XMLmind XML Editor - DITA Support

Hussein Shafie

April 2, 2013

XMLmind

Table of Contents

1. About DITA support in XMLmind XML Editor 1
2. DITA topic reference 1
    2.1. DITA topic menu 1
    2.2. DITA topic tool bar 5
    2.3. DITA topic bindings 7
    2.4. Using the indexterm editor 8
3. DITA map reference 10
    3.1. DITA map menu 10
    3.2. DITA map tool bar 14
    3.3. DITA map bindings 15
4. DITA bookmap reference 15
    4.1. About DITA bookmap support 15
5. Content inclusion 16
    5.1. Easy content inclusion 16
    5.2. Content inclusion: an alternative, low-level, method 17
    5.3. Limitations and specificities of the implementation of transclusion in XMLmind XML Editor 18
6. Rich media content 18
7. Preprocessing options 23

1. About DITA support in XMLmind XML Editor

Out of the box, XMLmind XML Editor (XXE for short) allows to edit topics, maps and bookmaps conforming to the DITA 1.0, 1.1, 1.2 DTD or W3C XML Schema.

By default, DITA documents created using XXE conform to the DTD and not to the W3C XML Schema, but this can be easily changed by customizing the DITA configuration as explained in *XMLmind XML Editor - Configuration and Deployment*.

Note that XMLmind XML Editor only supports "Technical content elements". Other vocabularies, "Learning and training elements", "Classification elements", "Task requirements domain", etc, are not yet officially supported.

As of version 4.5, XXE no longer relies on the DITA Open Toolkit to convert DITA documents to formats such as XHTML, PDF, RTF, etc. Instead XXE embeds *XMLmind DITA Converter*:

> XMLmind DITA Converter (ditac for short) allows to convert the most complex DITA 1.2 documents to production-quality XHTML 1.0, XHTML 1.1, HTML 4.01, XHTML 5.0, Web Help, Java™ Help, HTML Help, Eclipse Help, EPUB 2, EPUB 3, PDF, PostScript®, RTF (can be opened in Word 2000+), WordprocessingML (can be opened in Word 2003+), Office Open XML (`.docx`, can be opened in Word 2007+), OpenOffice (`.odt`, can be opened in OpenOffice.org 2+).
>
> XMLmind DITA Converter is free, open source, software licensed under the very liberal terms of the Mozilla Public License version 1.1.
>
> All this makes XMLmind DITA Converter a very serious alternative to using the DITA Open Toolkit.
>
> More information in http://www.xmlmind.com/ditac/.

2. DITA topic reference

2.1. DITA topic menu

When a DITA topic (of any kind) is opened in XMLmind XML Editor, the **XML** menu becomes the **Topic** menu and this menu is populated with items which are specific to DITA topics. This reference contains a description of such menu items.

Table editor

This table editor may be used to edit `simpletable`s as well as CALS `table`s. Most table editing commands can be repeated by using **Edit** → **Repeat** (`Ctrl-A`).

Note that using this table editor, or simply saving a topic, or checking a topic for validity, guarantees that the `cols` attribute of a `tgroup` is up to date. That is, you may forget about the `cols` attribute, XMLmind XML Editor will always compute it for you.

Menu	Item	Description
Column For a command in this menu to work, click anywhere inside a cell[(1)].	**Insert Before**	Insert a column before column containing specified cell.
	Insert After	Insert a column after column containing specified cell.
	Cut	Cut to the clipboard the column containing specified cell.
	Copy	Copy to the clipboard the column containing specified cell.

(1) or explicitly select a cell or an element having a cell ancestor

Menu	Item	Description
	Paste Before	Paste copied or cut column before column containing specified cell.
	Paste After	Paste copied or cut column after column containing specified cell.
	Delete	Delete the column containing specified cell.
	Sort Rows	Sort all the rows of the table according to the string values of the cells of the “selected column”. (The “selected column” is the column containing specified cell.) A dialog box is displayed allowing to specify the following sort options: **Order** **Dictionary** is the language-specific alphabetical order. Example: (`Charles`, `best`, `Albert`) is sorted as (`Albert`, `best`, `Charles`). **Numeric**. The string value of a cell is expected to start with a number. Example: (`+15.0%`, `1.50%`, `-20%`) is sorted as (`-20%`, `1.50%`, `+15.0%`). **Lexicographic** is the order of Unicode characters. Example: (`Charles`, `best`, `Albert`) is sorted as (`Albert`, `Charles`, `best`). **Dictionary** and **Numeric** orders will cause this menu item to fail, unless the language of the table can be determined (i.e. lookup for the `xml:lang` attribute). **Direction** **Ascending** means: A to Z, low to high. **Descending** means: Z to A, high to low. Note that: • Header rows (i.e. `thead`, `sthead`) are never sorted. • The contents of row groups (i.e. `tbody`) are sorted separately.
Row For a command in this menu to work, click anywhere inside a cell[(1)] or explicitly select a row.	**Insert Before**	Insert a row before row containing specified cell. **Note** Note that row editing commands are enabled, not only by implicitly or explicitly selecting a table cell or any of its descendants, but also by explicitly selecting a table row.
	Insert After	Insert a row before row containing specified cell.
	Cut	Cut to the clipboard the row containing specified cell.
	Copy	Copy to the clipboard the row containing specified cell.
	Paste Before	Paste copied or cut row before row containing specified cell.
	Paste After	Paste copied or cut row after row containing specified cell.
	Delete	Delete the row containing specified cell.
Cell	**Increment Column Span**	Increment the number of columns spanned by specified cell. Not relevant for `simpletables`.

Menu	Item	Description
For a command in this menu to work, click anywhere inside a cell[1].	**Decrement Column Span**	Decrement the number of columns spanned by specified cell. Not relevant for `simpletables`.
	Increment Row Span	Increment the number of rows spanned by specified cell. Not relevant for `simpletables`.
	Decrement Row Span	Decrement the number of rows spanned by specified cell. Not relevant for `simpletables`.

Paste plain text as one or more paragraphs or as a table

Paste As

The entries of this submenu allow to paste the *plain text* copied to the clipboard, typically using a third-party word processor or spreadsheet, as:

- one or more paragraphs,
- OR a `pre` element,
- OR one or more list items,
- OR an itemized list,
- OR one or more table rows,
- OR a table.

The last two menu entries assume that each text line specifies a table row and that, within a text line, the contents of the table cells are separated by tab characters.

Tip

If you need to paste the copied text as an ordered list, first paste this text as an itemized list then convert the pasted list to an ordered list using **Edit** → **Convert** (`Ctrl-T`).

The above menu entries paste elements *after* the implicitly or explicitly selected element.

The following entries of this submenu allow to paste the *image* copied to the clipboard as:

- `image`,
- `fig`.

Except for menu entry "**image**" which replaces the selection or pastes an element at caret position (like **Edit** → **Paste**), the other menu entries paste an element *after* the implicitly or explicitly selected element.

Indexterm editor

Insert or Edit indexterm

If the caret is anywhere inside an `indexterm` element or if a single element or node is explicitly selected anywhere inside an `indexterm` element, this menu item displays an `indexterm` editor dialog box allowing to modify this `indexterm` element.

Otherwise, this menu item displays an `indexterm` editor dialog box allowing to create a new `indexterm` element and then to insert it at caret position.

Tip

If some text has been selected, field **Term** of the dialog box is automatically initialized with the text selection. Therefore the simplest way to create an `indexterm` element is first to select the term in the body of the document, then invoke **Topic** → **Insert or Edit indexterm** and finally click **OK**.

Moving elements

Move Up
: Move selected element up, that is, swap it with its preceding sibling node. Requires the element to be explicitly selected.

Move Down
: Move selected element down, that is, swap it with its following sibling node. Requires the element to be explicitly selected.

Convert Document menu

Attention

The items of this menu are all disabled if the document being edited needs to be saved to disk.

Convert to XHTML
Convert to XHTML [one page]
: Converts the document being edited to multi page or single page XHTML 1.0.

