Saxon - Bug #4476

Type error evaluating (fn:collection(...))

2020-03-10 11:28 - Octavian Nadolu

Status:	Closed	**Start date:**	2020-03-10
Priority:	Normal	**Due date:**	
Assignee:	Michael Kay	**% Done:**	100%
Category:	Internals	**Estimated time:**	0:00 hour
Sprint/Milestone:		**Spent time:**	0:00 hour
Legacy ID:		**Fix Committed on Branch:**	9.9, trunk
Applies to branch:	9.9, trunk	**Fixed in Maintenance Release:**	10.0, 9.9.1.8

Description

I get a type error if I make a transformation using the following stylesheet ans Saxon 9.9.1.7. It works with Saxon 9.9.1.5 and 9.9.1.6. I think is related with: https://saxonica.plan.io/issues/2749

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    version="2.0">
    <xsl:template match="/">
        <xsl:variable name="url" select="'file:///D:/projects/eXml/samples/docbook/v5/out'"/>
        <xsl:variable name="FILELIST" select="
collection(concat($url, '?recurse=yes;select=*.indexterms'))"/>
        <xsl:variable name="terms" select="for $n in $FILELIST/*/* return $n"/>
        <xsl:value-of select="$terms"/>
    </xsl:template>
</xsl:stylesheet>

