Useful Linux Bash Commands

This is a very short summary of bash commands. Most but not all information comes from [1]

The bash terminal window

Good to know:

- Selected with mouse = automatically copied to clipboard
- Middle mouse button = paste
- <Up> = last command (editable)
- <Ctrl>-r someletters = search in command

Stopping bash scripts

`<ctrl> C or ^C` terminates a running script

This may leave the shell in a disordered state.
If this happens:

1. `<ctrl>J or ^J`
   gives you back a shell prompt (does the same as <enter>, even if <enter> doesn't work any more
2. type "`reset`" even if you don't see what you type, followed by <ctrl>J

Miscellanous

`history`	List of all used commands

`man mycommand`
 	
`man -k mykeyword`
	Display man page of mycommand
	(Leave with q)
	
	Search for mykeyword in the man pages

`info mycommand`	Get info on mycommand (Leave with Ctrl-C)
`type mycommand` `which mycommand`	see if mycommand is a builtin command displays location of mycommand (if in search path)

Many commands have a `--help` option.

Wildcards

Can be used for all bash commands and are expanded by the shell
Can be used for file and folder names

`*`	any char, any number of chars
`?`	any char, exactly one
`[abcd]`	any character contained in list abcd
`[^abcd]`	any character not contained list abcd

{a,b,c} works like [abc], but not only for files,it works for all bash commands
echo {a,b,c}test prints atest btest ctest

Redirection and pipes

Redirection

`mycommand > myfile`
`mycommand >> myfile`
`ls > dir.txt`
	Output of mycommand goes to myfile
	Output of mycommand is appended to myfile
	Output of ls into file dir.txt

`mycommand 2> myerrorfile`	Error output goes to myerrorfile (standard output remains on screen)
`mycommand > myfile  2> myerrorfile`	Output of mycommand goes to myfile, error output goes to myerrorfile
`mycommand &> myfile`	Output and error output of mycommand goes to myfile
`mycommand <  myfile`	mycommand gets input from myfile

Pipes

`command1	command2`	Execute command1 and use its output as input for command2
`ls	grep a`	List all files whose filename contains a letter “a”
`mycommand	tee myfile`	Execute mycommand , output is displayed and goes to myfile (tee copies from stdout to stdout and to one or more files)
`ls	tee dir.txt`	List directory to stdout and to file dir.txt

Administration

`sudo  mycommand`	execute mycommand as a super user (root)
`sudo su root`	Set user = root permanently
`sudo adduser jcf dialout`	Add user jcf to group dialout Necessary for working with serial ports e.g. using terminal
`id -Gn`	Display name and user groups

Installation

`sudo apt-get install myprogram`	Install myprogram
`sudo apt-add-repository ppa:myunity/ppa`	Add repository
`sudo apt-get update`	Update repository info
`sudo apt-get upgrade`	
`sudo apt-get remove`	
`sudo apt-get source`	get source files
`sudo apt-get check`	check for broken dependencies

System Analysis

`lspci`	List PCI devices
`lsusb`	List USB devices
`lsmod`	List drivers
`tail -f /var/log/syslog`	Continuously display syslog (leave with <ctrl>c
`dmesg`	Writes kernel boot messages to the standard output. dmesg reads the kernel ring buffer.
`dmesg | grep <keyword>`	filters the dmesg messages to show only those containing <keyword>
`uname -a`	Display kernel version
`blkid`	Get drives info (seems not always to work!)

Files

`ls`	List files
`ls > dir.txt`	List files, output into a file
`ls -l`	List files , detailed

`ls -l <myfile>`
 	
`ls -ld <myfile>`
	View detailed info on myfile
	(permissions, size, date)
	View detailed info on myfile
	(permissions, size, date + owner)

`ls d*`
`ls ?bus*`
`ls [def]*`
`ls [^def]*`
	Lists all files beginning with d
	Lists all files that contain “bus” after exactly one char that may be any char.
	Lists all files beginning with d or e or f
	Lists all files not beginning with d or e or f

`chown myname myfile`
`chown myname myfolder`
`chown -R myname myfolder`
 	
`chgrp myfile`
	change owner of myfile
	change owner of myfolder
	recursively change owner of myfolder
	
	change group
	(same options as chown)

`chmod +r <file>`
`chmod +w <file>`
`chmod +x <file>`
 	
`chmod -R +r <folder>`
	Give all users right to
	read
	write
	execute (for folders: allow entering folder)
	
