Cheat sheet Debian / Ubuntu

la référence ligne de commande que je garde ouverte

https://dev2go.vercel.app/blog/cheatsheet-debian-ubuntu
Franck Vienot

Les commandes préfixées par sudo modifient le système. Lire avant d'exécuter, tester avec -n / --dry-run quand l'option existe.

Les bases

- `pwd` — où suis-je ? affiche le répertoire courant
- `ls` / `ls -l` / `ls -a` — lister / avec détails / avec les fichiers cachés
- `cd /var/log` ; `cd ..` ; `cd ~` ; `cd -` — aller / remonter / home / répertoire précédent
- `mkdir dossier` ; `rmdir dossier` — créer / supprimer un répertoire vide
- `touch fichier` — créer un fichier vide (ou mettre à jour sa date)
- `cp source destination` — copier (`-r` pour un répertoire)
- `mv source destination` — déplacer ou renommer
- `rm fichier` ; `rm -r dossier` — supprimer (pas de corbeille : c'est définitif)
- `cat fichier` — afficher tout le contenu
- `less fichier` — lire page par page (flèches, `/recherche`, `q` pour quitter)
- `nano fichier` — éditeur simple (Ctrl+O enregistrer, Ctrl+X quitter)
- `vim fichier` — `i` pour écrire, Échap puis `:wq` pour enregistrer et quitter
- `echo 'texte'` ; `echo $HOME` — afficher du texte ou une variable
- `grep motif fichier` — chercher une chaîne dans un fichier
- `find . -name 'nom'` — chercher un fichier par son nom
- `clear` — nettoyer l'écran (ou Ctrl+L)
- `exit` / `logout` — fermer la session ou le shell
- `sudo commande` — exécuter en administrateur
- `whoami` ; `hostname` ; `date` — qui suis-je / nom de la machine / date et heure
- `df -h` ; `free -h` — espace disque / mémoire disponibles
- `chmod +x script.sh` ; `./script.sh` — rendre exécutable puis lancer
- Tab / Tab Tab — complétion automatique / lister les possibilités
- `* ? [abc] {a,b}` — jokers : tout / un caractère / un parmi / alternatives
- `. .. ~ /` — ici / parent / home / racine du système
- `apt install` / `apt remove` — installer / désinstaller un logiciel
- `reboot` ; `poweroff` — redémarrer / éteindre (avec `sudo`)

Aide & documentation

- `man 5 crontab` — manuel, section 5 = formats de fichiers ; 1 = commandes, 8 = admin
- `apropos reseau` / `man -k` — chercher une commande par mot-clé
- `cmd --help | less` — aide courte intégrée
- `type -a cmd` / `which cmd` — alias, fonction, builtin ou binaire ?
- `tldr tar` ; `tldr --update` — exemples concrets au lieu du man (`apt install tldr`)
- `cargo install tealdeer` — même chose en Rust, démarrage instantané
- `curl cheat.sh/tar` — aide par exemples, sans rien installer
- explainshell.com — décortique une ligne de commande option par option
- `history | grep ssh` — retrouver une commande passée
- `!!` / `!$` / `!123` — dernière commande / son dernier argument / commande n°123
- `sudo !!` — rejouer la dernière commande en root

Raccourcis clavier (readline, mode emacs)

- Ctrl+A / Ctrl+E — début / fin de ligne
- Ctrl+B / Ctrl+F — recule / avance d'un caractère
- Alt+B / Alt+F — recule / avance d'un mot
- Ctrl+W — supprime le mot précédent (jusqu'à l'espace)
- Alt+Retour arrière — supprime le mot précédent (s'arrête aux `/`, `-`, `.`)
- Alt+D — supprime le mot suivant
- Ctrl+U / Ctrl+K — coupe avant / après le curseur
- Ctrl+Y — recolle ce qui a été coupé ; Alt+Y parcourt les coupes
- Ctrl+_ (ou Ctrl+X Ctrl+U) — annule la dernière modification
- Ctrl+T / Alt+T — intervertit deux caractères / deux mots
- Alt+U / Alt+L / Alt+C — mot en majuscules / minuscules / capitalisé
- Ctrl+R — recherche arrière dans l'historique (répéter pour remonter)
- Ctrl+G / Ctrl+J — abandonne / valide la recherche en cours
- Ctrl+S — recherche avant (nécessite `stty -ixon` dans `~/.bashrc`)
- Ctrl+P / Ctrl+N — commande précédente / suivante
- Alt+. (ou Alt+_) — insère le dernier argument de la commande précédente
- Ctrl+X Ctrl+E — ouvre la ligne courante dans `$EDITOR`
- Ctrl+L — efface l'écran sans perdre la ligne en cours
- Ctrl+C / Ctrl+D / Ctrl+Z — interrompt / EOF / suspend (`fg` pour reprendre)
- Ctrl+S / Ctrl+Q — gèle / dégèle l'affichage (terminal qui semble figé)
- `set -o vi` ; `set -o emacs` — changer de mode d'édition (emacs par défaut)
- zsh : `bindkey -e` ; `bindkey -v` ; `bindkey -L` — même chose sous zsh (emacs par défaut) / lister
- `bind -P` ; `~/.inputrc` — lister / personnaliser les raccourcis readline
- Valable aussi dans `psql`, `python`, `irb` — tout ce qui utilise readline

Navigation & inspection de fichiers

- `ls -lAh --group-directories-first` — listing lisible, cachés inclus
- `ls -lt` / `ls -lS` — trier par date / par taille
- `cd -` — revenir au répertoire précédent
- `pushd /var/log` ; `popd` — pile de répertoires
- `tree -L 2 -a` — arborescence limitée à 2 niveaux
- `stat fichier` — inode, permissions, dates atime/mtime/ctime
- `file fichier` — type réel (ne pas se fier à l'extension)
- `readlink -f lien` — résoudre le chemin absolu réel
- `find . -name '*.log' -mtime +7 -delete` — supprimer les `.log` de plus de 7 jours
- `find /var -type f -size +100M` — fichiers de plus de 100 Mo
- `find . -type f -exec chmod 644 {} +` — appliquer une commande à chaque résultat
- `find . -newer ref.txt` — fichiers modifiés après `ref.txt`
- `locate nginx.conf` ; `sudo updatedb` — recherche instantanée via index (plocate)
- `du -sh * | sort -h` — poids de chaque élément, trié
- `ncdu` — explorateur d'espace disque interactif

