Linux Shortcuts and Commands:

Linux Newbie Administrator Guide
by Stan and Peter Klimas

This is a practical selection of the commands we use most often. Press `<Tab>` to see the listing of all available command (on your PATH). On my small home system, it says there are 2595 executables on my PATH. Many of these "commands" can be accessed from your favourite GUI front-end (probably KDE or Gnome) by clicking on the right menu or button. They can all be run from the command line. Programs that require GUI have to be run from a terminal opened under a GUI.

Legend:
<> = single special or function key on the keyboard. For example <Ctrl> indicates the "control" key.
italic = name of the file or variable you probably want to substitute with your own.
`fixed width` = in-line Linux commands and filenames.

Notes for the UNIX Clueless:
1. LINUX IS CASE-SENSITIVE. For example: Netscape, NETSCAPE and nEtscape are three different commands. Also my_filE, my_file, and my_FILE are three different files. Your user login name and password are also case sensitive. (This goes with the tradition of UNIX and the "c" programming language being case sensitive.)
2. Filenames can be up to 256 characters long and can contain letters, numbers, "." (dot), "_" (underscore), "-" (dash), plus some other not recommended characters.
3. Files with names starting with "." are normally not shown by the `ls` (list) or dir commands. Think of these files as "hidden". Use `ls -a` (list with the option "all") to see these files.
4. "/" is an equivalent to DOS "\" (root directory, meaning the parent of all other directories).
5. Under Linux, all directories appear under a single directory tree (there are no DOS-style drive letters).
6. In a configuration file, a line starting with # is a comment.

7.1 Linux essential shortcuts and sanity commands

`<Ctrl><Alt><F1>`
Switch to the first text terminal. Under Linux you can have several (6 in standard setup) terminals opened at the same time.

`<Ctrl><Alt><Fn>` (n=1..6)
Switch to the nth text terminal.

`tty`
Print the name of the terminal in which you are typing this command.

`<Ctrl><Alt><F7>`
Switch to the first GUI terminal (if X-windows is running on this terminal).

`<Ctrl><Alt><Fn>` (n=7..12)
Switch to the nth GUI terminal (if a GUI terminal is running on screen n-1). On default, nothing is running on terminals 8 to 12, but you can run another server there.

`<Tab>`
(In a text terminal) Autocomplete the command if there is only one option, or else show all the available options.
THIS SHORTCUT IS GREAT! It even works at LILO prompt!

`<ArrowUp>`
Scroll and edit the command history. Press <Enter> to execute.

`<Shift><PgUp>`
Scroll terminal output up. Work also at the login prompt, so you can scroll through your bootup messages.

`<Shift><PgDown>`
Scroll terminal output down.

`<Ctrl><Alt><+>`
(in X-windows) Change to the next X-server resolution (if you set up the X-server to more than one resolution). For multiple resolutions on my standard SVGA card/monitor, I have the following line in the file `/etc/X11/XF86Config` (the first resolution starts on default, the largest determines the size of the "virtual screen"):

```
Modes "1024x768" "800x600" "640x480" "512x384" "480x300" "400x300" "1152x864"
```

`<Ctrl><Alt><->`
(in X-windows) Change to the previous X-server resolution.

`<Ctrl><Alt><BkSpc>`
(in X-windows) Kill the current X-windows server. Use if the X-windows server crushes and cannot be exited normally.

`<Ctrl><Alt><Del>`
Shut down the system and reboot. This is the normal shutdown command for a user at the text-mode console. Don't just press the "reset" button for shutdown!

`<Ctrl>c`
Kill the current process (mostly in the text mode for small applications).

`<Ctrl>d`
Log out from the current terminal. See also the next command.

`<Ctrl>d`
Send [End-of-File] to the current process. Don't press it twice else you also log out (see the previous command).

`<Ctrl>s`
Stop the transfer to the terminal.

`<Ctrl>q`
Resume the transfer to the terminal. Try if your terminal mysteriously stops responding.

`<Ctrl>z`
Send the current process to the background.

`exit`
Logout. I can also use `logout` for the same effect. (If you have started a second shell, e.g., using `bash` the second shell will be exited and you will be back in the first shell, not logged out.)

`reset`
Restore a screwed-up terminal (a terminal showing funny characters) to default setting. Use if you tried to "cat" a binary file. You may not be able to see the command as you type it.

