

RÉSUMÉ DES PRINCIPALES COMMANDES LINUX

Guillaume Santini

guillaume.santini@iutv.univ-paris13.fr
IUT de Villetaneuse

13 février 2012

PLAN

1 ARBORESCENCE

- pwd
- ls
- cd
- mkdir
- rm
- cp
- mv
- find
- ln
- alias

2 PROCESSUS

3 ARCHIVAGE

4 MANIPULATION DE TEXTE

5 SYSTÈME DE FICHER

pwd

SYNTAXE

pwd

DESCRIPTION

- Affiche le nom du répertoire courant.

EXEMPLE D'UTILISATION:

```

/..... Répertoire Racine
├── home/
│   ├── chez_moi/..... Répertoire Courant
│   └── Etoiles/

```

```
[ login@localhost ~ ] pwd
/home/chez_moi
```

```

/..... Répertoire Racine
├── home/
│   ├── chez_moi/..... Répertoire Personnel
│   └── Etoiles/.... Répertoire Courant

```

```
[ login@localhost ~/Etoiles ] pwd
/home/chez_moi/Etoiles/
```

ls

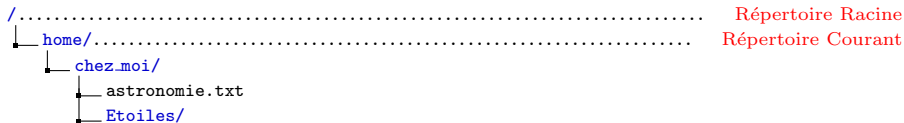
SYNTAXE

```
ls <source>
```

DESCRIPTION

- Affiche le contenu d'un répertoire.
- Par défaut si aucune source n'est indiquée, la commande affiche le contenu du répertoire courant.

EXEMPLE D'UTILISATION:



```
[ login@localhost /home/ ] ls
chez_moi/
```

```
[ login@localhost /home/ ] ls chez_moi/
Etoiles/ astronomie.txt
```

ls(bis)

SYNTAXE

```
ls -a <source>
```

DESCRIPTION

- Affiche le contenu d'un répertoire y compris les fichiers et répertoires cachés.
- Les fichiers et répertoires cachés ont un nom dont le premier caractère est un point.
- Les fichiers et répertoires cachés sont utilisés par le système ou certaines applications.

EXEMPLE D'UTILISATION:

chez_moi/.... **Rép. Courant**

```

./ssh/
├── id_rsa
├── id_rsa.pub
├── known_hosts
├── .bashrc
├── astronomie.txt
├── Etoiles/
│   └── soleil.jpg

```

Sans option -a

```

[ login@localhost ~ ] ls
astronomie.txt
Etoiles/

[ login@localhost ~ ] █

```

Avec option -a

```

[ login@localhost ~ ] ls -a
.
..
.ssh/
.bashrc
astronomie.txt
Etoiles/

[ login@localhost ~ ] █

```

ls(ter)

SYNTAXE

```
ls -l <source>
```

DESCRIPTION

- Affiche le contenu d'un répertoire en format long.
- Le format long donne le nom du propriétaire et son groupe, ainsi que les droits des différentes classes d'utilisateurs sur les fichiers et répertoires.

EXEMPLE D'UTILISATION:

```
chez_moi/..... Répertoire Courant
├── public_html/
│   └── index.html
└── astronomie.txt
```

```
[ login@localhost ~ ] ls -l
total 32
drwxr-xr-x 2 santini ensinfo 4096 20 jui 15 :50 public_html
-rw-r--r-- 1 santini ensinfo  25 20 jui 15 :49 telluriques.txt
```

Ici, le nom de l'utilisateur est **santini**, nom du groupe est **ensinfo** et les droits sont colorés en **vert**.

cd

SYNTAXE

```
cd <cible>
```

DESCRIPTION

- Change le répertoire courant (permet de naviguer dans l'arborescence).
- Si le chemin du répertoire cible est omit, le répertoire courant redevient par défaut le répertoire personnel.

EXEMPLE D'UTILISATION:



Commande #1 :

```
[ login@localhost /home ] cd
```

```
[ login@localhost ~ ] █
```

Commande #2 :

```
[ login@localhost /home ] cd chez_moi/Etoile
```

```
[ login@localhost ~/Etoile ] █
```

mkdir

SYNTAXE

```
mkdir chemin <chemin_2 ...>
```

DESCRIPTION

- Création d'un ou de plusieurs répertoires aux endroits spécifiés par les chemins.
- Si le chemin est occupé par un fichier ou un répertoire, il y a un message d'erreur.

EXEMPLE D'UTILISATION:

```
chez_moi/..... Répertoire Courant
├── astronomie.txt
├── Systeme_Solaire/..... Création Commande #1
├── Etoiles/
│   ├── Rouges/..... Création Commande #2
│   └── Bleues/..... Création Commande #3
└── Galaxies/..... Création Commande #3
```

Commande #1 :

```
[ login@localhost ~ ] mkdir Systeme_Solaire
```

Commande #2 :

```
[ login@localhost ~ ] mkdir Etoiles/Rouges
```

Commande #3 :

```
[ login@localhost ~ ] mkdir Galaxies Etoiles/Bleues
```


rm

SYNTAXE

```
rm chemin <chemin_2 ...>
```

DESCRIPTION

- La commande supprime le fichier pointé par le(s) chemin(s).
- Si le chemin pointe sur un répertoire, la commande affiche un message d'erreur.

