Operating Systems

github/2018_os_hw1/32169198

Simple Shell

Fuentes Javier
Dankook University
October 2018

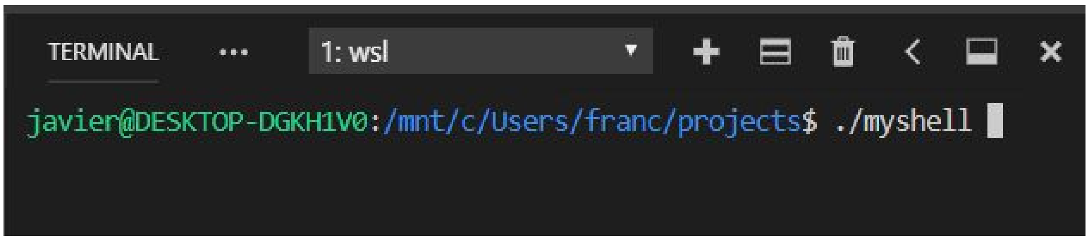
Introduction:

For this assignment we were asked to make a simple shell program utilising fork() wait(), execvp() among others.

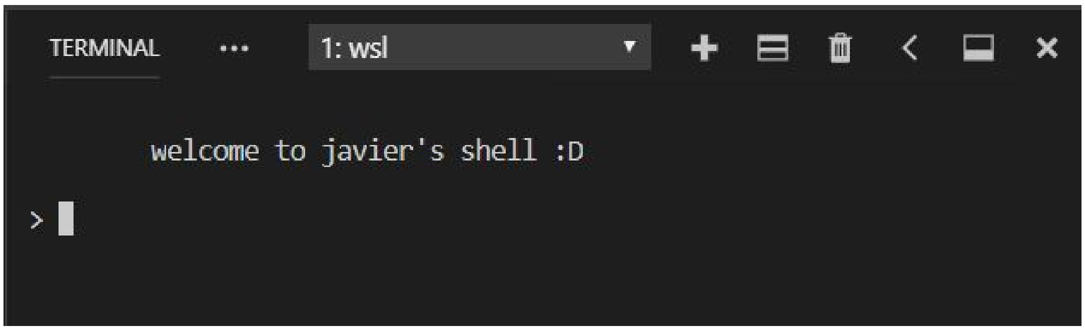
Functionality

Running the program.

In order to run the program you need to execute the program myshell
This is what the entry of the program looks like.



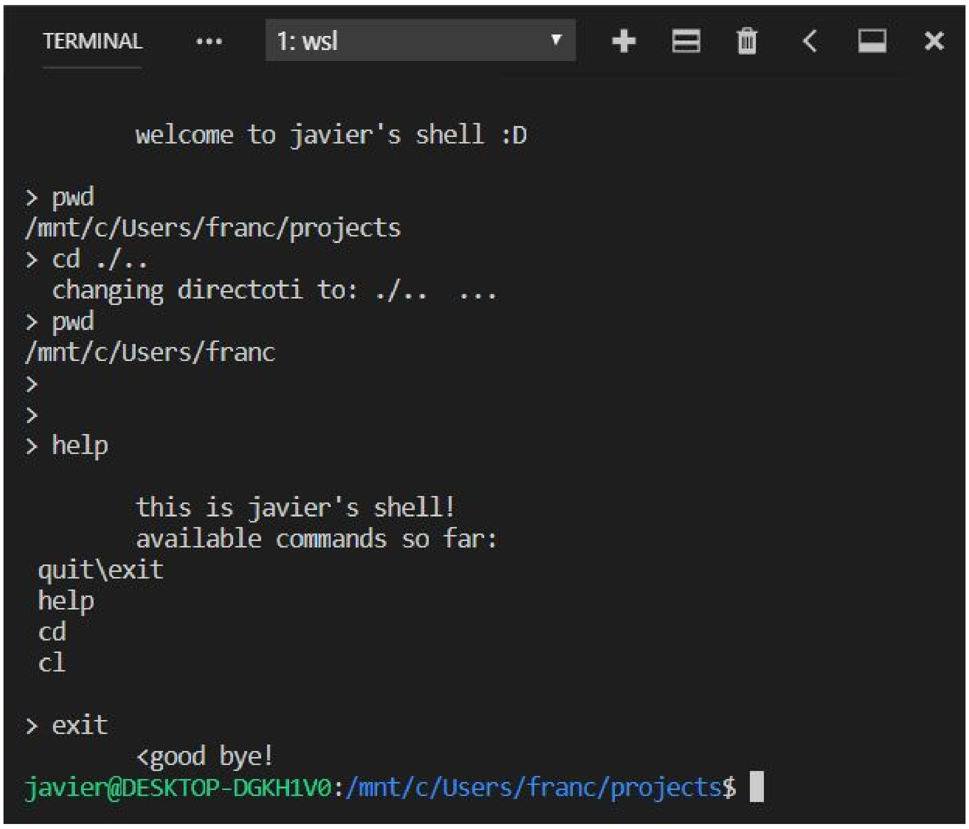
```
javier@DESKTOP-DGKH1V0:/mnt/c/Users/franc/projects$ ./myshell
```



```
        welcome to javier's shell :D
>
```

