

Introduction to LyX

by the LyX Team*

January 4, 2007

*If you have comments or error corrections, please send them to the LyX Documentation mailing list: lyx-docs@lists.lyx.org

Contents

1	The Philosophy of LyX	1
1.1	What is LyX?	1
1.2	LyX and Other Word Processors	2
1.3	What is L ^A T _E X?	4
2	Navigating the Documentation	5
2.1	The Format of the Manuals	5
2.2	Units used in the Manuals	7
2.3	The Manuals	7
3	Contributing to the LyX Project	8
3.1	Contributing to LyX	8
3.2	Contributing to the Documentation	9

1 The Philosophy of LyX

1.1 What is LyX?

LyX is a document preparation system. It excels at letting you create complex technical and scientific articles with mathematics, cross-references, bibliographies, indices, etc. It is very good at documents of any length in which the usual processing abilities are required: automatic sectioning and pagination, spellchecking, and so forth. It can also be used to write a letter to your mom, though granted, there are probably simpler programs available for that. It is definitely not the best tool for creating banners, flyers, or advertisements (we'll explain why later), though with some effort all these can be done, too. Some examples of what it is used for: memos, letters, dissertations and theses, lecture notes, seminar notebooks, conference proceedings, software documentation, books, articles in refereed scientific journals, scripts for plays and movies, business proposals, presentations . . .

LyX is a program that provides a modern approach to writing documents with a computer by using a markup language paradigm, an approach that breaks with the obsolete tradition of the “typewriter concept”. It is designed for authors who want professional output quickly with a minimum of effort without becoming specialists in typesetting. The job of typesetting is done mostly by the computer, not the author; with LyX, the author can concentrate on the contents of her writing.

Part of the initial challenge of using LyX comes from the change in thinking that you, the user, must make. At one time, all we had for creating documents were typewriters, so we all learned certain tricks to get around their limitations. Underlining, which is little more than overstriking with the “_” character, became a way to emphasize text. You were forced to figure out column sizes and tab stops, and set them, before creating a table. The same applied for letters and other right justified text. Hyphenation at the end of a line required a careful eye and a lot of foresight.

In other words, we've all been trained to worry about the little details of which character goes where. Consequently, almost all word processors have this mentality. They still use tab stops for adding whitespace. You still need to worry about exactly where on the page something will appear. Emphasizing text means changing a font, similar to changing the typewriter wheel. This is the underlying philosophy of a WYSIWYG word processor: “What You See Is What You Get”. Unfortunately, that paradigm often results in “What You See Is All You Get”.

This is where LyX differs from an ordinary word processor. You don't concern yourself with what character goes where. You tell LyX *what you're doing* and LyX takes care of the rest, following a set of rules called a *style*.¹ Let's look at a little example:

Suppose you are writing a report. To begin your report, you want a section called “Introduction.” So, you go into whatever menu it is in your word processor that

¹To be fair, most recent versions of the most popular office suites now have some sort of style sheets which follow a similar markup method. However, our experience is that they are still rarely used in practice.

changes font sizes and decide on a new font size. Then you turn on bold face. Then you type, “1. Introduction”. Of course, if you later decide that this section belongs someplace else in the document, or if you insert a new section before it, you need to change the numbering for this and all following sections, as well as any entry in the table of contents.

In L^AT_EX, you go to the pull-down on the far left of the button bar and select **Section**, and type “Introduction.”

Yes, that’s all. If you cut and paste the section, it will automatically be renumbered — everywhere. And if you enter references to that section correctly (by inserting cross-reference tags), L^AT_EX will automatically update them all throughout the file so that you never, ever type a section number.

Now let’s look at the problem of consistency. Five days later, you reopen your report and start Section 4. However, you forget that you were using 18pt bold instead of 16pt, so you type in the heading for Section 4 in a different font that what you used for Section 1. That problem doesn’t even exist in L^AT_EX. The computer takes care of all that silly bookkeeping about which thing has what size font, not you. After all, that’s what a computer is good at.

Here’s another example. Suppose you’re making a list. In other word processors, a list is just a bunch of tab stops and newlines. You need to figure out where to put the label for each list item, what that label should be, how many blank lines to put between each item, and so on. Under L^AT_EX, you have only two concerns: what kind of list is this, and what do I want to put in it. That’s it.

So, the basic idea behind L^AT_EX is: specify *what* you’re doing, not *how* to do it. Instead of “What You See Is What You Get,” the L^AT_EX model is “What You See Is What You *Mean*” or “WYSIWYM.” It’s a powerful idea that greatly simplifies the mechanics of writing documents. This is also why L^AT_EX isn’t so good for creating posters and flyers—in this case, you *do* want to specify exactly where everything goes, because there are no functional units like paragraphs, sections, etc. This doesn’t mean L^AT_EX is missing some cool function. It simply means that it isn’t the right tool for the job — you don’t use a screwdriver to drive in nails.

