XSLT Examples

Ulf Jahr
ulf.jahr@lhsystems.com

Lufthansa Systems Group AG

The basic thing about XSLT is that a XML document can be processed with an XSLT (Extensible Stylesheet Language Transformations) as transforming instructions. The Output can be any text based format such as HTML, SGML, rich text, LaTeX, plain text, and of course XML. XSLT is a “turing complete” programming language which means – generally speaking – every imaginable transformation can be done.

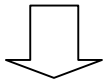
The examples shown here are simple and they are by far not capable to demonstrate all capabilities of XSLT. They should give an impression how the process of a transformation works.

The XML input file

```
<book>
<title>The Dilbert Principle</title>
<author>Scott Adams</author>
</book>
```

XSLT

```
<xsl:stylesheet version = '1.0'
xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>
<xsl:template match="/">
  <h1>
    <xsl:value-of select="//title"/>
  </h1>
  <h2>
    <xsl:value-of select="//author"/>
  </h2>
</xsl:template>
</xsl:stylesheet>
```



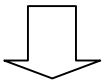
Output:

```
<h1>The Dilbert Principle</h1>
<h2>Scott Adams</h2>
```

if we exchange the rules the order of the output is changed also. The following XSLT is used with the same XML document:

XSLT

```
<xsl:stylesheet version = '1.0'
xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>
<xsl:template match="/">
  <h2>
    <xsl:value-of select="//author"/>
  </h2>
  <h1>
    <xsl:value-of select="//title"/>
  </h1>
</xsl:template>
</xsl:stylesheet>
```



output

```
<h2>Scott Adams</h2>
<h1>The Dilbert Principle</h1>
```

The next example shows a different possibility of writing XML processing rules with XSLT. Please note that there is no template “firstName”. Thus the default processing is applied.

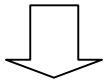
XML input file

```
<employees>
<employee>
     <firstName>Scott</firstName>
     <lastName>Adams</lastName>
</employee>
</employees>
```

XSLT

```
<xsl:stylesheet version = '1.0'
xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>
<xsl:template match="employee">
  <b>
    <xsl:apply-templates select="firstName"/>
  </b>
  <b>
    <xsl:apply-templates select="lastName"/>
  </b>
</xsl:template>

<xsl:template match="lastName">
  <i>
    <xsl:value-of select="."/>
   </i>
</xsl:template>
</xsl:stylesheet>
```



Output

```
<b>Scott</b>
<b>
  <i>Adams</i>
</b>
```

The last XSLT shows a possibility to set the content of the versionNb attribute of an eff.xml to “2”. Since XSLT is a kind of programming language there are – of course – many ways to do this.

```
<xsl:stylesheet version="1.0" xmlns:anc="http://aeec.arinc.net/633"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:fn="http://www.w3.org/2005/xpath-functions">
<xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>

<!-- Changes the content of the versionNb attribute of an eff.xml to 2 -->
<xsl:template match="anc:EFUSUB">
<xsl:element name="EFUSUB" namespace="http://aeec.arinc.net/633">
<xsl:apply-templates mode="root"/>
</xsl:element>
</xsl:template>

<xsl:template match="//anc:M633Header" mode="root" priority="3">2
<xsl:element name="M633Header" namespace="http://aeec.arinc.net/633" >
<xsl:attribute name="versionNb">2</xsl:attribute>
<xsl:attribute name="timestamp"><xsl:value-of se-
lect="@timestamp"/></xsl:attribute>
</xsl:element>
</xsl:template>

<xsl:template match="*" priority="2" mode="root">
<xsl:copy-of select="."/>
</xsl:template>

</xsl:stylesheet>
```