My Linux Commands Cheatsheets

Linux OS	
startx	Start GUI
sudo systemctl start|stop|restart|enable|disable <*service*>	Starting, stopping, restarting, enable or disable a service on a currently running system
sudo systemctl start|stop gdm (or sudo telinit 3)	Start or stop Graphical Display Manager (On Ubuntu versions before 18.04 LTS, substitute lightdm for gdm)
Alt+F2	Linux command terminal mit & ohne GUI
SUPER-L	Screen Lock
CTRL-F	Nautilus Suchleiste
ctrl + l im filemanager	Aktueller Pfad
CTR-DEL	Datei Löschen
Shift-DEL	Permanent Löschen
sudo find -type f -name '*suchbegriff*' -delete	Alles löschen das den Suchbegriff beinhaltet
var/log	Zeigt letzte modifizierte Datei
which	search programs
whereis	search programs in extended system range and locating source and man files
xdpyinfo | grep dim	Screen Solution
cat	used to type out a file (or combine files).
head	used to show the first few lines of a file.
tail	used to show the last few lines of a file.
pwd	Displays the present working directory
cd ~ or cd	Change to your home directory (shortcut name is ~ (tilde))
cd ..	Change to parent directory (..)
cd -	Change to previous directory (- (minus))
cd /	Change to the root directory
ls	List the contents of the present working directory
ls –a	List all files, including hidden files and directories (those whose name startwith.)
Ls -lR	Prints out dirs with files
ls -li	print out inode number which is a unique quantity for each file

tree	Displays a tree view of the filesystem
tree -d	view just the directories and to suppress listing file names.
ln	Create hard link
ln -s	Create soft link
pushd	change dir and put it on a system remenbering list
dirs	show remembering list
popd	jump backwards trough that list
cat	Used for viewing files that are not very long; it does not provide any scroll-back.
cat -n	shows line numbers
tac	Used to look at a file backwards, starting with the last line.
less	Used to view larger files because it is a paging program.
tail	Used to print the last 10 lines of a file by default. Change the number of lines by doing -*
-n 15 or just -15	if you wanted to look at the last 15 lines instead of the default.
head	The opposite of tail; by default, it prints the first 10 lines of a file.
wc	word count
touch <filename> -t 12092000:	Create a file with timestamp 12.09.2000
mkdir /usr/sampdir	Creates a sample directory called sampdir under /usr.
rmdir	Remove empty dir only on empty directories
rm	Remove a file
rm -f	Force remove
rm -i	Prompt before removing
rm -rf	Remove a directory and all of its contents
mv	Rename file
locate bin | grep zip	Findet alle einträge in denen bin und zip vorkommt
locate / find	Suchfunktion. Locate muss 1x pro tag mir updatedb aktualisiert werden, find nicht. Mit -ls sieht man zus. informationen.
du <filename>	Zeigt den Disk Usage eine Datei an
find /usr -name gcc	Searching for files and directories named gcc
find /usr -type d -name	Searching only for directories named gcc

gcc	
find /usr -type f -name gcc	Searching only for regular files named gcc
find -name "*.swp" -exec rm {} ’;’	Remove all files that end with **.**swp. The {} (squiggly brackets) is a placeholder that will be filled with all the file names that result from the find expression, and the preceding command will be run on each one individually.
find / -ctime 3	C: indode, a: accessed/last read, m: last modified; time_ days, min: Minuten usw.. value: - is leas tham, + more than. So find / -mmin -10 bedeutet finde file die vor weniger als 10 Minuten modified worden ist.
find / size 0	Note the size here is in 512-byte blocks, by default; you can also specify bytes (c), kilobytes (k), megabytes (M), gigabytes (G), etc. As with the time numbers above, file sizes can also be exact numbers (n), +n or -n. For details, consult the man page for find. For example, to find files greater than 10 MB in size and running a command on those files: $ find / -size +10M -exec command {} ’;’
dpkg --list	Get all packages on OS
man	used to view documentation.
man -f or whatis <socket>	Get a list of all man Pages with <socket>
man -k <socket	Get all man pages with <socket>
kill -SIGKILL <pid> or kill -9 <pid>	eliminate process. pid is the Process ID
w, top or uptime	load average. uptime gets more informations lie PID
<smth> &	execute task in the background
CTRL-Z	suspend foreground job
CTRL-C	terminate foreground job
bg or fg	run process in back/foreground
jobs -l	background processes with PID
ps	display all processes under current shell
ps -u or -ef or -eLF	info for specified username. The command **ps -ef** displays all the processes in the system in full detail. The command **ps -eLf** goes one step further and displays one line of information for every thread (remember, a process can contain multiple threads).
pstree	displays the processes running on the system in the form of a tree diagram showing the relationship between a process and its parent

