

Study_Sessions_Linux

Prepared by Andres Pascasio

[GitHub Profile](#)

Linux Basics Study Session

1. Overview

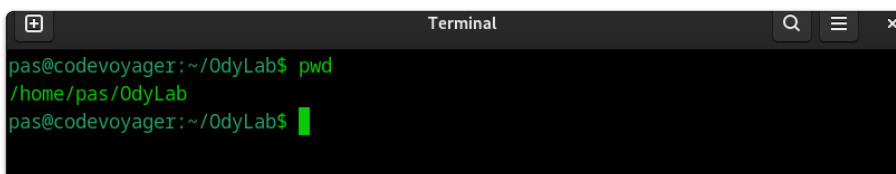
Objective: Learn basic Linux commands to navigate and manage files and directories.

2. Commands Overview

1. pwd - Print Working Directory

- **Purpose:** Displays the full path of the current working directory.
- **Usage:**

```
pwd
```

A terminal window titled "Terminal" with a search icon, a menu icon, and a close icon in the top right corner. The terminal shows the prompt "pas@codevoyager: ~/OdyLab\$ " followed by the command "pwd" in green. The output is "/home/pas/OdyLab" in green. The prompt is then followed by another "pwd" command in green, and the cursor is at the end of the line.

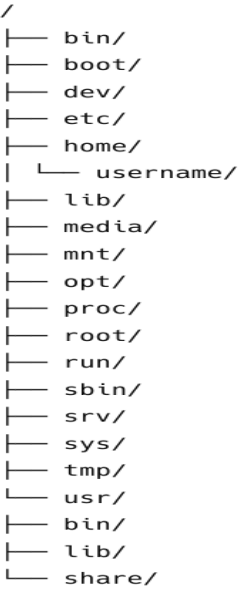
```
pas@codevoyager: ~/OdyLab$ pwd
/home/pas/OdyLab
pas@codevoyager: ~/OdyLab$
```

- **Description:**
The `pwd` command is useful to determine which directory you are currently in, especially when navigating through multiple folders in the terminal. When executed, it will show the complete path from the root of the filesystem to your current directory.

Note

Try it out in the Practice section: [Exercise 1 Navigating Directories](#).

The Unix/Linux filesystem is organized in a hierarchical structure, starting from the root directory `/`. Here's a brief overview of key directories:

Filesystem Diagram	Explanation
	<p><code>/</code>: Root Directory-The top level of the filesystem hierarchy.</p> <p><code>/bin/</code>: Contains essential command binaries needed for basic system operation.</p> <p><code>/boot/</code>: Holds files required for system booting, such as the kernel.</p> <p><code>/dev/</code>: Contains device files representing hardware components and peripherals.</p> <p><code>/etc/</code>: Stores configuration files for the system and applications.</p> <p><code>/home/</code>: Contains home directories for users. Each user has a personal directory under <code>/home/</code>, such as <code>/home/username/</code>.</p> <p><code>/lib/</code>: Contains essential shared libraries needed by binaries in <code>/bin/</code> and <code>/sbin/</code>.</p> <p><code>/media/</code>: Provides mount points for removable media like CDs and USB drives.</p> <p><code>/mnt/</code>: A generic mount point for temporarily mounting filesystems.</p> <p><code>/opt/</code>: Used for optional application software packages.</p> <p><code>/proc/</code>: A virtual filesystem providing information about system processes and hardware.</p> <p><code>/root/</code>: The home directory for the root user (system administrator).</p> <p><code>/run/</code>: Contains runtime data and information about the system's current state.</p> <p><code>/sbin/</code>: Contains system binaries used for system administration tasks.</p> <p><code>/srv/</code>: Holds data for services provided by the system, such as web or FTP services.</p> <p><code>/sys/</code>: A virtual filesystem providing information about the system's hardware.</p> <p><code>/tmp/</code>: Stores temporary files used by applications and the system.</p> <p><code>/usr/</code>: Contains user-related programs and data, including additional binaries, libraries, and shared data.</p>

