

DocBook to LaTeX Publishing

User Manual

Benoît Guillon

COLLABORATORS

	<i>TITLE :</i> DocBook to LaTeX Publishing		<i>REFERENCE :</i> Ref A1
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	Benoît Guillon	7 November 2020	
REVIEWED BY	Andreas Hoenen	7 November 2020	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME
01	2009/05/05	The manual does not include the change history anymore. The change history is now in the release note. The XSL Parameters are now described as reference entries.	B. Guillon
02	2009/06/21	Add a section about the new <code>set</code> support, and a section about how to extend the verbatim rendering.	B. Guillon
03	2011/07/03	Add a section about the new HTML table support, and a section about the PyPI distribution.	B. Guillon
04	2012/04/10	Update documentation for release 0.3.3.	B. Guillon
05	2012/06/03	Update documentation for release 0.3.4.	B. Guillon
06	2014/05/26	Update documentation for release 0.3.5.	B. Guillon
07	2015/05/15	Update documentation for release 0.3.6: add the ability to use a <code>texpost python</code> plugin, and new index capabilities.	B. Guillon
08	2015/08/06	Update documentation for release 0.3.7: add section Section 4.6.	B. Guillon

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME
09	2016/07/20	Update documentation for release 0.3.8: add appendix Appendix B, and sections Section 5.7.1, Section 4.11.	B. Guillon / Karl O. Pinc
10	2016/07/24	Update documentation for release 0.3.9: add new features in Section 5.7.1 and complete some sections.	B. Guillon / Karl O. Pinc
11	2017/04/03	Update documentation for release 0.3.10: add the features of Section 4.5, and put some cosmetic changes in Section 5.7.	B. Guillon
12	2019/09/10	Update release note for version 0.3.11 that is a bugfix release.	B. Guillon
13	2020/01/10	Update release note for version 0.3.11-py3: dlatex fork for python3.	B. Guillon
14	2020/11/07	Update release note for version 0.3.12 that is a bugfix release.	B. Guillon / P. Reinholdtsen

Contents

1	Documentation	1
1.1	Reference	1
2	Introduction	2
2.1	What about DB2LaTeX?	2
2.2	Features	2
2.3	Version	3
2.4	Change History	3
2.5	Publishing Principles	3
2.5.1	Backend Drivers	3
2.5.2	XSL Stylesheets	4
2.5.3	Python Post Processing	4
2.5.4	LaTeX Style Package	4
3	Installing the Package	5
3.1	Content	5
3.2	Installing on Unix like Systems	5
3.2.1	Dblatex Packages	5
3.2.2	Dependencies	6
3.2.3	Installation	6
3.2.3.1	Installing the dependencies	6
3.2.3.2	Installing from the source tarball	6
3.2.3.3	Installing from the Python Egg	7
3.3	Installing on Windows	7
3.3.1	Dependencies	7
3.3.2	Installation	7
3.3.2.1	Installing xsltproc	7
3.3.2.2	Installing MiKTeX	7
3.3.2.3	Installing dblatex	8

4	Using dblatex	9
4.1	Reference	9
	<code>dblatex</code>	9
4.2	Output Formatting Style	12
	4.2.1 How it works	12
	4.2.2 Adding a New Formatting Style	13
4.3	Publishing Outputs	14
	4.3.1 Publishing a single document	14
	4.3.2 Publishing a Set of Books	14
4.4	Global Page Setup	15
4.5	Book Covers	16
4.6	Chapter and Section numbering	18
	4.6.1 Depth of Section numbering and Table Of Content	18
	4.6.2 Using the <code>latex</code> style for section numbering	18
	4.6.3 Using the <code>label</code> attribute	19
4.7	Figure Inclusion	19
	4.7.1 Presentation	19
	4.7.2 Converting on the fly	20
	4.7.3 Paths Lookup	20
4.8	Creating Tables	21
	4.8.1 Limitations	21
	4.8.2 Table Width	21
	4.8.3 Tables without <code>colwidth</code>	22
	4.8.4 Tables with mixed <code>colspec</code>	22
	4.8.5 Tables with proportional and fixed <code>colwidth</code>	22
	4.8.6 Tables with fixed <code>colwidths</code>	23
	4.8.7 Automatic Column Width	23
	4.8.7.1 Global Setting	23
	4.8.7.2 Local Setting	24
	4.8.8 Informal Table LaTeX Styles	25
	4.8.9 Tables with <code>morerows</code>	26
	4.8.10 Landscape tables	26
	4.8.11 Smaller tables	28
	4.8.12 Coloured tables	28
	4.8.13 HTML Tables	29
4.9	Correcting Poor Formatting	30
	4.9.1 Floats	31
	4.9.2 Tables	31
	4.9.3 Examples	31

4.9.4	Hyphenation and over-long lines	32
4.9.5	Characters and Manual Spacing	32
4.10	Writing Mathematics	32
4.10.1	Writing LaTeX Mathematical Equations	32
4.10.1.1	Presentation	32
4.10.1.2	Implementation choice	33
4.10.1.3	Mathematical Delimiters	33
4.10.1.4	Compatibility	33
4.10.1.5	Examples	34
4.10.2	Writing MathML equations	35
4.11	Musical Notation	35
4.12	Extending the Verbatim Rendering	35
4.12.1	Dblatex Specific Options	35
4.12.2	Scaling Feature	36
4.12.3	Formatting embedded elements	36
4.12.4	Creating a new Verbatim Environment	38
4.13	Creating an Index	38
4.13.1	Internationalized Indexes	38
4.13.1.1	Indexing Tools	38
4.13.1.2	Index Sorting	39
4.14	Writing a Bibliography	39
4.14.1	Using Bibliography Entries	40
4.14.2	Using BibTeX Databases	40
4.14.3	Natbib Citations	41
4.15	Document Revisions	41
4.16	Locale Support	42
4.16.1	Document Encoding	42
4.16.2	Babel Languages	42
4.16.3	CJK Languages	42
4.16.3.1	Korean Support	42
4.16.4	Mixing the languages	42
4.17	Using XRefsyle and Olinks	43
4.17.1	Specific xrefstyle for <code>ulink</code>	43
4.18	Footnotes	43
4.18.1	Footnote Numbering Scope	43
4.18.2	End Notes	43
4.18.2.1	Writing a document with endnotes	44
4.18.2.2	Setup Properties	44
4.18.2.3	Endnotes Setup from scratch	45

4.18.2.4	Grouping Endnotes	45
4.18.2.4.1	Adding new Groups	45
4.18.2.4.2	Formatting the Headings text	46
4.18.2.4.3	LaTeX Command to make the Headings	46
4.18.2.4.4	Endnotes Counter Scope	46
5	Customization	47
5.1	Using XSL Parameters	47
5.2	Setting Command line Parameters	47
5.3	Using Processing Instructions	47
5.4	XSL User Stylesheet	48
5.4.1	Changing the XSL parameter values	48
5.4.2	Overriding some templates	49
5.5	Customized LaTeX style	50
5.5.1	Reusing an existing LaTeX style	50
5.5.2	Package options	51
5.5.3	Needed packages	51
5.5.4	DocBook interface	51
5.5.5	Debugging your Style	52
5.6	Latex post process script	52
5.6.1	Post latex compilations	53
5.6.2	Post processing with a Python Plugin	53
5.7	Dblatex Configuration File	54
5.7.1	XML Configuration File Format	54
5.7.1.1	<config>	55
5.7.1.2	<latex>	55
5.7.1.3	<xslt>	55
5.7.1.4	<engine>	56
5.7.1.5	<imagedata>	56
5.7.1.6	<converter>	57
5.7.1.7	<command>	57
5.7.1.8	<options>	58
5.7.2	Deprecated Text Configuration File Format	58
5.7.3	Style Paths	59
5.8	Customization Precedence	59

6	FAQ	61
6.1	My images are too big. What can I do?	61
6.2	How can I have the PDF fit to height by default?	61
6.3	How can I have all the PDF hyperlinks in blue color?	61
6.4	How can I remove that stupid float rules?	62
6.5	My long tables don't split in several pages. Why?	62
6.6	I cannot put a table in an example.	62
6.7	I cannot compile my cyrillic document. Why?	62
7	Thanks	63
7.1	Sponsors	63
7.2	Contributors	63
7.3	Pioners	63
A	Dblatex XSL Parameter Reference	64
A.1	Admonitions	64
	figure.caution	64
	figure.important	64
	figure.note	65
	figure.tip	65
	figure.warning	65
A.2	Callouts	65
	callout.linkends.hot	65
	calloutlist.style	66
	callout.markup.circled	66
	co.linkends.show	66
	imageobjectco.hide	66
A.3	ToC/LoT/Index Generation	67
	doc.lot.show	67
	doc.toc.show	67
	titleabbrev.in.toc	67
	toc.section.depth	68
	bibliography.tocdepth	68
	colophon.tocdepth	68
	dedication.tocdepth	69
	preface.tocdepth	69
	glossary.tocdepth	69
	index.tocdepth	70
	latex.index.tool	70

	latex.index.language	71
	refentry.tocdepth	71
A.4	Processor Extensions	71
	alt.use	71
	tex.math.in.alt	72
A.5	Automatic labelling	72
	bibliography.numbered	72
	glossary.numbered	72
	index.numbered	73
	refentry.numbered	73
A.6	Meta/*Info	74
	doc.pdfcreator.show	74
	make.single.year.ranges	74
	make.year.ranges	74
A.7	Reference Pages	75
	funcsynopsis.decoration	75
	funcsynopsis.style	75
	function.parens	75
	refclass.suppress	75
	refentry.generate.name	76
	refentry.xref.manvolnum	76
A.8	Tables	76
	newtbl.autowidth	76
	newtbl.bgcolor.thead	77
	newtbl.default.colsep	77
	newtbl.default.rowsep	78
	newtbl.format.tbody	78
	newtbl.format.tfoot	78
	newtbl.format.thead	78
	newtbl.use.hhline	79
	table.continue.caption	79
	table.default.position	79
	table.default.tabstyle	80
	table.in.float	80
	table.title.top	81
	default.table.rules	81
	default.table.width	82
A.9	Linking	82
	latex.hyperparam	82

Olink Parameters	83
A.10 Cross References	84
insert.xref.page.number	84
insert.xref.page.number.para	84
xref.hypermarkup	85
A.11 Lists	85
term.breakline	85
variablelist.term.separator	85
A.12 QAndASet	86
qanda.defaultlabel	86
A.13 Bibliography	86
biblioentry.item.separator	86
biblioentry.numbered	86
citation.default.style	87
citation.natbib.options	87
citation.natbib.use	87
latex.bibfiles	88
latex.biblio.output	88
latex.biblio.style	88
latex.bibwidelabel	89
A.14 Glossary	89
glossterm.auto.link	89
A.15 Miscellaneous	89
annotation.support	89
beginpage.as.pagebreak	90
doc.section.depth	90
endnotes.heading.command	90
endnotes.heading.groups	91
endnotes.heading.style	91
endnotes.properties	91
equation.default.position	92
example.default.position	92
example.float.type	93
figure.anchor.top	93
figure.default.position	94
figure.title.top	94
filename.as.url	94
footnote.as.endnote	94
hyphenation.format	95

linenumbering.scope	95
linenumbering.default	96
linenumbering.everyNth	96
literal.layout.options	96
literal.lines.showall	97
literal.width.ignore	97
literal.class	97
literal.role	97
literal.environment	98
literal.extensions	98
mediaobject.caption.style	99
monoseq.hyphenation	99
monoseq.small	99
pdf.annot.options	100
segmentedlist.as.table	100
seg.item.separator	101
show.comments	101
texlive.version	101
ulink.footnotes	102
ulink.show	102
xref.with.number.and.title	103
A.16 Graphics	103
imagedata.boxed	103
imagedata.default.scale	103
imagedata.file.check	104
keep.relative.image.uris	104
A.17 Chuncing	104
set.book.num	104
use.id.as.filename	105
A.18 Pagination and General Styles	105
page.height	105
page.margin.bottom	105
page.margin.inner	106
page.margin.outer	106
page.margin.top	106
page.width	107
paper.type	107
geometry.options	107
doc.alignment	108

doc.collab.show	108
doc.layout	108
doc.publisher.show	108
draft.mode	109
draft.watermark	109
latex.engine.options	109
latex.class.article	110
latex.class.book	110
latex.class.options	110
latex.encoding	111
latex.unicode.use	111
latex.output.revhistory	111
A.19 Font Families	111
body.font.family	111
cjk.font	112
monospace.font.family	112
sans.font.family	112
xetex.font	113
A.20 Localization	113
korean.package	113
latex.babel.language	114
latex.babel.use	114
A.21 Prepress	114
crop.marks	114
crop.paper.type	114
crop.page.width	115
crop.page.height	115
crop.mode	116
crop.options	116
B Dblatex Processing Instruction Reference	117
B.1 Bibliography	117
bibtex	117
dblatex citestyle	118
B.2 LaTeX	118
latex	118
B.3 Miscellaneous	120
dblatex angle	120
texmath delimiters	120

B.4	Tables	121
	<code>dblatex bgcolor</code>	121
	<code>dblatex table-width</code>	121
B.5	Lists	122
	<code>dblatex autowidth</code>	122
	<code>dblatex colwidth</code>	123
	<code>dblatex list-presentation</code>	124
8	Index	126

List of Figures

2.1 Transforming Process	3
4.1 Parameter Lengths used for Page Setup	16

List of Tables

4.1 [An HTML Table](#) 30

List of Examples

4.1	Choosing the DB2LaTeX style	12
4.2	DocBook 5 Front and Back Covers	17
4.3	DocBook 4 Front Cover	17
4.4	Figure inclusion	20
4.5	Figure conversion	20
4.6	Figures lookup	20
4.7	Equation taken from TDG	32
4.8	Equation with user delimiters	33
4.9	Inlined Equation	34
4.10	Equation in a block	34
4.11	Equation in a float	34
4.12	Equation without a title	34
4.13	Index Entry	38
4.14	XSL Index Language Setup	39
4.15	A Bibliography	40
4.16	Bibliography using BibTeX databases	41
4.17	Writing a document with endnotes	44
5.1	Table width specified with a Processing Instruction	48
5.2	Overriding templates	50
5.3	Reused LaTeX style	50
5.4	Texpost Python Plugin Example	53
5.5	User Manual Configuration File	54
5.6	Customization Precedence	59
A.1	Configuring with latex.hyperparam	83

Chapter 1

Documentation

1.1 Reference

[TDG] Norman Walsh and Leonard Muellner, *DocBook: The Definitive Guide*, Copyright © 1999, 2000, 2001 O'Reilly & Associates, Inc., 156592-580-7, O'Reilly.

Chapter 2

Introduction

2.1 What about DB2LaTeX?

Dblatex started as a **DB2LaTeX** clone, but since then many things have changed and new features have been added or (hopefully) improved. Now, the portion of shared code is small if any, and the **dblathex** purpose is different from **DB2LaTeX** on these points:

- The project is end-user oriented, that is, it tries to hide as much as possible the latex compiling stuff by providing a single clean script to produce directly DVI, PostScript and PDF output.
- The actual output rendering is done not only by the XSL stylesheets transformation, but also by a dedicated LaTeX package. The goal is to allow a deep LaTeX customisation without changing the XSL stylesheets.
- Post-processing is done by Python, to make publication faster, convert the images if needed, and do the whole compilation.

2.2 Features

With **dblathex** you can:

- transform a DocBook XML/SGML book or article to pure LaTeX,
 - compile the temporary LaTeX file with **latex**, **pdflatex**, or **xelatex** to produce DVI, PostScript and PDF files,
 - publish a set of books,
 - convert on the fly the figures included in the document,
 - have cross references with hot links,
 - olink to other documents built with **dblathex**,
 - write complex tables,
 - write several bibliographies,
 - reuse BibTeX bibliographies,
 - use callouts on program listings or on images,
 - create an index with **makeindex** or **xindy**,
 - write mathematical equations in LaTeX,
 - write mathematical equations in MathML,
 - have revision bars,
 - customise the output rendering with an XSL configuration file,
 - use your own LaTeX style package.
-

2.3 Version

This manual is for dblatex version *0.3.12*.

2.4 Change History

See the [Release Notes](#) in *Release Notes for dblatex* to have the dblatex change history.

2.5 Publishing Principles

Dblatex transforms a DocBook XML/SGML document to LaTeX. Once transformed into LaTeX, standard LaTeX tools are used to produce DVI, Postscript or PDF files.

Figure 2.1 explains the process applied. It shows the tools used and the steps. The emphasized tools are provided by the package.

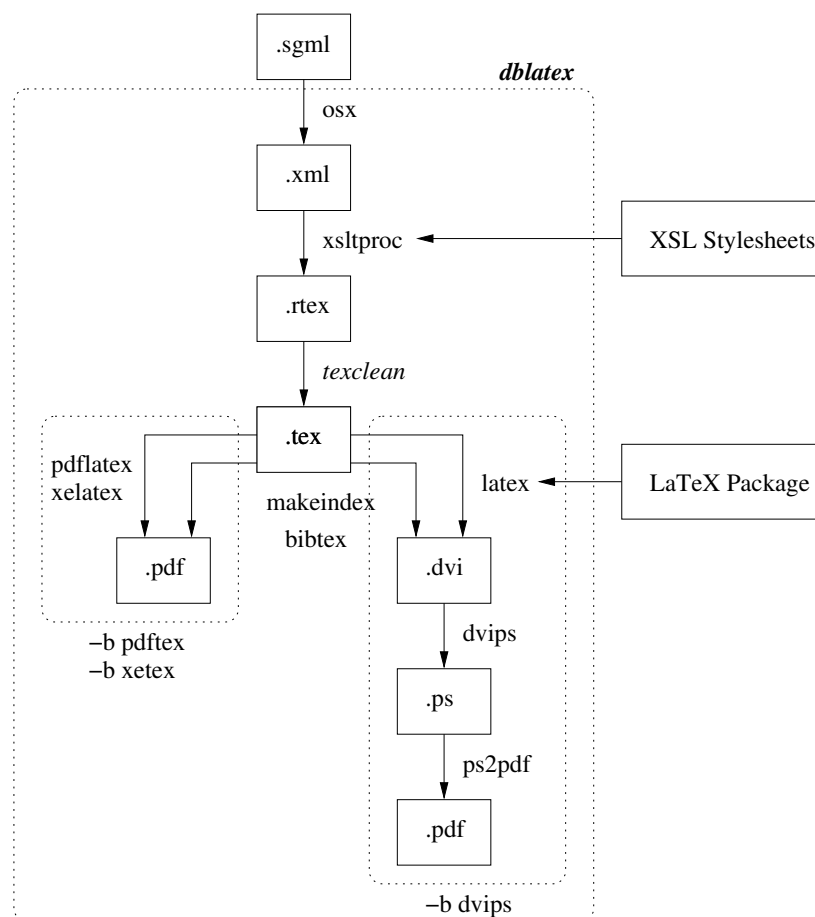


Figure 2.1: Transforming Process

2.5.1 Backend Drivers

The main script supports the following LaTeX backend drivers:

dvips

The driver calls **latex**, and produces DVI, Postscript and at the end PDF files. Latex natively accepts only EPS graphics. The drawback is that converting to PDF can take a while.

pdftex

The driver calls **pdflatex**, to directly produce PDF files. The conversion is fast, the file size is smaller. Pdflatex natively accepts PDF, PNG, JPEG, and TIFF graphics.

xetex

The driver calls **xelatex**, to directly produce PDF files through the **XeTeX** engine. This engine natively supports UTF-8 which improves multilingual support.

2.5.2 XSL Stylesheets

The XSL stylesheets located under `xsl/` are used to transform from XML to “raw” LaTeX. The main file is `latex_book_fast.xsl`, that includes the other stylesheets of the directory.

2.5.3 Python Post Processing

Actually the XSL stylesheets does not produce valid LaTeX. The reason is that some DocBook processing is too complex or too time-consuming for XSL transforming. Besides, some extra actions need sometimes to be done such like figure conversion. Here are the main actions done by Python Post processing:

- Transform the entities to valid LaTeX characters (e.g. ` ` is transformed to `'~'`). Python is suited and performant for this task.
- Convert the figures to be compatible with the backend driver. See Section 4.7 for more detail.
- Force some hyphenation in tables or for typed words.
- Do the whole LaTeX compilation sequence thanks to the **rubber** compilation engine.

2.5.4 LaTeX Style Package

Once valid LaTeX is available, the LaTeX style package (`docbook.sty`) under `latex/style/` is used to customize the output rendering. It includes the other files of the directory. You can also provide your own LaTeX style (cf. Chapter 5).

Chapter 3

Installing the Package

3.1 Content

The source package contains the following:

docs/

Contains the files of this document.

etc/

Contains the XML configuration schemas, to use to validate your configuration files.

latex/

Contains all the latex stuff: LaTeX style files, logos, and scripts to compile the LaTeX output.

scripts/

Several scripts, including the main script of the package.

xsl/

XSL stylesheets.

tests/

Test files.

3.2 Installing on Unix like Systems

3.2.1 Dblatex Packages

Dblatex is packaged for these Systems or Distributions:

- [Linux Debian, Ubuntu,](#)
- [Linux OpenSUSE \(RPM\), Linux Fedora \(RPM\),](#)
- [FreeBSD, NetBSD,](#)
- [Mac OS X \(Fink\).](#)

If you are installing on one of these distributions, follow their recommended way of installation, and you can safely ignore the next sections that give details for installing dblatex from the source tarball.

3.2.2 Dependencies

To work, the following items must be available:

- An XSLT. `xsltproc` is the default XSLT used, but one can also use [4suite](#) or [saxon](#).
- The XML DocBook DTD.
- A recent LaTeX distribution. The configure script checks that the needed latex packages are available.
- Python ≥ 3.2 .

3.2.3 Installation

3.2.3.1 Installing the dependencies

To use the package, install properly the dependencies:

1. Install Python if necessary.
2. Install LaTeX.
3. Install the XSLT. By default `xsltproc` is used.
4. Install the XML DocBook DTD.
5. Create a catalog file, that defines where to find the DTD. Here is an example:

```
PUBLIC "-//OASIS//DTD DocBook XML V4.1.2//EN"
    "file:///usr/local/share/xml/docbook/dtd/4.1.2/docbookx.dtd"
```

If the XML Gnome tools are available, it's a good idea to create an XML catalog by using `xmlcatalog` such like this:

```
% xmlcatalog --noout --create mycatalog
% xmlcatalog --noout --add 'public' '-//OASIS//DTD DocBook XML V4.1.2//EN' \
    'file://path/to/4.1.2/docbookx.dtd' mycatalog
```

6. Add the catalog path to the `SGML_CATALOG_FILES` variable:

```
export SGML_CATALOG_FILES=$SGML_CATALOG_FILES:/path/to/mycatalog
```

You can skip this step if you configure the `dblatex` installation with the `--catalogs` option.

3.2.3.2 Installing from the source tarball

The steps to follow are the following:

1. Untar the ball. For a bziped release, do as follow:

```
% tar xvfj dblatex-x.x.x.tar.bz2
```

For a gzipped release, do as follow:

```
% tar xvfz dblatex-x.x.x.tar.gz
```

2. Install the package. The installation script preliminary checks the dependencies. In the example, the `dblatex` script is installed under `/usr/local/bin` and the other files are installed under `/usr/local/share/dblatex`. Besides, the `--catalogs` option tells where to find the catalogs.

```
% cd dblatex-x.x.x
% python ./setup.py install --prefix=/usr/local --catalogs=/path/to/mycatalog
```

3.2.3.3 Installing from the Python Egg

Since version 0.3.2 dblatex is distributed as a Python Egg in the [Python Package Index \(PyPI\)](#). It requires to have `easy_install` installed.

The procedure to follow is:

1. Call simply `easy_install`:

```
% easy_install dblatex
Searching for dblatex
Reading http://pypi.python.org/simple/dblatex/
Reading http://dblatex.sf.net
Best match: dblatex 0.3.1.1
Downloading http://pypi.python.org/packages/2.5/d/dblatex/dblatex-0.3.1.1-py2.5.egg#md5 ←
=8520d596e473ff544eb9dc5175d7ae
Processing dblatex-0.3.1.1-py2.5.egg
creating /path/to/dblatex-0.3.1.1-py2.5.egg
Extracting dblatex-0.3.1.1-py2.5.egg to /path/to
Adding dblatex 0.3.1.1 to easy-install.pth file
Installing dblatex script to /path/to

Installed /path/to/dblatex-0.3.1.1-py2.5.egg
```

That's it!

3.3 Installing on Windows

The following packages to install and the procedure is for a native Windows installation. If you want to use dblatex via Cygwin instead, you should consider it like a unix like install.

3.3.1 Dependencies

The following applications are required:

- An XSLT. `xsltproc` is the default XSLT used, but one can also use [4suite](#).
- The XML DocBook DTD.
- [MiKTeX](#) > 2.5.
- [Python](#) >= 3.2.3.

3.3.2 Installation

3.3.2.1 Installing xsltproc

You can download the binaries and getting the installation instructions from: <http://www.zlatkovic.com/libxml.en.html>.

3.3.2.2 Installing MiKTeX

Install the minimal distribution, and add the following packages: `changebar`, `colortbl`, `fancybox`, `fancyhdr`, `fancyvrb`, `listings`, `overpics`, `rotating`, `subfigure`, `titlesec`, `bibtopic`, `enumitem`, `eepic`, `lm`, `lastpage`, `helvetic`, `times`, `symbol`, `courier`, `footmisc`, `ifxetex`, `pdfpages`, `wasysym`.

3.3.2.3 Installing dblatex

From the unpacked package directory just type:

```
python setup.py install
```

If the Python directory is C:\Python25 you can now try **dbl**atex by typing:

```
python C:\Python25\Scripts\dbl
```

atex file.xml

Chapter 4

Using dblatex

4.1 Reference

dblatex

dblatex — convert DocBook to LaTeX, DVI, PostScript, and PDF

Synopsis

```
dblatex [options] {file|-}
```

Description

dblatex is a program that transforms your SGML/XML DocBook documents to DVI, PostScript or PDF by translating them into pure LaTeX as a first process. MathML 2.0 markups are supported, too.

Options

A summary of options is included below.

-h, --help

Show a help message and exit.