	process and any other processes that it created.
top	highlights which processes are consuming the most CPU cycles and memory
mount	show mounted filesystems
df -Th	display information about mounted filesystems, including the filesystem type, and usage statistics about currently used and available space.
`diff [options] <filename1> <filename2>`	**`diff`** is used to compare files and directories. -c Provides a listing of differences that include three lines of context before and after the lines differing in content. -r Used to recursively compare subdirectories, as well as the current directory. -i Ignore the case of letters. -w Ignore differences in spaces and tabs (white space). -q Be quiet: only report if files are different without listing the differences. You can compare three files at once using **`diff3`**, which uses one file as the reference basis for the other two.
patch -p1 < patchfile or patch originalfile patchfile	Patch a file or dir. The first usage is more common, as it is often used to apply changes to an entire directory tree, rather than just one file, as in the second example. To understand the use of the -p1 option and many others, see the man page for patch.
file *	Examines the contents and certain characteristics to determine whether the files are plain text, shared libraries, executable programs, scripts, or something else of all files on the OS.
cp or rsync	cp can only copy files to and from destinations on the local machine (unless you are copying to or from a filesystem mounted using NFS), but rsync can also be used to copy files from one machine to another.
gzip *	Compresses all files in the current directory; each file is compressed and renamed with a .gz extension.
echo <text> > or >> <filename>	with > it crates file and inputs the text, with >> it appends textto the file
cat << EOF > <filename> . . EOF	Creates Textfile and appends so many lines text to it if you want
whoami / who	identify current user / list logged-on users
alias del=‘rm -r’	creates the alias del
set, env or export	Show enviromnet variables
echo $SHELL	Show the value of a specific variable

Edit ~/.bashrc and add the line export VARIABLE=value Type source ~/.bashrc or just . ~/.bashrc (dot ~/.bashrc); or just start a new shell by typing bash	Add a variable permanently
export VARIABLE=value (or VARIABLE=value; export VARIABLE)	Export a new variable value
history / CTRL-R	shows commands history / enable intelligent search
chown	Used to change user ownership of a file or directory
chown syntax	chown [optionen] [besitzer][:[gruppe]] datei
-c	Wie „-v“ aber nur, wenn auch etwas geändert wird.
-f	Unterdrückt Fehlermeldungen (–silent, –quiet)
-R	Steht für „rekursiv“. Mit diesem Schalter werden alle Ordner und Unterordner inkl. Dateien geändert. Also ganze Verzeichnisbäume.
-v	Steht für “verbose” und Zeigt alles an was der Befehl mv gerade macht.
chown root meinedatei.txt chown root:www-data meinedatei.txt chown -R www-data:www-data /var/www	Ändert den Besitzer der Datei „meinedatei.txt“ auf den User „root“ Ändert den Besitzer der Datei „meinedatei.txt“ auf den User „root“ und die Gruppe der Datei auf „www-data“ Ändert den Ordner, Inhalt und alle Unterordner von „/var/www“ auf den Besitzer und Gruppe“www-data“
chmod	**`chmod`** Used to change the permissions on the file, which can be done separately for **owner**, **group** and the rest of the **world** (often named as **other**)
chgrp	Used to change group ownership
strings book1.xls | grep my_string	search for the string my_string in book1.xls
clear	clear history
ls <filename> and than	View exit status value (0 is suceed, 2 mostly unsuceed)

echo $?	

