Xindice 1.2 XPath Guide

$Revision: 598114 $

by Vadim Gritsenko

NOTICE:

Table of contents

Note:
If you notice incorrectness in this documentation, please notify Xindice community. Your feedback will help create better documentation.

1. Querying the Database

Xindice currently supports XPath as a query language. Queries can be executed from within client application (please refer to the developers guide), through XML-RPC interface, or via a command line (please refer to the command line tool guide). This document describes what XPath queries are supported and what are the results of the query.

2. Sample Database

XPath queries and results described below were run against sample /db/addressbook collection. There are two documents in the addressbook collection.

Key: `address1`

```
<?xml version="1.0"?>
<person>
  <fname>John</fname>
  <lname>Smith</lname>
  <phone type="work">563-456-7890</phone>
  <phone type="home">534-567-8901</phone>
  <email type="home">jsmith@somemail.com</email>
  <email type="work">john@lovesushi.com</email>
  <address type="home">34 S. Colon St.</address>
  <address type="work">9967 W. Shrimp Ave.</address>
</person>
```

Key: `address2`

```
<?xml version="1.0"?>
<person>
  <fname>SlackJawedLocal</fname>
  <lname>Cletus</lname>
  <phone type="work">123-456-7890</phone>
  <phone type="home">234-567-8901</phone>
  <phone type="cell">345-678-9012</phone>
  <email type="home">cletus@hotmail.com</email>
  <email type="work">cletus@micrsquish.com</email>
  <address type="home">1234 S. Elm St.</address>
  <address type="work">4567 W. Pine St.</address>
</person>
```

3. Query for Document

One of the common usages for XPath is to obtain documents satisfying some criteria. Suppose we want to find everybody with the cell phone:

```
xindice xpath -c /db/addressbook -q "/person[phone/@type='cell']"
```

Result of the query will be one or more documents. If you have only two person entries in the collection, then only one result will be found:

```
<person xmlns:src="http://xml.apache.org/xindice/Query"
        src:col="/db/addressbook" src:key="address2">
  <fname>SlackJawedLocal</fname>
  <lname>Cletus</lname>
  <phone type="work">123-456-7890</phone>
  <phone type="home">234-567-8901</phone>
  <phone type="cell">345-678-9012</phone>
  <email type="home">cletus@hotmail.com</email>
  <email type="work">cletus@micrsquish.com</email>
  <address type="home">1234 S. Elm St.</address>
  <address type="work">4567 W. Pine St.</address>
</person>
```

4. Query for Element

Here we will issue a query resulting only in some elements from the document. Suppose we want to find everybody's home phone numbers:

```
xindice xpath -c /db/addressbook -q "/person/phone[@type='home']"
```

Result of the query will be all elements satisfying criteria from all documents.

```
<phone src:col="/db/addressbook" src:key="address1"
       xmlns:src="http://xml.apache.org/xindice/Query"
       type="home">534-567-8901</phone>
<phone src:col="/db/addressbook" src:key="address2"
       xmlns:src="http://xml.apache.org/xindice/Query"
       type="home">234-567-8901</phone>
```

5. Query for Text Node

With Xindice 1.1b4 and above, it is possible to query for text nodes. Each resulting text node will be wrapped into `result` element in the Query namespace.

```
xindice xpath -c /db/addressbook -q "/person[fname='John']/phone/text()"
```

Result of the query will be all phones for all Johns in the collection.

```
  <xq:result xmlns:xq="http://xml.apache.org/xindice/Query"
             xq:col="/db/addressbook"
xq:key="address1">563-456-7890</xq:result>
  <xq:result xmlns:xq="http://xml.apache.org/xindice/Query"
             xq:col="/db/addressbook"
xq:key="address1">534-567-8901</xq:result>
```

6. Query for String

XPath expressions with String result are also supported.

```
  xindice xpath -c /db/addressbook -q "string(/person[fname='John']/phone)"
```

Result of the query will be first phone number for all Johns in the collection, and empty result for each non-John.

```
  <xq:result xmlns:xq="http://xml.apache.org/xindice/Query"
             xq:col="/db/addressbook"
xq:key="address1">563-456-7890</xq:result>
  <xq:result xmlns:xq="http://xml.apache.org/xindice/Query"
             xq:col="/db/addressbook" xq:key="address2"></xq:result>
```

Note:

Because XPath is evaluated against each document, and because `string()` function always returns a result, such query will produce result from each document in the collection. In this example, result from second document is empty, as criteria `fname='John'` was not satisfied.

7. Query for Number

XPath expressions with Number result are also supported.

```
  xindice xpath -c /db/addressbook -q "count(/person/phone)"
```

This XPath will return count of phone numbers on file for each person. If person does not have phone numbers, result for this person will be `0.0`.

```
  <xq:result xmlns:xq="http://xml.apache.org/xindice/Query"
             xq:col="/db/addressbook" xq:key="address1">2.0</xq:result>
  <xq:result xmlns:xq="http://xml.apache.org/xindice/Query"
             xq:col="/db/addressbook" xq:key="address2">3.0</xq:result>
```

8. Using full text search in XPath

Full text search functionality is available in XPath queries in Xindice as an extention function.

Function: boolean ftcontains(string)

The ftcontains function returns true if text content of the context node matches full text query passed as the argument, otherwise it returns false. Text query syntax is the same as Lucene query syntax, except it does not use fields (using fields does not cause errors when parsing the query, but function will return false).

```
  xindice xpath -c /db/addressbook -q
"/person/address[@type='work']/text()[ftcontains('pine')]"
```

This XPath will return all work addresses of all people in database that contain word 'pine'.

```

  <xq:result xmlns:xq="http://xml.apache.org/xindice/Query"
             xq:col="/db/addressbook" xq:key="address2">4567 W. Pine
St.</xq:result>
```

Note:

Function results depend on the analyzer that is used to tokenize both node text content and a query. If LuceneIndexer exists for the collection being queried, same analyzer will be used for the ftcontains function to produce consistent results. If LuceneIndexer is not found, default analyzer will be used (at this time it is SimpleAnalyzer).