`<MiddleMouseButton>`
Paste the text which is currently highlighted somewhere else. This is the normal "copy-paste" operation in Linux. (It doesn't work with Netscape and WordPerfect which use the MS Windows-style "copy-paste". It does work in the text terminal if you enabled "gpm" service using "setup".) Best used with a Linux-ready 3-button mouse (Logitech or similar) or else set "3-mouse button emulation").

`~`
(tilde) My home directory (normally the directory `/home/my_login_name`). For example, the command `cd ~/my_dir` will change my working directory to the subdirectory "*my_dir*" under my home directory. Typing just "cd" alone is an equivalent of the command "cd ~".

`.`
(dot) Current directory. For example, `./my_program` will attempt to execute the file "my_program" located in your current working directory.

`..`
(two dots) Directory parent to the current one. For example, the command `cd ..` will change my current working directory one one level up.

7.2 Common Linux commands--system info

`pwd`
Print working directory, i.e., display the name of my current directory on the screen.

`hostname`
Print the name of the local host (the machine on which you are working).
Use `netconf` (as root) to change the name of the machine.

`whoami`
Print my login name.

`id` *`username`*
Print user id (uid) and his/her group id (gid), effective id (if different than the real id) and the supplementary groups.

`date`
Print or change the operating system date and time. E.g., I could change the date and time to 2000-12-31 23:57 using this command:

```
date 123123572000
```

To set the hardware (BIOS) clock from the system (Linux) clock, use the command (as root) `setclock`

`time`
Determine the amount of time that it takes for a process to complete + other info. Don't confuse it with the `date` command. E.g. I can find out how long it takes to display a directory content using:

```
time ls
```

```
who
```

Determine the users logged on the machine.

```
rwho -a
```

(=remote who) Determine all users logged on your network. The rwho service must be enabled for this command to run. If it isn't, run setup as root to enable "rwho".

```
finger user_name
```

System info about a user. Try: `finger root`

```
last
```

Show listing of users last logged-in on your system.

```
history | more
```

Show the last (1000 or so) commands executed from the command line on the current account. The "| more" causes the display to stop after each screenful.

```
uptime
```

Show the amount of time since the last reboot.

```
ps
```

(=print status) List the processes currently run by the current user.

```
ps axu | more
```

List all the processes currently running, even those without the controlling terminal, together with the name of the user that owns each process.

```
top
```

Keep listing the currently running processes, sorted by cpu usage (top users first). In KDE, you can get GUI-based Ktop from "K"menu under "System"-"Task Manager" (or by executing "ktop" in an X-terminal).

```
uname -a
```

(= Unix name with option "all") Info on your (local) server. I can also use `guname` (in X-window terminal) to display the info more nicely.

```
free
```

Memory info (in kilobytes).

```
df -h
```

(=disk free) Print disk info about all the filesystems (in human-readable form)

```
du / -bh | more
```

(=disk usage) Print detailed disk usage for each subdirectory starting at the "/" (root) directory (in human legible form).

```
cat /proc/cpuinfo
```

Cpu info--it show the content of the file `cpuinfo`. Note that the files in the `/proc` directory are not real files--they are hooks to look at information available to the kernel.

```
cat /proc/interrupts
```

List the interrupts in use.

```
cat /proc/version
```

Linux version and other info

```
cat /proc/filesystems
```

Show the types of filesystems currently in use.

```
cat /etc/printcap
```

Show the setup of printers.

```
lsmod
```

(As root. Use `/sbin/lsmod` to execute this command when you are a non-root user.) Show the kernel modules currently loaded.

```
set|more
```

Show the current user environment.

```
echo $PATH
```

Show the content of the environment variable "PATH". This command can be used to show other environment variables as well. Use "set" to see the full environment.

```
dmesg | less
```

Print kernel messages (the content of the so-called kernel ring buffer). Press "q" to quit "less". Use `less /var/log/dmesg` to see what "dmesg" dumped into this file right after the last system bootup.

7.3 Basic operations

```
any_command --help |more
```

Display a brief help on a command (works with most commands). "--help" works similar to DOS "/h" switch. The "more" pipe is needed if the output is longer than one screen.

`man topic`

Display the contents of the system manual pages (help) on the topic. Try `man man` first. Press "q" to quit the viewer. The command `info topic` works similar and may contain more up-to-date information. Manual pages can be hard to read. Try `any_command --help` for short, easy to digest help on a command. If more info needed, have a look to the directory `/usr/doc`. To display manual page from a specific section, I may use something like in this example: `man 3 exit` (this displays an info on the command `exit` from section 3 of the manual pages).