cp (copy)	
`-a` oder `--archive`	Beibehaltung von Besitzer-, Gruppen- und Zugriffsrechten und Erstellungs-, Modifikations- und Zugriffsdaten (entspricht `-dR --preserve=all`)
`-b` oder `--backup`	Sichert Dateien vor dem Überschreiben, wenn diese unterschiedlich sind
`-d`	erhält symbolische Links, folgt ihnen aber nicht beim Kopieren (entspricht `-P --preserve=links`)
`-i` oder `--interactive`	fragt vor Überschreiben nach
`-l` oder `--link`	kopiert nicht, sondern erstellt harten Link
`-n` oder `--no-clobber`	niemals vorhandene Dateien überschreiben (`-i` wird wirkungslos)
`-p` (klein) oder `--preserve=mode,ownership,timestamps`	erhält Standard-Dateiattribute wie Zeitpunkt des letzten Schreibzugriffs, Besitzer-, Gruppen- und Zugriffsrechten
`-P` (groß) oder `--no-dereference`	Symbolische Links als symbolische Links kopieren, statt den Links in der Quelle zu folgen
`-r` oder `-R` oder `--recursive`	Verzeichnisse rekursiv kopieren (Unterverzeichnisse eingeschlossen)
`-s` oder `--symbolic-link`	kopiert nicht, sondern erstellt symbolischen Link
`-u` oder `--update`	kopiert nur, wenn Zieldatei älter als Quelldatei

`-v` oder `--verbose`	Durchgeführte Tätigkeiten "erklären" / anzeigen
cp kennt noch eine Reihe von weiteren Optionen, die man mit Hilfe der Option --help aufrufen kann. Oder man schaut in die Manpage von cp.	
cp -u datei.txt /home/BENUTZER/Dokumente/datei.txt	Die Datei **datei.txt** aus dem aktuellen Verzeichnis nach **/home/BENUTZER/Dokumente** kopieren, falls diese neuer ist als die bestehende:
cp -uv datei.txt dokument.odt /home/BENUTZER/Dokumente	Die Dateien **datei.txt**, **dokument.odt** nach **/home/BENUTZER/Dokumente** kopieren, wenn neuer als bestehende, und Kopierfortschritt anzeigen:
cp -uv da*.txt do*.odt /home/BENUTZER/Dokumente	Dateien mit bestimmtem Anfangsbuchstaben **da*.txt**, **do*.odt** kopieren, wenn neuer als bestehende, und Kopierfortschritt anzeigen:
cp -a /home/BENUTZER/. /media/sda9/Sicherungen	Alle Dateien aus **/home/BENUTZER** samt der versteckten Einstellungsdateien rekursiv ins Verzeichnis **Sicherungen** kopiert, welches sich auf dem eingehängten Laufwerk **sda9** befindet.
shopt -s extglob	Leider enthält **cp** keine `--exclude`-Option wie andere Kopierprogramme, jedoch gibt bereits die Shell einen Weg her, der es ermöglicht ressourcensparend Muster auszuschließen. Dazu muss z.B. in bash die Shell-Option extglob gesetzt werden, die den erweiterten Musterabgleich aktiviert (extended globbing):
cp -av /QUELLE/!(Datei1.txt|Festplattenabbild*|Ordner/Unterordner1) /ZIEL/	Der Befehl muss jedes mal ausgeführt werden oder man trägt ihn in die bashrc ein. Links Folgenden der angewandte cp-Befehl. Durch das Muster `!(...)` werden alle Dateien und Ordner, die mit *"Datei1.txt*", *"Festplattenabbild*" und *"Unterorder1*" beginnen, ausgeschlossen. Darüber hinaus werden alle Dateien und Ordner von QUELLE nach ZIEL kopiert.

Linux and Bash Command Cheat Sheet: The Basics

Getting information

```
# return your user name
whoami
```

```
# return your user and group id
id
```

```
# return operating system name, username, and other info
uname -a
```

```
# display reference manual for a command
man top
```

```
# get help on a command
curl --help
```

```
# return the current date and time
date
```

Monitoring performance and status

```
# list selection of or all running processes and their PIDs
ps
ps -e
```

```
# display resource usage
top
```

```
# list mounted file systems and usage
df
```