Convert to Web Help
: Converts the document being edited to Web Help.

Convert to HTML Help
: Converts the document being edited to a `.chm` file. This command is disabled on platforms other than Windows.

Requires:

1. Download and install Microsoft®'s HTML Help Workshop.
2. Declare the HTML Help compiler, `hhc.exe`, as the helper application associated to files having a "`hhp`" extension. This can be specified by using the **Preferences** dialog box, **Helper Applications** section.

Convert to Java Help
: Converts the document being edited to a `.jar` file for use by the Java™ Help system.

Requires:

1. Download and install JavaHelp.
2. Declare the Java™ Help indexer, `jhindexer` (`jhindexer.bat` on Windows), as the helper application associated to files having a "`application/x-java-help-index`" MIME type. This can be specified by using the **Preferences** dialog box, **Helper Applications** section.

Convert to Eclipse Help
: Converts the document being edited to a directory containing various files for use by the Eclipse Help system.

Convert to EPUB
: Converts the document being edited to an `.epub` file.

Convert to RTF (Word 2000+)
: Converts the document being edited to RTF (Rich Text Format) using. The document generated by this command can be edited and printed using Microsoft® Word 2000 and above.

Requires downloading and installing the "*XMLmind FO Converter XSL-FO processor plug-in* " add-on using **Options** → **Install add-ons**.

Convert to WordprocessingML (Word 2003+)
: Converts the document being edited to WordprocessingML. The document generated by this command can be edited and printed using Microsoft® Word 2003 and above.

Same requirements as Convert to RTF.

Convert to Office Open XML (Word 2007+)
: Converts the document being edited to Office Open XML (.docx file) . The document generated by this command can be edited and printed using Microsoft® Word 2007 and above.

  Same requirements as Convert to RTF.

Convert to OpenDocument (OpenOffice.org 2+)
: Converts the document being edited to OpenDocument (.odt file). The document generated by this command can be edited and printed using OpenOffice.org 2.

  Same requirements as Convert to RTF.

Print PostScript
: Convert the document being edited to PostScript® and send the generated file to the chosen printer.

  Requires downloading and installing any of the following add-ons using **Options** → **Install add-ons**.

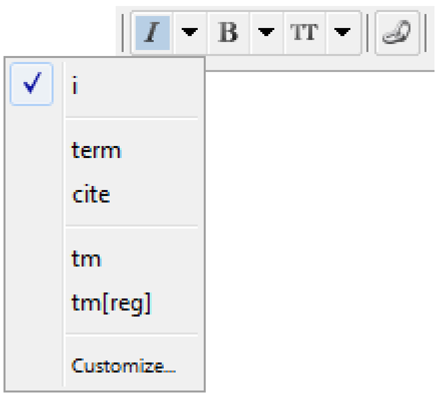
  - *Apache FOP 1.x XSL-FO processor plug-in*;
  - *RenderX XEP XSL-FO processor plug-in*[2].

Convert to PDF
: Convert the document being edited to PDF.

  Same requirements as Print PostScript.

2.2. DITA topic tool bar

When a DITA topic (of any kind) is opened in XMLmind XML Editor, buttons which are specific to this kind of document are automatically added to the tool bar. This reference contains a description of such buttons.

Button	Description
I Toggle i	"Toggle" element `i`. Next to this toggle is found an arrow button displaying a menu containing additional checkboxes for the following elements: `term`, `cite`, `tm`, `tm[reg]`. **Note** The DITA tool bar starts with a number of "text style" toggles. These toggles emulate the behavior of the **Bold**, **Italic**, **Underline**, etc, toggles found in the tool bars of almost all word-processors. More information about text style toggles in "About text style toggles". *Figure 1. Toggles found at the beginning of the DITA tool bar*  I B TT ✓ i term cite tm tm[reg] Customize...

[2] Unlike all the other add-ons, the RenderX XEP XSL-FO processor plug-in is not self-contained. You'll need to download, install and activate RenderX XEP (for example, free Personal Edition) prior to using the RenderX XEP XSL-FO processor plug-in.

Button	Description
	In the above screenshot, the caret is inside an `i` element and the user clicked the arrow button next to a "italic text style" toggle.
Toggle `b`	"Toggle" element `b`. Next to this toggle is found an arrow button displaying a menu containing additional checkboxes for the following elements: `keyword`, `uicontrol`, `option`.
Toggle tt	"Toggle" element `tt`. Next to this toggle is found an arrow button displaying a menu containing additional checkboxes for the following elements: `filepath`, `varname`, `cmdname`, `apiname`.
Toggle xref	"Toggle" element `xref`.
Add `p`	Add a `p` after node selection or after caret at a location where it is valid to do so and where it makes sense to do so. **Note** This command and all the following commands will never add an element *inside* a `p`, even it is valid to do so. These commands add elements always *after* a `p`. That is, a `p` element is always considered by these commands as being a plain paragraph and never as being a division.
Add list item	Add a list item of the right type after current list item. For this command to work, suffice to click anywhere inside an `sl`, `ul`, `ol`, `dl`, `choices`, `substeps`, `steps`, `steps-unordered`.
Add `ul`	Add an `ul` after node selection or after caret at a location where it is valid to do so and where it makes sense to do so (see note).
Add `ol`	Add an `ol` after node selection or after caret at a location where it is valid to do so and where it makes sense to do so (see note).
Add `dl`	Add a `dl` after node selection or after caret at a location where it is valid to do so and where it makes sense to do so (see note).
Add pre	Displays a menu which allows to add a `pre`, `lines`, `screen`, `codeblock` or a `msgblock` after node selection or after caret at a location where it is valid to do so and where it makes sense to do so (see note).
Add table	Displays a menu which allows to add a `simpletable` or a `table` after node selection or after caret at a location where it is valid to do so and where it makes sense to do so (see note).
Add image	Displays a menu which allows to • insert an `image` at caret position; • OR add a `fig` (containing an `image`) after node selection or after caret at a location where it is valid to do so and where it makes sense to do so (see note).
Insert media object	Displays a menu which allows to insert a "media object" at caret position. **object(audiovideo)** An `object` element allowing to add audio or video to your topic. **object(flash)** An `object` element allowing to add an Adobe® Flash® animation to your topic. **xref(play)** An `xref` element containing `<?onclick play()?>`. Last menu item allows to insert an `<?onclick?>` processing-instruction at caret position or, if an `<?onclick?>` processing-instruction is selected, to edit it.

Button	Description
	More information about the above “media objects” in Section 6.
§ Add `section`	Add a `section` after node selection or after caret at a location where it is valid to do so and where it makes sense to do so (see note).
Browse the DITA reference manual found on the OASIS Web site	Use the Web browser to display the documentation of the element explicitly or implicitly selected. (Contributed by Mark Fletcher.) *Requires to be connected to the Internet* because the reference manual of DITA elements is found on the OASIS Web site.

2.3. DITA topic bindings

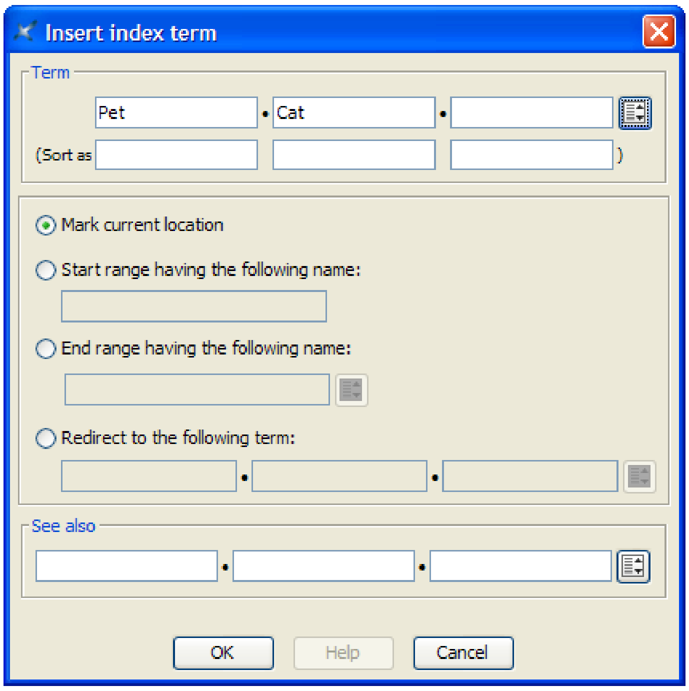
When a DITA topic (of any kind) is opened in XMLmind XML Editor, additional keyboard shortcuts and additional drag and drop facilities which are specific to this kind of document are automatically made available to the user. This reference contains a description of such user input/command bindings.

