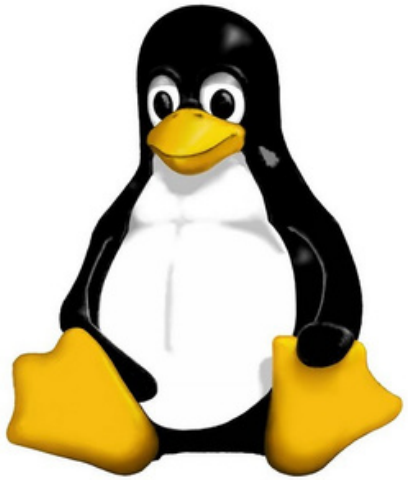
# Linux Bash Shell Cheat Sheet



(works with about every distribution, except for apt-get which is Ubuntu/Debian exclusive)

Legend:

Everything in “<>” is to be replaced, ex: <fileName> --> iLovePeanuts.txt
Don't include the '=' in your commands
'..' means that more than one file can be affected with only one command ex: rm file.txt file2.txt movie.mov .. ..

# Linux Bash Shell Cheat Sheet

## Basic Commands

### Basic Terminal Shortcuts

CTRL L = Clear the terminal CTRL D = Logout SHIFT Page Up/Down = Go up/down the terminal CTRL A = Cursor to start of line CTRL E = Cursor the end of line CTRL U = Delete left of the cursor CTRL K = Delete right of the cursor CTRL W = Delete word on the left CTRL Y = Paste (after CTRL U,K or W) TAB = auto completion of file or command CTRL R = reverse search history !! = repeat last command

CTRL Z = stops the current command (resume with fg in foreground or bg in background)

### Basic Terminal Navigation

ls -a = list all files and folders ls <folderName> = list files in folder ls -lh = Detailed list, Human readable ls -l *.jpg = list jpeg files only ls -lh <fileName> = Result for file only cd <folderName> = change directory

if folder name has spaces use “ “

cd / = go to root

cd .. = go up one folder, tip: ../../../

du -h: Disk usage of folders, human readable

du -ah: “ “ “ files & folders, Human readable

du -sh: only show disc usage of folders

pwd = print working directory

man <command> = shows manual (RTFM)

### Basic file manipulation

cat <fileName> = show content of file

(less, more)

head = from the top

-n <#oflines> <fileName>

tail = from the bottom

-n <#oflines> <fileName>

mkdir = create new folder

mkdir myStuff ..

mkdir myStuff/pictures/ ..

cp image.jpg newimage.jpg = copy and rename a file

cp image.jpg <folderName>/ = copy to folder

cp image.jpg folder/sameImageNewName.jpg

cp -R stuff otherStuff = copy and rename a folder

cp *.txt stuff/ = copy all of *<file type> to folder

mv file.txt Documents/ = move file to a folder

mv <folderName> <folderName2> = move folder in folder

mv filename.txt filename2.txt = rename file

mv <fileName> stuff/newfileName

mv <folderName>/ .. = move folder up in hierarchy

rm <fileName> .. = delete file (s)

rm -i <fileName> .. = ask for confirmation each file

rm -f <fileName> = force deletion of a file

rm -r <foldername>/ = delete folder

touch <fileName> = create or update a file

ln file1 file2 = physical link

ln -s file1 file2 = symbolic link

# Linux Bash Shell Cheat Sheet

## Basic Commands

### Researching Files

#### The slow method (sometimes very slow):

locate <text> = search the content of all the files
locate <fileName> = search for a file

sudo updatedb = update database of files

find = the best file search tool(fast)

find -name “<fileName>”
find -name “text” = search for files who start with the word text

find -name “*text” = “ “ “ “ end “ “ “ “

#### Advanced Search:

Search from file Size (in ~)

find ~ -size +10M = search files bigger than.. (M,K,G)

Search from last access
find -name “<filetype>” -atime -5
('-' = less than, '+' = more than and nothing = exactly)

Search only files or directory’s
find -type d --> ex: find /var/log -name "syslog" -type d
find -type f= files

