

2 - Naviguer et gérer les fichiers

Se déplacer dans l'arborescence et manipuler fichiers, dossiers et liens sans risque.

- [2.1 - Se repérer avec pwd, ls et cd](#)
- [2.2 - Créer, copier, déplacer et renommer](#)
- [2.3 - Supprimer sans se tromper](#)
- [2.4 - Types de fichiers et liens](#)

2.1 - Se repérer avec pwd, ls et cd

Le shell exécute les commandes dans un **dossier de travail courant**. Bien savoir où l'on se trouve évite de modifier le mauvais fichier.

Afficher le dossier courant

```
pwd
pwd -P      # résoudre les liens symboliques dans le chemin
```

Lister le contenu

```
ls
ls -l      # détails : droits, propriétaire, taille et date
ls -a      # inclure les noms commençant par un point
ls -lah    # détails, fichiers cachés et tailles lisibles
ls -lt     # trier par date de modification
ls -ltr    # plus ancien en premier
ls -ld /etc # afficher le dossier lui-même, pas son contenu
```

Dans une sortie `ls -l`, le premier caractère indique le type : `d` pour un dossier, `-` pour un fichier ordinaire et `l` pour un lien symbolique.

Changer de dossier

```
cd /var/log      # chemin absolu
cd ../lib        # dossier parent, puis lib
cd ~             # dossier personnel
cd               # identique à cd ~
```

```
cd - # revenir au dossier précédent
```

Les raccourcis essentiels :

Chemin	Signification
<code>.</code>	dossier courant
<code>..</code>	dossier parent
<code>~</code>	dossier personnel
<code>-</code> avec <code>cd</code>	dossier visité précédemment

Afficher une arborescence

Si la commande `tree` est installée :

```
tree
tree -L 2
tree -a -L 2 /etc
```

Sans `tree` :

```
find . -maxdepth 2 -print
```

Noms cachés et complétion

Sous Linux, un nom commençant par `.` est caché par convention : `.bashrc`, `.config`, `.ssh`, etc. Il n'est pas secret et reste visible avec `ls -a`.

Utilisez `Tab` pour compléter les chemins. Deux pressions peuvent afficher les possibilités. Cette méthode réduit fortement les erreurs de frappe.

Exemple de navigation contrôlée

```
cd /var/log
pwd
ls -lah
```

```
cd -
```

Avant une commande sensible, associer `pwd` et `ls` constitue un bon réflexe :

```
pwd  
ls -la -- .
```

Le séparateur `--` indique la fin des options. Il protège notamment contre un fichier dont le nom commence par `-`.

2.2 - Créer, copier, déplacer et renommer

Créer des dossiers et des fichiers

```
mkdir projet
mkdir -p projet/{docs,scripts,sauvegardes}
touch notes.txt
touch "compte rendu.txt"
```

`mkdir -p` crée aussi les dossiers parents manquants et ne signale pas d'erreur si le dossier existe déjà. `touch` crée un fichier vide ou actualise sa date de modification.

Copier

```
cp source.txt copie.txt
cp source.txt sauvegardes/
cp -i source.txt copie.txt      # demander avant d'écraser
cp -a dossier/ sauvegarde/      # copie récursive en préservant les attributs
cp -av dossier/ sauvegarde/     # mode détaillé
```

Attention à la barre finale :

```
cp -a dossier sauvegarde/      # crée sauvegarde/dossier si sauvegarde existe
cp -a dossier/. sauvegarde/    # copie le contenu de dossier dans sauvegarde
```

Déplacer ou renommer

La même commande réalise les deux opérations :

```
mv ancien.txt nouveau.txt
mv rapport.pdf Documents/
mv -i fichier.txt destination/
mv -n fichier.txt destination/    # ne jamais écraser
```

Créer rapidement du contenu

```
printf '%s\n' "Première ligne" "Deuxième ligne" > notes.txt
echo "Ligne ajoutée" >> notes.txt
```

> remplace le contenu existant. >> ajoute à la fin.

Comparer avant de remplacer

```
cmp fichier1.bin fichier2.bin
diff -u ancienne.conf nouvelle.conf
```

Pour synchroniser des dossiers, `rsync` offre un mode de simulation très utile :

```
rsync -avhn source/ destination/ # simulation
rsync -avh source/ destination/  # exécution
```

Gérer les noms particuliers

Protéger les espaces :

```
cp -- "rapport final.txt" "rapport final - copie.txt"
```

Traiter un nom commençant par un tiret :

```
mv -- -ancien-fichier nouveau-fichier
```

Vérifications recommandées

```
file sauvegarde.tar.gz      # identifier le type réel
stat fichier.txt            # métadonnées détaillées
ls -lah destination/
```

Pour une donnée importante, une copie locale n'est pas forcément une sauvegarde. Une vraie sauvegarde doit idéalement être stockée sur un autre support et testée par une restauration.

