

Navigation commands

pwd – Print Working Directory

```
Shows the current location in the filesystem.

**Output example:
/Users/yourname/projects
```

ls – List Directory Contents

Lists files and folders in the current directory.

- ls -l → Detailed list (permissions, size, date)
- ls $-a \rightarrow$ Shows hidden files (those starting with .)
- ls -la → Combined

MOutput image of the commands-

```
angel@angel-VirtualBox:~$ pwd
/home/angel
angel@angel-VirtualBox:~$ ls
Desktop first Music second Templates
Documents first.c Pictures second.c Videos
Downloads fourth.c Public snap vscode.deb
angel@angel-VirtualBox:~$ []
```

cd – Change Directory

Moves into a directory.

cd folder_name Examples:

cd Documents # Go to Documents cd .. # Go up one level cd / # Go to root cd ~ # Go to home directory

2. File and Directory Management

```
mkdir - Make Directory
Creates a new folder.

mkdir new_folder
touch - Create File
Creates an empty file.

touch file.txt
```

Output image of the commands-

```
angel@angel-VirtualBox:~/projects$ pwd
/home/angel/projects
angel@angel-VirtualBox:~/projects$ mkdir workfolder
angel@angel-VirtualBox:~/projects$ cd workfolder
angel@angel-VirtualBox:~/projects/workfolder$ touch file1.txt file2.txt file3.txt
angel@angel-VirtualBox:~/projects/workfolder$ ls
file1.txt file2.txt file3.txt
angel@angel-VirtualBox:~/projects/workfolder$ []
```

cp – Copy Files or Directories

```
cp source.txt destination.txt
Copy folder:
cp -r folder1 folder2
```

mv – Move or Rename Files

```
mv oldname.txt newname.txt
mv file.txt ~/Documents/ # Move file
```

rm – Remove Files

```
rm file.txt # Delete file
rm -r folder_name # Delete folder (recursively)

⚠ Be careful! There is no undo.
```

Image: (changing names, removing files using the mv, rm commands)

```
angel@angel-VirtualBox:~/projects/workfolder$ cp file1.txt copy1.txt
angel@angel-VirtualBox:~/projects/workfolder$ ls
copy1.txt file1.txt file2.txt file3.txt
angel@angel-VirtualBox:~/projects/workfolder$ mv file2.txt file123.txt
angel@angel-VirtualBox:~/projects/workfolder$ ls
copy1.txt file123.txt file1.txt file3.txt
angel@angel-VirtualBox:~/projects/workfolder$ [
```

📸3. File Viewing & Editing

```
cat – View File Contents
Displays content in terminal.

cat file.txt
nano – Edit Files in Terminal
```

```
A basic terminal-based text editor.

nano file.txt
Use arrows to move
CTRL + 0 to save
CTRL + X to exit
clear - Clears the Terminal
clear
Shortcut: CTRL + L
```

Table 1 Output image of the commands-

```
angel@angel-VirtualBox:~/projects/workfolder$ echo "hello,world" > file1.txt
angel@angel-VirtualBox:~/projects/workfolder$ cat file1.txt
hello,world
angel@angel-VirtualBox:~/projects/workfolder$ echo "hello,i am angel" > file3.txt
angel@angel-VirtualBox:~/projects/workfolder$ cat file3.txt
hello,i am angel
angel@angel-VirtualBox:~/projects/workfolder$ cat file3.txt file1.txt
hello,i am angel
hello,world
angel@angel-VirtualBox:~/projects/workfolder$ [
```

File Permissions with chmod and chown

1. Understanding File Permissions in Linux

Each file/directory in Linux has:

- **Owner** → The user who created the file.
- **Group** → A group of users who may share access.
- Others → Everyone else.
- Permission Types
 - **r** → Read (4 in numeric)
 - w → Write (2 in numeric)
 - **x** → Execute (1 in numeric)
- Permission Layout

Example from 1s -1:

```
-rwxr-xr--
```

Breakdown:

- - → Regular file (d = directory, l = symlink, etc.)
- rwx → Owner has read, write, execute

- r-x → Group has read, execute
- $r-- \rightarrow$ Others have read only

MOutput image of the commands-





Syntax

```
chmod [options] mode filename
```

Modes can be set in **numeric (octal)** or **symbolic** form.

(A) Numeric (Octal) Method

Each permission is represented as a number:

- Read = 4
- Write = 2
- Execute = 1

Add them up:

- 7 = rwx
- 6 = rw-
- 5 = r x
- 4 = r -
- O = ---

Example:

```
chmod 755 script.sh
```

Meaning:

- Owner: 7 → rwx
- Group: $5 \rightarrow r x$
- Others: 5 → r-x

(B) Symbolic Method

Use u (user/owner), g (group), o (others), a (all). Operators:

- + → Add permission
- - → Remove permission
- = → Assign exact permission

Examples:

```
chmod u+x script.sh  # Add execute for owner
chmod g-w notes.txt  # Remove write from group
chmod o=r file.txt  # Set others to read only
chmod a+r report.txt  # Everyone gets read access
```

(C) Recursive Changes

```
chmod -R 755 /mydir
```

• $-R \rightarrow$ applies changes recursively to all files/subdirectories.

