

Programmation système

Projet : création d'un mini-Shell sous Ubuntu



Réalisé par : Mohammed Fergoug

Encadrée par : Mr Anouar Bouchal

Introduction :

Dans le cadre de la programmation système sous Unix, on a réalisé un projet de création d'un **mini-Shell sous Ubuntu**.

Ce mini-Shell a pour objectif de permettre à l'utilisateur d'exécuter des commandes Unix classiques depuis un programme C.

Le programme repose sur l'utilisation des appels système **fork ()**, **execvp ()**, et **waitpid ()**, et propose un prompt personnalisé permettant de saisir et exécuter des commandes jusqu'à la saisie de la commande exit pour quitter le programme.

Objectifs du mini-Shell :

- Afficher un prompt personnalisé (mini-Shell>)
- Lire et analyser la commande saisie par l'utilisateur
- Exécuter la commande dans un processus fils
- Attendre la fin du processus fils avant de continuer
- Quitter le mini-Shell lorsqu'on tape exit

Environnement de travail :

- **Système d'exploitation** : Ubuntu (sur machine virtuelle VMware)
- **Langage utilisé** : C
- **Compilateur** : gcc
- **Éditeur de texte** : nano
- **Terminal Linux**

Développement du mini-Shell :

Le mini-Shell fonctionne en boucle :

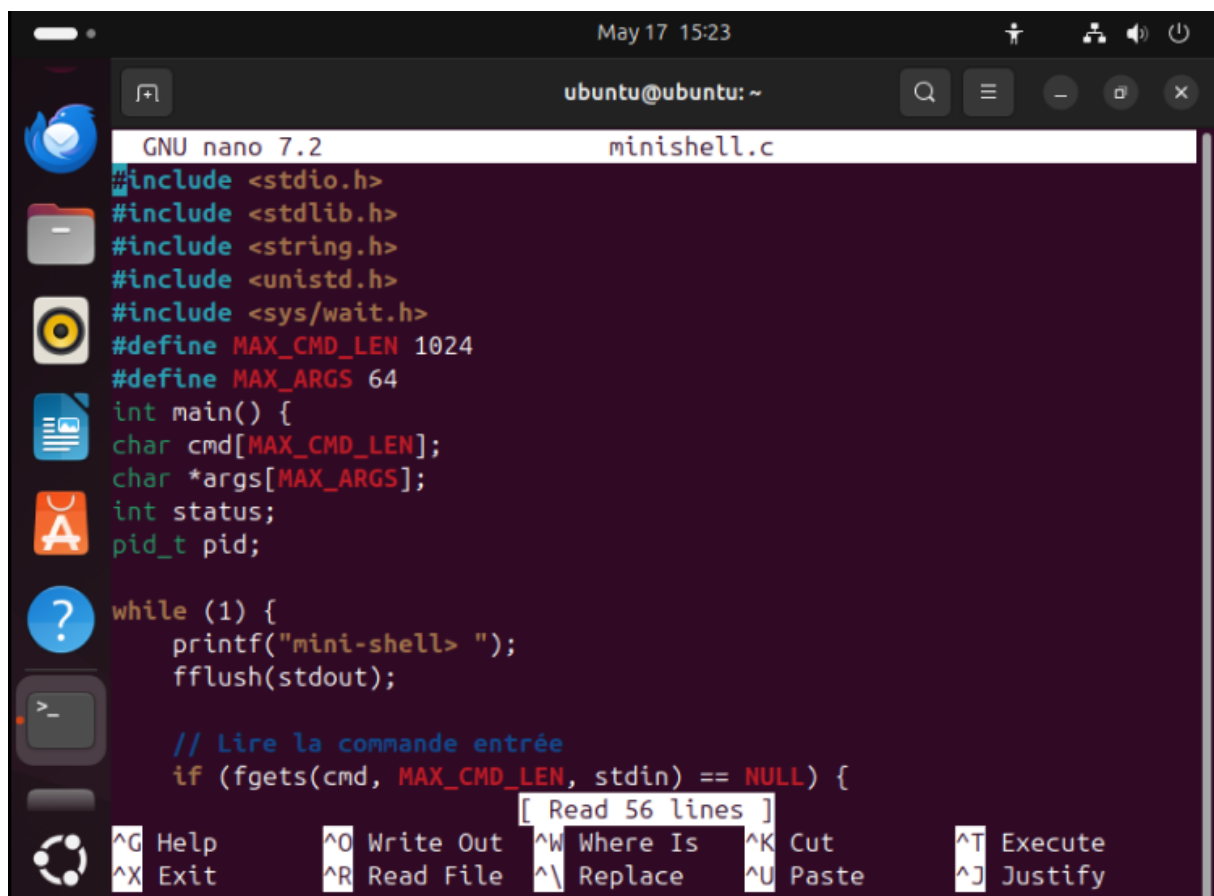
1. Affiche un prompt mini-Shell>
2. Lit la commande tapée par l'utilisateur

