Antenna House PDF5-*ML* plug-in User's Guide

A Data Usability Company
ANTENNA HOUSE

User's Guide
To get the newest information, visit our website at
https://github.com/AntennaHouse/pdf5-ml.

Preface

About Antenna House PDF5 plug-in

Originally PDF5 plug-in was created for Antenna House I18n Index Library. I18n Index Library is a Java library that makes index pages in the various languages used by the DocBook, DITA to XSL-FO stylesheet.

However PDF5 plug-in implements most of DITA 1.1 and additional 1.2 features, it is denoted to public domain under the Apache License, Version 2.0.

For more information about I18n Index Library, refer to http://www.antennahouse.com/antenna1/i18n-index-library/ for details.

PDF5 is configured to work without I18n Index Library. If you use I18n Index Library, set i18n.index.lib property to "yes" in the ant command-line parameter.

RELATED LINKS

- Plug-in Specific Parameters on page 5

PDF5 plug-in features

PDF5 plug-in has following features:

Newly created XSLT 2.0 stylesheets for DITA to XSL-FO transformation

PDF5 plug-in contains DITA to XSL-FO stylesheets. These stylesheets are written in XSLT 2.0. As DITA Open Toolkit 1.5 and later adopts Saxon-B9.1 as XSLT processor, you can use powerful XSLT 2.0 features in making stylesheets.

Supports most of DITA elements and attributes for XSL-FO output

The contained stylesheets implements most of DITA 1.1 and additional 1.2 elements and attributes for XSL-FO output. For instance these stylesheets have following features:

- In addition to <indexlist> element, <figurelist>, <tablelist> <glossarylist> elements have been implemented.
- If you use use.oid="yes" option, the topic's id is generated using original id and Antenna House Formatter extension attribute for PDF. This makes it possible to generate inter-book links between PDF files.
- Supports DITA indexterm features. Multiple level nested <indexterm> elements, multiple <index-see>,<index-see-also> elements, <index-sort-as> element and range index (indexterm/@start, @end attribute) are implemented. About the range index, if the indexterm/@start has no corresponding indexterm/@end element, this stylesheet automatically close the indexterm range according to the DITA specification.
- Implements character base <syntaxdiagram> element and related elements.

- Supports %display-atts; attributes. The %display-atts; attributes contains scale, frame, expanse attributes. You can use these attributes for <fig>, <pre>, <lines>,<codeblock>, <syntaxdiagram>, <properties>, <msgblock>, <simpletable>, <choicetable>, <screen> and <imagemap> elements.

Independent style definition

All of the style are defined in the external file called style definition file in the "config" folder. This style definition file is created for each language-code. Default style definition file, English and CJK style files are bundled in this plug-in. You can change the output document style only editing this style definition file without changing the stylesheets.

Easy stylesheet customization

If you want to customize the stylesheet algorithms, add the customization stylesheets in the "customization" folder and include it in dita2fo_custom.xsl. As customization is integrated into one folder, you can easily add the customized stylesheet.

About Antenna House PDF5-ML plug-in

PDF5-ML plug-in was derived from PDF5 plug-in and refined from the beginning to meet the actual business requirement.

As template interfaces are changed from PDF5 plug-in it is published as another plug-in distribution. PDF5-ML plug-in is also compatible for I18n Index Library.

PDF5-ML plug-in features

PDF5-ML plug-in has following features.

Supports multiple language formatting for one DITA document.

By specifying xml:lang attribute in any level element of topic, you can format it according to the specified language features including font-family, font-size and hyphenation, etc.

Realizes conditional variable and style definition

There are three predefined attributes for writing conditional variables and attribute-sets.

- @paper-size: Condition for the paper size. Paper size can be any value of standards. (e.g.) A4, B5, Letter
- @doc-type: Condition for the target document type. Useful for plural PDF outputs by one plug-in.
- @output-type: Condition for the output purpose. "web", "print-color", "print-mono" are already used for this value.

You can freely use these attributes in the variable and attribute-set elements in the external style definition file. These conditions can be automatically selected by plug-in parameter. This makes a great flexibility to make a stylesheet that is transparent from these conditions.

Enables the creation of free format cover pages

For most DITA stylesheet programmers it is a troublesome job to develop cover pages for PDF because these pages do not fit DITA topic concept. The component of the cover pages contains logo images, document titles, web links, barcodes, background images, etc. They are usually located in the absolute position.

PDF5-ML plug-in enables to generate cover pages by specifying the style information by authoring side using fo:prop property of ah-dita specialization and use the trick to generate fo:blok-container from <bodydiv> element. This makes a great flexibility to design cover pages by user themselves.

RELATED LINKS

Antenna House DITA specialization (ah-dita) https://github.com/AntennaHouse/ah-dita

Copyright Notices

This document and plug-in uses graphics in the following terms:

- From www.vectorbackground.net under the Creative Commons Attribution license.
- From pixabay.com under the Creative Commons CC0.

Contents

Figures

Tables

Chapter 1 Plug-in Installation

You can use this plug-in with DITA Open Toolkit. You need several procedures to integrate it into DITA-OT.

1.1 The plug-in zip file

If you download the ZIP file from GitHub repository, it contains this PDF and following files and folders.

```
+---com.antennahouse.pdf5.ml
|    |
|    +--- common-graphic
|    |     |
|    |     +--- common graphic files
|    |
|    +--- config
|    |     |
|    |     +--- default_style.xml, en_style.xml, ja_style.xml,
|    |          ko_style.xml, zh-CN_style.xml, zh-TW_style.xml, ...
|    |
|    +--- customization
|    |     |
|    |     +--- dita2fo_custom.xsl
|    |
|    +--- xsl
|    |     |
|    |     +--- dita2fo_xxxx.xsl, psmi.xsl (stylesheet files)
|    |
|    +--- build.xml, integrator.xml, plugin.xml, revision.txt, ...
|
+---jp.acme-corporation.pdf.ml
|    |
|    +--- Sample plug-in files
|
+---samples
|    |
|    +--- sample_en, sample_ja, sample_cover, sample_udhr
|          |
|          +--- Sample DITA instances and resut PDF files
|
+---notices
|    |
|    +--- LICENSE-2.0.txt, OASIS.txt, PSMI.txt
|
+--- ahf_setting.xml, run_en.bat, run_ja.bat, README.md,
     pdf5-ml_manual.pdf
```

1.2 Downloading DITA Open Toolkit

You can download DITA Open Toolkit from http://www.dita-ot.org/.

After downloading archive file, unzip it into the appropriate folder. From now this folder is called simply `[DITA-OT]` for convenience.

This plug-in has been tested with DITA Open Toolkit 1.8.5 or later.

Note

If you use DITA-OT 2.x, several versions have bugs in startcmd.bat that ANT_HOME environment variable is not set properly. Use DITA-OT 2.2.2 or later.

1.3 Updating DITA Open Toolkit batch file

There is a batch file called startcmd.bat at the `[DITA-OT]` folder. Add the following two commands before the `start` command line.

```
REM AH Formatter home and setting file
set AHF_DIR=C:\Program Files\Antenna House\AHFormatterV64
set AHF_OPT=%DITA_DIR%ahf_setting.xml
start "DITA-OT" cmd.exe
```

The 'AHF_DIR' environment variable should be changed according to your Antenna House Formatter version. The ahf_setting.xml is the sample option setting file for Formatter. It exists in the archive root folder. Please copy it to the `[DITA-OT]` folder.

Note

You can specify these environment variables via ant command-line parameter `-Dahf.dir` and `-Dahf.opt`.

RELATED LINKS

Chapter 2 Plug-in And Stylesheet Parameters on page 5

1.4 Installing PDF5-ML plug-in

Follow the next instructions.

1. Copy com.antennahouse.pdf5.ml folder to `[DITA-OT]\plugins` folder.
2. Copy `samples/sample_en, sample_ja` folder to `[DITA-OT]\samples` folder.
3. Copy all of the batch file in the root folder to `[DITA-OT]` folder.
4. Make `[DITA-OT]\out` folder.
5. From Explorer click `[DITA-OT]\startcmd.bat` file.

   The command window titled "DITA-OT" opens.
6. From command window enter `ant -f integrator.xml` command. This command integrates PDF5-ML plug-in into DITA Open Toolkit.