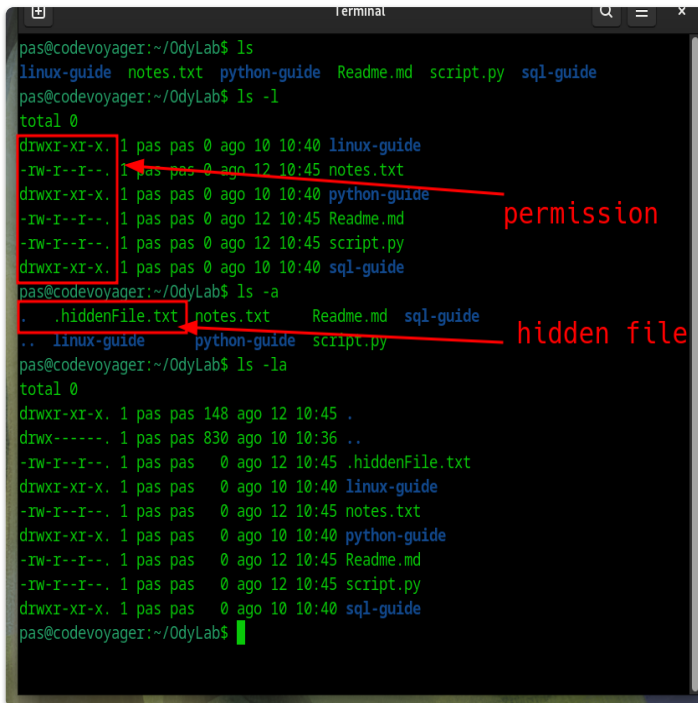
Tip

Understanding this structure will help you navigate the filesystem effectively and use commands like `cd`, `cp` and `mv` to manage files and directories.

2. `ls` - List Directory Contents

- **Purpose:** Lists files and directories in the current directory.
- **Usage:**

```
ls
ls -l # Detailed list
ls -a # Include hidden files
```



A terminal window showing the output of the `ls` command. The prompt is `pas@codevovager:~/OdyLab$`. The first command is `ls`, which lists files: `linux-guide`, `notes.txt`, `python-guide`, `Readme.md`, `script.py`, and `sql-guide`. The second command is `ls -l`, which shows a detailed listing. Red arrows point to the first column of the `ls -l` output, with the label "permission" in red text. The third command is `ls -a`, which lists hidden files including `..`, `.hiddenFile.txt`, and the other files. A red arrow points to `.hiddenFile.txt` with the label "hidden file" in red text. The fourth command is `ls -la`, which shows a detailed listing including hidden files.

```
pas@codevovager:~/OdyLab$ ls
linux-guide  notes.txt  python-guide  Readme.md  script.py  sql-guide
pas@codevovager:~/OdyLab$ ls -l
total 0
drwxr-xr-x. 1 pas pas 0 ago 10 10:40 linux-guide
-rw-r--r--. 1 pas pas 0 ago 12 10:45 notes.txt
drwxr-xr-x. 1 pas pas 0 ago 10 10:40 python-guide
-rw-r--r--. 1 pas pas 0 ago 12 10:45 Readme.md
-rw-r--r--. 1 pas pas 0 ago 12 10:45 script.py
drwxr-xr-x. 1 pas pas 0 ago 10 10:40 sql-guide
pas@codevovager:~/OdyLab$ ls -a
. .hiddenFile.txt notes.txt Readme.md sql-guide
.. linux-guide python-guide script.py
pas@codevovager:~/OdyLab$ ls -la
total 0
drwxr-xr-x. 1 pas pas 148 ago 12 10:45 .
drwx----- 1 pas pas 830 ago 10 10:36 ..
-rw-r--r--. 1 pas pas 0 ago 12 10:45 .hiddenFile.txt
drwxr-xr-x. 1 pas pas 0 ago 10 10:40 linux-guide
-rw-r--r--. 1 pas pas 0 ago 12 10:45 notes.txt
drwxr-xr-x. 1 pas pas 0 ago 10 10:40 python-guide
-rw-r--r--. 1 pas pas 0 ago 12 10:45 Readme.md
-rw-r--r--. 1 pas pas 0 ago 12 10:45 script.py
drwxr-xr-x. 1 pas pas 0 ago 10 10:40 sql-guide
pas@codevovager:~/OdyLab$
```

Note

Try it out in the Practice section: [Exercise 1 Navigating Directories](#).