Type error evaluating (fn:collection(...)) in xsl:variable/@select on line 6 column 110
 of Untitled.xsl:
  XPTY0019: The required item type of the first operand of '/' is node(); the supplied value
  xs:base64Binary("
PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiPz48aW5kZXggeG1sbnM9Imh0dHA6Ly93d3cub3h5Z2VueG1sLm
NvbS9ucy93ZWJoZWxwL2luZGV4Ii8+") is an atomic value
  In template rule with match="/" on line 4 of Untitled.xsl
The required item type of the first operand of '/' is node(); the supplied value xs:base64Binary("
PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiPz48aW5kZXggeG1sbnM9Imh0dHA6Ly93d3cub3h5Z2VueG1sLm
NvbS9ucy93ZWJoZWxwL2luZGV4Ii8+") is an atomic value
```

History

#1 - 2020-03-16 11:18 - Michael Kay

I've reproduced this using repo/samples/scm?select=*.scm as the collection URI - basically a directory that contains XML files, expects them to be treated as XML, but doesn't use a known file extension or HTTP media type that identifies them as XML. I think they were previously being recognized as XML by virtue of sniffing the initial bytes of the file. This changed with bug #4382; we were leaving the stream connection open after doing the sniffing, which led to exhaustion of the limit on open streams, so the design approach had to change.

The failure is because we haven't recognized this as an XML resource; we're delivering it as an unparsed Base64Binary object, which obviously can't appear on the lhs of the "/" operator.

The problem with the new (post-#4382) approach is we either have to open the file twice (once to do the sniffing, once to actually read the content), or we have to be prepared to defer recognising the file type until the file is actually opened.

We're treating the file as binary because that's the default for an unrecognized file extension. One possibility is to change the default to XML; since collection() historically only returned XML files, that's the option that most people are likely to be using, at least with directory-based collections which are perhaps the most common kind.

#2 - 2020-03-16 12:43 - Michael Kay

- Category set to Internals

- Status changed from New to In Progress

- Assignee set to Michael Kay

- Priority changed from Low to Normal

- Applies to branch trunk added

I have implemented (and am testing) the following solution:

(a) the default media type registered in the configuration is changed to "application/unknown". This can be changed using a call such as configuration.registerFileExtension("", "application/xml")

(b) the default media type is used the URI scheme is "file" and a media type cannot be inferred from the file extension.

(c) if the media type inferred from examination of the URI is "application/unknown", we allocate a new kind of Resource called UnknownResource. The getItem() method on UnknownResource sniffs the content (using URLConnection.guessContentTypeFromStream()) and then delegates to a more specific resource type obtained by calling configuration.getResourceFactoryForMediaType().

#3 - 2020-03-16 13:24 - Michael Kay

In testing this, I have one unit test failing (testCollectionWithHttp, which is using a collection catalog accessed over HTTP). This does not appear to be a new failure, the same test is failing under 9.9 where the changes have not yet been applied.

The failure occurs because inferStreamEncoding fails with an IOException "mark/reset not supported" while doing obtainCharacterContent(). Since we're reading JSON here, we could really assume an encoding of UTF-8.

A further complication is that the error isn't cleanly reported, because of the multi-threaded execution.

Setting the JSONResource encoding to UTF-8, rather than attempting to infer it, solves the particular test case - though it leaves the more general issue that with HTTP resources, inferring the encoding when not given in the HTTP headers isn't working.

#4 - 2020-03-16 13:31 - Michael Kay

- Status changed from In Progress to Resolved

- Fix Committed on Branch 9.9, trunk added

#5 - 2020-03-16 13:40 - Radu Coravu

It's great you added some more logic to the detection. Sometimes I would use collection() to iterate ".dita" files and probably with the latest Saxon changes they were considered non XML, right?

#6 - 2020-03-16 13:43 - Michael Kay

Yes, I think that before the fix for bug #4382 we were sniffing the file to detect content type if the file extension was unknown, but after the fix that stopped working, at least for file:// URIs. It's now reinstated. But you might like to consider registering additional file extensions (such as .dita) with the configuration.

#7 - 2020-03-17 09:18 - Octavian Nadolu

Maybe you can add a query parameter for the "collection()" function, that will allow you to specify the default content type.

#8 - 2020-03-17 13:48 - O'Neil Delpratt

- Fixed in Maintenance Release 10.0 added

Bug fix applied in the Saxon 10 major release.

#9 - 2020-03-17 13:59 - Octavian Nadolu

How gen I get this fix? Do you intend to have a minor release for Saxon 9.9 to include this fix? Is this fix added in Saxon 10?

#10 - 2020-03-17 15:42 - Michael Kay

The fix is included in 10.0; it's also committed on the 9.9 branch so will appear in the next maintenance release of 9.9.

#11 - 2020-03-17 15:45 - Octavian Nadolu

This is great. Thanks very much.

#12 - 2020-03-18 13:00 - Radu Coravu

One mention about the usage of this utility:

java.net.URLConnection.guessContentTypeFromStream(InputStream)

it works only if the input stream supports marking (java.io.InputStream.markSupported()). Maybe to be 100% sure the stream has mark support, it could have been wrapped in a buffered input stream: stream = new BufferedInputStream(stream); just to make sure this works no matter what stream

implementation is used.

#13 - 2020-03-18 13:23 - Michael Kay

- Status changed from Resolved to In Progress

Reopened, because that's a good suggestion.

#14 - 2020-03-19 11:28 - Michael Kay

For 10.0 only, I have added a query parameter content-type to the collection URI format recognised by directory and JAR collections; if present, this takes precedence over (and inhibits) any guessing of content type from the file name or file content.

This needs documenting at http://www.saxonica.com/documentation/index.html#!sourcedocs/collections

Also: there is a query parameter that appears to be implemented but undocumented, and has no effect: unparsed=yes|no. I'm going to get rid of it from the code. I think the idea was that you could retrieve unparsed XML if you wanted, but the implementation wasn't completed; you can now achieve this effect using /my/dir?select=*.xml;content-type=text/plain.

#15 - 2020-03-19 12:57 - Michael Kay

- Status changed from In Progress to Resolved

Having made this further change (including tests and documentation), closing the issue once again.

#16 - 2020-10-22 18:16 - O'Neil Delpratt

- Status changed from Resolved to Closed

- % Done changed from 0 to 100

- Fixed in Maintenance Release 9.9.1.8 added

Bug fix applied on the Saxon 9.9.1.8 maintenance release.