DocBook to LaTeX Publishing

User Manual

Ref A0 Ed. 20

COLLABORATORS			
	TITLE : DocBook to LaTeX Publishing		*REFERENCE* : Ref A0
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Benoît Guillon	27 Jan 2007	
REVIEWED BY	Jean-Yves Le Ruyet	27 Jan 2007	
APPROVED BY		27 Jan 2007	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
1	20/01/03	First release of the package.	B. Guillon
2	30/04/03	Changes: • The script `configure` now checks the latex package dependencies, i.e. it checks that the packages used by the default DocBook latex style are available. • The tool can be heavily customized thanks to a specification file and/or new extra options (cf. Chapter 5).	B. Guillon
3	11/06/03	Changes: • The xsltml library is included in the package to have a strong and consistent support of the MathML 2.0 specification. • A large excerpt fo the MathML Test Suite 2.0 is now available to validate the MathML stylesheets.	B. Guillon

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
4	03/07/03	Changes: • Dutch language is supported by the default latex stylesheets. • The `subtitle` element is displayed on the cover page. • Tables can be displayed in landscape, through the `orient` attribute. In addition, the table text size can be specified to be smaller by using the `role` attribute. • Hyphenation is forced in tables, so that no words can cover several cells.	B. Guillon
5	03/05/04	Changes: see Section 2.4.16	B. Guillon
6	15/06/04	Changes: see Section 2.4.15	B. Guillon
7	15/07/05	Changes: see Section 2.4.14	B. Guillon
8	25/09/05	Changes: see Section 2.4.13	B. Guillon
9	20/10/05	Changes: see Section 2.4.12	B. Guillon
10	28/11/05	Changes: see Section 2.4.11	B. Guillon
11	28/04/06	Changes: see Section 2.4.10	B. Guillon
12	21/07/06	Changes: see Section 2.4.9	B. Guillon
13	27/09/06	Changes: see Section 2.4.8	B. Guillon
14	27/10/06	Changes: see Section 2.4.7	B. Guillon
15	24/11/06	Changes: see Section 2.4.6	B. Guillon
16	23/12/06	Changes: see Section 2.4.5	B. Guillon

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
17	21/01/07	Changes: see Section 2.4.4	B. Guillon
18	24/02/07	Changes: see Section 2.4.3	B. Guillon
19	25/04/07	Changes: see Section 2.4.2	B. Guillon
20	22/06/07	Changes: see Section 2.4.1	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 Why a DB2LaTeX clone?

Dblatex started as a DB2LaTeX clone. So, why this project? The purpose is a bit different 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 purpose 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 or pdflatex, to produce DVI, PostScript and PDF files,
- convert on the fly the figures included in the document,
- 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 equation 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 *0.2.6*.

2.4 Changes

2.4.1 Release 0.2.6

Bug fix release.

- It is possible to number and/or put the preface, dedication, and colophon sections in the TOC and bookmarks.
- Add a draft watermark when the document is in draft mode. Moreover, the draft mode can be deduced from the document `status` attribute.
- Some other minor improvements and bug fixes.

2.4.2 Release 0.2.5

Contains some deep changes even if the list of changes is small.

- Basic support for Chinese, Japanese, and Korean (CJK) languages. The CJK package must be installed, as well as the cyberbit fonts, to have this feature available.
- Support native UTF8 latex compilation, thanks to the `ucs` package.
- Preamble code refactored. The expected benefit is to make dblatex more customizable.
- Some other minor improvements and bug fixes.

2.4.3 Release 0.2.4

A bug fix release.

- A step towards DocBook 5 support by reusing the DocBook Project namespace stripping templates.
- More consistent console output (use of logging python module).
- Some bug fixes, including:
  - Fix PostScript output bug introduced in 0.2.3.
  - Use of subprocess to handle correctly the latex compilation.
  - A general parser/encoding mechanism is added, allowing to encode `programlisting` characters to Latin1.

2.4.4 Release 0.2.3

This is mainly a bug fix release.