-b backend, --backend=backend

Backend driver to use: *pdftex* (default), *dvips*, or *xetex*. See also Section 2.5.1.

-B, --no-batch

All the tex output is printed.

-c config, -S config, --config=config

Configuration file. A configuration file can be used to group all the options and customizations to apply. See Section 5.7.

-d, --debug

Debug mode: Keep the temporary directory in which dblatex actually works. Section 5.5.5 explains how you can use it.

-D, --dump

Dump the error stack when an error occurs (debug purpose).

-e indexstyle, --indexstyle indexstyle

Index style file to pass to **makeindex** instead of the dblatex default index style.

- f *figure_format*, --fig-format=*figure_format***
Input figure format: *fig*, *eps*. Used when not deduced from figure file extension. See also Section 4.7.2.
 - F *input_format*, --input-format=*input_format***
Input file format: *sgml*, *xml* (default).
 - i *texinputs*, --texinputs *texinputs***
Path added to TEXINPUTS
 - I *figure_path*, --fig-path=*figure_path***
Additional lookup path of the figures. See Section 4.7.3.
 - l *bst_path*, --bst-path=*bst_path***
Additional lookup path of the BibTeX styles. See Section 4.14.2.
 - L *bib_path*, --bib-path=*bib_path***
Additional lookup path of the BibTeX databases. See Section 4.14.2.
 - m *xslt*, --xslt=*xslt***
XSLT engine to use. The available engines are: xsltproc (default), 4xslt, saxon.
 - o *output*, --output=*output***
Output filename. When not specified, the input filename is used, with the suffix of the output format. The option is ignored if several books are chunked from a set. In this case the -O option is applied instead.
 - O *output_dir*, --output-dir=*output_dir***
Output directory of the books built from a set. When not specified, the current working directory is used instead. The option is ignored if a single document is outputted, and the -o is taken into account.
 - p *xsl_user*, --xsl-user=*xsl_user***
An XSL user stylesheet to use. Several user stylesheets can be specified, but the option order is meaningful: a user stylesheet takes precedence over previously defined user stylesheets. See Section 5.1.
 - P *param=value*, --param=*param=value***
Set an XSL parameter from command line. See Section 5.2.
 - q, --quiet**
Less verbose, showing only TeX output messages and error messages.
 - r [*plugin:*] *script*, --texpost=[*plugin:*] *script***
Script called at the very end of the tex compilation. Its role is to modify the tex file or one of the compilation files before the last round. The *script* can be a python plugin. In this case add the prefix term 'plugin:'. See Section 5.6.
 - s *latex_style*, --texstyle=*latex_style***
Latex style to apply. It can be a package name, or directly a latex package path. A package name must be without a directory path and without the '.sty' extension. On the contrary, a full latex package path can contain a directory path, but must ends with the '.sty' extension. See Section 5.5.
 - t *format*, --type=*format***
Output format. Available formats: *tex*, *dvi*, *ps*, *pdf* (default).
 - dvi**
DVI output. Equivalent to -tdvi.
 - pdf**
PDF output. Equivalent to -tpdf.
 - ps**
PostScript output. Equivalent to -tps.
 - T *style*, --style=*style***
Output style, predefined are: *db2latex*, *simple*, *native* (default). See Section 4.2.
-

-v, --version

Display the dblatex version.

-V, --verbose

Verbose mode, showing the running commands

-x *xslt_options*, --xslt-opts=*xslt_options*

Arguments directly passed to the XSLT engine

-X, --no-external

Disable the external text file support. This support is needed for callouts on external files referenced by `textdata` or `imagedata`, but it can be disabled if the document does not contain such callouts. Disabling this support can improve the processing performance for big documents.

Files and Directories**`$HOME/.dblatex/`**

User configuration directory.

`/etc/dblatex/`

System-wide configuration directory.

The predefined output styles are located in the installed package directory.

Environment Variables**`DBLATEX_CONFIG_FILES`**

Extra configuration directories that may contain some dblatex configuration files.

Examples

To produce `myfile.pdf` from `myfile.xml`:

```
dblatex myfile.xml
```

To set some XSL parameters from the command line:

```
dblatex -P latex.babel.language=de myfile.xml
```

To use your XSL stylesheet:

```
dblatex -p myconfig.xsl myfile.xml
```

To use the `db2latex` output style:

```
dblatex -T db2latex myfile.xml
```

To apply your own latex style:

```
dblatex -s mystyle myfile.xml
dblatex -s /path/to/mystyle.sty myfile.xml
```

To pass extra arguments to the XSLT engine:

```
dblatex -x "--path /path/to/load/entity" myfile.xml
```

To use **dblatex** and profiling:

```
xsltproc --param profile.attribute "'output'" \
  --param profile.value "'pdf'" \
  /path/to/profiling/profile.xsl \
  myfile.xml | dblatex -o myfile.pdf -
```

To build a set of books:

```
dblatch -O /path/to/chunk/dir -Pset.book.num=all myfile.xml
```

4.2 Output Formatting Style

The output rendering done by **dblatch** can be widely customized like explained in Chapter 5. By default several rendering styles are provided, that one can choose by using the option `-T` (see Example 4.1). The available styles are:

native

The rendering uses the default LaTeX stylesheets. It is the style used by default if **dblatch** has been configured without using the option `--style`.

simple

The rendering is very close to original latex rendering. The wrapper around the default latex packages is very thin.

db2latex

The rendering tries to be as close as possible to the **DB2LaTeX** formatting.

Example 4.1 Choosing the DB2LaTeX style

```
dblatch -T db2latex file.xml
```

4.2.1 How it works

The rendering style stuff is under the `latex/` directory. You can see the XSL stylesheets under `xsl/` as the way to produce latex with as little as possible docbook specific things (even if a large amount of latex packages are used to do the work). Then, it's up to LaTeX stylesheets to format the document as you wish.

The organization under `latex/` is as follow:

contrib

Contains the non-default available LaTeX stylesheets (simple and db2latex).

graphics

Default graphics used in the admonitions (e.g. warning). These graphics are used by the default output formatting.

scripts

Scripts used to compile with **latex** or **pdflatex**.

specs

Contains all the configuration files describing the available styles. A specification file must have the extension `.specs`, `.conf`, or `.xml` to be detected as a style description, and its basename is the name of the style. For example the style **db2latex** is described by the configuration file `db2latex.xml`.

When **dblatch** is executed with no parameter, the usage is displayed. In particular, the list of the available styles is given, like this:

```
$ dblatex
dblathex [options] file.{sgml|xml}
Options:
-t {pdf|ps|dvi|tex|xml}: output format
...
-T style                : available latex styles (db2latex, native, simple)
```

The list is built by scanning the specs files found under `specs/`. The spec file syntax is described in Section 5.7.

style

Default LaTeX stylesheets.

4.2.2 Adding a New Formatting Style

To add a new formatting style, do the following steps:

1. Let's create the style directories that will contain all the specific data. We choose to put them under the default **dblathex** user configuration directory.

```
$ mkdir -p $HOME/.dblathex/mystyle/latex
$ mkdir -p $HOME/.dblathex/mystyle/xsl
```

Note that you could choose another configuration directory (see Section 5.7.3 for more details).

2. Create the latex stylesheets you need. It must define the expected DocBook interface and include some core definitions from the default latex stylesheets (cf. Section 5.5). Create also your XSL stylesheet if necessary.
3. Put these files under the appropriate directories:

```
$ mv mytexstyle.sty $HOME/.dblathex/mystyle/latex/.
$ mv param.xsl $HOME/.dblathex/mystyle/xsl/.
```

4. Create a configuration file under the directory `$HOME/.dblathex`. The configuration file must point to the new latex stylesheet, and give the specific parameters. Example:

```
$ cat $HOME/.dblathex/mystyle.conf
<?xml version="1.0" encoding="utf-8"?>
<!-- =====
      Dblathex config file for my new style.
      Note that the directories are relative to mystyle.conf
      ===== -->
<config xmlns="http://dblathex.sourceforge.net/config">
  <latex>
    <texinputs>mystyle/latex//</texinputs>
    <texstyle use="mytexstyle"/>
  </latex>
  <xslt>
    <stylesheet fileref="mystyle/xsl/param.xsl"/>
  </xslt>
  <options>-f fig</options>
</config>
```

5. That's it. Try to compile your document with your style, and check the output. The configuration file basename is the name of the style to call.

```
$ dblathex -T mystyle file.xml
```

4.3 Publishing Outputs

4.3.1 Publishing a single document

The default publishing document units are: `article` and `book`. The output file name is optionnaly specified by the `-o` option.

You can also publish an article or book subset, i.e. you can run `dblatex` on an XML input whose root element is a `chapter`, a `section`, or anything else. In this case, `dblatex` wraps the root element in an `article` or in a `book` and print out a warning. The output subset does not contain any front matter data found in an article or in a book (cover page, revision history, etc.), but it can contain some back matter materials like an index.

```
$ dblatex subset.xml
Build the book set list...
Build the listings...
XSLT stylesheets DocBook - LaTeX 2e (0.2.11)
=====
Warning: the root element is not an article nor a book
Warning: element section(sec-subset) wrapped with article
Build subset.pdf
...
```

4.3.2 Publishing a Set of Books

When the document root element is a `set`, and when `set.book.num` is set to 'all', `dblatex` outputs a file per book contained in the set (and in the nested sets). In this case the `-o` option is ignored, and only the `-O` option is taken into account to specify the output directory that will contain the generated files.

Instead of building all the books, the user can publish a single book from the set, by setting the `set.book.num` parameter to the absolute position of the book in the set(s). By default `set.book.num` is set to 1 to publish only the first book.

The output file names are the book identifiers when `use.id.as.filename` is non zero, and when an identifier exists. If one of the two conditions are not met, the filename pattern is "book<position in set>".

Example: given the following set:

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- setfile.xml. An example of set. All the books have an @id except one -->

<!DOCTYPE set PUBLIC "-//OASIS//DTD DocBook XML V4.4//EN"
"http://www.oasis-open.org/docbook/xml/4.4/docbookx.dtd">
<set lang="en" id="a_set" xmlns:xi="http://www.w3.org/2001/XInclude">
<title>Set Title</title>
  <set>
    <xi:include href="book1.xml"/> <!-- book #1 -->
    <xi:include href="book2.xml"/> <!-- book #2 -->
    <xi:include href="book3.xml"/> <!-- book #3 -->
  </set>
  <set>
    <set>
      <xi:include href="bookA.xml"/> <!-- book #4 -->
      <xi:include href="bookB.xml"/> <!-- book #5 -->
    </set>
    <set>
      <xi:include href="bookC.xml"/> <!-- book #6 -->
    </set>
  </set>
  <set>
    <xi:include href="book4.xml"/> <!-- book #7 -->
    <!-- The following book, at 8th position in the sets, has no @id -->
```

```

    <xi:include href="book5.xml"/>    <!-- book #8 -->
    <xi:include href="book6.xml"/>    <!-- book #9 -->
  </set>
</set>

```

Publishing this set produces 9 books in the `pdfdir` directory:

```

$ dblatex -O./pdfdir -Pset.book.num=all -Puse.id.as.filename=1 setfile.xml
Build the book set list...
Build the listings...
XSLT stylesheets DocBook - LaTeX 2e (0.2.11)
=====
Output all the books from the set
Writing sec1-mybook.rtex for book(sec1-mybook)
Writing sec2-mybook.rtex for book(sec2-mybook)
Writing sec3-mybook.rtex for book(sec3-mybook)
Writing secA-mybook.rtex for book(secA-mybook)
Writing book8.rtex for book
Writing secC-mybook.rtex for book(secC-mybook)
Writing sec4-mybook.rtex for book(sec4-mybook)
Writing sec5-mybook.rtex for book(sec5-mybook)
Writing sec6-mybook.rtex for book(sec6-mybook)
...
Files successfully built in '/path/to/set/pdfdir':
sec1-mybook.pdf
sec2-mybook.pdf
sec3-mybook.pdf
sec4-mybook.pdf
book8.pdf
sec6-mybook.pdf
secA-mybook.pdf
secB-mybook.pdf
secC-mybook.pdf

```

4.4 Global Page Setup

Since version 0.3.3 the user can specify to **dblatex** the global page layout to apply, through some XSL parameters (see Section A.18). It is also possible to produce pre-press layouts with physical pages having some crops (see Section A.21).

Figure 4.1 shows the meaning of each parameter length. Of course the example is a page with crops, only to be able to display the crop lengths with the other lengths.

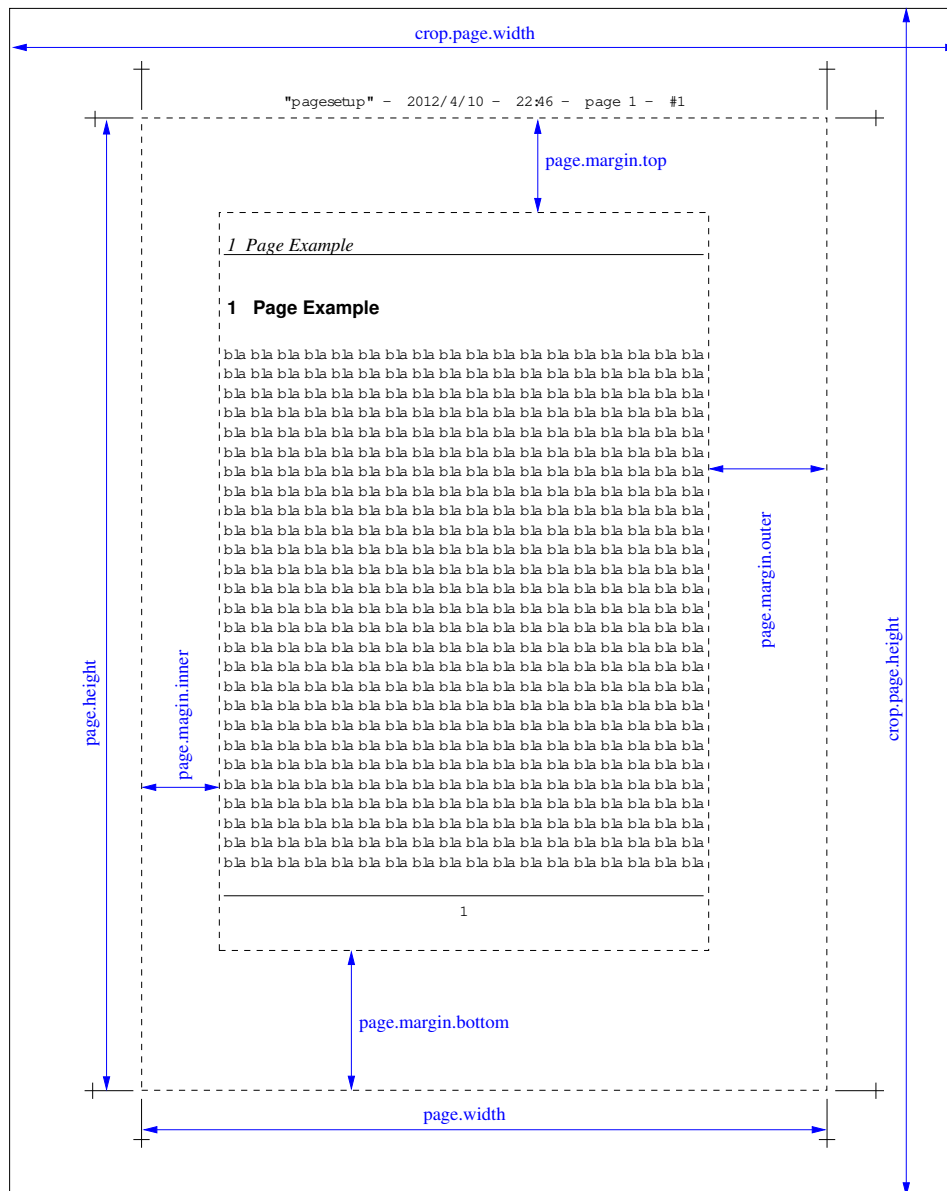


Figure 4.1: Parameter Lengths used for Page Setup

**Warning**

The native dlatex style still contains some hard-coded lengths to format the `revhistory` table. You should then disable the revision history display.

4.5 Book Covers

Since version 0.3.10 dlatex provides a basic support for page-size covers, when covers are defined as images to insert at the beginning of the document (front cover), or at the very end of the document (back cover).

To specify a cover, set in the book information element, `info` or `bookinfo`, the images to use in some cover elements (DocBook 5) or in `mediaobject` with role set to `cover` (DocBook 4).

If two covers are specified, it is assumed that the first one is for the front cover, and the second one for the back cover. You can play with standard `imagedata` width or depth attributes to ensure that the image is sized to fit the page covers.

Here are two examples:

Example 4.2 DocBook 5 Front and Back Covers

```
<book>
<info>
  ...
  <cover>
    <mediaobject>
      <imageobject>
        <imagedata fileref="cover-front-72dpi.png"
                    format="PNG" depth="100%" width="100%" align="left"/>
      </imageobject>
    </mediaobject>
  </cover>
  <cover>
    <mediaobject>
      <imageobject>
        <imagedata fileref="cover-back-72dpi.png"
                    format="PNG" depth="100%" width="100%" />
      </imageobject>
    </mediaobject>
  </cover>
</info>
  ...
</book>
```

Example 4.3 DocBook 4 Front Cover

```
<book>
<bookinfo>
  ...
  <mediaobject role="cover">
    <imageobject role="front-large">
      <imagedata fileref="cover-front-72dpi.png" format="PNG" width="100%" />
    </imageobject>
    <imageobject role="front">
      <imagedata fileref="cover-front-10dpi.png" format="PNG" width="444" />
    </imageobject>
    <imageobject role="front-small">
      <imagedata fileref="cover-front-10dpi.png" format="PNG" width="444" />
    </imageobject>
    <imageobject role="thumbnail">
      <imagedata fileref="cover-front-10dpi.png" format="PNG" width="444" />
    </imageobject>
  </mediaobject>
  ...
</bookinfo>
  ...
</book>
```

Note that in DocBook 4, **dblatex** filters on the same roles for `mediaobject` (`cover`) or for `imageobject` (`front-large`) than the DocBook Project does when it implements covers in `epub` format.

dblatex defines default templates `front.cover` and `back.cover` to implement covers through images, but you can overwrite them to create your own method to build covers.

4.6 Chapter and Section numbering

4.6.1 Depth of Section numbering and Table Of Content

Dblatex relies on latex to automatically compute the chapter and section numbers. It also relies on latex to produce the headings with these numbers, and to produce the Table Of Content containing these entries.

Some specific sections like `preface`, `colophon`, or `dedication` are not numbered because they are displayed in front and back matters, but they can be listed in the TOC if the related parameters are set (`dedication.tocdepth`, `preface.tocdepth`, `colophon.tocdepth`).

More generally you can configure the depth of the chapter and section numbering with the parameter `doc.section.depth`, and the depth of the entries in the TOC with the parameter `toc.section.depth`.

Note however that an unnumbered section becomes an informal component, and therefore you can only link to such a section by using a specific xrefstyle because their label built with the automatic number does not exist anymore. The other drawback is that all the sections included in an unnumbered section or chapter are unnumbered.

4.6.2 Using the latex style for section numbering

A well customized latex style can be a powerfull yet versatile alternative to the use of section and/or TOC depth parameter. With some latex packages you can easily format the headings to remove the numbers, or transform them. See Section 5.5 to know how to use your own latex style with **dblatex**.

The benefit of this method is that you can fully control how the titles must be displayed, you do not have to play with latex counters to have the right depth, and the sections do not lost their formal number label.

The following example shows how you can customize the chapter title by using the latex package `titlesec`, and shows how to remove the chapter label in the Table Of Content with the package `titletoc`. Look in particular in the listing where text is emphasized.

```
\usepackage{titlesec}

%% Example 1: Redefines the heading to remove the chapter label
%% The 2nd parameter only contains \filcenter without any label
\titleformat{\chapter}[block]
{\filcenter\huge}{\filcenter}{20pt}{\Huge}

%% Example 2: Put the chapter number in word
%% The title of the first chapter is then displayed like this:
%% "Chapter One <chapter title>"
\newcommand\makeletterof[1]{%
\ifthenelse{\equal{#1}{1}}{ONE}{%
\ifthenelse{\equal{#1}{2}}{TWO}{%
\ifthenelse{\equal{#1}{3}}{THREE}{%
\ifthenelse{\equal{#1}{4}}{FOUR}{%
\ifthenelse{\equal{#1}{5}}{FIVE}{%
\ifthenelse{\equal{#1}{6}}{SIX}{%
\ifthenelse{\equal{#1}{7}}{SEVEN}{%
\ifthenelse{\equal{#1}{8}}{EIGHT}{%
\ifthenelse{\equal{#1}{9}}{NINE}{%
\ifthenelse{\equal{#1}{10}}{TEN}{%
\ifthenelse{\equal{#1}{11}}{ELEVEN}{%
\ifthenelse{\equal{#1}{12}}{TWELVE}{%
#1}}}}}}}}}}}}}}

\titleformat{\chapter}[block]
{\filcenter\ttfamily\huge}%
{\filcenter\MakeUppercase{\chaptertitlename} \makeletterof{\thechapter}}{20pt}{\Huge}
```

```

%% Make TOC entries for chapters without label
\usepackage{titletoc}

\titlecontents{chapter} %
[1.5em] %
{\addvspace{1em plus 0pt}\bfseries} %
{\hspace{-1.3em}} % no number, remove room reserved for it
{\hspace{-1.3em}} %
{\hfill \contentspage} %
[\addvspace {0pt}]

```

4.6.3 Using the label attribute

If you need to mix numbered and not numbered chapters, the numbering depth parameters will not help. The latex style will not easily detect that for some chapters the number should be displayed and for others it should not.

To mix numbered and unnumbered chapters you can use a label with an empty string for the chapters that must not be numbered, as shown by the example below.

```

<chapter id="intro" label=""><title>Introduction</title>
  <!-- This chapter must be displayed with no number, like a preface -->
</chapter>
<chapter id="before" label=""><title>Pre-requisite</title>
  <!-- This chapter must be displayed with no number, like a preface -->
</chapter>
<chapter id="chap1"><title>First chapter</title>
  <!-- First numbered chapter -->
</chapter>
...
<chapter id="after" label=""><title>Conclusion</title>
  <!-- This chapter must be displayed with no number, like a colophon -->
</chapter>
...

```

You can also set a label to a specific integer to force a section counter. In this case there is no more automatic numbering. The automatic numbering then applies to the following sections that increment the counter set for this section if there is no label.

```

<chapter id="intro" label="2"><title>Introduction</title>
  <!-- This chapter is forced to have number 2 (it should be 1) -->
  <section id="s1" label="3"><title>Section 2.3</title>
    <!-- This section is forced to have number 3 (it should be 1) -->
    <section id="s11" label="4"><title>Section 2.3.4</title>
      <!-- This section is forced to have number 4 (it should be 1) -->
    </section>
  </section>
</chapter>

```

4.7 Figure Inclusion

4.7.1 Presentation

The expected format of the included figures depends on the backend driver used:

dvips:

EPS format is required.

pdftex:

PDF or PNG format is required.

In order to be able to use both backends, it is wise to not write the suffix of the file that references the figure. The suffix will be deduced from the backend used.

The figures must either already exists in the expected format, or must be able to be converted on the fly.

Example 4.4 Figure inclusion

```
<figure id="fig-exemple1">
  <title>Components</title>
  <mediaobject>
    <imageobject>
      <imagedata fileref="path/figure1" align="center" scale="70">
    </imageobject>
  </mediaobject>
</figure>
```

4.7.2 Converting on the fly

When it is needed dlatex tries to automatically convert the figures to the expected format (i.e. EPS or PDF). The principle is to detect the original figure format from the suffix of the fileref attribute. If no suffix is given, the tool checks if a file whose basename is conformant with the fileref attribute and with one of the predefined suffixes exists (that is, ".eps", ".fig", ".pdf", or ".png"). If such a file exists, conversion is done from the original format found.

The option `-f fig_format` allows to specify the default included figures format (*fig_format*), that will be used when automatic format scanning gives no result. Then, the tool converts the figures from the specified format to the expected one.

If the specified format is unknown, no conversion is done. The supported formats are:

fig:

native format of the figures produced by XFig.

eps:

Encapsulated PostScript format. This format shall be specified only when using the pdftex backend.

Example 4.5 Figure conversion

The following command compiles a document that contains figures produced with XFig.

