 <ZipCode>12345</ZipCode>
</PurchaseOrder>
```


Numbers and Strings

Numbers

- **Floating point**
- **No real representation for integer**

Example

count( ) counts #nodes in a node set and should be integer, but …

Strings

- **Unicode character based**
- **E.g., dates are just strings**

Abbreviated and Unabbreviated Syntax

- XPath is not in XML format
- Often used as an attribute value

Unabbreviated and abbreivated examples

```
<xsl:value-of select="/child::Book/child::Chapter/child::Section" />
```

```
<xsl:value-of select="/Book/Chapter/Section" />
```

attribute::*attributename*

@attributename

XPath 1.0 Axes - Navigation

- child axis
- attribute axis
- ancestor axis
- ancestor-or-self axis
- descendant axis
- descendant-or-self axis
- following axis
- following-sibling axis
- namespace axis
- parent axis
- preceding axis
- preceding-sibling Axis
- self axis

- Usage: *axis*::*name*

Child Axis

```
<Invoice>
<Date>2004-01-02</Date>
<Item quantity="4">QD123</Item>
<Item quantity="5">AC345</Item>
</Invoice>
```

Examples

child::Item
Item

child::node()
node()

child::text()
text()

attribute Axis

Examples

attribute::*
@*

* : wildcard

attribute::security
@security

Try It Out

Example: PersonData.xml

```
<?xml version='1.0'?>
<PersonData>
  <Name DOB="1920/11/25">
    <FirstName>Jack</FirstName>
    <LastName>Slack</LastName>
  </Name>
</PersonData>
```

XPath expressions

/PersonData/Name/FirstName
/PersonData/Name/LastName
/PersonData/Name/@DOB

Other Commonly Used Axes

- **parent: ..**
- **self: .**
- **descendant-or-self: //**

- **More examples**

Structure of XPath Expressions

- **An axis**
- **A node test (element, attribute, node(), text())**
- **An optional predicate (condition)**
  - **[*i*]: *i*th child (same as [position()=*i*])**
  - **[*T*]: if subelement *T* exists**
  - **[*T*="*value*"]: if subelement *T* with *value* exists**
  - **[@*A*]: if attribute exists**
  - **[@*A*="*value*"]: if attribute *A* with *value* exists**
- **Can have a sequence of predicates**
- **Examples**
  - /PersonData/Name[2]
  - /PersonData/Name[MiddleName]
  - /PersonData/Name[FirstName="John"]
  - /PersonData/Name[@title]
  - /PersonData/Name[@title][FirstName="John"]
  - //*[FirstName]
  - //Name/node()
  - //Name/node()/Name

Structure of XPath Expressions

```
<?xml version="1.0" encoding="UTF-8"?>
<PersonData>
 <Name DOB="1920/11/25">
  <FirstName>Jack</FirstName>
  <LastName>Slack</LastName>
 </Name>
 <Name DOB="1988/9/20" title="Mr.">
  <FirstName>John</FirstName>
  <LastName>Doe</LastName>
 </Name>
 <Name DOB="1956/4/8">
  <FirstName>John</FirstName>
  <MiddleName>Silver</MiddleName>
  <LastName>Long</LastName>
  <knows>
   <Name>
    <FirstName>Jack</FirstName>
    <LastName>Slack</LastName>
   </Name>
  </knows>
 </Name>
</PersonData>
```

XPath 1.0 Boolean Functions

- boolean()
- false()
- lang()
- not()
- true()

XPath 1.0 Node-Set Functions

- count()
- id()
- last()
- local-name()
- name()
- namespace-uri()
- position()

XPath 1.0 Numeric Funtions

- ceiling()
- floor()
- number()
- round()
- sum()

String Functions

- concat()
- contains()
- normalize-space()
- starts-with()
- string()
- string-length()
- substring()
- substring-after()
- substring-before()
- translate()

Looking Forward to XPath 2.0

- *The latest version of the XPath 2.0 specification is located at*

  http://www.w3.org/tr/xpath20/.
- *Functions for XPath 2.0 are specified in a separate document located at*

  http://www.w3.org/TR/xpath-functions/ .
- *At the time of writing, further general information on XPath 2.0 is found at*

  http://www.w3.org/XML/Query.
- *Currently, the XPath link from*

  http://www.w3.org/ describes only XPath 1.0.

XPath 2.0

- **Revised XPath Data Model**
- **W3C XML Schema Data Types**
- **Additional XPath 2.0 Functions**
- **XPath 2.0 Features**
  - **Better String Handling**
  - **Better Date and Time Handling**
  - **Creating New Sequences**
  - **Conditional Logic**
  - **Ability to Call User-Defined Functions**
  - **More Node Tests**