Customizing LyX

Chapter 1

Introduction

This manual covers the syntax of the `${HOME}/.lyxrc` file, as well as other customization features present in LyX.

Editor's Note: I'm not sure what else besides the .lyxrc should go in here, but I'm sure I'll find something....

WARNING!!!!! WARNING!!!!! WARNING!!!!

Danger, Will Robinson, Danger!

This file is ***very*** incomplete at the moment! It is borderline incoherent in places and has yet to undergo a thoughrough reorganization, as was done with `UserGuide.lyx`. Expect it to change!

Chapter 2

Creating Templates

Templates are created like usual documents. The only difference is, that usual documents contain all possible setting, including the fontscheme and the papersize. Usually a user don't want a template to overwrite his defaults in these cases. For that reason the designer of a template should remove the corresponding lyx-commands like `\fontscheme` or `\papersize` from the template lyx-file. This can be done with any simple text-editor, for example `vi` or `xedit`.

Chapter 3

The .lyxrc File

3.1 Customization with the lyxrc-file

LyX will read the file `$HOME/.lyxrc`, when it is invoked. This file may include various settings that are described below. Note that lines beginning with a "#" are just comments and not interpreted.

If `$HOME/.lyxrc` doesn't exist, LyX loads the system default file `(lyx_system_dir)/system.lyxrc` instead. Only system administrators should edit the `system.lyxrc` file. To design your own, copy `system.lyxrc` to `$HOME/.lyxrc` and use it as a template.

3.2 Bindings

command: `\bind`
format: `\bind <key combination> <lyx-function>`

`\bind` is used to, well, bind a function to a key. At present we're not sure if LyX support lyx-functions with parameters; anyway it will in the future. Both key combination and lyx-function must be enclosed in double quotes (""). Look in the bind-files in `(LyX_system_dir/bind/)` for examples on how to use this command. All the lyx-functions are listed in a later chapter.

3.3 Toolbar

The toolbar is also configurable from .lyxrc. This is done inside a *toolbar section:*

\begin_toolbar and \end_toolbar. The only legal command inside this section is \add.

\add has this format: \add <action> [<parameter>] (the parameter is optional), have a look in your lyxrc file to see how the standard toolbar could be declared.

3.4 Printer

There are a few commands to help you setup the printercommand from LyX.

- \printer
- \print_command
- \print_evenpage_flag
- \print_oddpage_flag
- \print_reverse_flag
- \print_landscape_flag
- \print_to_printer
- \print_to_file
- \print_file_extension
- \print_extra_options
- \print_spool_cmd

3.5 Misc

Different commands not belonging any particular place.

- \xdvi_extra_options
- \sgml_extra_options

3.6 Screen & Fonts

Ed. Note: I'll be adding things here at a later date. -jw

3.7 TEX

Some sites[1] still have the default LATEX version as 2.09. Though I can't imagine why this should be the default any longer, it will be necessary to deal with LYX in such an environment. Don't even *think* about "fixing" LYX to run under LATEX-2.09. Upgrading your LATEX will be easier, and is the right way to go, anyway. Most large systems should, even if they default to 2.09, have a version of LATEX 2_ε available, but perhaps using a different exectuable name. You can change the name of the appropriate latex command, say to the name "latex2e"[2] by placing the line

```
\latex_command latex2e
```

in your `.lyxrc` file.

3.8 LYX

[1] *like my office machine–Dave Johnson*
[2] *as it is at my office–Dave Johnson*

Chapter 4

Keybinding files

LYX allows you to define your own key sequences in your .lyxrc file. This section describes the way this is done. Also, you can define your own key binding file and include it. Check out the `system.lyxrc` file that came with the distribution.

Ed. Note: Eventually, all of that information will go in here. Right now, I have more pressing things to add to this document before v0.10 is officially released. -jw

4.1 Usage

The general syntax for key bindings is this:

\bind *"key-sequence" "action"*

here *key-sequence* consists of one or more keys separated by white space. A key is specified by giving its KeySym string optionally prefixed with 'S-', 'C-', 'M-' (shift, control, meta) or any combination of these. If you don't care about the setting of a particular modifier, specify it as '~S-', '~C-' or '~M-'; for example "~C-S-A" means "shift+A and maybe control".

The KeySym string can be found from the definitions in /usr/include/X11/keysymdef.h (with the leading 'XK_' removed) or by using the xev program.

The *action* is one of the names for LYX functions as defined in keybind.h (variable lyx_func_table).

Here are some examples:

- \bind "C-c C-l C-i" "layout-itemize"
- \bind "Delete" "delete-backward"
- \bind "KP_F1 KP_F3" "search"
- \bind "M-~S-less" "beginning-of-buffer"
- \bind "M-~S-greater" "end-of-buffer"

Note: If you use the Shift-modifier 'S-', you must also use the KeySym corresponding to the upper case letter and vice versa; eg in the above example 'M-greater' will not work if you have to press the shift key to get a '>'. Similarly, use 'C-S-A' instead of 'C-A' or 'C-S-a' (if the key can be produced with or without the shift modifier you can use the '~S-' modifier to tell LyX not to care about the setting of the shift key).