- Experimental `annotation` support (DocBook 5) enabled when the *`annotation.support`* paramater is set to 1. The annotation support is only for PDF output.
- Some `setup.py` improvements and bug fixes, thanks to Max Horn's feedbacks (Fink packager).
- Some minor improvements (*`doc.toc.show`* and *`draft.mode`* parameters added).
- A number of bug fixes.

2.4.5 Release 0.2.2

- BibTeX support added. See Section 4.8.2 for more details.
- A better bibliography support, and natbib citation styles can be used (see Section 4.8.3).
- Some table improvements:
  - Formal tables accross several pages are possible when setting the *table.in.float* parameter to 0. The limitation is that the caption must be on the top of the table.
  - Automatic table width allowed when setting the *newtbl.autowidth* parameter to default or all.
  - Footnotes in tables are possible.
  - Basic programlisting/screen support in tables.
- A number of bug fixes.

2.4.6 Release 0.2.1

- Better unicode support. Now the XML output is encoded in UTF-8, and Python uses the codecs to decode to ISO-Latin1 and replace the unsupported characters to some latex equivalent.
- Several XSLT processors can be used. The first port is for 4suite because it is fully written in Python and is quite a good tool. The processors are loaded as plugins, so that it can be easily extended to any other XSLT processor.

  The processor to use is specified from the command line with the -m *xslt* option, where *xslt* is the name of the plugin to load (actually the name of the dynamically loaded Python module).
- The XSL code is more conformant. It has been checked by using the 4suite XSLT processor as an alternative to **xsltproc**.
- Possibility to have some configuration files stored under $HOME/.dblatex or under /etc/dblatex for system-wide configurations. Some extra paths can be specified by using the DBLATEX_CONFIG_FILES environment variable.
- The remarks and comments are rendered as PDF text annotations when pdflatex is used. Otherwise, the comments are suppressed.
- Some other minor improvements:
  - The parameters *pdf.annot.options*, *latex.class.book*, and *latex.class.article* are added.
  - An hexadecimal color like <?dblatex bgcolor="#cceeff"?> is supported, and a named color like <?dblatex bgcolor="blue"?> is supported too (in the previous release named colors had to be enclosed in curly braces "{}").
  - Some cleaner locale handling is provided, and new latex commands are given to allow the user to customize the babel setup.
  - The cross-references now use key() instead of id(). It prevents from some bugs and makes writing a document easier, especially in modular parts.
- Some bug fixes.

2.4.7 Release 0.2

- Better **osx** integration. The SDATA entities are translated to the equivalent Unicode characters.
- Better Windows compatibility thanks to Nicolas Pernetty for his patches and feedbacks.
- Better table support:
  - Putting some verbatim text (literallayout, address, synopsis, classsynopsis) in tables now works.
  - Nesting some informaltables is possible.
  - Basic entrytbl support.
  - Better valign attribute support.

  - The columns, rows and entries can be coloured by using some special Processing Instructions like `<?dblatex bgcolor="..."?>`.
  - The table `floatstyle` attribute can be used to specify the float placement rules (like "[htbp]").
- A few `imageobject` improvements:
  - Like for the official DocBook XSL stylesheets, you can use the `role` attribute in `imageobject` to specify the image to use by **dblatex**. Set role to 'dblatex' to select the image used by **dblatex**.
  - Alternative `imageobject`s can be put in a `mediaobjectco` (DocBook 5).
- Some cleanups and a number of bug fixes.

2.4.8 Release 0.2pre

Major release. All the code to transform SGML, call the XSLT, convert the figures and finally compile with LaTeX has been re-written from scratch in Python, removing Perl and GNU make dependencies. The LaTeX compilation relies on a subset of the Rubber package. The new implementation is more robust, more consistent, and gives the possibility to integrates new features. Some other small improvements are included too:

- Some list attributes like `continuation`, `numeration` and `spacing` are now supported.
- The *`filename.as.url`* parameter is added to avoid forced hyphenation with spurious '-' characters.
- Some bug fixes.

2.4.9 Release 0.1.10

Bug fix release.

- Images can now have their default dimension limited to a specified maximum dimension (can be lower than the page boundaries).
- the following parameter is added:

  glossterm.auto.link
  Makes glossterms link to their glossary definition.
- A number of bug fixes.