EXEMPLE D'UTILISATION:

```
chez_moi/..... Répertoire Courant
├── astronomie.txt..... Supprimé par la Commande #1
├── Etoiles/
│   ├── soleil.jpg..... Supprimé par la Commande #2
│   └── aldebaran.gif..... Supprimé par la Commande #2
```

Commande #1 : [login@localhost ~] rm astronomie.txt

Commande #2 : [login@localhost ~] rm aldebaran.gif Etoiles/soleil.jpg

rm(bis)

SYNTAXE

```
rm -r chemin <chemin_2 ...>
```

DESCRIPTION

- L'option `-r` (Récuratif) permet de supprimer un répertoire et tout son contenu.

EXEMPLE D'UTILISATION:

```
chez_moi/..... Répertoire Courant
├── astronomie.txt
├── Etoiles/..... Supprimé par la Commande #1
│   ├── soleil.jpg..... Supprimé par la Commande #1
│   └── Galaxie/..... Supprimé par la Commande #1
│       └── Andromede.pdf..... Supprimé par la Commande #1
└── aldebaran.gif
```

Commande #1 :

```
[ login@localhost ~ ] rm -r Etoiles
```

cp

SYNTAXE

```
cp source cible
```

DESCRIPTION

- Copie le fichier source vers la cible.
- La source doit être un fichier ordinaire (pas un répertoire),
- Si la source est un répertoire la commande produit un message d'erreur.
- Si la cible :
 - est le chemin d'un répertoire existant, le fichier sera copié dans ce répertoire et conservera son nom,
 - ne correspond pas à un répertoire existant, le fichier sera copié avec le nom cible.

EXEMPLE D'UTILISATION:

```
chez_moi/..... Répertoire Courant
├── astronomie.txt..... Fichier Source Commande #1
├── Etoiles/..... Répertoire Cible Commande #1
│   └── astronomie.txt..... Copié/Créé par la Commande #1
└── cv.pdf
```

Commande #1 : `[login@localhost ~] cp astronomie.txt Etoiles`

cp(bis)

SYNTAXE

```
cp source <source_2 ...> cible
```

DESCRIPTION

- Copie plusieurs fichiers sources vers la cible.
- Les sources doivent être des fichiers ordinaires, et la cible un répertoire.

EXEMPLE D'UTILISATION:

```
chez_moi/..... Répertoire Courant
├── cv.pdf ..... Fichier Source Commande #2
├── motivations.pdf ..... Fichier Source Commande #2
└── Candidature/..... Répertoire Cible Commande #2
    ├── cv.pdf ..... Copié/Créé par la Commande #2
    └── motivations.pdf ..... Copié/Créé par la Commande #2
```

Commande #2 : `[login@localhost ~] cp cv.pdf motivations.pdf Candidature`

cp(ter)

SYNTAXE

```
cp -r source <source_2 ...> cible
```

DESCRIPTION

- L'option `-r` (**R**écursif) permet de copier un répertoire et son contenu si il apparait dans le(s) source(s).

EXEMPLE D'UTILISATION:

```
chez_moi/..... Répertoire Courant
├── astronomie.txt
├── Galaxie/..... Fichier Source Commande #3
│   ├── Andromede.pdf
│   └── Etoiles/..... Répertoire Cible #3
│       ├── soleil.jpg
│       └── Galaxie/..... Copié/Créé par la Commande #3
│           └── Andromede.pdf..... Copié/Créé par la Commande #3
└── aldebaran.gif
```

```
Commande #3 : [ login@localhost ~ ] cp -r Galaxies Etoiles
```

mv

SYNTAXE

```
mv source cible
```

DESCRIPTION

Déplace/Renomme un fichier ou répertoire.

- modifie le chemin d'accès à la source qui devient le chemin cible.
- Le chemin source disparaît et le chemin cible est créé.
- Le fichier ou répertoire pointé reste le même.
- La cible doit être un chemin non occupé ou un répertoire.

EXEMPLE D'UTILISATION: RENOMMER UN FICHIER

État Initial de l'arborescence :

```
chez_moi/..... Répertoire Courant
├─ AstroNomIe.TXT..... Fichier Source
```

État Final de l'arborescence :

```
chez_moi/..... Répertoire Courant
├─ astronomie.txt..... Fichier Renommé
```

```
[ login@localhost ~ ] mv AstroNomIe.TXT astronomie.txt
```

mv(bis)

EXEMPLE D'UTILISATION: DÉPLACER UN RÉPERTOIRE

État Initial de l'arborescence :

```
chez_moi/..... Répertoire Courant
├── astronomie.txt..... Fichier Source
└── Etoiles/..... Répertoire Cible
```

État Final de l'arborescence :

```
chez_moi/..... Répertoire Courant
├── Etoiles/..... Répertoire Cible
└── astronomie.txt..... Fichier Déplacé
```

```
[ login@localhost ~ ] mv astronomie.txt Etoiles
```