Keyboard shortcut	Description
`Enter`	Insert a newline character if possible. Otherwise, if caret is at the beginning of a paragraph, list item or a few other kinds of block, insert same block before. Otherwise, if caret is at the end of a block, insert same block after. Otherwise, split block.
`Del`	Delete selection if any. Otherwise, if caret is at the end of a paragraph, list item or a few other kinds of block, join with following block. Otherwise, delete character following caret.
`BackSpace`	Delete selection if any. Otherwise, if caret is at the beginning of a paragraph, list item or a few other kinds of block, join with preceding block. Otherwise, delete character preceding caret.
`Ctrl-Enter`	Add same block after the paragraph, list item or a few other kinds of block which is the ancestor of selected node.
`Ctrl+Shift-Enter`	Add same block before the paragraph, list item or a few other kinds of block which is the ancestor of selected node.
`Ctrl-F1`	Browse the DITA reference manual found on the OASIS Web site
Drop an object.	If the drop occurs above an element having an `href` attribute other than an `image` (e.g. an `xref`), the dropped string is considered to be an URL and is used to change the value of the `href` attribute. Note that this kind of drop attempts to *relativize* the dropped URL against the location of the drop site. For example, if you drop "`file://home/john/doc/topic1.dita`" onto an `xref` contained in file "`file://home/john/doc/ref/reference2.dita`", its `href` is set to "`../topic1.dita`". Elsewhere, normal behaviour which is: **Drop onto an `image` element** Considers the dropped string to be the URL or the filename of a graphics file. Displays a dialog box allowing to copy or reference this graphics file for use by the `image` element. **Drop elsewhere** If the object being dropped is an URL or an absolute filename, open the corresponding document. Otherwise, paste the dropped text or XML at or after the drop location.
Drag one of the “handles” displayed around an image. (The “handles” are	Resize the image, but always preserve its aspect ratio.

Keyboard shortcut	Description
displayed after clicking on the image.)	Pressing `Ctrl` (`Cmd` on the Mac) while dragging the handle allows to distort the image.
Drag a separator found between two table columns.	Resize the table column. More precisely this gives an appropriate proportional width (e.g. `<colspec colwidth="3*">`) to all table columns.

2.4. Using the `indexterm` editor

This dialog box, displayed by menu item **Topic** → **Insert or Edit indexterm**, allows to insert or edit an `indexterm` element.


Insert index term
Term
Pet
Cat
(Sort as
)
Mark current location
Start range having the following name:
End range having the following name:
Redirect to the following term:
See also
OK
Help
Cancel

We'll explain with examples how to use the `indexterm` editor.

- If you want to get this kind of entry in your back of the book index:

  ```
  P
  Pet 12
  ```

  specify **Term**=`Pet`.

- Back of the book index:

  ```
  P
  Pet
      Cat 26
  ```

  specify **Term**=`Pet`, **Term** #2=`Cat`.

- Back of the book index:

  ```
  P
  "+" 54
  ```

  specify **Term**=`"+"`, **Sort as**=`plus`. Without this **Sort as** specification, the index entry corresponding to `"+"` would have been found in the **Symbols** category:

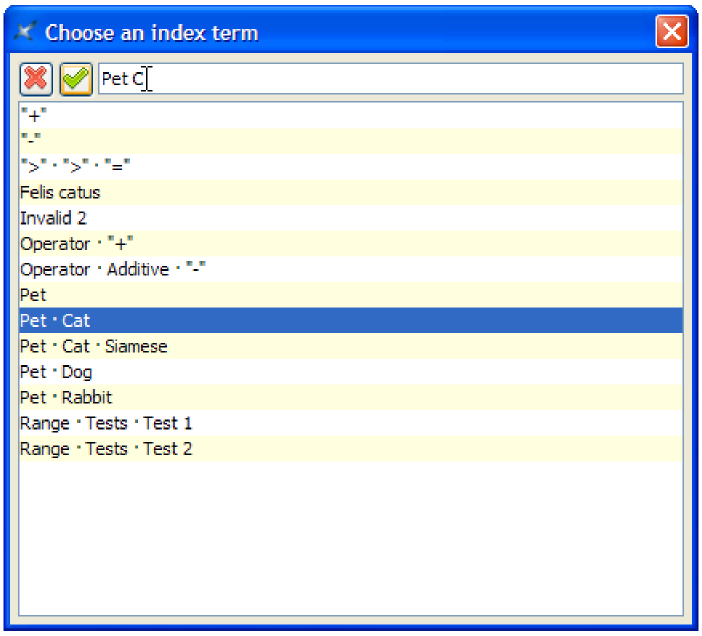
```
Symbols
"*" 53, 78
"+" 54
"-" 55, 91-95
...
```

- Back of the book index:

```
D
Domesticated animals 34 See also Pet
```

specify **Term**=`Domesticated animals`, **See also**=`Pet`.

Note that the content of the **See also** field must refer to an existing index entry. That's why instead of typing "`Pet`", you can select this index entry by using the dialog box displayed by the **Pick from list** button found at the right of the **See also** row.



"+"
"-"
">" · ">" · "="
Felis catus
Invalid 2
Operator · "+"
Operator · Additive · "-"
Pet
Pet · Cat
Pet · Cat · Siamese
Pet · Dog
Pet · Rabbit
Range · Tests · Test 1
Range · Tests · Test 2

The above dialog box supports autocompletion. Note that if, for example, you want specify compound term "`Pet Cat Siamese`", you must type a space character between each simple term.

- Back of the book index:

```
F
Felis catus See Pet, Cat
```

specify **Term**=`Felis catus`, select "**Redirect to the following term**" then specify **Redirect**=`Pet`, **Redirect** #2=`Cat`. (In the above example, notice that `Felis catus` has no associated page number.)

Like **See Also**, the content of the **Redirect** field must refer to an existing index entry. Unlike **See Also**, a **Redirect** entry is merely a redirection to an actual index entry.

- Back of the book index:

```
O
Operation
    Additive
        "+" 87-90
```

1. Insert a first `indexterm` element at the beginning the range (this will give us page number 87).

In order to do that, use **Topic** → **Insert or Edit indexterm** and specify **Term**=`Operation`, **Term** #2=`Additive`, **Term** #3=`"+"`, **Sort as** #3=`plus`.

Then check "**Start range having the following name**" and give your range an identifier by specifying "`plus_reference`" in the **Start range** field.

2. Insert another `indexterm` element at the end the range (this will give us page number 90).

   In order to do that, use **Topic** → **Insert or Edit indexterm**, check "**End range having the following name**" and specify the same identifier, "`plus_reference`", in the **End range** field. All the other fields must be left blank.

   Note that instead of typing "plus_reference" in the **End range** field, you can select this identifier by using the dialog box displayed by the **Pick from list** button found at the right of the **End range** field.

Related information

• The "Insert or Edit indexterm" menu item

3. DITA map reference

3.1. DITA map menu

When a DITA map is opened in XMLmind XML Editor, the **XML** menu becomes the **Map** menu and this menu is populated with items which are specific to DITA maps. This reference contains a description of such menu items.