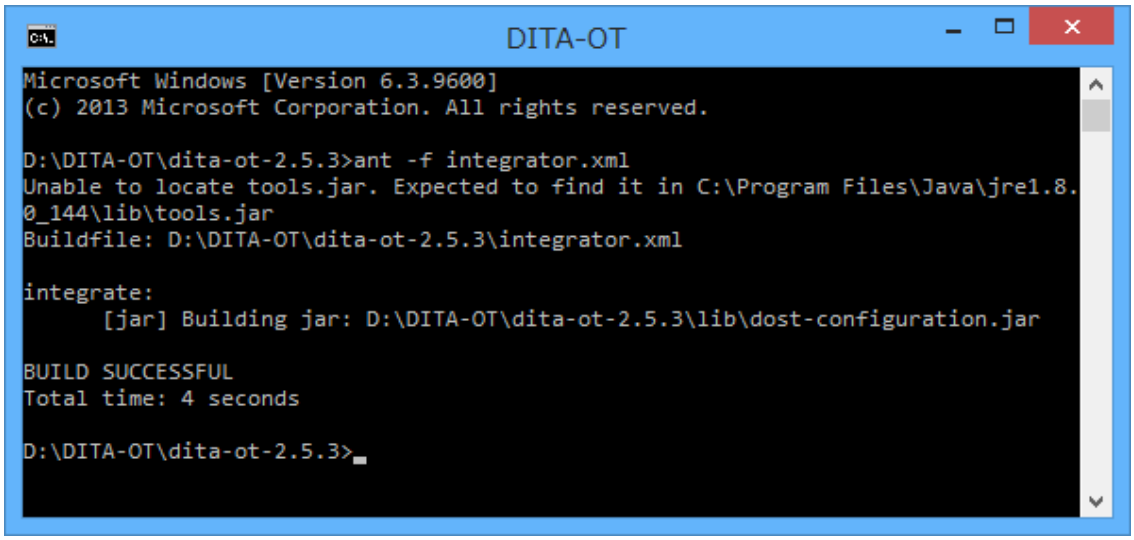

Figure 1.4-1 Integrating PDF5-ML plug-in into DITA Open Toolkit

7. Close the command window.

■ Using `dita` command instead of `startcmd.bat`

You can use `dita` command that is supported in DITA Open Toolkit 2.x.

1. Open command window at `[DITA-OT]` folder instead of clicking `[DITA-OT]\startcmd.bat`.
2. From command window enter `bin\dita -install` command. This command integrates PDF5-ML plug-in into DITA Open Toolkit.

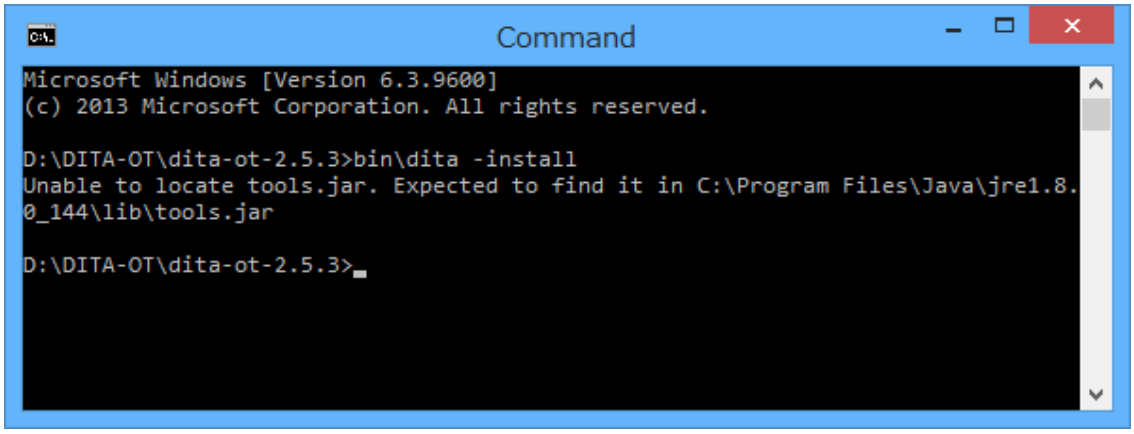

Figure 1.4-2 Integrating PDF5-ML plug-in into DITA Open Toolkit by `dita` command

1.5 Testing PDF5-ML Plug-in

This plug-in contains test data. Follow the next instructions.

1. From Explorer click `[DITA-OT]\startcmd.bat` file. The command window titled "DITA-OT" opens.

2. You can test `run_en.bat` batch files from this window. This batch file has no parameter.

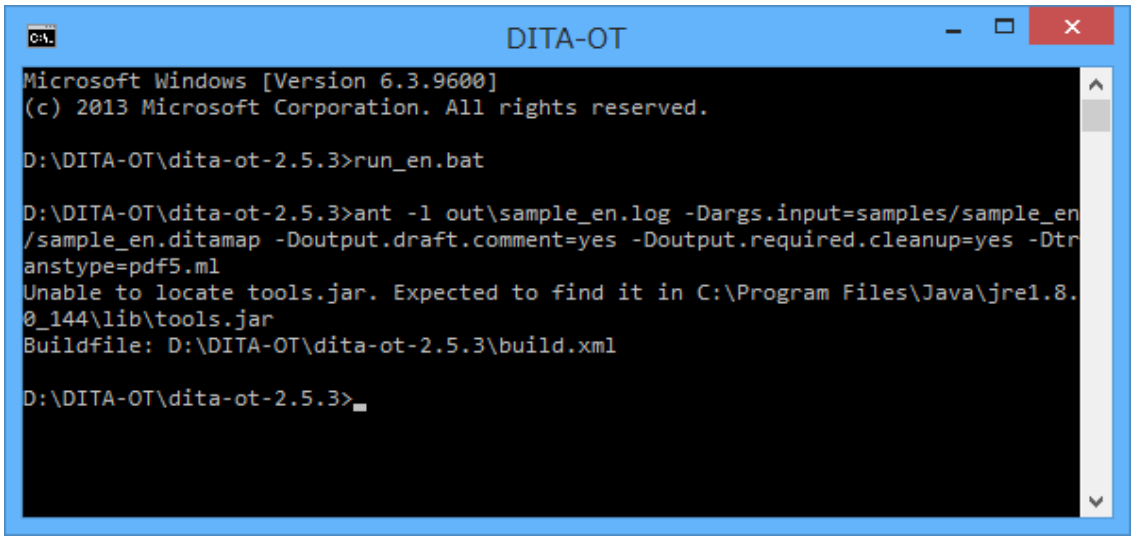

Figure 1.5-1 Example of running a batch file in DITA Open Toolkit

The target PDF file and log file will be generated in the `[DITA-OT]\out` folder.

3. View `[DITA-OT]\out\sample_en.pdf` by Adobe Reader.

Chapter 2 Plug-in And Stylesheet Parameters

There are two categories of ant command-line parameters. One is plug-in general and the other is plug-in specific.

■ General Plug-in Parameters

Following table describes DITA-OT general plug-in parameters:

Table 2-1 General plug-in parameters

Property name in build file	Name in the stylesheet	Default value	Notes
args.input	-	-	Specify the input map file path.
dita.temp.dir	-	-	Specify the folder path where DITA-OT will create temporary files during the transformation process.
clean.temp	-	yes	Specifies whether to clean up temporary folder.
args.filter	-	-	Specifies .ditaval file for flagging and filtering.
output.dir	-	-	Specifies the folder path where DITA-OT outputs the results.
transtype	-	-	Specify "pdf5.ml" to invoke this plug-in.

■ Plug-in Specific Parameters

This plug-in and stylesheet has following parameters. The plug-in parameter is defined in the `com.antennahouse.pdf5.ml\build.xml` as property. The stylesheet parameter is defined as `xsl:param` in `com.antennahouse.pdf5.ml\xsl\dita2fo_param.xsl`.

Table 2-2 Plug-in specific parameters

Property name in build file	Name in the stylesheet	Default value	Notes
add.numbering.title.prefix	PRM_ADD_NUMBERING_TITLE_PREFIX	yes	Specifies whether to add numbering to the part, chapter and descending titles. If this parameter is 'yes', the title will be numbered as '1.1.1' like style.

