

# Aides-mémoires Linux

**Christophe BLAESS**

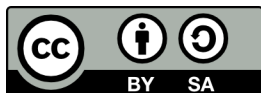
`christophe.blaess@logilin.fr`

<https://www.blaess.fr/christophe/>  
twitter: @chrisblaess



*Ingénierie et formations sur Linux et les logiciels libres*  
<https://www.logilin.fr>

Ce support de formation est distribué sous licence **Creative Commons 4.0**



*(Attribution - Partage dans les mêmes conditions).*

Vous êtes libres de copier et partager ce document, en mentionnant son origine. Si vous l'intégrez dans un contenu plus vaste, ce dernier devra être distribué avec les mêmes droits.

Ce chapitre a été rédigé en utilisant des logiciels libres sur système d'exploitation Linux :

– *LibreOffice Writer* pour le support et la mise en page

<https://www.blaess.fr/christophe/>

<https://www.logilin.fr>

# Aide-mémoire des commandes Linux (A à L)

Ce document regroupe les principales commandes susceptibles d'être employées régulièrement par les utilisateurs Unix, en rappelant leurs options les plus courantes. Pour avoir plus de détail sur une commande particulière, on consultera le manuel Unix (commande man).

Christophe Blaess pour Logilin

**a** **propos** Liste les pages « man » traitant d'un sujet  
| `apropos shell`

**arch** Affiche l'architecture de la machine.

**at**, **atq**, **atrm** Mémorise, examine ou supprime des jobs à exécuter ultérieurement.

-f lire les commandes dans le fichier indiqué.  
| `$ at now + 10 minutes < script.sh`  
| `$ at 20:55 -f start-record.sh`

**awk** Interpréteur du langage Awk.

**NF** nombre de champs sur la ligne

**FS** séparateur de champs

**NR** numéro d'enregistrement (de ligne)

**\$1**, **\$2**...**\$NF** champs successifs

| `ls -l | awk '{sum+= $5} END{print sum}'`  
| additionne les tailles des fichiers du répertoire courant

**b** **asename** Élimine le chemin d'accès et le suffixe éventuel d'un nom de fichier.

| `$ basename /usr/src/linux/signal.c`  
| `signal.c`  
| `$ basename /usr/src/linux/signal.c .c`  
| `signal`

**bash** Shell Gnu Bash

**batch** Lance un job en différé suivant la charge CPU  
| `batch << END`  
| `sort-records.sh`  
| `END`

**bc** Calculateur en précision arbitraire.

-l bibliothèque mathématique complète.  
| `$ pi=$(echo "a(1)*4" | bc -l)`  
| `$ echo $pi`  
| `3.14159265358979323844`

**bg** Relance à l'arrière-plan un job arrêté.

| `$ ./my-application`  
| (Ctrl-Z)  
| [1]+ Stopped my-application  
| `$ bg`  
| [1]+ my-application &

**bunzip2** Décompresse un fichier .bz2.

**bzip2** Comprime un fichier.

**C** **al** Affiche un calendrier.

| `cal 5 2024`

**cat** Concatène des fichiers sur la sortie standard.

-n numérote les lignes en sortie,  
-v caractères spéciaux sous forme symbolique.  
| `cat header.txt body.txt sign.txt > mail.txt`

**cc** Compilateur C.

**cd** Change de répertoire de travail.

| `cd /usr/src/linux`  
| `cd -`  
| revient au répertoire précédent  
| `cd`  
| revient dans le répertoire personnel.

**chgrp** Change le groupe propriétaire d'un fichier.

-R modifie récursivement les sous-répertoires.  
| `chgrp team2 file`

**chmod** Modifie les permissions d'un fichier.

-R modifie récursivement les sous-répertoires.

| `chmod 644 letter.txt`

lecture pour tous, écriture seulement pour propriétaire

| `chmod 755 script.sh`

lecture et exécution pour tous, écriture pour propriétaire

| `chmod u+s my-executable`

Activation du bit Set-UID du fichier.

