# Infrastructure - Bug #2459

## Index Processing encountering invalid XML character (0x19)

2012-03-08 18:25 - Skye Roseboom

| | | | |
|---|---|---|---|
| **Status:** | Closed | **Start date:** | 2012-03-08 |
| **Priority:** | Normal | **Due date:** | |
| **Assignee:** | Skye Roseboom | **% Done:** | 100% |
| **Category:** | d1_indexer | **Estimated time:** | 0.00 hour |
| **Target version:** | Sprint-2012.11-Block.2.2 | | |
| **Milestone:** | CCI-1.0.0 | **Story Points:** | |
| **Product Version:** | * | | |

**Description**

Noticed this starting 3-7 while rebuilding search index.

It appears an invalid character  (0x19, EOM - ascii end of medium control character) is entering system metadata document during MN Tier 3 tests:

[Fatal Error] :1:206: An invalid XML character (Unicode: 0x19) was found in the element content of the document.
ERROR indexer.XPathDocumentParser - An invalid XML character (Unicode: 0x19) was found in the element content of the document.
org.xml.sax.SAXParseException: An invalid XML character (Unicode: 0x19) was found in the element content of the document.
at org.apache.xerces.parsers.DOMParser.parse(Unknown Source)
at org.apache.xerces.jaxp.DocumentBuilderImpl.parse(Unknown Source)
at javax.xml.parsers.DocumentBuilder.parse(DocumentBuilder.java:124)
at org.dataone.cn.indexer.XPathDocumentParser.generateSystemMetadataDoc(XPathDocumentParser.java:376)
at org.dataone.cn.indexer.XPathDocumentParser.process(XPathDocumentParser.java:203)
at org.dataone.cn.index.processor.IndexTaskUpdateProcessor.process(IndexTaskUpdateProcessor.java:26)
at org.dataone.cn.index.processor.IndexTaskProcessor.processTask(IndexTaskProcessor.java:82)
at org.dataone.cn.index.processor.IndexTaskProcessor.processIndexTaskQueue(IndexTaskProcessor.java:72)
at org.dataone.cn.utility.SolrIndexBuildTool.processIndexTasks(SolrIndexBuildTool.java:134)
at org.dataone.cn.utility.SolrIndexBuildTool.generateIndexTasksAndProcess(SolrIndexBuildTool.java:117)
at org.dataone.cn.utility.SolrIndexBuildTool.refreshSolrIndex(SolrIndexBuildTool.java:101)
at org.dataone.cn.utility.SolrIndexBuildTool.main(SolrIndexBuildTool.java:89)
ERROR indexer.XPathDocumentParser - Could not load System metadata for ID:
testMNodeTier3:2012679267486_common-bmp-doc-example-□□□□□□□□□□□□□□□

Not sure how this invalid XML character is entering the xml document or why it doesn't surface in other d1 cn processing.

I trapped the text output provided by
http://cn-dev-2.dataone.org/cn/v1/meta/testMNodeTier3:2012679267486_common-bmp-doc-example-□□□□□□□□□□□□□□□ and created a test resource system metadata document and can successfully parse, index and retrieve the record from solr.  So the browser seems to be removing this character from the output stream?  Seems to also exclude the pid string name from the source of the error as it parses and indexes ok from the unit test.

Brief discussion of validity of 0x19 in xml:
http://stackoverflow.com/questions/1325379/hexadecimal-0x19-is-an-invalid-character

### History

#### #1 - 2012-03-08 18:30 - Skye Roseboom

*- Subject changed from Index Processing encountering invalid XML characters to Index Processing encountering invalid XML character (0x19)*

#### #2 - 2012-03-08 19:19 - Skye Roseboom

*- Status changed from New to In Progress*

#### #3 - 2012-03-16 02:32 - Dave Vieglais

*- Target version changed from Sprint-2012.09-Block.2.1 to Sprint-2012.11-Block.2.2*

*- Position set to 12*

#### #4 - 2012-03-21 16:42 - Skye Roseboom

Updated the index task update processor strategy to catch this SAX Parse exception. When this exception is caught, the systemMetadata object is refreshed from hzCast datastore and parsing is re-attempted. This has always succeeded in correcting the issue in testing. Issue effects less than 1% of d1 integration test data - haven't seen it occur yet on 'real' data.

Behavior described above seems to indicate an issue in index task persistence and/or system metadata serialization.

#### #5 - 2012-03-29 21:09 - Skye Roseboom

Could not reproduce in HSQL - only effected test class when switched to Postgresql.

Seems to be an issue between hibernate (jpa implementation) and postgres. Changing the JPA annotation from LOB to a plain column with length completely fixes the issue. This causes postgres to generate a varchar column instead of the non-ansii sql 'text' column type. This 'text' column type and how it was being converted/handled by hibernate may be the source of this bug.

Updated IndexTask.java JPA annotation for the 'systemMetadata' property.

#### #6 - 2012-03-30 21:04 - Skye Roseboom

*- Status changed from In Progress to Closed*

Resolved.

Found a JPA entity mapping for large text fields that works in postgres and hsql databases with 2-byte UTF-8 encoded characters:

@Column(columnDefinition = "TEXT")

Creates a 'text' column (better than varchar) that properly hydrates back into java object with UTF-8 encoding and no bad xml characters.