```
% dlatex -f fig mydoc.sgml
```

4.7.3 Paths Lookup

You can use and cumulate the option `-I path` to specify where the figures are. The given paths can be absolute or relative. The paths are added to the document root path.

Example 4.6 Figures lookup

This example shows how figure lookup is done. Let's consider this document source:

```
<figure id="fig-example1">
  <title>Composants</title>
  <mediaobject>
    <imageobject>
      <imagedata fileref="rep1/rep2/figure1" align="center" scale="70">
    </imageobject>
  </mediaobject>
</figure>
```

And the document is compiled like this:

```
% dblatex -I /another/path -I /last/case /initial/path/document.sgml
```

The figure1 lookup is done in the following directories, in respect of the order:

- /initial/path/rep1/rep2;
- /another/path/rep1/rep2;
- /last/case/rep1/rep2.

4.8 Creating Tables

DocBook tables can be quite complex, but **dblatex** should be able to drive most of cases thanks to the excellent newtbl implementation by David Hedley completely written in XSL.

Here is what is supported:

- Columns without specified widths (`colspec` without `colwidth` attribute) have the same size.
- A table width is always equal to the page width, if at least one column doesn't contain a fixed width attribute (e.g. `colwidth="12cm"`).
- Fixed column widths are supported (e.g. `colwidth="10cm"`). The unit can be whatever is understood by latex (e.g. cm, em, in, pt).
- Proportional column widths are supported (e.g. `colwidth="5%"`). Combination of fixed and proportional width is supported too (e.g. `colwidth="5*+10cm"`).
- The `morerows` attribute of a table entry (`entry` element) is supported.
- The `namest` and `nameend` attributes of a table entry (`entry` element) are supported. It is possible to have a cell spanned on several columns.
- The `orient` table attribute is supported (portrait and landscape).
- It is possible to have missing cell entries in a table.

4.8.1 Limitations

Currently the following things are known to fail with tables:

- Program listings and screens cannot be embedded in tables. Some other verbatim environments like `litterallayout` are allowed.
- Footnotes in table cells can fail, especially if the footnote contains several paragraphs. Moreover they are lost if a float like a table.

4.8.2 Table Width

A table width is explicit when all the columns have a fixed size. In this case it is the sum of the column widths. In the other cases (columns with no `colwidth` or proportional column widths) the total table width is deduced by `dblatex` as follows: it looks for the Processing Instruction `<?dblatex table-width="width" ?>` first, then the `@width` attribute, the `default.table.width` parameter, and finally use the page width.

A table width can be expressed as:

- A valid length (e.g. 15cm),
- A percentage of the page width (e.g. 75%),
- A keyword telling to apply an automatic column width (e.g. `autowidth.all`).

The automatic column width setup is detailed in Section [4.8.7](#).

4.8.3 Tables without colwidth

When none of the `colspec` elements contains the `colwidth` attribute, all the columns have the same size, and the table width is fixed to the maximum available size. Several examples of these tables are given.

Column 1
left aligned
no specified width, so it takes all the page

Column 1	Column 2
left aligned	centered cell
no specified width	idem

Column 1	Column 2	Column 3	Column 4	Column 5
left aligned	left aligned	right aligned	centered cell	centered
no specified width	idem	idem	idem	idem

4.8.4 Tables with mixed colspec

A table can have `colspec` elements containing `colwidth` attribute mixed with `colspec` elements without `colwidth`. Here is an XML source example:

```
<informaltable>
  <tgroup cols="5" colsep="1" rowsep="1" align="left">
    <colspec colname="c1"/>
    <colspec align="left" colwidth="4cm"/>
    <colspec align="right" colwidth="5cm"/>
    <colspec align="center"/>
    <colspec align="center" colwidth="3cm"/>
    <tbody>
      ...
    </tbody>
  </tgroup>
</informaltable>
```

It is rendered like this:

Column 1	Column 2	Column 3	Column 4	Column 5
left aligned (tgroup order)	left aligned	right aligned	centered cell	in the centre
no specified width	4 cm column width	5 cm column width	no width	3 cm column width

4.8.5 Tables with proportional and fixed colwidth

Proportional column widths are supported. Here is an example:

```
<informaltable>
  <tgroup cols="5" colsep="1" rowsep="1" align="left">
    <colspec colname="c1" colwidth="*"/>
    <colspec align="left" colwidth="2*"/>
    <colspec align="right" colwidth="3*"/>
    <colspec align="center"/>
    <colspec align="center" colwidth="3cm"/>
  </tgroup>
</informaltable>
```

```

<tbody>
...
</tbody>
</tgroup>
</informaltable>

```

It gives this table:

Column 1	Column 2	Column 3	Column 4	Column 5
left aligned (tgroup level)	left aligned	right aligned	centered cell	in the centre
proportional column (*)	proportional column (2*)	proportional column (3*)	no specified width	3 cm column width

4.8.6 Tables with fixed colwidths

All the columns can have fixed size, like this:

```

<informaltable>
  <tgroup cols="4" colsep="1" rowsep="1" align="left">
    <colspec colname="c1" colwidth="2cm"/>
    <colspec align="left" colwidth="2.5cm"/>
    <colspec align="right" colwidth="5cm"/>
    <colspec align="center" colwidth="3cm"/>
    <tbody>
      ...
    </tbody>
  </tgroup>
</informaltable>

```

It gives the following table:

Column 1	Column 2	Column 3	Column 4
left aligned (tgroup level)	left aligned	right aligned	centered cell
2 cm column width	2,5 cm column width	5 cm column width	4 cm column width

4.8.7 Automatic Column Width

4.8.7.1 Global Setting

In the previous sections the columns widths are computed from a proportional basis, when no colwidth is specified or when the colwidths contain a star ("*"). Of course, a colwidth containing a fixed width incidently sets the column width with this size.

It is possible to change this sizing policy of not-fixed-width columns by playing with the `newtbl.autowidth` parameter. The parameter can take the following values:

default

The automatic width (that is, latex is in charge to size the column width) is applied only to columns not having a specified colspec colwidth. It includes both undefined colspec, and colspec without the colwidth attribute.

all

the automatic width is applied to any column, whether a colspec is provided or not.

By default the parameter is unset, and no automatic width is applied. Using automatic width is handy in some situations but there is no more control if the tables fit in the page or not, since in this case the column is as wide as its content, with no more paragraph breaking. The parameter is global for all the tables in the document.

You can also perform the same thing by setting the `default.table.width` parameter to `autowidth.default` or `autowidth.all` instead of using `newtbl.autowidth`.

4.8.7.2 Local Setting

If you want to apply an automatic width only to some specific tables you can put the Processing Instruction `<?dblatex table-width="autowidth.scope" ?>` in the related tables. The PI has precedence over the `newtbl.autowidth` parameter.

The `scope` can take the same values and have the same effects than for the `newtbl.autowidth` parameter (default or all).

In addition to these keywords, `table.width` can also take for its value keywords of the form `autowidth.column: N` ..., where N is the number of the column (counting from 1) which is to have its width automatically sized.

See the following example:

```
<informaltable><?dblatex table-width="autowidth.column: 1 3"?>
  <tgroup cols="4" colsep="1" rowsep="1" align="left">
    <colspec colname="c1"/>                                <!-- automatic width is applied to column #1 -->
    <colspec align="left" colwidth="*/>
    <colspec align="right" colwidth="5cm"/> <!-- automatic width is applied to column #3 -->
    <colspec align="center" colwidth="3cm"/>
    ...
  </informaltable>
```

Here are two other examples with the all and default scopes:

The following table has columns 1 and 2 sized by latex with `autowidth.all`:

Column 1	Column 2	Column 3	Column 4
cell on 4 lines	simple cell	cell on 2 lines	cell without morerow attribute
	cell in column 2		cell on 2 lines
	left aligned on 2 lines	cell in line 3, column 3 4 cm column width	
			last cell in column 4

It is written as follow:

```
<informaltable><?dblatex table-width="autowidth.all"?>
  <tgroup cols="4" colsep="1" rowsep="1" align="left">
    <colspec colname="c1"/> <!-- 'default' and 'all' apply on this column -->
    <colspec align="left" colwidth="*/> <!-- only 'all' applies on this column -->
    <colspec align="right" colwidth="5cm"/>
    <colspec align="center" colwidth="3cm"/>
    ...
  </informaltable>
```

The following table has only columns 1 sized by latex with `autowidth.default`:

Column 1	Column 2	Column 3	Column 4
cell on 4 lines	simple cell	cell on 2 lines	cell without morerow attribute
	cell in column 2		cell on 2 lines
	left aligned on 2 lines	cell in line 3, column 3 4 cm column width	
			last cell in column 4

The only difference is that the PI attribute value is `autowidth.default`:

```
<informaltable><?dblatex table-width="autowidth.default"?>
  <tgroup cols="4" colsep="1" rowsep="1" align="left">
    <colspec colname="c1"/> <!-- 'default' and 'all' apply on this column -->
    <colspec align="left" colwidth="*/> <!-- only 'all' applies on this column -->
    <colspec align="right" colwidth="5cm"/>
    <colspec align="center" colwidth="3cm"/>
    ...
  </tgroup>
</informaltable>
```

4.8.8 Informal Table LaTeX Styles

By default **dblatex** translates an informal table to the LaTeX environment specified by the `table.default.tabstyle` parameter (usually `longtable`), but you can specify which LaTeX tabular environment to use, globally through the parameter, or per table through the `tabstyle` attribute. Of course the `tabstyle` attribute value is specific to `dblatex`. The supported values are:

longtable

The default table type used by `dblatex` when `table.default.tabstyle` is empty, in order to be able to split over several pages.

tabular

The most usual table type. Such table can only be on a single page.

tabularx

An advanced table type that allows to stretch column widths to the available remained page width. Such table can only be on a single page like for "tabular".

The two following examples show how `tabular` and `tabularx` rendering differ when the automatic width feature is used. Here is the XML source code:

```
<informaltable tabstyle="tabular">
  <?dblatex table-width="autowidth.column: 1 3"?>
  <tgroup cols="5" colsep="1" rowsep="1" align="left">
    <colspec colname="c1"/>
    <colspec align="left"/>
    <colspec align="right"/>
    <colspec align="center"/>
    ...
  </tgroup>
</informaltable>

<informaltable tabstyle="tabularx">
  <?dblatex table-width="autowidth.column: 1 3"?>
  <tgroup cols="5" colsep="1" rowsep="1" align="left">
    <colspec colname="c1"/>
    <colspec align="left"/>
    <colspec align="right"/>
    <colspec align="center"/>
    ...
  </tgroup>
</informaltable>
```

It is rendered as follow:

C1	Column 2	Column 3	Column 4	Column 5
A	left aligned	right aligned	centered cell	centered
B	idem	idem	idem	idem

C1	Column 2	Column 3	Column 4	Column 5
A	left aligned	right aligned	centered cell	centered
B	idem	idem	idem	idem

4.8.9 Tables with morerows

A table can contain entries that cover several lines. The following XML source contains an entry covering 4 lines:

```
<informaltable>
  <tgroup cols="4" colsep="1" rowsep="1" align="left">
    <colspec colname="c1" colwidth="*"/>
    ...
    <tbody>
      <entry morerows="3">it covers 4 lines</entry>
      ...
    </tbody>
  </tgroup>
</informaltable>
```

Here is an example of table containing several entries with morerows attribute:

Column 1	Column 2	Column 3	Column 4
cell on 4 lines	simple cell	cell on 2 lines	cell without morerow attribute
	cell in column 2		cell on 2 lines
	left aligned on 2 lines	cell in line 3, column 3	
		4 cm column width	last cell in column 4

4.8.10 Landscape tables

A table can be displayed in a lanscape format by using the orient attribute. Here is an XML source example:

```
<informaltable orient="land">
  <tgroup cols="5" colsep="1" rowsep="1" align="left">
    <colspec colname="c1" colwidth="*"/>
    ...
    <tbody>
      ...
    </tbody>
  </tgroup>
</informaltable>
```

Here is how it is displayed.

Column 1	Column 2	Column 3	Column 4	Column 5
left aligned	left aligned	right aligned	centered cell	centered
no specified width	idem	idem	idem	idem

4.8.11 Smaller tables

For big tables it can be usefull to have smaller text, so that the table is not too large or too long and it can be displayed within a page. It is possible to specify smaller table text by using the `role` attribute of the elements `table` or `informaltable`.

The values and the “role” dedicated to this attribute are specific to `dblatex`, but it is compliant with the DocBook specification because in general the `role` attribute purpose is never defined.

The available text size definitions supported by `role` are directly taken from LaTeX:

- `small`,
- `footnotesize`,
- `scriptsize`,
- `tiny`.

Here are examples for each size.

Column 1	Column 2	Column 3	Column 4	Column 5
left aligned	left aligned	right aligned	centered cell	centered
no specified width	idem	idem	idem	idem

Column 1	Column 2	Column 3	Column 4	Column 5
left aligned	left aligned	right aligned	centered cell	centered
no specified width	idem	idem	idem	idem

Column 1	Column 2	Column 3	Column 4	Column 5
left aligned	left aligned	right aligned	centered cell	centered
no specified width	idem	idem	idem	idem

Column 1	Column 2	Column 3	Column 4	Column 5
left aligned	left aligned	right aligned	centered cell	centered
no specified width	idem	idem	idem	idem

4.8.12 Coloured tables

You can color all the table by setting its `bicolor` attribute.

You can also color only some cells by using the Processing Instruction `<?dblatex bgcolor="color">`. The PI can apply to columns when put in a `colspec`, to rows when put at the beginning of a row, or to cells when put in a `entry`.

The entry colour has precedence over the row colour, that has precedence over the column colour, that has precedence over the table colour.

The color can be expressed in hexadecimal notation like for HTML (e.g. `#C0C0C0`) or in a syntax understood by the `colortbl` latex package.

Here is an example.

Column 1	Column 2	Column 3	Column 4
yellow	green column	yellow	yellow
blue row	red cell	blue row	blue row
yellow	green column	yellow	gray

This table is coded like this:

```
<informaltable id="tbl-color" bgcolor="{yellow}">
<tgroup cols="4" colsep="1" rowsep="1" align="left">
```

```

<colspec colname="c1" colwidth="2cm"/>
<colspec align="left" colwidth="2.5cm"><?dblatex bgcolor="#00FF00"?></colspec>
<colspec align="right" colwidth="5cm"/>
<colspec align="center" colwidth="3cm"/>
<thead>
  <row>
    <entry>Column 1</entry><entry>Column 2</entry>
    <entry>Column 3</entry><entry>Column 4</entry>
  </row>
</thead>
<tbody>
<row>
  <entry>yellow</entry><entry>green column</entry>
  <entry>yellow</entry><entry>yellow</entry>
</row>
<row>
<?dblatex bgcolor="{blue}"?>
  <entry>blue row</entry>
  <entry><?dblatex bgcolor="{red}"?>red cell</entry>
  <entry>blue row</entry><entry>blue row</entry>
</row>
<row>
  <entry>yellow</entry><entry>green column</entry>
  <entry>yellow</entry>
  <entry><?dblatex bgcolor="[gray]{0.8}"?>gray</entry>
</row>
</tbody>
</tgroup>
</informaltable>

```

4.8.13 HTML Tables

Since version 0.3.2 dblatex supports HTML tables. Some features are handled differently from CALS tables as illustrated by the following HTML table source example:

```

<table border="1" width="100%" rules="all"❶>
  <caption>An HTML Table</caption>❷
  <colgroup span="2" valign="top" align="right"/>
    <?dblatex bgcolor="red"?>❸
  </colgroup>
  <colgroup valign="bottom" align="left" width="5%"❹>
    <col align="right" span="2"/>
    <col valign="top"/>
  </colgroup>
  <colgroup bgcolor="yellow" width="0*"❺><?dblatex bgcolor="yellow"?></colgroup>
  <colgroup valign="bottom" align="left" width="15%"/>
  <colgroup valign="bottom" align="left" width="10%"/>
  <thead>
    <tr> <td width="10%">Head A1</td><td>Head B1</td><td>Head C1</td> </tr>
  </thead>
  <tfoot>
    <tr> <td width="10%">Foot A1</td><td>Foot B1</td><td>Foot C1</td> </tr>
  </tfoot>

  <tr>
    <td width="10%">A1</td><td>B1</td><td>C1</td>
  </tr>
  <tr>
    <td>A2</td><td bgcolor="yellow">B2</td><td colspan="3" rowspan="3">C-E2</td><td>F2</td>
  </tr>

```

```

<tr>
  <td>A3</td><td width="25.3"ⓐ>B3</td><td>F3</td><td rowspan="2">G3</td><td rowspan="3"> ←
    H3</td>
</tr>
<tr>
  <td>A4</td><td>B4</td><td>F4</td><td>G4</td>
</tr>
<tr>
  <td>A5</td><td>B5</td><td width="5%">C5</td><td>D5</td>
</tr>
<tr bgcolor="blue"ⓑ>
  <td>A6</td><td>B6</td><td>C6</td><td>D6</td><td>E6</td><td bgcolor="green"ⓓ>F6</td>
</tr>
</table>

```

- ❶ The cell borders are specified through the `table rules` attribute, and not by any `rowsep` and `colsep` attributes. Therefore it is not possible to set the borders of an individual cell. If no `rules` attribute is provided the default setup defined by `default.table.rules` parameter is used.
- ❷ The title of a formal HTML table is set through the `caption` element, because the `title` element is not available.
- ❸ The cell background colors can be set directly with the `bgcolor` attribute in `row` (for an entire row color setting), `th`, or `td`, but the `dblatex bgcolor` Processing Instruction is required in `colgroup` or `col` to set a column color.
- ❹ The relative widths can be expressed as a percentage. The proportional syntax (e.g. "3*") is still available in `colgroups`.
- ❺ To specify that a column must be as wide as its content, there is no need to use the `dblatex autowidth` Processing Instruction. You just need to set the related `colgroup width` attribute to "0*".
- ❻ The fixed widths must be expressed as numbers, and the implicit unit is the point ("pt"). You cannot set a width expressed in a unit like inches ("in"), centimeters ("cm") and so on.
- ❼ The closest enclosing element attribute has precedence over ancestor attributes. This applies to background color too.

This source example is rendered as follow:

Head A1	Head B1	Head C1						
A1	B1	C1						
A2	B2				F2			
A3	B3				F3			
A4	B4	C-E2			F4	G3		G4
A5	B5	C5	D5				H3	
A6	B6	C6	D6	E6	F6			
Foot A1	Foot B1	Foot C1						

Table 4.1: An HTML Table

4.9 Correcting Poor Formatting

LaTeX does not invariably produce good looking output. To remedy this it can sometimes be useful to guide LaTeX in the formatting of individual characters, words, sentences, and pages. It is best to adjust with caution, making adjustments only to fix specific problems and limiting adjustments to the affected area.

Tip

Adjustments to formatting are best done late in the production process so that subsequent changes to content do not produce new problems.

This section provides an overview of common problems and summarizes methods which may be used to resolve issues related to formatting. Much more detail can be found in the Chapter 5 chapter and the Appendix A and Appendix B appendixes. Be sure to look through these, especially the appendixes, to see whether there is an adjustment which will solve your particular problem. Of course the DocBook documentation should also be consulted to see if a DocBook attribute can be used to adjust document presentation.

4.9.1 Floats

Floats are figures, tables, examples, images, and so forth which do not necessarily appear in the output document in the location in which they appear in the DocBook source. The placement of floats can result in visually poor or otherwise undesirable output.

One direct approach is to use the “informal” versions of the affected DocBook elements. E.g. use `informaltable` instead of `table`. The “informal” elements do not float. Bear in mind though that “informal” elements do not appear in the table of contents.

The alternative to eliminating the float entirely is to make an adjustment by parameter, attributes, or processing instruction. For figure, example, and equation, you can play with these methods:

- In a specific element, use the `floatstyle` attribute to specify the latex float placement policy. For example ' [H] ' tries to place the element where it is declared.
- Use the parameter `element.default.position` to specify globally the placement policy for this element.

Look at page 79 for more details about the meaning of the placement choice letters.

4.9.2 Tables

Placement problems with tables relate to their classification as a float. For resolution regarding these issues see the section on floats above.

Tables exceeding the length of a page, or tables that otherwise need to cross page boundaries, have special requirements. They must not be floats and must be rendered using the LaTeX `longtable` environment. Among the available controls are: use of a `informaltable` element in place of a `table` element, the `table` element’s `tabstyle` attribute, the `table.default.tabstyle` parameter, the `table.in.float` parameter.

It is worth mentioning that good typesetting design practice for tables generally calls for eliminating all vertical rules (the lines between columns) and most horizontal rules.

Better vertical placement of table horizontal rules can be obtained by providing values for the `newtbl.format.thead` and `newtbl.format.tbody` parameters. Depending on the font in use, a value like `\rule{0pt}{2.6ex}\rule[-0.9ex]{0pt}{0}` can be used for both parameters. This inserts a *strut*, a 0 width character, in each row of the table which “pushes” the horizontal rules upwards and downwards to provide adequate vertical spacing.

4.9.3 Examples

Placement of examples has the same issue than tables, that is, it is an element that can contain many materials and need to split over several pages. In this case it cannot be considered as a float.

To avoid an example floating and to allow it cover several pages, set the parameter `example.float.type` to none.

4.9.4 Hyphenation and over-long lines

LaTeX is generally very good at hyphenation, but this applies only to actual words. Technical writing may include long character sequences that are not actual words. Hyphenation failure will typically result in lines of text that flow past the expected right-hand edge of a line. In LaTeX terminology this is known as a “overfull hbox”.

The `hyphenation.format` parameter can be helpful to flag some formats for more aggressive hyphenation.

Various adjustments can be made with the `<?latex?>` processing instruction to add raw latex directives and eliminate overfull hboxes but it often makes sense to address hyphenation problems directly. This can be done either on an ad-hoc basis, telling LaTeX how to hyphenate specific occurrences of words where a problem exists, or by telling LaTeX how to hyphenate words it does not know. The first method is accomplished by inserting “soft hyphens” into words in your DocBook source using the `latex` processing instruction. This is described in the section called “[Safe LaTeX Insertions](#)” section.

To educate LaTeX as to how to hyphenate your special vocabulary a custom LaTeX style is required. Setting this up is described in the [Section 5.5](#) section. The LaTeX command to use is `\hyphenation`. For example `\hyphenation{PostgreSQLtrans-mog-re-fi-ca-tion}` tells LaTeX that *PostgreSQL* should not be hyphenated and where to hyphenate *transmogrification*.

Finally, long URLs can cause over-long lines. Especially in footnotes. Using the LaTeX `breakurl` package is one way to solve this. This can be done with a custom latex stylesheet. See [Section 5.5](#).

4.9.5 Characters and Manual Spacing

The `<?latex?>` processing instruction may be used to limit character kerning, the joining of pairs of characters.

The following DocBook entity declarations can be useful to control spacing. These may be declared in your DocBook DTD within the trailing pair of square braces (`[]`) or elsewhere. Once declared the entities may be used in your document text. E.g. `Von Trapp` puts a non-breaking space between Von and Trapp, ensuring that these two words will not appear on separate lines.

```
<!ENTITY nbsp    " " >
<!ENTITY ensp    " " >
<!ENTITY emsp    " " >
```

4.10 Writing Mathematics

4.10.1 Writing LaTeX Mathematical Equations

4.10.1.1 Presentation

DocBook doesn’t define elements for writing mathematical equations. Only few elements exist that tell how equation should be displayed (inlined, block):

- `inlineequation` tells that the equation is inlined,
- `informalequation` tells that the equation is displayed as a block, without a title.
- `equation` tells that the equation is displayed as a block, with or without a title.

These tags include a `graphic` (`graphic` or `mediaobject`) or an alternative text equation, as shown by the example.

Example 4.7 Equation taken from TDG

```
<equation><title>Last Theorem of Fermat</title>
  <alt>x^n + y^n &ne; z^n &forall; n &ne; 2</alt>
  <graphic fileref="figures/fermat"></graphic>
</equation>
```

4.10.1.2 Implementation choice

The principle is to use only the `alt` element. If initially `alt` contains actually the text to print, it is chosen to use this element to embed LaTeX mathematical equations. This choice has the following advantages:

- The translation done by `dblatex` is really easy, since the equation is already written in LaTeX.
- LaTeX is one of the best word processor to render mathematical formulas.
- One doesn't need to write the equations in MathML.
- This method isn't specific to this tool (see the following section).

4.10.1.3 Mathematical Delimiters

The `dblatex` implementation is as light as possible. This is why it is up to the writer to properly use the mathematical delimiters (`$`, `\`, `\,`, `\,`, `\,`). By this way the writer fully controls how he writes equations.

By default **dblatex** checks that consistent mathematical delimiters or environment are used in `alt` and it inserts the default math mode delimiters if `dblatex` thinks they are missing, but you can ask `dblatex` to write directly the `alt` content without any action. To do this, use the `texmath` Processing Instruction with `delimiters` set to 'user'.

For example, Example 4.8 has user specific delimiters:

Example 4.8 Equation with user delimiters

$$S = \left\lfloor \frac{N \bmod 65536}{32768} \right\rfloor$$

$$E = \left\lfloor \frac{N \bmod 32768}{1024} \right\rfloor$$

$$M = N \bmod 1024.$$

EQUATION 4.1: Equation block with align tabs

In the XML source this equation uses a specific `align*` environment and the `texmath` PI to let the equation as is:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE para PUBLIC "-//OASIS//DTD DocBook XML V4.4//EN"
"http://www.oasis-open.org/docbook/xml/4.4/docbookx.dtd">
<equation>
  <title>Equation block with align tabs</title>
  <alt><?texmath delimiters="user"?><![CDATA[\begin{align*}
S &= \left\lfloor \frac{N \bmod 65536}{32768} \right\rfloor \\
E &= \left\lfloor \frac{N \bmod 32768}{1024} \right\rfloor \\
M &= N \bmod 1024. \\
\end{align*}]]></alt>
</equation>
```

4.10.1.4 Compatibility

This implementation is not contradictory nor specific. In particular, the **DBTeXMath** proposal to extend the DSSSL stylesheets used by jade follows the same approach, and is integrated in the Norman Walsh XSL stylesheets.

4.10.1.5 Examples

The following examples show how to write the equations.

Example 4.9 Inlined Equation

The formula $C = \alpha + \beta Y^\gamma + \varepsilon$ is inlined in the paragraph. Its XML source is:

```
<para>The formula
  <inlineequation id="eg-inlineequation">
    <alt>$C = \alpha + \beta Y^{\gamma} + \epsilon$</alt>
    <graphic fileref="figures/eq1"/>
  </inlineequation>
  is inlined in the paragraph. Its XML source is:</para>
```

Example 4.10 Equation in a block

The following formula:

$$C = \alpha + \beta Y^\gamma + \varepsilon$$

is displayed in a separate block. The XML source is:

```
<para>The following formula:
  <informalequation>
    <alt>\[C = \alpha + \beta Y^{\gamma} + \epsilon\]</alt>
    <graphic fileref="figures/eq1"/>
  </informalequation>
  is displayed in a separate block. The XML source is:</para>
```

Example 4.11 Equation in a float

The formula Equation 4.2 below:

$$C = \alpha + \beta Y^\gamma + \varepsilon$$

EQUATION 4.2: Simple Formula

is displayed in a block with a title. Its XML source is:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE para PUBLIC "-//OASIS//DTD DocBook XML V4.4//EN"
"http://www.oasis-open.org/docbook/xml/4.4/docbookx.dtd">
<para>The formula <xref linkend="eq-with-title"/> below:
  <equation id="eq-with-title">
    <title>Simple Formula</title>
    <alt>\[C = \alpha + \beta Y^{\gamma} + \epsilon\]</alt>
    <graphic fileref="figures/eq1"/>
  </equation>
  is displayed in a block with a title. Its XML source is:</para>
```

Example 4.12 Equation without a title

The formula 4.1 below:

$$C = \alpha + \beta Y^\gamma + \varepsilon \tag{4.1}$$

is displayed as a latex equation with its own equation numbering. Its XML source is:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE para PUBLIC "-//OASIS//DTD DocBook XML V4.4//EN"
"http://www.oasis-open.org/docbook/xml/4.4/docbookx.dtd">
<para>The formula <xref linkend="eq-with-no-title"/> below:
  <equation id="eq-with-no-title">
    <alt>C = \alpha + \beta Y^{\gamma} + \epsilon</alt>
    <graphic fileref="figures/eq4"/>
  </equation>
is displayed as a latex equation with its own equation numbering.
Its XML source is:</para>
```

4.10.2 Writing MathML equations

You can write MathML equations in a DocBook based document, by using the [MathML Module for DocBook XML](#) instead of the DocBook DTD.

dblatex now translates the MathML equations to latex by using the excellent stylesheets of the [XSLT MathML Library](#) by Vasil Yaroshevich. A large amount of tests from the [W3C MathML Test Suite 2.0](#) is supported (657 of 712 tests). The test file used to validate the MathML stylesheets is provided in the package.