Property name in build file	Name in the stylesheet	Default value	Notes
add.part.to.title	PRM_ADD_PART_TO_TITLE	yes	Specifies whether to add 'Part' or 'Chapter' to the title when input is specified using bookmap. This parameter is available when add.numbering.title.prefix="yes".
add.thumbnail.index	PRM_ADD_THUMBNAIL_INDEX	yes	Specifies whether to make thumbnail index to the right side of the odd page.
assume.equation.number.as.auto	PRM_ASSUME_EQUATION_NUMBER_AS_AUTO	no	If this parameter is "yes", plug-in ignores manually numbered `<equation-number>`
assume.sortas.pinyin	PRM_ASSUME_SORTAS_PINYIN	no	This parameter is only used with I18n Index Library when making zh-CN index.
auto.scale.down.to.fit	PRM_AUTO_SCALE_DOWN_TO_FIT	yes	Apply auto resizing for the block image when it exceeds the maximum width. This parameter corresponds to the XSL-FO property content-width="scale-down-to-fit".
copy.image.to.output.folder	PRM_IMAGE_IN_OUTPUT_FOLDER	no	Specifies to copy image files to the output folder. Copying image files is obsolete processing prior to DITA Open Toolkit 1.5.x. You should not change this parameter unless specifically needed.
debug.index.sort.result	PRM_DEBUG_INDEX_SORT_RESULT	no	Output index sorting result into out folder for debugging.
debug.style	PRM_DEBUG_STYLE	no	Specifies whether to output style debug information. If this parameter is "yes", this plug-in will output expanded style definition middle XML files into temp folder.
exclude.auto.numbering.from.equation.figure	PRM_EXCLUDE_AUTO_NUMBERING_FROM_EQUATION_FIGURE	yes	Exclude `<equation-block>` in `<figure>` when

Property name in build file	Name in the stylesheet	Default value	Notes
			number.equation.block.unconditionally = "yes".
exclude.cover.from.counting.page	PRM_EXCLUDE_COVER_FROM_COUNTING_PAGE	yes	Specifies whether to exclude cover pages from page numbering.
format.dl.as.block	PRM_FORMAT_DL_AS_BLOCK	yes	Specifies whether to format <dl> as block. If this parameter is 'no', <dl> is formatted using table.*1
include.frontmatter.to.toc	PRM_INCLUDE_FRONTMATTER_TO_TOC	no	Specifies whether to include frontmatter items in the toc page. Usually items in the frontmatter does not appear in the toc.
make.alt.text	PRM_MAKE_ALT_TEXT	no	Specifies whether to make ahf:alttext from alt element. ahf:alttext is effective in Tagged PDF.
make.see.link	PRM_MAKE_SEE_LINK	yes	Specifies whether to make link for "See" and "See Also" entry in the index page. If this parameter is 'yes', the author must make corresponding indexterm for "See" and "See Also".
make.toc.for.map	PRM_MAKE_TOC_FOR_MAP	yes	Specifies whether to make TOC for <map> input.
make.index.for.map	PRM_MAKE_INDEX_FOR_MAP	yes	Specifies whether to make INDEX for <map> input.
number.equation.block.unconditionally	PRM_NUMBER_EQUATION_BLOCK_UNCONDITIONALLY	no	Add equation number to `<equation-block>` even if it has no `<equation-number>`.
otversion	PRM_OT_VERSION	DITA Open Toolkit version	This parameter is used internally. You should not change this parameter value.
output.crop.region	PRM_OUTPUT_CROP_REGION	no	Specifies whether to output crop region for print PDF.
output.draft.comment	PRM_OUTPUT_DRAFT_COMMENT	no	Specifies whether to output <draft-comment> element content.
output.index	PRM_OUTPUT_INDEX	yes	If this parameter is "yes", plug-in outputs the index page.

Property name in build file	Name in the stylesheet	Default value	Notes
output.required.cleanup	PRM_OUTPUT_REQUIRED_CLEANUP	no	Specifies whether to output required-cleanup> element content.
output.pdf	-	[map file name].pdf	Specifies the output PDF file name with extension.
output.start.message	PRM_OUTPUT_START_MESSAGE	yes	If this parameter is "yes", plug-in outputs the start message using <xsl:message>.
output.type	PRM_OUTPUT_TYPE	web	Specifies PDF output type. Possible values are "web", "print-color", "print-mono". You can customize any other output type values in the style definition file.
paper.size	PRM_PAPER_SIZE	Letter	Specifies the paper size. Paper size is defined in style definition file "Paper_Info" entry. Default possible values are Letter, Government-Letter, Legal, B6, B5, B4, A6, A5, A4. You can freely customize the paper size and size of crop region.
style.def.file	PRM_STYLE_DEF_FILE	config\default_style.xml	Specifies default style definition file path based on plug-in folder.
use.i18n.index.lib	PRM_USE_I18N_INDEX_LIB	no	Decide whether to use I18n Index Library. The default value is "no". If you use I18n Index Library, set this parameter to "yes" explicitly.
use.oid	PRM_USE_OID	no	Decide whether to use original @id value for topic. If the input topics have duplicate @id value, this parameter should be set to "no". If you set this parameter to "yes", you can make inter-book links between PDF.
user.input.dir.url	PRM_MAP_DIR_URL	URL of the input map folder.	Specify the input map (bookmap) folder URL.

Property name in build file	Name in the stylesheet	Default value	Notes
			This parameter is used internally. You should not change this parameter.
xsl.file	-	com.antennahouse.pdf5.ml\xsl\dita2fo_shell.xsl	Specifies the main stylesheet file path.
xml.lang	PRM_LANG	The xml:lang attribute value of the map top element.	Specifies the different xml:lang value for the map.

*1 Traditionally <dl> is formatted as block. However DITA introduced title element <dlhead>. If you use dl/dlhead element, it is better to format it using table.

Chapter 3 Command-line Format And Parameter

After starting command window by clicking `[DITA-OT]\startcmd.bat`, you can enter the following ant command-line to invoke this plug-in.

```
ant -l [logfile path] -Dtranstype=pdf5.ml -D[property]=[value] ...
```

Figure 3-1 Ant command-line formats

The [property] is described in the previous chapter as "General Plug-in Parameters" and "Plug-in Specific Parameters". "Plug-in Specific Parameters" are almost defined in `com.antennahouse.pdf5.ml\build.xml`.

Chapter 4 Plug-in process flow

This plug-in has four steps to produce PDF from DITA-OT middle file generated via "preprocess" and "TopicMerge" processing. The simplified diagram is as follows:

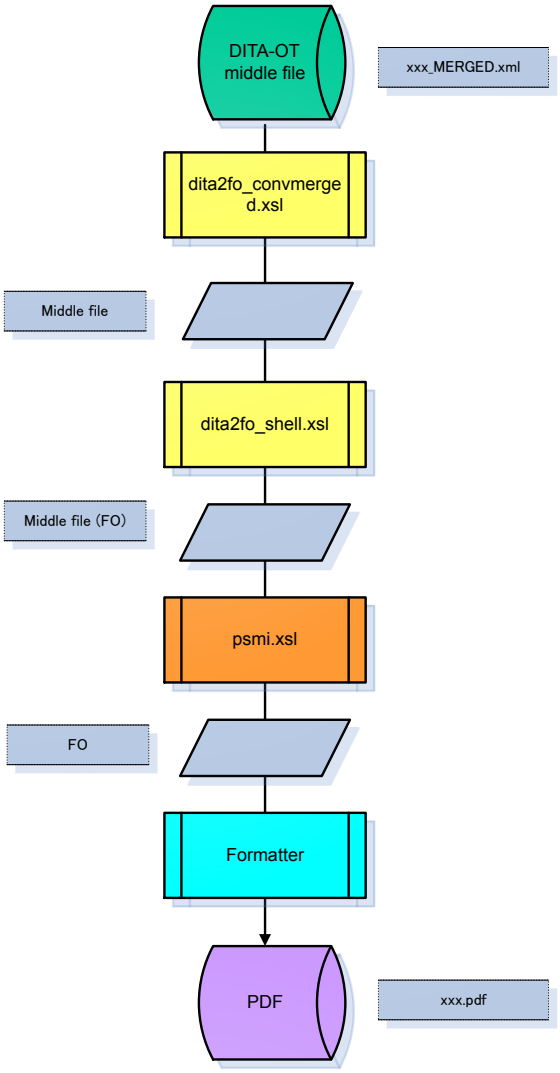

Figure 4-1 Plug-in process flow

xxxx_MERGED.xml

This file is a merged topic file. All of the topics referenced from the map are merged into one file. This file is located in the `[DITA-OT]\temp` folder by default. If you want to see this file, add `-Dclean.temp=no` to the command-line parameter.

Important

This file format is not compatible with DITA Open Toolkit PDF2 plug-in. PDF5-ML uses the original format that DITA-OT generates in the process called "TopicMerge".

dita2fo_convmerged.xsl

This stylesheet removes redundant elements from the input file and copy other elements to the output file.

dita2fo_shell.xsl

Main DITA to XSL-FO transformation stylesheet. The main stylesheet file name can be changed by the `-Dxsl.file` command-line parameter.

psmi.xsl

This stylesheet adjusts fo:page-sequence for index and toc pages.

Formatter

Makes PDF file from XSL-FO.

Chapter 5 Stylesheet Structure

The DITA to XSL-FO stylesheets has following names and contents. The stylesheet files are located at `com.antennahouse.pdf5.ml\xsl` folder.

