Cheat Sheet

LINUX

Linux is an open-source operating system used widely in servers, development, robotics, and cloud environments. It's powerful because everything is controlled by commands, scripts, and permissions. For developers and engineers, Linux is the foundation:

- *File system navigation (moving around, editing, copying files).*
- *Process control (starting, stopping, monitoring apps).*
- *Networking (checking connections, downloading files).*
- *System monitoring (CPU, memory, storage).*

Linux commands are short but powerful, often combined in scripts to automate tasks.

Basic Navigation & File Management

`ls`

Lists files and directories in the current folder

`ls -la`

Lists files with details (permissions, owner, size) including hidden files

`pwd`

Prints the full path of the current working directory

`cd folder`

Changes directory to folder, Use **cd ..** to go up one level

`mkdir folder`

Creates a new directory named folder

`rm file`

Removes a file. Be careful — no recycle bin!

`rm -r folder`

Removes a directory and its contents recursively

`rm -rf folder`

Forcefully and recursively deletes a folder and all its contents without asking for confirmation (very dangerous)

`mv old new`

Renames a file or directory if new is a name, or moves it if new is a folder

`touch file`

Creates an empty file, or updates the timestamp of an existing one

`cat file`

Displays the contents of a file directly in the terminal

`less file`

Opens a file for scrolling through page by page (q = quit)

`nano file`

Opens a simple terminal-based text editor

`grep text file`

Searches for text inside file

`find . -name "*.txt"`

Finds all files ending with .txt starting in the current directory

`history`

Shows all previously used commands

`clear`

Clears the terminal screen

`chmod +x file.sh`

Makes a file executable (so you can run scripts)

`tar -czvf file.tar.gz folder/`

Compresses a folder into a .tar.gz archive

`tar -xzvf file.tar.gz`

Extracts a .tar.gz archive

Filesystem / Disk

`tree`

Displays a visual directory tree (needs sudo apt install tree)

`du -sh folder/`

Shows the size of a folder in human-readable format

`df -h`

Shows free and used disk space on mounted filesystems

Processes

`ps aux`

Lists all running processes

`htop`

Interactive process viewer (more user-friendly than top)

`kill -9 PID`

Forcefully kills a process by its Process ID

Networking

`ping host`

Tests if a host (like google.com) is reachable

`curl URL`

Fetches content from a URL (useful for APIs)

`ifconfig / ip a`

Shows network interfaces and IP addresses

Package Management (APT)

`sudo apt update`

Updates the list of available packages from repositories

`sudo apt upgrade`

Installs the newest versions of all installed packages

`sudo apt install package`

Installs a package by name (e.g., sudo apt install git)

`sudo apt remove package`

Removes an installed package but keeps its config files

`sudo apt autoremove`

Cleans up unnecessary dependencies no longer needed

System Info

`uname -a`

Prints system information (kernel, architecture)

`uptime`

Shows how long the system has been running

`free -h`

Displays memory usage (RAM)

`lscpu`

Shows CPU architecture details

`lsusb`

Lists USB devices connected to the system

`lspci`

Lists PCI devices (e.g., graphics, network cards)

`dmesg | less`

Shows kernel and hardware messages (useful for debugging hardware)

Niko Korošec

Permissions & Users

sudo command

Runs a command as administrator (superuser)

whoami

Prints the current logged-in username

adduser name

Creates a new user

passwd

Changes the password for the current user.

DOCKER

Docker is a container platform that lets you run applications in isolated environments. Instead of installing everything on your computer, Docker packages code and dependencies into an image, which runs as a container. Key ideas:

- *Image: template with app + environment.*
- *Container: running instance of an image.*
- *Docker Hub: online library of images.*
- *Docker CLI: commands to start, stop, and manage containers.*

Developers use Docker for testing, consistency, and running apps the same way on any machine.

Containers

docker ps

Lists running containers

docker ps -a

Lists all containers, including stopped ones

docker run image

Runs a new container from an image

docker start container_id

Starts a stopped container

docker stop container_id

Stops a running container

docker restart container_id

Restarts a container

docker rm container_id

emoves a stopped container

docker exec -it container bash

Opens an interactive shell inside a running container-bash or sh

Images

docker images

Lists all downloaded images

docker pull image

Downloads an image from Docker Hub

docker rmi image_id

Removes an image

docker build -t name .

Builds an image from a Dockerfile in the current directory

System & Info

docker logs container_id

Shows logs of a container

docker inspect container_id

Displays detailed info about a container

docker system prune

Removes stopped containers, unused networks, and dangling images to free space

Docker Useful

docker stats

Shows real-time CPU, memory, and network usage of containers

docker network ls

Lists Docker networks

docker volume ls

Lists Docker volumes

docker-compose ps

Lists running containers managed by Compose - Multi-container application manager

GIT

Git is a version control system that tracks changes in code. It lets multiple developers work on the same project without overwriting each other. Key ideas:

- *Repository (repo): project folder tracked by Git.*
- *Commit: snapshot of your changes with a message.*
- *Branch: separate line of development (e.g., testing vs main).*
- *Push/Pull: sending and receiving changes from a remote repo*

Git is essential for collaboration, backups, and experimenting safely.

Networking

git init

Initializes a new Git repository in the current folder

git clone URL

Copies a remote repository to your local machine

git status

Shows the current state of changes (staged, unstaged, untracked)

git add .

Stages all modified and new files for the next commit

git commit -m "msg"

Saves staged changes with a descriptive message

git push -u origin main

Uploads local commits to the main branch of the remote repo

git pull

Fetches and merges changes from the remote repository into the current branch

git branch

Lists all local branches

git checkout branch

Switches to another branch

git checkout -b new-branch

Creates and switches to a new branch

git merge branch

Merges another branch into the current branch

git log --oneline

Shows a compact history of commits

git diff

Shows unstaged changes between working files and the last commit

git stash

Temporarily saves uncommitted changes and cleans your working directory

git reset --hard commit_id

Resets repository to a specific commit, discarding later changes (destructive)

git revert commit_id

Safely undoes a specific commit by creating a new one

Niko Korošec