2.3 - Supprimer sans se tromper

Sous Linux, `rm` ne place généralement pas les éléments dans une corbeille. Une suppression en ligne de commande peut donc être immédiate et difficile à récupérer.

Supprimer un fichier ou un dossier vide

```
rm fichier.tmp
rm -i fichier.tmp      # demander confirmation
rmdir dossier-vide
```

Supprimer plusieurs fichiers précis :

```
rm -- fichier1.tmp fichier2.tmp
```

Supprimer un dossier et son contenu

```
rm -r ancien-dossier/
rm -ri ancien-dossier/ # confirmation interactive
```

L'option `-f` force la suppression et réduit les protections. La combinaison `rm -rf` doit rester exceptionnelle et ne doit être utilisée qu'après vérification exacte de la cible.

Méthode sûre avec un motif

Avant de supprimer tous les fichiers `.tmp`, afficher ce que le shell sélectionne :

```
printf '%s\n' -- *.tmp
```

Puis supprimer seulement si la liste est correcte :

```
rm -i -- *.tmp
```

Avec `find`, commencer par `-print` :

```
find /var/tmp/mon-application -type f -name '*.tmp' -print
```

Après contrôle :

```
find /var/tmp/mon-application -type f -name '*.tmp' -delete
```

L'ordre des critères de `find` compte : `-delete` ne doit être ajouté qu'à la fin, après un test réussi avec `-print`.

Préférer une opération récupérable

Si l'environnement graphique fournit `gio` :

```
gio trash fichier.txt  
gio trash --list
```

Autre méthode : déplacer d'abord l'élément vers un dossier de quarantaine.

```
mkdir -p ~/A_supprimer  
mv -i ancien-dossier ~/A_supprimer/
```

Erreurs à éviter

- Ne jamais exécuter une suppression récursive sur `/`, `/home`, `~` ou une variable non vérifiée.
- Toujours mettre les variables entre guillemets : `rm -- "$fichier"`.
- Ne pas supposer qu'une variable est définie.
- Vérifier `pwd`, puis la cible avec `ls -ld -- chemin`.
- Ne pas précéder `rm` de `sudo` si les droits normaux suffisent.

Exemple de contrôle d'une variable :

```
cible="/var/tmp/mon-application"  
test -n "$cible" && test "$cible" != "/" && ls -ld -- "$cible"
```

Cette vérification n'autorise pas automatiquement la suppression ; elle sert seulement à examiner la cible.

2.4 - Types de fichiers et liens

Linux représente de nombreuses ressources sous forme de fichiers : documents, dossiers, périphériques, sockets et liens.

Identifier un type de fichier

```
file document.pdf
file image-sans-extension
stat document.pdf
ls -l document.pdf
```

Le début de `ls -l` indique le type :

Caractère	Type
-	fichier ordinaire
d	dossier
l	lien symbolique
b	périphérique bloc, par exemple un disque
c	périphérique caractère
s	socket
p	tube nommé

Lien symbolique

Un lien symbolique contient un chemin vers une autre ressource. Il ressemble à un raccourci.

```
ln -s /srv/app/config.yml ~/config-app.yml
ls -l ~/config-app.yml
readlink ~/config-app.yml
```

```
readlink -f ~/config-app.yml
```

Si la cible est déplacée ou supprimée, le lien devient cassé. Rechercher les liens cassés sous le dossier actuel :

```
find . -xtype l -print
```

Lien physique

Un lien physique est un second nom vers les mêmes données sur le même système de fichiers.

```
ln rapport.txt rapport-alias.txt
ls -li rapport.txt rapport-alias.txt
```

Les deux noms affichent le même numéro d'inode. Modifier l'un modifie le contenu visible par l'autre. Supprimer un nom ne détruit les données que lorsqu'aucun lien physique ne les référence plus et qu'aucun processus ne les maintient ouvertes.

Les liens physiques ne traversent généralement pas les systèmes de fichiers et ne s'appliquent normalement pas aux dossiers.

Extension et contenu réel

Linux ne dépend pas uniquement de l'extension :

```
file sauvegarde
file --mime-type sauvegarde
```

Un fichier nommé `.jpg` peut ne pas être une image. `file` examine sa signature, sans garantir pour autant qu'il est sûr.

Informations utiles avec stat

```
stat -c 'Nom=%n Taille=%s octets Droits=%A Propriétaire=%U Groupe=%G' fichier
```

Sur BSD ou macOS, les options de `stat` diffèrent. Les exemples ci-dessus visent les outils GNU courants sous Linux.