`apropos topic`

Give me the list of the commands that have something to to do with my topic.

`help command`

Display brief info on a bash (shell) build-in command.

`ls`

List the content of the current directory. Under Linux, the command "dir" is an alias to ls. Many users have "ls" to be an alias to "ls --color".

`ls -al |more`

List the content of the current directory, all files (also those starting with a dot), and in a long form. Pipe the output through the "more" command, so that the display pauses after each screenful.

`cd directory`

Change directory. Using "cd" without the directory name will take you to your home directory. "cd -" will take you to your previous directory and is a convenient way to toggle between two directories. "cd .." will take you one directory up.

`cp source destination`

Copy files. E.g., `cp /home/stan/existing_file_name .` will copy a file to my current working directory. Use the "-r" option (for recursive) to copy the contents of whole directories, e.g. , `cp -r my_existing/dir/ ~` will copy a subdirectory under my current working directory to my home directory.

`mcopy source destination`

Copy a file from/to a DOS filesystem (no mounting necessary). E.g., `mcopy a:\autoexec.bat ~/junk` . See `man mtools` for related commands: mdir, mcd, mren, mmove, mdel, mmd, mrd, mformat ....

`mv` *`source destination`*
Move or rename files. The same command is used for moving and renaming files and directories.

`ln` *`source destination`*
Create a hard link called *destination* to the file called *source*. The link appears as a copy of the original files, but in reality only one copy of the file is kept, just two (or more) directory entries point to it. Any changes the file are automatically visible throughout. When one directory entry is removed, the other(s) stay(s) intact. The limitation of the hard links are: the files have to be on the same filesystem, hard links to directories or special files are impossible.

`ln -s` *`source destination`*
Create a symbolic (soft) link called "destination" to the file called "source". The symbolic link just specifies a path where to look for the file. In contradistinction to hard links, the source and destination don't not have to tbe on the same filesystem. In comparison to hard links, the drawback of symbolic links are: if the original file is removed, the link is "broken", symbolic links can also create circular references (like circular references in spreadsheets or databases, e.g., "a" points to "b" and "b" points back to "a").

`rm` *`files`*
Remove (delete) files. You must own the file in order to be able to remove it. On many systems, you will be asked or confirmation of deleation, if you don't want this, use the "-f" (=force) option, e.g., `rm -f *` will remove all files in my current working directory, no questions asked.

`mkdir` *`directory`*
Make a new directory.

`rmdir` *`directory`*
Remove an empty directory.

`rm -r` *`files`*
(recursive remove) Remove files, directories, and their subdirectories. Careful with this command as root--you can easily remove all files on the system with such a command executed on the top of your directory tree, and there is no undelete in Linux (yet). But if you really wanted to do it (reconsider), here is how (as root): `rm -rf /*`

`cat` *`filename`* `| more`
View the content of a text file called "filename", one page a time. The "|" is the "pipe" symbol (on many American keyboards it shares the key with "\") The pipe makes the

output stop after each screenful. For long files, it is sometimes convenient to use the commands head and tail that display just the beginning and the end of the file. If you happened to use "cat" a binary file and your terminal displays funny characters afterwards, you can restore it with the command "reset".

```
less filename
```
Scroll through a content of a text file. Press q when done. "Less" is roughly equivalent to "more" , the command you know from DOS, although very often "less" is more convenient than "more".

```
pico filename
```
Edit a text file using the simple and standard text editor called `pico`.

```
pico -w filename
```
Edit a text file, while disabling the long line wrap. Handy for editing configuration files, e.g. `/etc/fstab`.

```
find / -name "filename"
```
Find the file called "filename" on your filesystem starting the search from the root directory "/". The "filename" may contain wildcards (*,?).

```
locate filename
```
Find the file name of which contains the string "filename". Easier and faster than the previous command but depends on a database that normally rebuilds at night.

```
./program_name
```
Run an executable in the current directory, which is not on your PATH.

```
touch filename
```
Change the date/time stamp of the file *`filename`* to the current time. Create an empty file if the file does not exist.

```
xinit
```
Start a barebone X-windows server (without a windows manager).

```
startx
```
Start an X-windows server and the default windows manager. Works like typing "win" under DOS with Win3.1

```
startx -- :1
```
Start another X-windows session on the display 1 (the default is opened on display 0).