2.4.10 Release 0.1.9

Few changes.

- The newtbl implementation now handles some tricky row spanning cells. Moreover it becomes the default table implementation used.
- Equations without title are now latex equations (not formula in a float).
- Xref to `varlistentry` or `term` is possible.
- the following parameters are added:

  make.year.ranges, make.single.year.ranges
  Change the rendering of year ranges in a copyright.
- A number of bug fixes.

2.4.11 Release 0.1.8

This is mainly a bug fix release.

- Better `programlisting` and `screen` support: inlinegraphic[@format='linespecific'] is handled.
- Better newtbl support: the case colwidth="1in+5*" is now correctly handled.
- GIF images are converted on the fly to PDF.
- `Bibliolist` support.
- Minor improvements:
  - DBLaTeX does not convert images when output is latex only.
  - `Emphasis` with role="underline" is supported.
  - `Trademark` with class="service" is supported.
  - `Xref` to `refnamediv` is now possible.
  - Automatic `biblioentry` abbreviation used if `abbrev` and @id reference are not defined.
- the following parameters are added:

  titleabbrev.in.toc
  When set to 1 the titleabbrev content is put in the TOC instead of the title. Set to 1 by default.

  set.book.num
  When the document root element is a `set` this parameter can be used to select the book to print. Set to 1 by default.

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

  qandaset.defaultlabel
  It defines the default label to use in a `qandadet` when the defaultlabel attribute is not specified. Set to "number" by default.

  imagedata.file.check
  Set to 1, it checks if the referenced image file exists. If not, the mediaobject alternative (textobject) is used. Set to 1 by default.

  doc.alignment
  It defines the text alignment for the whole document. The valid values are: "left", "center", "right", "justify". By default the parameter is empty, which is equivalent to "justify".
- A number of bug fixes.

2.4.12 Release 0.1.7

- Callouts are now supported:
  - Embedded callout markups `co`s are supported.
  - The `coref` markups are supported.
  - Callouts markups defined with `area`s in a `programlistingco` or `screenco` are supported.
  - Callouts on external text files (referenced by `textdata` or `imagedata` elements) are supported.
  - `Mediaobjectco`s is supported.
  - The `calloutlist`s are rendered as description lists where the terms are the callout markups.
  - The links between the callout markups (defined via `area`s or `co`s) and the `calloutlist` items (linkends attributes) are handled properly.

- `Programlisting` and `screen` improved: external text files referenced via `textdata` or `imagedata` are now supported.
- An `abstract` in an article is now printed.
- The `legalnotices` are now printed in the native docbook style.
- Better `xref` support. You can now make a cross-reference to an `itemizedlist` with title, and to a `refentry`.
- Hyphenation is forced for text using a typewriter font, and the font is smaller.
- Running **dblatex** on a root element different from a book or article does not fail anymore, except for `set`. The root element is now wrapped into a `book` or an `article`.
- Minor improvements:
  - A DBLaTeX logo can be put on the cover page.
  - The PDF information section can tell that the creator of the document is dblatex.
- the following parameters are added:

  co.linkends.show
  Next to a callout markup the links to the corresponding calloutlist items are shown if the parameter is set to 1. Set by default to 1.

  callout.markup.circled
  The callouts referenced in the callout list have the same rendering than the markups in the listing (or graphic), that is, white numbers in black circles. Set to 0 the references are simple numbers. Set to 1 by default.

  callout.linkends.hot
  The callouts referenced in the 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. Set to 1 by default.

  term.breakline
  Set to 1, the item following a term in a variable list is put on the next line. Set to 0 by default.

  doc.pdfcreator.show
  Set to 1, the creator field of the PDF information section says that dblatex is the creator. Set to 1 by default.

  doc.publisher.show
  Set to 1, the dblatex logo is printed on the cover page of the native docbook style. Set to 0 by default.

  literal.lines.showall
  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.
- Some bug fixes.