Table 5-1 Stylesheet file and contents

File Name	Contents
dita2fo_attribute.xsl	dita2fo_attribute.xsl Contains %univ-atts;, %display-atts; related function. The id generation is controlled in this stylesheet.
dita2fo_backmatter.xsl	Contains backmatter control templates.
dita2fo_bodyelements.xsl	Contains body related element templates such as <p>, , , etc.
dita2fo_bookmark.xsl	This stylesheet makes PDF bookmark.
dita2fo_chapter.xsl	Contains <part>, <chapter> control templates. It also contains main <topic> templates.
dita2fo_characterdomain.xsl	Contains <cm>, control templates.
dita2fo_common.xsl	Contains common templates and functions.
dita2fo_constants.xsl	Contains global constant definition.
dita2fo_convmerged.xsl	This stylesheet removes redundant elements from the merged middle file. The redundant element contains <topicref> elements that have print='no' attributes, <topicgroup> elements.
dita2fo_convmerged_import.xsl	This stylesheet defines import modules for dita2fo_convmerged.xsl.
dita2fo_convmerged_shell.xsl	The shell module for dita2fo_convmerged.xsl.
dita2fo_cover.xsl	This stylesheet makes general cover page.
dita2fo_cover_print.xsl	This stylesheet contains templates for making cover pages using fo:prop attribute.
dita2fo_custom.xsl	This stylesheet is prepared for customization. This stylesheet is located at `com.antennahouse.pdf5.ml\cutomization` folder.
dita2fo_dir_attributes.xsl	Contains @dir attribute processing template and function.
dita2fo_dita_util.xsl	Contains DITA related utility functions.
dita2fo_documentcheck.xsl	Contains document checking templates.
dita2fo_figurelist.xsl	This stylesheet makes figure list page.
dita2fo_flag_ditaval.xsl	Contains .ditaval flagging templates and functions.
dita2fo_float_fig.xsl	Contains experimental templates for floating figure.
dita2fo_fo_property.xsl	Contains @fo:prop and @fo:style processing functions.
dita2fo_footnote.xsl	This stylesheet handles footnote. This stylesheet does not output standard XSL-FO's <fo:footnote>. Instead

File Name	Contents
	outputs the footnotes at the end of <table>, <fig>, , elements.
dita2fo_frontmatter.xsl	Contains backmatter control templates.
dita2fo_generate_history_id.xsl	Contains unique id generation function used instead of generate-id().
dita2fo_global.xsl	This stylesheet defines globally used variables.
dita2fo_glossarylist.xsl	This stylesheet handles booklist/glossarylist element and generates glossary list pages.
dita2fo_glossaryrelatedelements.xsl	This stylesheet handles abbreviated-form element.
dita2fo_href_util.xsl	This stylesheet includes @href attribute handling templates.
dita2fo_import.xsl	This stylesheet includes most of other stylesheet modules. This stylesheet is imported in dita2fo_shell.xsl.
dita2fo_index.xsl	This stylesheet contains index pre-processing templates that does not use I18n Index Library.
dita2fo_indexcommon.xsl	This stylesheet contains index common templates and index output templates.
dita2fo_indexi18n.xsl	This stylesheet contains index pre-processing templates that use I18n Index Library.
dita2fo_indexshell.xsl	This stylesheet controls inclusion of dita2fo_index.xsl, dita2fo_indexi18n. xsl by referencing system property. This stylesheet also includes dita2fo_indexcommon.xsl. The system property is set in the build.xml file.
dita2fo_indexsort.xsl	This stylesheet sorts the expanded indexterm in temporary tree. This stylesheet was originally developed by Dimitre Novatchev. Refer to http://www.biglist.com/lists/xsl-list/archives/200303/msg00007.html for details.
dita2fo_indexterm.xsl	This stylesheet checks the <indexterm> element and generates corresponding index-key property or range indexing formatting objects.
dita2fo_layoutmasterset.xsl	This stylesheet generates fo:simple-page-master and fo:page-sequence- master.
dita2fo_main.xsl	This stylesheet contains main control templates.
dita2fo_message.xsl	This stylesheet defines variables for message output.
dita2fo_miscellaneouselements.xsl	Contains miscellaneous element templates such as <draft-comment>, <fn>, etc.
dita2fo_note.xsl	This stylesheet contains templates for <note> element.
dita2fo_numberingmap.xsl	This stylesheet makes temporary tree for making <table>, <figure> and <fn> number.
dita2fo_ot_version.xsl	This stylesheet contains DITA Open Toolkit version comparison functions.
dita2fo_param.xsl	This file defines stylesheet parameter <xsl:param>.

File Name	Contents
dita2fo_programmingelements.xsl	Contains programming element templates such as <apiname>, <codeblock>, etc.
dita2fo_prologelements.xsl	Contains dummy template only.
dita2fo_referenceelements.xsl	Contains reference element templates such as <properties>.
dita2fo_regacyconversionelements.xsl	This styleshhet handles the <required-cleanup> element.
dita2fo_relatedlinks.xsl	This stylesheet contains templates for <related-links>. This template does not outputs links for navigation.
dita2fo_shell.xsl	This is main entry point stylesheet. Plug-in build file specifies this file as main stylesheet.
dita2fo_softwareelements.xsl	Contains software related element templates such as <msgph>, <msgblock>, etc.
dita2fo_specializationelements.xsl	Contains specialization related element templates such as <itemgroup>, <required-cleanup>, etc.
dita2fo_staticcontent.xsl	This stylesheet generates contents of fo:static-contents.
dita2fo_style_get.xsl	This stylesheet defines style and variable getting templates and functions.
dita2fo_style_set.xsl	This stylesheet defines templates that expand style definition file into temporary tree.
dita2fo_tableelements.xsl	This stylesheet handles DITA table elements.
dita2fo_tablelist.xsl	This stylesheet makes table list page.
dita2fo_taskelements.xsl	Contains task related element templates such as <cmd>, <info>, etc.
dita2fo_thumbindexmap.xsl	This stylesheet makes the temporary tree for making thumbnail index.
dita2fo_title.xsl	This stylesheet contains title related templates.
dita2fo_toc.xsl	This stylesheet makes the toc page.
dita2fo_topicelements.xsl	Contains topic related element templates such as <abstract>, <shortdesc>, etc.
dita2fo_typographicelements.xsl	Contains typographic related element templates such as <b>, <i>, etc.
dita2fo_userinterfaceelements.xsl	Contains user interface related element templates such as <uicontrol>, <wintitle>, etc.
dita2fo_util.xsl	Contains utility templates and functions.
dita2fo_utilityelements.xsl	Contains utility related element templates such as <imagemap>, etc. Utility elements are used for HTML output.
dita2fo_xref.xsl	This stylesheet contains xref templates.
psmi.xsl	This stylesheet contains templates that adjust fo:page-sequences. Refer to http://www.cranesoftwrights.com/

File Name	Contents
	resources/psmi/index.htm for detais. This stylesheet was originally developed by Crane Softwrights Ltd.

Chapter 6 Style Definition File

Style definition file is an XML file that describes the information of the output PDF layout with XSLT stylesheet like notation. You can customize the layout by adding modification to the style definition file.

6.1 Style Definition File

Style definition file is a well formed XML file that have the following namespace:

```
http://www.antennahouse.com/names/XSLT/Document/Layout
```

This file has no strict structure, you can use following elements in any position.

<variable>

This element defines the variable. The name attribute is variable name and the child text is the variable value. Variable is referenced from other variable definition or attribute definition. Following is an example of variable definition and reference.

```
<variable name="General_Serif_Font">serif</variable>
<variable name="General_Text_Font">$General_Serif_Font</variable>
```

In this example, the variable value of "General_Serif_Font" and "General_Text_Font" is both "serif". The "$" mark is a placeholder of the variable reference.

The variable elements can hold language specific literals that can be obtained from stylesheet. For example following defines toc, index page title for English.

```
<variable name="Toc_Title">Contents</variable>
<variable name="Index_Title">Index</variable>
```

If the text of variable contains the token `%plug-in-path%`, it is replaced by the actual plug-in path.

```
<variable name="cNoteIcon">%plug-in-path%common-graphic/note.png</vari
able>
```

The variable "cNoteIcon" contents will be automatically expanded into file URI "file:/D:/DITA-OT2.1.0/plugins/com.antennahouse.pdf5.ml/common-graphic/note.png" if DITA-OT is installed in "file:/D:DITA-OT2.1.0". The value can be obtained using ahf:getVarValue function from the plug-in stylesheet. This notation is useful when you want to use plug-in local graphic file which is referenced from the plug-in stylesheet.

<attribute>

This element defines one attribute. This element has the notation that is mostly like the `xsl:attribute` element. The name attribute defines attribute's name. The child text is the attribute value. The attribute element must be the child of the attribute-set element. Following is an example of the attribute definition.

```
<attribute name="line-height">normal</attribute>
```

<attribute-set>

This element defines the attribute set. This element has the notation that is mostly like the xsl:attribute-set element. The name attribute is the attribute-set name. The use-attribute-sets attribute is used to incorporate the other attribute-set into this attribute-set. The attribute-set element can have the <attribute> child elements. Following is an example of the attribute-set definition.

```
<variable name="Base_Font_Size">11pt</variable>
<attribute-set name="atsBaseFontSize">
  <attribute name="font-size">$Base_Font_Size</attribute>
</attribute-set>
<attribute-set name="atsBaseLineHeight">
  <attribute name="line-height">normal</attribute>
</attribute-set>
<attribute-set name="atsRoot" use-attribute-sets="atsBaseFontSize atsB
aseLineHeight">
  <attribute name="xml:lang">en</attribute>
  <attribute name="font-family">$General_Text_Font</attribute>
</attribute-set>
```

This example refers to variable named "General_Text_Font" from font-family attribute definition.