You can have several GUI terminals running concurrently. Switch between them using <Ctrl><Alt><F7>, <Ctrl><Alt><F8>, etc.

```
xterm
```

(in X terminal) Run a simple X-windows terminal. Typing `exit` will close it. There are other, more advanced "virtual" terminals for X-windows. I like the popular ones: `konsole` and `kvt` (both come with kde) and `gnome-terminal` (comes with gnome). If you need something really fancy-looking, try `Eterm`.

```
xboing
```

(in X terminal). Very nice, old-fashioned game. Many small games/programs are probably installed on your system. I also like `xboard` (chess).

```
shutdown -h now
```

(as root) Shut down the system to a halt. Mostly used for a remote shutdown. Use <Ctrl><Alt><Del> for a shutdown at the console (which can be done by any user).

```
halt
reboot
```

(as root, two commands) Halt or reboot the machine. Used for remote shutdown, simpler to type than the previous command.

Network apps

```
netscape
```

(in X terminal) Run netscape (requires a separate Netscape installation). The current versions of Netscape (4.x) are known to be big and buggy. They occasionally crash by vanishing (no other harm done). Also, when not connected to the network , Netscape likes to refuse to do anything (looks like it hanged)-it revives when you connect.

```
netscape -display host:0.0
```

(in X terminal) Run netscape on the current machine and direct the output to machine named "host" display 0 screen 0. Your current machine must have a permission to display on the machine "host" (typically given by executing the command `xhost` *`current_machine_name`* in the xterminal of the machine host. Other X-windows program can be run remotely the same way.

```
lynx file.html
```

View an html file or browse the net from the text mode.

`pine`
A good text-mode mail reader. Another good and standard one is `elm`. Your Netscape mail will read the mail from your Internet account. `pine` will let you read the "local" mail, e.g. the mail your son or a cron process sends to you from a computer on your home network. The command `mail` could also be used for reading/composing mail, but it would be inconvenient--it is meant to be used in scripts for automation.

`elm`
A good tex-mode mail reader. See the previous command.

`mutt`
A really basic but extremally useful and fast mail reader.

`mail`
A basic operating system tool for e-mail. Look at the previous commands for a better e-mail reader. `mail` is good if you wanted to send an e-mail from a shell script.

`licq`
(in X term) An icq "instant messaging" client. Another good one is `kxicq`. Older distributions don't have an icq client installed, you have to do download one and install it.

`talk` *username1*
Talk to another user currently logged on your machine (or use "`talk` *username1@machinename*" to talk to a user on a different computer) . To accept the invitation to the conversation, type the command "`talk` *username2*". If somebody is trying to talk to you and it disrupts your work, your may use the command "`mesg n`" to refuse accepting messages. You may want to use "`who`" or "`rwho`" to determine the users who are currently logged-in.

`mc`
Launch the "Midnight Commander" file manager (looks like "Norton Commander" for Linux).

`telnet` *server*
Connect to another machine using the TELNET protocol. Use a remote machine name or IP address. You will be prompted for your login name and password--you must have an account on the remote machine to login. Telnet will connect you to another machine and let you operate on it as if you were sitting at its keyboard (almost). Telnet is not very secure--everything you type goes in open text, even your password!

```
rlogin server
```

(=remote login) Connect to another machine. The login name/password from your current session is used; if it fails you are prompted for a password.

```
rsh server
```

(=remote shell) Yet another way to connect to a remote machine. The login name/password from your current session is used; if it fails you are prompted for a password.

```
ftp server
```

Ftp another machine. (There is also ncftp which adds extra features and gftp for GUI .) Ftp is good for copying files to/from a remote machine. Try user "anonymous" if you don't have an account on the remote server. After connection, use "?" to see the list of available ftp commands. The essential ftp command are: ls(see the files on the remote system), ASCII, binary (set the file transfer mode to either text or binary, important that you select the proper one), get (copy a file from the remote system to the local system), mget (get many files at once), put (copy a file from the local system to the remote system), mput (put many files at once), bye (disconnect). For automation in a script, you may want to use ncftpput and ncftpget, for example:

```
ncftpput -u my_user_name -p my_password -a remote.host.domain remote_dir
*local.html
```

```
minicom
```

Minicom program (looks like "Procomm for Linux").