	The same with – instead of + : take rights away
	
	- R = recusively apply +r to all content of folder

`cp myfile myfolder` `cp -a myfiles myfolder`	Copy myfile to myfolder Copy recursively myfiles to myfolder, including subfolders
`mv myfile mynewfile` `mv myfile myfolder`	Rename myfile to mynewfile Move myfile to myfolder

`rm myfile`
`rm myfolder`
`rm -r myfolder`
	Remove (delete) myfile
	Remove myfolder
	Recursively remove myfolder, including subfolders

File permissions

Example:

```
ls -ld tmp.txt
-rw-rw-r-- 1 jcf jcf 0 Oct 15  2012 tmp.txt
```

Pos. 1	Pos. 2-4	Pos. 5-7	Pos. 8-10
`-`	`rw-`	`rw-`	`r--`

`Type`
 	
`- = file`
`d = dir`
`l = symbolic link`
`p = named pipe`
`c = char device`
`b = block device`
	`Permissions for owner`
	`*)`
		`Permissions for group`
		`*)`
			`Permissions for others`
			`*)`

*) r = read, w = write, x = execute

Folders

. = current directory
.. = parent directory

`cd`	Change dir to HOME
`pwd`	print working directory
`mkdir myfolder`	Make directory myfolder
`rmdir myfolder` `rm myfolder`	Remove directory myfolder

Display files

Edit file: gedit myfile (on Ubuntu)

`cat myfile`
 	
`cat *.dat | sort`
	display myfile
	Mainly used for piping
	
	pipe contents of all files *.dat through sort command

`less myfile`
 	
`less -N myfile`
	display myfile page by page
	(q = quit)
	display myfile page by page with line numbers

`head myfile` `head -5 myfile`	display first rows of myfile display first 5 rows of myfile

`tail myfile`
 	
`tail -f myfile`
	display last rows of myfile
	
	display last rows of myfile

`tail -f /var/log/syslog`	and leave file open, so that every update of myfile is displayed. This is very useful to watch logged data Use –retry option if myfile is not yet existing and you want to wait for it Continuously display syslog

`strings myfile`
`strings myfile | grep mytext`
`strings bash | grep error`
	extracts readable text out of binary files
	search for readable text mytext in binary file myfile
	search all strings containig “error” in bash

`xxd myfile`
`xxd -u myfile`
`xxd -u myfile > myfile.hex`
`xxd -r myfile.hex > myfile`
	hex dump of myfile
	hex dump of myfile with uppercase hex numbers
	hex dump into a file
	reverse hex dump

File creation

of course, with an editor!

otherwise:

`touch myfile`	create an empty file
`echo sometext > myfile` `echo moretext >> myfile`	write sometext into myfile append (in new line) moretext to myfile

File properties

`stat myfile` `stat myfile`	name, size, type, permissions, owner, time stamps info on file system containing myfile
`wc myfile`	word count: gives number of rows, words, bytes of a text file

Find files

find can look for files or folders with a specified name, size or time.

`find -name "myfile"` `find myfolder -name "myfile"`	search in current folder and it's subfolders for myfile search in myfolder and it's subfolders for myfile
`find -name "*,txt"`	the same using wilcards
`find -iname "myfile"`	as -name, but ignoring lower and uppercase chars
`find myfolder -size +100M`	find all files in myfolder and it's subfolders that are bigger than 100MB

`find myfolder -atime 3`
`find myfolder -mtime 3`
`find myfolder -ctime 3`
	find all files accessed in the last 3 days
	find all files modified in the last 3 days
	find all files that changed status in the last 3 days
	
	similar options : -amin, -cmin, -mmin for time in minutes
	use +3 for morre than 3, -4 for less than 4

`find -name "myfile" -ls`	gives more info on the found file with ls
`find -not -name "*.py"`	find all files that are not *.py files

(Many other options)

Find can be combined with xargs.
xargs in principle reads from the standard input and executes what it reads as a command, together with the options you give it.

`find -name "*.py" | xargs ls -l`	Finds all *.py files and pipes the resulting list through xargs that does ls -l with them, so details on the found files are displayed. (Corresponds to ls -l, but only on the found "*.py" files.)
`find -name "*.py" | xargs head`	Similar, but uses head to display the first lines of the found files as a preview
`find -name "*.py" | xargs grep mykeyword`	Uses grep to search in the found files for mykeyword (Take care: file names should not include whitespaces!)

Note: on some Linux systems (not on Ubuntu) the -print option is necessary for find to get output.