**chown** Modifie propriétaire et groupe d'un fichier.

-R modifie récursivement les sous-répertoires.

| `chown user1.team1 file`

**chsh** Change le shell appelé à la connexion.

-l liste des shells disponibles,

-s utilise le shell de connexion indiqué.

| `chsh -s /bin/ksh`

**cksum** Nombre d'octets et somme de contrôle.

**clear** Efface l'écran.

**cmp** Compare deux fichiers.

-l affiche le rang de chaque octet différent,

-s n'affiche rien, renvoie vrai ou faux.

| `if cmp -s file1 file2; then ...`

**col** Élimine les retours et sauts-de-ligne en arrière.

-b enlève tous les retours en arrière.

| `man col | col -b > col.man.txt`

**compress** Compression simple de fichier.

**cp** Copie de fichiers.

-R copie récursive des sous-répertoires,

-p garde horodatage, propriétaire, permissions,

-d copie les liens symboliques en tant que tels.

| `cp file file.bkup`

| `cp -Rdp file_* /other/directory`

**crontab** Édite le fichier crontab personnel.

-l affiche le contenu actuel,

-e édite le fichier crontab,

-r supprime le fichier crontab.

**csh** Shell C.

**csplit** Découpe un fichier suivant des lignes de contexte.

-f préfixe pour nommer les nouveaux fichiers.

| `csplit -f prefix file '/^$/' {'*}`

| crée prefix00 prefix01... en découpant le fichier à chaque ligne vierge

**cut** Supprime une partie de chaque ligne.

-b affiche les caractères indiqués,

-f affiche les champs indiqués,

-d caractère séparateur de champ

| `ls -l | cut -b 20-28`

| affiche uniquement les caractères 20 à 28, c'est-à-dire le groupe des fichiers.

**d** **ate** Affiche la date et l'heure du système.

-d indique la date à afficher

+ chaîne de format pour l'affichage.

| `date +"Date = %D, Heure = %X"`

| `date -d 20220102 +"%A"`

| affiche "mardi" (2 janvier 2022).

**dd** Copie générique et conversion de fichiers.

**if=** nom du fichier d'entrée,

**of=** nom du fichier de sortie,

**bs=** taille des blocs à copier,

**count=** nombre maximal de blocs à copier,

**skip=** position de début de lecture,

**seek=** position de début d'écriture.

| `dd if=file.iso of=/dev/sdb bs=4M`

**df** Place occupée sur les systèmes de fichiers.

-k affiche les tailles en kilo-octets,

-P affiche une ligne d'en-tête.

| `df -k /tmp/sauvegarde`

**diff** Trouve les différences entre des fichiers.

-i ignore les différences majuscule/minuscule,

-b ignore les différences d'espaces blancs,

- u** utilise un format compatible avec **patch**,
- r** étudie récursivement les sous-répertoires.

```
diff -u original copy > modifs.patch
```

**dirname** Affiche le répertoire d'un chemin d'accès.

```
$ dirname /usr/src/signal.c /usr/src
```

**dos2unix** Conversion de texte du format Dos vers Unix.

**du** Statistiques sur l'utilisation du disque.

- a** affiche les statistiques pour les fichiers,
- s** affiche seulement le total,
- x** ignore les autres systèmes de fichiers,
- k** affiche les tailles en kilo-octets.

**echo** Affiche une ligne de texte.

- e** interprète les caractères symboliques,
- n** évite le saut-de-ligne final.

```
echo "Warning!" >&2
echo -n "Your choice:"
echo -e "\r done: " $i "%"
echo -e "\007"
```

**ed** Éditeur ligne-à-ligne

**egrep** Synonyme de **grep** -E

**emacs** Éditeur Gnu pleine page  
Version X-Window : **xemacs**.

**env** Lance un programme en environnement modifié.

- i** Démarre dans un environnement vide.

```
env
affiche l'environnement en cours
env -i /bin/sh
démarré le shell dans un environnement neuf.
```

**expand** Convertit les tabulations en espaces.

- t** largeur de tabulation désirée,
- i** uniquement les tabulations en début de ligne.

```
expand -i < script.sh > listing.txt
```