More info: man find, man locate

### Extract, sort and filter data

grep <someText> <fileName> = search for text in file

-i = Doesn't consider uppercase words
-I = exclude binary files

grep -r <text> <folderName>/ = search for file names with occurrence of the text

#### With regular expressions:

grep -E ^<text> <fileName> = search start of lines with the word text
grep -E <0-4> <fileName> =shows lines containing numbers 0-4
grep -E <a-zA-Z> <fileName> = retrieve all lines with alphabetical letters

sort = sort the content of files

sort <fileName> = sort alphabetically
sort -o <file> <outputFile> = write result to a file
sort -r <fileName> = sort in reverse
sort -R <fileName> = sort randomly
sort -n <fileName> = sort numbers

wc = word count

wc <fileName> = nbr of line, nbr of words, byte size

-l (lines), -w (words), -c (byte size), -m (number of characters)

cut = cut a part of a file

-c --> ex: cut -c 2-5 names.txt
(cut the characters 2 to 5 of each line)
-d (delimiter) (-d & -f good for .csv files)
-f (# of field to cut)

more info: man cut, man sort, man grep

# Linux Bash Shell Cheat Sheet

## Basic Commands

### Time settings

date = view & modify time (on your computer)
View:

date “+%H” --> If it's 9 am, then it will show 09
date “+%H:%M:%Ss” = (hours, minutes, seconds)
%Y = years

Modify:

MMDDhhmmYYYY
Month | Day | Hours | Minutes | Year

sudo date 031423421997 = March 14th 1997, 23:42

### Execute programs at another time

use 'at' to execute programs in the future

Step 1, write in the terminal: at <timeOfExecution> ENTER
ex --> **at 16:45** or **at 13:43 7/23/11** (to be more precise)
or after a certain delay:

at now +5 minutes (hours, days, weeks, months, years)

Step 2: <ENTER COMMAND> ENTER
repeat step 2 as many times you need
Step 3: CTRL D to close input

atq = show a list of jobs waiting to be executed

atrm = delete a job n°<x>
ex (delete job #42) --> atrm 42

sleep = pause between commands

with ';' you can chain commands, ex: touch file; rm file
you can make a pause between commands (**m**inutes, **h**ours, **d**ays)
ex --> touch file; sleep 10; rm file <-- 10 seconds

### (continued)

crontab = execute a command regularly

-e = modify the crontab
-l = view current crontab
-r = delete you crontab

In crontab the syntax is
<Minutes> <Hours> <Day of month> <Day of week (0-6, 0 = Sunday)> <COMMAND>

ex, create the file movies.txt every day at 15:47:

47 15 * * * touch /home/bob/movies.txt
* * * * * --> every minute
at 5:30 in the morning, from the 1st to 15th each month:
30 5 1-15 * *
at midnight on Mondays, Wednesdays and Thursdays:
0 0 * * 1,3,4
every two hours:
0 */2 * * *
every 10 minutes Monday to Friday:
*/10 * * * 1-5

### Execute programs in the background

Add a '&' at the end of a command

ex --> cp bigMovieFile.mp4 &

nohup: ignores the HUP signal when closing the console
(process will still run if the terminal is closed)
ex --> nohup cp bigMovieFile.mp4
jobs = know what is running in the background
fg = put a background process to foreground

ex: fg (process 1), f%2 (process 2) f%3, ...

# Linux Bash Shell Cheat Sheet

## Basic Commands

### Process Management

w = who is logged on and what they are doing

tload = graphic representation of system load average
(quit with CTRL C)

ps = Static process list
-ef --> ex: ps -ef | less
-ejH --> show process hierarchy
-u --> process's from current user

top = Dynamic process list
While in top:

- q to close top
- h to show the help
- k to kill a process

CTRL C to top a current terminal process

kill = kill a process
You need the PID # of the process
ps -u <AccountName> | grep <Application>
Then
kill <PID> .. .. ..

kill -9 <PID> = violent kill

killall = kill multiple process's

ex --> killall locate

extras:
sudo halt <-- to close computer
sudo reboot <-- to reboot

### Create and modify user accounts

sudo adduser bob = root creates new user
sudo passwd <AccountName> = change a user's password
sudo deluser <AccountName> = delete an account
addgroup friends = create a new user group
delgroup friends = delete a user group
usermod -g friends <Account> = add user to a group
usermod -g bob boby = change account name
usermod -aG friends bob = add groups to a user without loosing the ones he's already in

