The LyX Tutorial

by the LyX Team[1]

July 17, 2006

[1] Principal maintainer of this file is Amir Karger. If you have comments or error corrections, please send them to the LyX Documentation mailing list, `lyx-docs@lists.lyx.org`.

Contents

Chapter 1

Introduction

1.1 Welcome to LyX!

This file is designed for all of you who have never heard of LaTeX, or don't know it very well. Now, don't panic - you won't need to learn LaTeX to use LyX. That is, after all, the whole point of LyX: to provide an almost-WYSIWYG interface to LaTeX. There are some things you will need to learn, however, in order to use LyX effectively.

Some of you probably found your way to this document because you tried to put two spaces after a "." or tried to put 3 blank lines between paragraphs. After much frustration, you found you couldn't. In fact, you'll find that most of the little tricks you're accustomed to using in other word processors just won't work in LyX. That's because most word processors you've used before allow you to manually enter all spacings, font changes, and so on. So you end up not only writing a document but typesetting it, too. LyX does the typesetting for you, in a consistent fashion, letting you focus on the important things, like the content of your writing.

So, bear with us and read on. Reading this tutorial is definitely worth the time.

1.2 What the Tutorial *is* and What it *isn't*

Before we get started with this section, we want to make a quick note of something. The *Tutorial* uses the notation outlined in the *Introduction*. If you came to this manual first, go read the *Introduction*. Yes, we mean now.

Now that you know which fonts mean what, we want to talk a bit about what this *Tutorial* is for.

1.2.1 Getting the Most out of the Tutorial

This tutorial consists of examples and exercises. To get the most out of this document, you should read through the document, typing all the silly little things we're telling you to type and trying out all of the exercises to see if you get them right. For convenience, you might want to print out the PostScript® version of this document.

If you are familiar with LaTeX, you'll probably be able to read the *Tutorial* somewhat faster, since many LyX ideas are just LaTeX ideas in disguise. However, LyX does have idiosyncrasies[1] you'll want to learn about. Even if you don't feel like reading the rest of the *Tutorial*, you should definitely check out Section 5.2, which is specifically written for experienced LaTeX users.

1.2.2 What You *Won't* Find:

- Detailed explanations of all of LyX's features.

  What, you want the *User's Guide* twice?

  Seriously, though, we're here to get you up and running so that all you need is the *User's Guide*. If we tried to duplicate all of the information about all of LyX's features in here, the *Tutorial* would be redundant, too long, and forever out of date. All we do here is introduce things; imagine there is a "see the *User's Guide*" at the end of every section.

- Detailed explanations of LaTeX.

  Unnecessary. If you're really curious about learning some of the neat tricks you can do with LaTeX, you can always go get a LaTeX book. There are several good ones on the market. No need to reinvent the wheel, after all...

So, brave soul, it's time to move onwards. Time for your first document ...

[1] or, more optimistically, "features"

Chapter 2

Getting Started with LyX

2.1 Your First LyX Document

OK. You're ready to start writing. Before you do, though, there are a few things we need to mention, which will hopefully make the *Tutorial* more instructive, useful, and fun.

Because there's lots of information that we won't be giving you, the *first* thing that you need to do is find the other help files. Luckily, this is very simple. Start up LyX. Choose the *User's Guide* from the Help menu. You may want to load the *Tutorial* as well (if you're not reading it on screen already). This way, you can read them while you're writing your own file[1]. Note that once you've got more than one document open, you can use the Documents menu to switch between them. The *Tutorial* will not cover in detail subjects which are described in the other LyX manuals. This may make life a bit harder for you at the beginning, but it will keep the *Tutorial* short. It will also get you in the habit of using the other manuals, which — in the long run — will save you a lot of time.

In this *Tutorial*, we're going to assume that you have a fully working version of LyX, as well as LaTeX, `xdvi` or some other dvi viewer, `dvips` or some other way of converting `dvi` documents to PostScript® documents, and a working printer. This is a lot to assume. If any of this is not true, you (or a friendly system administrator) will need to set up your system. You can find information on setup in other manuals.

Finally, we've written a file to let you practice your LyX skills on. It's called `example_raw.lyx`. Imagine that it was typed by someone who didn't know about any of LyX's great features. As you learn new LyX functions, we'll suggest that you fix those parts of `example_raw.lyx`. It also contains "subtle" hints about how to fix things[2]. If you want to cheat (or check what you've done), there's also a file called `example_lyxified.lyx` which contains the same text

[1] They can also serve as good examples of how to use the many features of LyX.

[2] The hints are located in yellow "Notes". Access the text in a note by clicking on it.

as written and typeset by a LyX master.

The example files can be found in the `examples/` directory, which you can get to by selecting File ▷ Open and then clicking on the Examples button. Open the raw document, and use File ▷ Save_As to save a copy in your own directory for you to work on. As you fix parts of the raw document, check to see how those changes affect the dvi output.

By the way, the `examples/` directory contains lots of other examples files. They will show you how to do various fancy things with LyX. They are especially useful to display things that (due to length or other reasons) won't fit in the documentation. After you read the *Tutorial*, or when you're confused about how to do something fancy in LyX, take a look at these files.

2.1.1 Typing, Viewing, and Printing

- Open a new file with File ▷ New
- Type a sentence like: `This is my first LyX document!`
- Save your document with File ▷ Save As.
- Run LaTeX to create a `dvi` file, with View ▷ DVI. You may see things being printed in the window you ran the `lyx` command from. These are messages from LaTeX, which you can ignore for now. LyX will run `xdvi` (or some other `dvi` viewer), which will pop up a new window displaying what your document will look like when printed.[3]
- Print by using File ▷ Print and hitting OK.

Congratulations! You've written and printed your first LyX document. All of the rest is just details, which is covered in the rest of the *Tutorial*, the *User's Guide*, and the *Extended Features*.

2.1.2 Simple Operations

LyX can of course do most of the things you're used to doing with a word processor. It will word-wrap and indent paragraphs automatically. Here's a quick description of how to do some simple actions.

Undo LyX has multiple levels of undo, which means you can undo everything you've done since your current editing session started, by selecting Edit ▷ Undo over and over again. If you undo too much, just select Edit ▷ Redo to get it back.

Currently, undo is limited to 100 steps. Undo also doesn't work for *everything*; for instance, changes to the document layout. Each of these is really a LyX bug.

[3] You can save time by leaving `xdvi` running in the background. Then, you can use View ▷ Update ▷ DVI and just click on the `xdvi` window (or unminimize it) after LaTeX finishes running.

Cut/Paste/Copy Use Edit ▷ Cut, Edit ▷ Copy, and Edit ▷ Paste to cut, copy, and paste. Or automatically paste selected text (including selections from other programs) with the *middle mouse button.*

Find/Replace Use Edit ▷ Find & Replace for a search. In the dialog, search with the Find button, and use the Replace button to replace a word you've found[4]. If you like, you can specify whether to make the search case-sensitive, or to search for only complete words; you can also search backwards through the document.

Character Formatting You can *emphasize* text (which will generally put characters in italics), put it in **bold face**, or in NOUN STYLE (usually small caps, used for people's names) from the toggle buttons in the Edit ▷ Text Style dialog.

Toolbar There are buttons on the toolbar (just below the menus) which allow you to do some of the more popular functions, such as Paste and Print.

Of course, you haven't yet written enough to make most of these functions useful. As you write more, though, try undoing, pasting, etc.

2.1.3 WYSIWYM: Whitespace in LYX

One of the hardest things for new users to get used to is the way that LYX handles whitespace. As many times as you hit Return, you'll only get one blank line. As many times as you hit Space, you'll only get one space. On a blank line, LYX won't let you type even one space. The Tab key won't move you forward one tab stop; in fact there *are* no tab stops! There's no ruler at the top of the page to let you set tabs or margins, either.

Many commercial word processors are based on the WYSIWYG principle: "What You See Is What You Get." LYX, on the other hand, is based on the principle that "What You See Is What You *Mean.*" You type what you mean, and LYX will take care of typesetting it for you, so that the output looks nice. A Return grammatically separates paragraphs, and a Space grammatically separates words, so there is no reason to have several of them in a row; a Tab has no grammatical function at all, so LYX does not support it. Using LYX, you'll spend more of your time worrying about the *content* of your document, and less time worrying about the *format.* See the *Introduction* for more information on the WYSIWYM concept.

LYX does have (many) ways to fine-tune the formatting of your document. After all, LYX might not typeset *exactly* what you mean. The *User's Guide* has information about all that. It includes HFills and vertical space — which are

[4] Close the window when you're done. Or leave it open if you find it more convenient. Most dialog boxes in LYX — including the Find & Replace, Table of Contents, and Layout dialogs, as well as the various math dialogs can operate like this. A few dialogs, like File ▷ Open, won't let you type anything in the main LYX window until you actually close the dialog. Just be sure you have the right window focus when you're trying to type in the main LYX window or give a command in some other LYX dialog.

more powerful and versatile than multiple spaces or blank lines — and ways to change font sizes, character styles, and paragraph alignments by hand. The idea, though, is that you can write your whole document, focusing on content, and just worry about that fine-tuning at the end. With standard word processors, you'll be distracted by document formatting throughout the writing process.

2.2 Environments

Different parts of a document have different purposes; we call these parts *environments*. Most of a document is made up of regular text. Section (chapter, subsection, etc.) titles let the reader know that a new topic or subtopic will be discussed. Certain types of documents have special environments. A journal article will have an abstract, and a title. A letter will have neither of these, but will probably have an environment that gives the writer's address.

Environments are a major part of the "What You See Is What You Mean" philosophy of LyX. A given environment may require a certain font style, font size, indenting, line spacing, and more. This problem is aggravated, because the exact formatting for a given environment may change: one journal may use boldface, 18 point, centered type for section titles while another uses italicized, 15 point, left justified type; different languages may have different standards for indenting; and bibliography formats can vary widely. LyX lets you avoid learning all the different formatting styles.

The Environment box is located on the left end of the toolbar (just under the File menu). It indicates which environment you're currently writing in. While you were writing your first document, it said "Standard," which is the default environment for text. Now you will put a number of environments in your new document so that you can see how they work. You'll do so with the Environment menu, which you open by clicking on the "down arrow" icon just to the right of the Environment box.

2.2.1 Sections and Subsections

Type the word `Introduction` on the first line of your LyX file, and select Section from the Environment menu[5]. Be sure to use Section and *not* Section*, which will be covered below. LyX numbers the section "1" and typesets the section heading (title) in a larger font. Now hit Return. Note that the Environment box changes from "Section" back to "Standard". Section headings, like most environments, are assumed to end when you type Return. Type the document introduction:

```
This is an introduction to my first LyX document.
```

[5] You don't have to *select* the line. If nothing is selected, LyX changes the paragraph you are currently in to the selected environment. Alternatively, you can change several paragraphs to a different environment by selecting them before picking an environment.

Hit Return again, and select Section from the Environment menu again. LyX writes a "2" and waits for you to type a title. Type `More Stuff`, and you'll see that LyX again sets it as a section title.

It gets better. Go to the end of Section 1 again (after "my first LyX document") and hit Return again, and select Section from the Environment menu again. Again, LyX writes "2" and waits for you to type a title. Type `About This Document`. Section "More Stuff", which used to be Section 2, has been automatically renumbered to Section 3! In true WYSIWYM fashion, you just need to identify the text that makes up the section titles, and LyX takes care of numbering the sections and typesetting them.

Hit Return to get back to the Standard environment, and type the following five lines:

```
Sections and subsections are described below.
Section Description
Sections are bigger than subsections.
Subsection description
Subsections are smaller than sections.
```

Click on the second line and select Subsection from the Environment menu. LyX numbers the subsection "2.1", and typesets it in a font which is bigger than regular text but smaller than the section title. Change the fourth line Subsection environment as well. As you probably expected, LyX automatically numbered the section "2.2". If you put yet another section before Section 2, Section 2 will be renumbered as Section 3, and the subsections will be renumbered to "3.1" and "3.2".

Further levels of sectioning include Subsubsection, Paragraph, and Subparagraph. We'll let you play with these on your own. You may notice that paragraph and subparagraph headings are not numbered by default, and that subparagraphs are indented; see the *User's Guide* to change this. Chapter headings are actually the highest level of sectioning, above Sections, but you're only allowed to use them in certain types (text classes) of LyX documents (see Section 3.1).

Finally, you may want to have sections or subsections that are not numbered. There are environments for this as well. If you change one of your section headings to the Section* environment (you may have to scroll down in the Environment menu to find it), LyX will use the same font size for the heading as it uses for a regular section, but it won't number that section. There are corresponding "starred" heading environments for Subsection and Subsubsection. Try changing some of your sections or subsections to the starred environments, and note how the other sections' numbers are updated.

Exercise: Fix the section and subsection headings in `example_raw.lyx`.

2.2.2 Lists and sublists

LyX has several different environments for typesetting lists. The various list environments free you from hitting Tab a million times when writing an outline, or from renumbering a whole list when you want to add a point in the middle of the list, and lets you concentrate on the list content.[6] Different types of documents logically require different list environments:

- A slide presentation might use the Itemize environment's bulleted lists to describe different points.
- An outline would use the Enumerate environment's numbered lists (and lettered sublists).
- A document describing several software packages could use the Description environment, where each item in the list begins with a bold-faced word.
- The List environment — not found in LaTeX — is a slightly different form of Description.

Let's write a list of reasons why LyX is better than other word processors. Somewhere in your document, type:

```
Lyx is better than other word processors because:
```

and hit Return. Now select Itemize from the Environment menu. LyX writes a "bullet" (actually, an asterisk, which will be converted to a round circle on output) on the line. Type in your reasons:

```
Typesetting is done for you.
Math is WYSIWYG
Lists are very easy to create!
```

List environments, unlike headings, do not end when you type Return. Instead, LyX assumes you're going on to the next item in the list. The above will therefore result in a three-item list. If you want more than one paragraph within one list *item*, one way is to use the Protected Break, which you get by typing C-Return. In order to get out of the list, you need to reselect the Standard environment (or just use the keybinding, M-p s).

You've got a beautiful itemized list. You might want to run LaTeX to see how the list looks when printed out. But what if you wanted to number the reasons? Well, just select the whole list[7] and choose Enumerate from the Environment menu. Pow! As we mentioned, if you add or delete a list item, LyX will fix the numbering.

While the list is still selected, you can change to the other two list environments, Description and List, in order to see what they look like. For those two environments, each list item is made up of a term, which is the item's first word, followed by a definition, which is the rest of the paragraph (until you hit Return.) The term is either typeset in boldface (Description) or separated by a

[6] Yes, we're overemphasizing this point throughout the *Tutorial.* But it *is* the main philosophy of LyX, so please forgive us.

[7] LyX won't let you select the first bullet unless you also select the paragraph *before* the list, which you probably don't want to do. Similarly, you can't select the actual number in a numbered section title. Don't worry about it.

“Tab”[8] (List) from the rest of the paragraph. If you want to have more than one word in the definition, then separate the words with Protected Blanks.

Exercise: Typeset the list in `example_raw.lyx`

You can nest lists within each other in all sorts of interesting ways. An obvious example would be writing outlines. Numbered and bulleted lists will have different numbering and bulleting schemes for sublists. See the *User’s Guide* for details on the different sorts of lists, as well as examples which use *a lot* of nesting.

2.2.3 Other Environments: Verses, Quotations, and More

There are two environments for setting quotations apart from surrounding text: Quote for short quotes and Quotation for longer ones. Computer code (the LyX-Code environment, also used in the *Tutorial* for the long typing examples) is written in a `typewriter` font; this environment is the only place in LyX where you’re allowed to use multiple spaces to allow code indenting. You can even write poetry using the Verse style, using Return to separate stanzas, and C-Return to separate lines within a stanza. See the *User’s Guide* for more complete descriptions of all of the available LyX environments.

Exercise: Correctly typeset the Quote, LyX-Code, and Verse in `example_raw.lyx`

[8] But a typesetter’s tab, which will change to fit the size of the largest term, not a pathetic, rigid, unchangeable typewriter Tab.

Chapter 3

Writing Documents

The previous chapter hopefully allowed you to get used to writing in LyX. It introduced you to the basic editing operations in LyX, as well as the powerful method of writing with environments. Most people who use LyX, though, will want to write documents: papers, articles, books, manuals, or letters. This chapter is meant to take you from simply writing text with LyX to writing a complete document. It will introduce you to text classes, which allow you to write different sorts of documents. It will then describe many of the additions that turn text into a document, such as titles, footnotes, cross references, bibliographies, and tables of contents.

3.1 Text classes

Different sorts of documents should be typeset differently. For example, books are generally printed double-sided, while articles are single-sided. In addition, many documents contain special environments: letters contain some environments — such as the sender's address and the signature — which do not make sense in a book or article. The LyX *text class*[1] takes care of these large scale differences between different sorts of documents. This *Tutorial*, for example, was written in the Book text class. Text classes are another major part of the WYSIWYM philosophy; they tell LyX how to typeset the document, so you don't need to know how.

Your document is probably being written in the Article text class[2]. Try changing to other text classes (using the Document ▷ Settings dialog) to see how they are typeset differently. If you change your document to the Book text class and look at the Environment menu, you'll see that most of the allowed environments are the same. However, you can now use the Chapter environment. If you're ever unsure about which environments you can use in a given text class, just consult the Environment menu.

[1] LaTeX users: this is equivalent to the LaTeX document class

[2] That's usually the default text class

Font sizes, one- or two-column printing, and page headings are just some of the ways journals' typesettings differ from one another. As the Computer Age continues to mature, journals have begun accepting electronic submissions, creating LaTeX "style files" so that authors can submit correctly typeset articles. LyX is set up to support this as well. For example, LyX supports typesetting (and extra environments) for the American Mathematics Society journals using the Article (AMS) text class.

Here's a very quick reference to some of the text classes. See the *Special Document Classes* section of the *Extended Features* manual for many more details.

Name	Notes
article	one-sided, no chapters
article (AMS)	layout & environments for American Math Society
report	longer than article, two-sided
book	report + front and back matter
slides	transparencies (also including FoilTeX)
letter	lots of extra environments for address, signature…

3.2 Templates: Writing a Letter

One of the most popular text classes is Letter. One way to write a letter would be to open a new file, and choose Letter class in the Document ▷ Settings dialog. While this is the most obvious way to write a letter, it seems like extra work. Every time you write a business letter, you want to have your address, the address you're sending to, a body, a signature, etc. LyX therefore has a *template* for letters, which contains a sample letter; once you have a template, you can just replace a couple parts of the letter with your text each time you write a letter.

Open a new file with File ▷ New from Template. Select `letter.lyx` as the template. Save and print the file to see how the various environments are typeset.

When you look at the Environment menu, you'll see several environments, like the My Address environment, which don't even exist in most other text classes. Others, like Quote and Description, are familiar. You can play around for a while to figure out how the various environments work. You'll notice for example that the Signature environment has the word "Signature:" in red before the actual text of the signature. This word doesn't show up in the actual letter, as you'll see if you try printing the file. It's just there to let you know where the signature goes. Also, note that it doesn't matter where in the file the Signature line is placed. Remember, LyX is WYSIWYM; you can put the Signature environment anywhere you want, but LyX knows that in the printout, the signature should be at the end.

A template is just a regular LyX file. This means you can fill in your address and signature and save the file as a new template. From now on, any time you want to write a letter, you can use the new template to save time. We probably don't have to suggest an actual "exercise" here; just write a letter to someone![3]

Templates can be a huge time-saver, and we urge you to use them whenever possible. In addition, they can help a person learn how to use some of the fancier text classes. Finally, they may be useful for a person who is configuring LyX for a bunch of less computer-aware users. When they're first learning LyX, it will be much less intimidating if they have a letter template customized for their company, for example.

3.3 Document Titles

LyX (like LaTeX) considers the title — which may contain the actual title, the author, the date, and even an abstract of a paper — to be a separate part of the document.

Go back to your `newfile.lyx` document and make sure it's using the Article text class.[4] Type a title on the first line, and change the line to the Title environment. On the next line, type your name and change it to the Author environment. On the next line, write the date in the Date environment. Type a paragraph or two summarizing your document using the Abstract environment. Notice how the title is presented when it's printed out. If you changed the document format to Book, you'll get a separate title page, like the first page of this tutorial.

Exercise: Fix the title, date, and author in `example_raw.lyx`

3.4 Labels and Cross-References

You can label a section (or subsubsection, or, more rarely, just a random piece of text) in your document. Once you do so, you can refer to this section in other parts of the document, using cross-references. You can refer either to the section's number, or to the page that the section appears on. As with sections and footnotes, LyX worries about the cross-references for you. Automatic labels and cross-references are one of the best advantages of LyX (and LaTeX) over conventional word processors.

Your first label

Let's mark our second section, whose title is "About This Document". Click at the end of the section title line, and select Insert ▷ Label. A dialog asks you for a

[3] One warning, if you're writing from a template. If you erase all of the text in an environment — for example, if you erase the whole My Address field so that you can replace it with your own — and then you move the cursor without writing any text, the environment may disappear. This is because most environments cannot exist without any text in them. Just reselect the environment from the Environment menu to get it back.

[4] You should not be using the letter any more, since the Letter textclass doesn't allow titles.

label name, and gives you a suggestion. When you click on OK, the label name will be placed in a box next to the section title.

By the way, you could have put the label right anywhere within the section as well; section references will refer to the last section or subsection whose heading comes before the label. However, putting it on the same line as the section title (or, perhaps, on the first line of the section's text) ensures that page references will reference the beginning of the section.

So far you haven't done anything — the dvi file will look exactly the same, since labels don't show up in the printed document. However, now that you've added a label, you can refer to that label with cross-references. We'll do that next.

Your first cross-references

Place the cursor somewhere in Section 2 of your document. Type

```
If you want to know more about this document, then see
Section , which can be found on page .
```

Now — with the cursor after the word "section" — choose Insert ▷ Cross Reference. The Reference dialog pops up. It shows a list of the possible labels you can reference. At the moment, there should be only one, "sec:aboutdocument". Select it (it may be selected by default), and click Apply. Now put the cursor after the word "page", and change the reference type to use the page number then click Apply. (To be really correct, you should put a Protected Blank in between the word "Section" and the reference. Same for the page reference.)

LyX puts the references in a box right where the cursor was. In the printed document, this reference marker will be replaced with either the page or section number (depending on what you selected in the Reference dialog). Use View ▷ Update ▷ DVI, and you'll see that on the last page we refer to "Section 2" and "Page 1" (or whatever page Section 2's title is on).

Conveniently, a cross-reference acts a hyperlink when you're editing a document in LyX; clicking on it will pop up the Reference dialog, clicking Go to Label will move the cursor to the referenced label.

More fun with labels

We told you that LyX worries about numbering cross-references; now you can test that. Add a new section before Section 2. Now rerun LaTeX, and — voilà! — the section cross reference changed to "3"! Change "About this Document" to a subsection, and the cross-reference will reference Subsection 2.1 instead of Section 3. The page reference won't change unless you add a whole page of text before the label, of course.

If you want some more practice with labels, then try putting a new label where your first cross-reference was, and refer to that label from elsewhere in the document. If you'll be inserting cross-references often (if, for example, you're writing a journal article), it may be convenient to leave the Reference dialog open.

If you want to make sure that the cross-referencing gets the pages right even for larger documents, Copy a couple pages of text from the *User's Guide* to the clipboard, and Paste the stolen text into your document[5].

Exercise: Fix the references in `example_raw.lyx`

3.5 Footnotes and Margin Notes

Footnotes can be added using the Insert Footnote button in the toolbar[6] or Insert ▷ Footnote. Click at the end of the word "LyX" somewhere in your document and hit the Insert Footnote button. A footnote box appears where you can enter the text of the footnote. LyX should place the cursor at the beginning of the footnote box. Type

```
LyX is a typesetting word processor.
```

Now click on the button labelled "foot." The footnote box disappears, leaving the button showing where the footnote marker will be in the printed text; this is called "folding" the footnote. You can unfold the footnote at any time — and re-edit its text, if you want — by clicking again on the "foot" button.

You may wonder why the footnote button is a word instead of a number. The answer is that LyX worries about the footnote numbering for you in the printed text. You can see this yourself by looking at the `dvi` file (or printout). If you add other footnotes, LyX will renumber the footnotes. Since LyX (well, LaTeX, actually) takes care of the footnote numbering, there's really no need to put the numbers in the LyX file.

A footnote can be cut and pasted like normal text. Go ahead; try it! All you need to do is select the footnote button[7] and Cut and Paste it. In addition, you can change regular text to a footnote, by selecting it and hitting the Insert Footnote button; change a footnote to regular text by clicking the Insert Footnote button when the cursor is in a footnote.

Margin notes can be added using the toolbar button (the button shows an arrow pointing to red text next to (i.e., in the margin of) black text, and should be next to the Insert Footnote button in the toolbar.) or Insert ▷ Marginal Note. Margin notes are like footnotes, except that:

- the on-screen boxes say "margin" instead of "foot"
- the notes will be placed in the margin, instead of below the text

[5] By the way, copying a chapter title may cause an error, because chapters aren't allowed in the article class. If this happens, just delete the chapter title. If you want to know why this happens, see Section 3.1.

[6] The button shows an arrow pointing to red text, which is just below some black text.

[7] It may be easier to select it using the keyboard. You might accidentally open the footnote if you're trying to select the marker itself with the mouse.

- margin notes are not numbered

Change your LyX footnote back to text, then select and change it to a margin note. Run LaTeX again to see what the margin note looks like.

Exercise: Fix the footnote in `example_raw.lyx`

3.6 Bibliographies

Bibliographies (at least in the exact sciences) are similar to cross references. The bibliography contains a list of references at the end of the document, and they can be referenced from within the document. Like section titles, LyX and LaTeX make your job easier by automatically numbering the bibliography items and changing citations when the items' numbers change.

Go to the end of the document and switch to the Bibliography environment. Now, each paragraph you type will be a reference. Type `The Lyx Tutorial, by the LyX Documentation Team` as your first reference. Note that LyX automatically puts a number in a box before each reference. Click on the boxed reference number, and a Bibliography item dialog box appears. You use the first field, the Key, to refer to this reference within the LyX document. By default, it is a number. Change the Key field to "lyxtutorial" to make it easy to remember.

Now pick somewhere in your document that you would like to insert a reference. Do so with Insert ▷ Citation. A Citation dialog appears. The right panel in this dialog lists all the bibliography entries, and this field allows you to choose which bibliography item you want to cite. Select "lyxtutorial" (right now, that's the only item in the bibliography), then use the left arrow in the center to insert it. (You can have multiple citations in the same place by transferring a number of keys this way.) Now run LaTeX, and you'll see that the citation appears in brackets in the text, referring to the bibliography at the end of the document.

How are the other fields used? The Text after field in the Citation dialog will put a remark (such as a reference to a page or chapter within the referenced book or article) in the brackets after the reference. If you want the references to have labels instead of numbers in the printed output (for example, some journals would use "[Smi95]" to refer to a paper written by Smith in 1995), use the Label field in the Bibliography Entry Settings dialog. As usual, you can see the *User's Guide* for details.

Exercise: Fix the bibliography and citation in `example_raw.lyx`

3.7 Table of Contents

You may want to put a table of contents at the beginning of your document. LyX makes this very easy to do. Just hit Return after your document title and before your first section title and choose Insert ▷ List / TOC ▷ Table of Contents. The words "Table of Contents" will appear in a button on the first line of the document.

This may not appear to be very useful. However, if you look at your `dvi` file, you will see that a table of contents has been generated, listing the various sections and subsections in your document. As usual, if you reorder sections or create new ones, you will see those changes in the `dvi` file when you update it.

The table of contents is not printed in the on-screen version of the document, because you can't edit it anyway. However, you can display the table of contents in a separate window by clicking on the table of contents button, or by using Document ▷ Table of Contents. The menu command will work even if you don't have a table of contents inset in your document. This is a very useful tool. You can use the Table of Contents window to move around your document. Clicking on a (sub)section title in the Table of Contents window will highlight that line and move the cursor (in the LyX editing window) to that place in the document. You can also use the arrow keys to move up and down in the table of contents. You may therefore find it convenient to leave this window open throughout editing sessions. You can get similar functionality from the Navigate menu, though, where the table of contents appears automatically.

To get rid of the Table of Contents, you can delete the table of contents button just like any other text.

Exercise: Fix the table of contents in `example_raw.lyx`

Chapter 4

Using Math

LaTeX is used by many scientists because it outputs great looking equations, avoiding the control characters used by word processors and their equation editors. Many of these scientists are frustrated, however, because writing equations in LaTeX is more like programming than writing. Happily, LyX has WYSIWYM support for equations. If you are used to LaTeX, you'll find that all of the usual LaTeX math commands can be typed in normally, but they will show up in a WYSIWYM fashion. If, on the other hand, you've never written in LaTeX, then the Math Panel will allow you to write professional-looking math quickly and easily[1].

4.1 Math Mode

Somewhere in your LyX document, type:

```
I like what Einstein said, E=mc^2, because it's so simple.
```

Now, that equation doesn't look very good, even in the `dvi` file; there's no space between the letters and the equals sign, and you'd like to write an actual superscript for the "2". That bad typesetting happened because we didn't tell LyX that we were writing a mathematical expression, so it typeset the equation like regular old text.

Instead, we create a formula that will get typeset properly. In order to create a formula, just click the toolbar button with $\frac{a+b}{c}$ written on it in blue. LyX will insert a little blue square, which is an empty math formula. LyX has placed the cursor in the blue square, so just type `E=mc^2` again. The expression is typed in blue, and the blue square disappears as soon as the formula is not empty. Now type Esc to leave the equation The purple markers disappear, leaving the cursor to the right of the expression, and now if you type something, it will be regular text.

[1] LyX can't check if the math you're writing is actually *correct*. Sorry.

Run LaTeX and look at the dvi file. Notice that the expression was typeset nicely, with spaces between the letters and the equals sign, and a superscript "2". Letters in math mode are assumed to be variables, and come out in italics. Numbers are just numbers.

This math editor is another example of the WYSIWYM philosophy. In LaTeX, you write a mathematical expression using text and commands like \sqrt; this can be frustrating, because you can't see what an expression looks like until you LaTeX the file, and may have to spend time to find missing brackets or other "bugs". On the other hand, LyX doesn't attempt to get the expression to look perfect (WYSIWYG), but it gives you an extremely good idea of what the expression will look like. LaTeX then takes care of the professional typesetting. 99% of the time, you won't have to make any changes to the font sizes or spacing that LaTeX outputs. This way (sorry to be so repetitive) you can focus on the *content* of your mathematical expressions, not their format.

4.2 Navigating an Equation

Now let's change $E = mc^2$ to $E = 1 + mc^2$. Use the arrow keys to move the cursor into the expression. Note that when you enter the expression, the purple markers appear to let you know you're editing math. Now you can use Left and Right to move the cursor past the equals sign, and just type "1+". Again, you can use the arrow keys or Esc to leave the formula.

Other than the special keys described below, typing in math mode is like editing regular text. Use Delete (or Backspace) to delete things. Select text either with the arrow keys or with the mouse. Edit ▷ Undo works in math mode, as does cutting and pasting. One thing to be careful of: if you're right outside a formula and you type Delete (or Backspace), it will delete the whole expression. Luckily, you can just use Undo to get it back.

What if you want to change $E = mc^2$ to $E = mc^{2.5} + 1$? Again, you can use the mouse to click in the right place. However, you can also use the arrow keys. If the cursor is just after the "c" but before the "2", then typing Up will move the cursor to the level of the superscript, just before the "2". Add the ".5". Now, hitting Down will move the cursor back to the regular level. In fact, if you hit Down from anywhere within the superscript, the cursor will be placed just *after* the superscript (so that you can then type the "+1").

4.3 Exponents and Indices

An exponent can be entered from the Math Panel (see below), but it's actually simpler just to type the caret key, "^". LyX will place another blue rectangle in the superscript, so that whatever you write next will be superscripted, and in a smaller font size. Everything you type until you hit a Space (or Esc to exit Mathed entirely) will be in the superscript.

Writing a subscript (index) is just as easy — start one by typing the underscore key, “_“. You can subscript and superscript both subscripts and superscripts like this: $A_{a_0+b^2} + C^{a_0+b^2}$.

Exercise: Put equation 1 of `example_raw.lyx` into math mode.

4.4 The Math Panel

The Math Panel is a convenient way to enter symbols or to perform many complicated Mathed functions. Many of these functions can be accomplished from the keyboard or the Edit ▷ Math or Insert ▷ Math menus. However, we're going to concentrate on using the Math Panel, just to let you know what's out there; you can learn keyboard shortcuts later, from other manuals. So open it using Insert ▷ Math ▷ Math Panel now and leave it open while reading this section.

Right-clicking on a formula will open the Math Panel for you.

4.4.1 Greek and symbols

The Math Panel which allow you to choose from a large array of symbols used in math: various arrows, relations, operators, and sums and integrals. Note that subscripting and superscripting allow you to put lower and upper limits on sums and integrals.

“Nothing you can do that can't be done. . . All you need is ♡.”

4.4.2 Square roots, accents, and delimiters

To type a square root, just click on the button with a square root sign on it. The square root appears, and the cursor is in a new insertion point inside the square root. You can type variables, numbers, other square roots, fractions, whatever you want. LyX will automatically resize the square root to fit what's inside.

Accenting a character ($\overrightarrow{v}$) or group of characters ($\overrightarrow{a+b}$) is done the same way. The Decoration types are available from the panel. Click on a decoration, and LyX will insert that decoration with an insertion point under (or over) it. Just type what you want in the insertion point. There are two sets of decorations: those that resize with the text you type, and those that have fixed size, and are most appropriate for a single letter.

Delimiters such as parentheses, brackets, and braces work similarly, but are a bit more complicated. Hit the Delimiter button, which features a blue square surrounded by brackets, to pop up the Delimiter dialog. Your current selection of delimiters is displayed in a box. It's a pair of parentheses by default, but you can choose a pair of braces, a brace and a parenthesis, or even choose the empty square to have something like “$a = \langle 7$” (the empty delimiter is displayed as a broken line in LyX, but won't show up in the output).

If you're lazy, you can type actual parentheses in math mode, rather than using the Delimiter window. However, those parentheses will be the same size as regular text, which will look bad if you have a big fraction or matrix inside the parentheses. Using the Delimiter window will guarantee that the delimiters are sized based on what's inside them.

You can also put delimiters or a square root sign or a decoration on already existing text. Select the portion of the formula that you want to adjust, and then click on the button you want from the Math Panel. Try using this to change Newton's second law from scalar to vector form ($f = ma$ to $\overrightarrow{f} = m\overrightarrow{a}$). Once you've learned about matrices, this is how you'll put parentheses or brackets around them.

4.4.3 Fractions

Fractions are very simple in Mathed. Just click on the Fraction button in the Math Panel, which shows a fraction with blue squares in the numerator and the denominator. LyX writes two insertion points in a fraction. As you would expect, you can use arrow keys or the mouse to move around a fraction. Click on the top square and type "1". Now hit Down and type "2". You've made a fraction! Of course you can type anything within each of the two boxes: variables with exponents, square roots, other fractions, whatever.

Exercise: Put equation 2 of `example_raw.lyx` into math mode.

4.4.4 TeX mode: Limits, log, sin and others

Because letters in math mode are considered to be variables, if you type "sin" in math mode, LyX thinks you're typing the product of the three variables s, i, and n. The three letters will be typeset in italics, when what you really wanted was the word "sin" typeset in Roman. In addition, LyX won't put a space between the word "sin" and the "x" (typing Space will just exit math mode). So how do you get "$\sin x$" instead of "$sinx$"?

Click on "sin" in the Functions list in the Math Panel. The word "sin" is written in black, in upright roman type. The whole word is treated as one symbol, so if you type Backspace, it will delete the whole word. Now type "x", which will be written in blue italics, like you expect in Mathed. In the `dvi` file, the expression will be correctly typeset. Try it.

Other commands you need to type in TeX mode using the Functions box include other trigonometric functions and their inverses, hyperbolic functions, logarithms, limits, and quite a few others. These functions can take subscripts and superscripts, important for typing "$\cos^2\theta$" or "$\lim_{n\rightarrow\infty}$".

Exercise: Put equation 3 of `example_raw.lyx` into math mode.

4.4.5 Matrices

Click on the Matrix button in the Math Panel. The dialog has two sliding bars which allow you to choose how many rows and columns you want in your matrix. Choose 2 rows and 3 columns and hit Apply or OK. LyX prints 6 insertion points in a 2×3 matrix. As usual, you can put any sort of Mathed expression (a square root, another matrix, etc.) in each insertion point. You can also leave some of the insertion points empty if you want.

Tab can be used to move horizontally between the columns of a matrix. Alternatively, you can use the arrow keys to move around - hitting Right at the end of one box will move to the next box, Down will move to the next row, etc.

If you suddenly need more rows or columns, use Edit ▷ Math ▷ Add Row and Add Column. They add a row or column just after the current position. Overdid it? Use Delete Row and Delete Column from the same menu.

See the *User's Guide* for information on how to change the horizontal alignment of each column, and how to change the vertical position of the whole matrix. Note that if you want to write a table containing text, you should use LyX' wonderful table support, rather than trying to write text in a matrix.

4.4.6 Display mode

All of the expressions we have written so far have been on the same line as the text that came before and after them, otherwise known as inline expressions. This is fine for short, simple expressions, but if you want to write larger ones, or if you want your expressions to stand out from the text, you need to write them in display mode. In addition, only displayed expressions can be labeled and numbered (see the *User's Guide*), and multi-line equations (see Sec. **??**) must be in display mode.

Click on the Display button in the Math Panel, which represents a couple lines of text before and after a centered blue box. LyX inserts a formula, but the insertion point is on a new line, and it's centered within that line. Now type an expression and run LaTeX to see how it looks. The Display button is actually a toggle; use it now to change a couple of your expressions to display mode and back.

Display mode has a couple differences from inline mode:

- The default font is larger for a few symbols, like $\sum$ and $\int$
- Subscripts and superscripts for limits and sums (but not integrals) are written under rather than next to the symbols
- Text is centered

Other than these differences, though, displayed expressions and inline expressions are very similar.

One final note about the way displayed formulae are typeset: be careful about whether you're putting your equation into a new paragraph or not. If your formula is in the middle of a sentence or paragraph, then don't press Return. Doing so will cause the text *after* the formula to start a new paragraph. That text will therefore be indented, which is probably not what you want.

Exercise: Put the various equations in `example_raw.lyx` into display mode, and see how they're typeset differently.

Exercise: Using various tools you've learned in this section, you should be able to write an equation like[2]:

$$f(x) = \begin{cases} \log_8 x & x > 0 \\ 0 & x = 0 \\ \sum_{i=1}^{5} \alpha_i + \sqrt{-\frac{1}{x}} & x < 0 \end{cases}$$

4.5 More Math Stuff

Mathed can do plenty more. By now, you're familiar with the basics, so we'll just refer to the *User's Guide* for tips on how to:

- Labeling and numbering expressions
- Multi-line equations
- Change typefaces, e.g., to write bold-face text in an expression.
- Fine-tune font sizes and spacing within an expression. (Don't worry about this until your final draft!)
- Write macros. These are very powerful, because you just define them once at the top of the document, and then you can use them throughout the document. If you change the macro definition, the references to the macro will be changed throughout the document. Macros can even take arguments.
- Do lots of other things we didn't have time to mention in this *Tutorial.*

[2] After you've done it the hard way, why don't you give Insert ▷ Math ▷ Cases environment a try?

Chapter 5

Miscellaneous

5.1 Other Major LyX Features

We haven't gone through all the possible commands in LyX, and we aren't planning on it. As usual, see the *User's Guide* for more information. We'll just mention a couple more major things LyX can do...

- LyX has WYSIWYM support for tables. Use the Insert ▷ Table to get a table. Click on the table with the *right button* to get a Table Settings dialog box which allows extensive table editing.
- LyX also supports including pictures in a number of formats (including JPEG and other bitmap formats, PostScript® and raw LaTeX) within documents. (You guessed it: Insert ▷ Graphics. Then click on the figure to choose the file to include, rotate or scale it, etc.) Tables and figures can have captions, and LyX will automatically generate lists of figures and/or tables.
- Version control is supported, using RCS (`man rcsintro` for more info).
- LyX is heavily configurable. Everything from how the LyX window looks to how the output comes out can be configured in a number of ways. Much configuration is done through Tools ▷ Preferences. For more information on this, check out Help ▷ Customization.
- LyX is being developed by a team of programmers on five continents. Therefore, LyX has better support for non-English languages (such as Dutch, German, French, Greek, Czech, Turkish, ...) than many word processors. Even some right-to-left languages like Hebrew or Arabic are supported. You can write documents in other languages, but you can also configure LyX to show its menus and error messages in other languages.
- The LyX menus feature keybindings. This means that you can do File ▷ Open by typing M-F followed by O or by using the binding which is shown

next to it in the menu (C-O by default). Keybindings are also configurable. For information on this, check out Help ▷ Customization.

- LyX can read in LaTeX documents. See Section 5.2.2.
- Spellchecking and thesaurus facilities are available.
- The text box near the bottom of the LyX window is called the minibuffer (after a similar feature in `emacs`). This gives you access to all sorts of interesting functionality, including functionality which could break your document. In other words, don't type in the minibuffer unless you know what you're doing.

5.2 LyX for LaTeX Users

If you don't know anything about LaTeX, you don't have to read this section. Actually, you might want to *learn* about LaTeX, and then read this chapter. However, many people who begin to use LyX will be familiar with LaTeX. If you are such a person, you may be wondering if LyX can really do everything LaTeX can do. The short answer is that LyX can do pretty much everything LaTeX can do in one form or another, and it definitely simplifies most parts of writing a LaTeX document. The tool that is used to convert a LaTeX document to LyX was rewritten completely for LyX 1.4. It should now be able to handle most LaTeX gracefully.

Because this is just a tutorial, we are only going to mention things that new LyX users will most likely be interested in. In the interests of keeping the *Tutorial* short, we will give only minimal information here. The *Extended Features* manual, specifically the *Secrets of the LaTeX Masters* chapter, has a great deal of information on differences between LyX and LaTeX, and how to do various LaTeX tricks in LyX.

5.2.1 TeX Mode

Anything that you enter in TeX mode will be passed straight to LaTeX, and will be displayed in red on the screen. You can use TeX commands in LyX by choosing Insert ▷ TeX Code. This creates a text box, and everything within it is passed straight to LaTeX.

In a math formula, TeX mode is handled a bit differently. Enter TeX mode by typing a backslash. The backslash is not written out, but anything you type afterwards will be in red. You exit TeX mode by typing Space or some other non-alphabetic character, like a number, underscore, caret, or parenthesis. Once you exit TeX mode, if LyX knows the TeX command you've typed in, it will convert it to WYSIWYM. So if, in a formula, you type `\gamma`, then when you type Space, LyX will change the red "gamma" to a blue "γ". This will work for almost all, non-complicated math macros. This may be faster than using the Math Panel, and will be especially convenient for experienced LaTeX users.

As a special case, if you type a brace in TeX mode, then the beginning *and* ending braces will be inserted in red, then take you *out* of TeX mode and place the cursor between the braces. This makes it more convenient to type commands that LyX doesn't know which take an argument.

LyX can't do absolutely everything that LaTeX can do (yet?). Some fancy functions are not supported at all, while some work but aren't WYSIWYM. TeX mode allows users to get the full flexibility of LaTeX, while having all the convenient features of LyX, like WYSIWYM math, tables, and editing. LyX could never support every LaTeX package. However, by typing `\usepackage{foo}` in the preamble (see Section 5.2.4.2), you can use any package you want — although you won't have WYSIWYM support for that package's features.

5.2.2 Importing LaTeX Documents — `tex2lyx`

You can import a LaTeX file into LyX by using the File ▷ Import ▷ LaTeX command in LyX. This will call `tex2lyx` which will create a file `foo.lyx` from the file `foo.tex` — and then open that file. If the translation doesn't work, you can try calling `tex2lyx` from the command line, possibly using fancier options.

`tex2lyx` will translate most legal LaTeX, but not everything. It will leave things it doesn't understand in TeX mode, so after translating a file with `tex2lyx`, you can look for red text and hand-edit it to look right.

`tex2lyx` has its own manpage. Read it to find out about which LaTeX commands and environments aren't supported, bugs (and how to get around them), and how to use the various options.

5.2.3 Converting LyX Documents to LaTeX

You might wish to convert a LyX Document to a LaTeX file. For example, a co-worker or co-author who doesn't have LyX might want to read it. This is very easy to do with LyX. Select File ▷ Export ▷ LaTeX. This will create a file `whatever.tex` from the `whatever.lyx` file you are editing. LyX always creates temporary LaTeX files when viewing or printing files, so it is very good at generating LaTeX.

5.2.4 LaTeX Preamble

5.2.4.1 Document Class

The Document ▷ Settings dialog takes care of many of the options that you would input in a `\documentclass` command. Change the class, default font size and paper size here. Put any extra options to the `\documentclass` command in the Extra Options area.

5.2.4.2 Other Preamble Matter

If you have special commands to put in the preamble of a LaTeX file, you can use them in a LyX document as well. Select Document ▷ LaTeX Preamble and type in the dialog window (or from the document settings dialog, depending on the frontend). Anything you type will (like with TEX mode) be sent directly to LATEX.

5.2.5 BibTEX

LYX has support for BibTEX, which allows you to build databases of bibliographical references to be used in multiple documents. Select Insert ▷ List / TOC ▷ BibTEX Reference to include a `bib` file. Click on the resulting "BibTEX Generated References" button, and you will get a BibTEX dialog. In the Database field, type what you would type inside the braces of a `\bibliography{}` command[1]. Similarly, in the Style field, type what you would type inside the braces of a `\bibliographystyle{}` command.

After you've done this, you can use citations from any bibliographies you're including with Insert ▷ Citation (see Section 3.6). LYX will take care of running BibTEX. The box in the Citation dialog will show a list of all the references in your `bib` file.

5.3 Errors!

Sometimes when you LATEX a document, there will be errors, things that LYX or LATEX can't understand. When this happens, LYX will open a LATEX Errors dialog. Clicking on individual errors in this dialog will take you to the place in the LYX document where the error occurs and also display the detailed LATEX error message.

[1] Like in regular LATEX, multiple bibliographies should be separated by commas, with no whitespace.

The LyX User's Guide

by the LyX Team*

July 17, 2006

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

Contents

1 Introduction

1.1 What is LYX?

LYX is a document preparation system. It is a tool for producing beautiful manuscripts, publishable books, business letters and proposals, and even poetry. It is unlike most other “word processors” in the sense that it uses the paradigm of a markup language as its core editing style. That means that when you type a section header, you mark it as a “Section”, not “Bold, 17 pt type, left justified, 5 mm space below”. LYX takes care of the typesetting for you, so you deal only with concepts, not the mechanics.

This philosophy is explained in much greater detail in the *“Introduction”.* If you haven’t read it yet, you need to. Yes, we mean now.

The “*Introduction*” describes several things in addition to LYX’s philosophy: most importantly, the format of all of the manuals. If you don’t read it, you’ll have a bear of a time navigating this manual. You might also be better served looking in one of the other manuals instead of this one. “*Introduction*” describes that, too.

1.2 Getting Started

1.2.1 Invoking LYX

Similar to other Linux [and other brands of Unix] programs, you start LYX by simply typing `lyx` at the command line. You can, of course, include several command-line options, including file names. We’re not going to repeat all of the command-line options here, since we’ve already done that in the `man` page for LYX. Check there for more info.

There are one or two things we’d like to comment on:

Please note that if you include more than one file name on the command line, LYX will load them all, though it won’t display them all simultaneously. More on that in a bit.

1.2.2 How LYX Looks

Like most applications, LYX has the familiar menu bar across the top of its window. Below it is a toolbar with a pulldown box and various buttons. There is, of course, a vertical scrollbar and a main work area for editing documents. Near the bottom of the window is a small window containing a single line of text. This is the *minibuffer* (a term which we’ve swiped, lock, stock, and barrel, from GNU Emacs), which really means "command buffer". Type M-x when you need to type a command in the minibuffer.

Note that there is no horizontal scroll bar. This is not a bug or an oversight, but intentional. When you read a book, you expect the end of a line to wrap around to the next line. Text overflows onto new pages in a vertical fashion, hence the need for only a vertical scrollbar.

There are three cases where you might want a horizontal scrollbar. The first case is large figures, displayed WYSIWYG. This, however, is due to a flaw in the routine that displays graphics on the LyX screen in a WYSIWYG fashion; it should rescale the graphics to fit in the window, just as you'd need to rescale graphics to fit on a page. The second and third cases are tables and equations which are wider than the LyX window. You can use the arrow keys to scroll horizontally through the table, but this doesn't work for equations yet.

1.2.3 HELP!

First, the bad news: the help system is not as thorough as that in many commercial applications. Patience. We're working on it.

Now the good news: the help system consists of the LyX manuals. You can read *all* of the manuals from inside LyX. Just select the manual you want read from the Help menu.

While we're at it, we'd like to make a comment about the manuals. They're not idiot-proof, not in the least. Here's what one of our authors, John Weiss, once said about manuals:

> I hate manuals.
>
> Yes, we've all dealt with the terse, poorly-translated, or cryptic manuals. They are aggravating. I find, however, that the overly simplified ones are even more aggravating. First, they spend about half their time carefully explaining to the user how to operate a mouse, what a menu is, et cetera, ad nauseum. Please, if someone doesn't know how to use their own computer, or a GUI, then they should sit down and learn *before* they start up a major piece of software.
>
> Second, what information they do provide seems to assume that the user is stupid. Utter nonsense! Most users, in my experience, are some combination of clueless and intimidated, not stupid. Besides, if someone is truly slow on the uptake, they need help that a manual for a piece of computer software can't give.

Editor's Note: With this in mind, I've instructed all of the other authors to avoid patronizing you, the reader, and to be more pedagogical than pedantic. As for those who are too lazy to read and understand the manuals — well, as we say here in America, there's no such thing as a free lunch. - jw

1.3 The LyX Interface

1.3.1 Basic File Operations

Under the File menu are the 9 basic operations for any word processor in addition to some more advanced operations:

- New
- New from Template
- Open
- Close
- Save
- Save As
- Revert
- Print
- Exit

They all do pretty much the same thing as in other word processors, with a few minor differences. The File ▷ New from Template command not only prompts you for a name for the new file, but also prompts you for a template to use. Selecting a template will automatically set certain layout features for the document, features you would otherwise need to change manually. They can be of use for certain classes, especially those for writing letters [see sec. 3.1.2]

Note: There is no "default file" or document named "Untitled" or "scratch." Unless you tell LyX to open a file or create a new one, that big, blank space is just that — a big, blank space.

The Revert command is useful if more people work on the same document at the same time[1]. It will simply reload the document from disk. You can of course also use it if you regret that you changed a document and want to restore it to the last save.

The second matter of note concerns the commands File ▷ Close and File ▷ Exit. They both feature a "nag box" to save us all from our own stupidity. That is, if you try to close a file with changes [or exit LyX], you'll be informed that there are unsaved files.

[1] If you plan to do this, you should check out the Version Control feature in LyX also. Read *Extended Features*.

1.3.2 Basic Editing Features

Like most modern word processors, LyX can perform cut and paste operations on blocks of text, can move by character, word, or screenful of text, and can delete whole words as well as individual characters. The next four sections cover the basic LyX editing features and how to access them. We'll start with cut and paste.

As you might expect, the Edit menu has the cut and paste commands, along with various other editing features. Some of these are special and covered in later sections. The basic ones are:

- Cut
- Copy
- Paste
- Find & Replace...

The first three are self-explanatory. One thing to note: whenever you delete a block of text that you've selected, it's automatically placed in the clipboard. That is, the Delete and Backspace keys also functions as the Cut command. Also, if you've selected text, be careful. If you hit a key, LyX will completely delete the selected text and replace it with what you just typed. You'll have to do an Undo to get back the lost text.

The Edit ▷ Find & Replace... item opens the Find & Replace dialog. The text you want to find goes in the Find box. Once you've found a word or expression, LyX selects it. Hitting the Replace button replaces the selected text with the contents of the Replace with box. You can click to search again to skip the current word.

Hit Replace All to replace all occurrences of the text in the document automatically.

The Case sensitive toggle button can be used if you want the search to consider the case of the search word. If the toggle is set, searching for "`Match`" will not match the word "`match`".

The Match Word toggle button can be used to force LyX to only find complete words. I.e., searching for "`match`" will not match "`matches`", "`matchbox`", etc.

1.3.3 Undo and Redo

If you make a mistake, you can easily recover from it. LyX has a large-capacity undo/redo buffer. Select Edit ▷ Undo to undo some mistake. If you accidently undo too much, use Edit ▷ Redo to "undo the undo." The undo mechanism is currently limited to 100 steps to minimise memory overhead.

Notice that if you revert back all changes to arrive to the document as it was last saved, the "changed" status of the document is unfortunately not reset. This is a consequence of the 100 step undo limit, above.

The Undo and Redo work on almost everything in LyX. They have some quirks, too. They won't Undo or Redo text character by character, but by blocks of text.

That can take some getting used to; you'll have to play with Undo and Redo to get a feel for just how much they'll undo/redo, and after time, you'll hopefully appreciate how it works.

1.3.4 Basic Mouse Bindings

We're not going to go into all of the mouse bindings here. Some of the other sections of this manual cover specific operations you can do with the mouse. Instead, we're going to cover the most basic mouse operations.

1. Motion

   - Click the *left mouse button* once anywhere in the edit window. The cursor moves to the text under the mouse.

2. Selecting Text

   - Hold down the *left mouse button* and drag the mouse. LYX marks the text between the old and new mouse positions. Use Edit ▷ Copy to create a copy of the text in LYX's buffer.
   - Re-position the cursor and then paste the text back into LYX using Edit ▷ Paste.

3. Footnotes, Margin Notes, Figure and Table Floats, etc.

   Single click the left mouse button to open or close any of these. Also check the appropriate section of this manual for more details.

4. Tables

   Single click the right mouse button to open a dialog that will allow you to manipulate the table.

1.3.5 Basic Key Bindings

Again, we're not going to cover all of the keybindings. Be aware that there are at least two different primary binding maps: CUA and Emacs. I guarantee you will cuss when you press Control-d to delete a character, and it starts up a DVI previewer instead (or vice versa).

Some keys, like Page Up, Page Down, Left, Right, Up, and Down, do exactly what you expect them to do. Other keys don't:

Tab
: There is no such thing as a tab stop in LYX. If you don't understand this, go read Sections 3.2.1 and 3.3, especially Section 3.3.6, right now. Yes, right now. If you're still confused, look in the *Tutorial*.

Esc This is the "cancel key." It's used, generically, to cancel operations. Other parts of the manual will go into greater detail about this.

Home and End These move the cursor, respectively, to the beginning and end of a line, unless you are using the Emacs bindings where they jump to the beginning or end of the file.

Backspace and Delete *If* you have your keyboard set up correctly under the X Windows System, Backspace works as expected and Delete deletes the character under the cursor [if no text is selected].

If you haven't set up your keyboard under X, or have no idea what we mean by that, go read section 2.3 immediately. You'll save yourself a lot of headaches.

Then there are the modifier keys:

Control- This has a couple of different uses, depending on which keys it's used in combination with:

- With Backspace or Delete, it deletes an entire word instead of a single character.
- With Left and Right, it moves by words instead of characters.
- With Home and End, it moves to the beginning and the end of the document, respectively.

Shift- Use this with any of the motion keys to select the text between the old and new cursor positions.

Meta- This is the Alt key on many keyboards, unless your keyboard has a distinct Meta key. Unfortunately, X sometimes has their functionality swapped, so if you have both keys, you will need to do a little trial and error to find out which one actually performs the Meta- function. This key does many different things, but it also activates the *menu accelerator keys*. If you use this in combination with any of the underlined letters in a menu or menu item, it selects that menu item.

For example, the sequence "M-e s" brings up the "Text Style" menu. Typing "M-f" opens the File menu.

There are also other things bound to the Meta- key, but you'll have to check in the *Reference manual* for more info.

Hopefully, you'll learn more and more keybindings and short-cut keys as you use LYX, because most mouse actions will prompt a small message in the minibuffer which describe the name of the action, you've just triggered, and any existing keybindings for that action. The notation for the keybindings is very similar to the notation used in this documentation, so you should not have any problems understanding it. However, notice that Shift-modifiers are explicitly mentioned, so "M-p S-A" means Meta-p followed by a capital A. "S-C-S" means Shift-Control-s.

1.4 Using LyX with Other Programs

1.4.1 Importing ASCII files

You can import text from an ASCII file using the File ▷ Import ▷ Ascii text as lines or File ▷ Import ▷ Ascii text as paragraphs options.

File ▷ Import ▷ Ascii text as_lines puts each line of the file into its own LyX paragraph. This is useful if you're importing a text file with a simple list in it. However, if your text file contains paragraphs in it, LyX will mangle the paragraphs if you use this form of import.

File ▷ Import ▷ Ascii text as paragraphs preserves paragraphs in text files. Often in a text file, you didn't put the contents of an entire paragraph on one line. You used Return to break up the paragraph into separate lines. Using the as paragraphs, LyX won't mangle such paragraphs. Anything between two consecutive blank lines goes into its own LyX paragraph. Remember: you must make sure there is a *completely blank* line between each and every paragraph in your text file. If not, LyX might end up merging two paragraphs.

1.4.2 Cut and Paste Between LyX and Other X Programs

The Cut, Copy, and Paste operations will transfer text to and from LyX. You can copy text from LyX to another window in this way: Select the text that you want to copy, then go to the destination window and paste the text with the middle mouse button.

Pasting text into LyX also works much the same way as in X. Select the text with the mouse in another X window. Go to the Lyx window and paste the text with the middle mouse button.

2 LyX Setup and Supporting Applications

2.1 Introduction

If you're using LyX on a system someone else has set up for you, then you can safely skip this chapter. It describes all of the things you need beyond the LyX binary and files distributed with it.

If you're installing LyX on your system, *you should read the README's that came with the LyX distribution and then* Help ▷ LaTeX Configuration. Do that first. This chapter does not describe installation or setup of the LyX binary [Well, not everything…]. It does describe all of the things you'll need to use LyX to its fullest.[1]

2.2 Basic LyX Setup

There are two ways to run LyX. The first way is to install LyX and all of its support files on your system. Of course, you need root (administrator) privileges to do that. The second way to run LyX doesn't require root access, letting you "install" LyX somewhere in your own account. LyX will automatically detect where it is as long as the supporting directories are put in the correct places.

There are several features of LyX that can be configured from inside LyX, without resorting to configuration files. First, LyX is able to inspect your system to see what programs, LaTeX document classes and LaTeX packages are available. It uses this knowledge to give reasonable defaults to several `preferences` variables. Although this configuration has already been done when LyX was installed on your system, you might have some items that you installed locally and which are not seen by LyX. To force LyX to re-inspect your system, you should use Tools ▷ Reconfigure. You should then restart LyX to ensure that the changes are taken into account. As far as LaTeX classes and packages are concerned, you will find information about what has been found under Help ▷ LaTeX Configuration.

The second set of settings that you might want to change comprises all the document-level setting that you can change via the Document ▷ Settings dialog. To do this, open a scrap document, set all these options according to your taste and save them with the Save as Document Defaults button in the Document dialog. This will create a

[1] This is basically where we decided to document a bunch of info about running LyX, including what other programs you'll need to make LyX useful.

template named `default.lyx` which is automatically loaded by LyX when you open a document without template such that the settings are automatically set-up as you defined them.

There are many other user-configurable options that you can feed to LyX. Upon startup, LyX reads a global options file called `lyxrc.defaults`. It will then attempt to read a file called `preferences`[2]

The Tools ▷ Preferences dialog can be used to change these options; the document *Customization* contains more information about the preferences dialog and these configuration files.

2.3 Setting Up the X Keyboard [obsolescent]

To use LyX properly, X *must* be set up correctly. This is especially vital if you're using the international support features of LyX and want to use non-English keyboard mappings. On modern distributions, this likely has been taken care of, but if not, you must do this yourself. Administrators of large systems often neglect this, so don't assume that you're safe if you're using a large system. Also ordinary users can instruct X how to use his or her keyboard.

2.3.1 xmodmap and xkeycaps

First of all read the man pages for these two programs. They are your best friends when you are trying to set up X key mapping correctly. If you don't have them, install them.

2.3.1.1 xmodmap

This document contains no information on how to use `xmodmap`. There is a sample `.Xmodmap` file in *Customization*. To load the new X keyboard mappings, place the command `xmodmap .Xmodmap` somewhere in your startup scripts [e.g. `.cshrc`, `.profile`, `.login` or `.xinitrc` are possible].

2.3.1.2 xkeycaps

This program brings up a graphical version of your keyboard, allows you to make modifications, and then spits those modifications out to the standard output in a form readable by `xmodmap`. It is very useful when you're trying to design a new `.Xmodmap` file, though it will require you to do a bit of cut-and-pasting.

[2] The `preferences` file is found in different directories on different systems. This directory is called LyX's *user directory*. To find out where it is, use Help ▷ About LyX. (You may set up an alternative user directory from the command line, using the switch `-userdir`.)