4.2 International Keyboard Support

Editor's Note: The following section is by Ivan Schreter. -jw

4.2.1 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>

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)	ñÑ
underbar (underb)	o̲
cedilla (ced)	çÇ
underdot (underd)	ọ
circumflex (circu)	âêîôû
circle (circl)	ÅůŮ
tie (tie)	o͡o
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 didn't have time.

Oh, and about keys: Backslash is escaped, so to enter it, you'll need double backslash. Also, quotes and # have different meaning. # marks comment, 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 fort your language, please mail it to me, so I can include it in the next keymaps 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).

4.3 A sample .Xmodmap

The deadkeys supported by X are defined in `/usr/X11R6/include/X11/keysymdef.h`. You should find appropriate locations for these on your keyboard. You only have to find places for the ones you actually use of course.

This is the .Xmodmap used by one of the LyX Team:

```
!  resets all modifiers
clear shift
add shift = Shift_L Shift_R
clear lock
add lock = Caps_Lock
clear control
add control = Control_L Control_R
clear mod1
add mod1 = Alt_L
clear mod2
add mod2 = Mode_switch
clear mod3
clear mod4
clear mod5

!  key codes
keycode   8 =
keycode   9 = Escape
keycode  10 = 1 exclam exclamdown onesuperior
keycode  11 = 2 quotedbl at twosuperior
keycode  12 = 3 numbersign sterling threesuperior
keycode  13 = 4 currency dollar cent
keycode  14 = 5 percent degree division
keycode  15 = 6 ampersand
keycode  16 = 7 slash braceleft onequarter
keycode  17 = 8 parenleft bracketleft onehalf
keycode  18 = 9 parenright bracketright threequarters
keycode  19 = 0 equal braceright
keycode  20 = plus question plusminus questiondown
keycode  21 = backslash dead_grave dead_acute dead_cedilla
keycode  22 = BackSpace
keycode  23 = Tab
keycode  24 = Q
keycode  25 = W
keycode  26 = e E ediaeresis Ediaeresis
keycode  27 = r R registered copyright
keycode  28 = t T thorn THORN
keycode  29 = y Y yen
keycode  30 = u U udiaeresis Udiaeresis
keycode  31 = i I idiaeresis Idiaeresis
keycode  32 = o O ograve Ograve
keycode  33 = p P paragraph
keycode  34 = aring Aring aacute Aacute
keycode  35 = dead_diaeresis dead_circumflex asciitilde dead_tilde
keycode  36 = Return
keycode  37 = Control_L
keycode  38 = a A agrave Agrave
keycode  39 = s S ssharp
keycode  40 = d D eth ETH
keycode  41 = f F ordfeminine masculine
```

```
keycode  42 = g G dead_abovering dead_abovedot
keycode  43 = H
keycode  44 = J
keycode  45 = K
keycode  46 = L
keycode  47 = oslash Ooblique odiaeresis Odiaeresis
keycode  48 = ae AE adiaeresis Adiaeresis
keycode  49 = bar section brokenbar
keycode  50 = Shift_L
keycode  51 = apostrophe asterisk grave dead_macron
keycode  52 = Z
keycode  53 = x X multiply
keycode  54 = c C ccedilla Ccedilla
keycode  55 = v V dead_doubleacute
keycode  56 = b B notsign
keycode  57 = n N ntilde Ntilde
keycode  58 = m M mu
keycode  59 = comma semicolon dead_breve dead_caron
keycode  60 = period colon periodcentered
keycode  61 = minus underscore hyphen macron
keycode  62 = Shift_R
keycode  63 = KP_Multiply
keycode  64 = Alt_L
keycode  65 = space nobreakspace
keycode  66 = Caps_Lock
keycode  67 = F1
keycode  68 = F2
keycode  69 = F3
keycode  70 = F4
keycode  71 = F5
keycode  72 = F6
keycode  73 = F7
keycode  74 = F8
keycode  75 = F9
keycode  76 = F10
keycode  77 = Num_Lock
keycode  78 = Scroll_Lock
keycode  79 = Home KP_7 KP_7 Home
keycode  80 = Up KP_8 KP_8 Up
keycode  81 = Prior KP_9 KP_9 Prior
keycode  82 = KP_Subtract
keycode  83 = Left KP_4 KP_4 Left
keycode  84 = Begin KP_5 KP_5 Begin
keycode  85 = Right KP_6 KP_6 Right
keycode  86 = KP_Add
keycode  87 = End KP_1 KP_1 End
keycode  88 = Down KP_2 KP_2 Down
keycode  89 = Next KP_3 KP_3 Next
keycode  90 = Insert KP_0 KP_0 Insert
keycode  91 = Delete KP_Decimal KP_Decimal Delete
keycode  92 = 0x1007ff00
keycode  93 =
keycode  94 = less greater guillemotleft guillemotright
keycode  95 = F11
keycode  96 = F12
keycode  97 = Home
keycode  98 = Up
keycode  99 = Prior
keycode 100 = Left
keycode 101 = Begin
keycode 102 = Right
keycode 103 = End
keycode 104 = Down
keycode 105 = Next
keycode 106 = Insert
keycode 107 = Delete
keycode 108 = KP_Enter
keycode 109 = Control_R
keycode 110 = Pause
keycode 111 = Print
keycode 112 = KP_Divide
keycode 113 = Mode_switch
keycode 114 = Break
keycode 136 = KP_7
keycode 137 = KP_8
keycode 138 = KP_9
keycode 139 = KP_4
```

```
keycode 140 = KP_5
keycode 141 = KP_6
keycode 142 = KP_1
keycode 143 = KP_2
keycode 144 = KP_3
keycode 145 = KP_0
keycode 146 = KP_Decimal
keycode 147 = Home
keycode 148 = Up
keycode 149 = Prior
keycode 150 = Left
keycode 151 = NoSymbol
keycode 152 = Right
keycode 153 = End
keycode 154 = Down
keycode 155 = Next
keycode 156 = Insert
keycode 157 = Delete
```

