Linux Command Line Fundamentals

Hands-On Beginner Lab Using JSLinux Fedora 33 RISC-V

Purpose

This document guides a brand-new Linux learner through a browser-based Linux lab environment and a sequence of step-by-step exercises. Each lab includes commands to type, what to observe, and a quick practice task.

Audience	Beginner / first-time Linux command-line learner
Estimated time	2 to 3 hours if completed carefully
Lab environment	JSLinux Fedora 33 RISC-V virtual machine in a web browser
Prerequisites	A computer with a modern browser and internet access
Created for	Personal learning, peer sharing, or guided practice

Training Roadmap

Complete the labs in order. The commands build on each other and are designed to be safe in the disposable browser-based training environment.

Lab	Topic	Main commands	Result
1	Start the JSLinux VM	browser, login shell	Working Linux prompt
2	Understand the prompt and basic commands	whoami, pwd, date, clear	Know where you are
3	Navigate the file system	ls, cd, ., .., /, ~	Move around safely
4	Create files and directories	mkdir, touch, echo	Build a practice workspace
5	View files	cat, less, head, tail	Read file content
6	Copy, move, rename, delete	cp, mv, rm, rmdir	Manage files
7	Use wildcards and history	*, ?, history, Ctrl+r	Work faster
8	Permissions basics	ls -l, chmod	Read permission strings
9	Search and filter text	grep, find, wc, sort, uniq	Locate information
10	Pipes and redirection	|, >, >>, 2>	Combine commands
11	Processes and system info	ps, top, uname, df, free	Inspect the system
12	Mini challenge	all prior commands	Prove practical skills

How to Use This Lab Guide

- Type each command exactly as shown, then press Enter.
- If a command produces an error, read the message carefully. Linux error messages often tell you what is wrong.
- Do not copy and paste the entire lab at once; practice one command at a time.
- Use the checkboxes to track progress as you complete each task.
- The JSLinux environment is for practice. If it breaks, refresh the browser and start over.

Safety reminder

These exercises are intended for the browser-based training VM only. Do not practice delete or permission-changing commands on production servers unless you fully understand the impact.

Lab 0: Set Up the Browser-Based Linux Training Environment

Goal

Open the JSLinux Fedora 33 RISC-V virtual machine and confirm you can type commands at a Linux shell prompt.

Commands you will practice: browser, Enter, whoami

Step 1: Open the JSLinux Fedora 33 RISC-V training VM:

https://bellard.org/jslinux/vm.html?cpu=riscv64&url=fedora33-riscv.cfg&mem=256

Step 2: Wait for the virtual machine to boot. It may take a minute or two depending on your browser and internet connection.

Step 3: Click inside the terminal window so your keyboard input goes to the Linux console.

Step 4: If you see a login prompt, try logging in using the information displayed by the JSLinux page or console. Many JSLinux images start directly at a shell or provide a simple root login. If you are already at a prompt, continue.

Step 5: Run these first commands:

```
whoami
pwd
date
```

What to observe: whoami prints the current user, pwd prints your current directory, and date prints the system date/time.

- ☐ I opened the JSLinux training environment.
- ☐ I clicked in the console and can type commands.
- ☐ I successfully ran whoami, pwd, and date.

Troubleshooting tips

If the page does not load, refresh the browser tab. If the keyboard does not type in the console, click inside the black terminal area. If the VM becomes unresponsive, refresh the page and wait for it to boot again.

Lab 1: Understand the Shell Prompt and Basic Commands

Goal

Learn what the command prompt represents and run simple commands that display useful information.

Commands you will practice: whoami, pwd, hostname, date, clear, man

1. Run the following commands one at a time:
2. Read the output after each command.
3. Clear the screen when finished.

```
whoami
pwd
hostname
date
clear
```

Try the manual/help system. Press q to exit the manual page.

```
man ls
```

Beginner explanation

The shell is the program that accepts your typed commands. The prompt usually shows that Linux is ready for your next command. Commands are case-sensitive, so LS and ls are not the same.

- ☐ I can identify the prompt.
- ☐ I can run a command and read the output.
- ☐ I know that Linux commands are case-sensitive.
- ☐ I exited the manual page using q.

Lab 2: Navigate the Linux File System

Goal

Move through directories and understand absolute paths, relative paths, the root directory, and the home directory.

Commands you will practice: pwd, ls, cd, /, ~, ..

```
pwd
ls
ls -l
cd /
pwd
ls
cd ~
pwd
cd ..
pwd
cd ~
```

Practice task: Move to /tmp, list the files, then return to your home directory.

```
cd /tmp
pwd
ls
cd ~
pwd
```

Key concepts

/ is the top of the Linux file system. ~ represents your home directory. .. means the parent directory above your current location. . means the current directory.

