

XML Transformation - XSLT

Ramesh Sankaranarayana

Department of Computer Science
Australian National University

COMP3410 IT in E-Commerce

Outline

- 1 XSLT
 - Introduction
 - Rules
 - An Example

Outline

- 1 XSLT
 - Introduction
 - Rules
 - An Example

XSLT Introduction

XSLT

- Is the transformation part of XSL
- Uses **stylesheets** that contain **style rules**. The style rules define the transformations to be applied to an XML document.
- Applies the rules to the input tree to generate a result tree
- Is a declarative language
- Uses patterns to match elements of the node tree. Different style rules can be created for different elements
- Uses XPath (with a few extensions) to select nodes and node values

XSLT Rules

Rules in XSLT are called **template rules**

- Template rules can be defined by the style sheet creator for specific nodes
- Built-in ones exist that will be applied to the remaining nodes
- Consists of two parts, the **pattern** and the **template**
- The pattern is used to select the nodes to which that template rule will apply
- The template contains the body of the rule

XSLT Rules

Patterns

- A pattern is used to select the nodes to which that template rule will apply
- Used in the **match** attribute of template rules to specify to which elements the rule applies
- Used in the body of the rule to select nodes for further processing

An Example

An example

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

  <!-- Generate HTML output -->
  <xsl:output method="html"/>

  <!-- Insert rules here -->

</xsl:stylesheet>
```

An Example

Rule for root

```
<xsl:template match="/">
  <html>
    <head>
      <title>Book Catalogue</title>
    </head>

    <body bgcolor="white">
      <h1 align="center">Book Catalogue</h1>
      <xsl:apply-templates select="node()"/>
    </body>
  </html>
</xsl:template>
```

An Example

Rule for catalogue

```
<xsl:template match="catalogue">
  <!-- Apply rule for book sorted in title order -->
  <xsl:apply-templates select="book">
    <xsl:sort select="title"/>
  </xsl:apply-templates>
</xsl:template>
```

An Example

Rule for book

```
<xsl:template match="book">
  <!-- Output title and author -->
  <p></p>
  <h2><xsl:value-of select="title"/></h2>
  <h3><xsl:value-of select="author"/></h3>

  <!-- If isbn exists, then apply isbn rule -->
  <xsl:if test="isbn">
    <xsl:apply-templates select="isbn"/>
  </xsl:if>
```

An Example

Rule for book

```
<!-- Apply the rules for publisher and year -->  
<xsl:apply-templates select="publisher|year"/>  
<br/>
```

```
<!-- Output the type and rating -->  
Type=<xsl:value-of select="@type"/>,  
Rating=<xsl:value-of select="@rating"/>,
```

An Example

Rule for book

```
<!-- Output the review -->  
Review=  
<!-- Transform numbers into text -->  
<!-- Mapping is as follows:  
  1 - Rubbish, 2 - Pretty bad  
  3 - Average, 4 - Good  
  5 - Excellent  
-->  
<xsl:choose>  
  <xsl:when test="@review='1'">  
    <xsl:text>Rubbish</xsl:text>  
  </xsl:when>
```

An Example

Rule for book

```
<xsl:when test="@review='2'">
  <xsl:text>Pretty bad</xsl:text>
</xsl:when>
<xsl:when test="@review='3'">
  <xsl:text>Average</xsl:text>
</xsl:when>
<xsl:when test="@review='4'">
  <xsl:text>Good</xsl:text>
</xsl:when>
<xsl:otherwise>
  <xsl:text>Excellent!</xsl:text>
</xsl:otherwise>
</xsl:choose>
</xsl:template>
```

An Example

Rules for isbn, publisher and year

```
<xsl:template match="isbn">  
  <xsl:value-of select="."/>,  
</xsl:template>
```

```
<xsl:template match="publisher">  
  <xsl:value-of select="."/>,  
</xsl:template>
```

```
<xsl:template match="year">  
  <xsl:value-of select="."/>  
</xsl:template>
```

Referencing style sheets

Assume style sheet is `catalogue.xsl`

```
<?xml version="1.0"?>  
<?xml-stylesheet type="text/xsl"  
  href="catalogue.xsl"?>  
  
<catalogue>  
  ...  
</catalogue>
```