EXEMPLE D'UTILISATION: RENOMMER UN RÉPERTOIRE

État Initial de l'arborescence :

```
chez_moi/..... Répertoire Courant
├── Etoiles/..... Répertoire Source
│   └── astronomie.txt
```

État Final de l'arborescence :

```
chez_moi/..... Répertoire Courant
├── Relativite/..... Répertoire Renommé
│   └── astronomie.txt
```

```
[ login@localhost ~ ] mv Etoiles Relativite
```

mv(ter)

EXEMPLE D'UTILISATION:

État Initial de l'arborescence :

```
chez_moi/..... Répertoire Courant
├── astronomie.txt ..... Fichier Source
├── relativite.pdf ..... Fichier Source
└── Etoiles/ ..... Répertoire Cible
```

État Final de l'arborescence :

```
chez_moi/..... Répertoire Courant
├── Etoiles/ ..... Répertoire Cible
├── astronomie.txt ..... Fichier Déplacé
└── relativite.pdf ..... Fichier Déplacé
```

```
[ login@localhost ~ ] mv astronomie.txt relativité.pdf Etoiles
```

EXEMPLE D'UTILISATION:

État Initial de l'arborescence :

```
chez_moi/..... Répertoire Courant
├── relativite.pdf ..... Fichier Source
├── Etoiles/ ..... Répertoire Source
│   └── astronomie.txt
└── Espace/ ..... Répertoire Cible
```

État Final de l'arborescence :

```
chez_moi/..... Répertoire Courant
├── Espace/ ..... Répertoire Cible
├── relativite.pdf ..... Fichier Déplacé
├── Etoiles/ ..... Répertoire Déplacé
│   └── astronomie.txt ..... Fichier Déplacé
```

```
[ login@localhost ~ ] mv relativité.pdf Etoiles Espace
```


find

SYNTAXE

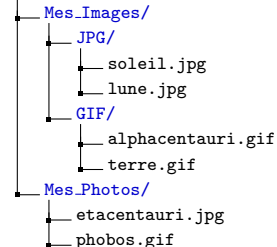
```
find depart -iname "motif"
```

DESCRIPTION

- Recherche dans les répertoires et sous-répertoires les fichiers dont le nom correspond au motif en partant du point de l'arborescence spécifié par le depart.
- L'option `-iname` indique que le motif sera recherché sans tenir compte des majuscules et minuscules.

EXEMPLE D'UTILISATION:

chez_moi/ . Répertoire courant



```

[ login@localhost ~ ] find . -iname *.gif
./Mes_Images/GIF/alphacentauri.gif
./Mes_Images/GIF/terre.gif
./Mes_Photos/phobos.gif

[ login@localhost ~ ] find . -iname *centauri*
./Mes_Images/GIF/alphacentauri.gif
./Mes_Photos/etacentauri.jpg

[ login@localhost ~ ] find Mes_Images/ -iname *e.*
Mes_Images/GIF/terre.gif
Mes_Images/JPG/lune.jpg

[ login@localhost ~ ] █

```

find(bis)

SYNTAXE

```
find depart -iname "motif" -exec commande \;
```

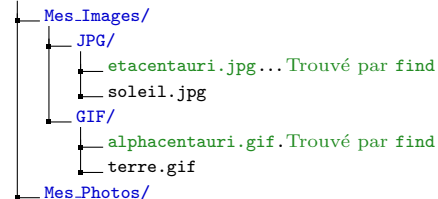
DESCRIPTION

- Exécute la commande sur la liste des fichiers identifiés par `find`,
- Dans la rédaction de la commande, la liste des fichiers est symbolisée par les caractères `{}`.

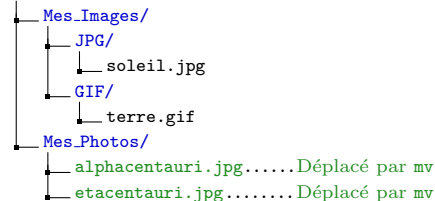
EXEMPLE D'UTILISATION:

```
[ login@localhost ~ ] find ./ -iname *centauri* -exec mv {} Mes_Photos \;
```

chez_moi/.....État Initial Répertoire courant



chez_moi/.....État Final Répertoire courant



ln

SYNTAXE

```
ln -s source cible
```

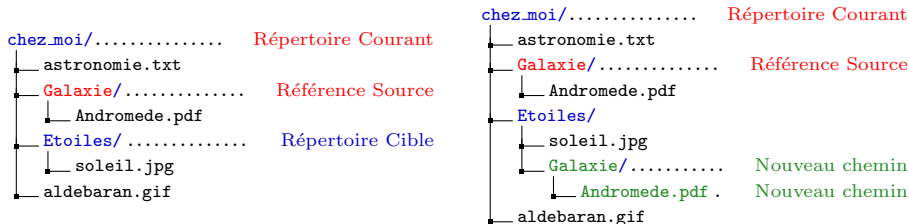
DESCRIPTION

- Crée un lien symbolique entre la référence source et le chemin cible..