4.11 Musical Notation

Dblatex does not itself present musical notation. Of course, images of musical scores can be incorporated into dblatex output as can any other sort of images. For a more comprehensive solution which better integrates the typesetting of text with musical notation the LilyPond project (<http://www.lilypond.org/>) provides a facility for including musical notation within DocBook documents. LilyPond provides a program, **lilypond-book**, that pre-processes DocBook documents containing LilyPond musical notation and which can automatically invoke **dblatex** to produce PDFs integrating musical notation with text.

Information regards **lilypond-book** is, at present, found in LilyPond's "Usage" manual, the "Running lilypond-book" chapter, the "Integrating music and text" section.

4.12 Extending the Verbatim Rendering

4.12.1 Dblatex Specific Options

There are few attributes or options specific to **dblatex** to render verbatim blocks:

- The `role` attribute of `screen`, `programlisting`, and `literallayout` can take the following special values:

wrap

The verbatim lines can break and wrap when they are longer than the available width. It is the default behaviour.

overflow

The verbatim lines never break and go into the margin when they are too long.

scale

The verbatim block is automatically scaled so that the longest line or specified column count fits in the available page width. See Section [4.12.2](#).

- The parameter `literal.role` can be used to set the default role to apply. By default the value is an empty string.
- The parameter `literal.class` can be used to set the default `literallayout` class when no class attribute is given. By default the value is `monospaced`.

4.12.2 Scaling Feature

The user can scale the verbatim block so that the longest line fits in the available page width, or so that the page contains at least a specified width expressed in columns.

The scaling feature is enabled when the parameter `literal.extensions` is set as follow:

scale

The scaling is performed only when the `role` attribute is set to "scale", or when the `role` attribute is not set and the parameter `literal.role` is set to "scale".

scale.by.width

The scaling is performed when the `role` attribute or `literal.role` is properly set, or when the attribute `width` is set. When `width` is set the block is scaled so that the specified width fits in the page width.

Here are some listing examples with several attribute combinations producing or not the scaling. In these examples the parameter `literal.extensions` is set to "scale.by.width".

```
<programlisting width="110">  
The listing is scaled and lines are wrapped after 110 characters. Check yourself:  
  
123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789  
          9 123456789  
0           1           2           3           4           5           6           7           8           9           10 ←  
          11  
  
</programlisting>
```

```
<programlisting width="110" role="overflow">
There is no scaling because the role has precedence over the width attribute:

123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789
0          1          2          3          4          5          6          7          8          9
11

</programlisting>
```

```
<programlisting role="scale">
The listing is scaled to display the longest line with no break:

123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789 1234567 9 123456789
0          1          2          3          4          5          6          7          8          9          10         11

</programlisting>
```

```
<programlisting width="110" role="scale">
The listing is scaled to fit up to 110 columns, and the lines are wrapped. In this case role is redundant ←
    because @width automatically produces scaling.

123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789 ←
    9 123456789
0      1      2      3      4      5      6      7      8      9      10 ←
    11

</programlisting>
```

4.12.3 Formatting embedded elements

The programlisting and screen environments are supported by dblatex, but the implementation is rather conservative, that is, most of the elements embedded in such environments are not rendered like in normal environment (e.g. bold, emphasis, etc.). Only the contained text is printed out. For the elements whose rendering is lost, **dblatex** prints out a warning message.

For example, let's compile the following programlisting fragment:

```
<programlisting>
zone <replaceable>zone_name</replaceable>
```

```
<optional><replaceable>class</replaceable></optional> {
    type delegation-only;
};

</programlisting>
```

dblatex warns that the `optional` and `replaceable` elements are not supported (i.e. not rendered) in the `programlisting`:

```
$ dblatex progfrag.xml
Build the book set list...
Build the listings...
XSLT stylesheets DocBook - LaTeX 2e (devel)
=====
Warning: the root element is not an article nor a book
Warning: programlisting wrapped with article
replaceable not supported in programlisting or screen
optional not supported in programlisting or screen
replaceable not supported in programlisting or screen
replaceable not supported in programlisting or screen
optional not supported in programlisting or screen
replaceable not supported in programlisting or screen
...
```

If you want those elements be formatted in bold, or italic you need to override the templates used in `latex.programlisting` mode, as follow:

```
<xsl:template match="replaceable|optional" mode="latex.programlisting">
  <xsl:param name="co-tagin" select="'&lt;:'" /> ❶
  <xsl:param name="rnode" select="/" /> ❷
  <xsl:param name="probe" select="0" /> ❸

  <xsl:call-template name="verbatim.boldseq"> ❹
    <xsl:with-param name="co-tagin" select="$co-tagin" />
    <xsl:with-param name="rnode" select="$rnode" />
    <xsl:with-param name="probe" select="$probe" />
  </xsl:call-template>
</xsl:template>
```

❶, ❷, ❸ These parameters are required in `latex.programlisting` mode.

❹ The predefined template makes bold the verbatim text of the element.

If formatting setup is not enough, you can also render these elements as if they were in a normal environment. To do this, you need to override the templates used in `latex.programlisting` mode, as follow:

```
<xsl:template match="replaceable|optional" mode="latex.programlisting">
  <xsl:param name="co-tagin" select="'&lt;:'" />
  <xsl:param name="rnode" select="/" />
  <xsl:param name="probe" select="0" />

  <xsl:call-template name="verbatim.embed"> ❶
    <xsl:with-param name="co-tagin" select="$co-tagin" />
    <xsl:with-param name="rnode" select="$rnode" />
    <xsl:with-param name="probe" select="$probe" />
  </xsl:call-template>
</xsl:template>
```

❶ To enable the normal mode rendering within a verbatim environment, call the `verbatim.embed` template, and pass the mandatory parameters.

4.12.4 Creating a new Verbatim Environment

dblatex heavily relies upon the listing latex package to display the screen, programlisting, and literallayout blocks.

The global listing setup can be overwritten with `literal.layout.options` but the user can also provide its own listing environment to use instead of the default environment, by using the following procedure:

1. Create the new listing environment in a customized latex style, like the following example. It is required that the environment name starts with the string "lst". If not, **dblatex** raises an error because it cannot recognize it as a special verbatim environment.

```
%%
%% This style is derivated from the db2latex one
%%
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{mystyle}[2012/02/03 My DocBook Style]

%% Just use the original package and pass the options
\RequirePackageWithOptions{db2latex}

%% New listing environment doing what I want
\lstnewenvironment{lstblock}[1][{}]{
  {\lstset{numbers=left,numberstyle=\tiny,float,#1}}
  {}
}
```

2. Specify to **dblatex** the listing environment name through the `literal.environment` parameter, either on the command line or with a user XSL stylesheet.

```
$ dblatex -s mystyle.sty -P literal.environment=lstblock file.xml
```

4.13 Creating an Index

An index is automatically generated if some index entries (`indexterm`), telling the terms to put in the index, are written in the document. The `keyword` elements are not printed but are also added to the index.

Example 4.13 Index Entry

```
<para>In this paragraph is described the function
<function>strcpy</function><indexterm><primary>strcpy</primary></indexterm>.
</para>
```

The index is put at the end of the document. It is not possible to put it somewhere else.

4.13.1 Internationalized Indexes

4.13.1.1 Indexing Tools

Makeindex is used by default to build the index. It works fine for latin charset but it is not suited for other charsets and UTF-8 encoding. Moreover its sorting rule is not suited for many languages.

Therefore **dblatex** gives the possibility to use **xindy** that allows internationalized indexing. When **xindy** is used, the sorting language used is deduced from the Docbook document lang. If the document lang has no correspondance for **xindy**, or if you want to force the use of a specific sorting language, you can specify the sorting language to use. With **Xindy** you can also provide your own sort rules, but see the official **Xindy** documentation for more details.

A parameter is provided to use **xindy** instead of **makeindex**:

- `latex.index.tool=xindy` tells dblatex to use xindy instead of makeindex (default).

A typical use is then:

```
dblatex -P latex.index.tool=xindy file.xml
```

4.13.1.2 Index Sorting

Makeindex has very few methods to change its index sorting rules, except the `-g` and `-L` options which are not very helpful in a DocBook context. On the contrary **xindy** can sort the index according to a specific language.

When xindy is used as index tool, **dblatex** passes to xindy through the xindy `-L` option the language name corresponding to the `lang` attribute of the document. If the document has no language or if **xindy** does not support the document language, the default sorting is applied. In this case you can use the following parameter:

- `latex.index.language` specifies the language used to sort the indexes. Currently this parameter is relevant only for xindy. The language set must be known by xindy (see the xindy documentation). When not set (or empty) the sorting language to apply is deduced from the document `lang`.

The parameter can be used like any other like this:

```
dblatex -P latex.index.language=german-din file.xml
```

But it can also be used through an XSL configuration file to provide new mapping rules between ISO lang codes and known xindy languages, or override the current mapping done by dblatex as shown in Example 4.14.

Example 4.14 XSL Index Language Setup

By default dblatex maps the `de` lang code to the language named `german-din`, and has no correspondance to map `cy` (Welsh) or `eu` (Basque). The following setup provides a mapping for these cases:

```
<xsl:param name="latex.index.tool">xindy</xsl:param>

<xsl:param name="latex.index.language">
  <xsl:variable name="lang">
    <xsl:call-template name="l10n.language">
      <xsl:with-param name="target" select="(/set|/book|/article) [1]"/>
      <xsl:with-param name="xref-context" select="true()"/>
    </xsl:call-template>
  </xsl:variable>

  <!-- Define Xindy lang to use in specific cases -->
  <xsl:choose>
    <xsl:when test="$lang='de'">german-duden</xsl:when>
    <xsl:when test="$lang='cy'">english</xsl:when>
    <xsl:when test="$lang='eu'">french</xsl:when>
  </xsl:choose>
</xsl:param>
```

4.14 Writing a Bibliography

A bibliography (bibliography) can be written and put anywhere in the document. It appears as a chapter or a section and is composed by several divisions (bibliodiv) displayed as sections or subsections.

4.14.1 Using Bibliography Entries

The writer selects information that describes each bibliography entry (`biblioentry`), and chooses the presentation order. The titles and authors are displayed first.

Example 4.15 A Bibliography

```
<bibliography><title>Bibliography Example</title>
  <bibliodiv><title>References</title>
    <biblioentry>
      <title>Document title</title>
      <author><firstname>J.</firstname><surname>Doe</surname></author>
      <pubsnumber>DEX000567325</pubsnumber>
    </biblioentry>
  </bibliodiv>
  <bibliodiv><title>White papers</title>
    <biblioentry>
      <title>Technical notes</title>
      <authorgroup>
        <author><firstname>J.</firstname><surname>Doe</surname></author>
        <author><firstname>R.</firstname><surname>Marion</surname></author>
      </authorgroup>
      <pubsnumber>DEX000704520</pubsnumber>
    </biblioentry>
  </bibliodiv>
</bibliography>
```

4.14.2 Using BibTeX Databases

Instead of writing the bibliographic materials in DocBook you can reuse some already available BibTeX databases. Of course, this feature is specific to **dblatex**, that will automatically call **bibtex** if some bibtex databases are used.

To do so, write a `bibliodiv` containing an empty `bibliomixed` element having a `bibtex` processing instruction specifying the databases to use and the style to apply.

More precisely here are the attributes supported by the `bibtex` PI:

bibfiles

This attribute is mandatory and specifies the databases to use. The databases are separated by commas, and must not contain the file suffix (`.bib`). The `bibfiles` paths must be absolute or relative to the base directory of the document. You can also add some `bibfile` paths by using the `-L` option.

bibstyle

Optional attribute specifying the bibliographic style to apply for rendering the databases. You can also change globally the style to apply with the [latex.biblio.style](#).

The actual style file used by **bibtex** is searched in the default paths, but some extra paths can be added by using the `-l` option.

mode

Optional print mode. The available values are:

all

Print all the entries contained in the databases.

cited

Print only the entries cited in the document.

notcited

Print only the entries *not* cited in the document.

When the attribute is not used, the `latex.biblio.output` parameter is used as print mode. By default the print mode is set to 'all'.

Some `bibliodivs` embedding bibliographic entries can be mixed with some `bibliodivs` using BibTeX databases, as shown by Example 4.16.

Example 4.16 Bibliography using BibTeX databases

```
<bibliography><title>Bibliography Example</title>
  <bibliodiv><title>References</title>
    <biblioentry>
      <title>Document Title</title>
      <author><firstname>J.</firstname><surname>Doe</surname></author>
      <pubsnumber>DEX000567325</pubsnumber>
    </biblioentry>
  </bibliodiv>
  <bibliodiv><title>Bibtex References</title>
    <bibliomixed><?bibtex bibfiles="bib/latex-bib" bibstyle="alpha"?></bibliomixed>
  </bibliodiv>
  <bibliodiv><title>Cited Bibtex References</title>
    <bibliomixed><?bibtex bibfiles="bib/database1,bib/database2"
      bibstyle="plain"
      mode="cited"?></bibliomixed>
  </bibliodiv>
</bibliography>
```

4.14.3 Natbib Citations

You can apply natbib citation styles by playing with the citation role attribute, or with a `dblatex` processing instruction. The natbib use is enabled only when the `citation.natbib.use` parameter is set to 1; if not (default) the role attribute or PI are not taken into account even if present. The natbib package can be loaded with user specific options by setting the `citation.natbib.options` parameter.

When using the role attribute, simply type the natbib citation command to apply. When using the `dblatex` PI, put the natbib command in the `citestyle` attribute.

If you need to put some square brackets "[]" in the citation texts, enclose the whole text with "{ }" to protect them (as you would do in latex).

Here are some examples:

```
<para>
<citation role="\citep[see][chap. #2]">texbook</citation>
<citation role="\citep[see][[chap. #2]]">texbook</citation>
<citation><?dblatex citestyle="\citep[see][chap. #2]">texbook</citation>
<citation>texbook</citation>
</para>
```

You can use a global natib citation style with the `citation.default.style` parameter. By default the parameter is empty, and therefor is not used.

4.15 Document Revisions

The attribute `revisionflag` is usefull to identify the changes between two revisions of a document. This information is managed by `dblatex`, that adds revision bars in the margin of the paragraphs changed, such like in this paragraph.

Adding the revision flags can be manual, but its is tedious and error prone. The `diffmk` tool by Norman Walsh can do the work for you. See <http://www.sagehill.net/docbookxsl/Changebars.html> for more details about how to use it.

Note

With old changebar packages the revision bars only appear when using the "dvips" driver. This limitation has been fixed with changebar greater or equal to v3.5c.

4.16 Locale Support

4.16.1 Document Encoding

By default the latex document produced by **dblatex** is encoded in latin1, that fits well for roman-characters. This said, a real international support involves some kind of Unicode (UTF8) support.

In dblatex, the Unicode support is done by two methods that can be selected by some parameters:

- `latex.unicode.use=1` asks for including the unicode package (initially provided by Passivetex) in order to handle many of the unicode characters in a latin1 encoded document.
- `latex.encoding=utf8` produces a document encoded in UTF8, that is compiled in UTF8. It requires to have the `ucs` package installed.

In some languages like Chinese, Japanese or Korean, the latex document must be in UTF8. Therefore, the UTF8 encoding is forced for these languages whatever the parameter values are.

4.16.2 Babel Languages

Dblatex should be able to handle most of the languages supported by the babel package. Just set the `lang=lang` attribute in the root document element and dblatex will load the appropriate babel language.

4.16.3 CJK Languages

Dblatex can handle the CJK languages thanks to the CJK package. The CJK package must be installed to have this support available.

As said in Section 4.16.1 the latex file is encoded in UTF8. Moreover, the Cyberbit fonts are then used.

The install of the CJK package and Cyberbit fonts are well described at: <http://kile.sourceforge.net/Documentation/html/cjk.html>.

4.16.3.1 Korean Support

Dblatex does not use the HLatex package to drive Korean documents. It does not use the **hmakeindex** nor the **hbibtex** tool. Currently, Korean is handled like Chinese and Japanese with the CJK package.

4.16.4 Mixing the languages

Dblatex cannot handle correctly a document containing several elements with different `lang` values. In particular, if the main document lang is not one of the CJK language, a portion of text written in CJK will not be handled correctly and it can result in a compilation crash.

Even if the langs mixed do not end to a compilation failure, only the main document lang will be taken into account.

4.17 Using XRefsyle and Olinks

Since version 0.2.7 you can use the `xrefstyle` attribute like you would do with the DocBook Project stylesheets for HTML output.

Furthermore, you can also use `olinks`. Note that Olinking is used in the PDF version of this manual, in Section 2.4.

Actually, the common DocBook Project stylesheets version 1.72 are now used by `dblatex` to handle all of these features.

These features are fully described in the [DocBook XSL: The Complete Guide](#) book by Bob Stayton. In particular, the following sections cover these topics:

- <http://www.sagehill.net/docbookxsl/CustomXrefs.html> explains how to use `xrefstyle`.
- <http://www.sagehill.net/docbookxsl/Olinking.html> explains how to use `olinks`.

4.17.1 Specific xrefstyle for ulink

Dblatex defines a template that can be applied through the `xrefstyle` attribute to format an `ulink`. The template has the following general form: `url[.{show|hide}][.{after|infoot}]`

The principle is that the `show` and `hide` parts override locally for this `ulink` the `ulink.show` parameter, and the `after` and `infoot` parts override locally the `ulink.footnotes` parameter.

When only one part is defined, only one setup is overridden, and the other setup depends on the corresponding global parameter.

See [ulink.show](#) to have several examples of use of the `ulink xrefstyle` template.

4.18 Footnotes

4.18.1 Footnote Numbering Scope

The footnotes are well handled by latex. Dblatex just takes care about special cases like footnotes in headings or in tables. Therefore the default latex behaviour is applied for footnote numbering.

By default the footnote numbers are reset for each new chapter. If you want to change the numbering scope, you can use the latex package `chngcntr` either in a customized latex style (see Chapter 5) or in the parameter `latex.begindocument`. The following example makes the footnote numbers document wide.

```
<xsl:param name="latex.begindocument">
  <xsl:text>
% Want document wide footnotes
\usepackage{chngcntr}
\counterwithout{footnote}{chapter}
\begin{document}
  </xsl:text>
</xsl:param>
```

4.18.2 End Notes

You can configure **dblatex** to manage the footnotes as endnotes. In this case the notes are expected to be displayed in a dedicated section. The following sections tell how to do this.

4.18.2.1 Writing a document with endnotes

To have footnotes behaving as endnotes, you just need to set the `footnote.as.endnote` to 1, and to put in your document an empty `index` element with the attribute `type` set to 'endnotes'. This index is supposed to be included in a section dedicated to contain the notes, and therefore it does not display any heading by itself. The section containing the endnotes index can also contain any other materials.

Example 4.17 Writing a document with endnotes

The following document contains some footnotes and a section for the End Notes:

```
<book>
  <chapter><title>A Chapter</title>
  <para>Bla bla<footnote id="fn1">First note</footnote> bla
  bla<footnote id="fn2">Second note</footnote> ...</para>
</chapter>

  <chapter><title>Another Chapter</title>
  <para>Bla bla<footnote id="fn3">Third note</footnote> bla
  bla<footnote id="fn4">Fourth note</footnote> ...</para>
</chapter>

  <chapter><title>End Notes</title>
  <para>This chapter contains all the notes of the book.</para>

  <index type="endnotes"/>
</chapter>
</book>
```

dblatex is called with:

```
$ dblatex -P footnote.as.endnote=1 file.xml
```

4.18.2.2 Setup Properties

Currently the endnotes are handled by the latex package `endnotes`. You can customize its use with the `endnotes.properties` attribute set. This attribute set is filled by default with the values given by the internal attribute set `endnotes.properties.default`, and you can override the following attributes:

package

Recall the latex package in charge to handle the endnotes. Attribute provided for provision since currently there is no real alternative with well established packages.

heading

Macros to render the heading of the endnotes to display.

font-size

Macro specifying the font size to apply for each endnote. The package default is `\footnotesize`. The `dblatex` default is `\normalsize`.

note-format

Macros specifying how the render a end note. The package default is the same formatting that footnotes. The `dblatex` default is to print out endnotes like a numbered list.

The default setup:

```
<xsl:attribute-set name="endnotes.properties.default">
  <xsl:attribute name="package">endnotes</xsl:attribute>

  <!-- No header: endnotes are embedded in another section -->
```

```

<xsl:attribute name="heading">\mbox{}\par</xsl:attribute>

<!-- Show end notes as a numbered list -->
<xsl:attribute name="font-size">\normalsize</xsl:attribute>
<xsl:attribute name="note-format">%
\leftskip=1.8em\noindent
\makebox[0pt][r]{\theenmark.\rule{0pt}{\baselineskip}}\ignorespaces
</xsl:attribute>
</xsl:attribute-set>

<xsl:attribute-set name="endnotes.properties"
use-attribute-sets="endnotes.properties.default"/>

```

4.18.2.3 Endnotes Setup from scratch

If you want to fully control the endnotes setup, add some other macros, you can directly override the template `endnotes.setup`, as shown by the example below. If the endnote setup is already defined in your latex style you can override the template to make it empty in order to avoid some conflict between the latex style definition and the default one done by dblatex.

```

<!-- Use the default endnotes package setup and change the heading -->
<xsl:template name="endnotes.setup">
  <xsl:text>\usepackage{endnotes}&#10;</xsl:text>
  <xsl:text>\def\noteheading{\paragraph*{List Of Notes}%
\mbox{}\par\vskip-\baselineskip}&#10;</xsl:text>
</xsl:template>

```

4.18.2.4 Grouping Endnotes

By default the endnotes are displayed for the whole document as a global list. You can configure dblatex to show the endnotes grouped per part or per chapter. A heading is put before each group of notes to recall the component (part, chapter, or another section) that contains these notes. The purpose is to make the research of the notes at the end of the document easier for the reader.

To group the endnotes according to the granularity you wish, you just need to set the `endnotes.heading.groups` with the expected section types separated by a space. For example, the following setup displays notes per part and per chapter:

```

<xsl:param name="footnote.as.endnote" select="1"/>
<xsl:param name="endnotes.heading.groups" select="'part chapter'"/>

```

4.18.2.4.1 Adding new Groups

By default dblatex can group the notes per chapter and/or per part, provided that the section type is declared in `endnotes.heading.groups` as explained previously. If you want to add another section type you need to override the template called in mode `endnotes`, and to add this type to the declared groups. Usually the endnotes mode template only calls the `endnotes.add.header` template. The example below makes a new group for preface.

```

<xsl:param name="endnotes.heading.groups" select="'part chapter preface'"/>

<xsl:template match="preface" mode="endnotes">
  <xsl:call-template name="endnotes.add.header">
    <xsl:with-param name="reset-counter" select="1"/>
  </xsl:call-template>
</xsl:template>

```

4.18.2.4.2 Formatting the Headings text

You can use the parameter `endnotes.heading.style` to format the headings of the groups with the same syntax than `xrefstyle` does. The example below gives a template where the title and the page of the chapter containing the notes are displayed.

```
<xsl:param name="endnotes.heading.style">
  <xsl:text>template:Notes of the chapter "%t", page %p</xsl:text>
</xsl:param>
```

4.18.2.4.3 LaTeX Command to make the Headings

By default **dblatex** creates the headings of the groups with a section command whose level is just below the level of the endnotes section. For example, if the endnotes section is a chapter, the headings are created by using a `\section*` latex command. If the endnotes section is a section, the headings are done by using `\subsection*`, and so on.

If this behaviour does not fit your needs you can specify the command to use with the parameter `endnotes.heading.command`. It can be usefull if you want to format these heading in a specific way. The heading markup produced following the `endnotes.heading` template is passed as an argument of the specified command. The example below uses the command `\enoteheader*` to create the headings. It assumes that this macro is defined somewhere else (e.g. in a user latex style).

```
<xsl:param name="endnotes.heading.command" select="'\enoteheader*'"/>
```

4.18.2.4.4 Endnotes Counter Scope

By default the endnotes counter restarts from one in each chapter or part when they are grouped. You can change this behaviour by overriding the `endnotes.counter.resetby` parameter that lists the section types where the counter is reset. If you want a global counter running for the whole document, just set the parameter empty.

Chapter 5

Customization

The transformation process (and thus the output rendering) can be heavily customized by:

- using some [configuration parameters](#) either in a [configuration stylesheet](#) or directly from the [command line](#),
- using [Processing Instructions](#) to create some instructions very specific to dblatex,
- using some [customized stylesheets](#),
- using a [customized LaTeX style package](#).
- using a [LaTeX post process script or plugin](#).

All these customization methods can be used independently and in exceptional cases, but it can also be combined and registered in a master configuration file, called a specification file (cf. Section 5.7) to create a new tool dedicated to your needs.

5.1 Using XSL Parameters

The PDF rendering can be customised by using some XSL configuration parameters. Appendix A contains the reference documentation of the available user-configurable parameters.

5.2 Setting Command line Parameters

You can set some XSL parameters directly from the command line without creating a configuration parameter stylesheet, with the `-P parameter=value` option.

The following example set the `latex.hyperparam` parameter value:

```
dblatex -P latex.hyperparam=colorlinks,linkcolor=blue myfile.xml
```

5.3 Using Processing Instructions

Dblatex has Processing Instructions (PI) which can modify the default document formatting. Usually these instructions alter the output formatting of specific DocBook elements, like table cells.

