Introduction to LyX

by the LyX Team[1]

December 20, 2006

[1]If you have comments or error corrections, please send them to the LyX Documentation mailing list, <lyx-docs@lists.lyx.org>.

Contents

Chapter 1

The Philosophy of LyX

> "Uncle Cosmo, why do they call this a word processor?"
>
> "It's simple, Skyler. You've seen what food processors do to food, right?"
>
> — Jeff MacNelly in "Shoe"

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 (on PostgreSQL, remote sensing, cryptology, fictional novels, poetry, and even a children's book or two), articles in refereed scientific journals, scripts for plays and movies, business proposals ... you get the idea.

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 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 LyX, 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), LyX 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 LyX. 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 LyX, you have only two concerns: what kind of list is this, and what do I want to put in it. That's it.

[1] To 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.

So, the basic idea behind LyX is: specify *what* you're doing, not *how* to do it. Instead of "What You See Is What You Get," the LyX 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 LyX 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 LyX 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 (unless your screwdriver comes with a lifetime warranty).

1.2 Differences between LyX and Other Word Processors[2]

Here's a list of things you won't find in LyX:

- 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 LyX. The program worries about where things go on the page, not you. Extra whitespace is similar; LyX 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 LyX, 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 LyX, you generally don't need them. LyX 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,

[2] No, we're not trying to start (or win) a word processor holy war here. But we do think it's important to describe LyX's features. And one of LyX's main features, WYSIWYM, is a fundamentally different concept than the one that most of people have about word processing.

the space between lines, between words, and between paragraphs is variable, set by LyX.[3]

Lastly, there are a few areas where we believe LyX (and LaTeX) 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 LyX is built upon the LaTeX document preparation system. LaTeX has been around for decades, and *works*.

1.3 What the heck *is* LaTeX (and why do I care)?

LaTeX is a document preparation system designed by Leslie Lamport in 1985.[4] It, in turn, was built up from a typesetting language called TeX, created by Donald Knuth in 1984. "TeX" is pronounced like "blech!" which is how many people feel about it. However, most folks don't understand just what TeX is. TeX takes a sequence of typesetting commands, written in a script in an ASCII file, and executes them. It's a bit more complicated than a typewriter, but not nearly as involved as an actual printing press; however, many of the "tricks" of the printing trade were modeled by Knuth as computer algorithms and incorporated into TeX, thus its excellent printed appearance. In any case, what comes directly out of TeX is the so-called "device independent" format file, or `dvi` for short. You can then feed the `dvi` file to anything that understands `dvi`, or converts `dvi` to other formats like PostScriptő, PDF, etc.

If it weren't for one other feature, all TeX would be is a typesetting engine. However, TeX also allows you to define macros. This is where the action begins.

Most people who use TeX are actually using a macro package which Knuth created to hide a lot of the typesetting details. This is what most people think of when they think of TeX. Ordinary users don't work with raw TeX, which are the bare-bones typesetting commands. People creating new macro-packages do that. This is where Leslie Lamport enters our story. He wanted a macro package

[3] There are ways to adjust all of these (only some of which require knowledge of LaTeX), 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.

[4] The source for the info in this section is "*A Guide to LaTeX 2ε*," by Helmut Kopka and Patrick Daly, which has an entry in the bibliography of the *User's Guide*.

that was more user-oriented and less typesetter-oriented, a set of commands that consistently typeset things like sections or tables or math formulae in a uniform, consistent fashion with as little fuss as possible. This is how LaTeX was born.

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

There are two ways to extend LaTeX: classes and styles. A *class* is a set of LaTeX (and TeX) 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, LyX controls page margins and line spacing using two different LaTeX 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, et cetera.

Here's a summary:

TeX:	Typesetting language with macro capability.
LaTeX:	Macro package built upon TeX.
classes:	Descriptions of a type of document, using LaTeX.
styles:	Alters the default behavior of LaTeX in some way.
LyX:	Visual, WYSIWYM word-processor that uses LaTeX in all its glory to do its printing.