EXEMPLE D'UTILISATION:

```
[ login@localhost ~ ] ln -s Galaxies Etoiles/Galaxies
```

Le lien symbolique sur un répertoire donne également accès à toutes les références contenues dans le répertoire pointé par le lien. Ainsi, le fichier ~/Galaxie/Andromede.pdf est aussi accessible par le chemin ~/Etoiles/Galaxie/Andromede.pdf.



alias

SYNTAXE

```
alias nom_de_la_commande=expression
```

DESCRIPTION

- crée un alias entre un nom de commande et une expression.
- l'expression est donnée entre *quotes* : *'expression ...'*

EXEMPLE D'UTILISATION:

chez_moi/.. Répertoire Courant

```
├── public_html/
│   └── index.html
└── astronomie.txt
```

```
[ login@localhost ~ ] ll
-bash : ll : command not found

[ login@localhost ~ ] alias ll='ls -l'

[ login@localhost ~ ] ls -l
total 32
drwxr-xr-x 2 santini ensinfo 4096 20 jui 15 :50 public_html
-rw-r--r-- 1 santini ensinfo  25 20 jui 15 :49 telluriques.txt
```

PLAN

1 ARBORESCENCE

2 PROCESSUS

- ps
- top
- chmod

3 ARCHIVAGE

4 MANIPULATION DE TEXTE

5 SYSTÈME DE FICHER

ps

SYNTAXE

```
ps <-eu>
```

DESCRIPTION

- Affiche les processus en cours d'exécution.
- L'option <-e> indique que tous les processus doivent être affichés,
- L'option <-u> restreint l'affichage aux processus de l'utilisateur.

EXEMPLE D'UTILISATION:

```
[ login@localhost ~ ] ps -eu
Warning : bad ps syntax, perhaps a bogus '-'? See http://procps.sf.net
  USER PID %CPU %MEM VSZ  RSS  TTY STAT  START  TIME COMMAND
santini 5905 0.0  0.2 4824 1656 pts/1  Ss 09 :27 0 :00 -bash LC_ALL=fr_FR.UTF
santini 5962 0.0  0.1 3884  896 pts/1  R+ 09 :48 0 :00 ps -eu MANPATH=/etc/jav


[ login@localhost ~ ] █
```

top

SYNTAXE

top

DESCRIPTION

- Permet de suivre dynamiquement (temps réel) les ressources matériel utilisées par chaque processus.
- Ouvre un interface dans la ligne de commande qui peut être quittée en pressant la touche 
- Donne pour chaque processus en autres choses, le PID, le nom du propriétaire, la date de lancement du processus, les %CPU et %MEM utilisés.

EXEMPLE D'UTILISATION:

```
Tasks : 85 total, 1 running, 84 sleeping, 0 stopped, 0 zombie
Cpu(s) : 5.7%us, 0.0%sy, 0.0%ni, 93.6%id, 0.0%wa, 0.7%hi, 0.0%si, 0.0%st
Mem : 772068k total, 231864k used, 540204k free, 2412k buffers
Swap : 995992k total, 0k used, 995992k free, 161316k cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
5116	root	20	0	33832	22m	6576	S	5.7	3.0	0 :19.49	X
5879	santini	20	0	16060	7344	6116	S	0.3	1.0	0 :01.06	xfce4-netload-p
1	root	20	0	1664	568	496	S	0.0	0.1	0 :02.95	init
2	root	20	0	0	0	0	S	0.0	0.0	0 :00.00	kthreadd
3	root	RT	0	0	0	0	S	0.0	0.0	0 :00.00	migration/0

chmod

SYNTAXE

```
chmod droit fichier
```

DESCRIPTION

- Modifie les droits et permissions accordés par le propriétaire aux différents utilisateurs du système.

EXEMPLE D'UTILISATION:

Retire au propriétaire le droit d'écriture sur le fichier `cv_2011.pdf`.

```
[ login@localhost ~ ] chmod u-w cv_2011.pdf
```

Retire aux utilisateurs qui ne sont ni le propriétaire ni membre de son groupe les droits de lecture, d'écriture et d'exécution.

```
[ login@localhost ~ ] chmod o-rwx listing.bash
```

Ajoute au propriétaire et aux membres de son groupe le droit d'exécution sur le fichier `listing.bash`.

```
[ login@localhost ~ ] chmod ug+x listing.bash
```

Ajoute à tous les utilisateurs, tous les droits.

```
[ login@localhost ~ ] chmod a+rx listing.bash
```


chmod(bis)

DESCRIPTION

Il existe plusieurs notations des droits.

- La notation alphanumérique :(uhoa) (+/-) (rwx)
- La notation octale :

Droit	---	--x	-w-	-wx	r--	r-x	rw-	rwx
Binaire	000	001	010	011	100	101	110	111
Octale	0	1	2	3	4	5	6	7

Alphabétique	r	w	x	r	-	x	-	-	x
Binaire	1	1	1	1	0	1	0	0	1
Octale	7			5			1		

EXEMPLE D'UTILISATION:

Alph.	Oct.	Alph.	Oct.
--- --- ---	000	---	---
rw- --- ---	600	rw- --- ---	700
rw- r-- r--	644	rw- r-x r-x	755
rw- rw- rw-	666	rw- rw- rw-	777

