XMLmind XML Editor - User's Guide

Hussein Shafie
Pixware

<xmleditor-support@xmlmind.com>

XMLmind XML Editor - User's Guide

Hussein Shafie
Pixware
<xmleditor-support@xmlmind.com>

Publication date July 30, 2010

Abstract

This document is essentially a comprehensive tutorial about XMLmind XML Editor (XXE for short). It also contains installation instructions and information about the contents of the distribution.

Chapter 1. Install

1. Installing XXE

1.1. Requirements

- Sun or Apple Java™ runtime 1.5 or above.
- At least 512Mb of memory and a 1GHz CPU.
- 60Mb of free disk space, 120Mb for a self-contained distribution which includes a Java™ 1.6 runtime.

XXE is officially supported on Windows XP/Vista/7, on Linux 2.6 and on Mac OS X 10.5 (Leopard) and 10.6 (Snow Leopard). It is possible to use it on other Java™ 1.5+ platforms (e.g. Mac OS X 10.4, Solaris), but without support from XMLmind.

XXE has been tested with:

- Sun Java™ runtime 1.5+ (up to 1.6.0_21) under Windows XP/Vista/7, openSUSE Linux 11.1, Ubuntu Linux 9.04 (Jaunty Jackalope).
- Mac OS X 10.5.8 (Leopard), 10.6.4 (Snow Leopard) and the latest Java™ 1.5 runtime or Java™ 1.6 runtime, depending on the version of Mac OS X.
- Apache HTTP server 2.2.3 and its `mod_dav` WebDAV server.

  Also tested against `mod_dav_svn` 1.4 (Subversion) with autoversioning turned on.
- The vsftpd FTP server 2.0.5, with both the FTP and FTPS protocols.
- The OpenSSH SFTP server 4.4p1.
- As an applet, XXE has been tested against: Internet Explorer 6/Windows XP, Firefox 3/Windows XP, Internet Explorer 8/Vista, Firefox 3/Linux. The applet does not work on the Mac whatever Web browser you may use.

1.2. Install on Linux (or *manual* install on the Mac)

Tip

On the Mac, it is strongly recommended to download and install the auto-installable `.dmg` file.

1. Make sure that the Java™ `bin/` directory is referenced in the $PATH and, at the same time, check that the Java™ runtime in the $PATH has the right version:

   ```
   ~$ java -version
   java version "1.6.0_20"
   Java(TM) SE Runtime Environment (build 1.6.0_20-b02)
   Java HotSpot(TM) Server VM (build 16.3-b01, mixed mode)
   ```

2. Unpack the XXE distribution inside any directory you want:

   ```
   $ cd
   $ tar zxvf xxe-perso-4_6_1.tar.gz
   $ ls xxe-perso-4_6_1
   addon/
   bin/
   demo/
   doc/
   ```

3. XXE is intended to be used directly from the `xxe-perso-4_6_1/` directory. That is, you can start XXE by simply executing:

```
$ xxe-perso-4_6_1/bin/xxe &
```

After that, you may want to add `xxe-perso-4_6_1/bin/` to your $PATH.

1.3. *Manual* install on Windows

Tip

On Windows, it is strongly recommended to download and install one of the two auto-installable `setup.exe` files.

1. Make sure that you have a Java™ 1.5+ runtime installed on your machine. To check this, open a command window and type "`java -version`" followed by Enter. You should get something looking like this:

```
C:\> java -version
java version "1.6.0_05"
Java(TM) SE Runtime Environment (build 1.6.0_05-b13)
Java HotSpot(TM) Client VM (build 10.0-b19, mixed mode, sharing)
```

2. Use a tool like WinZip, 7-Zip or Info-Zip [1] to unzip the XXE distribution inside any directory you want:

```
C:\> mkdir XMLmind
C:\> cd XMLmind
C:\XMLmind> unzip xxe-perso-4_6_1.zip
C:\XMLmind> dir xxe-perso-4_6_1
... <DIR> addon
... <DIR> bin
... <DIR> demo
... <DIR> doc
```

3. XXE is intended to be used directly from the `xxe-perso-4_6_1/` directory. That is, you can start XXE by simply executing:

```
C:\XMLmind> xxe-perso-4_6_1\bin\xxe.exe
```

After that, you may want to add a shortcut to "`C:\XMLmind\xxe-perso-4_6_1\bin\xxe.exe`" on your desktop.

Note that the `bin` directory contains not only `xxe.exe`, but also an equivalent `xxe.bat` which may be handy if you intend to customize the way XXE is started.

2. Contents of the installation directory

bin/
: Contains XXE code (`.jar` files) and many scripts used to start XXE and its associated utilities.

bin/xxe, xxe.bat
: Scripts used to start XXE. Use `xxe` on any Unix system. Use `xxe.bat` on Windows.

bin/xxe.exe, xxe.jstart
: Only when installing XXE on Windows using any of the `*setup*.exe` distributions. File `xxe.exe` is XXE launcher and `xxe.jstart` is its (plain text, UTF-8 encoded) associated parameter file.

bin/deploywebstart, deploywebstart.bat
: Scripts used to generate a Java™ Web Start configuration (`.jnlp` file, signed jars, etc) from a possibly customized XXE distribution. Use `deploywebstart` on any Unix system. Use `deploywebstart.bat` on Windows.

[1] Note that Windows XP has built-in support for `.zip` archives.

The `deploywebstart` command-line tool is documented in the Section 3.1, “The deploywebstart command-line tool” in *XMLmind XML Editor - Configuration and Deployment*.

bin/xmltool, xmltool.bat
: Scripts used to run **xmltool**. See *The xmltool command-line utility* for more information about this tool.

bin/csscheck, csscheck.bat
: Scripts allowing to check the syntax of CSS style sheets written for XXE.

bin/convertdoc, convertdoc.bat
: Scripts used to run **convertdoc**. This tool allows to execute XXE process commands from the command line, exactly as if these process commands were executed from XXE.

  See Section 3, “The **convertdoc** command-line tool” in *XMLmind XML Editor - Commands* for more information about this tool.

bin/*.jar
: All the (non-system) Java™ class libraries needed to run XXE:

  - `xxe.jar` contain the code of XXE.

    `xxe_help.jar` contains the online help of XXE.

    `xsc.jar` contains the code of the spell checker engine developed by XMLmind.

  - `jh.jar` is the standard Java™ help engine.

    `xerces.jar` contains Xerces 2.9.1 XML parser. (The version included in the Java™ runtime 1.6 has bugs which have been fixed in bundled version.)

    `resolver.jar` contains Apache XML Commons Resolver which implements catalog-based entity and URI resolution.

    Substantial parts of `xsdregex.jar`, James Clark's XSD to Java Regular Expression Translator, have been directly added to `xxe.jar` (which is why file `xsdregex.jar` is not included in the distribution). Download original package from http://www.thaiopensource.com/download/.

    Package `com.jclark.xsl.expr` contains the implementation of XPath 1.0 used by XT, James Clark's XSLT engine. A modified version of this package, renamed `com.xmlmind.xmledit.xpath`, has been directly added to `xxe.jar` (which is why file `xt.jar` is not included in the distribution). Download full XT from http://www.jclark.com/xml/xt-old.html or from http://www.blnz.com/xt/index.html.

    `relaxng.jar` is Jing version 20030619, James Clark's RELAX NG validator, slightly modified for use in XXE. The details of the modifications are found in `relaxng.README`.

    `saxon.jar` is Michael H. Kay's XSLT 1 engine. See http://saxon.sourceforge.net/.

    `saxon9.jar` is Michael H. Kay's XSLT 2.0 engine. See http://www.saxonica.com/.

    These *excellent* packages have *not* been developed by XMLmind. Copyright information is contained in the corresponding `.LICENSE` file. Read the corresponding `.README` file to have more details about these packages.

bin/legal/, legal.txt
: Contains legal information about XXE and about third-party components used in XXE.

bin/icons/
: Contains desktop icons for XXE.

bin/mac/
: Contains files (e.g. `Info.plist`) used on the Mac to create `XMLEditor.app` from a `.zip` or a `.tar.gz` distribution.

addon/
: The `addon/` directory is the place where XXE finds its extensions, whatever its type: configurations, plug-ins, translations to languages other than English, spell-checker dictionaries.

  This `addon/` directory is recursively scanned by XXE at startup time. Therefore, feel free to organize it as you want.

addon/config/
: Contains configuration files for a few document types: DocBook, DITA, XHTML, etc.

  The content of a configuration file, which specifies a customization of XXE for a specific XML application, is described in detail in XMLmind XML Editor - Configuration and Deployment.

addon/spell/
: Contains the bundled dictionaries used by the spell-checker. A dictionary is a file whose name is *`LL`*`.dar`, where *`LL`* is an ISO code for a language.

  The distribution of XXE only contains an English dictionary (with CA, GB, US variants). Additional dictionaries, German, French, Spanish, can be downloaded and installed using Options → Install Add-ons.

doc/
: Contains XMLmind XML Editor documentation in HTML and PDF (Acrobat) formats.

  Note that:

  doc/user/
  : Contains this User's Guide in HTML and PDF (Acrobat) formats.

    The source of the User's Guide is also available in DocBook format (`userguide.xml`) in case you want to open it in XXE.

demo/
: Contains XML documents that can be opened in XXE to demo some of its features.

3. Acknowledgments

On Windows, XMLmind XML Editor installer (i.e. `*setup*.exe`) is built using Inno Setup™ by Jordan Russell's software. XMLmind highly recommends this excellent and free-to-use tool.

Chapter 2. Tutorial: basics

What you'll not learn by reading this tutorial

This tutorial will *not* teach you how to use the most nifty features of XXE:

- how to use the **Ins** key or the **Enter** key to quickly insert text nodes and elements,
- how to change an image displayed in the document view by dragging and dropping a graphics file onto it,
- how to quickly insert or move columns by using the table editor,
- etc.

Most of these smart tricks are listed in the chapter Being productive with XXE [30]. Other features are specific to the type of the document being edited (XHTML, DocBook, DITA, etc), and are therefore documented in the online help which comes with the XXE configuration associated to the document type.

No, this tutorial will just teach you the *most basic way* to do things. On the other hand, what you'll learn here will work for any type of document, and not only for XHTML.

This tutorial assumes that the reader has a minimal knowledge of XML (that is, the reader knows what is an element, an attribute, etc) and a minimal knowledge of HTML or XHTML (that is, the reader knows that `p` is the tag for a paragraph, that `ul` is a list, etc).

It is recommended that, using XXE, the reader repeats each action described in this tutorial. (On Mac, use the Command key instead of the Control key, except for **Ctrl**+Tab and **Ctrl**+Space.)

1. Starting XXE

- If you have installed an auto-installable Windows distribution (`xxe-perso-4_6_1-setup.exe`), XXE can be started by double-clicking on the icon of `xxe.exe` or by using the "XMLmind XML Editor" shortcut added to the Start menu.

  If you have installed a zipped distribution, XXE can be started by typing `xxe` (or `xxe.bat`) from the command prompt, optionally followed by the name of one or several XML documents.

  Example:

  ```
  C> xxe C:\xxe-perso-4_6_1\doc\user\userguide.xml
  ```

- On Unix, XXE can be started by typing `xxe` from an xterm, optionally followed by the name of one or several XML documents.