File (de)compression

```
tar -zxvf filename.tar.gz
```

(=tape archiver) Untar a tarred and compressed tarball (*.tar.gz or *.tgz) that you downloaded from the Internet.

```
tar -xvf filename.tar
```

Untar a tarred but uncompressed tarball (*.tar).

```
gunzip filename.gz
```

Decompress a zipped file (*.gz" or *.z). Use gzip (also zip or compress) if you wanted to compress files to this file format.

```
bunzip2 filename.bz2
```

(=big unzip) Decompress a file (*.bz2) zipped with bzip2 compression utility. Used for big files.

`unzip` *`filename.zip`*
Decompress a file (*.zip) zipped with a compression utility compatible with PKZIP for DOS.

`unarj e` *`filename.arj`*
Extract the content of an *.arj archive.

`uudecode -o` *`outputfile filename`*
Decode a file encoded with `uuencode`. uu-encoded files are typically used for transfer of non-text files in e-mail (uuencode transforms any file into an ASCII file).

7.4 Process control

`ps`
(=print status) Display the list of currently running processes with their process IDs (PID) numbers. Use `ps axu` to see all processes currently running on your system (also those of other users or without a controlling terminal), each with the name of the owner. Use "top" to keep listing the processes currently running.

`fg` *`PID`*
Bring a background or stopped process to the foreground.

`bg` *`PID`*
Send the process to the background. Opposite to fg. The same can be accomplished with <Ctrl>z. If you have stopped jobs, you have to type `exit` twice in row to log out.

`any_command``&`
Run any command in the background (the symbol "&" means "run the proceeding command in the background").

`batch` *`any_command`*
Run any command (usually one that is going to take more time) when the system load is low. I can logout, and the process will keep running.

`at 17:00`
Execute a command at a specified time. You will be prompted for the command(s) to run, until you press <Ctrl>d.

`kill` *`PID`*
Force a process shutdown. First determine the PID of the process to kill using ps.

`killall` *program_name*
Kill program(s) by name.

`xkill`
(in an xwindow terminal) Kill a GUI-based program with mouse. (Point with your mouse cursor at the window of the process you want to kill and click.)

`lpc`
(as root) Check and control the printer(s). Type "?" to see the list of available commands.

`lpq`
Show the content of the printer queue. Under KDE (X-Windows), you may use GUI-based "Printer Queue" available from "K"menu-Utilities.

`lprm` *job_number*
Remove a printing job "job_number" from the queue.

`nice` *program_name*
Run *program_name* adjusting its priority. Since the priority is not specified in this example, it will be adjusted by 10 (the process will run slower), from the default value (usually 0). The lower the number (of "niceness" to other users on the system), the higher the priority. The priority value may be in the range -20 to 19. Only root may specify negative values. Use "top" to display the priorities of the running processes.

`renice -1` *PID*
(as root) Change the priority of a running process to -1. Normal users can only adjust processes they own, and only up from the current value (make them run slower).

`<Ctrl>c`, `<Ctrl>z`, `<Ctrl>s`, and `<Ctrl>q` also belong to this chapter but they were described previously. In short they mean: stop the current command, send the current command to the background, stop the data transfer, resume the data transfer.

7.5 Basic administration commands

`printtool`
(as root in X-terminal) Configuration tool for your printer(s). Settings go to the file `/etc/printcap`.

`setup`
(as root) Configure mouse, soundcard, keyboard, X-windows, system services. There

are many distibution-specific configuration utilities, `setup` is the default on RedHat. Mandrake 7.0 offers very nice `DrakConf` .

```
linuxconfig
```

(as root, either in text or graphical mode). You can access and change hundreds of setting from it. Very powerful--don't change too many things at the same time, and be careful with changing entries you don't understand.

```
xvidtune
```

(in X-terminal). Adjust the settings of the graphical display for all resolutions so as to eliminate black bands, shift the display right/left/up/down, etc. (First use the knobs on your monitor to fit your text mode correctly on the screen.) To make the changes permanent, display the frequencies on the screen and transfer them to the setup file `/etc/X11/XF86Config`.

```
alias ls="ls --color=tty"
```

Create an alias for the command "ls" to enhance its format with color. In this example, the alias is also called "ls" and the "color" option is only envoke when the output is done to a terminal (not to files). Put the alias into the file `/etc/bashrc` if you would like the alias to be always accessible to all users on the system. Type "`alias`" alone to see the list of aliases on your system.

```
adduser user_name
```

Create a new account (you must be root). E.g., `adduser barbara` Don't forget to set up the password for the new user in the next step. The user home directory is `/home/`*`user_name`*.

```
useradd user_name
```