2.3.2 Modifiers and Mode_switch

LyX supports three modifiers: Shift [S-], Control [C-], and Meta [M-]. Moreover, if one of the keys of your keyboard is configured as a Compose key, then you can use it to enter some characters not available on your keyboard. This compose key can be used either as a modifier (like Shift or Control) or as a prefix key. Here are some examples of what you can do with a Compose key:

- Compose+e+' → é
- Compose+O+R → ®
- Compose+1+2 → ½
- Compose+<+< → «

This input method is particularly handy when you use accented characters only from time to time. It works by default for latin1 characters, but other input methods will be used if you setup your locale correctly.

2.3.3 Helpful Hints and Tips

First, open up two xterminals. Use one to edit a new `.Xmodmap` file and run `xkeycaps` from the other. Using `xkeycaps`, remap your keyboard the way you want it. There's a button in `xkeycaps` to output the new keymap. Once you hit it, `xkeycaps` will spit a bunch of stuff on the xterm you executed it from. Just copy and paste all of that into your `.Xmodmap` file, and you're done.[3]

Also, there are some things you can do to help you get oriented. Try executing the command `xmodmap -v -pm`. This will show you all of the currently active modifiers. Also try `xmodmap -v -pke | more` to see which keycode numbers are mapped to which symbolic names. It will also give you some idea of the syntax of the `.Xmodmap` file.

There's one thing you'll need to check. Make sure that your Delete and BackSpace keys are *not* defined as the same key symbol by X! Note that giving these two keys unique symbol names will not necessarily alter the behavior of your programs. Some programs bind Delete and BackSpace to the same operation. Emacs is one. Other programs, however, use Delete and BackSpace for different operations. LyX is one of these programs, and if you have Delete and BackSpace labeled with the same key symbol name, you'll have trouble using LyX.

[3] You could also save yourself some typing by executing `xkeycaps > .Xmodmap`. This will create a usable map file.

2.4 LaTeX

If you want to do more with LyX than simply create documents and spit out `.tex` files, you'll need LaTeX.

In case you were wondering, LaTeX is a markup language front end for TeX, a document preparation system invented in 1984 by Donald Knuth.[4] TeX takes a set of commands in an ASCII file and converts it to a "device-independent" format, or Dvi, for short. The Dvi file can then be sent to printers. TeX is programmable, and LaTeX is nothing but a [really huge] set of TeX macros. LaTeX will typically come as part of a TeX distribution, so all you need is a TeX package.

Note that on some old systems you may find that only LaTeX 2.09 is installed (as opposed to the more current LaTeX 2_ε). LyX cannot be used with LaTeX 2.09.

If you're using Linux, LaTeX 2_ε should have come with your distribution. For other systems, you might need to install LaTeX yourself.You can obtain a LaTeX distribution (and anything and everything related to TeX and LaTeX) from a Comprehensive TeX Archive Network (CTAN) mirror. A complete list of mirrors may be found at
http://www.ctan.org

2.5 Dvips and Ghostscript

2.5.1 What You Need

There's one more step you need to take if you want to print your LyX documents. Obviously, you'll need to make sure your printer is configured [see next section]. You'll also need to install these programs (or compatibles), if you don't have them already:

- `dvips`
- `ghostscript`
- `xdvi`
- `ghostview`

The latter two programs are previewer for files in Dvi and PostScript®[5] format. If you don't know what a DVI file is, you've probably also never worked with LaTeX and should read the *Tutorial* document before proceeding further. `dvips` converts DVI files into PostScript, which is the format most printers use nowadays. For those of you using dot-matrix and inkjet printers, you'll want to filter the PostScript through `ghostscript`, which is capable of creating output for a variety of printers. The

[4] A note about pronunciation: TeX originated from the Greek letters, $\tau\epsilon\chi$, which rhymes with "blech." That's how you pronounce "TeX" and "LaTeX." [If you're American, just pronounce the "X" as a "k" and you've got it.]

[5] PostScript® is a registered trademark of Adobe Systems Incorporated, and is the main page description language in the UN*X world.

following section on printer setup describes how to do this automatically every time you print. For now, we'll concentrate on `dvips`.

2.5.2 Dvips

Whether you'll be running LyX on a large system or a Linux box at home, you should configure `dvips`. `dvips` will either "print" into a file, or send output directly to the printer, depending on how it's configured. If it is set up to print to a file, and if no filename is specified, it will simply turn `foo.dvi` into `foo.ps`. Most systems have `dvips` set up to send output to the default printer. For LyX, you'll want the flexibility to do both.

If you are not a mood to configure `dvips` to adapt its output to your printer, you can safely skip this section. Be warned however that the output will not match the quality that you could expect from your printer. At least, it will print.

If you are using teTEX (a TEX distribution which is particularly popular on Linux), you should run the program `texconfig`. To make the name of a new printer recognized by `dvips` you should then select menu entry Dvips, then add. Enter the required parameters and, before exiting, remember to select the function Rehash.

Let's turn now to manual configuration: in order to inform `dvips` how to automagically convert a `.dvi` file into a `.ps` file adapted to printer `foo`, you need to have a config-file, "`config.foo`," lying around somewhere. Typically, the `config.*` files for `dvips` will be in `/usr/lib/texmf/dvips` in most TEX distributions. Your system will probably be different, of course, so just look under the main TEX directory for a subdirectory called "`dvips`." It'll be there somewhere.

Typically, there will be at least one config-file: `config.ps`. This file is the default configuration file, which is *always* read by dvips[6]. Read this file and see what options could need to be changed for your particular printer. Then create a file `config.foo` containing only the relevant lines.

There's at least one thing you need to do to the config-file. There may exist a line that looks like, "`o | lpr`" [without the quotes, of course. . .]. Change it to "`o | lpr -Pfoo`", so that the output is sent by default to printer `foo`. However, you should probably investigate the entries "`M`" and "`D`", which define respectively the Metafont mode and the resolution of the printer. If you do not know what a Metafont mode is, you can see it as a printer driver: it adapts the design of TEX fonts to ensure that they give the best possible result on your printer. Be warned however that, if you define different Metafont modes for different printers, `dvips` will generate several copies of your TEX fonts on disk, and these take valuable space.

Once you are satisfied that your printers are correctly configured, you should tell LyX to make use of this configuration. To do this, you should launch the Preferences dialog (Tools ▷ Preferences), select the Printer tab, and set the entries Adapt output and Spool command.

You can use as many configuration files as you like, one for each of your printers.

[6] In particular, this file is not necessarily connected to the existence of a file named `ps`.

The default printer for LyX can be specified from the Preferences dialog or with the PRINTER environment variable. You can also choose the desired printer from inside LyX, as described in a later section. Once you've done all that, you can print to either a PostScript printer or file from LyX.

If your printer doesn't understand PostScript®, you'll need to use ghostscript as a filter for your print spooler. That's covered in numerous HOWTO's and manuals. We also have a section that covers a little bit of this.

Some people don't seem to like using the dvips plus ghostscript combination. As alternative, you can use a program that converts the DVI file directly into your printer language. You can specify this program in the Preferences dialog, too. There is a major disadvantage to this method. You can't include any PostScript files, such as graphics, in your documents, since the printer-specific conversion programs don't understand PostScript®. For that reason, the LyX team highly recommends using dvips and ghostscript for printing.

2.5.3 Ghostscript, Xdvi and Ghostview

Xdvi and ghostview are viewers. The former handles .dvi files, while the later interfaces with ghostscript to allow you to view PostScript files.

A quick note on both of these programs. Both automatically update themselves if the viewed file[7] changes. You can also force an update. So, once you've opened one of these two viewers, there's no reason to close it. Also, both programs are functionally the same, providing all of the same features.

The LyX team recommends using xdvi for fine tuning documents. Why? It's faster; there's one less layer of processing you need to do before you can view the changes. Here's an example:

1. Use xdvi to preview a document from LyX, and leave it running.
2. Make changes to the document using LyX.
3. To view those changes, just choose View ▷ Update ▷ DVI. When LaTeX's all done, click on the xdvi window, and voilà! xdvi will update itself.

Now, this doesn't mean ghostview is useless. ghostview is better suited to those occasions where you *must* view the PostScript version of the document. For repeated changes that aren't PostScript® dependent, you're better off previewing with xdvi. There is an alternative to ghostview which sports a much better interface: gv. LyX will automatically use it instead of ghostview if it is available.

2.6 The Printer

Anyone working on a large system shouldn't have any problems here. Your sysadmin [or you, if you are the sysadmin] should already have the printers set up for your

[7] That means the .dvi or .ps file, not the files used to make these.

system. All you need to do is find out the name of the printer you want to use, and configure your setup as described in the last section.

Those of you using Linux, however, may have a bit more work to do. Many people now install Linux from an ISO image of one of the popular distributions. They follow the install instructions, get Linux up and running, but never realize that they need to set up their printer. The more desktop friendly distributions may do this for you automatically. However, if you find that you need to do this by hand, we've written a little something to help you out with that; check out the "*A Printer Tutorial*" chapter in the *Customization* manual for help.

3 LyX Basics

3.1 Document Types

3.1.1 Introduction

Before you do anything else, before you ever start writing a document, you need to decide what *type* of document you want to edit. Different types of documents use different types of spacing, headings, numbering schemes, and so on. Additionally, different documents use different paragraph environments, and format the title of your document differently.

A *document class* describes a group of properties common to a particular set of documents. By setting the document class, you automagically select these properties, making it easier to create the type of document you want. If you don't choose a document class, LyX picks one for you by default. So, it behooves you to change the class of your document.

Read on for info about the document classes you can choose from LyX, and how to fine-tune some of their properties.

3.1.2 The Various Document Classes

3.1.2.1 Overview

There are five standard document classes in LyX. They are:

Article for basic articles

Report for basic reports

Book for writing a book

Letter for US-style letters

Slides is used to make transparencies

There are also some non-standard classes, which LyX only uses if you have a LaTeX setup that supports them:

Aapaper Journal articles in the style and format used in Astronomy & Astrophysics

Amsart Journal articles in the style and format used by the AMS [American Mathematical Society]. There are three amsart layouts available. The standard one uses a typical numbering scheme for theorems, *etc.*, that prepends the section number to the number of the result. All result-type statements (propositions, corollaries, and so on) are sequenced together, but definitions, examples, and the like have their own sequence. The "sequential numbering" scheme does not place the section number with each result, but numbers them throughout the article in a single sequence. Each type of result gets its own sequence. There is also a layout that dispenses with numbering of statements altogether.

Amsbook Books in the style and format used by the AMS. Only the standard numbering scheme is provided, under the assumption that you would not want to number results consecutively throughout a book, and that you would need to number results.

Dinbrief für Briefe nach deutscher Art

Foils is used to make transparencies, but is better than slides

Linuxdoc Used with the SGML-tools package (formerly known as LinuxDoc). It allows LyX to produce SGML output. SGML is a markup language and is the predecessor to HTML. The SGML-tools package allows you to convert SGML to HTML or to the format used by `man` pages.

Paper for use with the `paper` LaTeX document class [not in all LaTeX distributions]

Revtex is used to write articles for the publications of the American Physical Society (APS), American Institute of Physics (AIP), and Optical Society of America (OSA). This class is not completely compatible with all LyX features.

We won't go into any detail about how to use these different document classes here. You can find all the details about the non-standard classes in the *Extended Editing* manual. Here, we will settle with a list of some of the common properties of all of the document classes.

3.1.2.2 Selecting a Class

You can select a class using the Document▷Settings dialog. Select the class you want to use, and make any fine tunings of the options you may need.

3.1.2.3 Properties

Each class has a default set of options. Here's a quick table describing them:

	Pagestyle	Sides	Columns	Max. sectioning level
article	Plain	One	One	Section
report	Plain	One	One	Chapter
book	Headings	Two	One	Chapter
letter	Plain	One	One	none
linuxdoc	Plain	One	One	Section
aapaper	Plain	Two	Two	Section
amsart	Headings	One	One	Section
dinbrief	Plain	One	One	none
paper	Headings	One	One	Section

There is no default value of Extra Options for any of these classes.

You're probably also wondering what "Max. sectioning level" means. There are several paragraph environment used to create section headings. Different document classes allow different types of section headings. Only two use the Chapter heading; the rest do not and begin instead with the Section heading. Some document classes, such as the three for letters, don't use any section headings. In addition to Chapter and Section headings, there are also Subsection headings, Subsubsection headings, and so on. We'll describe these headings fully in section 3.3.4.

3.1.3 Fine-tuning the Defaults

Okay, we know we never told you what most of these "default options" set by the Class button do. That's what this section is for.

Pagestyle — This is another list, containing five options. It controls what sorts of headings and page numbers go on a page:[1]

- Default — Use default pagestyle of current class.
- Empty — No page numbers or headings.
- Plain — Page numbers only.
- Headings — Page numbers and either the current chapter or section title and number. Whether LyX uses the current chapter or the current section depends on which is the maximum sectioning level.
- Fancy — This allows you to create fully customizable headers and footers if you have the fancyhdr package installed. At the moment, support in LyX is limited to this setting. To use the full power of this package, you have to resort to magic codes in your preamble. Check the documentation for the fancyhdr package for more details.

[1] LaTeX does this part.

Sides
: No, LyX can't make your printer print on both sides of a sheet of paper! However, it can use a different format for odd-numbered pages than even-numbered pages. This way, if you *do* have a printer that duplexes[2], your page number will always be in the upper right corner of the page and the left margin will have extra room for a binding.

There are two radio buttons here: One for single-sided documents, Two for double-sided documents.

Columns
: Yes, this does control how many columns each page has. You can choose, using the toggle buttons, One or Two for the number of columns.

Note that LyX won't show two columns on screen. That's impractical, often unreadable, and not part of the WYSIWYM concept. However, there *will* be two columns in the generated output.

Extra Options
: The LaTeX command `\documentclass` takes several options. LyX sets some of these automatically for you. This text box allows you to enter in others. Just type in a comma-separated list of options. See a good LaTeX book to find out what kinds of additional options you can use.

Separation
: This has its own section. See sec. 3.2.1 for a description of what this does.

3.1.4 Paper Size, Orientation, and Margins

There are several other options to set in the Document Settings dialog. All of them are global options, but they have special purposes and only affect certain features. We describe what these options do in the same section that describes the features they affect.

There are two options that affect the overall layout of the document, so we'll describe them here. You'll find them in the Paper dialog under the Layout menu:

Orientation
: Two toggle buttons choose whether to print the output as Landscape or as Portrait.

Papersize
: What size paper to print on. The choices are

- Default
- A3, A4, A5
- B3, B4, B5
- US Letter

[2] i.e. prints on both sides of a sheet of paper

- US legal
- US executive
- Custom

Some of these settings require you to have the geometry package installed. This package will also allow you to set the margins in the Paper dialog.

3.1.5 Important Note:

If you change a document's class, LyX has to convert *everything* into the new class. That includes the paragraph environments. Some paragraph environments are standard; all of the document classes have them. Some classes have special paragraph environments, however. If this is the case, and you change document classes, LyX sets the missing paragraph environments to Standard and places an error box at the beginning of the paragraph. Just click on them and you'll get a message dialog that tells you about the conversion and why it failed.

3.2 Paragraph Indentation and Separation

3.2.1 Introduction

Before describing all of the various paragraph environments, we'd like to say a word or two about paragraph indentation.

Everyone seems to have their own convention for separating paragraphs. Most Americans indent the first line of a paragraph. Others don't indent but put extra space between the paragraphs. LyX uses the same convention you find among typographers. The *first* paragraph of a section, or after a figure, an equation, a table, a list, etc., is *not* indented. Only a paragraph following another paragraph gets indented. Some people don't like this convention, but if you want to use indented paragraphs, you'll have to live with it.[3]

The space between paragraphs, like the line spacing, the space between headings and text — in fact, all of the spacings for just about everything are pre-coded into LyX. As we said, you don't worry about how much space to add between what. LyX takes care of that. In fact, these pre-coded vertical spacings aren't a single number but a range. That way, LyX can squish or stretch the space between lines to make sure figures fit on a page with text, so that sections don't start at the bottom of a page, and so on.[4] However, pre-coded doesn't mean you can't change them. LyX gives you the ability to globally change *all* of these pre-coded spacings. We'll explain more later.

[3] There is a way to force LaTeX to indent all paragraphs. LyX won't show this, of course, but LaTeX *will* print it that way. You'll need to get a special package and insert an appropriate command in the preamble.

[4] Actually, LaTeX does this when LyX goes to produce a printable file.

3.2.2 Global Indentation Method

To select the default method of separating paragraphs, select Indent or Skip to indent paragraphs or add extra space between paragraphs, respectively.

3.2.3 Fine-Tuning

You can also change the separation method of a single paragraph. Open the Edit ▷ Paragraph Settings dialog and toggle the No Indent button to change the state of the current paragraph. If paragraphs indent by default, this button will be inactive at first. If paragraphs have no indentation but use extra space for separation, this button will be completely ignored (you can't indent a single paragraph by toggling this).

You should only need to change the indentation method for a single paragraph if you need to do some fine-tuning. Typically, you'll select Indent or Skip for the entire document and edit away.

3.2.4 Changing Line Spacing

In the Document ▷ Settings dialog you can choose your line spacing provided you have the setspace package installed.

3.3 Paragraph Environments

3.3.1 Overview

The paragraph environments correspond to the various

`\begin{`*environment*`} ... \end{`*environment*`}`

command sequences in an ordinary LaTeX file. If you don't know LaTeX, or the concept of a paragraph environment is totally alien to you, we urge you to read the *Tutorial.* The *Tutorial* also contains many more examples than this section does.

A paragraph environment is simply a "container" for a paragraph which gives that paragraph certain properties. This can include a particular style of font, different margins, a numbering scheme, labels, and so on. Additionally, you can "nest" the different environments inside one another, allowing one environment to inherit some of the properties of another. The different paragraph environments totally replace the need for messy tab stops, on the fly margin adjustment, and other hold-overs from the days of typewriters. There are several paragraph environments which are specific to a particular document type. We'll only be covering the most common ones here.

To choose a new paragraph environment, use the pull-down box on the left end of the toolbar. LyX will change the environment of the *entire* paragraph in which the cursor sits. You can also change the environment of an entire group of paragraphs if you select them before choosing the new environment.

Note that hitting Return will *typically* create a new paragraph using the Standard paragraph environment. We say "typically" because this isn't always the case.[5] Usually, starting a new paragraph resets both the paragraph environment and the nesting depth [more on nesting in section 3.4]. At the moment, all this is context-specific; you're better off expecting Return to reset the paragraph environment and depth. If you want a new paragraph to keep the current environment and depth, use M-Return instead.

3.3.2 Standard

The default paragraph environment is Standard for most classes. It creates a plain paragraph. If LyX resets the paragraph environment, this is the one it chooses. In fact, the paragraph you're reading right now [and most of the ones in this manual] are in the Standard environment.

You can nest a paragraph using the Standard environment in just about anything else, but you can't really nest anything in a Standard environment.

3.3.3 Document Titles

A LaTeX title page has three parts: the title itself, the name[s] of the author[s] and a "footnote" for thanks or contact information. For certain types of documents, LaTeX places all of this on a separate page along with today's date. For other types of documents, the title "page" goes at the top of the first page of the document.

LyX provides an interface to the title page commands through the paragraph environments Title, Author, and Date. Here's how you use them:

- Put the title of your document in the Title environment.
- Put the author name in the Author environment.
- If you want the date to have a certain appearance, want to use a fixed date, or want other text to appear in place of today's date, put that text in the Date environment. Note that using this environment is optional. If you don't provide any, LaTeX will automatically insert today's date.

Be sure to do this at the top of the document. You can use footnotes to insert "thanks" or contact information.

[5] If you are in one of these environments:

- Quote
- Quotation
- Verse
- Itemize
- Enumerate
- Description
- List

LyX keeps the old paragraph environment when you hit Return, rather than resetting it to Standard. LyX will still reset the nesting depth, however.

3.3.4 Headings

There are nine paragraph environments for producing section headings. LyX takes care of the numbering for you. All you need to do is decide what you're going to call section 3 of chapter 9.

3.3.4.1 Numbered Headings

There are 6 numbered types of section headings. They are:

1. Chapter
2. Section
3. Subsection
4. Subsubsection
5. Paragraph
6. Subparagraph

LyX labels each heading with a series of numbers, separated by periods. The numbers describe where in the document you are. These headings all subdivide your document into different pieces of text. For example, suppose you're writing a book. You group the book into chapters. LyX does similar grouping:

- Either Chapter or Section is the maximum sectioning level.
- Chapters are divided into Sections
- Sections are divided into Subsections
- Subsections are divided into Subsubsections
- Subsubsections are divided into Paragraphs
- Paragraphs are divided into Subparagraphs

Note: not all document types use the Chapter heading as the maximum sectioning level. In that case, the Section is the top-level heading.

So, if you use the Subsubsection environment to label a new sub-subsection, LyX labels it with its number, along with the number of the subsection, section, and, if applicable, chapter that it's in. For example: the fifth section of the second chapter of this book has the label “2.5”.

3.3.4.2 Unnumbered Headings

There are 3 types of unnumbered section headings. They are:

1. Section*
2. Subsection*
3. Subsubsection*

The "*" after each name means that these headings are not numbered. They work the same as their numbered counterparts.

3.3.4.3 Changing the Numbering

You can also alter which sectioning levels get numbered and which ones appear in the Table of Contents. Now, this doesn't remove any of the levels; that's preset in the document class. Certain classes start with Chapter and go down to the Subparagraph level. Others start at Section. Similarly, not all document classes number all sectioning levels. Most don't number Paragraph or Subparagraph. This is something you can change.

Open the Document ▷ Settings dialog. You should see a counter labelled Section number depth under the Extra tab. This counter controls how far down in the sectioning hierarchy LyX numbers a section heading. Unfortunately, the number you choose with the slider is really goofy, so here's a table of values and what they do:

Sec. Num. Depth value:	LyX numbers these section headings:
-2	no numbering of any kind
-1	add Parts
-0 and 0	add Chapters
1	add Sections
2	add Subsections
3	add Subsubsections
4	add Paragraphs
5	add Subparagraphs

The increasing numbers are cumulative: a setting of "0" will number parts and chapters, while "2" will number parts, chapters, sections, and subsections. Of course, if you're using a document class that doesn't use part or chapter headings (*e.g.* the default article class), then the numbering begins at the Section heading, and "0" also corresponds to "no numbering."

There's another counter in the dialog, called Table of contents depth. It works the same way as Section numbering depth, only it controls which sectioning levels appear in, you guessed it, the Table of Contents. This is a great control to have. Suppose you wanted to number *all* sectioning heading, but you only wanted Chapters, Sections, and Subsections in the Table of Contents. You'd just set Section numbering depth to "5" and Table of contents depth to "2" and voilà! You're all set.

3.3.4.4 Special Information

The following information applies to Chapter, Section, Subsection, Subsubsection, Paragraph, Subparagraph as well as Section*, Subsection*, and Subsubsection*:

- You cannot use a margin note in any of these environments.
- You can only use inlined math in these environments.
- You cannot nest other environments into these environments.
- You can use labels and cross-references to refer to their numbers.

As for examples of these paragraph environments - look around you! We're using them everywhere in the manuals.

3.3.4.5 Creating an Appendix

To create an appendix, simply start by adding a new chapter or section heading. Move the cursor back to the beginning of the heading and select Document ▷ Start Appendix Here. A red/brown box will be drawn around the remainder of the file to indicate there is something special about it, and the numerical chapter or section label(s) will be changed to a letter.

3.3.5 Quotes and Poetry

LyX has three paragraph environments for writing poetry and quotations. They are Quote, Quotation, and Verse. Forget the days of changing linespacing and twiddling with margins. These three paragraph environments already have those changes built-in. They all widen the left margin and add a bit of extra space above and below the text they contain. They also allow nesting, so you can put a Verse in a Quotation, as well as in some other paragraph environments.

There is another feature of these three paragraph environments: they do *not* reset to Standard when you start a new paragraph. So, you can type in that poem and merrily hit Return without worrying about the paragraph environment changing on you. Of course, that means that, once you're done typing in that poem, you have to change back to the Standard environment yourself.

3.3.5.1 Quote and Quotation

Now that we've described the similarities of these three environments, it's time for the differences. Quote and Quotation are identical except for one difference: Quote uses extra spacing to separate paragraphs and never indents the first line. Quotation *always* indents the first line of a paragraph and uses the same line spacing throughout.

Here's an example of the Quote environment:

> This is in the Quote environment. I can keep writing, extending this line out further and further until it wraps. See - no indentation!
>
> Here's the second paragraph of this quote. Again, there's no indentation, but there is extra space between me and the other paragraph.

That ends that example. Here's another example, this time in the Quotation environment:

> This is in the Quotation environment. If I keep writing, you'll see the indentation. If your country uses a writing style that shows off new paragraphs by indenting the first line, then Quotation is the environment for you! Well, you'd use it *if* you were quoting other text.
>
> Here's a new paragraph. I could ramble on and on, like a politician at election time. If I did that, though, you'd get bored.

That was our other example. As the example notes, Quote is for those people who use extra space to separate paragraphs. They should put quotes in the Quote environment. Those who use indentation to mark a new paragraph should use the Quotation paragraph environment for quoted text.

3.3.5.2 Verse

Verse is a paragraph environment for poetry, rhymes, verses, and so on. Here's an example:

> This is in Verse
> Which I did not rehearse!
>
> It could be much worse. This line could be long, very long, oh so long, so very long that it wraps around. It looks okay on screen, but in the printed version, the extra lines are indented a bit more than the first. Okay, so it's turned to prose and doesn't rhyme anymore. So sue me.
>
> To break a line
> And make things look fine
> Use C-Return.

As you can see, Verse does not indent both margins. Each stanza of the verse or poem is in its own paragraph. To separate the individual lines of a stanza, use the `break-line` function, C-Return.

3.3.6 Lists

LyX has four different paragraph environments for creating different kinds of lists. In the Itemize and Enumerate environments, LyX labels your list items with bullets or numbers, respectively. In the Description and List environments, LyX lets you provide your own label. We'll present the individual details of each type of list next after describing some general features of all four of them.

3.3.6.1 General Features

The four paragraph environments for lists differ from the other environments in several ways. First, LyX treats each paragraph as a list item. Hitting Return does *not* reset the environment to Standard but keeps the current environment and creates a new list item. The nesting depth is typically reset, however. If you want to keep both the current nesting depth and paragraph environment, you should use M-Return to break paragraphs.

You can nest lists of any type inside one another. In fact, LyX changes the labels on some list items depending on how its nested. If you intend to use any of the list paragraph environments, we suggest you read all of section 3.4.

3.3.6.2 Itemize

The first type of list we'll describe in detail is the Itemize paragraph environment. It has the following properties:

- Each item has a particular bullet or symbol as its label.
  - LyX uses the same symbol for all of the items in a given nesting level.
  - The symbol appears at the beginning of the first line.
- The items can be any length. LyX automatically offsets the left margin of each item. The offset is always relative to whatever environment the Itemize list may be in.
- If you nest an Itemize environment inside another Itemize environment, the label changes to a new symbol.
  - There are four different symbols for up to a four-fold nesting.
  - LyX always shows the same symbol, an asterisk, on screen.
  - See section 3.4 for a full explanation of nesting.

Of course, that explanation was also an example of an Itemize list. The Itemize environment is best suited for lists where the order doesn't matter.

We said that different levels use different symbols as their label. Here's an example of all four possible symbols. Note that those of you reading this manual online won't see any difference.

- The label for the first level Itemize is a large black dot, or bullet.
    - The label for the second level is a dash.
        - The label for the third is an asterisk.
            - The label for the fourth is a centered dot.
        - Back out to the third level.
    - Back to the second level.
- Back to the outermost level.

These are the default labels for an Itemize list. You can customize these labels in the Document ▷ Settings dialog in the Bullets tab.

Notice how the space between items decreases with increasing depth. We'll explain nesting and all the tricks you can do with different depths in section 3.4. Be sure to read it!

3.3.6.3 Enumerate

The Enumerate environment is the tool to use to create numbered lists and outlines. It has these properties:

1. Each item has a numeral as its label.
    a) The type of numeral depends on the nesting depth.
2. LyX automatically counts the items for you and updates the label as appropriate.
3. Each new Enumerate environment resets the counter to one.
4. Like the Itemize environment, the Enumerate environment:
    a) Offsets the items relative to the left margin. Items can be any length.
    b) Reduces the space between items as the nesting depth increases.
    c) Uses different types of labels depending on the nesting depth.
    d) Allows up to a four-fold nesting.

Unlike the Itemize environment, Enumerate *does* show the different labels for each item. Here is how LyX labels the four different levels in an Enumerate:

1. The first level of an Enumerate uses Arabic numerals followed by a period.
    a) The second level uses lower case letters surrounded by parentheses.
        i. The third level uses lower-case Roman numerals followed by a period.

A. The fourth level uses capital letters followed by a period.

B. Again, notice the decrease in the spacing between items as the nesting depth increases.

ii. Back to the third level

b) Back to the second level.

2. Back to the outermost level.

Once again, you can customize the type of numbering used in the Enumerate environment. It involves adding commands to the LaTeX preamble (see the *Extended Features* manual), however. As stated earlier, such customization only shows up in the printed version, not on the LyX screen.

There is more to nesting Enumerate environments than we've stated here. You *really* should read section 3.4 to learn more about nesting.

3.3.6.4 Description

Unlike the previous two environments, the Description list has no fixed label. Instead, LyX uses the first "word" of the first line as the label. Here's an example:

Example: This is an example of the Description environment.

LyX typesets the label in boldface and puts extra space between it and the rest of the line.

Now, you're probably wondering what we mean by, "uses the first 'word'." The Space key does not add a whitespace character, but separates words from one another. Inside of a Description environment, the Space key tells LyX to end the label if we're at the beginning of the first line of an item.

However, what if you want or need to use more than one word in the label of a Description environment? Simple: use a Protected Blank. [Use either C-Space or Special Formatting ▷ Protected Space from the Insert menu. See sec. 6.6.1 for more info.] Here's an example:

Second Example: This one shows how to use a Protected Blank in the label of a Description list item.

Usage: You should use the Description environment for things like definitions and theorems. Use it when you need to make one word in particular stand out in the text that describes it. It's not a good idea to use a Description environment when you have an entire sentence that you want to describe. You're better off using Itemize or Enumerate and nesting several Standard paragraphs into them.

Nesting: You can, of course, nest Description environments inside one another, nest them in other types of lists, and so on.

Notice that after the first line, LyX indents subsequent lines, offsetting them from the first line.

3.3.6.5 The LyX List

The List environment is a LyX extension to LaTeX.

Now, if you jumped here without reading sections 3.3.6.2-3.3.6.4, you've goofed. The List environment does *not* create numbered lists. That's what Enumerate does, and it's documented in section 3.3.6.3.

Like the Description environment the List environment has user-defined labels for each list item. There are some key differences between this list environment and the other three:

item labels LyX uses the first "word" of each line as the item label. The first Space after the beginning of the first line of an item marks the end of the label. If you need to use more than one word in an item label, use a protected blank as described above.

margins As you can see, LyX uses different margins for the item label and the body of the item text. The body of the text has a larger left margin, which is equal to the default label width plus a little extra space.

label width LyX uses one of two things for the label width: the actual width of the label, or the default width, whichever is larger. If the actual width is larger, then the label "extends" into the first line. In other words, the text of the first line isn't aligned with the left margin of the rest of the item text.

default width You can very easily set this default width. It's quite painless, actually. So, you can easily ensure that the text of all items in a List environment have the same left margin.

uses You should use the List environment the same way you'd use as Description list: when you need one word to stand out from the text that describes it. The List environment gives you another way to do this, using a different overall layout.

nesting You can nest List environments inside one another, nest them in other types of lists, and so on. They work just like the other list paragraph environments. Read section 3.4 to learn about nesting.

As you can see, this is a feature-packed paragraph environment!

To change the default width of the label, select the items in the list to change. You can also simply move the cursor into a List item if you want to change only its label width. Now open the Edit ▷ Paragraph Settings dialog and find the Label width text box. The text in the Label width box determines the default label width. If you really, really want to, you can use the text of your largest label here, but you don't need to. We recommend using the letter "M" multiple times. It's the widest character and is a standard unit of width in LaTeX. The default label width in the example List is 6

"M"s wide. Using "M" as your unit of width in the Label Width box has one more advantage: you don't need to keep changing the contents of Label Width every time you alter a label in a List environment.

There's yet another feature of the List environment we need to tell you about. As you can see in the examples, LyX left-justifies the item labels by default. You can use additional HFills to change how LyX justifies the item label. We'll document HFills later in section 6.6.1. Here are some examples:

Left The default for List item labels.

Right One HFill at the beginning of the label right justifies it.

Center One HFill at the beginning of the label and one at the end centers it.

Don't worry if you have no idea what HFills are yet. Just remember that you can use them to customize the look of the List environment.

That does it for the four paragraph environments for making lists. Oh - did we mention that you should read about nesting environments in section 3.4 if you want to use any of these list environments?

3.3.7 Letters

3.3.7.1 Address and Right Address: An Overview

Although LyX has document classes for letters, we've also created two paragraph environments called Address and Right Address. To use the letter class, you need to use specific paragraph environments in a specific order, otherwise LaTeX gags on the document. In contrast, you can use the Address and Right Address paragraph environments anywhere with no problem. You can even nest them inside other environments, though you can't nest anything in them.

Of course, you're not limited to using Address and Right Address for letters only. Right Address, in particular, is useful for creating article titles like those used in some European academic papers.

3.3.7.2 Usage

The Address environment formats text in the style of an address, which is also used for the opening and signature in some countries. Similarly, the Right Address environment formats text in the style of a right-justified address, which is used for the sender's address and today's date in some countries. Here's an example of each:

Right Address
WhoAmI
WhereAmI
When is it? What is today?

That was Right Address. Notice that the lines all have the same left margin, which LyX sets to fit the largest block of text on a single line. Here's an example of the Address environment:
WhoAreYou
Where do I send this
Your post office and country

As you can see, both Address and Right Address add extra space between themselves and the next paragraph. Speaking of which, if you hit Return in either of these environments, LyX resets the nesting depth and sets the environment to Standard. This makes sense, however, since Return is the `break-paragraph` function, and the individual lines of an address are not paragraphs. Thus, you'd use `break-line` [C-Return or Special Formatting ▷ Linebreak from the Insert menu] to start a new line in an Address or Right Address environment.

3.3.8 Academic Writing

Most academic writing begins with an abstract and ends with a bibliography or list of references. LyX contains paragraph environments for both of these.

3.3.8.1 Abstract

The Abstract environment is used for the abstract of an article. Technically, you *could* use this environment anywhere, but you really *should* only use it at the beginning of the document, after the title. The Abstract environment is only useful in the "article" and "report" document classes [as well as "amsart," which is just a specialized version of "article"]. The "book" document class ignores the Abstract completely, and it's utterly silly to use Abstract in the "letter" document class.

The Abstract environment does several things for you. First, it puts the centered label "Abstract" above the text. The label and the text of the abstract are separated by some extra vertical space. Second, it typesets everything in a smaller font, just as you'd expect. Lastly, it adds a bit of extra vertical space between the abstract and the subsequent text. Well, that's how it will appear on the LyX screen. If your document is in the "report" class, the abstract actually appears on a separate page in the printed version of the file.

Starting a new paragraph by hitting Return does *not* reset the paragraph environment. The new paragraph will still be in the Abstract environment. So, you will have to change the paragraph environment yourself when you finish entering the abstract of your document.

We'd love to give you an example of the Abstract environment, but we can't, since this document is in the "book" class. If you've never heard of an "abstract" before, you can safely ignore this environment.

3.3.8.2 Bibliography

The Bibliography environment is used to list references. Technically, you *could* use this environment anywhere, but you really *should* only use it at the end of the document. Also, don't bother trying to nest Bibliography in anything else or vice versa. It won't work.

When you first open a Bibliography environment, LyX add a large vertical space, followed by the heading "Bibliography" or "References," depending on the document class. The heading is in a large boldface font. Each paragraph of the Bibliography environment is a bibliography entry. Thus, hitting Return does *not* reset the paragraph environment. Each new paragraph is still in the Bibliography environment.

At the *beginning* of the *first line* of each paragraph, you will see a gray button showing a number. If you click on it, you will get a dialog in which you can set a key and a label. The key is the symbolic name by which you will refer to this bibliography entry. For example, suppose your first entry in the bibliography was a book about LaTeX. We could choose the key "latexguide" for that entry. You can also give a label, which will be displayed in the gray inset box.

The key field isn't useless. You can refer to your bibliography entries using the Insert ▷ Citation command. Just choose the key inside in the available keys list, then add a reference by clicking on the left arrow, which will add it to the selected keys list. Multiple references can be placed by selecting more than one key. An example of the Bibliography appears at the end of this document. "See [4] or [3, Chapter 3]" is an example of how to cite two of the entries in it. In the second one, we used the Text after field of the citation dialog to add the text "Chapter 3". The texts "latexguide" and "latexcompanion" that you see on screen will be replaced in print by the number or the label of the bibliography entry.

The more advanced LaTeX bibliography package BibTeX is also supported by LyX. For a description of how to use it, please refer to the *Extended LyX Features* document.

3.3.9 Special Purpose

There are three standard paragraph environments that simply don't fit any category, as they are very specialized for a particular purpose. We'll point out the highlights and uses of each.

3.3.9.1 Caption

The Caption environment is the default paragraph environment for Figure Floats and Table Floats. On the LyX screen, you'll see either the label "Figure #:" or "Table #:", depending on which type of Float it's in. The actual reference number is substituted in this label in the printed output.

You can't really nest things into a Caption environment. Additionally, hitting Return resets the paragraph environment to Standard, so a Caption can only be a single paragraph.

You cannot use a Caption environment outside of a Figure Float or a Table Float. See sections 4.3 and 4.4 for more information on Figure Floats and Table Floats.

3.3.9.2 LyX-Code

The LyX-Code environment is another LyX extension. It type-sets text in a typewriter-style font. It also treats the Space key as a fixed whitespace;[6] this is the only case in which you can type multiple whitespaces in LyX. If you need to insert blank lines, you'll still need to use C-Return [the `break-line` function]. Return breaks paragraphs. Note, however, that Return does *not* reset the paragraph environment. So, when you finish using the LyX-Code environment, you'll need to change the paragraph environment yourself. Also, you *can* nest the LyX-Code environment inside of others.

There are a few quirks with this environment:

- You cannot use C-Return at the beginning of a new paragraph [i.e. you can't follow Return with a C-Return].
- You can't follow a C-Return with a Space.
  - Use a Return to begin a new paragraph, then you can use a Space.
  - Or: use C-Space instead.
- You can't have an empty paragraph or an empty line. You must put at least one Space in any line you want blank. Otherwise, LaTeX generates errors.
- You cannot get the typewriter double quotes by typing " since that will insert *real* quotes. You get the typewriter double quotes with C-" (or C-q if you use Emacs-like key bindings).

Here's an example:

```
#include <stdio.h>

int main(void)
{
    printf("Hello World\n");
    return 0;
}
```

This is just the standard “Hello world!” program.

LyX-Code has one purpose: to typeset code, such as program source, shell scripts, rc-files, and so on. Use it only in those very, very special cases where you need to generate text as if you used a typewriter.

[6] In the LyX-Code environment, the Space key is treated as a Protected Blank instead of an end-of-word marker.

3.4 Nesting Environments

3.4.1 The Big Deal

Throughout the previous sections, we've been nagging you to "go read Section 3.4." So, you're probably wondering what the big deal is.

The big deal is that LyX differs rather strongly from the traditional "wordprocessor-as-overglorified-typewriter" concept. With a typewriter, text is merely ink on a page. Most word processors aren't much better, treating text as pixels on the screen and bytes in memory. In contrast, LyX treats text as a unified block with a particular context and specific properties. However, what if you wanted one "block" to inherit some of the properties of another "block" ?

Here's a more specific example: outlines. You have three main points in your outline, but point #2 also has two subpoints. In other words, you have a list *inside* of another list, with the inner list "attached" to item #2:

1. one
2. two
   a) sublist - item #1
   b) sublist - item #2
3. three

How do you put a list inside of a list? By now, the answer should be obvious: you nest one list inside the other.

How to nest an environment is quite simple. Select Increase Environment Depth or Decrease Environment Depth from the Edit menu to change the nesting depth of the current paragraph (the status bar will tell you how far you are nested).

You can also use the convenient key bindings S-M-Left and S-M-Right[7] to change the nesting level. The change will work on the current selection if you have made one (allowing you to change the nesting of several paragraphs at once), or the current paragraph.

Note that LyX only changes the nesting depth if it can. If it's invalid to do so, nothing happens if you try to change the depth. Additionally, if you change the depth of one paragraph, it affects the depth of every paragraph nested inside of it. It's hard to describe what exactly LyX does in this case. That depends specifically on what your text looks like. Your best bet is to simply play with changing the nesting depth and see what happens.

Nesting isn't just limited to lists. In LyX, you can nest just about anything inside anything else, as you're about to find out. This is the real power of nesting paragraph environments.

[7] M-p Left and M-p Right are alternatives, if you prefer those bindings

3.4.2 What You Can and Can't Nest

Before we fire a list of paragraph environments at you, we need to tell you a little bit more about how nesting works.

The question of nesting is a bit more complicated than a simple yes or no, can you or can't you. There's also the question of how. Can you nest this environment into anything else? Can you nest another environment into it? A "yes" to one of these doesn't guarantee a "yes" to the other.

The paragraph environments in LyX can do one of three things when it comes to nesting. First, an environment may be completely unnestable. Second, there are environments that are fully nestable. You can nest them inside of things and you can also nest other things inside of them. There is one last type of environment. You can nest them into other environments, but that's it. You can't nest anything into them.

Here's a list of the three types of nesting behavior, and which paragraph environments have them:[8]

Unnestable Can't nest them. Can't nest into them.

- Bibliography
- Title
- Author
- Date

Fully Nestable You can nest them. You can nest other things into them.

- Abstract
- Verse
- Quote
- Quotation
- Itemize
- Enumerate
- Description
- List
- LyX-Code

Nestable-Inside You can nest them inside of other things. You can't nest anything into them.

[8] For some odd reason, LyX allows you to fully nest both Bibliography and Abstract. Also, LyX allows you to nest Title, Author, and Date into other environments. We urge you not to. LaTeX may barf if you try it. Then again, it may not. We don't know for certain. However, it makes no sense contextually to perform any nesting with these environments, so why would you ever want to?

- Part
- Chapter
- Section
- Subsection
- Subsubsection
- Paragraph
- Subparagraph
- Part*
- Chapter*
- Section*
- Subsection*
- Subsubsection*
- Standard
- Right Address
- Address
- Caption

3.4.3 Nesting Other Things: Tables, Math, Floats, etc.

There are several things that aren't paragraph environments, but which are affected by nesting anyhow. They are:

- equations
- tables
- figures

[Note: if you put a figure or a table in a Float, this is no longer true. See below or look in sections 4.3 or 4.4 for more info.]

LyX can treat these three objects as either a word or as a paragraph. Well, you can't inline a table, but you can inline math and figures. If a figure or an equation is inlined, it goes wherever the paragraph it's in goes.

On the other hand, if you have an equation, figure or table in a "paragraph" of its own, it behaves just like a "nestable-inside" paragraph environment. You can nest it into any environment, but you [obviously] can't nest anything into it.

Here's an example with a table:

1. Item One

   a) This is (a) and it's nested.

a	b
c	d

   b) This is (b). The table is actually nested inside (a).

2. Back out again.

If we hadn't nested the table at all, the list would look like this:

1. Item One

   a) This is (a) and it's nested.

a	b
c	d

1. This is (b). The table is *not* nested inside (a). In fact, it's not nested at all.

2. Back out again.

Notice how item (b) is not only no longer nested, but is also the first item of a new list!

There's another trap you can fall into: nesting the table, but not going deep enough. LyX turns anything after the table into a new [sub]list.

1. Item One

   a) This is (a) and it's nested.

a	b
c	d

   a) This is (b). The table is actually nested inside Item One, but *not* inside (a).

2. Back out again.

As you can see, item (b) turned into the first item of a new list, but a new list *inside* item 1. The same thing would have happened to a figure or an equation. So, if you nest tables, figures or equations, make sure you go to the right depth!

Then there are the so-called Floats. A Float is a block of text associated with some sort of label, but which doesn't have a fixed location. It can "float" forward or backward a page or two, to wherever it fits best. Footnotes and Margin Notes are floats, as are Table Floats and Figure Floats. When you're editing a document in LyX, a closed Float looks like a gray button with a red label and goes wherever the paragraph it's in goes. However, because a Float has no fixed location in the final text, nesting has no effect on its actual location after you feed your document to LaTeX.

3.4.4 Usage and General Features

Speaking of levels, LyX can perform up to a six-fold nesting. In other words, "level #6" is the innermost possible depth. Here's an example to display what we mean:

1. level #1 - outermost

    a) level #2

        i. level #3

            A. level #4

                - level #5

                    – level #6

Once again, LyX has a maximum of 6 levels, regardless of which specific paragraph environments you're using at a given level.[9] That means that you can perform a six-fold nesting of a Description list, or a Verse environment, and so on. You can also mix environments, as we shall see later.

There are two exceptions to the six-fold nesting limit, and you can see both of them in the example. Unlike the other fully-nestable environments, you can only perform a four-fold nesting with the Enumerate and Itemize environments. For example, if we tried to nest another Enumerate list inside of item "A.", we'd get errors.[10]

3.4.5 Some Examples

The best way to explain just what you can do with nesting is by illustration. We have several examples of nested environments. In them, we explain how we created the example, so that you can reproduce them.

[9] Unfortunately, LyX doesn't enforce this limitation. If you try to exceed it, however, LaTeX will return errors when you go to produce output for your document.

[10] Once again, LyX doesn't enforce this limitation. If you try to exceed it, however, LaTeX will return errors when you go to produce output for your document.

3.4.5.1 Example #1: The Six-fold Way and Mixed Nesting

- #1-a This is the outermost level. It's a List environment.
  - #2-a This is level #2. We created it by using M-Return followed by M-p Right.
    - #3-a This is level #3. This time, we just hit Return, then used M-p Right twice in a row. We could have also created it the same way as we did the previous level, by hitting M-Return followed by M-p Right.

      This is actually a Standard environment, nested inside of "#3-a". So, it's at level #4. We did this by hitting M-Return, then M-p Right, then changing the paragraph environment to Standard. Do this to create list items with more than one paragraph - it also works for the Description, Enumerate, and Itemize environments!

      Here's another Standard paragraph, also at level #4, made with just a M-Return.
      - #4-a This is level #4. We hit M-Return and changed the paragraph environment back to List. Remember - we can't nest anything inside of a Standard environment, which is why we're still at level #4. However, we *can* keep nesting things inside of "#3-a".
        - #5-a This is level #5…
          - #6-a …and this is level #6. By now, you should know how we made these two.
        - #5-b Back to level #5. Just hit M-Return followed by a M-p Left.
      - #4-b After another M-Return followed by a M-p Left, we're back at level #4.
    - #3-b Back to level #3. By now it should be obvious how we did this.
  - #2-b Back to level #2.
- #1-b And last, back to the outermost level, #1. After this sentence, we'll hit Return and change the paragraph environment back to Standard to end the list.

There you have it! Oh — we could have also used the Description, Quote, Quotation, or even the Verse environment in place of the List environment. The example would have worked exactly the same.

3.4.5.2 Example #2: Inheritance

> This is the LyX-Code environment, at level #1, the outermost level. Now we'll hit Return, then M-p Right, after which, we'll change to the Enumerate environment.
>
> 1. This is the Enumerate environment, at level #2.
> 2. Notice how the nested Enumerate not only inherits its margins from its parent environment [LyX-Code], but also inherits its font and spacing!

We ended this example by hitting Return. After that, we needed to reset the paragraph environment to Standard and resetting the nesting depth by using M-p Left once.

3.4.5.3 Example #3: Labels, Levels, and the Enumerate and Itemize Environments

1. This is level #1, in an Enumerate paragraph environment. We're actually going to nest a bunch of these.

    a) This is level #2. We used M-Return followed by M-p Right. Now, what happens if we nest an Itemize environment inside of this one? It will be at level #3, but what will its label be? An asterisk?

    - No! It's a bullet. This is the *first* Itemize environment, even though it's at level #3. So, its label is a bullet. [Note: we got here by using M-Return, then M-p Right, then changing the environment to Itemize.]

        – Here's level #4, produced using M-Return, then M-p Right. We'll do that again…

        i. …to get to level #5. This time, however, we also changed the paragraph environment back to Enumerate. Notice the type of numbering! It's *lowercase Roman*, because we're the *thirdfold* Enumerate environment [i.e. we're an Enumerate inside an Enumerate inside an Enumerate].

        ii. What happens if we *don't* change the paragraph environment, but decrease the nesting depth? What type of numbering does LyX use?

        iii. Oh, as if you couldn't guess by now, we're just using M-Return to keep the current environment and depth but create a new item.

        iv. Let's use M-p Left to decrease the depth after the next M-Return.

        i. This is level #4. Look what type of label LyX is using!

        i. This is level #3. Even though we've changed levels, LyX is still using a lowercase Roman numeral as the label. Why?!

        ii. Because, even though the nesting depth has changed, the paragraph is *still* a thirdfold Enumerate environment. Notice, however, that LyX *did* reset the counter for the label.

    b) Another M-Return M-p Left sequence, and we're back to level #2. This time, we not only changed the nesting depth, but we also moved back into the twofold-nested Enumerate environment.

2. The same thing happens if we do another M-Return M-p Left sequence and return to level #1, the outermost level.

Lastly, we reset the environment to Standard. As you can see, the level number doesn't correspond to what type of labelling LyX uses for the Enumerate and Itemize environments. The number of *other Enumerate environments* surrounding it determines what kind of label LyX uses for an Enumerate item. The same rule applies for the Itemize environment, as well.

3.4.5.4 Example #4: Going Bonkers

1. We're going to go totally nuts now. We won't nest as deep as in the other examples, nor will we go into the same detail with how we did it. [level #1: Enumerate]

   [Return, M-p Right, Standard: level #2] We'll stick an encapsulated description of how we created the example in brackets someplace. For example, the two keybindings are how we changed the depth. The environment name is, obviously, the name of the current environment. Either before or after this, we'll put in the level.

2. [Return, Enumerate: level #1] This is the next item in the list.

   Now we'll add verse.
   It will get much worse.
   [Return, M-p Right, Verse: level #2]

   Fiddle dee, Fiddle doo.
   Bippitey boppitey boo!
   [M-Return]

   Here comes a table for you:

one-fish	two-fish
red-fish	blue-fish

   [M-Return, Table, M-p Right 3 times, M-Return, Verse, M-p Left]

3. [Return, Enumerate: level #1] This is another item. Note that selecting a Table resets the nesting depth to level #1, so we increased the nesting depth 3 times to put the table inside the Verse environment.

> We're now ending the Enumerate list and changing to Quotation. We're still at level #1. We want to show you some of the things you can do by mixing environments. The next set of paragraphs is a "quoted letter." We'll nest both the Address and Right Address environments inside of this one, then use another nested Quotation for the letter body. We'll use M-Return to preserve the depth. Remember that you need to use C-Return to create multiple lines inside the Address and Right Address environments. Here it goes:
>
> 1234 Nowhere Rd.
> Moosegroin, MT 00100
> 9-6-96
>
> Dear Mr. Fizlewitz:
>
> > We regret to inform you that we cannot fill your order for 50L of compressed methane gas due to circumstances beyond our control. Unfortunately, several of our cows have mysteriously exploded, creating a backlog in our orders for methane. We will place your name on the waiting list and try to fill your order as soon as possible. In the meantime, we thank you for your patience.
> >
> > We do, however, now have a special on beef. If you are interested, please return the enclosed pricing and order form with your order, along with payment.
> >
> > We thank you again for your patience.
>
> Sincerely,
> Bill Hick
>
> That ends that example!

As you can see, nesting environments in LyX gives you a lot of power with just a few keystrokes. We could have easily nested an Itemize list inside of a Quotation or Quote, or put a Quote inside of an Itemize list. You have a huge variety of options at your disposal.

3.5 Fonts and Text Styles

3.5.1 Overview

Many modern typesetting and markup languages have begun to move towards specifying character styles rather than specifying a particular font. For example, instead of changing to an italicized version of the current font to emphasize text, you use an “emphasized style” instead. This concept fits in perfectly with LyX. In LyX, you do things based on contexts, rather than focusing on typesetting details.

Right now, LyX allows you to specify a global default font, and has two character styles, Emphasized and Noun. The Emphasized style corresponds to an italics font. The Noun style corresponds to a font in smallcaps, which some languages and writing styles use to typeset proper names. The LyX Team has at last (as of LyX version 1.4) introduced true character styles, but currently these must be defined explicitly in the document layout file. There’s currently no GUI support to define new, or tweak existing, character styles to allow you, the user, to customize which font changes correspond to what styles.

3.5.2 Global Options

You can set the default font from the Document ▷ Settings dialog. There are two options of interest here, Fonts and Font Size. The possible options under Fonts include “default” and a list of fonts available on your system. The option “default” uses the standard TeX fonts, known as “computer modern” (cm) or “European modern” (ec). Most systems will typically have some version of a Times and Helvetica font, with other variants. You’ll have to examine this for yourself.

As for the Font Size option, there are three possible values: 10, 11, and 12. Remember, this is the *base* font size. LyX actually scales all of the other possible font sizes (such as those used in footnotes, superscripts, and subscripts) by this value. You can always fine-tune the font size from within the document if you need to. It’s also rather silly to use an 8pt or 24pt font as the default font size, as this typically renders your document unreadable.

Note that once you choose a new value for Fonts or Font Size, LyX does *not* change the screen. You’ll only see a difference once you generate the final output. This is part of the WYSIWYM concept. Besides, you have certainly noticed that "Roman" text on the LyX screen corresponds to the default font.

3.5.3 Using Different Character Styles

As we’ve already seen, LyX automatically changes the character style for certain paragraph environments. We also mentioned two other character styles, Emphasized and Noun. You can activate both of these styles via keybindings, the menus, and the toolbar.

To activate the Noun style, do one of the following:

- click on the toolbar button with the person-shaped icon
- use the keybinding M-c c

These commands are all toggles. That is, if Noun style is already active, they deactivate it.

One typically uses the Noun style for proper names. For example: "MATTHIAS ETTRICH is the original author of LyX."

A more widely used character style is the Emphasized style. You can activate [or deactivate - it's also a toggle] the Emphasized style by:

- clicking on the toolbar button with the "!" character on it
- using the keybindings M-c e

At the moment, the Emphasized style is equivalent to an italicized font. We have plans to make that association more user-configurable in the future.

We've been using the Emphasized style all over the place in this document. Here's one more example:

> *Don't overuse character styles!*

It's also a warning in addition to an example. One's writing should parallel ordinary conversation. Since we don't all constantly scream at each other, we should also avoid the common tendency to overuse character style.

Oh — one last note: You can always reset to the default font using the keybinding M-c Space.

3.5.4 Fine-Tuning with the Character Layout dialog

There are always occasions when you'll need to do some fine-tuning, so LyX gives you a way to create custom character style. For example, an academic journal or a corporation may have a style sheet requiring a sans-serif font be used in certain situations.[11] Also, writers sometimes use a different font to offset a character's thoughts from ordinary dialogue.

Before we document how to use custom character style, we want to issue a warning yet again: Don't overuse character styles. Many modern word processors have a vast array of fonts available to them, providing you with the power of a printing press. Unfortunately, there is a tendency to overuse that power. The phrase, "Using a sledgehammer to swat a fly," comes to mind. And, as the old saying implies, documents that overuse different fonts and sizes tend to look like someone's knocked huge holes in them.

Enough complaining.

[11] Note from JOHN WEISS: There is, in fact, such a style sheet for the LyX Documentation, since manuals need a certain amount of consistency.

To use custom fonts, open the Edit ▷ Text Style dialog. There are seven buttons on this dialog, each corresponding to a different font property which you can choose. You can choose an option for one of these seven properties, or select No change, which keeps the current state of that property. The item Reset will reset the property to whatever is the default for the hosting paragraph environment. You can use this to reset attributes across a bunch of different paragraph environments in a snap.

The seven font properties, and their options [in addition to No change and Reset] are:

Family The "overall look" of the font. The possible options are:

- Roman This is the Roman font family.
  It's also the default family. [keybinding = M-c r]
- Sans Serif This is the Sans Serif font family.
  [keybinding = M-c s]
- Typewriter `This is the Typewriter font family.`
  [keybinding = M-c p]

Series This corresponds to the print weight. Options are:

- Medium This is the Medium font series.
  It's also the default series.
- Bold **This is the Bold font series.**
  You can toggle this series on or off with the keybinding M-c b.

Shape As the name implies. Options are:

- Upright This is the Upright font shape.
  It's also the default shape.
- Italic *This is the Italic font shape.*
- Slanted *This is the Slanted font shape* (although it might not be visible on screen, this is different from italic).
- Small Caps This is the Small caps font shape.

Size Alters the size of the font. You'll find no numerical values here; all possible sizes are actually proportional to the default font size. Once again, you don't feed LyX the details, but a general description of what you want to do.

The options [and their keybindings] are:

Tiny This is the “Tiny” font size.
[keybinding = M-s t or M-s 1]

Smallest This is the “Smallest” font size
[keybinding = M-s 2]

Smaller This is the “Smaller” font size
[keybinding = M-s S or M-s 3]

Small This is the “Small” font size.
[keybinding = M-s s or M-s 4]

Normal This is the “Normal” font size.
It’s also the default size. [keybinding = M-s n or M-s 5]

Large This is the “Large” font size.
[keybinding = M-s l or M-s 6]

Larger This is the “Larger” font size.
[keybinding = M-s S-L or M-s 7]

Largest This is the “Largest” font size.
[keybinding = M-s 8]

Huge This is the “Huge” font size.
[keybinding = M-s h or M-s 9]

Huger This is the “Huger” font size.
[keybinding = M-s H or M-s 0]

We’ll warn you *yet again*: don’t go crazy with this feature. You should almost never need to change the font size. LyX automatically changes the font size for different paragraph environments - use that instead. This is here for fine-tuning *only!*

Misc Here you can change a few other things at the character level. Options are:

Emph *This is text with emphasize on.*
This might seem like the same as *Italic*, but it is actually a bit different. If you use emphasize on italicized text, it will make it upright. In future versions of LyX, we hope to let you customize the exact behavior of this *logical* property.

Underbar This is text with Underbar on.
[keybinding = M-c u]

Noun THIS IS TEXT WITH NOUN ON.

Like Emph, this is a logical attribute. For the moment, it is equivalent to Small Caps, but that is bound to change some day.

Avoid using underbar if you can! It's a holdover from the typewriter days, when you couldn't change fonts. We no longer need to resort to emphasizing text by overstriking it with an underscore character. It's only included in LyX because it's also in LaTeX, and because some people *may* need it in order to follow style sheets for journal submissions (and in fact we use it in these manuals to indicate keyboard shortcuts for menu items).

Color You can adjust the color of the text with this control. Of course, you need to have a color printer to exploit this, but you also need to have the color LaTeX package installed. Notice that `xdvi` is not able to display these colors. Besides No color, which is the standard "color", you can choose between Black, White, Red, Green, Blue, Cyan, Magenta and Yellow text.

Language This is used to mark regions of text as having a different language from the language of the document. Text marked in this way will be underlined in blue to indicate the change.

You have a huge number of combinations to choose from.

Once you've chosen a new character style via the Edit ▷ Text Style dialog, you can activate it using the toolbar button labelled "Font", or select Apply. The toolbar button lets you toggle the state of your custom character style even when the dialog isn't visible.

As we stated earlier, to completely reset the character style to the default, use M-c Space. If you want to toggle only those properties that you have just changed (suppose you just sent the shape to "slanted" and the series to "bold"), set the Toggle on all these switch and press Apply.

We conclude with the same warning we've been spewing: Don't overuse the fonts. They are, more often than not, a kludge and a horrible substitute for good writing. Your writing should speak for itself — and will.

3.6 Printing and Previewing

3.6.1 Overview

Now that we've covered some of the basic features of document preparation using LyX, you probably want to know how to print out your masterpiece. Before we tell you that, however, we want to give you a quickie explanation of what goes on behind-the-scenes. We cover this information in much greater detail in the *Extended Features* manual as well.

LyX uses a program called "LaTeX" as its backend. (Actually, LaTeX is just a macro package for the TeX typesetting system, but to prevent confusion, we'll just refer to the whole magilla as "LaTeX.") Think of it this way: LyX is what you use to do your actual writing. Then, LyX calls LaTeX to turn your writing into printable output. This happens in a couple of stages:

1. First, LyX converts your document to a series of text commands for LaTeX, generating a file with the extension, "`.tex`".

2. Next, LaTeX uses the commands in the `.tex` file to produce printable output. It doesn't know anything about your printer, however. Instead, LaTeX produces what's known as a *device-independent* file, or DVI for short. The actual output is in a file with the extension, "`.dvi`". DVI files are completely portable; you can move them from one machine to another without needing to do any sort of conversion.

   NOTE: The DVI file only contains what was in the LaTeX file itself. If you have included PostScript pictures in your document, there will only be a link to these files. So don't forget these files if you move your `.dvi` file to another computer.

