devopscycle

Build. Ship. Run. Repeat.

THE ULTIMATE LINUX BASICS CHEAT SHEET

Get Help and Check Your Location

Bash/GNU Linux: replace <placeholders>; omit $. Quote paths containing spaces. Most successful changes produce no output.

```
# --help lists usage; no universal short equivalent
$ <command> --help

# man (manual) opens a command's documentation
# use arrow keys to scroll; q quits
$ man <command>

# type identifies an alias, builtin, or executable
$ type <command>

# pwd (print working directory) shows where you are
$ pwd

# ls lists entries; --all (-a) includes hidden ones
# --format=long (-l) shows details
# --human-readable (-h) shows sizes in K/M/G
$ ls --format=long --all --human-readable
```

Navigate and Find Files

Paths: `/` is root, `~` is home, `.` is here, `..` is the parent. find searches inside a directory and its subdirectories.

```
# cd (change directory) moves to another directory
$ cd <directory>

# - means the previous directory; it is not a flag
$ cd -

# no argument returns to your home directory
$ cd

# -type f: regular files; -name: match a name pattern
# find uses single-dash words, not short/long pairs
# quote the pattern; -iname ignores case
$ find <directory> -type f -name '<pattern>'

# -v shows how the shell resolves a command name
# this Bash builtin has no long option for -v
$ command -v <command>
```

Create, Copy, Move, and Link

cp copies; mv moves or renames; ln creates links. Add --interactive to cp and mv to ask before overwriting. > replaces contents; >> appends.

```
# mkdir (make directory) creates a directory
# --parents (-p) also creates missing parents
$ mkdir --parents <directory>

# touch creates a file or updates its timestamp
$ touch <file>

# printf: %s inserts text; \n adds a newline
# > writes the result to a file, overwriting contents
$ printf '%s\n' '<text>' > <file>

# >> appends text without replacing existing contents
$ printf '%s\n' '<text>' >> <file>

# copy one file to another destination
$ cp <source-file> <destination-file>

# --archive (-a) copies trees, preserving attributes
$ cp --archive <source-dir> <destination-dir>

# move or rename a file or directory
$ mv <source> <destination>

# --symbolic (-s) makes a link pointing to the target
# an absolute target works from any link location
$ ln --symbolic <target> <link>
```

Read and Search Text

grep finds matching lines using regular expressions; --fixed-strings searches literal text. No matches means no output. Ctrl+C stops a command.

```
# cat (concatenate) prints the contents of a short file
$ cat <file>

# less pages through a long file; / searches, q quits
$ less <file>

# head shows the start; --lines (-n) sets the count
$ head --lines <count> <file>

# tail shows the end; --lines (-n) sets the count
$ tail --lines <count> <file>

# --follow (-f) shows new lines as the file grows
# useful for watching logs; Ctrl+C stops
$ tail --follow <file>

# --ignore-case (-i) matches either letter case
# --line-number (-n) shows each match's line number
$ grep --ignore-case --line-number '<pattern>' <file>

# --invert-match (-v) prints nonmatching lines
$ grep --invert-match '<pattern>' <file>
```

Manage Permissions and Ownership

Permissions apply to owner / group / others. r = read (4), w = write (2), x = execute (1). Directories need x to traverse, wx to add/remove entries.

```
# --format=long (-l): permissions, owner, size, date
# - = file, d = directory; then three rwx groups
$ ls --format=long <file>
## ⇒ output
### -rw-r----- 1 <user> <group> <size> <date> <file>

# chmod (change mode) changes permissions
# u = owner; +x adds execute permission
$ chmod u+x <file>

# 640: owner gets rw-; group gets r--; others get ---
$ chmod 640 <file>

# 755: owner gets rwx; group and others get r-x
$ chmod 755 <directory>

# chown (change owner) sets owner and group
# sudo runs the command as an administrator
$ sudo chown <user>:<group> <file>

# id (identity) shows your user ID and groups
$ id

# +t (sticky bit) restricts deletion in shared folders
# only entry/directory owners or admins may delete
$ chmod +t <directory>
```

Remove Files and Directories Safely

Deletion bypasses the trash. Check the exact target first. --interactive asks for confirmation; answer n to keep a file.

```
# inspect all entries: --format=long (-l), --all (-a)
$ ls --format=long --all <directory>

# rmdir only removes empty directories
$ rmdir <empty-directory>

# rm deletes a file; --interactive (-i) asks first
$ rm --interactive <file>

# --recursive (-r or -R) removes a directory tree
# --interactive (-i) asks each time; check each path
$ rm --recursive --interactive <directory>
```

Blog: https://devopscycle.com/blog/the-ultimate-linux-cheat-sheet/
Linux: https://www.kernel.org/doc/man-pages/
Consulting: https://devopscycle.com/