Chapter 5

A Printer Tutorial

5.1 Introduction

If you have a Linux system, and need to set up your printer, start by reading the file `Printing-HOWTO` [often in the directory `/usr/doc/FAQ/HOWTO/` or `/usr/doc/HOWTO/` depending on your system]. Also, you will need to install the Ghostscript package from your Linux CD-ROM, if you never did so.

Some Linux distributions now come with so-called "magic filters" and automatically set up your system to use them. You can use magic filters in place of or in addition to what we describe below, but it may be more difficult to take advantage of all of your printer's features. If your Linux distribution comes with some insta-setup tool for the printer, you might just want to use that, then look at the `Printing-HOWTO` when you're ready to customize things.

So if you're feeling adventurous, or have no other choice, read on!

5.2 The `/etc/printcap` File

5.2.1 Overview

The `/etc/printcap` file is how you tell Linux about your printer. Each entry defines a printer, giving it a name that you and other users call it by, and telling Linux how to handle output to it. You can actually have multiple "printers" on your system that all access the same physical printer, but use different features to do so.[1] Of course, you will need a printcap entry for each physical printer you have.

Here's a sample `/etc/printcap` file:

```
lp|Text on an Epson:\
        :lp=/dev/lp1:\
        :sd=/var/spool/lpd/lp:\
        :lf=/var/spool/lpd/lp/errs:\
        :mx#0:\
        :sh:\
        :sf:
draft|Postscript on an Epson Stylus Color IIs,
180dpi:\
        :lp=/dev/lp1:\
        :sd=/var/spool/lpd/draft:\
        :lf=/var/spool/lpd/draft/errs:\
        :if=/var/spool/lpd/draft/draft.filter:\
        :mx#0:\
        :sh:\
        :sf:
ps|Postscript on an Epson Stylus Color IIs, 360dpi:\
        :lp=/dev/lp1:\
        :sd=/var/spool/lpd/ps:\
        :lf=/var/spool/lpd/ps/errs:\
        :if=/var/spool/lpd/ps/ps.filter:\
        :mx#0:\
        :sh:\
        :sf:
```

Before we start explaining these heiroglyphs, we have an important message:

WARNING! WARNING! WARNING!

[1] Note from John Weiss: I do this with my Epson Stylus Color IIs. Since you need to change print heads to use color on the Stylus IIs, I set up a separate printer, `color`, for printing color documents. I also created a `draft` printer that uses lower print resolution and less-accurate printing features. So you see, there are lots of things you can do with the printcaps.

Use only one line between entries!!!
Make sure the line between entries is COMPLETELY BLANK!!!

Even if you leave a space on the blank line between entries, weird things can happen, and the system will not parse your `/etc/printcap` correctly.

Take a look at one of the entries in the above example. Each part of the entry is followed by a "`:`" character. The "\" character at the end of each line tells Linux that the entry continues on the next line. Don't put anything after the "\", not even a space. The "\" should come right after the "`:`" of the last part of the entry on the line you're breaking.

If you do choose to break up the `printcap` entry into multiple lines, be sure to put a "`:`" at the beginning of the *text* of the new line, after any space. For example, we could have written the first entry in the previous example as:

```
lp|Text:lp=/dev/lp1:sd=/var/spool/lpd/lp:lf=/var/spool/lpd/lp/errs:
```

Notice that there is only one "`:`" between each part of the entry. If we wanted to break this entry into two lines:

```
lp|Text:lp=/dev/lp1:sd=/var/spool/lpd/lp:\
      :lf=/var/spool/lpd/lp/errs:
```

We break the line at a "`:`" by adding a "\", starting a new line, adding some space, then adding a "`:`" before continuing. It's customary to indent the next line a bit if you break up a `printcap` entry into multiple lines.

Now we'll explain each part of the `/etc/printcap` entry. Note that there are other options you can add to an entry. We'll only cover the four basic ones here. See the man page for the `printcap` file for additional information.