### File Permissions

chown = change the owner of a file

ex --> chown bob hello.txt
chown user:bob report.txt = changes the user owning report.txt to 'user' and the group owning it to 'bob'
-R = recursively affect all the sub folders
ex --> chown -R bob:bob /home/Daniel
chmod = modify user access/permission – simple way

u = user g = group o = other d = directory (if element is a directory) l = link (if element is a file link) r = read (read permissions) w = write (write permissions) x = eXecute (only useful for scripts and programs)

# Linux Bash Shell Cheat Sheet

Basic Commands

## File Permissions (continued)

'+' means add a right '-' means delete a right '=' means affect a right ex --> chmod g+w someFile.txt

(add to current group the right to modify someFile.txt)

more info: man chmod

## Flow redirection

Redirect results of commands:

'>' at the end of a command to redirect the result to a file

ex --> ps -ejH > process.txt

'>>' to redirect the result to the end of a file

### Redirect errors:

'2>' at the end of the command to redirect the result to a file
ex --> cut -d , -f 1 file.csv > file 2> errors.log

'2>&1' to redirect the errors the same way as the standard output
Read progressively from the keyboard
<Command> << <wordToTerminateInput>

```
ex --> sort << END <-- This can be anything you want
       > Hello >
       Alex    >
       Cinema  >
       Game    >
       Code    >
       Ubuntu  >
       END
```

## Flow Redirection (continued)

terminal output: Alex Cinema Code Game Ubuntu

Another example --> wc -m << END

## Chain commands

'|' at the end of a command to enter another one

ex --> du | sort -nr | less

## Archive and compress data

Archive and compress data the long way:

Step 1, put all the files you want to compress in the same folder: ex --> mv *.txt folder/

Step 2, Create the tar file:

tar -cvf my_archive.tar folder/

-c : creates a .tar archive
-v : tells you what is happening (verbose)
-f : assembles the archive into one file

Step 3.1, create gzip file (most current):

gzip my_archive.tar
to decompress: gunzip my_archive.tar.gz

Step 3.2, or create a bzip2 file (more powerful but slow):

bzip2 my_archive.tar
to decompress: bunzip2 my_archive.tar.bz2

# Linux Bash Shell Cheat Sheet

Basic Commands

## Archive and compress data (continued)

step 4, to decompress the .tar file:

tar -xvf archive.tar archive.tar

Archive and compress data the fast way:
gzip: tar -zcvf my_archive.tar.gz folder/

decompress: tar -zcvf my_archive.tar.gz Documents/

bzip2: tar -jcvf my_archive.tar.gz folder/
decompress: tar -jxvf archive.tar.bz2 Documents/

Show the content of .tar, .gz or .bz2without decompressing it:
gzip: bzip2: tar:

gzip -ztf archive.tar.gz
bzip2 -jtf archive.tar.bz2
tar -tf archive.tar

tar extra:
tar -rvf archive.tar file.txt = add a file to the .tar

You can also directly compress a single file and view the file without decompressing:
Step 1, use gzip or bzip2 to compress the file:

gzip numbers.txt

Step 2, view the file without decompressing it:
zcat = view the entire file in the console (same as cat)
zmore = view one screen at a time the content of the file (same as more)
zless = view one line of the file at a time (same as less)

## Installing software

When software is available in the repositories:
sudo apt-get install <nameOfSoftware>

ex--> sudo apt-get install aptitude

If you download it from the Internets in .gz format (or bz2) - “Compiling from source”
Step 1, create a folder to place the file:
mkdir /home/username/src <-- then cd to it

Step 2, with 'ls' verify that the file is there (if not, mv ../file.tar.gz /home/username/src/) Step 3, decompress the file (if .zip: unzip <file>) <-- Step 4, use 'ls', you should see a new directory Step 5, cd to the new directory Step 6.1, use ls to verify you have an INSTALL file, then: more INSTALL If you don't have an INSTALL file: Step 6.2, execute **./configure** <-- creates a makefile Step 6.2.1, run make <-- builds application binaries Step 6.2.2 : switch to root --> su Step 6.2.3 : make install <-- installs the software Step 7, read the readme file