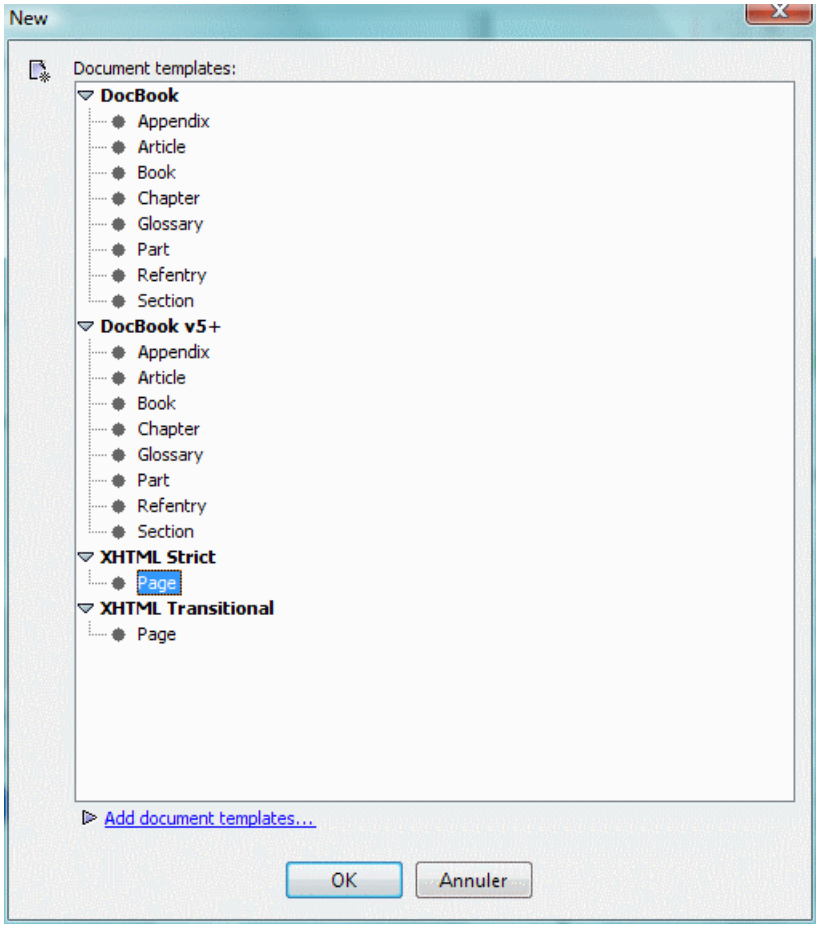
  Example:

  ```
  $ xxe &
  ```

2. Creating a new document

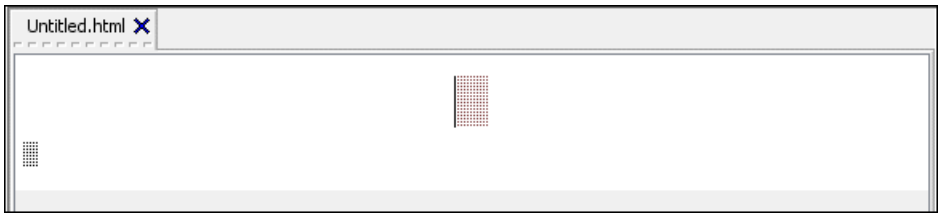
Use File → New and choose a document template from the following dialog box.

Figure 2.1. The File → New dialog box

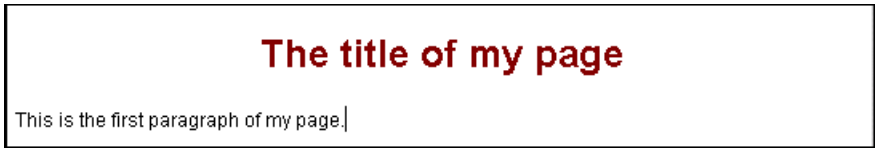


In this tutorial, we have chosen to create a XHTML page conforming to the strict DTD.

The newly created document (`Untitled.html`) looks like this:



The ``blobs" are placeholders for text. Click on the first placeholder and type the title of your XHTML page. Click on the second placeholder and type a few words.



Click again anywhere on the title and then anywhere in the first paragraph. You'll notice that, at the top of the window, just below the tool bar icons, the *node path bar* changes its label.

The node path bar shows where the *caret* (also called insertion cursor) is.

The node path bar does so by displaying the name of the node (anonymous text node `#text`) containing the caret preceded by the name of each of its ancestors elements in the containment hierarchy (element `p` contained in element `body` contained in element `html`).

3. Inserting elements

In this section, we'll get familiar with the following commands:

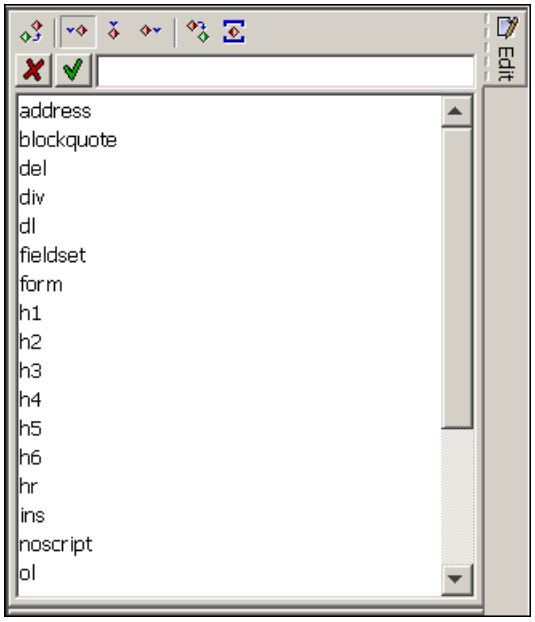
1. Insert before element
2. Insert inside element, at caret location
3. Insert after element

These commands are accessible using the Edit menu bar menu, using the Edit tool ``tab" or using keyboard shortcuts, but in this tutorial we'll limit ourselves to using the Edit popup menu. Therefore the notation Edit → Insert means "click anywhere in the document view using the right mouse button and select menu item Insert in the Edit popup menu".

Figure 2.2. The Edit popup menu

Click anywhere inside the paragraph and execute Edit → Insert Before. The keyboard focus is given to the Edit tool. This pane now lists all the elements you can insert before a `p` and therefore, is ready to use.

Figure 2.3. The Edit tool listing all elements you can insert before a `p`



Select `h1` from the list (a single click on the list item is sufficient). The newly inserted element has a red border around it. Do not care about that, just type the text of the heading and the red border will go away.

Click inside the `p` before a word and execute Edit → Insert, the same element chooser dialog pops up listing all the inline elements you can insert inside a `p`. Select `strong` from the list and type a few words in bold font.

Your document should look like this:

This time, use Edit → Insert After and add a `ul` (unordered list) after the `p`. Type the text of the first list item (`li`).

If you are in the `strong` element -- check it with the node path bar -- click in the `p` outside the bold words because inserting an `ul` inside a `p` after a `strong` is not allowed in XHTML.

Click to get rid of the red border, then using what you have learned, add two more `li`s: one before the first `li` you have created, the second after it.

4. Selecting elements

Editing commands are applied to the selection.

XXE supports three types of selection:

- The text selection, found in all text editors and word processors.
- The node selection, which can contain one or several nodes.
- The implicit element selection.

Some commands can be applied to any type of selection, for example: Edit → Convert. Other are more restrictive, for example: Edit → Insert Before can be applied to the implicit element selection or to an explicit single node selection.

4.1. The implicit element selection

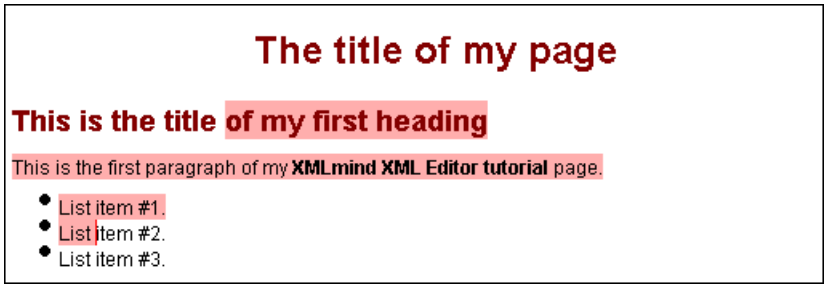
The element containing the caret is implicitly selected. Therefore you can apply commands to it without making any special effort.

This is one of XXE's nicest features, even if it needs to be learned because it has no equivalent in the word processor world.

In the first section of this tutorial, you have already used the implicit element selection to insert an `h1` before the `p` and an `ul` after it.

4.2. The text selection

Selecting text in XXE is no different from selecting text in any text editor. Click in the middle of the `h1` and drag the mouse across the document until the middle of the second `li` is reached.



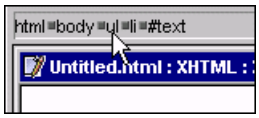
4.3. The node selection

4.3.1. Most general method

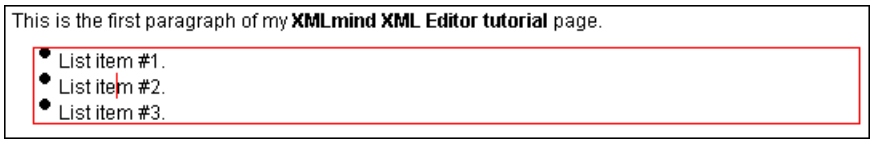
Now we want to add a `pre` (preformatted text) after the `ul`.

We have learned that clicking inside the a `li` implicitly selects it therefore we know how to add a new `li` after this one but how to select its parent, the `ul`, to add a `pre` after it?

For that we need to use *explicit element selection*. Click inside any `li` and then click on the word `ul` inside the node path bar.



This explicitly selects the corresponding node in the document view.



Note that explicitly selected nodes are drawn with a red border around them.

Alternatively, click inside any `li` and type **Ctrl**+Up 3 times: first time to select the `#text` node, second time to select the `li`, third time to select the `ul`.

Now that `ul` is selected, use Edit → Insert After to add a `pre` after it.

Type 2 lines of text in this `pre`. Do not be worried by the ``carriage return icon" always displayed by XXE if the last character of a `#text` node is a newline character.

4.3.2. Directly selecting a node

Clicking near some text moves the caret inside the text and no node is explicitly selected.

Instead of simply clicking, try to *Ctrl-click* on the first `p`. A `#text` node is explicitly selected and the node path bar tells you which one.

Selecting a `#text` node is rarely what you want to do. You'll often need to Ctrl-click again without moving the mouse. This will select to parent of the selected `#text`, that is, the `p`. Ctrl-clicking again without moving the mouse would select the parent of the `p` which is a `body` and so on. When doing a series of Ctrl-clicks, always look at the node path bar to know precisely where you are. Also, do not Ctrl-click several times too fast otherwise the editor will think you are double-clicking or triple-clicking and therefore, selecting elements that way would not work.

Click on the first `p` to cancel the selection because we are going to study another method to select it. Now try to Ctrl-click, not inside the text itself, but *in the blank space at the right of the text* of the first `p`. Notice that one Ctrl-click is sufficient to directly select the whole `p`.

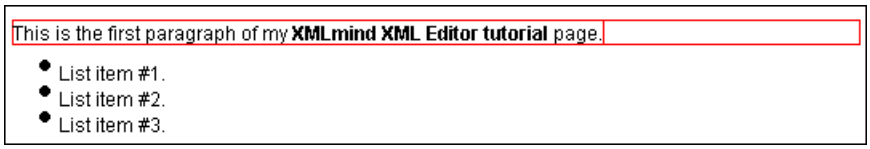
Similarly, you can directly select a node when a non-editable ``decor" has been generated for it. For example, simply *click on the bullet* of a `li` to select it. This type of non-editable decor is very common, for example: the image displayed for a figure, the number of a section, the border of a table, a node icon or element name in the tree view, etc.

4.3.3. Selecting a node range

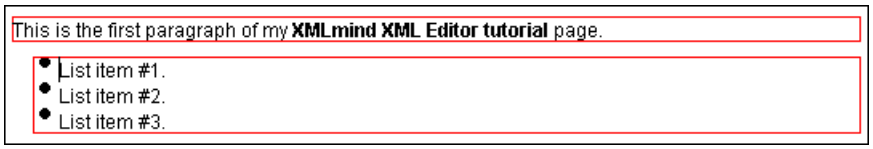
What if you want to put first `p` and `ul` inside a `blockquote`? The answer is select both of them first and then use Edit → Convert (described below in this tutorial).

With XXE it is possible to select a *node range*, that is, adjacent children of the same parent element.

Select first `p` using any of the methods described above.