3. Consider the `.dvi` file to be the "final output." Once you have it, you can view it, print it, or convert it to other formats.

   a) You can view `.dvi` files using a program called `xdvi`.

   b) Some printers and Unix systems understand DVI, and can print your `.dvi` file directly.

   c) Nowadays, most printers understand the PostScript format. LyX automatically converts the `.dvi` file to a PostScript file for you when you go to print out your document. LyX will also let you preview a PostScript version of your document using the program `ghostview`.

      One advantage of using PostScript® is that the converter program [called `dvips`] takes any PostScript graphics you may have included in your document and puts it into the resulting PostScript version of your document. It also includes any special fonts you may have used. That makes the PostScript version much, much more portable than the DVI version.

LyX does all of these steps automagically for you.

As you have seen, a lot of things happen before you get a hardcopy or a preview of your document. So, don't worry if printing requires a bit more time than with other word processors. The printed result is worth the wait. Quality always has its price.

3.6.2 Quick Viewing with xdvi

To get a look at the final version of your document, with all of the pagebreaks in place, the footnotes correctly numbered, and so on, select View ▷ DVI. Then wait a while.

When all of the behind-the-scenes action is done, LyX calls the program `xdvi`. You can now look at the results. [If you want more info on the `xdvi` program, see the `man`-pages.]

Helpful-Tip: Keep the `xdvi` window open, maybe moving it to another desktop. Then, after you make changes to your document, just use View ▷ Update ▷ DVI. Now click on the `xdvi` window. The `xdvi` program will automatically reread the `.dvi` file and give you an updated view.

3.6.3 Viewing the PostScript Version with ghostview

In general, using `xdvi` to view your document is the easiest and fastest way. There may be times, however, when you want to look at the PostScript version. One reason is fonts.[12] You can use PostScript fonts in a LaTeX document, but `xdvi` won't show this. You'll need to use `ghostview` or some other PostScript file viewer to see the actual results.

To view the PostScript version of your document, select PostScript from the View menu. When all of the magic behind-the-scenes is done, LyX calls the program `ghostview`. You can now look at the results.

You've guessed what the Update ▷ Postscript command from the View menu does, haven't you? Remember to click once in the `ghostview` window after this command to update the view.

3.6.4 Printing the File

To print a file, select Print from the File menu, or click on the toolbar button with the printer on it. This opens the Print dialog.

You can choose to only print even-numbered or odd-numbered pages - this is useful for printing on two sides: you can re-insert the pages after printing one set of pages, to print on the other side. Some printers spit out pages face-up, others, face-down. By choosing a particular order to print in, you can take the entire stack of pages out of the printer without needing to reorder them.

You can set the parameters in the Destination box as follows :

Printer This is the name of the printer to print to.[13] The printer should understand PostScript files.

[12] Note from John Weiss: Another reason is paranoia. I always like to look at the PostScript file before I print it, just so I see exactly what went to the printer…

[13] Note that this printer name isn't for the `lpr` command but for `dvips`. That means `dvips` has to be configured for this printer name. See the section 2.5.2 or the `dvips` documentation for details. The default printer can also be set in `lyxrc`.

File The name of a file to print to. The output will be in PostScript format. The file will generally be written in the current directory, unless you specify the full path.

Note that printing may need little time, since LaTeX, `dvips` and, if you don't have a PostScript printer, `ghostscript` have to process your document.

3.7 A Few Words about Typography

3.7.1 Hyphens and Hyphenation

In LyX, the “`-`” character comes in three lengths, often called the *hyphen*, the *en dash*, and the *em dash*:

1. hyphen	-	made with “`-`“
2. en dash	–	made with “`--`”
3. em dash	—	made with “`---`”
4. minus sign	$-$	a “`-`” in math mode

You generate these by using the “`-`” character multiple times in a row. LyX automatically converts them to the appropriate length dash in the final output.

The three types of dash are distinct from the minus sign, which appears in math mode and has a length of its own. Here are some examples of the “`-`” in use:

1. line- and page-breaks	(*hyphen*)
2. From A–Z	(*en dash*)
3. Oh — there's a dash.	(*em dash*)
4. $x^2 - y^2 = z^2$	(*minus sign*)

Those of you reading this from within LyX will see no difference, though there is one in the printed version.

One more note about hyphenation — LyX automatically breaks up words and inserts hyphens in English text. The words won't be hyphenated until you generate the final output.

Actually, it's LaTeX that does this, and it will also hyphenate words in *some* other languages. To know whether (PDF)LaTeX hyphenates for *your* language, look at any log file produced by a LaTeX run: it will say

```
Babel <v3.7h> and hyphenation patterns for american, french, german,
ngerman, nohyphenation, loaded.
```

This tells you that, e.g., if you write in Finnish, you're out of luck. Study (for the teTEX distribution of LATEX) the utilities `texconfig` and `fmtutil` in order to switch hyphenation on for your language by "uncommenting" the relevant line in a file typically named `language.dat`. Sorry for the inconvenience.

If, for whatever reason, LATEX *still* can't break a word correctly (e.g., a compound word), you can set hyphenation points manually. This is done with the menu item Hyphenation Point under Special Formatting in the Insert menu. Note that these extra hyphenation points are only recommendations to LATEX. If no hyphenation is necessary, LATEX will totally ignore them.

3.7.2 Punctuation Marks

3.7.2.1 Abbreviations and End of Sentence

When LYX calls LATEX to generate the final version of your document, LATEX automatically distinguishes between words, sentences, and abbreviations. LATEX then adds the "appropriate amount of space": sentences get a little bit more space between the period and the next word. Abbreviations get the same amount of space after the period as a word uses.

Unfortunately, the algorithm for figuring out what's an abbreviation and what's the end of a sentence is really quite brain-dead. If a "`.`" is at the end of a lowercase letter, it's the end of a sentence; if it's at the end of a capitalized letter, it's an abbreviation.

Here are some examples of *correct* abbreviations and the end of a sentence:

- M. Butterfly
- Don't worry. Be happy.

. . . and here's an example of the algorithm going wrong:

- e. g. this is too much space!
- This is I. It's okay.

You won't see anything wrong until you view a final version of your document.

To fix this problem, use one of the following:

1. Use an Inter-word Space after lowercase abbreviations (see section 6.6.5.1).
2. Use a Thin Space between two tokens of an abbreviation (see section 6.6.5.3).
3. Use an End of sentence period found under the Insert ▷ Special Character menu to force the use of inter-sentence spacing. This function is also bound to C-period for easy access.

With the corrections, our earlier examples look like this:

- e. g. this is too much space!
- This is I. It's okay.

Some languages don't use extra spacing between sentences. If your language is such a language, you don't need to worry about all of this. For those that do need to bother, there is help to catch those sneaky errors: check out the Tools ▷ Check TeX feature described in *Extended Editing.*

3.7.2.2 Quotes

LyX usually sets quotes correctly. Specifically, it will use an opening quote at the beginning of quoted text, and use a closing quote at the end. For example, "open close". The keyboard character, ", generates this automatically.

New in version 1.4: To get single quotation marks, you have to press C-". This produces quotation marks like this: ‘’.

You can also select quotes for different languages via the Type option. There are six choices:

“Text” Use quotes like this “double” or ‘single’

”Text” Use quotes like ”this” or ’this’

„Text“ Use quotes like „this“ or ‚this‘

„Text” Use quotes like „this” or ‚this’

«Text» Use quotes like «this» or ‹this›

»Text« Use quotes like »this« or ›this‹

Again, this affects what character the " key produces.

On the other hand, if you want to produce a bona-fide quote character, type M-". This produces: ".

3.7.3 Ligatures

It is standard typesetting practice to group certain letters together and print them as single characters. These groups are known as *ligatures.* Since LaTeX knows about ligatures, your LyX documents will contain them, too. Here are the possible ligatures:

- ff
- fi
- fl
- ffi

- ffl

Once in a while, though, you don't want a ligature in a word. While a ligature may be okay in the word, "graffiti," it looks really weird in compound words, such as "cufflink" or the German "Dorffest." To break a ligature, use Insert ▷ Special Formatting ▷ Ligature Break. This changes "cufflinks" to "cufflinks" and "Dorffest" to "Dorffest".

3.7.4 Widows and Orphans

In the early days of word processors, page breaks went wherever the page happened to end. There was no regard for what was actually going on in the text. You may remember once printing out a document, only to find the heading for a new section printed at the very bottom of the page, the first line of a new paragraph all alone at the bottom of a page, or the last line of a paragraph at the top of a new page. These dangly-bits of text became known as *widows* and *orphans*.

Clearly, LyX can avoid breaking pages after a section heading. That's part of the advantage of paragraph environments. But what about widows and orphans, where the page breaks leave one line of a paragraph all alone at the top or bottom of a page? There are rules built into LaTeX governing page breaks, and some of those rules are there to specifically prevent widows and orphans. This is the advantage LyX has in using LaTeX as its backend.

There's no way we can go into how TeX and LaTeX decide to break a page, or how you can tweak that behavior. Some LaTeX books listed in the bibliography [such as [3] or [4]] may have more information. You will almost never need to worry about this, however.

4 Floats: Tables, Figures, Footnotes and Margin Notes

4.1 Footnotes

Unlike other typesetting programs, LyX uses “foldable” boxes instead of displaying its footnotes at the bottom of the screen or somewhere else in your text. When you insert a footnote with Insert ▷ Footnote, you’ll first see a grey box with a red label “foot” appearing within your text. This box is LyX’s representation of your footnote. You can enter your text into this box. If you click the “foot” label, the box will “fold”. Clicking on the button again will “unfold” the footnote.[1] You will not see any numbers within LyX. You don’t need to worry about those, anyhow, because LyX does the numbering for you, as well as putting the footnote at the bottom of the correct page, when it processes your file. If you want to turn already existing text into a footnote, simply mark it and click on the footnote button (a picture of text with an arrow pointing to stuff in the bottom margin).

What LyX cannot do, yet, is take care of special needs like setting the footnote numbering back to 1 after each section in the “`article`” document class or changing the counter style. You’ll need to insert LaTeX commands like those described in the *Tricks for Footnotes and Marginpars* section of “*Extended Features*.”

NOTE: A float in LaTeX and LyX isn’t a simple paragraph as with usual word processors. It is a complex text structure that may contain everything except floats. That means you can use all the layouts inside a float, even figures and tables. You may not need this too often, but if you do occasionally need it, it’s a neat feature.

4.2 Margin Notes

Margin notes look and behave just like footnotes in LyX. When you insert a margin note via Insert ▷ Margin Note or the toolbar button (which contains a picture of text in a margin with an arrow pointing to it), you’ll see a grey box with a red label “margin” appearing within your text. This box is LyX’s representation of your margin note. You can enter your text into this box. If you click the “margin” label, the box will “fold”. You can access it at a later time by clicking on the label again, thereby “unfolding” the margin note.

This is a margin note.

[1] To close this footnote, click on the red box at the top left.

As a default, LyX uses 1.9 cm (0.75 inches) as the margin width to allow room for margin notes. This might not be what you're looking for, but as with footnotes, LyX cannot yet do everything LaTeX has to offer. You might want to consult your LaTeX handbook for additional commands.

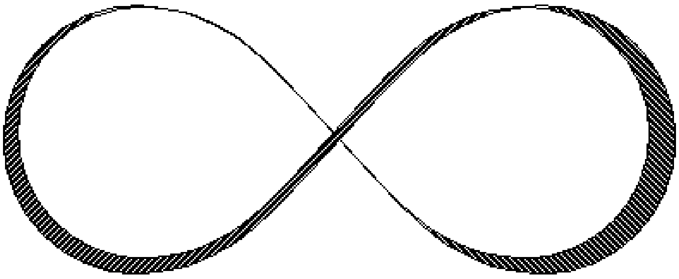
4.3 Figures and Imported Graphics

No document preparation system is complete without the ability to import graphics from other utilities into the document. In LyX, these are referred to as "figures" whether they are actually figures in the traditional sense or simply some kind of imported image.

Note that figures referred to here are do not have captions and sit wherever in the document you place them. If you need one of these features, see sec. 4.3.2.1 below.

To place a figure in your document, click on the second right-most icon on the toolbar, or select Insert ▷ Graphics... from the menu.

A dialog will appear for you to choose the file to load. You can also change any settings you need to in this dialog.



This dialog has numerous parameters, though most should be self-explanatory. The File tab allows you to choose your image file (note that a wide variety of image formats are supported automatically). The figure can be transformed by setting a rotation angle, using a bounding box, and scaling. Table 4.1 describes all available units. It is possible to set a bounding box automatically for some image formats (see the Bounding Box tab). Note that it is possible to control the display of the figure in LyX and the display in the final document separately, which can be very useful for large figures. LaTeX wizards can specify additional LaTeX options in the Extras tab. You can also set the Subfigure option here, for use in figure floats (see Section 4.3.2.1).

4.3.1 How it works

LyX has the ability to handle literally any graphics format in the known universe so long as a conversion path from this graphics format to the target output format can be created. If that sounds a little obtuse, consider how LyX handles Encapsulated PostScript® figures. LaTeX provides native support for this format, so LyX needs do

Table 4.1: Units for setting the image size

unit	name/description
mm	millimetre
cm	centimetre
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 ≈ 37.6 mm)
cc	cicero (1cc = 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)

nothing other than use the `\includegraphics` LaTeX command to insert the figure in the final document.

To view the figure on the LyX screen, however, some additional work is required because neither the XForms nor the Qt GUI libraries can load PostScript® figures themselves. The XForms library can load figures in the following, widely used graphics formats: `bmp`, `gif`, `jpeg`, `pbm`, `pgm`, `ppm`, `tif` and `xbm` whilst the Qt library can also handle `mng`, `png` and `xpm` format figures. Thus, LyX must initiate a conversion from Encapsulated PostScript® to a loadable graphics format.

It does this using the powerful, configurable converters mechanism exposed in the Converters section of the Edit ▷ Preferences dialog. If LyX cannot create a conversion path (which might have many steps) from Encapsulated PostScript® to one of the loadable formats listed above, then it defaults to the use of ImageMagick's `convert` utility. If, after all that, LyX *still* cannot load the figure, then it'll tell you so with a message "Error converting to loadable format" in place of an on-screen view of your figure. If you're presented with such a message, then you'll need to augment the list of known converters.

This strategy is used both to generate on-screen views of your image and when generating the final document. In the latter case, the LaTeX compiler must be supplied with graphics files in PostScript® format. Similarly the PDFLaTeX compiler requires files in `pdf`, `png` or `jpeg` format. LyX will handle the necessary conversions behind the scenes.

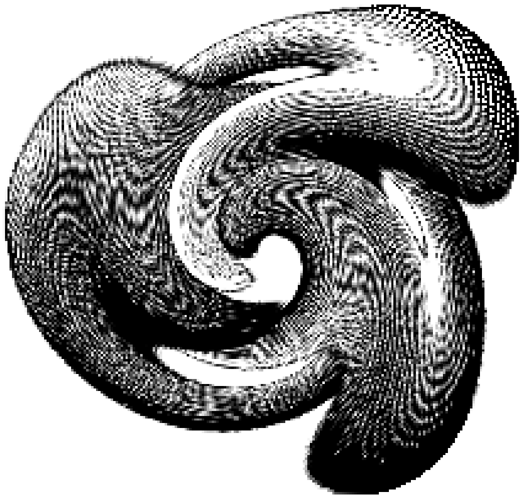
4.3.2 Figure Floats

4.3.2.1 Using Figure Floats

The problem with inserting figures straight into your text is that they might make the pagination of your document extremely awkward. To suit the LyX mentality of automating such processes, you might find it preferable to use Figure Floats, which LyX (actually, LaTeX) is free to move about your document as it deems necessary for a good fit. In return, LyX automates the listing of these figures and allows you to place a caption on them, using the Caption environment explained in Section 3.3.9.1.

To place a Figure Float simply select Insert ▷ Float ▷ Figure from the menu bar. You will get a float without a figure in it; use the toolbar icon described above to insert the actual figure.

Figure 4.1: M.C. Escher on acid.



It seems simple, but there is subtlety involved in the placement of the caption. If you prefer your caption to appear below the figure, then you must press return when the cursor is at the very start of the caption, and insert the figure in the new paragraph created above the caption; or you can delete the caption and recreate it by selecting the Caption environment after the figure has been inserted. This is what we did for figure 4.2. If the cursor is in a paragraph after the caption when you insert the Figure then it will be inserted after the caption, as was the case for 4.1. It is preferred to use one Figure per Float. This allows LyX [actually LaTeX] to best position each figure.

Right-clicking on a float opens a dialog where you can alter the placement options that LaTeX uses for positioning the float (see 4.3.2.2). Span columns is only useful for

Figure 4.2: A severely distorted platypus in a float.

two-column documents: if you select it, the float will span across both columns on the page instead of being confined to just one.

This figure also shows how we place a label and create a cross-reference to it; as you would expect from reading section 6.1 you can simply insert a Label in the caption and refer to it using a Cross Reference as normal. It is especially important to use these with figure floats, rather than using vague references to “the above figure,” as LaTeX will reposition your floats for you in the final document; it might not be “above” at all. If it is not possible to fit the floats neatly on the same page as the text which refers to it, the figures will be placed on a separate page by themselves. Rest assured that the overall effect is usually quite nice.

Note that the caption is used in a List of Figures (as described in Section 4.5.2) automatically, should you choose to include one in your document.

4.3.2.2 Float Placement

Now, the whole idea behind Figure Floats [as well as Table Floats, which we introduce later] is to allow LyX to place a figure [or table] on a page in a consistent, sensible fashion. The rules LaTeX uses are rather arcane; refer to the LaTeX documentation for the exact details. You can use check boxes in the float dialog to set placement for a particular float. By default, each float uses the document’s default placement rules. You can change these, if you wish, in the Document ▷ Settings dialog. The Float placement box takes a LaTeX-style placement specification. You can place any combination of four letters in the Float placement box, in any order:

- h for *here*
- t for *top*
- b for *bottom*
- p for *page*

The letters correspond to the following behaviour:

Here: LyX tries to put the Float at the same point in the text where you put it.

If there isn’t enough room, LyX tries one of the other three location types.

Top: LyX tries to put the Float at the top of the current page. If the figure won’t fit on the current page, it goes to the next page.

Bottom: LyX tries to put the Float at the bottom of the current page. If there isn’t room, it goes to the next page.

Page: LyX tries to put the Float (or a number of Floats) on a page of its own.

There is some subtlety to how this all works. The order specifies what location LyX should try first. If that one fails, it tries the next one, and so on, though "h" will always take precedence if it appears in the list. The default placement list is "tbp": try the top of a text page first, then the bottom of a text page, then on a page by itself. If you want LyX to try "really hard" to place the figure where you command it, precede the list with an exclamation point; for example "!htbp". Here are some example entries and what they do:

1. hbp

   Try putting the figure/table at its actual position in the text. If that doesn't work, put it on the bottom of the page. If that fails, put it on a separate page.

2. !hbp

   Try really hard to put the figure/table at its actual position in the text. Then the bottom of the page, then on a separate page.

3. tp

   Put the figure/table at the top of each page. If it's too long, put it on a separate page.

4. p

   Always put figures and tables on their own page.

4.3.3 XFig and LyX

One obvious question is "how would I create the figures?" Fortunately, the answer is included in most Linux and/or LaTeX distributions. `XFig` is a powerful though slightly awkward drawing tool. If you want to include figures that you have created with `XFig` there are several ways. We recommend the following:

1. Export the figure as Encapsulated PostScript. This could be very easy included into LyX as described in the previous sections. The great advantage of this way is, that you have the full power of PostScript® available. That means Bezier curves, colors, all line thicknesses and many more. If you have inserted text into your fig-document this will be printed with PostScript fonts, which is OK. The figure can be manipulated like any other EPS figure, as described above.

   The only disadvantage is that you cannot create formulas as PostScript text except by hand. If you also need formulas or simple exponents or indices in your figure, the next way is recommended.

2. Export the figure as LaTeX. This is just as easy to include into LyX, with the advantage that you may use all LaTeX commands within the text inside XFig. Therefore you have to set the *special flag* for text in XFig. This is automatic if you invoke XFig with `xfig -specialtext`. If this is done and you have also

chosen a LaTeX font you may simply write “H_2” in XFig. If you export this figure as LaTeX and include it in LyX with Insert ▷ Child Document (see description in *Extended Features*) this text will appear as H_2.

The disadvantage of this way is that the graphical power of LaTeX isn't as strong as PostScript®. You cannot use all thicknesses of lines and, more annoyingly, not all slopes. This is why we recommend the third way for more complex figures.

3. Export the figure as LaTeX/PostScript combined. Then XFig [transfig, really] will generate two files:

   a) the PostScript part foo.pstex, that contains all painting.

   b) the LaTeX part foo.pstex_t, that contains all text and a link to the PostScript part.

Then you just have to include the LaTeX part as described above. This will automatically include the PostScript part, too.[2] This way you have the full PostScript® and LaTeX power combined except for the possibility to scale the figure after creating. So if you want scalable pictures, the PostScript format is your only choice. Another little advantage of letting LaTeX typeset the font is that the same font will appear in your figures as in your text, which looks a little nicer.

4.4 Tables

LyX has powerful table support, but LaTeX can do many more things with tables than LyX is currently capable of, so you might want to look at a good LaTeX book if the features described here should turn out to be inadequate.

You can insert a table using either the table toolbar button or Insert ▷ Table. A dialog will appear, asking you for the number of rows and columns. The default table has lines at the top and to the left of every cell, a line to the right of the rightmost column and a line at the bottom of the lowest row, forming a box around the table. Additionally, the topmost row also has a line at the bottom, which causes this row to appear separated from the rest of the table. Here's an example:

[2] If you get an error like “unknown graphics extension pstex” you have to declare these graphic extensions. I think this is a transfig bug that occurs with LaTeX 2$_\varepsilon$. Simply add a line like

```
\@namedef{Gin@rule@ps_tex}#1{{eps}{ps_tex}{#1}}
```

in the file /usr/lib/texmf/tex/latex/graphics/dvips.def. Then add pstex to the extension:

\def\Gin@extensions{eps, ps, pstex, eps.gz, ps.gz, eps=2EZ}

This should fix the whole thing. Alternatively you may export the postscript part as foo.eps and change the LaTeX part foo.pstex_t manually. But this is annoying.

	12	45	98
A			
B		multi	
C			

4.4.1 The Table dialog

You can alter a table by clicking on it with the right mouse button, which brings up a settings dialog. Among these options are:

- Adding/removing border lines from a row or column. If you remove the top line from one of the rows, you'll get a dotted line in LyX, but no line will appear in the printout. If you set the bottom line of one row and the top line of the row below, then the rows are separated by a small space, as you can see with the top row in the example above. You can do the same vertically if you set the right line of a column and the left line of the column to the right.
- Text alignment in a column
- Appending rows and columns
- Deleting rows, columns, or the entire table
- Multicolumn
- Setting a fixed width for a column
- Longtable options - this is useful if your table is higher than the paper. Then the table is split on the bottom of the page and continued on the next one, instead of running of the end of the page.
- Rotate the whole table or a single cell sideways, by 90 degrees

You can also use the menu to perform these operations. Try Edit ▷ Table or Edit ▷ Rows&Cols when the cursor is inside a table.

Most of these options also work on selections. This means that if you select more cells, columns or rows the action is done on all of your selection. Note that there is a difference between selecting the *contents* of the cell, and the cell itself. If you can see a red border inside a cell, then a selection will select the contents. If you press Escape or click outside of the box, then the selection will select cells (whether you use the mouse or the normal cursor-movement keys).

When you append a row, it is added *below* the row containing the cursor. Similarly, columns are appended to the *right* of the cursor. This makes it difficult to add columns on the left edge of a table without a lot of cutting and pasting. Deletion is always performed on the row or column containing the cursor.

The multicolumn option merges two or more adjacent cells on a given row. For example, in the above table, row “B” has had multicolumn applied to the columns labelled “45” and “98.” To use it, you must first select the cells, then choose Multicolumn from the menu. This will not work vertically - see the Table Examples document for how to do this.

You can also use Multicolumn if you need to have a special handling for a single table cell’s top and bottom border lines and text alignment. Here an example of this special handling of a cell:

*	x	y
point a & b	103	9
point b & a	599	340
point abc	1009	52
point abcd	96	11

You see here that the header line cells are aligned to the center, whereas the left column is aligned to the left, and the other columns are aligned to the right. Also the bottom and top line of two cells have been removed.

If you want your column to have a fixed width, then you can insert a width in the Width input-field of the Table dialog. This will then allow the cell to have multiple paragraphs of text.

If your table becomes too large to fit on a portrait document layout, you can select the Rotate 90° button, and the table will appear sideways (this means landscape in a portrait document style). You might also like to rotate single table cells to give them more horizontal space. The example below demonstrates why it is useful to rotate single cells.

Note: This Rotate 90° option will *not* display on screen, and works *only* for PostScript output. So, if you want to preview them, use View ▷ Postscript, as View ▷ DVI will not show the table properly.

Description	Flag 1	Flag 2	Flag 3	Flag 4	Flag 5	Flag 6	Flag 7	Flag 8	Flag 9	Flag 10	Total
Desc. 1	*		*	*		*		*	*	*	7
Desc. 2		*	*	*	*	*	*	*	*	*	9
Desc. 3	*	*	*		*	*	*	*		*	8
Desc. 4	*		*	*			*	*	*		6
Total											30

4.4.2 What can be placed inside a table cell?

Many objects can be placed inside a table cell. Any single line of text, an equation (not a displayed or multilined equation, though), or a figure can be in a cell; in fact, all three kinds of objects can be placed in the same cell. Font sizes and shapes can be altered, and the table will adjust to display them properly. However, you can’t put a special environment in a cell (like Section*, etc.), nor set spacing options etc. for the cell’s paragraph.

4.4.3 Cut & Paste in Tables

Cutting and pasting between tables works reasonably well. You can cut and paste even more than one row. Selection with the mouse or with Shift plus the arrow keys works as usual. The values in the second table below were cut and pasted from the first, using the mouse to select and paste.

1	2	3
4	5	6
7	8	9

1	2	3
4	5	6
7		

Note that you can also copy and paste the entire table as a single unit by starting the selection from outside the table.

4.4.4 Multiple lines in cells

It is possible to have multi-line entries in tables, but not in a completely WYSIWYM manner. Define a fixed length for the column in the Table dialog. After this, your text is automatically split into more lines and the cell enlarged vertically when the length of the text exceeds the given fixed length. An example:

1	2	3
4	This is a multi-line entry in a table.	5
6	This is longer now.	7
8	This is a multi-line entry in a table. This is longer now.	9

Text within a cell will not normally wrap to fit the page, so if a line of text in a table is too long, the table will extend beyond the right margin of the page. Similarly, tables will not split themselves at the bottom of a page, and so might extend below the bottom margin. You have these options to resolve this problem:

1. Split it into two tables.
2. Select the Longtable button in the Table dialog. This automatically splits the table over more pages, if it is too tall. After doing this, the list of Longtable buttons activate themselves and you may now define:

a) First header: The current row and all rows above that don't have any special options defined are defined to be the header rows of the first page of the longtable.

b) Header: The current row and all rows above that don't have any special options defined are defined to be the header rows of all pages of the longtable; except for the first page, if First header is defined.

c) Footer: The current row and all rows below that don't have any special options defined are defined to be the footer rows of all pages of the longtable; except for the last page, if Last footer is defined.

d) Last footer: The current row and all rows below that don't have any special options defined are defined to be the footer rows of the last page of the longtable.

If you set more than one option in the same table row, you should be aware of the fact that only the first flag is used in the given table rows. The others will then be defined as *empty*. In this context, first means first in this order: Footer, Last footer, Header, First header. See the `TableExamples.lyx` example file to see how this works.

The check box in the long table options can be used to specify specific rows to break the page on as well.

3. A table can also be placed in a float, as described below, which will allow TEX to place it as well as it can within the page.

4.4.5 Table Floats

Outside of a float, the table will be positioned exactly where it is placed in the document. Using a Table Float from the Insert ▷ Float menu will enable LATEX to place the table where it fits best, rather than exactly where you insert it. Float placement for table floats is similar to that for figure floats 4.3.2.1, and is described in section 4.3.2.2. Captions also work the same way as with figure floats, as described in section 4.3.2.1. Table 4.2 is an example of a table float.

Table 4.2: A table float.

1	2	3
Joe	Mary	Ted
$\int x^2dx$	$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$	$1+1=2$

4.5 Table of Contents and other Listings

One of the really nice features of LaTeX is the ease with which it lets you create various "Lists," such as a Table of Contents. All you need to do is to use certain environments and insert a reference at the place where you want the list to appear.

4.5.1 The Table of Contents

In order to get a Table of Contents, you need to do four things:

1. Use a document class that includes support (all but `letter`).
2. Set paragraph environments appropriately: Chapter, (Sub...) Section, (Sub...), Paragraph. Note that styles with a *, like Section*, will *not* appear in the Table of Contents.
3. Make sure you set the Section number depth and Table of contents. depth in the Document Layout dialog to the appropriate value as described in 3.3.4.3
4. Insert the ToC command at some place in the document. You'll find it under Insert ▷ List / TOC ▷ Table of Contents.

You can also bring up a dialog for navigating through your document with Document ▷ Table of Contents.

4.5.2 List of Figures, Tables and Algorithms

Table, figure, and algorithm lists are very much like the table of contents. You can insert them from the Insert ▷ List / TOC submenu. If you want figures, tables, or algorithms to appear in the list, you must place them inside a float of the relevant type and add a caption.

5 Mathematical Formulae

5.1 Basic Math Editing

To create a math formula, you can just click on the toolbar icon with $\frac{a+b}{c}$ on it. That will open a little blue square, with purple markers around it, on the corners. That blue square is the formula itself; the purple markers indicate what level of nesting within the formula you are at. You can also choose a particular formula type to insert via the Insert ▷ Math menu; or you can use a keyboard macro, M-c m, M-m m, or C-m (CUA binding only).

If you simply need to type a single Greek letter, such as α, there is a special shortcut. Just type M-m g a to get α, M-m g b to get β, etc.

Editing the parameters of a formula may be done from the Insert ▷ Math ▷ Math Panel dialog, by Edit ▷ Math, or by clicking the right mouse button on the formula. The math panel is very useful, so you may want to open it and leave it somewhere on the screen. If you're not already in a formula, selecting anything from the math panel will insert a formula for you.

5.1.1 Navigating a Formula

The best control over cursor position within an existing formula is achieved with the arrow keys. Mathed uses small squares to indicate places where something can be inserted. The arrow keys can be used to navigate between parts of a formula. Pressing Space will leave a fraction or other formula construct (a square root $\sqrt{2}$, or parentheses (f), or a matrix $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$). Pressing Escape will leave the formula, placing the cursor after the formula. Tab can be used to move horizontally in a formula; for example, through the cells of a matrix or the positions in a multi-line equation.

Space seems to do nothing in Mathed, since it does not in fact add a space between characters, but it does exit a nested structure. For this reason, you have to be careful about using Space. For example, if you want $\sqrt{2x+1}$, type `\sqrt` then Space, then `2x+1`, not `\sqrt` Space `2x` Space `+` Space `1`, since in the latter case only the $2x$ will be under the square root sign, $\sqrt{2x}+1$. For those who learned to space out expressions in this way, it takes a little unlearning.

You can leave many parts of a formula, like this matrix, partially filled in, such as:

$$\begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix}.$$

If you leave a fraction only partially filled in, or a subscript with nothing in it, the results will be unpredictable, but most constructs don't mind.

5.1.2 Selecting Text

You can select text within a formula in two different ways. Place the cursor at one end of the string of text you want, and press Shift and a cursor movement key to select text. It will be highlighted as with regular text selection. Alternatively, you can select text with the mouse in the usual way. That text can then be cut or copied, and then pasted within any formula (not in a plain text region in LyX, though).

5.1.3 Exponents and Subscripts

You can use the math panel to add superscripts or subscripts, but the much easier way is to use the standard TeX method. To get x^2, type (in Mathed) `x^2` then Space. The final Space puts the cursor back down on the base line of the expression, instead of in the superscript. If you type `x^2y`, you will get x^{2y}, to get x^2y, type `x^2` then Space then `y`. Subscripts are similar, to get a_1, type (in Mathed) `a_1` then Space. Note that by default, the superscript or subscript is only for the single symbol to the left, which changes the spacing and alignment; you should read section 5.3 if you need to alter this.

5.1.4 Fractions

Create a fraction with either `\frac` (in Mathed) or using the fraction icon in the Math Panel dialog or the Math menu item Fraction. You will be presented with an empty fraction, with two Mathed insertion squares top and bottom. The cursor moves immediately to the top of the fraction. To move to the bottom, simply press Down. To move back up, press Up. Any math structure can be placed in a fraction, as this example shows:

$$\left[\frac{1}{\left(\begin{array}{cc} 2 & 3\\ 4 & 5 \end{array}\right)}\right]$$

5.1.5 Sums and Integrals

Sum ($\sum$) and integral ($\int$) signs are very often decorated with one or more sets of "limits". These limits can be entered in LyX by entering them as you would enter a superscript or subscript, directly after the symbol. Sum will automatically place its "limits" over and under the symbol in display style, but will move them to the side when inlined, such as $\sum_{n=0}^{\infty}\frac{1}{n!}=e$, versus

$$\sum_{n=1}^{\infty}\frac{x^n}{n}=\ln\left(\frac{1}{1-x}\right).$$

Integral signs, however, will not by default move the limits to directly over and under the integral sign in display style, as in $\int_a^x f(t)dt := F(x)$, versus

$$\int_{-\infty}^{\infty} \frac{dx}{1+x^2} = \pi.$$

Both symbols will be automatically re-sized when placed in display mode. In display mode, the placement of the limits (directly above and below, or offset to the right from the sign) can be changed by placing the cursor in front of the sign and hitting M-m l. Exactly what change occurs depends on the sign.

Certain other mathematical expressions have this "moving limits" feature as addition, such as

$$\lim_{x\to\infty} f(x),$$

which will place the $x \to \infty$ underneath the "lim" in display mode, but not in inlined mode, $\lim_{x\to\infty} f(x)$. Note that the lim was entered as a function - you get it in LyX by typing `\lim` in math-mode, or choosing from the "functions" menu in the math panel; see 5.1.9.

5.1.6 The Math Panel

The Math Panel dialog (accessible via Insert ▷ Math) has a more extensive list of symbols and structures. As stated earlier, you can keep the math panel open when writing mathematics. The use of the panel should be fairly obvious; we'll describe some of the details in later sections.

Note that right-clicking on a formula opens the panel as well.

5.1.7 Other Math Symbols

Most math symbols can be found in the math panel under one of several categories; including Greek $\Gamma\rho\epsilon\epsilon\kappa$, operators $\pm\times$, relations $\leq\cong$, arrows $\uparrow\Leftrightarrow$, large operators $\sum\int$, and the dreaded miscellaneous. There are also the additional symbols provided by the American Mathematical Society (AMS). If you know the standard LaTeX macro for a particular symbol you which to use, you do not have to use these dialogs, but they will help for those symbols whose LaTeX name you do not know. Note that the AMS symbols will not be displayed as symbols in LyX unless you install the right fonts as described in the relevant manual.

It is possible to get an nth root symbol. In the minibuffer, type `math-insert root`. This generates a root symbol with an extra box above the root sign. Use Up and Down to move between the two boxes. You can also use the key binding M-m-r.

5.1.8 Altering spacing

You may want to create blank spaces that differs from the standard spacing that LaTeX provides. We don't recommend this as a matter of course, since the whole idea of WYSIWYM is that you don't think about the typesetting, but the content. However, there are situations where you will want to add spaces. The first thing to do is to type C-Space. This generates a small space, and shows a small marker on the screen within LyX: $a\,b$. The next trick is to change that space to different sizes. *Before* you move the cursor, after typing C-Space, if you hit Space again, you will change the size of the space, through a number of variable sizes. The last ones in the list are red, and are a negative space. For example: $a\quad b$, or $a\!\!b$. You can also insert these spaces via the math panel.

5.1.9 Math functions

The math panel contains a number of "functions", such as sin, lim, *etc.* (you can type them in a formula by typing `\sin` etc). Standard mathematical practice is that functions which are names, like sin, should not be italicized. Entering just the letters *sin* within Mathed will give italics, of course, so these special macros are available. They do more to the final output than just change the typeface, however. For example, the expression $\sin t$ will typeset with a little extra space between the n and the t. For words which are more sophisticated mathematical objects, like lim, the macro changes the way that subscripts are placed, depending on whether the math-inset is inlined or displayed: $\lim_{x\to0}f(x)=L$ versus

$$\lim_{x\to0}f(x)=L.$$

These two expressions were typed the same way, but using the macro `\lim` alters the appearance (actually, it is the inlined version that is altered, to improve linespacing).

5.1.10 Accents

In a formula you can insert accented characters in the same way as in text mode. This may depend on your keyboard, or the bindings file you use. You can also use TeX macro equivalents, as macros. That is, you can enter $\hat{a}$ to get the same effect if your keyboard does not have accents enabled. This is entered by typing "`\hat a`" in Mathed. These are the equivalences between the text names and the macro names for the various accents:

text	math	example
circumflex	hat	$\hat{a}$
grave	grave	$\grave{a}$
acute	acute	$\acute{a}$
umlaut	ddot	$\ddot{a}$
tilde	tilde	$\tilde{a}$
dot	dot	$\dot{a}$
breve	breve	$\breve{a}$
caron	check	$\check{a}$
macron	bar	$\bar{a}$
—	vec	$\vec{a}$

Finally, you can choose one of these accents by selecting an item from the Decorations symbol set in the math panel; this will apply to any selection you have made within a formula too.

5.1.11 The math editor for LaTeX users

Editing mathematical expressions in LyX can be done in one of two ways. You can use the native LyX support for rendering the formulae in a WYSIWYM fashion. LaTeX users might like to be able to use the keyboard to enter things like `$\alpha$` (this gets, in ordinary TeX, an α in the final document), believing that it is faster than chasing around menus for a symbol. Here's a testimonial of one of those old LaTeX users, David Johnson:

> I was finally convinced that the math editor was the way to go when I found that, with a few modifications, I could use it the same way I was accustomed to writing TeX. As an example, I created this α by typing the following keys: First type M-c m to enter `math-mode`, then type `\alpha` , then Space and Esc. As soon as I typed that Space the α was right there on the screen.

The M-c m sequence inserts a formula (you may also use C-m or M-m m), the `\alpha` is of course the standard TeX command for a Greek alpha letter, and the Esc leaves the formula. Some of the advantages of this approach are:

- You have immediate visual feedback to be sure your TeX was correct
- You have the real mathematical expression on the screen, correctly displayed, to make sure your mathematics is correct (correctly written, at least)
- All the new LaTeX fuss with special environments and such are taken care of by LyX, not you
- You won't have to chase through the code trying to find that missing $ or extra { any more

- If you don't remember the LaTeX name of a particular symbol, like $\wp$, you can find it in the dialogs

5.2 Brackets and decorations

There are several brackets available through LyX. For most purposes, using just the keys `[{]}()|\<>` should suffice, but the effect, especially if you want to surround a large structure, such as a matrix or a fraction, or if you have several layers of brackets, is better using the math panel's Delimiter dialog [see sec. 5.1.6]. For example, that's how you would construct the brackets around a standard matrix such as :

$$\left[\begin{array}{cc} 1 & 2\\ 3 & 4 \end{array}\right],$$

and to make it easier to see the layers of parentheses of an abomination such as:

$$\frac{1}{\left(1+\left(\frac{1}{1+\left(\frac{1}{1+x}\right)}\right)\right)}$$

or:

$$f\left(g\left(h\left(k\left(l\left(x\right)\right)\right)\right)\right).$$

The parentheses, and other brackets, from that menu will automatically re-size to accommodate the size of what is inside (This is done in straight LaTeX by `\left( blah \right)`).

It is very easy to construct the braces you want to use. Click on the brace you want on the left side with the left mouse button, the right side with the right button, and place them in the document by clicking on the button. If you want one side to not have a bracket, use the blank button. It will appear in LyX with a dotted line, but nothing will print.

If you decide after the fact to place parentheses (or other math structure, like a square root, or other decoration) around some math structure, you can do that by highlighting (selecting) the structure that is to go inside the parentheses (that is done by holding the Shift key down and moving the cursor with the arrow keys, or selecting with the mouse). Then, choose the appropriate brackets for left and right, and click on Apply. The parentheses will be drawn around the selected structure.

If you're trying to enter a LaTeX `{` for grouping, you should read 5.3.

5.3 Grouping

You may need to group a set of symbols. In LaTeX, for example, the typesetting of `{x^y}^z` is different from `x^{y^z}`:

$${x^{y}}^{z}\quad\text{differs from}\quad x^{y^{z}}$$

However, trying to type the } in LyX gives an actual closing brace in the output. To create this grouping, you need to use the key sequence \{ then type space. Inside LyX, you will see red braces indicating the grouping. The example directly above shows how this works.

5.4 Arrays and Multi-line Equations

Arrays, such as matrices, are easily entered in LyX. In the Math Panel there is a matrix button, which will open a dialog for you to choose the number of rows/columns. Here is an example:

$$\left(\begin{array}{ccc}
1 & 2 & 3\\
4 & 5 & 6\\
7 & 8 & 9
\end{array}\right).$$

The parentheses aren't automatic, but you can add them as usual. Remember that you can add this after the fact, by highlighting the matrix inside Mathed (Position the mouse on one side of the matrix, hold the Shift key down, and hit the appropriate arrow key to move the cursor across the matrix). You can, when you construct the matrix, decide whether the columns (or some of them) will be left-, right-, or center-justified. The specification is `ccc` by default. Each letter corresponds to the relevant column. For example, `lcr` means that the first column will be left-justified, the second will be centered, and the third column will be right-justified.. It will look like this:

$$\begin{array}{lcr}
this & this\,column & this\,column\\
column & has & has\,right\\
has\,left\,alignment & center\,alignment & alignment
\end{array}.$$

You can add more rows to an existing matrix by hitting C-Enter while in the matrix, and you can add columns, or delete either, via the Edit ▷ Math and Edit ▷ Rows&Cols menus.

There are numerous other arrays used in LaTeX math-mode, particularly with the AMS-LaTeX packages included, such as `\cases` and commutative diagrams. Not all of these are supported in LyX but some are, see Insert ▷ Math.

Multi-line equations are very easy to construct in LyX. A formula will automatically switch to an `eqnarray` format (LaTeX's multi-line displayed equation format) if you hit C-Enter. The best way to do this, if you decide you want a multi-line displayed equation, is to insert a new line (with C-Enter) immediately. Each line then has three regions, left, center, and right, which you can move through using either the arrow keys, the mouse, or the Tab key. Here is an example:

$$\begin{array}{rcl}
3 & = & 1+2\\
4+5 & = & 9.
\end{array}$$

You can also turn an existing displayed formula into a multi-line formula by hitting C-Enter while the cursor is anywhere on the original formula. However, LyX will *not* try to decide where to break the formula up into three parts, but places everything in the left side of the line. To change the alignment points of the equation, place the cursor where you want to start the middle part of the line, and hit C-Tab. It then puts everything to the right of the cursor in the middle region of the equation (which, by the way, is not typeset by LaTeX in display-math size, so you should not put large expressions like fractions there). Move to where you want the right side of the line to begin, and hit C-Tab again. The “extra” insertion points in the line will disappear.

5.5 Equation Numbering and Labels

Equation numbering is very easy in LyX. All it takes to change a displayed equation like:

$$1+2=3$$

into the numbered equation :

$$1+2=3 \tag{5.1}$$

is to go to the Insert menu, and select the Label... option. This opens a dialog in which you must enter some string as the label. There is no need to call it by a specific number, since LaTeX will take care of re-numbering the equation. Labels will not appear as such on the final output. LaTeX will insert appropriate numbers for the equations. The labels are used internally for cross-referencing. You can turn on numbering without a specific label with the menu option Edit ▷ Math ▷ Toggle Numbering while the cursor is in the equation, such as:

$$1+1=2. \tag{5.2}$$

You can toggle it on or off with this menu item. You can reference a labelled (not just numbered) equation, (cf. (5.1)) by using the Cross-Reference dialog, which you open using Insert ▷ Cross-Reference... .

For numbered (or labelled) multi-line formulas, the default is that all lines are numbered separately. Once you attach a label to make the equation numbered, all subsequent lines receive a label of #. That label can be changed to another so that you can refer to that line, like (5.4) below.

$$\begin{aligned} 1 &= 3-2 \tag{5.3}\\ 2 &= 4-2 \tag{5.4}\\ 4 &\leq 7. \tag{5.5}\end{aligned}$$

You can turn off numbering of a specific line with Edit ▷ Math ▷ Toggle numbering of line while the cursor is on that line of a multi-line numbered equation. This also toggles. For example :

$$\begin{aligned} 1 &= 4-3 \tag{5.6}\\ 2 &= 7-5 \tag{5.7}\end{aligned}$$

$$\begin{array}{rcl} 1 & = & e^{2\pi i} \\ 16 & \equiv & 2\,(mod\,7) \end{array} \tag{5.8}$$

Note that the first equation in this set (5.6) is labelled, the next is numbered but unlabelled, the third is unnumbered, and the last (5.8) is again labelled.

5.6 User defined macros in math mode

LyX allows the user to define macros for use in math mode. A macro definition box appears on screen as purple box with the name of the macro in blue (math color). It contains two cells initially marked empty by blue rectangles that can be edited as if it were ordinary math. Just try it: The contents of the first cell will be used when the macro definition is written during export as LaTeX. The contents of the second cell, however, will be used for drawing the macro's expansion on screen. In the common case where both export and drawing use the same representation, the second cell can be left empty and LyX will use the contents of the first cell will be used for export and drawing automatically.

Now, to use this macro in other math boxes just type the name in TeX mode, in this case `\macro`, and it will be automatically expanded: $c=a+b$. As you can verify, the cursor can't go inside the macro, the whole macro is like a single character, and the TeX generated code of this expression is `c = \macro`.

However the cursor could go inside of some kind of macros, those that have *arguments*. In a macro definition box an argument looks like a `#` followed by the argument number:

Once expanded, this macro includes the usual empty rectangle to indicate that you can insert there whatever you want: `\macroarg =` $2+\sqrt{}$. Example: $b=2+\sqrt{x-2}$.

When exported to LaTeX, a macro definition will produce the command
`\newcommand{\macroarg}[1]{2+\sqrt{#1}}`

5.6.1 How to create macros

To create a macro definition box use this syntax in the minibuffer:[1]

`math-macro <macro name> [number of arguments]`

For example, `\macro` was created with "`math-macro macro`", and `\macroarg` was created with "`math-macro macroarg 1`".

To insert an argument mark (only inside a macro definition box) simply type
`#<number>` or use `math-macro-arg <number>`

The argument mark in `\macroarg` was introduced with "`math-macro-arg 1`".

You can use no more than 9 arguments, numbered from 1 to 9. An argument can be repeated inside the macro definition box, but of course can be edited only once.

[1] Macro names mustn't contain numbers!

5.6.2 How to navigate in macros

With the arrow keys: Opening a macro from the left side will put the cursor in the first argument, to move to the second argument use the TAB key. Remember that pressing the Space bar will get the cursor out and at the right side of the macro.

With the mouse: As usual, click on the desired argument box. Sometimes this fails if the box is empty or too small.

Currently it is only possible to define command macros, but not environment macros.

5.7 Fine-Tuning

5.7.1 Typefaces

You can use various typefaces in a formula. The standard font for text is italic, *text*, but for numbers the standard is Roman. To set a font in a formula, choose it from the math panel, or by entering the LaTeX command for it directly, as follows:

Font	LaTeX Command
Roman	`\mathrm`
Bold	`\mathbf`
Italic	`\mathit`
`Typewriter`	`\mathtt`
$\mathbb{BLACKBOARD}$	`\mathbb`
$\mathfrak{Fraktur}$	`\mathfrak`
$\mathcal{CALLIGRAPHIC}$	`\mathcal`
SansSerif	`\mathsf`

LaTeX's math mode does not support all characters in all fonts, and only letters will be supported with these font styles; some only support capital letters.

For any of these fonts, you have to be careful how you enter the text. If there is text to the right of the entry point, the font reverts to that style after one character. To be able to type a string in a particular font, make sure there is a protected-space to the right of the cursor. Also, entering a protected-space will revert subsequent text to standard font. The font styles are nestable, as LaTeX does. This can be a little confusing, as selecting a different font on a selection will *not* change the selection, but insert a new nested level with the new typeface.

It is possible (in AMS-LaTeX) to embolden (not italicize) numbers and special symbols. However, LyX does not yet support this in WYSIWYM manner. It will print correctly, though. To get emboldened symbols, for example a bold α, enter `\boldsymbol{\alpha}` in Mathed. The closing brace appears (in red) automatically when you type the opening brace. This works for all symbols, as well as numbers.

A number of other options are available as well, via Insert ▷ Math ▷ Font Change.

5.7.2 Math Text Mode

Typefaces are useful for entering variable names in some given font, but certainly not for anything else, and in particular not text. For typing longer pieces of text, use math text mode, which is obtained by typing M-m m while already in math mode. (The same command will get out of math text mode, too.) Math text mode appears on the screen in black instead of blue. You cannot enter punctuation or font changes in your text[2], but it works for simple text. Here's an example:

$$f(x) = \begin{array}{rl} x & \textrm{if I say so} \\ -x & \textrm{otherwise} \end{array}$$

5.7.3 Font Sizes

There are four (relative) font sizes (or "styles") used in math-mode, which are automatically chosen in most situations. These are called *textstyle*, *displaystyle*, *scriptstyle*, and *scriptscriptstyle*. For most characters, *textstyle* and *displaystyle* are actually the same size, but fractions, superscripts and subscripts, and certain other effects, are set larger or placed differently in *displaystyle*. Except for some operators, which re-size themselves to accommodate various situations, all text will be set in these various sizes as LaTeX thinks is appropriate. These choices can be over-ridden by using the `math-size` function in the minibuffer. For example, you can set $\frac{1}{2}$ normally (*textstyle*), or you can make it larger, which also changes the line-spacing, by entering `math-size displaystyle` in the minibuffer while the cursor is in the main line of the math-inset, $\dfrac{1}{2}$. Careful, though, if the cursor is on the denominator of that fraction, only the numerator will be enlarged, e.g. $\frac{1}{2}$! This reflects a LaTeX "unintended feature"[3], not a LyX one. These font-size changes are not as apparent in LyX as they are in the output. Here are some text in the various styles: *displaystyle*, *textstyle*, *scriptstyle*, *scriptscriptstyle*.

All these math-mode font sizes are relative, that is, if the whole math inset and surrounding text are set in a particular size, all these sizes will be adjusted. Similarly, if the base font size of the document is changed, all fonts will be adjusted to correspond.

Here is a paragraph in "largest" font, with symbols: α.

This applies to math fonts in titles, etc. as well.

[2] Moreover, math text mode outputs its contents inside a `\textrm{}`, whereas and `\mbox` (or AMS-LaTeX's `\text`) might have been a better choice

[3] That is, a bug.

5.8 AMS-LaTeX

The American Mathematical Society (AMS) provide a LaTeX packages that are in common use. LyX includes some support for these packages.

5.8.1 Enabling AMS-Support

In the Document ▷ Settings dialog there is a checkbox, Use AMS Math. If selected, this will include the AMS-package in the document, and make the facilities available.

5.8.2 AMS-Symbols

The AMS-LaTeX packages add support for some mathematical symbols that are not accessible from plain LaTeX (or LyX), but are fairly common in mathematical typesetting, such as the old-German Fraktur font and the stylized "blackboard bold" fonts commonly used to denote the real or complex numbers, or the integers. Once activated, all AMS-LaTeX symbols and environments are available. You will run into trouble if you include these packages from the preamble, since LyX now defines a few of the macros used in these packages on its own. The AMS-layouts include these packages automatically.

5.8.3 AMS-Formula Types

AMS-LaTeX provides a selection of different formula types. LyX allows you to choose between `align`, `alignat`, `flalign`, `gather`, and `multline`. Refer to the AMS-documentation for the differences between these formula types.

6 More Tools

6.1 Cross-References

Those of you reading this manual online will see a grey box with text in it, right before the beginning of this sentence. This is a Label. Properly speaking, it is one half of a cross-reference. The other half is the Reference proper, and it looks like this: 6.1. Again, those of you reading the manual online will see a gray box with text in it. Those reading printed versions, however, will see a number — in this case, the number of this section. There are also other varieties of cross-reference: for example, 83. This is the page number containing the location of the label. That's what cross-references do: they let you reference other parts of your document. You don't need to remember which section number was what anymore — LyX will do that for you! All you need to do is use a Label to mark a section, figure, table, formula, etc., and then refer to it via a Reference.

To insert a label, use Insert▷Label. A box will appear where you can enter your label. You can change the name of the label at a later time by simply clicking on the gray box and reopening the label dialog.

To insert a reference, select Insert▷Cross Reference. The Insert Cross-Reference dialog appears with a list of labels. Selecting a list item, then clicking OK inserts a reference into the text; changing the Reference type allows you to insert a page number or other reference variant instead.

Note that if you cut & paste text from another document that contains a Label or Reference, or if you delete a label in your text, LaTeX will complain:

```
LaTeX Warning: Reference ‘X’ on page Y undefined on input line
Z
LaTeX Warning: There were undefined references
```

You'll also see two question marks in the output instead of the reference.

There are a few more comments we need to make about the Labels. They always print the number of the section heading closest to them. So — if you want to put a label on a Chapter, but a Section heading immediately follows it, you need to put the Label *into* the Chapter environment. It doesn't matter where, and it will look weird on the LyX screen. However, you need to do this if you want to label the Chapter separately from the Section. The same goes for all other section headings.

Also, a Label *only* makes sense in *numbered* section headings and table and figure floats. Bare figures and tables aren't numbered, so, like unnumbered section headings, you can't really use a Label on it.[1] See sections 4.3.2.1, 4.4.5, and 5.5 for details on using a Label with figures, tables, and equations, respectively.

6.2 URLs (Uniform Resource Locators)

It is often desirable to include long "verbatim" items in a document such as Web site URLs, e-mail addresses, etc.; these things typically do not contain any spaces and are thus difficult to typeset properly. Such items will often fall on a line boundary if they cannot be split, resulting in an overfull or underfull line depending on the circumstances. You can use Insert ▷ URL within LyX to enter a long URL and have it split gracefully (if necessary) along automatically determined boundaries.

At the point in the document where you want to enter the URL (or other address-like entity) simply select Insert ▷ URL; a dialog will appear where you can enter the full URL (in the Url: field). In its simplest usage, that's all you need to do. Click on the following gray box to see how LyX's homepage would be entered: http://www.lyx.org.[2]

If you would like to associate some definite phrase with the URL, enter it into the Name field of the dialog; it will be typeset as plain text immediately before the URL. For example, I might say that you can find all things related to LaTeX at CTAN http://ctan.tug.org. On the printed page, the last sentence ends as "all things related to LaTeX at CTAN http://ctan.tug.org".

[*Author's Note: somebody needs to document the* HTML Type *button*]

6.3 Specifying Short Titles with Optional Arguments

Some section or chapter titles, such as this one, can get quite long. This can cause over-runs when there is limited horizontal space. For example, if the header of the page is set to show the current section title, a long title will over-run past the edges, and look awful.

LaTeX allows you to specify an optional argument to the section commands that specifies a shorter version of the title[3]. This shorter version is used in the header and in the actual Table of Contents, avoiding the problem mentioned. LyX allows you to specify this optional argument by selecting Insert ▷ Short Title. This will insert a box

[1] Well, you *can*, but only if you use the Page number reference. The regular Reference — the one that refers to a section/table/figure number — won't work, because there's no numbered thingy to refer to! You could also use bare Labels as page markers, then refer back to them using the Page number reference. Once again, the regular Reference won't work very well. It will refer to something, but that something will typically be the number of the previous numbered section heading.

[2] **Important note**: When you use the following characters: "%", "#", "^", you have to write them with a backslash before, e.g. "\#". URLs mustn't end with a backslash!

[3] For those who don't know LaTeX, commands look like this: `\command[optionalargument]{the content}`

(labelled “opt”, which stands for “optional”) which you can use to enter the short title text. This also works for captions inside floats.

The title of this section is a good example of using this feature.

6.4 Branches

Sometimes you wish to be able to output to paper multiple versions of the same document. The most extreme version of this is, when you want to prepare a single document in two or more different languages, but as a single document file, with corresponding pieces of text adjacent in the file and on-screen. This can be achieved with *branches*.

First, you have to *define* the branches available within a particular document. This is done in the Document ▷ Settings ▷ Branches tab. You can also associate a background colour with each branch, e.g., red for the English language, blue for the German language branch. Then, you create a branch inset from the Insert ▷ Branch menu item. The inset will contain the text that you want to be output when this branch is activated.

Activation or *deactivation* of a branch is done from the document settings menu. All insets belonging to deactivated branches will be automatically closed, those belonging to activated branches automatically opened.

Other possible applications of the Branches paradigm include a “teacher’s version” of a textbook containing the answers to questions, etc.

6.5 Previewing snippets of your document

LyX allows you to generate previews of sections of your document on the fly so you can see how they’ll look in the final document without having to break your train of thought with View ▷ DVI. If you’d like to see your math formulae typeset by LaTeX then install the necessary software (see below) and select the Instant preview “On” pulldown item in the Tools ▷ Preferences dialog. (It can be found in the Look and feel ▷ Graphics pane in the Qt frontend and the Look & Feel ▷ Misc tab in the XForms frontend.) Previews are generated when you load a document into LyX and when you finish editing an inset. Previews of an already loaded document are *not* generated just by selecting the Instant preview check box.

LyX will generate previews of math insets. It will also generate previews of include insets or “child documents” if you select the Show preview check box in the inset’s dialog. This latter is useful if you wish to generate a preview of a LaTeX figure, for example. Coming in version 1.4 are previews of the external inset also.

To get previews working, you’ll need some additional software. First, you’ll need the preview.sty LaTeX package. Find it on your local CTAN mirror at `CTAN/support/preview-latex/`. Thereafter, you’ll need the usual tools: `latex`, `dvips` and `gs`.Finally, you’ll obtain prettier results if you install `pnmcrop`from the `netpbm` package.

6.6 Spacing, pagination and line breaks

6.6.1 Extra Horizontal Space

HFills are a special LyX feature for adding extra space in a uniform fashion. An HFill is actually a variable length space, whose length always equals the remaining space between the left and right margins. If there is more than one HFill on a line, they divide the available space equally between themselves.

Note: if an HFill is at the beginning of a line, and it's *not* the first line in a paragraph, LyX ignores it. This prevents HFills from accidentally being wrapped onto a new line.

HFills can be inserted with Insert▷Special Formatting▷Horizontal Fill. Here a few examples what you can do with them:

> This is on the left side This is on the right
>
> Left Middle Right
>
> Left 1/3 Left Right

That was an example in the Quote environment. Here: :is one in a standard paragraph. It may or may not be apparent in the printed text, but it *is* sitting in-between the two ":".

Remember that we said that an HFill always fills the remaining space between the margins? There may be more than one set of margins on a line. Here's an example with the List environment.

one two :three four five six

The ":" marks the beginning of the item. (There is actually a "hidden" HFill inside of the label of the List environment; it's put at the end of the label automatically.) HFills work similarly in other "multi-margin" situations, like two-column mode.

6.6.2 Extra Vertical Space

To add extra vertical space above or below a paragraph, use Edit▷Paragraph Settings to open the Paragraph Settings dialog.

We will not provide an example of a VFill, as it would waste paper. They work the same as any other type of filler, including HFills: they fill the remaining vertical space on a page with blank space. If there are several VFills on a page, they divide the remaining vertical space equally between themselves. You can therefore use VFills to center text on a page, or even place text 2/3 down a page, or 1/4, and so on.

Note that for paragraphs at the top/bottom of a page, the extra space is only added if you have also checked the option Edit▷Paragraph Settings▷Spacing▷Keep space.

6.6.3 Changing Paragraph Alignment

You can also change the paragraph alignment with the Edit ▷ Paragraph Settings dialog. There are four possibilities:

- Justified
- Left
- Right
- Center

The default in most cases is justified alignment, in which the inter-word spacing is variable and each line of a paragraph fills the region between the left and right margins. The other three alignments should be self-explanatory, and look like this:

This paragraph is right aligned,

this one is centered,

this one is left aligned.

In some paragraph environments, the default is something other than justified alignment.

6.6.4 Forcing Page Breaks

If you don't like the way LaTeX does the page breaks in your document, you can force a pagebreak where you want one. In general, this will *not* be necessary because LaTeX is good at pagebreaking, as was already mentioned in section 3.7.4.

So in general there is no need to use the option described below, and we recommend not using it until the text is finished, and until you have checked in the preview to see if you *really* have to change the pagebreaking.You can force a pagebreak above or below a paragraph in the Edit ▷ Paragraph Settings dialog by selecting the checkboxes to add a pagebreak above or below the paragraph.

You might try to use a pagebreak to ensure that a figure or table appears at the top of a page. This is, of course, the wrong way to do it. LyX gives you a way of automatically ensuring that your figures and tables appear at the top of a page [or the bottom, or on their own page] without having to worry about what precedes or follows your figure or table. See sections 4.3 and 4.4 and read about Floats to learn more.