Manipuler fichiers & répertoires

- `cp -a src/ dst/` — copie en préservant droits, liens, dates
- `cp -r --reflink=auto` — copie légère sur Btrfs/XFS si supporté
- `mv -n a b` — ne pas écraser la cible
- `rm -rf --one-file-system dir` — suppression sûre, ne traverse pas les montages
- `mkdir -p a/b/c` — crée toute l'arborescence
- `ln -s /opt/app/bin/app /usr/local/bin/app` — lien symbolique
- `touch -d '2024-01-01' f` — forcer une date de modification
- `shred -u fichier` — écrasement avant suppression (HDD)
- `rsync -avh --progress src/ dst/` — copie incrémentale (le `/` final compte !)
- `rsync -avz --delete -e ssh src/ user@host:/dst/` — miroir distant compressé
- `rsync -avn ...` — `-n` = dry-run, à tester systématiquement d'abord
- `install -m 750 -o www-data f /srv/` — copier en fixant droits et propriétaire

Texte : lire, filtrer, transformer

- `less +F fichier.log` — comme `tail -f` mais navigable (`q` pour sortir)
- `tail -f -n 100 app.log` — suivre un fichier en temps réel
- `head -n 20` / `tail -n +2` — 20 premières lignes / à partir de la ligne 2
- `grep -rniE 'error|fail' /var/log` — récursif, insensible à la casse, regex étendue
- `grep -v '^#' -e '^$' conf` — exclure commentaires et lignes vides
- `grep -c` / `-l` / `-o` — compter / lister les fichiers / n'afficher que la correspondance
- `grep -A3 -B3 'motif'` — contexte après / avant
- `rg 'motif' --type ts` — ripgrep : bien plus rapide que grep
- `sed -i 's/ancien/nouveau/g' f` — remplacement en place (`-i.bak` pour sauvegarder)
- `sed -n '10,20p' f` — afficher seulement les lignes 10 à 20
- `sed '/^#/d' f` — supprimer les lignes commentées
- `awk '{print $1, $NF}' f` — première et dernière colonne
- `awk -F: '$3>=1000 {print $1}' /etc/passwd` — utilisateurs humains
- `awk '{s+=$1} END {print s}'` — somme d'une colonne
- `cut -d, -f1,3 data.csv` — extraire des colonnes
- `sort -u` / `sort -k2 -n` / `sort -h` — unique / par colonne numérique / tailles humaines
- `uniq -c | sort -rn | head` — top des occurrences
- `tr -d '\r' < f > f.unix` — supprimer les CR (fichiers Windows)
- `wc -l` / `-c` / `-w` — compter lignes / octets / mots
- `diff -u a b` ; `vimdiff a b` — comparer deux fichiers
- `jq '.items[] | .name' f.json` — parser du JSON
- `column -t -s, f.csv` — aligner en colonnes lisibles
- `xargs -a liste.txt -I{} cp {} /dst` — construire des commandes depuis une liste
- `comm -13 a.txt b.txt` — lignes présentes seulement dans `b` (fichiers triés)

sed ou awk : lequel choisir ?

C'est la question qui revient à chaque fois, alors autant poser la règle en une ligne : **grep filtre, sed transforme, awk calcule.**

- Je pense « remplacer / supprimer / insérer » -> **sed** (la ligne est une chaîne de caractères)

- Je pense « colonne 3, somme, si tel champ » -> **awk** (la ligne est un tableau de champs)
- J'ai besoin de mémoire entre les lignes -> **awk** (compteurs, tableaux, bloc `END`)
- Ça dépasse 3 lignes d'awk -> **passer à Python**, ce sera plus lisible dans 6 mois
- Position fixe et simple -> **cut**, plus rapide à écrire que les deux

sed : substituer et éditer des lignes

- `sed 's/ancien/nouveau/' f` — 1re occurrence de chaque ligne
- `sed 's/ancien/nouveau/g' f` — toutes les occurrences
- `sed -i.bak 's/a/b/g' f` — modifié en place en gardant `f.bak`
- `sed 's#/usr/local#/opt#g' f` — changer de délimiteur pour éviter les `\/`
- `sed -E 's/([0-9]+)-([0-9]+)/\2-\1/' f` — regex étendue + groupes de capture
- `sed -n '10,20p' f` ; `sed -n '$p' f` — afficher lignes 10 à 20 / la dernière
- `sed -n '/debut/,/fin/p' f` — extraire un bloc entre deux motifs
- `sed -n '120,175p;175q' f | nl -ba -v120` — extrait 120-175 numérotées comme dans le fichier
- `...p;175q` — le `q` arrête sed à 175 au lieu de lire tout le fichier
- `nl -ba -v120` — `-ba` numérote aussi les lignes vides, `-v` fixe le départ
- `sed '/^#/d ; /^$/d' f` — enlever commentaires et lignes vides
- `sed '1d' f` ; `sed '$d' f` — supprimer l'entête / la dernière ligne
- `sed '/motif/s/a/b/' f` — substituer seulement sur les lignes contenant `motif`
- `sed 's/^/# /' f` — commenter toutes les lignes
- `sed 's/[[:space:]]*$//' f` — supprimer les espaces en fin de ligne
- `sed '3i\ligne' f` ; `sed '3a\ligne' f` — insérer avant / après la ligne 3
- `sed -n 's/^Version: //p' f` — extraire une valeur (`-n` + `p` = seulement ce qui matche)
- `sed 's/.*/\U&/' f` — tout en majuscules (GNU sed)

Toujours tester sans `-i` d'abord. Et attention : `-i` n'existe pas tel quel sur macOS/BSD.

