# I - Manipulation de fichiers

## Manuel des commandes (man)

La commande man permet d’afficher le manuel de la commande demandée.

## Création d'un fichier (touch)

La commande touch permet de créer un nouveau fichier vide.

Syntaxe : touch *fichier*

Appliquée à un fichier déjà existant, elle modifie son heure de dernier accès.

Vous pouvez créer plusieurs fichiers en une seule ligne : touch fichier1 fichier2 fichier3

## Copie un fichier (cp)

La commande cp permet de copier un fichier

Syntaxe : cp [option] source cible

Exemple : cp toto /home/ => copie le fichier toto de l’emplacement où je suis, dans le dossier home.

Cp toto /home/tata => comme ci-dessus mais renomme le fichier en tata

## Déplace un fichier (mv)

La commande move mv permet de placer un fichier.

Syntaxe : mv [option] source cible

Exemple : mv toto /home/ => déplace le fichier toto de l’emplacement où je suis, dans le dossier home. Il n’existe donc plus là où je suis.

mv toto /home/tata => comme ci-dessus mais renomme le fichier en tata

## Afficher le contenu d'un fichier (cat)

La commande ***cat*** permet d'afficher sur votre shell le contenu d'un fichier.

Options utiles :  
- n (Number) numérote les lignes  
- b (number nonBlank) numérote que les lignes non vides

Syntaxe : *cat fichier*

Affichage inverse (tac)

La commande tac est homologue à cat mais affiche le contenu d'un fichier en partant de la dernière ligne vers la première.

Afficher écran par écran (more et less)

Les commandes more et less permettent d'afficher page par page des fichiers volumineux sur le shell.

## Afficher l'entête (head)

La commande head permet de n'afficher que les permières lignes d'un fichier (10 par défaut). En voici les options :  
- cN affiche les N premiers octets  
- nN affiche les N premières lignes  
- q n'affiche pas le nom du fichier  
- v affiche le nom du fichier avant d'en afficher l'entête.

Syntaxe : head *fichier*

Exemple : head *lettre*

Dans cet exemple, il s'affiche les 10 premières lignes du fichier *lettre*, si ce dernier en contient moins de 10, il sera affiché dans sa totalité.

Exemple : head -n5 *lettre*

Affichage des 5 premières lignes.

Exemple : head -vc20 *lettre*

Affichage des 20 permiers caractères après le nom du fichier.

## Afficher la fin (tail)

La commande tail permet de n'afficher que les dernières lignes d'un fichier (10 par défaut). Elle est homologue à head et possède les mêmes attributs.

Syntaxe : tail *fichier*

Exemple : tail *lettre*

Dans cet exemple, il s'affiche les 10 dernières lignes du fichier *lettre*, si ce dernier en contient moins de 10, il sera affiché dans sa totalité.

Exemple : tail -n5 *lettr*

Affichage des 5 dernières lignes.

Exemple : tail -vc20 *lettre*

Affichage des 20 derniers caractères après le nom du fichier.

## Numérotation des lignes (nl)

La commande nl permet l'affichage du contenu d'un fichier et en numérote les lignes. En voici les options :  
- bt numérote les lignes non-vides (par défaut)  
- ba numérote toutes les lignes

## Différences (diff)

La commande diff permet de comparer le contenu de deux fichier pour en connaître les différences. Ceci est très pratique pour savoir si deux fichiers ont le même contenu. En voici quelques options :   
- b ignore les différences dû à des espaces blancs  
- B ignore les différences dû à des lignes blanches  
- i ignore les différences minuscules/MAJUSCULES  
- q indiquer seulement si les fichiers sont différents et ne pas afficher les différences elles-mêmes  
- s indiquer lorsque deux fichiers sont identiques  
- r comparaison récursive des fichiers d'un répertoire, sous répertoires...

Syntaxe : diff [options] *fichier\_1* *fichier\_2*

## Statistiques Word Count (wc)

La commande wc permet de compter le nombre de caractères, de mots et de lignes d'un fichier. Ses options sont :  
- l (Lignes) compte le nombre de lignes  
- w (Words) compte le nombre de mots  
- c (Chars) compte le nombre de caractères  
- L (Length max ligne) affiche la longueur de la ligne la plus longue

Syntaxe : wc *fichier*

Sans options, wc renvoie automatiquement le nombre de de lignes (l), de mots (w) et de caractères (c).

## Recherche de fichiers (find)

La commande find permet de faire une recherche de fichier.

Syntaxe : find *répertoire* *critères* [-print]

Exemple : find . -name "\*.c" -print

Recherche depuis le répertoire courant ‘ . ‘ et affiche le résultat de la recherche ‘ print ‘. Le critère de recherche porte sur le nom ‘ name ‘ et doit comprendre les caractères suivants : "\*.c" c'est-à-dire tous les fichiers contenant dans leur nom ‘ .c ‘.

Exemple : find . -mtime +7 -print

On recherche les fichiers dont la date de dernière modification (mtime : Modify Time) remonte à plus de 7 jours (+7).