Also this attribute-set incorporates "atsBaseFontSize" and "atsBaseLineHeight" attribute- sets defined in another location.

<instream-object>

This element defines the svg graphic. The child element should have svg namespace `http://www.w3.org/2000/svg`. Following is an example of the instream-object definition.

```
<instream-object name="Note_Icon">
    <svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3
.org/1999/xlink" width="65" height="21" viewBox="0 0 65 21" overflow="
visible" enable-background="new 0 0 65 21" preserveAspectRatio="xMidYm
id meet" xml:space="preserve">
        <g id="layer_1">
            ...
        </g>
    </svg>
</instream-object>
```

<formatting-object>

This element defines the XSL Formatting Objects. The child element should have FO namespace `http://www.w3.org/1999/XSL/Format`. Following is an example of the formatting-object definition.

```
<formatting-object name="foColSpanDummyBlock">
    <fo:block span="all" line-height="0mm" xmlns:fo="http://www.w3.org
```

```
/1999/XSL/Format"> </fo:block>
</formatting-object>
```

<include>

This element indicates to include another style definition file specified in `href` attribute. The `href` attribute has a relative path from the current file. Following is an example from en-US_style.xml.

```
<style-definition
 xmlns:axf="http://www.antennahouse.com/names/XSL/Extensions"
 xmlns="http://www.antennahouse.com/names/XSLT/Document/Layout">
    <include href="en_style.xml"/>
</style-definition>
```

Refer to the `com.antennahouse.pdf5.ml\config\default_style.xml` file for the actual examples.

6.2 How Style Definition File Works?

• The kinds of style definition file

There are two kinds of style definition file in the `com.antennahouse.pdf5.ml\config` folder.

Default style definition file

The file name is fixed to `default_style.xml`. This file defines basic variable and styles based on English document.

Language specific style definition file

The file names are fixed to `[language-code]_style.xml`. The `language-code` is gathered from xml:lang attribute of the whole DITA documents including map and topic. This means that if DITA document contains multiple xml:lang attributes, the corresponding language specific style definition files are used.

• Style definition file integration

Prior to the stylesheet document processing the two kinds of style definition files are integrated into one temporary tree. In this processing default style definition file and each language specific style definition files are expanded using xml:lang that latter file belongs. This means that the temporary tree contains multiple styles and variable definitions for each language. This is illustrated as follows:

```
+--------------------------------------+
| DITA instance                        |
| Contains xml:lang="en-US" & "ja-JP"  |
+--------------------------------------+
                ↓
+--------------------------------------+
| Style definition temporary tree      |
| attribute-set & variables for "en-US" |
```

```
| attribute-set & variables for "ja-JP" |
+---------------------------------------+
```

- **Getting the style information**

DITA to XSL-FO stylesheets requests the style information by using the name attribute of the <attribute-set> element and the relevant language-code associated to the DITA element. In this case the attribute-set temporary tree that matches the language code will be searched. If there are multiple attribute-set entries that have the same name, the whole attributes will be returned as the result ordered by occurrence.

In other words if the result attributes are serialized into the XSLT result tree, the last defined attribute will be appeared as the result XSL-FO property. By using this mechanism, you can write the language specific style overrides.

- **Getting the variable value**

DITA to XSL-FO stylesheets requests the variable value by using the name attribute of the <variable> element and the relevant language-code associated of the DITA element. In this case the variable temporary tree that matches the language code will be searched. If there are multiple variable entries that have the same name, the last positioned variable will be adopted.

By using this mechanism, you can write the language specific variables for the various kinds of literal text in the output.

- **Language specific style definition files**

There are several language specific style definition files in the `com.antennahouse.pdf5.ml\config` folder.

```
en_style.xml, en-US_style.xml, ja_style.xml, ja-JP_style.xml, ko_style.xml, ko-KR_style.xml, zh-CN_style.xml, zh-TW_style.xml
```

The en_style.xml is a dummy file, it has no style definition. The en-US_style.xml only includes en_style.xml. This file only exists to handle derived language codes.

Note

These language specific style definition file contents are not complete yet. You may need to add override definitions when you make these language publishings.

6.3 How To Make Language Specific Style Definition File

It is not difficult to make language specific style definition file. For instance, if you want to make a German file:

1. Make `com.antennahouse.pdf5.ml\config\de_style.xml`
2. Copy XML declalation and the header part from `default_style.xml`.
3. Translate the variable value from English to German.

4. If you want to change style, copy the corresponding style definition from `default_style.xml` and change the style value.
5. Specify *xml:lang="de"* in the DITA instance. Then `de_style.xml` will be used in this plug-in.

Refer to the bundled langauge-specific style definition file in the `com.antennahouse.pdf5.ml\config` folder for actual examples.

Chapter 7 Customizing Stylesheet

This plug-in has `com.antennahouse.pdf5.ml\customization` folder for stylesheet customization. In this folder there is dita2fo_custom.xsl. You can add your customized stylesheet into this file.

Note

If you want to change the style of output, it will be better to make your own plug-in that is described in another chapter.

7.1 How To Add Customized Stylesheet

If you make a customized stylesheet named `dita2fo_my_customization.xsl`, simply add the `xsl:include` instruction to the `com.antennahouse.pdf5.ml\customization\dita2fo_custom.xsl`

```
<xsl:stylesheet version="2.0"
xmlns:fo="http://www.w3.org/1999/XSL/Format"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:axf="http://www.antennahouse.com/names/XSL/Extensions"
xmlns:ahf="http://www.antennahouse.com/names/XSLT/Functions/Document"
exclude-result-prefixes="ahf"
>
<!-- Add your customization xsl here. -->
<xsl:include href="dita2fo_my_customization.xsl"/>
</xsl:stylesheet>
```

Figure 7.1-1 Adding customized stylesheet to dita2fo_custom.xsl

As `dita2fo_custom.xsl` is included in the shell stylesheet `dita2fo_shell.xml`, this customization will work immediately.

```
<xsl:stylesheet version="2.0"
xmlns:fo="http://www.w3.org/1999/XSL/Format"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:import href="dita2fo_import.xsl"/>
<xsl:strip-space elements="menucascade uicontrol abstract"/>
<xsl:include href="../customization/dita2fo_custom.xsl"/>
</xsl:stylesheet>
```

Figure 7.1-2 dita2fo_shell.xsl

Note

For stylesheet customization mechanisms, refer to `dita2fo_shell.xml`, `dita2fo_import.xsl`, `dita2fo_custom.xsl`. According to the XSLT import mechanisms, the customization stylesheet is most honored.

7.2 How To Add Logo Image To Output PDF

Usually the image files are referenced from DITA instances by the <image> element. They are referenced from the output XSL-FO file based on the input map path if it has relative path. However if you want to embed the company logo image that is not referenced from DITA instances, it must be treated manually.

To use the plug-in specific local image file simply put them in the `com.antennahouse.pdf5.ml\common-graphic` folder and add variable definition in the style definition file. The style definition processing will automatically expand the variable definition into the file URI if it contains the token "%plug-in-path%". If you put my-logo.jpg file in the `com.antennahouse.pdf5.ml\common-graphic` folder, it can be referenced from the stylesheet according to the following example.

```
<variable name="companyLogo">%plug-in-path%common-graphic/my-logo.jpg<var
iable>
```

Figure 7.2-1 Defining the common graphic file as variable in style definition file

```
<xsl:template name="putLogo">
  <xsl:variable name="companyLogoPath" as="xs:string" select="ahf:getVarV
alue('companyLogo')"/>
  <fo:block>
    <fo:external-graphic src="{$companyLogoPath}"/>
  </fo:block>
</xsl:template>
```

Figure 7.2-2 Referencing logo image path from stylesheet

7.3 How To Customize The Output Style

If you want to change the output style this plug-in generates, you must add customizations to the style definition file.

Note

This method needs direct editing of plug-in distribution. Every time you update the new plug-in versions, you must maintain the style definition file.

■ Changing The Default Style Definition File

1. Open the config/default_style.xml.
2. Insert the <include href="..."/> element before the last line.

```
<style-definition>
  ...
  ...
  <!-- The last line-->
  <include href="my_customization_style.xml"/>
</style-definition>
```

3. As for the variables this plug-in adopts the value of the last defined one. So your style customization will take precedence as long as the include element exists at the end of the default style definition file.
4. As for the attribute-set, this plug-in adopts all of the attributes defined in the same name. So your customization will be honored even if there are same attribute-set names defined in the default style definition file.

■ Changing The Language Specific style definition file

Changing the language specific style definition takes same steps as default style definition file.