6.6.5 Blanks/Spaces

A blank is a blank? Not in good typography. While you might be used to press the space key anytime you want to separate two words in ordinary word processors, LyX offers you more spaces: Spaces of different width and spaces which can or cannot be broken at the end of a line. The following sections will show you some examples where those spaces are useful.

6.6.5.1 Inter-word Space

Some languages (e. g. English) have the typographical convention to add extra space after an end-of-sentence punctuation mark, and LyX honors those conventions (see section 3.7.2.1). Sometimes, you want a normal space nevertheless. In this case, insert one with Insert ▷ Special Formatting ▷ Inter-word Space or with C-M-Space.

6.6.5.2 Protected Space

The protected space: It is used to tell LyX (and LaTeX) not to break the line at that point. This may be necessary to avoid unlucky linebreaks, like in:

> A good documentation should weight no more than 1
> kg.

Obviously, it would be a good thing to put a protected space between “1” and “kg”. A protected space is set with Insert ▷ Special Formatting ▷ Protected Space or with C-Space.

6.6.5.3 Thin Space

A “thin space” is a blank which has half the size of a normal space (and it is also “protected”). The typographical conventions in a lot of languages propose the use of thin spaces in cases where normal spaces would be too wide, for instance inside abbreviations:

> D. E. Knuth has developped our beloved typesetting program, i. e. TeX.

You can insert a thin space with Insert ▷ Special Formatting ▷ Thin Space or with C-S-Space.

6.6.5.4 More Spaces

Apart from the ones described, there are still some more spaces. Although LyX supports them natively, they can only be reached via the *minibuffer*. To get them, just type `space-insert <command>` into the minibuffer, where `<command>` is one of the following:

command	width	protected?
`normal`	1/3 em	no
`protected`	1/3 em	yes
`thin`	1/6 em	yes
`enspace`	0.5 em	yes
`enskip`	0.5 em	no
`quad`	1 em	no
`qquad`	2 em	no
`negthinspace`	-1/6 em	yes

6.6.6 Line breaking

You can force line breaks within a paragraph by selecting Insert ▷ Special Formatting ▷ Linebreak or with C-Return. You should, however, not use this to correct LaTeX's linebreaking, as LaTeX is *very* good at linebreaking. . . (see section 6.6.4). There are, however, a number of situations where it is necessary to actively set a linebreak, e.g. in a poem or for an Address (see sections 3.3.5.1, 3.3.5.2 and 3.3.7.2).

6.7 Spellchecking

LyX itself has no built-in spellchecker. Rather it uses the external `ispell` program as a backend or the newer and generally better `aspell`. This section assumes you have already installed and set up one of these programs.

The spellchecker can be started with the menu entry Tools ▷ Spellchecker. Checking will start just after the current cursor position. A dialog window will appear showing any incorrect (or unknown) word found, allowing you to edit and replace it in a second line. Whenever an unknown word is found, the word is highlighted and the view in your text buffer is updated to make the word visible. In the Spellchecker dialog, there is also a box showing suggestions for a correction, if any could be found. Clicking on one of the corrections will copy the near miss into the replace input field (double-click to invoke replace).

6.7.1 Spellchecker Options

The following options can be set in the Tools ▷ Preferences dialog.

6.7.1.1 Dictionary

By default, the dictionary file to use is determined by the language of the text you're checking, which is set in the Document ▷ Settings dialog. If you do not have a dictionary for the document language, the spellchecker will not work. In this case, you can specify another dictionary file in the dialog by specifying a different "alternative language".

If you're using `ispell`, you may need to make a link from say `deutsch.(aff|hash)` to `german.(aff|hash)` or whatever applies for your language. This is because these `ispell` files normally have the native language name ("deutsch") whereas `ispell`, when started from LyX, searches for the English version of the name used with the LaTeX babel package ("german").

You may also have problems the font encoding is not correct for that dictionary. If you use a language with `latin1` encoding and set the Encoding option in the Document ▷ Settings dialog to latin1 (or other than default), you must have this option in your language dictionary as well. If your dictionary doesn't support the Encoding you chose, you'll have an error like this on stderr:

```
ispell:  unrecognized formatter type 'latin1'
```

The spellchecker gives you an error that it couldn't start the `ispell` process and that you probably have some problems with your dictionary file.

There are four solutions to this problem. The easiest is to try the Use Input Encoding option. If that does not help, you can set Encoding to default when calling the spellchecker (which is probably annoying). The third is to add the `latin1` option to your dictionary `<`*language*`>.aff` file and recompile the dictionary (which probably isn't easy if you installed the whole stuff with some distribution and don't have the language directory of the `ispell` sources). Read the `ispell` documentation for this task! The fourth is to send a message to your package-maintainer, or better yet to the maintainer of the dictionary file in question and ask him to solve your problem.

6.7.1.2 Personal dictionary

If you want to use a different file from the spellchecker's default choice as your personal dictionary, you can set this in the dialog. Specifying a filename which does not already exist will result in an error message on stderr which you can ignore (`ispell` will create the file when you finish checking your spelling).

6.7.1.3 Further Options

The Spellchecker Options dialog has some additional options which are self-explanatory:

- Accept compound words
  Prevent the spellchecker from complaining about compounded words like ”passthrough”.
- Escape characters
  Allows you to add nonstandard characters to what the spellchecker considers words, e.g. German umlauts. This should not normally be needed.

6.7.2 Limitations

Some users have expressed a wish to be able to globally change the spelling of a particular word, rather than having to change the spelling separately for each occurrence of the word. Per-document word lists would also be useful. Neither of these features are present as of this writing.

Unless you're using the `pspell` spellchecker, LyX cannot correctly spellcheck documents containing multiple languages. This, does, however, work with `pspell`, assuming you have marked the different languages appropriately.

6.8 International Support

This section describes how to use LyX with any language you want. LyX comes with a default configuration which supports the English language on a U.S.-style keyboard, with a standard U.S. paper size and the spellchecker set to U.S. English. You can change any or all of these settings as desired, and you can make the changes apply to the current session only, or use them as your new default configuration.

If you have a keyboard suited to the language you are using (for example, a German keyboard for writing in German), and you have correctly configured your X environment, all you need to do for LyX is tell it your language, the character encoding, and desired paper size. Refer to 6.8.1 for more information.

If, however, you have a U.S.-style keyboard and want to write in a different language than English, you can use an alternate keymap. For example, if you have a U.S.-style keyboard but want to write in Italian, you can configure LyX to use an Italian keymap. Refer to 6.8.2 for details.

Finally, you may just want to change a few key mappings or create an entirely different keymap (for Vulcan, for instance). You may, for example, normally write in Italian on a U.S. keyboard but want to include an occasional quotation in German. In such a case, you can write your own keyboard mapping or modify an existing one to support the characters you want.

The details of how to customize LyX to your own language are *way* beyond the scope of this manual. You can not only alter the keyboard layout, you can also change the names of the menus buttons, etc., to reflect your language. If you want to learn more about writing keymap files and tailoring LyX to your native tongue, please see the *Customization* manual for details.

6.8.1 Language Options

The Document ▷ Settings dialog lets you set the language and character encoding for your language.

Choose your language by clicking on the arrow in the Language combobox of the Document ▷ Settings dialog. The default is U.S. English. Scroll to find the language you want and then click on your choice. The language name appears in the window.[4]

[4] In LaTeX terms, selecting a language other than default adds Babel support. If you do not have Babel installed, refer to the different LaTeX distributions for it.

The Encoding box lets you choose the character encoding map you want to use. The default is the `Latin1` encoding, which includes the characters required by the various Western European languages.

6.8.2 Keyboard mapping configuration

The preferences dialog allows you to choose up to two keyboard mappings. This allows you to choose the keymap of your choice for your U.S.-style keyboard. You can choose primary and secondary keyboard languages and then select which one you want to use.

6.8.3 Character Tables

Table 6.1 shows the `Latin1` character set. You should be able to enter the characters in the first eight columns directly from the keyboard.

There are a few things you need to know about this table. This manual is set up — by hand, mind you — to print all of these characters. That ain't the default. Nowhere near, in fact. Here are some of the details you'll need to bear in mind when using characters from the `Latin1` character set:

- The characters at entries A2, A4, A5, A6 and AD – the cent, the yen, the generic-currency-symbol, the broken vertical bar and the short dash are just plain missing in the default encodings. We don't know where they are or why this is the case.

- Even if you've selected latin1 in the Document ▷ Settings dialog, users who have only the `T1`-fonts for LaTeX [or who have the `T1`-fonts but aren't using them] will still miss a few characters: D0, F0, DE, FE, AB, and BB – the uppercase and lowercase eth and thorn, and the french quotes won't show up.

- Users of `T1`-fonts can, however, get the french quotes [characters AB and BB] if they include the either the package `umlaute.sty` or `german.sty` in their documents.[5]

The following is a full list of all of the accented characters LyX can display directly. It includes not only the accented characters from the previous table, but also the characters from `ISO8859--2` through `4`.

- From `ISO8859--1`:

¨ Ä Ë Ï Ö Ü ä ë ï ö ü ÿ	diaeresis
^ Â Ê Î Ô Û â ê î ô û	circumflex

[5] This only holds when you want to input these quotes by yourself. The automatic quote feature described in Section 3.7.2.2, will generate automatically LaTeX code adapted to available fonts and packages.

Table 6.1: The `latin1` character set

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
00				0	@	P	’	p				°	À	Ð	à	ð
01			!	1	A	Q	a	q			¡	±	Á	Ñ	á	ñ
02			“	2	B	R	b	r			¢	²	Â	Ò	â	ò
03			#	3	C	S	c	s			£	³	Ã	Ó	ã	ó
04			$	4	D	T	d	t			¤	´	Ä	Ô	ä	ô
05			%	5	E	U	e	u			¥	μ	Å	Õ	å	õ
06			&	6	F	V	f	v			¦	¶	Æ	Ö	æ	ö
07			‘	7	G	W	g	w			§	·	Ç	×	ç	÷
08			(	8	H	X	h	x			¨	¸	È	Ø	è	ø
09			)	9	I	Y	i	y			©	¹	É	Ù	é	ù
0A			*	:	J	Z	j	z			ª	º	Ê	Ú	ê	ú
0B			+	;	K	[	k	{			«	»	Ë	Û	ë	û
0C			,	<	L	\	l	|			¬	¼	Ì	Ü	ì	ü
0D			-	=	M	]	m	}			–	½	Í	Ý	í	ý
0E			.	>	N	^	n	~			®	¾	Î	Þ	î	þ
0F			/	?	O	_	o				¯	¿	Ï	ß	ï	ÿ

‘ À È Ì Ò Ù à è ì ò ù	grave
´ Á É Í Ó Ú Ý á é í ó ú ý	acute
˜ Ã Ñ Õ ã ñ õ	tilde
¸Çç	cedilla
¯	macron[6]

- From `ISO8859--2` through `4`:

ĤĴĥĵĈĜŜĉĝŝ	circumflex
ŚŹśźŔĹĆŃŕĺćń	acute
ĨĩŨũ	tilde
ŞşŢţŖĻĢŗļģŅĶņķ	cedilla[7]
ĒēĀĪŌŪāīōū	macron
ŐŰőű	hungarian umlaut

All the characters above are actively supported by TeX fonts. In addition TeX allows diacritical marks on almost all characters . Also make sure you're using the `T1`

[6] The dead macron in usually not needed, as you will use a non–dead key for this instead. For example, S-M-minus, or if `.Xmodmap` is correct, S-M-macron.

[7] These characters might not look very nice on screen, but they will be just fine when run through LaTeX and printed.

font-encoding and have the package `umlaute.sty` with the definition file `iso.def` installed.

7 Credits

The documentation is a collaborative effort between many different people (and we would encourage people to contribute !).

First, we need to give due credit to those who came before us. They gave us the base upon which the new manuals are built, and some continue to provide information:

- Matthias Ettrich wrote the original documentation, from which this manual is built, as well as the introduction to this manual [or the “LyX Manifesto,” as some of us call it].
- Lars Gullik Bjønnes wrote several minidocs, including some of the information about international support in LyX.
- Ivan Schreter also wrote a minidoc about international support, specifically about international keyboard maps and customization.
- Pascal André originally documented the LinuxDoc SGML interface.
- Alejandro Aguilar Sierra originally documented math mode and provided the entries for the math functions in `Reference.lyx`
- Special thanks to the LyX Team [1] for help and answers to questions.

Next, it’s time to give credit to the “LyX Documentation Team,” all of the people who helped rewrite the old documentation into the form it had after LyX version 0.10:

- David Johnson:
  - Contributor to the FAQ and the old “`HowDoI-.lyx`” [now defunct].
  - General editing assistance.
  - Documentation of:
    - math mode
    - tables
    - spellchecking
- Rich Fields:
  - Primary contributor to `Reference.lyx`

  - Documentation of the basic LyX interface in `UserGuide.lyx`

- Paul Evans:
  - Former maintainer of the FAQ and the old “`HowDoI-.lyx`” [now defunct].
  - Documentation of LinuxDoc in `UserGuide.lyx`
- Paul Russel:
  - Documentation of figures and imported graphics in `UserGuide.lyx`
- John Raithel:
  - Documentation of internationalization features in `UserGuide.lyx`
- Robin Socha:
  - Documentation of:
    - footnotes
    - margin notes
    - table of contents
    - cross-references
- Amir Karger
  - Primary contributor to `Tutorial.lyx`
- Matthias Zenker:
  - Documentation of
    - manual fine-tuning
    - using LaTeX from within LyX
- John Weiss:
  - General organization and format of the documents.
  - Documentation of :
    - LyX setup
    - paragraph environments, document layout, nesting, typography notes, fonts
  - Also responsible for Introduction in `Tutorial.lyx`
  - Editor of the documents. [from 6/96-fall 1997]

After fall of 1997, the LyX Team as a whole took over maintenance of the documentation.

Bibliography

[1] The LyX Team: *CREDITS*

[2] Leslie Lamport: *LaTeX: A Document Preparation System.* Addison-Wesley, second edition, 1994

[3] Michel Goossens, Frank Mittelbach and Alexander Samarin: *The LaTeX Companion.* Addison-Wesley, 1994

[4] Kopka and Daly: *A Guide to LaTeX 2ε*

[5] Donald E. Knuth. *The TeXbook*

Extended LyX Features

by the LyX Team[1]

July 17, 2006

[1] Principal maintainer of this file is Mike Ressler. If you have comments or error corrections, please send them to the LyX Documentation mailing list, <lyx-docs@lists.lyx.org>.

Contents

Chapter 1

Introduction

The *Extended LyX Features* manual, which you are now reading, is essentially Part II of the *User's Guide.* The reason for splitting this document is simple: the *User's Guide* is already huge, and it contains all of the basic features one needs to know in order to prepare most documents. However, the LyX Team has a long-term goal of making LyX extensible through various configuration files and external packages. That means that if you want to support the Fizzwizzle LaTeX package, you can create a layout file for it without having to alter LyX itself. We've already had contributions of several new features this way. This is the place where all of that gets documented.

This manual also documents some special features, like fax support, version control, and SGML support, which require additional software to work properly. Lastly, there's a chapter of LaTeX tools and tips, things you can use to spruce up your documents by directly using the powerful features of LaTeX. After all, LyX *is* only WYSIWYM, and will only ever interface to certain LaTeX features.

Of course, with all of this extra documentation, *Extended LyX Features* may itself grow too big for its britches. In that case, you can just call it the "Overextended Manual" for fun!

If you haven't read the *Introduction* yet, you are definitely in the wrong manual. The *Introduction* is the first place to go, since it will direct you to the correct manual, and it also describes the notation and format of all of the manuals. You should also be thoroughly familiar with the *User's Guide* and all of the basic features of LyX.

In this document, many sections are independent articles contributed by an individual and are noted as such. This person is generally whoever wrote the layout file for the new document class or LaTeX package, or implemented the feature. If there is no mention of an author to a chapter [or chapter sections], that means it was written by the LyX Documentation Team.

Since all the topics in this manual depend heavily on LyX's interaction with LaTeX, this first chapter covers the inner workings of LyX and how to direct LyX to generate exactly the LaTeX code you want. It is obviously for more seasoned LyX users.

Chapter 2

LyX and LaTeX

2.1 How LyX Uses LaTeX

This chapter is for both TeX-nicians and the LaTeX-curious. In it, we'll explain how LyX and LaTeX work together to produce printable output. This is the only place in any of the manuals where we assume you know something about LaTeX.

At one time, we called LyX a "WYSIWYM frontend to LaTeX," but that's no longer true. There are frontends to LaTeX out there. They are basically editors with the ability to run LaTeX and mark any errors in the file you're editing. Although LyX *is* an editor, and it *does* run LaTeX, and it also marks errors in the file, it also does much, much more. Thanks to the WYSIWYM concept, you don't need LaTeX to use LyX effectively. LyX has also added a few extensions to LaTeX. Try the following sometime: select Export ▷ LaTeX from the File menu, then look at the preamble of the resulting `.tex` file. You'll notice a variety of new macros defined specifically by LyX. These macros are defined automatically, according to the features you use in the document.

There are several commands that automatically invoke LaTeX. They are:

- View ▷ View *Format*
- View ▷ Update ▷ *Format*
- File ▷ Print
- File ▷ Fax

They will only invoke LaTeX if the file has changed since the last time LaTeX was run.

When you run LaTeX on the file you're editing, LyX performs these steps:

1. Convert the document to LaTeX and save to a file with the extension `.tex` in place of `.lyx`.

2. Run LATEX on the `.tex` file (maybe several times).
3. If there are any errors, insert error boxes in the document to mark where they are. These boxes are transient and are not saved along with the document.

If you've run LATEX using View DVI, LYX then executes `xdvi` on the Dvi file. If you've used View PostScript or Print, LYX performs two more steps:

- Run `dvips` to convert the Dvi file to PostScript®:
  - For View PostScript, the output file has the extension `.ps_tmp`
  - For Print , the output file has the extension `.ps`, as expected.
- Execute `ghostview` or send the PostScript® file to the printer.

2.2 "Help! LYX generated an unreadable `.tex` file!"

Die-hard LATEX users will scream and howl this into the night, then declare LYX useless, simply because they didn't RTFM.

We're going to set the record straight. LYX produces two kinds of LATEX files. One is human readable. The other is LYX readable. Every time LYX executes LATEX, it produces a LATEX file that it can easily scan for errors. The resulting `.tex` file is not human readable. Don't even try to read it. If you want a `.tex` file that you can send to a colleague, select Export ▷ LaTEX from the File menu.

2.3 Translating LATEX files into LYX

You can import a LATEX file into LYX by using the File ▷ Import ▷ LaTEX command in LYX. This will call a Perl script named `reLyX`—which will create a file `foo.lyx` from the file `foo.tex`—and then open that file. If the translation doesn't work, you can try calling `reLyX` from the command line, possibly using fancier options.

`reLyX` will translate most legal LATEX, but not everything. It will leave things it doesn't understand in TEX mode, so after translating a file with `reLyX`, you can look for red text and hand-edit it to look right.

`reLyX` has its own section in the *Extended Features* manual (as well as a Unix manpage equivalent), which you should read to find out about what LATEX isn't supported, bugs (and how to get around them), and how to use the various options.

If you can't get `reLyX` to work, or you just want to put a piece of LATEX code into a LYX file, see Section2.4.

2.4 Inserting LaTeX Code into LyX Documents

This is a rather important point: You can always insert LaTeX code into any LyX document. LyX simply cannot, and will probably never be able to, display every possible LaTeX construct. If ever you need to insert LaTeX commands into your LyX document, you can use the ERT box, which you can insert into your document with Insert ▷ TeX. The ERT box comes in three forms: collapsed, open, and inlined. The first two are used just like any other collapsable (foldable) box (such as footnotes), and are useful for significant amounts of LaTeX commands. An "inlined" ERT box displays its content as part of the button, and is useful for very short sections of LaTeX commands.

You can switch between all three by right-clicking on the ERT. Note that if you want more than one line of LaTeX commands, you cannot use the inlined mode.

Here's an example of inserting LaTeX commands in a LyX document. The code looks like this:

```
\begin{tabular}{ll}
\begin{minipage}{5cm}
This is an example for a minipage environment. You
can put nearly everything in it, even (non-floating)
figures and tables.
\end{minipage}
&
\begin{minipage}{5cm}
\begin{verbatim}
\begin{minipage}{5cm}
This ...
\end{minipage}
\end{verbatim}
\end{minipage}
\end{tabular}
```

The ERT box containing this text is directly after this paragraph. Those of you reading the manual online will only see a bunch of funky text in red. Those reading a printed version of the manuals will see the actual results:

This is an example for a minipage environment. You can put nearly everything in it, even (non-floating) figures and tables.	`\begin{minipage}{5cm}`
	`This ...`
	`\end{minipage}`

In addition to these two methods, you can also create a separate file containing some complex LaTeX structure. You can then use Insert ▷ Child Document to include your file (you should select the type Input). We recommend that you only do this if you have a `.tex` file which you *know* works already. Otherwise, you'll have a big job tracking down LaTeX errors…

There are a few last notes to emphasize:

- Inside of LyX, LaTeX code appears *in red*
- LyX *does not* check if your LaTeX code is correct.
- Beware reinventing the wheel.

That last note refers to two things. First, LyX does have quite a few features tucked into it, and more are coming. Be sure to check the manuals to make sure that LyX doesn't have such-and-such feature before you go off merrily coding LaTeX. Second, there are numerous LaTeX packages out there to do all sorts of things, from labels to envelopes to fancy multipage tables. Check out a CTAN site for details (see Section "Requirements" of the *User's Guide*).[1]

If you do need to do some wild and fancy things within your document, be sure to check out a good LaTeX book for assistance. There are a number of them listed in the bibliography of the *User's Guide*.

There are a number of LaTeX commands which have to be placed before the beginning of the actual text. They go into the preamble, and this is explained in the next section.

2.5 LyX and the LaTeX Preamble

2.5.1 About the LaTeX Preamble

If you already know LaTeX, there is no need to explain here what the preamble is good for. If you don't, the following will give you some ideas — we recommend again that you consult a LaTeX book for further information. In any case, you should read the points below, because they explain what you can do and what you don't need to do in the LaTeX preamble of a LyX document.

The LaTeX preamble comes at the very beginning of a document, *before* the text. It serves to:

- declare the document class. LyX already does this for you.

  If you're a seasoned LaTeX-nician, and you have some custom document class you want to use, check out the *Customization Manual* for information on how to make LyX interface to it. Be sure to submit your efforts to the LyX Team for inclusion in future versions!

- declare the usage of packages. LaTeX packages provide special commands, which are only available within a document when the package has been declared in the preamble. For example, the package `indentfirst` forces all paragraphs to be indented. There are other packages for labels, envelopes, margins, etc.

[1] Note from John Weiss: I seem to do this an awful lot. Sat down and merrily began coding something to print out labels, only to learn that there were already 2 different LaTeX packages to do this. Worse yet — I had them already!

- set counters, variables, lengths and widths. There are several LaTeX counters and variables which *must* be set globally from within the preamble in order to have the desired effect. [There are other variables which you can set and reset inside the document, too.] Margins are a good example of something which must be set in the preamble. Another example is the label format for lists. You can actually set these just about anywhere, but it's best to do it just once, inside the preamble.

- declare user defined commands [with `\newcommand` or `\renewcommand`], mostly abbreviations for LaTeX commands which appear very often inside a document. Although the preamble is a good place to declare such commands, they *can* be declared anywhere else [but *before* they are used for the first time, of course...]. This can be useful if there is a lot of raw LaTeX code in your document, which normally should not be the case.

LyX adds its own set of definitions to the preamble of the `.tex` file it produces. This makes LaTeX files generated by LyX portable.

2.5.2 Changing the Preamble

The commands which LyX adds to the preamble of a LaTeX file are fixed; you can't change them without patching LyX itself. You can, however, add your own stuff to the preamble. There are two ways to do this:

1. Select LaTeX Preamble from the Document menu, or via the Document ▷ Settings dialog, depending on your frontend. Note that the LyX keybindings will not work in this dialog, alas.

2. Use the preamble contents you've added as your default template (see "Basic LyX Setup" in the *User's Guide*), so that it will be the default preamble for any file you create.

LyX adds anything in the Preamble dialog to its own built-in preamble. Before adding your own declarations in the preamble, you should make sure that LyX doesn't already support what you want to do (remember what we said about reinventing the wheel?). Also, *make sure your preamble code is correct.* LyX doesn't check it.

2.5.3 Examples

Here are some examples of what you can add to a preamble, and what they do:

2.5.3.1 Example #1: Offsets

There are two variables under LaTeX that control page position: `\hoffset` and `\voffset`. Their names should be self-explanatory. These variables are useful if you think for a moment about computer labels. Sometimes, the size of a print medium and the area of the medium that you can actually print on aren't the same. This is where \hoffset and \voffset come in.

The default values for \hoffset and \voffset are both 0 pt., i. e. the page isn't shifted.

Unfortunately, some DVI drivers always seem to shift the page. We have no idea why, or why the sysadmin hasn't fixed such behavior. If you're using LyX on a system that you don't personally maintain, and your sysadmin is a doofus, \hoffset and \voffset can save the day. Suppose you're left and top margins are always 0.5 inches too big. You can add this to the preamble:

```
\setlength{\hoffset}{-0.5 in}
\setlength{\voffset}{-0.5 in}
```

. . . and your margins should now be correct.

2.5.3.2 Example #2: Labels

Speaking of labels, suppose you wanted to print out a bunch of address labels. There's a rather nice package, available at your nearest CTAN archive, for printing sheets of labels, called labels.sty. Now, your system may not have this package installed by default. We leave that up to you to check. You'll also want to read the documentation for it; we're not going to do that for you. Since this is an example, however, we'll give you an example of how you use this package.

First, make sure you're using the article document class. Next, you need to put the following in your preamble:

```
\usepackage{labels}
\LabelCols=3
\LabelRows=7
\LeftBorder=8mm
\RightBorder=8mm
\TopBorder=9mm
\BottomBorder=2mm
```

This sets things up for Avery® label sheets, stock #5360. You're now ready to print labels, but you'll need to insert LaTeX code, placing the commands \begin{labels} and \end{labels} around each label text. This and other special features of labels.sty are explained in its documentation.

Someday, someone may write a LyX layout file to support this package directly. Maybe that someone is you.

2.5.3.3 Example #3: Paragraph Indentation

Americans are trained to indent the first line of *every* paragraph. As with all of their other weird quirks, most Americans will whine and moan until they can have their way and indent the first line of all paragraphs.[2]

Of course, this behavior isn't standard typography. In books, you typically only indent the first line of a paragraph *if* it follows another one. The idea behind indenting the first line of a paragraph is to distinguish neighboring paragraphs from one another. If there is no previous paragraph, for example, it follows a figure, or is the first paragraph in a section, then there is no special indentation.

If you're a typical American, though, you don't care about such esoteric things; you want your indentation! Add this to the preamble:

```
\usepackage{indentfirst}
```

If your TEX distribution isn't a braindead one, you'll have this package, and all of your paragraphs will get the indentation you think they deserve.

2.5.3.4 Example #4: This Document

You can also check out the preamble of this document to get an idea of some of the advanced things you can do. You'll probably need to make the Preamble... dialog full-screen to see most of it. Also, there are more examples and an assortment of LATEX "dirty tricks" given in Chapter 7.

2.6 LYX and LATEX Errors

When LYX calls LATEX, it tells LATEX to blithely ignore any errors and keep going. It then uses the log-file from the LATEX run to do a post-mortem. As we stated earlier in the chapter, LYX generates two kinds of `.tex` files, one of which it uses to locate errors in the document. If there was an error someplace, LYX will put a box with the word "Error" at the appropriate place in the document.[3] It will also display a message alerting you to the fact that there were errors.

You can navigate through the errors by using Error in the Navigate menu. You can "open" the error-boxes and view the error message LATEX produced by clicking on it.

Some folks also like to look at the log file directly, accessible from Document ▷ LaTEX Log File. There are some fairly common error messages and warnings. We'll cover those here. You should look at a good LATEX book for a complete listing.

- "`LaTeX Warning:`"

  Anything beginning with these word is a warning message for the purpose of "debugging" the LATEX code itself. You'll get messages like this if you

[2] Note from JOHN WEISS: This was written by an American — *me*! It's my perception of my fellow countrymen. Tough if you don't like it. Thpbpbpbpbpbpbpbp!

[3] LYX will occasionally misguess where the error was. This will typically happen with tables, figures, math, and the preamble.

added or changed cross-references or bibliography entries, in which case, LATEX is trying to tell you that you need to make another run.

You can by-and-large ignore these.

- "`LaTeX Font Warning:`"

  Another warning message, this time about fonts which LATEX couldn't find. The rest of the message will often say something about a replacement font that LATEX used.

  You can safely ignore these.

- "`Overfull \hbox`"

  LATEX absolutely *loves* to spew these out. They are warning you about lines that were too long and run past the right margin. Almost always, this is unnoticeable in the final output. Or, only one or two characters extend past the margin. LATEX seems to generate at least one of these messages for just about any document you write.

  You can ignore these stupid messages. Your eyes will tell you if there's a problem with something that's too wide; just look at the output.

- "`Underfull \hbox`"

  Not quite as common as its cousin. LATEX seems to like to print lines that are a bit too wide as opposed to ones that are a bit too narrow. We have no idea why.

  You can ignore these, too.

- "`Overfull \vbox`" and "`Underfull \vbox`"

  Warnings about troubles breaking the page. Once again, just look at the output. Your eyes will tell you where something has gone wrong.

- " `LaTeX Error: File 'Xxxx' not found`"

  The file "Xxxx" isn't installed on this system. This usually appears because some package your document needs isn't installed. If you didn't touch the preamble or didn't use the `\usepackage{}` command, then one of the packages LYX tried to load is missing. Use Help ▷ LaTEX Configuration, to get a list of packages that LYX knows about. This file is updated whenever you reconfigure LYX (using Tools ▷ Reconfigure) and tells you which packages have been detected and what they do.

  If you *did* use the `\usepackage{}` command, and the package in question isn't installed, you'll need to install it yourself.

- "`LaTeX Error: Unknown option`"

  Error messages beginning with this are trying to tell you that you specified a bad or undefined option to a package. Check the package's documentation.

- "`Undefined control sequence`"

  If you've inserted LATEX code into your document, but made a typo, you'll get one of these. You may have forgotten to load a package. In any case, this error message usually means that you used an undefined command.

There are other error and warning messages. Some are self-explanatory. These are usually LATEX messages. Others are downright cryptic. These are actually TEX error messages, and we really have *no clue* what they mean or how to decipher them.

There's a general sequence you should follow if you get error messages:

1. Look at the LATEX code you inserted for typos.
2. If there are no typos, check and see that you used the command(s) correctly.
3. If you get a bunch of error boxes piled up at the very top of the document, it means that there are errors in the preamble. Start debugging your preamble.
4. If you didn't add anything to the preamble and didn't add any LATEX code to the document, the first suspect is your LATEX distribution itself. Check for missing packages and install them.
5. Okay, so there are no missing packages. Did you use any of the fine-tuning options in LYX? Specifically, did you *misuse* any of them, like trying to manually insert lots of Protected Blanks, Linebreaks, or Pagebreaks? Did you try to kludge something together with these instead of using the appropriate paragraph environment?
6. All right, you didn't use any of the fine-tuning options, you played by the rules. Did you try to pull a fancy maneuver? Did you do something funky inside a table or an equation, like inserting a graphic into a table cell?
7. Do you have long sections of text where LATEX cannot find a place to break a line? By default, LATEX is rather strict about how much extra inter-word spacing it will add in order to break a line. Preferrably, you should rework the paragraph to avoid the problem. If this isn't an option, you can wrap your text in `\sloppypar` to make LATEX's line breaking more, well, sloppy.
8. Did you go overboard with the nesting? LYX (currently) doesn't check to make sure you're in the limits for nesting environments. If you nested a bunch of environments to the 17th level, that's the problem.
9. Okay, you didn't get any error messages, but your output looks whacked. If you have a table or figure that's too wide or long for the page, you need to:

   (a) rescale the figure so it fits.

(b) trim down the table so it fits.

If something else is wrong with the output, and you didn't try to pull anything fancy or kludge the fine-tuning options, we're not sure what's wrong.

If all this doesn't help — well, then *perhaps* you might have found a bug in LYX. . .

Chapter 3

Supplemental Tools

3.1 Preparing a Bibliography with BibTEX

by Mike Ressler and Jürgen Spitzmüller

STOP! If you don't know what BibTEX is, or have a reasonably good idea of how to use it (*e.g.* setting up your own bibliographic databases), *run*, do not walk, to your nearest copy of the 2nd edition of Lamport's *LATEX: A Document Preparation System*, particularly Appendix B. The rest of this discussion assumes you have created a correct bibliography file, that you have all relevant environment variables set correctly (esp. `BIBINPUTS`, `BSTINPUTS`, and `TEXINPUTS`), and that if sufficiently desperate, you could create and "TEX" a LATEX file with a BibTEX database.

For those who don't know what BibTEX is, it is a system for creating a large database of your most used journal references. For all future articles you write, you only need to include this standard database and reference the appropriate key to each reference. Even if you write only a few papers with handful of references each, it is well worth your time to examine BibTEX and decide whether it will be useful to you.

To use BibTEX with LYX, first read the *User Guide* where it describes how to insert citations. The basic mechanism for inserting BibTEX references is the same. Then, at the very end of your document, select Insert ▷ List / TOC ▷ BibTEX Reference. In the resulting dialog, fill out the dialog boxes as follows:

Database: enter the name of your `.bib` file *without* the `.bib` extension. For searching multiple `.bib` files, just enter them in the desired order, separated by commas.

Style: enter the name of your BibTEX style file *without* the `.bst` extension. The default style is `plain` (which should be included in your LATEX distribution, so you don't have to worry about creating it).

For each citation, assuming that the source is in the `.bib` file, just call Insert ▷ Citation Reference at the correct location in the text, and enter the appropriate reference key. Nothing else is required; when invoking View ▷ DVI, for example, you should see that BibTex and LaTeX are invoked as needed, including multiple invocations of LaTeX.

3.1.1 Alternative Citation Styles

Standard BibTeX uses numbers (e. g. "[12]") to refer to a cited work. However, in many scientific disciplines, other citation styles are in use. The most common one is the author-year style (e. g. "Knuth 1984a"). LyX supports two packages that provide this style, `natbib` and `jurabib`. Both packages have their own pros and cons, which cannot be listed in detail. If you only want to have simple author-year (or author-numerical) style or if you want to use one of the countless style files for natbib, than the established `natbib` package is probably your choice. If you need special features like short title references, ibidem etc., you might consider the fairly new `jurabib` package.

The handling of both packages in LyX is basically the same. Go to Document ▷ Settings and select the Bibliography pane (with the xforms frontend: the Extras tab). Then select Natbib or Jurabib. With both packages, you will get some extra features in the citation dialog and you can select the style of the reference ("Knuth 1984", "Knuth (1984)", "Knuth, 1984", "1984" etc.). Note that both packages need specifically designed style files (they both ship their own, while there are lots of additional style files and even an interactive style file builder[1] for `natbib` available).

3.1.2 Sectionated Bibliographies

Sometimes you might need to divide your bibliography into several sections. If you are, for instance, a historian, the possibility to separate sources and scientific works is most likely a "must have". Unfortunately, BibTeX itself does not allow you to do this. The good news is, though: With the help of some LaTeX packages, BibTeX can be extended to fit your historical needs.

As of version 1.4, LyX provides native support for one of these packages, Stefan Ulrich's `bibtopic`.[2] The advantage of this package (compared to other packages like `multibib`) is that you don't need to define new citation commands. Instead, you need to prepare different bibliographic databases which include the entries for the different sections of the bibliography. For example: If you want to divide your bibliography into the sections "Sources" and "Scientific works", you first need to create two bibliographic databases, e. g. `sources.bib` and `scientific.bib`.

In LyX, go to Document ▷ Settings and select the Bibliography pane (with the xforms frontend: the Extras tab). Check Sectionated bibliography. Now you can insert multiple BibTeX references (as described in section 3.1), one for

[1] See `ftp://ctan.tug.org/tex-archive/macros/latex/contrib/custom-bib/`

[2] Available from `ftp://ctan.tug.org/tex-archive/macros/latex/contrib/bibtopic/`

each section of your bibliography. Returning to our example: Insert a BibTEX reference for the database `sources.bib` and a second one for the database `scientific.bib`. You are free to use the same or different styles for each section. Additionally, you can chose if the bibliography section should contain "all cited references" of the specified database(s) (which is the default), "all uncited references" or even "all references". This might be useful if you would like to separate your bibliography into three sections: "Cited sources", "Uncited sources", and "Scientific works". The titles for the sections can be added as ordinary sections or subsections. Since `bibtopic` removes the bibliography title, you have manually re-add that, too (as a chapter* or section*, for instance).

3.1.3 Multiple Bibliographies

Multiple bibliographies, e. g. a bibliography for each section or chapter of the document, are not supported by BibTEX itself. But the `bibtopic` package, which is used for the creation of sectionated bibliographies in LYX (cf. section 3.1.2), provides an easy way to solve this task, if you are willing to use some LATEX-Code (ERT, cf. section 2.4).[3]

First, go to Document ▷ Settings and select the Bibliography pane (with the xforms frontend: the Extras tab). Check Sectionated bibliography. In the document, you have to enclose the sections, which shall contain their own bibliography (including the BibTEX reference itself), between `\begin{btUnit}` and `\end{btUnit}` (those commands have to be inserted as ERT). The bibliography will contain all references which have been cited in the current btUnit. N. B.: If you are using this approach, then *every* citation reference has to be inside some btUnit. Also, the btUnits cannot be nested.

3.2 Making an Index

A good index is one of the hardest things to make in a lengthy document, but LYX helps make things a bit simpler by interfacing to the `makeindex` program which is found in most recent LATEX distributions.[4] Inserting an index and marking words to include in it works much the same way as preparing a bibliography as mentioned in the last section.

First, go to the end of your file and select Insert ▷ List / TOC ▷ Index List. Then, for each word you would like to include in the index, go to the end of that word and click on Insert ▷ Index Entry. This will insert a tag showing the word as it will appear in the index. That's all there is to it; LYX will automatically call `makeindex` for you and create the index itself. The text in the dialog available from right-clicking on the index button accepts LATEX, so you'll need to be careful to avoid using any special characters. On the positive side, you can use the advanced options - have a look at the documentation which

[3] An alternative approach is to use the `chapterbib` or `bibunits` package, respectively.

[4] In the Outputs ▷ LaTeX section of the preferences dialog, however, you can customize the index command, if you prefer an alternative program like `xindy`.

comes with your LaTeX distribution to find out how to do things like “nested entries”, etc.

Be careful not to put spaces between the word in the text and the index marker; apparently the wrong page number can be produced if this happens.

3.3 Multipart Documents

3.3.1 General Operation

When you are working on a large file with many sections, it is often convenient to break up the document into several files, or perhaps you have something where a table may change from time to time, but the preceding text does not. In these cases, you should seriously consider using multipart documents. For example, scientific papers often have five major sections: the introduction, observations, results, discussion, and conclusion. Each of these could be its own separate LyX file, with one “master” file which contains the title, authors, abstract, references, etc., plus the five included files. It is important to note that each of these files is a full LyX file which can be formatted and printed on its own, as well as included in a master file. Each of these files must have the same document class, however— don’t attempt to mix book classes with article classes. You may also include LaTeX files; however, these files must not have their own preamble *(i.e.* everything up to and including the `\begin{document}` line as well as the `\end{document}` line must be deleted) or else errors will be generated when you try to make a DVI file.

LyX allows you to include files quite easily with Insert ▷ Child Document. When you click on this selection a small box is inserted into the file at the current cursor location. Clicking on the box raises a dialog which allows you to select the file to be included, and the method of its inclusion.

The file selection box should by now be obvious. The three inclusion methods are “include”, “input”, and “verbatim”. The difference between “include” and “input” is really only meaningful to LaTeXperts, but the practical difference is that files which are “included” are typeset beginning on a new page, while files which are “inputted” are typeset starting on the current page. Perhaps the labeling in LyX will be changed someday to reflect this.

Generally, the master file is converted into a full LaTeX file before typesetting, while the included files are converted to LaTeX files which do not have all the preamble information. Checking the Don’t typeset button prevents this conversion.

A “verbatim” included file allows you to include a file typeset exactly as it appears in the file, i.e. verbatim mode, with the characters set in a fixed-width typewriter font. Normally, spaces in this file are invisible, though two consecutive spaces are conserved, unlike LyX’s normal treatment of spaces. However, setting the Mark spaces in output checkbox typesets a mark to unambiguously define the presence of a space.

3.3.2 Cross-References Between Files

It is possible to set up cross-references between the different files. First, open all the files in question: let's call them A and B in a two file example, where B is included in A. Let's say you insert a label in A, then want to reference it in B. Open the cross-reference dialog in whilst in document B, and you can select the "buffer" to use.

3.4 Algorithms

The package algorithm is needed by LyX to be able to output algorithm floats. These are useful in placing short algorithms across page breaks and support an index of algorithms too.

3.5 Subfigures

The package subfigure is used by LyX when you select "subfigure" in the graphics dialog and enter the subfigure caption. Several figures marked in this way can be packed into a single float with individual sub-captions.

3.6 Fancy Headers and Footers

The default page layout is rather plain; for an article document class, all you get is a centered page number at the bottom of the page. This document is the book class, so it appears to be a bit fancier, but to really put on a show, you need to set the document page style to "fancy", as mentioned in the User Guide. This section describes the LaTeX codes you need to insert in your LaTeX preamble or the text in order to get the desired effects.

The page header is divided into three fields, not surprisingly labeled "left", "center", and "right". The footer is also divided into these three fields. The LaTeX commands to set these fields in the simplest manner are `\lhead`, `\chead`, `\rhead`, `\lfoot`, etc. Suppose you wish to put your name in the upper left hand corner of each page. Simply insert the following command in the preamble:

```
\lhead{John Q. DocWriter}
```

You will now see your name in the upper left. If a field has a default entry that you would like to get rid of (often the page number appears in the central footer, simply include a command with a blank argument, e.g.

```
\cfoot{}
```

Let's get really fancy: lets put the section number with the word "Section" (e.g. Section 3) in the upper left, the page number (e.g. Page 4) in the upper right, your name in the lower left, and the date in the lower right. The following commands should now appear in the preamble:

```
\lhead{Section \thesection}
\chead{}
```

```
\rhead{Page \thepage}
\lfoot{John Q. DocWriter}
\cfoot{}
\rfoot{\today}
```

The codes `\thesection` and `\thepage` access LaTeX's section and page counters, and so print out the current section and page numbers. `\today` simply prints out today's date.

The thicknesses of the horizontal rules drawn beneath the header and above the footer can also be modified. If you don't want one of the headers, set its thickness to 0. The header rule has a default thickness of 0.4pt, the footer rule is 0pt. Use the commands, e.g. `\renewcommand{\headrulewidth}{0.4pt}` and `\renewcommand{\footrulewidth}{0.4pt}` to set the thicknesses.

You can switch the header/footer settings on and off for individual pages using commands like `\thispagestyle{empty}`, `\thispagestyle{plain}`, and `\thispagestyle{fancy}`. Simply insert them in the text on the page you want changed and mark them as TeX code. In fact, title pages are marked as plain by default, while following pages are marked fancy when using the global fancy setting.

There are more complex commands which will let you insert things in the upper left on odd numbered pages, etc., but I will refer you to the `fancyhdr` package documentation for more descriptions. For example, if you have a teTeX installation, look for `/usr/share/texmf/doc/latex/fancyhdr/fancyhdr.dvi`.

As a final example, it is possible to include an Encapsulated PostScript® file in the header or footer. Suppose you want to put a company logo in the upper lefthand corner. You might try something like

```
\lhead{\resizebox{1in}{!}{\includegraphics{logo.eps}}}
```

(you may need to preface this with `\usepackage{graphics}` if you don't include EPS files elsewhere in your document).

3.7 Minipages

LaTeX provides a mechanism to produce essentially a page within a page, called minipages. Within a minipage, all the usual rules of indentation, line wrapping, etc. apply. LyX also provides some of the minipage capability.

Minipages in LyX have their own collapsable box; insert one via Insert ▷ Minipage. Right-clicking on the box allows you to alter the minipage's width and alignment within the page. Warning: if the minipage is too long to fit on a page, it is truncated, not wrapped onto the next page.

If you place two minipages side-by-side, you can use Insert ▷ Special Character to insert a special instruction known in the LaTeX world as an `hfill` to put a maximum amount of space between them; it forces one minipage to the left edge, the other to the right edge. The examples below show the difference.

This is a minipage which does not use hfill. This is the second sentence of a minipage which does not use hfill.

This is a second minipage which does not use hfill. This is the second sentence of a second minipage which does not use hfill.

Here is some normal text to separate the two examples.

This is a minipage which does use hfill. This is the second sentence of a minipage which does use hfill.

This is a second minipage which does use hfill. This is the second sentence of a second minipage which does use hfill.

3.8 Wrapping Text Around Figures

A very frequently asked question is whether text can be made to "wrap" around figures so that a figure occupies some fraction of the column width and text fills the rest. If you have the LaTeX package `floatflt` installed (you can find out about it in the *LaTeX Configuration* manual) you can do this.

At the right is a figure of a mobius strip—you should have already seen this in the *User's Guide*. To wrap the text like this insert a wrap box via Insert ▷ Floats ▷ Floatflt Figure.

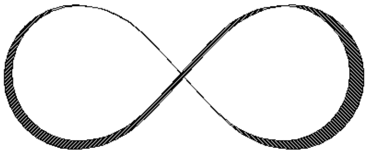


Figure 3.1: This is a wrapped figure, and this is the brilliant caption that describes it

Note: this package is very fragile! For example, having a figure too close to the bottom of the page will mess things up, as will having two figures close together. Use this package sparingly and do read the documentation that came with it (which will also tell you how to wrap text around tables).

3.9 Extra Table Options

While the standard table layout will suffice in 99% of all tables you generate, occasionally you will run into one which requires a bit of extra tweaking. The table dialog which appears on a right-click of a table allows these tweaks to be made. It will give you access to some extra column alignment parameters. A little bit of LaTeX background is useful here: when you set up a table in LaTeX, each column is given an alignment type. For example, you would give it "`l`", "`c`", or "`r`" for left-aligned, centered, and right-aligned columns respectively (which appear as the left/center/right radio buttons in LyX). A fourth type is "`p`", which will make a column of a specified width (the width box in LyX), and will wrap text within that box. A fifth type is "`|`" (vertical bar) which rather than making a column will make a vertical rule at that point; this manifests itself in LyX as the "borders" buttons. Finally, there is a type "`@`", which allows you to use whatever is enclosed in the accompanying braces as the column separator, including a null argument. The reasons for doing this may not be obvious, but they can be very powerful. They are best demonstrated by example.

3.9.1 Removing Extra Column Space

Here is a standard table:

Type	Example
Rock	Granite
Mineral	Quartz

Notice that the horizontal rule extends a bit past the text on both sides. If you wanted the line to end even with the text, we can put a null separator on the ends to get rid of the bit of extra space LaTeX adds by default. Here is the example:

Type	Example
Rock	Granite
Mineral	Quartz

In this case, the column specifier for the left column was set to "`@{} l`", while the right column was set to "`l @{}`", in order to put the null characters on the edges.

3.9.2 Changing the Column Separator Character

Now suppose you really wanted, for reasons that are completely opaque, to use $\sqrt{\pi}$ with some space around it for the column separator. Simply turn off the vertical border, then set the right column specifier to "`@{~$\sqrt{\pi}$~} l`". You could now make a table like this:

Type	$\sqrt{\pi}$	Example
Rock	$\sqrt{\pi}$	Granite
Mineral	$\sqrt{\pi}$	Quartz

3.9.3 Making a Decimal Point Aligned Column

Okay, that last example was very silly, but here is one that is not. Suppose you want to make a table that has a column which is aligned on a decimal point. A standard LaTeX trick to do this is to set the whole number part in a right-aligned column, use a decimal point for the column separator, then set the fractional part as a left-aligned column. A variation on this is to include the decimal point explicitly with the whole part, then use just a null separator in between. The latter variation is demonstrated here:

Expression	Value
π	3.1416
π^π	36.462
$(\pi^\pi)^\pi$	80663.
π^{π^π}	$1.3402{\times}10^{18}$

Though it appears a bit funny in LyX, on paper it will produce what appears to be a 2-column table in which the right column is aligned on the decimal point and the header appears to be centered over it.

Perhaps it is best if I described just what I did: first, create a 3×3 table and remove all the borders. Then re-add a bottom border to the top row, and a right border to the first column. Type in the values for the first column and set its alignment to center. Type in the `3.`, `36.`, `80663.`, and `1.` and set that column's alignment to right. Type in the `1416`, `462`, and `3402`$\times 10^{18}$ and set the extra column alignment to `@{} l`. Finally type in the word `Value` in the middle column, highlight it and the blank entry to its right, and check the Special Cell entry multicolumn. Easy, right?

3.9.4 A Better Decimal-Alignment Solution

An alternative way to have decimal alignment in tables is through the `dcolumn` package. Add the following to the LateX preamble:

```
\usepackage{dcolumn}
\newcolumntype{d}[1]{D{.}{.}{#1}}
```

To have a column decimally aligned, enter in the Special Column Alignment box of the Table dialog the following:

```
d{number of decimals of the data}
```

To create extra column space just increase the number of decimals in `d{}`. Setting the multicolumn attribute for a single cell makes it insensitive to the decimal alignment which comes in handy as well. A drawback of this method is that math mode is not allowed in a column with decimal alignment except if the multicolumn attribute is set.

This method offers the same flexibility as the `dcolumn` package. One could, for example, change the alignment separator, and have different alignment separators for different columns by defining multiple column types in the preamble. The syntax is as follows:

```
D{inputsep}{outputsep}{decimal places}
```

The interested reader is directed towards the `dcolumn` package documentation for more details.

3.10 Itemize Bullet Selection

by Allan Rae

3.10.1 Introduction

LyX provides 216 bullet shapes that can be accessed from a simple dialog. Using this dialog you can easily specify what bullet shape to use at each level of an itemized list. These settings are document-wide so you won't be able to specify different sets of bullets for different paragraphs[5].

3.10.2 How it looks

Open the dialog by selecting the Document ▷ Settings menu item and then select the Bullets tab.

The dialog provides you with a table of bullet shapes. A column of buttons on the left of the table provides access to the six different panels of bullet shapes. The row of buttons across the top is used to select which bullet depth you are changing. A text entry under the table shows the currently selected bullet shape's LaTeX equivalent and this can be edited if desired. If you do modify the text you will also need to specify any needed packages in the LaTeX preamble.

The six panels are divided up by the packages they require. The following table shows the mappings from button name to LaTeX packages.

Button	Packages Required
Standard	base LaTeX
Maths	`amssymb.sty`
Ding1	`pifont.sty`
Ding2	`pifont.sty`
Ding3	`pifont.sty`
Ding4	`pifont.sty`

LyX doesn't stop you using bullets from packages you don't have. If you get errors from LaTeX when you try to view or print the file then its likely you are missing a package. LyX doesn't restrict your use since you may be editing locally and exporting elsewhere.

3.10.3 How to use it

Select which bullet depth you want to change then select the bullet shape and size. Any changes will not be visible in LyX, but are visible when viewing the document using xdvi or ghostview.

You can reset a bullet shape to the default simply by clicking your right mouse button on the appropriate bullet depth button.

[5] Well, actually you can but you'll have to do it by hand.

If you *really* want to have multiple sets of paragraphs with different sets of bullets in each then you're going to have to get your hands dirty. The itemize bullet selection dialog can help though because it provides you with the LaTeX code for a wide range of bullet shapes. To make your own custom paragraphs you have the following options:

♯ Use the LaTeX command `\renewcommand{}{}` to specify a new bullet shape for a given depth. You'll also need to save the current bullet shape so you can restore it again afterwards. In this itemized list the following LaTeX code was used to change the bullet used for the first depth.

```
\let\savelabelitemi=\labelitemi
\renewcommand\labelitemi[0]{\small\(\sharp\)}
```

Note that the itemize depth is specified in Roman numerals as part of the `\labelitem` command.

⋆ Specify each individual entry by starting each item with the bullet shape enclosed in square brackets and set as TeX. For example, this item was started with `[\(\star\)]`.

You'll also need to revert the labelitem back to its previous setting for the global bullet shape settings to remain in effect. The way used here was:

```
\renewcommand\labelitemi[0]{\savelabelitemi}
```

Chapter 4

Special Document Classes

4.1 AMS LaTeX

by David Johnson

The AMS LaTeX layouts are set up to conform to suggested styles for mathematical papers to be submitted to American Mathematical Society publications. The layouts are not tailored to a specific journal, but easily can be. You should refer to the AMS documentation for specific instructions for each journal (usually it will entail only changing a single line in the TeX output). That documentation is available on the Web at `http://www.ams.org` or by ftp at `ftp://ftp.ams.org/pub/tex/amslatex/`.These layouts are appropriate, and useful, for any mathematical writing. There are currently 4 distinct AMS LaTeX layouts:

1. amsart: The standard AMS-article format. All results and similar statements are numbered as $(n.m)$, where the first number refers to the section, and the second refers to the total number of results (Theorems, Corollaries, Propositions, Definitions and Remarks, etc.) in that section. There are also many (but not all) environments available unnumbered, which is occasionally needed. Unnumbered environments indicated by an asterisk at the end.

2. amsart-seq: Here, numbering for each type of statement is in its own sequence, with no reference to the section number. There are also many (but not all) environments available unnumbered, which is occasionally needed. Unnumbered environments indicated by an asterisk at the end.

3. amsart-plain: This one is even more terse, since all the environments are unnumbered.

4. amsbook: the standard AMS book (really, monograph) format. Numbering is similar to the amsart layout, except that all numbering is by

($n.m.p$), where the first number refers to the chapter, the second to the section, and the third is the number of the results (Theorems, Corollaries, Propositions, Definitions and Remarks, etc.) in that section. There are also many (but not all) environments available unnumbered, which is occasionally needed. Unnumbered environments indicated by an asterisk at the end.

Any AMS LyX file can be converted to either of the numbering schemes by simply changing the document class in the Document▷Settings dialog.

4.1.1 What these layouts provide

There is a long list of included environments provided by these layouts. Most mathematical papers or books will set as special statements most of these environments, in AMS-LaTeX there is an opportunity to define an unlimited variety of such declarations. However, the AMS recommends the environments that are available in LyX. The list of environments (not counting the standard environments such as sections, bibliography, title, author, date), is:

Theorem This is typically used for the statements of major results. The word "Theorem" appears in bold type, along with an automatically-determined number (an unnumbered version, Theorem*, is also available). The text is italicized.

Corollary This is used for statements which follow fairly directly from previous statements. Again, these can be major results. Unnumbered version Corollary* is available.

Lemma These are smaller results needed to prove other statements.

Proposition These are less major results which (hopefully) add to the general theory being discussed.

Conjecture These are statements provided without justification, which the author does not know how to prove, but which seem to be true (to the author, at least).

Criterion A required condition.

Algorithm A general procedure to be used.

Axiom This is a property or statement taken as true within the system being discussed.

Definition Guess what this is for. The font, both on-screen and in the output, is different for this environment than for the previous ones. The heading ("definition") is still set in boldface, along with the number, if any, but the rest is set upright.

Example Typeset similarly to Definition.

Condition

Problem

Exercise

Remark This environment is also a new type of theorem. This is set with the word Remark in italics, and the rest upright.

Note Set similarly to the Remark environment.

Notation

Claim

Summary

Acknowledgement

Case Generally, these are used to break up long arguments, using specific instances of some condition. The numbering scheme for cases is on its own, not together with other numbered statements.

Conclusion

Fact

Proof The word "*Proof*" is set in italics, but the rest is set upright. At the end of this environment (other environments can be nested within this one, of course) a QED symbol (usually a square, but it can vary with different styles) is placed.

Address This should be the author's permanent address.

Current Address This should be the author's temporary address at the time of submission, if different from the Address.

Email Author's e-mail address

URL Author's Web address, if desired.

Keywords Key words or phrases used to identify specific topics discussed in the paper.

Subjectclass These refer to the AMS Subject Classifications, published and described in *Mathematical Reviews.* These are also available online at the AMS cites listed above.

Thanks

Dedicatory

Translator

In addition, these environments automatically provide the AMS LaTeX and AMS fonts packages. They need to be available on your system in order to use these environments.

4.2 Dinbrief

The document class dinbrief can be used to type letters according to German conventions. A template file is included in `.../lyx/share/templates` for you to use as a starting point.

4.3 Paper

The document class paper provides an alternative to the standard article class. It provides similar functionality, but you might prefer this layout with sans serif sections, headings, and more.

4.4 A&A Paper

by Peter Sütterlin

4.4.1 Introduction

This section describes how LyX can be used to write articles for submission to the scientific journal *Astronomy and Astrophysics* (www.edpsciences.fr/aa/ `http://www.edpsciences.fr/aa/`) using Version 5.01 of the document class `aa.cls`. This package can be downloaded from the ftp site

`ftp://ftp.edpsciences.org/pub/aa/readme.html`

A manual comes together with that package, and this text is not meant to replace the original manual but merely a short guide how to realize the correct form of your paper.

Please note that the publisher of the journal was changed from Springer to EDP Sciences starting January 1, 2001. That change implicated also some slight changes of the style files, namely the removal of the thesaurus command. The LyX class aa supports the newest version of these style files, V 5.01. If you have an older version installed, please upgrade. For compatibility, the old (version 4) layout has been kept as article (A&A V4). Please refer to the comments in `LyXDir/layouts/aapaper.layout`.

4.4.2 Getting started

It is recommended you start from the example template distributed with LyX. If you are not using a template, note the following settings:

- Select article (A&A) in the Document ▷ Settings dialog (OK, that one was obvious).
- Don't change the option Page style: Leave it set to default. The whole layout is done by the macros, you shouldn't change anything.

4.4.3 The header block

First thing to enter is the header information. It consists of seven entries, of which some are optional. They are

- Title: [required]
- Subtitle: [optional]
- Author: [required]
- Address: [required]
- Offprints: [optional] if more than one author: whom to contact for offprint requests.
- Mail: [optional] mail address for contacts.
- Date: [required]. Suggested format is `Received: <date>; Accepted <date>`

There is no need to issue the `\maketitle` command, this is done automatically by LyX when the header is finished. Although the order of the single header entries doesn't matter it is advised to keep the above sequence, just to get the best optics and meets the layout of the real document.

If you want to place footnotes in the header block, e.g. to state your present address, just use the standard footnote via Insert ▷ Footnote. LyX will automagically use the term `\thanks{}` in that case.

In addition to these topics, the macros use three additional LaTeX commands that have no counterpart in LyX:

- `\and` to separate different names for more than one author and institute, respectively.
- `\inst{<nr>}`to mark corresponding author/institute pairs. The institutes are numbered sequentially as they appear in the Address field, so you have to put a marker to each author.
- `\email{address}` to supply an email address for fast contact.

In all cases, the appropriate command has to be entered in LyX an marked as LaTeX code. See the examples.

4.4.4 The abstract

The abstract should immediately follow the header block. With version 5 the abstract environment was changed to a command, and there is now a resctriction to only one paragraph. In addition, it should contain an entry with the keywords. This is not yet implemented for LyX, therefore you have to enter the LaTeX command `\keywords{}` by hand and mark it as LaTeX code. Refer to the example paper.

4.4.5 Supported environments

The A&A paper layout supports the following environments for structuring your text:

- Standard
- Section
- Subsection
- Subsubsection
- Itemize
- Enumerate
- Description
- Caption
- Abstract
- Acknowledgment
- Bibliography
- LaTeX

4.4.6 Commands not supported by LyX

Some commands are not yet supported by the paper (A&A) layout for LyX. Some have already been mentioned. For the sake of completeness, they are listed all together here:

- `\and`
- `\email`
- `\appendix`
- `\authorrunning`
- `\inst{}`
- `\keywords{}`
- `\object{}`
- `\titlerunning{}`

If you want to use any of these commands, you have to enter them yourself. **Do not forget to mark them as LaTeX code!**

4.4.7 Figure and Table Floats

LyX provides support for the necessary float environments figure, figure*, table and table*, therefore we won't tell much about it here. Refer to the *User's Guide*. Just remember that tables should be left-aligned. For that, select the table and change the alignment in Edit ▷ Paragraph Settings.

There is only one special thing: the figures with caption besides the figure. To create such a figure, you have to do the following:

1. Create a wide figure float: Insert ▷ Float ▷ Figure, then right click in the figure and select Span columns.
2. Enter your caption text.
3. Press Return to move the cursor above the caption.
4. Insert your figure
5. Position the cursor behind the figure and insert a horizontal fill: Insert ▷ Special Character ▷ Horizontal Fill.
6. Switch to LaTeX mode: M-c t.
7. Enter `\parbox[b]{55mm}{`. **Do not close the brace!**
8. Position the cursor behind the caption text, switch to LaTeX mode and insert the closing brace: M-c t }.

Also, refer to the figures in the example paper.

4.4.8 Referee layout

For submission, the paper has to be formated in a special double-spacing layout. For this purpose, you have to give the option `referee` to the documentclass. This must be done using the extra class options field in the Document ▷ Settings dialog. Just enter the string `referee` there.

