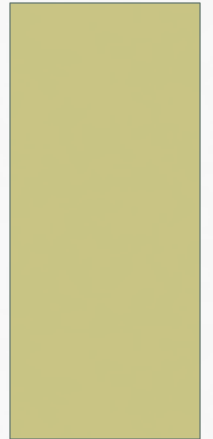


TP 6 – PART 2

SYSTEMES D'EXPLOITATION



1ERE PARTIE

- Exemple:

```
pid_t pid = fork();  
//in child  
if (pid==0){  
    //execute job with given parameters for example with execvp  
    ...  
}  
//in parent (shell)  
else if (pid>0){  
    foreground_job = pid  
    int status;  
    waitpid(pid, &status, 0);  
    //print in shell that job exited  
    ...  
    foreground_job = 0  
}
```

TÂCHE DE FOND

```
pid_t pid = fork();  
//in child  
if (pid==0){  
    // ** Redirect stdin to /dev /null **  
    ...  
    //execute job with given parameters  
    ...  
}  
//in parent (shell)  
else if (pid>0){  
    background_job = pid;  
}
```

SIGNAUX

- SIGTERM et SIGQUIT seront ignorés
- SIGINT sera redirigé vers le processus principal
- SIGHUP sera redirigé vers TOUS les processus/jobs du shell et terminera le shell

- Le shell gère les signaux avec sigaction:

```
sa.sa_handler = functionName or SIG_IGN;  
sa.mask = mask; //if needed to make setting the flags easier.  
sa.sa_flags = SA_RESTART|flag1|...;  
sigaction(signal, &sa, NULL);
```

GESTION DES ENFANTS

- Redirection des signaux avec `kill(pid, signal)`
 - Si le shell termine (SIGHUP) il doit rediriger le signal aux enfants!
- Evitement des zombies:
 - Signal SIGCHLD **avec flag SA_SIGINFO**
 - `sa.sa_sigaction`:

```
static void child_process_signal(int signum, siginfo_t *siginfo, void*
unused) {
    int pid_sig = siginfo->si_pid;
    if (pid_sig==background_job){
        //What do we do here??
    }
}
```

GESTION DES ENFANTS

- Redirection des signaux avec `kill(pid, signal)`
- *Evitement des zombies:*
 - Signal `SIGCHLD` **avec flag `SA_SIGINFO`**
 - `sa.sa_sigaction:`

```
static void child_process_signal(int signum, siginfo_t *siginfo, void*
unused) {
    int pid_sig = siginfo->si_pid;
    if (pid_sig==background_job){
        int status;
        waitpid(pid_sig, &status, 0);
        background_job = 0;
    }
}
```

BONUS

- Exécution de plusieurs jobs en tâche de fond
- Remettre un job en tâche principale
- Redirection des entrées/sorties (symbole '>')
- Implémentation de pipes (symbole '|')