5.2.2 The Printer Name

The first line of a `printcap` entry is the printer name. It contains the printer name, followed by a "|" character, followed by a text description of the printer, and ending, of course, with a "`:`" character. For example:

```
draft|OurPrinter in Draft Mode:\
```

is a printer called "`draft`" with the description "OurPrinter in Draft Mode" [Notice the "\" at the end of the line, indicating that we'll continue the entry on a new line.] The name `draft` is what you pass to many printing commands using the `-P` option. [The `lpr` program is such a command.]

It's traditional to name your main PostScript® printer "`ps`" and to call your text printer "`lp`".

5.2.3 The Printer Device

The next part of our sample `printcap` entries identifies the device to print to. In all of the example entries, there is a line:

```
:lp=/dev/lp1:\
```

[Again, notice the "\" at the end of the line, as well as the beginning "`:`" after the space we used to indent the text proper.]

The tag "`lp=`" takes the name of the printing device. In this case, it is `/dev/lp1`, the parallel port. There are printers that connect to the serial port. If you have one, you'd use `/dev/ttyS0` for the first serial port, `/dev/ttyS1` for the second, and so on. You'll also want to add other commands the the `printcap` entry, such as the printer's baud rate. See the `printcap` man page for more information.

Most printers, however, use the parallel port. In addition, the printer port on *most* PC-compatible computers will be `/dev/lp1`. If you want to know why, look in this footnote.[2] Your system may be different, however, so check to be sure which device is the parallel port.

[2]Why is `/dev/lp1` and *not /dev/lp0* the first printer port? It turns out that the I/O address of each parallel port is hard-coded into the Linux kernel. Address `0x3BC` is `/dev/lp0`; address `0x378` is `/dev/lp1`; and address `0x278` is `/dev/lp2`. Since most PC-compatible computers assign the first parallel port to the I/O address `0x378` by default, `/dev/lp1` will be the parallel port on most systems.

If you have more than one parallel port, the situation changes. Also, if you altered the I/O address of the parallel port for some reason, then one of the other `lp` devices will point to your parallel port. Other systems may just simply be weird. In any case, you can go into DOS and check out which I/O address your parallel port has. Then you can use that address to figure out whether your parallel port is `/dev/lp0`, `/dev/lp1`, or `/dev/lp2`.

5.2.4 The Spool Directory and Log File

In addition to naming your printer and telling Linux which device it is, you'll also need a directory for print spooling. The tag "`sd=`" identifies the spool directory for the printer. It's also a good idea to have a log file for errors. The tag "`lf=`" points to that.

Most systems put all of the print spools into the directory `/var/spool/lpd`; some use only `/var/spool`. In any case, each printcap entry traditionally has its own spool directory named after the printer. You'll need to log on a "root" and make these directories yourself. You'll also neeed to change to ownership on these directories, so that their group is the `daemon` group. You can do this with the command, "`chgrp daemon` *dirname*", where *dirname* is the name of the spool directory.

For example, look at the entry for the printer "`ps`" in the example. It contains the lines:

```
:sd=/var/spool/lpd/ps:\
:lf=/var/spool/lpd/ps/errs:\
```

This indicates that the spool directory is `/var/spool/lpd/ps` and that the log file is the file `/var/spool/lpd/ps/errs`. You should "create" this file using the command "`touch errs`" in the spool directory.

Similarly, the entry for the printer "`lp`" contains:

```
:sd=/var/spool/lpd/lp:\
:lf=/var/spool/lpd/lp/errs:\
```

Again, the spool directory is `/var/spool/lpd/lp` and the log file is `/var/spool/lpd/lp/errs`.

5.2.5 The Filter File

Unlike "`lp`", which simply passes text to the printer unformatted, the "`draft`" and "`ps`" printers need a filter [that's why they have an additional tag in them]. The `printcap` tag for the filter program is "`if=`" [stands for `input filter`, meaning input to the spool].

There are actually several tags for filter programs, allowing you to specify a different filter for different purposes. You can also select different filters via options to `lpr`, but let's not get into that here.

Let's look at the entry for the "`ps`" printer:

```
:if=/var/spool/lpd/ps/ps.filter:\
```

The file "`ps.filter`" is actually a shell script that resides in the spool directory for the "`ps`" printer. You could also specify the path and filename of a program; however, calling a program via a shell script allows you to specify command line options.

All filters, whether an acutal program or a shell script, should take their input from `stdin` [the standard input stream] and send their output to `stdout` [the standard output stream].

5.2.6 Additional Tags

There are three additional things you should put in your `printcap` entries. You can see them at the end of all three examples. They are:

```
:mx#0:\
:sh:\
:sf:
```

The "`mx#0`" means "don't limit the size of the file we can print." The "`sf`" prevents a form-feed from being sent after the document ends. Lastly, the "`sh`" prevents a header page from being sent at the beginning of each job. You'll probably always want to add these to your `printcap` entries.

5.3 Creating a Print Filter

5.3.1 Printing PostScript® Files Using Ghostscript

The easiest way to make your printer understand PostScript® is by using the Ghostscript program as a print filter. There are probably other ways, but this is the most common. If, for some reason, you don't want to or can't use Ghostscript, you're on your own [see the `Printing-HOWTO` for a start].