awk : colonnes, filtres et calculs

- `awk '{print $1, $3}' f` — champs 1 et 3 (`$0` = ligne entière, `$NF` = dernier)
- `awk -F: '{print $1}' /etc/passwd` — choisir le séparateur d'entrée
- `awk -F'\t' -v OFS=, '{print $1,$2}' f` — TSV vers CSV
- `awk 'NR==1 {next} {...}' f` — sauter la ligne d'entête
- `awk 'NR>=10 && NR<=20' f` — plage de lignes (`NR` = numéro de ligne)
- `awk '$3 > 100' f` — filtrer sur une valeur numérique
- `awk '/erreur/ {print $2}' f` — filtrer par motif puis extraire
- `awk '{s+=$2} END {print s}' f` — somme d'une colonne
- `awk '{s+=$1} END {print s/NR}' f` — moyenne
- `awk '{a[$1]++} END {for (k in a) print a[k], k}' f` — compter par clé (puis `| sort -rn`)
- `awk '!vu[$0]++' f` — dédoublonner sans trier (garde l'ordre)
- `awk '{print NF}' f` ; `awk 'NF==0'` — nombre de champs / lignes vides
- `awk 'length > 80' f` — lignes trop longues
- `awk -v seuil=100 '$2 > seuil' f` — passer une variable du shell à awk
- `awk '{printf "%-20s %6.2f\n", $1, $2}' f` — mise en forme en colonnes
- `awk 'BEGIN {...} {...} END {...}'` — avant / pour chaque ligne / après : la structure d'awk
- `ss -tulpn | awk 'NR>1 {print $5}'` — usage typique en bout de pipe
- `du -s * | awk '$1 > 1e6 {print $2}'` — filtrer une sortie chiffrée

Redirections, pipes & substitutions

- `cmd > f` / `>> f` — écraser / ajouter (stdout)
- `cmd 2> err.log` / `&> tout.log` — stderr / stdout+stderr
- `cmd 2>&1 | tee -a log` — tout dans le pipe et dans un fichier
- `cmd < f` / `cmd <<< 'texte'` — entrée depuis fichier / depuis une chaîne
- `cat <<'EOF' > f ... EOF` — heredoc (quotes = pas d'expansion)
- `cmd1 && cmd2 || cmd3` — si succès / sinon
- `$(cmd)` — substitution de commande
- `diff <(cmd1) <(cmd2)` — substitution de processus
- `cmd | tee /dev/tty | wc -l` — voir et traiter en même temps
- `cmd > /dev/null 2>&1` — silence total

Permissions, propriétés & sudo

- `chmod 640 f` ; `chmod u+x,g-w f` — octal ou symbolique
- `chmod -R u=rwX,go=rX dir` — `X` = `+x` seulement sur les répertoires
- `chown -R user:group dir` — changer propriétaire et groupe
- `umask 022` — droits par défaut des nouveaux fichiers
- `chmod g+s dir` — setgid : héritage du groupe dans le répertoire
- `chmod +t /tmp` — sticky bit : seul le propriétaire peut supprimer
- `getfacl f` ; `setfacl -m u:jean:rw f` — droits fins (ACL)
- `lsattr` / `chattr +i f` — rendre un fichier immuable
- `sudo -i` / `sudo -u www-data cmd` — shell root / exécuter en tant qu'un autre utilisateur
- `sudo visudo` — éditer sudoers avec vérification syntaxique
- `sudo -l` — lister ses propres droits sudo

Utilisateurs & groupes

- `adduser jean` — création interactive (Debian, préférer à `useradd`)
- `usermod -aG docker,sudo jean` — ajouter à des groupes (`-a` obligatoire !)
- `deluser --remove-home jean` — supprimer un compte et son home
- `passwd jean` ; `passwd -l jean` — changer / verrouiller un mot de passe
- `id jean` ; `groups jean` — uid, gid, groupes
- `su - jean` — changer d'utilisateur avec son environnement
- `chage -l jean` — expiration du mot de passe
- `who` / `w` / `last` / `lastlog` — sessions actives et historique de connexion
- `useradd -r -s /usr/sbin/nologin svc` — compte de service sans shell

Paquets : APT & dpkg

- `sudo apt update && sudo apt upgrade` — mettre à jour l'index puis les paquets
- `sudo apt full-upgrade` — upgrade autorisant les suppressions (ex-`dist-upgrade`)
- `apt search motif` ; `apt show paquet` — chercher / détailler
- `sudo apt install ./paquet.deb` — installer un `.deb` avec ses dépendances
- `sudo dpkg -i paquet.deb` — installation d'un `.deb` local, sans résolution des dépendances
- `sudo apt -f install` — à lancer juste après `dpkg -i` pour compléter les dépendances
- `sudo dpkg -r paquet` ; `sudo dpkg -P paquet` — désinstalle / + fichiers de conf (sans les dépendances)
- `sudo apt remove` / `purge paquet` — désinstaller / + fichiers de conf
- `sudo apt autoremove --purge` — nettoyer les dépendances orphelines
- `apt list --installed | grep nginx` — vérifier ce qui est installé
- `apt list --upgradable` — mises à jour disponibles
- `apt-mark hold/unhold paquet` — figer une version
- `apt policy paquet` — versions candidates et dépôts sources
- `dpkg -l | grep ^rc` — paquets désinstallés avec conf résiduelle
- `dpkg -L paquet` ; `dpkg -S /chemin` — fichiers du paquet / paquet fournissant un fichier
- `sudo dpkg --configure -a` — réparer une installation interrompue
- `sudo apt --fix-broken install` — résoudre des dépendances cassées
- `sudo add-apt-repository ppa:xxx/ppa` — ajouter un PPA (Ubuntu)
- `ls /etc/apt/sources.list.d/` — dépôts additionnels
- `sudo unattended-upgrade --dry-run -d` — tester les MAJ de sécurité automatiques
- `cat /var/run/reboot-required` — un redémarrage est-il nécessaire ?
- `snap list` / `sudo snap refresh` — paquets snap (Ubuntu)