3. Si la commande est exit, le Shell se ferme
4. Sinon, la commande est découpée en arguments à l'aide de strtok ()
5. Un processus fils est créé avec fork ()
6. La commande est exécutée dans le fils avec execvp ()
7. Le processus père attend la fin du processus fils avec waitpid ()
8. La boucle recommence

Annexe

Le code pour la création de Shell :

```
ubuntu@ubuntu:~$ nano minishell.c
```



```
GNU nano 7.2 minishell.c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/wait.h>
#define MAX_CMD_LEN 1024
#define MAX_ARGS 64
int main() {
    char cmd[MAX_CMD_LEN];
    char *args[MAX_ARGS];
    int status;
    pid_t pid;

    while (1) {
        printf("mini-shell> ");
        fflush(stdout);

        // Lire la commande entrée
        if (fgets(cmd, MAX_CMD_LEN, stdin) == NULL) {

[ Read 56 lines ]

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify
```



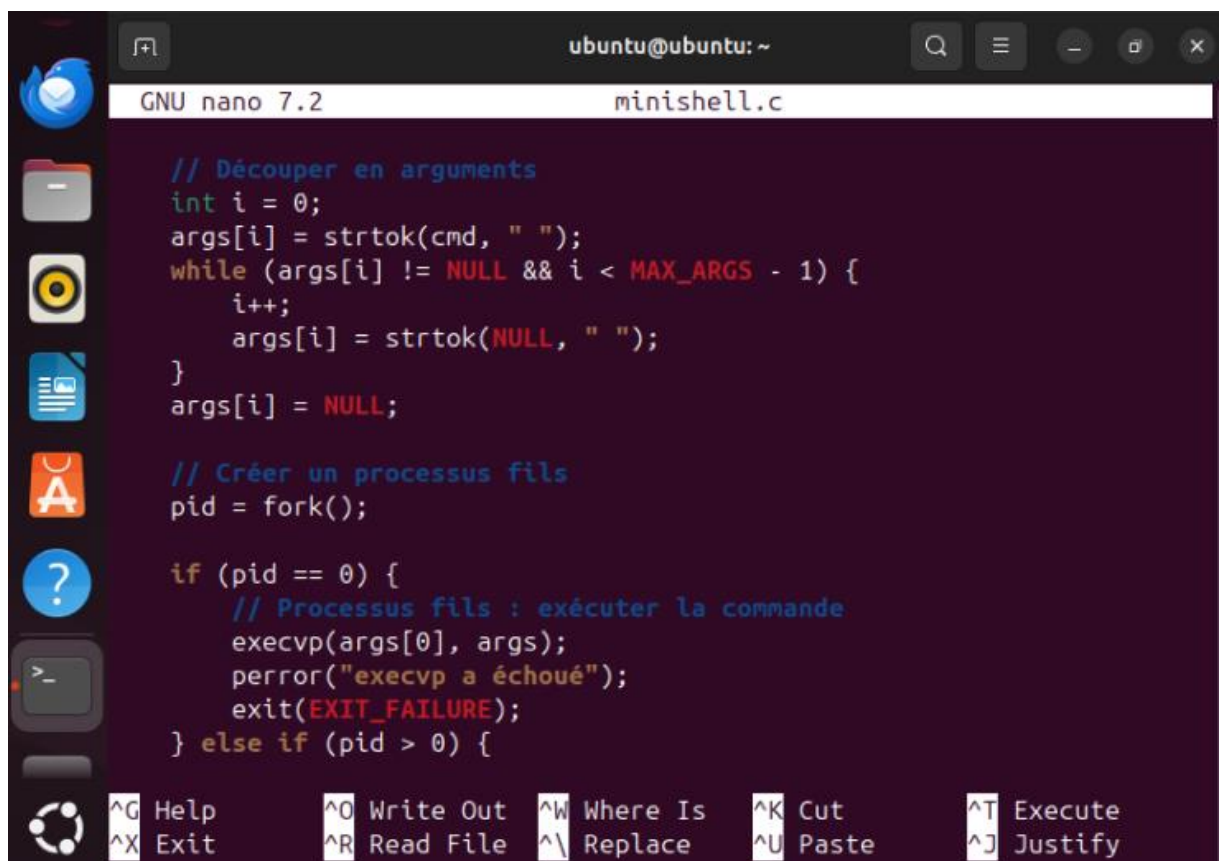
```
ubuntu@ubuntu: ~
GNU nano 7.2 minishell.c
pid_t pid;

while (1) {
    printf("mini-shell> ");
    fflush(stdout);

    // Lire la commande entrée
    if (fgets(cmd, MAX_CMD_LEN, stdin) == NULL) {
        break;
    }

    // Retirer le retour à la ligne
    cmd[strcspn(cmd, "\n")] = 0;

    // Quitter si "exit"
    if (strcmp(cmd, "exit") == 0) {
        break;
    }
}
```



