

Unix: Scripts Shell

ASR1 Système

Département d'informatique
IUT Bordeaux 1

Novembre 2010

Processus

- 1 Définitions
- 2 Table des processus
- 3 kill
- 4 Pilotage des processus
- 5 Une application
 - Un exemple de service
 - Le code principal

Processus

Définition : Un processus est un programme **en cours d'exécution**

Un programme lancé depuis le shell peut

- **s'exécuter en avant-plan** (foreground)
- **s'exécuter en arrière-plan** (background)
- être **stoppé**

Manipulation

- **lancer** une commande en **avant-plan** : `xclock`
- **stopper** la commande en avant-plan : `CTRL-Z`
- **relancer** la commande stoppée **en avant-plan** : `fg`
- **relancer** la commande stoppée **en arrière-plan** : `bg`
- **lancer** une commande **en arrière-plan** : `xclock &`

NB : Si plusieurs commandes tournent en arrière-plan : `jobs`, `fg %n...`

La table des processus

Pour visualiser les processus `ps`, `top`

Les processus sont identifié par un **PID** (*Process Identifier*)

Exemple

```
$ ps
  PID TTY          TIME CMD
 4056 pts/1    00:00:00 bash
 4236 pts/1    00:00:07 xpdf.bin
 4243 pts/1    00:00:10 emacs
 4471 pts/1    00:00:00 xterm
 4613 pts/1    00:00:00 ps
```

- `ps` sans option montre les processus issus du shell
- options intéressantes de `ps` : `axule...`
- voir aussi `pstree`

La commande kill

- La commande `kill` envoie un **signal** à des processus
- **Syntaxe** : `kill [-signal] PID ...`

Exemple

```
$ xclock &
```

```
$ ps
```

PID	TTY	TIME	CMD
4056	pts/1	00:00:00	bash
4734	pts/1	00:00:00	xclock
4739	pts/1	00:00:00	ps

```
$ kill -TERM 4734
```

- le signal `TERM` (9) termine le programme (utilisé par défaut)
- le signal `STOP` arrête un processus
- le signal `CONT` le relance
- `kill -1` affiche la liste des signaux

Pilotage des processus

- commande `&` lance une commande en arrière-plan
- la variable `$!` contient son numéro de processus
- la variable `$$` = numero du shell courant

Exemple

```
mplayer funny-music.mp3 >/dev/null &  
music=$!
```

```
# sauvegardes  
tar czf .....
```

```
# arrêter la musique à la fin  
kill -9 $music
```

Wait

La commande `wait nb_proc` attend un processus

Exemple

```
mplayer funny-music.mp3 >/dev/null &  
music=$!  
  
# sauvegardes en parallèle  
tar czf archive1.tar ..... &  
svgd1=$!  
tar czf archive2.tar ..... &  
svgd2=$!  
  
wait $svgd1  
wait $svgd2  
  
kill -9 $music # arrête la musique
```

Un exemple de service

Une commande pour faire apparaître / disparaître une pendule sur le bureau

Usage

- `./pendule.sh start`
- `./pendule.sh stop`
- `./pendule.sh usage`
- `./pendule.sh restart`

Un exemple de service

Constantes et Fonctions

```
prog=/usr/bin/xclock
pid_file=/tmp/$USER.pid

function do_start {
    $prog &
    echo $! > $pid_file
}

function do_stop {
    kill -9 $(cat $pid_file)
}

function usage {
    echo "usage: pendule {start|
        stop|restart|usage}"
}
```

Code principal

```
...
case "$1" in
start)
    do_start ;;
stop)
    do_stop ;;
restart)
    do_stop
    do_start ;;
usage)
    print_usage ;;
*)
    print_usage
    exit 1
esac
```

Utilise 3 fonctions : do_start, do_stop et print_usage