The idea of this section was to try and explain *why* LyX works somewhat differently from other word processors. The reason is simple: LyX uses LaTeX as its printing backend. Just like LaTeX, LyX focuses on the context of your writing — *what* you are typing. The computer then handles how it should look.

Oh — one last thing. LaTeX is pronounced like TeX is. It rhymes with "hey blech."[5] Usually. Lamport says in his book, though, that "*lay*-tecks is also possible". "LyX," on the other hand, is pronounced "licks." Or "lucks," or "looks," depending on what country you're from ... but numerous holy wars and flame fests have been started over this issue on the LyX mailing lists, so

[5] or "ha blech", depending on how you pronounce your "a"s...

please just pronounce it however you please. Just don't pronounce it "word". :-)

Chapter 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.

Although LyX is now well past the "version 1.0" mark, 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 Chapter 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 of you may have printed out the manual(s). Others may be reading it online, within LyX as a file. For those reading online, there are some differences from 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 visible. To open a footnote, which looks like this,[1] click on it with the left mouse button. For the Table of Contents,

[1] Hi!

either click on the grey box, or click on the Navigate menu, where the contents are displayed automatically. (Try it!)

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 grey box like the following: 3. (The printed manuals show a number instead.) 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 one named "chp:Contrib". You can go to the section the referred to by clicking the button labeled "Go to reference". Going back to where you came from is just as easy. Clicking on "Go back" or typing C-< will take you back to your earlier location. (What does "C-<" mean? See below.)

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, LaTeX code, and 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.

For menu accelerator keys and other more obscure 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.

To close me, click on the grey box on the top left of this box, the one with the word "foot" in it.

- 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.

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 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 LaTeX before, you should start here. If you think LaTeX is the stuff they make condoms out of, then you definitely need to reread Chapter 1, then read the *Tutorial*. Note that after you read the *Tutorial*, you'll probably still think LaTeX is some stretchy substance — but you *will* know how to use LyX.

If you *have* used LaTeX before, you should still read the *Tutorial*, starting with the section on "LyX for LaTeX users." (Skimming the rest of the document wouldn't hurt, either.)

If, at any time, you find yourself feeling a bit clueless with respect to LyX, try perusing the *Tutorial* before diving into any of the other manuals. It's a good springboard.

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 Features

Extension of the *User's Guide*. Documents how to use raw LaTeX commands, additional layouts, and special-purpose editing features, and includes some of the (rather bizarre, but nifty) tricks of the LaTeX masters.

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. Don't even think of going in here until you read the *Tutorial*.

LaTeX 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.

Chapter 3

Contributing to the LyX Project

3.1 Contributing to LyX

LyX is mostly written in C++. 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 behaviour 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 `LyX bug tracker 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, `LyX website http://www.lyx.org/`.

A good (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.

If you can make LyX crash, please take the time to produce a backtrace with a non-stripped lyx executable. The one built in the source directory is per

default not stripped, while the installed binary is stripped. So, run LyX from gdb by typing, for example, "`gdb /users/steve/lyx-1.0.x/src/lyx`" and then "`run`". Make LyX crash and you'll return to gdb. Use "`bt`" to produce a backtrace and include the output in the bug report[1]. If possible, then a description of a way to reproduce the bug is more important than a backtrace, because then we have the possibility to roll our own backtrace. If the bug is not easily reproducible, a backtrace is essential, because then that might be all we've got.

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 assignment *first*:

1. Get the latest LyX source code. Untar it.[2] You will find a directory inside the main tree called `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

[1] if you want to be thorough, use the "`info locals`" and "`up`" commands to print out the values of local variables at a few stack levels.

[2] The more adventurous can grab the latest documentation anonymously from the LyX CVS repository - it is contained in the `lyxdoc` module.

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.