Processus & charge

- `ps aux --sort=-%mem | head` — top consommateurs mémoire
- `ps -ef --forest` — arborescence des processus
- `pgrep -af node` ; `pkill -f 'node app'` — chercher / tuer par motif de ligne de commande
- `kill -15 PID` puis `kill -9 PID` — arrêt propre (TERM) puis forçage (KILL)
- `htop` / `btop` — moniteur interactif (F5 arbre, F6 tri, F9 kill)
- `top -o %CPU` — sans installation supplémentaire
- `nice -n 10 cmd` ; `renice -n 5 -p PID` — baisser la priorité
- `nohup cmd &` / `setsid cmd` — détacher du terminal
- `jobs` / `fg %1` / `bg %1` / `disown` — gestion des jobs du shell
- `lsof -i :8080` ; `lsof -p PID` — qui utilise ce port / fichiers ouverts
- `fuser -k 8080/tcp` — tuer ce qui occupe un port
- `timeout 30s cmd` — limiter la durée d'exécution
- `watch -n 2 'df -h'` — réexécuter toutes les 2 s
- `strace -p PID -f -e trace=openat` — tracer les appels système

systemd : services & unités

- `systemctl status nginx` — état, PID, extrait des logs
- `sudo systemctl start|stop|restart|reload nginx` — cycle de vie du service
- `sudo systemctl enable --now nginx` — activer au boot + démarrer tout de suite
- `sudo systemctl disable --now nginx` — désactiver et arrêter
- `systemctl is-active` / `is-enabled` / `is-failed` — réponses courtes, idéales en script
- `systemctl list-units --type=service --state=running` — services actifs
- `systemctl --failed` — unités en échec

- `systemctl cat nginx` ; `systemctl show nginx` — définition / toutes les propriétés
- `sudo systemctl edit nginx` — override propre dans `/etc/systemd/system/*.d/`
- `sudo systemctl daemon-reload` — après modification d'un fichier `.service`
- `systemctl list-timers --all` — tâches planifiées systemd
- `systemd-analyze blame` ; `critical-chain` — ce qui ralentit le démarrage
- `sudo systemctl mask/unmask svc` — interdire totalement le démarrage
- `loginctl` / `hostnamectl` / `timedatectl` — sessions / nom d'hôte / heure et NTP
- `sudo timedatectl set-timezone Europe/Paris` — fuseau horaire

Journaux & diagnostic

- `journalctl -u nginx -f` — suivre les logs d'un service
- `journalctl -u nginx --since '1 hour ago'` — fenêtre temporelle (`--until` aussi)
- `journalctl -p err -b` — erreurs depuis le dernier boot
- `journalctl -b -1` — logs du boot précédent (utile après crash)
- `journalctl --disk-usage` ; `--vacuum-time=7d` — taille des logs / purge
- `journalctl -k` / `dmesg -T -w` — messages du noyau (OOM killer, disques, USB)
- `tail -f /var/log/syslog /var/log/auth.log` — logs classiques Debian/Ubuntu
- `logger -t monapp 'message'` — écrire dans le journal depuis un script
- `sudo logrotate -d /etc/logrotate.conf` — tester la rotation sans l'appliquer
- `uptime` ; `free -h` ; `vmstat 1` — charge, mémoire, activité système
- `iostat -xz 1` ; `iotop` — IO disque (paquet `sysstat`)
- `dmesg | grep -i 'out of memory'` — confirmer un kill par manque de RAM

Disques, partitions & montage

- `df -hT` ; `df -i` — espace libre par système de fichiers / inodes
- `lsblk -f` — disques, partitions, UUID, points de montage
- `blkid` — UUID et type de système de fichiers
- `sudo mount /dev/sdb1 /mnt` ; `umount /mnt` — monter / démonter
- `sudo mount -a` — monter tout `/etc/fstab` (à tester avant reboot !)
- `UUID=xxx /data ext4 defaults,noatime 0 2` — ligne type de `/etc/fstab`
- `sudo mkfs.ext4 -L data /dev/sdb1` — formater
- `sudo fdisk -l` ; `sudo parted /dev/sdb print` — table de partitions
- `sudo e2fsck -f /dev/sdb1` — vérifier un ext4 (démonté)
- `sudo resize2fs /dev/sdb1` — agrandir après extension de partition
- `sudo smartctl -a /dev/sda` — santé du disque (`smartmontools`)
- `sudo swapon --show` ; `free -h` — état du swap
- `sudo dd if=x.iso of=/dev/sdX bs=4M status=progress conv=fsync` — clé USB bootable (vérifier `/dev/sdX` !)

LVM : volumes logiques

Principe : **PV** (disque) -> **VG** (pool) -> **LV** (volume). Un LV se formate et se monte comme une partition.

- `sudo pvs` ; `vgs` ; `lvs` — vue rapide des volumes physiques, groupes, logiques
- `sudo pvdisplay` ; `vgdisplay` ; `lvdisplay` — même chose en détaillé
- `sudo lsblk` — voir l'empilement disque -> PV -> LV
- `sudo pvcreate /dev/sdb1` — initialiser un disque comme volume physique
- `sudo vgcreate vg_data /dev/sdb1` — créer un groupe de volumes
- `sudo vgextend vg_data /dev/sdc1` — ajouter un disque à un groupe existant
- `sudo lvcreate -L 50G -n lv_srv vg_data` — créer un volume logique de 50 Go
- `sudo lvcreate -l 100%FREE -n lv_srv vg_data` — utiliser tout l'espace restant
- `sudo mkfs.ext4 /dev/vg_data/lv_srv` — formater le volume logique
- `sudo mount /dev/vg_data/lv_srv /srv` — monter (puis ligne dans `/etc/fstab`)
- `sudo lvextend -L +20G -r /dev/vg_data/lv_srv` — agrandir de 20 Go, `-r` redimensionne le FS
- `sudo lvextend -l +100%FREE -r /dev/vg_data/lv_srv` — absorber tout l'espace libre du VG
- `sudo lvreduce -L -10G -r /dev/vg_data/lv_srv` — réduire (sauvegarder avant, ext4 seulement)
- `sudo lvcreate -L 5G -s -n snap /dev/vg_data/lv_srv` — instantané avant une mise à jour risquée
- `sudo lvconvert --merge /dev/vg_data/snap` — restaurer l'instantané (au prochain montage)
- `sudo lvremove /dev/vg_data/lv_srv` — supprimer un volume logique
- `sudo vgreduce vg_data /dev/sdc1` ; `pvremove` — retirer un disque du groupe
- `sudo pvmove /dev/sdb1` — déplacer les données avant de retirer un disque
- Ubuntu : `ubuntu-vg` / `ubuntu-lv` — noms par défaut de l'installateur (souvent à étendre)
- `df -h` vs `lvs` — si `lvs` est plus grand, le FS n'a pas été redimensionné (`-r`)