Working example.

In this example the command PWD was used to show the change in the working directory, while the command CD was used to change the directory.
Also the command help was used, which only prints the command names at the screen.



```
        welcome to javier's shell :D

> pwd
/mnt/c/Users/franc/projects
> cd ./..
  changing directoti to: ./..  ...
> pwd
/mnt/c/Users/franc
>
>
> help

        this is javier's shell!
        available commands so far:
 quit\exit
 help
 cd
 cl

> exit
        <good bye!
javier@DESKTOP-DGKH1V0:/mnt/c/Users/franc/projects$
```

The implementation of the commands CD and HELP can be checked at the code section of this document.

Personal ideas

For this homework I had more problems than I expected, at the beginning I was trying to run the programs by looking at the environment variable path and looking for such program name on each one of the directories, which gave me various errors at compile time.
At the end I check some examples online and learned how to use the command execvp which was a better approach, I also found some ways to simplify my code while doing that.

Code

```
#include <sys/wait.h>
#include <sys/types.h>
#include <unistd.h>
#include <stdlib.h>
```

```
#include <stdio.h>
#include <string.h>

//declarations
//my comands
int cm_cd(char **by_words);
int cm_help();
int cm_exit();
int cm_clear();
int other(char **by_words);
// my Functions
char **split_line(char *string);


int main(int argc, char **argv)
{
   int continar = 1;
   char *line;
   char **by_words;
   size_t size = 0;

   printf("\033[H\033[J");
   printf("\n\twelcome to javier's shell :D\n\n");

   while (continar)
   {
      printf("> ");
      getline(&line, &size, stdin);
      line[strlen(line)-1] = '\0';
      //printf("input<%s.\n", line); //validation
      by_words = split_line(line);

      char *first = by_words[0];

      if (first == NULL)
      {
         /*do nothing*/
      }

      else if (strcmp(first, "cl") == 0)
      {
         cm_clear();
```

```
        }

        else if (strcmp(first, "help") == 0)
        {
            cm_help();
        }

        else if ( (strcmp(first,"quit")==0) || (strcmp(first,"exit")==0) )
        {
            continar = 0;
            cm_exit();
        }

        else
        {
            //if it does not find it
            other(by_words);
        }
    }

    return 0;
}

// command: change directory.
int cm_cd(char **by_words)
{
    if (by_words[1] != NULL)
    {
        chdir(by_words[1]);
    }
    return 0;
}

// command: print help.
int cm_help()
{
    char *comands = "quit\\exit\n help\n cd\n cl\n";
    printf("\n\tthis is javier's shell!\n");
    printf("\tavailable commands so far:\n %s\n", comands);
    return 0;
}
```

```
// command: clear screen.
int cm_clear()
{
    printf("\033[H\033[J");
    return 0;
}

// command: terminates shell.
int cm_exit()
{
    printf("\t<good bye!\n");
    return 0;
}

// runs the command to the console
int other(char **by_words)
{
    pid_t pid;

    pid = fork();

    if (pid == 0) // Child
    {
        execvp(by_words[0], by_words);
    }

    else // Parent
    {
        wait(0);
    }

    return 0;
}

//really i just need the 2 first words
char **split_line(char *string)
{
    char **chain = malloc(3 * sizeof(char *));
    const char s[2] = " ";

    chain[0] = strtok(string, s); //first one
    chain[1] = strtok(NULL, s); //second one
    chain[2] = NULL; //las one is null

    //printf("1:%s\n2:%s\n3:%s\n", chain[0], chain[1], chain[2]); //control

    return chain;
}
```