4.4.9 The example paper

The Examples directory contains an example paper written with LyX. It is the example paper from the original macro package, translated to LyX. Use it for inspiration, and compare the original LaTeX code with LyX way of writing.

4.5 AASTEX

by Mike Ressler

4.5.1 Introduction

AASTEX is a set of macros produced by the American Astronomical Society to facilitate electronic manuscript submission to the three journals they publish: the Astrophysical Journal (including the Letters and Supplement), the Astronomical Journal, and the Publications of the Astronomical Society of the Pacific. LyX has proven to be an excellent tool for generating these documents, especially given its equation, citation, and figure handling capabilities. LyX requires version 5.0 (or higher) of these macros; preferably 5.2, which is the version described here, or higher. Versions prior to 5.0 are intended for use with LaTeX2.09 and are fundamentally incompatible with LyX. The AASTEX package may be downloaded from the AASTEX Web site

`http://www.journals.uchicago.edu/AAS/AASTeX`

A complete user guide is contained in that package and you should familiarize yourself with it thoroughly before embarking on writing a paper in LyX. LyX will not reduce the need to figure out all the AASTEX commands, it will only reduce the drudgery of typing everything in. It is your responsibility to ensure that the final exported LaTeX document conforms completely to the requirements of the journal to which you are submitting your paper.

4.5.2 Starting a New Paper

I strongly suggest that you start with the AASTEX template file. Click on File ▷ New from Template, enter the new file name, then choose the `aastex.lyx` template. This will show the most common fields found in a manuscript. Simply overwrite the existing text (including the brackets, <>) with the correct information. Many of the AASTEX commands and environments can be implemented directly in LyX, but some cannot: most noticeably `\altaffilmark` and `\altaffiltext`, which should stick out like a sore thumb if you actually just opened the template file. For commands such as these, the LaTeX code must be entered directly and marked as such. Such commands are referred to as ERT, or Evil Red Text. I tried to minimize the amount of ERT needed in an AASTEX document, but there is still a bit more required than any of us would like.

4.5.3 Finishing Your Paper

When the paper is finished to your satisfaction and previews/prints correctly, there are a few "postprocessing" actions which need to be done before you submit it to the journals.

1. Export your paper as a LaTeX file (File ▷ Export ▷ LaTeX).
2. Edit the resulting `.tex` file with your favorite text editor
   (a) remove the comment lines before the `\documentclass` command

(b) remove the `\usepackage...{fontenc}` line if it appears (usually just after `\documentclass}`; also remove the `\secnumdepth` line if it appears.

(c) remove everything between (and including) the `\makeatletter` and `\makeatother` commands, except for any commands you specifically put into the LaTeX preamble (which should appear immediately after the "User specified LaTeX commands" comment in the `.tex` file).

3. Run the resulting file through LaTeX to make sure it still processes correctly.

4. Reread the journal requirements to make sure your filenames and formats are correct.

5. Submit it.

4.5.4 Comments On Specific Commands

I will not describe the detailed usage of the individual AASTEX commands: the AASTEX User Guide (`aasguide.tex`) gives a good description of each. Thus it's probably easiest for me to go down the list as found in the guide and offer comments where necessary. So let's begin ...

4.5.4.1 Things that work as expected

Because they work as you might expect, I simply list them and the section they are found in: `\documentclass` (2.1.1), `\begin{document}` (2.2), `\title` (2.3), `\author` (2.3), `\affil` (2.3), `\abstract` (2.4), `\keywords` (2.5), `\section` (2.7), `\subsection` (2.7), `\subsubsection` (2.7), `\paragraph` (2.7), `\facility` (2.10), `\begin{displaymath}` (2.12), `\begin{equation}` (2.12), `\begin{eqnarray}` (2.12), `\begin{mathletters}` (2.12), `\begin{thebibliography}` (2.13.1), `\bibitem` (2.13.2), all the cite commands and their variations (2.13.2), the generic graphicx figure commands (2.14.1), `\begin{table}` (2.15.4), `\begin{tabular}` (2.15.4), `\caption` (2.15.4), `\label` (2.15.4, amongst other places), `\tablerefs` (2.15.5), `\tablecomments` (2.15.5), `\url` (2.17.4), `\end{document}` (2.18).

The following style options also work correctly: `longabstract` (2.4), `preprint` (3.2.1), `preprint2` (3.2.2), `eqsecnum` (3.3), `flushrt` (3.4). Simply put them in the Options box in Layout ▷ Document.

4.5.4.2 Things that work, but require more comment

The following items work, but require a little more discussion:

- These items are reserved for use by the journal editors, but you can put them into the LaTeX preamble if you feel compelled to do so: `\received`, `\revised`, `\accepted`, `\ccc`, `\cpright` (all from 2.1.3)

- These items may be placed in the LaTeX preamble, and are included as blanks in the template file: `\slugcomment` (2.1.4), `\shorttitle` (2.1.5), `\shortauthors` (2.1.5)
- `\email` (2.3) – can only be used "standalone", not in the middle of a paragraph. Use ERT if you need to embed it.
- `\and` (2.3) – will have extra {} after it. This should not cause an error.
- `\notetoeditor` (2.6) – can only be used "standalone", not in the middle of a paragraph. Use ERT if you need to embed it.
- `\placetable` (2.8) – can't insert a cross-reference tag, you must type the tag name by hand
- `\placefigure` (2.8) – same as for `\placetable`
- `\acknowledgements` (2.9) – will have extra {} after it. This should not cause an error.
- `\appendix` (2.11) – will have extra {} after it. This should not cause an error.
- `\figcaption` (2.14.2) – you can insert an optional filename argument by placing the cursor at the beginning of the text and selecting Insert ▷ Short Title. "Short Title" inserts an optional argument of the type needed by `\figcaption`. Hopefully it will be renamed someday.
- `\objectname` (2.17.1) – same as `\figcaption` for the catalog ID optional parameter
- `\dataset` (2.17.1) – same as `\figcaption` for the catalog ID optional parameter

4.5.4.3 Things not implemented, use ERT

`\altaffilmark` (2.3), `\altaffiltext` (2.3), `\eqnum` (2.12), `\setcounter{equation}` (2.12), Journal name abbreviations (2.13.4), `\figurenum` (2.14.1), `\epsscale` (2.14.1), `\plotone` (2.14.1), `\plottwo` (2.14.1), `\tablenum` (2.15.4), `\tableline` (2.15.4, insert it as the first element in the lefthand cell after where you want it. Don't use any of LyX's rules in the table), `\tablenotemark` (2.15.5), `\tablenotetext` (2.15.5), much of Misc (2.17, except `\objectname`, `\dataset`, `\url`, and `\email`; see above), `\singlespace` (3.1), `\doublespace` (3.1), `\onecolumn` (3.2), `\twocolumn` (3.2)

4.5.4.4 Things that cannot be implemented

. . . at least in any meaningful sort of way, so I suggest ignoring them. They are the references environment (2.13.3), and the deluxetable environment (2.15). If you really, really need to use deluxetable, I suggest editing it in a separate file with a text editor, then using Insert ▷ Child Document to include it in your LyX document. See the `aas_sample.lyx` file to see an example of this.

4.5.5 FAQs, Tips, Tricks, and Other Ruminations

4.5.5.1 Getting LyX and AASTEX to cooperate

It can be a bit tricky to get LyX to recognize a new layout and document class. When all else fails, do this:

1. Make certain that LaTeX can find AASTEX. Copy sample.tex (and perhaps table.tex) from the AASTEX distribution into a directory completely unrelated to LaTeX or AASTEX and run LaTeX on `sample.tex`.
2. Make certain that `aastex.layout` appears in `/usr/.../share/lyx/layouts` or `~/.lyx/layouts`.
3. Rerun Tools ▷ Reconfigure in LyX, then restart LyX.
4. Open a regular new file, not from a template. Does AASTEX appear in the class list in Document ▷ Settings?

If you get a warning from an existing AASTEX document about not being able to find the AASTEX layout or a message about “You should not mix title layouts with normal ones”, things haven’t been installed correctly.

4.5.5.2 LaTeX error processing a table

LyX, by default, attempts to center the table caption/title. This seems to produce a bad interaction in AASTEX so you should click somewhere in the caption/title, then select Edit ▷ Paragraph Settings, then set the Alignment to Block. This took care of it for me.

4.5.5.3 References

A couple of things: 1) I have noticed some funny spacing in the reference entries in the text. When you enter the bibliography item data, make sure their is *no* space between the last author and the parenthesis setting off the year; *e.g.* type `Ressler(1992)`, not `Ressler (1992)`. 2) Entering the references at all is not obvious. The easiest thing is to start typing your first reference at the end of the document, then mark it as type References. That will put a small gray box in front of what you just typed. Click on the box to fill in the rest of the information. For new references, go to the end of an existing reference and press return. That will create a new line with its own box, etc.

4.5.5.4 Including EPS files

Even though AASTEX provides its own figure commands (`\plotone`, for example), I much prefer LATEX's standard figure commands (with the default graphicx). You can insert the `\plotone`, etc. commands as ERT into a Figure Float box if you desire, but I never have much luck getting the layout right. With the standard graphics, LYX will insert a `\usepackage{graphicx}` command into the LATEX preamble and handle the figures in the standard LATEX 2_ε way, interspersing the figures in the text. I believe ApJ accepts figures exactly this way now; AJ might still use the "stack everything at the end" technique.

4.5.5.5 Things I could have done, but didn't

There are a few "pretty" things I could have implemented, but chose not to. For instance, I saw no point in double-spacing the text in the LYX window, even though it is double-spaced in the paper manuscript. Also, I chose not to make separate layouts for the preprint and preprint2 styles. Since I assume you will spend most of your time in the plain manuscript mode anyway, I decided not to chew up more disk space with this.

4.5.6 Final Caveat

Your mileage may vary. I've now had papers published by both ApJ and AJ that have had 98% of the effort done in LYX; the last 2% was the LATEX post-processing and a few cleanups. I have had no trouble with the submission process, and I'm sure the journals were never aware that there might be a difference. So, go forth and publish!

4.6 qijmpc and ijmpd

by Panayotis Papasotiriou

4.6.1 Overview

The ijmpc package is a set of macros that facilitates electronic manuscript submission to the *International Journal of Modern Physics C.* Similarly, the ijmpd package is for creating manuscripts to be submitted to the *International Journal of Modern Physics D.* Both journals are published by World Scientific. The corresponding document classes are named `ws-ijmpc.cls` and `ws-ijmpd.cls`, respectively. These files, together with instructions for the authors, can be downloaded from the sites `http://www.worldscinet.com/ijmpc/mkt/guidelines.shtml` and `http://www.worldscinet.com/ijmpd/mkt/guidelines.shtml`. Both packages are modified versions of the standard "article" package, and they are almost (but not exactly) identical. Most of their features are supported by LYX. I have used LYX successfully to write articles submitted to both journals without any problem.

4.6.2 Writing a paper

As usual, the easiest way to write a paper is to start with a template. Click on File ▷ New from Template, then choose the `ijmpc.lyx` or `ijmpd.lyx` template. This will give an (almost) empty document that includes the most common fields found in a manuscript. Simply overwrite the existing text (including the brackets, <>) with your text. You should keep in mind the following remarks.

1. LyX won't let you change the font size and the page style of the document, because such modifications are not allowed by both packages.

2. The language of the document should not be changed. Before previewing your paper, be sure that the babel package is not used. To do this, click on Tools ▷ Preferences, select the Lang Opts tab, deselect the Use babel checkbox in the language settings, and click on Apply (or Save, if you wish to make this change permanent).

3. The "Keywords" style must be used to define keywords.

4. The ijmpc package provides a style named "Classification Codes", which can be used to define classification codes, such as PACS numbers. Note that this facility is not supported by the ijmpd package.

5. Several new environments are available: "Definition", "Step", "Example", "Remark", "Notation", "Theorem", "Proof", "Corollary", "Lemma", "Proposition", "Prop", "Question", "Claim", and "Conjecture". Their use is more or less obvious. LyX supports all these environments; it will use the proper label, text style, and numbering scheme for each of them.

6. Both packages use basic citations; the natbib package should not be used. In LyX, citation references are shown as usual; in the output, citations are shown as superscripts. If you want to use a citation as normal text, you should use the `refcite` command, e.g., "See Ref. `\refcite{key}`".

7. There is no "Acknowledgments" section in both packages. To put acknowledgments, just use the "Section*" environment.

8. Appendices may be added to the paper, *after* the Acknowledgments and *before* the References. LyX provides a special environment, called "Appendices Section" which marks the beginning of the appendices. This environment should be left blank; it just sends a LaTeX command, but nothing is really printed. In LyX, the word "Appendix" is printed with blue letters, as a signal that all sections after that point are appendices. To write an appendix, use the "Appendix" environment. LyX will number each appendix with capital letters, as required by both journals. Note that "Appendices Section" *must* be present before the first appendix; if not, all appendices will be numbered as normal sections in the output.

9. The ijmpc and the ijmpd packages use the `tbl` command to implement table captions. As a result, a table created by LyX is printed correctly, but its caption is ignored. However, you can use some ERT to overpass this problem, so that captions are printed as expected. To do so, create a float table as usual, remove the caption, and replace it with the ERT `\tbl{`*`your table caption`*`}{` (sic); you must also the ERT `}` immediately after the tabular material. Study the example table included in the template files to see how this trick is implemented. Alternatively, If you need table captions, you should implement the whole table float in a `.tex` file, then include this file to the LyX document (Insert ▷ File ▷ Child Document). Details on how to create a table float can be found in the files `ws-ijmpc.tex` and `ws-ijmpd.tex`, included in the corresponding packages.

4.6.3 Preparing a paper for submission

Before you submit your paper you must export the LyX document as a LaTeX file (File ▷ Export ▷ LateX), then make the following changes to the resulting `.tex` file.

1. Remove the comment lines before the `\documentclass` command.
2. Remove everything between (and including) the `\makeatletter` and `\makeatother` commands, except for any commands you specifically put into the LaTeX preamble.

The modified `.tex` file should be saved and processed through LaTeX as many times as necessary. You may also want to check the resulting `.dvi` document.

4.6.4 Use of ERT

The use of ERT is reduced to two commands, which must be placed at the top of the document. If you started writing your paper by using the `ijmpc.lyx` or the `ijmpd.lyx` template, the ERT needed is already in its place; you usually don't need to delete it. You may only modify the first ERT to specify the information printed to the top of odd and even pages (authors' names and short paper's title, respectively). This ERT must have the form `\markboth{Authors' Names}{Short Paper's Title}`.

4.7 Kluwer

by Panayotis Papasotiriou

4.7.1 Overview

The Kluwer package is a set of macros produced by Kluwer Academic Publishers that facilitates electronic manuscript submission to the journals they publish.

Most known of them (at least in my domain of interest) are *Astrophysics and Space Science* and *Solar Physics*, but there are many others (see a complete list at `http://www.wkap.nl/jrnllist.htm/JRNLHOME`). The Kluwer package may be downloaded from the site `http://www.wkap.nl/kaphtml.htm/STYLEFILES`. A complete user guide is contained in that package (but it can also be downloaded separately).

LyX supports many features of the package but not everything. However, the ERT needed is reduced to some "peculiar" commands of the package (see 4.7.4). I have recently used LyX to write an article submitted to the *Astrophysics and Space Science* without any problem.

4.7.2 Writing a paper

The easiest way to write a paper is to start with the Kluwer template file. Click on File ▷ New from Template, then choose the `kluwer.lyx` template. This will give an (almost) empty document that includes the most common fields found in a manuscript and a short description of their use. As in most templates, simply overwrite the existing text (including the brackets, `<>`) with the correct information.

4.7.3 Preparing a paper for submission

As in the AASTEX package, before you submit your paper to a journal you must "postprocess" it as follows.

1. Export your paper as a LaTeX file. To do this, click on File ▷ Export ▷ LateX.
2. Edit the resulting `.tex` file with a text editor and make the following changes

   (a) remove the comment lines before the `\documentclass` command,

   (b) remove everything between (and including) the `\makeatletter` and `\makeatother` commands, except for any commands you specifically put into the LaTeX preamble.

   Save the resulting `.tex` file.
3. Run the `.tex` file through LaTeX as many times as necessary (usually up to three).
4. View the resulting `.dvi` document using, e.g., xdvi, and check if everything is ok (it should, if you didn't make any mistake).

4.7.4 "Peculiarities" of the Kluwer package

The Kluwer package has the following "peculiarities".

1. It is possible to write multiple articles in the same LaTeX file[1]. Each article must be included in the environment "article". Unfortunately, this environment cannot be omitted, even if you write just one article. Therefore, each article starts with the command `\begin{article}` and, obviously, ends with the command `\end{article}`. Although this can be implemented in LyX, I didn't included it, since it looks ugly and can confuse the novice user. Therefore, you need to enter them directly and mark them as LaTeX code (the well-known "ERT").

2. Information given at the beginning of the article (i.e., title, subtitle, author, institution, running title, running author, abstract and keywords) must be included in an environment called "opening". This is not implemented in LyX, so you must enter title, subtitle etc. between two ERT lines (`\begin{opening}` and `\end{opening}`).

3. According to the user manual, the label of each bibliography item must be written as `\protect\citeauthoryear{`*author(s)*`}{`*year*`}`.

The `kluwer.lyx` template takes care of all these "peculiarities". If you start a new paper using this template you don't need to do anything special. Just

1. don't delete the ERT included in the template, and

2. copy the example bibliography item included in the template and modify it as necessary to enter new bibliography items.

4.8 Koma-Script

by Bernd Rellermeyer

4.8.1 Overview

The LyX document classes *article (koma-script)*, *report (koma-script)*, *book (koma-script)*, and *letter (koma-script)* correspond to the LaTeX document classes `scrartcl.cls`, `scrreprt.cls`, `scrbook.cls`, and `scrlettr.cls`, resp. of the Koma-Script family. They are replacements for the standard document classes `article.cls`, `report.cls`, `book.cls` and `letter.cls`, resp., and fit better to European typography conventions in a number of points.

- Standard character size is 11pt in *article (koma-script)*, *report (koma-script)*, and *book (koma-script)*, and 12pt in *letter (koma-script)*.

- Headings, labels of the description environment, and a number of elements of the *letter (koma-script)* document class are set in a bold sans serif

[1] I can't imagine any good reason to do this.

font.[2] The numbering of chapter headings is made in the same way as the numbering of section headings, that is without the extra line "Chapter…". In addition, the appearance of the headings can be modified by using a number of options (in LyX to be entered in the field Extra Options of the dialog Layout ▷ Document). A detailed German description of these options can be found in the Koma-Script documentation *scrguide*.

- The main means in the Koma-Script document classes to design the type area are the options BCOR and DIV (in LyX to be entered in the extra class options field in the dialog Document ▷ Settings). They make a clearer modification of page margins possible as do the options of the dialog Document ▷ Settings. A detailed German description of these and other type area options can be found in the Koma-Script documentation *scrguide*.
- The LaTeX document classes of the Koma-Script family define a number of additional commands. Those part of it which makes sense in LyX is implemented in corresponding paragraph types.

A detailed German description of the LaTeX document classes of the Koma-Script family can be found in the Koma-Script documentation *scrguide*.[3] The following sections describe only those aspects, which are relevant in LyX.

4.8.2 article (koma-script), report (koma-script), and book (koma-script)

The document classes *article (koma-script)*, *report (koma-script)*, and *book (koma-script)* are implemented in the layout files `scrartcl.layout`, `scrreprt.layout`, and `scrbook.layout`, resp. They contain all the paragraph types of the corresponding standard document classes *article*, *report*, and *book*, resp., partly modified, with the exception of the LyX specific List-type, which is replaced by the new Labeling-type having the same functionality. Beside the Labeling-Type there is a number of new paragraph types added. They are *not* part of *letter (koma-script)*.

- Addpart, Addchap, Addsec: are equivalents to Part*, Chapter* and Section*, resp., additionally inserting an entry in the table of contents. Addpart and Addchap are not contained in *article (koma-script)*.
- Addchap*, Addsec*: behave exactly as Addchap and Addsec, resp., additionally clearing running heads. Addchap* is not contained in *article (koma-script)*.[4]

[2] There is a big difference between the bold sans serif old cm fonts and new ec fonts, especially in the appearance of headings. In comparison, the ec bold sans serif fonts look a bit thin. Here the LaTeX package `cmsd.sty by` Walter Schmidt helps to produce the "usual" appearance when using the ec fonts.

[3] There is an English translation *screnggu*, but it is not a complete one.

[4] There is also an `\addpart*` command in *book (koma-script)* and in *report (koma-script)*, but since this is identical to Part*, is has not been implemented in LyX.

- Minisec: generates a heading directly above the following paragraph in the standard character size without affecting the structure of the document.

- Captionabove and Captionbelow are special captions which respect the different space settings needed for captions placed above or below an element (if you follow strict typographic rules, you might want to place table captions always above the table). You can also use the class option `tablecaptionsabove`, which will switch caption to captionabove for tables and captionbelow for figures. You need at least Koma-Script version 2.8q to use this.

- Dictum: can be used to set a bonmot, e. g. at the beginning of a chapter. If you use the optional argument (Insert ▷ Short Title), you can insert the dictum's author there. Dictum and author are separated by a line. You need at least Koma-Script version 2.8q to use this. Dictum is not contained in *article (koma-script)*.

The following types, together with the standard types Title, Author, and Date, form the title area of the document. They must be entered ahead of the first "ordinary" paragraph.[5] When such a type is used more than once, the latter usage overwrites the former one, that means, for every type only the latest usage is valid. The order of the different types however has, like Title, Author, and Date, no effect on the appearance of the produced document.

- Subject: produces a centered paragraph above the ordinary title (Title, Author, Date) for the subject of the document.

- Publishers: produces a centered paragraph below the ordinary title (Title, Author, Date) for the publishers' name.

- Dedication: in *report (koma-script)* and *book (koma-script)* produces a centered paragraph on its own page behind the title page, or in *article (koma-script)* produces a centered paragraph below the ordinary title (Title, Author, Date, Publishers) for a dedication.

- Titlehead: produces a left aligned paragraph above the ordinary title (Title, Author, Date, Subject) for a document‘s head.

- Uppertitleback: produces in a double-sided print in *report (koma-script)* and *book (koma-script)* a left-aligned paragraph at the top of the title page‘s back or has no effect in a single-sided print or in *article (koma-script)*.

- Lowertitleback: produces in a double-sided print in *report (koma-script)* and *book (koma-script)* a left-aligned paragraph at the bottom of the title page‘s back or has no effect in a single-sided print or in *article (koma-script)*.

[5] The corresponding LaTeX commands must appear before the `\maketitle` command.

- Extratitle: produces a special "dirty" page ahead of the actual document containing a paragraph without special formatting.

The layout files for the document classes *article (koma-script)*, *report (koma-script)*, and *book (koma-script)* do include the file `scrmacros.inc`. This is thought of as a place to define your own types. Copy `scrmacros.inc` in your personal layout directory and edit the file!

4.8.3 letter (koma-script)

The document class *letter (koma-script)* is implemented in the layout file `scrlettr.layout`. It contains all the paragraph types of the corresponding standard document class *letter*, partly modified, with the exception of the LyX specific types LyX-Code and Comment and the List type, which is replaced by the new Labeling type. In addition, it contains, in contrast to the standard document class, the standard types LaTeX, Quotation, Quote, and Verse. Furthermore, there are a number of new letter specific types.

The appearance of the letter produced by this document class can be controlled by a number of LaTeX commands, which you can put in the LaTeX preamble.[6] A detailed German description of such LaTeX commands can be found in the Koma-Script documentation *scrguide*. With it, the letter's author can produce his personal letter layout.

The types Letter and Opening define the beginning of the letter and must be used in every letter. To emphasize them in the LyX document class, they are marked with the letter *L* or *O*, resp. in the left margin. It is possible to write any number of letters in one file. An Opening type produces a new letter using the same addressee and a Letter type produces a new addressee. The types Closing, PS, CC, and Encl are ordinary paragraph types and can also be used several times in one and the same letter.

- Letter: produces a paragraph for the addressee and implicitly defines the beginning of the letter.

- Opening: produces a paragraph for the form of address and implicitly produces a new letter.

- Closing: produces a paragraph for a close.

- PS: produces a paragraph for a postscript.

[6] For example, the standard appearance of the letter‘s heading, consisting of name and address, is quite self-willed. An "ordinary" heading is produced by the following LaTeX commands in the preamble:

```
\firsthead{\parbox[b]{\textwidth}
  {\ignorespaces \fromname\\ \ignorespaces \fromaddress}}
\nexthead{\parbox[b]{\textwidth}
  {\ignorespaces \fromname \hfill \ignorespaces \pagename\ \thepage}}
```

- CC: produces a paragraph for a distribution list.
- Encl: produces a paragraph for enclosures.

The types Name, Signature, Address, Telephone, Place, Backaddress, Specialmail, Location, Title, and Subject are input types provided with a label to enter information, which will be processed by the document class.[7] The types must be used ahead of the corresponding Opening type.

An implementation of these types in a WYSIWYG fashion does not seem to make sense, because the real appearance of the produced letter does not only depend on the usage of the particular type, but also on other factors. For example, a signature entered in the Signature type will in the standard behavior appear in the produced letter only, when in the same letter also a Closing type is used. The entered value of the Telephone type will in the standard behavior not appear in the produced letter at all. The possibility to design the letter‘s heading freely is already indicated in a footnote above.

The input types can also be used as empty paragraphs. This makes sense e. g. for the Signature type. If the Signature type is not used at all, in the standard behavior the value of the Name type is used as signature, whereas if an empty Signature type is used, no signature value is defined.

By using the input types it is possible to write a letter template, containing filled input types with your personal dates (name, address, etc.) and empty input types for other dates you want to enter.

- Name: sender's name, in the standard behavior appears as a centered paragraph in small caps in the letter‘s heading.
- Signature: sender's signature, in the standard behavior appears below the Closing type. If no Signature type is used, the value of the Name type appears instead.
- Address: sender's address, in the standard behavior appears in a centered paragraph in the letter‘s heading below the sender's name.
- Telephone: sender's telephone number, in the standard behavior only sets the LaTeX variable `\telephonenum`.
- Place: place of the letter‘s making.
- Date: date of the letter‘s making. Place and Date, in the standard behavior, produce the place and the date in a right-aligned line below the addressee's field. If an empty Date type is used, neither place nor date appear, independent of the value of the Place type. If no Date type is used, the date of the letter ‘s production is used.

[7] It could be seen as a matter of inconsequence, that the types Letter and Opening described above are not such input types as well. Because of the special meaning of those types, however, I have implemented them as ordinary paragraph types with a one letter mark in the left margin. Moreover, it would affect my feeling of symmetry, if the Opening type and the Closing type had such a serious different appearance.

- Backaddress: sender‘s back address, in the standard behavior appears above the addressee’s field in a small sans serif font.
- Specialmail: special mail information, in the standard behavior appears underlined above the addressee’s field below the back address.
- Location: additional information, in the standard behavior appears on right side below the addressee‘s field.
- Title: the letter’s title, in the standard behavior appears in a big, bold, sans serif font above the subject.
- Subject: the letter’s subject, in the standard behavior appears in a bold font above the Opening paragraph.

The types Yourref, Yourmail, Myref, Customer, and Invoice produce a business letter like line above the Title line containing the fields “Your ref.”, “Your letter of”, “Our ref.”, “Customer no.”, “Invoice no.”, and “Date”. For the date field, the value of the Date type is used. If one of these “business letter types” is used, the value of the Place type however does not appear, but only the LaTeX variable `\fromplace` is set. The ordinary output of place and date in a right-aligned line below the addressee‘s field is suppressed. The types are implemented as input types provided with a label and must be used ahead of the corresponding Opening type.

- Yourref: Your ref.
- Yourmail: Your letter of.
- Myref: Our ref.
- Customer: Customer no.
- Invoice: Invoice no.

4.8.4 The new letter class: letter (koma-script v.2)

by Jürgen Spitzmüller

Koma-Script version 2.8 has introduced a new letter class `scrlttr2` which superceeds the now unsupported `scrlettr`. It has — on the LaTeX side — a completely new interface and is not compatible with the old class. Therefore, LyX supports both, though it is recommended to use the new class.

This class covers the same functionality as *letter (koma-script),* and a few more. The basic items are Address (receiver’s address, same as Letter in the old layout), Opening, and Closing. NextAddress will start a new letter (i. e. you can write several letters per document). New elements are sender’s E-Mail, URL, Fax, Bank and the possibility to use a Logo (via Insert ▷ Graphics) in the header.

The biggest improvement is, though, that the letter’s layout is configurable at almost any needs. This can be done via the preamble or with a special style file (Letter Class Option, extension `*.lco`), that will be read in as a class option.[8] Have a look at the *koma-letter2* template that is included in LyX for examples. A detailed description is to be found in the Koma-Script documentation (*scrguide*).

4.8.5 Problems

Visualizing the Koma-Script document classes in LyX, the LyX internals cause some problems.

- The chapter number of a Chapter type appears on a line of its own above the chapter heading instead of appearing in the same line ahead of it. The cause for that is the LyX internal behavior for the labeltype Counter_Chapter in the layout file.
- The headings of the types Addchap and Addsec are only put in the "true" LaTeX table of contents, but not in the LyX table of contents (Document ▷ Table of Contents).
- The paragraphs in a *letter* document class appear in a skip separation mode, not indented. This is the standard behavior, no special LaTeX commands are needed for that. But in the Document ▷ Settings dialog the corresponding radio button indicates Indent. A Skip value always has the effect that extra LaTeX commands are inserted in the document to produce the gap, which is not what is wanted in this case.

4.9 Springer Journals (svjour)

by Martin Vermeer

4.9.1 Description

These are the layout files for some of the journal formats used by Springer Verlag and listed on `http://www.springer.de/author/tex/help-journals.html`, where you should also go to fetch the class files (yes, these are LaTeX 2ε now!). It is a modular system: the things common to all journals are implemented in `svjour.inc`, which journal-specific layout files (such as, e.g., `svjog.layout` for Journal of Geodesy) can include.

This means that implementing support for any other Springer journal on this list is as simple as writing your own `sv`<myjournal>`.layout` file following the outline given in `svjog.layout`.

It is reasonably well tested only for the Journal of Geodesy. `svjour` and `svjog` come with the standard LyX distribution. Install the relevant class file

[8] The KOMA package comes with some default `*.lco` files. There is, for instance, a `DIN.lco` file that follows german typesetting rules, or a `KOMAold.lco` that provides the default layout of the old `scrlettr` class. The latter can be loaded with the class option `KOMAold`, inserted via the Layout ▷ Document ▷ Extra Options field.

(downloaded from Springer) in a proper directory, reconfigure LaTeX (in the teTeX case by running `texhash`, as root if necessary — doesn't LyX take care of this?), reconfigure LyX and it should work.

4.9.2 New styles

A large number of theorem-like styles — Claim, Conjecture, ... Theorem.

Headnote, Dedication, Subtitle, Running_LaTeX_Title, Author_Running, Institute, Mail, Offprints, Keywords, Acknowledgements, Acknowledgement. See the Springer class file documentation for details.

4.9.3 Supported journals

- *Journal of Geodesy*: `svjog.layout` — Martin Vermeer
- *Probability Theory and Related Fields*: `svprobth.layout` — Jean-Marc Lasgouttes

Add your own, it isn't so hard!

4.9.4 Credits

These files are partly based on the older `ejour2.layout`, which was again based on a tinkered-with version of an old LaTeX 2.09 style file from Springer. All this, and the `ejour2` layout, are now defunct. Jean-Marc Lasgouttes helped out big in making me find my way around the LyX layout file mechanism.

4.9.5 Bugs

Probably. But probably less than in the old hacked-LaTeX `ejour2`.

Limitations e.g.: does not display the number for theorem-like layouts, just #.

4.10 AGU journals (aguplus)

by Martin Vermeer

4.10.1 Description

These are the layout files for some of the journals of the American Geophysical Society. It is assumed that you have both the AGU's own class files and AGUplus installed (everything to be found at`ftp://ftp.agu.org/journals/latex/journals`).

4.10.2 New styles

Redefined are Paragraph, Paragraph*. They are still called this in the LyX GUI, though their LaTeX equivalents in the AGU classes are Subsubsubsection and Subsubsubsection*.

Newly defined styles are Left_Header, Right_Header, Received, Revised, Accepted, CCC, PaperId, AuthorAddr, SlugComment. These are mostly manuscript attributes and defined in the AGU class documentation.

I suspect this is still badly incomplete.

4.10.3 New floats

Planotable and Plate. We also have a new Table_Caption.

4.10.4 Supported journals

- *Journal of Geophysical Research*: `jgrga.layout` — Martin Vermeer

Add your own, it isn't so hard! Look at the `jgrga.layout` example and `aguplus.inc`.

4.10.5 Bugs and things to remember

In order to use the new layouts, you must remember to do the following for a new document:

1. *Turn off babel.* This can be done in the layout ▷ document or document ▷ settings menu item. (AGU articles are always in English, right? So *don't* choose a language.)

2. Enter `jgrga` into the document's Extra Options field. (Yes, this is a bug.)

3. Make sure you use the `agu.bst` bibliography style, by entering `agu` into the second field of the BibTeX inset. None of the standard styles will do.

4.11 EGS journals (egs)

by Martin Vermeer

4.11.1 Description

This is the layout file for the European Geophysical Society journals. The needed `egs.cls` can be downloaded from the web site of the EGS under `www.copernicus.org`.

4.11.2 New styles

Right_address, Latex_Title, Affil, Journal, msnumber, FirstAuthor, Received, Accepted, Offsets. The current layout file is unfortunately very unmodular and would benefit from using the various std*.inc file inclusions.

4.12 Slides [aka SLITEX]

by John Weiss

4.12.1 Introduction

This section describes how to use LYX to make slides for overhead projectors. There are two document classes that can do this: the default slides class and the FoilTEX slides class. This section documents the former.

I'm going to say this again, nice and clear, so that there's no misunderstanding:

This section documents the class "slides (default)" *only.*

If you're looking for the documentation for "slides (FoilTEX)", check out section 4.13. The foils class ["slides (FoilTEX)"] is actually somewhat better than the default slides class,[9] which this section documents.

This class is the LATEX 2ε improvement of the old SLITEX package. Every LATEX 2ε distribution includes this class [which I'll just refer to as "slides" from now on], so you're bound to have it. As I noted earlier, there are other classes, such as foils, which also produce slides for overhead projectors and do a better job at it. However, there are some things which slides can do which the others can't, such as generate overlays. Read on to learn more!

4.12.2 Getting Started

Obviously, to use this document class, you need to select "slides (default)" from the class list in the Document ▷ Settings dialog. There are some other special things you should know about this class:

- Don't bother changing the options Sides and Columns. They're not supported by the slides class, anyways.
- The option Page style behaves a bit differently for this class. The possible choices and what they do are as follows:

  plain The final output contains page numbers in the lower right corner.

[9] . . . or so I've been told repeatedly by its advocates. Having never used it, I have no idea if this claim is true or not.

headings Like plain, but also prints out any time markers you've put in. This is the default.

empty The final output contains no page numbers, time markers, or alignment markers.

- The slides class has an extra option: `clock`. To use it, put "`clock`" in the extra class options.

  Using this options allows you to add time markers to Notes. See section 4.12.4.3 for more details.

You can also use the template file "`slides.lyx`" to automatically set up a document to use the slides class [using File ▷ New from Template to open your new document]. The template file also contains some examples of the special paragraph environments used by this class. I'll describe those next.

4.12.3 Paragraph Environments

4.12.3.1 Supported Environments

The first thing you'll notice when you start up a new slides document is the font size and type: it's the equivalent of the size "Largest" in the Sans Serif font. This is also what's used in the output. Think of this as a "visual cue" to remind you that this is a slide. Your final slides will use a larger font; ergo, you'll have less space. Of course, the larger default screen font isn't WYSIWYG, only a reminder.

The next thing that becomes obvious is the changes to the paragraph environment pull-down box [at the far-left end of the toolbar]. Most of the paragraph environments you're used to seeing are missing. There are also five new ones. That's because the slides class itself only supports certain paragraph environments:

- Standard
- Itemize
- Enumerate
- Description
- List
- Quotation
- Quote
- Verse
- Caption
- LyX-Code

- Comment

All of the other standard environments, including the section-heading environments, aren't used in the slides class.

On the other hand, you'll notice the following new environments:

- Slide
- Overlay
- Note
- InvisibleText
- VisibleText

These five are kind of quirky, due to a "feature" in LyX. You see, LyX doesn't permit you to nest any other paragraph environment into an empty environment. Now, that's fine and dandy, but it means that you wouldn't be able to start a slide with anything except plain text. To deal with this, I've performed a little "LaTeX magic."

4.12.3.2 Quirks of the New Environments

All five of the new paragraph environments are somewhat quirky due to inherent limitiations in the current version of LyX. As I just mentioned, LyX forbids environments that begin with another environment. To get around this, the Slide environment isn't a paragraph environment as described in the *User's Guide*.

You should consider Slide, Overlay, and Note to be "pseudo-environments." They look like a section heading or a "Caption," but really begin a [and, if necessary, end the previous] paragraph environment. Likewise, treat InvisibleText and VisibleText as "pseudo-commands." These two perform some action.

A common feature of all five environments, Slide, Overlay, Note, InvisibleText and VisibleText, is a rather long-ish label. The text following this label — ordinarily the contents of the paragraph environment — is utterly irrelevant for Slide, Overlay, Note, InvisibleText and VisibleText. LyX completely ignores it. In fact, you can leave these five environments completely empty.

While you don't *have* to put any text after the rather long-ish label, you might want to. This could be a short description of the contents of the Slide, for example. In that case, enter in your descriptive comment and hit Return as you normally would.

If, on the other hand, you don't want to enter in any descriptive text, you'll hit another LyX quirk. LyX, like nature, abhors a vacuum, and will not let you start a new paragraph environment until you put something in the old one. So, do this:

- Start entering the text that will *follow* the new Slide, Overlay, Note, InvisibleText or VisibleText.

- Now move to the beginning of that paragraph.
- Next, hit Return.
- Finally, change this new, empty paragraph to a Slide, Overlay, Note, InvisibleText or VisibleText.

Some future version of LyX will, hopefully, resolve this quirkiness. . .

4.12.4 Making a Presentation with Slide, Overlay and Note

4.12.4.1 Using the Slide Environment

If you're expecting this section to teach you how to actually make a presentation, you'll be sorely disappointed. Naturally, I'll describe all of the ways the slides class can assist you in preparing the materials for a presentation. Filling in the contents, however, is up to you. [Then again, that *is* the LyX philosophy.]

Choosing the Slide environment [in the manner described in section 4.12.3.2] tells LyX to begin a new slide [duh]. The label for this environment/"pseudo-command" is an "ASCII line," in cool blue, followed by the label, "NewSlide:". Any text or paragraph environments that follow this one go on the new slide. It's that simple.

Slides are probably the only time you'll need to forcibly end pages in LyX (this can be specified in the Paragraph Layout dialog). In fact, you'll want to, once you finish entering the contents of one slide. If you've entered more text than can physically fit on a slide, the extra overflows onto a new slide. I don't recommend doing this, however, since the overflow slide won't have any page number on it. Furthermore, it may interfere with any Overlay you've made to accompany the oversized Slide.

The Overlay and Note environments work the same way as the Slide environment. They both create an "ASCII line" followed by a label ["NewOverlay:" and "NewNote:", respectively]. The color is a stunning magenta instead of blue, and the "ASCII line" will look different, in style and in length. The label fonts of all three also differ from one another.

As with a Slide, if the contents of a Note or Overlay exceed the physical size of a slide or sheet of paper, the extra will overflow onto a new sheet. Again, you should avoid this. It defeats the whole purpose of Notes and Overlays.

4.12.4.2 Using Overlay with Slide

The idea behind an Overlay is a slide that sits atop another slide. Perhaps you wish to discuss a figure on the main Slide before displaying the text associated with it. One way to accomplish this is tape a flap of dark paper over the part of the Slide you want to display later. This method fails, however, if you wish to overlap one graph with another, for example. You would then have to fumble while speaking to align the two separate, overlapping Slides to align the two graphs. The use of an Overlay environment in both cases makes life much easier.

Each Overlay receives the page number of its "parent" Slide, appended by "-a".[10] Clearly, you want the contents of both the Slide and the Overlay to each fit on a single physical slide! You should probably consider an Overlay as "part of" a Slide. Indeed, the LyX slides class provides a visual cue for this: the label at the start of an Overlay is shorter than that at the start of a Slide. Lastly, when you generate printable output, you'll find alignment markers in all four corners of both the Overlay page and its parent Slide. These will assist you in lining up the two physical slides.

The major problem in overlaying two slides is aligning the contents of the two transparencies. How much space should you leave for that graph on the second slide? Worse still, what if you want a graph and a sentence on second slide, but there is text on the main transparency that goes in between them? You could try and insert vertical space of the right size. The better way is to use InvisibleText and VisibleText.

As their names imply, InvisibleText and VisibleText are two command-like paragraph environments that make all subsequent text invisible and visible, respectively. Note from section 4.12.3.2 that you don't place anything *into* these two environments, however. When you create an InvisibleText, it inserts a centered, sky-blue label into the page reading "<Invisible Text Follows>". For paragraphs following this label, the parts of the Slide [or Overlay; it doesn't matter which] where they would be contain instead blank space.

For VisibleText, the corresponding centered label is "<Visible Text Follows>" in blazing green. Paragraphs following this label behave normally. Note that the beginning of a new Slide, Overlay, or Note automatically shuts off an InvisibleText. It's therefore not necessary to use VisibleText at the end of a Slide.

By now, it should be obvious how to create overlay transparencies using the proper combination of InvisibleText and VisibleText on a Slide and Overlay:

1. Create a Slide, including everything that will appear on it, whether on the main slide or on the Overlay.

2. Before each figure or paragraph that will appear only on the Overlay, insert an InvisibleText environment. If necessary, insert a VisibleText environment after the Overlay-only text.

3. Start an Overlay immediately following the Slide.

4. Copy the contents of this Slide into the Overlay.

5. Within the Overlay, change all of the InvisibleText lines to VisibleText and vice-versa.

That's it. You've just made an Overlay.

There's one problem with the way I've designed the LyX slides class: you can't make text in the middle of a paragraph invisible, nor make text in the

[10] Presumably, mutliple Overlays would have "-a", "-b", "-c", etc. appended to the page number of the parent Slide.

middle of an invisible paragraph visible again. To accomplish this feat, you'll need to use some inlined LaTeX codes.[11]

4.12.4.3 Using Note with Slide

Like an Overlay, a Note is associated with a "parent" Slide. Here, too, the LyX slides class provides visual cues. The label for a Note is shorter than that of a Slide [yet longer than that of an Overlay] and, like the label of an Overlay is shockingly magenta. Additionally, the printed Note has the page number of its "parent" Slide, appended by "-1", "-2", "-3", etc. You can have multiple Notes associated with a single Slide, and, as with Slide and Overlay, you'll probably want to break up long Notes so that they fit on a single sheet of paper.

The purpose of a Note is obvious: it contains anything additional you might want to say about a Slide. It could also be used as a sheet of reminders for a particular Slide. In the case of the latter, you might want to make use of time markers. Currently, the LyX slides class has no "native" support for time markers, a SliTeX feature. So, you'll have to resort to using the LaTeX codes.

To use time markers, you'll need to specify the extra class option "`clock`" [see section 4.12.2]. This option turns on timing marks, which will appear in the lower-left-hand corner of every Note you generate. To set what appears in the time marker, you use the LaTeX commands "`\settime{}`" and "`\addtime{}`". The arguments of both commands are time measured in seconds. "`\settime{}`" sets the time marker to a given time. "`\addtime{}`" increments the time marker by the specified amount. Using time markers and Notes in this fashion, you can remind yourself how much time to spend on a particular Slide.

There's one last feature to describe. Clearly, you'd like to print out all of your Slides and Overlays on transparencies while printing all of your Notes on plain paper. However, a Note *must* follow the Slide with which it is associated. What's a person to do?

Luckily, there are two LaTeX commands that allow you to select what to print out. Both must be placed into the preamble of your document. The command "`\onlyslides{\slides}`" will cause the output to contain only the Slides and Overlays. Correspondingly, the command "`\onlynotes{\notes}`" prevents the output of anything but Notes. I'd advise placing both commands in the preamble and initially comment both out. You can then preview your entire presentation as you write. When you're done writing, you can then uncomment one of the two to select what you want to print. I like to uncomment "`\onlyslides{\slides}`" , print to a file with "`-slides`" in its name, comment it back out, then uncomment "`\onlynotes{\notes}`" and print to a "`*-notes.ps`" file. I can then send

[11] The commands of interest are:

- `{\invisible ... }`
- `{\visible ... }`

. . . and need to be marked as TeX. The text whose "visibility" you wish to change goes in between the brackets [and after the `\invisible` or `\visible` command]. If you don't know how to mark text as TeX, see the apprpriate section of the *User's Guide*.

either file to a printer, loading transparencies or plain paper as appropriate.

You can also provide other arguments to the "\onlyslides{}" and "\onlynotes{}" commands. See a good LaTeX book for details.

4.12.5 The slides Class Template File

I have also provided a template file, "slides.lyx", with the slides class. To use it, begin your new presentation with File ▷ New from Template. Your new LyX presentation file will contain an example Slide – Overlay – Note triplet. The Slide and Overlay additionally contain an example of the use of InvisibleText and VisibleText. Lastly, the preamble will contain:

```
% Uncomment to print out only slides and overlays
%
%\onlyslides{\slides}

% Uncomment to print out only notes
%
%\onlynotes{\notes}
```

One final thing: I created this class to support the LaTeX 2ε "SliTeX emulation" class, one of the built-in LaTeX 2ε classes. Neither I nor the rest of the LyX Team endorse or oppose the use of this built-in slide class. It's here if you want it or need it. There exist other LaTeX 2ε classes for creating presentations, such as the Foils class [see section 4.13] or the "seminar" package [present on some TeX distributions]. The latter is not yet supported under LyX.[12] I know nothing about these other classes. Try them out to see what sort of alternative they provide.

4.13 Foils [aka FoilTeX]

by Allan Rae

4.13.1 Introduction

This section describes how to use LyX to make slides for overhead projectors. There are two document classes that can do this: the default slides class and the FoilTeX slides class. This section documents the latter.

I'm going to say this again, nice and clear, so that there's no misunderstanding:

This section documents the class "slides (FoilTeX)" *only.*

[12] Perhaps you can take on the task…

If you're looking for the documentation for "slides (default)", check out section 4.12. If your machine doesn't have the foils class ["slides (FoilTeX)"] installed, you'll probably have to use the default slides class, which isn't quite as good as foils.

The foils class is designed for use with version 2.1 of the foils.cls LaTeX class file which is now an integral part of LaTeX 2ε.

4.13.2 Getting Started

Obviously, to use this document class, you need to select "slides (FoilTeX)" from the Class entry in the Document Layout dialog. There are some settings in the Document Layout dialog that you should know about that are specific to this class:

- Don't change the options Sides and Columns on the Document Layout dialog. They're ignored by the foils class.
- The default font size is 20pt with the other options being 17pt, 25pt and 30pt.
- The default font is sans serif but all math equations are still typeset in the usual roman font.
- FoilTeX supports A4 and Letter paper sizes as well as a special size for working with 35mm slides. It doesn't support A5, B5, legal or executive paper sizes.
- Don't bother changing the Float Placement settings because they are ignored anyway. All floats appear where they are defined in the text.
- The Pagestyle setting behaves a bit differently for this class. FoilTeX provides extensive footer and header capabilities including a user-defined logo. See section 4.13.4.6 for more details. The title page is treated differently to all other pages in the document and is *always* unnumbered and *always* has the logo centered at the bottom of the page (if one is defined). The possible page style choices and what they do are as follows:

  empty The final output contains no page numbers, or other headers or footers (except footnotes of course).

  plain The final output contains page numbers centered at the bottom of the page. No other headings or footers (other than footnotes).

  foilheadings Page numbers in lower right corner. Additional headers and footers are also shown. This is also the default.

  fancy Gives you access to the fancyheadings package although its use with FoilTeX is discouraged by the writer of the FoilTeX package because of some potential page layout clashes.

4.13.2.1 Extra Options

The following options may be used in the extra class options in the Document ▷ Settings dialog.

35mmSlide This sets up the page layout for 7.33in by 11in paper, which is about the same aspect ratio as a 35mm slide, making it a bit easier to work with this medium.

headrule Places a rule across the page below the header on every page except the title page.

footrule Places a rule across the page above the footer on every page except the title page.

dvips This is automatically set each time you create a new foils document. This option tells FoilTEX to use the dvips driver to rotate those pages that are set as landscape foils.

landscape Simply changes the page dimensions to those of a landscape page but doesn't do any rotation. Thus if you use this option you need to use an external program to rotate each page or feed your paper through your printer as landscape. Note that this option effectively reverses the roles of the Foilhead and Rotatefoilhead environments (don't worry these are described in the next section).

leqno Equation numbers on the left.

fleqn Flush-left equations.

4.13.3 Supported Environments

Most of the environments commonly supported in other classes are also supported by the foils class. There are several additional environments provided by FoilTEX as well as a couple added by LyX. The following environments are shared with other classes:

- Standard
- Itemize
- Enumerate
- Description
- List
- LyX-Code
- Verse
- Quote
- Quotation
- Title
- Author
- Date
- Abstract
- Bibliography

- Address
- RightAddress
- Caption
- Comment

That is, all the major environments apart from the sectioning environments. Since foils are essentially self-contained sections, with a title and body, FoilTeX provides specific commands for starting new foils and these are:

- Foilhead
- Rotatefoilhead

LyX also provides slightly modified versions of these two environments called:

- ShortFoilhead
- ShortRotatefoilhead

and the differences will be explained in the next section.

Since foils are often used in presenting ideas or new theorems and such FoilTeX also provides a comprehensive box of goodies for presenting them:

- Theorem
- Lemma
- Corollary
- Proposition
- Definition
- Proof
- Theorem*
- Lemma*
- Corollary*
- Proposition*
- Definition*

The starred versions are unnumbered while the unstarred versions are numbered. There are also two list environments added by LyX and these are:

- TickList
- CrossList

FoilTeX provides some powerful header and footer capabilities that are best set in the preamble although they may be set at any point in a document. If you want to change these settings in your document the best place to do so is at the very top of a foil, *i.e.* straight after the foilhead.

For this purpose, the following command styles are provided [Martin Vermeer]:

- My Logo
- Restriction
- Right Footer
- Right Header
- Left Header

There are also a few commands provided by FoilTEX that aren't directly supported by LYX but I'll tell you what they do and how to use them in section 4.13.5.

4.13.4 Building a Set of Foils

This section will give a simple introduction to using the different environments to build a set of foils. If you want to see an example set of foils take a look at the `Foils.lyx` file accessible from the File ▷ Open... dialog under the Examples button.

4.13.4.1 Give It a Title Page

Unlike other classes that provide Title, Author, Date and Abstract environments, foils creates the title on a page of its own. If you leave out the Date environment LATEX will substitute the current date (every time you regenerate the output).

4.13.4.2 Start a New Foil

As I mentioned earlier, there are four ways of starting a new foil. For portrait foils you should use Foilhead or ShortFoilhead. The difference between these two environments is the amount of space between the title of the foil (the foilhead) and the body of the foil.

Landscape foils are generated using the Rotatefoilhead and ShortRotatefoilhead environments. Again the only difference is the spacing between foilhead and body. Both of the short versions have 0.5 inches less separation between the foilhead and the body.

One problem with the support for landscape foils is the requirement that you have to use the `dvips` driver to generate the PostScript® output otherwise the foils won't be rotated. It is possible to get landscape foils even if you haven't got the `dvips` driver provided you can feed your foils sideways through your printer ;-)

4.13.4.3 Theorems, Lemmas, Proofs and more

Due to a small bug in LYX you can't have two of the same type of these environments directly following each other. They must be separated by something. If you try, you will just be extending the previous environment as if you had merged the two environments together. So, how do you get around this problem? The simplest option is to insert some text between the two environments or add a LATEX environment between the two with just a "%" in it. This will force LYX to produce two separate environments and hence the correct LATEX output.

An example is provided in the example file included with the LyX distribution. Remember, this problem only occurs if you are trying to place two of the same type of theorem-like environments one directly after the other.

4.13.4.4 Lists

You get all the commonly supported list styles found in other classes as well as two new ones. I'll only describe the new ones here. If you want to find out more about the other list environments check out the *User's Guide.* If you intend to use itemized lists you might also want to read about the Itemize Bullet Selection dialog described above in section 3.10.

The two new list styles, TickList and CrossList, are designed to make it easier for you to create lists of do's and don'ts or right and wrong by providing dedicated environments that use a tick or a cross as the label of the list. These lists are in fact dedicated variants of the Itemize environment. They do however require that you have the `psnfss` packages installed.

4.13.4.5 Figures and Tables

FoilTEX redefines the floating tables and figures so that they appear exactly where they are in the text rather than pushing them to the top of the page or to some user specified location. In fact if you change the float placement settings they are simply ignored.

4.13.4.6 Page Headers and Footers

My Logo and Restriction are two commands used to control the left-footer text string. The first is meant to allow you to include a graphic logo on your foils and defaults to "-Typeset by FoilTEX-". While the second is meant to provide a classification for the audience, *e.g.* Confidential. It is empty by default.

The remaining page corners can be filled by Right Footer (which defaults to page numbers), Right Header (top right) and Left Header (top left).

4.13.5 Unsupported FoilTEX Goodies

All the commands mentioned below need to be set in a LATEX environment or as TEX within another environment.

4.13.5.1 Lengths

All lengths are adjusted using the `\setlength{`*lengthname*`}{`*newlength*`}` command. Where *lengthname* should be replaced by the name given to the length you want to change and *newlength* is the length value. All lengths should be specified in units of length such as inches (`in`), millimeters (`mm`) or points (`pt`) or relative to some document or font-based length such as `\textwidth`.

It's possible to change the spacing between a foilhead and the body of the foil by adjusting the length specified by `\foilheadskip`. For example, to make *all* foilheads 0.5 inches closer to their bodies put the following in the preamble: `\setlength{\foilheadskip}{-0.5in}`

The spacings around floats can be adjusted by setting these lengths:

`\abovefloatskip`	Separation between the text and the top of the float
`\abovecaptionskip`	Separation between the float and the caption
`\belowcaptionskip`	Separation between the caption and the following text
`\captionwidth`	You can make the captions narrower than the surrounding text by adjusting this length. Best done relative to `\textwidth`.

There are also several title page related lengths that you may find useful if you have a long title or several authors:

`\abovetitleskip`	Separation from headers to Title
`\titleauthorskip`	between Title and Author environments
`\authorauthorskip`	between multiple Author lines
`\authordateskip`	between the Author and the Date
`\dateabstractskip`	between the Date and the Abstract

The last length related command affects all the list environments. If you place `\zerolistvertdimens` *inside* a list environment then all the vertical spacing between the list items is removed. Note that this is a command not a length so it doesn't require `\setlength` like the stuff mentioned above.

4.13.5.2 Headers and Footers

The `\LogoOn` and `\LogoOff` commands control whether the logo in the MyLogo definition appear on a given page. If you put `\LogoOff` in the preamble then none of the foils will have the logo on them. If you don't want the logo on a particular page place the `\LogoOff` directly after the foilhead of that page and the `\LogoOn` directly after the next foilhead.

If you decide to use the fancy page style setting in the Document Layout dialog you should probably add `\let\headwidth\textwidth` to your preamble so headers and footers on landscape pages are correctly placed when rotated. This is due to some clashes between the page layouts provided by the fancyheadings package and the foils class.

4.14 Latex8 (IEEE Conference Papers)

by Allan Rae

4.14.1 Introduction

Since this class is specifically for writing submissions to IEEE sponsored conferences I strongly recommend that you get a copy of their Authors Kit. The latex.sty package and associated bibliography style file is included in the kit. The Authors Kit is usually sent out by email once your initial submission has been accepted. There is a lot of useful information in the Authors Kit explaining formatting restrictions and so on and I will assume you have read this since that means I don't have to repeat it all here.

4.14.2 Getting Started

[AR. more to come]

4.14.3 Supported Environments

- Standard
- Title
- Author
- E-mail
- Affiliation
- Abstract
- Section
- SubSection
- Caption

4.14.4 Differences Between Screen and Paper

There are slight differences in appearance mainly with the presentation of section counters. On screen the trailing period of the section counter is missing but it will appear in the output so don't let this worry you.

4.15 Hollywood (Hollywood spec scripts)

by Garst Reese

4.15.1 Introduction

Getting the format of a Hollywood script right is a "rite of passage." It is designed to make the readers focus on content and to be easy and familiar for the actors to read. Each page of a script should be one minute of film. Nothing goes in a script that you cannot see or hear on screen. The courier 12 pt font should be used throughout. No italics.

4.15.2 Special problems

Speakers' lines should NEVER break in mid-sentence. If a speaker's lines continue over a page break, repeat the Speaker title followed by (Cont'd).

4.15.3 Special features

Insert the Speaker names as labels then cross-reference the label to insert the name. The cross-reference dialog will show the current cast of characters. You can use this to insert the speaker name in narratives also.

4.15.4 Paper size and Margins

USLetter, left 1.6in, right 0.75in, top 0.5in, bottom 0.75in

4.15.5 Environments

The following environments are available. You can use hollywood.bind to get the bind keys shown at the right.

- Standard
  Used where nothing else works. Try to avoid it.

- FADE_IN: M-z S-I
  Usually followed by something like "on Sally waking up."

- INT: M-z i
  Introduces a new INTERIOR camera set-up. Always followed by DAY or NIGHT, or something similar to define the lighting required. Everthing on this line in CAPS.

- EXT: M-z e
  Introduces a new EXTERIOR camera set-up. Everthing on this line in CAPS.

- Speaker M-z s
  The character speaking.

- Parenthetical M-z p
  Instructions to the speaker. The () are automatically inserted, but only the (will show in LyX. Both will be printed.

- Dialogue M-z d
  What the Speaker says.
- Transition M-z t
  Camera movement instruction. e.g. CUT TO:
- FADE OUT: M-z S-I
- Author M-z S-A
- Title M-z S-T
- Right_Address M-z r

4.15.6 Script jargon

- (O.S) — off screen
- (V.0) — voice over
- b.g. — background
- C.U. — close-up
- PAN — camera movement
- INSERT — cut to close-up of

4.16 Broadway

by Garst Reese

4.16.1 Introduction

Broadway is for writing plays. The format is more decorative than Hollywood, and much less standardized. This format should be suitable for workshops.

4.16.2 Special problems

The same as in Hollywood.

4.16.3 Special features

Insert the Speaker names as labels then cross-reference the label to insert the name. The cross-reference dialog will show the current cast of characters.

4.16.4 Paper size and Margins

USLetter, left 1.6in, right 0.75in, top 0.5in, bottom 0.75in

4.16.5 Environments

The following environments are available. You can use broadway.bind to get the bind keys shown at the right.

- Standard
  You should not have to use this, but it is here for anything that does not fit otherwise.

- Narrative M-z n
  Used to describe stage setting and the action. First use of speaker names in all CAPs.

- ACT M-z a
  Automatically numbered. On screen it will be arabic, but will print as Roman.

- ACT* M-z S at
  Subtitle for ACT. It is just centered text.

- SCENE M-z S-S
  Not automatically numbered. You supply the number. This is because I couldn't figure out how.

- AT_RISE: M-z S-R
  A special case of Narrative to describe the setting and action as the curtain rises.

- Speaker M-z s
  The speaker's (actor's) title, centered in all CAPS.

- Parenthetical M-z p
  Instructions to the speaker. The parentheses are automatically inserted. The (will appear on screen, but both will be in the printed play. This environment is only used within Dialogue.

- Dialogue M-z d
  What the Speaker says.

- CURTAIN M-z S-C
  The curtain comes down.

- Title M-z S-T

- Author M-z S-A

- Right_Address M-z r

Hello there.

4.17 RevTEX4

by Amir Karger

The Revtex 4 textclass works with the American Physical Sociey's RevTEX 4.0 (the β release of May, 1999) class.

LYX has a Revtex textclass, which works with RevTEX 3.1. However, v3.1 is basically obsolete, as it works with LATEX 2.09. That means that it doesn't interact very well with LYX, which requires LATEX 2_ε, although it has been kludged to work. Since RevTEX 4.0 has been designed to work much more cleanly with LATEX 2_ε, LYX with the RevTEX 4 textclass should also be pretty easy to use.

These documents are supposed to be used in *addition* to the RevTEX 4.0 documents, so we don't describe any of the special RevTEX macros, and assume you'll know what to put in the preamble if necessary.

4.17.1 Installation

All you need to do is install RevTEX 4, as described in the package's README file. the package can be found at The RevTeX 4 Web Site `http://publish.aps.org/revtex4/`. Install it somewhere that LATEX can see it. Test it by trying to LATEX a short RevTEX 4 document in some random directory (i.e., not the directory where you installed the class file.) Then, if you reconfigure LYX, it will find the class file and let you use the RevTEX4 textclass.

Probably the easiest way to get started is either to import a RevTEX 4 document using `reLyX`, or to use the Revtex 4 template, found in the templates directory.