The same as the command " `adduser` *`user_name`* ".

```
userdel user_name
```

Remove an account (you must be a root). The user's home directory and the undelivered mail must be dealt with separately (manually because you have to decide what to do with the files).

```
groupadd group_name
```

Create a new group on your system. Non-essential but can be handy even on a home machine with a small number of users.

```
passwd
```

Change the password on your current account. If you are root, you can change the password for any user using: `passwd` *`user_name`*

```
chmod perm filename
```

(=change mode) Change the file access permission for the files you own (unless you are root in which case you can change any file). You can make a file accessible in three modes: read (r), write (w), execute (x) to three classes of users: owner (u), members of the same group as the owner (g), others on the system (o). Check the current access permissions using:

```
ls -l filename
```

If the file is accessible to all users in all modes it will show:

```
rwxrwxrwx
```

The first triplet shows the file permission for the owner of the file, the second for his/her group, the third for others. A "no" permission is shown as "-".

E.g., this command will **add** the permission to read the file "junk" to all (=user+group+others):

```
chmod a+r junk
```

This command will remove the permission to execute the file junk from others:

```
chmod o-x junk
```

Also try here for more info.

You can set the default file permissions for the news files that you create using the command `umask` (see `man umask`).

```
chown new_ownername filename
chgrp new_groupname filename
```

Change the file owner and group. You should use these two commands after you copy a file for use by somebody else.

```
su
```

(=substitute user id) Assume the superuser (=root) identity (you will be prompted for the password). Type "exit" to return you to your previous login. Don't habitually work on your machine as root. The root account is for administration and the su command is to ease your access to the administration account when you require it. You can also use "su" to assume any other user identity, e.g. `su barbara` will make me "barbara" (password required unless I am a superuser).

```
kernelcfg
```

(as root in X terminal). GUI to to add/remove kernel modules. You can do the same from the command line using the command "`insmod`", but "`insmode`" is less "newbie-friendly".

```
lsmod
```

List currently loaded kernel modules. A module is like a device driver--it provides operating system kernel support for a particular piece of hardware or feature.

```
modprobe -l |more
```

List all the modules available for your kernel. The available modules are determined by how your Linux kernel was compliled. Every possible module/feature can be compiled on linux as either "hard wired" (fast, non-removable), "module" (maybe slower, but loaded/removable on demand), or "no" (no support for this feature at all).

```
insmod parport
insmod ppa
```

(as root) Insert modules into the kernel (a module is roughly an equivalent of a DOS device driver). This example shows how to insert the modules for support of the external parallel port zip drive (it appears to be a problem to get the external zip drive to work in any other way under RH6.0).

`rmmod` *`module_name`*

(as root, not essential). Remove the module *module_name* from the kernel.

```
setserial /dev/cua0 port 0x03f8 irq 4
```

(as root) Set a serial port to a non-standard setting. The example here shows the standard setting for the first serial port (cua0 or ttyS0). The standard PC settings for the second serial port (cua1or ttyS1) are: address of i/o port 0x02f8, irq 3. The third serial port (cua2 or ttyS2): 0x03e8, irq 4. The forth serial port (cua3 or ttyS3): 0x02e8, irq 3. Add your setting to `/etc/rc.d/rc.local` if you want it to be set at the boot time. See man setserial for good a overview.

```
fdisk
```

(as root) Linux hard drive partitioning utility (DOS has a utility with the same name).

```
cd /usr/src/linux-2.0.36
make xconfig
```

(as root in X terminal). Nice GUI front-end for configuration of the kernel options in preparation for compilation of your customized kernel. (The directory name contains the version of your Linux kernel so you may need to modify the directory name if your Linux kernel version is different than 2.0.36 used in this example. You also need the "Tk" interpreter and the kernel source code installed.) The alternatives to "make xconfig" are: "make config" (runs a scripts that asks you questions in the text mode) and "make menuconfig" (runs a text-based menu-driven configuration utility).
Try: `less /usr/doc/HOWTO/Kernel-HOWTO` for more information.
After the configuration, you may choose to proceed with kernel compilation of the new kernel by issuing the following commands:

```
make dep
make zImage
```

The last command will take some time to complete (maybe 0.5 h, depending on your hardware). It produces the file "zImage", which is your new Linux kernel. Next:

```
make modules
make modules_install
```

