File commands (ls, cp, mv, ln) Everyday file handling

START HERE

`ls -lah`
List with sizes and hidden files

`ln -s <target> <link>`
Make a symlink

`cp -r <src> <dst>`
Copy a directory

`rm -ri <dir>`
Delete, asking first

`mv <src> <dst>`
Move or rename

`touch <file>`
Create empty file

`mkdir -p <a/b/c>`
Create nested directories

`stat <file>`
Size, mode, timestamps

CONCEPTS

No trash can: rm and overwriting cp/mv are permanent. Add -i or -n while learning, and expand globs with echo first.

Directory entry name
Inode owner, mode, times
Data blocks contents

A directory entry names an inode; data lives until the last name goes

A hard link is another name for the same inode (same filesystem, files only). A symlink is a small file holding a path and can dangle.

mv within a filesystem is a rename: instant and keeps inode and timestamps. Across filesystems it copies, then deletes.

LISTING (LS)

`ls`
List the current directory

`ls -l`
Long format: mode, owner, size, date

`ls -a`
Include hidden dotfiles

`ls -lh`
Human-readable sizes

`ls -lt`
Newest first

`ls -ltr`
Oldest first, newest at the bottom

`ls -lS`
Largest first

`ls -d */`
List only directories
Without -d, ls <dir> lists the directory contents instead of the entry itself.

`ls -R`
Recurse into subdirectories

`ls -1`
One name per line

`ls -i`
Show inode numbers
Same inode number means the same file (hard links).

`ls -lA`
All entries except . and .. dirs

`ls -F`
Mark types: / dir, @ link, * exec

`ls -ld <path>`
Show the directory itself, not its contents

COPYING (CP)

`cp <src> <dst>`
Copy a file

COPYING (CP) (cont.)

`cp -r <dir> <dst>`
Copy a directory recursively

`cp -a <src> <dst>`
Archive: recurse, keep links, mode, times
Also keeps ownership when run as root. The right choice for backups.

`cp -i <src> <dst>`
Ask before overwriting

`cp -n <src> <dst>`
Never overwrite existing files

`cp -u <src> <dst>`
Only copy newer or missing files

`cp -v <src> <dst>`
Show each copy

`cp -p <src> <dst>`
Keep mode, owner and times

`cp -t <dir> <files>`
Target dir first, files after (xargs-friendly)

`cp --backup=numbered <src> <dst>`
Keep numbered backups of overwritten files

`cp --reflink=auto <src> <dst>`
Copy-on-write clone where supported
Instant on btrfs and XFS. auto falls back to a normal copy.

`cp <src>/. <dst>`
Copy directory contents, including dotfiles

MOVING (MV)

`mv <src> <dst>`
Move or rename

`mv <a> <b> <dir>`
Move several items into a directory

`mv -i <src> <dst>`
Ask before overwriting

`mv -n <src> <dst>`
Never overwrite

REMOVING (RM, RMDIR)

`rm <file>`
Delete a file

`rm -i <file>`
Confirm each deletion

`rm -r <dir>`
Delete a directory tree

`rm -rf <dir>`
Recursive, no prompts, ignore missing
Check the path with ls first. An empty variable or a stray space in rm -rf $d/ can erase far more than intended.

`rm -I <files>`
One prompt for 4+ files or recursion

REMOVING (RM, RMDIR) (cont.)

`rm -- -<name>`
Delete a file whose name starts with -

`rmdir <dir>`
Remove an empty directory only

`rm --one-file-system -r <dir>`
Do not cross mount points

Prefer rm -I or a dry run (ls <glob>) before any recursive delete. Use rmdir when the directory should already be empty.

LINKS (LN, READLINK)

`ln -s <target> <link>`
Create a symbolic link

`ln <file> <link>`
Create a hard link

`ln -sf <target> <link>`
Replace an existing link

`ln -sfn <target> <link>`
Replace a link to a directory in place
Without -n, an existing link to a directory is followed and the new link lands inside it.

`ln -sr <target> <link>`
Make the link path relative

`ls -l <link>`
Show where a link points

`readlink <link>`
Print the link target

`readlink -f <path>`
Resolve fully to an absolute path

`find -L <dir> -type l`
Find broken symlinks

A relative symlink target is resolved from the link's directory, not from where you ran ln.

DIRECTORIES AND FILES

`mkdir <dir>`
Create a directory

`mkdir -p <a/b/c>`
Create parents, no error if present

`mkdir -p src/{app,lib}`
Create several siblings with brace expansion

`touch <file>`
Create empty file or update its time

`touch -d "2 days ago" <file>`
Set a specific modification time

DIRECTORIES AND FILES (cont.)

`touch -r <ref> <file>`
Copy the timestamp of another file

INSPECTING PATHS

`stat <file>`
Size, inode, mode, owner, times

`stat -c '%a %U %s %n' <file>`
Custom format: octal mode, owner, size, name

`stat -c %y <file>`
Last modification time

`realpath <path>`
Absolute path, symlinks resolved

`realpath --relative-to=<dir> <path>`
Path relative to a directory

`basename <path>`
Last path component

`basename -s .txt <path>`
Strip a suffix as well

`dirname <path>`
Everything but the last component

TEMPORARY FILES (MKTEMP)

`mktemp`
Create a unique temp file, print its path

`mktemp -d`
Create a unique temp directory

`mktemp /tmp/<name>.XXXXXX`
Use your own template
Needs at least three trailing X characters.

Always use mktemp rather than a fixed name like /tmp/out: it avoids clashes and symlink attacks. Remove it with a trap.

RECIPES

Safe scratch directory

```
tmp=$(mktemp -d) || exit 1
trap 'rm -rf "$tmp"' EXIT
cp -a <src>/. "$tmp"
```

Atomic symlink switch (releases)

```
ln -sfn releases/<new>
current.tmp
mv -T current.tmp current
# current flips in one rename
```

Back up a file before editing

```
cp -a --backup=numbered <file>
<file>.bak
ls -l <file>*
```

Preview, then delete

```
ls -d <glob>
rm -rI <glob>
```