XQuery Summer Institute

Winona Salesky

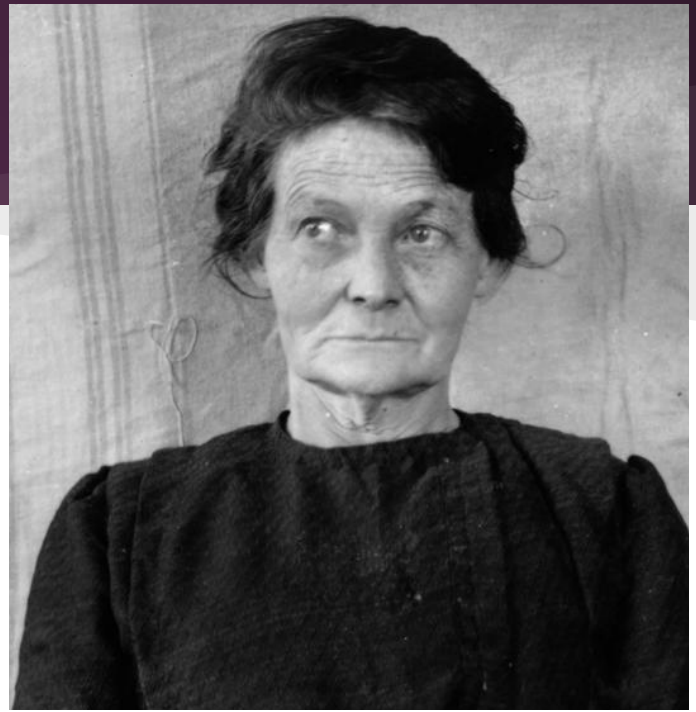
Me

Selected Projects

http://syriaca.org/

http://cdi.uvm.edu/

http://www.cpanda.org/cpanda/



Portrait of a seated older woman., Original version: 4" x 5" black and white negative in Tennie Toussaint Photographs, Folder 3, Special Collections, University of Vermont, Special Collections, University of Vermont Library, http://cdi.uvm.edu/collections/item/uvmsctous3016b (accessed May 27, 2014)

wsalesky.com

What is an XML database

“An XML database is a data persistence software system that allows data to be stored in XML format. These data can then be queried, exported andserialized into the desired format. XML databases are usually associated with document-oriented databases…

Native XML (NXD): the internal model of such databases depends on XML and uses XML documents as the fundamental unit of storage, which are, however, not necessarily stored in the form of text files.”

http://en.wikipedia.org/wiki/XML_database

Why eXist

- NoSQL document database (XML and Binary)
- Can serve as Web server
- Sophisticated document search engine
- Document editing and creation (XForms)
- Embeddable libraries for use in your own application.
- Open source

Introduction to eXist

- Tour of the Dashboard
- What's in the database?
  - /db - root
  - /db/system - system files, most importantly index configuration files
  - /db/apps - Your apps will be developed here.
- What is on your filesystem?
  - Configuration tools: EXIST-HOME/conf.xml, EXIST-HOME/webapp/WEB-INF/controller-config.xml
  - EXIST-HOME/webapp/Web-INF/data
  - EXIST-HOME/webapp/Web-INF/logs

Adding/Removing data

- Dashboard collection browser
- eXide
- WebDAV
- Java Admin Client
- oXygen database explorer
- REST API: PUT, POST
- Programmatically via xmldb:store() or xmldb:store-files-from-pattern()
- Ant build

Support

eXist mailing list:

http://exist-open.markmail.org/search/?q=

tei-eXist:

http://sourceforge.net/p/exist/mailman/exist-teixml/

XQuery and eXist

XQuery 3.0 is partially implemented:

- Try/catch
- Switch
- Higher order functions
- annotations
- group by

more: http://exist-db.org/exist/apps/doc/xquery.xml#xquery-30

Missing:

- "tumbling" and "sliding window" in FLWOR expressions
- "count" clause in FLWOR expressions
- "allowing empty" in FLWOR clause

Extension modules

Two kinds of modules, Java and XQuery.

