# Saxon - Bug #4560

## Failure running fn:transform() from XQuery under Saxon-HE

2020-05-26 11:23 - Michael Kay

| | | | |
|---|---|---|---|
| **Status:** | Closed | **Start date:** | 2020-05-26 |
| **Priority:** | Normal | **Due date:** | |
| **Assignee:** | Michael Kay | **% Done:** | 100% |
| **Category:** | XQuery conformance | **Estimated time:** | 0:00 hour |
| **Sprint/Milestone:** | | **Spent time:** | 0:00 hour |
| **Legacy ID:** | | **Fix Committed on Branch:** | 10 |
| **Applies to branch:** | 10 | **Fixed in Maintenance Release:** | 10.3 |

**Description**

Transferred from help forum, post #7895

When I run the XQuery code

```
transform(
  map {
    'initial-match-selection' :
      transform(
          map {
            'source-location' : 'input1.xml',
            'stylesheet-location' : 'sheet1.xsl',
            'delivery-format' : 'raw'
          }
       )?output,
    'stylesheet-location' : 'sheet1.xsl'
  }
)?output
```

with some input XML and a not schema-aware stylesheet using Saxon-HE 10.1J with the command line java -cp saxon-he-10.1.jar net.sf.saxon.Query -t .\chain-two-xslts2.xq it succeeds while using Saxon-EE 10.1J with java -cp saxon-ee-10.1.jar net.sf.saxon.Query -t .\chain-two-xslts2.xq fails with an error "Cannot use a schema-validated source document unless the stylesheet is schema-aware".

### History

#### #1 - 2020-05-26 11:31 - Michael Kay

My findings so far:

(a) It works under Saxon-EE, even when running without a license file

(b) Both under Saxon-HE 10.0 and 10.1 it fails in a different way from that reported:

```
java.lang.NullPointerException
 at net.sf.saxon.s9api.Xslt30Transformer.transform(Xslt30Transformer.java:379)
 at net.sf.saxon.functions.TransformFn.call(TransformFn.java:802)
 at net.sf.saxon.expr.FunctionCall.iterate(FunctionCall.java:543)
 at net.sf.saxon.expr.Expression.evaluateItem(Expression.java:852)
 at net.sf.saxon.expr.LookupExpression.iterate(LookupExpression.java:316)
 at net.sf.saxon.expr.parser.Evaluator$7.evaluate(Evaluator.java:199)
 at net.sf.saxon.expr.SystemFunctionCall.evaluateArguments(SystemFunctionCall.java:449)
 at net.sf.saxon.expr.FunctionCall.iterate(FunctionCall.java:541)
```

Line 379 is in fact on an error path where it is trying to call the ErrorListener; it's presumably failing because we don't have an ErrorListener, we have an ErrorReporter. To find out what the primary error is (the one it's trying to report) I'll need to build an HE-only project for debugging.

#### #2 - 2020-05-26 12:20 - Martin Honnen

Can my result of the failure with EE be caused by my particular license which is not a full EE license but EE-T? But the stylesheet does nothing but

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="3.0">

<xsl:mode on-no-match="shallow-copy" streamable="yes"/>

<xsl:template match="/*">
  <xsl:next-match/>
  <xsl:comment select="
'Processed at', current-dateTime(), 'with xsl:supports-streaming', system-property('xsl:supports-streaming')"
/>
</xsl:template>

</xsl:stylesheet>
```

still not sure at what point the XQuery or XSLT creates "a schema-validated source document".

#### #3 - 2020-05-26 13:02 - Michael Kay

Yes, it could be that the EE-T license is an issue.

The problem is probably xs:untyped vs xs:anyType. If there's a node annotated xs:anyType that counts as a "schema-validated node" for the purpose of this error message. If XSLT can handle typed nodes (as it can with an EE-T license) then under appropriate conditions it can generate xs:anyType rather than xs:untyped, and if XQuery is non-schema-aware (as it is with an EE-T license) then it will reject xs:anyType. I think with this kind of license there are probably going to be interoperability issues passing data from XSLT to XQuery or vice versa.

Shallow-copy implicitly uses validation=preserve, which despite its name gives shallow-copied nodes a type of xs:anyType (that's because the content of the element might contain typed nodes, which would be invalid for xs:untyped). So nodes passed to the query will be annotated xs:anyType and the non-schema-aware XQuery isn't prepared to handle that.

In principle it would be better if non-schema-aware code ignored type annotations rather than rejecting them. But it would be hard to implement that reliably.

Can you work around it by using a default template rule with <xsl:copy validation="strip">?

#### #4 - 2020-05-26 13:24 - Michael Kay

*- Fix Committed on Branch 10 added*

As regards the NullPointerException:

(a) there are three places in the code that AbstractXsltTransformer.getErrorListener() is called. These should be changed to use the new getErrorReporter() interface instead.

(b) The documentation for  AbstractXsltTransformer.getErrorListener() says that if no user-supplied error listener is available, a system-defined one will be returned. This isn't what the code is doing; it is returning null. We should change the code to match the documentation, by returning an ErrorListener that wraps the ErrorReporter.

I have applied these changes

#### #5 - 2020-05-26 13:35 - Martin Honnen

Michael Kay wrote:

> Yes, it could be that the EE-T license is an issue.
>
> The problem is probably xs:untyped vs xs:anyType. If there's a node annotated xs:anyType that counts as a "schema-validated node" for the purpose of this error message. If XSLT can handle typed nodes (as it can with an EE-T license) then under appropriate conditions it can generate xs:anyType rather than xs:untyped, and if XQuery is non-schema-aware (as it is with an EE-T license) then it will reject xs:anyType. I think with this kind of license there are probably going to be interoperability issues passing data from XSLT to XQuery or vice versa.
>
> Shallow-copy implicitly uses validation=preserve, which despite its name gives shallow-copied nodes a type of xs:anyType (that's because the content of the element might contain typed nodes, which would be invalid for xs:untyped). So nodes passed to the query will be annotated xs:anyType and the non-schema-aware XQuery isn't prepared to handle that.
>
> In principle it would be better if non-schema-aware code ignored type annotations rather than rejecting them. But it would be hard to implement that reliably.
>
> Can you work around it by using a default template rule with <xsl:copy validation="strip">?

Yes, using <xsl:copy validation="strip"> instead of relying on xsl:mode on-no-match="shallow-copy" fixes the problem and now that I have understood the reasons I will be able to apply that.

#### #6 - 2020-09-14 01:00 - Michael Kay

*- Status changed from New to Resolved*

I'm going to mark this resolved. The effect of the patch was to give an error message attempting to explain the situation, rather than crashing. I don't think it's easy to do much about the error (a non-schema-aware stylesheet doesn't accept schema-typed nodes as input); we could get better at explaining it in error messages or documentation, but I don't think it's going to be a common enough problem to justify the effort.

#### #7 - 2020-10-28 18:57 - O'Neil Delpratt

Bug fix applied in the Saxon 10.3 maintenance release

#### #8 - 2020-10-28 19:13 - O'Neil Delpratt

*- Status changed from Resolved to Closed*

*- % Done changed from 0 to 100*

*- Fixed in Maintenance Release 10.3 added*