1.2 Differences between L^AT_EX and Other Word Processors

Here’s a list of things you won’t find in L^AT_EX:

- The document ruler
- Tab stops
- Extra whitespace (i.e. hitting **Enter** or **Space** two or more times)

Tab stops, along with a ruler showing you the position of things on the page, are useless in L^AT_EX. The program worries about where things go on the page, not you. Extra whitespace is similar; L^AT_EX adds them where necessary, depending on context.

Not being able to type two blank lines in a row will be annoying at first, but it makes more sense once you're thinking in WYSIWYM terms.

Here are some things that exist in L^AT_EX, but aren't used as you might think:

- Indenting controls
- Page breaks
- Line spacing (i.e. single spaced, double spaced, etc.)
- Whitespace, horizontal and vertical
- Fonts and font sizes
- Typefaces (bold, italic, underline, etc.)

Although they exist in L^AT_EX, you generally don't need them. L^AT_EX will take care of these things for you, depending on what you're doing. Different parts of the document are automatically set in a different typeface and font size. Paragraph indenting is context dependent; different types of paragraphs get indented differently. Page breaks get handled automatically, as well. In general, the space between lines, between words, and between paragraphs is variable, set by L^AT_EX.²

Lastly, there are a few areas where we believe L^AT_EX (and L^AT_EX) surpasses many word processors:

- Hyphenation
- Lists of any type
- Mathematics
- Tables
- Cross-referencing

Granted, many modern word processors can handle mathematical symbols, tables, and hyphenation, and many have moved towards style definitions and the WYSIWYM concept. However, they've only recently been able to do so, whereas L^AT_EX is built upon the L^AT_EX document preparation system. L^AT_EX has been around for over 20 years, and *works*.

²There are ways to adjust all of these (only some of which require knowledge of L^AT_EX), either for a whole document or for a specific location in a document. See the *User's Guide* and/or the *Extended Features* manual for details.

1.3 What is L^AT_EX?

L^AT_EX is a document preparation system designed by Leslie Lamport in 1985.³ It was built up from a typesetting language called T_EX, created by Donald Knuth in 1984. T_EX takes a sequence of typesetting commands, written in a script in an ASCII file, and executes them. Many of the “tricks” of the printing trade were modeled by Knuth as computer algorithms and incorporated into T_EX, thus its excellent printed appearance. What comes directly out of T_EX is the portable document format *pdf* or the so-called “device independent” format file *dvi*. The *dvi* format is often used for previews and can later be converted to other formats like PostScript.

T_EX isn’t only a typesetting engine, it also allows you to define macros. Most people who use T_EX are actually using a macro package which Knuth created to hide a lot of the typesetting details. This is where Leslie Lamport enters our story. He wanted a macro package that was more user- and less typesetter-oriented, with a set of commands that consistently typeset things like sections, tables or math formulas in an uniform, consistent fashion. This is how L^AT_EX was born.

Now, in parallel with the development and growth of L^AT_EX, other folks were creating their own custom macro packages for T_EX, ones to make slides or articles for math journals and so on. Some used the raw T_EX facilities to do this, others began modifying L^AT_EX. To try and unify this mess, a team of L^AT_EX-nicians began to work on L^AT_EX 2_ε, the current version of L^AT_EX, during the late 1980’s. This new version of L^AT_EX has commands which provide an easier-to-use interface to T_EX’s macro-creating commands, aid in the use of new fonts, and so on. In fact, L^AT_EX is quite an extensive language in its own right! Users around the world have been creating their own add-ons for L^AT_EX beyond the standard ones.

There are two ways to extend L^AT_EX: classes and styles. A *class* is a set of L^AT_EX macros describing a new type of document, like a book, or an article. There are classes for slides, for physics and math journals... many universities even have a class for their thesis format! A *style* differs from a class in that it doesn’t define a new type of document, but a different type of *behavior* that any document can use. For example, L^AT_EX controls page margins and line spacing using two different L^AT_EX style-files designed for these purposes. There are style-files for a whole slew of things: printing labels or envelopes, changing indentation behavior, adding new fonts, manipulating graphics, designing fancy page headings, customizing bibliographies, altering the location and appearance of footnotes, tables, and figures, customizing lists, etc.

Here is a summary:

T_EX: Typesetting language with macro capability.

L^AT_EX: Macro package built upon T_EX.

classes: Descriptions of a type of document, using L^AT_EX.

³The source for the info in this section is “A *Guide to L^AT_EX 2_ε*,” by Helmut Kopka and Patrick Daly, which has an entry in the bibliography of the *User’s Guide*.

styles: Alters the default behavior of \LaTeX in some way.