To alter formatting globally it is better to set an appropriate [stylesheet parameter](#).

Processing instructions are written `<?name content ?>`. Most often *content* will have the form of one or more XML attributes with a value, as follows: `<?name attribute="value" ?>`.

The list of the available Processing Instructions are given in Appendix B.

Here is an example of a PI used in a table:

Example 5.1 Table width specified with a Processing Instruction

```
<informaltable><?dblatex table-width="autowidth.column: 1 3"?>
  <tgroup cols="4" colsep="1" rowsep="1" align="left">
    <colspec colname="c1"/>          <!-- automatic width is applied to column #1 -->
    <colspec align="left" colwidth="*/>
    <colspec align="right" colwidth="5cm"/> <!-- automatic width is applied to column #3 -->
    <colspec align="center" colwidth="3cm"/>
    ...
  </tgroup>
</informaltable>
```

5.4 XSL User Stylesheet

You can provide your own XSL stylesheet to set some of the XSL parameters, and/or to override some of the dblatex XSL templates. The user stylesheet is specified by using the option `-p custom.xml`.

5.4.1 Changing the XSL parameter values

The parameters can be stored in a user defined XSL stylesheet. An example of configuration stylesheet is given with this manual:

```
<?xml version='1.0' encoding="iso-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version='1.0'>

<!-- Target Database set by the command line -->

<xsl:param name="target.database.document">olinkdb.xml</xsl:param>
-->

<!-- Use the Bob Stayton's Tip related to olinking -->
<xsl:param name="current.docid" select="/*/@id"/>

<!-- Use the literal scaling feature -->
<xsl:param name="literal.extensions">scale.by.width</xsl:param>

<!-- We want the TOC links in the titles, and in blue. -->
<xsl:param name="latex.hyperparam">colorlinks,linkcolor=blue,pdfstartview=FitH</xsl:param>

<!-- Put the dblatex logo -->
<xsl:param name="doc.publisher.show">1</xsl:param>

<!-- Show the list of examples too -->
<xsl:param name="doc.lot.show">figure,table,example</xsl:param>

<!-- DocBook like description -->
<xsl:param name="term.breakline">1</xsl:param>

<!-- Manpage titles not numbered -->
<xsl:param name="refentry.numbered">0</xsl:param>

<xsl:template match="parameter">
  <xsl:variable name="name" select="."/>
  <xsl:variable name="target" select="key('id',$name)[1]"/>

  <xsl:choose>
```

```

<xsl:when test="count($target) > 0">
  <!-- Hot link to the parameter refentry -->
  <xsl:call-template name="hyperlink.markup">
    <xsl:with-param name="linkend" select="$name"/>
    <xsl:with-param name="text">
      <xsl:apply-imports/>
    </xsl:with-param>
  </xsl:call-template>
  <!-- Index entry for this parameter -->
  <xsl:text>\index{Parameters!}</xsl:text>
  <xsl:value-of select="$name"/>
  <xsl:text>></xsl:text>
</xsl:when>
<xsl:otherwise>
  <!--
  <xsl:message>No reference for parameter: '<xsl:value-of
  select="$name"/>'</xsl:message>
  -->
  <xsl:apply-imports/>
</xsl:otherwise>
</xsl:choose>
</xsl:template>

<xsl:template match="sgmltag[@class='xmlpi']">
  <xsl:variable name="name" select="normalize-space(.)"/>
  <xsl:variable name="nameref" select="concat('pi-',translate($name,' ','_'))"/>
  <xsl:variable name="target" select="key('id',$nameref)[1]"/>

  <xsl:choose>
  <xsl:when test="count($target) > 0">
    <!-- Hot link to the parameter refentry -->
    <xsl:call-template name="hyperlink.markup">
      <xsl:with-param name="linkend" select="$nameref"/>
      <xsl:with-param name="text">
        <xsl:apply-imports/>
      </xsl:with-param>
    </xsl:call-template>
    <!-- Index entry for this parameter -->
    <xsl:text>\index{Processing Instructions!}</xsl:text>
    <xsl:value-of select="$name"/>
    <xsl:text>></xsl:text>
  </xsl:when>
  <xsl:otherwise>
    <!--
    <xsl:message>No reference for parameter: '<xsl:value-of
    select="$name"/>'</xsl:message>
    -->
    <xsl:apply-imports/>
  </xsl:otherwise>
  </xsl:choose>
</xsl:template>

</xsl:stylesheet>

```

5.4.2 Overriding some templates

You can directly put the overriding templates in your XSL stylesheet, but do not try to import the default dblatex stylesheets in it: it is automatically done by the tool. So, just focus on the template to override and dblatex will ensure that your definitions will get precedence over the dblatex ones.

You can of course split your templates in several files, and import each of them in the main user stylesheet by calling `xsl:import`.

Example 5.2 Overriding templates

```
<?xml version='1.0' encoding="iso-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version='1.0'>

<!-- Let's import our own XSL to override the default behaviour. -->
<xsl:import href="mystyle.xsl"/>

<!-- Let's patch directly a template here -->
<xsl:template match="article" mode="docinfo">
  <xsl:apply-imports/>
  <xsl:text>\let\mymacro=\DBKrelease</xsl:text>
</xsl:template>

</xsl:stylesheet>
```

5.5 Customized LaTeX style

The actual output rendering is done by the latex style package used, and not by the XSL stylesheets whose role is only to translate to latex. Users can provide their own LaTeX style file, in respect of some rules:

- The LaTeX style package preamble must support all the options that the XSL stylesheets can pass to the package.
- Some packages must be used to make all the thing work.
- The docbook interface must be defined: the XSL stylesheets register some elements information in LaTeX commands. These commands or macro are the only ones specific to DocBook that are explicitey used by the XSL stylesheets. Other specific macros are used but are not intended to be changed by the user. These hidden macros are defined in the `dbk_core` latex package.

The latex style file to use is specified by using the option `--texstyle latex_style`. An example of a simple LaTeX DocBook style is provided in the package.

The `--texstyle latex_style` option accepts a package name (no path and no `.sty` extension) or a full style file path. If a full path is used, the filename must ends with `.sty`.

```
# Give a package name and assume its path is already in TEXINPUTS
dblet --texstyle=mystyle file.xml

# Give the full package path. The TEXINPUTS is then updated by dblet
dblet --texstyle=./mystyle.sty file.xml
```

5.5.1 Reusing an existing LaTeX style

You can either create your latex style from scratch, in respect of the interfaces described in the following sections, or you can simply reuse an already existing style and override what you want. The latter method is easier for small things to change.

Here is an example of a style package reusing the default docbook style:

Example 5.3 Reused LaTeX style

```
%%
%% This style is derivated from the docbook one
%%
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{mystyle}[2007/04/04 My DocBook Style]
```

```

%% Just use the original package and pass the options
\RequirePackageWithOptions{docbook}

%% Redefine the paragraph layout
\setlength\parskip{\medskipamount}
\setlength\parindent{5pt}

%% Redefine some french settings
\babelsetup{fr}{%
  \catcode'\T1\guillemetleft =\active
  \catcode'\T1\guillemetright =\active
  \def\T1\guillemetleft {u\og\ignorespaces}
  \def\T1\guillemetright {v\unskip\fg}
}

```

5.5.2 Package options

A compliant LaTeX style package supports the following options. The options are provided by the XSL stylesheets according to the document attributes.

Option	Role
hyperlink, nohyperlink	Indicates if links in the document are provided or not
article, book	The document is an article or a book

5.5.3 Needed packages

A LaTeX style package must at least include the following packages.

Package	Description
dbk_core	Core LaTeX definitions and macros needed for DocBook

5.5.4 DocBook interface

All the latex commands beginning with DBK are related to elements under `bookinfo` or `articleinfo`.

Command	Description
\DBKreference	mapped to <code>pubsnumber</code>
\DBKsite	mapped to <code>address</code>
\DBKcopyright	mapped to <code>copyright</code>
\DBKdate	mapped to <code>date</code>
\DBKedition	mapped to <code>edition</code>
\DBKpubdate	mapped to <code>pubdate</code>
\DBKsubtitle	mapped to <code>subtitle</code>
\DBKreleaseinfo	mapped to <code>releaseinfo</code>
\DBKlegalnotice	environment mapped to a <code>legalnotice</code> . The legal notices are all put into the <code>\DBKlegalblock</code> command. It is up to the latex stylesheet to decide where to put it in the document.
\DBKlegalblock	wrapper command for the <code>\DBKlegalnotice</code> environments, used by the latex stylesheet to decide where to put the legal notices in the document.

Command	Description
<code>\DBKindexation</code>	This command contains the <code>othercredit</code> information translated to latex by the XSL. This command must be placed where the <code>othercredit</code> shall appear in the document.
<code>\DBKintable</code>	This environnement must be defined by the user to render the <code>othercredit</code> list. It can be displayed as a table, listitem, description list, or anything that suits your need.
<code>\DBKinditem</code>	This is an <code>othercredit</code> item.
<code>\DBKrevtable</code>	This environnement must be defined by the user to render the <code>revhistory</code> table. Untill now it is not really possible to customize it, since it must be a table with four columns, each column for a <code>revhistory</code> piece of information.
float example	This float is expected to be defined, and is mapped to <code>example</code> . It is not defined by default by the <code>dbk_core</code> package to allow the user to define its rendering (ruled or not, etc.)
float dbequation	This float is expected to be defined, and is mapped to <code>equation</code> . It is not defined by default by the <code>dbk_core</code> package to allow the user to define its rendering (ruled or not, etc.)

5.5.5 Debugging your Style

It is not surprising if your first `dblatex` compilation fails with a fresh LaTeX style. So, how to debug it when used with `dblatex`?

The following steps can help you:

1. Compile your file in the debug mode (option `-d`). When the compilation is done, the temporary working directory will not be removed.

```
$ dblatex --texstyle=./mystyle.sty -d file.xml
...
/tmp/tpub-ben-99629 is not removed
```

2. Go under the building temporary directory, and set the environment with the file `env_tex`.

```
$ cd /tmp/tpub-ben-99629
$ . env_tex
```

3. Compile the temporary latex file produced by the XSL stylesheets. Its name has the suffix `"_tmp.tex"`.

```
$ pdflatex file_tmp.tex
$ [ many outputs here ]
```

4. Now latex stops when it encounters an error so that you can debug your stylesheet.

5.6 Latex post process script

Extra user actions can be processed on the latex file produced by the XSL stylesheets or on its temporary working files produced by the latex compilation.

For instance, in the documents I write the cover page must display the number of pages of the document, but written in full letters (e.g. 23 is written “twenty three”). The latex post process script is then helpfull, and in this particular case it patches the `.aux` file.

The post process script is called just before the last latex compilation, and takes one parameter, the latex file compiled by the tool.

5.6.1 Post latex compilations

The latex compilations done once the script is called depend on the return code of the script:

- When the return code is 0, **dblatex** continues the compilation as many times as necessary.
- When the return code is 1, no more compilation is done by dblatex. This case is useful if the script needs to control precisely the number of compilation to apply. It is up to the script to perform the expected compilations.

To do so, the script can retrieve in the LATEX environment variable the actual compiler used by **dblatex**.

- When the return code is another value, an error is raised to signal a failed post process script execution.

5.6.2 Post processing with a Python Plugin

You can use a python plugin instead of a script by prefixing the plugin name with the string "plugin:". When using a plugin you must not put the python suffix in the plugin name. If the plugin is in one of the Python system directories, or in the current directory where you call dblatex, or in one of the directories of the PYTHONPATH environment variable, you don't need to specify a directory location. Otherwise put the plugin directory path before the plugin name.

Here are several plugin call examples:

```
# The texpost.py module is in one of the python paths
dblatex -r plugin:texpost file.xml

# The texpost.py module location is specified with an absolute path
dblatex -r plugin:/path/to/texpost file.xml

# The texpost.py module is specified through a relative path from current dir
dblatex -r plugin:relative/path/from/current/dir/texpost file.xml
```

The plugin must contain a main entry point. **Dblatex** will pass the following parameters to the entry point: *latex_file* to specify the latex file to post process, and *stdout* to specify the output stream to use to be consistent with the dblatex verbosity.

Example 5.4 Texpost Python Plugin Example

```
import sys
import os

def main(latex_file, stdout):
    """
    Texpost Plugin Entry point
    """
    # Let's find out the backend used
    tex_engine = os.environ["LATEX"]

    # Let's log something
    print >>stdout, "Plugin called on '%s' file" % (latex_file)

    # Open the latex file and parse it
    texfile = open(latex_file)
    ...

    # Now decide if a new compilation must occur
    if has_changed:
        sys.exit(0)
    else:
        sys.exit(1)
```

5.7 Dblatex Configuration File

A configuration file, also called a specification file, can be used to list all the customizations and options to apply. Such a file is passed by using the option `-S config_file`. Several configuration files can be specified if needed; it can be usefull to have a default setup and additional setup for specific needs that override or complete the default configuration.

5.7.1 XML Configuration File Format

You should use the XML format to configure dblatex. It contains more features than the text format used up to the 0.3.7 release that is now deprecated. The principle remains the same, that is, most of the configuration parameters correspond to a command line option.

Here is a full example of a configuration file. In the next sections the meaning and use of the configuration tags are detailed.

```
<?xml version="1.0" encoding="utf-8"?>
<!-- ===== -->
<!-- Dblatex Configuration example for building a book -->
<!-- ===== -->
<config xmlns="http://dblatex.sourceforge.net/config" version="1.0">
  <latex>
    <backend use="xetex"/>
    <texstyle use="dbsimple"/>
    <texpost fileref="dblatex-postprocess"/>
    <indexstyle fileref="../myindexstyle.ist"/>
    <texinputs>../mystyledir:/etc/texmf/userfont</texinputs>
  </latex>
  <xslt>
    <stylesheet fileref="user_param.xsl"/>
    <stylesheet fileref="xetex_param.xsl"/>
    <stylesheet fileref="pdf.xsl"/>
  </xslt>
  <imagedata>
    <converter src="svg" dst="*" docformat="pdf">
      <command>
        inkscape -z -D --export-dpi=600 --export-%(dst)s=%(output)s %(input)s
      </command>
    </converter>
  </imagedata>
  <options>-X -V</options>
</config>
```

Another example, much simpler, is the configuration file used for this manual.

Example 5.5 User Manual Configuration File

```
<?xml version="1.0" ?>
<!--
=====
Configuration file for dblatex documentation (manual, release notes)
=====
-->
<config xmlns="http://dblatex.sourceforge.net/config">
  <latex>
    <texinputs>../latex//</texinputs>
    <texstyle use="docbook"/>
    <backend use="pdftex"/>
  </latex>
  <xslt>
    <stylesheet fileref="manual.xsl"/>
  </xslt>
</config>
```

The following sections detail the meaning of each tag. Note that dblatex provides some schema that you can use to validate your configuration file.

5.7.1.1 <config>

It is the root element of a configuration file. A valid configuration file must have a `config` root element.

Attributes:

xmlns

The namespace to use for a dblatex configuration file is: `http://dblatex.sourceforge.net/config`.

The following elements occur in `config`: [latex](#), [xslt](#), [imagedata](#), [options](#).

5.7.1.2 <latex>

The `latex` element contains the configuration data related to the latex processing.

Attributes: none

The following elements occur in `latex`:

backend

It defines the latex engine to use, like the option `--backend` does. The backend name is specified through the `use` attribute.

texstyle

It defines the docbook latex style to use, like the option `--texstyle` does. The latex package is specified through the `use` attribute.

texpost

It defines the latex post-processing script to use, like the option `--texpost` does. The script name is specified through the `use` attribute or the `fileref` attribute.

indexstyle

It defines index style file (.ist) to use, like the option `--indexstyle` does. The filename is specified through the `fileref` attribute.

texinputs

The element text defines the extra paths to add to TEXINPUTS, like the option `--texinputs` does.

bibinputs

The element text defines the lookup paths of BIBINPUTS, like you would do by successive use of the option `--bib-path`.

bstinputs

The element text defines the lookup paths of BSTINPUTS, like you would do by successive use of the option `--bst-path`.

5.7.1.3 <xslt>

The `xslt` element contains the configuration data related to the XSL processing.

Attributes: none

The following elements occur in `xslt`:

stylesheet

It defines a user stylesheet to use, like the option `--xsl-user` does. The stylesheet name is specified through the `fileref` attribute. If several `stylesheet` elements are set, the precedence is the same like using the option `--xsl-user` several times.

engine

See [engine](#).

5.7.1.4 <engine>

It defines the XSLT processor to use; it can be either a predefined engine, or a user-defined command to run.

When the attribute `use` is used, it works like the option `--xslt`. The engine name is specified through the `use` attribute.

When the element contains the `command` or `commandchain` child element it defines the command(s) to run to perform the XSLT processing. The core keywords defined in [command](#) apply. The additional keywords of an XSLT engine command are the following:

% (xmlfile) s

Replaced by the XML source file to process.

% (stylesheet) s

Replaced by the stylesheet to use to process the XML file `xmlfile`.

% (param_list) s

Replaced by the list of the XSL parameters to set. The parameter pair formatting (the parameter name and its value) is specified through the `param-format` attribute given to the `engine` element.

Attributes:

param-format

This attribute is mandatory when the engine is specified through a `command`. It tells how a parameter name and value pair shall be formatted when added to the XSLT engine command. The formatting is given by a string containing the keywords replaced by the name or value of the parameter to set:

% (param_name) s

Replaced by the name of the parameter to set.

% (param_value) s

Replaced by the value to set to the parameter.

The following example re-writes with a user-defined command the Saxon engine with an additional `-T` option:

```
<xslt>
  <engine param-format="% (param_name) s=% (param_value) s">
    <command>
      saxon-xslt -T -o %(output)s %(xmlfile)s
                                     %(stylesheet)s %(param_list)s</command>
    </engine>
  </xslt>
```

5.7.1.5 <imagedata>

The `imagedata` element contains the rules and commands to convert on the fly the images included in the docbook document.

Attributes: none

The following elements occur in `imagedata`:

figpath

It defines a path where to find the images, like the option `--fig-path` does.

figformat

It defines the default image format when no suffix is available to deduce the actual format, like `--fig-format` does.

formatrule

It specifies in which format an image must be converted, depending on the context, given by some attributes:

docformat

The output document for which the rule applies. When not set the rule applies for any format

backend

The rule applies only when this backend is used

dst

When the conditions are met the image shall be converted to this format.

The following example:

```
<formatrule docformat="pdf" backend="xetex" dst="pdf"/>
```

tells that when a PDF document is built with the xetex backend, the graphics not natively supported shall be converted to PDF.

converter

It defines an image transformation rule. See [converter](#) for more details.

5.7.1.6 <converter>

The converter element defines a rule about how to convert on the fly an image in a docbook document to the corresponding image used in the built latex file. The goal is to have an image format compatible with the latex engine, whatever the original format is. **Dblatex** has a default set of rules, but one can overwrite or add some rules.

Attributes:

src

Format of the original image referenced in the XML document. The converter only applies to the images matching this format.

dst

Target format once converted. The converter only applies to the original images having the format `src` and that need to be converted to the format `dst`. `dst` can take the special value "*" meaning that the rule applies for all the possible target formats (e.g. eps, pdf, png).

docformat

Specify for which document output format the converter applies. For example, if `docformat` is set to "pdf", it means that the converter applies only if the document output format is PDF. The converter will not apply if you want to build a PostScript or DVI document.

backend

The converter applies only if it is this latex engine that is used. For example, it is usefull if you want to convert an image in a specific way only when xetex is used.

The following elements occur in `converter`:

commandchain

List of the commands to run. A `commandchain` contains one or more `command` elements.

command

command to execute to perform the conversion, or a part of the conversion if several commands are chained. See [command](#).

5.7.1.7 <command>

The command element contains the arguments of the command to run. The arguments can contain some predefined keywords that are replaced by the actual values when the command is executed. The known core keywords are:

% (src) s

Replaced by the `src` value (original image format).

% (dst) s

Replaced by the `dst` value (target image format).

% (input) s

Replaced by the input file required by the command. In the case of an image conversion, it is the filename of the original image to convert.

% (output) s

Replaced by the name of the output file required by the command. In the case of an image conversion, it is the filename of the output image once converted.

Attributes:

input

Standard input of the command to run. It can specify a file or it can specify to use the output of the preceding command if the keyword "PIPE" is used.

output

Standard output of the command to run. It can specify a file or it can specify that the output shall be redirected to the next command if the keyword "PIPE" is used.

shell

When set to "1" or "true", the command is run in a shell environment.

5.7.1.8 <options>

The options element lists the extra arguments to pass to **dblatex**.

5.7.2 Deprecated Text Configuration File Format

The format of the file is the following:

- Every comment starts with a "#", and is ignored.
- The file must contain one parameter by line.
- The format of a parameter is the following:

```
<keyword>: <value>
```

- Every parameter is mapped to an option that can be passed to **dblatex**.
- An unknown parameter is silently ignored (the whole line is dropped).
- The parameters defining a path (a file or a directory) can take absolute or relative paths. A relative path must be defined from the specification file itself. For instance, a specification file under `/the/spec/directory/` with a parameter describing the file `../where/this/file/is/myfile` points to `/the/spec/where/this/file/is/myfile`.

The following table lists the supported parameters and the corresponding command line option.

Keyword	Value	Corresponding option	Description
TexInputs	Directories	--texinputs	Defines extra path to add to TEXINPUTS
TexStyle	Latex package name	--texstyle	Defines the LaTeX style package to use.
TexPost	Script file name	--texpost	Defines the LaTeX post process script to use.

Keyword	Value	Corresponding option	Description
XslParam	Parameter file name	-p	Defines the parameter file to use.
FigInputs	Directories	-I	Defines the extra figures path.
Options	Command line options	None	Lists command options to use by default when using the tool. The options specified by the parameter are directly passed to dblatex

5.7.3 Style Paths

By default **dblatex** tries to find the configuration files related to a style (specified with option `-T`) in the following paths, in respect of the order:

1. The current directory
2. `$HOME/.dblatex`
3. `/etc/dblatex`
4. The dblatex package configuration directories.

You can add some extra paths where to look for by setting the `DBLATEX_CONFIG_FILES` environment variable. The paths are separated by ":" in Unix like systems, and by ";" on Windows. These paths are used only when nothing is found in the default paths.

5.8 Customization Precedence

All the customization queries are translated to the corresponding command line options. Thus, using several customization methods can be inconsistent because each of them override the same option with another value.

For instance, you can specify the use of a specification file in which it is said to use a latex style (parameter `TexStyle`) and explicitly use the `--texstyle` command line option. So, what is the behaviour?

The options order is the following:

- If a specification file is used (`-S` option), the options are set to the specification file parameters.
- If several specification files are used (`-S` option), the precedence is given to the last specified files.
- The options explicitly passed override the specification file setting, whatever is the position of the options (i.e. before or after the `-S` option).
- If an option is passed several times, this is the last occurrence that is used.

Example 5.6 Customization Precedence

Let's consider the specification file containing the following parameters:

```
<?xml version="1.0" ?>
<config xmlns="http://dblatex.sourceforge.net/config">
  <xslt><stylesheet fileref="file3.xsl"/></xslt>
  <options>-b pdftex</options>
  <latex><texstyle use="mystyle1"/></latex>
</config>
```

And now the command line:

```
dbletex -b dvips -p file1.xml -p file2.xml -S file.specs -s mystyle2 mydoc.xml
```

The setting used is the following:

- “-b dvips” overrides “-b pdftex” set by the spec file.
 - “-p file2.xml” overrides “-p file1.xml” since it is defined after, and overrides “file3.xml” set by the spec file.
 - “-s mystyle2” override “mystyle1” set by the spec file.
-

Chapter 6

FAQ

The purpose of this mini FAQ is to give some tips about how customizing or tweaking the PDF output.

6.1 My images are too big. What can I do?

When an image is included via `imagedata` with no scaling attributes (e.g. `width`, `height`, `contentwidth`) it is its natural size that is used.

One can change individually the size of an `imagedata` by defining its attributes (see [\[TDG\]](#) for more details). One can also use the parameter `imagedata.default.scale` to apply a systematic scaling rule on every image that has no explicit attribute.

The parameter `imagedata.default.scale` can take:

- The default predefined value "pagebound": the image natural size is used, up to the page boundaries. That is, if an image natural width is greater than the page width its size is proportionally reduced so that it is contained in the page. The same control is done for height.
- Any combination of valid `\includegraphics` options. For example

`imagedata.default.scale=scale=40%`

The scale 40% is applied on the images.

`imagedata.default.scale=width=40%,height=3in`

This example is weird but shows that several options can be used. In this case the image width is 40% of the page width, and the height is fixed to 3 inches. The risk to have an anamorphous result is very high here.

6.2 How can I have the PDF fit to height by default?

The behaviour of the PDF file when opened by a reader like Acrobat Reader can be customized with the parameter `latex.hyperparam`. See the section called "[Description](#)" for more details about this parameter.

To answer precisely to the question, set the parameter with the option "`pdfstartview=FitV`".

6.3 How can I have all the PDF hyperlinks in blue color?

Same answer than for the previous question.

For this particular case, set the parameter with the options "`linktocpage,colorlinks,linkcolor=blue,citecolor=blue,urlcolor=blue`".

6.4 How can I remove that stupid float rules?

If you wonder about this, you probably use the `db2latex` style. To remove the rules, you need to patch the `db2latex.sty`. You can:

- Simply remove the `floatstyle` definition for the floats for which you don't want the rules.
- Explicitly use the plain `floatstyle`. Note that using this explicit style does not allow to change the float title position anymore. The plain style always puts the title at the bottom of the float.

6.5 My long tables don't split in several pages. Why?

A formal table (`table` element) is put in a float, so that it can have a numbered caption and be placed by `tex` at the best place. The limitation is that a float cannot split over several pages.

For long tables that need to split, use `informaltable` instead.

6.6 I cannot put a table in an example.

A formal table (`table` element) is put in a float, and cannot be put in another float like an example. You can use an `informaltable` instead.

6.7 I cannot compile my cyrillic document. Why?

A document using some characters different from the roman alphabet may face some troubles, because `latex` natively handles only `latin1` encoding.