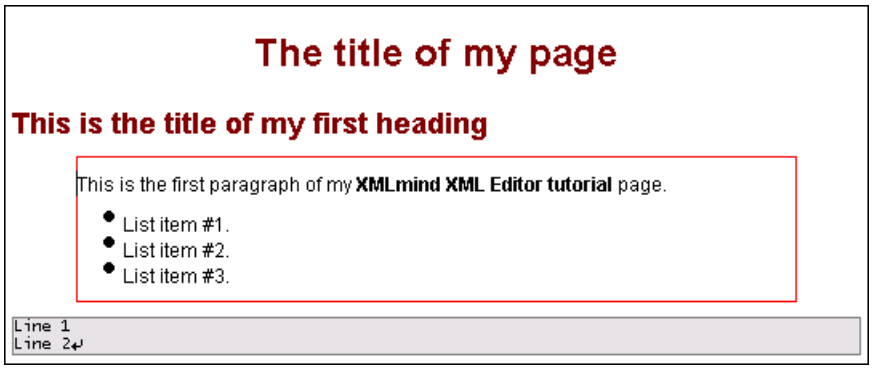
Then *Ctrl+Shift-click* anywhere on the `ul`.



Note that extending the node selection is easy because you do not have to be precise. The reason is that there is no ambiguity about what to select: a `#text` or a `li` is not a sibling of first `p`, only the `ul` is a sibling of first `p`.

Alternatively, you could have typed **Esc** then Right to extend node selection to the `ul`.

Now use Edit → Convert and choose `blockquote`.



5. Navigating through elements

5.1. Using Tab to go from a `#text` to the other, just like in a form

The Tab key may be used to move the caret from the current `#text` to the beginning of the next one. **Shift**+Tab moves the caret from the current `#text` to the beginning of the previous one.

Click on the `title` and use Tab and **Shift**+Tab to move the caret from one `#text` to the other.

What if you really want to insert a Tab character into a `pre` (or any element which allows Tab characters to be inserted in the text flow just as any ordinary character)? The answer is: type **Ctrl**+Tab instead of Tab.

5.2. XXE makes a difference between the end of a `#text` node and the beginning of the `#text` node next to it

The simplest way to move the caret is of course to use the Left or Right arrow keys.

Click on the first `p` to the left of the `strong` and press on the Right key to move the caret in the direction of the `strong`.

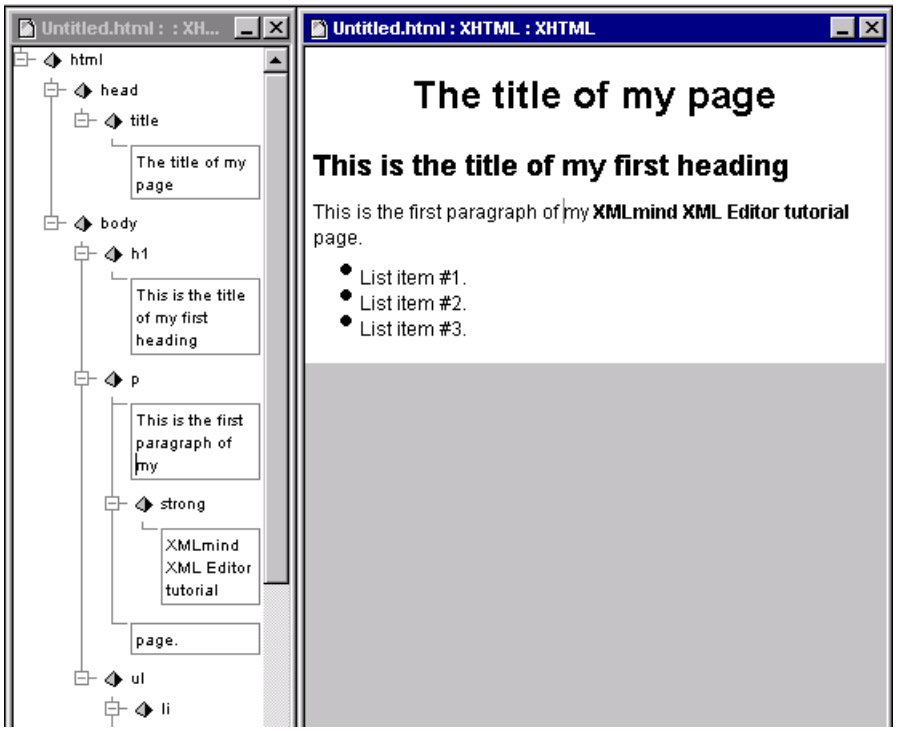
Just before reaching the `strong` element, you'll notice that pressing on the Right key has caused no perceptible caret movement. Then after this ``dead'' Right key, the caret seems to move as expected.

Go back to the left using the Left arrow and at the `p`/`strong` boundary, you'll notice a ``dead'' Left key then the caret seems to move as expected.

Note that on the `p`/`strong` boundary, the caret makes no visible movement but the node path bar displays different paths (`html·body·p·#text` and `html·body·p·strong·#text`).

Use menu command View → Add to add a low-level, hierarchical, view called the tree view, at left of the styled view.

Figure 2.4. Both tree and styled Views side by side



Repeat what you have done with the Right and Left arrow keys and you'll notice that with the tree view, there is a visible caret movement from the end of the `p·#text` to the beginning of the `strong·#text` and vice-versa.

Close the tree view by clicking anywhere inside it and by using View → Close.

This ``dead'' key behavior also occurs when using the **Del** and Backspace keys.

Click on the first `p` to the left of the `strong` and type on the **Del** key several times. Notice what happens on the `p`/`strong` boundary. Then use Edit → Undo as many times as needed to undo this typing.

Click inside the `strong` and type on the Backspace key several times. Notice what happens on the `p`/`strong` boundary. Then use Edit → Undo as many times as needed to undo this typing.

6. Copy, cut, paste, delete

6.1. Copy, cut, paste, delete applied to the text selection

The text selection is used not only to specify a range of characters but also a range of nodes and child elements contained in a common ancestor element.

When applied to the text selection, the Edit → Copy menu command copies all the characters and nodes in the specified range to the system clipboard. It is then possible to paste these characters and nodes in any other application including XXE itself.

Select the characters displayed in bold font from the first `p`, copy them to the clipboard using Edit → Copy, click on the `title` and paste the copied characters using the Edit → Paste.

After copying the text selection, you'll notice that, near the *Clipboard Content* button, the status bar displays `#text`: a piece of text is stored in the clipboard.

Note that if you have not been precise when selecting text, the status bar may display `strong` (you have copied the whole `strong` element to the clipboard) or it may display `[2]` (you have copied 2 nodes to the clipboard: the `strong` element and a piece of text next to it).

Edit → Cut and Edit → Delete basically work the same: they delete all the characters and nodes in the range specified by text selection. The only difference is that Edit → Cut copies the characters and nodes in the specified range to the clipboard.

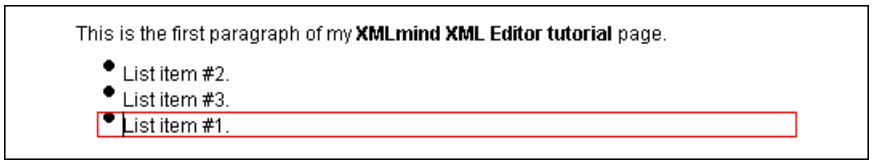
Of course, elements and nodes are deleted only if the DTD or schema constraining the document allows to do so. If this is not the case, selected characters are removed from such elements and nodes and that's it.

6.2. Copy, cut, paste, delete applied to the node selection

The Copy, Cut, Paste, Delete commands can be applied to a node range or to the implicitly selected element.

Two more Paste commands are available: Paste before and Paste after. These commands can only be applied to a single explicitly selected node or to the implicitly selected element.

Select the first `li` (implicit selection is fine for that). Use the Edit → Cut to move it to the clipboard. Select the last `li`. Use the Edit → Paste After to paste the `li` stored in the clipboard after the last `li`.



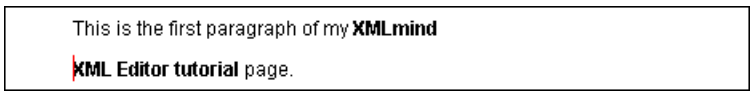
After copying the first `li`, you'll notice that the status bar displays `li`: a list item element is stored in the clipboard.

Note that when the clipboard contains a node range, the status bar displays the number of nodes of the range. For example, copy all list items to the clipboard and you'll see:

7. Splitting and joining elements

7.1. Simple Split and Join

Click in the middle of the `strong` contained in the first `p`. *Type Enter*. The `p` is split in two parts, each part being a `p`, as expected in any word processor.



Now *type Backspace* at the beginning of the second part. The two parts are joined to form our original single paragraph.

Typing Del at the end of the first part would have given the same result.

This is also a handy method for inserting elements. Type Enter at the end of the `p`.

Type some text in the newly created paragraph. Type Enter at the beginning of newly created `p`. This creates another `p` before it.

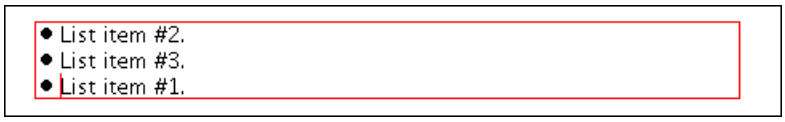
Use Edit → Undo three times to undo the creation of last two paragraphs.

7.2. Split and Join generalized

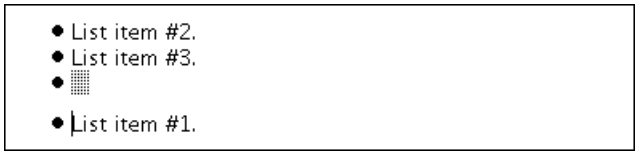
The Split and Join commands bound to Enter, Backspace or Del keystrokes are very handy but they can only be applied to paragraphs.

What if we want to split the list in two parts in order to insert a paragraph before the second part?

Move the caret at the beginning of the last list item and explicitly select the whole list using the node path bar.



Now execute Edit → Split. This command splits the explicitly selected element at caret position, which gives us two adjacent lists.



The generalized form of Join is the inverse command of Split. The Join command joins the explicitly selected element to its *preceding* sibling, an element of same type.

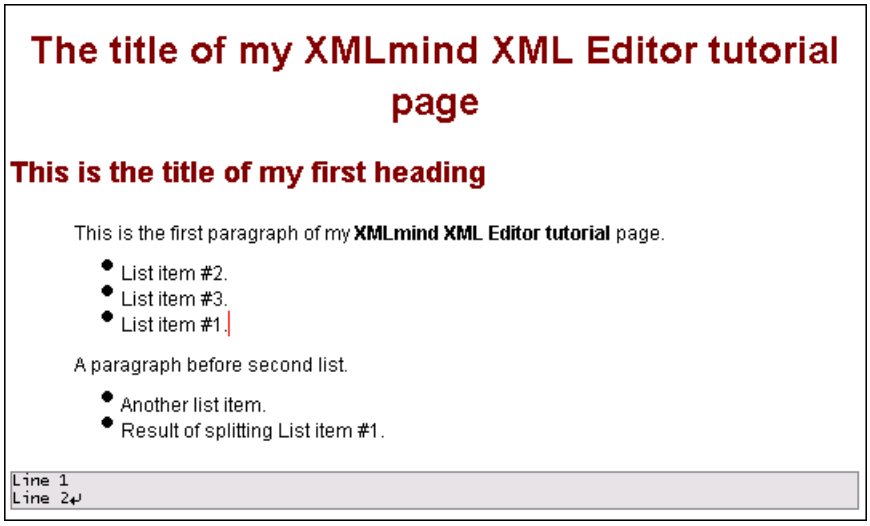
Note that the generalized form of Split and Join are the only commands that cannot be applied to the implicitly selected element.

Explicitly select the second `ul` and execute Edit → Join. The two adjacent lists are now joined to form our original single list.

Undo last Join because we really want to split the list in two parts.

- Add a `p` before the second `ul`, type some text in it.
- Copy text "List item #1" (as usual, by dragging the mouse and by pressing Ctrl-C) and paste it (by pressing Ctrl-V) in the empty last `li` of first `ul`.
- Add another `li` before the only `li` of second `ul`, for example by clicking in the `li`, using Edit → Copy and then using Edit → Paste Before.
- Change all the text of the two list items of the second `ul`, for example by selecting "List item #1" as usual and by typing a few words.

