DocBook to LaTeX Publishing

User Manual

Ref A1 Ed. 02

COLLABORATORS

	TITLE :	*REFERENCE* :
	DocBook to LaTeX Publishing	Ref A1

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Benoît Guillon	21 June 2009	
REVIEWED BY	Andreas Hoenen	21 June 2009	

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 `set` support, and a section about how to extend the verbatim rendering.	B. Guillon

Contents

List of Figures

List of Examples

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 dblatex 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 **dblatex** 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 **dblatex**,
- write complex tables,
- write several bibliographies,
- reuse BibTeX bibliographies,
- use callouts on program listings or on images,
- create an index,
- 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 *0.2.11*.

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, Postcript 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.

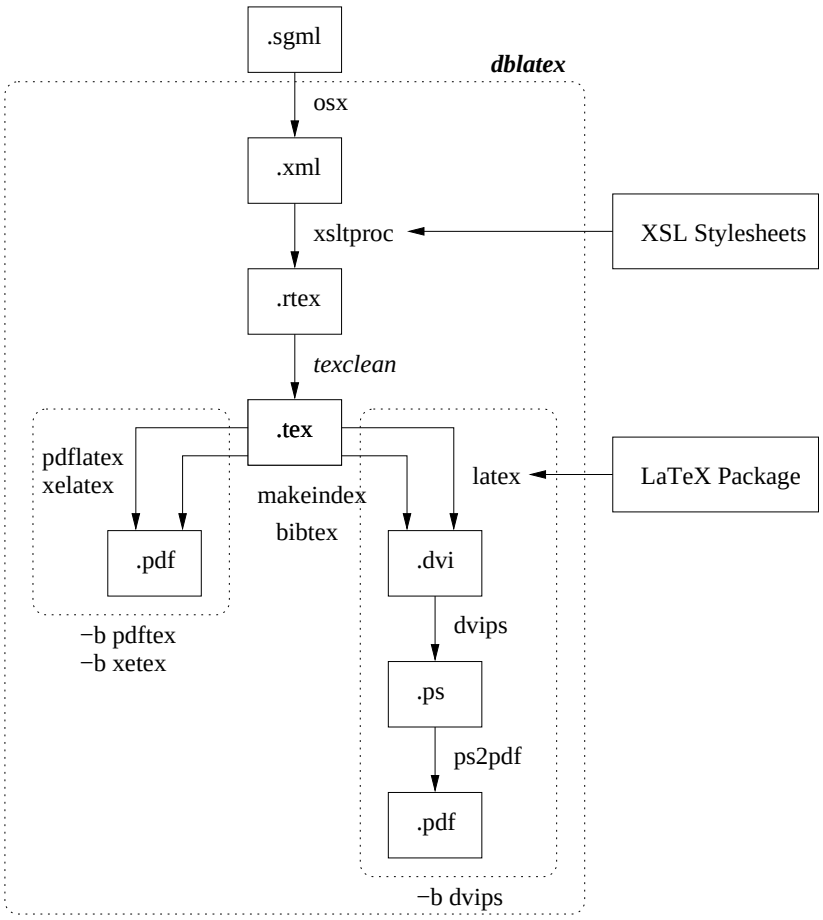

Figure 2.1: Transforming Process

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 4.4 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 package contains the following:

docs/
Contains the files of this document.

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.
- The XML DocBook DTD.
- A recent LaTeX distribution. The configure script checks that the needed latex packages are available.
- Python >= 2.4.

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 the tool

The steps to follow are the following:

1. Untar the ball. For a bzipped 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
   ```

3.3 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 >= 2.4.

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 **dblatex** by typing:

```
python C:\Python25\Scripts\dblatex 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 5.6.

-d, --debug
Debug mode: Keep the temporary directory in which dblatex actually works. Section 5.4.5 explains how you can use it.

-D, --dump
Dump the error stack when an error occurs (debug purpose).

-f *figure_format*, --fig-format=*figure_format*
Input figure format: *fig*, *eps*. Used when not deduced from figure file extension. See also Section 4.4.2.

-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 4.4.3.

-l *bst_path*, --bst-path=*bst_path*
Additional lookup path of the BibTeX styles. See Section 4.9.2.

-L *bib_path*, --bib-path=*bib_path*
Additional lookup path of the BibTeX databases. See Section 4.9.2.

-m *xslt*, --xslt=*xslt*
XSLT engine to use. The available engines are: xsltproc (default), 4xslt.

-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 outputed, 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. See Section 5.1.