Try the different unicode supports provided by `dblatex`:



- Use the XeTeX backend, a new `tex` engine that natively supports Unicode characters: `dblatex -b xetex file.xml`.
- Ask for using the `latex` unicode support by setting the `latex.encoding=utf8` parameter: `dblatex -P latex.encoding=utf8 file.xml`.
- Use the Passivetex extensions by setting the `latex.unicode.use=1` parameter: `dblatex -P latex.unicode.use=1 file.xml`.

See Section [4.16.1](#) for more details.

Chapter 7

Thanks

7.1 Sponsors

Thanks to the companies that sponsored dblatex at a time or another:  ¹, and the Raphael Hertzog's company  ².

7.2 Contributors

Thanks to Andreas Hoenen who packages and maintains dblatex for Debian, provides patches and so on.

Thanks also to other contributors: David Hedley (newtbl implementor), and Nicolas Pernetty (Windows port).

7.3 Pioners

Thanks to the people who contributed to the project at its early age: Jean-Yves Le Ruyet, precursory and hard-working user, Julien Ducourthial for his precious help, Vincent Hottier who asked for the embedded LaTeX equations support.

Special thanks to the KDE documentation team (especially Philip Rodrigues), Michael Smith (from the DocBook Project), and Kai Bormmann, for their feedbacks, encouragements, and advice. They were the first people who believed in this project and boosted dblatex.

¹<http://www.froglogic.com/>

²<http://www.freexian.com/>

Appendix A

Dblatex XSL Parameter Reference

This is reference documentation for all user-configurable parameters in the dblatex XSL stylesheets.

The organization of the reference entries mimics the [FO Parameter Reference](#) for people familiar with the DocBook Project documentation.

A.1 Admonitions

figure.caution

figure.caution — Figure to use to render a `caution` block.

Synopsis

```
<xsl:param name="figure.caution">warning</xsl:param>
```

Description

Figure to use to render a `caution` block. This parameter is added to allow new latex styles to use their own figures in admonitions.

figure.important

figure.important — Figure to use to render a `important` block

Synopsis

```
<xsl:param name="figure.important">warning</xsl:param>
```

Description

Figure to use to render a `important` block. This parameter is added to allow new latex styles to use their own figures in admonitions.

figure.note

figure.note — Figure to use to render a `note` block

Synopsis

```
<xsl:param name="figure.note"/>
```

Description

Figure to use to render a `note` block. This parameter is added to allow new latex styles to use their own figures in admonitions.

figure.tip

figure.tip — Figure to use to render a `tip` block

Synopsis

```
<xsl:param name="figure.tip"/>
```

Description

Figure to use to render a `tip` block. This parameter is added to allow new latex styles to use their own figures in admonitions.

figure.warning

figure.warning — Figure to use to render a `warning` block

Synopsis

```
<xsl:param name="figure.warning">warning</xsl:param>
```

Description

Figure to use to render a `warning` block. This parameter is added to allow new latex styles to use their own figures in admonitions.

A.2 Callouts

callout.linkends.hot

callout.linkends.hot — Hot links callout items

Synopsis

```
<xsl:param name="callout.linkends.hot" select="'1'"/>
```

Description

The callouts referenced in a callout list are hot links if the parameter is set to 1. Then, the references are in red such like any other cross-reference link in the document.

calloutlist.style

calloutlist.style — Callout list style to apply

Synopsis

```
<xsl:param name="calloutlist.style" select="'leftmargin=1cm,style=sameline'"/>
```

Description

Defines how the callout list items are displayed. The value must be some valid enumitem description list options.

callout.markup.circled

callout.markup.circled — Use black circles for numbering the callout items?

Synopsis

```
<xsl:param name="callout.markup.circled" select="'1'"/>
```

Description

Set to 1 the callouts references in a `calloutlist` are white numbers in black circles, like the markups in the listing (or graphic). Set to 0 the references are simple numbers.

co.linkends.show

co.linkends.show — Show the references to calloutlist items next to the markup

Synopsis

```
<xsl:param name="co.linkends.show" select="'1'"/>
```

Description

Next to a callout markup the links to the corresponding calloutlist items are shown when the parameter is set to 1. Set to 0 the links are not shown.

imageobjectco.hide

imageobjectco.hide — Hide the callout markups on the image

Synopsis

```
<xsl:param name="imageobjectco.hide" select="0"/>
```

Description

When set to 1 the callout numbered circles are not drawn on the image. Only the anchors are put, allowing callout list items to jump at the referenced position on the image. The purpose of this parameter is to allow the use of images that already contain the callout numbers (like for GIMP manual).

A.3 ToC/LoT/Index Generation

doc.lot.show

doc.lot.show — Specifies the Lists of Titles to display

Synopsis

```
<xsl:param name="doc.lot.show">figure,table</xsl:param>
```

Description

Specifies which Lists of Titles should be printed after the Table of Content. The value is a comma separated list of the LoTs to show. The supported LoTs are "figure", "table", "equation", and "example". The list order represents the LoTs order in the output document.

doc.toc.show

doc.toc.show — Print the Table Of Contents

Synopsis

```
<xsl:param name="doc.toc.show">1</xsl:param>
```

Description

Print the table of contents when set to 1.

titleabbrev.in.toc

titleabbrev.in.toc — Should titleabbrev be put in the TOC instead of title?

Synopsis

```
<xsl:param name="titleabbrev.in.toc">1</xsl:param>
```

Description

Set to 1 the titleabbrev content is put in the TOC instead of the title.

toc.section.depth

toc.section.depth — How deep should recursive sections appear in the TOC?

Synopsis

```
<xsl:param name="toc.section.depth">5</xsl:param>
```

Description

Depth of the TOC. Used to set the latex `tocdepth` counter.

bibliography.tocdepth

bibliography.tocdepth — How bibliography section and subsections appear in TOC

Synopsis

```
<xsl:param name="bibliography.tocdepth">5</xsl:param>
```

Description

Same than *preface.tocdepth* for bibliography sections. Meaningful only when *bibliography.numbered* is set to 0.

See Also

bibliography.numbered.

refentry.numbered, *refentry.tocdepth*.

glossary.numbered, *glossary.tocdepth*.

index.numbered, *index.tocdepth*.

colophon.tocdepth

colophon.tocdepth — How colophon section and subsections appear in TOC

Synopsis

```
<xsl:param name="colophon.tocdepth">0</xsl:param>
```

Description

Same than *preface.tocdepth* for colophon sections.

dedication.tocdepth

dedication.tocdepth — How dedication section and subsections appear in TOC

Synopsis

```
<xsl:param name="dedication.tocdepth">0</xsl:param>
```

Description

Same than *preface.tocdepth* for dedication sections.

preface.tocdepth

preface.tocdepth — How preface section and subsections appear in TOC

Synopsis

```
<xsl:param name="preface.tocdepth">0</xsl:param>
```

Description

When greater than 0, the preface headings appear in the TOC. The parameter value define the preface section depth appearing in the TOC and in the bookmarks. If set to 0, none of the sections are put in the TOC. If set to 1, only the chapter level appears in the TOC and bookmarks, and so on. When the parameter is negative, it behaves like with 0, but it uses the previous implementation (use of unnumbered sections, that is, with latex heading commands ending with '*').

glossary.tocdepth

glossary.tocdepth — How glossary section and subsections appear in TOC

Synopsis

```
<xsl:param name="glossary.tocdepth">5</xsl:param>
```

Description

Same than *preface.tocdepth* for glossary sections. Meaningful only when *glossary.numbered* is set to 0.

See Also

[glossary.numbered](#).

[refentry.numbered](#), [refentry.tocdepth](#).

[bibliography.numbered](#), [bibliography.tocdepth](#).

[index.numbered](#), [index.tocdepth](#).

index.tocdepth

[index.tocdepth](#) — How index section and subsections appear in TOC

Synopsis

```
<xsl:param name="index.tocdepth">5</xsl:param>
```

Description

Same than [preface.tocdepth](#) for index sections. Meaningful only when [index.numbered](#) is set to 0.

See Also

[index.numbered](#).

[refentry.numbered](#), [refentry.tocdepth](#).

[bibliography.numbered](#), [bibliography.tocdepth](#).

[glossary.numbered](#), [glossary.tocdepth](#).

latex.index.tool

[latex.index.tool](#) — Define the tool to use to build an index

Synopsis

```
<xsl:param name="latex.index.tool"/>
```

Description

This parameter defines the LaTeX compatible index tool to use to build the document index. When empty the default tool used is **makeindex**. Currently **xindy** is the only other tool supported by dblatex. Read Section [4.13](#) for more information about index building.

See Also

[latex.index.language](#).

latex.index.language

latex.index.language — Force the language to use for index sorting

Synopsis

```
<xsl:param name="latex.index.language"/>
```

Description

This parameter forces the use of the specified index language whatever the document language is. The index language is used by some index tools like **xindy** that builds and sorts the index list. Read Section 4.13 for more information about index building.

See Also

latex.index.tool.

refentry.tocdepth

refentry.tocdepth — How refentry section and subsections appear in TOC

Synopsis

```
<xsl:param name="refentry.tocdepth">5</xsl:param>
```

Description

Same than *preface.tocdepth* for refentry sections. Meaningful only when *refentry.numbered* is set to 0.

A.4 Processor Extensions

alt.use

alt.use — Always use alt to display equations

Synopsis

```
<xsl:param name="alt.use" select="0"/>
```

Description

When an (informal) equation contains both alt and another element (graphic, etc.), and when *tex.math.in.alt* is not set to 'latex', the alt element is not used to display the equation since it is considered as a fallback element. Set *alt.use* to force the use of alt as default rendering element even when *tex.math.in.alt* is not set to 'latex'.

tex.math.in.alt

tex.math.in.alt — TeX notation used for equations

Synopsis

```
<xsl:param name="tex.math.in.alt" select="'latex'"/>
```

Description

Specifies if the `alt` element in an (informal) equation contains some tex equation. If so, and if the tex equation is in 'latex' format, the content is directly used by dblatex.

A.5 Automatic labelling

bibliography.numbered

bibliography.numbered — Should bibliography headings be numbered?

Synopsis

```
<xsl:param name="bibliography.numbered">1</xsl:param>
```

Description

Defines either the bibliography titles are numbered or not. When numbered, it is displayed as any other numbered section.

See Also

bibliography.tocdepth.

glossary.numbered, *glossary.tocdepth*.

refentry.numbered, *refentry.tocdepth*.

index.numbered, *index.tocdepth*.

glossary.numbered

glossary.numbered — Should glossary headings be numbered?

Synopsis

```
<xsl:param name="glossary.numbered">1</xsl:param>
```

Description

Defines either the glossary titles are numbered or not. When numbered, it is displayed as any other numbered section.

See Also

[*glossary.tocdepth*](#).

[*refentry.numbered*](#), [*refentry.tocdepth*](#).

[*bibliography.numbered*](#), [*bibliography.tocdepth*](#).

[*index.numbered*](#), [*index.tocdepth*](#).

index.numbered

[*index.numbered*](#) — Should index headings be numbered?

Synopsis

```
<xsl:param name="index.numbered">1</xsl:param>
```

Description

Defines either the index titles are numbered or not. When numbered, it is displayed as any other numbered section.

See Also

[*index.tocdepth*](#).

[*refentry.numbered*](#), [*refentry.tocdepth*](#).

[*bibliography.numbered*](#), [*bibliography.tocdepth*](#).

[*glossary.numbered*](#), [*glossary.tocdepth*](#).

refentry.numbered

[*refentry.numbered*](#) — Should refentry headings be numbered?

Synopsis

```
<xsl:param name="refentry.numbered">1</xsl:param>
```

Description

Defines either the refentry titles are numbered or not. When numbered, it is displayed as any other numbered section.

See Also

[refentry.tocdepth](#).

[glossary.numbered](#), [glossary.tocdepth](#).

[bibliography.numbered](#), [bibliography.tocdepth](#).

[index.numbered](#), [index.tocdepth](#).

A.6 Meta/*Info

doc.pdfcreator.show

[doc.pdfcreator.show](#) — Set the PDF metadata Creator field

Synopsis

```
<xsl:param name="doc.pdfcreator.show">1</xsl:param>
```

Description

Fill the Creator field of the PDF document information section with "DBLaTeX-<version>" if the parameter is set to 1. Set to 0 this field is keep untouched.

make.single.year.ranges

[make.single.year.ranges](#) — Print single-year ranges (e.g., 1998-1999)

Synopsis

```
<xsl:param name="make.single.year.ranges" select="0"/>
```

Description

If non-zero, year ranges that span a single year will be printed in range notation (1998-1999) instead of discrete notation (1998, 1999). Parameter taken from the DocBook XSL stylesheets.

make.year.ranges

[make.year.ranges](#) — Collate copyright years into ranges?

Synopsis

```
<xsl:param name="make.year.ranges" select="0"/>
```

Description

If non-zero, copyright years will be collated into ranges. Parameter taken from the DocBook XSL stylesheets.

A.7 Reference Pages

funcsynopsis.decoration

funcsynopsis.decoration — Decorate elements of a *funcsynopsis*?

Synopsis

```
<xsl:param name="funcsynopsis.decoration" select="1"/>
```

Description

If non-zero, elements of the *funcsynopsis* will be decorated (e.g. rendered as bold or italic text). The decoration is controlled by templates that can be redefined in a customization layer.

This parameter is taken from the DocBook Project XSL parameters.

funcsynopsis.style

funcsynopsis.style — What style of *funcsynopsis* should be generated?

Synopsis

```
<xsl:param name="funcsynopsis.style">ansi</xsl:param>
```

Description

If *funcsynopsis.style* is *ansi*, ANSI-style function synopses are generated for a *funcsynopsis*, otherwise K&R-style function synopses are generated.

This parameter is taken from the DocBook Project XSL parameters.

function.parens

function.parens — Generate parens after a function?

Synopsis

```
<xsl:param name="function.parens">0</xsl:param>
```

Description

If non-zero, the formatting of a *function* element will include generated parentheses.

This parameter is taken from the DocBook Project XSL parameters.

refclass.suppress

refclass.suppress — Suppress display of *refclass* contents?

Synopsis

```
<xsl:param name="refclass.suppress" select="0"/>
```

Description

Parameter taken from the DocBook Project.

See [refclass.suppress](#).

refentry.generate.name

refentry.generate.name — Output NAME header before refnames?

Synopsis

```
<xsl:param name="refentry.generate.name" select="0"/>
```

Description

If non-zero, a "NAME" section title is output before the list of refnames.

refentry.xref.manvolnum

refentry.xref.manvolnum — Output manvolnum as part of refentry cross-reference?

Synopsis

```
<xsl:param name="refentry.xref.manvolnum" select="'1'"/>
```

Description

if non-zero, the manvolnum is used when cross-referencing refentrys, either with xref or citerefentry.

A.8 Tables

newtbl.autowidth

newtbl.autowidth — Table column widths sized by latex

Synopsis

```
<xsl:param name="newtbl.autowidth"/>
```

Description

Defines if the table column widths must be automatically sized by latex. The allowed values are:

default

The automatic width (that is, latex is in charge to size the column width) is applied only to columns not having a specified `colspec` colwidth. It includes both undefined `colspec`, and `colspec` without the `colwidth` attribute.

all

the automatic width is applied to any column, whether a `colspec` is provided or not.

The default value is empty. This is the same as `none`.

The values `all` and `default` have identical effect in the tabular presentation of `segmentedlists`. (`segmentedlist` elements have no `colwidth` attribute.)

See Also

- Section [4.8.7.1](#)
- Section [4.8.7](#)
- [default.table.width](#)

newtbl.bgcolor.thead

[newtbl.bgcolor.thead](#) — Background color of the `thead` rows

Synopsis

```
<xsl:param name="newtbl.bgcolor.thead"/>
```

Description

Background color of the `thead` rows.

newtbl.default.colsep

[newtbl.default.colsep](#) — By default draw a vertical line between columns

Synopsis

```
<xsl:param name="newtbl.default.colsep" select="'1'"/>
```

Description

Set to 1, print the column separators when no `colspec` attribute is specified.

newtbl.default.rowsep

newtbl.default.rowsep — By default draw a horizontal line between rows

Synopsis

```
<xsl:param name="newtbl.default.rowsep" select="'1'"/>
```

Description

Set to 1, print the row separators when no `rowspec` attribute is specified.

newtbl.format.tbody

newtbl.format.tbody — LaTeX formatting for body table cells

Synopsis

```
<xsl:param name="newtbl.format.tbody"/>
```

Description

LaTeX formatting for body table cells.

newtbl.format.tfoot

newtbl.format.tfoot — LaTeX formatting for foot table cells

Synopsis

```
<xsl:param name="newtbl.format.tfoot"/>
```

Description

LaTeX formatting for foot table cells.

newtbl.format.thead

newtbl.format.thead — LaTeX formatting for head table cells

Synopsis

```
<xsl:param name="newtbl.format.thead">\bfseries%  
</xsl:param>
```

Description

LaTeX formatting for head table cells.

newtbl.use.hhline

newtbl.use.hhline — Draw the horizontal lines with the `hhline` package

Synopsis

```
<xsl:param name="newtbl.use.hhline" select="'0'"/>
```

Description

Set to 1, use the `hhline` package to draw the table row separators instead of `cline`. Using `hhline` seems more suited for colored tables.

table.continue.caption

table.continue.caption — Caption text for continued table titles

Synopsis

```
<xsl:param name="table.continue.caption" select="'(continued)'/>
```

Description

Text to display in the second (and following) captions of long tables expanding over more than one page. The default parameter value is `'(continued)'` (including the parentheses).

table.default.position

table.default.position — Default table float placement policy

Synopsis

```
<xsl:param name="table.default.position" select="'[htbp]'/>
```

Description

Default table float placement algorithm. The value must contain enclosing square braces (`[]`). Order is significant, the first match which produces acceptable output is used.

The meaning of the characters are:

h

Meaning: “here”

Place the float where it occurs in the DocBook source.

t

Meaning: “top”

Place the float at the top of the page.

b

Meaning: “bottom”

Place the float at the bottom of the page.

p

Meaning: “page”

Place the float on a special page just for floats.

Any of the characters may be followed by an exclamation point (!) to force LaTeX to be more aggressive when trying the placement algorithm.

The default value is `[htbp]`.

table.default.tabstyle

table.default.tabstyle — Default table style to apply

Synopsis

```
<xsl:param name="table.default.tabstyle"/>
```

Description

The parameter applies only for `informaltables` that do not have a `tabstyle` attribute. With `dblatex` its role is to specify the latex table environment to use, so the set value must be a valid latex environment. The supported values are:

longtable

The default table type used by `dblatex`, in order to be able to split over several pages. When the parameter is empty and no `tabstyle` attribute is defined, it is the applied value.

tabular

The most usual table type. Such table can only be on a single page.

tabularx

An advanced table type that allows to stretch column widths to the available remained page width. Such table can only be on a single page.

table.in.float

table.in.float — Use or emulate a float to display a formal table?

Synopsis

```
<xsl:param name="table.in.float" select="'1'"/>
```


Description

Determines whether formal tables float. Floating tables do not necessarily appear in the output document in the location in which they appear in the DocBook input, they may move when this improves the overall document layout. Non-floating tables always appear in the rendered document in the same place as they appear in the input DocBook document.

Informal tables never float. Tables that do not float are processed using the `longtable` package. Unlike the methods used to process floating tables, tables processed with `longtable` may span page boundaries. Floating tables may not cross page boundaries and so may not be larger than a single page.

The `longtable` package is limited in that titles must be positioned at the top of the table. This limits the positioning of non-floating table titles to the top of the table.

When `table.in.float` is 0 formal tables do not float.

The default is 1, formal tables float.

table.title.top

`table.title.top` — Title on top of the table float

Synopsis

```
<xsl:param name="table.title.top" select="'0'"/>
```

Description

Set to 1 the table float title position is above the table. Set to 0 the title is under the table. Meaningless when the table is not in a float (see `table.in.float`).

default.table.rules

`default.table.rules` — The default column and row rules for tables using HTML markup

Synopsis

```
<xsl:param name="default.table.rules">none</xsl:param>
```

Description

Tables using HTML markup elements can use an attribute named `rules` on the `table` or `informaltable` element to specify whether column and row border rules should be displayed. This parameter lets you specify a global default style for all HTML tables that don't otherwise have that attribute.

These are the supported values:

all

Rules will appear between all rows and columns.

rows

Rules will appear between rows only.

cols

Rules will appear between columns only.

groups

Rules will appear between row groups (thead, tfoot, tbody). No support for rules between column groups yet.

none

No rules. This is the default value.

The border after the last row and the border after the last column are not affected by this setting. Those borders are controlled by the `frame` attribute on the table element.

default.table.width

default.table.width — The default width of tables

Synopsis

```
<xsl:param name="default.table.width"/>
```

Description

If non-empty, this value controls table width. It is used for the `width` attribute on `tables` that do not specify an alternate width (with the `<?dblatex table-width?>` processing instruction, or with the *newtbl.autowidth* parameter).

The parameter takes one of the following values:

- A valid length (e.g. 15cm),
- A percentage of the page width (e.g. 75%),
- A keyword telling to apply an automatic column width (e.g. `autowidth.all`).

See Also

- Section [4.8.2](#)
- Section [4.8.7](#)
- *newtbl.autowidth*

A.9 Linking

latex.hyperparam

latex.hyperparam — Options/parameters passed to `hyperref`

Synopsis

```
<xsl:param name="latex.hyperparam"/>
```

Description

This parameter gives the options to pass to the LaTeX hyperref package. No validity check is done.

For instance, the Table of Content rendering (link color, etc.) can be changed. Look at the hyperref.sty documentation to know all the hyperref options available.

Example A.1 Configuring with latex.hyperparam

```
<?xml version='1.0' encoding="iso-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version='1.0'>

<!-- We want TOC links in the titles (not in the page numbers), and blue.
-->
<xsl:param name="latex.hyperparam">colorlinks,linkcolor=blue</xsl:param>

</xsl:stylesheet>
```

Olink Parameters

current.docid, insert.olink.page.number, insert.olink.pdf.frag, olink.base.uri, olink.debug, olink.doctitle, olink.lang.fallback.sequence, prefer.internal.olink, target.database.document, targets.filename, use.local.olink.style
 — Parameters to configure Olinks

Synopsis

```
<xsl:param name="current.docid"/>

<xsl:param name="insert.olink.page.number">yes</xsl:param>

<xsl:param name="insert.olink.pdf.frag" select="1"/>

<xsl:param name="olink.base.uri" select="''"/>

<xsl:param name="olink.debug" select="0"/>

<xsl:param name="olink.doctitle" select="'yes'"/>

<xsl:param name="olink.lang.fallback.sequence" select="''"/>

<xsl:param name="prefer.internal.olink" select="0"/>

<xsl:param name="target.database.document" select="''"/>

<xsl:param name="targets.filename" select="'target.db'"/>

<xsl:param name="use.local.olink.style" select="0"/>
```

Description

These parameters are taken from the DocBook Project XSL parameters and must be used as described in the original reference:

- *current.docid*,
 - *insert.olink.page.number*,
 - *insert.olink.pdf.frag*,
-

- `olink.base.uri`,
- `olink.debug`,
- `olink.doctitle`,
- `olink.lang.fallback.sequence`,
- `prefer.internal.olink`,
- `target.database.document`,
- `targets.filename`,
- `use.local.olink.style`,

A.10 Cross References

`insert.xref.page.number`

insert.xref.page.number — Turns page numbers in xrefs on and off

Synopsis

```
<xsl:param name="insert.xref.page.number">maybe</xsl:param>
```

Description

The value of this parameter determines if cross references (`xrefs`) in printed output will include page number citations. It has three possible values.

no

No page number references will be generated.

yes

Page number references will be generated for all `xref` elements. The style of page reference may be changed if an `xrefstyle` attribute is used.

maybe

Page number references will not be generated for an `xref` element unless it has an `xrefstyle` attribute whose value specifies a page reference.

`insert.xref.page.number.para`

insert.xref.page.number.para — Turns page numbers in xrefs to paragraphs on and off

Synopsis

```
<xsl:param name="insert.xref.page.number.para">yes</xsl:param>
```

Description

The value of this parameter determines if cross references (`xrefs`) to paragraphs in printed output will include page number citations.

Historically, cross references to paragraphs included page number citations unconditionally, regardless of the `insert.xref.page.number` value.

yes

Page number references will be generated for paragraphs.

maybe

Whether page number references are generated for an `xref` element referring to a paragraph will be controlled by the `insert.xref.page.number` rules.

xref.hypermarkup

`xref.hypermarkup` — Wrap the entire the `xref` markups with an hyperlink?

Synopsis

```
<xsl:param name="xref.hypermarkup" select="0"/>
```

Description

When set to 1, the whole cross reference markup produced for an `xref` is converted to an hyperlink.

When not set, the default latex hyperlinking is done, that usually means that only the reference numbers are hot.

A.11 Lists**term.breakline**

`term.breakline` — Put the term description on the next line?

Synopsis

```
<xsl:param name="term.breakline">0</xsl:param>
```

Description

Set to 1 the item following a term in a variable list is put on the next line.

variablelist.term.separator

`variablelist.term.separator` — Text to separate terms within a multi-term varlistentry

Synopsis

```
<xsl:param name="variablelist.term.separator">, </xsl:param>
```

Description

When a `varlistentry` contains multiple `term` elements, the string specified in the value of the `variablelist.term.separator` parameter is placed after each `term` except the last.

A.12 QAndASet**qanda.defaultlabel**

`qanda.defaultlabel` — Sets the default for defaultlabel on QandASet.

Synopsis

```
<xsl:param name="qanda.defaultlabel">number</xsl:param>
```

Description

If no `defaultlabel` attribute is specified on a `qandaset`, this value is used. It must be one of the legal values for the `defaultlabel` attribute, one from `none`, `number` or `qanda`. The default value is 'number'.

Meaning

`qanda` - questions are labeled "Q:" and answers are labeled "A:".

`number` - The entries are enumerated.

`none` - No distinguishing label precedes Questions or Answers.

A.13 Bibliography**biblioentry.item.separator**

`biblioentry.item.separator` — Text to separate bibliography entries

Synopsis

```
<xsl:param name="biblioentry.item.separator">, </xsl:param>
```

Description

Text to separate bibliography entries.

biblioentry.numbered

`biblioentry.numbered` — Should `biblioentry` item labels be numbered?