1. Open the `config\[language code]_style.xml`.
2. Insert the <include href="..."/> element before the last line.

```
<style-definition>
  ...
  ...
  <!-- The last line-->
  <include href="en-US_my_customization_style.xml"/>
</style-definition>
```

3. The customization will also work as default style definition style.

Chapter 8 Making Your Own Plug-in

Developing PDF plug-in from scratch needs lots of time and efforts. If you are intending to make PDF plug-in by changing this plug-in, the most convenient way is to make your own plug-in that imports stylesheet and style definition file of this plug-in.

This plug-in distribution contains `jp.acme-corporation.pdf.ml` plug-in as the import model. It contains minimum customization examples:

- Change the base serif font and sans-serif font (`config\en_style.xml`).
- Change the cover page title (`xsl\dita2fo_global.xsl`).

You can test this model after integrating it to DITA-OT by specifying `-Dtranstype=acme-corporation.pdf.ml` in ant command-line.

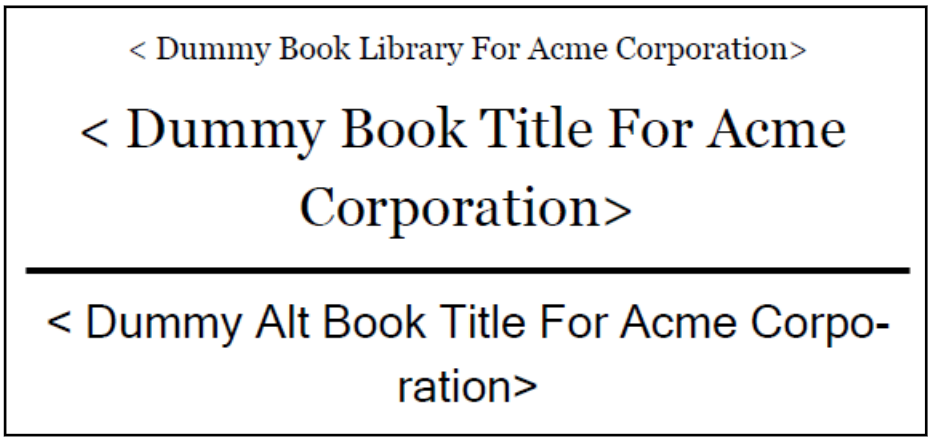

Figure 8-1 Sample customization result (sample_en.ditamap cover page)

Important

As you know, Acme Corporation is a fictional corporation described at Wikipedia. The sample plug-in folder name `jp.acme-corporation.pdf.ml` does not concern any actual organizations.

8.1 Importing And Customizing Stylesheets

It is important to obey following guidelines:

Table 8.1-1 Stylesheet customization guidelines

File	Necessity	Notes
dita2fo_convmerged.xsl	Optional	Needed if you customize merged middle file processing.
dita2fo_convmerged_shell.xsl	Mandatory	This is the entry point stylesheet for merged middle file processing and it must import dita2fo_convmerged_shell.xsl of PDF5-ML plug-in.
dita2fo_global.xsl, dita2fo_param.xsl, etc	Optional	Include it if you have any customization for these stylesheets.
dita2fo_import.xsl	Mandatory	It must import dita2fo_style_set.xsl. Also it must import your customization templates if necessary.

File	Necessity	Notes
dita2fo_shell.xsl	Mandatory	This is the main entry point stylesheet and it must import dita2fo_shell.xsl of PDF5-ML plug-in and dita2fo_import.xsl of this plug-in.
dita2fo_style_set.xsl	Mandatory	It must override one global variable based on the URI of the stylesheet file location.
psmi.xsl	Mandatory	It must import psmi.xsl of PDF5-ML plug-in.

8.2 Importing And Customizing Style Definition File

In most cases the style definition file imports (includes) PDF5-ML's same file and add the customizations.

Following code shows the import & customization examples of `config/en_style.xml`.

```
<style-definition
    xmlns:axf="http://www.antennahouse.com/names/XSL/Extensions"
    xmlns="http://www.antennahouse.com/names/XSLT/Document/Layout">

    <include href="../../com.antennahouse.pdf5.ml/config/en_style.xml"/>

    <!-- Sample customization of font-family -->
    <variable name="General_Serif_Font">Century Schoolbook</variable>
    <variable name="General_Sans_Serif_Font">Century Gothic</variable>

</style-definition>
```

Figure 8.2-1 Style definition file customization example

The include element should come to the first line and you can define your variable and attribute-set customizations in the proceeding line.

Chapter 9 Tips For this Plug-in

There are several authoring and coding tips for this plug-in.

9.1 Paying attention to topic/@xml:lang

PDF5-ML plug-in supports multi-language formatting in one DITA document. This is based on the @xml:lang attribute in map or topic. Especially specifying the xml:lang attribute to topic is very important.

If you are using CMS for your DITA content and translate it into multiple languages, the xml:lang attribute of topic is indispensable. But in some cases this attribute is explicitly omitted and map/@xml:lang attribute takes over the topic/@xml:lang attribute. But there are several problems in omitting topic/@xml:lang.

- In the DITA specification, xml:lang attribute is applicable for most of the elements. But it does not inherit from map to topic.

  2.1.3.9.1 The @xml:lang attribute

  The @xml:lang attribute can be specified on the <map> element. The @xml:lang attribute cascades within the map the same way that it cascades within a topic. The @xml:lang value does not cascade from one map to another or from a map to a topic, and an @xml:lang value specified in a map does not override @xml:lang values specified in other maps or in topics.

- DITA-OT automatically generates xml:lang="en-US" if it is not specified in topic. This may be based on the specification of "The default value of the processor may be either fixed".

  If the @xml:lang attribute on the document (outermost) element of a map or of a top-level topic has no value, the processor should assume a default value. The default value of the processor may be either fixed, configurable, or derived from the content itself, such as the @xml:lang attribute on the primary map file.

So specified xml:lang to map isn't applied to each topic automatically. This behavior sometimes causes the unexpected formatting result because PDF5-ML honors the xml:lnag attribute for generating XSl-FO. So it is important to explicitly specify the xml:lang attribute to get the expected formatting result

The @xml:lang attribute should be explicitly set on the root element of each map and topic.

9.2 How To Make The Inter-Book Links

Making the inter-book links is not so difficult. You can accomplish it by changing the target PDF build parameter and making the link to PDF by `<xref>` specifying topic/@id.

● Building the target PDF

Follow the next steps to make the link target PDF:

- Build the PDF file using this plug-in by specifying the parameter `-Duse.oid=yes`. This is indispensable requirement.
- In this case all of the `topic/@id` value must be unique over the document,
- This plug-in stylesheet generates the XSL-FO that indicates Formatter to make the named destination in the output PDF.

```
<!-- Original authoring -->
<topic id="topic_ja_5">
```

```
<!-- Generated XSL-FO -->
<fo:block id="topic_ja_5" axf:destination-type="xyz-top">
```

- **Authoring the inter-book link by `<xref>` element.**

You can make the inter-book link by using the following @href format:

```
<xref href="[target PDF path]#[topic/@id value] format="pdf" scope="exter
nal">
```

Of course you must know the @id value of the topic in the target PDF. For instance, the following authoring:

```
<xref href="sample_ja.pdf#topic_ja_5" format="pdf" scope="external">
```

will be converted into the following XSL-FO:

```
<fo:basic-link external-destination="sample_ja.pdf#nameddest=topic_ja_5"
axf:action-type="gotor"/>
```

This link will surely work when sample_ja.pdf is built using `-Duse.oid=yes` parameter.

9.3 Importing Base Element Template

There are some difficulties in importing the base class element by your customization stylesheet.

If your customization stylesheet that imports this plug-in needs to override base class templates such as `<ph>` element, the difficult problem will be raised. If you write the following override template in your customization,

```
<xsl:template match="*[contains(@class, ' topic/ph ')]">
   ...
</xsl:template>
```

all of the other specialization element of ' topic/ph ' in this plug-in will be never be called because the override template has highest precedence in the call tree. For instance, the next template in this plug-in will be never called.

```
<xsl:template match="*[contains(@class,' hi-d/b ')]" priority="2">
    ...
</xsl:template>
```

This is because the <b> element has the class value "+ topic/ph hi-d/b ".

To avoid this phenomenon, this plug-in has the following template structure for base class elements:

```
<xsl:template match="*[contains(@class, ' topic/ph ')]">
   <xsl:call-template name="processPh"/>
</xsl:template>
<xsl:template name="processPh">
   ...
</xsl:template>
```

So you can override the `processPh` template in your customization. This technique will honor the @priority value defined in the this plug-in template. This plug-in adopts this template structure for <abstract>, <data>, <keyword>, <li>, <ph>, <ol>, <pre>, <section>, <term>, <ul> elements.

9.4 Understanding The Template Model

PDF5-ML plug-in adopts a bit different template model for domain elements.

Here you can see several implementation patterns about <p> element. The comments are added by author.

