XTraining

Part 3: Functions

Regine Heberlein 5/22/2018

To Begin

- Download the latest test file from the ADG team drive
- Download AC123 from the ADG team drive
- Open both in oXygen

Homework:
Concatenated string v. Sequence

```
count((//c[@level='series'])[1]/c)
|| ' ' ||
count((//c[@level='series'])[2]/c)
```

v.

```
for $i in
(count((//c[@level='series'])[1]/c),
count((//c[@level='series'])[2]/c))
return $i
```

Aside for the Determined: What Happens to Quantified Expressions Executed on an Empty Sequence

every $x in //controlaccess satisfies $x/pirate → *false*

BUT

every $x in //controlaccess/pirate satisfies $x → *true*

"Note that if the input sequence is empty, the "some" expression will always be false, while the "every" expression will always be true. This may not be intuitive to everyone, but it is logical—the "every" expression is true if there are no counter-examples; for example, it's true that every unicorn has one horn, because there are no unicorns that don't have one horn. Equally, and this is where the surprise comes, it is also true that every unicorn has two horns." –Michael Kay, XSLT 2.0 and XPath 2.0, 4th ed. (2008)

Challenge

The Dread Pirate Roberts has hidden treasure on the island of AC387. Find it and tally the number of pearls it contains as well as how much they are worth!

How many types of pearls?

```
count(
//cave[@mark='x']/chest
        [@owner='Dread Pirate Roberts']
                /treasure[@type='pearls']//pearl
)
```

OR

```
for $i
in //cave[@mark='x']/chest[@owner='Dread Pirate Roberts']/treasure[@type='pearls']//pearl return $i/count
```

Sum up the count value of each of the four types:

```
//cave[@mark='x']/chest[@owner='Dread Pirate Roberts']/treasure[@type='pearls']//pearl[1]/count +

//cave[@mark='x']/chest[@owner='Dread Pirate Roberts']/treasure[@type='pearls']//pearl[2]/count +

//cave[@mark='x']/chest[@owner='Dread Pirate Roberts']/treasure[@type='pearls']//pearl[3]/count +

//cave[@mark='x']/chest[@owner='Dread Pirate Roberts']/treasure[@type='pearls']//pearl[4]/count
```

Shorter version: sum()

```
sum(
//cave[@mark='x']/chest
        [@owner='Dread Pirate Roberts']
        /treasure[@type='pearls']//pearl/count
)
```

What denominations is their value measured in?

```
distinct-values(
        //cave[@mark='x']/chest
                [@owner='Dread Pirate Roberts']
                        /treasure[@type='pearls']
                                //pearl/value/@unit
)
```

How many instances of dubloons?

```
count(
        //cave[@mark='x']/chest
                [@owner='Dread Pirate Roberts']
                        /treasure[@type='pearls']
                                //pearl/value
                                        [@unit='dubloon']
)
```

Sum up the value of each of the three instances of dubloon

```
//cave[@mark='x']/chest[@owner='Dread Pirate Roberts']/treasure[@type='pearls']//pearl[value/@unit='dubloon'][1]/value +
//cave[@mark='x']/chest[@owner='Dread Pirate Roberts']/treasure[@type='pearls']//pearl[value/@unit='dubloon'][2]/value +
//cave[@mark='x']/chest[@owner='Dread Pirate Roberts']/treasure[@type='pearls']//pearl[value/@unit='dubloon'][3]/value
```

Return the value of the single instance of silver dollar

```
//cave[@mark='x']/chest[@owner='Dread Pirate Roberts']/treasure[@type='pearls']//pearl[value/@unit='silver dollar']/value
```