```
[ login@localhost ~ ] chmod 700 dir_parano
[ login@localhost ~ ] chmod 644 fichier_pub
```

PLAN

1 ARBORESCENCE

2 PROCESSUS

3 ARCHIVAGE

- gzip
- gunzip
- tar

4 MANIPULATION DE TEXTE

5 SYSTÈME DE FICHIER

gzip

SYNTAXE

```
gzip fichier <fichier.2 ...>
```

DESCRIPTION

- Comprime un ou plusieurs fichiers dont le nom est passé en paramètre.
- Le fichier source (initial non compressé) est supprimé et seul subsiste le fichier compressé.
- Le fichier compressé qui apparaît porte le même nom que le fichier initial avec l'extension `.gz` ajoutée à la fin.

EXEMPLE D'UTILISATION:

<pre>chez_moi/..... Répertoire Courant ├─ tellurique.tsv └─ astronomie.txt..... Avant gzip</pre>	<pre>chez_moi/..... Répertoire Courant ├─ tellurique.tsv └─ astronomie.txt.gz..... Après gzip</pre>
--	---

```
[ login@localhost ~ ] ls
astronomie.txt telluriques.tsv

[ login@localhost ~ ] gzip astronomie.txt

[ login@localhost ~ ] ls
astronomie.txt.gz telluriques.tsv
```

gunzip

SYNTAXE

```
gunzip fichier <fichier.2 ...>
```

DESCRIPTION

- Décompresse un ou plusieurs fichiers dont le nom est passé en paramètre.
- Le fichier source (compressé) est supprimé et seul subsiste le fichier décompressé.
- Le fichier décompressé qui apparaît porte le même nom que le fichier initial sans l'extension .gz ajoutée à la fin.

EXEMPLE D'UTILISATION:

```
chez_moi/..... Répertoire Courant
├─ tellurique.tsv
└─ astronomie.txt.gz..... Avant gunzip
```

```
chez_moi/..... Répertoire Courant
├─ tellurique.tsv
└─ astronomie.txt..... Après gunzip
```

```
[ login@localhost ~ ] ls
astronomie.txt.gz telluriques.tsv

[ login@localhost ~ ] gunzip astronomie.txt.gz

[ login@localhost ~ ] ls
astronomie.txt telluriques.tsv
```

tar

SYNTAXE

```
tar cv nom_archive fichier_ou_repertoire <autres_sources>
```

DESCRIPTION

- Crée un fichier archive dont le nom (chemin) est donné en premier argument et porte classiquement l'extension `.tar`.
- Les fichiers sources qui servent à créer l'archive sont préservés par la commande `tar`.
- L'option `c` (**C**reate), indique que la commande `tar` doit utiliser un algorithme d'archivage.
- L'option `v` (**V**erbose), permet d'afficher le déroulement de l'archivage.

EXEMPLE D'UTILISATION:

Regroupe dans la même archive `espace.tar` le fichier `astronomie.txt` et le répertoire `Images/` et son contenu :

```
[ login@localhost ~ ] tar cv espace.tar astronomies.txt Images/
```

tar(bis)

SYNTAXE

```
tar xv nom_archive
```

DESCRIPTION

- Extrait les fichiers et répertoires d'une archive.
- Les fichiers sont placés dans le répertoire courant.
- L'option **x** (**eXtarct**) indique que la commande `tar` doit utiliser un algorithme de désarchivage.

EXEMPLE D'UTILISATION:

Extrait le contenu de l'archive `espace.tar` :

```
[ login@localhost ~ ] tar xv espace.tar
```

tar(ter)

SYNTAXE

```
tar cvz nom_archive fichier_ou_repertoire <autres_sources>
```

SYNTAXE

```
tar xvz nom_archive
```

DESCRIPTION

- L'option `z` permet de créer ou d'extraire une archive compressée.
- L'extension donnée aux fichiers contenant une archive compressée par ce moyen est classiquement : `.tgz`

EXEMPLE D'UTILISATION:

Crée une archive compressée `espace.tgz` avec le fichier `astronomie.txt` et le répertoire `Images/` et son contenu :

```
[ login@localhost ~ ] tar cvz espace.tar astronomie.txt Images/
```

Extrait le contenu d'une archive compressée `espace.tgz` :

```
[ login@localhost ~ ] tar xvz espace.tar
```

PLAN

1 ARBORESCENCE

2 PROCESSUS

3 ARCHIVAGE

4 MANIPULATION DE TEXTE

- more
- less
- head
- tail
- cut
- grep
- cat
- sort
- uniq
- sed
- wc
- echo



5 SYSTÈME DE FICHER

more

SYNTAXE

```
more fichier <fichier_2 ...>
```

DESCRIPTION

- Affiche le contenu du (des) fichier(s) page par page,
- L'affichage s'adapte à la taille du shell,
- Pour passer à la ligne suivante, l'utilisateur presse la touche .
- Pour passer à la page suivante, l'utilisateur presse la touche .
- Une fois que tout le contenu du fichier a défilé, l'utilisateur retrouve un nouveau prompt.

EXEMPLE D'UTILISATION:

- Cette commande est utilisée pour parcourir des documents dont l'affichage dépasse la taille de la fenêtre du terminal.
- Utilisée avec un tube (cf. Partie sur les Redirections) elle permet de visualiser tous les résultats d'une commande qui dépasserait la taille de la fenêtre du terminal. Par exemple, si un répertoire contient de très nombreux fichiers, la commande `ls` qui affiche le contenu du répertoire peut produire un affichage très long. Si l'on souhaite passer en revue tous les fichiers il faut alors utiliser la commande suivante :