[PDF2]

```
<!--The style for <p> is fixed by <xsl:attribute-set name="p">.
    You cannot introduce conditional attribute or language specific style
    as long as <xsl:attribute-set name="p"> does not contain any logic in it.
 -->
<xsl:template match="*[contains(@class, ' topic/p ')]">
   <fo:block xsl:use-attribute-sets="p">
       <!-- Only copy the @id attribute -->
       <xsl:call-template name="commonattributes"/>
       <xsl:apply-templates/>
   </fo:block>
</xsl:template>
```

[PDF5]

```
<!--The style for <p> is defined in the style definition file named "atsP".
    You can introduce attribute that have variable reference or language
    specific attribute.
 -->
<xsl:template match="*[contains(@class, ' topic/p ')]">
   <fo:block>
      <xsl:copy-of select="ahf:getAttributeSet('atsP')"/>
      <!-- Handle %univ-atts -->
      <xsl:call-template name="ahf:getUnivAtts"/>
      <!-- Handle fo:prop attribute introduced in ah-dita specialization -->
      <xsl:copy-of select="ahf:getFoStyleAndProperty(.)"/>
      <xsl:apply-templates/>
   </fo:block>
</xsl:template>
```

[PDF5-ML]

```
<!--The style for <p> is defined in the style definition file
    and it is named as "atsP".
    This style name can be obtained from mode="MODE_GET_STYLE" template.
    This template will be automatically called from getAttributeSetWithLang
    template.
    getAttributeSetWithLang template will return attribute()* sequence
    considering xml:lang attribute of the ancestor or self element.
 -->
<xsl:template match="*[contains(@class, ' topic/p ')]" mode="MODE_GET_STYLE"
              as="xs:string*">
   <xsl:sequence select="'atsP'"/>
</xsl:template>

<xsl:template match="*[contains(@class, ' topic/p ')]">
   <fo:block>
      <!-- Get attribute for fo:block considering xml:lnag -->
      <xsl:call-template name="getAttributeSetWithLang"/>
      <!-- Handle %univ-atts -->
      <xsl:call-template name="ahf:getUnivAtts"/>
      <!-- Handle fo:prop attribute introduced in ah-dita specialization -->
      <xsl:copy-of select="ahf:getFoStyleAndProperty(.)"/>
      <xsl:apply-templates/>
   </fo:block>
</xsl:template>
```

If you are intending to add a template for your customization or newly specialized domain elements, it is better to adopt the above PDF5-ML template model because this model supports the automatic conversion of the language specific attribute. If you don't need this feature, PDF5 based code is still available.

Chapter 10 Stylesheet Messages

The DITA to XSL-FO stylesheets outputs the following messages. The suffix character of the message number means:

"I"

This is an information message.

"W"

This is a warning message.

"F"

This is a fatal error message. In some cases XSLT processor will stop processing after this message has been outputted.

Table 10-1 Stylesheet messages

Message	Contents
[General 001W] No template is defined for this element. element=%elem file=%file	This stylesheet does not have template for %elem. Check the stylesheet file and add template for this element.
[getAttributeSet 005F] Attribute-set name not found attribute-set=%attrsetname file=%file	Check the style definition file=%file.
[getAttributeValue 006F] Attribute-set name not found. attributeset=%attrsetname file=%file	Check the style definition file=%file.
[getAttributeValue 007F] Attribute name not found. attribute-name=%attrname attribute-set-name= %attrsetname file=%file	Check the style definition file=%file.
[getVarValue 008F] Variable %var is not found in %file.	Check the style definition file=%file.
[getInstreamObject 009F] Instream object name not found. objname=%objname file=%file	Check the style definition file=%file.
[getFormattingObject 010F] Formatting object name not found. obj-name=%objname file=%file	Check the style definition file=%file.
[getVarValueAsInteger 012F] Variable value is not castable as xs:integer. variable=%var value=%value file=%file	Check the style definition file=%file.
[getVarRecursive 023F] Referenced variable %varname is not exist in %stylefile	Check the style definition file=%stylefile.
[getCssStyle 025F] Style %style does not defined in %file.	Check the style definition file=%file. This is inner error.
[percentToNumber 028W] Invalid @scale value. Assumed 100. scale=%scale element=%elem file=%file	Correct the authoring.
[processLocalXref 030F] Xref/@href destination topic does not found. Probably href format is illegal. href=%h file=%file.	Correct the xref/@href authoring.

Message	Contents
[getXrefTitle 031W] Xref/@href destination section has no title element. Section id=%id file=%file	Correct the xref or section authoring.
[getXrefTitle 032W] Xref/@href destination example has no title element. Example id=%id file=%file	Correct the xref or example authoring.
[getXrefTitle 033W] Xref/@href destination table has no title element. Table id=%id file=%file	Correct the xref or t able authoring.
[getXrefTitle 034W] Xref/@href destination fig has no title element. Fig id=%id file=%file	Correct the xref or fig authoring.
[getXrefTitle 035W] Xref/@href destination %elem has no title content. %elem id=%id file=%file	Correct the xref authoring.
[getFootnotePrefix 037W] The fn element does not have table,simpletable,ul,ol,dl,glossdef as ancestor. This fn will be ignored. Content='%cont' file=%file	Correct the fn authoring.
[bodyelements 041W] Element object is not suitable for PDF output. Ignored. file=%file classid=%class	Correct the object authoring.
[getKeyCol 050W] The keycol attribute is not positive integer. Assumed as not specified. file=%file element= %elem keycol=%keycol	Correct the @keycol authoring.
[dlhead 055W] As PRM_FORMAT_DL_AS_BLOCK='yes', the dl/dlhead element is ignored. file=%file	Change the parameter PRM_FORMAT_DL_AS_BLOCK='no' or delete dl/ dlhead element.
[synnoteref 060W] The href attribute of synnoteref cannot be resolved. file=%file trace=%trace @href= %href	Check the synnoteref authoring.
[processLink 062W] Ignored invalid link in related-links or reltable. file=%file href=%href	Check the reltable or relatedlinks authoring.
[processLink 063W] Ignored invalid link in related-links or reltable. file=%file href=%href	Check the reltable or relatedlinks authoring.
[processLink 070W] topicref/@href target does not found. href='%href' file=%f ile	Check the topicref authoring.
[processLocalXref 072F] Xref target is not contained in map. href='%href' file=%file	Check the xref target.
[makeBasicLinkDestination 074F] Topic in href target does not found. href='%href' file=%file element= %element	Check the href attribute of the specified element.
[processTopicref 076F] topicref/@href target does not found. href='%href' file=%file	Check the href attribute of the topicref element.
[genGlossaryListMain 078F] Referenced topicref does not found. id of topic='%id' file=%file	Check the corresponding glossentry element.
[genGlossaryListBookMark 079F] Referenced topicref does not found. id of topic='%id' file=%file	Check the corresponding glossentry element.
[Frontmatter 080W] Element:%elem without @href attribute is ignored. file=%file	Check the corresponding element.

Message	Contents
[Frontmatter 082W] Element:%elem without @href attribute is not supported. file=%file	Check the corresponding element.
[Backmatter 084W] Element:%elem without @href attribute is ignored. file=%file	Check the corresponding element.
[Backmatter 086W] Element:%elem without @href attribute is not supported. file=%file	Check the corresponding element.
[Figurelist 087W] Element:%elem is ignored. There is no figure with title. file=%file	Check the corresponding element.
[Tablelist 088W] Element:%elem is ignored. There is no table with title. file=%file	Check the corresponding element.
[Glossarylist 089W] Element:%elem is ignored. There is no glossentry as the child of %elem. file=%file	Check the corresponding element.
[Glossarylist 090W] Element:%elem should be written in backmatter otherwise all of indexterm after the %elem will be automatically ignored. file=%file	Check the corresponding element.
[abbreviatedForm 092W] Referenced topic does not found. @keyref=%keyref @href=%href file=%file	Check the corresponding element.
[ditamapClass 100F] Undefined ditamap class. class= %class file=%file	Check the map file.
[documentLang 101W] xml:lang is not specified in map. Adopt 'en' as default language.	Check the ditamap file.
[altstyledef 102W] [altstyledef 102W] Alternate style definition file %file does not found. Stylesheet use %default as style definition file.	Check the existence of %file.
[altstyledef 103W] Alternate style definition file %file does not found. Stylesheet use %default as alternate style definition file.	Check the xml:lang attribute of map root element or check the well-formedness of the alternate style definition file.
[styledef 104F] Style definition file %file does not found.	Check the existence of %file.
[styledef 105F] Language specific style definition file is not found. file="%file"	Check the file is prepared in config folder.
[getPropertyNu 400W] Invalid non-numeric property value='%val'. Treated as 1.0.	Some font-size attribute in the style definition file is not numeric.
[documentCheck 500F] topicref/@href target does not found. ditamap=%xtrf href=%ohref	Check the ditamap file.
[documentCheck 501F] Detected same topic/@id value. It must be unique. @id='%id' file1=%xtrf1 file2=%xtrf2	Correct the id value of the topic.
[documentCheck 503F] The part and chapter level element coexists. Either part or chapter only document is supported. ditamap=%xtrf	This stylesheet does not accept this configuration. Correct the ditamap file.
[documentCheck 504F] The part level element contains attribute toc='no'. file=%ohref ditamap=%xtrf	This stylesheet does not accept this pattern. Correct the ditamap file.
[documentCheck 505F] The chapter level element contains attribute toc='no'. file=%ohref ditamap=%xtrf	This stylesheet does not accept this pattern. Correct the ditamap file.