Réseau

- `ip a` ; `ip -br a` — adresses IP (vue compacte)
- `ip r` ; `ip route get 1.1.1.1` — table de routage / route utilisée
- `ss -tulpn` — ports en écoute avec les processus
- `ss -tan state established` — connexions établies
- `ping -c4` / `mtr host` — connectivité / traceroute continu
- `dig +short example.com` ; `dig @1.1.1.1 A x` — résolution DNS
- `resolvectl status` ; `resolvectl query x` — DNS via systemd-resolved
- `curl -I https://site` — en-têtes HTTP seulement
- `curl -sS -X POST -H 'Content-Type: application/json' -d '{}' url` — requête API
- `curl -w '%{time_total}\n' -o /dev/null -s url` — mesurer un temps de réponse
- `wget -c url` — téléchargement reprenable
- `nc -zv host 443` — tester l'ouverture d'un port
- `tcpdump -i eth0 -nn port 443 -c 20` — capture de paquets
- `hostnamectl set-hostname srv01` — changer le nom d'hôte
- `sudo netplan try` ; `sudo netplan apply` — config réseau Ubuntu (`/etc/netplan/*.yaml`)
- `nmcli device status` ; `nmcli con up X` — NetworkManager (poste de travail)
- `ip link set eth0 up/down` — activer / désactiver une interface

SSH & accès distant

- `ssh-keygen -t ed25519 -C 'poste-perso'` — générer une clé moderne
- `ssh-copy-id -i ~/.ssh/id_ed25519.pub user@host` — installer sa clé publique
- `ssh -p 2222 user@host` — port non standard
- `ssh -i cle.pem -o IdentitiesOnly=yes user@host` — forcer une clé précise
- `~/.ssh/config` : `Host` / `HostName` / `User` / `Port` / `IdentityFile` — alias de connexion
- `ssh -L 8080:localhost:80 user@host` — tunnel local (accès à un service distant)
- `ssh -R 9000:localhost:3000 user@host` — tunnel inverse (exposer un service local)
- `ssh -D 1080 user@host` — proxy SOCKS
- `ssh -J bastion user@interne` — rebond via une machine intermédiaire
- `scp -r dir user@host:/dst` ; `sftp user@host` — copie de fichiers
- `ssh-add -l` ; `ssh-agent bash` — clés chargées dans l'agent
- `sudo sshd -t && sudo systemctl reload ssh` — valider la conf avant de recharger
- `PermitRootLogin no` / `PasswordAuthentication no` — durcissement dans `/etc/ssh/sshd_config`
- `tmux new -s work` ; `tmux a -t work` — sessions persistantes (Ctrl+B d pour détacher)

Pare-feu & sécurité

- `sudo ufw status verbose` — état du pare-feu
- `sudo ufw allow 22/tcp` ; `sudo ufw enable` — ouvrir SSH **avant** d'activer !
- `sudo ufw allow from 192.168.1.0/24 to any port 5432` — règle sur une source précise
- `sudo ufw delete allow 8080` ; `ufw status numbered` — supprimer une règle
- `sudo ufw default deny incoming` — politique par défaut
- `sudo nft list ruleset` ; `iptables -L -n -v` — règles bas niveau
- `sudo fail2ban-client status sshd` — IP bannies
- `sudo fail2ban-client set sshd unbanip 1.2.3.4` — débannir
- `grep 'Failed password' /var/log/auth.log | tail` — tentatives d'intrusion SSH
- `sudo lynis audit system` — audit de durcissement
- `gpg --verify fichier.sig fichier` — vérifier une signature
- `sha256sum -c SHA256SUMS` — vérifier une empreinte

Archives & compression

- `tar -czf arch.tar.gz dir/` — créer une archive gzip
- `tar -xzf arch.tar.gz -C /dst` — extraire vers un répertoire
- `tar -tzf arch.tar.gz | head` — lister sans extraire
- `tar -cJf arch.tar.xz dir/` — xz : plus lent, plus compact
- `tar --exclude='node_modules' -czf a.tgz .` — exclure des chemins
- `zip -r a.zip dir` ; `unzip -l a.zip` — format zip
- `gzip -9 f` ; `gunzip f.gz` — compresser un fichier seul
- `zstd -19 f` ; `unzstd f.zst` — excellent ratio vitesse/taille
- `7z x archive.7z` — paquet `p7zip-full`

Planification : cron, at, timers

- `crontab -e` ; `crontab -l` — tâches de l'utilisateur courant
- `sudo crontab -u www-data -e` — cron d'un autre utilisateur
- `*/15 * * * * cmd` — min, heure, jour, mois, jour-semaine
- `0 3 * * 1 /opt/backup.sh >> /var/log/bk.log 2>&1` — tous les lundis à 3 h, avec logs
- `@reboot cmd` — au démarrage
- `/etc/cron.d/` ; `/etc/cron.daily/` — cron système (champ utilisateur en plus)
- `at 22:00 -f script.sh` ; `atq` ; `atrm 3` — exécution unique différée
- `monjob.timer` + `monjob.service` — alternative moderne à cron
- `OnCalendar=*-*-* 03:00:00` ; `Persistent=true` — syntaxe timer (rattrape si machine éteinte)
- `systemd-analyze calendar 'Mon *-*-* 03:00'` — vérifier une expression `OnCalendar`