Liste des critères de recherche

Le tableau suivant récapitule les principales options de la commande find.

|  |  |
| --- | --- |
| Option | Description |
| print | affiche le résultat |
| name motif | nom du fichier |
| iname motif | idem mais sans tenir compte de la casse |
| mtime +n|-n|n | nombre de jours depuis la date de dernière modification |
| ctime +n|-n|n | nombre de jours depuis la date de création |
| atime +n|-n|n | nombre de jours depuis la date de dernier accès |
| mmin +n|-n|n | nombre de minutes depuis la date de dernière modification |
| cmim +n|-n|n | nombre de minutes depuis la date de création |
| amin +n|-n|n | nombre de minutes depuis la date de dernier accès |
| type type | type du fichier b (fichier spécial en mode bloc), c (fichier special en mode caractère), d (répertoire), p (tube nommé), f (fichier normal), l (lien symbolique), s (socket) |
| uid n | le fichier porte le numéro de propriétaire spécifié |
| gid n | idem pour numéro de groupe |
| size n | le fichier est de taille spécifiée dont il faut préciser l'unité : b (bloc de 512 octets, par défaut), c (octet), k (Ko) |
| used n | nombre de jours entre la création et le dernier accès au fichier |
| perm +n|-n|n | valeur numérique des droits d'accès au fichier |
| user nom | le propriétaire du fichier a pour nom celui passé en paramètre |
| group nom | idem pour le groupe |
| inum n | numéro d'inode du fichier |
| links n | nombre de liens du fichier |
| newer fichier | le fichier est plus récent que celui passé en paramètre |
| anewer fichier | on a accédé au fichier plus récemment qu'on a modifié celui passé en paramètre |
| cnewer fichier | on a accédé au fichier plus récemment qu'on a créé celui passé en paramètre |
| nouser | aucun utilisateur dans le système ne correspond au UID du fichier |
| nogroup | aucun groupe ne correspond au GID du fichier |
| empty | le fichier est vide et il est soit un fichier normal soit un répertoire |
| false | renvoie toujours faux |
| true | renvoie toujours vrai |

Les options doivent êtres précédées d'un trait d'union (-) et leur paramètre éventuel séparé par un espace.

Les paramètres numériques n peuvent être passé en valeur absolue (par exemple 7) ou bien en valeur relative (+2 ou -20).

## Recherche de contenu dans un fichier (grep)

grep [mot cherché] [fichier où lancer la recherche]

grep [mot cherché] -r [dossier où lancer la recherche]

Le mot cherché doit être écrit entre guillemet s’il comprend des espaces

## Découpage (split)

La commande split permet de découper un fichier en plusieurs plus petits. Ses options sont :  
- b *n* (Bytes) découpage par blocs de *n* octets  
ou  
- l *n* (Lignes) découpage par blocs de *n* lignes

Syntaxe : split *fichier*

Exemple : split -b 135000 *mer.mpeg* => Découpage du fichier mer.mpeg qui est une vidéo volumineuse en fichiers de 1.35 Mo afin de la sauvegarder sur disquettes (de capacité maximum de 1.44 Mo).

Par défaut, les fichiers issus de la découpe auront un nom ayant pour préfixe *x* et pour suffixe une suite de lettre du type *aa*, *ab*,*ac*... créés dans l'ordre lexicographique naturel (descendant de 'a' vers 'z').   
Pour changer le préfixe, il suffit de le spécifier en fin de commande.

Exemple : wc -l 100 */var/log/httpd/access.log* access.log.

Ici on découpe le fichier de log du serveur HTTP en plus petits fichiers de 100 lignes chacun. Dont le nom sera du type *access.log.aaa*, *access.log.aab*...

Note : pour découper des fichiers texte brut, faire la découpe en nombre de lignes. Tandis que pour découper les fichiers binaires, utiliser la découpe en nombre d'octets.

# II - Manipulation de dossiers

## Création d'un dossier (mkdir)

La commande mkdir permet de créer un nouveau dossier.

Syntaxe : mkdir dossier

Vous pouvez créer plusieurs dossiers à même temps : mkdir dossier1 dossier2 dossier3

- p créer le dossier mère et les dossiers filles

Exemple : mkdir -p cours/linux/reseau => créer le dossier cours, dans lequel il y aura le dossier fille linux, qui aura lui-même un dossier fille réseau.

## Copier un dossier (cp)

La commande cp permet aussi de copier un dossier

Syntaxe : cp [option] source cible

Exemple : cp -R cours courstssi => copie le dossier cours et tous ses sous dossiers et fichiers contenu à l’emplacement où je suis, et le renomme en courtssi.

## Déplace un fichier (mv)

La commande move mv permet de placer un dossier.

Syntaxe : mv [option] source cible

Exemple : mv cours /home/cours => déplace le dossier cours de l’emplacement où je suis, dans le dossier home. Il n’existe donc plus là où je suis.

mv cours /home/courstssi => comme ci-dessus mais renomme le dossier principal en courstssi.

## Supprimer un dossier ou un fichier rm / rmdir

La commande remove rm permet de supprimer un fichier ou un dossier, ou même plusieurs.

Syntaxe : rm [option] cible

-r recursif : supprime dossier, sous-dossiers et fichiers

-f force la suppression

- R vide le dossier de tout son contenu, sans le supprimer