Our XHTML document now looks like this.

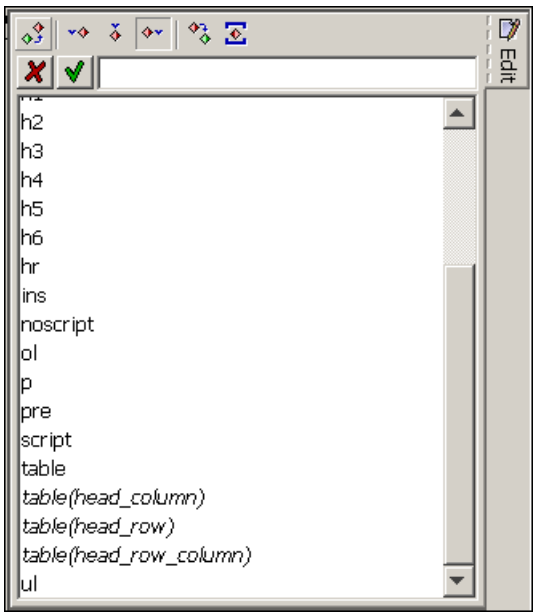


8. Replacing elements

The Edit → Replace command is equivalent to deleting the node selection or implicitly selected element and inserting a new element or `#text` which replaces the deleted nodes.

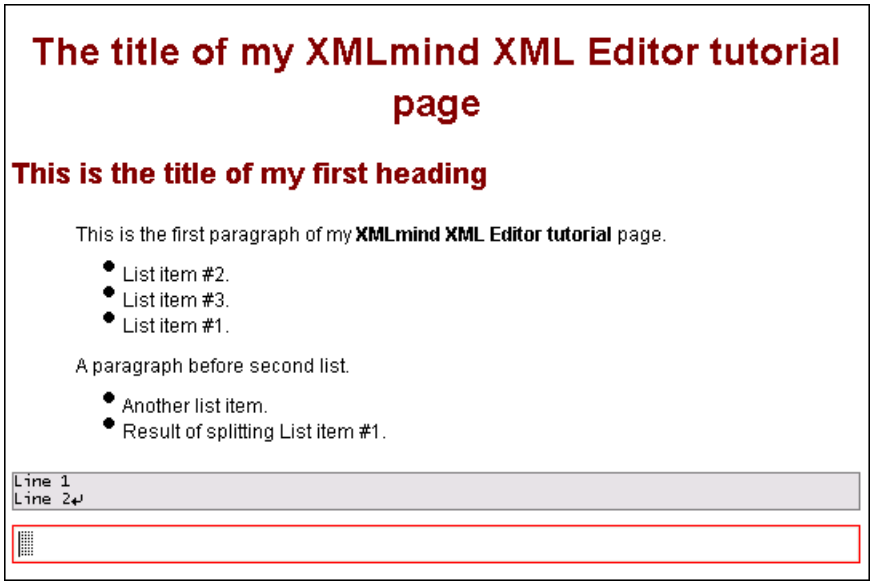
This command is useful because it is often not allowed to delete the selection: doing so would create an invalid document.

Select the `pre`. Insert a `table` after it.



table(head_column), ..., table(head_row_column) are preconfigured *table templates*. Most of the time, you'll choose one of these but for this exercise, we'll choose table which is the simplest table possible in XHTML.

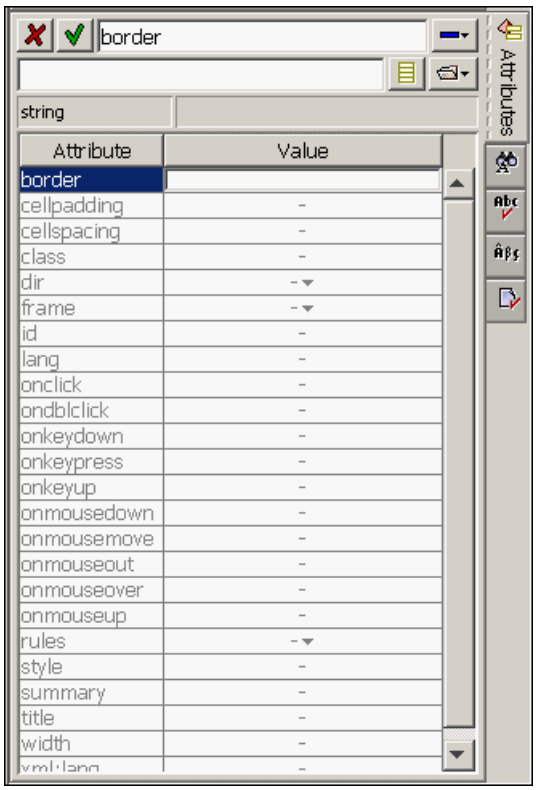
The table is created with a single `tr` (row) containing a single `th` (a ``heading'' cell where text is displayed using a bold font). This is often what you want for the skeleton of `table` but not always.



This very simple table has no borders, even if the footprint of a cell is displayed using a very light gray. This would make the screenshots of this tutorial hard to read. Therefore we'll immediately add a border to this table.

Select the `table` (using the node path bar) if it is not already the case. The Attributes tool should be already displayed, otherwise click on the corresponding tab.

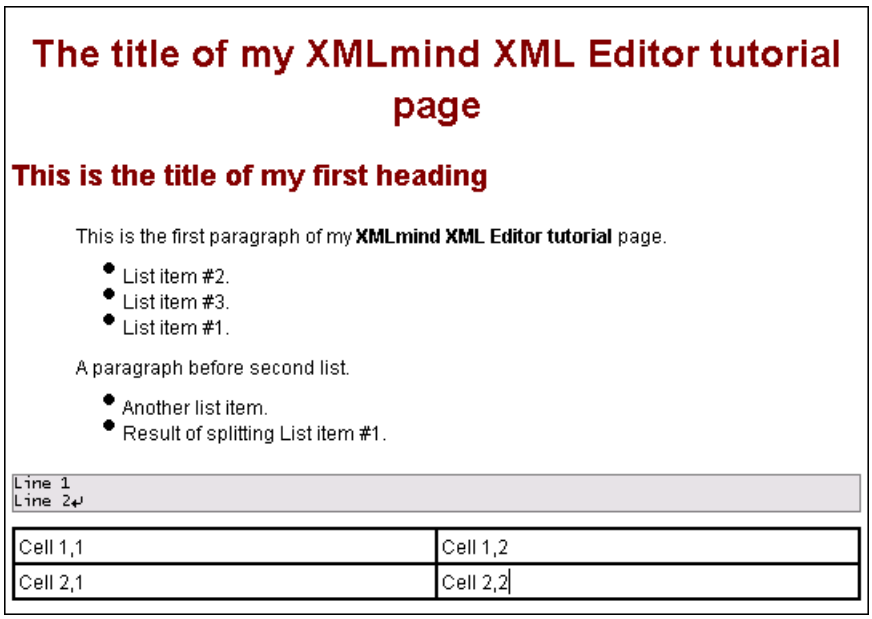
First row of the attribute table is for attribute `border`. Click inside the attribute value cell at the right of the attribute name cell containing `border`, type `1` and press Enter.



Now the table has a nice black border. (We'll learn how to use the Attributes tool later in this tutorial.)

Exercise:

1. Select the `th` contained in the `tr` (implicit selection is fine for that). Using Edit → Replace, replace it with a `td` (a plain cell).
2. Select this `td`. Use Edit → Insert After to add another `td` after it.
3. Using the node bar path, select the `tr` containing the two cells . Use Edit → Copy to copy it to clipboard.
4. Use Edit → Paste After to paste a copy of the `tr` after currently selected `tr`.
5. Type some text in each cell.

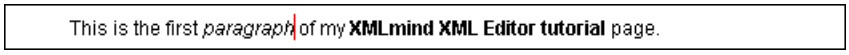


9. Converting elements

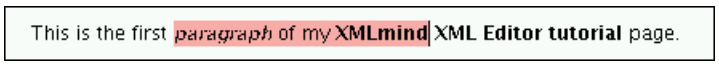
9.1. Convert applied to the text selection

Select a non-bold word in the first p.

Use the Edit → Convert to convert it to em (emphasis).



Remember that text selection is used not only to specify a range of characters but also a range of nodes and child elements contained in a common ancestor element. This feature is very useful when doing a conversion. For example: select text from word "`paragraph`" to word "`XMLmind`".



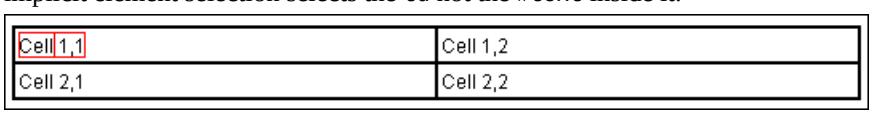
Use the Edit → Convert to convert it to `button`.

Use Edit → Undo to undo the last conversion.

9.2. Convert applied to the node selection

Converting a `#text` node rather than an equivalent text selection is often more convenient.

Select the `#text` node contained in the first cell of the table. Note that explicit node selection is needed to do that: implicit element selection selects the `td` not the `#text` inside it.



Used Edit → Convert to convert it to em.

Cell 1,1	Cell 1,2
Cell 2,1	Cell 2,2

Unlike Edit → Replace which creates an *empty* new element, Edit → Convert transfers the content of the selection to the new element which is the result of the conversion.

More precisely, in the case of the node selection:

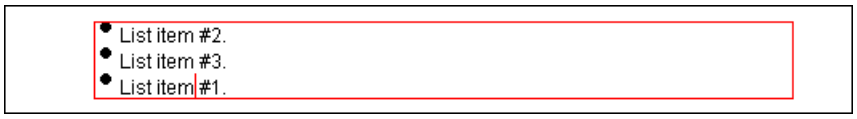
- When several nodes or a single non-element node are selected, all these nodes are given a new parent element which is the result of the conversion.

  We have already seen two examples of this behavior. First one is when we ``wrapped" a `p` and a `ul` into a `blockquote`. Second one is just above.

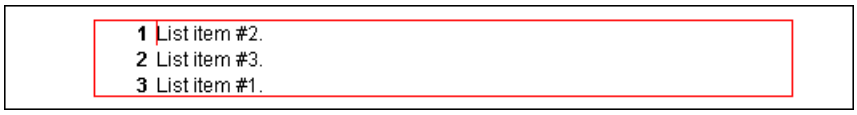
- When a single element is selected, all its children (but not its attributes) are transferred to the result of the conversion.

  What follows is an example of this second behavior. Here we want to ``morph" single selected element which is an `ul` to an `ol`.

Select the first `ul`.



Used Edit → Convert to convert it to an `ol` (ordered list).

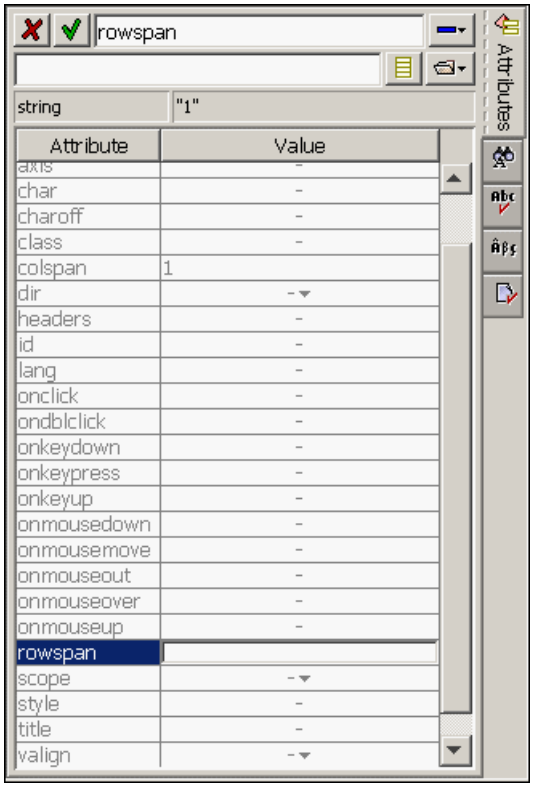


This operation is valid because the `ul` parent, a `body`, accepts `ul`s as well as `ol`s at this place and because the element content of this `ul` is ``compatible" with an `ol`.