**export** Passe une variable dans l'environnement du shell

```
export APP_DIR=/usr/local/lib/app/
APP_VERSION=1.5
export APP_VERSION
```

**expr** Évalue des expressions.

```
expr 4 "*" 3 + 2 affiche 14
```

(les guillemets protègent l'étoile par rapport au shell)

**false** Échoue en ne faisant rien.

```
until false; do ...
```

**fc** Édite la dernière ligne de l'historique avec l'éditeur mentionné dans la variable d'environnement **FCEDIT**.

**fg** Ramène un job à l'avant-plan.

**fgrep** Synonyme de **grep** -F

**file** Affiche le type d'un fichier

**find** Recherche des fichiers dans une arborescence.

- name** *motif* recherche sur le nom du fichier,
- regex** *expr* recherche sur le nom complet,
- atime** *n* dernier accès il y a *n* jours,
- ctime** *n* dernière modif. de l'état du fichier,
- mtime** *n* dernière modif. du contenu du fichier,
- perm** *mode* autorisations d'accès au fichier,
- size** *n* taille du fichier (en blocs),
- type** *t* type du fichier,
- print** affiche les noms des fichiers trouvés,
- exec** ...**\{\}** \; exécute l'action indiquée en remplaçant **\{\}** par le nom du fichier,
- ok** ...**\{\}**; **exec** avec confirmation.

```
find /tmp -ctime +30 -ok rm \{\} \;
find /home -name core -exec rm \{\} \;
```

**fold** Coupe les lignes d'un fichier à une largeur donnée.

**ftp** Transfert de fichiers entre machines.

**fuser** Identifie les processus utilisant un fichier.

- k** leur envoie le signal **SIGKILL**,
- i** confirmation avant d'envoyer le signal,
- m** tous processus accédant au système de fichiers.

```
fuser -k /mnt/cdrom
```

**git** Gestionnaire de version.  
De nombreux tutoriels sont disponibles sur Internet.

**grep** Affiche les lignes correspondant à un motif.

- E** le motif est une expression rationnelle étendue,
- F** le motif est une chaîne pas une expression,
- i** ignore différences majuscules/minuscules,
- v** affiche les lignes ne correspondant pas,
- l** affiche seulement le nom des fichiers.

```
grep -i "Pattern" files_*
grep -v "absent" file
```

**groups** Affiche les groupes d'un utilisateur.

**gunzip** Décompresse un fichier .gz.

**gzip** Comprime un fichier.

**head** Affiche le début d'un fichier.

- c** *n* affiche les *n* premiers octets,
- n** *n* affiche les *n* premières lignes.

**hostid** Affiche l'identifiant de la machine

**hostname** Affiche le nom de la machine

**iconv** Convertit des textes d'un jeu de caractères vers un autre

```
iconv -f LATIN1 -t UTF8 < file
```

**id** Affiche les UIDs et GIDs effectifs et réels.

- u** affiche seulement l'UID,
- g** affiche seulement le GID,
- r** affiche les identifiants réels.

```
if [ $(id -u) == 0 ]; then...
```

**jobs** Affiche la liste des jobs en cours.

**join** Fusionne les lignes de deux fichiers triés.

```
join file_1 file_2 > file_3
```

**kill** Envoie un signal à un processus.

- numéro** le signal dont le numéro est indiqué,
- l** affiche la ligne des signaux disponibles.

```
kill -9 30582
```

**killall** Envoie un signal aux processus de même nom.

- i** demande confirmation individuellement,
- l** affiche la liste des signaux disponibles.

```
killall xterm
```

**ksh** Shell Korn

**less** Affiche un fichier page-par-page.  
(alternative libre et puissante à **more**.)

**lex** Générateur d'analyseur lexical

**ln** Crée des liens entre fichiers.

- f** force l'écrasement du fichier s'il existe,
- s** crée un lien symbolique.

```
ln -sf appli-1.4.sh appli
```

