Using XPath in oXygen

James Cummings *April 2015*

1 XPath

Contents

1.1 Introduction

In this exercise we will be using a modified version of Hamlet, from the Bodleian First Folio project. The full project is available at `http://firstfolio.bodleian.ox.ac.uk`. You may also be interested in using the documentation for the XPath Tutorial at W3Schools as a reference:

- `http://www.w3schools.com/xpath`
- And the functions list at `http://www.w3schools.com/xpath/xpath_functions.asp`.

1.2 Starting up

- Start the oXygen XML Editor, and load up the file `hamlet.xml` file.
- Have a look through the file. There may be sections that you don't understand but note the following items:
  - The long `<teiHeader>` is quite detailed: `<fileDesc>` with a detailed `<titleSmt>`, `<publicationStmt>`, and `<sourceDesc>` which contains a `<msDesc>` element. There is also a `<listPerson>` documenting each of the characters in the play.
  - Each character's `<person>` element contains a standardised form of their character's name, as well as all the forms in which it is found in the text as a speaker designation.
  - The text itself is divided into modern acts and scenes regardless of whether these were indicated in the original. It uses a *@rend* attribute of 'notPresent' for those that have been added.
  - Each `<sp>` (speech) is marked and has a *@who* attribute that points to characters in the `<listPerson>`in the header.
- Clearly you don't have time to read through the whole file, but have a glance through some of it to make sure you understand its layout.

1.3 Using the XPath Search in the toolbar

- Assuming you haven't changed your toolbar setup, you should have and 'XPath 2.0' toolbar in the upper-left of the oXygen editor. (If you have got rid of this, ask and we'll show you how to restore it.)
- This toolbar should look like this:

It contains:

  - a dropdown menu on the left for selecting the version of XPath you are using (for now use 'XPath 2.0')
  - a box in which to type xpath queries for the current file
  - a setting dropdown menu enabling you to change options (these should be unchecked by default)

- Let's say you want to find all of the speeches in the text. You know each one of these is in its own `<sp>` element. So we should be able to find these with: `//sp`. (You may have to press enter twice, as it will be prompting you to select 'sp' as one of the elements it knows exists there.)
- When you do a search in oXygen it lists the hits in a window at the bottom of the editor. This contains a description, the XPath location, what file it is in, and location. Click on one of the results and you will be taken to that result. This can be an easy way to navigate through a large file. Notice what it lists as the 'Description' and that next to the heading it also lists how many results there are. **How many speeches are there in Hamlet as a whole?**
- We could also get this using the `count(//sp)` function in the XPath box, but then wouldn't have access to each one. Try it and see the difference!
- Can you find the speech which doesn not have a *@who* attribute? Perhaps using the 'not()' function in a filter might help? e.g. `//sp[not(@who)]`
- Each character's `<person>` record contains a variety of information including a unique id (*@xml:id*), a standardised form of the name, each form it is found in speaker identifiers, and some of them also have an indication of their gender, their age, and socio-economic status (that aren't present in the project's original files). Notice that the *@xml:id* values are in a structured format 'F-' (for Folio), 'ham-' (for the play) and 'ham' (for the character). Or for Horatio: 'F-ham-hor'. This means they are unique across the entire collection of plays.

```
<!-- Hamlet's record --><person xml:id="F-ham-ham">
 <persName type="standard">Hamlet, son of the former king and nephew to the
present king</persName>
 <persName type="form">Ha.</persName>
 <persName type="form">Ham.</persName>
 <persName type="form">Hamlet.</persName>
 <persName type="form">Hem.</persName>
 <sex value="1"/>
 <age value="1"/>
 <socecStatus>noble</socecStatus>
</person>
<!-- Horatio's record -->
<person xml:id="F-ham-hor">
 <persName type="standard">Horatio, friend to Hamlet</persName>
 <persName type="form">Hor.</persName>
 <persName type="form">Hora.</persName>
 <persName type="form">Horat.</persName>
 <persName type="form">Hor. & Mar.</persName>
```

```
 <sex value="1"/>
 <age value="1"/>
 <socecStatus>noble</socecStatus>
</person>
```

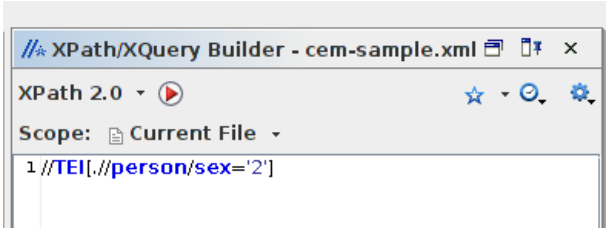
If we wanted to find all the characters we could do this be looking for `//person`. **How many characters are there in Hamlet?**

- Those characters that have a `<sex>` element with a value of '2' are female. We could find this by searching for `//person/sex[@value='2']`. **How many nominally female characters are there in Hamlet?**

1.4 Using the XPath Builder

- If you type too long of an XPath in the XPath search box it will suggest you use the XPath Builder and offer to open it for you. Otherwise you can open it by going to `Window -> Show View -> XPath/XQuery Builder`. This will open up a window on the right-hand side of oXygen which enables you to type longer XPaths. (You can also run XQueries in them as well, but that is a slightly more complicated XPath-based query language for XML Databases.)

