XSLT 3.0 xsl:fork instruction

Balisage Conference 2026, Open Mic presentation

6 August

Author : Mukul Gandhi

E-mail : gandhi.mukul at gmail.com

Affiliation : Apache Xalan-J committer, and PMC member

Introduction

This document, illustrates details about XSLT 3.0 xsl:fork instruction, which is newly introduced within W3C XSLT 3.0 specification.

"An xsl:fork instruction may be used to split an input dataset (which could be XML, JSON or a combination of both) to produce different independent results, with a single read on an input data by an XSL transformation processor". An xsl:fork instruction is specified to do this efficiently by processing parallelly, the xsl:fork's contained XSLT instructions.

An XSL content model for xsl:fork instruction is described within the following document sections.

An xsl:fork instruction is particularly useful to use when using XSLT 3.0 streaming, but may also be used when XSLT 3.0 streaming is not used.

xsl:fork instruction, content model

The content model for xsl:fork instruction has two possible forms:

1) A sequence of zero or more xsl:sequence instructions.

2) A single xsl:for-each-group instruction. As specified by XSLT 3.0 specification, this case will normally use xsl:for-each-group instruction's 'group-by' attribute, since with all other xsl:for-each-group usage patterns the containing xsl:fork instruction has no useful effect.

An xsl:fork instruction's previously mentioned content model, is equivalent to the following XSLT grammar fragment:

```
<xsl:fork>
   <!-- Content: (xsl:sequence* | xsl:for-each-group) -->
</xsl:fork>
```

xsl:fork instruction constraints, and benefits

Although, XSLT 3.0 specification suggests to parallelly process xsl:fork instruction's contained xsl:sequence instructions, XSLT 3.0 specification also requires an XSL transformation processor to produce xsl:fork -> xsl:sequence* results in an order with which these xsl:sequence instructions are provided within an XSL stylesheet.

The advantage of using xsl:fork instruction, over an XSLT conventional xsl:for-each instruction is that, xsl:fork instruction's contained xsl:sequence instruction's may produce results concurrently. This may be beneficial when, xsl:fork's contained xsl:sequence instructions have significant processing that take time to produce their results.

xsl:fork instruction, examples

Example 1

This XSL stylesheet example, illustrate use of xsl:fork instruction to combine XSLT 3.0 sequences, where each of the sequence produces, sequence of xs:integer values. This is a trial illustration of xsl:fork instruction's syntax and processing.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                xmlns:xs="http://www.w3.org/2001/XMLSchema"
                xmlns:fn0="http://fn0"
                exclude-result-prefixes="#all"
                version="3.0">

   <xsl:output method="xml" indent="yes"/>

   <xsl:variable name="seq1" select="(1, 2, 3)" as="xs:integer*"/>
   <xsl:variable name="seq2" select="(5, 6, 7, 8)" as="xs:integer*"/>

   <xsl:template match="/">
      <result>
         <xsl:copy-of select="fn0:xsl3DoFork1($seq1, $seq2)"/>
      </result>
   </xsl:template>

   <xsl:function name="fn0:xsl3DoFork1" as="xs:integer*">
      <xsl:param name="seq1" as="xs:integer*"/>
      <xsl:param name="seq2" as="xs:integer*"/>
      <xsl:fork>
         <xsl:sequence select="$seq1"/>
         <xsl:sequence select="$seq2"/>
      </xsl:fork>
   </xsl:function>

</xsl:stylesheet>
```

An XSL stylesheet transformation result, for this XSL stylesheet is following:

<?xml version="1.0" encoding="UTF-8"?><result>1 2 3 5 6 7 8</result>

Example 2

This XSL stylesheet example, illustrate use of xsl:fork instruction to combine XSLT 3.0 sequences, where each of the sequence produces, sequence of xs:string values. This is a trial illustration of xsl:fork instruction's syntax and processing.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                xmlns:xs="http://www.w3.org/2001/XMLSchema"
                xmlns:fn0="http://fn0"
                exclude-result-prefixes="#all"
                version="3.0">

   <xsl:output method="xml" indent="yes"/>

   <xsl:variable name="seq1" select="('a', 'b', 'c')" as="xs:string*"/>
   <xsl:variable name="seq2" select="('d', 'e', 'f', 'g')" as="xs:string*"/>

   <xsl:template match="/">
      <result>
         <xsl:copy-of select="fn0:xsl3DoFork1($seq1, $seq2)"/>
      </result>
   </xsl:template>

   <xsl:function name="fn0:xsl3DoFork1" as="xs:string*">
      <xsl:param name="seq1" as="xs:string*"/>
      <xsl:param name="seq2" as="xs:string*"/>
      <xsl:fork>
         <xsl:sequence select="$seq1"/>
         <xsl:sequence select="$seq2"/>
      </xsl:fork>
   </xsl:function>

</xsl:stylesheet>
```

An XSL stylesheet transformation result, for this XSL stylesheet is following:

<?xml version="1.0" encoding="UTF-8"?><result>a b c d e f g</result>

Example 3

This is an XSLT 3.0 xsl:fork non-streaming example, similar to an XSL stylesheet xsl:fork example provided within XSLT 3.0 specification.

XML input document:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<transactions>
   <transaction value="5.60"/>
   <transaction value="11.20"/>
   <transaction value="-3.40"/>
   <transaction value="8.90"/>
   <transaction value="-1.99"/>
</transactions>
```

XSL stylesheet:

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

    <xsl:output method="xml" indent="yes"/>

    <xsl:template match="/">
       <xsl:fork>
          <xsl:sequence>
             <xsl:result-document href="credits.xml">
                <credits>
                  <xsl:for-each select="transactions/transaction[@value &gt;= 0]">
                    <xsl:copy-of select="."/>
                  </xsl:for-each>
                </credits>
             </xsl:result-document>
          </xsl:sequence>
          <xsl:sequence>
             <xsl:result-document href="debits.xml">
                <debits>
                  <xsl:for-each select="transactions/transaction[@value &lt; 0]">
                    <xsl:copy-of select="."/>
                  </xsl:for-each>
                </debits>
             </xsl:result-document>
          </xsl:sequence>
       </xsl:fork>
    </xsl:template>

</xsl:stylesheet>
```

This XSL stylesheet example, splits an XML input document into two separate XML documents containing individual relevant information.

Summary

An xsl:fork instruction is useful to process multiple XSL input sequences parallelly, thereby being able to efficiently process an XSL stylesheet input data. An xsl:fork instruction may also process a single xsl:for-each-group instruction as well.

This article has also provided few XSL stylesheet examples, demonstrating uses of xsl:fork instruction.

The XSL stylesheet examples provided within this article, have been run and verified with Saxon-HE XSLT 3.0 processor, and Apache Xalan-J XSLT 3.0 development code.

References

The following W3C specifications are relevant, for the details provided within this document.

1) XSLT 3.0 specification : https://www.w3.org/TR/xslt-30

2) XPath 3.1 specification : https://www.w3.org/TR/xpath-31/

3) XPath 3.1 F&O specification : https://www.w3.org/TR/xpath-functions-31/