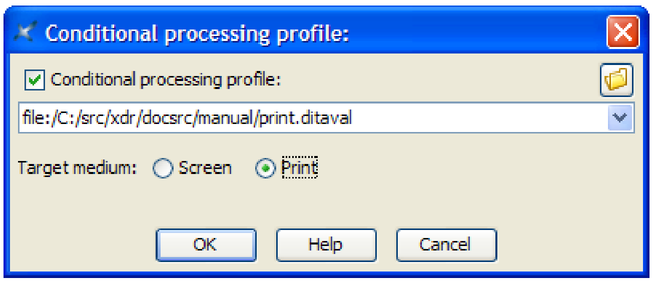
Check map

Conditional Processing Profile

Displays a dialog box allowing to specify a conditional processing profile (a `.ditaval` file) which is to be applied to the map being edited and also the medium targeted by this map. The conditional processing profile is used by the **Check Map** command and also by all the **Convert** commands found in the **Convert Document** menu.

The target medium specified in this dialog box is used only by the **Check Map** command. If you specifically target a print form (PDF, PostScript, RTF, etc) for your deliverable, check **Print**. In any other case, check **Screen**.

*Figure 2. The dialog box displayed by menu item "**Conditional Processing Profile**"*



[✓] Conditional processing profile:

file:/C:/src/xdr/docsrc/manual/print.ditaval

Target medium: Screen [✓ Print]

Remember

Note that the values specified in the above dialog box are remembered for use during subsequent editing sessions. For example, in the case of the above screenshot, if you reopen the same map later, this map will still be filtered by `print.ditaval` and its target medium will still be `Print`, and this even if you do not explicitly use menu item **Map → Conditional Processing Profile** during the new editing session.

Check Map

Extensively check the map being edited. This task which can be lengthy is run in background. While this task is running, a non-modal dialog box displays all the errors and warnings found in the map being edited, its submaps and all the topics referenced by these maps. If no errors or warnings are found, the dialog box is automatically closed. Otherwise it stays opened allowing you to review each error or warning. After you are done, you'll have to close the dialog box by clicking **Close** if you want to be able to re-run **Check Map**.

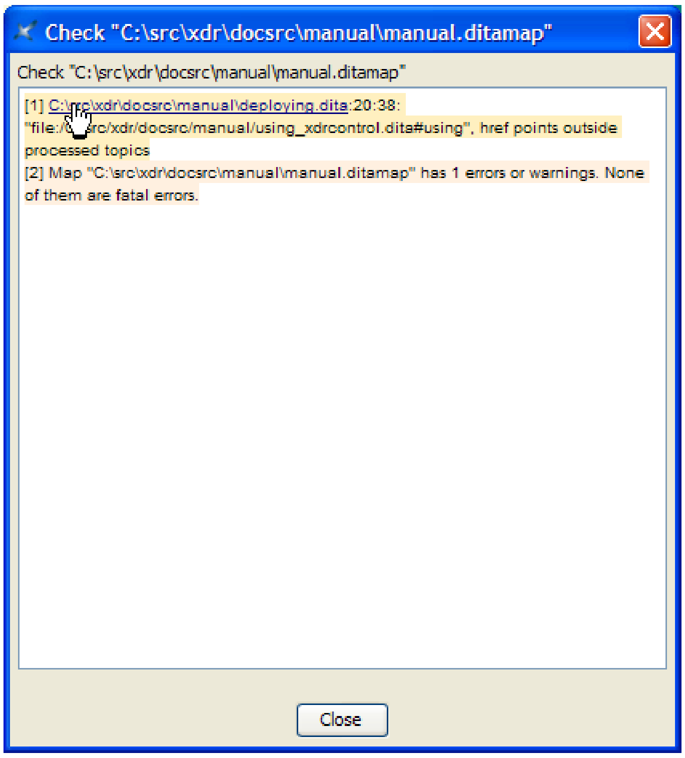


*Figure 3. The dialog box displayed by menu item "**Check Map**"*

Check "C:\src\xdr\docsrc\manual\manual.ditamap"

Check "C:\src\xdr\docsrc\manual\manual.ditamap"

```
[1] C:\src\xdr\docsrc\manual\deploying.dita:20:38:
"file:/C:/src/xdr/docsrc/manual/using_xdrcontrol.dita#using", href points outside
processed topics
[2] Map "C:\src\xdr\docsrc\manual\manual.ditamap" has 1 errors or warnings. None
of them are fatal errors.
```

As you can see it in the above screenshot, clicking on an underlined filename or URL opens the corresponding topic or map in XMLmind XML Editor and selects the element having the error or a warning.

Convert Document menu

Attention

The items of this menu are all disabled if the document being edited needs to be saved to disk.

Convert to XHTML
Convert to XHTML [one page]

Converts the document being edited to multi page or single page XHTML 1.0.

Convert to Web Help

Converts the document being edited to Web Help.

Convert to HTML Help

Converts the document being edited to a `.chm` file. This command is disabled on platforms other than Windows.

Requires:

1. Download and install Microsoft®'s HTML Help Workshop.
2. Declare the HTML Help compiler, `hhc.exe`, as the helper application associated to files having a "`hhp`" extension. This can be specified by using the **Preferences** dialog box, **Helper Applications** section.

Convert to Java Help

Converts the document being edited to a `.jar` file for use by the Java™ Help system.

Requires:

1. Download and install JavaHelp.
2. Declare the Java™ Help indexer, `jhindexer` (`jhindexer.bat` on Windows), as the helper application associated to files having a "`application/x-java-help-index`" MIME type. This can be specified by using the **Preferences** dialog box, **Helper Applications** section.

Convert to Eclipse Help

Converts the document being edited to a directory containing various files for use by the Eclipse Help system.

Convert to EPUB

Converts the document being edited to an `.epub` file.

Convert to RTF (Word 2000+)

Converts the document being edited to RTF (Rich Text Format) using. The document generated by this command can be edited and printed using Microsoft® Word 2000 and above.

Requires downloading and installing the "*XMLmind FO Converter XSL-FO processor plug-in* " add-on using **Options** → **Install add-ons**.

Convert to WordprocessingML (Word 2003+)

Converts the document being edited to WordprocessingML. The document generated by this command can be edited and printed using Microsoft® Word 2003 and above.

Same requirements as Convert to RTF.

Convert to Office Open XML (Word 2007+)

Converts the document being edited to Office Open XML (.docx file) . The document generated by this command can be edited and printed using Microsoft® Word 2007 and above.

Same requirements as Convert to RTF.

Convert to OpenDocument (OpenOffice.org 2+)

Converts the document being edited to OpenDocument (.odt file). The document generated by this command can be edited and printed using OpenOffice.org 2.

Same requirements as Convert to RTF.

Print PostScript

Convert the document being edited to PostScript® and send the generated file to the chosen printer.

Requires downloading and installing any of the following add-ons using **Options** → **Install add-ons**.

- *Apache FOP 1.x XSL-FO processor plug-in*;
- *RenderX XEP XSL-FO processor plug-in*[3].

Convert to PDF

Convert the document being edited to PDF.

(3) Unlike all the other add-ons, the RenderX XEP XSL-FO processor plug-in is not self-contained. You'll need to download, install and activate RenderX XEP (for example, free Personal Edition) prior to using the RenderX XEP XSL-FO processor plug-in.

Same requirements as Print PostScript.

Changing the look and contents of the files generated by the Convert Document menu

There are three ways to change the look and contents of the files generated by the items of the **Convert Document** menu.

1. Specifying custom XSLT stylesheet parameters. This is done by using **Options** → **Customize Configuration** → **Change Document Conversion Parameters**.

   For example, selecting parameter group "**Convert to HTML Help**" and adding parameter `number` with value `all` allows to number the sections of the generated `.chm` file.

   The reference manual of the parameters of the XSLT stylesheets used to perform the conversion is found in XMLmind DITA Converter Manual - XSLT stylesheets parameters. This reference manual can be directly accessed from the "**Change Document Conversion Parameters**" dialog box.