The first thing you should do is install Ghostscript. There should be a version in your Linux distribution at the very least. If not, you can certainly get one from the Linux FTP archive `sunsite.unc.edu`. Once you've installed it, execute it using

```
gs -h | more
```

from the command line. This lists the available output devices and formats. Here's a list to help decipher what Ghostscript device name corresponds to which printer:

- Apple Printers:

  appledmp Apple dot matrix printers

  iwhi Imagewriter in high-res mode

  iwlo Imagewriter in low-res mode

  iwlq Imagewriter LQ in 320x216 dpi mode

- Canon Printers:

  bj10e BubbleJet BJ-10e

  bj200 BubbleJet BJ-200

  bjc600 Color BubbleJet BJC-600, BJC-4000 and BJC-70.[3]

  bjc800 Color BubbleJet BJC-800

  lbp8 LBP-II8 laser printer

  lips3 LIPS-III laser printer in CaPSL mode

- DEC Printers:

  la50 LA50

  la70 LA70

  la70t LA70 with low-res text enhancement

  la75 LA75

  la75plus LA75 Plus

  ln03 LN03

  lj250 LJ250 color printer

  declj250 Alternate LJ250 driver0

- Epson Printers:

[3]Also good for Apple printers like the StyleWriter 2x00

ap3250 ActionPrinter AP3250 [and any printer using the ESC/P2 printer language]

escp2 All Epson printers using the ESC/P2 command set [thsi driver is possibly defunct. Use **ap3250** instead.]

epson All 9- and 24-pin dot matrix printers [and compatibles]

eps9mid All 9-pin dot matrix printers [and compatibles] in intermediate resolution

eps9hi All 9-pin dot matrix printers [and compatibles] in triple resolution

epsonc LQ-2550 color printer [also works for the Fujitsu 1200, 2400, and 3400 printers

st800 Stylus 800 inkjet

stcolor the Stylus Color, Stylus Color Pro, Stylus Color II and IIs color inkjet printers

- Hewlet Packard [HP] Printers:

lp2563 2563B line printer

cdeskjet DeskJet 500C with 1 bpp color

cdjcolor DeskJet 500C [and 540C] with 24 bpp color and Floyd-Steinberg dithering

cdjmono DeskJet 500C [and 510C, 520C, 540C], monochrome only.

cdj500 same as **cdjcolor**

cdj550 DeskJet 550C and 560C

deskjet DeskJet and DeskJet Plus

djet500 DeskJet 500

djet500c Alternate DeskJet 500C driver [does not work with the 550C or the 560C]

dnj650c DesignJet 650C

ljet2p LaserJet IId, LaerJet IIp, and LaserJet III* with TIFF compression

ljet3 LaserJet III* with Delta Row compression

ljet3d LaserJet IIID with duplex capability

ljet4 LaserJet 4 [default res=600dpi]

lj4dith LaserJet 4 with Floyd-Steinberg dithering

ljetplus LaserJet Plus

pj PaintJet XL

pjetxl Alternate PaintJet XL driver

pjxl PaintJet XL color

pjxl300 PaintJet XL300 color [and 1200C]

paintjet Alternate PaintJet color driver

- Tektronix Printers:

t4693d2 4693d color printer with 2 bits/RGB component [=6 bpp]

t4693d4 4693d color printer with 4 bits/RGB component [=12 bpp]

t4693d8 4693d color printer with 8 bits/RGB component [=24 bpp]

tek4696 4695 and 4696 inkjet plotters

- Miscellaneous Printers:

cp50 Mitsubishi CP50 color printer

ibmpro IBM 9-pin Proprinter

imagen Imagen ImPress printers

jetp3852 IBM Jetprinter ink-jet color printer (Model #3852)

m8510 C.Itoh M8510 printer

necp6 NEC P6/P6+/P60 printers at 360 dpi

nwp533 Sony Microsystems NWP533 laser printer [Sony only]

oce9050 OCE 9050 printer

oki182 Okidata MicroLine 182

okiibm Okidata MicroLine IBM-compatible printers

r4081 Ricoh 4081 laser printer

sj48 StarJet 48 inkjet printer

sparc SPARCprinter

xes Xerox XES printers (2700, 3700, 4045, etc.)

If the Ghostscript package that came with your distribution doesn't include your printer, you can always hunt down the source code and compile it yourself. If your printer isn't *directly* supported by Ghostscript at all, try choosing the a printer that's compatible.

Note that some printers require special command line arguments. For example, the `stcolor` driver requires that the file `stcolor.ps` be read before the input file. You should read the Ghostscript documentation for more details.