Put it all together like so:

```
'The Dread Pirate Roberts has hidden '
|| (
//cave[@mark='x']/chest[@owner='Dread Pirate
Roberts']/treasure[@type='pearls']//pearl[1]/count +
//cave[@mark='x']/chest[@owner='Dread Pirate
Roberts']/treasure[@type='pearls']//pearl[2]/count +
//cave[@mark='x']/chest[@owner='Dread Pirate
Roberts']/treasure[@type='pearls']//pearl[3]/count +
//cave[@mark='x']/chest[@owner='Dread Pirate
Roberts']/treasure[@type='pearls']//pearl[4]/count
) ||
' pearls in his cave. They are valued at '
|| (
//cave[@mark='x']/chest[@owner='Dread Pirate
Roberts']/treasure[@type='pearls']//pearl[value/@unit='dubloon'][1]/value +
//cave[@mark='x']/chest[@owner='Dread Pirate
Roberts']/treasure[@type='pearls']//pearl[value/@unit='dubloon'][2]/value +
//cave[@mark='x']/chest[@owner='Dread Pirate
Roberts']/treasure[@type='pearls']//pearl[value/@unit='dubloon'][3]/value
) ||
' dubloons and ' ||
//cave[@mark='x']/chest[@owner='Dread Pirate
Roberts']/treasure[@type='pearls']//pearl[value/@unit='silver dollar']/value ||
' silver dollars all told. Arrgh!'
```

Same, using sum()

```
'The Dread Pirate Roberts has hidden '
||
sum(//cave[@mark='x']/chest[@owner='Dread Pirate
Roberts']/treasure[@type='pearls']//pearl/count)
||
' pearls in his cave. They are valued at '
||
sum(//cave[@mark='x']/chest[@owner='Dread Pirate
Roberts']/treasure[@type='pearls']//pearl[value/@unit='dubloon']/val
ue)
||
' dubloons and '
||
sum(//cave[@mark='x']/chest[@owner='Dread Pirate
Roberts']/treasure[@type='pearls']//pearl[value/@unit='silver
dollar']/value)
||
' silver dollars all told. Arrgh!'
```

Functions, at Long Last!

count()
not()
distinct-values()
normalize-space()
max()
contains()
last()
position()
substring() / substring-after() / substring-before()
number()

How to use functions

```
//controlaccess/*/position()

//controlaccess/*[position()=last()]

//p/text()[contains(normalize-space(.), 'no permission')]

//c/@id/substring-after(., 'AC387_c')

max(//unittitle/string-length())

count(//c/@id/substring-after(., 'AC387_c') ! number()[.>40])

//unitdate/translate(normalize-space(.), '-', '*')
```

//unitdate[not(@normal)] //unitdate[not(@normal)]/text()
[not(contains(., "undated"))]
distinct-values(//container[@type='box']/text())
better: [not(matches(., '\p{Z}'))]

- How many components are in AC387?
  count(//c)
- How many components that lack a unitdate?
  count(//c[not(did/unitdate)])
- How many components whose unitdate lacks the normal attribute?
  count(//c[did/unitdate[not(@normal)]])
- What types of containers are present in AC387?
  distinct-values(//container/@type)
- What is the highest box number in AC387?
  max(distinct-values(//container[@type='box' and number()]))
  ☝ NB: this will not catch non-numerical box numbers!

Aside for the Determined: What Happens to `not( )` v. `!=` Executed on an Empty Sequence

(//container)[1] returns "27"

(//container)[1]!=28 → TRUE

not((//container)[1]=28) → TRUE

BUT:

(//container)[60] returns an empty sequence

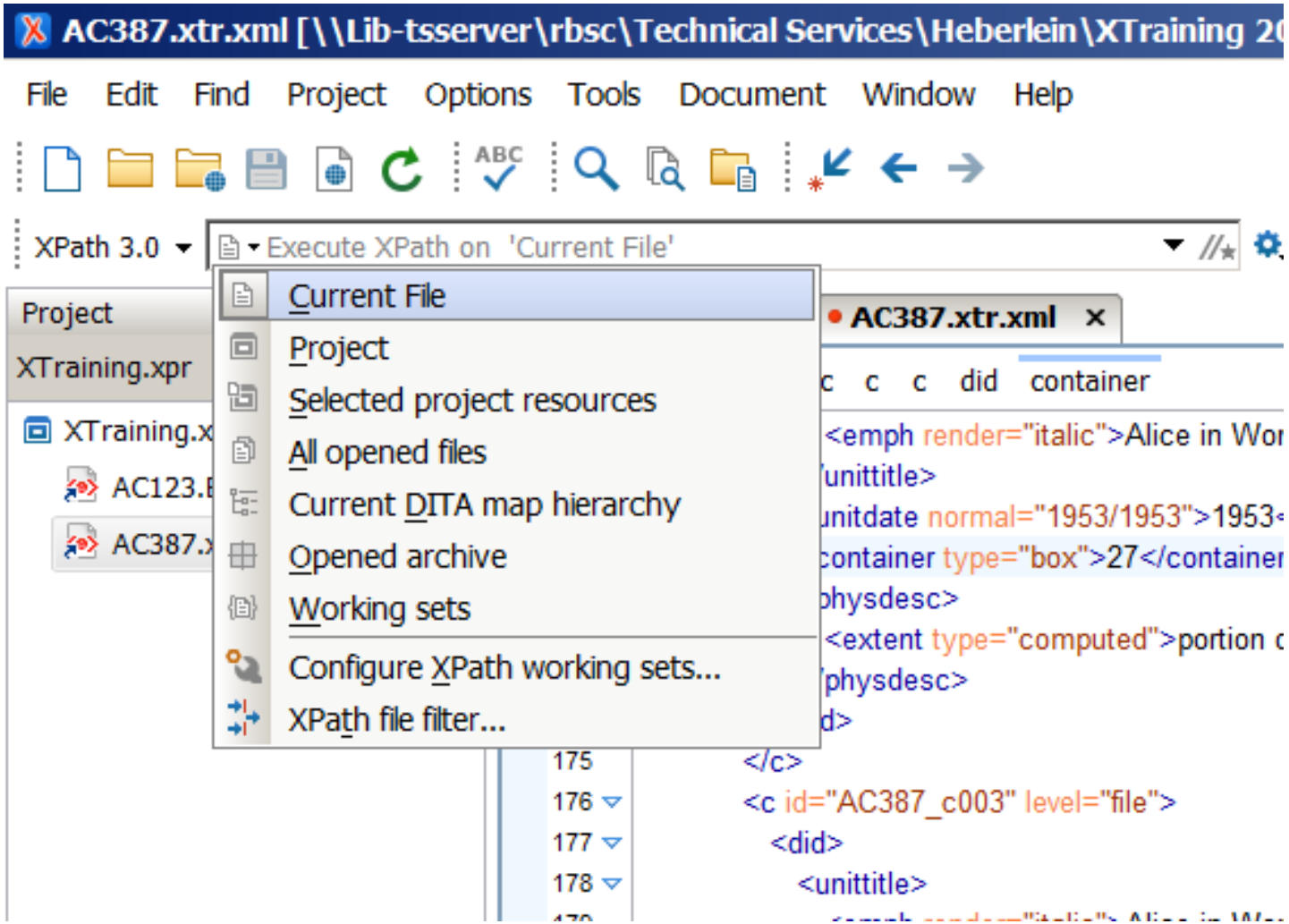
(//container)[60]!=28 → FALSE