2. Using menu item **Options** → **Customize Configuration** → **Customize Document Conversion Stylesheets** is also a relatively simple way to influence the layout and style of the deliverable (PDF, RTF, HTML, etc) which results from the document conversion.

   The document being edited is converted to other formats by the means of XSLT stylesheets. This menu item allows to:

   - select an XSLT stylesheet other the default one,
   - create a custom XSLT stylesheet on the fly,
   - invoke a specialized editor —XMLmind XSL Customizer— to modify a user-created XSLT stylesheet.

   However, when the document being edited is converted to an HTML-based format (Web Help, EPUB, HTML Help, etc), the HTML pages which are automatically generated by the aforementioned XSLT stylesheets are styled mainly by *CSS stylesheets*. When this is the case, this menu item allows additionally to:

   - select a CSS stylesheet other the default one,
   - create a custom CSS stylesheet on the fly,
   - invoke a helper application (generally, a text editor) to modify a user-created CSS stylesheet.

3. To a lesser extent, changing the options of the XMLmind DITA Convert (ditac) preprocessor. This is done by using **Options** → **Customize Configuration** → **Preprocessing Options**.

   For example, selecting option group "**Convert to PDF, PostScript**" and then selecting "**Generate as backmatter**" in the **Index** combobox allows to add an index at the end of the generated PDF files.

   More information about this facility in Section 7.

Note that a technical writer is not expected to know which parameter, option or style is to be specified to get the desired effect. Unless she/he is the local guru, a technical writer is expected to post a support request to the xmleditor-support public, moderated, mailing list in order to learn this. But at least the three above facilities allow her/him to customize her/his deliverables without having to hand edit configuration files.

Syntax highlighting

You can automatically colorize the source code contained in `pre`, `codeblock` or any other element specializing `pre`. This feature, commonly called *syntax highlighting*, has been implemented using an open source software component called "XSLT syntax highlighting".

If you want to turn on syntax highlighting in a DITA document, suffice to add attribute `outputclass` to a `pre`, `codeblock` or any other element specializing `pre`. The value of attribute `outputclass` must be any of: `language-c`, `language-cpp`, `language-csharp`, `language-delphi`, `language-ini`, `language-`

java, language-javascript, language-m2, language-perl, language-php, language-python, language-ruby, language-tcl, language-xml.

If you want to customize syntax highlighting for an HTML-based output format (XHTML, EPUB, etc), redefine any of the following CSS styles:

- .hl-keyword (keywords of a programming language),
- .hl-string (string literal),
- .hl-number (number literal),
- .hl-comment (any type of comment),
- .hl-doccomment (comments used as documentation, i.e. javadoc, or xmldoc),
- .hl-directive (preprocessor directive or in XML, a processing-instruction),
- .hl-annotation (annotations or "attributes" as they are called in .NET),
- .hl-tag (XML tag, i.e. element name),
- .hl-attribute (XML attribute name),
- .hl-value (XML attribute value),
- .hl-doctype (<!DOCTYPE>and all its content).

Example:

```
.hl-keyword {
    font-weight: bold;
    color: #602060;
}
```

This can be done from within XXE using **Options** → **Customize Configuration** → **Customize Document Conversion Stylesheets**.

If you want to customize syntax highlighting for an XSL-FO-based output format (PDF, RTF, etc), redefine any of the following attribute-sets: hl-keyword, hl-string, hl-number, hl-comment, hl-doccomment, hl-directive, hl-annotation, hl-tag, hl-attribute, hl-value, hl-doctype.

Example:

```
<xsl:attribute-set name="hl-keyword" use-attribute-sets="hl-style">
  <xsl:attribute name="font-weight">bold</xsl:attribute>
  <xsl:attribute name="color">#602060</xsl:attribute>
</xsl:attribute-set>
```

This can be done from within XXE using **Options** → **Customize Configuration** → **Customize Document Conversion Stylesheets**.

3.2. DITA map tool bar

When a DITA map is opened in XMLmind XML Editor, buttons which are specific to this kind of document are automatically added to the tool bar. This reference contains a description of such buttons.

Button	Description
Insert topicref before	Insert a topicref before selected topicref, topichead, topicgroup.
Insert topicref into	Append a topicref child element to the selected topicref, topichead, topicgroup.
Insert topicref after	Insert a topicref after selected topicref, topichead, topicgroup.
Move up	If selected element is a topicref, topichead, topicgroup, navref or anchor and if this selected element is preceded by a topicref, topichead,

Button	Description
	`topicgroup`, `navref` or `anchor`, this command moves selected element one position up in its container element.
⬇ Move down	If selected element is a `topicref`, `topichead`, `topicgroup`, `navref` or `anchor` and if this selected element is preceded by a `topicref`, `topichead`, `topicgroup`, `navref` or `anchor`, this command moves selected element one position down in its container element.
⬅ Promote	If selected element is a `topicref`, `topichead`, `topicgroup`, itself contained in a `topicref`, `topichead`, `topicgroup`, this command removes selected element from its parent and makes it a sibling following its ex-parent. Moreover, all elements that followed selected element in the ex-parent are also removed from this ex-parent and made children of the newly ``promoted" element.
➡ Demote	If selected element is a `topicref`, `topichead`, `topicgroup`, itself preceded by a `topicref`, `topichead`, `topicgroup`, this command moves selected element at the end of its predecessor.
Edit referenced topic or map	Opens in XMLmind XML Editor the topic or map referenced in the `conref`, `href` or `mapref` attribute of selected element. If this document is already opened in XMLmind XML Editor, this command brings its view to front and makes it the active one. If this document does not exist yet, the **File** → **New** dialog box is displayed and allows the user to choose which document (i.e. `topic`, `task`, `reference`, `map`, etc) is to be created at the location specified by attribute `href`, `mapref` or `conref`.

3.3. DITA map bindings

When a DITA map is opened in XMLmind XML Editor, additional keyboard shortcuts which are specific to this kind of document are automatically made available to the user. This reference contains a description of such keyboard shortcuts.

Keyboard shortcut	Description
`F2 Up`	Move up
`F2 Down`	Move down
`F2 Left`	Promote
`F2 Right`	Demote
`F2 e`	Edit referenced topic or map

4. DITA bookmap reference

4.1. About DITA bookmap support

When a DITA bookmap is opened in XMLmind XML Editor, the **XML** menu becomes the **BookMap** menu and this menu is populated with items which are specific to DITA bookmaps. Similarly, buttons which are specific to this kind of document are automatically added to the tool bar. The **BookMap** menu and tool bar are identical to those specific to DITA maps.

Related information

• Section 3.1. DITA map menu

5. Content inclusion

In the next two sections, we'll learn how to reference in topic A some contents found in a topic B. We'll first learn how to do it the easy way by using **Copy as Reference**/**Paste**. Then, for those who prefer to control everything to the finest degree, we'll learn how to achieve the same results using a low-level method.

5.1. Easy content inclusion

Before you begin

The transclusion of elements having a `conref` attribute must be turned on (which is the case by default).

Note

As of XMLmind XML Editor v4.9, it's possible to completely turn off the transclusion of conref by using menu item **Options** → **Customize Configuration** → **Conref Transclusion**. Note that this user preference is specified separately for topics and for maps.

About this task

The `conref` attribute of an element allows to reference the contents (text, child elements, some of the attributes) of another element.

Instead of just seeing an empty element having a `conref` attribute (that is, the ``pointer''), it is nicer to see the referenced contents. This process is called *transclusion* and XMLmind XML Editor can do it for you.

Note

Everything explained here should also work for DITA maps.

