Infrastructure - Decision #8765

Consider changing how BaseSolrFieldXPathTest works

2019-02-13 00:36 - Bryce Mecum

Status:	Closed	**Start date:**	2019-02-13
Priority:	Low	**Due date:**	
Assignee:	Bryce Mecum	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
Milestone:	None		

Description

Ran into a weird thing while expanding the https://repository.dataone.org/software/cicore/trunk/cn/d1_cn_index_processor/src/test/java/org/dataone/cn/index/SolrFieldXPathEmlTest.java to test an EML 2.2.0 doc.

SolrFieldXPathEmlTest compares values extracted via various subprocessors to a set of expectations stored in a HashMap<String, String>() (of form <fieldName, expectedValue>). This data structured limits expectations to one value per field. Forever ago, in r7985,

```
r7985 | sroseboo | 2012-03-23 12:12:53 -0800 (Fri, 23 Mar 2012) | 1 line

initial commit of search index support for parsing FGDC science metadata docs.

Index: src/test/java/org/dataone/cn/index/BaseSolrFieldXPathTest.java
```

support was added for testing multiple expectations for a single field by defining a convention of smushing multiple values into a single string, separated by two # characters (##). For example,

```
eml210Expected.put("project", "Random Project Title##Another Random Project Title");
```

would test the project field for two values, "Random Project Title" and "Another Project Title" not a literal "Random Project Title##Another Random Project Title". I imagine this was picked because it's rare to see a ## in a metadata record which seems reasonable.

I ran afoul of this today because I wanted to test an expectation for a field with a # in it and I couldn't because it was being split when I didn't want it to be. Why did a single # break things when the convention above is a double #? Because BaseSolrFieldXPathTest.java splits the expectation string using StringUtils.split like this:

```
StringUtils.split(expectedForField, "##") // Where expectedForField might equal "Random Project Title##Another Random Project Title"
```

According to https://commons.apache.org/proper/commons-lang/apidocs/org/apache/commons/lang3/StringUtils.html#split-java.lang.String-java.lang.String-, the second arg to StringUtils.split, separatorChars, is "the characters used as the delimiters, null splits on whitespace" which tells me we're using it wrong. I think the method we needed to use was StringUtils.splitByWholeSeparator which correctly splits only on double #.

I could just change the line of code and move on but that breaks a ton (98) of tests. Before I did that, I wanted to ask what others thought. I see a few routes:

1. Change the code to correctly split only on ## and not # *and* update all the tests I break
2. Do 1 *and* use a different separator to something more future proof. I suggest "&&" because that'd be invalid in XML outside a CDATA section.

3. Change how the expectations get tested so field expectations can have a one to many relationship. This'd take some time so I'd opt for (1) or (2) instead

Any preferences out there?

History

#1 - 2019-02-13 19:20 - Bryce Mecum

Brought this up on Slack and got feedback from Jing, Rob, and Dave. Majority was in favor of (2). I'll start on that today.

#2 - 2019-02-13 19:21 - Bryce Mecum

- % Done changed from 0 to 30

- Priority changed from Normal to Low

- Assignee set to Bryce Mecum

- Status changed from New to In Progress

#3 - 2019-02-14 00:39 - Bryce Mecum

- % Done changed from 30 to 100

- Status changed from In Progress to Closed

I started swapping out the separators from #/## to && and ran into an edge case where some assertions needed to test multiple URLs with trialing & symbols (weird), like "http://example.com/ASDF&" which would make the assertion "http://example.com/ASDF&&&http://example.com/ASDF" which doesn't split properly. I decided to try <> instead, first confirming no test assertions include that string and things look all good now.

I think the switch is probably pretty agreeable to people but let me know if it isn't.