3. `cd` - Change Directory

- **Purpose:** Changes the current directory.
- **Usage:**

```
cd /path/to/directory
cd .. # Go up one directory
cd ~ # Go to the home directory
```

Note

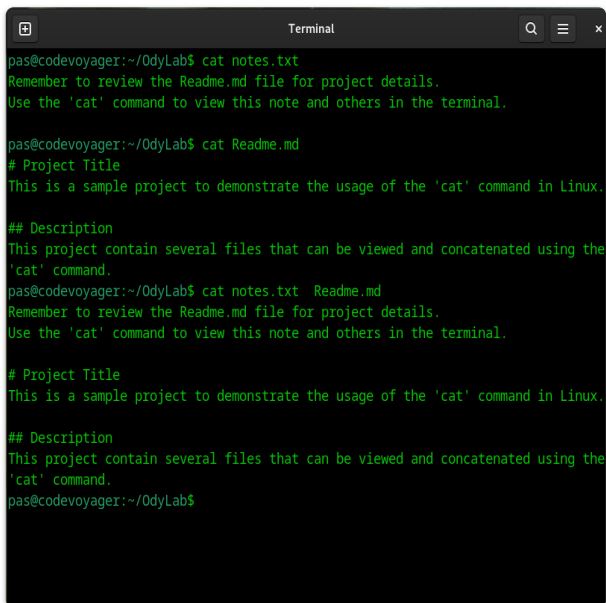
Try it out in the Practice section: [Exercise 1 Navigating Directories](#).

Try it out in the Practice section: [Exercise 2 Managing Files and Permissions](#).

4. `cat` - Concatenate and Display Files

- **Purpose:** Concatenates files and prints their contents to the terminal.
- **Usage:**

```
cat file.txt # Display contents of a file
cat file1.txt file2.txt # Concatenate and display multiple files`
```

A terminal window titled "Terminal" with search, menu, and close icons. It shows the following commands and output:
pas@codevoyager:~/OdyLab\$ cat notes.txt
Remember to review the Readme.md file for project details.
Use the 'cat' command to view this note and others in the terminal.

pas@codevoyager:~/OdyLab\$ cat Readme.md
Project Title
This is a sample project to demonstrate the usage of the 'cat' command in Linux.

Description
This project contain several files that can be viewed and concatenated using the 'cat' command.
pas@codevoyager:~/OdyLab\$ cat notes.txt Readme.md
Remember to review the Readme.md file for project details.
Use the 'cat' command to view this note and others in the terminal.

Project Title
This is a sample project to demonstrate the usage of the 'cat' command in Linux.

Description
This project contain several files that can be viewed and concatenated using the 'cat' command.
pas@codevoyager:~/OdyLab\$

Note

Try it out in the Practice section: [Exercise 3: Editing Files](#).

5. `touch` - Create an Empty File

- **Purpose:** Creates a new empty file or updates the timestamp of an existing file.
- Usage:**

```
touch newfile.txt # Create a new empty file
```

```
pas@codevoyager:~/OdyLab/sql-guide$ touch createTable.sql
pas@codevoyager:~/OdyLab/sql-guide$ ls
createTable.sql
pas@codevoyager:~/OdyLab/sql-guide$ cat createTable.sql
pas@codevoyager:~/OdyLab/sql-guide$
```

Note

Try it out in the Practice section: [Exercise 4: Directory and File Management](#).

6. `mkdir` - Make Directory

- **Purpose:** Creates a new directory.

Usage:

```
mkdir new_directory # Create a new directory
mkdir -p /path/to/new_directory # Create nested directories
```

Note

Try it out in the Practice section: [Exercise 2 Managing Files and](#)

Try it out in the Practice section: [Exercise 4: Directory and File Management](#).

7. `rm` - Remove Files or Directories

- **Purpose:** Deletes files or directories.

Usage:

```
rm file.txt # Remove a file
rmdir directory_name # Remove a directory
```

Caution

The `rm` command permanently deletes files and directories. Once deleted, files cannot be easily recovered. Always double-check before using `rm` to avoid accidental data loss. For safer file removal, consider using `rm -i` to prompt for confirmation before deletion.

8. `cp` - Copy Files or Directories

- **Purpose:** Copies files or directories from one location to another.

