# 爬蟲with Selenium 的大神器: Xpath

## What is XPath?

XPath is a major element in the XSLT standard.

XPath can be used to navigate through elements and attributes in an XML document.

|  |  |
| --- | --- |
| XPath | * XPath stands for XML Path Language * XPath uses "path like" syntax to identify and navigate nodes in an XML document * XPath contains over 200 built-in functions * XPath is a major element in the XSLT standard * XPath is a W3C recommendation |

## XPath Path Expressions

XPath uses path expressions to select nodes or node-sets in an XML document.

## XPath Terminology

### Nodes

In XPath, there are seven kinds of nodes: element, attribute, text, namespace, processing-instruction, comment, and document nodes.

XML documents are treated as trees of nodes. The topmost element of the tree is called the root element.

Look at the following XML document:

<?xml version="1.0" encoding="UTF-8"?>  
  
<bookstore>  
  <book>  
    <title lang="en">Harry Potter</title>  
    <author>J K. Rowling</author>  
    <year>2005</year>  
    <price>29.99</price>  
  </book>  
</bookstore>

Example of nodes in the XML document above:

<bookstore> (root element node)  
  
<author>J K. Rowling</author> (element node)  
  
lang="en" (attribute node)

### Atomic values

Atomic values are nodes with no children or parent.

Example of atomic values:

J K. Rowling  
  
"en"

### Items

Items are atomic values or nodes.

## Relationship of Nodes

### Parent

Each element and attribute has one parent.

In the following example; the book element is the parent of the title, author, year, and price:

<book>  
  <title>Harry Potter</title>  
  <author>J K. Rowling</author>  
  <year>2005</year>  
  <price>29.99</price>  
</book>

### Children

Element nodes may have zero, one or more children.

In the following example; the title, author, year, and price elements are all children of the book element:

<book>  
  <title>Harry Potter</title>  
  <author>J K. Rowling</author>  
  <year>2005</year>  
  <price>29.99</price>  
</book>

### Siblings

Nodes that have the same parent.

In the following example; the title, author, year, and price elements are all siblings:

<book>  
  <title>Harry Potter</title>  
  <author>J K. Rowling</author>  
  <year>2005</year>  
  <price>29.99</price>  
</book>

### Ancestors

A node's parent, parent's parent, etc.

In the following example; the ancestors of the title element are the book element and the bookstore element:

<bookstore>  
  
<book>  
  <title>Harry Potter</title>  
  <author>J K. Rowling</author>  
  <year>2005</year>  
  <price>29.99</price>  
</book>  
  
</bookstore>

### Descendants

A node's children, children's children, etc.

In the following example; descendants of the bookstore element are the book, title, author, year, and price elements:

<bookstore>  
  
<book>  
  <title>Harry Potter</title>  
  <author>J K. Rowling</author>  
  <year>2005</year>  
  <price>29.99</price>  
</book>  
  
</bookstore>

## The XML Example Document

We will use the following XML document in the examples below.

<?xml version="1.0" encoding="UTF-8"?>  
  
<bookstore>  
  
<book>  
  <title lang="en">Harry Potter</title>  
  <price>29.99</price>  
</book>  
  
<book>  
  <title lang="en">Learning XML</title>  
  <price>39.95</price>  
</book>  
  
</bookstore>

## Selecting Nodes

XPath uses path expressions to select nodes in an XML document. The node is selected by following a path or steps. The most useful path expressions are listed below:

|  |  |
| --- | --- |
| **Expression** | **Description** |
| *Nodename* | Selects all nodes with the name "*nodename*" |
| / | Selects from the root node(只找在父層級) |
| // | Selects nodes in the document from the current node that match the selection no matter where they are(無論在何層級皆可) |
| . | Selects the current node |
| .. | Selects the parent of the current node |
| @ | Selects attributes |

In the table below we have listed some path expressions and the result of the expressions:

|  |  |
| --- | --- |
| **Path Expression** | **Result** |
| Bookstore | Selects all nodes with the name "bookstore" |
| /bookstore | Selects the root element bookstore  **Note:** If the path starts with a slash ( / ) it always represents an absolute path to an element! |
| 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 |

## Predicates

Predicates are used to find a specific node or a node that contains a specific value.

