Transforming XML with XSLT

Author: Mihai Rotaru

Date: 1 Dec 2011

The XSLT transformation pipeline involves four elements: the source XML document(s) with the XSLT stylesheets, an XSLT processor (template processing engine), and the resulting document(s).

The output format of an XSLT transformation (actually, in the context of XSLT a transformation doesn't actually transform the source XML document - but instead uses it as input for creating other documents) can range from PDF files to plain text files; this is due to XSLT's powerful templating mechanism and versatile XSL Formatting Objects.

Here's an XSL template which will create an XML file based on movies.xml, and add a column which will represent the 'value' of the movie - it's rating divided by it's price:

```
01 <?xml version="1.0" encoding="ISO-8859-1"?>
02 <xsl:stylesheet version="1.0"
03 xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
04
05 <xsl:output method="xml" version="1.0" indent="yes"/>
06 <xsl:template match="/">
07 <movies>
08 <xsl:for-each select="movies/movie">
09 <movie>
00 <xsl:copy-of select="./*"/>
11 <value><xsl:value-of select="rating div price"/></value>
12 </movie>
13 </xsl:for-each>
14 </movies>
15 </xsl:template>
16 </xsl:stylesheet>
```

Notes

- this solution is not ideal, since some elements are hard-coded (movies and movie); these should be somehow deduced
- xsl:for-each is used on L08 to select each movie node in turn (in the order they appear in the original XML file - 'document order')
- xsl:copy-of is used on L10 to create a copies of all the child nodes of the current node. This statement will copy the title, year, etc elements for each movie element.
- on L11, the value element is created. xsl:value-of is used to insert the value resulting from dividing the value of the rating element with the value of the price element of the current node.

Microsoft's msxsl tool (http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=21714) can be used to transform the original XML file into a new one, with the added <value> element:

```
msxsl movies.xml transform.xsl -o new_movies.xml
```

Sorting is accomplished by using the xsl:sort element (http://www.w3.org/TR/xslt#sorting). xsl:sort can only appear as a child of an xsl:apply-templates or an xsl:for-each element. To sort the list of movies by the name of the title, we can simply add a line after L08 in the previous XSLT stylesheet:

```
9 <xsl:sort select="title">
```

The resulting XML document will have it' elements sorted corresponding to the alphabetical order of their 'title' elements. So, 'Apocalypto' will appear the first, and 'Year One' as the last one.

XSLT offers much more powerful features then what was used in this example; one stylesheet can contain multiple templates, and each template can be recursive. XSLT also includes support for regular expressions and math (as shown in the example). But if one finds XSLT's features are not sufficient, then XSLT has an extension mechanism, which allows the language to be extended and new features added to it.