Usage:

```
cp source_file.txt destination.txt # Copy a file
cp -r source_directory/ destination_directory/ # Copy a directory and its
contents
```

9. `mv` - Move or Rename Files and Directories

- **Purpose:** Moves or renames files and directories.

Usage:

```
mv old_name.txt new_name.txt # Rename a file
mv file.txt /path/to/destination/ # Move a file to a new location
```

Editors

10. `vi` - Text Editor

- **Purpose:** A powerful text editor.
- **Usage:**

```
vi file.txt # Open a file in vi
```

- **## Key Commands**

Important

Command	Description
i	Enter insert mode
Esc	Return to command mode
:wq	Save and quit
:q!	Quit without saving
o	Insert a new line below
dd	Delete the current line



11. nano - Simple Text Editor

- **Purpose:** A user-friendly text editor.
- **Usage:**

```
nano file.txt # Open a file in nano
```

- **## Key Commands**

Info

Command	Description
Ctrl + O	Save the file
Ctrl + X	Exit Nano
Ctrl + K	Cut the current line
Ctrl + U	Paste the cut line
Ctrl + W	Search for text
Ctrl + G	Display help screen

File Management

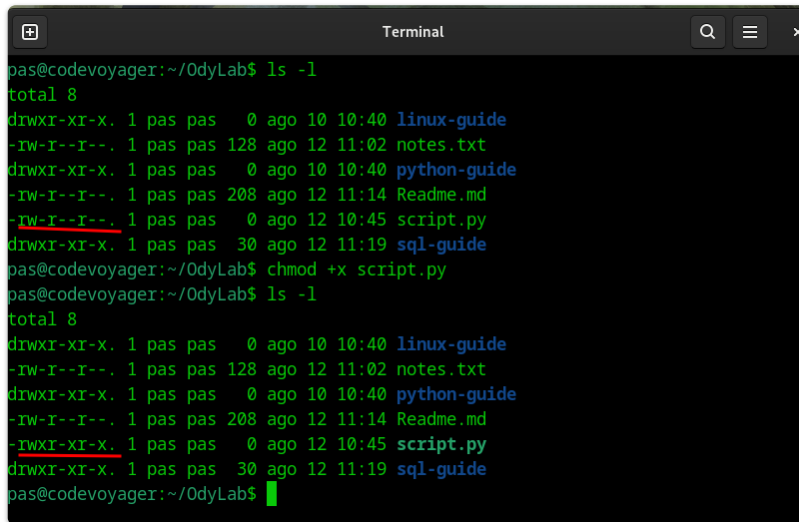
While basic file navigation commands like `cd`, `ls`, and `mkdir` are fundamental, managing file permissions is also crucial for effective file management. Understanding how to

control file permissions is essential for both securing and properly configuring files, especially when working with scripts or sensitive data.

12. `chmod` - Change File Permissions

- **Purpose:** Changes the permissions of a file or directory.
- **Usage:**

```
chmod 755 file.txt # Set read, write, execute for owner, read and execute for others
chmod +x script.sh # Add execute permission
```



```
pas@codevoyager:~/OdyLab$ ls -l
total 8
drwxr-xr-x. 1 pas pas  0 ago 10 10:40 linux-guide
-rw-r--r--. 1 pas pas 128 ago 12 11:02 notes.txt
drwxr-xr-x. 1 pas pas  0 ago 10 10:40 python-guide
-rw-r--r--. 1 pas pas 208 ago 12 11:14 Readme.md
-rw-r--r--. 1 pas pas  0 ago 12 10:45 script.py
drwxr-xr-x. 1 pas pas 30 ago 12 11:19 sql-guide
pas@codevoyager:~/OdyLab$ chmod +x script.py
pas@codevoyager:~/OdyLab$ ls -l
total 8
drwxr-xr-x. 1 pas pas  0 ago 10 10:40 linux-guide
-rw-r--r--. 1 pas pas 128 ago 12 11:02 notes.txt
drwxr-xr-x. 1 pas pas  0 ago 10 10:40 python-guide
-rw-r--r--. 1 pas pas 208 ago 12 11:14 Readme.md
-rwxr-xr-x. 1 pas pas  0 ago 12 10:45 script.py
drwxr-xr-x. 1 pas pas 30 ago 12 11:19 sql-guide
pas@codevoyager:~/OdyLab$
```

Description: The `chmod` (change mode) command is used to modify the access permissions of files and directories. Permissions determine who can read, write, or execute a file. Mastery of `chmod` is key for managing file security and ensuring that files have the appropriate access controls.