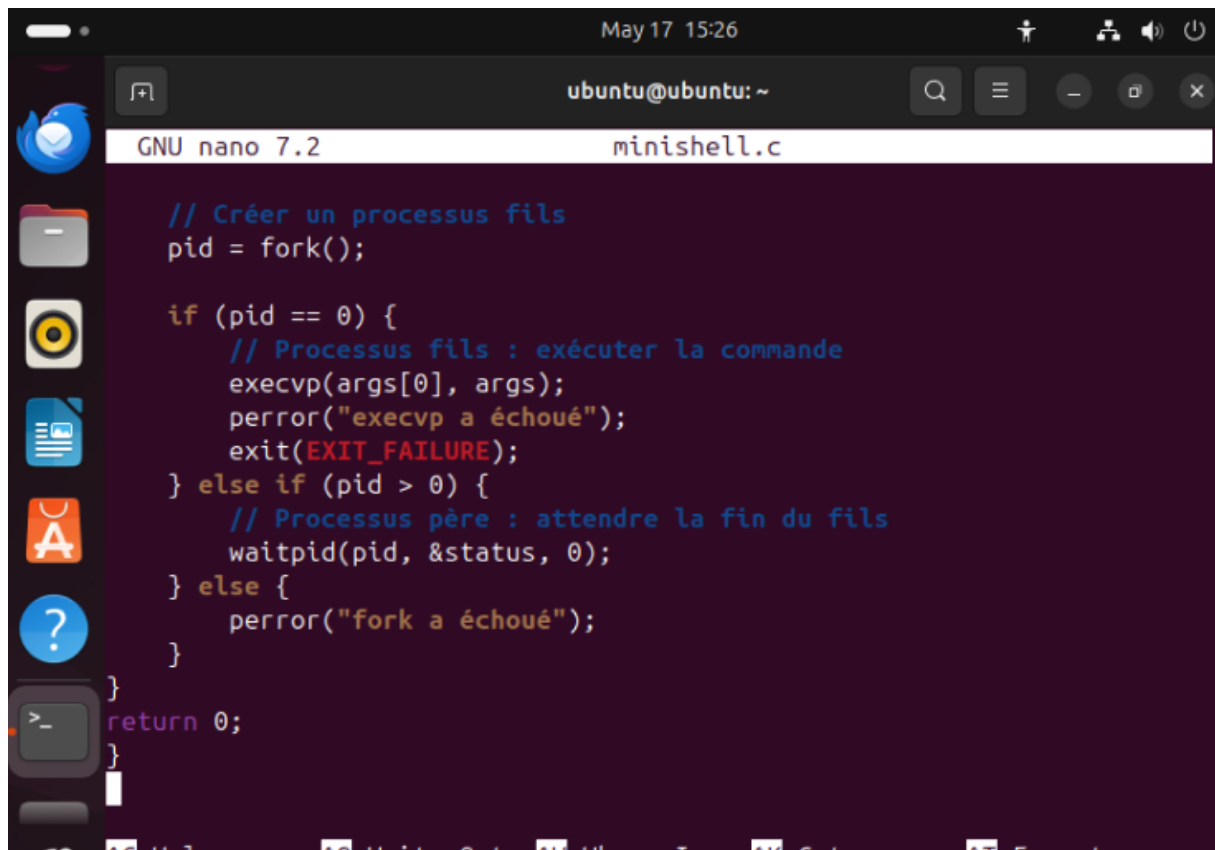
```

// Découper en arguments
int i = 0;
args[i] = strtok(cmd, " ");
while (args[i] != NULL && i < MAX_ARGS - 1) {
    i++;
    args[i] = strtok(NULL, " ");
}
args[i] = NULL;

// Créer un processus fils
pid = fork();

if (pid == 0) {
    // Processus fils : exécuter la commande
    execvp(args[0], args);
    perror("execvp a échoué");
    exit(EXIT_FAILURE);
} else if (pid > 0) {
}
```