```
[ login@localhost ~ ] ls Ma_Musique | more
```

less

SYNTAXE

```
less fichier
```






DESCRIPTION







- Affiche le contenu d'un fichier,
- Permet de naviguer en avant et en arrière dans le fichier.
- Permet d'effectuer des recherches de mot(if)s.

La commande ouvre une interface dans la fenêtre du terminal. Contrairement à la commande `more`, on ne revient pas à la ligne de commande lorsqu'on atteint la fin du fichier, pour cela il faut quitter l'application.

EXEMPLE D'UTILISATION:

Pour avoir une description complète des commandes de navigation dans l'interface de visualisation `less`, reportez-vous aux pages de `man`. Les commandes les plus utilisées sont :

Combinaison de touches	Action
	Affiche l'aide (abrégé des commandes)
	Avancer d'une page (forward)
	Reculer d'une page (backward)
	Avancer d'une ligne
	Reculer d'une ligne

Combinaison de touches	Action
	Quitter
	Aller à la première ligne
 + 	Aller à la dernière ligne
 + 	Aller à la ligne numéro num

head

SYNTAXE

```
head < -int > fichier
```

DESCRIPTION

- Affiche par défaut les 10 premières lignes d'un fichier.
- Si un entier *n* précède le nom du fichier, la commande affiche les *n* premières lignes du fichier.

EXEMPLE D'UTILISATION:

Soit le fichier `planetes.txt` contenant les lignes suivantes :

planetes.txt

```
# Premier groupe
1 Mercure Tellurique
2 Venus Tellurique
3 Terre Tellurique
4 Mars Tellurique
# Deuxième groupe
1 Jupiter Gazeuse
2 Saturne Gazeuse
3 Uranus Gazeuse
4 Neptune Gazeuse
```

La commande suivante affiche les 5 premières lignes du fichier :

```
[ login@localhost ~ ] head -5 planetes.txt
# Premier groupe
1 Mercure Tellurique
2 Venus Tellurique
3 Terre Tellurique
4 Mars Tellurique

[ login@localhost ~ ] █
```

tail

SYNTAXE

```
tail < -int > fichier
```

DESCRIPTION

- Affiche par défaut les 10 dernières lignes d'un fichier.
- Si un entier *n* précède le nom du fichier, la commande affiche les *n* dernières lignes du fichier.

EXEMPLE D'UTILISATION:

Soit le fichier `planetes.txt` contenant les lignes suivantes :

planetes.txt

```
# Premier groupe
1 Mercure Tellurique
2 Venus Tellurique
3 Terre Tellurique
4 Mars Tellurique
# Deuxième groupe
1 Jupiter Gazeuse
2 Saturne Gazeuse
3 Uranus Gazeuse
4 Neptune Gazeuse
```

La commande suivante affiche les 4 dernières lignes du fichier :

```
[ login@localhost ~ ] tail -4 planetes.txt
1 Jupiter Gazeuse
2 Saturne Gazeuse
3 Uranus Gazeuse
4 Neptune Gazeuse

[ login@localhost ~ ] █
```

cut

SYNTAXE

```
cut -d 'sep' -f n fichier
```

DESCRIPTION

- Affiche une colonne du fichier.
- L'option `<-d 'sep'>` permet de changer le séparateur par défaut qui est la tabulation. Le séparateur est donné entre guillemets simples.
- L'option `<-f n>` indique que la commande doit afficher la n^{ème} colonne.

EXEMPLE D'UTILISATION:

Cas#1 : les mots (les champs) sont séparés par des tabulations :

tellur.tsv		
1	Mercure	Venus
2	Terre	Mars

Commande #1

```
[ login@localhost ~ ] cut -f 2 tellur.tsv
Mercure
Terre
[ login@localhost ~ ] █
```

Cas#2 : les mots (les champs) sont séparés par le caractère = :

jov.txt	
1=Jupiter=Saturne	
1=Uranus=Neptune	

Commande #2

```
[ login@localhost ~ ] cut -d '=' -f 3 jov.txt
Saturne
Neptune
[ login@localhost ~ ] █
```

grep

SYNTAXE

```
grep "motif" fichier
```

DESCRIPTION

- Affiche les lignes du fichier qui comportent le "motif".
- Les lignes sont affichées dans leur ordre d'apparition dans le fichier.

EXEMPLE D'UTILISATION:

Soit le fichier `planetes.txt` contenant les lignes suivantes :

`planetes.txt`

```
# Premier groupe
1 Mercure Tellurique
2 Venus Tellurique
3 Terre Tellurique
4 Mars Tellurique
# Deuxième groupe
1 Jupiter Gazeuse
2 Saturne Gazeuse
3 Uranus Gazeuse
4 Neptune Gazeuse
```

Commandes :

```
[ login@localhost ~ ] grep 'Tellurique' planetes.txt
1 Mercure Tellurique
2 Venus Tellurique
3 Terre Tellurique
4 Mars Tellurique