🔥 Important

- **Importance:** Properly setting file permissions is essential for protecting files and scripts from unauthorized access and ensuring that they function as intended. It is especially relevant when working with bash scripts or shared files.

🔥 Tip

- **Further Reading:** For a deeper understanding of file permissions and `chmod`, consider exploring detailed guides or tutorials on Linux file permissions. This

knowledge is fundamental for both everyday file management and more advanced administrative tasks.

Claro, aquí tienes una sección para `chown` que cubre su uso para cambiar la propiedad de archivos y directorios en Linux:

13. `chown` - Change File Ownership

- **Purpose:** Changes the ownership of a file or directory, allowing you to set the user and/or group that owns the file or directory.
- **Usage:**

```
chown user file.txt           # Change the owner of file.txt to
'user'
chown user:group file.txt     # Change the owner of file.txt to
'user' and the group to 'group'
chown -R user:group /path/to/dir # Recursively change ownership of all
files and directories within /path/to/dir
```

- **Options:**
 - `user`: The username or user ID to assign as the new owner.
 - `group`: The group name or group ID to assign as the new group owner.
 - `-R` or `--recursive`: Apply changes recursively to all files and directories within the specified directory.
- **Examples:**

```
chown alice file.txt          # Set 'alice' as the owner of
'file.txt'
chown bob:admin file.txt      # Set 'bob' as the owner and 'admin' as
the group for 'file.txt'
chown -R alice:users /home/alice # Recursively set 'alice' as the owner
and 'users' as the group for all files in /home/alice
```

Note

Only the root user or a user with appropriate privileges can change the ownership of files that they do not own.

Changing ownership of system files or files owned by other users can affect system security and operation, so use this command with caution.

3. Practice Exercises

Exercise 1: Navigating Directories

Objective: Use the `pwd`, `ls`, and `cd` commands to navigate the filesystem.

- **Step 1:** Open your terminal (or use one of the resources provided).
- **Step 2:** Run `pwd` to display your current directory.
- **Step 3:** List all files and directories in the current location using `ls -l`.
- **Step 4:** Navigate to a different directory using `cd /path/to/directory` and confirm the change with `pwd`.
- **Step 5:** Return to your home directory with `cd ~`.

Exercise 2: Managing Files and Permissions

Objective: Practice creating, modifying, and managing files and directories, along with setting permissions.

- **Step 1:** Create a new directory named `practice_dir` using `mkdir practice_dir`.
- **Step 2:** Navigate into this directory with `cd practice_dir`.
- **Step 3:** Create a new file using `touch testfile.txt`.
- **Step 4:** Change the permissions of this file to `755` using `chmod 755 testfile.txt`.
- **Step 5:** Add execute permission to the file with `chmod +x testfile.txt`.
- **Step 6:** Verify the permissions using `ls -l`.

Exercise 3: Editing Files

Objective: Learn to edit files using `vi` and `nano`.

- **Step 1:** Open `testfile.txt` in `vi` using `vi testfile.txt`.
- **Step 2:** Enter insert mode by pressing `i`, then type the following text:

Remember to review the README.md file for project details.
Use the ``cat`` command to view this note and others in the terminal.

then press `Esc` followed by `:wq` to save and exit.

- **Step 3:** Reopen the file in `nano` using `nano testfile.txt`, make additional edits, and save with `Ctrl + O`, then exit with `Ctrl + X`.
- **Step 4:** Use the `cat` command to view the contents of the file: `cat testfile.txt`
This will display the contents of `testfile.txt` in the terminal.

Exercise 4: Directory and File Management

Objective: Practice creating, renaming, and removing files and directories.