LyX: Visual, WYSIWYM word-processor that uses \LaTeX to do its printing.

\LaTeX is the reason why LyX differs from other word processors. LyX is a graphical user interface (GUI) frontend for \LaTeX . It is designed to typeset documents with by using \LaTeX without knowing its macro commands. But although LyX can't never support all commands graphically, it offers its usage, so that LyX is as functional as \LaTeX .

Just like \LaTeX , LyX focuses on the context of your writing — *what* you are typing. The computer then handles how it should look.

2 Navigating the Documentation

To make it easier to answer your questions and describe all of the features of LyX, the documentation has been split up into several different files. Each one has its own purpose, as described below. Before you go plowing into any of those files, however, you should read this chapter thoroughly first, since it contains a lot of useful information and commentary that can save you some time.

The developing of LyX will hopefully never stop, so that some of the documentation may be incomplete or a bit out of date, though we try to keep up. Like the rest of LyX, the manuals are the work of a group of volunteers who have “Real Jobs”, families, dishes to clean, kitty litter to dispose of, et cetera. If you want to help out, be sure to read Section 3 in addition to the rest of this document.)

Also, please do us a favor – if anything in these manuals confuses you, is unclear, or wrong, don't hesitate to let us know! You can reach the current document maintainers by mailing to lyx-docs@lists.lyx.org. If you have questions which are not obviously answered in the documentation, and need help fast, there is an active users' mailing list which you can reach at lyx-users@lists.lyx.org.

2.1 The Format of the Manuals

Some may have printed out the manuals. Others may be reading it within LyX. There are some differences between the LyX-file and the printed version. First, the title is simply at the top of the document, not formatted on a separate page as in some of the printed versions. Nor are any of the footnotes or the Table of Contents fully visible. To open a footnote, which looks like this: foot, click on it with the left mouse button. For the Table of Contents, either click on the grey box or click on the **Navigate** menu, where the contents are displayed automatically.

In the printed manuals, all cross-references appear as the actual numbers for a chapter, section, subsection, and so on. Online, however, all cross-references appear as a light-grey box like the following: Ref. sec:Contrib. If you click on that box with

the left mouse button, a dialog box will appear containing a list of all the cross-references in the document. This introduction has only two, the first one is named “sec:Contrib”. You can go to the referred section by right-clicking on the box or by clicking the button **Go to Label** in the opened dialog. Going back to where you came from is just as easy. Clicking on **Go back** to go back to your earlier location.

Now that we’ve cleared up some of the differences between the printed and online versions of this file, we can start looking at the format of this document. You’ll occasionally notice things in different fonts:

- *Emphasized Style* is used for general emphasis, generic arguments, book titles, names of sections of other manuals, and notes from the authors.
- **Typewriter** is used for program and file names, LyX code and functions.
- **Sans Serif** is used for menu, button, or dialog box names, and the names of keyboard keys.
- **NOUN STYLE** is used for people’s names.
- **Bold** is used for L^AT_EX code

For menu accelerator keys and keybindings, you’ll probably get referred to the *Key Bindings* section of the “*Extended Features Manual*” (the file `Extended.lyx`). When we do need to reference keys, we’ll use the following prefixing convention:

- “C-” indicates the **Control-** key.
- “S-” indicates the **Shift-** key.
- “M-” indicates the **Meta-** key, which on some keyboards will be the **Alt-** key.
- “F1” ... “F12” are the function keys.
- “Esc” is the escape key.
- “Left” “Right” “Up” “Down”: self-explanatory.
- “Insert” “Delete” “Home” “End” “PageUp” “PageDown”: these are the 6 keys that appear above the cursor keys on many PC keyboards. “PageUp” and “PageDown” are called “Prior” and “Next” on some keyboards.
- **Return** and **Enter** both refer to the same key. Some keyboards label the **Return** key as “Return,” others as “Enter,” still others have two keys. LyX treats all of them as the same key, so we’ll use **Return** and **Enter** interchangeably.

Table 1: Units

unit	name/description
mm	millimeter
cm	centimeter
in	inch
pt	point (72.27 pt = 1 in)
pc	pica (1 pc = 12 pt)
sp	scaled point (65536 sp = 1 pt)
bp	big point (72 bp = 1 in)
dd	didot (72 dd \approx 37.6 mm)
cc	cicero (1 cc = 12 dd)
Scale%	% of original image width
text%	% of text width
col%	% of column width
page%	% of paper width
line%	% of line width
theight%	% of text height
pheight%	% of paper height
ex	height of letter x in current font
em	width of letter M in current font
mu	math unit (1 mu = 1/18 em)

You'll also see something like “(See ‘*Extended*’)” from time to time. We've listed the possible default keybindings for a function in its entry in the “*Extended Features Manual*,” so check there, too. Note that there are two different keybinding maps in common use in the LyX community: the “CUA” style which is the default and familiar to those coming from the PC world, and the “emacs” style, which is common with those who “grew up” on Unix systems and use the Emacs editor. Unless specifically noted, the keybindings in the documentation are from the default CUA map. If you like Emacs, you should be smart enough to read the documentation and figure out the bindings on your own.