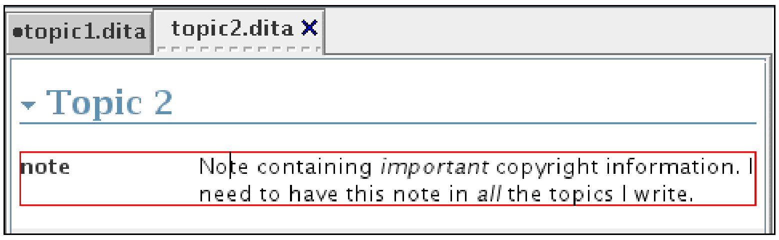
Note

The following procedure (**Copy as Reference** then **Paste**) is not specific to DITA. The same procedure could be used to add references to DocBook or XHTML documents. This is why it is explained in great details in our tutorial.

Procedure

1. Open in XMLmind XML Editor the topic containing the element you want to reference.
2. Select this element.

   Let's call this element the *conref target*.



Topic 2

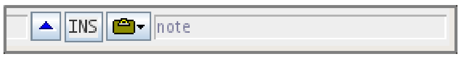
note Note containing *important* copyright information. I need to have this note in *all* the topics I write.

3. If this selected element has no `id` attribute, specify one using the **Attributes** tool.
4. If you want to reference a *range of nodes* rather a single element, extend the selection (**Select** → **Extend Selection to Following Sibling**, `Esc Right-Arrow`) to some nodes following this first selected element.

Just make sure that the end of the node range is an element having the same type as the first selected element and that this end of range element has an `id` attribute.

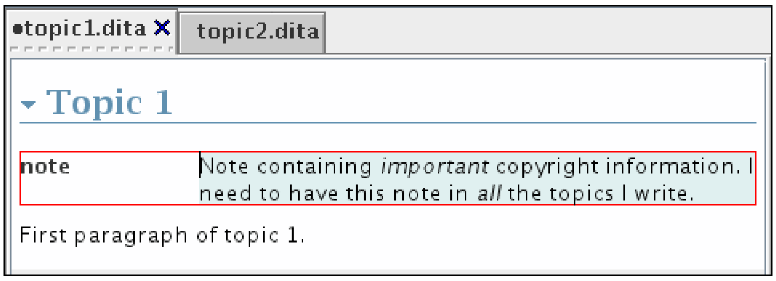
5. Press Ctrl+Shift-C (**Edit** → **Reference** → **Copy as Reference**).

   You'll see the name of the element copied as reference displayed in dimmed blue at the bottom right of XMLmind XML Editor main window.

6. Switch to the topic where you want to create the reference.
7. Use Ctrl-U (**Edit** → **Paste Before**), Ctrl-V (**Edit** → **Paste**) or Ctrl-W (**Edit** → **Paste After**) to paste a reference to the conref source.

   Let's call this pasted reference the *conref source*.

   Topic 1

   note Note containing *important* copyright information. I need to have this note in *all* the topics I write.

   First paragraph of topic 1.

8. Sometimes, you'll want to add attributes which are specific to the conref source (typically an `id` attribute). In such case:

   a. Select the conref source.
   b. Use **Edit** → **Reference** → **Untransclude** to un-transclude the conref source.

      You'll see an element having the same name as the conref source but having no content and having a `conref` attribute pointing the conref target.
   c. Use the **Attributes** tool to specify one or more attributes.
   d. Use **Edit** → **Reference** → **Retransclude** to re-transclude the conref source.

Related information

• Section 5.2. Content inclusion: an alternative, low-level, method

5.2. Content inclusion: an alternative, low-level, method

Procedure

1. Insert the element (the *conref source*) you wish to transform into a reference to another element contained elsewhere (the *conref target*).

   You may use Ctrl-H (**Edit** → **Insert Before**), Ctrl-I (**Edit** → **Insert**) or Ctrl-J (**Edit** → **Insert After**) to do this.
2. Using the **Attributes** tool, specify a `conref` attribute for the conref source.

   Specifying a value “by hand” for the `conref` attribute is tedious and error-prone. That's why using this method rather than the easy one described in Section 5.1 is not recommended.

3. If you want to reference a *range of nodes* rather a single element, use to **Attributes** tool to also give a `conrefend` attribute to the conref source.
4. Specify other attributes, for example an `id` attribute, if you want.
5. Use **Edit** → **Reference** → **Retransclude** to transclude the conref source.

Related information

- Section 5.1. Easy content inclusion

5.3. Limitations and specificities of the implementation of transclusion in XMLmind XML Editor

Limitations

- Content inclusion achieved using **Copy as Reference**/**Paste** does not perform every possible check on the validity of the reference. That's why it's possible to use **Copy as Reference**/**Paste** successfully in a document and still get errors when you'll convert this document to other formats.
- Content *pushed* from one topic to another (the `conaction` attribute) is not transcluded by XMLmind XML Editor.
- Something like `<keyword keyref="product-name"/>`, where the definition of key `product-name` contains `<keyword>Thing-O-Matic</keyword>` is not transcluded by XMLmind XML Editor.

All these limitations apply only to XMLmind XML Editor as an *authoring tool*. They do not apply when you'll use XMLmind XML Editor to convert a DITA document to formats such as HTML, PDF, RTF, etc.

Specificities

If your topics make use of attributes `keyref` and/or `conkeyref`, it is strongly recommended:

1. To turn on option "**Enable the 'File|Document Set' Submenu**" (**Options** → **Preferences**, **General**|**Features** section).
2. If your topics make use of attribute `conkeyref`, turn on "**Automatically update all inclusions in member documents**" (**Options** → **Preferences**, **Tools**|**Document Set** section).
3. If your topics make use of attribute `keyref` to point to image files, turn on "**Automatically redraw member documents**" (**Options** → **Preferences**, **Tools**|**Document Set** section).
4. To open your DITA map as a *document set* using **File** → **Document Set** → **Open Document Set**.

By doing this, you'll instruct XMLmind XML Editor to use your DITA map as a *key space* for all the topics referenced by this map. More information about document sets in "*XMLmind XML Editor - Online Help*".

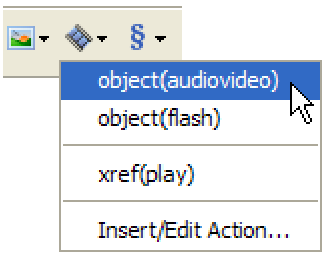
6. Rich media content

This chapter explains how to add SVG, MathML, audio, video and Flash animations to your DITA topics and how **`ditac`** processes this rich media content in the case where the output format supports rich media (e.g. XHTML 5, EPUB 3) and also in the case where the output format does not support rich media (e.g. XHTML 1, PDF, RTF).

Note

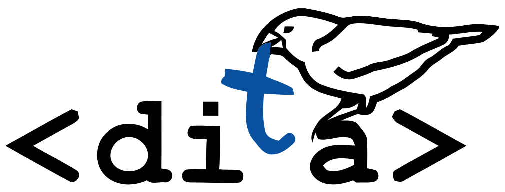
XMLmind XML Editor has an "**Insert media object**" button in its DITA **Topic** tool bar which allows to easily insert any of the elements and processing-instructions described in this chapter.

*Figure 4. The menu displayed by the "**Insert media object**" button*



SVG

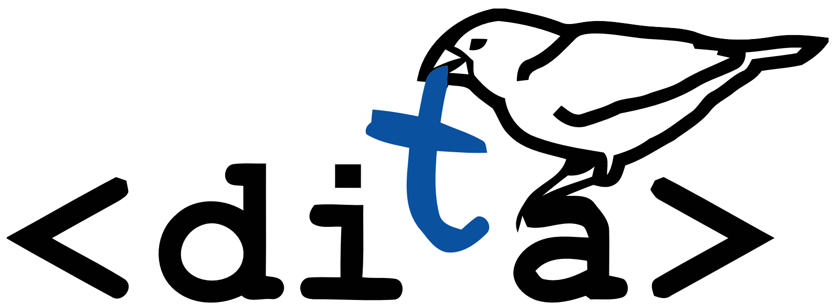
It is possible to include SVG graphics in a DITA topic either by reference or by inclusion. Use an `image` element pointing to an SVG file to include it by reference. Example:



The XML source code corresponding to the above example is:

```
<p><image href="media/graphic.svg"/></p>
```

Embed the `svg:svg` element in a `foreign` element (or any element specializing `foreign`) to include it by inclusion. Example:



The XML source code corresponding to the above example is:

```
<p><foreign><svg:svg height="140" id="svg2" version="1.1"
  viewBox="0 0 264 120" width="320"
  xmlns:svg="http://www.w3.org/2000/svg">
  <svg:defs id="defs39"/>
  ...
  </svg:g>
</svg:svg></foreign></p>
```

- It is recommended to include SVG graphics by reference as the `image` element has useful attributes (`width`, `height`, `scale`, `scalefit`) allowing to adjust the dimension of the image.
- Only the following screen formats may contain SVG: XHTML 5, XHTML 5 Web Help and EPUB 3. Note that only modern web browsers support XHTML 5 and XHTML 5 Web Help. Very few EPUB readers (e.g. iBooks) support EPUB 3.