Next, create the filter. We suggest giving it the name "*pname*`.filter`", where "*pname*" is the name you gave the printer in its `printecap` entry. We also suggest putting the filter script in the printer's spool directory. The filter script should contain the following:

```
#!/bin/sh
/usr/bin/gs -q -dSAFER  -dNOPAUSE -sDEVICE=devname \
        [-rResNum ExtraOptions] -sOutputFile=-
[precmd.ps] -
```

where *devname* is the name of the Ghostscript device; *ResNum* is the resolution you want to use; *ExtraOptions* indicates any additional options to pass to Ghostscript; and "*precmd*`.ps`" is the name of a file to be read before the input [like the file `stcolor.ps` for the `stcolor` driver]. The "\" at the end of the first line is how you break up a command into multiple lines in a shell script.

Note that the above example assumes that `gs` is in the directory `/usr/bin`. Yous system may be different.

Different Ghostscript drivers treat the "`-r`" argument differently. Some are looking for a single number, for example, "`-r600`". Others want separate DPI resultions for the x and y directions, e.g. "`-r360x360`". The printer driver may take other options, as well. You'll need to consult the Ghostscript documentation for more information.

After you've created the printer filter, use "`chmod a+x`" on the script file to make it executable. [You need to be logged in as root to do this].

Here's an example of a Ghostscript print filter, used by John Weiss for his Epson Stylus Color IIs:

```
#!/bin/sh
/usr/bin/gs -q -dSAFER -sDEVICE=stcolor \
            -r360x360 -sDithering=fsmono \
            -dNOPAUSE -dnoWeave \
            -sOutputFile=- stcolor.ps -
```

This filter generates 360dpi monochrome output on an Epson Stylus Color IIs.[4]

5.3.2 If You Already Have a PostScript® Printer...

Many modern laser printers now come with PostScript® built-in. On some laser printers, PostScript® is actually the native printer language. If you have such a printer, you won't need a filter and can pass a PostScript® file directly to the printer. In contrast, if you want to print a text file, you'll need to format it in PostScript® somehow.

We know of three programs that translate plain ASCII text to PostScript®: `nenscript`, `a2ps`, and `mpage`. `mpage` can also print multiple pages on a single sheet of paper. You should create a filter script using one of these programs for the `lp` printer. [The `lp` printer is traditionally the name of the printer that prints ASCII text files.] Read the `Printing-HOWTO` for more information.

5.4 The Print Spool Daemon, `lpd`

The program that actually does the spooling and printing of files is `lpd`. It's also the program that reads the `/etc/printcap` file. Somewhere in your bootup scripts,[5] you'll need to execute `lpd` to begin print spooling every time you boot your machine. You can also execute `lpd` as a background job from the command line if you're logged on as root.

Once `lpd` is running on your system, you can control its behavior and that of the print spool using the `lpc` command. Again, you'll need to log in as root to do this.

[4] Note from John Weiss: This is my `ps` printer. I also use a similar filter for my `draft` printer, which has the option `-r180x180` instead of what you see above. And for my `color` printer, I use a different value for the `-sDithering` option.

[5] Some machines keep their bootup scripts in the `/etc/rc.d` directory. Others use `/etc/init.d`. You'll need to figure this out on your own.

5.5 Putting It All Together

We'll conclude this mini-HOWTO with the steps you need to follow to get a minimal setup. When finished, you'll have two printers, one named "`lp`" for printing ASCII text, and a second called "`ps`" for printing PostScript® files. The filter file for the "`ps`" printer [included below] works for any Epson-compatible printer [which includes almost all dot-matrix printers these days], using the default printing resolution.

1. Log in as "root" or use "`su root`".
2. Execute the following commands:
   (a) `cd /var/spool`
   (b) `mkdir lpd`
       [if it doesn't already exist]
   (c) `chgrp lp lpd`
       [You may need to use "`daemon`" instead of "`lp`"]
   (d) `cd lpd`
   (e) `mkdir lp ps`
       [only do this if these directories don't exist yet]
   (f) `touch lp/errs ps/errs`
   (g) `chgrp daemon lp ps lp/errs ps/errs`
3. Now edit the file `/etc/printcap`
   (a) If this file exists, either...
       i. ...rename the old version to something like "`/etc/printcap.orig`"
       ii. ...or, comment out everything in the file by putting a "`#`" character at the beginning of each line.
   (b) Put the following into `/etc/printcap`:

```
lp|Text printer:\
        :lp=/dev/lp1:\
        :sd=/var/spool/lpd/lp:\
        :lf=/var/spool/lpd/lp/errs:\
```

```
                :mx#0:sh:sf:
        ps|Postscript printer:\
                :lp=/dev/lp1:\
                :sd=/var/spool/lpd/ps:\
                :lf=/var/spool/lpd/ps/errs:\
                :if=/var/spool/lpd/ps/ps.filter:\
                :mx#0:sh:sf:
```

(c) MAKE SURE THAT:

i. the blank line between the two entries has nothing in it, not even whitespace characters.

ii. nothing follows the "\" character at the end of each line.

4. Next, create the filter script for the "`ps`" printer.

(a) Put the following into the file `/var/spool/lpd/ps/ps.filter`:

```
#!/bin/sh
/usr/bin/gs -q -dSAFER -dNOPAUSE \
            -sDEVICE=epson -sOutputFile=- -
```

[If you don't have an Epson-compatible dot-matrix printer, replace the "`epson`" with the appropriate Ghostscript device name. Also, if the program `gs` isn't in `/usr/bin`, put in the correct path.]