2.4.13 Release 0.1.6

- Better `figure` and `informalfigure` rendering:
  - Caption and title are printed separately, in a consistent way.
  - Default image scaling is possible.
- Better `programlisting` and `screen` rendering:
  - All the attributes are supported
  - A default verbatim layout is provided. The text is put in a framed box with a yellow background color.
  - Long lines are wrapped.
- Minor improvements:
  - A breakline is forced after a `term` when it is immediately followed by a list.

- the following parameters are added:

  imagedata.default.scale
  It defines the default scaling rule to apply on every `imagedata` that contains no scaling attribute.
  By default the parameter is set to 'pagebound', that is the images keep their natural size up to the page boundaries.

  figure.title.top
  Set to 1 it specifies to put the title above the image. By default it is set to 0 (title below).

  > **Note**
  > This parameter has no effect if an explicit float style is used for the figures (e.g. ruled style), since the title position is then fixed by the chosen style.

  mediaobject.caption.style
  Font style applied to the caption text. Default is slanted.

  literal.width.ignore
  Set to 1 the programlisting and screen width attribute is ignored.

  literal.layout.options
  Overwrite the default verbatim layout options.

  seg.item.separator
  Defines the separator to use between several `segitem`s.

- Some bug fixes.

2.4.14 Release 0.1.5

- **dblatex** supports the new option `-T target_style`. It specifies which latex style to use for formatting the output. See Section 4.2 for more details.
- The configure script can select the default latex style to use with the option `--target`. Example:

```
./configure --prefix=/where/to/install --target=db2latex
```

- The use of **make** instead of **gmake** is now detected by configure.
- Any document language should be well supported, since babel is now included for the related language.
- New table support, completely re-written by David Hedley. It is very good and no Perl parsing is needed. One can use this new XSL table code by setting the parameter newtbl.use=1.
- The following XSL parameters are added:

  latex.babel.use
  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. Default is 1.

  latex.babel.language
  Empty by default, this parameter forces the use of the specified babel language whatever the document language is.

  newtbl.use
  Set to 1, use the David Hedley table support. By default it is set to 0.

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

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

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

- XML source files with any extension are correctly handled. Previously one needed to give XML files with extension .xml.
- Better `footnote` support: it can be used in section titles and in `term`s.
- Some latex rendering aspects are removed from the XSL stylesheets (they should never have been in these stylesheets): \parindent value, \parskip value, \thispagestyle{fancy} for pages containing chapters.
- Bug fixes.

2.4.15 Release 0.1.4.1

- Significant `imagedata` improvement: almost all the attributes (align, valign, depth, width, scale, scalefit, contentdepth, contentwidth) are correctly managed. However percentage used for both contentdepth and contentwidth is not managed (only contentwidth percentage is then applied).
- Dblatex tries to automatically detect the image file formats of the included graphics, and convert them if necessary (and if possible) to be compatible with the TeX backend driver. It is usefull when several image formats are used within the same document, in which case the `-f fig_format` cannot be used.
- The dblatex option `-P param=value` is added. One can then set XSL parameter values directly from the command line. This is an alternative to the `-p custom.xsl` option.
- The `align` attribute is now managed for table cells spanned on several columns (i.e. row entries with `nameend` or `spanname` attributes).

2.4.16 Release 0.1.4

- Deep code cleanup.
- Better table support
  - `Multicolumn` support (use of the attributes `namest`, `nameend`, `spanname`).
  - Better `frame`, `rowsep`, `colsep` attributes inheritance.
- Better bibliography support
  - `Bibliography` can be nested under any section.
  - `Biblioset` support.
  - Basic `bibliomixed` support.
- `Indexterm` `sortas` and `class` attributes support added.
- `Imagedata` `width`, `depth`, `scale` attributes support improved. In previous releases, `scale` was used instead of `width`. Now, you should use `width` or `scale` properly.
- `Programlisting` `linenumbering` attribute support added.
- Basic `glossary` support added.

- Better `reference` support. `Refnamediv` title is no more hard-coded (use of $refnamediv.title if not empty, or name automatically generated according to the lang).
- `Qandaset` improved. `Qandadiv` can be nested under any section.
- Better `xref` support. Now `xreflabel` and `endterm` work.
- The latex hyperref package is now automatically included in the dbk_core package. A customized LaTeX style package shouldn't include hyperref anymore.
- `Link` now works.
- `Trademark` `class` attribute managed (except class='service').
- A `keyword` is not displayed but is inserted in the index entries.
- Some bug fixes.