Predicates are always embedded in square brackets.

In the table below we have listed some path expressions with predicates and the result of the expressions:

|  |  |
| --- | --- |
| **Path Expression** | **Result** |
| /bookstore/book[1] | Selects the first book element that is the child of the bookstore element.  **Note:** In IE 5,6,7,8,9 first node is[0], but according to W3C, it is [1]. To solve this problem in IE, set the SelectionLanguage to XPath:  *In JavaScript: xml*.setProperty("SelectionLanguage","XPath"); |
| /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 |
| /bookstore/book[position()<3] | Selects the first two book elements that are children of the bookstore element |
| //title[@lang] | Selects all the title elements that have an attribute named lang |
| //title[@lang='en'] | Selects all the title elements that have a "lang" attribute with a value of ‘en’ (記得:超坑!!!:  **包起屬性內容的是單引號**,單引號,單引號,不是雙引號!!!)(所以不像css selector屬性內容外不用加引號!) |
| /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 |

## Selecting Unknown Nodes

XPath wildcards can be used to select unknown XML nodes.

|  |  |
| --- | --- |
| **Wildcard** | **Description** |
| \* | Matches any element node |
| @\* | Matches any attribute node |
| node() | Matches any node of any kind |

In the table below we have listed some path expressions and the result of the expressions:

|  |  |
| --- | --- |
| **Path Expression** | **Result** |
| /bookstore/\* | Selects all the child element nodes of the bookstore element |
| //\* | Selects all elements in the document |
| //title[@\*] | Selects all title elements which have at least one attribute of any kind |

## Selecting Several Paths

By using the | operator in an XPath expression you can select several paths.

In the table below we have listed some path expressions and the result of the expressions:

|  |  |
| --- | --- |
| **Path Expression** | **Result** |
| //book/title | //book/price | Selects all the title AND price elements of all book elements |
| //title | //price | Selects all 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 |

## The XML Example Document

We will use the following XML document in the examples below.

<?xml version="1.0" encoding="UTF-8"?>  
  
<bookstore>  
  
<book>  
  <title lang="en">Harry Potter</title>  
  <price>29.99</price>  
</book>  
  
<book>  
  <title lang="en">Learning XML</title>  
  <price>39.95</price>  
</book>  
  
</bookstore>

## XPath Axes

An axis defines a node-set relative to the current node.

|  |  |
| --- | --- |
| **AxisName** | **Result** |
| 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 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 |
| self | Selects the current node |

## Location Path Expression

A location path can be absolute or relative.

An absolute location path starts with a slash ( / ) and a relative location path does not. In both cases the location path consists of one or more steps, each separated by a slash:

An absolute location path:  
  
/step/step/...  
  
A relative location path:  
  
step/step/...

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)

The syntax for a location step is:

axisname::nodetest[predicate]

### Examples

|  |  |
| --- | --- |
| **Example** | **Result** |
| 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 as well if it is a book node |
| child::\*/child::price | Selects all price grandchildren of the current node |

## XPath Operators

Below is a list of the operators that can be used in XPath expressions:

|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** | **Example** |
| | | Computes two node-sets | //book | //cd |
| + | Addition | 6 + 4 |
| - | Subtraction | 6 - 4 |
| \* | Multiplication | 6 \* 4 |
| div | Division | 8 div 4 |
| = | Equal | price=9.80 |
| != | Not equal | price!=9.80 |
| < | Less than | price<9.80 |
| <= | Less than or equal to | price<=9.80 |
| > | Greater than | price>9.80 |
| >= | Greater than or equal to | price>=9.80 |
| or | or | price=9.80 or price=9.70 |
| and | and | price>9.00 and price<9.90 |
| mod | Modulus (division remainder) | 5 mod 2 |