Modules can be enabled from the conf.xml file:

```
<!--
    <module uri="http://exist-db.org/xquery/xslfo "
        class="org.exist.xquery.modules.xslfo.XSLFOModule">
        <parameter name="processorAdapter"
            value="org.exist.xquery.modules.xslfo.ApacheFopProcessorAdapter"/>
    </module>
-->
```

A few examples

- **xmldb:** for manipulating database contents
  - `xmldb:get-child-resources($collection)`
- **sm:** (security module) for handling users and permissions
  - `sm:chmod($path-to-doc, 'rwxr-xr-x')`
- **util:** A module convenient utility functions
  - `util:document-name($node), util:eval($path)`
- **kwic:** keyword in context (KWIC) highlighting functions
  - `kwic:summarize($hit, <config xmlns="" width="60"/>)`
- **ft:** Interacting with the full text index, built on the Lucene library
  - `//tei:sp[ft:query(.,'pox')]`
- **request:** HTTP requests
  - `request:get-parameter('records',10)`

function documentation: http://exist-db.org/exist/apps/fundocs/index.html

Optimizing XQueries

First things first:

Unlearning all those good habits:

Top ~~5~~ 3 XQuery for eXist tips

1. Shorter XPaths

```
tei:TEI/tei:teiHeader/tei:fileDesc/tei:titleStmt/tei:title
```

becomes:

```
//tei:title
```

~~2. Most selective filters first~~

~~3. Avoid unnecessary nested filters~~

Top 5 XQuery for eXist tips

4. Predicates are faster than where expressions

```
for $doc in collection('/db/apps/xq-institute/data/indexed-plays')
let $title := $doc//tei:titleStmt/tei:title/text()
where $doc//tei:body[ft:query(.,'pox')]
return $title
```

becomes:

```
for $doc in collection('/db/apps/xq-institute/data/indexed-plays')//tei:body
[ft:query(.,'pox')]
let $title := $doc//tei:titleStmt/tei:title/text()
return $title
```

Top 5 XQuery for eXist tips

5. Use group by rather than distinct-values

```
for $doc in collection('/db/apps/xq-institute/data/indexed-plays')
let $titles := $doc//tei:titleStmt/tei:title
let $genre := $doc//xqi:genre/text()
group by $genre
return
    <div>
        <div>{$genre}</div>
        <ul>{for $title in $titles/text() return <li>{$title}</li>}
</ul>
    </div>
```

One more

eXist does NOT do lazy evaluation.

Structure you queries with this in mind.

Exercises

1. Build simple browse list with title and date
   a. functions to use:
      i. collection()/xmldb:get-child-resources
      ii. base-uri()/doc()

2. Add sort:
   a. title, date *

eXist Indexes

Index types:

1. Structural Index
2. Range Indexes
3. Ngram Indexes
4. Full-text/Lucene Indexes
5. Constructed fields

Range Index

```
<create path="//tei:titleStmt/tei:title" type="xs:string"
/>
<create qname="tei:title" type="xs:string"/>
```

To query the range index:

```
//tei:titleStmt/tei:title = 'Hamlet'
```

Range Indexes 2.2

```
<range>
    <create qname="tei:title" type="xs:string"/>
    <create qname="tei:witness">
        <field name="xqy-id" match="@xml:id" type="xs:string"/>
      <field name="orig-pub" match="date" type="xs:string"/>
    </create>
</range>
```

To query the range index:

```
//tei:witness[@xml:id = 'shakespeare-online'][tei:date = '1598']
```

Query the new range index with the same value comparison operators and string functions as the original index.

Ngram Indexes

Create ngram index:

```
<create ngram="tei:title"/>
```

To query the range index:

```
//tei:titleStmt/ngram:starts-with(tei:title, ‘Henry’)
```

Full Text Indexes

Create index:

```
<lucene>
    <text qname="tei:title" boost="2.0"/>
    <ignore qname="tei:c"/>
    <text match="//tei:speaker/*"/>
</lucene>
```

To query the range index:

```
//tei:titleStmt/tei:title[ft:query(., 'Ham*')]
```

OR

```
<query>
    <term>pox</term>
</query>
```

Constructed fields

Add to index:

```
ft:index($uri, <doc>
    <field name="play" store="yes">{
        for $word in  $doc//tei:body/descendant::*/tei:w/text()
        return concat($word,' ')
    }</field>
</doc>)
```

Query a constructed field (field name ‘play’):

```
//ft:search('play:pox')/search
```

