How to Develop Stylesheet for XML to XSL-FO Transformation

July, 2005
2nd Edition

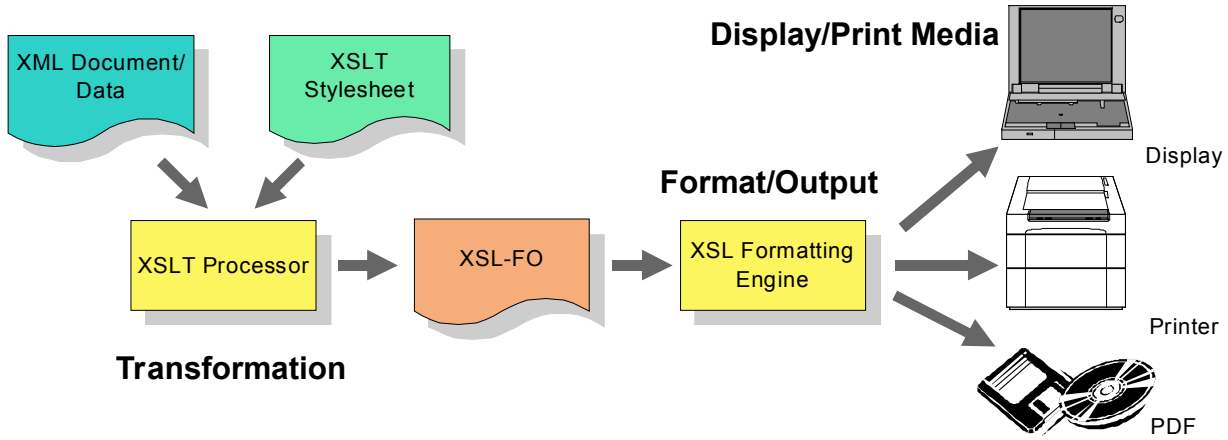
Antenna House, Inc.

Table of Contents

Preface

Extensible Stylesheet Language (XSL) (Source Information [1]) has been brought to the attention of a wide audience as a specification recommended by W3C on 15th October, 2001 for displaying and printing the XML document. The following is the general process to transform the XML document into XSL Formatting Objects (XSL-FO) and print it.

1. Develop the stylesheet that conforms to the DTD of the source XML document to create the target output.
2. Input the XML document and the XSL stylesheet to the XSLT processor to create XSL-FO.
3. Get the target outputs such as paper-based output, PDF output, by the XSL-FO processor.



Generating XSL-FO and Display/Print by XSL Formatter

The knowledge about XSLT and XSL is necessary to develop XSLT stylesheet. The knowledge about XSLT and XSL is necessary to develop XSLT stylesheet. As for XSLT, many reference books has been published other than the specification (Source Information [2]) . Probably, XSLT is already much more familiar one, since it is often used for the conversion from XML to HTML. The XSL specification (Source Information [1]) has a huge amount of contents, it has over 400 pages. It is pretty hard to understand this specification. But basically it is intended for implementers. It is not necessary for XSLT stylesheet designers to understand everything. You can fully write stylesheet by knowing some regular contents and patterns.

This report explains how to edit stylesheet which is used for transforming XML documents into XSL-FO according to the example of SimpleDoc. SimpleDoc is made for this report as a format to write a simple document. This is based on PureSmartDoc (Source Information [4]) presented by Tomoharu Asami. To make it a simple, the number of the elements is reduced and the useful functions for writing and formatting documents are added.

This report explains how to edit a stylesheet which is used for transforming a SimpleDoc document into XSL-FO. This report itself is the instance XML document of SimpleDoc.dtd, and it is then ready to be formatted by Antenna House XSL Formatter with the stylesheet which transforms the SimpleDoc document explained here into XSL-FO.

Step for XSL-FO Transformation

Now, what steps are necessary to develop XSL stylesheet? These steps are explained briefly as below.