Output image of chmod commands-

```
angel@angel-VirtualBox:~/projects/workfolder$ touch test.sh
angel@angel-VirtualBox:~/projects/workfolder$ ls -l test.sh
-rw-rw-r-- 1 angel angel 0 Sep 10 21:21 test.sh
angel@angel-VirtualBox:~/projects/workfolder$ chmod +x test.sh
angel@angel-VirtualBox:~/projects/workfolder$ ls -l test.sh
-rwxrwxr-x 1 angel angel 0 Sep 10 21:21 test.sh
angel@angel-VirtualBox:~/projects/workfolder$ echo 'echo "hello"' > test.sh
angel@angel-VirtualBox:~/projects/workfolder$ ./test.sh
hello
angel@angel-VirtualBox:~/projects/workfolder$ []
```

3. chown – Change File Ownership

Syntax

```
chown [options] new_owner:new_group filename
```

Examples:

🚀 4. Putting It All Together

Example Scenario

```
touch project.sh
ls -l project.sh
```

Output:

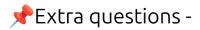
```
-rw-r--r 1 angel dev 0 Aug 19 12:00 project.sh
```

Now:

```
chmod 700 project.sh  # Only owner has rwx
chmod u+x,g-w project.sh  # Add execute for user, remove write for group
chown root:admin project.sh # Change owner to root and group to admin
```


Numeric	Permission	Meaning
0		No access
1	X	Execute only
2	-W-	Write only
3	-wx	Write + Exec
4	Г	Read only
5	Г-Х	Read + Exec
6	rw-	Read + Write
7	ГWХ	Full access

W Key Tip: Use numeric for quick settings (e.g., 755, 644) and symbolic for fine adjustments (u+x, g-w).



Q1 Difference Between chmod and chown?

In Unix-based operating systems (Linux, macOS), chmod and chown are two important commands used to manage files and directories. While both are related to permissions and ownership, they serve different

purposes.

chmod - Change Mode

The **chmod** command is used to change the **permissions** of a file or directory. Permissions define who can read, write, or execute a file.

Permissions:

- **Read (r)**: Permission to view the contents of a file.
- Write (w): Permission to modify or delete the file.
- Execute (x): Permission to run the file as a program (for executable files or scripts).

Example Usage:

1. chmod 755 myfile

Grants:

- Read, write, and execute permissions to the **owner**.
- Read and execute permissions to the **group** and **others**.
- 2. chmod u+x script.sh

Adds execute permission to the **owner** of **script.sh**.

3. chmod 644 myfile.txt

Grants:

- Read and write permissions to the **owner**.
- Read permission to the **group** and **others**.

chown - Change Owner

The **chown** command is used to change the **ownership** of a file or directory. This includes both the **user owner** and the **group owner**.

- [user] specifies the new owner of the file.
- [group] specifies the new group of the file (optional).

Example Usage:

```
1. chown alice:admins file.txt
  Changes the owner of file.txt to alice and the group to admins.
```

2. chown bob file.txt

Changes the **owner** of file.txt to bob, while keeping the existing group unchanged.

3. chown :staff file.txt
Changes the group of file.txt to staff (no change to the owner).

Summary of Differences:

Command	Purpose	Example Usage	
chmod	Changes permissions of files	<pre>chmod 755 myfile (Set read, write, execute)</pre>	
chown	Changes ownership of files	<pre>chown alice:admins file.txt (Change owner and group)</pre>	

Understanding the difference between chmod and chown is crucial for managing file access and ownership in a secure environment. Use chmod to modify what users can do with a file, and chown to modify who owns the file and its associated group.

Q2 How do you check current directory and user?

Checking the Current Directory and Current User

In Unix-based systems (Linux, macOS), you can easily check your **current directory** and **current user** using simple terminal commands.

Check the Current Directory

To check your **current directory**, use the pwd command:

pwd

Checking the Current Directory and Current User

In Unix-based systems (Linux, macOS), you can easily check your **current directory** and **current user** using simple terminal commands.

1. Check the Current Directory

To check your **current directory**, use the pwd command:

pwd

\$ pwd

/home/alice/projects

Checking the Current User

In Unix-based systems (Linux, macOS), you can check the **current user** using the **whoami** command.

Command

To find out the current user, use the following command in the terminal:

whoami