- All XSL-FO based formats support SVG whatever the XSL-FO processor you may use.

MathML

Embed the `mml:math` element in a `foreign` element (or any element specializing `foreign`) to add math to your DITA topics. Example:

$$\begin{cases} \nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \\ \nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} \end{cases}$$

The XML source code corresponding to the above example is:

```
<p><foreign><mml:math display="block"
  xmlns:mml="http://www.w3.org/1998/Math/MathML">
  <mml:mrow>
    <mml:mo>{</mml:mo>
    <mml:mtable>
      <mml:mtr>
        <mml:mtd>
        ...
        </mml:mtd>
      </mml:mtr>
    </mml:mtable>
  </mml:mrow>
</mml:math></foreign></p>
```

- Only the following screen formats may contain MathML: XHTML 5, XHTML 5 Web Help and EPUB 3. Most modern web browsers (Firefox, Chrome) support XHTML 5 and XHTML 5 Web Help containing MathML. Very few EPUB readers (e.g. iBooks) support EPUB 3.

- XSL-FO based formats support MathML dependin g on the XSL-FO processor you use:

  – Apache FOP requires you to download and install the the JEuclid FOP plug-in.
  – RenderX XEP does not support MathML.
  – Antenna House Formatter supports MathML as an option.
  – XMLmind XSL-FO Converter supports MathML out of the box.

Audio

Use the `object` DITA element to add audio to your DITA topics. Example:

audio.mp3 (audio/mpeg)

The XML source code corresponding to the above example is:

```
<p><object data="media/audio.mp3" type="audio/mpeg">
     <param name="source.src" value="media/audio.ogg"/>
     <param name="source.type" value="audio/ogg"/>

     <param name="source.src" value="media/audio.m4a"/>
     <param name="source.type" value="audio/mp4"/>

     <param name="source.src" value="media/audio.wav"/>
     <param name="source.type" value="audio/wav"/>

     <param name="controls" value="true"/>
</object></p>
```

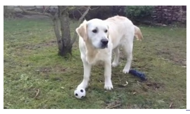
- The `data` and `type` attributes are required. The value of the `type` attribute must start with "`audio/`".

- It is strongly recommended to specify *alternate audio files* as modern web browsers, while all supporting the HTML 5 `audio` element, vary in their support of audio formats. This is done by adding pairs of param child elements to the object element. The first param of a pair must have name="source.src" attribute and its value attribute must reference an audio file. The second param of a pair must have name="source.type" attribute and its value attribute must contain the media type of the preceding audio file.

- It is possible to add param elements corresponding to the attributes supported by the HTML 5 audio element (crossorigin, preload, autoplay, mediagroup, loop, muted, controls). In the above example, we have added a param element corresponding to the controls HTML 5 attribute. Note that in the case of HTML 5 *boolean* attributes (autoplay, loop, muted, controls), the value attribute of a param is not significant. For example, in the case of the above example, you could have specified "yes", "on", "1", etc, instead of "true".
- If the object element has a desc child element, then this desc element is used to generate fallback content in case audio is not supported. If the object element has no desc child element, then a simple fallback content is automatically generated by ditac. This automatic fallback content basically consists in a link allowing to download the audio file.
- When ditac is used to generate an XSL-FO based format (PDF, RTF, etc), only the fallback content appears in the ouput file.

Video

Use the object DITA element to add video to your DITA topics. Example:


video.mp4 (video/mp4)

The XML source code corresponding to the above example is:

```
<p><object data="media/video.mp4" type="video/mp4">
     <param name="source.src" value="media/video.ogv"/>
     <param name="source.type" value='video/ogg; codecs="theora, vorbis"'/>

     <param name="source.src" value="media/video.webm"/>
     <param name="source.type" value="video/webm"/>

     <param name="width" value="320"/>
     <param name="controls" value="yes"/>
     <param name="poster" value="media/video_poster.jpg"/>
</object></p>
```

- The data and type attributes are required. The value of the type attribute must start with "video/".
- It is strongly recommended to specify *alternate video files* as modern web browsers, while all supporting the HTML 5 video element, vary in their support of video formats. This is done by adding pairs of param child elements to the object element. The first param of a pair must have name="source.src" attribute and its value attribute must reference an video file. The second param of a pair must have name="source.type" attribute and its value attribute must contain the media type of the preceding video file.
- It is possible to add param elements corresponding to the attributes supported by the HTML 5 video element (crossorigin, poster, preload, autoplay, mediagroup, loop, muted, controls, width, height). In the above example, we have added a param element corresponding to the width, controls and poster HTML 5 attributes. Note that in the case of HTML 5 *boolean* attributes (autoplay, loop, muted, controls), the value attribute of a param is not significant. For example, in the case of the above example, you could have specified "true", "on", "1", etc, instead of "yes".
- If the object element has a desc child element, then this desc element is used to generate fallback content in case video is not supported. If the object element has no desc child element, then a simple fallback content is automatically generated by ditac. This automatic fallback content basically consists

in a link allowing to download the video file. The `param` element corresponding to the `poster` HTML 5 attribute, if present, is used to generate a nicer automatic fallback content.

- When ditac is used to generate an XSL-FO based format (PDF, RTF, etc), only the fallback content appears in the ouput file.

Flash animation

Use the `object` DITA element to add Adobe® Flash® animations to your DITA topics. Example:

`animation.swf (application/x-shockwave-flash)`

(You may have to right-click on the above screenshot and select **Play** from the Flash popup menu to replay the animation.)

The XML source code corresponding to the above example is:

```
<p><object data="animation.swf"
           type="application/x-shockwave-flash"
           width="431" height="123">
    <param name="movie" value="animation.swf"/>

    <param name="menu" value="true"/>
    <param name="quality" value="low"/>
</object></p>
```

- The `data`, `type`, `width` and `height` attributes are required. The `param name=movie` child element having the same value as attribute `data` is required too.
- You may add any other `param` child element supported by the Flash object. In the above example, you'll find `menu` and `quality` in addition to required `movie`.
- If the `object` element has a `desc` child element, then this `desc` element is used to generate fallback content in case Flash is not supported. If the object element has no `desc` child element, then a simple fallback content is automatically generated by ditac. This automatic fallback content basically consists in a link allowing to download the `.swf` file.
- When ditac is used to generate an XSL-FO based format (PDF, RTF, etc), only the fallback content appears in the ouput file.

Actions

Unless you add `param name="controls"` (see above), you'll not be able to play audio or video. Even worse, without the `controls` `param`, an audio object is not rendered on screen (that is, it is invisible).

A simple solution for this problem is to insert a `<?onclick?>` processing-instruction in a DITA element (typically an *inline* element such as `xref` or `ph`). The `<?onclick?>` processing-instruction allows to specify an number of actions:

`play`
: Play the associated resource from the beginning. Only applicable to video or audio targets.

`pause`
: Pause playing . Only applicable to video or audio targets.

`resume`
: Resume playing . Only applicable to video or audio targets.

`mute`
: Mute sound . Only applicable to video or audio targets.