**logger** Journalise un message système.

**login** Relance une connexion sur le système.

**logname** Nom de connexion de l'utilisateur.

**lp** Requête d'impression

- d** sélection de l'imprimante
- n** nombre de copies

```
pr -l 66 appli.c | lp -d listing
```

**ls** Liste les fichiers et le contenu des répertoires.

- a** aussi les fichiers commençant par un point,
- d** noms des répertoires, pas leur contenu,
- i** affiche les numéros d'i-nœud,
- l** utilise un format d'affichage long,
- R** affiche récursivement les sous-répertoires.

```
ls -al /home/usera
```



# Aide-mémoire des commandes Linux (M à Z)

Ce document regroupe les principales commandes susceptibles d'être employées régulièrement par les utilisateurs Unix, en rappelant leurs options les plus courantes. Pour avoir plus de détail sur une commande particulière, on consultera le manuel Unix (commande man).

Christophe Blaess pour Logilin

**make** Construction d'application, et gestion des dépendances.

**man** Affiche une page du manuel Unix.

**numéro** recherche dans la section indiquée,

- a affiche toutes les pages correspondant,
- t écrit la page Postscript sur la sortie standard,
- k équivalent à la commande **apropos**.

```
man 1 c
man -k socket
```

**md5sum** Calcule et affiche un compte-rendu MD5.

**mkdir** Crée des répertoires.

- p crée récursivement les répertoires parents,
- m **mode** fixe les autorisations d'accès.

```
mkdir -p /var/lib/new-app/font/big
```

**mkfifo** Crée des FIFOs (tubes nommés).

- m **mode** fixe les autorisations d'accès.

```
mkfifo -m 666 /tmp/server-fifo
```

**mknod** Crée des fichiers spéciaux.

**b** ou **c** fichier spécial bloc ou caractère

```
mknod /dev/sda1 b 8 1
```

crée le noeud de numéros majeur/mineur 8/1.

**more** Consulte un fichier page par page (voir **less**)

**mv** Déplace ou renomme des fichiers.

- f force l'écrasement du fichier destination.

```
for i in *.JPG; do mv $i ${i%.JPG}.jpg; done
renomme tous les fichiers .JPG en .jpg
```

**nice** Exécute un programme avec une priorité d'ordonnancement diminué.

- n **va leur** diminue la priorité de la valeur indiquée.

**nl** Numérote les lignes d'un fichier.

- f **a** numérote aussi les lignes vides.

**nohup** Exécute un programme en le rendant insensible aux déconnexions.

```
$ nohup ~/bin/compute &
[1] 17300
$ exit
```

**od** Affiche le contenu d'un fichier en octal ou sous d'autres formats.

- c affiche les caractères imprimables en Ascii,
- x affiche les codes hexadécimaux.

**passwd** Change le mot de passe.

```
$ passwd
# passwd user
```

**patch** Applique une série de modifications à un fichier.

- pn enlève **n** répertoires au début des noms de fichiers

```
$ patch -p1 < ../new_version/patch_1
```

**pathchk** Vérifie la validité d'un nom de fichier.

- p vérification stricte de la portabilité.
- ```
if pathchk "$rep/$fic" ; then...
```

**ping** Test de liaison entre machines.

- c nombre de tentatives
- w délai maximal en secondes

**pr** Prépare des fichiers de texte pour l'impression.

- h **texte** indique l'en-tête de chaque page,

- l **n** affiche **n** lignes par pages,

- t supprime les en-têtes et pieds de pages.

**printf** Affiche des données numériques formatées.

```
printf "%05d %4.2f" $x $y
```

**ps** Affiche l'état des processus en cours.

- ax** tous les processus (BSD)

- u** informations complètes (BSD)

- e tous les processus (SysV)

- f informations complètes (SysV)

- w lignes larges.

```
ps aux (BSD)
```

```
ps -ef (SysV)
```

**pwd** Affiche le nom du répertoire de travail.

**quota** Affiche les quotas d'utilisation du disque.

**rcp** Copie de fichiers entre systèmes différents.

**renice** Modifie la priorité d'un processus en cours.

```
renice +20 14210
```

**rev** Inverse les lignes d'un fichier (voir aussi **tac**).

**rlogin** Connexion sur un système distant (préférer **ssh**)

**rm** Efface des fichiers.

- f pas de confirmation,

- i confirmation avant chaque effacement,

- r efface récursivement les sous-répertoires.

```
rm -rf /home/usera/tmp
```

**rmdir** Suppression de répertoires vides.

**rsh** Exécution de commande sur système distant. (préférer **ssh**).

**Script** Enregistre une session de travail .

- a **fic** ajoute le résultat dans le fichier.