Alias & fonctions du shell

- `alias ll='ls -lAh'` — crée un raccourci (valable pour la session courante)
- `alias` — liste tous les alias définis
- `unalias ll` — supprime un alias
- `\ls` — contourne ponctuellement un alias (antislash devant)
- `type ll` — alias, fonction, builtin ou binaire ?
- `~/.bash_aliases` — fichier dédié, chargé par `~/.bashrc` sur Debian/Ubuntu
- `source ~/.bashrc` (ou `. ~/.bashrc`) — recharge la conf sans rouvrir le terminal
- `alias gs='git status'` ; `alias ..='cd ..'` — exemples classiques
- `alias rm='rm -i'` — filet de sécurité, mais masque le comportement réel
- `alias sudo='sudo '` — l'espace final permet à sudo de résoudre tes alias
- `mkcd() { mkdir -p "$1" && cd "$1"; }` — un alias ne prend pas d'arguments : dès qu'il en faut un, écrire une fonction
- `/etc/profile.d/mesalias.sh` — alias pour tous les utilisateurs de la machine
- zsh : `~/.zshrc` au lieu de `~/.bashrc` — même syntaxe pour les alias et les fonctions

Variables du shell

- `var=valeur` — aucun espace autour du `=`
- `echo "$var"` ; `echo "${var}_suffixe"` — toujours guillemeter ; accolades pour délimiter
- `export VAR=valeur` — transmet la variable aux processus enfants
- `VAR=valeur commande` — variable définie pour cette seule commande
- `unset VAR` ; `readonly VAR=x` — supprime / rend non modifiable
- `env` ; `printenv VAR` ; `set` — environnement / une variable / tout, shell inclus
- `export PATH="$HOME/bin:$PATH"` — ajouter un répertoire d'exécutables
- `$HOME $USER $PWD $SHELL $LANG $EDITOR` — variables usuelles fournies par le système
- `$? $$ $0 $# $@` — code retour, PID, nom du script, nb d'arguments, arguments
- `resultat=$(commande)` — récupère la sortie d'une commande
- `${var:-defaut}` — valeur de repli si la variable est vide
- `${var:=defaut}` ; `${var:?message}` — affecte le repli / échoue avec un message
- `${#var}` — longueur de la chaîne
- `${var#prefixe}` ; `${var%.txt}` — retire le début / la fin
- `${var/ancien/nouveau}` — remplace (`//` pour toutes les occurrences)
- `${var^^}` ; `${var,,}` — majuscules / minuscules (bash)
- `arr=(a b c)` ; `${arr[1]}` ; `${arr[@]}` ; `${#arr[@]}` — tableaux : élément, tous, nombre
- `local var=x` — variable limitée à une fonction
- `set -a ; source .env ; set +a` — exporter d'un coup toutes les variables d'un `.env`
- `~/.bashrc` vs `~/.profile` — shell interactif vs session de connexion
- `/etc/environment` — variables globales (pas d'`export`, pas d'expansion)
- `printenv | grep -i proxy` — vérifier une variable d'environnement

Scripts bash : l'essentiel

- `#!/usr/bin/env bash` — shebang portable
- `set -Eeuo pipefail` — arrêt sur erreur, variable non définie, échec dans un pipe
- `IFS=$'\n\t'` — séparateur sûr : évite les surprises avec les espaces
- `trap 'rm -rf "$tmp"' EXIT` — nettoyage garanti à la sortie
- `tmp=$(mktemp -d)` — répertoire temporaire sûr
- `"${var:-defaut}"` / `"${var:?message}"` — valeur par défaut / erreur si vide
- `if [[ -f "$f" && -r "$f" ]]; then ... fi` — tests (préférer `[[ ]]` à `[ ]`)
- `for f in *.log; do echo "$f"; done` — boucle (toujours guillemeter les variables)
- `while read -r line; do ...; done < f` — lire un fichier ligne par ligne
- `case "$1" in start) ...;; *) ...;; esac` — aiguillage
- `mafonc() { local x=$1; ...; }` — fonction avec variable locale
- `$?` / `$#` / `$@` / `$0` / `$$` — code retour / nb args / args / nom / PID
- `shellcheck script.sh` — linter indispensable
- `bash -x script.sh` — mode trace pour déboguer

Neovim + kickstart.nvim

- `~/.config/nvim/init.lua` — le fichier unique de kickstart, à lire et éditer directement
- `lua/kickstart/plugins/` ; `lua/custom/plugins/` — modules optionnels / tes propres plugins
- `<leader>` = Espace — appuyer sur Espace seul affiche which-key
- `:Lazy` ; `:Lazy update` ; `:Lazy sync` — gestionnaire de plugins (`lazy-lock.json` figé les versions)
- `:Mason` — installer serveurs LSP, formateurs et linters
- `:checkhealth` — diagnostic complet (clipboard, parsers, providers)
- `:ConformInfo` ; `<leader>f` — formateur actif / formater le buffer

Telescope (recherche)

- `<leader>sf` ; `<leader>sg` — chercher un fichier / grep dans le projet
- `<leader><leader>` ; `<leader>s.` — buffers ouverts / fichiers récents
- `<leader>sw` ; `<leader>/` — mot sous le curseur / recherche dans le buffer courant
- `<leader>sh` ; `<leader>sk` ; `<leader>sn` — aide / raccourcis / fichiers de config nvim
- `<leader>sd` ; `<leader>sr` — diagnostics / relancer la dernière recherche

LSP (défauts standardisés depuis nvim 0.11/0.12)

- `grd` ; `grr` ; `gri` ; `grt` — définition / références / implémentation / type
- `grn` ; `gra` — renommer le symbole / code action
- `K` ; `gO` ; `gW` — doc flottante / symboles du fichier / du projet
- `]d` ; `[d` ; `<leader>q` — diagnostic suivant / précédent / liste quickfix
- `Ctrl+o` ; `Ctrl+i` — revenir / avancer dans la liste de sauts

Édition & fenêtres

- `gcc` ; `gc` en visuel — commenter une ligne / une sélection
- `sa)` ; `sd)` ; `sr)'` — mini.surround : ajouter / supprimer / remplacer un entourage
- `Ctrl+h/j/k/l` — naviguer entre les fenêtres (`:vsplit`, `:split`)
- `<Esc>` — effacer la surbrillance de recherche

Les mappings évoluent avec les versions : en cas de doute, `<leader>` puis attendre, ou `:map`.