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.

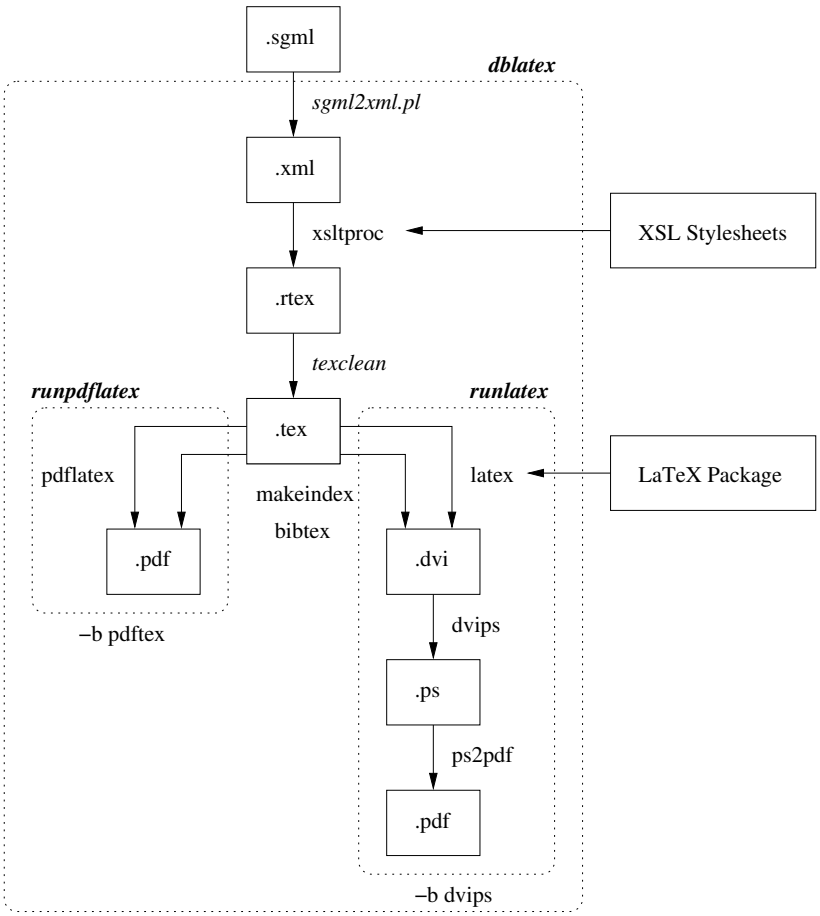

Figure 2.1: Transforming Process

2.5.1 Backend Drivers

The main script allows to use two LaTeX backend drivers:

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

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 doesn't 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 4.3 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 Systems

3.2.1 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.

3.2.2 Installation

3.2.2.1 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.

3.2.2.2 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

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.

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 dblatex

Name

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`*, *`dvips`* (default). 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 mode: Keep the temporary directory in which dblatex actually works. Section 5.4.5 explains how you can use it.

`-f` *`figure_format`*, `--fig-format=`*`figure_format`*
Input figure format: *`fig`*, *`eps`*. Used when not deduced from figure file extension. See also Section 4.3.2.

`-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 4.3.3.

-l *bst_path*, --bst-path=*bst_path*
Additional lookup path of the BibTeX styles. See Section 4.8.2.

-L *bib_path*, --bib-path=*bib_path*
Additional lookup path of the BibTeX databases. See Section 4.8.2.

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

-o *output*, --output=*output*
Output filename. When not used, the input filename is used, with the suffix of the output format.