not((//container)[60]=28) → TRUE

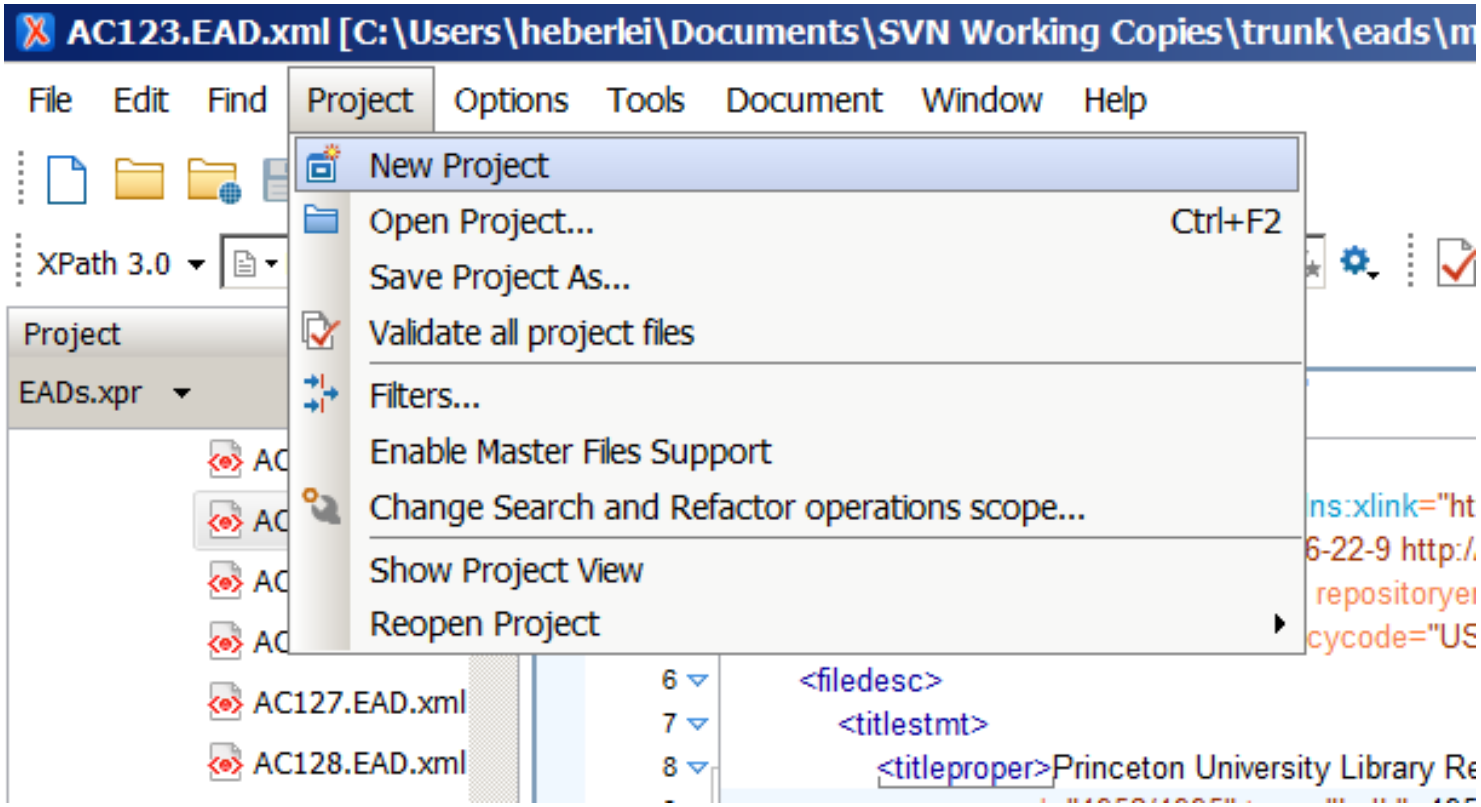
Yet More Functions!