Working with files

```
# copy a file
cp file.txt new_path/new_name.txt
```

```
# change file name or path
mv this_file.txt that_path/that_file.txt
```

```
# remove a file verbosely
rm this_old_file.txt -v
```

```
# create an empty file, or update existing file's timestamp
touch a_new_file.txt
```

```
# change/modify file permissions to 'execute' for all users
chmod +x my_script.sh
```

```
# get count of lines, words, or characters in file
wc -l table_of_data.csv
wc -w my_essay.txt
wc -m some_document.txt
```

```
# return lines matching a pattern from files matching a filename pattern - case insensitive and whole words only
grep -iw hello \*.txt
```

```
# return file names with lines matching the pattern 'hello' from files matching a filename pattern
grep -l hello \*.txt
```

Navigating and working with directories

```
# list files and directories by date, newest last
ls -lrt
```

```
# find files in directory tree with suffix 'sh'
find -name '\*.sh'
```

```
# return present working directory
pwd
```

```
# make a new directory
mkdir new_folder
```

```
# change the current directory: up one level, home, or some other path
cd ../
cd ~ or cd
cd another_directory
```

```
# remove directory, verbosely
rmdir temp_directory -v
```

Printing file and string contents

```
# print file contents
cat my_shell_script.sh
```

```
# print file contents page-by-page
more ReadMe.txt
```

```
# print first N lines of file
head -10 data_table.csv
```

```
# print last N lines of file
tail -10 data_table.csv
```

```
# print string or variable value
echo "I am not a robot"
echo "I am $USERNAME"
```

Compression and archiving

```
# archive a set of files
tar -cvf my_archive.tar.gz file1 file2 file3
```

```
# compress a set of files
zip my_zipped_files.zip file1 file2
zip my_zipped_folders.zip directory1 directory2
```

```
# extract files from a compressed zip archive
unzip my_zipped_file.zip
unzip my_zipped_file.zip -d extract_to_this_direcory
```

Performing network operations

```
# print hostname
hostname
```

```
# send packets to URL and print response
ping www.google.com
```

```
# display or configure system network interfaces
ifconfig
ip
```

```
# display contents of file at a URL
curl <url>
```

```
# download file from a URL
wget <url>
```

Bash shebang

```
#!/bin/bash
```

Pipes and Filters

```
# chain filter commands using the pipe operator
ls | sort -r
```

```
# pipe the output of manual page for ls to head to display the first 20 lines
man ls | head -20
```

Shell and Environment Variables

```
# list all shell variables
set
```

```
# define a shell variable called my_planet and assign value Earth to it
my_planet=Earth
```

```
# display shell variable
echo $my_planet
```

```
# list all environment variables
env
```

```
# environment vars: define/extend variable scope to child processes
export my_planet
export my_galaxy='Milky Way'
```

Metacharacters

```
# comments
# The shell will not respond to this message
```

```
# command separator
echo 'here are some files and folders'; ls
```

```
# file name expansion wildcard
ls *.json
```

```
# single character wildcard
ls file_2021-06-??.json
```

Quoting

```
# single quotes - interpret literally
echo 'My home directory can be accessed by entering: echo $HOME'
```

```
# double quotes - interpret literally, but evaluate metacharacters
echo "My home directory is $HOME"
```

```
# backslash - escape metacharacter interpretation
echo "This dollar sign should render: \$"
```

I/O Redirection

```
# redirect output to file
echo 'Write this text to file x' > x

# append output to file
echo 'Add this line to file x' >> x

# redirect standard error to file
bad_command_1 2> error.log

# append standard error to file
bad_command_2 2>> error.log

# redirect file contents to standard input
$ tr “[a-z]” “[A-Z]” < a_text_file.txt

# the input redirection above is equivalent to
$cat a_text_file.txt | tr “[a-z]” “[A-Z]”
```

Command Substitution

```
# capture output of a command and echo its value
THE_PRESENT=$(date)
echo "There is no time like $THE_PRESENT"
```

Command line arguments

```
./My_Bash_Script.sh arg1 arg2 arg3
```

Batch vs. concurrent modes

```
# run commands sequentially
start=$(date); ./MyBigScript.sh ; end=$(date)

# run commands in parallel
./ETL_chunk_one_on_these_nodes.sh & ./ETL_chunk_two_on_those_nodes.sh
```