4.17.2 Preamble Matter

Optional arguments to `\documentclass`, like "preprint" and "aps", go in the Extra Options field in the Document Layout dialog, as usual. Remember that in RevTEX, at least one optional argument is required!

Other preamble matter, like `\draft` etc. goes in the Latex Preamble dialog, also as usual.

4.17.3 Layouts

The layouts basically correspond to the commands in RevTEX4.0. For example, the Email layout corresponds to `\email{}`. Note that (at least as of RevTEX 4.0 Beta), the Address and Affiliation layouts are exactly equivalent, so you shouldn't need to use both.[13]

[13] In case you're curious, both were included so that `reLyX` would be able to translate both `\address` and `\affiliation`.

4.17.4 Important Notes

There are a couple of important unique aspects of RevTEX 4 which might cause bugs that will be even more confusing in LYX.

In RevTEX, the `\thanks` command goes *outside* the `\author` command. The LYX equivalent is that there is a separate Thanks layout. Do *not* write footnotes in the Author layout, or weird things may happen. See the RevTEX 4 documentation for more details.

Also, the Author Email, Author URL, and Thanks layouts must be placed *in between* the Author layout and the corresponding Address (or equivalent Affiliation) layout. If you put the Thanks after the Address, the LATEX won't compile.

4.17.5 Drawbacks

The main problem with this layout is that you can't use the optional arguments to layouts like Email and Title. (The problem is not unique to this layout; you can't use optional arguments to the Section layouts either.) This means that after you export that file to LATEX (which you'll need to do eventually to send it in to APS), you'll need to edit the LATEX file with a text editor to add the optional arguments to set, e.g., the running title for the page headers. Lacking these layouts makes the `\altaffiliation` (and the equivalent `\altaddress`) useless, so the corresponding layouts don't exist, and will have to be added by hand.[14]

4.18 Article (mwart), book (mwbk) and report (mwrep)

by Tomasz Luczak

The LYX document classes *article (mwart)*, *report (mwrep)* and *book (mwbk)* correspond to the LATEX document classes `mwart.cls`, `mwrep.cls` and `mwbk.cls`, resp. They are replacements for the standard document classes `article.cls`, `report.cls` and `book.cls`, resp., and fit better to Polish typography conventions in a number of points.

Basic differences:

- Unnumbered titles (with star, eg. Section*) are added into table of contents,

- Additional page styles:

  uheadings header with separated lines,

  myheadings custom header, contents headers via commands: `\markright` and `\markboth`,

[14] *Note from JMarc:* actually, LYX 1.3.0 supports some forms of optional arguments, but this layout has not been updated yet to take advantage of it.

myuheadings custom header with separated lines,

outer page number is placed on outer side of page

- Options

rmheadings serif titles — default,

sfheadings sansserif titles,

authortitle on title page first placed is author next title — default,

titleauthor on title page first placed is title next author,

withmarginpar reserve place on page for margins.

4.19 Elsevier Journals

By Rod Pinna

Elsevier Science Publishers B.V. provides a standard LaTeX document class (`elsart.cls`) for submitting articles to their various journals. The style file can be downloaded directly from their web site: `http://authors.elsevier.com/`. Instructions are supplied along with the class file, which details the requirements of the publishers. LyX includes package that allows for the use of this class, by a layout and a template file. Installation of the class file is the same as for any other LaTeX package; instructions are provided in the Elsevier documentation.

To make use of `elsart.cls`, a file `elsart.layout` is supplied. As the Elsevier class file is based mainly on the standard article class, most of the normal functionality is provided. The Elsevier class defines a number of mathematical environments, which are similar to the AMS environments. These commands are all described in the Elsevier documentation, and are available in LyX.

The easiest way to use the Elsevier style is to base documents on the included template file. It is best not to use options such as fancy headings or the geometry package, as elements such as these are defined by Elsevier in their style file. Ideally, no extra packages except those mentioned in the Elsevier documentation should be used. Essentially, Elsevier require as "clean" a LaTeXfile as possible, as their intention is to take the supplied file and replace the class file with one for the particular journal to which the paper has been submitted. This also means that not too much time should be spent on the formating of the document. When it comes to be published, this will change anyway. The rest of the usage for this layout is substantially the same as for the normal article class. For details of what Elsevier do and don't allow, refer to their documentation.

4.20 Memoir

By Jürgen Spitzmüller

4.20.1 Overview

Memoir is a very powerful and constantly evolving class. It has been designed with regard to fictional and non-fictional literature. Its aim is to let the user have maximum control over the typesetting of his document. Memoir is based on the standard book class, but it can also emulate the article class (see below).

Peter Wilson, the developer of Memoir, is known as the author of lots of useful packages in the LaTeX world. Most of them have been merged with Memoir. Therefore, it is much easier to layout the table of contents, appendices, chapter designs and such. LyX, though, does not support all of these goodies natively. Some of them might be added to forthcoming releases[15], lots will probably never, due to the limitations of LyX's framework. Of course you can still use all features with the help of some native LaTeX commands (ERT[16]). In this section, we can only list those features which are natively supported by LyX. For detailed descriptions (and for the rest of features) we are recommending to have a look at the detailed manual of the Memoir class[17], which is not only a user guide for the class, but also both a comprehensive description on good typesetting and a superb example for good typesetting itself.

4.20.2 Basic features and restrictions

Memoir supports basically all features of the standard book classes. There are, however, some differences, as follows:

Font sizes: Memoir has a broader range of font sizes: 9, 10, 11, 12, 14, 17

Page style: The fancy page style is not supported, due to a command clash between Memoir and the fancyhdr package (they are both defining a command with the same name, which confuses LaTeX). Instead, Memoir comes with a bunch of own page styles (see Layout ▷ Document ▷ Page Style). If you want to use these for the chapter pages, you have to use the command `\chapterstyle` in the main text or in preamble (e. g. `\chapterstyle{companion}`).

Sectioning: Sectionings (chapter, section, subsection etc.) are coming with an optional argument in the standard classes. With this, you can specify an alternative version of the title for the table of contents and the headers (for instance, if the title is too long). In LyX, you can do this via Insert ▷ Short Title at the beginning of a chapter/section. Memoir features a second optional argument and thus separates the table of contents from the header. You can define three variants of a title with this: one for the main text, one for the table of contents, and one for the headers. Simply insert two optional arguments if you need this feature, the first one containing the short title for the Table of Contents, the second one containing an alternative short title for the headers.

[15] You are invited to send suggestions to `lyx-devel@lists.lyx.org`.

[16] Cf. section 2.4 for details.

[17] Cf. `CTAN:/macros/latex/memoir/memman.pdf`.

TOC/LOT/LOF: In the standard classes (and in many other classes), the table of contents, the list of figures and the list of table start a new page automatically. Memoir does not follow this route. You have to insert a page break yourself, if you want to have one.

Titlepage: For some unknown reason, Memoir uses pagination on the title page (in the standard classes, title pages are "empty", i. e. without pagina). If you want an empty title page, type `\aliaspagestyle{title}{empty}` in the preamble.

Article: With the class option *article* (to be inserted in Layout ▷ Document ▷ Extra Options), you can emulate article style. That is, counters (footnotes, figures, tables etc.) will not be reset on new chapters, chapters don't start a new page (but are—in contrary to "real" article classes—still allowed), parts, though, use their own page, as in book.

Oldfontcommands: By default, Memoir does not allow the use of the deprecated font commands, which have been used in the old LaTeX version 2.09 (e. g. `\rm`, `\it`). It produces an error and stops LaTeX whenever such a command appears. The class option *oldfontcommands* reallows the commands and spits out warnings instead (which does at least not stop LaTeX). Since a lot of packages and particularly BibTeX style files are still using those commands, we have decided to use this option by default.

4.20.3 Extra features

We will only describe the features supported by LyX (which is not much currently). Please consult the Memoir manual[18] for details.

Abstract: You may wonder why an abstract is an extra feature. Well, it is in book class. Usually books don't have abstracts. Memoir, however, has. You can use it whereever and how often you like.

Chapterprecis: You may know this from belletristic: The contents of a chapter is shortly described below the title and also in the table of contents (e. g. *Our hero arrives in Troia; he loses some friends; he finds others*). Chapterprecis does exactly this. It is therefore only sensible below a chapter.

Epigraph: An epigraph is a smart slogan or motto at the beginning of a chapter. The epigraph environment provides an elegant way of typesetting such a motto. The motto itself (text) and its author (source) are divided by a short line. Unfortunately, we have to fool LyX a bit here again, since the environment needs two arguments (text and source). In this case, we have to use curly brackets (in TeX mode) between the two arguments: *<smart slogan>* `}{` *<author of the slogan>*.

[18] Cf. `CTAN:/macros/latex/memoir/memman.pdf`.

Poemtitle: Memoir has lots of possibilities to typeset poetry (up to very complex figurative poems). Lyx can only support a few of them. One is poemtitle, which is a centered title for poems, which will also be added to the table of contents (verse is the standard environment for poems. Memoir has some enhanced versions of verse, but you need to use ERT, because they have to be nested inside regular verse environments, which is not possible with LyX).

Poemtitle*: Same as poemtitle, but it adds no entry to the table of contents.

Chapter 5

Importing and Exporting Alternate File Formats

5.1 Considerations

Importing and exporting LyX documents from/to other formats has been touched on briefly in the *User Guide*. Here we describe more of the gory details needed to understand just what is going on when you click on the File ▷ Import and File ▷ Export menu items.

5.2 Importing Other Formats

5.2.1 LaTeX

Translating from LaTeX into LyX is performed by a Perl script called reLyX. Although it is a standalone program which can be called from the command line, LyX will call it automatically when a LaTeX document is imported. See section 5.4 for a complete description. There are no user tunable parameters for reLyX within LyX.

5.2.2 ASCII Text

When importing plain ASCII text, there are two methods of reading the file. Importing "as lines" preserves all the linebreaks in the ASCII; to LyX, then, each line looks like a paragraph. Importing "as paragraphs" assumes that consecutive lines separated by only a single linebreak form a single paragraph. Successive linebreaks with no intervening text are thus assumed to be paragraph delimiters.

5.2.3 Noweb

[Editor's note: Needs to be written, obviously - any volunteers? — mer]

5.3 Exporting Other Formats

5.3.1 LaTeX

LyX generates two types of LaTeX files: stripped down versions for the normal processing (View DVI, etc.) which one normally never sees[1], and human readable forms which are suitable for exchanging with your colleagues. The only settable option for the translation is the line length of the output file. The default is 65 characters, but it can be set in Tools ▷ Preferences using the Ascii line length field.

5.3.2 Device Independent Files

Device Independent files (DVI files) are produced by running LaTeX on your document. There are no user settable options.

5.3.3 PostScript®

The next step in the conversion chain is converting a DVI file into Postscript®. You can either use File ▷ Export ▷ Postscript or, if you need more control on the result, File ▷ Print. If you use the later, note that it is possible to configure, in Tools ▷ Preferences, the options passed to the dvips program to achieve different effects.

5.3.4 ASCII text

Exporting as ASCII attempts to preserve the "shape" of the document as well as possible, but things like centering and indentation are thrown out; paragraphs are separated by blank lines. Section numbering and cross-references are done correctly, so the resulting text files is remarkably readable. The only changeable option is the length of lines, as for LaTeX output.

5.3.5 HTML

LyX documents can be converted to hypertext markup, usually by converting to LaTeX first, then converting that to HTML. Four LaTeX→HTML converters are currently known to LyX: `tth`, `latex2html`, `hevea` and `htlatex`. Though they are autodetected, you can overide the selection in preferences. You can also include further command line options in this dialog.

5.3.6 PDF

by Dekel Tsur (mostly)

[1] The resulting file is a perfectly valid LaTeX file, though the preamble might look a bit strange since it includes some definitions used by LyX which wouldn't show up in most human-written files.

The fastest way to generate a basic PDF file (no tags, links, etc.) with any version of LyX is to save the document as a Postscript® file, then run the `ps2pdf` command on it. Starting with version 1.1.6, the menu item File->Export->PDF will do all this for you. There are some issues with fonts that you need to pay attention to: see Section 5.3.6.2. Also, as of version 1.1.6, there is a better method that will generate much more sophisticated files.

5.3.6.1 Use pdfLaTeX

With pdfLaTeX you need to convert your eps figures to PDF (see Section **??**), and you cannot use pstricks. On the other hand, with pdfLaTeX it is possible to insert directly images in JPEG or PNG format, use TrueType fonts, and more.

5.3.6.2 Why does the text look so bad when viewed with Acrobat Reader?

The problem is that bitmap fonts are displayed poorly by Acrobat Reader. When creating a PDF from the LyX file, you need to use outline font instead of the default bitmap fonts (in fact, you should also use outline fonts for Postscript files). Recent LaTeX distributions come with Postscript® Type 1 version of the standard (Computer Modern) fonts. pdfLaTeX uses these font by default. Dvips doesn't use these fonts by default, so to make it use them, add the following to lines to your `~/.dvipsrc` file

```
p+ psfonts.cmz
p+ psfonts.amz
```

If the default LaTeX font encoding (OT1) is used, nothing else need to be done. However, if the T1 font encoding is used, then LaTeX uses the newer EC fonts, for which there are no Type1 version. The solution is to use the ae package which emulates T1 coded fonts using the standard CM fonts. This is done by adding `\usepackage{ae,aecompl}` to the preamble of the LyX file. However, some glyphs are missing from the CM fonts (e.g. eth, thorn), and they are taken from the EC fonts. Therefore you get these glyphs as bitmaps.

Note: LyX uses by default the T1 font encoding. If you wish to use the default font encoding (this is not recommended, unless you only write English documents), clear the field TeX encoding in preferences (tabs Outputs, Misc).

An alternate option is to use the standard Postscript® fonts instead of the Computer Modern fonts. To do that, you need to select pslatex as the global font in the document layout dialog. When using the Postscript® fonts, the result PDF file is smaller as the fonts are not saved into the file. Furthermore, the Postscript® fonts include all T1 glyphs. On the other hand, the Postscript® fonts have no bold symbol font, so poor man's bold must be used (see Section 5.3.6.3). The Postscript® fonts also look different from the Computer Modern fonts.

To sum up, both the Computer Modern and the Postscript® fonts gives good results (with few exceptions). The decision of which one to use is a matter of taste.

5.3.6.3 Why doesn't the \boldsymbol{} command work when I use pslatex?

The Postscript® fonts do not have a bold symbol font. The solution is to use the `\pmb{}` (poor man's bold) command.

It is possible to redefine the `\boldsymbol` command to use `\pmb` by putting

```
\renewcommand{\boldsymbol}[1]{\pmb{#1}}
```

in the preamble.

5.3.6.4 Is it possible to do write latex code which is processed only when running pdfLaTeX?

Yes. Here is an example:

```
\newif \ifpdf
   \ifx \pdfoutput \undefined
      \pdffalse
   \else
      \pdftrue
\fi
\ifpdf
   \pdfinfo { /Author (your name and e-mail address)
      /Title (official title -- i.e., title element)
      /Subject (one line description of the document)
   }
   \pdfcatalog { /PageMode (/UseNone)
   % /OpenAction (fitbh)
   }
   \usepackage[pdftex]{hyperref}
\else
   \usepackage[ps2pdf]{hyperref}
\fi
```

5.3.6.5 How can I make URLs clickable ?

See the references here :
`http://wiki.lyx.org/pmwiki.php/FAQ/PDF`

5.3.7 Custom

Custom exports are possible if you have some particularly weird format you wish to convert to, assuming you have the relevant converter, of course. The format of the *input* file can be chosen in the File ▷ Export ▷ Custom dialog; LyX will automatically convert the file to this point, then feed it to your custom converter. The possible values are all formats that LyX can produce from its own documents.

The converter command is also specified in the dialog.It should be a completely qualified command line which uses the variable `$$FName` to specify the name of the file. If this variable is not given, then the file will be sent to the standard input of your command. You may have to apply a bit of ingenuity to escape this sequence correctly so that it is compatible with your shell.

While it is not possible to save this command using the Preferences dialog, you can manually edit your `.lyx/preferences` to add a line like

```
\custom_export_command "mycommand $$FName"
```

5.4 The Complete reLyX Description

5.4.1 Synopsis

The simplest way to use reLyX is via the File ▷ Import command in LyX. That runs reLyX on the given file and loads the resulting file into LyX. You should try that first, and call it from the command line only if you need to use more complicated options.

reLyX [**-c** *textclass*] [**-df**] [**-o** *outputdir*] [**-r** *renv1*[,*renv2...*]] [**-s** *sfile1*[,*sfile2...*]] *inputfile*

reLyX -p -c *textclass* [**-df**] [**-o** *outputdir*] [**-r** *renv1*[,*renv2...*]] [**-s** *sfile1*[,*sfile2...*]] *inputfiles*

reLyX -h

5.4.2 Options

-c Class. By default, when reLyX sees a `\documentclass{foo}` command, it creates a file of textclass "foo" and reads the LyX layout file for that class. Use **-c** to declare a different textclass (and read a different layout file).

-d Debug. By default, reLyX gives sparse output and deletes the temporary files which were created during translation. Using the **-d** flag will create much more output (both to stdout and stderr) and leave the temporary files around.

-f Force. reLyX will not run if the `.lyx` file it would generate already exists Use the **-f** option (carefully) to clobber any existing files.

-h Help. Print out usage information and quit

-o Output directory. With this option, all temporary files and LyX output files (for the given input file, for any included files, or for any file fragments given with the **-p** option) will be put into *outputdir*. Otherwise, for each file *dir/foo.tex*, the temporary files and the LyX output file will be created in *dir*. This can be useful if a file includes files from other directories which you want to consolidate in one directory, or if you don't have write permission on the directory the LaTeX files are in.

-p Partial file. The input files are LaTeX fragments, with no preamble matter or `\begin{document}` commands. This option requires the **-c** option, since there are no `\documentclass` commands in the files reLyX is translating. When using this option, you can translate more than one file, as long as all files are the same class. The LyX file created by reLyX can be included in an existing LyX file using Insert ▷ File ▷ LyX Document.

-r Regular environments (see the Section 5.4.5.4). If you give more than one environment, separate them with commas (not spaces). You'll probably need to quote the environment list, especially if it has asterisk environments (foo*) in it. If you use this command often, considering creating a personal syntax file.

-s Syntax files. Input (one or more quoted, comma-separated) syntax files to read in addition to the default. (see the section Section 5.4.5.4 for details).

5.4.3 Description

5.4.3.1 Introduction

reLyX will create a LyX file *dir/foo.lyx* from the LaTeX file *dir/foo.tex* (unless the **-o** option is used).

Suffixes `.tex`, `.ltx` and `.latex` are supported. If *inputfile* does not exist and does not have one of these suffixes, reLyX will try to translate *inputfile.tex*. (This is similar to the behavior of LaTeX.)

The purpose of reLyX is to translate *well-behaved* LaTeX 2$_\varepsilon$ into LyX. If your LaTeX file doesn't compile—or if you do weird things, like redefining standard LaTeX commands—it may choke. LaTeX209 will often be translated correctly, but it's not guaranteed.

reLyX has some bugs and lacks a few features. However, its main goals are:

- Get through a well-behaved LaTeX 2$_\varepsilon$ file without crashing
- Translate a lot of that file.
- Localize the parts that can't be translated and copy them in TeX mode

It achieves these main goals pretty well on most files.

There are many improvements that can and will be made to reLyX in the future. However, we wanted to get reLyX out there early on, to make it easier for new LyX users to read in their existing LaTeX files.

5.4.3.2 Usage

Here's a more lengthy description of what you should do to translate a LaTeX document into LyX.

- Run reLyX.

  reLyX will inform you of its progress and give any warnings to stderr, so if you don't want any output at all, try (in csh) "`reLyX foo.tex >& /dev/null`" or (in bash) "`reLyX foo.tex 2>&1 >/dev/null`". You should NOT redirect standard output to `foo.lyx`.

- Run LyX on the resulting .lyx file.

  In theory, most of the file will have been translated, and anything that's untranslatable will be highlighted in red (TeX mode). In theory, LyX will be able to read in the file, and to create printed documents from it, because all that untranslated red stuff will be passed directly back to LaTeX, which LyX uses as a backend. Unfortunately, reality doesn't always reflect theory. If reLyX crashes, or LyX cannot read the generated LyX file, see Section 5.4.3.5 or the `BUGS` file.

- Change things that are in ERT boxes (TeX code) by hand in LyX.

  As mentioned above, you should be able to print out the LyX file even without doing this. However, changing a command in TeX mode to the corresponding LyX object will allow you to take advantage of LyX's WYSIWYM editing.

  reLyX is not guaranteed to create a LyX file which generates exactly the same output as the LaTeX file, but it should come close. reLyX will generally err on the side of translating less to ensure that dvi or ps files are accurate, even though this leads to more "evil red text" and less WYSIWYM.

- PROOFREAD THE DOCUMENT!!

  I'm sure you were planning on doing this anyway, but it's particularly important after translating a LaTeX document. reLyX is, at least now, better at "macro-translating" (translating the whole document) than "micro-translating" (translating every little detail). For example, you may see extra spaces or deleted spaces. Space handling has improved, but it's not perfect.

5.4.3.3 What reLyX Can Handle

reLyX understands many LaTeX commands. It will translate:

- regular text, including mini-commands like ~, ", `\@`, `\TeX`, as well as accented characters like `\'{a}`, and the special cases ¿ and ¡

- title commands like `\author`, `\date`, `\title`, `\thanks` and the abstract environment
- heading commands like `\section` including starred commands (`\section*`)
- Environments: `quote`, `quotation`, and `verse`; `center`, `flushright`, and `flushleft`
- `itemize`, `enumerate`, and `description` environments, and their `\item` commands. Also, well-behaved nested lists
- cross-referencing commands: `\ref`, `\pageref`, `\label`, and `\cite`
- `\footnote` and `\margin`
- font-changing commands including `\em`, `\emph`, `\textit`, and corresponding commands to change family, size, series, and shape
- `\input{foo}` (or `\input{foo.blah}`) and `\include{foo}`. Plain TeX `\input` command ”`\input foo.tex`” is also supported.
- `tabular` environment, and commands that go inside it like `\hline`, `\cline`, and `\multicolumn` (but see below)
- float environments `table` and `table*`, as well as `\caption` commands within them
- `thebibliography` environment and `\bibitem` command, as well as BibTeX's `\bibliography` and `\bibliographystyle` commands
- miscellaneous commands: `\hfill`, `\\`, `\noindent`, `\ldots`...
- documentclass-specific environments (and some commands) which can be translated to LyX layouts
- arguments to certain untranslatable commands (e.g. `\mbox`)

Some of this support may not be 100% yet. See below for details

reLyX copies math (almost) verbatim from your LaTeX file. Luckily, LyX reads in LaTeX math, so (almost) any math which is supported by LyX should work just fine. A few math commands which are not supported by LyX will be replaced with their equivalents, e.g., `\to` is converted to `\rightarrow`. See the section on *Syntax Files* for more details.

reLyX will also copy any preamble commands (i.e., anything before `\begin{document}`) verbatim, so fancy stuff you've got in your preamble should be conserved in dvi and printed documents, although it will not of course show up in the LyX window. Check the preamble to make sure.

5.4.3.4 What reLyX Can't Handle — But it's OK

- figures and `tabular*` tables
- minipages
- spacing commands (`\vspace`, `\pagebreak`, `\par`)
- `\centering`, `\raggedleft`, `\raggedright`
- `\verb` and `verbatim` environment. reLyX is careful to copy *exactly* in this case, including comments and whitespace.
- some unknown (e.g., user-defined) environments and commands

reLyX copies unknown commands, along with their arguments, verbatim into the LyX file. Also, if it sees a `\begin{foo}` where it doesn't recognize the "foo" environment, it will copy verbatim until it sees `\end{foo}` (unless you use the `-r` option). Hopefully, then, most of these unknown commands won't cause reLyX to break; they'll merely require you to do some editing once you've loaded the file up in LyX. That should be less painful than editing either the `.tex` or the `.lyx` file using a text editor.

5.4.3.5 What reLyX Handles Badly — a. k. a. BUGS

Since reLyX is relatively new, it's got a number of problems. As it matures, these bugs will be squished. A number of bugs and missing features can be found listed on the LyX bug tracker, LyX Bugzilla `http://bugzilla.lyx.org/`.

If reLyX is choking on something, or LyX can't read it after reLyX translates it, the best thing to do is to put `\begin{reLyXskip}` before the offending text, and `\end{reLyXskip}` after it. I call this a "skip" block. reLyX will copy this block exactly, in TEX mode. Then edit the resulting LyX file, and translate the unknown stuff by hand. The `reLyXskip` environment is magical; the `\begin` and `\end` commands will not be put into the LyX file.

- "Exact" copying of unknown environments and commands isn't quite exact. Specifically, newlines and comments may be lost. This will yield ugly LyX, but in almost all cases the output will be the same. However, certain parts of the file will be copied perfectly, including whitespace and comments. This includes: the LaTeX preamble, `verbatim` environments and `\verb` commands, and skip blocks.
- reLyX translates only a few options to the `\documentclass` command. (Specifically 1[012]pt, [letter|legal|executive|a4|a5|b5]paper, [one|two]side, landscape, and [one|two]column.) Other options are placed in the extra class options field in the Document ▷ Settings dialog.

  More importantly, reLyX doesn't translate `\usepackage` commands, margin commands, `\newcommands`, or, in fact, anything else from the preamble. It simply copies them into the LaTeX preamble. If you have margin commands in your preamble, then the LyX file will generate the right margins. However, these margins will override any margins you set in the LyX Document▷Settings dialog. So you should remove the options from the preamble to be safe. The same goes for setting your language with babel, `\inputencoding`, `\pagestyle`, etc.

- The foil class has a couple bugs. reLyX may do weird things with optional arguments to `\foilhead` commands. Also, it may handle `\begin{dinglist}` incorrectly (although the stuff in the environment should translate normally).

reLyX is hopefully rather robust. As mentioned above, it may not translate your file perfectly, but it shouldn't crash. If it does crash—and the problem is not one of those mentioned above or in the *BUGS* file—see Section 5.4.5.1.

5.4.3.6 What LyX Can't Handle

LyX itself is missing a couple features, such that even if reLyX translates things perfectly, LyX may still have trouble reading it. If you really need these features, you can export your final document as LaTeX, and put them back in. See *BUGS* for more details on these bugs.

- For a number of commands, LyX does not support the optional argument. Examples include `\sqrt`, `\chapter` (and other sectioning commands), and \\. reLyX will automatically discard the optional arguments with a warning to stdout. LyX also ignores the width argument for the `thebibliography` environment.
- Centering (or right or left justifying) works on full paragraphs.
- LyX support for tables isn't perfect. For complicated tables, use a "skip" block, so that they will be copied in TeX mode.
- The LyX math editor can't handle the AMS-LaTeX math environments align, split, etc. So those environments will be copied in TeX mode. You can change `equation*` environments to the exactly equivalent displaymath, and then they will be translated correctly.

5.4.4 Examples

```
reLyX -df -o ‘‘my/dir’’ -r ‘‘myenv’’ foo.tex > foo.debug
```

The above will create a file my/dir/foo.lyx from foo.tex, overwriting if necessary. When it finds a `\begin{myenv} ... \end{myenv}` block, it will translate the stuff within the block, but copy the `\begin` and `\end` commands in TeX mode. Finally, I'm going to keep the temporary files around (they will also be in my/dir/) and output lots of debugging information into the file foo.debug.

5.4.5 Notes

5.4.5.1 Bug Reports

If reLyX is crashing or otherwise acting strangely—in ways other than those described in Section 5.4.3.5 or the bug tracker—then please run reLyX **-d**. That will allow you to figure out where in the reLyXing process it crashed. That, in turn, will allow you to write a better bug report, which will allow the developers to fix it more quickly and easily.

Bug reports should be sent to the LyX developers' mailing list. Its address is currently `lyx-devel@lists.lyx.org`. If you are running reLyX on a huge file, please do not send all of the output in your bug report. Just include the last ten or twenty lines of output, along with the piece of the LaTeX file it crashed on. Or, even better, attach a small but complete file which causes the same problem as your original file.

5.4.5.2 Implementation Details:

reLyX makes several "passes" in order to translate a TeX file. On each pass, it creates one or two files.

Pass 0
Before doing anything, read the syntax file (or files).

Pass 1a
Split preamble (anything before a `\begin{document}` command) off the rest of the file. It saves the two pieces in separate files. This is necessary because there may be very strange stuff in a preamble. It also ignores anything after the `\end{document}`, on the assumption that it isn't LaTeX.

Pass 1b
Translate the preamble. Currently, that just means translating the `\documentclass` command and copying the rest exactly into the LyX preamble.

Once you know what class the document is, read the LyX layout file for that class.

Pass 2
"Clean" the TeX file, generating slightly stricter LaTeX. This includes:

- Change, e.g., `x^2` to the equivalent but clearer `x^{2}`
- Removing optional arguments that LyX can't handle (e.g., from `\sqrt`)
- Changing `{\em foo}` to `\emph{foo}`, etc. This is necessary because LyX always writes out the non-local forms anyway. This should very rarely make a difference.

Pass 3
Translate LaTeX text, commands, and environments to LyX.

Pass 4
Put the two pieces back together, and do some final tweaking, to generate the LyX file

If there are any `\input` or `\include` commands, reLyX will loop back to the beginning and translate those. It assumes that the included files are the same class as the main file, and that they have no preamble matter. (If you have an `\input` command in the preamble of a file, the command will be copied exactly into the LaTeX preamble portion of the LyX file, so the included file won't be translated.) So when translating included files, it skips passes 0 and 1.

If reLyX doesn't find a file you wanted to include, it will give a warning, but will continue to translate any files it does find.

5.4.5.3 Layout Files

reLyX reads a LyX layout file to know how to handle LaTeX environments and commands which get translated to LyX layouts. This file will include all "normal" non-math environments (i.e., including quote and itemize, but not tabular, minipage, and some other fancy environments), and commands like `\section` and `\title`. If you want to reLyX a class that doesn't have an existing layout file, then you'll have to create a layout file. But you have to do this anyway, in order to LyX the file, since LyX depends on layout files to know how to display and process its files. Check the LyX documentation for help with this task (which can be hard or easy, depending on the class you want to create a layout file for.) If your class is quite similar to a class that has a layout file, then consider using the **`-c`** option.

5.4.5.4 Syntax Files

reLyX always reads at least one syntax file, called the default syntax file. reLyX will read your personal syntax file if it exists; otherwise it will read the system-wide file. reLyX will read additional syntax files if you specify them with the **`-s`** option. (These extra files should have the same format as the default file, but will tend to be shorter, since they only have to specify extra commands not found in the default file.) A syntax file tells reLyX a few things.

First, it describes the syntax of each command, that is, how many required arguments and how many optional arguments the command takes. Knowing this makes it easier for reLyX to copy (in TeX mode) commands that it doesn't know how to translate. The syntax file simply has a command, followed by braces or brackets describing its arguments in the correct order. For example, a syntax file entry `\bibitem[]{}` means that the `\bibitem` command takes an optional argument followed by a required one, while the entry `\bf` means that the `\bf` command takes no arguments at all. When reLyX encounters a token that it doesn't know how to translate into LyX, it will copy the token—along with the correct number of arguments—exactly. If the token is not in the syntax file, then reLyX just copies as many arguments as it finds. This means that it may copy too much. But since the user can specify additional syntax files, that shouldn't happen often.

Some commands that cannot be translated to LyX, like `\mbox`, have as one of their arguments regular LaTeX text. If the string "translate" is put into an argument of an (untranslatable) command in the syntax file, then reLyX will translate that argument instead of copying it verbatim. So, for example, the default syntax file has `\raisebox{}[][]{translate}`. This means that the `\raisebox` command and the first argument (and optional arguments if they exist) are copied in TeX mode, but the last argument (which may contain math, complicated LaTeX, other untranslatable commands, etc.) will be translated into LyX. You can't use "translate" on optional arguments.

User-defined syntax files are allowed to define new commands and their syntax, or override the number of arguments for a command given in the default syntax file. (E.g., if you're using a style that gives an extra argument to some command...) However, this will only be useful for commands copied in TeX mode. Commands which are actually translated by reLyX (like `\item`) have their argument syntax hard-coded. The hard-coded commands are identified in the default syntax file.

Second, the syntax file describes any "regular environments". Usually, an entire unknown environment will be copied in TeX mode. If you define a regular environment "foo", though, then only the `\begin{foo}` and `\end{foo}` commands will be copied in TeX mode; the text within the environment will be treated (i.e., translated) by reLyX as regular LaTeX, rather than being copied into TeX mode. Don't try to declare `tabbing` and `picture` as regular environments, as the text within those environments will confuse reLyX; use this capability for new environments you create that have plain text or math or simple commands in them. You also can't declare unknown math environments (like `equation*`) as regular environments, either, since the LyX math editor won't understand them. The names of regular environments appear, whitespace-separated, between `\begin{reLyXre}` and `\end{reLyXre}` statements in the syntax file. (If you have a regular environment which you won't use very often, you can use the `-r` option rather than writing a syntax file.)

Third, the syntax file describes a math translation table. The LyX math editor doesn't support a few commands. For example, `_` is supported, but the equivalent `\sb` is not. Put any commands you'd like translate between `\begin{reLyXmt}` and `\end{reLyXmt}` statements. The statement "`\| {\Vert}`" means that any `\|` in math mode will be converted to "`\Vert` " (in cases where a token made up of a backslash and a non-letter is translated to something with letters at the end, a space is added by reLyX. That way, "`\|a`" is correctly translated to "`\Vert a`").

5.4.5.5 Miscellaneous

You need Perl version 5.002 or later to run reLyX. <plug> If you don't have Perl, you should get it anyway (at Perl `http://www.perl.com/`), because it's a really useful tool for pretty much anything. </plug>

5.4.6 Diagnostics

reLyX should always explain why it crashes, if it crashes. Some diagnostics may be very technical, though, if they come from the guts of the code. reLyX gives much more information while running if you use the **-d** option, but you shouldn't need that unless something goes wrong.

When it's finished, reLyX will tell you if it finished successfully or died due to some error.

5.4.7 Warnings

Always keep a copy of your original LaTeX files either under a different name or in a different directory. There are a couple ways in which using LyX could lead to overwriting the original LaTeX file.

If you import `foo.tex` to create `foo.lyx`, then edit `foo.lyx` and want to re-export it, note that it will overwrite the original `foo.tex`. (LyX will *not* ask you if you want to overwrite it.)

If you have chosen not to use a temporary directory in the preferences, then LyX will create its temporary files in your current directory, which means your LaTeX original may be overwritten (without a warning from LyX) when you "view dvi" or print the LyX document.

5.4.8 Files

`MY_LYXDIR/layouts/*.layout`
User's personal layout files for document classes

`MY_LYXDIR/reLyX/syntax.default`
User's personal syntax file

`LIBDIR/layouts/*.layout`
System-wide layout files for document classes

`LIBDIR/reLyX/syntax.default`
System-wide LaTeX syntax file

`LIBDIR` is the system-wide LyX directory, usually something like `/usr/local/share/lyx/`. `MY_LYXDIR` is your personal LyX directory, something like `.lyx/` in your home directory. You can see their actual values in the Help ▷ About LyX dialog.

5.4.9 See also

lyx(1), *latex*(1)

5.4.10 Authors

Copyright (c) 1998–9 Amir Karger (`karger@voth.chem.utah.edu`)

Code contributors:

- John Weiss wrote the original CleanTEX pass.
- Etienne Grossmann
- José Abílio Oliveira Matos
- David Suarez de Lis

Other contributors:

- Jean-Marc Lasgouttes worked on the wrapper script and offered lots of bug reports, advice, and feature suggestions.
- Asger K. Alstrup Nielsen and Marc Pavese provided advice.
- Various members of the LYX developers' and users' lists provided bug reports and feature suggestions.

reLYX uses a modified version the Perl TEX parser `Text::TeX` package written by Ilya Zakharevich (`ilya@math.ohio-state.edu`), available on CPAN.

Chapter 6

LyX Features needing Extra Software

6.1 Using LyX with SGML-Tools (aka LinuxDoc)

by Paul Evans

6.1.1 Overview

LinuxDoc is a document class available in LyX if you have the `sgml-tools` package installed. You can use it to produce documents in the so-called Standardized General Mark-up Language (SGML) in the particular format used by the Linux Documentation Project. That is obviously helpful if you are contributing to that project. You can use the SGML format with the `sgml-tools` package of scripts and programs (to produce other formats, including Latex, HTML, plain text, man pages and...). You may therefore prefer to use this document class if you want to write something that can be easily translated into other formats.

You will find that LinuxDoc has fewer layout options than the other text classes in LyX. This is mainly so that the translations into other formats have a chance of making some sense. In this section we describe:

- how to setup and use a document in LinuxDoc
- how to use the tags in LinuxDoc to layout your document
- how to use the SGML packages to produce the various formats
- how to sort out some problems.

6.1.2 Preparing and using a LinuxDoc document

6.1.2.1 Getting started

You start by selecting the LinuxDoc class using the Document ▷ Settings dialog. Then you will find that there are fewer paragraph environments than for most other classes. You can see them on the pull down box on the left of the tool bar. How to use them is described in section 6.1.3.2.

You *must* enter a title for the document, followed by an author, marking each with the appropriate paragraph environment. If you don't do this, you will get errors when you try to print the file. You can then enter the date and an abstract. The document proper must start with a Section paragraph environment rather than any standard layout.

After that you can prepare a document as usual using the available range of paragraph environments. See section 6.1.3.2 for the full list and their uses.

6.1.2.2 Output from LinuxDoc

You can print and save these documents in the normal way. To use the other features of the SGML package you need to save your document as LinuxDoc; this is a version in which the document is translated into the basic sgml tags. Use File ▷ Export ▷ LinuxDoc. You will get a file with the same name and a `.sgml` extension rather than a `.lyx` extension. See 6.1.4 on how you than make use of this file.

6.1.3 Using the paragraph environments in LinuxDoc

6.1.3.1 The Structure of a LinuxDoc Document

There is a formal structure for LinuxDoc which limits how you can place tags. There are two parts to all documents:

Header: this is everything up to the first time you insert a Section layout marker. It can include title, author, date, abstract and ToC. You must include the first two.

Body: from the beginning of the first section onwards. All other tags are allowed.

6.1.3.2 The LinuxDoc Paragraph Environments

Here is a list of all the tags you will find listed on the layout bar in the order they come there, with some comments where the purpose or use is not obvious:

- Standard: works as described in [cross reference]
- Title: This will appear at the top left of the document when printed, above a heavy horizontal rule, although you will not see this on the LYX screen.

- Section, Subsection, Subsubsection, Paragraph and Subparagraph: all do what you would expect and in the usual order. Whether they are numbered or not is controlled by the Section number depth setting. You cannot get the equivalent number free versions in any other way; there is no Section* or similar
- Enumerate: As usual this produces a numbered and indented list as described in the *User's Guide*.
- Itemize: Again much the same as in the other classes: see the *User's Guide*.
- Description: As explained in the *User's Guide*. Remember that if you want the bold element at the start of a description to be more than one word then you need to put protected spaces between the words.
- Verbatim: As usual.
- Code: similar to the Lyx-Code environment
- Author: Anything you mark with this will appear on the left of the heading of the document, under the heavy rule.
- Date: Anything you mark with this will appear on the right of the heading under the rule. You do not have to make this a date. Any text can be entered, e. g. a version number.
- Abstract: You can use this to produce a free standing paragraph after the author and date, and before the first section. You are only allowed one such paragraph.[1]
- Displaymath:[2]

6.1.3.3 Other document features

You can also use the Layout menu to set fonts or to emphasis words. You can also use the table of contents as usual; see the corresponding section of the *User's Guide*. Although you will find some some other features on the menus e. g. inserting footnotes. There is some doubt about whether these will work correctly.[3]

6.1.3.4 Cross references and HTML

On the Insert menu you will find two new options relating to the inclusion of URL addresses. If you use either option you will find some highlighted TEX code inserted into your document in three separate blocks with spaces available between. The blocks will be:

[1] *Author's note.* This needs checking —*pe.*

[2] *Author's note:* I have not yet checked this —*pe.*

[3] *Author's note:* Again still checking to see whether this is my system —*pe.*

`\htmlurl{` or `\url{` space `}{` space `}`

You insert a full HTML tag between the first and second blocks. This can be `http://any.address` or other valid tags such as `mailto:me@my.address.`Then you insert some description between the second and third blocks. The differences are:

- URL: both the HTML tag and the description will appear in the document
- HTML URL: only the description appears in the printed version

6.1.4 Using the LinuxDoc Sgml scripts

You can use LinuxDoc as a text class without any additional scripts or programs, but there is not much point in doing this. All you will get is a document that looks like a *Linux Documentation Project Howto*. To do the document translation you need to get and install the `sgml-tools-1.0.x.tar.gz` (with $x \geq 3$) package from the SGML-Tools WWW Page at

```
http://pobox.com/~cg/sgmltools
```

Alternatively, you can go to the `sunsite` archive at[4]

```
ftp://sunsite.unc.edu/pub/Linux/utils/text/sgml-tools-1.
0.x.tar.gz
```

The file `sgml-tools-1.0.x.tar.gz` contains everything that you need to write SGML documents and convert them to groff, LaTeX, HTML, GNU info, LyX, and RTF.

This package was renamed from `linuxdoc-sgml-1.5.tar.gz` in January 1997.

Follow the instructions in that package on how to install it and how to use it. All this has to be done outside of LyX, before you can use the File ▷ Export ▷ as LinuxDoc option.

6.1.5 Troubleshooting LinuxDoc

When you print or preview a LinuxDoc document some checking is done of the tags before LaTeX is run. Some errors are trapped here, especially those concerning the structure of the document. LyX may produce an error message, but not leave an error box in the document for you to open. You may have to look at the files directly to discover what is wrong. Most problems seem to come from the use of options that are not fully available in the text class.

[4] Note that, at the time of this writing (01/1998), version 1.0.3 of sgml-tools has not yet been made available at `sunsite`.

6.2 Checking TEX

by Asger Alstrup

6.2.1 Introduction

Under the Tools menu, you'll find a Check TEX command. This feature requires you to have the `chktex` program installed, and is grayed out if you don't have it. You can get it from your nearest CTAN mirror, or over the Web from `http://www.ifi.uio.no/~jensthi/chktex/`.

The ChkTEX package is a program that was written by Jens T. Berger Thielemann in frustration because some constructs in LATEX are sometimes non-intuitive, and easy to forget. The program runs over your LATEX file and checks the integrity of the file, and flags some common errors. In other technical words, it is `Lint` for LATEX.

Well, what is a syntax checker doing in LYX which is supposed to produce correct LATEX anyways? The answer is simple: Just as `Lint` not only checks the *syntax* of C programs, but also does *semantic* checks for type-errors, ChkTEX catches some common *typographic* errors, in addition to the syntactical ones. Specifically, ChkTEX is capable of detecting several common errors, such as

- Ellipsis detection:
  Use . . . instead of ...

- No space in front of/after parenthesis:
  (wrong spacing)

- Enforcement of normal space after common abbreviations:
  e. g. is too wide spacing.

- Enforcement of end-of-sentence space when the last sentence ends with a capital letter:
  This is a TEST. And this is wrong spacing.

- Space in front of labels and similar commands:
  The label should stick right up to the text to avoid falling to a wrong page. [5] The label is separated too much.

- Space in front of references, instead of hard spaces:
  In you are in bad luck, the text will break right between the referenced text and reference number, and that's a pity. See section 6.2.1.

- Use of "x" instead of × between numbers:
  2x2 looks cheap compared to 2 × 2.

and more . . . It is an invaluable tool when you are "finishing up" your document before printing, and you should run it right after the obligatory spelling check, and before you go fine tuning the typesetting.

[5] This footnote is in danger of falling off to a wrong page

6.2.2 How to use it

If you have the program installed, usage is as simple as choosing Tools ▷ Check TEX. This will make LYX generate a LATEX file of your document, start ChkTEX to check it, and then make LYX insert "error boxes" with the warnings from ChkTEX, if there were any. The warnings will be placed close to the point of the mistake, and you can quickly find them by using the Navigate ▷ Error menu item, or the shortcut key C-g from the default cua bind file. Open the error boxes by clicking on them with the mouse, or use the shortcut key C-i from cua bindings, or the corresponding C-o for the alternate emacs bind file. Read the warning and correct the mistake, if it is a mistake. If you have trouble understanding what the warning is about, you can safely ignore it. Remember that there is a hidden layer between the document on screen and the technical details in invoking ChkTEX, and this gap can make some warnings seem arcane or just right down plain silly.

This document is an excellent testing bed for the feature, and it should provide quite a few warnings for you to fiddle with. Since computers are only so smart, expect most of the warnings to be false alarms, though.

6.2.3 How to fine tune it

Sometimes, you'll find that ChkTEX makes more noise than suits your mood. Then you can choose not to use it, wait until your mood changes, or try to customize ChkTEX to get better along with you. Another choice in the most desperate situations is to use View ▷ Remove All Error Boxes, which will get rid of all warnings instantly.

Although ChkTEX *is* very configurable and extensible, you shouldn't expect to solve all problems with ChkTEX in LYX this way. Since LYX has to generate a somewhat special LATEX file to be able to match the line numbers from the ChkTEX output[6] to the internal document structure, some of the warnings will not seen to appear correctly. There are two things you can do about this:

- Fine tune the ChkTEX invocation command line in Preferences (tabs Outputs, Misc), or the global ChkTEX installation configuration file (usually with the file /usr/local/share/chktexrc). See below to learn what warnings can be enabled and disabled on the command line.

- Export your document as a raw LATEX file using File ▷ Export ▷ LaTEX and run chktex manually on that. Invoked in this way, it can be a hassle to find the corresponding place in the document inside LYX, but with a little patience, you should be able to do it.

Here follows the warning messages that can be enabled and disabled in Preferences. Use -n# to disable a warning, and -w# to enable a warning. The

[6] You can inspect the specific output from chktex by using Edit ▷ View LATEX Log right after a chktex run.

emphasized entries are disabled by default, because the default is `"chktex -n1 -n3 -n6 -n9 -n22 -n25 -n30 -n38"`.

Notice that you should only use the options that enable and disable warnings, because LYX relies on some of the other command line parameters to be set in a specific way to have a chance to communicate with `chktex`.

1. *Command terminated with space.*
2. Non-breaking space ("~") should have been used.
3. *You should enclose the previous parenthesis with "{}".*
4. Italic correction ("\/") found in non-italic buffer.
5. Italic correction ("\/") found more than once.
6. *No italic correction ("\/") found.*
7. Accent command "`cmd`" needs use of "`cmd`".
8. Wrong length of dash may have been used.
9. *"%s" expected, found "%s".*
10. Solo "`%s`" found.
11. You should use "`%s`" to achieve an ellipsis.
12. Inter-word spacing ("\ ") should perhaps be used.
13. Inter-sentence spacing ("\@") should perhaps be used.
14. Could not find argument for command.
15. No match found for "`%s`".
16. Math mode still on at end of LATEX file.
17. Number of "`char`" doesn't match the number of "`char`".
18. You should use either `` or '' as an alternative to """.
19. You should use "'" (ASCII 39) instead of "´" (ASCII 180).
20. User-specified pattern found.
21. This command might not be intended.
22. *Comment displayed.*
23. Either ''\,' or '\,'' will look better.
24. Delete this space to maintain correct page references.
25. *You might wish to put this between a pair of "{}".*
26. You ought to remove spaces in front of punctuation.

27. Could not execute LaTeX command.
28. Don't use `\/` in front of small punctuation.
29. `$\times$` may look prettier here.
30. *Multiple spaces detected in output.*
31. This text may be ignored.
32. Use `` ` ` `` to begin quotation, not `'`.
33. Use `'` to end quotation, not `` ` ` ``.
34. Don't mix quotes.
35. You should perhaps use "`cmd`" instead.
36. You should put a space in front of/after parenthesis.
37. You should avoid spaces in front of/after parenthesis.
38. *You should not use punctuation in front of/after quotes.*
39. Double space found.
40. You should put punctuation outside inner/inside display math mode.
41. You ought to not use primitive TeX in LaTeX code.
42. You should remove spaces in front of "`%s`"
43. "`%s`" is normally not followed by "`%c`".

In later versions of LyX, we hope to provide a more complete interface to this tool (and it's smaller cousin `lacheck`) to exploit the full power of it. But it's not exactly useless as it is now: go try it on one of your existing documents of a certain length and be surprised.

6.3 Version Control in LyX

by Lars Gullik Bjønnes

6.3.1 Introduction

A friend of mine wanted to try LyX for a group project. When he didn't find support for version control or file locking, he dropped it. This angered me a bit, so I thought that I should at least make support for RCS (with the possibility of CVS and/or SCCS as a future improvement.) This has now been done. LyX now supports some of the most basic RCS commands. If you need to something a bit more sophisticated you will have to do that manually in an xterm.

Before you begin to use the version control features in LyX, you should read "rcsintro" (a man file, read it with `man rcsintro`). This file describes all the basic features of RCS. You should especially notice the comment about a RCS directory, and the notion of a master RCS file (the file ending in `,v`).

The implementation in LYX assumes a recent version of the GNU RCS package—no guarantees are made for older versions.

6.3.2 RCS commands in LYX

The following sections describe the RCS commands supported by LYX. You can find them in the File▷Version Control submenu.

6.3.2.1 Register

If your document is not under revision control, this is the only item shown in the menu. And if it is under revision control, the Register item is grayed out.

This command registers your document with RCS. You are asked interactively to supply an initial description of the document. The document is now set in Read-Only mode and you have to Check Out For Edit, before making any changes to it. A document under revision control has a "[RCS:<version> <locker>]" item tagged to the filename in the minibuffer.

RCS command that is run: `ci -q -u -i -t-"<initial description>" <file-name>`

Read `man ci` to understand the switches.

6.3.2.2 Check In Changes

When you are finished editing a file, you check in your changes. When you do this, you are asked for a description of the changes. This is stored in the history log. The version number is bumped, your changes are applied to the master RCS file, the document is unlocked and set to Read-Only mode.

RCS command: `ci -q -u -m"<description>" <file-name>`

6.3.2.3 Check Out For Edit

By doing this you lock the document so that only you can edit it. This will also make the document Read-Write only for you. You will usually continue editing for a while and when you are finished you check in your changes. The status line is changed to reflect that you have locked the file.

RCS command: `co -q -l <file-name>`

6.3.2.4 Revert To Last Version

This will discard all changes made to the document since the last check in. You get a warning before changes are discarded.

RCS command: `co -f -u<version> <file-name>`

6.3.2.5 Undo Last Checkin

This makes as if the last check in never happened. No changes are made to the document loaded into LYX, but the last version is removed from the master RCS file.

RCS command: `rcs -o<version> <file-name>`

6.3.2.6 Show History

This show the complete history of the RCS document. The output of `rlog <file-name>` is shown in a browser. See `man rlog` for more info.

6.4 Literate Programming

Updated by Kayvan Sylvan (kayvan@sylvan.com), original documentation written by Edmar Wienskoski Jr. (edmar-w-jr@technologist.com)

6.4.1 Introduction

The main purpose of this documentation is to show you how to use LYX for literate programming. Where it is assumed that you are familiar with this programming technique, and know what "tangling" and "weaving" means. If that is not the case, please follow the web links provided in the following sections. There is a lot of good documentation out there covering old development history to the latest tools tips.

It is also assumed that you are familiar with LYX itself to a point that you are comfortable changing your LYX preferences, and X resources file. If that is not the case please refer to other LYX documentation to cover your specific needs.

6.4.2 Literate Programming

From the Literate Programming FAQ:

> Literate programming is the combination of documentation and source together in a fashion suited for reading by human beings. In fact, literate programs should be enjoyable reading, even inviting! (Sorry Bob, I couldn't resist!) In general, literate programs combine source and documentation in a single file. Literate programming tools then parse the file to produce either readable documentation or compilable source. The WEB style of literate programming was created by D.E. Knuth during the development of his TEX typesetting software.

Another excerpt says:

> *How is literate programming different from verbose commenting?*
>
> There are three distinguishing characteristics. In order of importance, they are:
>
> - flexible order of elaboration
> - automatic support for browsing
> - typeset documentation, especially diagrams and mathematics

Now that I sparked your curiosity, take a look in the references.

6.4.2.1 References

The complete Literate Programming FAQ can be found at:

> Literate Programming FAQ `http://shelob.ce.ttu.edu/daves/lpfaq/faq.html`

The FAQ lists 23 (twenty three!) different literate programming tools. Where some are specialized or "tailored" for particular programming languages, while other have general scope. I selected NOWEB for my own use for several reasons:

- It can generate the documentation either in latex or html.
- It has a open architecture, i.e., it is easy to plug in new filters and to perform special processing that you may need.
- There is a good selection of filters available already (the html is one of them).
- It is free.

The Noweb web page can be found at:

> Noweb home page `http://www.cs.virginia.edu/~nr/noweb/`

Starting from there you can reach many other interesting links and even some literate program examples.

6.4.3 LYX and Literate Programming

The LYX support for Literate Programming is provided by using the generic LYX convertors mechanism. This support is provided in a "Noweb independent" way, i.e., you will be able to use this new LYX feature with some other literate programming tool of your choice by just changing your LYX preferences.

6.4.3.1 Generating documents and code (weaving and tangling)

Selecting the document class If you have installed Noweb and LyX successfully, whenever you open a new document or try to change the document class of an existing one, you will find that there are three new document classes available:

- Article (Noweb)
- Book (Noweb)
- Report (Noweb)

You must select one of them to create your literate documents from.

Note that literate documents are not limited to these three classes. New classes can be generated from other styles like letter or in combination with other class variations like Article (AMS). If you have special needs that cannot be covered by one of the existing classes, let the LyX developers list (lyx-devel@lists.lyx.org) know and we will arrange to insert a new entry, or teach you how to do it.[7] Moreover, if you use a literate tool other than Noweb you may need to create a new set of document classes for it.

Typing code in LyX enables you to write code with a layout named Scrap.[8] Noweb delimits scraps like this:

```
<<My scrap>>=
  code
  more code
  even more code
  @
```

The problem is that whatever is written in between the << and the @ must be taken literally, i.e., LyX should be prevented from making any special interpretation of what has been written. This is handled by a special layout named Scrap, that works like a normal paragraph but has a free spacing capability.

The down side of the Scrap paragraph layout is that consecutive paragraphs of code will be spaced with one empty line in the source code and also in the printed documentation. The work around is to enter each line of code within a single Scrap, with a newline (ctrl-return). The example above will look like this:[9]

[7] It is very simple, it involves the creation of a file with four lines, and re-running of the auto configuration.

[8] The equivalent Noweb term is “Chunk”. For historical reasons, I got used to the term “scrap” introduced by other literate tool named Nuweb, which I used for many years before rendering myself to Noweb.

[9] If you have a printed version of this document you will not see any difference between the previous example and this one.

```
<<My scrap>>=
  code
  more code
  even more code
  @
```

This layout works fine. The only real inconvenience is that you have to type ctrl-return instead of a plain return.[10]

As a special note, you can also use the "%def" construct of Noweb in your scraps to add items to Noweb's identifier cross-reference:

```
<<My scrap>>=
  def some_function(args):
    "This is the doc string for this function."
    print "My args: ", args
@ %def some_function
```

For an example of this usage and the resulting cross-reference output, look at the Literate python program in *LIBDIR/examples/listerrors.lyx* which should make this all clear.

Generating the documentation At this point you already have a new document file with a proper document class, and with some code and text on it. How do I print it? The answer is simple, you select View ▷ DVI, etc. Just like you would do for a plain document. No special procedure is required.

To help orientate you, I will now explain what happens inside LyX:

1. When the Update ▷ DVI menu option is chosen, a latex file is generated.

   If the document is of any literate class the generated file will be named with an extension name defined by the "literate" format (defined in the Preferences panel), otherwise the file will have the usual `.tex` extension.

2. Note that the only difference so far is in the name of the file, no special processing is required by LyX. Given that you formatted the code using the Scrap layout that, by itself, takes care of the business.

3. If the document is of any literate class LyX will then use the internal LyX to Noweb converter, followed by the Noweb to LaTeX converter[11] to generate the LaTeX file.

   Otherwise it will just skip this step.

4. Finally, LaTeX is invoked and the regular post processing continues as in a plain document.

Independence from a particular "literate tool" is easily achieved by changing the commands that are run by the various converters.

[10] It is in my list of "improvements" to fix that.

[11] The converters are defined in the Tools ▷ Preferences panel, under the "Conversion" tab.

Generating the code When the build menu option is chosen or the corresponding button in the toolbar is pressed, a latex file is generated just like step 1 above. Next, LyX invokes the `Noweb->Program` converter. Typically, this converter (like any other converter), has two parts:

1. The converter program itself. This program performs the conversion from the one format to the other (in this case, from the Noweb format to the Program pseudo-format).
2. The error log parser. This is a program whose sole purpose is to rewrite error messages in a format that LyX understands. This makes it possible for LyX to place error boxes in the right places in the file buffer.

The first part, the "Converter" setting, should be set to "`build-script $$i`". This basically means that LyX will call "build-script" (a program or script) with the name of the Noweb file (generally a file in the LyX temp directory).

This is an implementation of "build-script" that you can place in a directory on your path:

```
#!/bin/sh
#
notangle -Rbuild-script $1 | env NOWEB_SOURCE=$1 sh
```

The next part of the converter setting is the "Flags" which is to be set to "`originaldir,parselog=listerrors`". This will run any errors that are generated by the "build-script" process through the "listerrors" program.

The converter code looks in *MYLYXDIR/scripts* first, then in *LIBDIR/scripts* then on the path for the "listerrors" program.

Build instructions in the document The last piece of the integration between LyX and noweb is the "build-script" scrap. Generally, the instructions for building your program should be embedded in a scrap of its own. The noweb-specific "build-script" above uses the notangle command to look for this scrap (called "build-script") and runs its contents through "sh".

Typically, such a scrap would look something like this:

```
<<build-script>>=
#!/bin/sh

if [ -z "${NOWEB_SOURCE}" ]
then
  NOWEB_SOURCE=myfile.nw
fi
[... code to extract files ...]
[... code to compile files ...]
@
```

Look in *LIBDIR/examples/listerrors.lyx* or in *LIBDIR/examples/Literate.lyx* which implement two versions of the "listerrors" program for some illustrations of how all of these pieces go together or in *LIBDIR/examples/noweb2lyx.lyx.* Interestingly, these three files show off the language-indepence of the LyX literate programming support since they are written in Python, C and Perl respectively.

6.4.3.2 Configuring LyX

All the Literate Programming support is configured by the Tools ▷ Preferences panel in the "Conversion" tab. The important parts are:

the "literate" format Set up via the Formats tab, this is where the Noweb-specific pieces are set up. The GUI Name is set to `NoWeb`, the file extension is set to `.nw`. This tells LyX to create a file with a `.nw` extension in the first step of the conversion process.

the Program format This is an empty format whose sole purpose is to be the endpoint of a conversion (which then allows us to set up a converter for it).

NoWeb->LaTeX This converter performs the "weaving" of the literate document. For Noweb, it is set to "`noweave -delay -index $$i > $$o`"

NoWeb->Program This performs the "tangling step". As stated above, the Converter is set to "`build-script $$i`", with Flags set to "`originaldir,parselog=listerrors`".

6.4.3.3 Debug extensions

There is also a new function implemented in the LyX server, the "server-goto-file-row" function, to be used with ddd/gdb or other debugger.

When debugging code with ddd/gdb, it is possible to invoke a text editor at the current execution position with a single key stroke. The default ddd configuration for that is shift-ctrl-V. It happens that you can define the editor command line invocation in ddd by accessing the Edit ▷ Preferences ▷ Helpers dialog and changing the "Edit Sources" entry.

I take advantage of the new created LyX server function and this ddd feature, and set "Edit Sources" to:

```
echo "LYXCMD:monitor:server-goto-file-row:@FILE@ @LINE@" >~/.lyxpipe.in
```

With this, whenever you are using ddd and find a point in the program that you want to edit, you just press shift-ctrl-V (in the ddd window), and ddd you forward this information to LyX through the LyX server and then the LyX window will show the same file with the cursor at the same position ddd was pointing to. No more guessing or long scrolling to locate a point in the program back from debugging !

Note however that you must enable the LYX server to get this feature working (it is disabled by default). You can enable it in Preferences (tabs Inputs, Paths) by entering in the LYXserver pipe a path like "`/home/<your-home-directory>/.lyx/lyxpipe`"

Read the LYX server documentation in the *Customization Manual* for further information.

6.4.3.4 Toolbar extensions

There are six new buttons that can be added to your LYX toolbar. Five of these buttons are short cuts to layout styles: Standard, Section, LATEX, LYX-Code, and Scrap. The last one is a short cut to the "Build Program" File menu entry.

LYX has a range of buttons that are available for tool bar customization. In my toolbar I like to combine the six short cuts above with two more: One for View ▷ Update ▷ DVI and the other for View ▷ DVI File menu entries. Here is how it looks like:

```
Toolbar
  Layouts
  Icon "layout Standard"
  Icon "layout Section"
  Icon "layout LaTeX"
  Icon "layout LyX-Code"
  Icon "layout Scrap"
  Separator
  Icon "buffer-view"
  Icon "buffer-typeset"
  Icon "build-program"
  Separator
.
.
.
End
```

6.4.3.5 Colors customization

There are a number of colors in LYX that can be customized in Preferences. One of the things that bothers people is the LATEX font color. The default color is red, since the scraps uses LATEX font, and there is a lot of scraps in literate documents, you may get tired of seeing everything in red. You can change it by going to the tabs Look&Feel, Colors.

