# Séance de TP N°6
## `06.signaux` – Traitement de signaux

Créez votre répertoire de TP *~/l3i6/06.signaux* et placez-vous dessus. Récupérez dans votre répertoire de TP le fichier *Fourniture06.tgz* sur le site de cet enseignement[1] et déballez-le. Ce répertoire contient :

- un script **configure** et son ***Makefile*** générique ***Makefile.gen*** ;
- ***sigusr.c*** ;
- ***printWindowSize.c*** ;

Les programmes à construire sont les suivants :

- ***sigusr2.c***,
- ***signauxDivers.c*** ;
- ***horloge.c***,

# Démonstrations

## *Démonstration de traitement des signaux SIGUSR et SIGTERM*

On considère le programme ***sigusr.c***. Compilez-le et effectuez l'exécution suivante en arrière-plan :

```
$ sigusr &
[4] 32412
```

Envoyer alors à ce programme, deux fois le signal *SIGUSR1* et deux fois le signal *SIGUSR2*, puis terminez le programme par le signal *SIGTERM* :

```
$ kill -USR1 32412 (ou %4)
$ kill -USR1 32412
$ kill -USR2 32412
$ kill -USR2 32412
$ kill -TERM 32412
```

Étudiez le source et interprétez les résultats.

## *Programme de démonstration du traitement du signal SIGWINCH*

On considère le programme ***printWindowSize.c***. Compilez-le et effectuez les exécutions suivantes qui d'abord échouent, sauf la dernière en avant-plan, la seule qui réussit :

```
$ printWindowSize > Z
$ printWindowSize < Z
$ printWindowSize &
$ printWindowSize
[4] 412
```

Modifiez plusieurs fois, avec la souris, la taille de la fenêtre **xterm**. Terminez l'exécution par la touche **Ctrl c** (signal *SIGINT*). Étudiez le source et interprétez les résultats.

[1] http://deptinfo.unice.fr/~lahire/enseignement/SYSL3

# Traitement du signal SIGTERM

Écrire le programme **sigusr2** qui modifie le programme **sigusr** fourni pour :

- qu'il ajoute, pour le signal *SIGTERM*, un traitant qui affichera un message de refus de terminaison et poursuivra l'exécution du processus.
- qu'il tente de faire de même pour le signal *SIGKILL*.

Exécutez le programme **sigusr2** en effectuant les expériences suivantes :

- exécution en arrière-plan et envoi du signal *SIGUSR1*, en utilisant la commande *kill(1)* ;
- lancement d'une deuxième fenêtre **xterm** en arrière-plan, puis lancement du programme **sigusr2** en avant-plan, dans l'une des fenêtres. Alors on enverra le signal *SIGUSR1* depuis l'autre fenêtre avec les commandes *ps -x* et *kill*.

Interprétez les résultats. Ensuite, testez l'envoi des signaux suivants au programme **sigusr2**, quel que soit son plan de travail :

```
$ kill -TERM numéro_du_ processus
$ kill numéro_du_ processus
$ kill -KILL numéro_du_ processus
```

Interprétez les résultats.

# Traitement de signaux divers

Écrire un programme ***signauxDivers.c*** qui traite les signaux suivants comme indiqué (voir conseils plus loin pour les écritures) :

- *INT* (touche **Ctrl c** ) : envoi d'un message d'identification du signal et terminaison par exit.
- *TSTP* (touche **Ctrl z** ) : envoi d'un message d'identification du signal et arrêt du processus par *STOP*,
- *CONT* (commande **fg**) : nettoyage de l'écran, envoi du message "reprise" en haut et à droite de l'écran et positionnement du curseur sur la 1ère colonne de la 2ème ligne. On utilisera les séquences suivantes pour effacer l'écran et positionner le curseur sur l'écran :

  ```
  #define CLEAR_SCREEN "\033[;H\033[2J"
  #define MOVE_CURSOR "\033[%d;%dH"
  ```

  Pour placer le curseur sur la 3ème ligne et la 4ème colonne, il suffit d'émettre au terminal la séquence "*\033[3;4H*" . Pour afficher le message "reprise" en haut et à droite, on tiendra compte de la longueur de ce message et de la largeur de la fenêtre, obtenue par la technique indiquée dans le programme **printWindowSize** (utilisation d'un **ioctl**).
- *WINCH* (action avec la souris) : message d'identification du signal et nouvelles dimensions de la fenêtre,
- *QUIT* (touche **Ctrl **) : message "je m'endors" (pendant 10 secondes),
- *SIGUSR1*, *SIGUSR2* (commande *kill(1)*) : réveil du processus, désarmement du *timer*, message d'identification.

Ce programme aura la même structure que le programme **sigusr**, après avoir armé les traitants des signaux contrôlés, il s'arrêtera dans une boucle d'attente avec la primitive *pause(3)*. Pour écrire sur le terminal, quelque soit son plan de travail ou d'éventuelles redirections, on ignorera le signal *SIGTTOU* et on écrira

sur *stderr* ou un fichier ouvert par */dev/tty*. Dans les deux cas, les écritures ne devront pas être tamponnées, soit en utilisant *write(2)*, soit en utilisant des *fprint(tty,...)* et *setvbuf(tty, NULL, _IONBF, 0)*. Pour tester ce programme, on aura intérêt à lancer d'abord une autre fenêtre **xterm** pour envoyer les signaux *SIGUSR1* et *SIGUSR2* par la commande **kill**.

# Pour les plus courageux: horloge à périodicité d'affichage variable

Écrire un programme horloge qui :

- efface l'écran et affiche de manière centrée, au milieu de l'écran : l'heure sous le format *HH:MM:SS.CC* (avec le centièmes de secondes). Le réaffichage se fera au départ toutes les *INTERV* secondes, avec INTERV == 1.
- traite les signaux *SIGINT* (**Ctrl c** ), *SIGTSTP* ( **Ctrl z** ) et *SIGQUIT* ( **Ctrl ** ) de la manière suivante :
    - *SIGINT* double l'intervalle *INTERV* à chaque fois ;
    - *SIGTSTP* divise par 2 l'intervalle *INTERV* à chaque fois ;
    - *SIGQUIT* termine le programme.