Synopsis

```
<xsl:param name="biblioentry.numbered" select="0"/>
```

Description

When non-zero, the reference labels of the bibliography entries are numbers instead of label strings, even if explicit labels are set through an `xreflabel` attribute or an `abbrev` tag.

citation.default.style

citation.default.style — Default natbib citation style to apply

Synopsis

```
<xsl:param name="citation.default.style"/>
```

Description

Default natbib citation style to apply when natbib is used.

See Also

- Section [4.14.3](#)
- [dblatex citestyle](#)

citation.natbib.options

citation.natbib.options — Specifies the natbib package options

Synopsis

```
<xsl:param name="citation.natbib.options"/>
```

Description

Options to pass to the natbib package when it is loaded. See also Section [4.14.3](#).

citation.natbib.use

citation.natbib.use — Use natbib to display citations

Synopsis

```
<xsl:param name="citation.natbib.use" select="'0'"/>
```

Description

Load the natbib package, and allows the use of natbib citation styles. The package is loaded if the parameter is set to 1. See Section [4.14.3](#).

latex.bibfiles

latex.bibfiles — Defines the default BibTeX database to use

Synopsis

```
<xsl:param name="latex.bibfiles">'</xsl:param>
```

Description

Defines the default BibTeX database to use. Used when the bibtex PI does not have a "bibfiles" attribute. See Section [4.14.2](#) for more details.

latex.biblio.output

latex.biblio.output — Defines how the BibTeX bibliographic entries are printed out

Synopsis

```
<xsl:param name="latex.biblio.output">all</xsl:param>
```

Description

The available print mode values are:

all

Print all the entries contained in the databases.

cited

Print only the entries cited in the document.

notcited

Print only the entries *not* cited in the document.

The default print mode is 'all'.

See Also

- Section [4.14.2](#)
- [bibtex](#)

latex.biblio.style

latex.biblio.style — Default BibTeX style to apply

Synopsis

```
<xsl:param name="latex.biblio.style"/>
```

Description

Defines the default BibTeX style to apply. Meaningful when not empty, only for the used bibtex databases.

See Also

- Section [4.14.2](#)
- [bibtex](#)

latex.bibwidelabel

[latex.bibwidelabel](#) — Template of the widest bibliography label

Synopsis

```
<xsl:param name="latex.bibwidelabel">WIDELABEL</xsl:param>
```

Description

The environment bibliography accepts a parameter that indicates the widest label, which is used to correctly format the bibliography output. The value of this parameter is output inside the `\begin{thebibliography}[]` LaTeX command.

A.14 Glossary

glossterm.auto.link

[glossterm.auto.link](#) — Generate links from glossterm to glossentry automatically?

Synopsis

```
<xsl:param name="glossterm.auto.link" select="0"/>
```

Description

When set to 1, the glossterms in the document are linked to their definition in the glossary.

A.15 Miscellaneous

annotation.support

[annotation.support](#) — Enable the annotation support

Synopsis

```
<xsl:param name="annotation.support" select="'0'"/>
```

Description

Set to 1 the experimental DocBook 5 `annotation support` is enabled.

beginpage.as.pagebreak

beginpage.as.pagebreak — Force a page break when set to 1. Otherwise `beginpage` does nothing.

Synopsis

```
<xsl:param name="beginpage.as.pagebreak" select="1"/>
```

Description

The legacy behaviour of `dblatex` is that it inserts a page break when `beginpage` appears. You can keep this behaviour by setting this parameter to 1 (default value) or inhibit it by setting to 0.

doc.section.depth

doc.section.depth — Depth of the section numbering

Synopsis

```
<xsl:param name="doc.section.depth">5</xsl:param>
```

Description

Depth of the section numbering. Used to set the latex `secnumdepth` counter.

endnotes.heading.command

endnotes.heading.command — LaTeX command of the endnotes headings

Synopsis

```
<xsl:param name="endnotes.heading.command"/>
```

Description

With this parameter you can provide your own latex macro to make the endnotes headings. If not set, **dblatex** uses the default sectioning commands like `\section*`, `\subsection*`, and so on. The command takes one parameter that is the title markup of the heading. For more details see Section 4.18.2.4.

See Also

[*endnotes.heading.groups*](#)

[*endnotes.heading.command*](#)

endnotes.heading.groups

[*endnotes.heading.groups*](#) — Display the endnotes by groups

Synopsis

```
<xsl:param name="endnotes.heading.groups"/>
```

Description

In the endnote section, the notes are grouped according to the level defined by the elements listed in this parameter. Each group has a heading that references the element that contains these notes. By default, dblatex can group per chapter or per part. For more details see Section [4.18.2.4](#).

See Also

[*endnotes.heading.style*](#)

[*endnotes.heading.command*](#)

endnotes.heading.style

[*endnotes.heading.style*](#) — Style of the heading of endnotes groups

Synopsis

```
<xsl:param name="endnotes.heading.style" select="'select:title'"/>
```

Description

This parameter is provided to format the heading of an endnote group. It uses the `xrefstyle` syntax, and therefore one can specify directly a template, or select some keywords to substitute. For more details see Section [4.18.2.4](#).

See Also

[*endnotes.heading.groups*](#)

[*endnotes.heading.command*](#)

endnotes.properties

[*endnotes.properties*](#) — Properties of the latex package `endnotes` setup

Synopsis

```
<xsl:attribute-set name="endnotes.properties.default">
  <xsl:attribute name="package">endnotes</xsl:attribute>

  <!-- No header: endnotes are embedded in another section -->
  <xsl:attribute name="heading">\mbox{}\par</xsl:attribute>

  <!-- Show end notes as a numbered list -->
  <xsl:attribute name="font-size">\normalsize</xsl:attribute>
  <xsl:attribute name="note-format">%
    \leftskip=1.8em\noindent
    \makebox[0pt][r]{\theenmark.~~\rule{0pt}{\baselineskip}}\ignorespaces
  </xsl:attribute>
</xsl:attribute-set>

<xsl:attribute-set name="endnotes.properties"
  use-attribute-sets="endnotes.properties.default"/>
```

Description

These properties are used to configure the `endnotes` package when the footnotes are printed as endnotes. Each attribute defines the content of the setup macros of the package. The user should always override the attributes of `endnotes.properties` and leave `endnotes.properties.default` untouched. See Section 4.18.2 for more details.

See Also

[*footnote.as.endnote*](#)

equation.default.position

[*equation.default.position*](#) — Default equation float placement to apply

Synopsis

```
<xsl:param name="equation.default.position">[H]</xsl:param>
```

Description

Default equation float placement algorithm to apply. See `figure.default.position` for more details about how to use latex float specifications.

example.default.position

[*example.default.position*](#) — Default example float placement to apply

Synopsis

```
<xsl:param name="example.default.position">[H]</xsl:param>
```

Description

Default example float placement algorithm to apply. See `figure.default.position` for more details about how to use latex float specifications.

example.float.type

`example.float.type` — Select the type of float for example elements

Synopsis

```
<xsl:param name="example.float.type">none</xsl:param>
```

Description

Selects the type of float for example elements.

- If `example.float.type` is “none”, then no float is used. It may be usefull when the example must be displayed exactly where it is defined or when it must cover several pages.
- If `example.float.type` is "float", then the float environment is used. The actual float placement is computed by latex according to the `floatstyle` attribute if specified, or according to the `example.default.position` parameter.

See Also

`example.default.position`.

`figure.default.position`.

figure.anchor.top

`figure.anchor.top` — Cross reference anchor on top of the figure float

Synopsis

```
<xsl:param name="figure.anchor.top" select="$figure.title.top"/>
```

Description

By default the anchor of a formal figure is where the figure title appears. It means that a link to the figure points after the image when the title is below the image, and at the top of the image when the title is on the top. This parameter allows to put the anchor at the top of the image even if the title is below the figure.

Note that it is not possible to have the anchor below the image if the title is on the top.

See Also

`figure.title.top`

figure.default.position

figure.default.position — Figure float placement policy

Synopsis

```
<xsl:param name="figure.default.position">[htbp]</xsl:param>
```

Description

Default figure float placement algorithm to apply. The default parameter value is [htbp] meaning that latex tries to place the figure where it occurs first (h, here), then at the top of the page (t), at the bottom of the page (b), and finally on the next page (p).

figure.title.top

figure.title.top — Title on top of the figure float

Synopsis

```
<xsl:param name="figure.title.top">0</xsl:param>
```

Description

Set to 1 the *figure* float title position is above the image. Set to 0 the title is under the image.

filename.as.url

filename.as.url — Hyphenate a filename like if it was an URL

Synopsis

```
<xsl:param name="filename.as.url">1</xsl:param>
```

Description

Set to 1 the *filenames* are handled as URLs, with the same hyphenation rules. Set to 0 the *filename* hyphenation is forced for each character.

footnote.as.endnote

footnote.as.endnote — Use footnotes as endnotes

Synopsis

```
<xsl:param name="footnote.as.endnote" select="0"/>
```

Description

When set to 1, the footnotes are displayed as endnotes, that is, all the notes appear in a section at the end of the document. See Section 4.18.2 for more details.

See Also

[endnotes.properties](#)

hyphenation.format

[hyphenation.format](#) — Predefined formats to hyphenate

Synopsis

```
<xsl:param name="hyphenation.format">monoseq,sansseq</xsl:param>
```

Description

This parameter replaces the parameter [monoseq.hyphenation](#) and its role is more general. It contains a comma separated list of the predefined inline formats that must be aggressively hyphenated. It means that all the elements rendered with these formats are then hyphenated.

The predefined formats that can be hyphenated when contained in the list are:

monoseq

Monospaced font.

sansseq

Sans serif like font.

boldmonoseq

Bold monospaced font.

italicmonoseq

Italic monospaced font.

If one wants no hyphenation at all, he must set the parameter to "nohyphen". In this case dblatex tries to avoid overfull boxes (words in the margins) but keeps the words splittable only on their natural hyphenation points which can end up to no hyphenation at all for literal words like URLs or technical terms.

See Also

[monoseq.hyphenation](#)

linenumbering.scope

[linenumbering.scope](#) — Templates on which the global linenumbering setup applies

Synopsis

```
<xsl:param name="linenumbering.scope"/>
```

Description

List of the templates for which the `linenumbering.default` parameter applies. The list separator is a space. For example, when the parameter is set to "screen programlisting", the two other linenumbering parameters will have effects only on `screen` and `programlisting` templates.

See Also

`linenumbering.default`, `linenumbering.everyNth`.

linenumbering.default

`linenumbering.default` — Default linenumbering setup for literal environments

Synopsis

```
<xsl:param name="linenumbering.default"/>
```

Description

This parameter defines the default linenumbering setup to apply to the templates listed in `linenumbering.scope` that do not have a `linenumbering` attribute.

The parameter value must be consistent and therefore compliant to the values allowed for the `linenumbering` attribute.

See Also

`linenumbering.scope`, `linenumbering.everyNth`.

linenumbering.everyNth

`linenumbering.everyNth` — Indicate which lines should be numbered

Synopsis

```
<xsl:param name="linenumbering.everyNth"/>
```

Description

If line numbering is enabled, everyNth line will be numbered. Note that numbering is one based, not zero based.

literal.layout.options

`literal.layout.options` — Override the options passed to the `listing` package

Synopsis

```
<xsl:param name="literal.layout.options"/>
```

Description

Overwrite the default options passed to the `\lstset` command.

literal.lines.showall

literal.lines.showall — Show the last empty lines in the literal environments?

Synopsis

```
<xsl:param name="literal.lines.showall">1</xsl:param>
```

Description

Set to 1, all the lines in a verbatim environment like `programlisting` or `screen` are printed, even if they are empty. Set to 0, the last empty lines are not printed. It is set to 1 by default.

literal.width.ignore

literal.width.ignore — Ignore the literal environment width attribute

Synopsis

```
<xsl:param name="literal.width.ignore">0</xsl:param>
```

Description

When set to 1 the `programlisting` and `screen` width attribute is ignored. In this case all the verbatim environment widths are equal to the enclosing environment width.

literal.class

literal.class — Default class to apply to `literallayout` blocks

Synopsis

```
<xsl:param name="literal.class">monospaced</xsl:param>
```

Description

Default class to apply when no `class` attribute is set, to render `literallayout` environments.

literal.role

literal.role — Default role to apply to verbatim blocks

Synopsis

```
<xsl:param name="literal.role"/>
```

Description

When not empty and when no `role` attribute is specified for a block verbatim block to render, this parameter is applied as formatting role.

The allowed values are specified in Section 4.12.1. If the value is not one of the allowed value, the role is then ignored.

literal.environment

literal.environment — Latex environment used to format verbatim blocks

Synopsis

```
<xsl:param name="literal.environment">
<xsl:choose>
<xsl:when test="contains($literal.extensions,'scale')">
  <xsl:text>lstcode</xsl:text>
</xsl:when>
<xsl:otherwise>
  <xsl:text>lstlisting</xsl:text>
</xsl:otherwise>
</xsl:choose>
</xsl:param>
```

Description

It specifies the Latex environment to use to render a verbatim block.

Since **dblatex** uses the listings package capabilities to handle the DocBook attributes, the specified environment must have the same capabilities if written from scratch, but it is recommended to just extend the listings environment with you own listings extension. See Section 4.12.4 for more details.

The specified environment name must starts with "lst". Otherwise **dblatex** raises an error and exits.

See Also

literal.extensions.

literal.extensions

literal.extensions — Enable extended verbatim features

Synopsis

```
<xsl:param name="literal.extensions">0</xsl:param>
```

Description

When this parameter is empty, **dblatex** only uses options supported by the listings package, in order to ensure that any verbatim environment built upon this standard package will work.

When this parameter contains "scale", **dblatex** uses additional options handled only by the dblatex version of the verbatim environment to provide scaling capabilities (see Section 4.12.2).

See Also

[literal.environment](#).

mediaobject.caption.style

[mediaobject.caption.style](#) — Font style of the mediaobject caption text

Synopsis

```
<xsl:param name="mediaobject.caption.style">\slshape</xsl:param>
```

Description

Font style of the mediaobject caption text. Its value can be any valid latex font style command combinations. By default this parameter put the caption text to italics.

monoseq.hyphenation

[monoseq.hyphenation](#) — Specifies one of the supported monoseq hyphenation policy

Synopsis

```
<xsl:param name="monoseq.hyphenation">1</xsl:param>
```

Description

This parameter is obsoleted in version 0.3.4 and has no more effect. It has been replaced by the [hyphenation.format](#).

Its role was: when set to 1, aggressively hyphenates the inlined element rendered with monoseq fonts. When set to 0, let latex do as default. When set to 'nohyphen', dblatex tries to avoid overfull boxes (words in the margins) but keeps the monoseq words not splittable.

See Also

[hyphenation.format](#)

monoseq.small

[monoseq.small](#) — Use a smaller font to render monoseq portions of text

Synopsis

```
<xsl:param name="monoseq.small">0</xsl:param>
```

Description

When set to 1, choose a smaller font to the element rendered with monoseq fonts.

Tip

If you use the XeTeX backend you do not need to use this parameter. Instead, you can configure the monospaced font used with a scale option as follow, through the parameter *xetex.font* or via a user latex style:

```
\setmonofont[Scale=MatchLowercase]{DejaVu Sans Mono}
```

pdf.annot.options

pdf.annot.options — PDF text annotations rendering options

Synopsis

```
<xsl:param name="pdf.annot.options"/>
```

Description

Options to change how the PDF text annotations should look. The supported options are width, height, depth. The options must be comma separated like: width=5cm, depth=10cm.

segmentedlist.as.table

segmentedlist.as.table — Format segmented lists as tables?

Synopsis

```
<xsl:param name="segmentedlist.as.table" select="0"/>
```

Description

If non-zero, segmentedlists will be formatted as tables. If you want a local setup for a specific segmentedlist you can use the list-presentation Processing Instruction instead.

When formatted as table, some table Processing Instructions can be used to customize the rendering.

See Also

- [dblatex list-presentation](#)
- Inline presentation:
 - [seg.item.separator](#)
- Table presentation:
 - Section [4.8.2](#)
 - [newtbl.autowidth](#)
 - [default.table.width](#)
 - [dblatex colwidth](#)
 - [dblatex autowidth](#)
 - [dblatex table-width](#)

seg.item.separator

seg.item.separator — Separator to use between several segs

Synopsis

```
<xsl:param name="seg.item.separator">, </xsl:param>
```

Description

Defines the separator to use between several segs.

show.comments

show.comments — Display remark elements?

Synopsis

```
<xsl:param name="show.comments" select="1"/>
```

Description

If non-zero, comments will be displayed, otherwise they are suppressed. Comments here refers to the `remark` element (which was called `comment` prior to DocBook 4.0), not XML comments (`<--` like this `-->`) which are unavailable.

Note

Dblatex uses the PDF Text Annotations capabilities to handle comments and remarks.

texlive.version

texlive.version — Version of the installed Tex Live Distribution

Synopsis

```
<xsl:param name="texlive.version">2007-10</xsl:param>
```

Description

The version number is used to adapt the tex output. For instance the URL output depends on the url package behaviour that differs from one release to another.

ulink.footnotes

ulink.footnotes — Generate footnotes for ulinks?

Synopsis

```
<xsl:param name="ulink.footnotes" select="0"/>
```

Description

If non-zero, and if *ulink.show* also is non-zero, the URL of each ulink will appear as a footnote.

Dblatex Limitation

The URL cannot be shown in a footnote if the ulinks are in list terms or heading titles.

ulink.show

ulink.show — Display URLs after ulinks?

Synopsis

```
<xsl:param name="ulink.show" select="0"/>
```

Description

If non-zero, the URL of each ulink will appear after the text of the link. If the text of the link and the URL are identical, the URL is suppressed.

See also *ulink.footnotes*.

The global *ulink.show* and *ulink.footnotes* setting can be overridden for each ulink that uses an xrefstyle like the following examples:

```
<!-- show 'Hot Text [URL]' or 'Hot Text (foot)' even if $ulink.show=0 -->
<ulink xrefstyle="url.show" url="http://www.a.site.org/path">Hot Text</ulink>

<!-- show 'Hot Text' even if $ulink.show=1 -->
<ulink xrefstyle="url.hide" url="http://us3.a.site.org/path">Hot Text</ulink>

<!-- show 'Hot Text [URL]' even if $ulink.show=0 and $ulink.footnotes=1 -->
<ulink xrefstyle="url.show.after" url="http://www.a.site.org/path">Hot Text</ulink>
```

```
<!-- show 'Hot Text (foot)' even if $ulink.show=0 and $ulink.footnotes=0 -->
<ulink xrefstyle="url.show.infofoot" url="http://www.a.site.org/path">Hot Text</ulink>

<!-- show 'Hot Text (foot)' if $ulink.show=1 and even if $ulink.footnotes=0 -->
<ulink xrefstyle="url.infofoot" url="http://www.a.site.org/path">Hot Text</ulink>

<!-- show 'Hot Text [URL]' if $ulink.show=1 and even if $ulink.footnotes=1 -->
<ulink xrefstyle="url.after" url="http://www.a.site.org/path">Hot Text</ulink>
```

xref.with.number.and.title

xref.with.number.and.title — Use number and title in cross references

Synopsis

```
<xsl:param name="xref.with.number.and.title" select="0"/>
```

Description

A cross reference may include the number (for example, the number of an example or figure) and the `title` which is a required child of some targets. This parameter inserts both the relevant number as well as the title into the link.

A.16 Graphics

imagedata.boxed

imagedata.boxed — Put the images into a framed box

Synopsis

```
<xsl:param name="imagedata.boxed">0</xsl:param>
```

Description

If set to 1, put the images into a framed box.

imagedata.default.scale

imagedata.default.scale — Specifies the default image scaling properties

Synopsis

```
<xsl:param name="imagedata.default.scale">pagebound</xsl:param>
```

Description

Default scale to apply to every `imagedata` that does not contain any scaling attribute.

By default this parameter is set to ``pagebound`` so that the included images keep their natural size up to the page boundaries.

Two other special parameters are available: `'maxwidth=width'` and `'maxheight=height'` where *width* and *height* define the maximum image dimensions, i.e. the image keeps its natural size up to the specified maximum dimension. Both `'maxwidth'` and `'maxheight'` settings can be combined in a comma separated list.

Example:

```
dblatex -P imagedata.default.scale=maxwidth=10cm,maxheight=8cm file.xml
```

Except these special reserved values, the expected value of the parameter must be some valid options passed to the `\includegraphics` command.

imagedata.file.check

imagedata.file.check — Make the latex compilation robust to missing images

Synopsis

```
<xsl:param name="imagedata.file.check">1</xsl:param>
```

Description

When set to 1 some tex code is added to ensure that latex compilation does not fail when the referenced `imagedata` file does not exist.

keep.relative.image.uris

keep.relative.image.uris — Should image URIs be resolved against `xml:base`?

Synopsis

```
<xsl:param name="keep.relative.image.uris" select="0"/>
```

Description

If non-zero, relative URIs (in, for example `fileref` attributes) will be used in the generated output. Otherwise, the URIs will be made absolute with respect to the base URI.

Note that the stylesheets calculate (and use) the absolute form for some purposes, this only applies to the resulting output.

A.17 Chunccking

set.book.num

set.book.num — Select a single book or all the books to compile from a set

Synopsis

```
<xsl:param name="set.book.num">1</xsl:param>
```

Description

When the document root element is a `set` this parameter can be used to select the book to print, or to publish all the books contained in the set when the value is "all". **dblatex** can only chunk the set in several books, and is not able to publish several books in a single file.

See Section 4.3.2 for more details.

use.id.as.filename

use.id.as.filename — Use ID value of chunk elements as the filename?

Synopsis

```
<xsl:param name="use.id.as.filename" select="0"/>
```

Description

If *use.id.as.filename* is non-zero, the filename of chunk elements that have IDs will be derived from the ID value.

This parameter is used only when chunking occurs, that is, when a set of books is published.

A.18 Pagination and General Styles

page.height

page.height — The height of the physical page

Synopsis

```
<xsl:param name="page.height"/>
```

Description

The page height is generally calculated from the *paper.type* parameter.

page.margin.bottom

page.margin.bottom — The bottom margin of the page

Synopsis

```
<xsl:param name="page.margin.bottom"/>
```

Description

The bottom page margin is the distance from the bottom of the body including the footer to the physical bottom of the page. When empty the used latex style layout applies.

page.margin.inner

page.margin.inner — The inner page margin

Synopsis

```
<xsl:param name="page.margin.inner"/>
```

Description

The inner page margin is the distance from bound edge of the page to the first column of text.

The inner page margin is the distance from bound edge of the page to the outer edge of the first column of text.

In left-to-right text direction, this is the left margin of recto (front side) pages. For single-sided output, it is the left margin of all pages.

In right-to-left text direction, this is the right margin of recto pages. For single-sided output, this is the right margin of all pages.

page.margin.outer

page.margin.outer — The outer page margin

Synopsis

```
<xsl:param name="page.margin.outer"/>
```

Description

The outer page margin is the distance from non-bound edge of the page to the outer edge of the last column of text.

In left-to-right text direction, this is the right margin of recto (front side) pages. For single-sided output, it is the right margin of all pages.

In right-to-left text direction, this is the left margin of recto pages. For single-sided output, this is the left margin of all pages.

page.margin.top

page.margin.top — The top margin of the page

Synopsis

```
<xsl:param name="page.margin.top"/>
```

Description

The top page margin is the distance from the physical top of the page to the top of the body including the header.

page.width

page.width — The width of the physical page

Synopsis

```
<xsl:param name="page.width"/>
```

Description

The page width does not need to be specified when *paper.type* is set.

paper.type

paper.type — Select the paper type

Synopsis

```
<xsl:param name="paper.type"/>
```

Description

The paper type is a convenient way to specify the paper size. The list of known paper sizes includes USletter and most of the A, B, and C sizes.

When the parameter is empty, the default layout of the latex style used is applied.

When not empty, the value is directly passed to the latex geometry package, and therefore it must be a valid value understood by this package.

geometry.options

geometry.options — Raw options to pass to the geometry package

Synopsis

```
<xsl:param name="geometry.options"/>
```

Description

The content of this parameter is directly passed as options to the latex geometry package, when the page layout setup is customized by the user. Page layout customization is done through setting one or more of the pagination parameters (see below). If *geometry.options* is set but none of the other parameters is set, nothing is passed to geometry because the package is loaded only when the layout is not the default one.

See Also

[*page.height*](#), [*page.margin.bottom*](#), [*page.margin.inner*](#), [*page.margin.outer*](#), [*page.margin.top*](#), [*page.width*](#), [*paper.type*](#).

doc.alignment

[*doc.alignment*](#) — Specifies the text alignment of the document

Synopsis

```
<xsl:param name="doc.alignment"/>
```

Description

Defines the text alignment for the whole document. The valid values are: "left", "center", "right", "justify". An empty string is equivalent to "justify".

doc.collab.show

[*doc.collab.show*](#) — Print the document collaborators (authors, etc.) in a table

Synopsis

```
<xsl:param name="doc.collab.show">1</xsl:param>
```

Description

Show the collaborators (authors, contributors) defined in the document information block.

doc.layout

[*doc.layout*](#) — Specifies the overall document layout.

Synopsis

```
<xsl:param name="doc.layout">coverpage toc frontmatter mainmatter index </xsl:param>
```

Description

The parameter configures the overall document layout, i.e. it specifies if the document must contain a coverpage, some TOC/LOTs, a frontmatter layout (for metadata/preface/dedication rendering), an index section, etc.

See also the [*doc.lot.show*](#) and [*doc.toc.show*](#) parameters to specify the content of the TOC/LOTs.

doc.publisher.show

[*doc.publisher.show*](#) — Print the dblatex logo on the cover page?

Synopsis

```
<xsl:param name="doc.publisher.show">0</xsl:param>
```

Description

Print the dblatex logo on the cover page for the native docbook style if the parameter is set to 1.

draft.mode

draft.mode — Select draft mode

Synopsis

```
<xsl:param name="draft.mode">maybe</xsl:param>
```

Description

Print `releaseinfo` in a framed box in the header, when the parameter is set to 'yes'. The `releaseinfo` is ignored if the parameter is set to 'no', or if the `releaseinfo` content is empty. When the parameter is set to 'maybe', the draft mode is deduced from the `status` attribute of the root element if set to 'draft'.

draft.watermark

draft.watermark — Print a Watermak on each page in draft mode?