**sed** Éditeur non-interactif.

- e "... " commandes fournies sur la ligne,

- f **fic** commandes dans un fichier,

- n supprime l'affichage des lignes traitées.

**Commandes essentielles de Sed :**

**p** affiche la ligne sélectionnée

**d** ignore la ligne sélectionnée

**n** affiche la ligne et passe à la suivante

**s** recherche un motif et le remplace

```
sed -ne '1,/^\$/p' < mail.txt
```

extrait l'en-tête d'un mail.

**sh** Shell Bourne.

**sleep** Attend une durée déterminée.

```
sleep 14
```

(14 secondes)

```
sleep 3m
```

(3 minutes)

**sort** Trie les lignes d'un fichier texte.

- b ignorer les blancs en début de ligne,

- f ignorer les différences majuscules/minuscules,

- r inverser l'ordre du tri.

**split** Découpe un fichier en différentes partie.

- l **n** en fichiers de **n** lignes,

- b **n** en fichiers de taille **n**.

```
split -b 1M big-file part-
```

**ssh** Connexion sécurisée sur un système distant.

```
ssh user@hostname
```

```
ssh -l user hostname /usr/bin/command
```

**strings** Cherche les chaînes Ascii dans un fichier

**stty** Configuration du terminal.

- a affiche la configuration en cours,

**sane** revient en configuration normale,

- echo pas d'écho des caractères frappés,

- icanon min 0 time 1

lecture des caractères à la volée.

**su** Exécute un shell avec un UID et un GID différents.

- exécute un shell de login.

**sum** Somme de contrôle, et nombre de blocs.

**tac** Concatène et affiche des fichiers à l'envers.

**tail** Affiche la fin d'un fichier.

-**num** affiche le nombre de lignes indiqué,

-**f** affiche en continu les modifications.

| `tail -f /var/log/messages`

**tar** Utilitaire de gestion d'archives.

**f** **fic** nom de l'archive,

**c** crée une archive,

**t** affiche le contenu d'une archive,

**x** extrait le contenu d'une archive,

**z** invoque Gnu gzip pour les (dé)compressions,

**j** invoque bzip2 pour les (dé)compressions,

-**v** mode volubile.

| `tar cjf save.tar.gz /home/usera/*`

| `tar xf appli-1.10.tgz`

**tee** Copie entrée sur sortie standard et dans un fichier.

-**a** ajout en fin de fichier sans écrasement.

| `... | tee file1.log | ...`

**telnet** Connexion sur un système distant.

(préférer ssh)

**test** Type d'un fichier, ou comparaison de valeurs.

Synonyme de la commande shell [ ... ].

| `if test -f ${file} ; then`

**time** Chronomètre une commande simple.

**touch** Modifie l'horodatage d'un fichier.

-**t** **MMJJhhmm** utilise l'horodatage indiqué,

-**r** **fichier** utilise l'horodatage du fichier,

**tr** Transpose ou élimine des caractères.

| `tr 'âäåçêëîïôöù' 'aaaceeeiioou' | ...`

**true** Réussit à ne rien faire...

| `while true; do ...`

**tty** Affiche le nom du terminal de l'entrée standard.

**uname** Affiche des informations sur le système.

-**m** type de matériel,

-**n** nom d'hôte,

-**r** version du système,

-**s** système d'exploitation,

-**a** toutes les informations.

**uncompress** Décompression de fichier .Z.

**unexpand** Convertit les espaces en tabulation.

(voir aussi expand)

**uniq** Ôte les lignes dupliquées d'un fichier trié.

-**u** n'affiche que les lignes uniques,

-**d** n'affiche que les lignes dupliquées,

-**c** affiche le nombre d'occurrences des lignes.