Help Write Out Where Is Cut Execute
Exit Read File Replace Paste Justify



```
May 17 15:26
ubuntu@ubuntu: ~
GNU nano 7.2 minishell.c

// Créer un processus fils
pid = fork();

if (pid == 0) {
    // Processus fils : exécuter la commande
    execvp(args[0], args);
    perror("execvp a échoué");
    exit(EXIT_FAILURE);
} else if (pid > 0) {
    // Processus père : attendre la fin du fils
    waitpid(pid, &status, 0);
} else {
    perror("fork a échoué");
}

return 0;
}
```

❖ Avant la compilation, les outils ont été installés via :

```
ubuntu@ubuntu:~$ sudo apt update
```

```
ubuntu@ubuntu:~$ sudo apt install build-essential
```

Compilation : gcc

```
ubuntu@ubuntu:~$ gcc minishell.c -o minishell
```

Exécution :

```
ubuntu@ubuntu:~$ ./minishell
```

Résultat :

```
ubuntu@ubuntu:~$ ./minishell
mini-shell>
```

Exemples de commandes :

ls :

```
ubuntu@ubuntu:~$ ./minishell
mini-shell> ls
Desktop    Downloads  Pictures   Templates  bonjour    minishell.c
Documents  Music      Public     Videos     minishell  snap
mini-shell>
```

ls -l :

```
mini-shell> ls -l
total 24
drwxr-xr-x 2 ubuntu ubuntu    60 May 17 12:47 Desktop
drwxr-xr-x 2 ubuntu ubuntu    40 May 17 12:49 Documents
drwxr-xr-x 3 ubuntu ubuntu    60 May 17 14:49 Downloads
drwxr-xr-x 2 ubuntu ubuntu    40 May 17 12:49 Music
drwxr-xr-x 2 ubuntu ubuntu    40 May 17 12:49 Pictures
drwxr-xr-x 2 ubuntu ubuntu    40 May 17 12:49 Public
drwxr-xr-x 2 ubuntu ubuntu    40 May 17 12:49 Templates
drwxr-xr-x 2 ubuntu ubuntu    40 May 17 12:49 Videos
drwxrwxr-x 2 ubuntu ubuntu    40 May 17 16:41 bonjour
-rwxrwxr-x 1 ubuntu ubuntu 16528 May 17 16:43 minishell
-rw-rw-r-- 1 ubuntu ubuntu  1149 May 17 16:42 minishell.c
drwx----- 7 ubuntu ubuntu   140 May 17 15:00 snap
mini-shell>
```

Pwd :

```
mini-shell> pwd
/home/ubuntu
```

date:

```
mini-shell> date
Sat May 17 17:10:10 UTC 2025
mini-shell>
```

mkdir:

```
mini-shell> mkdir Mohammed_Fergoug
mini-shell> ls
Desktop    Mohammed_Fergoug  Public    bonjour    snap
Documents  Music             Templates minishell
Downloads  Pictures          Videos   minishell.c
mini-shell>
```

rm -rf :

```
mini-shell> rm -rf Mohammed_Fergoug
mini-shell> ls
Desktop    Downloads  Pictures   Templates bomjour    minishell.c
Documents  Music      Public     Videos   minishell  snap
mini-shell> 
```