Saxon - Bug #4693

Error message is different when comparing Saxon-HE with Saxon-PE.

2020-08-25 16:49 - Gerben Abbink

Status:	Closed	**Start date:**	2020-08-25
Priority:	Low	**Due date:**	
Assignee:	Michael Kay	**% Done:**	100%
Category:	Diagnostics	**Estimated time:**	0:00 hour
Sprint/Milestone:		**Spent time:**	0:00 hour
Legacy ID:		**Fix Committed on Branch:**	10, trunk
Applies to branch:	10, trunk	**Fixed in Maintenance Release:**	10.3

Description

I am using test file "conflict-resolution-0702.xsl".

I am calling Saxon-HE from the command line, like this:

- java.exe -jar saxon-he-10.1.jar -xsl:conflict-resolution-0702.xsl -s:whatever.xml

I get this warning:

```
Warning at procedure xsl:unnamed on line 0
  XTDE0540  Ambiguous rule match for /root
Matches both "element()" on line 16 of file:///C:/folder/conflict-resolution-0702.xsl
and "(element()|(comment()|(text()|processing-instruction())))" on line 20 of << THIS LINE IS DIFF
ERENT
  file:///C:/folder/conflict-resolution-0702.xsl
```

Then i cali Saxon-PE from the command line, like this:

- java.exe -jar saxon-pe-10.1.jar -xsl:conflict-resolution-0702.xsl -s:whatever.xml

I get this warning:

```
Warning at procedure xsl:unnamed on line 0
  XTDE0540  Ambiguous rule match for /root
Matches both "element()" on line 16 of file:///C:/folder/conflict-resolution-0702.xsl
and "(processing-instruction()|(text()|(element()|comment())))" on line 20 of << THIS LINE IS DIFF
ERENT
  file:///C:/folder/conflict-resolution-0702.xsl
```

These warning look the same but they are slightly different. Is there a reason for this, is this an error?

This is the content of conflict-resolution-0702.xsl:

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="2.0">

<?spec xslt#unprefixed-qnames?>
<!-- Testing xsl:xpath-default-namespace -->
<xsl:template match="u:doc" xmlns:u="http://some.uri/">
   <out xsl:xpath-default-namespace="http://some.uri/">
      <xsl:apply-templates select="foo"/>
   </out>
</xsl:template>

<xsl:template match="u:foo" xmlns:u="http://some.uri/">
    <xsl:text>Match-of-qualified-name</xsl:text>
</xsl:template>
```

```
<xsl:template match="*">
    <xsl:text>Match-of-wildcard</xsl:text>
</xsl:template>

<xsl:template match="node()">
    <xsl:text>Match-of-node-type</xsl:text>
</xsl:template>

</xsl:stylesheet>
```

This is the content of conflict-resolution-0702.xsl:

```
<root></root>
```

History

#1 - 2020-08-25 17:05 - Michael Kay

The pattern is represented internally as an instance of MultipleNodeKindTest with uType = an integer with 4 bits set representing node kinds ELEMENT, TEXT, COMMENT, PI.

The toString() method on this pattern constructs a Java Set from this uType and then iterates over the set; the order of items in a set is undefined.

I've no idea why the result should be different in the two cases, but it's intrinsically unpredictable.

So not a bug, I think.

(It has, however, been suggested that we should try to retain the original pattern for use in error messages, rather than reconstituting it from its internal compiled form.)

#2 - 2020-08-25 19:43 - Gerben Abbink

The problem lies with UType's decompose:

```
    public Set<PrimitiveUType> decompose() {
        Set<PrimitiveUType> result = new HashSet<PrimitiveUType>();
        for (PrimitiveUType p : PrimitiveUType.values()) {
            if ((bits & (1<<p.getBit())) != 0) {
                result.add(p);
            }
        }
        return result;
    }
```

The order of a HashSet is quite random: "It makes no guarantees as to the iteration order of the set; in particular, it does not guarantee that the order will remain constant over time." https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/HashSet.html

A LinkedHashSet has a fixed order. https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/LinkedHashSet.html

#3 - 2020-09-07 12:59 - Michael Kay

- Category set to Diagnostics

- Status changed from New to In Progress

- Assignee set to Michael Kay

Until 9.6, we maintained originalText as a property of Pattern. I think we dropped it when we started creating patterns indirectly, by first creating an Expression and then converting the Expression to a Pattern. But I have reinstated it, making it work properly this time, and this means that the warning messages on ambiguities will now be phrased in terms of the original pattern "as written".

This won't work when a stylesheet is exported and reimported. I would like to think that ambiguities like this will only survive while stylesheets are under development, and not when they are in production. When a stylesheet is imported from a SEF file, we will still represent the pattern text in diagnostics by decompiling it.

#4 - 2020-09-07 13:48 - Michael Kay

The changes to implement this were a bit more extensive than expected: the copy() method on all subclasses of Pattern needs to change to retain the originalText property, and the toString() method needs to change to use it (I've done this by introducing a new method reconstruct() which decompiles the compiled pattern to a string, and Pattern.toString() calls this when necessary.

I've also improved the location information:

```
Warning at procedure xsl:unnamed on line 0
```

"Procedure" is now replaced by "mode"; xsl:unnamed is replaced by "(unnamed)", and we don't output the line number if it's zero (in this case the mode is implicitly declared so there's no relevant line number). The actual line numbers of the two ambiguous rules are present in the message.

#5 - 2020-09-08 13:54 - Michael Kay

- Status changed from In Progress to Resolved

- Applies to branch 10, trunk added

- Fix Committed on Branch 10, trunk added

#6 - 2020-10-28 18:57 - O'Neil Delpratt

Bug fix applied in the Saxon 10.3 maintenance release

#7 - 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