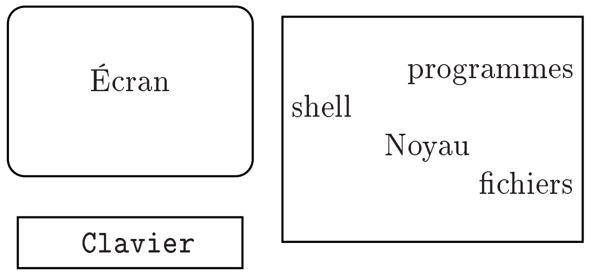
LA LIGNE DE COMMANDE



Les lignes de commande sont interprétées par le shell, et traduites en ordres transmis au noyau, et aux autres programmes.

Shells

csh, tcsh (C-shells à la Berkeley)
bash (héritier GNU du Bourne shell), ksh, ..., zsh.
Attention, sh n'est plus un shell précis, mais le shell par défaut : sous Linux, en général un bash restreint.
Un bash plus conforme : `bash --posix`
Initialisations dans `/etc/profile`, ... , `~/.bashrc`.

Édition de la ligne

On modifie les options de readline dans `~/.inputrc`, par exemple édition style emacs :

`C-k, C-u, C-w, C-y, C-a, C-e, C-l`

Suppr ← et les flèches.

Instruction élémentaire

`commande` [*options*] [arguments] Entrée

Commandes usuelles

Aide

```
man commande, info commande,
commande -h, --help, -help,
         -?, ... --pasglop
```

Commande `history`, raccourcis en `!`

Fichiers et répertoires

```
ls, cp, mv, rm,
ln,
pwd, cd, mkdir, rmdir,
chmod, chown,
umask, ulimit
```

Chercher un fichier

```
locate, find
```

Contenu

```
file, wc,
cat, more, less,
grep, diff,
head, tail, sort, uniq, cut,
tr, recode,

sed, awk, ...
```

Archives et compression

```
gzip, gunzip, zcat,
bzip2, bunzip2, bzcat

tar
zip, unzip
```

Utilisateurs

```
passwd, who, w,
id, groups,
```

Système

```
date, touch,
uptime, uname,
du, df, free,
```

Impression

```
lpr, lpq, lprm,
a2ps, enscript, ...
```

Contrôle, parallélisme

```
C-c, C-d, C-z

fg, bg,
ps, top,
kill, killall

 bidule & chose &
jobs, %b
wait
```

Réseau

```
ping, host, dig, whois

wget $URL
rsync [options] source/ cible/

telnet ailleurs.net [ port ]
ssh ma.becane
scp fichier toto@machine:repert/
```

Environnement

```
$PATH, $CDPATH, $LINES, $COLUMNS,
$TERM, $DISPLAY,
$LANG, $LC_ALL, $LC_TIME, ...

echo, printf,
type, which,
set, unset, export,
alias, unalias
```

Expansions

```
~/ ou $HOME/ => /home/moi/
~toto        => /home/toto

$TRUC, f*a[l-z]*.??, fich.{tex,aux,log}

$(sous-shell), `sous-shell`
$(( 2 * 17 + 42 )), $[ 2 + 2 ]
```

Redirections

0 = entrée, 1 = sortie normale, 2 = erreurs.
Attention à l'ordre pour `2>&1`, préférer `&>`.

```
<entree, >sortie, 2>>erreurs, &>le_tout
```

```
 cat <<BALISE
bla bla
...
BALISE
```

Structures

Groupes et pipes

```
un | deux | tee sortie_deux | trois
{ comm1 ; comm2 ; }
( a ; b ; c )
```

Tests

```
test && echo oui || echo non
[ -f fichier ], [ $COLUMNS -gt 80 ],
[[ a < z ]]
```

le code de sortie est dans `$?`.

Alternatives

```
if liste
then
        echo oui
else
        echo non
fi
```

```
case $var in
     a*)
     ...
     ;;
     b*|cc*)
     ...
     ;;
     *)
     defaut
esac
```

Boucles

```
for f in *.jpg ; do action $f ; done

for(( n=0 ; n<42 ; n++ )) # bash 2.x
do
        echo $n ; action
done

while encore ; do actions ; done
until OK     ; do actions ; done

break, continue
```

Fonction, sous-programme

```
nom_de_fonction () {
        # contenu
}

exit, exec
source fichier_de_commandes
./fichier_script
```

Un fichier script *exécutable* commence par la ligne « shebang » `#!/bin/bash` ou `#!/bin/sh`. Les arguments sont dans `$*` ou `$@`, et leur nombre dans `$#`. Voir aussi `shift`, etc.

Liens

```
http://www.eleves.ens.fr/tuteurs/unix/
http://www-inf.enst.fr/~dax/polys/shell/
news:fr.comp.os.unix
```