10. Editing element attributes

Use Tools → Edit Attribute (**Ctrl+E**) to edit the attributes of the selected element. This action displays and gives the keyboard focus to the Attributes tool. Alternatively, if this pane is already displayed, you can click inside it and use it right away.

Figure 2.5. The Attributes tool with the `rowspan` attribute of a `td` being edited



There are two methods for adding or changing the attributes of the (explicitly or implicitly) selected element:

1. Using the attribute form (the upper side of the Attributes tool).This should be the method of choice for persons who prefer to use the keyboard.

2. Using the attribute table (the lower side of the Attributes tool). This should be the method of choice for persons who prefer to use the mouse.

The ``minus" button of the attribute form can be used to remove an attribute. Removing an attribute directly from the attribute table is possible too: simply right-click on the attribute row and use the displayed popup menu.

The content of the attribute table can be described as follows:

- All attributes set for the selected element are displayed by the table.

- All possible attributes for the selected element, *even those not set*, are also displayed by the table.

  Attributes which have not been set are displayed in gray. Attributes which have been set are displayed in black.

- Attributes are listed sorted in alphabetical order.

- The names of required attributes are displayed using a bold font.

- The names of fixed attributes are displayed using an italic font.

Exercise:

- Select second `td` in the `table`. Set its `align` attribute to `center`.

- Set its `rowspan` attribute to `2`.

- Set its `valign` attribute to `middle`.

- Add an extra `td` after it to make the `table` look more balanced.

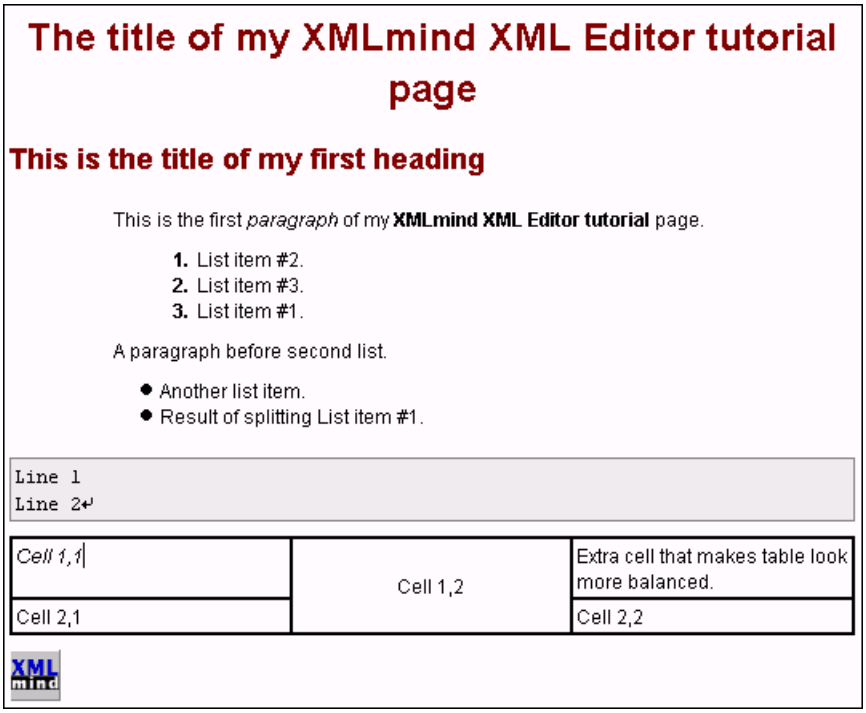
Cell 1,1	Cell 1,2	Extra cell that makes table look more balanced.
Cell 2,1		Cell 2,2

10.1. Required attributes in newly created elements

It is important to remember that, by default, XXE automatically gives a placeholder value ("???") to required attributes of newly created elements. This means that you have to replace this placeholder value by the actual one as soon as the element has been created.

Add a `p` after the `table` using Edit → Insert After and insert an `img` in it using Edit → Insert.

The `img` has two required attributes `src` and `alt`. XXE has set those attributes to string `???`. Use the Attributes tool to give these attributes an actual value.



For this `img`, we used *`XXE_install_dir`*`/doc/user/tutorial/xxe.gif` and `XMLmind logo` for `alt` (directory *`XXE_install_dir`*`/doc/user/tutorial/`, generally `C:\Program Files\XMLmind_XML_Editor\doc\user\tutorial\`, contains all the files used in this tutorial) .

The useless `#text` node has been removed by explicitly selecting it and using Edit → Delete.

11. Checking document validity

You cannot check the validity of a document without an associated DTD, W3C XML Schema or RELAX NG schema. When editing a document not constrained by a grammar, XXE guarantees that what you'll create will be well-formed without making any special effort.

This section, like most of this tutorial, describes the behavior of XXE when editing documents constrained by a grammar.

Checking document validity is automatically performed each time you save your document.

Unless you use Edit → Force Deletion, XXE will never allow editing commands that would make the document *structurally* invalid (a document where some elements have invalid child elements or attribute names).

Therefore explicitly checking document validity is rarely needed. You may have to use the Tools → Check Validity command when:

1. You have loaded an invalid document and you are fixing it. After each editing command, you want to know if it is fixed now.

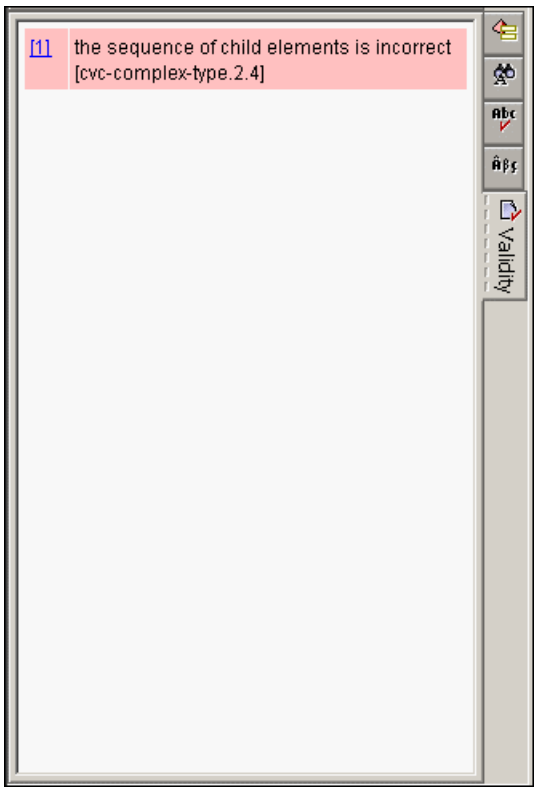
2. XXE creates elements where the *value* of required attributes, if any, is invalid: unless configured differently, the required attributes are given `???` as a placeholder value.

After explicitly or implicitly checking document validity, its validity status is displayed at the left of the status bar.

Figure 2.6. After forcing the deletion of all the items of a list, "Untitled.html" is structurally invalid.

The Validity tool ``tab" displays validity error messages if any.

Figure 2.7. Validity error message displayed after forcing the deletion of all the items of a list.



The color of the message reflects the severity of the error. Clicking on the number of an error message selects the element where the validity error was found.

Chapter 3. Tutorial: creating a modular document

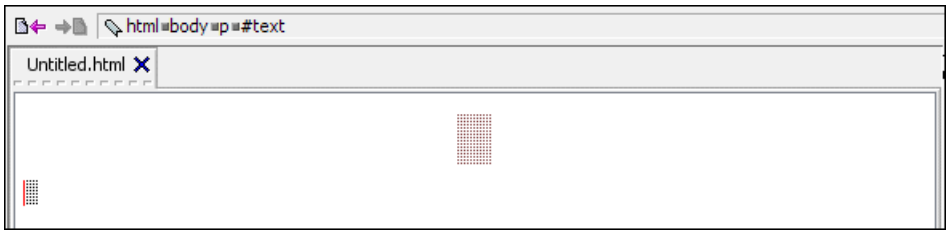
A *modular document* is a document which not only has its own content but which also includes XML nodes contained in other documents.

Let's say you want to add the same copyright information at the bottom of all the XHTML pages you create. The first step is to author the copyright information in a separate document we'll call `Copyright.html`. The second step is to include this copyright information at the bottom of the XHTML page created during this tutorial.

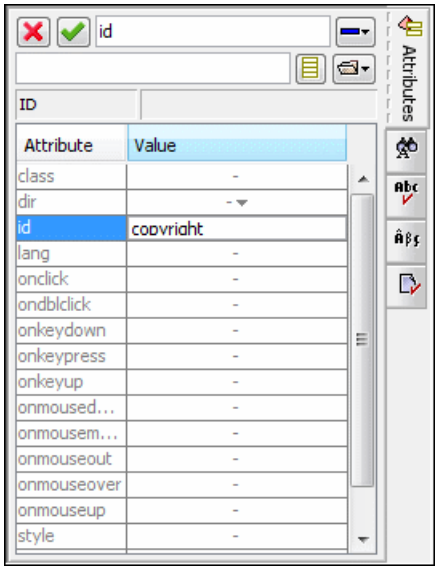
1. Creating the Copyright.html document

Use File → New and once again choose XHTML Strict/Page as the document template.

Using Edit → Replace, replace the inner `p` by a `div`.



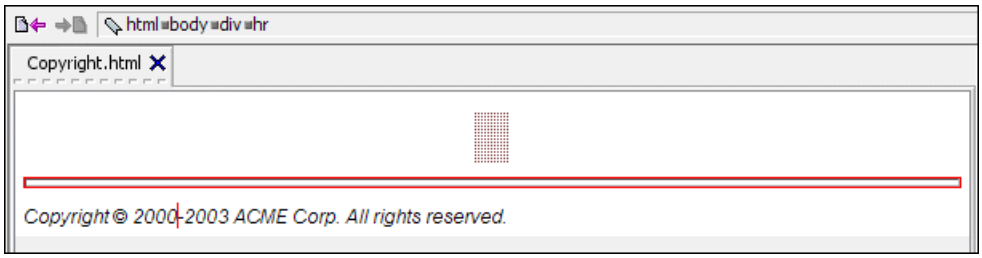
Using the Attributes tool, specify `copyright` as the `id` attribute of the newly insert `div` element.



Then select the text node which is the default content of the `div` by clicking on `#text` in the node path bar and replace it by an `address` element. Finally, type the copyright information. (The copyright sign © has been inserted using the Characters tool.)

Using Edit → Insert Before, insert a `hr` (horizontal rule) before the `address` to visually separate the copyright information from the body of the XHTML page.

Using File → Save As (or File → Save which behaves like File → Save As for a new document), save this document as `Copyright.html` in a place where it can be shared by all the XHTML pages you'll create.



2. Inserting a reference to Copyright.html into the XHTML page

The procedure to do this is very simple. It is similar to copying an element in a document and pasting it in another document:

1. Open in XXE the document containing the element you want to reference. If the document containing the element you want to reference is already opened in XXE, simply display its window.

2. Explicitly select the element you want to reference.

3. Use menu bar menu Edit (not popup menu Edit), select sub-menu Reference and choose entry Copy as Reference (shortcut **Ctrl**+**Shift**+**C**).

   Unlike the usual Copy command which copies XML data to the clipboard, this special command copies to the clipboard a *reference* to an element (that is, a *pointer* to the element).

   Note that unless selected element has an ID attribute or is the root element of the document, Copy as Reference will not work.

4. Now switch to the window containing the modular document.

5. Select the element where you want to insert the reference.

6. Use one of the standard Paste Before, Paste or Paste After commands to insert the reference into the modular document. As always, the Paste Before, Paste or Paste After commands are enabled only if the DTD or schema constraining the modular document allows it.