Scheduling jobs with Cron

```
# open crontab editor
crontab -e

# job scheduling syntax
m h dom mon dow command
```

minute, hour, day of month, month, day of week

`*` means any

```
# append the date/time to file every Sunday at 6:15 pm
15 18 * * 0 date >> sundays.txt
```

```
# run a shell script on the first minute of the first day of each month
1 0 1 * * ./My_Shell_Script.sh
```

```
# back up your home directory every Monday at 3 am
0 3 * * 1 tar -cvf my_backup_path\my_archive.tar.gz $HOME\
```

```
# deploy your cron job
Close the crontab editor and save the file
```

```
# list all cron jobs
crontab -l
```

Command Line Shortcuts

Keyboard Shortcut	Task
CTRL-L	Clears the screen
CTRL-D	Exits the current shell
CTRL-Z	Puts the current process into suspended background
CTRL-C	Kills the current process
CTRL-H	Works the same as backspace
CTRL-A	Goes to the beginning of the line
CTRL-W	Deletes the word before the cursor
CTRL-U	Deletes from beginning of line to cursor position
CTRL-E	Goes to the end of the line
Tab	Auto-completes files, directories, and binaries

Package Manager

Operation	rpm	deb
Install package	`rpm -i foo.rpm`	`dpkg --install foo.deb`
Install package, dependencies	`dnf install foo`	`apt-get install foo`
Remove package	`rpm -e foo.rpm`	`dpkg --remove foo.deb`
Remove package, dependencies	`dnf remove foo`	`apt-get autoremove foo`
Update package	`rpm -U foo.rpm`	`dpkg --install foo.deb`
Update package, dependencies	`dnf update foo`	`apt-get install foo`
Update entire system	`dnf update`	`apt-get dist-upgrade`
Show all installed packages	`rpm -qa` or `dnf list installed`	`dpkg --list`
Get information on package	`rpm -qil foo`	`dpkg --listfiles foo`
Show packages named `foo`	`dnf list "foo"`	`apt-cache search foo`
Show all available packages	`dnf list`	`apt-cache dumpavail foo`
What package is `file` part of?	`rpm -qf file`	`dpkg --search file`

OS Shutdown

Syntax	*shutdown [OPTIONS] [TIME] [MESSAGE]*
sudo shutdown	shutdown in 1 minute
sudo shutdown now	shutdown immediately
sudo shutdown 10:00	shutdown at 10 AM
sudo shutdown +30	shutdown in 30 Minutes
sudo shutdown -h	halt system

sudo shutdown -r now	restart immediately
sudo shutdown -r 10:00	restart at 10 AM
sudo shutdown -c	cancel shutdown
date -d @`cat /run/systemd/shutdown/scheduled | head -n 1 | cut -c6-15`	check scheduled shutdown

Text Manipulation	
cat file1 file2	Concatenate multiple files and display the output; i.e. the entire content of the first file is followed by that of the second file
cat file1 file2 > **newfile**	Combine multiple files and save the output into a new file
cat file >> **existingfile**	Append a file to the end of an existing file
cat > **file**	Any subsequent lines typed will go into the file, until **CTRL-D** is typed
cat >> **file**	Any subsequent lines are appended to the file, until **CTRL-D** is typed
echo string > newfile	**echo string > newfile**
echo string >> existingfile	**echo string >> existingfile**
echo $variable	**echo $variable**
less <file> or cat <file> | less	View the contents of large files, scrolling up and down page by page, without the system having to place the entire file in memory before starting.
head or head -n */path/…*	**head** reads the first few lines of each named file (10 by default) and displays it on standard output. You can give a different number of lines in option -n.

`tail -f somefile.log`	tail work same as head. The left command will continuously display any new lines of output in somefile.log as soon as they appear. Thus, it enables you to monitor any current activity that is being reported and recorded.
`grep [pattern] <filename>`	Search for a pattern in a file and print all matching lines
`grep -v [pattern] <filename>`	Print all lines that do **not** match the pattern
`grep [0-9] <filename>`	Print the lines that contain the numbers **0** through **9**
`grep -C 3 [pattern] <filename>`	Print context of lines (specified number of lines above and below the pattern) for matching the pattern. Here, the number of lines is specified as 3