(b) Execute these commands:

i. `chmod a+x /var/spool/lpd/ps/ps.filter`

ii. `chgrp daemon /var/spool/lpd/ps/ps.filter`

5. Lastly, start up the printer daemon, `lpd`:

(a) If it's not running already, use "`lpd &`" to start it.

(b) If it is running already, execute "`lpc`". Then, from within `lpc`, do a "`restart all`" and a "`up all`" to activate the new print spools. [The command "`quit`" leaves `lpc`].

(c) Be sure to add the appropriate command to your bootup scripts so that `lpd` is executed every time you boot your system.

6. You may now log out as root and test your new printers to make sure they work. Issue the command "`lpr -Plp`" followed by the name of a short ASCII text file to verify that the "`lp`" printer works. Then use "`lpr -Pps`" followed by the name of a short PostScript® file to test the "`ps`" printer.

A word of caution: when testing "`ps`", if anything goes wrong, the system may end up passing garbage to your printer. If that happens, shut off your printer and use the `lpq` and `lprm` commands to kill the print job. After you kill the print job, you can turn your printer back on.

If either of these printers doesn't work, check everything again, consult the `man` pages for `printcap` and `lpd`, and read the `Printing-HOWTO` for help.

If everything does work, congratulations!

Chapter 6

The LyX-textclass format

This document describes the LyX-textclass format for a patched LyX-0.9.23. Later versions of LyX will possible support an enhanced format.

This is no tutorial but a short reference. When you plan to realize a new textclass, it might be very helpful to look at the example textclasses contributed with LyX. If you use a nice LaTeX textclass that might be of interest for others, too, and have a nice corresponding LyX textclass, feel free to contribute the stuff to us, so we may put it into the distribution.

The possible values are printed in brackets after the feature's name. The default value if a feature isn't specified inside a textclass-description is typeset *italic*. If the argument is a datatype like "string" or "float", the default is shown like this: "float=*default* ".

6.1 General textclass parameters

Columns [*1*, 2] Wether the class-default should be one- or twocolumn. Can be changed in the *Layout->Document* popup.

Sides [*1*, 2] Wether the class-default should be printing on one or both sides of the paper. Can be changed in the *Layout->Document* popup.

PageStyle [*plain*, empty, headings] The class-default pagestyle. Can be changed in the *Layout->Document* popup.

MaxCounter [*Counter_Chapter*, Counter_Section, Counter_Subsection, Counter_Subsubsection, Counter_Paragraph, Counter_Subparagraph, Counter_EnumI, Counter_EnumII, Counter_EnumIII, Counter_EnumIV] The highest level of automatic enumeration. Useful values are Counter_Chapter for a book or report and Counter_Section for an article. If there is no enumeration needed at all, for example inside a letter-class, MaxCounter should be set to Counter_EnumI.

DefaultFamily [*Roman*, Sans, Typewriter] This family is used if a specific layout's family is defined as "environment_default" and there are no environments around this paragraph, or these environments also define the family as "environment_default".

DefaultSeries [*Medium*, Bold] Similar to DefaultFamily.

DefaultShape [*Up*, Italic, SmallCaps] Similar to DefaultFamily.

DefaultSize [tiny, small, *normal*, large, larger, largest, huge, giant] Similar to DefaultFamily.

6.2 Specific Layouts

A layout description looks like that:

`Style` <*Name of Layout*>
 <*Description*>
`End`

with <*Description*> is the following:

Border [*Static*, Manual, Dynamic, First_Dynamic, Right_Address_Box] The kind of border the layout has on the left side. "Static" means just a static border. "Manual" means that the left border depends on the string entered in the *layout->paragraph* popup. This is used to typeset nice lists without tabulators. "Dynamic" means that the border 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 border (as wide as "5.4.3.2.1") than "3.2 Very long headline", even if other wordprocessors 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. Used for example for descriptions. "Right_Address_Box" means the border is chosen in a way that the longest row of this paragraph fits to the right border. Used to typeset an address on the very right of a paper.

LatexType [*Paragraph*, Command, Environment, Item_Environment, List_Environment] How the layout should be translated into LaTeX. "Paragraph" means nothing special. "Command" means "\foo{...}" and "Environment" means "\begin{foo}...\end{foo}". "Item_Environment" is the same than "Enviroment", except that a "\item" is generated for each paragraph of this environment. "List_Environment" is the same than "Item_Environment", except that the labelwidthstring is passed as an argument to the environment. The labelwidthstring can be defined in the *layout->paragraph* popup.

LatexName The name of the corresponding LaTeX-stuff. Either the environment or command name.

NextNoIndent [1, *0*] Wether the following (standard-) 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 means "MM" that the paragraph is indented with the width of "MM" in the normal font. This way was chosen that the look is the same with each used screen font. The Parindent will be fix 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. So the look stays the same with different screen fonts.