Message	Contents
[documentCheck 514F] This plug-in does not support a indexlist element in frontmatter. ditamap=%xtrf	Correct the authoring.
[documentCheck 599F] Terminated by document fatal error.	Check the previous error message.
[genIndex 600F] Failed to sort indexterm. Counts before sorting= %before, after sorting=%after	This will be the difficulty of the I18n Index Library.
[genIndex 601I] Index debug start.	This is a debug message.
[genIndex 602I] Index debug end.	This is a debug message.
[indexterm 610W] Authoring error! Indexterm has sibling index-see element. index-key='%key' file=%file Stylesheet will ignore this index-see element.	Correct the indexterm authoring.
[indexterm 611W] Authoring error! Indexterm has sibling indexsee-also element. index-key='%key' file= %file Stylesheet will ignore this index-see-also element.	Correct the indexterm authoring.
[indexterm 612W] Authoring error! Indexterm has both index-see and index-see-also child element. index-key='%key' file=%file Stylesheet will ignore this index-see element.	Correct the indexterm authoring.
[indexterm 620W] Authoring error! Text of indexterm is empty. index-key='%key' file=%file Stylesheet will ignore this indexterm element.	Correct the indexterm authoring.
[indexterm 621W] Authoring error! Text of indexterm is too long. It must be less than %max characters for sorting. index-key='%key' file=%file Stylesheet will trim this indexterm text for index sorting.	Correct the indexterm authoring. This message will not be outputted when you use I18n Index Library.
[indexterm 630W] Authoring error! Start attribute is specified in nesting indexterm more than once. previous='%prev' current='%curr' index-key='%key' file=%file Stylesheet will ignore this start attribute.	Correct the indexterm authoring.
[indexterm 640W] End attribute authoring error! Corresponding indexterm that has same start attribute value does not found. end='%end' index-key='%key' file=%file Stylesheet will ignore this end attribute.	Correct the indexterm authoring.
[indexterm 641W] End attribute authoring error! Corresponding indexterm that has same start attribute value exist plurally. end='%end'index-key='%key' file= %file Stylesheet will ignore this end attribute.	Correct the indexterm authoring.
[indexterm 642W] Authoring warning! Indexterm element that has end attribute should not have ancestor indexterm element. end='%end' index-key='%key' file= %file Stylesheet will ignore ancestor indexterm.	Correct the indexterm authoring.
[indexterm 643W] Authoring warning! Indexterm element that has end attribute should not have child indexterm element. end='%end' index-key='%key' file= %file Stylesheet will ignore child indexterm.	Correct the indexterm authoring.
[indexterm 650W] Authoring error! Attribute start and end cannot be specified together in one indexterm.	Correct the indexterm authoring.

Message	Contents
start='%start' end='%end' index-key='%key' file=%file Stylesheet will ignore this indexterm.	
[indexterm 651W] Authoring error! End attribute cannot be specified to the descendant of indexterm that has start attribute. start='%start' end='%end' index-key='%key' file=%file Stylesheet will ignore this indexterm.	Correct the indexterm authoring.
[indexterm 652W] Authoring error! Start attribute cannot be specified to the descendant of indexterm that has end attribute. start='%start' end='%end' index-key='%key' file=%file Stylesheet will ignore this indexterm.	Correct the indexterm authoring.
[makeChapterMap 700W] Illegal class is found in topicref. class='%class' file=%file.	Check the ditamap file.
[getFoProperty 800F] Missing ':' in style description. @fo:prop='%foAttr' @xtrc='%xtrc' @xtrf='%xtrf'	Check the @fo:prop property.
[getFoProperty 802F] Property value is invalid. Property='%propName' @xtrc='%xtrc' @xtrf='%xtrf'	Check the @fo:prop property name.
[getFoPropertyReplacing 804F] Missing ':' in style description. @fo:prop='%foAttr' @xtrc='%xtrc' @xtrf='%xtrf'	Check the @fo:prop property separator.
[getFoPropertyReplacing 806F] Property value is invalid. Property='%propName' @xtrc='%xtrc' @xtrf='%xtrf'	Check the @fo:prop property name.
[attribute-set 900F] Illegal attribute found in attribute-set element in style definition. attribute-set-name='%attribute-set-name' attribute='%attribute'.	Check the attribute of attribute-set element in the style definition file.
[attribute-set 902F] Attribute-sets references itself by @use-attribute-sets attribute. attribute-set-name='%attribute-set-name'.	Check the use-attribute-sets attribute of attribute-set element in the style definition file.
[processUseAttributeSet 904F] There is no attribute-set referenced by @use-attribute-sets. use-attribute-set='%attribute-set-name'.	Check the use-attribute-sets attribute of attribute-set element in the style definition file.
[convmerged 1001I] Removing topicref. href=%href ohref=%ohref	An topicref element is removed because it has @print="no" attribute.
[convmerged 1002I] Removing topic. id=%id xtrf=%xtrf	An topic element is removed because it is referenced from topicref that has @print="no" attribute.
[convmerged 1003I] Removing link. href=%href	An link element is removed because it refers to the removed topic.
[convmerged 1004W] Warning! Xref refers to removed topic. href=%href	An xref element refers removed topic.
[parameter 1500F] Paper size parameter is invalid. Parameter=%param Supported value=%sptval.	Check the paper size parameter.

Chapter 11 Limitation

This plug-in has following important limitations.

■ Plug-in feature

Making index page

If this plug-in is used without I18n Index Library, the language support for making index is very limited. The supported languages are xml:lang="en", "ja" only. If you want to use indexing features of other language, please consider:

- Develop your own index template for needed languages.
- Purchase Antenna House I18n Index Library.

Using Formatter other than Antenna House Formatter for PDF output

In this plug-in the ant build file is designed to use Antenna House Formatter to convert XSL-FO to PDF. However if you want to use another Formatter, it is possible to rewrite `com.antennahouse.pdf5.ml\build.xml` to support another one. As this stylesheet uses XSL 1.1 features the target formatter should support XSL 1.1. Please add modifications to build file on your own responsibility.

■ Plug-in implementation

Using composite topic

Composite topic has top <dita>elements and child <topic> elements.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE dita PUBLIC "-//OASIS//DTD DITA Composite//EN" "ditabase.dtd">
<dita>
    <topic id="topic_u2k_csc_jbb">
        <title>Topic in composite topic</title>
        ...
    </topic>
    <concept id="concept_u2k_csc_jbb">
        <title>Concept in composite topic</title>
        ...
    </concept>
    <reference id="reference_u2k_csc_jbb">
        <title>Reference in composite topic</title>
        ...
    </reference>
</dita>
```

If you reference composite topic from map by <topicref>, it becomes vague how to treat the hierarchy of child topics of <dita> element in comparison of ditamap structure. For instance if composite topic is referenced from <chapter>, it will become difficult to treat the child topics as series of <chapter>. In composite topic the <dita> element fulfill the role

of topic container only. It will be better to take topics in composite topic apart and reference them from map directly. Based on these reasons this plug-in does not support composite topics.

Reference to the nested topic

The following topicref/@href is not supported in a map (without reltable)

```
<topicref href="abc.xml#topicId">
```

where "topicId" is not the top level topic/@id of abc.xml.

<fn> element does not generate <fo:footnote> defined in XSL-FO specification

This plug-in assumes that <fn> elements exist in <table>, <simpletable>, <ul>, <ol>, <dl>, <glossdef> elements and outputs the <fn> contents right under these elements. The <fn> elements contained in other elements will be ignored.

Specifying parameter $PRM_DISPLAY_FN_AT_END_OF_TOPIC="yes" will outputs the <fn> contents at the end of topic. (This parameter is deprecated and not included in build.xml.)

Absolute path reference in @href attribute of <image> element is not supported

This plug-in accepts @href attribute with relative path if scope is not specified as "external".

```
<image href="image/tys125f.jpg" placement="break">
```

Running header that uses `xsl:retrive-marker/@retrieve-position="first-including-carryover"` is not supported

This is the problem of stylesheet design. If you want implement this feature, you should generate `<fo:marker>` at topicref level.

Powered by

Visit us on the World Wide Web
http://www.antennahouse.com

Antenna House, Inc. Ina Branch
8077-1 Minaminowa Vil. Kamiina Co. Nagano Pref. 399-4511 Japan