My Linux Dictionary	
udev	User-Device
bash	GNU-Bourne Again Shell
usr	User
SUPER-Key	Keyboard Windows-Key
Nautilus	Default File manager
gedit	GNOME Default text editor
dpkg	Package Manager
apt	Advanced-package-tool (Debian based OS)
rpm	Red-hat-package-manager (Red Hat based OS)
yast	Yet-another-setup-tool pckge manager (Open Suse based OS)

- **`cat`**, short for concatenate, is used to read, print, and combine files.
- **`echo`** displays a line of text either on standard output or to place in a file.
- **`sed`** is a popular stream editor often used to filter and perform substitutions on files and text data streams.
- **`awk`** is an interpreted programming language, typically used as a data extraction and reporting tool.
- **`sort`** is used to sort text files and output streams in either ascending or descending order.
- **`uniq`** eliminates duplicate entries in a text file.
- **`paste`** combines fields from different files. It can also extract and combine lines from multiple sources.
- **`join`** combines lines from two files based on a common field. It works only if files share a common field.
- **`split`** breaks up a large file into equal-sized segments.

- Regular expressions are text strings used for pattern matching. The pattern can be used to search for a specific location, such as the start or end of a line or a word.
- **`grep`** searches text files and data streams for patterns and can be used with regular expressions.
- **`tr`** translates characters, copies standard input to standard output, and handles special characters.
- **`tee`** saves a copy of standard output to a file while still displaying at the terminal.
- **`wc`** (word count) displays the number of lines, words, and characters in a file or group of files.
- **`cut`** extracts columns from a file.
- **`less`** views files a page at a time and allows scrolling in both directions.
- **`head`** displays the first few lines of a file or data stream on standard output. By default, it displays 10 lines.
- **`tail`** displays the last few lines of a file or data stream on standard output. By default, it displays 10 lines.
- **`strings`** extracts printable character strings from binary files.
- The **`z`** command family is used to read and work with compressed files.

Linux-Filesystem

/bin	Ccontains executable binaries, essential commands used to boot the system or in single-user mode, and essential commands required by all system users, such as **cat**, **cp**, **ls**, **mv**, **ps**, and **rm**.
/usr/bin	Commands that are not essential (theoretically) for the system to boot or operate in single-user mode are placed in the /usr/bin and /usr/sbin directories.
/dev	The /dev directory contains device nodes, a type of pseudo-file used by most hardware and software devices, except for network devices.

/proc	Certain filesystems, like the one mounted at /proc, are called pseudo-filesystems because they have no permanent presence anywhere on the disk. The /proc filesystem contains virtual files (files that exist only in memory) that permit viewing constantly changing kernel data. /
/var	The /var directory contains files that are expected to change in size and content as the system is running. System log files: /var/log Packages and database files: /var/lib Print queues: /var/spool Temporary files: /var/tmp. Network services directories such as /var/ftp (the FTP service) and /var/www (the HTTP web service) are also found under /var.
/etc	The /etc directory is the home for system configuration files. It contains no binary programs, although there are some executable scripts. For example, /etc/resolv.conf tells the system where to go on the network to obtain host name to IP address mappings (DNS). Files like passwd, shadow and group for managing user accounts are found in the /etc directory.
/boot	The /boot directory contains the few essential files needed to boot the system. For every alternative kernel installed on the system there are four files: vmlinuz The compressed Linux kernel, required for booting. initramfs The initial ram filesystem, required for booting, sometimes called initrd, not initramfs. config The kernel configuration file, only used for debugging and bookkeeping. System.map Kernel symbol table, only used for debugging.
/lib and /lib64	/lib contains libraries (common code shared by applications and needed for them to run) for the essential programs in /bin and /sbin. These library filenames either start with ld or lib.
/media, /run and /mnt	One often uses removable media, such as USB drives, CDs and DVDs. While historically this was done under the /media directory, modern Linux distributions place these mount points under the /run directory.
/opt	Optional application software packages
/sys	Virtual pseudo-filesystem giving information about the system and the hardware Can be used to alter system parameters and for debugging purposes
/srv	Site-specific data served up by the system Seldom used
/tmp	Temporary files; on some distributions erased across a reboot and/or may actually be a ramdisk in memory

/usr	Multi-user applications, utilities and data