- The XPath/XQuery Builder looks like:



- On the left-hand side is a drop-down menu for choosing the version of XPath, and next to this a 'play' button to run your search. Beneath this you can control the scope of the XPath search to be one of:
  - Current File (Default and what you should be using
  - Project
  - Selected project resources
  - All opened files
  - Current DITA Map hierarchy
  - Working sets
  - And you can set some options

- To the right-hand side there is a star to allow you to 'favourite' queries, along with a drop-down list of those you have favourited. There is another drop-down list of recent queries. Finally, there is a drop-down settings menu.

- Try the queries you have made already in the XPath Builder and mark at least one of them as a 'favourite'.

- **Try to find all speeches by Horatio.** To do this you need to use his unique person id which is 'F-ham-hor'. Each speech looks like:

```
<sp who="#F-ham-hor">
 <speaker rend="italic">Hor.</speaker>
 <l>Friends to this ground.</l>
</sp>
```

What is the best way to find this? Try `//sp[@who='#F-ham-hor']`. **How many speeches of Horatio does this find?**

- Why might that not be accurate? Have a look at speeches like:

```
<sp who="#F-ham-hor #F-ham-mar">
 <speaker rend="italic">Both.</speaker>
 <l>We will my Lord.</l>
</sp>
```

In cases like this there are two speakers delivering a single speech. We can't know who else might be speaking with Horatio (though it does happen always to be Marcellus).

- The 'contains()' function is a very useful one for finding nodes whose text contains some other bit of text. It is used by saying something like `//l[contains(lower-case(.), 'lord')]`. Note how this uses two nested functions. 'lower-case()' modifies the input to be lowercase. The 'contains()' function takes two strings, it checks to see if the second string is present in the first. In this case the first string is the lowercase contents of '.' which, since this is in a square-bracket predicate or filter, is the contents of the 'l' (metrical line) element.
- **How many speeches does Horatio really speak?** To find this you'll need to do something like `//sp[contains(@who, '#F-ham-hor')]`
- You can return attribute values. **What is the *@who* attribute value for those speeches that mention the word 'death'?**
- You can navigate up and down the document hierachy from any point to any other point. Consider how you might find this: A list of the distinct-values of the standardised name for the characters whose single-spoken speeches mention the string of characters 'death' spelled precisely that way. To do this we must do several things:

  1. the standardised form of the name. This is in each `<person>` element as `persName[@type='standard']`
  2. the *@who* attribute of any speeches containing 'death'. You've just found this above with something like: `//sp[contains(., 'death')]/@who`
  3. The *@xml:id* of a `<person>` element perhaps with a '#' prefixing it. We could do this with `//person[concat('#', @xml:id) ]`
  4. the distinct values of all of the results. We can use the `distinct-values()` function for this.

  If we put this all together we might get something like:
  `distinct-values(//person[concat('#',@xml:id)=//sp[contains(., 'death')]/@who]/persName[@type='standard'])` To reiterate this in more detail:

  distinct-values(...) This overarching function will make a list of all the distinct values of the result.

  //person[...] This will find any `<person>` elements at any level.

concat('#', @xml:id) This concatenates the string '#' to the *@xml:id* value of the `<person>`.

=//sp[contains(., 'death')]/@who] This compares the previous string to the *@who* attribute of any `<sp>` which (when it and its children are taken as a string) contains the characters 'death'. The final closing ']' is that opened by filtering the `<person>` above.

/persName[@type='standard'] Back inside the context of the person, return the `<persName>` whose *@type* attribute is equal to 'standard'.

Go ahead and try it!

1.5 Using XPath with Find and Replace

- You can also use XPath to narrow the search context in the `Find -> Find/Replace` window.
- If you search for the word 'death' (without the single quotes) and click 'Find All', you will find many hits. This should be '38' or '37 if 'Case sensitive' is checked.
- You could narrow this in several ways. One way is the use the XPath box immediately under the Replace with: box. This allows you to filter your search to only be in the XPath you have selected. In this case, let's limit it to prose passages: put `//p`. If you 'Find All' again you should find a lot less.
- You should also look under the 'Enable XML search options' box (by checking it) to expose a sub-menu of more fine-grained places to look.
- Notice that if you close the 'Find/Replace' window and load it up again (control-f on windows, probably command-f on mac), that the options you had selected before are still selected. Always check this when doing a Find or Replace.
- Experiment finding more bits of the document through increasingly more complicated XPaths! Use the function list at: `http://www.w3schools.com/xpath/xpath_functions.asp`.

2 Application to your own project

You may wish to explore texts from your own project using XPath. Can you use it to discover something you did not already know?