2.2 Units used in the Manuals

To understand the units described in this documentation, Table 1 explains all units available in LyX.

2.3 The Manuals

The following list describes the contents of each of the files in the documentation:

Introduction This file.

Tutorial If you are new to LyX, and have never used L^AT_EX before, you should start here. If you have used L^AT_EX before, you should still read the *Tutorial*, starting with the section on “LyX for L^AT_EX users.” (Skimming the rest of the document wouldn’t hurt, either.)

User’s Guide The primary documentation. We’ll cover *most* of the basic operation and available features of LyX here. The main manual assumes that you’ve read the *Tutorial*.

Extended-Insets Extension of the *User’s Guide*. Documents in detail how to use tables, graphics, floats, notes, and boxes. It also includes many tricks of the L^AT_EX masters.

Extended Features Extension of the *User’s Guide*. Documents how to use raw L^AT_EX commands, additional layouts, and special-purpose editing features.

Customization A description of advanced LyX features, including how to customize the overall behavior of LyX. This includes such things as keybindings, internationalization, and configuration files.

L^AT_EX configuration LyX investigates your system upon installation. This file contains info on what LyX learned about your installation. Check it to see if you’re missing something you might like to have.

These files will reference one another as necessary. For example, the *User’s Guide* contains *some* information on installation and customization, but refers the reader to the *Customization Manual* for more information.

We’ll state again an important point:

If you are new to LyX, read the *Tutorial*. Now.

Otherwise, you could needlessly frustrate yourself. LyX does all that you need a word processor to do, but using a different approach.

3 Contributing to the LyX Project

3.1 Contributing to LyX

LyX is mostly written in C++ (the L^AT_EX importer is written in Python). It is a large project, and as a result it is not free from bugs, or the need for improvements in the source code.

3.1.1 Reporting a bug

While using LyX, you may find behavior which you consider a bug. Crashes, though rare, can happen. User interface problems are considered major bugs by the LyX team: especially helpful are indications of parts of the LyX interface you find confusing, or unclear.

LyX has a bug tracking system, which you can find at <http://bugzilla.lyx.org/>. You should check the bug tracker before reporting any bugs, in case it has already been reported. If you have a comment on an existing bug, or wish to report a new bug, you may either use the bug tracker, or send an e-mail to the development mailing list, lyx-devel@lists.lyx.org. Archives of this list are linked from the main LyX website, <http://www.lyx.org/>.

A useful bug report will at a minimum include the version of LyX you are having the problem with. Accurate, detailed descriptions are preferred - the more time developers have to spend to pinpoint the source of a bug, the less time they have for other improvements. Mention the system and system version you are running LyX with. Give the versions of the libraries you have installed on your system, and, if relevant, the versions of external programs that LyX uses. If it's a compilation or configuration problem, include the file `config.log`, and mention which compiler you are using.

3.1.2 Contributing fixes and new features

If you have made changes to LyX's source that you think should become part of LyX, send your changes as a diff file (in unified format) to the development list referenced above, along with a change log, and a description of what your patch does.

3.2 Contributing to the Documentation

LyX's documentation is extensive; however LyX is under constant development, and each new release adds new features. You may find some documentation needs improvement. This section describes what to do if you find an error, or have some suggestions for improving the documentation.

3.2.1 Reporting Errors in the Manuals

If you find a problem with the documentation, send a message to the mailing list lyx-docs@lists.lyx.org. The documentation team will make any necessary fixes.

3.2.2 Joining the Documentation Team.

The LyX Documentation Project, like anything else in the LyX project, can always use assistance! If you're interested in contributing to the Documentation Project, you need to do the following:

1. Get the latest LyX source code from <http://www.lyx.org/trac/browser/lyx-devel/trunk/lib/doc>
Inside that directory is a file called `DocStyle.lyx`. Read it; it's the style sheet for the documentation.
2. Next, read the *User's Guide* and the *Tutorial*.
The point of this exercise is to give you ideas. The *Tutorial* and *User's Guide* is likely to be the most up-to-date of all of the documentation. You should be able to glean some insights into how we want the manuals to read and to look.
3. Contact the team at:
`lyx-docs@lists.lyx.org`
to discuss your intended changes, and get some feedback on them.

The changes you wish to make may range from improving clarity of the text, to doing major re-structuring of the documentation. Any and all improvements are gladly received.