The xxeconvert command-line utility

Hussein Shafie, XMLmind Software <xmleditor-support@xmlmind.com>

June 30, 2026

Abstract

This document is the reference manual of the **xxeconvert** command-line utility. This tool is based on the XMLmind XML Editor desktop application, but without a graphical user interface (**GUI**). It can be used to convert XML documents to other formats (HTML, PDF, RTF, etc) from within a script or a makefile.

This utility, like all the other command-line utilities, is found in *XXE_install_dir*/bin/.

Table of Contents

1. Why use the **xxeconvert** command-line utility? 1
2. Using **xxeconvert** 1
    2.1. The online documentation of **xxeconvert** 1
    2.2. Using **xxeconvert** to convert documents 2
A. About **xxeconvert** and user authentication 7

1. Why use the xxeconvert command-line utility?

This tool is based on the XMLmind XML Editor (**XXE** for short) desktop application, but without a graphical user interface (**GUI**). It can be used to convert XML documents to other formats (HTML, PDF, RTF, etc) from within a script or a makefile.

Because **xxeconvert** shares a lot of code with **XXE**, this command-line utility will automatically make use of most the installed add-ons (e.g. "Apache FOP 1.x XSL-FO processor plug-in", "Apache Batik image toolkit plug-in", etc) and also of some of the last specified user preferences (e.g. conditional processing profiles).

This utility, like all the other command-line utilities, is found in *XXE_install_dir*/bin/.

Mac users

If you install **XXE** on the Mac using the recommended `.dmg` distribution, you'll not find *XXE_install_dir*/bin/. The **xxeconvert** command-line utility, like all the other command-line utilities, is found in `XMLmind.app/Contents/Resources/xxe/bin/`.

If you plan to use **xxeconvert** a lot, may be you'll prefer to install the `.zip` distribution on your Mac rather the recommended `.dmg` distribution. Installing the `.zip` distribution on your Mac requires also installing a Java 1.8+ runtime.

2. Using xxeconvert

2.1. The online documentation of xxeconvert

xxeconvert is fully auto-documented:

- Run "`xxeconvert`" or "`xxeconvert -?`" to list its command-line options:

```
$ xxeconvert

usage: xxeconvert [Advanced option]* [Special option]*
   [-t XSLT_stylesheet_file_or_URL]?
   [-r|-ru resource_name resource_value]*
   [-p XSLT_stylesheet_param_name XSLT_stylesheet_param_value]*
   process_command_name doc_file_or_URL
   [-s|-u process_command_arg]*
...
```

- Run "`xxeconvert` *`whatever`*" to list all available process commands:

```
$ xxeconvert foo

xxeconvert: *** ERROR *** 'foo', not a documented process command.
asm.toEclipseHelp
asm.toEpub
asm.toHTML
asm.toHTML1
asm.toHTMLHelp
asm.toPSFile
asm.toRTF
asm.toWebHelp
db5.toEclipseHelp
db5.toEpub
...
```

- Run "`xxeconvert dita.toEPUB`" to print how to use the "`dita.toEPUB`" process command:

```
$ xxeconvert dita.toEPUB

xxeconvert [options] dita.toEPUB map_or_topic_file_or_URL \
  -u epub_file_or_URL

  Convert DITA map or topic map_or_topic_file_or_URL to
  EPUB file epub_file_or_URL.

  Example:
  xxeconvert convert dita.toEPUB doc.ditamap -u out/doc.epub
```

2.2. Using xxeconvert to convert documents

Usage:

```
xxeconvert [Advanced Option]* [Special Option]*
     [-t XSLT_stylesheet_file_or_URL]?
     [-r|-ru resource_name resource_value]*
```

```
[-p|-pu XSLT_stylesheet_param_name XSLT_stylesheet_param_value]*
process_command_name doc_file_or_URL
[-s|-u process_command_arg]*
```

Converts XML document *`doc_file_or_URL`* using process command in *XMLmind XML Editor - Commands* called *`process_command_name`*, found in any of the **XXE** configuration files scanned during the startup of **xxeconvert** (see XMLmind XML Editor - Configuration and Deployment).

Common options:

`-t` *`XSLT_stylesheet_file_or_URL`*

Use this alternate XSLT style sheet instead of the one specified in the first `transform` child element of the process command.

If specified process command has no `transform` child element but has `subProcess` child elements, these sub-processes are searched recursively for a `transform` child element.

`-r|-ru` *`resource_name resource_value`*

Copy specified resource rather than the one specified in the `<copyProcessResources name="resource_name">` child element of the process command.

`-ru` is useful when the resource value is a relative filename that needs to be converted to an absolute "`file:`" URL.