Now let's apply this procedure to our example:

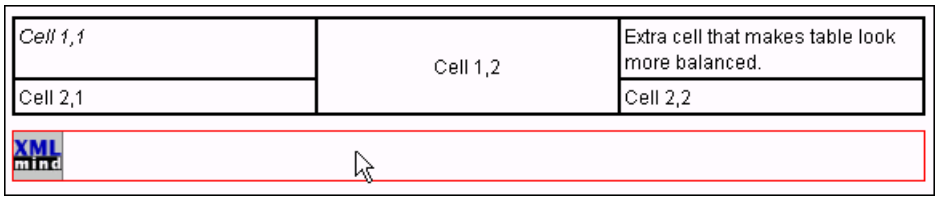
1. The window containing `Copyright.html`, the document we want to reference, is already displayed. There is nothing to do in step #1.

2. We want to reference the `div` contained in `Copyright.html`, therefore we need to select it. For example, click on word `div` in the node path bar to do this.

3. Use **Ctrl**+**Shift**+**C** to copy to the clipboard a reference to the `div`. Notice that at the bottom of XXE window, near the "View Clipboard Content" icon, the word `div` is displayed using a dimmed color. This means that the clipboard contains a reference rather than ordinary data.

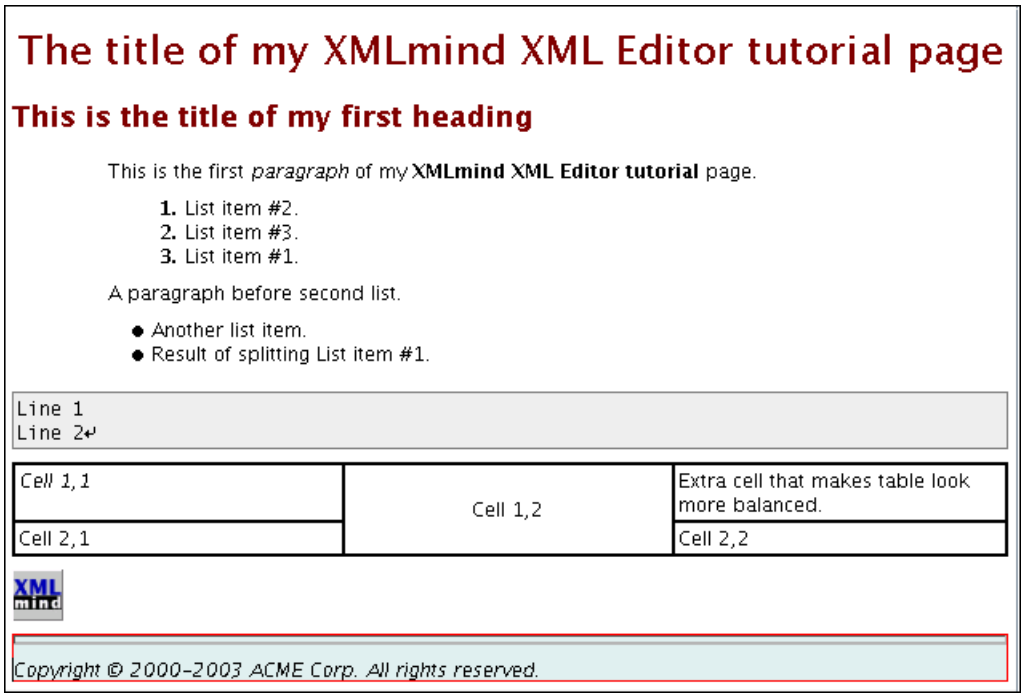
4. Switch to the window containing our XHTML page, for example by clicking on the tab having "`Untitled.html`" as its title.

5. Select last `p` at the bottom of the page, for example by simply clicking at the right of the XMLmind logo. (There are many quick ways to select nodes using the keyboard or the mouse: they are all explained in the next chapter of this user's guide: being productive with XXE [30].)



6. Use command Paste After (shortcut **Ctrl**+**W**) to paste the reference to the `div` after selected `p`.

The XHTML page now looks like this.



Notice that the copyright information inserted at the bottom of the page is displayed with a light blue-gray background. This is used to indicate that this part of the document has been included from an other document and that,consequently, *it cannot be edited in this window*.

Click anywhere inside the copyright information and try to type some text: no characters are inserted. (You can still delete, cut or replace the whole copyright information: it is the `div` which is not editable, not its parent `body`.)

Also notice that the node path bar displays non-editable nodes using a dimmed color.

At the left of the node path bar, the button with a right arrow switches to the window of the included document. Try it and you'll be able to edit `Copyright.html`.

Once having `Copyright.html` in front of you, use the button at the left of the node path bar showing a left arrow to go back to the XHTML page created during this tutorial.

3. Extensive use of the "Copy as Reference" command

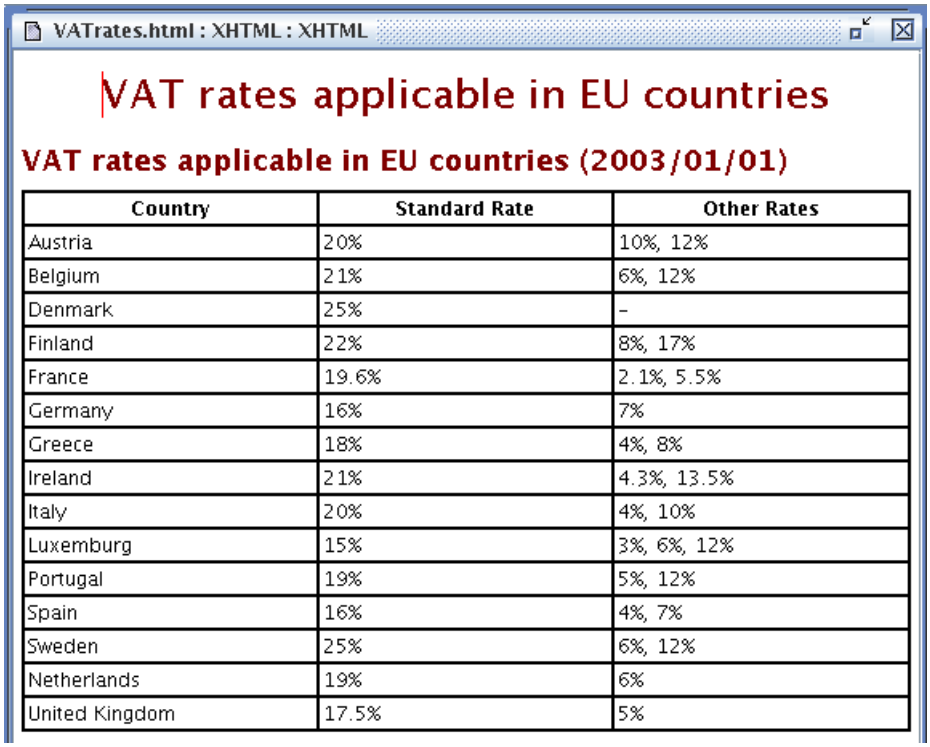
Let's suppose you need to write an article about taxes, but you don't want to type the values of the different VAT rates used in European countries directly in your document because you know that these VAT rates are about to change. Let's say that you already have an XML document detailing these VAT rates.

(In this tutorial, creating the VAT rates document from scratch would be tedious which is why you'll find `VAT-rates.html` in *`XXE_install_dir`*`/doc/user/tutorial/`. You'll also find in this directory: `Untitled.html`, the XHTML page we are trying to create, `Copyright.html` and `xxe.gif`. That is, all the files used in this tutorial.)

Add a new `p` at the end of our XHTML page and type in it ``The VAT rate of France is higher than the VAT rate of Germany.".

Now we need to use `VATrates.html` to insert the numerical values of the VAT rates in our new paragraph.

1. Open `VATrates.html` in XXE.



Country	Standard Rate	Other Rates
Austria	20%	10%, 12%
Belgium	21%	6%, 12%
Denmark	25%	-
Finland	22%	8%, 17%
France	19.6%	2.1%, 5.5%
Germany	16%	7%
Greece	18%	4%, 8%
Ireland	21%	4.3%, 13.5%
Italy	20%	4%, 10%
Luxemburg	15%	3%, 6%, 12%
Portugal	19%	5%, 12%
Spain	16%	4%, 7%
Sweden	25%	6%, 12%
Netherlands	19%	6%
United Kingdom	17.5%	5%

2. Click inside the cell which contains the VAT rate of France to implicitly select it, then use **Ctrl**+**Shift**+**C** to copy it as a reference.

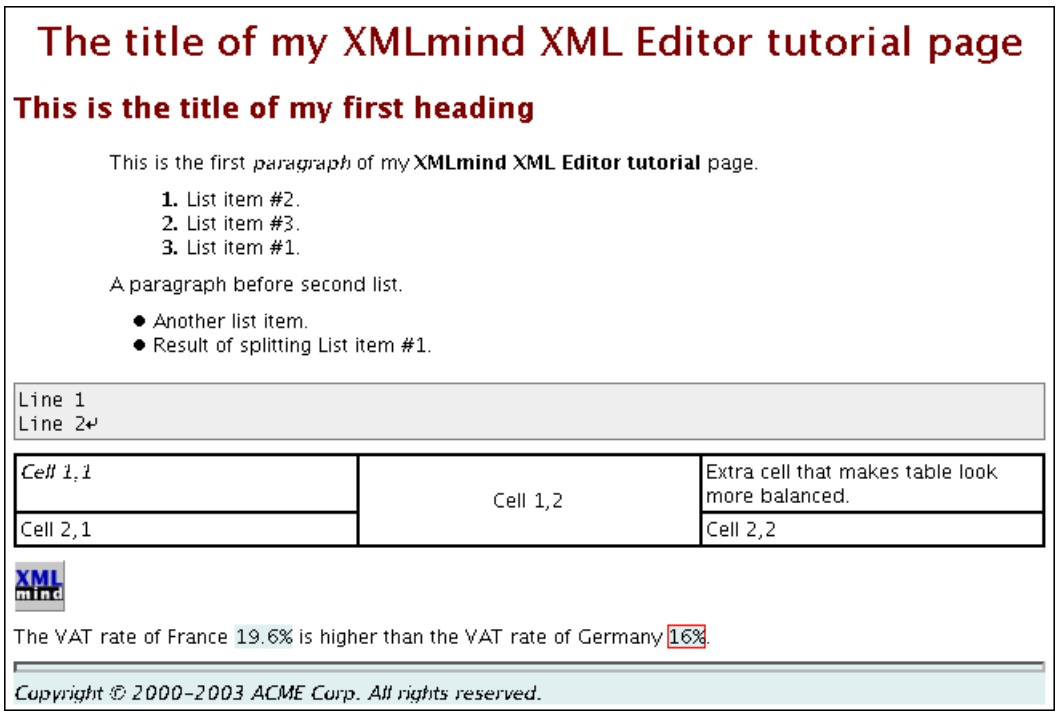
3. Switch to the window displaying our XHTML page: `Untitled.html`.

4. Move the caret after ``The VAT rate of France" .

5. Paste the copied reference using **Ctrl**+**V**.

Do the same for the VAT rate of Germany.

The XHTML page now looks like this.



This has worked smoothly because:

a. Each table cell of `VATrates.html` contains a single `span` element rather than plain text. A `span` element can be inserted almost anywhere in an XHTML document.

b. Each `span` of second column is identified using an `id` attribute: `austria_vat`, `belgium_vat`, etc.

Now type this sentence at the end of last added p: ``Note that the VAT rate of France is lower than the VAT rate of Italy." and insert the value of the VAT rate of Italy as explained above.

