XPath

XPath

- XPath is used to navigate through elements and attributes in an XML document.
- XPath is a language for finding information in an XML document.
- It uses path expressions to navigate in XML documents and contains a library of standard functions
- XPath is a W3C recommendation

XPath Path Expressions

- XPath uses path expressions to select nodes or node-sets in an XML document.
- XPath includes over 100 built-in functions - for string values, numeric values, date and time comparison, node and QName manipulation, sequence manipulation, Boolean values, and more.

XPath Nodes

- In XPath, there are seven kinds of nodes:
  - Element node,
  - Attribute node,
  - Text node,
  - Namespace node,
  - Processing-instruction node,
  - Comment node,
  - Document node.

XPath Terminology

- XML documents are treated as **trees of nodes**.
- The topmost element of the tree - **root element**.
- **Atomic values** - nodes with no children or parent.
- Relationship of nodes – Parent, children, siblings, Ancestors, Descendants

XPath Syntax

- XPath uses **path expressions** to select nodes or node-sets in an XML document.
- The node is selected by following **a path or steps.**

Sample XML document

- <?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>

Selecting Nodes

- **Path-Expressions**
  - ***nodename*** - Selects all nodes with the name "nodename"
  - ***/*** - Selects from the root node
  - ***//*** - Selects nodes in the document from the current node that match the selection no matter where they are
  - **.** - Selects the current node
  - **..** - Selects the parent of the current node
  - @ - Selects attributes

Examples – Path Expressions

- **bookstore** - Selects all nodes with the name "bookstore"
- /**bookstore** - Selects the root element bookstore
- **bookstore/book** - Selects all book elements that are children of bookstore
- **//book** - Selects all book elements no matter where they are in the document
- **bookstore//book** - Selects all book elements that are descendant of the bookstore element, no matter where they are under the bookstore element
- **//@lang** - Selects all attributes that are named lang

Selecting Unknown Nodes

- XPath wildcards can be used to select unknown XML elements.
  - ***** - Matches any element node
  - @***** - Matches any attribute node
  - **node()** - Matches any node of any kind

Selecting Unknown Nodes

- **/bookstore/*** - Selects all the child nodes of the bookstore element
- **//*** - Selects all elements in the document
- **//title[@*]** - Selects all title elements which have any attribute

Selecting Several Paths

- By using the | operator in an XPath expression we can select several paths.
- Eg:
- **//book/title | //book/price** -Selects both the title and price elements of all book elements
- **//title | //price** - Selects both the title and price elements in the document
- **/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

Predicates

- Predicates are used to find a **specific node or a node that contains a specific value.**
- **Eliminate unwanted items**
- A predicate is similar to an If/Then statement- predicate is TRUE, then the element will be selected, if the predicate is FALSE, it will be excluded.
- An XPath predicate is contained within square brackets [], and comes after the parent element to be tested.

Examples – Xpath with Predicates

- **/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[last()-1]** - Selects the last but one book element that is the child of the bookstore element
- **//title[@lang]** - Selects all the title elements that have an attribute named lang
- **//title[@lang='eng']** - Selects all the title elements that have an attribute named lang with a value of 'eng'
- **/bookstore/book[price>35.00]** - Selects all the book elements of the bookstore element that have a price element with a value greater than 35.00
- **/bookstore/book[price>35.00]/title** - Selects all the title elements of the book elements of the bookstore element that have a price element with a value greater than 35.00

XPath Axes

- An axis defines a node-set relative to the current node.
  - **self -** Selects the current node
  - **attribute** - Selects all attributes of the current node
  - **parent -** Selects the parent of the current node
  - **child** - Selects all children of the current node
  - **namespace -** Selects all namespace nodes of the current node

XPath Axes

- **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
- **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
- **preceding** - Selects all nodes that appear before the current node in the document, except ancestors, attribute nodes and namespace nodes
- **preceding-sibling** - Selects all siblings before the current node

Location Path Expression

- A location path can be absolute (starts with /) or relative.
  - **Absolute location path:** /step/step/...
  - **Relative location path:** step/step/...
- Each step is of form

  axisname::nodetest[predicate]
- Each step is evaluated against the nodes in the **current node-set.**
- A step consists of:
  - **an axis** (defines the tree-relationship between the selected nodes and the current node)
  - **a node-test** (identifies a node within an axis)
  - **zero or more predicates** (to further refine the selected node-set)

Examples

- **child::book** - Selects all book nodes that are children of the current node
- **attribute::lang -** Selects the lang attribute of the current node
- **child::*** - Selects all element children of the current node
- **attribute::*** - Selects all attributes of the current node
- **child::text() -** Selects all text node children of the current node
- **child::node()** - Selects all children of the current node
- **descendant::book** - Selects all book descendants of the current node
- **ancestor::book -** Selects all book ancestors of the current node
- **ancestor-or-self::book** - Selects all book ancestors of the current node - and the current node if it is a book node
- **child::*/child::price** - Selects all price grandchildren of the current node

XPath Operators

- An XPath expression returns either a node-set, a string, a Boolean, or a number.
- **XPath Operators that can be used in XPath expressions:**
- **|** - Computes two node-sets Eg: //book | //cd - Returns a node-set with all book and cd elements
- **+** - Addition
- **-** - Subtraction
- ***** - Multiplication
- **div** - Division - 8 div 4 = 2

XPath Operators

- **= - Equal** Eg: price=9.80 - true if price is 9.80 false if price is 9.90
- **!= - Not equal** Eg: price!=9.80 - true if price is 9.90 false if price is 9.80
- **< - Less than** Eg: price<9.80 - true if price is 9.00 false if price is 9.80
- **<= - Less than or equal to -** Eg: price<=9.80- true if price is 9.00 false if price is 9.90
- **> - Greater than** Eg: price>9.80 - true if price is 9.90 false if price is 9.80
- **>= - Greater than or equal to** - Eg: price>=9.80 - true if price is 9.90 false if price is 9.70
- **or -** Eg: price=9.80 or price=9.70 - true if price is 9.80 false if price is 9.50
- **and** - Eg: price>9.00 and price<9.90 - true if price is 9.80 false if price is 8.50
- **mod** - Modulus (division remainder) - Eg: 5 mod 2 = 1