- https://www.w3.org/TR/xpath-functions-31/
- http://www.xqueryfunctions.com/

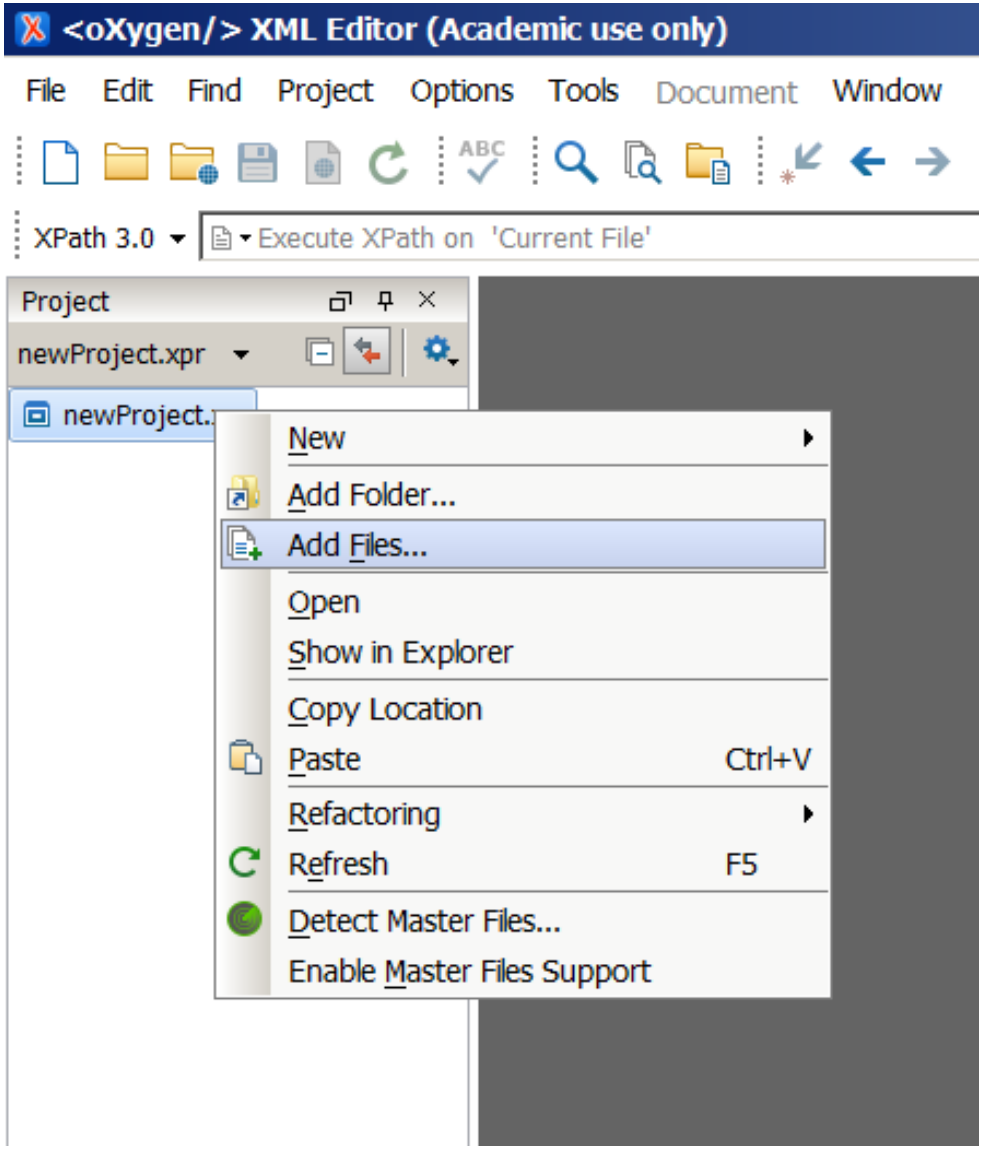
Useful Tool: XPath Scope



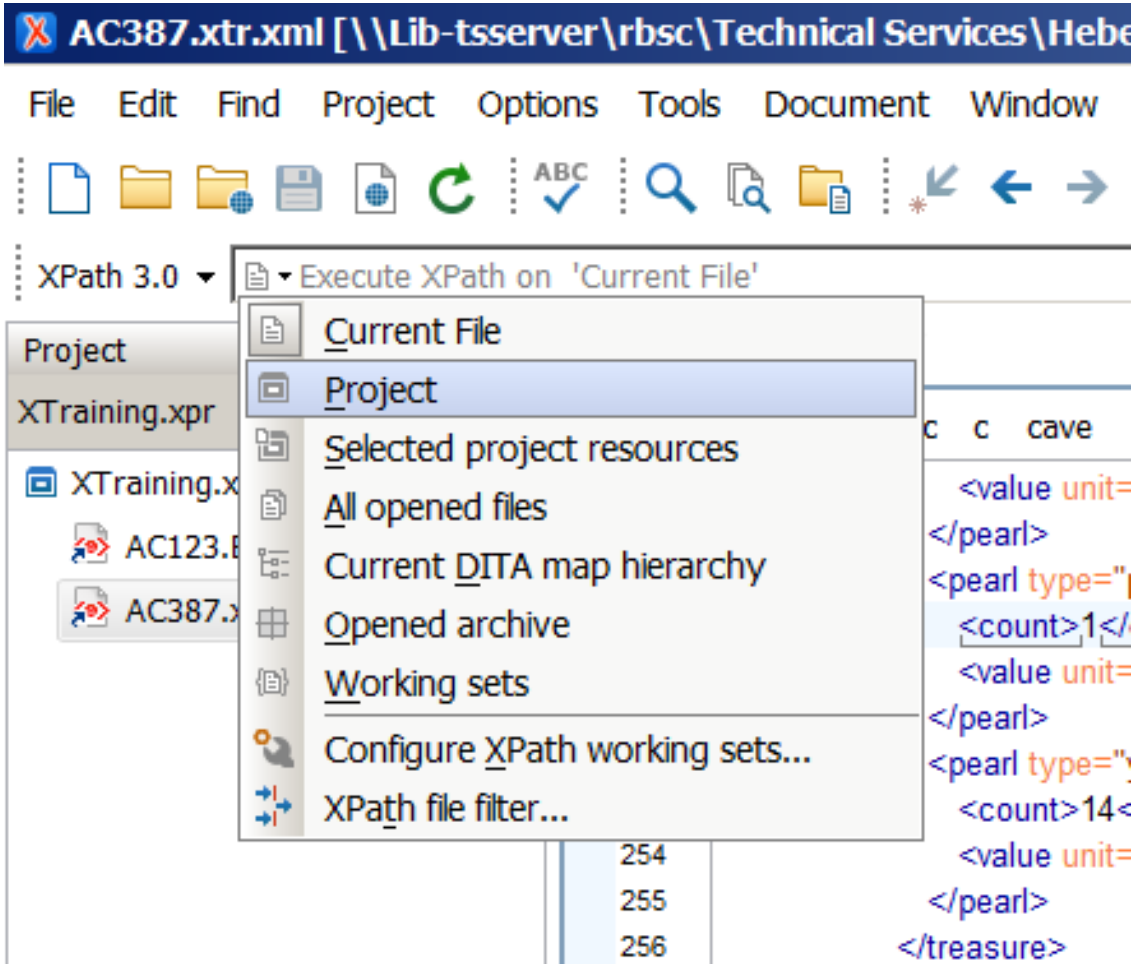
Useful Tool: Project



Useful Tool: Project



Project Data Analysis



Project Data Analysis

- //did[not(unitdate)]
- //did[unittitle = unitdate]
- //c[contains(string(.[not(child::c)]), 'accession') and not(unitid[@type='accession number'])]
- //c[matches(string(.[not(child::c)]), 'accession number', 'i') and not(unitid[@type='accession number'])]

Project Data Analysis

- find extents lacking the unit attribute

  //extent[not(@unit)]

- find components whose unittitle is the same as their untidate (you'll need parentheses!)

  //c[did/(unittitle=unitdate)]

- find controlaccess terms with a local source attribute

  //controlaccess/*[@source='local']