`unmute`
: Unmute sound . Only applicable to video or audio targets.

`show`
: Set the visibility property of the target element to visible.

`hide`
: Set the visibility property of the target element to hidden.

The above actions are exactly those supported by EPUB 3's `epub:trigger`.

The `<?onclick?>` processing-instruction is processed by **`ditac`** for the following output formats: XHTML 5, XHTML 5 Web Help and EPUB 3. It is discarded for any other output format.

The syntax for the content of `<?onclick?>` is:

```
onclick_data -> action (S action)*
action -> op '(' target_id? ')'
op -> 'play'|'pause'|'resume'|'mute'|'unmute'
      'show'|'hide'
```

When *target_id* is not specified, it is taken from the `href` attribute of the element containing the `<?onclick?>` processing-instruction. For example, `<xref href="#media/target_audio"><?onclick` **`play()`**`?>` is equivalent to: `<xref href="#media/target_audio"><?onclick` **`play(media/target_audio)`**`?>`.

Example 1: Say: "*Viens Hubble!*", which, in French, means: "Come here Hubble!".

No audio. Say: "*Viens Hubble!*", which, in French, means: "Come here Hubble!".

The XML source code corresponding to the above example is:

```
<p>Example 1: <xref href="#media/target_audio"><?onclick play()?>
Say "<ph xml:lang="fr">Viens Hubble!</ph>"</xref>
...
<object data="media/audio.wav" id="audio_sample" type="audio/wav">
  <desc> ... </desc>
</object></p>
```

Example 2: Hide Hubble. Show Hubble.

Figure 5. My name is Hubble. I'm a 7-month old Golden Retriever.

The XML source code corresponding to the above example is:

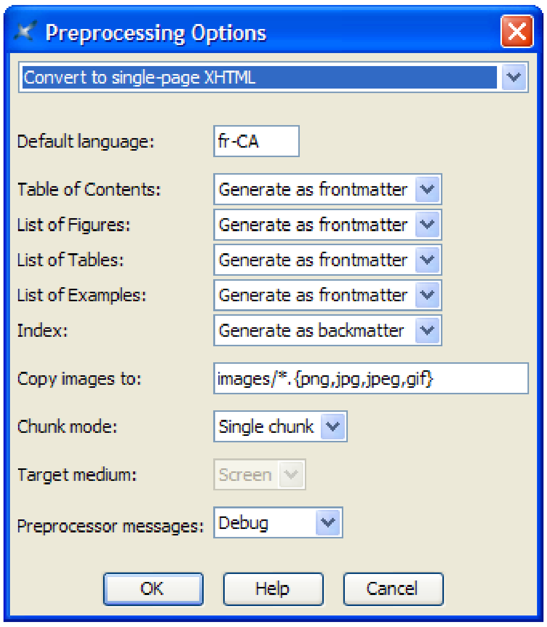
```
<p>Example 2:
<xref href="#media/target_image"><?onclick hide()?>Hide Hubble</xref>.
<xref href="#media/target_image"><?onclick show()?>Show Hubble</xref>.</p>
```

7. Preprocessing options

Converting a DITA document to formats such as HTML, PDF, RTF, etc, comprises two steps. First step consists in preprocessing the DITA document. Second step consists in translating the preprocessed DITA document to the other format by the means of XSLT stylesheets.

The XSLT stylesheets are parameterized by using **Options** → **Customize Configuration** → **Change Document Conversion Parameters**, while the preprocessor is parameterized by using **Options** → **Customize Configuration** → **Preprocessing Options**. The latter menu item displays a dialog box which is described in this section.

*Figure 6. The **Preprocessing Options** dialog box*



Convert to single-page XHTML

Default language:	fr-CA
Table of Contents:	Generate as frontmatter
List of Figures:	Generate as frontmatter
List of Tables:	Generate as frontmatter
List of Examples:	Generate as frontmatter
Index:	Generate as backmatter
Copy images to:	images/*.{png,jpg,jpeg,gif}
Chunk mode:	Single chunk
Target medium:	Screen
Preprocessor messages:	Debug

The top combobox allows to select the group of options to be edited. Each group of options is completely separated from the other. For example, specifying that an index is to be generated as backmatter for group "**Convert to single-page XHTML**" will have an effect when you'll use **Map** → **Convert Document** → **Convert to XHTML [one page]** and no effect at all when you'll use **Map** → **Convert Document** → **Convert to HTML Help** or when you'll use **Topic** → **Convert Document** → **Convert to XHTML [one page]** (because there is a separate "**Convert to single-page XHTML**" group of options for the **Map**, **BookMap** and **Topic** configurations).

Default language

Specifies the main language of the document. Examples: `en`, `en-US`, `fr`, `fr-CA`. This information is needed in order to sort the index entries. By default, this information is taken from the `xml`:lang attribute of the root element of the topic map (if any, "`en`" otherwise).

Table of Contents

Specifies whether to automatically generate a **Table of Contents** and, if a **Table of Contents** is to be generated, where to generate it. *Frontmatter* means at the beginning of the document. *Backmatter* means at the end of the document.

This option, like **List of Figures**, **List of Tables**, **List of Examples** and **Index**, is mainly useful when working with maps or individual topics. When working with a bookmap, the preferred way to specify the location, if any, of a **Table of Contents** is to do it in the bookmap itself. In all cases, what's specified in the bookmap has priority over the value of this option.

List of Figures

Specifies whether to automatically generate a **List of Figures** and, if a **List of Figures** is to be generated, where to generate it.

List of Tables

Specifies whether to automatically generate a **List of Tables** and, if a **List of Tables** is to be generated, where to generate it.

List of Examples

Specifies whether to automatically generate a **List of Examples** and, if a **List of Examples** is to be generated, where to generate it.

Index

Specifies whether to automatically generate an **Index** and, if an **Index** is to be generated, where to generate it.

Copy images to

Copy the image files referenced in the topics to specified directory. If specified path is relative, it is relative to the output directory.

In the above screenshot, "`images/*.{png,jpg,jpeg,gif}`" means:

- copy to directory `images/`, relative to the output directory,
- as is (that is, without having to convert the image to another image format),
- all the images referenced in the document source, having a `png`, `jpg`, `jpeg` or `gif` filename extension.
- Any image referenced in the document source having a filename extension other than `png`, `jpg`, `jpeg` or `gif` (e.g. `svg`, `tif`) will be automatically converted to an image having a `png`, `jpg`, `jpeg` or `gif` filename extension.

When this field is left empty, the generated document will reference the image files using absolute URLs. This is harmless for PDF, RTF, etc, files because at the end of the conversion process, such files will *embed* a copy of the image files. However, this is rarely what is wanted for HTML-based formats (XHTML, Java Help, HTML Help, Eclipse Help, EPUB, etc).

Chunk mode

Allowed values are **Automatic**, **Single** and **None**.

Chunk **Automatic** means: ignore the chunk specification found in the topic map and output a single chunk for the **Print** medium; honor the chunk specification for the **Screen** medium.

Chunk **None** means ignore the chunk specification found in the topic map and output a single chunk. As explained above, chunk **None** is implicit for some formats (PostScript, PDF, RTF, etc).

Both the **None** and **Single** values may be used to force the generation of a single output file. Chunk **Single** allows to reuse a map designed to output multiple HTML pages in order to generate a single HTML file or a PDF file.

Target medium

Explicitly specifies the output medium: **Screen** (XHTML, HTML Help, Eclipse Help, etc) or **Print** (PDF, RTF, etc). By default, the output media is guessed using the extension of the output file.

Preprocessor messages

Specifies the level of verbosity of the preprocessor. Allowed values are (from not verbose to very verbose): **None**, **Information**, **Verbose**, **Debug**.

Some fields may be ``grayed out'' (disabled). This happens in two cases:

1. The DITA configuration has been customized by the local guru. This automatically prevents the end user from making any change to the preprocessing options.
2. Changing the values of some options (e.g. **Target medium**) would break the stock configuration.