-P *param=value* , --param=*param=value*
Set an XSL parameter from command line. See Section 5.2.

-r *script*, --texpost=*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. See Section 5.5.

-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 5.4.

-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 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 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:

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

4.2 Output Formatting Style

The output rendering done by **dblatex** 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 dblatex 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

```
dblatex -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 specification files describing the available styles. A specification file must have the extension .specs 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 specification file `db2latex.specs`.

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

```
$ dblatex
dblatex [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 5.6.

`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 directores that will contain all the specific data. We choose to put them under the default **dblatex** user configuration directory.

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

   Note that you could choose another configuration directory (see Section 5.6.2 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 5.4). Create also your XSL stylesheet if necessary.

3. Put these files under the appropriate directories:

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

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

   ```
   $ cat $HOME/.dblatex/mystyle.conf
   #
   # Dblatex config file for my new style.
   # Note that the directories are relative to mystyle.conf
   #
   TexInputs: mystyle/latex//
   TexStyle:  mytexstyle
   XslParam:  mystyle/param.xsl
   Options:   -f fig
   ```

5. That's it. Try to compile your document with your style, and check the output.

   ```
   $ dblatex -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 ouputs 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 Figure Inclusion

4.4.1 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 4.2 Figure inclusion

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

4.4.2 Converting on the fly

When it is needed dblatex 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 4.3 Figure conversion

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

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

4.4.3 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 4.4 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`.

4.5 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.

4.5.1 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 is a float like a table.

4.5.2 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

4.5.3 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

4.5.4 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"/>
    <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

4.5.5 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

4.5.6 Automatic column width

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

It is possible to change this sizing policy by playing with the *newtbl.autowidth* parameter. It 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, whatever 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.

4.5.7 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

4.5.8 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

4.5.9 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

4.5.10 Coloured tables

You can color all the table by setting its `bgcolor` 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>
```

4.6 Writing Mathematics

4.6.1 Writing LaTeX Mathematical Equations

4.6.1.1 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 4.5 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>
```

4.6.1.2 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).

Besides, the 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.

4.6.1.3 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.

4.6.1.4 Examples

The following examples show how to write the equations.

Example 4.6 Inlined Equation

The formula $C = \alpha + \beta Y^{\gamma} + \epsilon$ 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 4.7 Equation in a block

The following formula:

$$C = \alpha + \beta Y^{\gamma} + \epsilon$$

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 4.8 Equation in a float

The formula Equation 4.1 below:

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

EQUATION 4.1: 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 4.9 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>
```

4.6.2 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.

4.7 Extending the Verbatim Rendering

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, enphasis, 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 rendered as they are in a normal environment 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.embed"> ❹
     <xsl:with-param name="co-taging" 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 To enable the normal mode rendering within a verbatim environment, call the verbatim.embed template, and pass the mandatory parameters.

4.8 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 4.10 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.

4.9 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.

4.9.1 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 4.11 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>
```

4.9.2 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 4.12.

Example 4.12 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>
```

4.9.3 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* paramater 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.

4.10 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 perl script diffmk by Norman Walsh can do the work for you. It works fine, but it depends on several Perl modules.

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.

4.11 Locale Support

4.11.1 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.

4.11.2 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.

4.11.3 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 4.11.1 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`.

4.11.3.1 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.

4.11.4 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.

4.12 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`.

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 some customized stylesheets,
- using a customized LaTeX style package.
- using a LaTeX post process script.

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 5.6) 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 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.xsl`.

5.3.1 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"/>


<!-- 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:stylesheet>
```

5.3.2 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 5.1 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>
```

5.4 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 explicitely 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
dblatex --texstyle=mystyle file.xml