- ☐ I navigated to / and back to my home directory.
- ☐ I used cd .. to move up one level.
- ☐ I used cd /tmp and returned home using cd ~.

Lab 3: Create a Practice Workspace

Goal

Create directories and files that will be used throughout the rest of the labs.

Commands you will practice: mkdir, touch, echo, ls

```
cd ~
mkdir linux-lab
cd linux-lab
pwd
mkdir notes scripts logs backups
touch notes/day1.txt
echo "Linux lab started" > notes/day1.txt
ls -R
```

What happened: mkdir created directories, touch created an empty file, and echo with > wrote text into a file.

- ☐ I created ~/linux-lab.
- ☐ I created notes, scripts, logs, and backups directories.
- ☐ I created notes/day1.txt.
- ☐ I listed the workspace recursively with ls -R.

Lab 4: View Text Files

Goal

Display file contents and learn which viewer works best for short files versus longer files.

Commands you will practice: cat, less, head, tail

```
cd ~/linux-lab
cat notes/day1.txt
echo "Command line practice" >> notes/day1.txt
echo "Navigation, files, and directories" >> notes/day1.txt
cat notes/day1.txt
head notes/day1.txt
tail notes/day1.txt
less notes/day1.txt
```

In less, use the arrow keys to scroll and press q to quit.

- ☐ I used cat to print a file.
- ☐ I used >> to append more lines.
- ☐ I used head and tail.
- ☐ I opened a file with less and exited with q.

Lab 5: Copy, Rename, Move, and Delete Files

Goal

Practice basic file management safely inside your lab workspace.

Commands you will practice: cp, mv, rm, rmdir

```
cd ~/linux-lab
cp notes/day1.txt notes/day1-copy.txt
ls notes
mv notes/day1-copy.txt backups/day1-backup.txt
ls backups
mkdir temp-delete
touch temp-delete/remove-me.txt
ls temp-delete
rm temp-delete/remove-me.txt
rmdir temp-delete
```

Important caution

rm deletes files. In many Linux environments there is no Recycle Bin. Always check your current directory with pwd and review file names with ls before deleting.

- ☐ I copied a file.
- ☐ I renamed or moved a file with mv.
- ☐ I deleted a test file only inside the lab workspace.
- ☐ I removed an empty directory with rmdir.

Lab 6: Use Command History, Tab Completion, and Wildcards

Goal

Learn productivity shortcuts that make the command line faster and reduce typing mistakes.

Commands you will practice: history, Tab, *, ?

Run these commands, then use the Up Arrow key to repeat earlier commands:

```
cd ~/linux-lab
touch logs/app.log logs/system.log logs/security.log
ls logs
ls logs/*.log
ls logs/s*.log
history | tail
```

Practice Tab completion: type the beginning of a path, then press Tab to complete it.

```
cat notes/da
```

After typing the command above, press Tab. The shell should complete the filename if it is unique.

- ☐ I used the Up Arrow to recall a previous command.
- ☐ I used a wildcard to list multiple .log files.
- ☐ I practiced Tab completion.

Lab 7: Understand File Permissions

Goal

Read permission strings and safely change permissions on a test script.

Commands you will practice: ls -l, chmod

```
cd ~/linux-lab
echo "echo Hello from my first script" > scripts/hello.sh
ls -l scripts/hello.sh
cat scripts/hello.sh
chmod +x scripts/hello.sh
ls -l scripts/hello.sh
./scripts/hello.sh
```

Permission string basics

In ls -l output, r means read, w means write, and x means execute. Permissions are shown for the owner, group, and others. chmod +x adds execute permission.

- ☐ I viewed permissions with ls -l.
- ☐ I added execute permission to hello.sh.
- ☐ I ran the script using ./scripts/hello.sh.

Lab 8: Search and Filter Text

Goal

Find files and search within text files using common command-line tools.

Commands you will practice: grep, find, wc, sort, uniq

```
cd ~/linux-lab
echo "error: disk almost full" >> logs/system.log
echo "info: login successful" >> logs/security.log
echo "error: failed login" >> logs/security.log
grep error logs/*.log
find . -name "*.log"
wc -l logs/*.log
printf "web
db
web
app
db
" > notes/servers.txt
sort notes/servers.txt
sort notes/servers.txt | uniq
```

- ☐ I searched log files with grep.
- ☐ I found files with find.
- ☐ I counted lines with wc -l.
- ☐ I sorted and removed duplicates using sort and uniq.