Nouveautés nvim 0.12

- `:restart` — redémarrer nvim sans quitter (réattache l'UI)
- `:lsp` — nouvelle commande ; `LspInfo`, `LspRestart` et `LspLog` ont disparu
- `vim.pack.add { { src = 'https://...' } }` — gestionnaire de plugins intégré (alternative à lazy.nvim)
- `vim.o.autocomplete = true` — autocomplétion native en insertion, sans plugin
- `completeopt = menu,menuone,noselect,popup` — à régler avec `autocomplete`
- `:Undotree` ; `:Diff` — plugins désormais fournis avec nvim
- `vim.net.request()` ; `:edit https://...` — requêtes HTTP natives
- Code lens en lignes virtuelles, diagnostics de workspace — autres apports 0.12

Docker (auto-hébergement)

- `docker ps -a` ; `docker stats` — conteneurs / consommation temps réel
- `docker logs -f --tail 100 nom` — suivre les logs
- `docker exec -it nom bash` — shell dans un conteneur
- `docker compose up -d` ; `down` — démarrer / arrêter une stack
- `docker compose pull && docker compose up -d` — mettre à jour les images
- `docker compose logs -f service` — logs d'un service de la stack
- `docker system df` ; `docker system prune -a --volumes` — espace utilisé / grand nettoyage (destructif)
- `docker volume ls` ; `docker inspect nom` — volumes / configuration détaillée
- `docker run --rm -it -v $PWD:/w -w /w image cmd` — conteneur jetable sur le répertoire courant
- `docker image prune -f` — supprimer les images orphelines

Infos système & dépannage

- `uname -a` ; `lsb_release -a` ; `cat /etc/os-release` — noyau et version de la distribution
- `hostnamectl` — résumé : OS, noyau, architecture, virtualisation
- `lscpu` ; `lsmem` ; `free -h` — CPU et mémoire
- `lspci` ; `lsusb` ; `lsmod` — matériel et modules noyau
- `sensors` ; `uptime` — températures / durée de fonctionnement

- `systemctl reboot` ; `systemctl poweroff` — redémarrer / éteindre proprement
- `sudo update-alternatives --config editor` — choisir l'outil par défaut
- `env` / `printenv` / `export VAR=val` — variables d'environnement
- `echo $PATH` ; `source ~/.bashrc` — PATH/ recharger la conf du shell
- `ls /etc/*release` ; `dpkg --print-architecture` — identifier précisément la machine
- GRUB -> Advanced options -> recovery mode — récupération : console root, fsck, réinstallation de paquets
- `mount -o remount,rw /` — repasser en écriture en mode secours

Fichiers & chemins à connaître

- `/etc/fstab` — montages permanents
- `/etc/hosts` ; `/etc/resolv.conf` — résolution locale ; DNS (géré par resolved)
- `/etc/ssh/sshd_config` ; `~/.ssh/config` — serveur SSH ; client SSH
- `/etc/apt/sources.list.d/` — dépôts de paquets
- `/etc/systemd/system/` — unités locales et overrides (prioritaires)
- `/lib/systemd/system/` — unités fournies par les paquets (ne pas éditer)
- `/etc/netplan/*.yaml` — configuration réseau Ubuntu
- `/etc/crontab` ; `/etc/cron.d/` — tâches planifiées système
- `/var/log/{syslog,auth.log,dpkg.log}` — journaux principaux
- `/etc/environment` ; `~/.bashrc` ; `~/.profile` — variables et conf du shell
- `/etc/sudoers.d/` — droits sudo par fichier (éditer avec `visudo -f`)
- `/proc/cpuinfo` ; `/proc/meminfo` — infos noyau en temps réel
- `/var/lib/docker/` — images, volumes et conteneurs

Signaux, codes retour & rappels

- `SIGTERM` 15 / `SIGKILL` 9 / `SIGHUP` 1 — arrêt propre / brutal / rechargement de conf
- `SIGINT` 2 (Ctrl+C) / `SIGSTOP` 19 / `SIGCONT` 18 — interruption / pause / reprise
- `echo $?` -> 0 succès ; toute valeur non nulle = échec
- 126 / 127 / 130 — non exécutable / commande introuvable / Ctrl+C
- **Toujours guillemeter** `"$var"` — sinon les espaces cassent tout
- **Tester avec** `-n` / `--dry-run` — rsync, apt, ufw et logrotate le proposent
- **Sauvegarder avant d'éditer une conf** — `cp fichier fichier.bak-$(date +%F)`
- **Vérifier la syntaxe avant reload** — `sshd -t`, `nginx -t`, `visudo -c`, `netplan try`
- **Ne jamais `rm -rf` avec une variable vide** — `rm -rf "${dir:?}"/` protège du pire

Index : à quoi sert cette commande ?

Le trajet inverse du reste de la fiche : on croise une commande dans un tuto ou dans la sortie d'un agent, et on veut savoir ce qu'elle fait avant de l'exécuter.

