Linux

Move around, manage files, read and search text, set permissions.

01 Navigate & list

```
pwd             # current directory
cd /var/log     # go to a path
cd ~            # home (or just cd)
cd ..           # up one level
cd -            # previous directory
ls -la          # long list + hidden files
ls -lh          # human-readable sizes
ls -lt          # newest first (-lS: largest)
```

Tab completes paths. ls -R lists subdirectories too.

02 Create & delete

```
mkdir -p src/components  # + parents
mkdir -m 755 public      # with a mode
touch notes.txt          # empty file
rm old.txt               # delete a file
rm -i old.txt            # ask first
rm -r old_dir/           # a directory
rm -rf build/            # never prompts
rmdir empty_dir/         # empty only
```

rm -rf never asks, so check the path first.

03 Copy & move

```
cp file.txt copy.txt
cp a.txt b.txt /target/dir/  # many
cp -r src_dir/ dest_dir/     # folder
cp -i src.txt dest.txt       # ask first
cp -p file.txt backup.txt    # keep perms
mv old.txt new.txt           # rename
mv a.txt b.txt /target/dir/  # move
mv -i src.txt dest.txt       # ask first
```

cp needs -r for directories; mv does not.

04 View files

```
cat file.txt
cat -n file.txt          # line numbers
cat a.txt b.txt          # join files
head -n 20 file.txt      # first 20 lines
tail -n 30 file.txt      # last 30 lines
tail -f /var/log/syslog  # follow a log
less big.log             # page through
```

head and tail show 10 lines unless you pass -n.

05 Keys: less, nano & vim

Space b q	less: next page, back, quit
/text n N	less: search, next, previous
Ctrl+O Ctrl+X	nano: save, exit
i Esc	vim: insert mode, normal mode
:wq :q!	vim: save and quit, discard
dd yy p	vim: cut line, copy line, paste

Run vimtutor for an interactive vim lesson.

06 Find files & commands

```
find . -name "*.js"      # by name
find . -iname "readme*" # any case
find . -type d           # dirs (f: files)
find . -size +100M       # over 100 MB
find . -name "*.log" -delete
find . -type f -exec chmod 644 {} \;
locate nginx.conf        # fast, indexed
which -a python3         # matches in PATH
```

-exec runs a command on each match; {} is the path. locate misses new files until sudo updatedb runs.

07 Search text: grep

```
grep "error" app.log
grep -rn "TODO" src/   # recursive, numbered
ps aux | grep nginx    # filter output
```

-i	Ignore case
-r	Search a whole directory
-n	Show line numbers
-c	Count matching lines
-v	Lines that do not match

Flags combine: grep -rin "todo" src/

08 Permissions: chmod

```
chmod 755 deploy.sh    # rwxr-xr-x
chmod 644 index.html   # rw-r--r--
chmod 600 private.key  # rw-------
chmod u+x script.sh    # user: +execute
chmod g-w file.txt     # group: -write
chmod o+r file.txt     # others: +read
chmod -R 755 public/   # recursive
```

Digits are user, group, others: r=4, w=2, x=1.

09 Ownership: chown & chgrp

```
sudo chown alice file.txt
sudo chown alice:devs file.txt
sudo chown :devs file.txt    # group only
sudo chown -R alice:devs app/
chgrp devs file.txt
groups                       # your groups
```

You need sudo to change files you do not own.

Linux

Manage processes, check system resources, work remotely, archive files.

10 Processes: ps & top

```
ps aux                        # every process
ps -u alice                   # one user
ps auxf                       # as a tree
ps aux | grep nginx           # find a PID
ps aux --sort=-%mem | head    # by memory
top                           # live monitor
```

P M	top: sort by CPU, memory
k	top: kill a PID
q h	top: quit, help

The PID is the second column of ps aux. top refreshes CPU and memory use every few seconds.

11 Kill processes

```
kill 1234          # SIGTERM: clean exit
kill -9 1234       # SIGKILL: force it
killall node       # every "node" process
pkill node         # by name
kill -l            # list signals
```

Try plain kill first so the process can clean up; -9 gives it no chance. killall hits every process with that name.

12 Background jobs

```
./build.sh &          # run in background
jobs                  # list jobs and IDs
fg %1                 # bring job 1 back
bg                    # resume in background
nohup ./server.sh &   # survive logout
```

Ctrl+Z suspends the running command; bg then resumes it in the background.

13 System info

```
uname -a              # kernel, host, arch
uname -r              # kernel version
cat /etc/os-release   # distribution
lsb_release -a        # distro and release
hostname
uptime                # running time, load
whoami                # current user
id                    # uid, gid, groups
who                   # logged-in users
w                     # and what they run
```

uptime shows load averages for the last 1, 5 and 15 minutes.

14 Disk & memory

```
df -h                    # free disk space
du -sh node_modules/     # folder total
du -h --max-depth=1      # one level deep
du -ah | sort -rh | head -20
free -h                  # RAM and swap
cat /proc/meminfo        # full detail
```

du -ah | sort -rh puts the biggest files first. In free, read the available column for usable memory.

15 Network checks

```
ping -c 4 google.com   # 4 pings, then stop
ping -i 2 google.com   # every 2 seconds
ip a                   # interfaces and IPs
ss -tuln               # listening ports
ss -ant                # all TCP connections
netstat -tuln          # older equivalent
```

Without -c, ping runs until Ctrl+C. ss is the faster, modern replacement for netstat.

16 SSH & scp

```
ssh user@host
ssh -p 2222 user@host            # port
ssh -i ~/.ssh/key.pem user@host # key
ssh user@host 'ls -la'           # run it
scp file.txt user@host:/path/    # upload
scp user@host:/path/file.txt ./ # download
scp -r dir/ user@host:/path/     # folder
```

scp takes the port as uppercase -P, not -p.

17 curl & wget

```
curl https://api.example.com
curl -I <url>         # headers only
curl -L <url>         # follow redirects
curl -O https://example.com/a.zip
curl -X POST -d "key=value" <url>
wget https://example.com/a.zip
wget -c <url>         # resume
```

-O keeps the remote name; -o page.html sets one.

18 Archives: tar, gzip, zip

```
tar -czvf site.tar.gz site/     # create
tar -xzvf site.tar.gz           # extract
tar -xzvf site.tar.gz -C /opt/  # to dir
tar -tvf backup.tar             # list
gzip -k file.txt                # keep orig
gunzip file.txt.gz              # gzip -d
zip -r site.zip site/
unzip site.zip -d /dest/
unzip -l site.zip               # list
```

c create, x extract, t list, z gzip, j bzip2, v verbose, f file. gzip replaces the original unless you add -k.