Find inside a file: grep

`grep "mykeyword" myfile` `grep "usb" /var/log/syslog`	search for mykeyword in myfile search for "usb" in log file
`grep [options] pattern [files]`	options: -n gives line number -i ignores case -w compares only whole words -x compares only whole lines -C N gives N lines before and behind compare match **-r recursive search through subfolders**
`grep -r "usb-audio" /var/log`	recursive search for "usb-audio" in all log files in and under the /var/log directory

grep can use regular expressions
egrep is the same, but uses an extended syntax for regular expressions
fgrep accepts a list of keywords

Manipulate files or output

cut	extract columns
paste	combines several files seen as columns to one output on stdout
tr	translates characters
sort	sorts alphabetically or numerically (or other)
uniq	remove double results in a sort output
awk	recognizes patterns and filters
sed	recognizes patterns and filters
m4	translates macros

Compare files

kompare nice GUI based tool for file comparison, easy to read difference and common text

bash commands: diff, comm, cmp

`md5sum myfile`	creates a checksum
`md5sum myfile > chk.txt`	creates a chk.txt file containing checksum and filename
`md5sum myfile1 myfile2 ... > chk.txt`	id, for more files
`md5sum --check chk.txt`	checks if checksum(s) and file(s) listed in chk.txt match

Disk tools

`df`
`df -h`
`df /dev/sda1`
	displays disk space and usage for all drives
	the same but easy to read
	the same for sda1 only

`mount`
`mount | grep sd`
`sudo umount /media/sda3`
`sudo mount /dev/sda3 /media/sda3`
	shows all mounted devices
	shows mounted devices, but only those beginning with sd...
	unmount /media/sda3
	mount /dev/sda3 to /media/sda3

`sync`	write all pending operations to disk Very important to do before retreiving an USB stick or a memory card!
`umount /dev/sde1` `sudo fsck /dev/sde1`	do a file system check for device sde1 Caution: device must be unmounted before check!

View and kill processes

`ps -U jcf`
`ps -C thunderbird`
`ps 2301`
`ps -efww`
`ps -efH`
	View all processes belonging to user jcf
	View process ID of program "thunderbird"
	View status and program command of process number 2301
	View all processes with their command invocation
	View all processes and their child processes

`pidof firefox`	get PID of "firefox"
`top` `top -c`	View processes sorted by their activity id, with command line
`free -m -c`	Display free RAM in Megabyte
`kill 2103`	End process number 2103

Time processes

`watch -n 2 myprogram` `watch -n 2 date`	Execute myprogram every 2 seconds Example: Display date every 2 s

`at mytime`
 	
`at now+1min`
`> ls > at.txt`
`>.... (other commands)`
	at mytime do something
	(enter commands at the prompt, end with Ctrl-D)
	
	in one minute …
	…. list files to at.txt
	
	at does not work with all commands (why?)
	This seems to have something to do with the output of the program.
	Bash commands that output into a file are OK

`crontab -e`
`crontab -l`
`crontab -r`
	Edit crontab file for jobs that will be done regularly
	View crontab file
	Remove crontab file
	The edit mode shows a self explaning template

Network and Internet

Own PC

`ip addr`
 	
`ip addr show eth0`
	displays IP and MAC address etc. for all ports
	
	the same, but only for eth0

`ifconfig`	does the same as ip addr

Network

`ping <IP Adress>` `ping 192.168.0.100`	tests if a host is accessible

Internet

`host <IP Adress>`
`host 91.189.90.41`
 	
`host <www Adress>`
`host www.ltam.lu`
	asks DNS for host name
	
	Asks DNS for IP address

`dig <IP Adress>` `dig <www Adress>`	very powerful command for DNS enquiry, with many options
`whois <www Adress>` `whois ltam.lu`	Get registering data of an Internet domain Attention: omit "www." in the address! e.g. "www.ltam.lu" → "ltam.lu"

`ping <IP Adress>`
`ping 192.168.0.100`
`ping google.com`
	tests if a host is accessible

`traceroute <IP Adress>`
`traceroute google.com`
`traceroute 173.194.35.132`
	displays the network path from the own PC to a distant host, and the time needed

Ports

`lspci -v | grep -i serial`	Display serial port info from lspci (-i = ignore case)
`dmesg | grep tty`	Look for available ports in dmesg
`sudo adduser jcf dialout`	Add user jcf to group dialout Necessary for working with serial ports e.g. using terminal
`echo x > /dev/ttyS4`	Send “x” to serial port ttyS4
`setserial -ag` `/dev/ttyS[012345]`	Display information on serial ports ttyS0...5 Take care: this info may be wrong! Especially: “UART: unknown” means no device present! Look in man setserial for more info
`sudo setserial /dev/ttyS0` `baud_base 38400`	Set baud rate of ttyS0 to 38400

Literature

[1] D.J. BarrettLinux Pocket Guide, o'Reilly