`-p|-pu` *`XSLT_stylesheet_param_name XSLT_stylesheet_param_value`*

Add/replace corresponding XSLT style sheet parameter in the first `transform` child element of the process command.

`-pu` is useful when the parameter value is a relative filename that needs to be converted to an absolute "`file:`" URL.

If specified process command has no `transform` child element but has `subProcess` child elements, these sub-processes are searched recursively for a `transform` child element.

`-p|-pu` *`param_group_name/param_name param_value`*

Add/replace specified parameter to parameter group called *`param_group_name`*.

`-pu` is useful when the parameter value is a relative filename that needs to be converted to an absolute "`file:`" URL.

DocBook example: `-p docb.toRTF.XFCParameters/docx.variant 15`[1], where parameter group `docb.toRTF.XFCParameters` is declared as follows in *`XXE_install_dir`*`/addon/config/docbook/xslMenu.incl`:

```
<command name="docb.toRTF">
  <process>
    ...
    <processFO processor="XFC" file="__doc.fo" to="__doc.%0">
      <parameter name="outputFormat">%0</parameter>
      <parameter name="outputEncoding">%1</parameter>
      <parameter name="imageResolution">120</parameter>
```

[1]This XMLmind XSL-FO Converter option marks generated DOCX file as being compatible with MS-Word 2013.

```
      <parameter name="prescaleImages">false</parameter>
      <parameterGroup name="docb.toRTF.XFCParameters" />
    </processFO>
    ...
  </process>
</command>
```

`-s|-u` *`process_command_arg`*

Pass these arguments to the process command as the values of process variables `%0`, `%1`, ..., `%9`.

If `-s` (String) is specified, the argument is passed as is.

If `-u` (URL) is specified, the argument, a file or directory name, is first converted to an URL.

Special options:

`-profile -|`*`file_or_URL`*

Apply this conditional processing (profiling) file to the document being converted.

This file may be
- a "`.ditaval`" file optionally ending with "`?media=screen`" (default media) or "`?media=print`"
- OR a "`.profiles`" file ending with a fragment (e.g. "`#my_profile`") specifying the ID of the selected profile.

Specify "`-`" to suppress profiling.

`-fop`

Make sure to use Apache FOP to generate PDF (*just for this document conversion*; sometimes useful because Apache FOP has MathML support).

`-d`

Sets the `debug` attribute of the process command to value `true` (no matter what has been specified in the `process` element).

This prevents the process command from deleting its work directory (`/tmp/xxe`*`NNNN`*`/`) at the end of the processing.

`-v|-vv|-vvv`

Turn verbosity on. The more Vs, the more verbose.

Advanced options:

`-auth` *`credentials`*

This option can be used to specify authentication credentials for a given server. This allows to connect to the specified server without interactively asking the user to enter a username and a password.

String *`credentials`* consists in 6 fields: *`host`*, *`port`*, *`prompt`*, *`scheme`*, *`username`*, *`password`*, in that order, separated by a newline character (`'\n'`). Fields *`host`*, *`port`*, *`prompt`*, *`scheme`* can be left empty, which means: match any. The UTF-8 bytes of the string are then encoded in base-64.

Command-line utility *`XXE_install_dir`*`/bin/authvalue` allows to generate such encoded string. Example: encode string "`\n\nDocument Store\n\nvictoria\nsecret`":

```
/opt/xxe/bin$ authvalue victoria secret - "Document Store"
CgpEb2N1bWVudCBTdG9yZQoKanZpY3RvcmlhCnNlY3JldA==

/opt/xxe/bin$ xxeconvert convert -auth CgpEb2N1bWVudCBTdG9yZQoKanZpY3RvcmlhCnNlY3JldA== \
docb.toHTML http://www.acme.com/docstore/push_up.xml -u docs/
```

Command-line utility **authvalue** is auto-documented. Type **`authvalue`**, then press Enter to display a short documentation explaining how to use this utility.

See also Appendix A, *About **xxeconvert** and user authentication* [7] for an alternative way to let the user of **xxeconvert** authenticate to a server.

`-putpref` *`key value`*

Adds or replace preference specified by *`key`*/*`value`* to the set of the user's preferences.

Note that the `-putpref`, `-putprefs`, `-delprefs` options change the user's preferences only during this invocation of **xxeconvert**. **xxeconvert** reads, but never writes the contents of file *`XXE_user_preferences_dir`*`/preferences.properties`.

`-putprefs` *`property_file_or_URL`*

Similar to `-putpref` except that several *`key`*/*`value`* pairs may be read from specified property file.