Steps	Contents
Know the structure of the XML document	First, the information about the structure of XML source documents is required. XSLT processor can transform XML document into XSL-FO without a DTD. But the information described in the DTD such as types of elements, contents of elements, appearing order of elements and values of properties are necessary for developing a stylesheet.
Specify a printing form	This is the printing form as a final output, in other words the output specification. XSL is a formatting specification. Printing forms has various range of specifications such as sizes and layouts of printing paper, layouts of head and body, deciding whether or not to output index, table of contents, and so on.
Apply a printing form to formatting objects	After determining the specification of printing, you have to know what XSL formatting objects and properties are applied in order to print in this style. It is better to practice how to specify by referring to a simple stylesheet.
Develop an XSL stylesheet	Put the instructions to the stylesheet in order to transform XML source documents into the target printing form. Map the XML source document to XSL formatting objects that can generate the output specification. The stylesheet have the similar aspect as the general programming languages, while it may be difficult if you do not understand the feature of the XSL. (1)

(1) Refer to the definition list template in this report. In XSL, Structure for control of the conditional branch can be made, but it is impossible to assign a value to a variable. The technique to realize by calling loops recursively is necessary.

SimpleDoc Organization

The following table shows the structure of SimpleDoc treated in this report. For more detail, refer to SimpleDoc.dtd.

Element	Meaning	Definition
a group of block elements	−	p | figure | ul | ol | dl | table | program | div
a group of inline elements	−	a | note | span | b | i | em | code | br
doc	root element	(head, body)
head	header	(date | author | abstract | title)*
date, author, abstract, title	header elements, date, author, abstract, title	(#PCDATA | a group of inline elements)*
body	body	(chapter | part | section | a group of block elements | a group of inline elements)*
part	part	(title, (chapter |a group of block elements | a group of inline elements)*)
chapter	chapter	(title, (section | a group of bock elements | a group of inline elements)*)
section	section	(title, (subsection | a group of block elements | a group of inline elements)*)
subsection	subsection	(title, (subsubsection | a group of block elements | a group of inline elements)*)
subsubsection	subsubsection	(title, (a group of block elements | a group of inline elements)*)
title	title	(#PCDATA | a group of inline elements)*
p	paragraph	(#PCDATA | a group of block elements | a group of inline elements)*
figure	figure	(title?) Specify a file by the src property.
ul	unordered list	(li*) Specify a character for label of line by the type property.
ol	ordered list	(li*) Specify format of number in the label by the type property.
dl	definition list	(dt, dd)* Specify whether to format the block in horizontal way or in vertical way by the type property.
dt	definition term	(#PCDATA | a group of block elements | a group of inline elements)*
dd	description of details	(#PCDATA | a group of block elements | a group of inline elements)*
table	entire table	(title?, col*, thead?, tfoot?, tbody) Specify whether to make auto layout or fixed by the layout property. Specify the width of the entire table by the width property.
col	column format	EMPTY Speciry the number of the column by the number property, the width of the column by the width property.
thead	table header	(tr*)
tfoot	table footer	(tr*)
tbody	table body	(tr*)
tr	table row	(th | td)* Specify the height of the row by the height property.
th	table header	(#PCDATA | a group of inline elements | a group of block elements)*

Element	Meaning	Definition
		Specify the number of the columns to be expanded across, the number of the rows to be expanded vertically by the colspan and rowspan properties. The align property allows horizontal alignment to be set to left, right, or center. The valign property allows vertical alignment to be set to top, middle, bottom or baseline.
td	table data	(#PCDATA | a group of inline elements | a group of block elements)* Specify the number of the columns to be expanded across, the number of the rows to be expanded vertically by the colspan and rowspan properties. The align property allows horizontal alignment to be set to left, right, or center. The valign property allows vertical alignment to be set to top, middle, bottom or baseline.
program	program code	(#PCDATA | title)*
div	general block element	(title, (general block element | general inline element)*) The div element expands the type by the class property.
a	anchor(link)	(#PCDATA | a group of inline elements)* Specify URL as the value of href property.
note	note	(#PCDATA | a group of inline elements)*
b	bold typeface	(#PCDATA | a group of inline elements)*
i	italic typeface	(#PCDATA | a group of inline elements)*
em	emphasis	(#PCDATA | a group of inline elements)*
code	program code of the in-line elements	(#PCDATA | a group of inline elements)*
span	general in-line element	(#PCDATA | a group of inline elements)*
br	line break	EMPTY