**unix2dos** Conversion de textes du format Unix vers Dos.

**unxz** Décompresse un fichier .xz.

**unzip** Décompresse un fichier .zip.

**uptime** Temps fonctionnement et charge système.

**users** Nom des utilisateurs connectés.

**uudecode** Décode un fichier .uu.

**uuencode** Code un fichier binaire en Ascii.

**vi** Éditeur interactif

Version graphique : gvim

**wait** Attend la fin d'un processus.

| `$ ./commande &`

| `[1] 2927 ./commande`

| `$ wait 2927`

| `[1]+ Done ./commande`

| `$`

**wc** Nombres de caractères, mots et lignes d'un fichier.

**whereis** Recherche les fichiers exécutables, les sources et les pages de manuel d'une commande.

**which** Affiche le chemin d'accès des commandes.

**who** Montre qui est connecté.

**whoami** Affiche notre UID effectif.

**xargs** Construit et exécute une ligne de commande.

| `find . -name "*.c" | xargs grep "init()"`

**xz** Comprime un fichier

**yacc** Générateur d'analyseur syntaxique.

**yes** Affiche indéfiniment une chaîne

(par défaut 'y')

| `yes | rm -r /var/old-backup/`

**zcat** Affiche le contenu d'un fichier compressé.

**zip** Comprime un fichier.

## Expressions rationnelles

**grep, sed, find -regex**

utilisent des expressions rationnelles *simples*.

**grep -e, awk, perl**

utilisent des expressions rationnelles *étendues*.

### Éléments communs

\ supprime la signification des caractères spéciaux,

| `price=\$25`

. remplace n'importe quel caractère,

| `g.n.rique`

^ représente le début de chaîne,

\$ représente la fin de chaîne,

| `^$` (ligne vide)

\* indique zéro, une ou plusieurs occurrences,

[ ] représente une liste, un intervalle ou une classe,

| `[eéêëë]`

| `[0-9]`

| `[[:upper:]]`

\i contenu du i<sup>ème</sup> regroupement entre parenthèses.

### Classes de caractères

**alpha** caractères alphabétiques,

**digit** chiffres décimaux,

**xdigit** chiffres hexadécimaux,

**alnum** caractères alphanumériques,

**lower** minuscules,

**upper** majuscules,

**blank** caractères blancs,

**space** caractères séparateurs,

**punct** signes de ponctuation,

**graph** symboles visibles,

**print** symboles visibles ou blancs,

**cntrl** caractères de contrôle d'impression.

### Éléments des expressions rationnelles étendues

| représente une alternative,

| `Y|y`

+ réclame une ou plusieurs occurrences,

? réclame zéro ou une occurrence,

| `[+-]?[[:digit:]]+`

{ } réclament un certain nombre de répétitions,

( ) regroupent des éléments.

### Équivalences pour les expr. rationnelles simples

\| correspond au | des expressions étendues,

\+ correspond au + des expressions étendues,

\? correspond au ? des expressions étendues,

\{ \} correspondent aux {} des expressions étendues,

\( \) correspondent aux () des expressions étendues.