TopSep [float=0] The vertical space the very first of a chain of paragraphs with this layout is separated from the previous one. If the previous paragraph has another layout, the separations are not simple added, but the maximum is taken.

BottomSep [float=0] The same like "TopSep" for the very last one.

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 are separated with the environment's "Parsep". But the whole items of the environment are additionally separated with this "Itemsep".

Leftborder [string=""] If you put layouts into environments, the leftborders are not simply added, but added with factor "4/(depth+1)*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.

Rightborder [string=""] Similar to LeftBorder.

Labeltype [*No_Label*, Manual, Static, Top_Environment, Centered_Top_Environment, Sensitive, Counter_Chapter, Counter_Section, Counter_Subsection, Counter_Subsubsectio Counter_Paragraph, Counter_Subparagraph, Counter_EnumI, Counter_EnumII, Counter_EnumIII, Counter_EnumIV] "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. "Senistive" is a special case for the caption-labels "Figure" and "Table". Sensitive means the (hardcoded) labelstring depends on the kind of float. The counter labeltypes should be self explanatory.

Labelsep [string=""] The horizontal space between the label and the textbody. Only used for labels that are not above the textbody.

LabelBottomsep [float=0] The vertical space between the label and the textbody. Only used for labels that are above the textbody ("Top_Environment", "Centered_Top_Environment").

Labelstring [string=""] The string used a label in static labeltypes.When the border is "manual" this string is also used as a suggestion for the labelwidthstring that can be set in the *layout->paragraph* popup.

Align [*block*, left, right, center]

AlignPossible [[*block*, left, right, center] "," [*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 top"-switch of the *layout->paragraph* popup can be set when initializing a paragraph with this layout. Used for the lyxtitle-layout.

Fill_Bottom [*0*,1] Similar to "Fill_Top".

Newline [0, *1*] Wether newlines are translated into LaTeX newlines ("\\") or not. The translated can be switched of to allow more comfortable LaTeX editing inside LyX.

FreeSpacing [*0*, 1] Usually LyX doesn't allow to insert more than one blank between words, since a blank 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 programming-code or plain LaTeX code. For this reason "FreeSpacing" could be enabled. Note that LyX will create protected blanks for the additional blanks when in another mode than LaTeX-mode.

Baselinestretch [float = 1.5] The baselinestretch inside a paragraph. 1.5 looks on the screen similar to 1.0 on a LaTeX-print. That's why the default is 1.5. For real 1.5 baselinestretch you should set this value for about 3.0.

Font The font used for the textbody. See next section.

LabelFont The font used for the label. See next section.

6.3 Font description

A font description looks like that:

`Font` *or* `LabelFont`
<*Description*>
`EndFont`

with <*Description*> is the following:

Family [*Environment_Default*, Roman, Sans, Typewriter]

Series [*Environment_Default*, Medium, Bold]

Shape [*Environment_Default*, Up, Italic, SmallCaps]

Size [*Environment_Default*, tiny, small, normal, large, larger, largest, huge, giant]

Latex [*no_latex*, latex, formula] "no_latex" means LyX will translate everything into LaTeX. "latex" means a red screeprint and no translation at all. "formula" means blue screenprint. If the layout's latex flag is "no_latex" LyX would also put a string of "formula"-character into a LaTeX math environment. This is not the case if the layout itself is "formula". In this case the layout itself is responsible for switching to math mode.

Chapter 7

The LyX Server

Editor's Note: I have not idea how to use the LyX server, nor does anyone else on the doc-team have a clue. All of this is just a shameless copy from the old documentation. I don't guarantee that any of this is correct, nor do I even know!

The LyX server enables other programs to talk to LyX, invoke LyX commands and retrieve information about the LyX internal state. This 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 .lyxrc file by adding a line like:

\serverpipe "/home/myhome/.lyxpipe"

LyX will add the '.in' and '.out' to create the pipes. The above line also has the effect of activating the LyXserver. 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 .lyxrc between the start of the programs.

There are three forms of communication between LyX and a client program:

- lyx call (the client tells LyX to invoke a particular function)
- lyx request (the client requests a certain information from LyX)
- notification (LyX informs a client about some event that has happened)

7.1 LyX call

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 in the \bind command above.

argument is an optional argument which is meaningful only to some functions (eg 'insert-self').

7.2 LyX request

This is just like a LyX call except that the request functions will make LyX send an answer (through the out-pipe) instead of performing an action. The format of the answer is:

INFO:*clientname*:*function*:*data*

Here *clientname* and *function* are just echoed from the request; *data* contains the actual information for the client.

7.3 Notification

LyX can notify clients of events going on asynchronously. Currently it will only do this if the user binds a key sequence to 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' command set: 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.

Note: if LyX crashes, it may not manage to remove the pipes; in this case you should remove them manually. If LyX starts and the pipes exist already, it will not start any server.

Examples:

echo "LYXCMD:test:beginning-of-buffer:" >~/.lyxpipe.in

echo "LYXCMD:test:get-xy:" >~/.lyxpipe.in
read a <~/.lyxpipe.out
echo $a

Remainder of Document is Incomplete.