

SDN Community Contribution

(This is not an official SAP document.)

Disclaimer & Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.

Table of Contents

Table of Contents	2
Applies To:.....	2
Summary	2
Date: 25 July 2005	2
Files Involved.....	3
Requirement Scenario.....	3
Solution.....	3
input.xml.....	3
padding_xslt.xsl	4
output.xml	7
Author Bio.....	7

Applies To:

This piece of code was written primarily for XSLT mapping in XI 3.0, but it can be used in any product where XSLT mapping can be used.

Summary

This XSLT code is a sequel to my earlier code sample, [Generic XSLT Mapping for SAP XI](#). In the earlier article we had discussed how XSLT can be used for grouping and aggregation, and then generating the output in an XML format. This XSLT code will be useful in places where we need the output to be in a format retaining white spaces and also having padded output.

Any enhancements to this piece of code are welcome.

By: Sameer Rahim Shadab

Company: Intelligroup Asia Private Ltd. (www.intelligroup.com)

Date: 25 July 2005

Files Involved

input.xml: This is the source xml file for the XSLT code.

padding_xslt.xsl: This is the XSL document which does the grouping and padding.

output.xml: This is how the output file would look after executing the XSLT code.

Requirement Scenario

In the "input.xml" file there are four 'item' nodes under the 'salesOrder' node. Out of the four 'item' nodes, three have the same 'itemDescription' value of 'Blazer' and the fourth has value of 'Trouser'.

The requirement is to identify nodes with same 'itemDescription' and then aggregate the subsequent 'itemRetailPrice' and 'itemSalePrice' into a single node.

Apart from grouping and aggregating, we need to pad the output with white-spaces depending on the length of element-values. Given below are the lengths of each element:

Element	Length
itemDescription	20
itemRetailPrice	6
itemSalePrice	6

So for example if the length of 'itemDescription' is 10 in the input file, then we have to pad it with 10 trailing spaces. The final output will look like the file 'output.xml'.

Solution

input.xml

```
<?xml version="1.0"?>  
<salesOrder>  
<item>
```

```
<itemDescription>Blazer</itemDescription>
<itemRetailPrice>29.95</itemRetailPrice>
<itemSalePrice>24.95</itemSalePrice>
</item>
<item>
<itemDescription>Blazer</itemDescription>
<itemRetailPrice>29.95</itemRetailPrice>
<itemSalePrice>24.95</itemSalePrice>
</item>
<item>
<itemDescription>Blazer</itemDescription>
<itemRetailPrice>29.95</itemRetailPrice>
<itemSalePrice>24.95</itemSalePrice>
</item>
<item>
<itemDescription>Trouser</itemDescription>
<itemRetailPrice>10.55</itemRetailPrice>
<itemSalePrice>8.95</itemSalePrice>
</item>
</salesOrder>
```

padding_xslt.xsl

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

<!-- Build a key to group item by itemDescription -->
<xsl:key name="itemDescription" match="salesOrder/item"
use="itemDescription"/>

<xsl:template match="/">
<salesOrder>
<!-- iterate on each group -->
```

```
<xsl:for-each select="salesOrder/item[generate-id(.) = generate-
id(key('itemDescription', itemDescription)[1]) ]">

<xsl:variable name="group" select="key('itemDescription', itemDescription)"/>

<xsl:variable name="itemDescription">
<xsl:call-template name="pad">
<xsl:with-param name="value" select="itemDescription"/>
<xsl:with-param name="size" select="20"/>
</xsl:call-template>
</xsl:variable>

<xsl:variable name="itemRetailPrice">
<xsl:call-template name="pad">
<xsl:with-param name="value" select="sum($group/itemRetailPrice)"/>
<xsl:with-param name="size" select="6"/>
</xsl:call-template>
</xsl:variable>

<xsl:variable name="itemSalePrice">
<xsl:call-template name="pad">
<xsl:with-param name="value" select="sum($group/itemSalePrice)"/>
<xsl:with-param name="size" select="6"/>
</xsl:call-template>
</xsl:variable>

<!-- Generate the final XML file -->
<item>
<itemDescription>
<xsl:value-of select="$itemDescription"/>
</itemDescription>
<itemRetailPrice>
<xsl:value-of select="$itemRetailPrice"/>
</itemRetailPrice>
<itemSalePrice>
```

```
<xsl:value-of select="$itemSalePrice"/>
</itemSalePrice>
</item>
</xsl:for-each>
</salesOrder>
</xsl:template>

<!--create templates which will do the padding based on string length-->
<xsl:template name="pad">
<xsl:param name="value"/>
<xsl:param name="size"/>
<xsl:param name="pad_char" select="' '"/>
<xsl:value-of select="$value"/>
<xsl:call-template name="padding">
<xsl:with-param name="value" select="$value"/>
<xsl:with-param name="size" select="$size"/>
</xsl:call-template>
</xsl:template>

<xsl:template name="padding">
<xsl:param name="value"/>
<xsl:param name="size"/>
<xsl:param name="pad_char" select="' '"/>
<xsl:if test="$size > string-length($value)"><xsl:value-of
select="$pad_char"/>
<xsl:call-template name="padding">
<xsl:with-param name="value" select="$value"/>
<xsl:with-param name="size" select="$size - 1"/>
</xsl:call-template>
</xsl:if>
</xsl:template>

</xsl:stylesheet>
```

output.xml

```
<?xml version='1.0' encoding='UTF-8' ?>
<salesOrder>
  <item>
    <itemDescription>Blazer          </itemDescription>
    <itemRetailPrice>89.85  </itemRetailPrice>
    <itemSalePrice>74.85  </itemSalePrice>
  </item>
  <item>
    <itemDescription>Trousers Fabric  </itemDescription>
    <itemRetailPrice>10.55  </itemRetailPrice>
    <itemSalePrice>8.95  </itemSalePrice>
  </item>
</salesOrder>
```

Author Bio

Sameer Rahim Shadab has about seven years of experience in the IT industry. He has been with Intelligroup Asia Pvt Ltd. since January 2003. He is a part of Intelligroup's SAP NetWeaver competency group and has been working on SAP XI for around a year now.