Example Index

Shakespeare xconf

Exercises

1. Basic
   a. Configure indexes for Shakespeare data
   b. Write a full text search using ft:query()

2. Advanced
   a. add subsequence() to limit results
   b. Add and query a constructed field
   c. Add KWIC to results **

User Defined Functions

- Recursion
- Reusable code
- More readable code

Anatomy of a Function

Prefix | Function name | Arguments, argument types | Result Type | Occurrence Indicator

```
local:list-plays($collection as xs:anyURI) as item()*
    {
        Function Body
    };
```

Types and Occurrence Indicators

Commonly used types:

- xs:string
- xs:decimal
- xs:integer
- xs:boolean
- xs:date
- element()
- node()
- item()

Occurrence Indicators

- ? zero or one items
- * zero or more
- + one or more

blank indicates a single item is required

Type Hierarchy: http://www.w3.org/TR/xpath-datamodel/type-hierarchy.png

Exercises

- **Simple exercise**
  - Write a function to construct a full text search using query element. Pass a search string to this function (from a global variable)

```
<query>
    <near/> | <phrase/> | <term/>
</query>
```

  - Call the function in main search expression

- **Advanced exercise:**
  - Write paging and count functions for search results

User Defined Modules

- Reuse functions across multiple queries
- Share functions across applications
- Smaller more modular queries

Anatomy of a module

Version declaration

```
xquery version "3.0";
```

Module namespace declaration

```
module namespace alchemy="http://xqueryinstitute.org/alchemy";
```

Import module declaration (s)

```
import module namespace config="http://localhost:
   8080/exist/apps/xq-institute/config" at "config.xqm";
```

Namespace declarations, options and variables

```
declare namespace httpclient="http://exist-db.
   org/xquery/httpclient";
```

Module functions

```
declare function alchemy:build-tone-node($fulltext as xs:string?)
   as node()*{
   Function Body
};
```

Exercises

Create module from query function

Import query module into search xquery

Call xquery function in search

Connecting to external resources

Selected examples:

- Send data to external API, process results:
  - Alchemy API for text analyisis
  - Integrate RSS feeds
  - Solr

Connecting to external resources

- Push your data to an API
  - Twitter
  - External blog
- Page scraping
- Build your own API to expose your data!

Functions to use

- Integration of external APIs

eXist's httpclient extension functions

```
httpclient:get($url as xs:anyURI, $persist as xs:boolean,
    $request-headers as element()?) as item()
httpclient:post($url as xs:anyURI, $content as item(),
    $persist as xs:boolean, $request-headers as element()?) as item()
```

See the rest

EXPath HTTP Client

```
http:send-request($request as element()?) as item()+
```

See the rest

Examples with our data:

- Page scraping dates
- Alchemy API for text analysis
- DPLA API for related title information

Examples

XQuery update

eXist documentation

http://exist-db.org/exist/apps/doc/update_ext.xml

W3C recommendation 2011

http://www.w3.org/TR/xquery-update-10/

Insert

Insert element or attribute into document/element matching expression

Syntax

```
update insert insert-exp
    (into | following | preceding ) target-expr
```

EXAMPLE

```
    update insert
      <genre xmlns="http://xqueryinstitute.org/ns"
>history</genre>
     following $doc//tei:text[last()]
```

Replace

Replace element/attribute or text node

Syntax

```
update replace expr with exprSingle
```

EXAMPLE

```
update replace $doc/xqi:genre with $genre
```

Value

Updates the content of the selected node/s

Syntax

```
update value expr with exprSingle
```

EXAMPLE

```
update value
   $doc//tei:publicationStmt/tei:date with
current-date()
```

Delete

Delete element or attribute matching expression (can not be used on doc root)

Syntax:

```
update delete expr
```

EXAMPLE

```
update delete $doc//tei:front
```

Rename

Rename nodes

```
update rename expr as expr2
```

EXAMPLE

```
update rename $doc//tei:title as ‘sub-title’
```

Exercise

Add new to a play, or all the plays

```
<change xmlns="http://www.tei-c.org/ns/1.0"
 who="http://xqueryinstitute.org/YOURNAME"
 when="{current-date()}">
   Adding change element For: XQueryInstitute
</change>
```

Query the plays to find your change

Delete your change