- `alias` — crée un raccourci de commande
- `apt` / `dpkg` — gestionnaire de paquets / couche bas niveau
- `at` — exécute une commande une seule fois plus tard
- `awk` — traite du texte colonne par colonne
- `base64` — encode/décode en base64
- `basename` / `dirname` — extrait le nom du fichier / le chemin du dossier
- `bc` — calculatrice en ligne de commande
- `cal` / `date` — calendrier / date et heure
- `cat` / `tac` — affiche un fichier / à l'envers
- `chmod` / `chown` — change les droits / le propriétaire
- `chroot` — exécute dans une autre racine (dépannage)
- `cmp` / `diff` — compare octet par octet / ligne par ligne
- `column` — aligne du texte en colonnes
- `cp` / `mv` / `rm` — copie / déplace / supprime
- `crontab` — édite les tâches planifiées
- `curl` / `wget` — requêtes HTTP / téléchargement de fichiers
- `cut` / `paste` — découpe des colonnes / les recolle
- `dd` — copie bloc à bloc (images disque, clés USB)
- `df` / `du` — espace libre par disque / poids des fichiers
- `dig` / `nslookup` — interroge le DNS
- `dmesg` — messages du noyau (matériel, OOM)
- `echo` / `printf` — affiche du texte / avec formatage
- `env` / `export` — environnement / définit une variable
- `file` — devine le type réel d'un fichier
- `find` / `locate` — cherche des fichiers à la volée / via un index
- `free` / `vmstat` — mémoire / activité système
- `fuser` / `lsof` — qui utilise ce fichier ou ce port
- `getent` — interroge les bases système (passwd, hosts)
- `git` — gestion de versions
- `grep` / `rg` — cherche un motif dans du texte
- `gzip` / `xz` / `zstd` — compresse un fichier seul
- `head` / `tail` — début / fin d'un fichier
- `hostname` / `hostnamectl` — nom de la machine
- `htop` / `top` / `btop` — moniteur de processus interactif
- `id` / `groups` / `whoami` — identité et groupes de l'utilisateur
- `ip` — adresses, routes, interfaces réseau
- `iostat` / `iotop` — activité disque
- `jobs` / `fg` / `bg` — gère les tâches du shell
- `journalctl` — consulte les journaux systemd
- `jq` — manipule du JSON
- `kill` / `pkill` / `killall` — envoie un signal à un processus
- `less` / `more` — lit un fichier page par page
- `ln` — crée un lien (symbolique avec `-s`)
- `logger` — écrit dans le journal système
- `ls` / `tree` — liste un répertoire / son arborescence
- `lsblk` / `blkid` / `parted` — disques, partitions, UUID
- `lscpu` / `lspci` / `lsusb` / `lsmod` — matériel et modules du noyau
- `man` / `tldr` / `apropos` — documentation, exemples, recherche
- `md5sum` / `sha256sum` — empreinte d'un fichier
- `mkdir` / `rmdir` — crée / supprime un répertoire
- `mktemp` — crée un fichier ou dossier temporaire de manière sûre
- `mount` / `umount` — monte / démonte un système de fichiers
- `mkfs` / `fsck` / `resize2fs` — formate / vérifie / redimensionne un FS
- `nano` / `vim` — éditeurs de texte en terminal
- `nc` (netcat) / `socat` — teste ou relaie une connexion réseau
- `nice` / `renice` — ajuste la priorité CPU
- `nl` / `wc` — numérote les lignes / compte lignes, mots, octets
- `nohup` / `setsid` / `disown` — détache un processus du terminal
- `nproc` / `uptime` — nombre de CPU logiques / charge et durée de marche
- `openssl` — certificats, chiffrement, tests TLS
- `passwd` / `chage` — mot de passe et son expiration
- `ping` / `traceroute` / `mtr` — teste et trace un chemin réseau
- `ps` / `pgrep` — liste ou cherche des processus
- `pv` — affiche la progression d'un flux dans un pipe
- `pvs` / `vgs` / `lvs` — volumes LVM (physiques, groupes, logiques)
- `pwd` / `cd` — répertoire courant / changer de répertoire
- `readlink` / `realpath` — résout un chemin réel
- `rename` — renomme en masse avec une regex
- `rsync` — synchronise des fichiers, local ou distant
- `scp` / `sftp` — copie de fichiers via SSH
- `sed` — remplace ou supprime du texte par flux
- `seq` / `shuf` — génère une suite / mélange des lignes
- `shellcheck` — analyse et corrige les scripts bash
- `sleep` / `timeout` / `watch` — attend / limite / répète une commande
- `sort` / `uniq` — trie / déduplique des lignes
- `split` / `csplit` — découpe un gros fichier
- `ss` / `netstat` — connexions et ports en écoute
- `ssh` / `ssh-keygen` / `ssh-copy-id` — connexion distante et gestion des clés
- `stat` — métadonnées d'un fichier (dates, inode)
- `strace` / `ltrace` — trace les appels système d'un processus
- `su` / `sudo` — change d'utilisateur / exécute en root
- `systemctl` — pilote les services et unités systemd
- `tar` — crée et extrait des archives
- `tee` — écrit dans un fichier et sur la sortie
- `time` — mesure la durée d'exécution
- `tmux` / `screen` — sessions terminal persistantes
- `touch` — crée un fichier vide ou change sa date
- `tr` — remplace ou supprime des caractères
- `ufw` / `iptables` / `nft` — pare-feu
- `umask` — droits par défaut des nouveaux fichiers
- `uname` / `lsb_release` — noyau / version de la distribution
- `unzip` / `zip` / `7z` — archives zip
- `useradd` / `usermod` / `adduser` — gestion des comptes
- `which` / `type` / `command -v` — localise une commande
- `who` / `w` / `last` — sessions ouvertes et historique
- `xargs` — construit des commandes depuis une liste
- `xxd` / `hexdump` — affiche un fichier en hexadécimal
- `yes` — répond automatiquement à une invite

`man`, `tldr` et `shellcheck` restent les meilleurs compléments à cette fiche. Et en cas de doute sur une commande destructrice : `--dry-run` d'abord.

Si vous voulez le contexte — pourquoi j'ai eu besoin de construire ça après des années à « faire du Linux » — c'est dans cet article (https://dev2go.vercel.app/blog/maitriser-linux-ligne-de-commande).