Read: /usr/doc/HOWTO/Kernel-HOWTO for information on how to install the new kernel. You will probably also find it useful to read "man depmode". Configuration, compilation and installation of a new kernel is not difficult but it CAN lead to problems if you don't know what you are doing.
Compilation of a kernel is a good way to test your hardware, because it involves a massive amount of computing. If your hardware is "flaky", you will most likely receive the "signal 11" error (read the beatiful `/usr/doc/FAQ/txt/GCC-SIG11-FAQ`). **See this for details on kernel upgrade.**

```
depmod -a
```

(as root) Build the module dependency table for the kernel. This can, for example, be useful after installing and booting a new kernel. Use "`modprobe -a`" to load the modules.

```
ldconfig
```

(as root) Re-create the bindings and the cache for the loader of dynamic libraries ("ld"). You may want to run ldconfig after an installation of new dynamically linked libraries on your system. (It is also re-run every time you boot the computer, so if you reboot you don't have to run it manually.)

```
mknod /dev/fd0 b 2 0
```

(=make node, as root) Create a device file. This example shows how to create a device file associated with your first floppy drive and could be useful if you happened to accidentally erase it. The options are: b=block mode device (c=character mode device, p=FIFO device, u=unbuffered character mode device). The two integers specify the major and the minor device number.

```
fdformat /dev/fd0H1440
mkfs -c -t ext2
```

(=floppy disk format, two commands, as root) Perform a low-level formatting of a floppy in the first floppy drive (/dev/fd0), high density (1440 kB). Then make a Linux filesystem (-t ext2), checking/marking bad blocks (-c). Making the files system is an equivalent to the high-level format.

```
badblocks /dev/fd01440 1440
```

(as root) Check a high-density floppy for bad blocks and display the results on the screen. The parameter "1440" specifies that 1440 blocks are to be checked. This command does not modify the floppy.

```
fsck -t ext2 /dev/hda2
```

(=file system check, as root) Check and repair a filesystem. The example uses the partition hda2, filesystem type ext2.

```
dd if=/dev/fd0H1440 of=floppy_image
dd if=floppy_image of=/dev/fd0H1440
```

(two commands, dd="data duplicator") Create an image of a floppy to the file called "floppy_image" in the current directory. Then copy `floppy_image` (file) to another floppy disk. Works like DOS "DISKCOPY".

Program installation

```
rpm -ivh filename.rpm
```

(=RedhatPackageManager, install, verbose, hashes displayed to show progress, as root.) Install a content of RedHat rpm package(s) and print info on what happened. Keep reading if you prefer a GUI installation.

```
rpm -qpi filename.rpm
```

(=RedhatPackageManager, query, package, list.) Read the info on the content of a yet uninstalled package *filename.rpm*.

```
rpm -qpl filename.rpm
```

(=RedhatPackageManager, query, package, information.) List the files contained in a yet uninstalled package *filename.rpm*.

```
rpm -qf filename
```

(=RedhatPackageManager, query, file.) Find out the name of the *.rpm package to which the file *filename* (on your hardrive) belongs.

```
rpm -e packagename
```

(=RedhatPackageManager, erase=uninstall.) Uninstall a package *pagckagename*. *Packagname* is the same as the beginning of the *.rpm package file but without the dash and version number.

```
kpackage
gnorpm
glint
```

(in X terminal, as root if you want to be able to install packages) GUI fronts to the Red Hat Package Manager (rpm). "glint" comes with RH5.2, "gnorpm" with RH6.0, "kpackage" comes with RH6.1 or must be installed separately but is the best of the three. Use any of them to view which software packages are installed on your system and the what not-yet-installed packages are available on your RedHat CD, display the info about the packages, and install them if you want (installation must be done as root).

Accessing drives/partitions

```
mount
```

See here for details on mounting drives. Examples are shown in the next commands.

```
mount -t auto /dev/fd0 /mnt/floppy
```

(as root) Mount the floppy. The directory `/mnt/floppy` must exist, be empty and NOT be your current directory.

```
mount -t auto /dev/cdrom /mnt/cdrom
```

(as root) Mount the CD. You may need to create/modify the `/dev/cdrom` file depending where your CDROM is. The directory `/mnt/cdrom` must exist, be empty and NOT be your current directory.

```
mount /mnt/floppy
```

(as user or root) Mount a floppy as user. The file `/etc/fstab` must be set up to do this. The directory `/mnt/floppy` must not be your current directory.

```
mount /mnt/cdrom
```

(as user or root) Mount a CD as user. The file `/etc/fstab` must be set up to do this. The directory `/mnt/cdrom` must not be your current directory.

```
umount /mnt/floppy
```

Unmount the floppy. The directory `/mnt/floppy` must not be your (or anybody else's) current working directory. Depending on your setup, you might not be able to unmount a drive that you didn't mount.