The features of SimpleDoc are:

- You can start writing a document from part, also start from section. It has a flexible structure so that it can map to various kinds of documents.
- The number of the block element and in-line element are reduced to a minimum amount. The div in the block element and the span in the in-line element are defined to give various extensions.
- The br element is defined so that you can break lines inside of the lists, or the cells in the table, also inside of the paragraph without ending the paragraph.
- Output format of the list and table can be specified by the attributes.
- The problem is that the content and style is mixed.

Hello! World

Simple Example of Transforming SimpleDoc into XSL-FO

Now, we show the simplest simple stylesheet that transforms SimpleDoc to XSL-FO as follows.

Source XML Document (Hello.xml)

```
<?xml version="1.0" encoding="UTF-16" ?>
<doc>
  <head>
    <title>Simple</title>
  </head>
  <body>
    <p>Hello World!</p>
    <p>This is the first<b>SimpleDoc</b></p>
  </body>
</doc>
```

XSL Stylesheet(Simple.xsl)

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

<xsl:template match="doc">
  <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
      <fo:simple-page-master page-height="297mm" page-width="210mm"
          margin="5mm 25mm 5mm 25mm" master-name="PageMaster">
        <fo:region-body   margin="20mm 0mm 20mm 0mm"/>
      </fo:simple-page-master>
    </fo:layout-master-set>
    <fo:page-sequence master-reference="PageMaster">
      <fo:flow flow-name="xsl-region-body" >
        <fo:block>
          <xsl:apply-templates select="body"/>
        </fo:block>
      </fo:flow>
    </fo:page-sequence>
  </fo:root>
</xsl:template>

<xsl:template match="body">
  <xsl:apply-templates/>
</xsl:template>

<xsl:template match="p">
  <fo:block>
    <xsl:apply-templates/>
  </fo:block>
</xsl:template>

<xsl:template match="b">
  <fo:inline font-weight="bold">
    <xsl:apply-templates/>
```

