Navigation & Filesystem Concepts (Core)

Linux Commands Course · Section 1

What is a File?

In Linux, everything is treated as a file — ordinary files, directories, devices, sockets, even processes.

A file is a named collection of data stored on disk.

Examples:

- Text files contain readable text.
- Binary files contain executable or machine data.
- Directories special files that list other files.

Long Format (ls -l) Explained

The long format shows multiple properties of each file:

Example output:

-rw-r--r-- 1 student users 4096 Oct 22 10:30 notes.txt

Parts of this line:

- 1. **Type & permissions** file type and access rights
- 2. Links number of hard links
- 3. **Owner** user who owns the file
- 4. **Group** group owning the file
- 5. Size file size in bytes6. Date & time last modification
- 7. **Name** file name

This view helps you identify files and their properties at a glance.

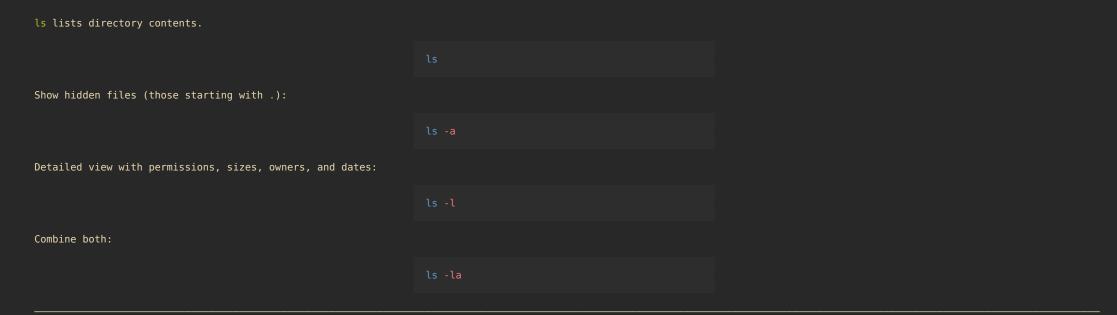
Home Directory Meaning

Every user has a personal home directory — their private workspace.		
It's where:		
 Your personal files and folders are stored. Configuration files (dotfiles) live. You usually start when logging in. 		
Path example:		
	/home/student	
Shortcut:		
~ always expands to your current user's home directory.		

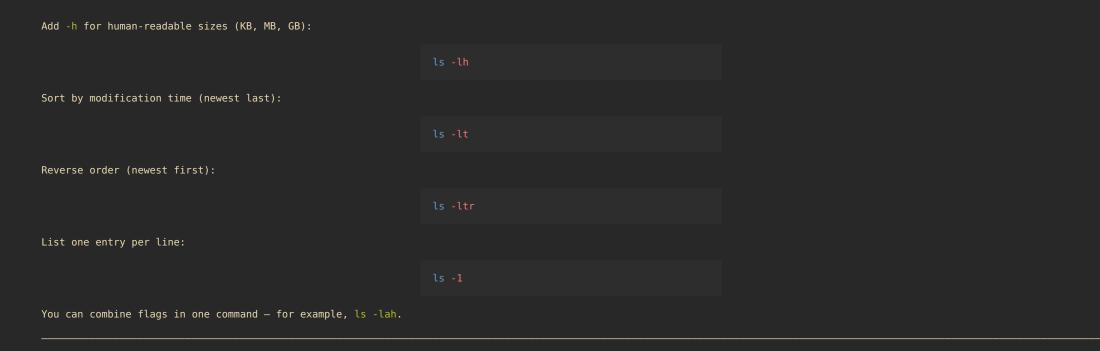
Where am I?

Show your current working directory:		
	pwd	
Example output:		
	/home/student	
This tells you your exact location in the filesystem hierarchy.		

Listing files - ls



Human-friendly details



Colorized output & file types

Many distros colorize ls output automatically (directories in blue, executables in green).

Each leading character in ls -l shows type:

regular file d directory l symbolic link	Symbol	Туре
, ,	d l c	directory symbolic link character device

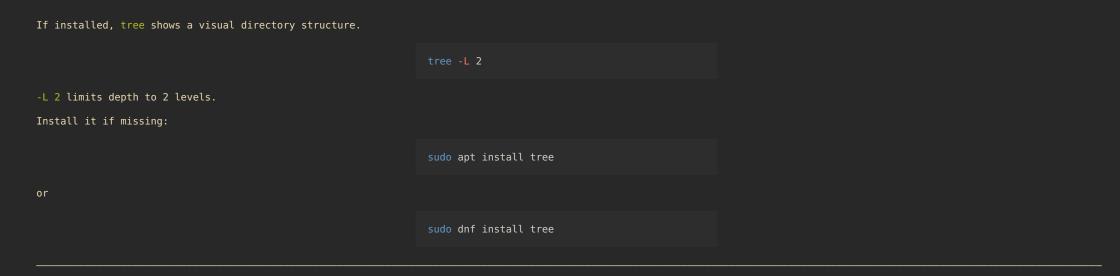
Changing directories — cd

Move to another location:		
	cd /etc	
Return to your home directory:		
	cd	
Go up one level (parent directory):		
	cd	
Return to the previous working directory:		
	cd -	

Home directory shortcut

~ always represents your home directory.		
	cd ~	
To access subfolders in home, append paths:		
	cd ~/Documents	
~user accesses another user's home (if permitted).		

Tree view (optional tool)



Absolute vs Relative Paths

Absolute paths start from / (root).
Relative paths start from your current directory.

Examples:

Туре	Example	Meaning
Absolute Relative		Always points to the same location Moves relative to where you are

Tip: Use pwd before running a command to confirm your location.

Globbing — Wildcards

The shell expands wildcard patterns automatically before running the command.

Pattern	Matches
* ? [abc] [0-9]	any number of any characters any single character any one of a, b, or c any digit anything except x

Example:

ls *.txt

lists all files ending in .txt in the current directory.

Brace Expansion {}

Create multiple arguments or names in one command.

echo file_{a,b,c}.txt

→ expands to file_a.txt file_b.txt file_c.txt

Make multiple directories:

mkdir project/{src,bin,docs}

Environment Variables

Special variables store information about your shell environment.		
Show your home directory path:		
	echo \$HOME	
Show your PATH (where executables are searched):		
	echo \$PATH	
PATH is a colon-separated list of directories.		

PATH in action

When you type a command name, the shell searches each directory in \$PATH from left to right.		
You can inspect the order by printing it:		
	echo \$PATH	
To see where a command is found:		
	which ls	
If multiple versions exist, the first one found in \$PATH runs.		

Recap

- pwd print current directory
 ls, ls -lah list files with detail
 cd, cd .., cd - move around
 tree visual hierarchy
 Absolute vs relative paths know your context
 Wildcards * ? [] { } powerful pattern matching
 \$HOME, \$PATH key environment variables