[ login@localhost ~ ] grep '1' planetes.txt
1 Mercure Tellurique
1 Jupiter Gazeuse

[ login@localhost ~ ] █
```

cat

SYNTAXE

```
cat fichier <fichier_2 ...>
```

DESCRIPTION

- Affiche le contenu des fichiers les uns à la suite des autres.
- Les fichiers sont concaténés dans l'ordre des paramètres.

EXEMPLE D'UTILISATION:

Cette commande est en générale utilisée pour concaténer des fichiers textes. On l'utilise avec une commande de redirection (*cf.* Partie Redirections) pour enregistrer le résultat de la concaténation dans un nouveau fichier.

Soient les deux fichiers suivants :

tellur.txt

Mercure, Venus
Terre, Mars

jov.txt

Jupiter, Saturne
Uranus, Neptune

La commande :

```
[ login@localhost ~ ] cat tellur.txt jov.txt
Mercure, Venus
Terre, Mars
Jupiter, Saturne
Uranus, Neptune

[ login@localhost ~ ] █
```

sort

SYNTAXE

```
sort <-r> fichier
```

DESCRIPTION

- Affiche les lignes du fichier triées par ordre croissant.
- L'option `-r` inverse l'ordre de tri.

EXEMPLE D'UTILISATION:

Soit le fichier :

donnees.txt

```
a
A
1
7
8
71
```

```
[ login@localhost ~ ] sort donnees.txt
1
7
71
8
A
a
[ login@localhost ~ ] █
```


uniq

SYNTAXE

```
uniq fichier
```

DESCRIPTION

- Affiche les lignes du fichier en supprimant les lignes consécutives identiques.

EXEMPLE D'UTILISATION:

Soit le fichier :

```
donnees.txt
```

```
1
lune
Terre
Terre
lune
```

```
[ login@localhost ~ ] uniq donnees.txt
1
lune
Terre
lune

[ login@localhost ~ ] █
```

sed

SYNTAXE

```
sed 's/motif/new/g' fichier
```

DESCRIPTION

La commande `sed` est une commande qui permet de faire de nombreuses opérations. Nous ne verrons ici que la syntaxe permettant de substituer un motif dans un texte.

- Affiche le contenu du fichier après avoir remplacé les occurrences du motif par `new`.

EXEMPLE D'UTILISATION:

Soit le fichier :

```
dialogue.txt
```

```
- C'est par ici!!!
- Où ça, "ici"?
```

```
[ login@localhost ~ ] sed 's/ici/là/' dialogue.txt
- C'est par là!!!
- Où ça, "là"?

[ login@localhost ~ ] █
```

SYNTAXE

```
wc fichier <fichier.2 ...>
```

DESCRIPTION

- Affiche des statistiques sur le nombre de lignes, de mots et de caractères (comptés en nombre d'octets) contenus dans le fichier dont le chemin est donné en paramètre.

EXEMPLE D'UTILISATION:

Soit le fichier suivant :

tellur.tsv	
1	Mercure Venus
2	Terre Mars

Commande #1 :

```
[ login@localhost ~ ] wc tellur.tsv
2      6      29 tellur.tsv

[ login@localhost ~ ] █
```

L'affichage produit indique que le fichier `tellur.tsv` comporte :

- 2 lignes,
- 6 mots et
- 29 caractères. La taille du fichier texte est donc de 29 octets ...

wc(bis)

SYNTAXE

```
wc -l fichier <fichier_2 ...>
```

DESCRIPTION

- L'option `-l` indique que l'on affiche que le nombre de lignes.

EXEMPLE D'UTILISATION:

Soit le fichier suivant :

tellur.tsv	
1	Mercure Venus
2	Terre Mars

L'affichage produit indique que le fichier `tellur.tsv` comporte :

- 2 lignes.

Commande #1 :

```
[ login@localhost ~ ] wc -l tellur.tsv
2      tellur.tsv
[ login@localhost ~ ] █
```

echo

SYNTAXE

```
echo expression
```

DESCRIPTION

- Affiche sur la sortie standard l'expression après interprétation.

EXEMPLE D'UTILISATION:

Affiche 'Bonjour' :

```
[ login@localhost ~ ] echo Bonjour
Bonjour
[ login@localhost ~ ] █
```

Définie une variable puis affiche sa valeur :

```
[ login@localhost ~ ] Astre=Terre
[ login@localhost ~ ] echo $Astre
Terre
[ login@localhost ~ ] echo La planete $Astre
La planete Terre
[ login@localhost ~ ] █
```

PLAN

1 ARBORESCENCE

2 PROCESSUS

3 ARCHIVAGE

4 MANIPULATION DE TEXTE

5 SYSTÈME DE FICHIER

- mount
- df
- du
- which
- dirname
- basename
- wget

mount

SYNTAXE

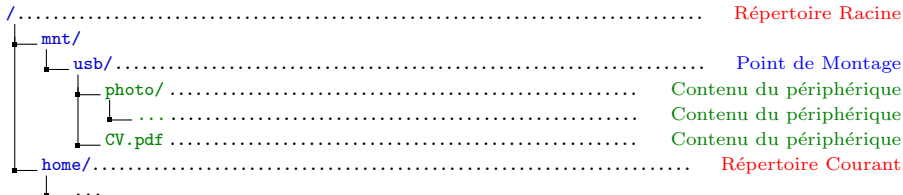
```
mount périphérique point_de_montage
```

DESCRIPTION

- `périphérique` correspond soit à un fichier de périphérique (`/dev/xxx`), soit à l'adresse d'un disque (`nom_réseau_du_disque :répertoire_du_disque`).
- `point_de_montage` correspond à un nom de répertoire valide dans l'arborescence principale donnant accès au contenu de l'arborescence du périphérique.

EXEMPLE D'UTILISATION:

```
[ login@localhost /home ] mount /dev/sda1 /mnt/usb
```



df

SYNTAXE

df -h

DESCRIPTION

- Affiche les disques montés et leur capacité de mémoire.
- L'option -h (human readable) convertie l'affichage des tailles mémoires en unités conventionnelles (en nombre de blocs par défaut).

EXEMPLE D'UTILISATION:

```
[ login@localhost ~ ] df -h
Sys. de fichiers      Taille  Uti.   Disp.  Uti%   Monté sur
/dev/sda1             56G    16G    37G    31%    /
myserver :/home/sant  1,8T   1,6T   192G   90%    /users/santini
...                   ...     ...     ...     ...     ...

[ login@localhost ~ ] █
```


du

SYNTAXE

`du -sh`

DESCRIPTION

- Affiche l'espace mémoire utilisé par un fichier ou un répertoire.
- L'option `-h` (human readable) convertie l'affichage des tailles mémoires en unités conventionnelles (en nombre de blocs par défaut).
- L'option `-s` (sumurize) n'affiche pas le détail des fichiers et des sous-répertoires.

EXEMPLE D'UTILISATION:

```
[ login@localhost ~ ] du -sh Documents/  
5,2G Documents/  
  
[ login@localhost ~ ] █
```

which

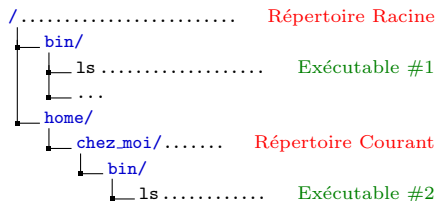
SYNTAXE

```
which nom_de_la_commande
```

DESCRIPTION

- Affiche le chemin du fichier correspondant à une commande.
- Parcours successivement les répertoires de la variable \$PATH. Dès qu'il trouve un fichier correspondant au nom de la commande il renvoie son chemin.

EXEMPLE D'UTILISATION:



```

[ login@localhost /home/chez_mo ] echo $PATH
/bin:/usr/bin:/usr/local/bin:/home/chez_moi/bin

[ login@localhost /home/chez_moi ] which ls
/bin/ls
  
```

dirname

SYNTAXE

```
dirname chemin
```

DESCRIPTION

- Ne conserve que la partie répertoire d'un chemin d'accès.
- Il n'est pas nécessaire que le chemin existe dans l'arborescence. Le chemin est traité comme une chaîne de caractères.

EXEMPLE D'UTILISATION:

```
[ login@localhost ~ ] dirname Documents
.
[ login@localhost ~ ] dirname Documents/cv.txt
Documents
[ login@localhost ~ ] dirname Documents/Photos/
Documents
[ login@localhost ~ ] dirname Documents/Photos/Soleil.jpg
Documents/Photos
[ login@localhost ~ ] █
```

basename

SYNTAXE

```
basename chemin
```

DESCRIPTION

- Élimine le chemin d'accès et le suffixe d'un nom de fichier.
- Il n'est pas nécessaire que le chemin existe dans l'arborescence. Le chemin est traité comme une chaîne de caractères.

EXEMPLE D'UTILISATION:

```
[ login@localhost ~ ] basename curriculum.pdf
curriculum
[ login@localhost ~ ] basename Documents/cv.txt
cv
[ login@localhost ~ ] basename Documents/Photos/Soleil.jpg
Soleil
[ login@localhost ~ ] █
```

wget

SYNTAXE

```
wget chemin
```

DESCRIPTION

- Client HTTP, HTTPS et FTP .
- Permet de récupérer du contenu d'un serveur Web ou FTP (télécharger).

EXEMPLE D'UTILISATION:

```
[ login@localhost ~ ] wget http://www-lipn.univ-paris13.fr/~santini/intro_syste
me/2011_2012_S1D_cours_1.pdf .
Résolution de www-lipn.univ-paris13.fr... 10.10.0.68
Connexion vers www-lipn.univ-paris13.fr[10.10.0.68] :80... connecté.
requête HTTP transmise, en attente de la réponse... 200 OK
Longueur : 4568618 (4,4M) [application/pdf]
Sauvegarde en : «2011_2012_S1D_cours_1.pdf»

100%[=====>] 4 568 618 10,4M/s ds 0,4s

2012-01-02 16 :02 :59 (10,4 MB/s) - «2011_2012_S1D_cours_1.pdf» sauvegardé
[4568618/4568618]

[ login@localhost ~ ] ls -l ./2011_2012_S1D_cours_1.pdf
-rw-r--r-- 1 santini users 4,4M 2011-12-14 10 :33 ./2011_2012_S1D_cours_1.pdf
```