Lab 9: Use Pipes and Redirection

Goal

Connect commands together and save command output to files.

Commands you will practice: |, >, >>, 2>

```
cd ~/linux-lab
ls -l /etc | head
ls -l /etc | wc -l
grep error logs/*.log > notes/errors.txt
cat notes/errors.txt
echo "Reviewed errors" >> notes/day1.txt
ls /does-not-exist 2> logs/command-errors.log
cat logs/command-errors.log
```

Redirection summary

> writes output to a file and replaces existing content. >> appends output to the end of a file. | sends the output of one command into another command. 2> redirects error messages.

- ☐ I used a pipe to send output to another command.
- ☐ I redirected output to a file.
- ☐ I appended output to an existing file.
- ☐ I captured an error message with 2>.

Lab 10: Inspect Processes and System Information

Goal

Use commands that help you understand what is running and what resources the system has.

Commands you will practice: ps, top, uname, df, free, uptime

```
uname -a
uptime
ps
ps aux | head
df -h
free -m
top
```

In top, press q to quit.

- ☐ I viewed kernel/system information with uname.
- ☐ I viewed uptime.
- ☐ I listed processes with ps.
- ☐ I checked disk and memory usage.
- ☐ I opened top and exited with q.

Lab 11: Use a Text Editor for Simple File Edits

Goal

Make a small text change using a terminal editor.

Commands you will practice: vi, nano if available

Editor note

Some tiny browser-based Linux images may not include nano. vi is usually available. This exercise provides a safe vi workflow.

Create a file and open it in vi:

```
cd ~/linux-lab
vi notes/editor-practice.txt
```

Inside vi, do the following:

4. Press i to enter insert mode.
5. Type: This file was created with vi.
6. Press Esc to leave insert mode.
7. Type :wq and press Enter to save and quit.
8. Confirm the file content with cat.

```
cat notes/editor-practice.txt
```

☐ I opened vi.

☐ I entered insert mode with i.

☐ I saved and quit with :wq.

☐ I verified the file with cat.

Lab 12: Mini Challenge: Build a Simple Server Inventory File

Goal

Apply everything learned by creating, editing, searching, sorting, and saving command output.

Commands you will practice: mkdir, echo, cat, grep, sort, uniq, wc, >, >>

Complete this challenge without skipping steps. If you get stuck, look back at the earlier labs.

```
cd ~/linux-lab
mkdir -p challenge
printf "server01,linux,web
server02,linux,database
server03,windows,file
server04,linux,web
" > challenge/server_inventory.csv
cat challenge/server_inventory.csv
grep linux challenge/server_inventory.csv > challenge/linux_servers.csv
cat challenge/linux_servers.csv
cut -d, -f3 challenge/server_inventory.csv | sort | uniq > challenge/roles.txt
cat challenge/roles.txt
wc -l challenge/server_inventory.csv challenge/linux_servers.csv
```

Challenge questions:

- How many total server records are in server_inventory.csv?
- How many Linux server records were copied to linux_servers.csv?
- Which unique roles are listed in roles.txt?
- Which command was used to split fields by comma?

☐ I created the challenge directory.

☐ I created a CSV-style inventory file.

☐ I filtered Linux records into a new file.

☐ I generated a unique role list.

☐ I answered the challenge questions.

Command Cheat Sheet

Command	What it does	Example
pwd	Print current directory	pwd
ls	List files	ls -l
cd	Change directory	cd /tmp
mkdir	Create directory	mkdir lab
touch	Create empty file	touch file.txt
cat	Print file content	cat file.txt
less	Page through file	less file.txt
cp	Copy file	cp a.txt b.txt
mv	Move or rename	mv old.txt new.txt
rm	Delete file	rm test.txt
grep	Search text	grep error app.log
find	Find files	find . -name "*.log"
chmod	Change permissions	chmod +x script.sh
ps	Show processes	ps aux | head
df	Disk usage	df -h
free	Memory usage	free -m
history	Show command history	history | tail

Next steps after this lab

Repeat the labs without looking at the explanations. Then try creating your own directory structure, notes files, and script. After that, move to Linux users/groups, services, package management, SSH, logs, and basic shell scripting.

Completion Sign-Off

Learner name: _____

Date completed: _____

Notes or questions:

_____

_____

_____


TOVARTECH
LEARN • BUILD • MASTER
LINUX
SQL SERVER
WINDOWS SERVER
AUTOMATION
REAL SKILLS. REAL LABS. REAL SUCCESS.