The next thing is the visible presence of the newline character in the screen. You can choose the color of this particular character and make it blend in the background. I recommend you choosing a color that is close to the background but not equal, that way you still can see it is there, but it is not bothering you anymore.

Chapter 7

Secrets of the LaTeX Masters

Though LyX is a powerful tool, it cannot hope to support everything that can be done with pure TeX/LaTeX. However, many familiar dirty TeX and LaTeX tricks can be done within LyX, as long as you are not afraid to use that "TeX" button on the toolbar or add things to the LaTeX preamble. This section lists some tips, tricks, and otherwise cool ideas to give your document that extra little flair. *Do try this at home*, just start with something a little smaller and less important than your dissertation!

Most ideas in this section require less common files in your LaTeX installation. If you have a system like teTeX, most will already be available. A few, however, will need to be downloaded from one of the CTAN archives. Often, there are several ways to do something, or several LaTeX style files which do the same thing. We do not endorse one choice over another, we simply claim that we have done a particular task with a particular file. Put on your wizard hat, keep an eye out for dragons, and let us begin.

7.1 Tricks for Footnotes and Margin Notes

suggested by Robin Socha

7.1.1 Footnotes

LyX cannot yet take care of setting the footnote numbering back to 1 after each section in the "article" document class or changing the counter style. You'll need to insert LaTeX commands like the following to achieve that:

Using `\setcounter{footnote}{0}` will set the counter back to 1[1].

The following command will change the numbering to small letters. Take a look at the next footnote in your xdvi or ghostview :[b]

[1] The counter has been set back to 1.

[b] This is an example for a footnote with alphabetic numbering.
Use `\renewcommand{\thefootnote {\alph{footnote}}` to get this.

The next command sets the counter style back to default, i.e. `\arabic`[c].

You can use `\arabic`, `\roman`, `\Roman`, `\alph` or `\Alph` and others as counter styles. Just replace the LaTeX command in the above example and rerun TeX to see what those styles can do.

7.1.2 Margin Notes

Here are two examples of neat things you can do to margin notes using LaTeX commands.

The following command will make a vertical line appear alongside your text—great for "thumbing": `\marginpar{\rule[-10mm]{30mm}{5mm}}`.

Check your dvi- or ghostview-output to see what the `\reversemarginpar` command does to the following margin note.

This is a margin note.

7.2 Multiple Columns

by Lars Gullik Bjønnes

7.2.1 Purpose

The aim for this chapter[d] is to show how the LaTeX package `multicol` can be used in a LyX document. As LyX doesn't support the `multicol` package natively yet, we have to use some small hacks. By reading this section it should be obvious how to do this.

7.2.2 Limitations

The `multicol` package allows switching between one and multicolumn format on the same page. Footnotes are handled correctly (for the most part), but will be placed at the bottom of the page and not under each column. LaTeX's float mechanism, however, is partly disabled in the current implementation. At the moment only page-wide floats can be used within the scope of the environment.

7.2.3 Examples

7.2.3.1 Two columns

If you want to have two columns in your text, you have use LaTeX mode to insert `\begin{multicols}{2}` at the point where you want the two column layout to start, and then `\end{multicols}` where you want it to end. Like this:

[c] Use `\renewcommand{\thefootnote}{\arabic{footnote}}` to set the counter–style back to LyX's default, i.e. \arabic.

[d] Editor's note: Lars' original chapter was a masterful description of how to use the `multicol` package. However, it was too long to flow smoothly in this document. I have therefore chosen to excerpt the most important sections here (sorry, Lars); you can read the original chapter (and more of the story!) in the example file `examples/multicol.lyx`. — mer

The Adventure of the Empty House
by Sir Arthur Conan Doyle

It was in the spring of the year 1894 that all London was interested, and the fashionable world dismayed, by the murder of the Honourable Ronald Adair under most unusual and inexplicable circumstances. The public has already learned those particulars of the crime which came out in the police investigation, but a good deal was suppressed upon that occasion, since the case for the prosecution was so overwhelmingly strong that it was not necessary to bring forward all the facts. Only now, at the end of nearly ten years, am I allowed to supply those missing links which make up the whole of that remarkable chain. The crime was of interest in itself, but that interest was as nothing to me compared to the inconceivable sequel, which afforded me the greatest shock and surprise of any event in my adventurous life. Even now, after this long interval, I find myself thrilling as I think of it, and feeling once more that sudden flood of joy, amazement, and incredulity which utterly submerged my mind. Let me say to that public, which has shown some interest in those glimpses which I have occasionally given them of the thoughts and actions of a very remarkable man, that they are not to blame me if I have not shared my knowledge with them, for I should have considered it my first duty to do so, had I not been barred by a positive prohibition from his own lips, which was only withdrawn upon the third of last month.

7.2.3.2 Multiple columns

The same pattern is used when you want more than two columns:

It can be imagined that my close intimacy with Sherlock Holmes had interested me deeply in crime, and that after his disappearance I never failed to read with care the various problems which came before the public. And I even attempted, more than once, for my own private satisfaction, to employ his methods in their solution, though with indifferent success. There was none, however, which appealed to me like this tragedy of Ronald Adair. As I read the evidence at the inquest, which led up to a verdict of willful murder against some person or persons unknown, I realized more clearly than I had ever done the loss which the community had sustained by the death of Sherlock Holmes. There were points about this strange business which would, I was sure, have specially appealed to him, and the efforts of the police would have been supplemented, or more probably anticipated, by the trained observation and the alert mind of the first criminal agent in Europe. All day, as I drove upon my round, I turned over the case in my mind and found no explanation which appeared to me to be adequate. At the risk of telling a twice-told tale, I will recapitulate the facts as they were known to the public at the conclusion of the inquest.

You can have have more than 3 columns if you want to, but that might not be very pleasant for the eye.

7.2.3.3 Columns inside columns

You can even have columns inside columns:

The Honourable Ronald Adair was the second son of the Earl of Maynooth, at that time governor of one of the Australian colonies. Adair's mother had returned from Australia to undergo the operation for cataract, and she, her son Ronald, and her daughter Hilda were living together at 427 Park Lane.

The youth moved in the best society–had, so far as was known, no enemies and no particular vices. He had been engaged to Miss Edith Woodley, of Carstairs, but the engagement had been broken off by mutual consent some months before, and there was no sign that it had left any very profound feeling behind it. For the rest {sic} the man's life moved in a narrow and conventional circle, for his habits were quiet and his nature unemotional. Yet it was upon this easy-going young aristocrat that death came, in most strange and unexpected form, between the hours of ten and eleven-twenty on the night of March 30, 1894.

Ronald Adair was fond of cards–playing continually, but never for such stakes as would hurt him. He was a member of the Baldwin, the Cavendish, and the Bagatelle card clubs. It was shown that, after dinner on the day of his death, he had played a rubber of whist at the latter club. He had also played there in the afternoon. The evidence of those who had played with him– Mr. Murray, Sir John Hardy, and Colonel Moran–showed that the game was whist, and that there was a fairly equal fall of the cards. Adair might have lost five pounds, but not more. His fortune was a considerable one, and such a loss could not in any way affect him. He had played nearly every day at one club or other, but he was a cautious player, and usually rose a winner. It came out in evidence that, in partnership with Colonel Moran, he had actually won as much as four hundred and twenty pounds in a sitting, some weeks before, from Godfrey Milner and Lord Balmoral. So much for his recent history as it came out at the inquest.

Please do read the file `examples/multicol.lyx` for more advanced examples including column and header spacing, vertical separator lines, and more.

7.3 Numbering in the Enumerate Paragraph Environment

by John Weiss

The default numbering for the Enumerate paragraph environment begins with Arabic numbers and ends with uppercase letters. Suppose, however, you wanted a different type of numbering scheme. Here's a quickie example of how to change the numbering scheme:

```
\renewcommand{\labelenumi}{\Roman{enumi}.}
\renewcommand{\labelenumii}{\Alph{enumii}.}
\renewcommand{\labelenumiii}{\arabic{enumiii}.}
\renewcommand{\labelenumiv}{\alph{enumiv}.)}
```

. . . which changes the numbering scheme to uppercase Roman numerals, uppercase letters, Arabic numbers, and lowercase letter.

Additionally, the previous example also adds a little bit extra to the numbering scheme. For example, the first level label actually looks like: “I.”. For ease of reading, we'll describe what the numbering schemes look like using a notation something like this: <“I.”, ”A.”, ”1.”, “a.)”>.

As you can see in the example, there is a label command for each nesting level, `\labelenumi` . . . `\labelenumiv`, as well as a counter, `enumi` . . . `enumiv`. There are also five “number printing” commands, `\arabic{}`, `\roman{}`, `\Roman{}`,

`\alph{}`, and `\Alph{}`, each of which take one counter as an argument. You can add characters before or after these, but there's no need to add spaces.

You can get really fancy with these. For example:

```
\renewcommand{\labelenumi}{\#\Alph{enumi}\#}
\renewcommand{\labelenumii}{\Alph{enumi}.\arabic{enumii}}
\renewcommand{\labelenumiii}{\alph{enumiii}+}
\renewcommand{\labelenumiv}{(\roman{enumiv})}
```

produces the somewhat out of hand numbering scheme: <“#A#”, ”A.1”, ”a+”, “(i)”>.

7.4 Extra Space Between Table Rows

by Mike Ressler

LaTeX allows you to put a bit of extra space between rows in a table by giving an optional argument to the end-of-row specifier (\\). LyX has not yet implemented this in a formal way, so here are two dirty little tricks to do the same job.

The first is the more formal, but longwinded way to do it. In the LaTeX preamble, add the following command definition:

`\newcommand{\extratablespace}[1]{\noalign{vskip#1}}` This command takes a single argument—the amount of space you would like to insert. Insert the command in the first column of the row *after* where you would like the space to appear. Here is an example (I've removed all the borders using Layout ▷ Table):

Minerals	Calcite	Dolomite
	Quartz	Graphite
Rocks	Limestone	Sandstone
	Granite	Andesite

The second method is faster, but will make typographers and TeXperts all over the world groan. Simply put an end of row specifier with optional argument at the same spot. No fancy definitions are needed as in the above example, but there will be more space inserted than you specified because you essentially added a blank row plus the extra space. If the space added is too much, simply use a negative number, like so:

Minerals	Calcite	Dolomite
	Quartz	Graphite
Rocks	Limestone	Sandstone
	Granite	Andesite

It's short, sweet, and gets the job done quickly, even if it is really ugly. You may put away the rotten vegetables now! I promise I won't suggest anything else like that!

7.5 Dropped Capitals

by Mike Ressler

Those of you who like the style of old books probably also like "dropped capitals"—those large capital letters which begin each new chapter or section. Implementing them with plain LyX/LaTeX is straightforward (assuming you know some plain TeX!) but does require a lot of work and many iterations, as you can see by all the ugly TeX-mode stuff at the beginning of this paragraph.

`\bigdrop{-1em}{3}{ptmri}{T}`here is a much easier way of doing this, of course. The `dropcaps` (or the newer `dropping`) package from CTAN allows a simple way to add such letters to your documents. Since this package is not a standard part of teTeX, I can't demonstrate it within this document, but if you copy this paragraph to a new document, delete the "`\verb`" and the pluses from the TeX code at the beginning of the paragraph, and add `\usepackage{dropcaps}` to your LaTeX preamble, you will get a nice Times Roman Italic "T", whose height is three lines of text and which protrudes 1 em into the margin. (Make certain you have copied "`dropcaps.sty`" into a directory where TeX can see it.) The first argument is the amount of indentation; in this case the negative sign moves it into the margin. The second argument is the height of the letter in number of lines of text. The third argument is the font name: virtually anything which has a tfm file should work (wade through the `.../texmf/fonts/tfm` directory for possibilities). My personal favorite is "`yinit`", a fancy German font specifically designed for dropped capitals. The fourth argument is the letter (or letters) to be dropped. The `dropping` package also offers the `\bigdrop` command, as well as a slightly simplified `\dropping` command.

7.6 Non-standard Paragraph Shapes

by Mike Ressler

There are times when the tyranny of rectangular paragraphs must be overthrown. In such situations, a call to the delightful plain TeX command `\parshape` is called for. As you can see, completely arbitrary shapes can be laid out with a suitable set of linelength definitions. While this parshape may look a bit silly and useless, one could conceive of situations such as finely tuned dropped capitals, word wrapping around non-rectangular graphics, etc. which will benefit from such handcrafting.

The syntax is `\parshape numlines #1indent #1length #2indent #2length ... #nindent #nlength`, where `numlines` is the number of lines of text which define the paragraph. If there turn out to be fewer lines, the shape is truncated; if there are more, the excess lines have the same dimensions as the last line of the definition. The `#nindent` and `#nlength` entries specify the indentation of the line from the left margin, and the length of the line as measured from that point. The shape applies only to the current paragraph; everything is reset to normal for the next paragraph.

7.7 Summary

As you can see, the examples in this section range from the useful to the whimsical. While I don't expect that anyone will ever need the paragraph shape demonstrated in the last section, the important point is that you can do almost anything you want in LyX if you are willing to figure out how to do it in TeX and LaTeX. TeX is a fantastically powerful typesetting system and all that power is available to you since LyX uses it as its backend. Happy LyXing!

Customizing LyX: Features for the Advanced User

by the LyX Team[1]

July 17, 2006

[1] Principal maintainer of this file is Mike Ressler. If you have comments or error corrections, please send them to the LyX Documentation mailing list, <lyx-docs@lists.lyx.org>.

Contents

Chapter 1

Introduction

This manual covers the customization features present in LyX. In it, we discuss issues like keyboard shortcuts, screen previewing options, printer options, sending commands to LyX via the LyX Server, internationalization, installing new LaTeX classes and LyX layouts, etc. We can't possibly hope to touch on everything you can change—our developers add new features faster than we can document them—but we will explain the most common customizations and hopefully point you in the right direction for some of the more obscure ones.

Chapter 2

LyX configuration files

This chapter aims to help you to find your way through the LyX configuration files. Before continuing to read this chapter, you should find out where your LyX library directory is by using Help ▷ About LyX. This directory is the place where LyX places its system-wide configuration files, and we will simply name it `LyXDir` in the remainder of this document.

2.1 What's in `LyXDir`?

`LyXDir` and its sub-directories contain a number of files and that can be used to customise LyX's behaviour. You can change many of these files from within LyX itself through the Tools ▷ Preferences dialog. Most customization that you might want to do to LyX is possible through this dialog. However, many other inner aspects of LyX can be customized by modifying the files in `LyXDir`. They fall in different categories, described in the following subsections.

2.1.1 Automatically generated files

These files are generated when you configure LyX. They contain various default values that are guessed by inspection. In general, it is not a good idea to modify them, since they might be overwritten at any time.

`lyxrc.defaults` contains defaults for various commands.

`packages.lst` contains the list of packages that have been recognized by LyX. It is currently unused by the LyX program itself, but the information extracted, and more, is made available with Help ▷ LaTeX Configuration.

`textclass.lst` is the list of text classes that have been found in your `layout/` directory, along with the associated LaTeX document class and their description.

`doc/LaTeXConfig.lyx` is automatically generated during configuration from the file `LaTeXConfig.lyx.in`.

2.1.2 Directories

`bind/` this directory contains files with the extension `.bind` that define the keybindings used in LyX (see section 3.3). If there exists an internationalized version of the bind file named `$LANG_xxx.bind`, that will be used first. See Chapter 4, and section 3.3for details.

`clipart/` contains graphics files that can be included in documents.

`doc/` contains LyX documentation files (including the one you are currently reading). The file `LaTeXConfig.lyx` deserves special attention, as noted above. If there exists an internationalized version of the help-document with `$LANG_` prepended to the name, that will be used first. See Chapter 4 for details.

`examples/` contains example files that explain how to use some features. In the file browser, press the Examples button to get there.

`images/` contains image files that are used by the Document dialog. In addition, it also contains the individual icons used in the toolbar and the banners that can be shown when LyX is launched.

`kbd/` contains keyboard keymapping files. See Chapter 4.4 for details.

`layouts/` contains the text class files described in Chapter 5.

`reLyX/` contains lots of files that together make up reLyX, the translator of "well behaved" LaTeX into LyX.

`scripts/` contains some files that demonstrate the capabilities of the External Template feature.

`templates/` contains the standard LyX template files described in Chapter 5.4.

`tex/` contains some LaTeX cls files distributed with LyX.

`ui/` contains files with the extension `.ui` that define the user interface to LyX. That is, the files define which items appear in which menus and the items appearing on the toolbar. See Chapter 3.4 for details.

2.1.3 Files you don't want to modify

These files are used internally by LyX and you generally do not need to modify them unless you are a developer.

`CREDITS` this file contains the list of LyX developers. The contents are displayed with the menu entry Help ▷ About LyX.

`chkconfig.ltx` this is a LaTeX script used during the configuration process. Do not run directly.

`configure` this is the script that is used to re-configure LyX. It creates configuration files in the directory it was run from.

2.1.4 Other files needing a line or two...

`encodings` this contains tables describing how different character encodings can be mapped to unicode

`external_templates` this file contains the templates available to the new External Template feature.

`languages` this file contains a list of all the languages currently supported by LyX.

`lyxrc.example` Deprecated and definitely obfuscated. This is the old style preferences file. It will probably disappear in the near future.

2.2 Your local configuration directory

Even if you are using LyX as an unprivileged user, you might want to change LyX configuration for your own use. The `UserDir` directory contains all your personal configuration files. This is the directory described as "user directory" in Help ▷ About LyX. This directory is used as a mirror of `LyXDir`, which means that every file in `UserDir` is a replacement for the corresponding file in `LyXDir`. Any configuration file described in the above sections can be placed either in the system-wide directory, in which case it will affect all users, or in your local directory for your own use.

To make things clearer, let's provide a few examples:

- The preferences set in the Tools ▷ Preferences dialog are saved to a file `preferences` in `UserDir`.
- When you reconfigure using Tools ▷ Reconfigure, LyX runs `configure` and the resulting files are written in your local configuration directory (see section 3.9 to have a list of the `preferences` settings affected by this section). This means that any additional text class file that you might have added in `UserDir/layouts` will be added to the list of classes in the Layout ▷ Document dialog.
- Similarly, if you have installed some LaTeX document classes in your home directory, that LaTeX can find with your `TEXINPUTS` path, they will show up in your list of text classes.[1]

[1] as long as LyX or yourself have a `.layout` file for it, of course.

- If you get some updated documentation from LyX ftp site and cannot install it because you do not have sysadmin rights on your system, you can just copy the files in `UserDir/doc/` and the items in the Help menu will open them!

2.3 Running LyX with multiple configurations

The configuration freedom of the local configuration directory may not suffice if you want to have more than one configuration at your disposal. For example, you may want to be use different key bindings or printer settings at different times. You can achieve this by having several such directories. You then specify which directory to use at run-time.

Invoking LyX with the command line switch `-userdir` *<some directory>* instructs the program to read the configuration from that directory, and not from the default directory (you can determine the default directory by running LyX without this switch as described above). If this directory does not exist, LyX offers to create it for you, just like it does for the default directory on the first time you run the program. You can modify the configuration options in this additional `Userdir` exactly as you would for the default directory. These directories are completely independent (but read on). Note that setting the environment variable `LYX_USERDIR_13x` to some value has exactly the same effect.

Having several configurations also requires more maintenance: if you want to add a new layout to `Userdir/layouts` which you want available from all your configurations, you must add it to each directory separately. You can avoid this with the following trick: after LyX creates the additional directory, most of the subdirectories (see above) are empty. If you want the new configuration to mirror an existing one, replace the empty subdirectory with a symbolic link to the matching subdirectory in the existing configuration. Take care with the `doc/` subirectory, however, since it contains a file written by the configuration script (also accessible through Tools ▷ Reconfigure 3.9) which is configuration-specific.

Chapter 3

The Preferences dialog

3.1 Using the dialog for the first time

The `UserDir/preferences` file will contain only changes that you have made to the default behaviour, some of which is hard-coded into LyX and some of which is contained in the system file `LyXDir/lyxrc.defaults`. Note that in both files lines beginning with a "#" are just comments and not interpreted. However, only system administrators should edit `LyXDir/lyxrc`. Users should use the Tools ▷ Preferences dialog to create and modify their own `UserDir/preferences` file.

We hope that the Tools ▷ Preferences dialog will be largely self-explanatory. Almost all the commands have an associated comment, so you shouldn't have too much trouble modifying it to taste. Before we highlight a few of the more important commands below, however, a word of warning: Applying some of your changes (e.g., screen fonts) will have an instant effect. Others (e.g. changing the bind file) will not. If nothing appears to have changed, just Save the changes and restart LyX.

3.2 On-screen fonts

The font used to display your documents on the LyX screen is very important, since you'll be reading all your documents with this font. Therefore it is important that the font is as readable and good-looking as possible. The LyX team tried to provide the best possible default font for you, but since practically all X11 systems are different, it's likely that the default fonts will be sub-optimal on your system. Fortunately, you can do something about this. Before we explain how to do this, you should learn a bit more about fonts so that you are better prepared for choosing your fonts, because it is a trade-off that is specific to your preferences and the capabilities of your system.

Notice that this section only deals with the fonts on the *screen* inside the LyX window. The fonts that appear on the *paper output* are independent from these fonts, and are determined by the document class. Read the *User's Guide* to learn how to change the font of the printed version of your document.

Basically, screen fonts come in two different kinds: scalable outline fonts and non-scalable bitmap fonts. This distinction seems a bit arbitrary, since non-scalable fonts are actually scalable in most modern font renderers. The difference lies in the *quality* of the scaling, and the *speed* of display. The most important decision is thus whether you should use non-scalable bitmap fonts or scalable outline fonts.

The scalable fonts are built from *outlines* of the single glyphs (i.e. characters) in the font. This means that each glyph is defined using mathematical curves that are well suited for scaling to any requested size. This mathematical definition is interpreted by the font renderer and turned into a small picture composed of pixels according to which size and glyph, the programmer requests. This means that scalable fonts will look pretty good in all sizes. Well, almost all sizes. Since scalable fonts are defined in an abstract way, it can be hard to provide a good rendering at small sizes, where each pixel has to be very carefully computed to provide a good image. Technically it is possible to do this from the mathematical definition, but in order to keep the rendering reasonably fast, tradeoffs have to be made, and the result is that scalable fonts can be difficult to read at small sizes.

Bitmap fonts on the other hand, are defined by bitmap graphics from the start, so they will look good at all the sizes they are meant for. However, they don't scale well, because in order to scale a glyph, each pixel is enlarged into several pixels. It is the same effect that happens if you try to enlarge a picture in `xv` or any other picture manipulation program. In order to relieve this effect, bitmap fonts are typically provided in several fixed sizes typically from around 8 pixels high up to 34 pixels or so high in steps according to what is believed to be useful. The advantage of bitmap fonts is that no complicated computations are necessary to display each glyph, so bitmap fonts are thus faster displayed than scalable fonts. The disadvantage is that sizes that don't exists as fixed versions have to be scaled by doubling pixels, and thus look bad.

The net result of all this, is that bitmap fonts are generally best for the small sizes, where they are available, while scalable fonts are generally best for large sizes. The logical conclusion would thus be to use bitmap fonts for the small sizes, and scalable fonts for the large sizes. Unfortunately, this is not a good idea, since bitmap fonts and scalable fonts are not designed to be used together, so the overall look of such a scheme would be bad. The best you can do is thus to try both schemes and decide for yourself what suits you.

By default, LyX uses non-scalable bitmap fonts (when using the XForms frontend). For serif fonts, *times* is used, for sans serif fonts, *helvetica* is used, while *courier* is used as the monospaced/typewriter font.

In the following, we will describe what to do if the text does not look good in LyX. We'll start with the most important parameters: DPI and font zoom.

3.2.1 DPI setting and Font Zoom

LyX automatically tries to scale the fonts to look as close as the paper output size as possible, except for the so-called font zoom factor.

In order for this to work on all systems, it relies on the screen DPI (dots per inch) setting to be correct. The DPI setting for your system is autodetected by LyX using the information the X server can provide. You can check what LyX autodetects the DPI setting to, by running LyX as `lyx -dbg 2`.

On many systems, X is not set up correctly, so you should check that it is correct by hand. Run “`xdpyinfo | more`” and write down what the DPI is for the resolution you use (this will be close to the value LyX detects). It is the number mentioned as “resolution”. Also write down the number of pixels you have in the width (the first number under “dimensions”).

Then get the good old ruler out of the closet, and measure the width of the visible screen-image on your monitor. Convert this measurement to inches if you used a centimeter ruler by dividing by 2.54. Now you can determine the correct DPI setting for your screen by dividing the number of pixels in the width by the width of the screen-image on the monitor. If this number is more than, say, 5 DPI from the detected value, you should either fix the X setup, or at least tell LyX that the DPI is different than the detected value.

If you can't fix the X setup (which of course is best since other programs than LyX will benefit from this as well), you can tell LyX the correct DPI using the Preferences dialog.

If the text is too small or too big for your taste, you should fiddle with the font zoom setting. This setting is used to scale the point size of the text. If your DPI setting is correct, and the font zoom setting is set to 100, this means that LyX will try to display the text exactly the same size as it will appear on the paper-output. If you set the zoom factor to 200, the text will try to be 2 times as big as on paper. Of course, this will only happen if LyX can find a font that has the appropriate size, which you can't count on. Since LyX is a WYSIWYM system anyways, this limitation isn't much of an issue.

The default font zoom setting is 150, since a monitor is typically wider than a piece of paper, but you should try to fiddle with it through the Font Zoom setting in the Preferences dialog to find a size that you like. When you've found a setting that seems to work nicely for you (tip: use the Apply button to keep the dialog open while you experiment), you can make this setting the default by using the Save button.

While it is often possible to find a suitable size for the text on the screen, this doesn't necessarily mean that the fonts are the best ones available on your system. In order to help you get the most out of your system, you can use the font definition commands to fine-tune the look of the text in greater detail than merely size.

3.2.2 Font definition commands

As mentioned, LyX uses non-scalable bitmap fonts by default with the XForms frontend. For serif fonts, *times* is used, for sans serif fonts, *helvetica* is used, while *courier* is used as the monospaced/typewriter font.

You can change all of these from within the Preferences dialog. The number of fonts that are available on different systems vary, but the program `xfontsel` should be available everywhere. Use that program to find candidate fonts. When you've found a font that you like, try to insert the first two elements of the name (called "fndry" and "fmly" in `xfontsel`) in the appropriate field in the Preferences dialog and press Apply. LyX will then reformat your document using the new font, and if you like the font, you should Save it. One place to start for a new font is to see if the scalable font "utopia" is available. Tip: You can see whether a font is a bitmap font or a scalable font by checking the "resx" or "resy" fields in `xfontsel`. If the value 0 is available, the font is scalable. If the value 0 isn't available, the font is a bitmap font.

Before you go about scrapping a bitmap font because the larger sizes look "blocky", you should toggle the "Use scalable fonts" button. This is only useful if you use bitmap fonts, because only these don't scale well. If you define this flag, LyX will only use the fixed font sizes that are available, and this guarantees that all bitmap fonts look well. (You can see which individual font sizes are available with the `xlsfonts` command. Try `man xlsfonts`.) However, the prize is that the difference between the size of the fonts on screen and the size of fonts on paper will be larger because LyX will have to be satisfied with the closest available size, and not try to scale a size to fit. Also, you can risk that some logically different sizes, such as Large and Larger, will be mapped to the same screen font, making it hard for you to see the difference on screen. We've decided not to use scalable fonts by default because of these artifacts, but since LyX is a WYSIWYM system, many people like to use the flag anyways, well-knowing that the font size on the screen can't be trusted. But remember that this flag only makes a difference when you use bitmap fonts. Scalable fonts won't be affected for reasons you should understand by now.

One final note regarding this flag: you should know that there is nothing wrong with using bitmap and scalable fonts at the same time for different purposes. For instance, it's common to use the scalable "Utopia" for the serif text together with a bitmap version of "Helvetica". And you can safely select the "Use scalable fonts" button without worries: It will only apply to the Helvetica font.

Sometimes the artifacts introduced by use of the flag can be relieved by using the fine-detail screen font sizes which defines which point sizes the different logical font sizes correspond to. Run LyX as `lyx -dbg 513` to see exactly what concrete fonts the logical sizes map to, and try adjusting the corresponding entries in the Preferences dialog until you've managed to hit the nail and get the fonts you want. This can be hard to do, because LyX uses the DPI setting and the font zoom settings to calculate which exact screen font size to ask the X server for, thus obfuscating the mapping. If you can't make it by trial-and-error, you can make the process more transparent if you set both the DPI setting and font zoom settings to 100—even when this is known to be wrong. This will of course make your scalable fonts look weird, so use with care.

3.2.3 Font encoding

By default, LyX will use fonts meant to write Western European text, including all kinds of English. This is defined through the so-called *font encoding.* If you want to use LyX to write for instance Eastern European text, Cyrillic or any other language not covered by the ISO-8859-1 font encoding, you can define a different one with the encoding setting. This requires you to have special fonts installed. You can use `xfontsel` to see whether this is the case: check the "rgstry" and "encdng" fields for ISO-8859-X values different from ISO-8859-1, and search for one that contains the national characters of your language. If you find any, enter this encoding in the dialog. If not, go searching the Web for appropriate fonts. For the Qt frontend, it's recommended you use an iso646 font set.

When you've set LyX up to use a different font encoding, you should also consider changing the font used by dialog windows in LyX. For instance, the Table of Contents dialog will not be understandable unless you tell LyX to use a different font for this. By default the menu font is set to `-*-helvetica-medium-r`, but often Helvetica is not available in the font encoding you need, so the dialog allows this to be changed.

As you can see, there are quite a few options that can be used to fine tune the look of your fonts. This should not scare you from fiddling with the settings, because after all, you will hopefully be using LyX for many hours in the future. And contrary to real WYSIWYG word processors where you are tied to using fonts that have to look good both on paper and on screen, LyX gives you the possibility of using fonts that are designed to look good on the screen while using a different set of fonts to look good on paper.

3.3 Bindings

Bindings are used to, well, bind a function to a key. Several prepackaged binding files are available: a CUA set of bindings (familiar as the typical set of PC and CDE set of keyboard shortcuts), an Emacs set of bindings, for those of us who follow the One True Way and refuse to lower our standards,[1] as well as specialty bindings (broadway and hollywood) and other languages (French, German, etc.).

If, however, you'd like to customise the keybindings to your own exacting tastes, then copy the best-fit file in `LyXDir/bind/` to your own `UserDir/bind/` and modify that. Don't forget to load this new file into LyX using the Preferences dialog. (For the moment you'll have to restart LyX for these changes to take effect.)

[1] I'm kidding here, of course!

LyX supports internationalization of the user interface (see Chapter 4). If your *locale* is set, with the environment variable `$LANG`, LyX will try to use bindfiles by prepending `$LANG_` to their name. For example, you can put a translated copy of some standard bind file in your personal `bind/` directory, and LyX will use it automatically.

The syntax of the `.bind` files is straightforward:

```
\bind <key combination> <lyx-function>
```

Both key combination and lyx-function (including any arguments) must be enclosed in "double quotes". All the LyX functions are listed in the *Reference Guide*.

3.4 User Interface

The appearance of both the menu and toolbar may both be changed using the Preferences dialog. Simply change the `.ui` file in `LyXDir/ui/`. For the moment, only one file exists, `default.ui`, but feel free to experiment. Just copy the file to the `UserDir/ui/` directory and play! Note that, for the moment, you'll have to restart LyX for these changes to take effect.

The syntax of the `.ui` files is straightforward: have a look at `default.ui. The Menubar`, `Menu` and `Toolbar` entries must be ended with an explicit `End`. They may contain `Submenus`, `Items`, `OptItems`, `Separators`, `Icons` and in the case of the "file" menus, a `Lastfiles` entry. One small word of warning. `Submenus` may be inserted in a `Menubar` or `Menu`, but they are defined as `Menus`, not as `Submenus`.

3.5 Converters, Formats, Viewers, Editors and Copiers

LyX has a powerful mechanism to convert to and from any file format using external programs. Define a pair of formats, e.g. `LaTeX` and `PDF`. Now define a converter from one format to the other. In our example, two possible mechanisms exist.

1. A direct conversion, from LaTeX to PDF using pdflatex
2. A more convoluted route using intermediate formats and converters: LaTeX to DVI (using latex) to PostScript® (using dvips) to PDF (using ps2pdf).

LyX will always choose the shortest possible route, so you must specify two different Format names for `.pdf` files to be able to use either. Both are included by default in the Preferences dialog. Have a look and then invent your own!

Moreover, each Format can have a Viewer associated with it. For example, you might want to use `ghostview` to examine PostScript® files, or `xdvi` to preview the LaTeX output. You can alter the viewer to use (and what options to pass to it) via the Tools ▷ Preferences:Conversion dialog. For example, to change the dvi viewer, select the DVI format in the dialog, change the viewer to be kdvi (or whatever), and hit Modify.

If the operating system has a default viewer associated to a format it is used instead of the one you can define via the Tools ▷ Preferences:Conversion dialog. This does currently only work in the Windows® port of LyX, but it is planned to implement this feature on all other ports that can support it, too.

Editors are like viewers: Each Format can have an Editor associated to it, and they can be altered via the Tools ▷ Preferences:Conversion dialog. LyX uses them whenever an included file[2] needs to be edited.

Finally, each Format can have a Copier associated to it. Since all conversions from one Format to another take place in a temporary directory, it is sometimes necessary to modify a file before copying it to the temporary directory[3]. This is done by the Copier: It copies a file to (or from) the temporary directory and may modify it in the process.

3.6 BibTEX and makeindex

Both the bibliography generating command (default bibtex) and the index generating command (default makeindex with options -c and -q) can be changed. As an alternative for makeindex, xindy can be recommended.

The command to enter is

```
makeindex.sh -m $$lang
```

where the placeholder $$lang will be replaced by the chosen document (babel) language. For this, you must

have installed the packages xindy and make-rules (xindy-make-rules). Type makeindex.sh at a shell prompt for a help page.

3.7 ASCII export options

There are a couple of commands that can be used to "clean up" exported ASCII text files. Note that LyX automatically detects and uses the best settings for your system at installation time, but you can modify them if you disagree with its interpretation.

ASCII roff This option defines the command used to produce better ASCII tables with the groff/troff/nroff UNIX-commands (refer to their manpages for more information about them). Setting this as empty tells LyX to use the internal (inferior) formatter.

[2] This can be an included .tex file, a verbatim included text file, external material or an included graphics file.

[3] For example, the file may reference other files with relative filenames, which will become invalid in the temporary directory

`ASCII line length` With this command you can set the default line length of the ASCII output file. Setting it to 0 means endless lines.

3.8 Printer

There are a bunch of configuration options that are used for interaction with the external print command from LyX. Normally the defaults are fine: if, however, your print command takes different option names, you can modify them here.

3.8.1 Changing Colors

You can change the colors used by LyX on-screen using the new Preferences dialog. Alternatively, if you're feeling particularly perverse you could use the `set-color` bindable function (see the *Reference Guide*). Input would have the format:

`set-color LyXName X11Color`

Here is a (partial) list of the functions and default colors:

LyX Name	Purpose	Default Color (X11)
background	text background	black
foreground	text foreground	linen
latex	LaTeX code	DarkRed
math	Mathed formulae	DarkBlue
mathline	fraction Lines, brackets, etc.	Blue
mathbg		AntiqueWhite
mathframe		Magenta
mathcursor		black
selection	selection background	LightBlue

3.9 The autodetected settings

There are several items that are detected for you when you run Tools ▷ Reconfigure. In this section, we list those which pertain to the user preferences.

`\ascii_roff_command` uses either `groff` or `nroff`+`tbl`, depending on what is available.

`\chktex_command` is set to `chktex` plus a bunch of options.

`\print_spool_command` is set to `lp` on systems (so-called System V) who have this command, and `lpr` otherwise (BSD systems).

`\print_spool_printerprefix` is set to `-d` or `-P`, depending on whether `lp` or `lpr` was found.

`\font_encoding` is set to `T1` if the `ec` fonts are found and LaTeX has support for these fonts built-in. You can set it manually if you only have the so-called `dc` fonts.

3.10 The rest

There are many other configuration options that can be used to customize LYX behavior. We still need to document them here, but again, most should be fairly obvious. Please ask on the mailing lists if you need some more information; it may even prompt us to expand this section.

Chapter 4

Internationalizing LyX

LyX supports using a translated interface. Last time we checked, LyX provided text in 14 languages together with the default English text. The language of choice is called your *locale*. (For further reading on locale settings, see also the documentation for locale that comes with your operating system. For Linux, the manual page for locale(5) could be a good place to start).

Notice that these translations will work, but do contain a few flaws. In particular, all dialogs have been designed with the English text in mind, which means that some of the translated text will be too large to fit within the space allocated. This is only a display problem and will not cause any harm. Also, you will find that some of the translations do not define short-cut keys for everything. Sometimes, there are simply not enough free letters to do it. Other times, the translator just hasn't got around to doing it yet. Our localization team – which you may wish to join – will try to fix these shortcomings in future versions of LyX.

4.1 Selecting an alternative language for the user interface

This feature is disabled by default, meaning that system default language will be used. To enable an alternative language, you have to set an appropriate environment variable. Use `"setenv LANG xx"` for csh class shells or `"export LANG=xx"` for sh class shells. Substitute the `xx` with the two letter code (or four letter code, like `en_GB` for British English) for the language you want. For instance, `no` is Norwegian. Besides the user interface texts being translated, also the appropriate manuals will be presented under the Help menu – if available.

On some systems, you may have to redefine `LC_ALL` or `LC_MESSAGES` instead of `LANG`, to override the system settings; their preference is in this order[1], which

[1] The shell variable LANGUAGE has been disabled in LyX for technical reasons. Don't use it.

corresponds to the way GNU `gettext` does it. Consult your system documentation. Normally, you'll want to put the appropriate line in a shell script run on start-up, so that the translation is on by default. Remember that this affects *all* localized packages, not only LYX!

If LYX is configured and compiled with "`--disable-nls`", this mechanism will not work.

4.2 Translating LYX

4.2.1 Translating the graphical user interface (text messages).

LYX uses the GNU `gettext` library to handle the internationalization of the interface. To have LYX speak your favorite language in all menus and dialogs, you need a `po`-file for that language. When this is available, you'll have to generate a mo-file from it and install the `mo`-file. The process of doing all of this is explained in the documentation for GNU `gettext`, but in short, this is what you do (`xx` denotes the language code):

- Copy `LYX-SOURCE-DIR/po/lyx.pot` to `xx.po` (if `lyx.pot` doesn't exist, it can be remade with `make lyx.pot` in that directory, or you can use an existing po-file for some other language as a template).
- Edit `xx.po`[2]. For some menu- and widget-labels, there are also shortcut keys that should be translated. Those keys are marked after a '|', and should be translated according to the words and phrases of the `xx`-language. There is a tool named `scgen.pl` written in Prolog in `LYX-SOURCE-DIR/development/tools/` that may be useful to help determine short-cut keys. Note that XForms (version 0.86 at least) can't handle anything but 7-bit characters as shortcut keys. You should also fill also out the information at the beginning of the new `po`-file with your email-address, etc., so people know where to reach you with suggestions and entertaining flames.
- Generate `xx.mo`. This can be done with `msgfmt -o xx.mo < xx.po`
- Copy the `mo`-file to your locale-tree, at the correct directory for application messages for the language `xx`, and under the name `lyx.mo` (e.g. `/usr/local/share/locale/xx/LC_MESSAGES/lyx.mo`)

Adding a new po-file to the *distribution* of LYX involves altering the configure scripts and more, but the way `gettext` works, you don't actually need the source-code of LYX to translate it—having `lyx.pot` (or an existing `po`-file) and the `gettext` tools suffices.

[2] We recommend that you use Emacs to do this, since the `gettext` distribution includes a nice mode that supports you in doing this.

If you've written a translation file for a language that LYX does not currently support, feel free to submit it for inclusion by sending a patch. In this case, we recommend that you read the `README` provided in the `LYX-SOURCE-DIR/po/` directory for more instructions.

4.2.1.1 Ambigous messages

Sometimes it turns out that one english message needs to be translated into different messages in the target language. One example is the message `To` which has the german translation `Nach` or `Bis`. `gettext` does not handle such ambigous translations. Therefore you have to add some context information to the message: Instead of `To` it becomes `To[[as in 'From format x to format y']]` and `To[[as in 'From page x to page y']]`. Now the two occurences of `To` are different for `gettext` and can be translated correctly to `Nach` and `Bis`, respectively.

Of course the context information needs to be stripped off the original message when no translation is used. Therefore you have to put it in double square brackets at the end of the message (see the example above). The translation mechanism of LYX ensures that everything in double square brackets at the end of messages is removed before displaying the message.

4.2.2 Translating the documentation.

The online documentation (in the Help-menu) can (and should!) be translated. If there are translated versions of the documentation available[3], and the locale is set accordingly, these will be used automagically by LYX. LYX looks for translated versions as `LyXDir/doc/xx_DocName.lyx`, where `xx` denotes the language as set by the environmental variable `$LANG`. If there are none, the default English versions will be displayed. Note that the translated versions must have the same filenames (`DocName` above) as the original. If you feel up to translating the documentation (an excellent way to proof-read the original documentation BTW!), there are a few things you should do right away:

- Read `DocStyle.lyx`, the guide to writing LYX documentation. Pay special attention to the translator's section.
- Check out the documentation translation web page at The LyX Developer's Web Site `http://www.devel.lyx.org`. That way, you can find out which (if any) documents have already been translated into your language. You can also find out who (if anyone) is organizing the effort to translate the documentation into your language. If no one is organizing the effort, please let us know that you're interested.

Once you get to actually translating, here's a few hints for you that may save you trouble:

[3] As of February 2003, almost all of the docs have been translated into German and French. The *Tutorial* has been translated into at least 12 other languages, with other translations in progress. The library of translated documents is growing rapidly.

- Join the documentation team! There is information on how to do that in `Intro.lyx` (Help ▷ Introduction), which by the way is the first document you should translate.
- Learn the typographic conventions for the language you are translating to. Typography is an ancient art and over the centuries, a great variety of conventions have developed throughout different parts of the world. Also study the professional terminology amongst typographers in your country. Inventing your own terminology will only confuse the users. *(Warning! Typography is addictive!)*
- Make a copy of the document. This will be your working copy. You can use this as your personal translated help-file by placing it in your `~/.lyx/doc/`-directory.
- Sometimes the original document (from the LyX-team) will be updated. Use the ViewCVS tool available at `http://www.lyx.org/viewcvs.cgi/lyxdoc/` to see what has been changed[4]. That way you can easily see which parts of the translated document need to be updated.
- If you ever find an error in the original document, fix it and notify the rest of the documentation team of the changes! (You didn't forget to join the documentation team did you?)

4.3 International Keyboard Support

[Editor's Note: The following section is by Ivan Schreter. It needs to be fixed to conform to the new Documentation Style sheet and to make use of the new v1.0 features. The whole thing also needs to be merged with the section following it.-jw]

4.3.1 Defining Own Keymaps: Keymap File Format

Let's look at a keyboard definition file a little closer. It is a plain ASCII file defining

- key-to-key or key-to-string translations
- dead keys
- dead keys exceptions

To define key-to-key or key-to-string translation, use this command:

```
\kmap key outstring
```

[4] Alternatively, you can keep a copy of the latest version of the English document which you've translated.

where `key` is the key to be translated and `outstring` is the string to be inserted into the document. To define dead keys, use:

```
\kmod key deadkey
```

where `key` is keyboard key and `deadkey` is dead key name. The following dead keys are supported (shortcut name is in parentheses):

Name	*Example*
acute (acu)	áéíóú
grave (gra)	àèìòù
macron (mac)	ō
tilde (til)	ñÑ̃
underbar (underb)	o̱
cedilla (ced)	çÇ
underdot (underd)	ọ
circumflex (circu)	âêîôû
circle (circl)	ÅůŮ
tie (tie)	o͡
breve (bre)	ăŏ
caron (car)	čšž
hungarian umlaut (hug)	őű
umlaut (uml)	äöü
dot (dot)	żṡ

Since in many international keyboards there are exceptions to what some dead keys should do, you can define them using

```
\kxmod deadkey key outstring
```

For example, on Slovak keyboard, if you enter caron-o, it generates circumflex-o, so you put in

```
\kxmod caron o "\^o"
```

to make it work correctly. Also, you have to define as exceptions dead keys over i and j, to remove the dot from them before inserting an accent mark. I will change this when the time comes, but so far I haven't had time.

Oh, and about characters: backslash is escaped, so to enter it, you'll need double backslash. Also, quotes and `#` have different meaning. `#` marks comments, quotes start and end LaTeX-style commands. To enter quote, you'll need to use `\"`, to enter `#`, use `\#`.

If you make a keyboard description file that works for your language, please mail it to me, so I can include it in the next keymap distribution.

More keywords will be supported in keymap configuration file in future, like

- `\kinclude filename` `include` another file
- `\kprog program` `define` an external keymap translation program

Also, it should look into `lyxrc` file for defaults, too (for example, a `\kinclude` option to include default keyboard).

4.4 International Keymap Stuff

The next two sections describe the `.kmap` and `.cdef` file syntax in detail. These sections should help you design your own key map if the ones provided do not meet your needs.

4.4.1 The .kmap File

A .`kmap` file maps keystrokes to characters or strings. As the name suggests it sets a keyboard mapping. The `.kmap` file keywords `kmap,kmod,ksmod`, and `kcomb` are described in this section.

`kmap` Map a character to a string

`\kmap` *char string*

This will map *char* to *string.* Note that in *string*, the double-quote (") and the backslash (\) must be escaped with a preceding backslash (\).

An example of a kmap statement to cause the symbol / to be output for the keystroke & is:

```
\kmap & /
```

`kmod` Specify an accent character

`\kmod` *char accent allowed*

This will make the character *char be an accent* on the *allowed* character(s). This is the dead key[5] mechanism.

If you hit *char* and then another key not in *allowed*, you will get a *char* followed by the other, unallowed key, as output. Note that a Backspace cancels a dead key, so if you hit *char Backspace*, the cursor will not go one position backwards but will instead cancel the effect that *char* might have had on the next keystroke.

The following example specifies that the character ' is to be an acute accent, allowed on the characters a, e, i, o, u, A, E, I, O, and U:

```
\kmod ' acute aeiouAEIOU
```

`ksmod` Specify an exception to the accent character

`\kxmod` *accent char result*

[5] The term *dead key* refers to a key that does not produce a character by itself, but when followed with another key, produces the desired accent character. For example, a German character with an umlaut like *ä* can be produced in this manner.

This defines an exception for *accent* on *char*. The *accent* must have been assigned a keystroke with a previous `\kmod` declaration and *char* must not belong in the *allowed* set of *accent*. When you enter the *accent char* sequence, *result* is produced. If such a declaration does not exist in the `.kmap` file and you enter *accent char*, you get *accent_ key char* where *accent_ key* is the first argument of the `\kmod` declaration.

The following command produces causes äi to be produced when you enter acute-i ('i):

```
\kxmod acute i "\\'{\\i}"
```

`kcomb` Combine two accent characters

`\kcomb` *accent1 accent2 allowed*

This one is getting pretty esoteric. It allows you to combine the effect of *accent1* and *accent2* (in that order!) on *allowed* chars. The keystrokes for *accent1* and *accent2* must have been set with a `\kmod` command at a *previous* point in the file.

Consider this example from the `greek.kmap` file:

```
\kmod ; acute aeioyvhAEIOYVH \kmod : umlaut iyIY \kcomb acute umlaut iyIY
```

This allows you to press ;:i and get the effect of `\'{\"{i}}`. A backspace in this case cancels the last dead key, so if you press ;: Backspace i you get `\'{i}`.

4.4.2 The .cdef File

After the `.kmap` mapping is performed, a `.cdef` file maps the strings that the symbols generate to characters in the current font. The LyX distribution currently includes at least the `iso8859-1.cdef` and `iso8859-2.cdef` files.

In general the `.cdef` file is a sequence of declarations of the form

char_ index_ in_ set string

For example, in order to map \'`{e}` to the corresponding character in the iso-8859-1 set (233), the following declaration is used

```
233 "\\'{e}"
```

with \ and `"` being escaped in *string*. Note that the same character can apply to more than one string. In the `iso-8859-7.cdef` file you have

```
192 "\\'{\\\"{i}}"
192 "\\\"{\\'{i}}"
```

If LyX cannot find a mapping for the string produced by the keystroke or a deadkey sequence, it will check if it looks like an accented char and try to draw an accent over the character on screen.

4.4.3 Dead Keys

There is a second way to add support for international characters through so-called dead-keys. A dead-key works in combination with a letter to produce an accented character. Here, we'll explain how to create a really simple dead-key to illustrate how they work.

Suppose you happen to need the circumflex character, "ˆ". You could bind the ^-key [a.k.a. Shift-6] to the LyX command `accent-circumflex` in your `lyxrc` file. Now, whenever you type the ^-key followed by a letter, that letter will have a circumflex accent on it. For example, the sequence "^e" produces the letter: "ê". If you tried to type "^t", however, LyX will complain with a beep, since a "t" never takes a circumflex accent. Hitting Space after a dead-key produces the bare-accent. Please note this last point! If you bind a key to a dead-key, you'll need to rebind the character on that key to yet another key. Binding the ,-key to a cedilla is a bad idea, since you'll only get cedillas instead of commas.

One common way to bind dead-keys is to use Meta-, Ctrl-, and Shift- in combination with an accent, like "˜" or "," or "^". Another way involves using `xmodmap` and `xkeycaps` [remember them from section?] to set up the special `Mode_Switch` key. The `Mode_Switch` acts in some ways just like Shift and permits you to bind keys to accented characters. You can also turn keys into dead-keys by binding them to something like `usldead_cedilla` and then binding this symbolic key to the corresponding LyX command.[6] You can make just about anything into the `Mode_Switch` key: One of the Ctrl- keys, a spare function key, etc. As for the LyX commands that produce accents, check the entry for `accent-acute` in the *Reference Manual.* You'll find the complete list there.

4.4.4 Saving your Language Configuration

You can edit your preferences so that your desired language environment is automatically configured when LyX starts up, via the Edit ▷ Preferences dialog.

[6] Note from John Weiss: This is exactly what I do in my `~/.lyx/lyxrc` and my `~/.xmodmap` files. I have my Scroll Lock key set up as `Mode_Shift` and a bunch of these "`usldead_*`" symbolic keys bound such things as Scroll Lock-^ and Scroll Lock-˜. This is how I produce my accented characters.

Chapter 5

Installing New Document Classes, Layouts, and Templates

In this chapter, we describe the procedures for creating and installing new LyX layout and template files, as well as offer a refresher on correctly installing new LaTeX document classes. Some definitions: a document class is a LaTeX file (usually ending in `.cls` or `.sty`) which describes the format of a document such as an article, report, journal preprint, etc. and all the commands needed to realize that format. A layout file is a LyX file which corresponds to a LaTeX document class and which tells LyX how to "draw" things on the screen to make the display look something like the final printed page. More precisely, a layout file describes a "text class" which is the internal construct LyX uses to render the screen display. "Layout" and "text class" can be used somewhat interchangeably, but it is better to refer to the file as the layout, and the thing living in LyX's memory as the text class. A template file is simply a LyX document which contains a set of predefined entries for a given document class which are generally required for that class. Templates are especially useful for things like journal manuscripts which are to be submitted electronically.

5.1 Installing a new LaTeX package

Some installations may not include a LaTeX package that you would like to use within LyX. For example, you might need FoilTeX, a common (and very powerful) package for preparing slides or viewgraphs for overhead projectors. Here are the formal steps involved in getting the package up and running if you are using teTeX or some other web2c based distribution.

1. Get the package from CTAN or wherever.[1]

[1] See the *Inventory of your LaTeX configuration* manual for details of what CTAN is and

2. Read the file `texmf.cnf` (this usually lives in the directory `$TEXMF/web2c`, though you can run `kpsewhich texmf.cnf` to locate it). It describes how to add a local `texmf` directory; follow the instructions. You need to insert the name of your local `texmf` directory in `texmf.cnf`. Under Linux, `/usr/local` is a logical place to install software that did not come with your distribution, so you might use `/usr/local/texmf`. Usually, you will have to modify only two things:

   (a) Set `TEXMFLOCAL` to the directory you chose; e.g.
   `TEXMFLOCAL = /usr/local/texmf`

   (b) Make sure `TEXMF` includes the `TEXMFLOCAL` variable; e.g.
   `TEXMF = {$HOMETEXMF,!!$TEXMFLOCAL,!!$TEXMFMAIN}`

3. Create your local `texmf` directory (e.g. `/usr/local/texmf`). You must follow the directory structure of your existing `texmf` directory (for example, latex packages should go under `/usr/local/texmf/tex/latex/`).

4. Install the package. For example, you would unpack the FoilTEX tarball and create `/usr/local/texmf/tex/latex/foiltex`. The `foiltex` directory contains various files.

5. Run: `texhash`. This should create `/usr/local/texmf/ls-R` amongst others.

6. From within LyX, do: Tools ▷ Reconfigure. Restart LyX.

Now you should see your new package—for example slides (FoilTEX)—under Layout ▷ Document, field Class. Note that there are simpler ways of installing packages: you can add a link to the new package directory in the system LaTeX directory (`$TEXMF/tex/latex`, don't forget to then run `texconfig`), or sometimes simply set the `$TEXINPUTS` environment variable to include the new package. However, the formal procedure described in `texmf.cnf` is guaranteed to work, so you should follow it unless circumstances absolutely prevent it: such as, when you don't have superuser access.

5.2 Layouts

This section describes how to write and install your own LyX layout files (also known as text classes) and walks through the `article` text class format as an example. The `.layout` files describe what paragraph styles are available for a given document class and how LyX should display them. We try to provide a thorough description of the process here; however, there are so many different types of documents supported by LaTeX classes we can't hope to cover every different possibility or problem you might encounter.

where supported document classes can be found.

When you plan to write a new layout, it is extremely helpful to look at the example layouts distributed with LyX. If you use a nice LaTeX document class that might be of interest for others, too, and have a nice corresponding LyX layout, feel free to contribute the stuff to us, so we may put it into the distribution.

All the tags described in this chapter are case-insensitive; this means that `Style`, `style` and `StYlE` are really the same command. The possible values are printed in brackets after the feature's name. The default value if a feature isn't specified inside a text class-description is typeset ***`emphasized`***. If the argument has a datatype like "string" or "float", the default is shown like this: `float=`***`default`***.

5.2.1 Supporting new document classes

There are two situations you are likely to encounter when wanting to support a new LaTeX document class, involving LaTeX 2$_\varepsilon$ class (`.cls`) and style (`.sty`) files.

5.2.2 A layout for a sty file

If your new document class is provided as a style file that is used in conjunction with an existing, supported document class, start by copying the existing class's layout file into your local directory. For the sake of example we'll assume that the style file is called myclass.sty and it is meant to be used with report.cls which is a standard class.

```
cp report.layout ~/.lyx/layouts/myclass.layout
```

Then edit `myclass.layout` and change the line:

```
\DeclareLaTeXClass{report}
```

to read

```
\DeclareLaTeXClass[report, myclass.sty]{report (myclass)}
```

Then add:

```
Preamble
    \usepackage{myclass}
EndPreamble
```

near the top of the file.

Start LyX and select Tools ▷ Reconfigure. Restart LyX and try creating a new document. You should see "report (myclass)" as a document class option in the Document ▷ Settings dialog. It is likely that some of the sectioning commands and such will differ from how the base class[2] works, so you can fiddle around with the settings for the different sections if you wish. See below for more discussion on this.

5.2.3 Layout for a cls file

In this case, you will probably have to "roll your own" layout. We strongly suggest copying an existing layout file which uses a similar LaTeX class and modifying it if at all possible. At least use an existing file as a starting point so you can find out what items you need to worry about. Again, the specifics are covered below.

5.3 Declaring a new text class

When it's finally time to get your hands dirty and create or edit your own layout file, the following sections describe what you're up against. Our advice is to go slowly, save and test often, listen to soothing music, and enjoy one or two of your favorite adult beverages; more if you are getting particularly stuck. It's really not that hard, except that the multitude of options can become overwhelming if you try to do to much in one sitting. Go have another adult beverage, just for good measure.

Here we go!

Lines in a layout file which begin with a # are comments. There is one exception to this rule: all layouts should begin with lines like:

```
#% Do not delete the line below; configure depends on this
#  \DeclareLaTeXClass{article}
```

The second line is used when you configure LyX. The layout file is read by the LaTeX script `chkconfig.ltx`, in a special mode where # is ignored. The first line is just a LaTeX comment, and the second one contains the declaration of the text class. If these lines appear in a file named `article.layout`, then they define a text class of name `article` (the name of the layout file) which uses the LaTeX document class `article.cls` (the default is to use the same name as the layout). The string "article" that appears above is used as a description of the text class in the Document ▷ Settings dialog.

Let's assume that you wrote your own text class that uses the `article.cls` documentclass, but where you changed the appearance of the section headings. If you put it in a file `myarticle.layout`, the header of this file should be:

```
#% Do not delete the line below; configure depends on this
#  \DeclareLaTeXClass[article]{article (with my own headings)}
```

This declares a text class `myarticle`, associated with the LaTeX document class `article.cls` and described as "article (with my own headings)". If your text class depends on several packages, you can declare it as:

[2] `report` in this example

```
#% Do not delete the line below; configure depends on this
#  \DeclareLaTeXClass[article,foo.sty]{article (with my own headings)}
```

This indicates that your text class uses the foo.sty package. Finally, it is also possible to declare classes for SGML and DocBook code. Typical declarations will look like

```
#% Do not delete the line below; configure depends on this
# \DeclareSGMLClass{SGML (LinuxDoc)}
```

or

```
#% Do not delete the line below; configure depends on this
# \DeclareDocBookClass[article]{SGML (DocBook article)}
```

Note that these declarations can also be given an optional parameter declaring the name of the document class (but not a list).

When the text class has been modified to your taste, all you have to do is to copy it either in `$LyXDir/layouts/` or in `$UserDir/layouts` and run Tools ▷ Reconfigure. Exit LyX and restart it; then your new text class should be available along with the others.

5.3.1 File format

The first non-comment line must contain the file format number:

`Format` [`int`] This tag was introduced with LyX 1.4.0 (layout files of LyX 1.3.x and earlier don't have an explicit file format). The file format that is documented here is 2.

5.3.2 General text class parameters

These are the general parameters which describe the form of the entire document:

`Columns` [*1* , 2] Whether the class-default should have one or two columns. Can be changed in the Document ▷ Settings dialog. This setting (same goes for `Sides`, too) is important: if your text class has two columns by default but you forget to set it correctly, the `twocolumn` LaTeX option will *not* be output when you select Two columns in Document ▷ Settings.

`Sides` [*1* , 2] Whether the class-default should be printing on one or both sides of the paper. Can be changed in the Document ▷ Settings dialog.

`PageStyle` [*`plain`* , `empty`, `headings`] The class default pagestyle. Can be changed in the Document ▷ Settings dialog.

`ClassOptions...End` This section describes various global options supported by the document class. See Section 5.3.3 for a description.

`ProvidesAmsmath` [*0*, `1`] Whether the class already loads the `amsmath` package. This is the case of the `amsart` and `amsbook` document classes.

`ProvidesMakeidx` [*0*, `1`] Whether the class already provides the functionality of the `makeidx` package. This is the case of the `amsart` and `amsbook` document classes.

`ProvidesUrl` [*0*, `1`] Whether the class already provides the functionality of the `url` package. This is the case of the `AASTeX` document class.

`DefaultFont` This is used to describe the default font of the document. See Section 5.3.8 for a description.

`DefaultStyle` [`string`] This is the style that will be assigned to new paragraphs, usually Standard. This will default to the first defined style if not given, but you are highly encouraged to use this directive.

`TitleLatexType` [*CommandAfter*, `Environment`] Indicates what kind of markup is used to define the title of a document. `CommandAfter` means that the macro with name `TitleLaTeXCommand` will be inserted after the last layout which has "`InTitle 1`". `Environment` corresponds to the case where the block of paragraphs which have "`InTitle 1`" should be enclosed into the `TitleLaTeXCommand` environment.

`TitleLatexCommand` [`string="maketitle"`] The name of the command/environment mentionned above.

`Preamble...EndPreamble` A set of macro definitions that will be output at the beginning of the LaTeX files. Use this for global definitions.

`Input` As its name implies, this command allows you to include another layout definition file within yours to avoid duplicating commands. Common examples are the standard layout files, for example, `stdclass.inc`, which contains most of the basic layouts.

`Style...End` This sequence defines a new style. If the style already exists, it will redefine some of its parameters instead. See Section 5.3.4 for details.

`NoStyle` This command deletes an existing style. This is particularly useful when you want to suppress a style that has be defined in an input file.

`Float...End` This sequence defines a new float. See Section 5.3.5 for details.