7.6 Network administration tools

```
netconf
```

(as root) A very good menu-driven setup of your network.

```
pingmachine_name
```

Check if you can contact another machine (give the machine's name or IP), press <Ctrl>C when done (it keeps going).

```
route -n
```
Show the kernel routing table.

```
nslookup host_to_find
```
Query your default domain name server (DNS) for an Internet name (or IP number) *host_to_find*. This way you can check if your DNS works. You can also find out the name of the host of which you only know the IP number.

```
traceroute host_to_trace
```
Have a look how you messages trave to `host_to_trace` (which is either a host name or IP number).

```
ipfwadm -F -p m
```
(for RH5.2, seen next command for RH6.0) Set up the firewall IP forwarding policy to masquerading. (Not very secure but simple.) Purpose: all computers from your home network will appear to the outside world as one very busy machine and, for example, you will be allowed to browse the Internet from all computers at once.

```
echo 1 > /proc/sys/net/ipv4/ip_forward
ipfwadm-wrapper -F -p deny
ipfwadm-wrapper -F -a m -S xxx.xxx.xxx.0/24 -D 0.0.0.0/0
```
(three commands, RH6.0). Does the same as the previous command. Substitute the "x"s with digits of your class "C" IP address that you assigned to your home network. See here for more details. In RH6.1, masquarading seems broken to me--I think I will install Mandrake Linux:).

```
ifconfig
```
(as root) Display info on the network interfaces currently active (ethernet, ppp, etc). Your first ethernet should show up as eth0, second as eth1, etc, first ppp over modem as ppp0, second as ppp1, etc. The "lo" is the "loopback only" interface which should be always active. Use the options (see `ifconfig --help`) to configure the interfaces.

```
ifup interface_name
```
(`/sbin/ifup` to it run as a user) Startup a network interface. E.g.:

```
ifup eth0
ifup ppp0
```
Users can start up or shutdown the ppp interface only when the right permission was checked during the ppp setup (using `netconf`). To start a ppp interface (dial-up connection), I normally use kppp available under kde menu "internet".

```
ifdown interface_name
```

(`/sbin/ifdown` to run it as a user). Shut down the network interface. E.g.: `ifdown ppp0` Also, see the previous command.

```
netstat | more
```

Displays a lot (too much?) information on the status of your network.

Music-related commands

```
cdplay play 1
```

Play the first track from a audio CD.

```
eject
```

Get a free coffee cup holder :))). (Eject the CD ROM tray).

```
play my_file.wav
```

Play a wave file.

```
mpg123 my_file.mp3
```

Play an mp3 file.

```
mpg123 -w my_file.wav my_file.mp3
```

Create a wave audio file from an mp3 audio file.

```
knapster
```

(in X terminal) Start the program to downolad mp3 files that other users of napster have displayed for downloading. Really cool!

```
cdparanoia -B  "1-"
```

(CD ripper) Read the contents of an audio CD and save it into wavefiles in the current directories, one track per wavefile. The "1-"
means "from track 1 to the last". -B forces putting each track into a separate file.

```
playmidi my_file.mid
```

Play a midi file. `playmidi -r my_file.mid` will display text mode effects on the screen.

```
sox
```

(argument not given here) Convert from almost any audio file format to another (but not mp3s). See `man sox`.

Graphics-related commands

```
kghostview my_file.ps
```

Display a postscript file on screen. I can also use the older-looking `ghostview` or `gv` for the same end effect.

```
ps2pdf my_file.ps my_file.pdf
```

Make a pdf (Adobe portable document format) file from a postscript file.

```
gimp
```

(in X terminal) A humble looking but very powerful image processor. Takes some learning to use, but it is great for artists, there is almost nothing you can't do with gimp. Use your mouse right button to get local menus, and learn how to use layers. Save your file in the native gimp file format *.xcf (to preserve layers) and only then flatten it and save as png (or whatever). There is a large user manual /usr/

```
gphoto
```

(in X terminal) Powerful photo editor.

```
giftopnm my_file.giff > my_file.pnm
pnmtopng my_file.pnm > my_file.png
```

Convert the propriatory giff graphics into a raw, portable pnm file. Then convert the pnm into a png file, which is a newer and better standard for Internet pictures (better technically plus there is no danger of being sued by the owner of giff patents).