# Aide-mémoire de la programmation shell

Christophe Blaess pour Logilin

## Évaluation des expressions

| *variable*=*value*  
affectation de *variable*. Pas d'espace autour du signe égal !  
| *array*[*index*]=*value*  
affectation du rang *index* du tableau *array*.  
| \${*variable*}  
remplacé par le contenu de la *variable*.  
| \${*array*[*index*]}  
remplacé par le contenu du rang *index* du tableau *array*.  
| \${*variable*-*value*}  
remplacé par la valeur si la *variable* n'est pas définie.  
| \${*variable*=*value*}  
affectation de la *variable* si elle n'est pas définie.  
| \${*variable*?*value*}  
remplacé par la valeur si la *variable* est vide.  
| \${#*variable*}  
est remplacé par la longueur du contenu de la *variable*.  
| \${*variable*#*pattern*}  
est remplacé par le contenu de la *variable* privé du plus court  
préfixe correspondant au *pattern*.  
| \${*variable*%*pattern*}  
est remplacé par le contenu de la *variable* privé du plus court  
suffixe correspondant au *pattern*.  
| \${*variable*##*pattern*} \${*variable*%*pattern*}  
suppression du préfixe ou suffixe le plus long possible.  
| ~*user*/  
remplacé par le répertoire personnel de l'utilisateur *user*.  
| ab{c,d,e}fg  
est développé en abcfg abdfg abefg  
| \$(*command*)  
remplacé par la sortie standard de la *command*,  
| \$((*expression*))  
remplacé par le résultat de l'évaluation arithmétique entière  
de l'*expression*.

## Protection des caractères spéciaux

| "\$var1 \$var2"  
garde la chaîne en remplaçant les variables par leurs valeurs,  
| '\$var1 \$var2'  
garde la chaîne inchangée (pas de remplacement),  
| \\$var  
le *backslash* protège le \$. Il n'est plus considéré comme  
caractère spécial (pas de remplacement).

## Structures de contrôle

### Boucles

```
| while cmd_1 ; do  
|     commands  
| done
```

Répète les commandes tant que *cmd\_1* renvoie vrai (0).

```
| until cmd_1 ; do  
|     commands  
| done
```

Répète les commandes tant que *cmd\_1* renvoie faux.

```
| for variable in list ; do  
|     commands  
| done
```

Répète les commandes en remplissant la variable avec les  
éléments successifs de *list*.  
break  
sort directement d'une boucle *for*, *while* ou *until*.  
continue  
passe à l'itération suivante de la boucle.

### Tests

```
| if cmd_1 ; then  
|     cmd_2  
| elif cmd_3 ; then  
|     cmd_4  
| else  
|     cmd_5  
| fi
```

Si *cmd\_1* renvoie vrai exécute *cmd\_2*. Sinon si *cmd\_3*  
renvoie vrai, exécute *cmd\_4*, sinon exécute *cmd\_5*.

```
| case expression in  
|     pattern1 ) cmd_1 ;;  
|     pattern2 | pattern3 ) cmd_2 ;;  
|     * ) cmd_default ;;  
| esac
```

Si l'*expression* peut correspondre au *pattern1*, exécute  
*cmd\_1*, sinon si elle correspond au *pattern2* ou *pattern3*,  
exécute *cmd\_2*, sinon exécute *cmd\_default*.

## Fonctions

```
| function_1 ()  
| {  
|     commandes...  
| }
```

définit la *function\_1*.  
| *function\_1* *value\_1* *value\_2*...  
invocation de *function\_1*; dans la fonction les arguments  
sont dans \$1, \$2... et leur nombre dans \$#.  
| local *variable*  
déclare une variable locale à la fonction  
| return *value*  
termine la fonction en renvoyant la valeur en retour.

## Motifs du shell

\*  
n'importe quelle chaîne de caractères (même vide),  
?  
n'importe quel caractère,  
| \\* \? \\  
Caractères \*, ?, \,  
| [liste]  
Caractères l, i, s, t, e  
| [b-e]  
Caractères b, c, d, e  
| [^liste]  
N'importe quel caractère hors de la liste

## Redirections

```
| commande < fichier  
entrée standard depuis fichier,  
| commande > fichier  
sortie standard vers fichier,  
| commande >> fichier  
sortie standard ajoutée en fin de fichier,  
| commande 2> fichier  
sortie d'erreur vers fichier,  
| commande 2>&1  
sortie d'erreur identique à sortie standard,  
| commande <<- LABEL  
lignes à envoyer  
vers l'entrée standard  
de la commande  
| LABEL  
document en ligne envoyé vers l'entrée standard.
```

## Exécution des commandes

### Ligne shebang

```
| #! /bin/sh
```

en tout début de script.