Inserting the VAT rate of France in second sentence is easier because we have already pasted it in our XHTML page. In such case, there is no need to switch to the window displaying `VATrates.html`:

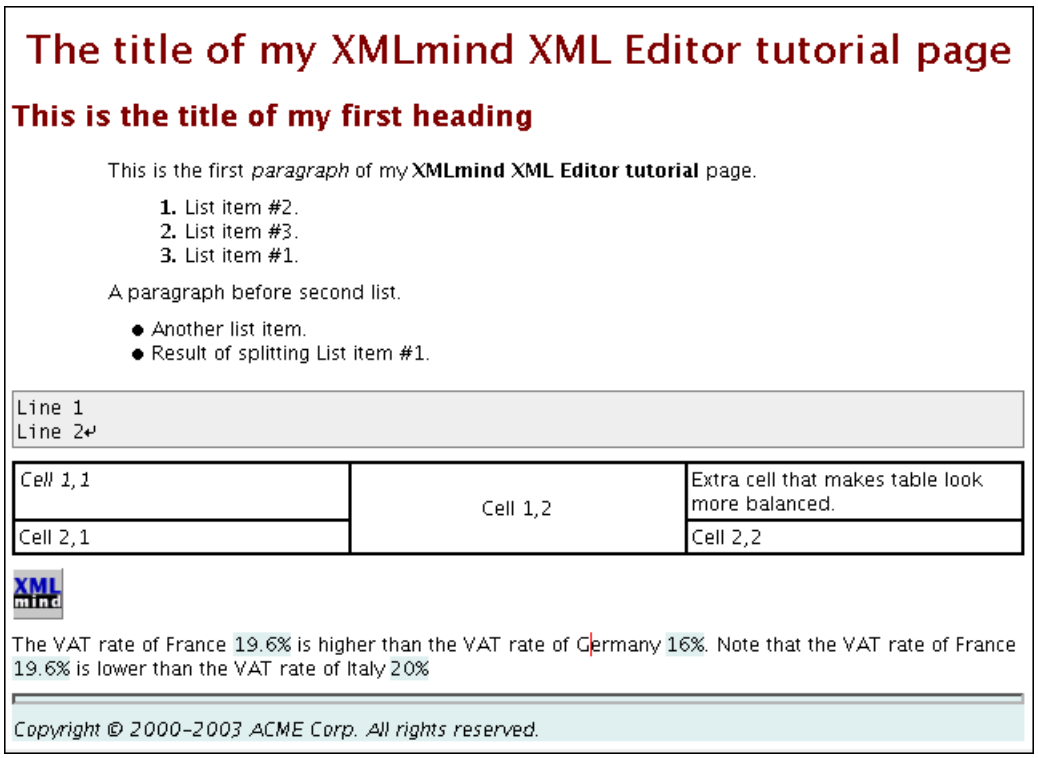
1. Select the `span` previously pasted in our XHTML page which contains the VAT rate of France. For example, implicitly select it by clicking inside it.

2. **Ctrl**+**C** to copy it normally.

3. Click in second sentence after ``Note that the VAT rate of France".

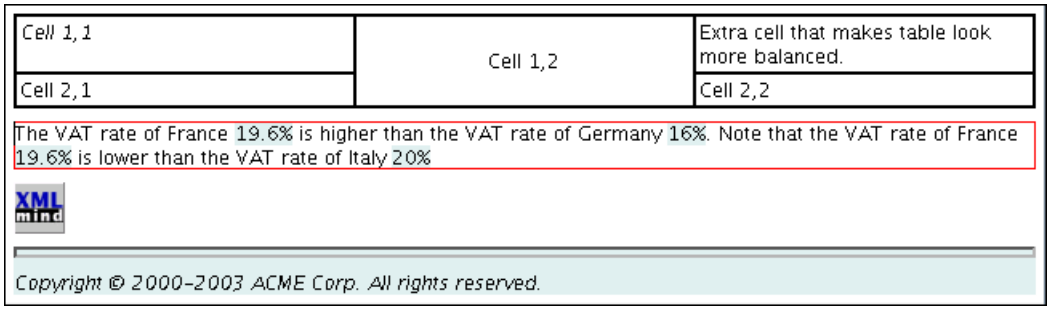
4. Paste the copied reference using **Ctrl**+**V**.

The XHTML page finally looks like this.



Note that it is not possible to copy a reference as a reference. Only an ``original" element can be copied as reference. That's why we used Copy (**Ctrl**+**C**) here and not Copy As Reference (**Ctrl**+**Shift**+**C**). In fact, the standard Cut, Copy and Paste, Paste After, Paste Before commands all know about included elements and will take care of preserving the fact that they are references to elements contained in other documents.

For example, use Cut (**Ctrl**+**X**) and Paste Before (**Ctrl**+**U**) to move the p containing all the VAT spans before the p containing the XMLmind logo. You'll get this:



4. Creating modular documents which are highly interchangeable with other applications

Creating modular documents in XXE should work very smoothly, without much limitations. Converting such modular documents to other formats, within XXE, using its "Convert document" menu[1], should work equally well.

Despite the fact that XXE *exclusively uses a standard mechanism* (XInclude), naively creating modular documents may lead to interchange problems with third-party applications.

These problems come from two situations:

A. The application does not support XInclude at all. Example: the Saxon XSLT 1 processor.

[1] The "Convert document" menu is present only in XMLmind XML Editor Professional Edition.

B. The application supports XInclude, but with limitations. Example: the xsltproc XSLT processor.

The workarounds are:

Situation A: no support for XInclude
: Use a preprocessor to create a flat copy of the document before submitting it to the application which does not support XInclude. XMLmind XML Editor comes with such preprocessor. You'll find `xmltool` in the *`XXE_install_dir`*`/bin/` directory.

Situation B: limited support for XInclude
: Avoid anything which, even remotely, looks like a recursive inclusion.

  For example, even if this complies with the standard and is very well supported by XXE, avoid to reference in a modular document, an element contained in the modular document itself (e.g. you want the same tip to appear at different places).

Chapter 4. Being productive with XXE

1. Do not use the tree view

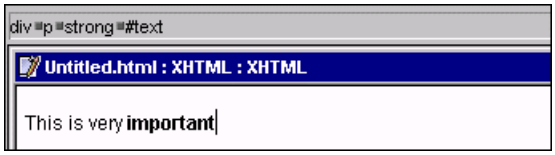
Do not use the tree view. You can do everything efficiently in the styled view. If this is not the case, then you have missed something in the tutorial.

The tree view has been designed to allow editing documents for which a style sheet has not yet been written.

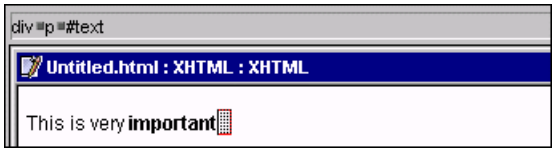
2. Quickly type plain text after a bold or italic element

Use the **Insert** key to insert a `#text` node after the explicitly selected node or implicitly selected element.

This is often handy in the following situation: you have typed some text in a paragraph then inserted a `strong` for which you have typed text.



Then how to quickly continue typing plain text after the `strong`? The answer is: use the **Insert** key of your keypad.



Also note that **Shift**+**Insert** inserts a `#text` node *before* the explicitly selected node or implicitly selected element.

Mac Users

There is no **Insert** key on the Mac. Instead, you need to use the **F1** function key.

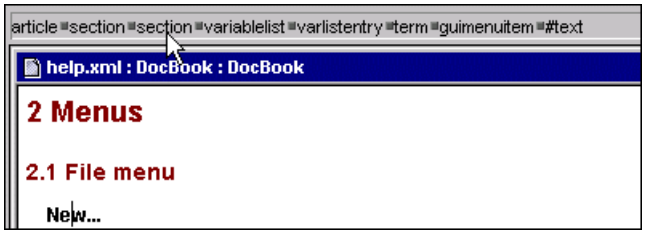
3. Quickly insert after an element, another element of the same type

- Use **Ctrl**+**Insert** to insert an element of the same type after the explicitly or implicitly selected element.

  Example: you want to quickly add a section after current section. Select the section using the node path bar and type **Ctrl**+**Insert**.

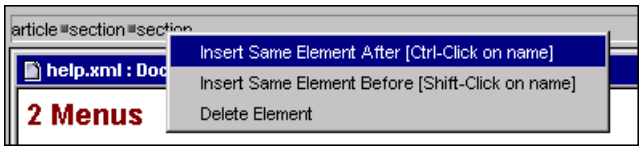
  Also note that **Ctrl**+**Shift**+**Insert** inserts an element of the same type *before* the explicitly or implicitly selected element.

- Even quicker than previous tip, **Ctrl**+click in the node path bar on the name of the element you want to duplicate.



Also note that if you **Shift**+click in the node path bar on the name of the element, this selects this element and then creates a new element of same type *before* this element.

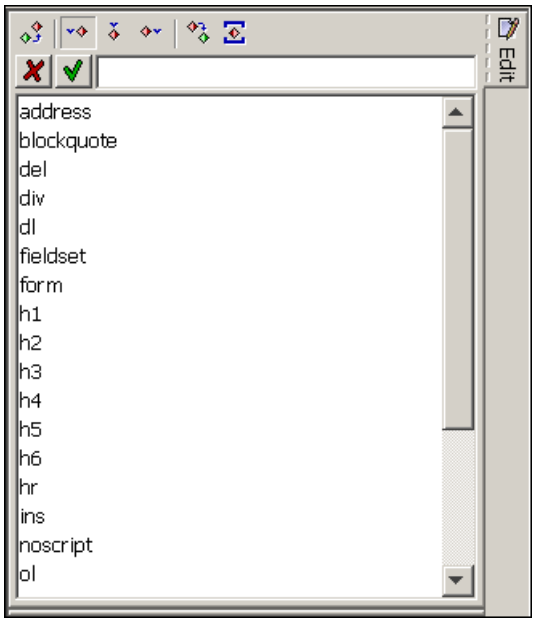
Remembering these two tips is easy, the node path bar has a contextual menu which is displayed when you click using the right mouse button.



- Most XXE configurations (XHTML, DocBook, DITA, etc) bind keystrokes **Ctrl**+**Enter** and **Ctrl**+**Shift**+**Enter** to the following actions:
  - **Ctrl**+**Enter** pressed anywhere inside a paragraph or a list item (i.e. any commonly used, repeatable, element) inserts a new paragraph or a list item after it.
  - **Ctrl**+**Shift**+**Enter** pressed anywhere inside a paragraph or a list item inserts a new paragraph or a list item before it.

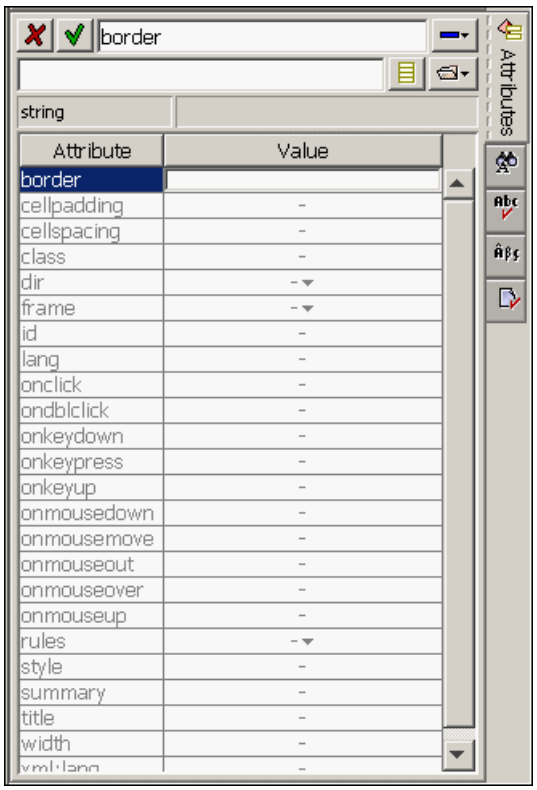
4. Use auto-completion as much as possible

- When choosing an element using Edit tool, type the element name (or "(text)" to specify a text node) in the text field instead of clicking on it in the element list. This text field supports *auto-completion*.



User preferences related to auto-completion can be changed using the Preferences dialog box, Edit section.

- When adding an attribute to an element or when changing the value of an attribute, use the attribute form rather than the attribute table:



1. Type the name of the attribute in the name field (first field of the form).
2. Press **Enter** to move to the value field (second field of the form).
3. Type the value of the attribute in the value field.
4. Press **Enter** to commit the change and to give the keyboard focus back to the document view.