-p *xsl_user*, --xsl-user=*xsl_user*
An XSL user stylesheet to use. 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
```

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 Figure Inclusion

4.3.1 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>
```

4.3.2 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
```

4.3.3 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`.

4.4 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.

4.4.1 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.

4.4.2 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

4.4.3 Tables with mixed colspec

A table can have `colspec` elements containing `colwidth` attribute mixed with `colspec` elements without `colwidth`. The following XML source:

```
<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>
```

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

4.4.4 Tables with proportional and fixed colwidth

Proportional column widths are supported. The following XML source:

```
<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>
```

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

4.4.5 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

4.4.6 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.

4.4.7 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

4.4.8 Landscape tables

A table can be displayed in a lanscape format by using the `orient` attribute. The following XML source is an 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

4.4.9 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

4.4.10 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 expression must be 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 bgcolor="{yellow}">
<tgroup cols="4" colsep="1" rowsep="1" align="left">
```

```
  <colspec colname="c1" colwidth="2cm"/>
  <colspec align="left" colwidth="2.5cm">
  <?dblatex bgcolor="{green}"?>
  </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>
```

4.5 Writing LaTeX Mathematical Equations

4.5.1 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>
```

4.5.2 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.

4.5.3 Compatibility

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

4.5.4 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 SGML source is:

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

```
<para>The following formula:
  <informalequation>
    <alt>\[C = \alpha + \beta Y^{\gamma} + \epsilon\]</alt>
    <graphic fileref="figures/eq2"/>
  </informalequation>
is displayed in a separate block. The SGML 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 SGML 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/eq3"/>
  </equation>
is displayed in a block with a title. Its SGML 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 SGML 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/eq3"/>
  </equation>
is displayed as a latex equation with its own equation numbering.
Its SGML source is:</para>
```

4.6 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 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.

4.8 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.

4.8.1 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>
```