```
  </fo:inline>
</xsl:template>

</xsl:stylesheet >
```

Generated XSL-FO

```
<?xml version="1.0" encoding="UTF-16"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:layout-master-set>
    <fo:simple-page-master page-height="297mm" page-width="210mm"
          margin="5mm 25mm 5mm 25mm" master-name="PageMaster">
      <fo:region-body margin="20mm 0mm 20mm 0mm"/>
    </fo:simple-page-master>
  </fo:layout-master-set>
  <fo:page-sequence master-reference="PageMaster">
    <fo:flow flow-name="xsl-region-body">
      <fo:block>
        <fo:block>Hello World!</fo:block>
        <fo:block>This is the first
          <fo:inline font-weight="bold">SimpleDoc</fo:inline>
        </fo:block>
      </fo:block>
    </fo:flow>
  </fo:page-sequence>
</fo:root>
```

The above XSL-FO is formatted/displayed as follows.

Hello World!
This is the first **SimpleDoc**

Stylesheet Structure

The above Simple.xsl and XSL-FO show the following facts:

- A stylesheet is a set of templates. The descendant of the root element, xsl:stylesheet consists of xsl:template elements. Each xsl:template is applied so that the xxx tag of the source XML document may be processed by match="xxx".
- Formatting objects and the XML source text in each template are output to result XSL-FO tree. Then, templates that match to the descendant elements are called by an instruction of xsl:apply-templates.

XSLT processor loads the source XML document, starts processing from the root node. It finds the templates that match each node, and processes them as described in the templates. The processor processes child elements recursively, continues until the processor returns to the root element and there are no more templates to be processed.

Block Elementand Inline Element

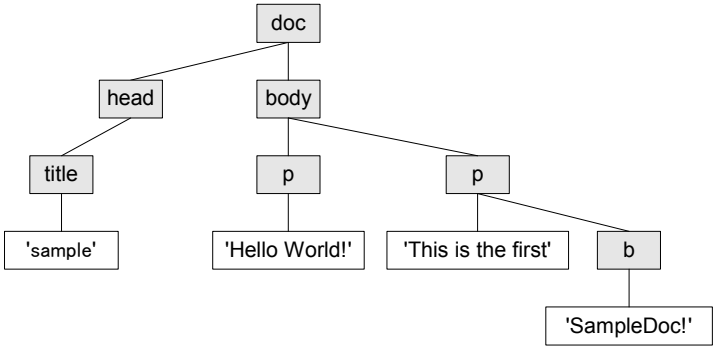
Please note how the XSL stylesheet maps block elements and inline elements in source element to formatting objects.

- In the stylesheet, p elements are transformed into fo:block objects, b elements are transformed into fo:inline objects. The base of XSL-FO transformation is to map the elements of the source XML document to either fo:block elements or fo:inline elements according to the layout instruction.
- The elements that intend to break lines by the end tag normally can be mapped to the fo:block objects. the elements of which the end tag do not intend to break lines can be mapped to the fo:inline objects. The attributes of source

elements specify properties of formatting objects. In this case, the b element means emphasis and property of the inline object generated from b is specified as bold.

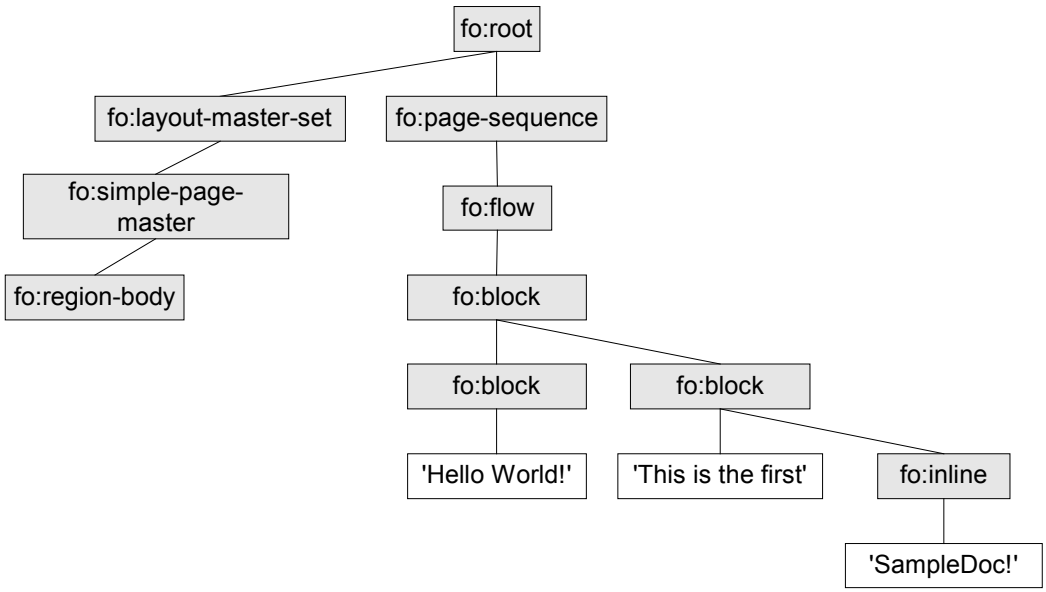
XSL-FO Tree Structure

Next, please pay attention to the XSL-FO tree structure. The following illustrates the structure of the XML document.



Hello.xml Tree Structure

In contrast, the tree structure of XSL-FO is as follows. The root of the XSL-FO tree is fo:root, which has two children, fo:layout-master-set and fo:page-sequence. Fo:layout-master-set defines the page layouts and fo:page-sequence has a flow of contents arranged in in pages.



XSL-FO Tree after XSL Processing

fo:layout-master-set that defines the page layouts should precede fo:page-sequence(preceding-sibling)that is an actual content of pages. XSL processor processes the XML source document from the root element, seeking the templates (xsl:template) to be matched. Therefore, **fo:layout-master-set element should be an output from the template that processes the root element of the XML source document.** In this case, <xsl:template match="doc"> takes this processing.