`-delpref` *`key`*

Removes preference specified by *`key`* from the set of the user's preferences.

Examples:

Example 1. Convert a DocBook document to multi-page HTML

1. First run "`xxeconvert convert`" to list all availables process commands:

```
$ xxeconvert dontknow

...
db5.toEclipseHelp
db5.toEpub
...
docb.toHTML
...
```

2. Process command `docb.toHTML`[2] looks good. Run "`xxeconvert docb.toHTML`" to print its online help:

```
$ xxeconvert docb.toHTML

xxeconvert [options] docb.toHTML docbook_file_or_URL \
  -u output_dir_filename_or_URL

  Convert DocBook document docbook_file_or_URL to multi-page HTML.
```

[2]The `docb.toHTML` process command is defined in *`XXE_install_dir`*`/addon/config/docbook/xslMenu.incl`.

```
  Create the HTML pages in directory output_dir_filename_or_URL.

  Example:
  xxeconvert convert docb.toHTML doc.xml -u out/
```

3. Use process command `docb.toHTML` to convert DocBook document `help.xml` to multi-page HTML created in directory `docs/help/`.

```
$ xxeconvert -p toc.section.depth 4 -p chunk.section.depth 2 \
    docb.toHTML help.xml \
    -u docs/help
```

> **IMPORTANT:** If file `help.xml` is a DocBook 5.0 document and not a DocBook 4 document, please make sure to invoke command **db5.toHTML** and not command **docb.toHTML**. If you make this mistake, you'll get the following error:
>
> ```
> xxeconvert: *** error: did not find command 'docb.toHTML'¬
> in configuration 'DocBook 5.0'
> ```

Example 2. Convert a DocBook document to PDF

1. First run "`xxeconvert convert`" to list all availables process commands:

```
$ xxeconvert whichone

...
db5.toEclipseHelp
db5.toEpub
...
docb.toPSFile
...
```

2. Process command `docb.toPSFile`[3] looks good. Run "`xxeconvert convert docb.toPSFile`" to print its online help:

```
$ xxeconvert docb.toPSFile

xxeconvert convert [options] docb.toPSFile docbook_file_or_URL \
  -s pdf -s "|pdf" -u pdf_file_or_URL

  Convert DocBook document docbook_file_or_URL to PDF file pdf_file_or_URL.

  Example:
  xxeconvert convert docb.toPSFile doc.xml -s pdf -s "|pdf" -u out/doc.pdf
...
```

[3] The `docb.toPSFile` process command is defined in *`XXE_install_dir`*`/addon/config/docbook/xslMenu.incl`.

3. Use process command `docb.toPSFile` to convert DocBook document `doc.xml` to `commands.pdf`.

```
$ xxeconvert -t fo_docbook.xsl \
    -p toc.section.depth 4 -p callout.graphics 0 -p variablelist.as.blocks 1 \
    docb.toPSFile doc.xml \
    -s pdf -s "|pdf" -u docs/commands/commands.pdf
```

   Notice that an alternate, customized, XSLT style sheet, `fo_docbook.xsl`, is used instead of the stock `docbook.xsl`.

   IMPORTANT: If file `help.xml` is a DocBook 5.0 document and not a DocBook 4 document, please make sure to invoke command **db5.toPSFile** and not command **docb.toPSFile**. If you make this mistake, you'll get the following error:

```
xxeconvert: *** error: did not find command 'docb.toPSFile'¬
in configuration 'DocBook 5.0'
```

A. About xxeconvert and user authentication

Most remote document repositories (e.g. WebDAV servers) will require the user of **xxeconvert** to authenticate herself/himself.

xxeconvert being a command-line tool designed to be used in makefiles, batch files, shell scripts, etc, authentication by the means of an interactive dialog with the user is not a solution. That's why **xxeconvert** automatically uses the credentials stored by XMLmind XML Editor in the preferences file of the user.

Example:

User john needs to run **xxeconvert** to convert `http://www.acme.com/docs/foo.xml` to HTML.

Server `http://www.acme.com/docs/` requires user john to authenticate himself.

User john starts XMLmind XML Editor and opens `http://www.acme.com/docs/foo.xml`.

A dialog box is displayed prompting user john for his credentials.

User john types his user name and password and, by checking the "**Remember these user name and password**" checkbox, allows XMLmind XML Editor to store the credentials in *`XXE_user_preferences_dir`*`/preferences.properties`.

From then, user john may run **xxeconvert** to convert any document stored on `http://www.acme.com/docs/`.

See also the `-auth` command-line option [4] for an alternative way to let the user of **xxeconvert** authenticate to a server.