4.8.2 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>
```

4.8.3 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.

4.9 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.

4.10 Locale Support

4.10.1 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.

4.10.2 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.

4.10.3 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 4.10.1 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.

4.10.3.1 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.

4.10.4 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.

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 XSL Parameters

The PDF rendering can be customised by using some XSL configuration parameters. The available configuration parameters are the following:

Parameter	Role	Default Value
annotation.support	Set to 1 the experimental DocBook 5 `annotation` support is enabled.	0
callout.markup.circled	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.	1
callout.linkends.hot	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.	1
calloutlist.style	Defines how the callout list items are displayed. The value must be some valid enumitem description list options.	"leftmargin=1cm,style=samel
citation.default.style	Default natbib citation style to apply when natbib is used. See Section 4.8.3.	Empty
citation.natbib.options	Options to pass to the natbib package when it is loaded. See also Section 4.8.3.	Empty
citation.natbib.use	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 4.8.3.	0

Parameter	Role	Default Value
co.linkends.show	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.	1
colophon.tocdepth	Same than *preface.tocdepth* for colophon sections.	0
dedication.tocdepth	Same than *preface.tocdepth* for dedication sections.	0
doc.alignment	Defines the text alignment for the whole document. The valid values are: "left", "center", "right", "justify". An empty string is equivalent to "justify".	Empty
doc.collab.show	Show the collaborators (authors, contributors) defined in the document information block.	1
doc.lot.show	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.	"figure,table"
doc.pdfcreator.show	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.	1
doc.publisher.show	Print the dblatex logo on the cover page for the native docbook style if the parameter is set to 1.	0
doc.section.depth	Depth of the section numbering. Used to set the latex `secnumdepth` counter.	5
doc.toc.show	Print the table of contents when set to 1.	1
draft.mode	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'.	maybe
draft.watermark	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'.	1
figure.caution	Figure to use to render a `caution` block. This parameter is added to allow new latex styles to use their own figures in admonitions.	"warning"
figure.default.position	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).	[htbp]
figure.important	Figure to use to render a `important` block. This parameter is added to allow new latex styles to use their own figures in admonitions.	"warning"
figure.note	Figure to use to render a `note` block. This parameter is added to allow new latex styles to use their own figures in admonitions.	Empty
figure.tip	Figure to use to render a `tip` block. This parameter is added to allow new latex styles to use their own figures in admonitions.	Empty
figure.title.top	Set to 1 the `figure` float title position is above the image. Set to 0 the title is under the image.	0
figure.warning	Figure to use to render a `warning` block. This parameter is added to allow new latex styles to use their own figures in admonitions.	"warning"

Parameter	Role	Default Value
filename.as.url	Set to 1 the `filenames` are handled as URLs, with the same hyphenation rules. Set to 0 the `filename` hyphenation is forced for each character.	1
glossterm.auto.link	When set to 1, the glossterms in the document are linked to their definition in the glossary.	0
imagedata.boxed	If set to 1, put the images into a framed box.	0
imagedata.default.scale	cf. Section 5.1.1	pagebound
imageobjectco.hide	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).	0
latex.babel.use	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.	1
latex.babel.language	This parameter forces the use of the specified babel language whatever the document language is.	Empty
latex.biblio.output	Defines how the BibTeX bibliographic entries are printed out. The available values are defined in Section 4.8.2.	all
latex.biblio.style	Defines the default BibTeX style to apply. Meaningful when not empty, only for the used bibtex databases. See Section 4.8.2.	Empty
latex.class.article	This parameter sets the document class to use for `article` documents.	article
latex.class.book	This parameter sets the document class to use for `book` documents.	report
latex.class.options	Options passed to the \documentclass command.	Empty
latex.encoding	Encoding of the latex document to produce. The supported values are: "latin1" and "utf8". See Section 4.10.1 for more details about how to use it.	"latin1"
latex.hyperparam	See Section 5.1.2	empty
latex.unicode.use	Set to 1 the passivetex unicode support is included, allowing to handle a wider range of Unicode characters (like cyrillic).	0
literal.layout.options	Overwrite the default options passed to the \lstset command.	Empty
literal.lines.showall	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.	1
literal.width.ignore	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.	0
make.year.ranges	If non-zero, copyright years will be collated into ranges. Parameter taken from the DocBook XSL stylesheets.	0
make.single.year.ranges	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.	0
mediaobject.caption.style	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.	\slshape
newtbl.autowidth	Defines if the table column widths must be automatically sized by latex. See Section 4.4.6.	Empty

Parameter	Role	Default Value
newtbl.default.colsep	Set to 1, print the column separators when no `colspec` attribute is specified.	1
newtbl.default.rowsep	Set to 1, print the row separators when no `rowspec` attribute is specified.	1
newtbl.format.thead	LaTeX formatting for head table cells.	\bfseries%
newtbl.format.tbody	LaTeX formatting for body table cells.	Empty
newtbl.format.tfoot	LaTeX formatting for foot table cells.	Empty
newtbl.use.hhline	Set to 1, use the `hhline` package to draw the table row separators instead of cline. Using `hhline` seems more suited for colored tables.	0
pdf.annot.options	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`.	Empty
preface.tocdepth	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 '*').	0
qandaset.defaultlabel	Defines the default label to use in a `qandadet` when the defaultlabel attribute is not specified.	"number"
seg.item.separator	Defines the separator to use between several `segitems`.	", "
set.book.num	When the document root element is a `set` this parameter can be used to select the book to print, because **dblatex** can output only one book from the set.	1
table.default.position	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).	[htbp]
table.in.float	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.	1
table.title.top	Set to 1 the `table` float title position is above the table. Set to 0 the title is under the table.	0
term.breakline	Set to 1 the item following a term in a variable list is put on the next line.	0
titleabbrev.in.toc	Set to 1 the titleabbrev content is put in the TOC instead of the title.	1
toc.section.depth	Depth of the TOC. Used to set the latex `tocdepth` counter.	5

5.1.1 imagedata.default.scale

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.

5.1.2 latex.hyperparam

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 5.1 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>
```

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'>

<!-- 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>

</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 5.2 Overriding templates

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

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

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

</xsl:stylesheet>
```

5.4 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 5.3 Reused LaTeX style

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

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

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

%% Redefine some french settings
\babelsetup{fr}{%
  \catcode`\«=\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`

Command	Description
\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.

Keyword	Value	Corresponding option	Description
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 5.4 User Manual Configuration File

```
#
# Specification file example
#
TexInputs: ../latex//
PdfInputs: ../latex/graphics
TexStyle:  docbook
XslParam:  myparam.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 5.5 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 Section 5.1.2 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.

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.