Both the name and value fields support auto-completion. However auto-completion in the value field only works for attributes having the following types: any enumerated type, `ID`, `IDREF`, `IDREFS`.

This auto-completion feature can be configured using the Preferences dialog box, Edit section.

5. When possible, apply commands to node ranges

Use the keyboard to select a node range:

- **Esc** Down Arrow (that is: type **Esc**, then type Down Arrow) selects all child nodes of the implicitly or explicitly selected element.
- **Esc** Right Arrow extends node selection to following sibling.
- **Esc** Left Arrow extends node selection to preceding sibling.

Note **Esc** Right Arrow (and **Esc** Left Arrow) will first select element containing caret if there is no explicit node selection, therefore typing **Esc** Right Arrow several times is often the quickest way to select a node range.

6. Learn important keyboard shortcuts

Command	Keyboard shortcut	Trick to remember the keyboard shortcut
Undo	**Ctrl**+**Z**	Standard shortcut
Redo	**Ctrl**+**Y**	Standard shortcut

Command	Keyboard shortcut	Trick to remember the keyboard shortcut
Repeat	**Ctrl**+**A**	**A** like **A**gain
Cut	**Ctrl**+**X**	Standard shortcut
Copy	**Ctrl**+**C**	Standard shortcut
Paste	**Ctrl**+**V**	Standard shortcut
Paste Before	**Ctrl**+**U**	**Ctrl**+**V** means Paste and **U** is before **V**
Paste After	**Ctrl**+**W**	**Ctrl**+**V** means Paste and **W** is after **V**
Delete	**Ctrl**+**K**	**K** like **K**ill
Replace	**Ctrl**+**R**	**R** like **R**eplace
Insert	**Ctrl**+**I**	**I** like **I**nsert
Insert Before	**Ctrl**+**H** (**Cmd**+**B** on the Mac)	**Ctrl**+**I** means Insert and **H** is before **I**
Insert After	**Ctrl**+**J**	**Ctrl**+**I** means Insert and **J** is after **I**
Convert	**Ctrl**+**T**	**T** like **T**ransform
Wrap	**Ctrl**+**Shift**+**T**	**T** like **T**ransform (variant of Convert)
Split	**Esc Enter**	A paragraph-specific form of Split is often bound to the **Enter** keystroke
Join	**Esc** Backspace	A paragraph-specific form of Join is often bound to the Backspace keystroke
Search	**Ctrl**+**F**	Standard shortcut **F** like **F**ind
Replace	**Ctrl**+**M**	**M** like **M**odify
Find Next	**Ctrl**+**G**	Standard shortcut

Mac Users

- Use the **Cmd** key instead of the **Ctrl** key, except for **Ctrl**+Tab (insert tab character) and **Ctrl**+Space (insert non breaking space).
- Use **Ctrl**+button1 to emulate mouse button3.
- Use **Alt**+button1 to emulate mouse button2.

Note that using a 2-button mouse with a scroll wheel on the Mac is not only very well supported but also highly recommended.

Tip

A quick reference card in PDF format is available in four different flavors: A4, Letter, for the Mac/A4, for the Mac/Letter.

Tip

Use XXE add-on manager (Options → Install Add-ons) to download and install the add-on containing a sample `customize.xxe` file. This customization file contains many *truly useful* macro-commands and their associated bindings.

7. Quickly paste selected text using mouse button #2

Use the mouse to quickly copy and paste text:

- Selecting text automatically ``copies it as system selection'' on platforms supporting system selection (X-Window) and automatically copies it to an internal clipboard on other platforms.
- Clicking with mouse button #2 (middle button or mouse wheel) pastes the content of system selection on platforms supporting system selection and pastes the content of an internal clipboard on other platforms.

Note that unlike Edit → Copy which copies characters as well as nodes, selecting text this way just copies *characters* to the system selection.

This functionality is disabled by default. To enable it, please use the Preferences dialog box, Edit section and check "Clicking with middle button pastes system selection".

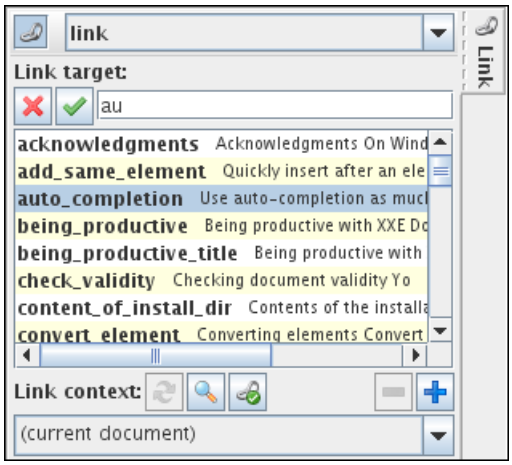
8. Use the Link tool to quickly create navigation links

Note

The Link tool is hidden by default. You need to enable it by checking "Enable the Link Tool" in Options → Preferences, Features section.

Using the Link tool, creating navigation links[1] is as quick and easy as (DocBook example):

1. Select some text.
2. Press **Ctrl**+**Shift**+**X** (shortcut of Edit → Add or Change Link). The keyboard focus is moved to the Link tool which is activated and now displays the list of all possible targets, given the selected *link context*.



3. Type the first few characters of the chosen link target. Press SPACE to auto-complete the link target it if needed to.
4. Press ENTER when the desired link target is unambiguously selected. Doing this creates a new `link` element wrapping the selected text.

While very convenient to use when editing a monolithic document, the Link tool really shines when it comes to creating navigation links in the context of a modular document (modular DocBook `book`, DITA `map` or `bookmap`, etc). More information in Section 6.2, “Link tool” in *XMLmind XML Editor - Online Help*.

9. Drop graphics files on `img`, `imagedata`, `image`, etc, elements

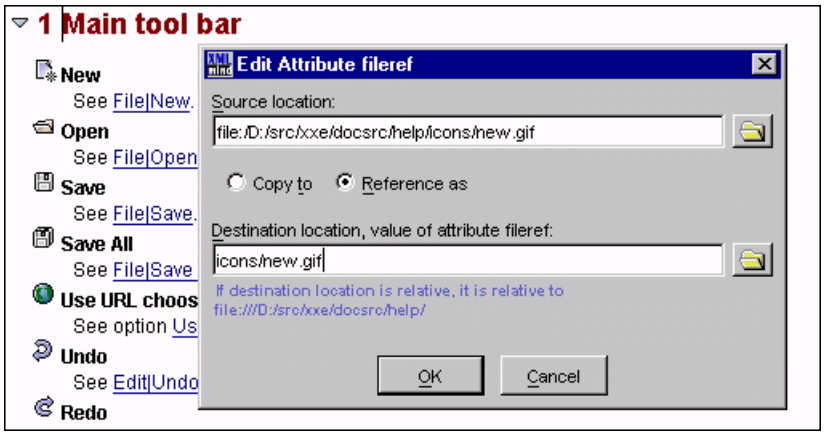
When an image is displayed in a styled document view, the simplest way to change it is to drag and drop a file on it.

[1]Examples: DITA xref and link elements, DocBook xref and link elements, XHTML a element.

When a file is dropped on an image, a pre-filled, specialized dialog box is displayed to let the user specify exactly what he wants to do with the image file.

For example, if the image file is referenced in attribute `fileref` of element `imagedata` (like in screenshot below), the user is given the choice between

- copying the image to the document directory and then updating the reference to the image file in the attribute,
- OR just updating the reference to the image file in the attribute.



The same dialog box can be opened without having to drop a file on an image displayed in the document view: simply double-click on the image.

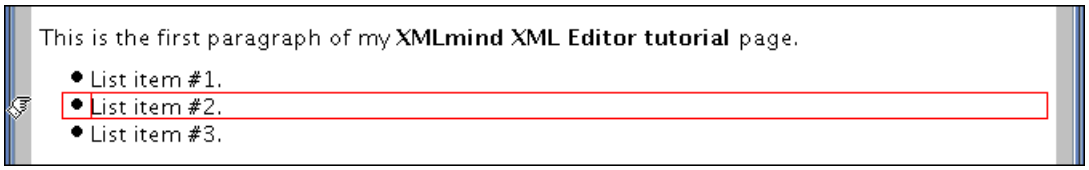
This functionality has been implemented mainly to allow users to upload images to the remote site when they edit documents stored on a FTP or WebDAV server, but this is also very handy when working on the local file system.

10. Easily select paragraphs, list items, table rows, etc, using the ``interactive gray margins''

The easiest way to select a paragraph or a table row is to use the option which adds ``*interactive gray margins*'' at the left and/or at the right of the document view.

By default these interactive margins are absent. To enable them, you need to use Options → Preferences, Edit section and check one or both the "Add interactive margin to the styled view" toggles. After doing this, you'll also need to reload any opened document.

Example, using interactive margins to select an `ul` in an XHTML document:



Move the mouse in the gray margin found at the left and/or at the right of the document view. Here you'll notice that the cursor changes its shape. Move the cursor in front of any list item and click once. Mouse clicks in the left or in the right margin selects the ``block'' (paragraph, row, row group, table) which is in front of the click. In this case, this selects the `li`. Click again without moving the mouse and this will select the parent of the `li`: the `ul`. Clicking again without moving the mouse would select the parent of the `ul`: the `body`, and so on. Do not click several times too fast otherwise the editor will think you are double-clicking or triple-clicking and therefore, selecting elements that way would not work.

Note

Because this way of selecting blocks is so easy to use, it tends to ``cannibalize'' the other ways of selecting nodes, especially the implicit element selection. Therefore we do not recommend turning on this option.

11. Quickly insert ``XML variables'' in your document using the Include tool

Note

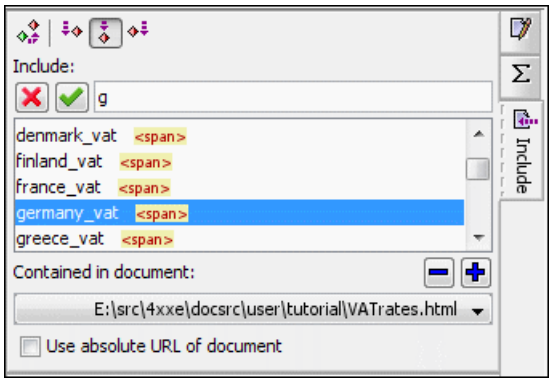
The Include tool is hidden by default. You need to enable it by checking "Enable the Include Tool" in Options → Preferences, Features section.

You'll often want to put all the ``XML variables'': product names, product versions, copyright information, addresses, phone numbers, etc, you use in all your documents in a single, special purpose, document and then, paste references to these ``XML variables'' in your actual documents.

By working this way, if one day, the value of an ``XML variable'' changes, you don't need to manually update all the documents making use of this value.

In Extensive use of the "Copy as Reference" command [25], we have already explained that this is done by copying an element from the document containing the collection of ``XML variables'' (using Copy As Reference — **Ctrl**+**Shift**+**C**) and then pasting the reference in the actual document (using Paste — **Ctrl**+**V**).

However if you need to do that one hundred times a day, you'll quickly find very tedious switching from one document to the other. Fortunately, the Include tool has been specially designed to handle the case of ``XML variables''.



Let's suppose you want to use the Include tool to insert in your report the VAT rate of Germany (same example as in Extensive use of the "Copy as Reference" command [25]). Here's what you'll have to do:

1. Use the ✚ button to specify *once and for all* the filename or URL of the document containing your collection of ``XML variables''. In the above screenshot, this document is `VATrates.html`.

2. Use Edit → Reference → Insert Reference (**Ctrl+Shift+R**) to display the Include tool.

3. Type the ID of the element for which you want to insert a reference. Typing the first few letters is generally sufficient as the "Include" text field supports auto-completion.

4. Press **Enter**.

That's it. You have inserted a reference to the `span` element containing the VAT rate of Germany.

Use the right tool for the right job

For example, do *not* attempt to use the Include tool to compose a modular book. Inserting references to the chapters in the modular book is best achieved by using the Copy As Reference/Paste (**Ctrl**+**Shift**+**C**/**Ctrl**+**V**) approach described in the tutorial.