Synopsis

```
<xsl:param name="draft.watermark">1</xsl:param>
```

Description

Print the draft text (that is, "DRAFT") as a watermark on each page, if the document is in draft mode and if the parameter is set to '1'.

latex.engine.options

latex.engine.options — Extra arguments to pass to the TeX engine

Synopsis

```
<xsl:param name="latex.engine.options"/>
```

Description

This parameter enables to pass some options or extra arguments to the TeX engine used to compile the latex file built by **dblatex**. This can be usefull to tweak the PDF generation to meet some specific needs that cannot be set through latex commands. The following example asks to **xelatex** to produce PDF version 1.3 and to embed all the fonts in the PDF file:

```
$ dblatex -b xetex -V -P latex.engine.options="-output-driver='xdvipdfmx -V3 -E'" file.xml
[...]
xelatex -output-driver=xdvipdfmx -V3 -E -interaction=batchmode file.tex
[...]
xelatex -output-driver=xdvipdfmx -V3 -E -interaction=batchmode file.tex
[...]
```

latex.class.article

latex.class.article — LaTeX document class to use for article documents

Synopsis

```
<xsl:param name="latex.class.article">article</xsl:param>
```

Description

This parameter sets the document class to use for article documents.

latex.class.book

latex.class.book — LaTeX document class to use for book documents

Synopsis

```
<xsl:param name="latex.class.book">report</xsl:param>
```

Description

This parameter sets the document class to use for book documents.

latex.class.options

latex.class.options — Options passed to the `\documentclass` command

Synopsis

```
<xsl:param name="latex.class.options"/>
```

Description

Options passed to the `\documentclass` command.

latex.encoding

latex.encoding — Encoding of the latex document to produce

Synopsis

```
<xsl:param name="latex.encoding">latin1</xsl:param>
```

Description

Encoding of the latex document to produce. The supported values are: "latin1" and "utf8". See Section 4.16.1 for more details about how to use it.

latex.unicode.use

latex.unicode.use — Use passivetex unicode support?

Synopsis

```
<xsl:param name="latex.unicode.use">0</xsl:param>
```

Description

Set to 1 the passivetex unicode support is included, allowing to handle a wider range of Unicode characters (like cyrillic).

latex.output.revhistory

latex.output.revhistory — Print the revhistory table?

Synopsis

```
<xsl:param name="latex.output.revhistory">1</xsl:param>
```

Description

The *revhistory* data are formatted as a table of the revisions if the parameter is non-zero. If the parameter is zero all the *revhistory* data are skipped.

A.19 Font Families

body.font.family

body.font.family — The default font family for body text

Synopsis

```
<xsl:param name="body.font.family">DejaVu Serif</xsl:param>
```

Description

The body font family is the default font used for text in the page body.

Currently this parameter is taken into account only when the xetex backend is used, through the [xetex.font](#).

See Also

[xetex.font](#)

cjk.font

[cjk.font](#) — Fonts to use in CJK environments

Synopsis

```
<xsl:param name="cjk.font">cyberbit</xsl:param>
```

Description

Fonts to use in CJK environments (i.e. for Chinese, Japanese or Korean documents handled by the CJK package).

monospace.font.family

[monospace.font.family](#) — The default font family for monospace environments

Synopsis

```
<xsl:param name="monospace.font.family">DejaVu Sans Mono</xsl:param>
```

Description

The monospace font family is used for verbatim environments (program listings, screens, etc.).

Currently this parameter is taken into account only when the xetex backend is used, through the [xetex.font](#).

See Also

[xetex.font](#)

sans.font.family

[sans.font.family](#) — The default sans-serif font family

Synopsis

```
<xsl:param name="sans.font.family">DejaVu Sans</xsl:param>
```

Description

The default sans-serif font family.

Currently this parameter is taken into account only when the xetex backend is used, through the [xetex.font](#).

See Also

[xetex.font](#)

xetex.font

[xetex.font](#) — Specifies the fonts that XeTeX must use

Synopsis

```
<xsl:param name="xetex.font">
  <xsl:value-of select="concat(' \setmainfont{' , $body.font.family, ' }&#10;' )"/>
  <xsl:value-of select="concat(' \setsansfont{' , $sans.font.family, ' }&#10;' )"/>
  <xsl:value-of
    select="concat(' \setmonofont{' , $monospace.font.family, ' }&#10;' )"/>
</xsl:param>
```

Description

Font specification for XeTeX. Meaningful only when the xetex backend is used. By default the parameter sets the XeTeX configuration from the parameters [body.font.family](#), [sans.font.family](#), and [monospace.font.family](#) but you can completely override it to define a more complex setup.

See Also

[body.font.family](#), [sans.font.family](#), [monospace.font.family](#)

A.20 Localization

korean.package

[korean.package](#) — Package included when Korean language is used

Synopsis

```
<xsl:param name="korean.package">CJK</xsl:param>
```

Description

When lang is set to 'ko' and the parameter is set to 'CJK' the CJK package is included to handle the Korean language.

latex.babel.language

latex.babel.language — Force the loaded babel language

Synopsis

```
<xsl:param name="latex.babel.language"/>
```

Description

This parameter forces the use of the specified babel language whatever the document language is.

latex.babel.use

latex.babel.use — Disable the use of babel, whatever the document language is

Synopsis

```
<xsl:param name="latex.babel.use">1</xsl:param>
```

Description

Set to 1 the babel package corresponding to the document language is included. Set to 0 no babel package is included whatever the document language is.

A.21 Prepress

crop.marks

crop.marks — Output crop marks?

Synopsis

```
<xsl:param name="crop.marks" select="0"/>
```

Description

If non-zero, crop marks will be added to each page. Crop marks are produced with the latex crop package.

crop.paper.type

crop.paper.type — Select the paper type for paper with crops

Synopsis

```
<xsl:param name="crop.paper.type"/>
```

Description

The paper type is a convenient way to specify the paper size. The list of known paper sizes includes USletter and most of the A, B, and C sizes.

This parameter gives the paper type of the physical page containing crops and the usefull page. The crop paper type is valid only if the overall size actually contains the usefull page whose sizes are defined with the page layout parameters.

The crop paper type is used only if *crop.marks* parameter is not set to zero.

Note

The usefull page is always centered in the crop page, and there is no parameter to change this behaviour.

crop.page.width

crop.page.width — The width of the physical crop page

Synopsis

```
<xsl:param name="crop.page.width"/>
```

Description

The crop page width does not need to be specified when *crop.paper.type* is set.

The parameter is used only if *crop.marks* parameter is not set to zero.

Note

The usefull page is always centered in the crop page, and there is no parameter to change this behaviour.

crop.page.height

crop.page.height — The height of the physical crop page

Synopsis

```
<xsl:param name="crop.page.height"/>
```

Description

The crop page height is generally calculated from the `crop.paper.type` parameter.

The parameter is used only if `crop.marks` parameter is not set to zero.

Note

The usefull page is always centered in the crop page, and there is no parameter to change this behaviour.

crop.mode

`crop.mode` — How to display crops

Synopsis

```
<xsl:param name="crop.mode" select="'cam'"/>
```

Description

Mode to pass to the crop package defining the form of the crops to display when `crop.marks` is not set to zero.

The allowed values are detailed in the crop package manual.

crop.options

`crop.options` — Raw options passed to the crop package

Synopsis

```
<xsl:param name="crop.options"/>
```

Description

Options directly passed to the crop package when crop marks are asked with the `crop.marks` set to non-zero.

Appendix B

Dblatex Processing Instruction Reference

This is reference documentation for all processing instructions in the dblatex XSL stylesheets.

B.1 Bibliography

bibtex

`<?bibtex?>` — Specify how to fill a Bibliography with BibTeX

Synopsis

```
<?bibtex bibfiles="bib/database1,bib/database2"
          bibstyle="plain"
          mode="cited"?>
```

Description

This Programming Instruction is usefull to include a bibliography coming from an external bibliography database written in the well known BibTeX format. **Dblatex** will call **bibtex** to process the database.

The attributes supported by the PI are:

bibfiles

This attribute is mandatory and specifies the databases to use. The databases are separated by commas, and must not contain the file suffix (`.bib`). The bibfiles paths must be absolute or relative to the base directory of the document. You can also add some bibfile paths by using the `-L` option.

bibstyle

Optional attribute specifying the bibliographic style to apply for rendering the databases. You can also change globally the style to apply with the [latex.biblio.style](#).

The actual style file used by **bibtex** is searched in the default paths, but some extra paths can be added by using the `-l` option.

mode

Optional print mode. The available values are:

all

Print all the entries contained in the databases.

cited

Print only the entries cited in the document.

notcited

Print only the entries *not* cited in the document.

When the attribute is not used, the `latex.biblio.output` parameter is used as print mode. By default the print mode is set to 'all'.

Context

Recognized in the `bibliomixed` element.

See Also

- Section [4.14.2](#)
- [latex.biblio.style](#)
- [latex.biblio.output](#)

dblatex citestyle

`<?dblatex citestyle?>` — Sets Natbib Citation Style

Synopsis

```
<?dblatex citestyle="\citep[see][chap. #2]"?>
```

Description

Applies the given natbib citation command to a citation. As the value of `citestyle` is a LaTeX command, if square brackets “[]” are part of the citation text then the text must be enclosed in curly brackets “{ }” to protect them.

Context

Recognized in the `citation` element, only when the [citation.natbib.use](#) parameter is set to 1.

See Also

[citation.default.style](#)

Section [4.14.3](#)

B.2 LaTeX

If you do inject arbitrary LaTeX into the output stream you will probably need to use the techniques found in Section [5.5.5](#).

latex

`<?latex?>`, `<?db2latex?>` — Insert Arbitrary text into the LaTeX file

Synopsis

```
<?latex content ?>
```

Description

These Processing Instructions are unusual in that they have no “attribute”. Instead, the *content* is put directly into the generated LaTeX document.

The `<?db2latex?>` PI is deprecated and should not be used anymore. It was provided for backward compatibility. Use the `<?latex?>` PI instead.

This can provide the ultimate in customization, but can be very dangerous since it breaks the abstractions provided by dblatex.

Warning



These Processing Instructions can be used to inject arbitrary LaTeX into the output stream. Please check if a safer customization method can be used instead. See the other [customization methods](#).

While it is pretty clear that insertion of certain sorts of LaTeX text, at some points within the document, will always be expected to work, there is no guarantee that *any* inserted LaTeX will continue to work in the future with newer versions of dblatex or its underlying software stack.

Inserting LaTeX into CDATA, i.e. in places where document text appears, is likely to be more robust than inserting LaTeX elsewhere. But again, no guarantees are made.

Care must be taken with whitespace. Leading and trailing spaces matter within the processing instruction. The *content* consists of everything following the first space and before the closing `>`. Further care must be taken with any whitespace which follows the processing instruction -- the TeX tokenization scan may consume whitespace which follows the processing instruction. One possible solution when this happens is to end your *content* with the `\` character.

The `<?latex?>` Processing Instruction takes special steps to work in verbatim blocks, `screen`, `programlisting`, and `literallayout` elements, while `<?db2latex?>` is directly written as is.

Safe LaTeX Insertions

Although the `latex` processing instruction can be dangerous there are a few `latex content` values that are safe to use within CDATA words, within the words of the actual text of your document. The following *content* values provide LaTeX with information it can use to improve your document’s formatting:

`\-`

Soft hyphen. A backslash followed by a dash indicates a soft hyphen. LaTeX may or may not break the line and hyphenate at this point. Useful when over-long variable names and similar fail to hyphenate and thereby cause lines to exceed their normal lengths.

Once a soft hyphen is inserted into a word the insertion point becomes the only place at which hyphenation is permitted in that occurrence of the word.

`{ }`

Do not kern (join together) characters. An empty pair of curly braces placed between two characters indicates that the characters should not be joined together. Depending on the font and the characters, some pairs of characters may be joined together. E.g. a pair of lower-case `f` characters may be have a their crossbars joined. This may not be desirable, as when each of the characters is a component of a separate word within a larger compound word. Placing this PI between the characters which are to remain separated prevents the two characters from kerning.

`{\kern0pt }`

Really, do not kern (join together) characters. A stronger version of `{ }`.

`\@`

A backslash followed by the at sign, when placed after a capital letter and before a period, indicates that the period is the end of a sentence. Periods following capital letters do not otherwise end sentences and sentences may be used by LaTeX when determining layout.

Context

Recognized in all elements.

B.3 Miscellaneous

dblatex angle

`<?dblatex angle?>` — Rotates an imagedata Image

Synopsis

```
<?dblatex angle="90" ?>
```

Description

Rotates the designated image by the angle given, expressed in degrees.

The default is no rotation.

Context

Recognized in the `imagedata` element.

texmath delimiters

`<?texmath delimiters?>` — Disable The LaTeX Mathematical Delimiters Control

Synopsis

```
<?texmath delimiters="user" ?>
```

Description

By default **dblatex** checks that consistent mathematical delimiters or environment are used in `alt` and it inserts the default math mode delimiters if `dblatex` thinks they are missing, but you can ask `dblatex` to write directly the `alt` content without any action. To do this, use the `texmath` Processing Instruction with `delimiters` set to 'user'.

Context

Recognized in the `alt` element.

See Also

Section [4.10.1.3](#).

B.4 Tables

dblatex bgcolor

`<?dblatex bgcolor?>` — Sets the Background Color of A Table Element

Synopsis

```
<?dblatex bgcolor="#00FF00"?>
<?dblatex bgcolor="{blue}"?>
<?dblatex bgcolor="[gray]{0.8}"?>
```

Description

Set the background color of a table element. The color of a child element is inherited from its parent, with the exception of the color of row cells. These have a color which is inherited from the column in which they appear so long as no color is specified for the row. How the color is set, whether by attribute or by processing instruction, does not matter.

The color can be expressed in hexadecimal notation like for HTML (e.g. #C0C0C0) or in a syntax understood by the `colortbl` latex package.

The default is the value of the table's `bgcolor` attribute, or white when not otherwise specified.

Context

Recognized in the following elements¹:

- `col`
- `colgroup`
- `entry`
- `entrytbl`
- `row`

See Also

- See Section [4.8.12](#)

dblatex table-width

`<?dblatex table-width?>` — Control the Automatic Width of a Table

Synopsis

```
<?dblatex table-width="12cm" ?>
```

¹HTML table elements have attributes which allow color to be specified.

Description

Set the width of a table. Used only when there is at least one column in the table without a fixed width, the *table-width* has precedence over other determinants of a table's width.

The *table.width* takes the values allowed for the [default.table.width](#) param, plus more. The values shared with [default.table.width](#) are:

- A valid length (e.g. 15cm),
- A percentage of the page width (e.g. 75%),
- A keyword telling to apply an automatic column width (e.g. `autowidth.all`).

Automatic Column Width

If you want to apply an automatic width only to some specific tables you can put the Processing Instruction `<?dblatex table-with="autowidth.scope" ?>` in the related tables. The PI has precedence over the [newtbl.autowidth](#) parameter.

The *scope* can take the following values, like in the [newtbl.autowidth](#) parameter:

default

The automatic width (that is, latex is in charge to size the column width) is applied only to columns not having a specified `colspec` colwidth. It includes both undefined `colspec`, and `colspec` without the `colwidth` attribute.

all

the automatic width is applied to any column, whether a `colspec` is provided or not.

In addition to these keywords, *table.width* can also take for its value keywords of the form `autowidth.column: N ...`, where *N* is the number of the column (counting from 1) which is to have its width automatically sized.

There is no default.

Context

Recognized in the `table` element.

See Also

- Section [4.8.2](#)
- Section [4.8.7.2](#)
- [default.table.width](#)
- [newtbl.autowidth](#)

B.5 Lists

dblatex autowidth

`<?dblatex autowidth?>` — Column sizing by LaTeX in `segmentedlist` tables

Synopsis

```
<?dblatex autowidth="all" ?>
```

Description

Defines, when `segmentedlists` are presented in tabular form, whether the column widths are automatically sized by latex. The allowed values are:

default

The automatic width (that is, latex is in charge to size the column width) is applied only to columns not having a specified `colspec` colwidth. It includes both undefined `colspec`, and `colspec` without the `colwidth` attribute.

all

the automatic width is applied to any column, whether a `colspec` is provided or not.

In addition to these keywords, `autowidth` can also take for its value keywords of the form `column: N ...`, where N is the number of the column (counting from 1) which is to have its width automatically sized. Columns with numbers not referenced, and not otherwise sized, will all have widths equally apportioned from available space.

The values `all` and `default` have identical effect in the tabular presentation of `segmentedlists`. (`segmentedlist` elements have no `colwidth` attribute.)

The `<?dblatex autowidth?>` processing instruction supersedes column width sizing specified by the `default.table.width` parameter or `dblatex table-width` processing instruction.

The default depends on whether the overall table width is specified. When there is no fixed table width columns default to the LaTeX default, the width of the widest cell. This is the default when no parameters or processing instructions are used. When a fixed table width is specified the default is “*”, proportional spacing.

Context

Recognized in the `segmentedlist` element.

See Also

- Section [4.8.2](#)
- [dblatex table-width](#)
- [dblatex colwidth](#)
- [segmentedlist.as.table](#)
- [dblatex list-presentation](#)
- [newtbl.autowidth](#)
- [default.table.width](#)

dblatex colwidth

`<?dblatex colwidth?>` — Specifies a column width for `segmentedlists` presented as tables

Synopsis

```
<?dblatex colwidth="3cm" ?>
```

Description

Specifies a column width for the columns of a `segmentedlist` when presented as a table. (See [segmentedlist.as.table](#).) This is the width of the cell content and does not include inter-cell spacing.

The `colwidth` attribute may take any of the `colspec`'s `colwidth` attribute's values. These are:

- A valid TeX length (e.g. 4cm).
- A proportional width specification of the form $N*$, where N is a positive integer.
- A combination of the above (e.g. $3*+1\text{cm}$). Spaces are not allowed.
- Any of the keywords allowed the [default.table.width](#) parameter: `autowidth.default`, `autowidth.all`, and `autowidth.none`. The first two, `autowidth.default` and `autowidth.none`, have the same meaning -- LaTeX determines the width of the column based on the width of the widest cell value. `autowidth.none` means the same as “*”, proportional spacing.

The `<?dblatex colwidth?>` processing instruction supersedes any column width specified by the [default.table.width](#) parameter or [dblatex table-width](#) processing instruction.

The default depends on whether the overall table width is specified. When there is no fixed table width columns default to the LaTeX default, the width of the widest cell. This is the default when no parameters or processing instructions are used. When a fixed table width is specified the default is “*”, proportional spacing.

Context

Recognized in the `segtitle` element.

See Also

- Section [4.8.2](#)
- [dblatex table-width](#)
- [segmentedlist.as.table](#)
- [dblatex list-presentation](#)
- [newtbl.autowidth](#)
- [default.table.width](#)

dblatex list-presentation

`<?dblatex list-presentation?>` — Specifies presentation style for a `segmentedlist`

Synopsis

```
<?dblatex list-presentation="list|table" ?>
```

Description

Use the `<?dblatex list-presentation?>` PI as a child of a `segmentedlist` to control the presentation style for the list (to cause it, for example, to be displayed as a table).

The value of `list-presentation` must be one of the following:

list

Display as a term-by-term list. The `segmentedlist`'s `seg` values are output like `variablelist listitem` values, one per line. The `segtitle` values appear as `variablelist terms` values do. Unlike a `variablelist`, the list is formatted without indentation.

table

Display as a table. The `segtitle` values are the column headings.

Context

Recognized in the `segmentedlist` element.

See Also

- [segmentedlist.as.table](#)
- Inline presentation:
 - [seg.item.separator](#)
- Table presentation:
 - Section [4.8.2](#)
 - [dblatex colwidth](#)
 - [dblatex table-width](#)
 - [newtbl.autowidth](#)
 - [dblatex autowidth](#)
 - [default.table.width](#)

Chapter 8

Index

P

Parameters, 64

- alt.use, 71
- annotation.support, 89
- beginpage.as.pagebreak, 90
- biblioentry.item.separator, 86
- biblioentry.numbered, 86
- bibliography.numbered, 68, 70, 72–74
- bibliography.tocdepth, 68, 70, 72–74
- body.font.family, 111, 113
- callout.linkends.hot, 65
- callout.markup.circled, 66
- calloutlist.style, 66
- citation.default.style, 41, 87
- citation.natbib.options, 41, 87
- citation.natbib.use, 41, 87
- cjk.font, 112
- co.linkends.show, 66
- colophon.tocdepth, 18, 68
- crop.marks, 114–116
- crop.mode, 116
- crop.options, 116
- crop.page.height, 115
- crop.page.width, 115
- crop.paper.type, 114–116
- dedication.tocdepth, 18, 69
- default.table.rules, 30, 81
- default.table.width, 21, 24, 82
- doc.alignment, 108
- doc.collab.show, 108
- doc.layout, 108
- doc.lot.show, 67, 108
- doc.pdfcreator.show, 74
- doc.publisher.show, 108
- doc.section.depth, 18, 90
- doc.toc.show, 67, 108
- draft.mode, 109
- draft.watermark, 109
- endnotes.heading.command, 46, 90, 91
- endnotes.heading.groups, 45, 91
- endnotes.heading.style, 46, 91
- endnotes.properties, 44, 91, 95
- equation.default.position, 92
- example.default.position, 92, 93
- example.float.type, 31, 93
- figure.anchor.top, 93
- figure.caution, 64
- figure.default.position, 93, 94
- figure.important, 64
- figure.note, 65
- figure.tip, 65
- figure.title.top, 93, 94
- figure.warning, 65
- filename.as.url, 94
- footnote.as.endnote, 44, 92, 94
- funcsynopsis.decoration, 75
- funcsynopsis.style, 75
- function.parens, 75
- geometry.options, 107
- glossary.numbered, 68–70, 72–74
- glossary.tocdepth, 68–70, 72–74
- glossterm.auto.link, 89
- hyphenation.format, 32, 95, 99
- imagedata.boxed, 103
- imagedata.default.scale, 103
- imagedata.file.check, 104
- imageobjectco.hide, 66
- index.numbered, 68, 70, 72–74
- index.tocdepth, 68, 70, 72–74
- insert.xref.page.number, 84, 85
- insert.xref.page.number.para, 84
- keep.relative.image.uris, 104
- korean.package, 113
- latex.babel.language, 114
- latex.babel.use, 114
- latex.bibfiles, 88
- latex.biblio.output, 41, 88, 118
- latex.biblio.style, 40, 88, 117
- latex.bibwidelabel, 89
- latex.class.article, 110
- latex.class.book, 110
- latex.class.options, 110
- latex.encoding, 42, 62, 111
- latex.engine.options, 109
- latex.hyperparam, 82
- latex.index.language, 39, 70, 71

- latex.index.tool, [39](#), [70](#), [71](#)
- latex.output.revhistory, [111](#)
- latex.unicode.use, [42](#), [62](#), [111](#)
- linenumbering.default, [96](#)
- linenumbering.everyNth, [96](#)
- linenumbering.scope, [95](#), [96](#)
- literal.class, [35](#), [97](#)
- literal.environment, [38](#), [98](#), [99](#)
- literal.extensions, [36](#), [98](#)
- literal.layout.options, [38](#), [96](#)
- literal.lines.showall, [97](#)
- literal.role, [35](#), [36](#), [97](#)
- literal.width.ignore, [97](#)
- make.single.year.ranges, [74](#)
- make.year.ranges, [74](#)
- mediaobject.caption.style, [99](#)
- monoseq.hyphenation, [95](#), [99](#)
- monoseq.small, [99](#)
- monospace.font.family, [112](#), [113](#)
- newtbl.autowidth, [23](#), [24](#), [76](#), [82](#), [122](#)
- newtbl.bgcolor.thead, [77](#)
- newtbl.default.colsep, [77](#)
- newtbl.default.rowsep, [78](#)
- newtbl.format.tbody, [31](#), [78](#)
- newtbl.format.tfoot, [78](#)
- newtbl.format.thead, [31](#), [78](#)
- newtbl.use.hhline, [79](#)
- page.height, [105](#), [108](#)
- page.margin.bottom, [105](#), [108](#)
- page.margin.inner, [106](#), [108](#)
- page.margin.outer, [106](#), [108](#)
- page.margin.top, [106](#), [108](#)
- page.width, [107](#), [108](#)
- paper.type, [105](#), [107](#), [108](#)
- pdf.annot.options, [100](#)
- preface.tocdepth, [18](#), [68–71](#)
- qanda.defaultlabel, [86](#)
- refclass.suppress, [75](#)
- refentry.generate.name, [76](#)
- refentry.numbered, [68](#), [70–73](#)
- refentry.tocdepth, [68](#), [70–74](#)
- refentry.xref.manvolnum, [76](#)
- sans.font.family, [112](#), [113](#)
- seg.item.separator, [101](#)
- segmentedlist.as.table, [100](#)
- set.book.num, [14](#), [104](#)
- show.comments, [101](#)
- table.continue.caption, [79](#)
- table.default.position, [79](#)
- table.default.tabstyle, [25](#), [31](#), [80](#)
- table.in.float, [31](#), [80](#), [81](#)
- table.title.top, [81](#)
- term.breakline, [85](#)
- tex.math.in.alt, [71](#), [72](#)
- texlive.version, [101](#)
- titleabbrev.in.toc, [67](#)
- toc.section.depth, [18](#), [68](#)
- ulink.footnotes, [43](#), [102](#)
- ulink.show, [43](#), [102](#)
- use.id.as.filename, [14](#), [105](#)
- variablelist.term.separator, [85](#), [86](#)
- xetex.font, [100](#), [112](#), [113](#)
- xref.hypermarkup, [85](#)
- xref.with.number.and.title, [103](#)

Processing Instructions, [117](#)

- bibtex, [117](#)
- dblatex angle, [120](#)
- dblatex autowidth, [122](#), [123](#)
- dblatex bgcolor, [121](#)
- dblatex citestyle, [118](#)
- dblatex colwidth, [123](#), [124](#)
- dblatex list-presentation, [124](#), [125](#)
- dblatex table-width, [121](#)
- latex, [32](#), [118](#), [119](#)
- texmath delimiters, [120](#)