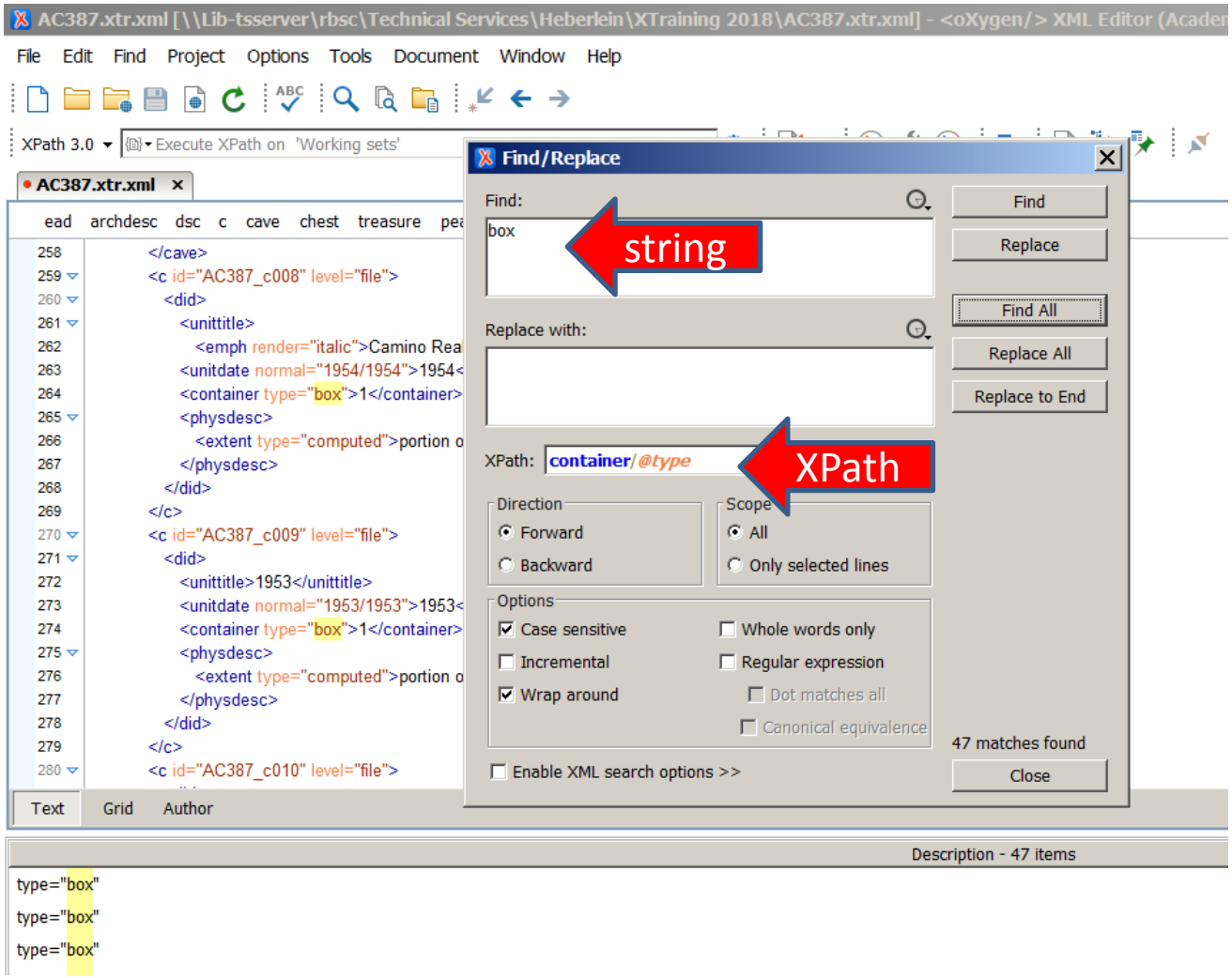
- Return the date the finding aid was encoded, if it falls within 2008. (hint: use substring)

  //profiledesc/creation/date[substring(@normal, 1, 4)='2008']

- Return the call number of the finding aid encoded in 2008

  //profiledesc[creation/date[substring(@normal, 1, 4)='2008']]/../eadid

Useful Tool: Query a Directory or Project from Find/Replace Dialog



Let’s get real (sort of)

Replace all box-type containers with basket-type containers.

1. Use find/replace
2. In "find", enter the search string
3. In XPath, return the node you want to change
4. **Test** by clicking on "find all"
5. In "replace", enter the replacement string

Challenge

Across all our repos…

- how many finding aids have no dsc?
- how many finding aids have more than one dsc?
- how many finding aids contain extents with a unit attribute?
- what are the unique values used in the unit attribute?