- **Step 1:** Open your terminal.
 - **Step 2:** Create a new directory named `work_dir`: `mkdir work_dir`
 - **Step 3:** Navigate into `work_dir`: `cd work_dir`
 - **Step 4:** Create two new files named `file1.txt` and `file2.txt`: `touch file1.txt file2.txt`
 - **Step 5:** Rename `file1.txt` to `renamed_file1.txt`: `mv file1.txt renamed_file1.txt`
 - **Step 6:** Remove `file2.txt`: `rm file2.txt`
 - **Step 7:** Move `renamed_file1.txt` to the parent directory: `mv renamed_file1.txt ..`
 - **Step 8:** Navigate back to the parent directory and remove the `work_dir`: `cd ..`
`rmdir work_dir`
-

4. Q&A and Wrap-up

4.1 Questions and Answers

- **Common Questions:**
 - **How do I undo a command in Linux?**
 - Linux does not have a built-in "undo" for commands, but you can use commands like `mv` to rename files back or restore from backups if available.
 - **What should I do if a command fails?**
 - Check for syntax errors, ensure you have the necessary permissions, and verify file paths.
 - **How can I get help with a specific command?**
 - Use `man command_name` or `command_name --help` to get more information about the command.
- **Troubleshooting Tips:**
 - **Check for Typos:**

- Ensure commands are typed correctly. Use tab completion to help avoid typos.
- **File Permissions:**
 - Verify and modify file permissions using `chmod` and `chown` if you encounter permission issues.
- **Command Help:**
 - Use `man` pages and `--help` options to understand command usage and options.

4.2 Key Takeaways

- **Basic Commands:**
 - **Listing Files:** `ls` shows files and directories.
 - **Navigating Directories:** Use `cd` to change directories.
 - **Managing Files:** `cp` (copy), `mv` (move), `rm` (remove), `mkdir` (create directory).
- **Navigation:**
 - **Paths:** Understand relative and absolute paths for efficient navigation.
- **File Management:**
 - **Permissions and Ownership:** Use `chmod` and `chown` to manage file permissions and ownership.

4.3 Feedback and Next Steps

- **Feedback:**

We value your input and would love to hear your thoughts on this guide. Please provide feedback by:

 - **Submitting Issues or Pull Requests on GitHub:**
 - Go to the [GitHub repository](#) for this guide.
 - Open an [issue](#) to report any errors, suggest improvements, or share your experience.
- **Next Steps:**
 - **Practice:**
 - Apply the commands and concepts you've learned by practising in a Linux environment. Setting up a virtual machine or using an online Linux terminal can provide hands-on experience.
 - Work on small projects or tasks that require using Linux commands to solidify your understanding.
 - **Advanced Topics:**

- Explore more advanced commands and features in Linux. Topics to consider include:
 - **Scripting:** Learn shell scripting to automate tasks and improve efficiency.
 - **System Administration:** Dive into system administration commands and tools for managing Linux systems.
 - **Networking:** Understand basic networking commands and configurations in Linux.
 - **Package Management:** Explore how to manage software packages using tools like `apt`, `yum`, or `dnf`.

Continuing to build on these topics will enhance your Linux skills and open up more possibilities for working with Linux systems.

5. Resources

If you don't have access to a Linux environment for practice, there are several websites and resources that offer interactive terminal environments directly in your browser. These platforms allow you to execute Linux commands and gain hands-on experience without needing to install anything on your local machine. Here are a few recommendations:

Additionally, if you wish to deepen your understanding of Linux commands, some of these resources provide comprehensive tutorials and explanations to help you build a strong foundation.

1. [JSLinux](#)

- **Description:** JSLinux is a fully functional Linux emulator that runs in your browser. It offers an experience that closely resembles a real Linux environment.
- **Functionality:** It provides an online terminal where you can practice commands and run programs without needing to install anything.
- **Ideal for:** Users who want a complete and realistic Linux experience within the browser.

2. [LinuxCommand](#)

- **Description:** This site offers a comprehensive tutorial on the Linux command line. Although it doesn't have an interactive terminal, it provides practical exercises and detailed explanations for each command.
- **Functionality:** It combines theory with practical examples. You can read about a command and then practice it on your own Linux system or another online

environment.

- **Ideal for:** Those who prefer a structured guide with examples and explanations before practicing.

3. [Webminal](#)

- **Description:** Webminal is an interactive online Linux lab. It allows you to practice Linux commands, write shell scripts, and learn various aspects of Linux system administration.
- **Functionality:** It provides a complete terminal environment where you can execute commands in real-time and receive instant feedback.
- **Ideal for:** Beginners and those who want to practice in a safe environment without needing to install anything.

Prepared by Andres Pascasio

[GitHub Profile](#)