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

5.4.1 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 5.2 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`\«=\active
  \catcode`\»=\active
  \def«{u\og\ignorespaces}
  \def»{v\unskip\fg}
}
```

5.4.2 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`

5.4.3 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

5.4.4 DocBook interface

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

Command	Description
\DBKreference	mapped to `pubsnumber`
\DBKsite	mapped to `address`
\DBKcopyright	mapped to `copyright`
\DBKdate	mapped to `date`
\DBKedition	mapped to `edition`
\DBKpubdate	mapped to `pubdate`
\DBKsubtitle	mapped to `subtitle`
\DBKreleaseinfo	mapped to `releaseinfo`
\DBKlegalnotice	environment mapped to a `legalnotice`. The legal notices are all put into the \DBKlegalblock command. It is up to the latex stylesheet to decide where to put it in the document.
\DBKlegalblock	wrapper command for the \DBKlegalnotice environments, used by the latex stylesheet to decide where to put the legal notices in the document.
\DBKindexation	This command contains the `othercredit` information translated to latex by the XSL. This command must be placed where the othercredit shall appear in the document.
\DBKindtable	This environnement must be defined by the user to render the `othercredit` list. It can be displayed as a table, listitem, description list, or anything that suits your need.
\DBKinditem	This is an `othercredit` item.
\DBKrevtable	This environnement must be defined by the user to render the `revhistory` table. Untill now it is not really possible to customize it, since it must be a table with four columns, each column for a `revhistory` piece of information.
float example	This float is expected to be defined, and is mapped to `example`. It is not defined by default by the dbk_core 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 `equation`. It is not defined by default by the dbk_core package to allow the user to define its rendering (ruled or not, etc.)

5.4.5 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.

5.5 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.

5.5.1 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 Dblatex Configuration File

A master 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`*.

5.6.1 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.
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**

Here is the specification file used for this manual.

Example 5.3 User Manual Configuration File

```
#
# Configuration file for dblatex documentation (manual, release notes)
#
TexInputs: ../latex//
PdfInputs: ../latex/graphics
TexStyle:  docbook
XslParam:  manual.xsl
Options:   -b pdftex
```

5.6.2 Configuration Paths

By default **dblatex** tries to find the configuration files 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.

5.7 Customization Precedence

All the customization queries are translated to the corresponding command line options. Thus, using several customization methods can be unconsistent 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 explicitely 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.
- The options explicitely 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 occurence that is used.

Example 5.4 Customization Precedence

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

```
XslParam: file3.xsl
Options: -b pdftex
TexStyle: mystyle1
```

And now the command line:

```
dblatex -b dvips -p file1.xsl -p file2.xsl -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.xsl" overrides "-p file1.xsl" since it is defined after, and overrides "file3.xsl" 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 propably 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.
- Explicitely use the plain floatstyle. Note that using this explicit style does not allow to change the float title position anymore. The plain style always put 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 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 4.11.1 for more details.

Chapter 7

Thanks

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.

Thanks also to the current contributors: David Hedley (newtbl implementor), Andreas Hoenen (Debian maintainer), and Nicolas Pernetty (Windows port).

Special thanks to the KDE documentation team (especially Philip Rodrigues), Michael Smith (from the DocBook Project), and Kai Brommann, for their feedbacks, encouragements, and advice.

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.

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*.

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

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`*.

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.

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 `refentry`s, 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. See Section 4.5.6.

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.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 to apply. The default parameter value is [htbp] meaning that latex tries to place the table 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).

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

Set to 0 the formal tables are no more put in table floats. They are displayed with the longtable package, allowing to have formal tables covering several pages (which is not possible with floats). The limitation is that the title must necessarily be on the top of the table.

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`*).

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 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.11 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.12 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.

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 Section 4.9.3.

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 4.9.3.

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 4.9.3.

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 4.9.2 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

Defines how the BibTeX bibliographic entries are printed out. The available values are defined in Section 4.9.2.

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 Section 4.9.2.

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.13 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.14 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.

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.

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.

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 is was an URL

Synopsis

```
<xsl:param name="filename.as.url">1</xsl:param>
```

Description

Set to 1 the `filename`s are handled as URLs, with the same hyphenation rules. Set to 0 the `filename` hyphenation is forced for each character.

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.

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 monseq hyphenation policy

Synopsis

```
<xsl:param name="monoseq.hyphenation">1</xsl:param>
```

Description

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.

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.

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`.

seg.item.separator

seg.item.separator — Separator to use between several `segitems`

Synopsis

```
<xsl:param name="seg.item.separator">, </xsl:param>
```

Description

Defines the separator to use between several segitems.

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.

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 overriden 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.infoot" 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.infoot" 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>
```

A.15 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.

A.16 Chuncking

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 chuncking occurs, that is, when a set of books is published.

A.17 Pagination and General Styles

doc.alignment

doc.alignment — Specifies the text alignement 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.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 4.11.1 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.18 Font Families

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).

xetex.font

xetex.font — Specifies the fonts that XeTeX must use

Synopsis

```
      <xsl:param name="xetex.font">
  <xsl:text>\setmainfont{DejaVu Serif}
</xsl:text>
  <xsl:text>\setsansfont{DejaVu Sans}
</xsl:text>
  <xsl:text>\setmonofont{DejaVu Sans Mono}
</xsl:text>
</xsl:param>
```

Description

Font specification for XeTeX. Meaningful only when the xetex backend is used.

A.19 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.