`NoFloat` This command deletes an existing float. This is particularly useful when you want to suppress a float that has be defined in an input file.

`CharStyle...End` This section defines a new character style. See Section 5.3.6 for a description.

`Counter...End` This sequence defines a new counter. See Section 5.3.7 for details.

5.3.3 `ClassOptions` section

The `ClassOptions` section can contain the following entries:

`FontSize` [`string="10|11|12"`] The list of available font sizes for the document's main font, separated by "`|`".

`PageStyle` [`string="empty|plain|headings|fancy"`] The list of available page styles, separated by "`|`".

`Other` [`string=""`] Some document class options, separated by a comma, that will be added to the optional part of the `\documentclass` command.

5.3.4 Specific Paragraph Layouts

A paragraph layout description looks like this[3]:

```
Style name
 ...
End
```

where the following commands are allowed:

`CopyStyle [string]` This is used to copy all the features of an existing layout into the current one.

`LatexType` [*`Paragraph`*, `Command`, `Environment`, `Item_Environment`, `List_Environment`] How the layout should be translated into LaTeX. `Paragraph` means nothing special. `Command` means `\`*`LatexName`*`{...}` and `Environment` means `\begin{`*`LatexName`*`}...\end{`*`LatexName`*`}`. `Item_Environment` is the same as `Environment`, except that a `\item` is generated for each paragraph of this environment. `List_Environment` is the same as `Item_Environment`, except that `LabelWidthString` is passed as an argument to the environment. `LabelWidthString` can be defined in the Layout ▷ Paragraph dialog. *`LatexType`* is perhaps a bit misleading, since these rules apply to SGML classes, too. Visit the SGML class files for specific examples.

`InTitle [1, 0]` If 1, marks the layout as being part of a title block (see also the `TitleLatexType` and `TitleLatexCommand` global entries)

`LatexName` The name of the corresponding LaTeX stuff. Either the environment or command name.

[3] Note that this will either define a new layout or modify an existing one.

`LatexParam` The optional parameter for the corresponding `LatexName` stuff. This parameter cannot be changed from within LyX.

`OptionalArgs` [`int=0`] The number of optional arguments that can be used with this layout. This is useful for things like section headings, and only makes sense with LaTeX.

`Margin` [*`Static`*`, Manual, Dynamic, First_Dynamic, Right_Address_Box`] The kind of margin that the layout has on the left side. `Static` just means a fixed margin. `Manual` means that the left margin depends on the string entered in the Edit ▷ Paragraph Settings dialog. This is used to typeset nice lists without tabulators. `Dynamic` means that the margin depends on the size of the label. This is used for automatic enumerated headlines. It is obvious that the headline "5.4.3.2.1 Very long headline" must have a wider left margin (as wide as "5.4.3.2.1" plus the space) than "3.2 Very long headline", even if other word processors are not able to do this. `First_Dynamic` is similar, but only the very first row of the paragraph is dynamic, while the others are static; this is used, for example, for descriptions. `Right_Address_Box` means the margin is chosen in a way that the longest row of this paragraph fits to the right margin. This is used to typeset an address on the right edge of the page.

`NextNoIndent` [`1,` *`0`*] Whether the following Paragraph is allowed to indent its very first row. `1` means that it is not allowed to do so, `0` means it could do so if it wants to.

`ParIndent` [`string=""`] The indent of the very first line of a paragraph. The argument is passed as a string. For example `"MM"` means that the paragraph is indented with the width of `"MM"` in `the` normal font. You can get a negative width by prefixing the string with `"-"`. This way was chosen so that the look is the same with each used screen font. The `Parindent` will be fixed for a certain layout. The exception is Standard layout, since the indentation of a Standard layout paragraph can be prohibited with `NextNoIndent`. Also, Standard layout paragraphs inside environments use the `Parindent` of the environment, not their native one. For example, Standard paragraphs inside an enumeration are not indented.

`Parskip` [`float=0`] LyX allows to choose either "indent" or "skip" to typeset a document. When "indent" is chosen, this value is completely ignored. When "skip" is chosen, the parindent of a LaTeXtype "Paragraph" layout is ignored and all paragraphs are additionally separated by this parskip argument. The vertical space is calculated with `value*DefaultHeight()` where `DefaultHeight()` is the height of a row with the normal font. This way, the look stays the same with different screen fonts.

`TopSep` [`float=0`] The vertical space with which the very first of a chain of paragraphs with this layout is separated from the previous paragraph. If the previous paragraph has another layout, the separations are not simply added, but the maximum is taken.

`BottomSep` [`float=0`] The same as `TopSep` for the very last paragraph.

`Parsep` [`float=0`] The vertical space between two paragraphs of this layout.

`Itemsep` [`float=0`] This is an extra space between the paragraphs of an environment layout. If you put other layouts into an environment, each is separated with the environment's `Parsep`. But the whole items of the environment are additionally separated with this `Itemsep`.

`LeftMargin` [`string=""`] If you put layouts into environments, the leftmargins are not simply added, but added with a factor $\frac{4}{depth+4}$. Note that this parameter is also used when the border is defined as `Manual` or `Dynamic`. Then it is added to the manual or dynamic border. This string has the same meaning as for `ParIndent`.

`RightMargin` [`string=""`] Similar to `LeftMargin`.

`Labeltype` [*`No_Label`* `, Manual, Static, Top_Environment, Centered_Top_Environment, Sensitive, Counter`]
`Manual` means the label is the very first word (up to the first real blank). `Static` means it is defined in the layout (see `LabelString`). `Top_Environment` and `Centered_Top_Environment` are special cases of `Static`. The label will be printed above the paragraph, but only at the top of an environment or the top of a chain of paragraphs with this layout. Usage is for example the Abstract layout or the Bibliography layout. This is also the case for `Manual` labels with latex type `Environment`, in order to make layouts for theorems work correctly. `Sensitive` is a special case for the caption-labels "Figure" and "Table". `Sensitive` means the (hardcoded) label string depends on the kind of float. The `Counter` label type defines automatically numbered labels.

`LabelCounter` [`Chapter, Section, Subsection, Subsubsection, Paragraph, Subparagraph, EnumI, EnumII, EnumIII, EnumIV`]
The name of the counter for automatic numbering. This must be given if `Labeltype` is `Counter`.

`Labelsep` [`string=""`] The horizontal space between the label and the text body. Only used for labels that are not above the text body.

`LabelBottomsep` [float=0] The vertical space between the label and the text body. Only used for labels that are above the text body (`Top_Environment`, `Centered_Top_Environment`).

`LabelString` [`string=""`] The string used for a label with a `Static` labeltype. When the border is `Manual` this string is also used as a suggestion for the `LabelWidthString` that can be set in the Edit ▷ Paragraph Settings dialog.

When `LabelCounter` is set, this string can be contain special formatting commands as explained in Section 5.3.7.

`LabelStringAppendix` [`string=""`] If non-empty, this is used inside the appendix instead of `LabelString`.

`TocLevel [int]` The level of the style in the table of contents. This is used for automatic numbering of section headings.

`EndLabeltype` [*`No_Label`*`, Box, Filled_Box, Static`] The type of label that stands at the end of the paragraph (or sequence of paragraphs if `LatexType` is `Environment`, `Item_Environment` or `List_Environment`). `No_Label` means "nothing", `Box` (resp. `Filled_Box`) is a white (resp. black) square suitable for end of proof markers, `Static` is an explicit text string.

`EndLabelString` [`string=""`] The string used for a label with a `Static EndLabelType`.

`Align` [*`block`*`, left, right, center`] Paragraph alignment.

`AlignPossible` [*`block`*`, left, right, center`] A comma separated *list* of possible aligns. Some LaTeX styles prohibit certain alignments, since those wouldn't make sense. For example a right-aligned or centered enumeration isn't possible.

`Fill_Top` [*`0`*`,1`] With this parameter the Fill value of the "Vertical space above" list of the Edit▷Paragraph Settings dialog can be set when initializing a paragraph with this layout[4].

`Fill_Bottom` [*`0`*`,1`] Similar to `Fill_Top`.

`NeedProtect` [*`0`*`,1`] Whether fragile commands in this layout should be `\protect`'ed.

`Newline` [`0,` *`1`*] Whether newlines are translated into LaTeX newlines (\\) or not. The translation can be switched off to allow more comfortable LaTeX editing inside LyX.

`PassThru` [*`0`*`, 1`] Whether the contents of this paragraph should be output in raw form, meaning without special translations that LaTeX would require. This somehow replaces the older `Latex` font property.

`FreeSpacing` [*`0`*`, 1`] Usually LyX doesn't allow you to insert more than one space between words, since a space is considered as the separation between two words, not a character or symbol of its own. This is a very fine thing but sometimes annoying, for example when typing program code or plain LaTeX code. For this reason, `FreeSpacing` can be enabled. Note that LyX

[4] *Note from Jean-Marc:* I'm not sure that this setting has much use, and it should probably be removed in later versions.

will create protected blanks for the additional blanks when in another mode than LaTeX-mode.

`KeepEmpty` [*0* , 1] Usually LyX does not allow you to leave a paragraph empty, since it would lead to empty LaTeX output. There are some cases where this could be desirable however: in a letter template, the required fields can be provided as empty fields, so that people do not forget them; in some special classes, a layout can be used as some kind of break, which does not contain actual text.

`Spacing` [*single* , `onehalf`, `double`, `other` *value*] This defines what the default spacing should be in the layout. The arguments `single`, `onehalf` and `double` correspond respectively to a multiplier value of 1, 1.25 and 1.667. If you specify the argument `other`, then you should also provide a numerical argument which will be the actual multiplier value. Note that, contrary to other parameters, `Spacing` implies the generation of specific LaTeX code, using the package `setspace.sty`.

`Font` The font used for both the text body *and* the label. See section 5.3.8. Note that defining this font automatically defines the `LabelFont` to the same value.

`TextFont` The font used for the text body . See section 5.3.8.

`LabelFont` The font used for the label. See section 5.3.8.

`Preamble...EndPreamble` A set of macro definitions that will be output at the beginning of the LaTeX files when the layout is used. Use this to define the macros needed by this particular layout.

`DependsOn` the name of a style which preamble should be output *before* the one mentionned above. This allows to ensure some ordering of the preamble snippets when macros definitions depend on one another[5].

5.3.5 Floats

Since version 1.3.0 of LyX, it is necessary to define the floats (figure, table, . . .) in the text class itself. If you are looking here to learn how to upgrade an existing text class, it will probably turn out that all you have to do is to add

```
Input stdfloats.inc
```

at a reasonable location of the text class.[6] If you want to implement a text class that proposes some other float types (like the AGU class bundled with LyX), the information below will hopefully help you:

[5] Note that, besides that functionality, there is no way to ensure any ordering of preambles. The ordering that you see in a given version of LyX may change without warning in later versions.

[6] Don't forget to also have a look at counters in next section.

`Type` [`string`=""] The "type" of the new class of floats, like program or algorithm. After the appropriate `\newfloat`, commands such as `\begin{program}` or `\end{algorithm*}` will be available.

`GuiName` [`string`=""] The string that will be used in the menus and also for the caption.

`LaTeXBuiltin` [*`0`*, `1`] Set to `1` if the float is already defined by the documentclass. If this is set to `0`, the float will be defined using the LaTeX package `float`.

`NumberWithin` [`string`=""] This (optional) argument determines whether floats of this class will be numbered within some sectional unit of the document. For example, if within is equal to `chapter`, the floats will be numbered within chapters.

`Style` [`string`=""] The style used when defining the float using `\newfloat`.

`Placement` [`string`=""] The default placement for the given class of floats. They are like in standard LaTeX: `t`, `b`, `p` and `h` for top, bottom, page, and here, respectively.[7] On top of that there is a new type, `H`, which does not really correspond to a float, since it means: put it "here" and nowhere else. Note, however that the `H` specifier is special and, because of implementation details cannot be used in non-builtin float types. If you do not understand what this means, just use `"tbp"`.

`Extension` [`string`=""] The file name extension of an auxiliary file for the list of figures (or whatever). LaTeX writes the captions to this file.

`ListName` [`string`=""] The heading used for the list of floats.

5.3.6 Character styles

You can define character styles since version 1.4.0 of LyX. The `CharStyle` section can contain the following entries:

`Font` The font used for both the text body *and* the label. See section 5.3.8. Note that defining this font automatically defines the `LabelFont` to the same value.

`LabelFont` The font used for the label. See section 5.3.8.

`LatexName` The name of the corresponding LaTeX stuff. Either the environment or command name.

`LatexParam` The optional parameter for the corresponding `LatexName` stuff. This parameter cannot be changed from within LyX.

`LatexType` See section 5.3.4.

`Preamble...EndPreamble` See section 5.3.4

[7] Note that the order of these letters in the string is irrelevant, like in LaTeX.

5.3.7 Counters

Since version 1.3.0 of LyX, it is necessary to define the counters (chapter, figure, . . .) in the text class itself. If you are looking here to learn how to upgrade an existing text class, it will probably turn out that all you have to do is to add

```
Input stdcounters.inc
```

The definition of counters is presently a bit primitive in LyX, since many things are still hardcoded. The following two parameters can be used:

`Name` [`string`=""] The name of the counter

`Within` [`string`=""] If this is set to the name of another counter, the present counter will be reset everytime the other one is increased (is that unclear enough?).

When a counter has been associated to a style, it is possible to use some special constructs in `LabelString` and `LabelStringAppendix`:

- `@`*`style-name`*`@` will be replaced the expanded `LabelString` of style *`style-name`*. This is used for example to define the label of a subsection in terms of the label of a section.
- counter values can be expressed using LaTeX-like macros `\`*`numbertype`*`{`*`counter`*`}`, where *`numbertype`* can be:

  `arabic` to translate `counter` to arabic numerals, like 1, 2, 3. . . [8]

  `alph` for lower-case letters: a, b, c, . . .

  `Alph` for upper-case letters: A, B, C, . . .

  `roman` for lower-case roman numerals: i, ii, iii, . . .

  `Roman` for upper-case roman numerals: I, II, III. . .

  `hebrew` for hebrew numerals.

5.3.8 Font description

A font description looks like that:

```
Font or LabelFont
  ...
EndFont
```

and the following commands are available:

[8] Actually, the situation is a bit more complicated than that: any *`numbertype`* other than those descibed below will produce arabic numerals. It would not be surprising to see this change in the future.

`Family` [*Roman* , `Sans, Typewriter`]

`Series` [*Medium* , `Bold`]

`Shape` [*Up* , `Italic, SmallCaps, Slanted`]

`Size` [`tiny, small,` *normal* , `large, larger, largest, huge, giant`]

`Color` [*none* , `black, white, red, green, blue, cyan, magenta, yellow`]

5.3.9 Upgrading old layout files

The file format of layout files changes from time to time, so old layout files need to be converted. This process has been automated in LyX 1.4.0: If LyX reads an old format layout file it will call the conversion tool `$LyXDir/scripts/layout2layout.py` and convert it to a temporary file in current format. The original file is left untouched, so that you can still use it with LyX 1.3.x. If you want to convert the layout file permanently, just call the converter by hand:

```
    python $LyXDir/scripts/layout2layout.py myclass.layout myclassnew.layout
```

Then copy `myclassnew.layout` to `$UserDir/layouts/`.

The automatic conversion does only handle syntax changes. It cannot handle the case where the contents of included files was changed. For example, layout files based on `book.layout` need to include `numreport.inc` in addition to `stdclass.inc`. If you get error messages about undefined counters, try to convert your file with `layout2layout.py` and then add one of `numarticle.inc`, `numreport.inc` and `numrevtex.inc`.

5.4 Creating Templates

Templates are created just like usual documents. The only difference is that usual documents contain all possible settings, including the fontscheme and the papersize. Usually a user doesn't want a template to overwrite his defaults in these cases. For that reason, the designer of a template should remove the corresponding commands like `\fontscheme` or `\papersize` from the template LyX file. This can be done with any simple text-editor, for example `vi` or `xedit`.

Put the edited template files you create in `$UserDir/templates/`, copy the ones you use from the global template directory in `$LyXDir/templates/` to the same place, and redefine the template path in the Edit ▷ Preferences dialog (tabs Input, Path).

Note that there is a template which has a particular meaning: `defaults.lyx`. This template is loaded everytime you create a new document with File ▷ New in order to provide useful defaults. To create this template from inside LyX, all you have to do is to open a document with the correct settings, and use the Save as Document Defaults button.

Chapter 6

Including External Material

6.1 Background

One often requested feature from LyX users is to be able to interface LyX with XFig, Dia, or other similar applications that specialize in producing a certain kind of diagram, figure, schematic or whatever material might be relevant to include in your document. Previously, it was only possible to include boring, static, fixed images in LyX documents with the graphics feature, but there are several limitations attached to this approach:

- If you want to change the figure, you have to invoke an external program by hand
- LyX does not notice that the referenced files change, so the on-screen display can fast become obsolete, and this is aggravated by the lack of a means of updating the display
- The graphics stuff does not provide any mechanisms for coping with different exported formats such as DocBook, HTML or raw Ascii

The external material facility attempts to solve all of these problems[1]. It does this by offering a general method to interface LyX to external applications. Instead of introducing a long list of different constructs tailored for each specific application, we chose to sacrifice the in-line displaying of the included material in order to provide a general construct to cover a wide range of applications. The result is the external material construct. External material presents itself in the document simply as a button, but don't let this fool you. When you click on it, a dialog will appear that allows you to chose exactly what material to include, and in the following sections you will learn that this is indeed a powerful mechanism that can solve all of the above problems, and more.

[1] Even if the graphics facility can't solve all problems, it is still valuable because it does provide in-line preview of the graphics, and supports advanced geometric transformations with a comfortable user interface.

6.2 How does it work?

The external material feature is based on the concept of a *template*. A template is a specification of how LyX should interface with a certain kind of material. As bundled, LyX comes with predefined templates for XFig figures, Dia diagrams, various raster format images, gnuplot, and more. You can check the actual list by using the Insert ▷ External Material command. Furthermore, it is possible to roll your own template to support a specific kind of material. Later we'll describe in more detail what is involved, and hopefully you will submit all the templates you create so we can include them in a later LyX version.

Another basic idea of the external material feature is to distinguish between the original file that serves as a base for final material and the produced file that is included in your exported or printed document. For example, consider the case of a figure produced with XFig. The XFig application itself works on an original file with the `.fig` extension. Within XFig, you create and change your figure, and when you are done, you save the `fig`-file. When you want to include the figure in your document, you invoke `transfig` in order to create a PostScript file that can readily be included in your LaTeX file. In this case, the `.fig` file is the original file, and the PostScript file is the produced file.

This distinction is important in order to allow updating of the material while you are in the process of writing the document. Furthermore, it provides us with the flexibility that is needed to support multiple export formats. For instance, in the case of an Ascii resulting file, it is not exactly an award-winning idea to include the figure as raw PostScript. Instead, you'd either prefer to just include a reference to the figure, or try to invoke some graphics to Ascii converter to make the final result look similar to the real graphics. The external material management allows you to do this, because it is parameterized on the different export formats that LyX supports.

Besides supporting the production of different products according to the exported format, it supports tight integration with editing and viewing applications. In the case of an XFig figure, you are able to invoke `xfig` on the original file with a single click from within the external material dialog in LyX, and also preview the produced PostScript file with `ghostview` with another click. No more fiddling around with the command line and/or file browsers to locate and manipulate the original or produced files. In this way, you are finally able to take full advantage of the many different applications that are relevant to use when you write your documents, and ultimately be more productive.

So, all in all, LyX has information about a number of different programs to use behind the scenes in order to realize all of this machinery. This information, in fact, is exactly what is contained in the templates. To each template, there is associated a list of command lines that are used to invoke applications, convert the original file to the produced file, and more. This mechanism allows the advanced user to extend the capabilities of LyX without fiddling with the source code. It requires some footwork to define all the different commands and flags, but luckily, the LyX team did all the hard work and specified these for you.

But before the trees grow into the skies, we have to admit that we did take one tiny short-cut. Since you can produce many different kinds of files to go with each exported format, one could also expect that it would be possible to preview each product. The LyX team decided against this in order to keep the user interface simple. Instead of providing a button for each exported file format, we decided to introduce the concept of the primary file format and just have one button. When you press View result in the external material dialog, you will get a view of the produced file in the primary file format. And the primary file format is specified by your document class. For most document classes, the primary file format is LaTeX, but for the DocBook document classes, the primary file format is DocBook. So, when you view the produced file, keep in mind that it will only be a preview of what the main result will be. If you want to see how other exported formats turn out, you have to export them and preview them by hand.

6.3 The external material dialog

You insert external material from the Insert menu. When you do this, a button is inserted into your document, and the external material dialog is shown. This dialog allows you to describe exactly what material should be included, and also how it should be included. Furthermore, it provides access to the external applications to either view, edit or produce the material that is used in the resulting file.

At the top of this dialog, there is a drop-down list where you can chose which template should be used. Just below the template drop-down, there's a text area with a short blurb about the chosen template that should help you use it. Most often, it will provide a short description of the template, and a few hints on how to parameterize the use of it. Further down, you'll find a filename input field along with a "Browse" button that allows you to chose which file should be included with the standard file browser. Thus this field specifies the original file. Since the produced file is automatically generated when needed, there is no need to give access to it in the user interface.

At the bottom of the dialog, you'll find a general input box called Parameters. This box is generally used to parameterize the specific template. The specific use should be covered in the help blurb associated with the template, but it typically allows you to define variations on how the produced file should be generated.

At the right side of the dialog, you'll find three buttons: Edit file, View result, and Update result. These in turn allow you to edit your original file with the appropriate editing application, view the produced file as included in the primary format document, and finally force an update of the resulting material in the primary format. Normally, the Update result button will be disabled, because most templates are configured to automatically update the produced file when needed. In those cases, there is no need to force the production of a new produced file. However, some templates are configured to not be automatically producing the residual product, because the cost of producing the produced file might be so large that it would be a pain to do it all the time. Those types are known as *manual* external material. In those cases, you can use the button to force the production of the produced file exactly when you need it, and thus control the amount of work that is done. In fact, it is *your* responsibility to do this to keep the produced files current at all times: before printing, before exporting, before viewing, etc. At some time in the future, it might be possible that LyX will help you with this task.Any changes in the template, filename or parameters are actually applied whenever you press Edit file, View result or Update result buttons.

6.4 Examples

In this section, we should include some examples of use of the external material. Those examples could include:

- External raster images
- External XFig figures
- Chess diagrams
- Sound samples
- The use of makefiles
- Recursive external LyX templates

6.5 The external template configuration file

It is relatively easy to add custom external template definitions to LyX. However, be aware this doing this in an careless manner most probably *will* introduce an easily exploitable security hole. So before you do this, please read the discussion about security which will follow later.

Having said that, we encourage you to submit any interesting templates that you create.

The external templates are defined in the `lib/external_templates` file. You can place your own version in `.lyx/external_templates`.

A typical template looks like this:

```
Template XFig
GuiName "XFig: $$AbsOrRelPathParent$$Basename"
HelpText
An XFig figure.
HelpTextEnd
InputFormat fig
FileFilter "*.fig"
AutomaticProduction true
```

```
Transform Rotate
Transform Resize
Format LaTeX
TransformCommand Rotate RotationLatexCommand
TransformCommand Resize ResizeLatexCommand
Product "$$RotateFront$$ResizeFront
         \\input{$$AbsOrRelPathMaster$$Basename.pstex_t}
         $$ResizeBack$$RotateBack"
UpdateFormat pstex
UpdateResult "$$AbsPath$$Basename.pstex_t"
Requirement "graphicx"
ReferencedFile latex "$$AbsOrRelPathMaster$$Basename.pstex_t"
ReferencedFile latex "$$AbsPath$$Basename.eps"
ReferencedFile dvi "$$AbsPath$$Basename.eps"
FormatEnd
Format PDFLaTeX
TransformCommand Rotate RotationLatexCommand
TransformCommand Resize ResizeLatexCommand
Product "$$RotateFront$$ResizeFront
         \\input{$$AbsOrRelPathMaster$$Basename.pdftex_t}
         $$ResizeBack$$RotateBack"
UpdateFormat pdftex
UpdateResult "$$AbsPath$$Basename.pdftex_t"
Requirement "graphicx"
ReferencedFile latex "$$AbsOrRelPathMaster$$Basename.pdftex_t"
ReferencedFile latex "$$AbsPath$$Basename.pdf"
FormatEnd
Format Ascii
Product "$$Contents(\"$$AbsPath$$Basename.asc\")"
UpdateFormat asciixfig
UpdateResult "$$AbsPath$$Basename.asc"
FormatEnd
Format DocBook
Product "<graphic fileref=\"$$AbsOrRelPathMaster$$Basename.eps\">
         </graphic>"
UpdateFormat eps
UpdateResult "$$AbsPath$$Basename.eps"
ReferencedFile docbook "$$AbsPath$$Basename.eps"
ReferencedFile docbook-xml "$$AbsPath$$Basename.eps"
FormatEnd
Format LinuxDoc
Product "[XFig: $$FName]"
FormatEnd
TemplateEnd
```

As you can see, the template is enclosed in `Template` … `TemplateEnd`. It contains a header specifying some general settings, and for each supported primary document file format a section `Format` ... `FormatEnd`.

6.5.1 The template header

`Template <id>` A unique name for the template. It must not contain substitution macros (see below).

`GuiName <guiname>` The text that is displayed on the button. This command must occur exactly once.

`HelpText <text> HelpTextEnd` The help text that is used in the External dialog. Provide enough information to explain to the user just what the template can provide him with. This command must occur exactly once.

`InputFormat <format>` The file format of the original file. This must be the name of a format that is known to LyX (see the Tools ▷ Preferences:Conversion dialog). Use `"*"` if the template can handle original files of more than one format. LyX will attempt to interrogate the file itself in order to deduce its format in this case. This command must occur exactly once.

`FileFilter <pattern>` A glob pattern that is used in the file dialog to filter out the desired files. If there is more than one possible file extension (e.g. tgif has `.obj` and `.tgo`), use something like `"*.{obj,tgo}"`. This command must occur exactly once.

`AutomaticProduction true|false` Wether the file represented by the template must be generated by LyX. This command must occur exactly once.

`Transform Rotate|Resize|Clip|Extra` This command specifies which transformations are supported by this template. It may occur zero or more times. This command enables the corresponding tabs in the external dialog. Each `Transform` command must have either a corresponding `TransformCommand` or a `TransformOption` command in the `Format` section. Otherwise the transformation will not be supported by that format.

6.5.2 The Format section

`Format LaTeX|PDFLaTeX|Ascii|DocBook|LinuxDoc` The primary document file format that this format definition is for. Not every template has a sensible representation in all document file formats. Please define nevertheless a `Format` section for all formats. Use a dummy text when no representation is available (see the LinuxDoc format in the example above). Then you can at least see a reference to the external material in the exported document.

`TransformCommand Rotate RotationLatexCommand` This command specifies that the built in LaTeX command should be used for rotation. This command may occur once or not at all.

`TransformCommand Resize ResizeLatexCommand` This command specifies that the built in LaTeX command should be used for resizing. This command may occur once or not at all.

`TransformOption Rotate RotationLatexOption` This command specifies that rotation is done via an optional argument. This command may occur once or not at all.

`TransformOption Resize ResizeLatexOption` This command specifies that resizing is done via an optional argument. This command may occur once or not at all.

`TransformOption Clip ClipLatexOption` This command specifies that clipping is done via an optional argument. This command may occur once or not at all.

`TransformOption Extra ExtraLatexOption` This command specifies that an extra optional argument is used. This command may occur once or not at all.

`Product <text>` The text that is inserted in the exported document. This is actually the most important command and can be quite complex. This command must occur exactly once.

`UpdateFormat <format>` The file format of the converted file. This must be the name of a format that is known to LyX (see the Tools ▷ Preferences:Conversion dialog). This command must occur exactly once.

`UpdateResult <filename>` The file name of the converted file. The file name must be absolute. This command must occur exactly once.

`ReferencedFile <format> <filename>` This command denotes files that are created by the conversion process and are needed for a particular export format. If the filename is relative, it is interpreted relative to the master document. This command may be given zero or more times.

`Requirement <package>` The name of a required LaTeX package. The package is included via `\usepackage{}` in the LaTeX preamble. This command may occur zero or more times.

`Preamble <name>` This command specifies a preamble snippet that will be included in the LaTeX preamble. It has to be defined using `PreambleDef` ... `PreambleDefEnd`. This command may occur zero or more times.

`Option <name> <value>` This command defines an additional macro `$$<name>` for substitution in `Product`. `<value>` itself may contain substitution macros. The advantage over using `<value>` directly in `Product` is that the substituted value of `$$<name>` is sanitized so that it is a valid optional argument in the document format. This command may occur zero or more times.

6.5.3 Preamble definitions

The external template configuration file may contain additional preamble definitions enclosed by `PreambleDef` ... `PreambleDefEnd`. They can be used by the templates in the `Format` section.

6.6 The substitution mechanism

When the external material facility invokes an external program, it is done on the basis of a command defined in the template configuration file. These commands can contain various macros that are expanded before execution. Execution always take place in the directory of the containing document.

Also, whenever external material is to be displayed, the name will be produced by the substitution mechanism, and most other commands in the template definition support substitution as well.

The available macros are the following:

$$FName The filename of the file specified in the external material dialog. This is either an absolute name, or it is relative to the LyX document.

$$Basename The filename without path and without the extension.

$$Extension The file extension (including the dot).

$$FPath The path part of `$$FName` (absolute name or relative to the LyX document).

$$AbsPath The absolute file path.

$$RelPathMaster The file path, relative to the master LyX document.

$$RelPathParent The file path, relative to the LyX document.

$$AbsOrRelPathMaster The file path, absolute or relative to the master LyX document.

$$AbsOrRelPathParent The file path, absolute or relative to the LyX document.

$$Tempname A name and full path to a temporary file which will be automatically deleted whenever the containing document is closed, or the external material insertion deleted.

$$Contents("filename.ext") This macro will expand to the contents of the file with the name `filename.ext`.

$$Sysdir This macro will expand to the absolute path of the system directory. This is typically used to point to the various helper scripts that are bundled with LyX.

All path macros contain a trailing directory separator, so you can construct e.g. the absolute filename with `$$AbsPath$$Basename$$Extension`.

The macros above are substituted in all commands unless otherwise noted. The command `Product` supports additionally the following substitutions if they are enabled by the `Transform` and `TransformCommand` commands:

$$ResizeFront The front part of the resize command.

$$ResizeBack The back part of the resize command.

$$RotateFront The front part of the rotation command.

$$RotateBack The back part of the rotation command.

The value string of the `Option` command supports additionally the following substitutions if they are enabled by the `Transform` and `TransformOption` commands:

$$Clip The clip option.

$$Extra The extra option.

$$Resize The resize option.

$$Rotate The rotation option.

You may ask why there are so many path macros. There are mainly two reasons:

First, relative and absolute file names should remain relative or absolute, respectively. Users may have reasons to prefer either form. Relative names are useful for portable documents that should work on different machines, for example. Absolute names may be required by some programs.

Second, LaTeX treats relative file names differently than LyX and other programs in nested included files. For LyX, a relative file name is always relative to the document that contains the file name. For LaTeX, it is always relative to the master document. These two definitions are identical if you have only one document, but differ if you have a master document that includes part documents. That means that relative filenames must be transformed when presented to LaTeX. Fortunately LyX does this automatically for you if you choose the right macros.

So which path macro should be used in new template definitions? The rule is not difficult:

- Use `$$AbsPath` if an absolute path is required.
- Use `$$AbsOrRelPathMaster` if the substituted string is some kind of LaTeX input.
- Else use `$$AbsOrRelPathParent` in order to preserve the user's choice.

There are special cases where this rule does not work and e.g. relative names are needed, but normally it will work just fine. One example for such a case is the command `ReferencedFile latex "$$AbsOrRelPathMaster$$Basename.pstex_t"` in the XFig template above: We can't use the absolute name because the copier for `.pstex_t` files needs the relative name in order to rewrite the file content.

6.7 Security discussion

The external material feature interfaces with a lot of external programs and does so automatically, so we have to consider the security implications of this. In particular, since you have the option of including your own filenames and/or parameter strings and those are expanded into a command, it seems that it would be possible to create a malicious document which executes arbitrary commands when a user views or prints the document. This is something we definately want to avoid.

However, since the external program commands are specified in the template configuration file only, there are no security issues if LyX is properly configured with safe templates only. This is so because the external programs are invoked with the `execvp`-system call rather than the `system` system-call, so it's not possible to execute arbitrary commands from the filename or parameter section via the shell.

This also implies that you are restricted in what command strings you can use in the external material templates. In particular, pipes and redirection are not readily available. This has to be so if LyX should remain safe. If you want to use some of the shell features, you should write a safe script to do this in a controlled manner, and then invoke the script from the command string. In the `lib/scripts` directory of the LyX installation, you can find a safe wrapper script `general_command_wrapper.py` that supports redirection of input and output. That can serve as an example for how to write safe template scripts. For a more advanced example that uses `fork` and friends, take a look at the `pic2ascii.py` converter script.

It is possible to design a template that interacts directly with the shell, but since this would allow a malicious user to execute arbitrary commands by writing clever filenames and/or parameters, we generally recommend that you only use safe scripts that work with the `execvp` system call in a controlled manner. Of course, for use in a controlled environment, it can be tempting to just fall back to use ordinary shell scripts. If you do so, be aware that you *will* provide an easily exploitable security hole in your system. Of course it stands to reason that such unsafe templates will never be included in the standard LyX distribution, although we do encourage people to submit new templates in the open source tradition. But LyX as shipped from the official distribution channels will never have unsafe templates.

Including external material provides a lot of power, and you have to be careful not to introduce security hazards with this power. A subtle error in a single line in an innocent looking script can open the door to huge security problems. So if you do not fully understand the issues, we recommend that you consult a knowledgable security professional or the LyX development team if you have any questions about whether a given template is safe or not. And do this before you use it in an uncontrolled environment.

Chapter 7

The LyX Server

7.1 Introduction

The LyX server is a method implemented in LyX that will enable other programs to talk to LyX, invoke LyX commands, and retrieve information about the LyX internal state. This is only intended for advanced users, but they should find it useful.

7.2 Starting the LyX Server

The LyX server works through the use of a pair of named pipes. These are usually located in your home directory and have the names "`.lyxpipe.in`" and "`.lyxpipe.out`". External programs write into `.lyxpipe.in` and read back data from `.lyxpipe.out`. The stem of the pipe names can be defined in the Tools▷Preferences dialog, for example `"/home/myhome/.lyxpipe"`.

LyX will add the '`.in`' and '`.out`' to create the pipes. The above setting also has the effect of activating the LyX server. If one of the pipes already exists, LyX will assume that another LyX process is already running and will not start the server. To have several LyX processes with servers at the same time, you have to change the configuration between the start of the programs.

If you are developing a client program, you might find it useful to enable debugging information from the LyX server. Do this by starting LyX as `lyx -dbg lyxserver`.

Warning: if LyX crashes, it may not manage to remove the pipes; in this case you must remove them manually. If LyX starts and the pipes exist already, it will not start any server.

Other than this, there are a few points to consider:

- Both server and clients must run on UNIX or OS/2 machines. Communications between LyX on UNIX and clients on OS/2 or vice versa is not possible right now.

- On OS/2, only one client can connect to LYXServer at a time.
- On OS/2, clients must open inpipe with `O_WRONLY` mode.

You can find a complete example client written in C in the source distribution as `development/server_monitor.c`.

7.3 Normal communication

To issue a LYX call, the client writes a line of ASCII text into the input pipe. This line has the following format:

LYXCMD:*clientname*:*function*:*argument*

Here *clientname* is a name that the client can choose arbitrarily. Its only use is that LYX will echo it if it sends an answer - so a client can dispatch results from different requesters.

function is the function you want LYX to perform. It is the same as the commands you'd use in the minibuffer.

argument is an optional argument which is meaningful only to some functions (for instance "self-insert" which will insert the argument as text at the cursor position.)

The answer from LYX will arrive in the output pipe and be of the form

INFO:*clientname*:*function*:*data*

where *clientname* and *function* are just echoed from the command request, while *data* is more or less useful information filled according to how the command execution worked out. Some commands will return information about the internal state of LYX, such as "font-state", while other will return an empty data-response. This means that the command execution went fine.

In case of errors, the response from LYX will have this form

ERROR:*clientname*:*function*:*error message*

where the *error message* should contain an explanation of why the command failed.

Examples:

```
echo "LYXCMD:test:beginning-of-buffer:" >~/.lyxpipe.in
echo "LYXCMD:test:get-xy:" >~/.lyxpipe.in
read a <~/.lyxpipe.out
echo $a
```

7.4 Notification

LyX can notify clients of events going on asynchronously. Currently it will only do this if the user binds a key sequence with the function "notify". The format of the string LyX sends is as follows:

> `NOTIFY:`*key-sequence*

where *key-sequence* is the printed representation of the key sequence that was actually typed by the user.

This mechanism can be used to extend LyX's command set and implement macros: bind some key sequence to "notify", start a client that listens on the out pipe, dispatches the command according to the sequence and starts a function that may use LyX calls and LyX requests to issue a command or a series of commands to LyX.

7.5 The simple LyX Server Protocol

LyX implements a simple protocol that can be used for session management. All messages are of the form

> LYXSRV:*clientname*:*protocol message*

where *protocol message* can be "hello" or "bye". If "hello" is received from a client, LyX will report back to inform the client that it's listening to it's messages, while "bye" sent from LyX will inform clients that LyX is closing.

Appendix A

Bindings

This appendix is a huge cross-reference to all the English language keybindings. Originally, we simply wanted to list all of the key bindings followed by the function it's bound to. That way, a user can look up a key to find out what it does. We then decided, what the hey, why not include the default toolbar and menu bindings, too. Please note this section is likely to be very out of date.

The form is really self-explanatory, but here are a few tips: all entries are arranged roughly alphabetically for a given modifier (C-a, C-b, etc.). For the general keyboard layout, simpler prefixes precede the more complex (C-s before C-S-c). All entries were gleaned from the default user interface and binding files located in the directories `.../share/lyx/ui` and `.../share/lyx/bind`; they should be treated as the final word on the bindings.

As a final note, be aware that some window managers (such as FVWM) take control of some of the function keys or motion keys. C-right is listed here as generating `word-forward`, but FVWM grabs it and uses it to change virtual desktops instead. Very annoying unless you instruct your window manager to stop intercepting such sequences.

A.1 Toolbar

```
Toolbar
    Layouts
    Icon "buffer-open"
    Icon "buffer-write"
    Icon "buffer-print"
    Separator
    Icon "cut"
    Icon "copy"
    Icon "paste"
    Separator
    Icon "font-emph"
```

```
        Icon "font-noun"
        Icon "font-free"
        Separator
        Icon "tex-mode"
        Icon "math-mode"
        Separator
        Icon "footnote-insert"
        Icon "marginpar-insert"
        Icon "depth-next"
        Separator
        Icon "figure-insert"
        Icon "dialog-tabular-insert"
    End
```

A.2 Menu

A.2.1 File

M-f a	`buffer-write-as`
M-f c	`buffer-close`
M-f d	`buffer-reload`
M-f e	`file_export` submenu
M-f f	`buffer-export fax`
M-f i	`file_import` submenu
M-f n	`buffer-new`
M-f o	`buffer-open`
M-f p	`buffer-print`
M-f s	`buffer-write`
M-f t	`buffer-new-template`
M-f v	`file_vc` submenu

h	`vc-history`
i	`vc-check-in`
l	`vc-revert`
o	`vc-check-out`
r	`vc-register`
u	`vc-undo-last`

M-f x	`lyx-quit`

A.2.2 Edit

M-e a	paste	
M-e c	cut	
M-e d	redo	
M-e e	error-remove-all	
M-e f	find-replace	
M-e h	buffer-chktex	
M-e i	`edit_floats` submenu	
	a	tabular-feature append-row
	b	tabular-feature toggle-line-bottom
	c	tabular-feature align-center
	d	tabular-feature delete-column
	e	tabular-feature align-left
	i	tabular-feature align-right
	l	tabular-feature toggle-line-left
	m	tabular-feature multicolumn
	n	tabular-feature valign-center
	o	tabular-feature valign-top
	r	tabular-feature toggle-line-right
	t	tabular-feature toggle-line-top
	u	tabular-feature append-column
	v	tabular-feature valign-bottom
	w	tabular-feature delete-row
M-e l	math-panel	
	a	floats-operate openfoot
	c	floats-operate closefoot
	f	floats-operate openfig
	m	melt
	o	open-stuff
	t	floats-operate closefig
M-e o	copy	
M-e p	dialog-preferences	

M-e r	reconfigure	
M-e s	spellchecker	
M-e t	edit_tabular submenu	
M-e u	undo	
M-e x	edit_paste submenu	
	l	primary-selection-paste
	p	primary-selection-paste paragraph

A.2.3 Insert

M-i a	insert_floats submenu	
	a	buffer-float-insert algorithm
	d	buffer-float-insert wide-tab
	f	buffer-float-insert figure
	t	buffer-float-insert table
	w	buffer-float-insert wide-fig
M-i b	dialog-tabular-insert	
M-i c	citation-insert	
M-i d	math-display	
M-i e	buffer-child-insert	
M-i f	footnote-insert	
M-i g	figure-insert	
M-i h	math-mode	
M-i i	index-insert	
M-i l	label-insert	
M-i m	marginpar-insert	
M-i n	note-insert	
M-i o	insert_toc submenu	
	a	loa-insert
	b	bibtex-insert
	c	toc-insert

	f	`lof-insert`
	i	`index-print`
	t	`lot-insert`
M-i r	`reference-insert`	
M-i s	`insert_special` submenu	
	b	`protected-space-insert`
	e	`end-of-sentence-period-insert`
	h	`hfill-insert`
	i	`dots-insert`
	l	`break-line`
	m	`menu-separator-insert`
	p	`hyphenation-point-insert`
	q	`quote-insert`
	s	`command-sequence math-insert ^;math-mode;`
	u	`command-sequence math-insert _;math-mode;`
M-i t	`insert_file` submenu	
	l	`file-insert-ascii lines`
	p	`file-insert-ascii paragraphs`
	x	`file-insert`
M-i u	`url-insert`	
M-i w	`index-insert-last`	
M-i x	`external-insert`	

A.2.4 Layout

M-l a	`appendix`
M-l b	`font-bold`
M-l c	`layout-character`
M-l d	`layout-document`
M-l e	`font-emph`
M-l l	`layout-preamble`
M-l n	`font-noun`

M-l p	`layout-paragraph`
M-l s	`layout-save-default`
M-l t	`layout-tabular`
M-l v	`depth-increment`
M-l x	`tex-mode`

A.2.4.1 Layout ▷ Character

M-c b	`font-bold`
M-c c	`font-noun`
M-c e	`font-emph`
M-c m	`math-mode`
M-c p	`font-code`
M-c r	`font-roman`
M-c s	`font-sans`
M-c u	`font-underline`
M-c space	`font-default`
M-c Down	`word-lowcase`
M-c Up	`word-upcase`
M-c Right	`word-capitalize`
M-s h	`font-size huge`
M-s l	`font-size large`
M-s n	`font-size normal`
M-s s	`font-size small`
M-s t	`font-size tiny`
M-s 0	`font-size huger`
M-s 1	`font-size tiny`
M-s 2	`font-size smallest`
M-s 3	`font-size smaller`
M-s 4	`font-size small`

M-s 5	`font-size normal`
M-s 6	`font-size large`
M-s 7	`font-size larger`
M-s 8	`font-size largest`
M-s 9	`font-size huge`
M-s S-H	`font-size huger`
M-s S-L	`font-size larger`
M-s S-S	`font-size smaller`
M-s plus	`font-size increase`
M-s minus	`font-size decrease`

A.2.5 View

A.2.6 Navigate

A.2.7 Help

A.2.8 Paragraph Style

M-p a	`layout Abstract`
M-p b	`layout Itemize`
M-p c	`layout LyX-Code`
M-p d	`layout Description`
M-p e	`layout Enumerate`
M-p f	`layout ShortFoilhead`
M-p i	`layout Itemize`
M-p l	`layout List`
M-p n	`layout Enumerate`
M-p q	`layout Quote`
M-p r	`layout ShortRotatefoilhead`
M-p s	`layout Standard`
M-p t	`layout Title`

M-p v	`layout Verse`
M-p space	`drop-layouts-choice`
M-p 0	`layout Part`
M-p 1	`layout Chapter`
M-p 2	`layout Section`
M-p 3	`layout Subsection`
M-p 4	`layout Subsubsection`
M-p 5	`layout Paragraph`
M-p 6	`layout Subparagraph`
M-p asterisk 0	`layout Part*`
M-p asterisk 1	`layout Chapter*`
M-p asterisk 2	`layout Section*`
M-p asterisk 3	`layout Subsection*`
M-p asterisk 4	`layout Subsubsection*`
M-p asterisk 5	`layout Paragraph*`
M-p asterisk 6	`layout Subparagraph*`
M-p S-A	`layout Author`
M-p S-B	`layout Bibliography`
M-p S-C	`layout Comment`
M-p S-D	`layout Date`
M-p S-F	`layout Foilhead`
M-p S-L	`layout LaTeX`
M-p S-Q	`layout Quotation`
M-p S-R	`layout Rotatefoilhead`
M-p C-a	`layout RightAddress`
M-p M-a	`layout Address`
M-p M-c	`layout Caption`
M-p Left	`depth-decrement`

M-p Right	`depth-increment`

These ones are kept for backwards compatibility, but only make sense on a qwerty keyboard:

M-p S-at	`layout Section*`
M-p S-dollar	`layout Subsubsection*`
M-p S-numbersign	`layout Subsection*`

A.3 Keyboard

A.3.1 Specific to `emacs.bind`

C-a	`line-begin`
C-b	`char-backward`
C-d	`delete-forward`
C-e	`line-end`
C-f	`char-forward`
C-g	`cancel`
C-h	`hyphenation-point-insert`
C-i	`hfill-insert`
C-k	`line-delete-forward`
C-l	`screen-recenter`
C-m	`mark-toggle`
C-n	`down`
C-o	`open-stuff`
C-p	`up`
C-q	`quote-insert`
C-s	`find-replace`
C-u	`font-underline`
C-v	`screen-down`
C-w	`cut`

C-y	`paste`
C-S-Y	`layout-paste`
C-x a	`buffer-auto-save`
C-x b	`buffer-previous`
C-x c	`lyx-quit`
C-x d	`buffer-new`
? C-x f	`buffer-open`
C-x g	`buffer-view-ps`
C-x k	`buffer-close`
C-x p	`buffer-view`
C-x r	`buffer-typeset`
? C-x s	`buffer-write`
C-x t	`buffer-typeset`
C-x u	`undo`
C-x v c	`vc-undo-last`
C-x v h	`vc-history`
C-x v i	`vc-register`
C-x v u	`vc-revert`
C-x v v	`vc-check-in`
? C-x w	`buffer-write-as`
C-x bracketleft	`screen-up`
C-x bracketright	`screen-down`
C-x C-a	`buffer-auto-save`
C-x C-b	`menu-open Documents`
C-x C-c	`lyx-quit`
C-x C-d	`buffer-new`
C-x C-f	`buffer-open`
C-x C-g	`buffer-view-ps`

C-x C-p	`buffer-view`
C-x C-q	`buffer-toggle-read-only`
C-x C-s	`buffer-write`
C-x C-t	`buffer-typeset`
C-x C-w	`buffer-write-as`
Home	`buffer-begin`
End	`buffer-end`
S-Home	`line-begin-select`
S-End	`line-end-select`
S-Up	`up-select`
S-Down	`down-select`
S-Next	`screen-down-select`
S-Prior	`screen-up-select`
S-Left	`backward-select`
S-Right	`forward-select`
C-Up	`paragraph-up`
C-Down	`paragraph-down`
C-Left	`word-backward`
C-Right	`word-forward`
C-Delete	`word-delete-forward`
C-BackSpace	`word-delete-backward`
C-Return	`break-line`
C-period	`end-of-sentence-period-insert`
C-space	`protected-space-insert`
C-S-at	`mark-on`
C-S-greater	`label-goto`
C-S-less	`reference-back`
C-S-slash	`undo`

C-S-underscore	`undo`
C-S-quotedbl	`quote-insert`
C-S-Home	`buffer-begin-select`
C-S-End	`buffer-end-select`
C-S-Up	`paragraph-up-select`
C-S-Down	`paragraph-down-select`
C-S-Left	`word-backward-select`
C-S-Right	`word-forward-select`
Escape	`meta-prefix`
M-d	`word-delete-forward`
M-w	`copy`
M-x	`command-execute`
M-S-W	`layout-copy`
M-period	`dots-insert`
M-Return	`break-paragraph-keep-layout`
M-S-percent	`find-replace`

A.3.2 Specific to `cua.bind`

C-b	`font-bold`
C-c	`copy`
C-d	`buffer-view`
C-e	`font-emph`
C-f	`find-replace`
C-g	`error-next`
C-i	`open-stuff`
C-k	`font-noun`
C-l	`tex-mode`
C-m	`math-mode`
C-n	`buffer-new`

C-o	`buffer-open`
C-p	`buffer-print`
C-q	`lyx-quit`
C-r	`buffer-reload`
C-s	`buffer-write`
C-t	`buffer-view-ps`
C-u	`font-underline`
C-v	`paste`
C-w	`buffer-close`
C-x	`cut`
C-z	`undo`
C-space	`protected-space-insert`
C-S-C	`layout-copy`
C-S-D	`buffer-typeset`
C-S-M	`math-display`
C-S-N	`buffer-new-template`
C-S-P	`font-code`
C-S-S	`buffer-write-as`
C-S-T	`buffer-typeset-ps`
C-S-V	`layout-paste`
C-S-Z	`redo`
S-Insert	`paste`
S-Delete	`cut`
C-period	`end-of-sentence-period-insert`
C-S-greater	`label-goto`
C-S-less	`reference-back`
C-minus	`hyphenation-point-insert`
C-S-quotedbl	`quote-insert`

C-S-space	`protected-space-insert`
M-x	`command-execute`
S-Home	`line-begin-select`
S-End	`line-end-select`
S-Up	`up-select`
S-Down	`down-select`
S-Delete	`cut`
S-Insert	`paste`
S-Next	`screen-down-select`
S-Prior	`screen-up-select`
S-Left	`backward-select`
S-Right	`forward-select`
C-Home	`buffer-begin`
C-End	`buffer-end`
C-Up	`paragraph-up`
C-Down	`paragraph-down`
C-Delete	`word-delete-forward`
C-BackSpace	`word-delete-backward`
C-Insert	`copy`
C-Return	`break-line`
C-Left	`word-backward`
C-Right	`word-forward`
C-S-Down	`paragraph-down-select`
C-S-End	`buffer-end-select`
C-S-Home	`buffer-begin-select`
C-S-Left	`word-backward-select`
C-S-Right	`word-forward-select`
C-S-Up	`paragraph-up-select`

C-S-quotedbl	`quote-insert`
M-Return	`break-paragraph-keep-layout`
M-period	`dots-insert`
M-S-Right	`depth-increment`
M-S-Left	`depth-decrement`
Escape	`cancel`
F2	`buffer-write`
F3	`buffer-open`
F5	`screen-recenter`
F7	`spellchecker`
F9	`meta-prefix`
C-F4	`buffer-close`
M-F4	`lyx-quit`

A.3.3 Specific to `sciword.bind`

These are LyX keyboard definitions for mathematics, similar to those of Scientific Word.

The bindings file and the present documentation were prepared by Serge Winitzki with assistance from Jean-Marc Lasgouttes. Version 1.3, for LyX 1.2.x and 1.3.x.

These definitions make it a lot easier to type equations without using the mouse, especially for people familiar with Scientific Word. The standard LyX bindings such as `M-m` or `M-o` are unmodified.

Tip: to find the "LyX bind name" for a key, look at the status bar after typing some non-existent key combination. E.g. to find how "Ctrl-&" is referenced, press Ctrl-S and then Ctrl-&: the status bar shows "C-s S-C-ampersand." (This does not work in LyX 1.3.0!)

C-c `copy` – Copy, cut, paste is as in Sciword, `C-c`, `C-x`, and `C-v`.

C-d `math-display` – Display equation toggle: type `C-d` to insert a displayed formula (`d` for “displayed”). You can also type `C-d` in a displayed formula to convert it back to an inline formula.

C-f `math-insert \frac` – Fractions: type `C-f` to insert a fraction (`f` for “fraction”). You can also select an expression and type `C-f` to convert it to the numerator of a fraction. Note: pressing `Backspace` at the *left* end of the denominator will delete the numerator and convert the denominator to a non-fraction.

C-i `math-insert \int` – Inserts $\int$ (`i` for "integral")

C-k `line-delete-forward` – Emacs-like binding: delete forward of cursor to end of line.

C-m `math-mode` – A text/math toggle (`m` for "math"): switches to math in text mode, and also inserts roman text in math mode. Also bound to `C-t` (`t` for "text").

C-n `math-number` – Add/remove numbering in a single equation.

S-C-N `math-nonumber` – Add/remove numbering at a line in equation arrays.

The above commands are toggles that control the numbering of equations (`N` for "number"). Note: when deleting a number in a labeled eqnarray, the label is not really removed (the TeX code becomes "`\label{} \nonumber`") and this generates a (harmless) LaTeX warning.

C-o `file-open` – Open a new document. (W*ndows)

C-q `quote-insert` – Insert a quote character `"` (`q` for "quote"). This is not the "smart" double quote character that you get by default.

C-r `math-insert \sqrt` – Square root sign $\sqrt{x}$ (`r` for "root").

S-C-R `math-insert \root` – root sign $\sqrt[n]{x}$.

C-t `math-mode` – Another binding for a switch between the text and the math mode (`t` is for "text"). Note that the roman text inserted in math mode is special.

C-u `font-underline` – Underline the selected text (text mode only, use things like `\overline` or `\underbar` for math).

C-v `paste` – W*ndows heritage.

C-w `buffer-close` – Close the current document (again, a W*ndows heritage).

C-x `cut`

C-z `undo`

S-C-Z `redo` – the "Redo" operation, or "undo the undo".

Bracket delimiters: press `Ctrl-`<bracket key> to insert a matching pair of delimiters. For example, `Ctrl-[` inserts a pair of parentheses []. (Note: `Ctrl-]` does the same thing.) It will switch to math mode if needed. The supported characters are ([{ < |. It is the same to press the right or the left bracket. The corresponding delimiters are () [] {} ⟨⟩ ||. The delimiters are "smart" and resize with their contents. Use Math Panel to get other or non-matching delimiters. Press backspace on the *left* delimiter to remove both "smart" delimiters without removing their contents.

C-9 `math-delim ( )` – for convenience, pressing `Ctrl-9` is the same as `Ctrl-(` or `Ctrl-)`

S-C-parenleft `math-delim ( )`

C-0 `math-delim ( )`

S-C-parenright `math-delim ( )`

C-bracketleft `math-delim [ ]`

C-bracketright `math-delim [ ]`

C-S-less `math-delim langle rangle` – angular delimiters ⟨⟩, not to confuse with ordinary $<$ $>$ signs.

C-S-greater `math-delim langle rangle`

The bar bracket: on some keyboards (e.g. some British ones), the bar character is bound to an `Alt`-something and on some wayward Unices to "brokenbar". So we define all of these keys as well.

C-S-bar `math-delim | |`

C-S-brokenbar `math-delim | |`

C-M-bar `math-delim | |`

S-C-braceleft `math-delim { }`

S-C-braceright `math-delim { }`

Accents are in most cases `Ctrl-`<accent key>, e.g. `Ctrl-.` for overdot, `Ctrl-'` for acute accent, `Ctrl-~` for tilde (you also need to press `Shift` here) etc. For example, "`Ctrl-' a`" inserts á. Some accents work only in math mode and others only in text mode.

C-period `accent-dot` – overdot accent, ȧ (text mode only).

C-comma `math-insert \dot` – overdot accent, $\dot{a}$ (math mode only – in physics this denotes a first derivative).

C-equals `math-insert \overrightarrow` – Vector accent over math $\overrightarrow{x}$.

S-C-quotedbl `accent-umlaut` – umlaut accent, ä (text mode only)

S-C-colon `math-insert \ddot` – double dot accent, $\ddot{a}$ (math mode only – in physics this denotes a second derivative). To get a triple dot in math mode, use `\dddot`

C-quoteleft `accent-grave` – grave accent à (text mode only, use `\grave` for math)

S-C-asciitilde `accent-tilde` – tilde accent ã (text mode only, use `\tilde` for math)

C-apostrophe `accent-acute` – acute accent á (text mode only, use `\acute` for math)

S-C-asciicircum `accent-circumflex` – circumflex (caret) accent â (text mode only, use `\hat` for math).

Function keys. The new key S-F2 for creating a LaTeX file seems handy.

F2 `buffer-write` – Save current document.

S-F2 `buffer-export latex` – Write a LaTeX file for the current document.

F3 `find-replace` – Find and replace dialog.

C-F4 `buffer-close` – same as `C-w`.

M-F4 `lyx-quit` – `Alt-F4` to quit LyX is the W*ndows w*ndow manager's mnemonic.

`F4` to `F8` are used to switch fonts. Use `F4` to switch back to the normal font. The non-default font switches `F5 - F8` all work as toggles. They also work on the whole word if you put the cursor in the middle of the word, or if you select some text.

F4 `font-default` – stop using any special font

F5 `font-bold` – make **bold** text.

F6 `font-emph` – make *emphasized* text.

F7 `font-code` – make `typewritten` text.

F8 `font-noun` – make CAPS/SMALL CAPS text. (Used sometimes for people's names.)

`F9` is bound as "meta-prefix", same as the `Alt` key (useful e.g. if the keyboard has no working `Alt`).

Here are some Sciword-inspired mnemonics for frequently used math symbols. Many symbols start with a `C-s` sequence. Therefore `C-s` cannot be itself bound to anything.

M-apostrophe `math-insert \prime` – The "prime" symbol $\prime$ in math mode. This is frequently unnecessary: in most cases the normal apostrophe works just fine, e.g. $x' + 2x = 0$, but in some cases this would generate a double superscript error in LaTeX. For example: x'^2 (x prime squared) must be entered with the prime character.

C-s apostrophe `command-sequence math-superscript; math-insert \prime;` – Insert a prime as a superscript (see example above).

C-Up `math-insert ^` – Insert an upper index. Also, ^

C-Down `math-insert _` – Insert a lower index. Also, _

C-s d `math-insert \partial` – Partial derivative symbol ∂.

C-s e `math-insert \sum` – Summation symbol $\sum$ (not the same as the Greek letter uppercase Sigma Σ because it can resize and allows smart upper/lower limits).

C-s p `math-insert \prod` – Product symbol $\prod$ (not the same as the Greek letter uppercase Pi, Π).

C-s i `math-insert \infty` – Infinity ∞.

C-s x `math-insert \times` – Cross product $\times$.

C-s v `math-matrix 1 2` – Insert a stacked array . (Frequently useful in formulae.)

C-s m `math-matrix 3 3` – Insert a 3×3 matrix . (Then you can modify its size using the Edit ▷ Math menu.)

C-s S-plus `math-insert \dagger` – The "dagger" †.

C-s equal `math-insert \equiv` – "Identical equality" $\equiv$.

M-equal `math-insert \approx` – "Approximate equality" $\approx$.

M-minus `math-insert \sim` – The "of order" sign $\sim$.

C-minus `math-insert \rightarrow` – The arrow $\rightarrow$ as in $\lim_{x\rightarrow 0}$.

S-M-less `math-insert \leq` – Less-or-equal $\leq$.

S-M-greater `math-insert \geq` – Greater-or-equal $\geq$.

C-s S-less `math-insert \ll` – "Much less than" $\ll$ (useful in physics)

C-s S-greater `math-insert \gg` – "Much greater than" sign $\gg$.

A.3.4 Standard math bindings

M-m b	`math-insert \overline`
M-m d	`math-display`
M-m e	`math-insert ^`
M-m f	`math-insert \frac`
M-m g	`math-greek`
M-m h	`accent-circumflex`
M-m i	`math-insert \int`
M-m l	`math-limits`
M-m m	`math-mode`
M-m n	`math-number`
M-m o	`math-insert \oint`
M-m p	`math-insert \partial`
M-m r	`math-insert \sqrt`
M-m s	`math-insert \sqrt`
M-m u	`math-insert \sum`
M-m v	`math-insert \vec`
M-m x	`math-insert _`
M-m 8	`math-insert \infty`
M-m S-G	`math-greek-toggle`
M-m S-N	`math-nonumber`
M-m S-period	`accent-dot`
M-m S-asciitilde	`accent-tilde`
M-m S-apostrophe	`math-insert \prime`
M-m S-parenleft	`math-delim ( )`
M-m S-bracketleft	`math-delim [ ]`
M-m S-braceleft	`math-delim { }`
M-m S-less	`math-delim langle rangle`

M-m S-greater `math-delim rangle langle`

M-m S-bar `math-delim | |`

M-m S-plus `math-insert \pm`

M-m S-equal `math-insert \neq`

A.3.5 Other Accelerators

M-k o	`keymap-off`
M-k t	`keymap-toggle`
M-k x	`keymap-off`
M-k 1	`keymap-primary`
M-k 2	`keymap-secondary`