### Pipeline

| *command* | *command* | *command*  
sortie standard injectée dans l'entrée de la suivante

### Liste de pipelines

| *pipeline* ; *pipeline*  
(exécutions séquentielles)  
| *pipeline* & *pipeline*  
(exécutions parallèles)  
| *pipeline* && *pipeline*  
(exécutions dépendantes)  
| *pipeline* || *pipeline*  
(exécutions alternatives)

### Commandes composées

| { *list of pipelines* }  
(regroupement de commandes)  
| ( *list of pipelines* )  
(sous-shell)

## Commandes internes essentielles

### echo

| *echo* arguments  
affiche les arguments séparés par des espaces.  
-n supprime le saut de ligne final  
-e interprète les séquences spéciales.

### read

| *read* variables...  
remplit les *variables* avec les mots successifs de la ligne lue (séparateur : contenu de la variable IFS).  
Dernière variable reçoit tout ce qui reste. Par défaut, utilise variable *REPLY*. Renvoie faux en fin de fichier.

### exec

| *exec* *command*  
remplace le (script) shell en cours par la *commande*.  
| *exec* *redirections*  
applique les *redirections* indiquées au shell courant.

### source

| *source* *script*  
| . *script*  
interprète le *script* dans le shell en cours.

### exit

| *exit* *value*  
termine le (script) shell courant en renvoyant la valeur.

### test

| *test* *condition*  
| [ *condition* ]  
Laisser des espaces autour des crochets ! Renvoie une valeur vraie ou fausse suivant la condition.  
Comparaisons de valeurs numériques :  
-eq ... égale à ...  
-ne ... différente de ...  
-lt (-le) ... inférieure (ou égale) à ...  
-gt (-ge) ... supérieure (ou égale) à ...

Test sur les chaînes :

-n longueur non nulle  
-z longueur nulle.

Comparaisons de chaînes : =, !=, <, >

Tests sur les fichiers :

-a existence du fichier,  
-b périphérique mode bloc,  
-c périphérique caractère,  
-d répertoire,  
-f fichier normal,  
-g bit Set-GID validé,  
-G appartenant au groupe de l'utilisateur,  
-h lien symbolique,  
-k bit Sticky validé,  
-N modifié depuis la dernière lecture,

-O appartient à l'utilisateur,  
-p tube nommé (fifo),  
-r peut être lu,  
-s taille non-nulle,  
-S socket,  
-u bit Set-UID validé,  
-w peut être écrit,  
-x peut être exécuté.

Comparaisons de fichiers :

-ef ... même fichier physique que ...,  
-nt ... modifié plus récemment que ...,  
-ot ... modifié plus anciennement que ...

Test sur les descripteurs :

-t est un terminal

### cd

| *cd* *directory*  
change de répertoire de travail,  
*cd* - revient au répertoire précédent,  
*cd* revient au répertoire de connexion.

### pwd

affiche le répertoire de travail en cours.

### export

| *export* *variable*  
Transfère la *variable* du shell dans l'environnement qui sera transmis aux processus fils ultérieurs.

### env

affiche le contenu de l'environnement

### set

*set*  
affiche les variables du shell et l'environnement,  
| *set* *options*

configure des paramètres du shell :

-a exporter toutes les variables  
-u refuser les variables indéfinies  
-v afficher les lignes de commandes avant exécution  
-x afficher les développements avant exécution

### unset

| *unset* *variable*  
efface la *variable*.

### getopts

```
while getopts "ab:c" variable ; do
  case $variable in
    a) echo "opt° -a";;
    b) echo "opt° -b, arg. $OPTARG";;
    c) echo "opt° -c";;
    *) echo "opt° invalide"; exit 1;;
  esac
done

shift $((OPTIND - 1))
echo "arguments restants : "
echo "$@"
exit 0
```

Analyse la ligne de commande en fonction de la liste d'options. Si une option prend un argument (':' après sa lettre), il est dans *OPTARG*. Une fois les options lues, le rang du premier argument restant est dans *OPTIND*.

### shift

| *shift* *n*  
décale les arguments en ligne de commande de *n* rangs : \$0 reste inchangé, \$n+1 passe dans \$1, \$n+2 dans \$2, etc.

