



Projet Système

Find The Cat

Lucie BOUCHER
Lola MONTIGNIER

18 décembre 2022

0.1 Introduction

Ce projet consiste à coder un FindTheCat. Nous avons implémenté des options permettant de trouver des fichiers ou dossiers en fonctions de différents critères comme le nom, la date de dernière de modification ou encore la taille.

Notre binôme formé de Lucie BOUCHER et Lola MONTIGNIER. Ayant déjà travaillé ensemble sur plusieurs projets, nous nous faisons confiance. Cela nous a apporté un certain confort de travail. Afin d'être le plus efficace possible, nous nous sommes réparties les différentes fonctionnalités à implémenter, tout en communiquant très régulièrement afin de connaître l'avancée du projet.

0.2 Choix de conception

Au début du projet, nous implémentions les fonctionnalités une par une dans des fichiers différents. Rapidement, nous nous sommes aperçus qu'il serait plus pratique de réunir certaines fonctionnalités dans un même fichier. Nous avons aussi décidé de travailler sur une liste de structure qui est remplie initialement avec tout les fichiers et répertoires du chemin passé en paramètre du FindTheCat. Chaque fichier est ajouté à la liste avec une structure qui contient :

- Son nom
- Sa taille
- La date de sa dernière modification
- Les permissions d'accès
- Un pointeur vers la structure suivante

Afin de gérer, les différents flags passés en paramètre, nous avons décidé de construire une liste de flags où chaque élément à la structure suivante :

- Une option
- Un paramètre associé

Par exemple pour `"/. /ftc /home -date +4m"`, l'option stockée est date et son paramètre associé est +4m.

Cette structure de liste de flags nous a permis de gérer le "ET", ainsi plusieurs options et leur paramètres associés pouvaient être passé en ligne de commande.

Ces deux structures, nous ont permis d'avancer rapidement dans l'implémentation du projet car une fois la mise en place terminée chacune des options était implémentée sous le même format. Lors de l'exécution, on accède à la partie du code correspondant à l'option voulue puis si le fichier courant ne satisfait pas les exigences associées au flag, ce dernier est retiré de la liste des fichiers et le fichier suivant est étudié. A la fin, seuls les fichiers restants dans la liste sont affichés sur la sortie standard, c'est à dire ceux correspondants aux options passées en ligne de commande.

Notre code est composé de plusieurs fichiers .c et .h, nos fichiers respectent une sorte d'arborescence. La racine est le fichier util.c, où toutes les petites fonctions, utiles à

l'implémentation des options, sont réunies, le fichier `util.h`, associé, réalise les différents imports nécessaires. Ensuite, le fichier `list_of_files.c` contient toutes les fonctions en lien avec la gestion de la liste de fichiers :

- ajout d'un fichier à la liste
- suppression d'un fichier de la liste
- la taille de la liste
- affichage de la liste
- etc...

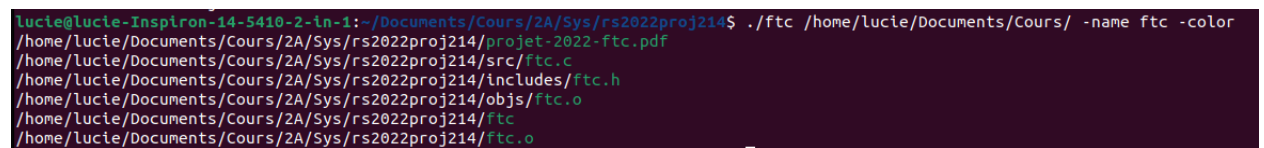
Ce fichier contient également tous les getters et setters liés à notre structure de données d'un fichier.

Nos choix de conceptions se sont révélés être avantageux car nous avons pu implémenter toutes les options sans grande difficulté et sans avoir à modifier à plusieurs reprises nos structures.

0.3 Extensions implémentées

Deux extensions ont été implémentées : `"-color"` et `"-perm"`.

L'accès à l'extension `"-color"` se fait lorsque l'on passe en ligne de commande nos options voulues ainsi que leurs paramètres associées, rajouter l'option `"-color"` va colorer en vert le nom des fichiers.



```
lucie@lucie-Inspiron-14-5410-2-1n-1:~/Documents/Cours/2A/Sys/rs2022proj214$ ./ftc /home/lucie/Documents/Cours/ -name ftc -color
/home/lucie/Documents/Cours/2A/Sys/rs2022proj214/projet-2022-ftc.pdf
/home/lucie/Documents/Cours/2A/Sys/rs2022proj214/src/ftc.c
/home/lucie/Documents/Cours/2A/Sys/rs2022proj214/includes/ftc.h
/home/lucie/Documents/Cours/2A/Sys/rs2022proj214/objs/ftc.o
/home/lucie/Documents/Cours/2A/Sys/rs2022proj214/ftc
/home/lucie/Documents/Cours/2A/Sys/rs2022proj214/ftc.o
```

FIGURE 0.1 – resultat du `ftc` avec l'option `-color`

L'extension `"-perm"` permet de connaître les permissions alloués à chaque fichier, ici la permission passée en option est sous forme octal, l'option est un entier composé de 3 entiers, chaque entier représente respectivement les permissions du propriétaire, des groupes et des autres.

Les entiers sont calculés de la manière suivante :

- Droit de lire le fichier/dossier = 4
- Droit de modifier le fichier/dossier = 2
- Droit d'exécuter le fichier/dossier = 1

Ainsi si la commande `"-perm 661"` est passé en paramètre, seuls les fichiers avec les permissions suivantes seront retournés.

- Le propriétaire a le droit de lire et de modifier le fichier/dossier.
- Les groupes ont les même droit que le propriétaire.
- Les autres ont uniquement le droit d'exécuter le fichier/dossier.

0.4 Difficultés rencontrées

Lors de ce projet, nous n'avons pas rencontré de grande difficulté. Nous avons réussi à travailler de manière régulière, en commençant dès que possible ce projet afin d'éviter de devoir tout faire au dernier moment. Les principales difficultés, que nous avons rencontrées, ont résidé dans la validation des tests. En effet, jusqu'à assez tard dans le projet nous ne passions pas le test build, le problème était en fait que notre Makefile possédait trop d'options. Nous sommes donc revenus à un Makefile un peu plus basique. Par la suite, nous avons réussi à passer les tests un par un en nous aidant des indications sur gitlab lorsqu'un test échoué alors que l'option avait été implémentée. Des fois, nous nous rendions alors compte que nous avions mal compris la consigne et que nous avions donc mal implémenté cette option. Par exemple pour l'option "-date", le membre en charge de cette option avait compris que comme pour l'option "-size" il fallait implémenter un "+" et "-" et ainsi si "-date 4m" était passé en argument seuls les fichiers ayant été changés 4min exactement auparavant devaient être affichés, or il fallait afficher tous les fichiers modifiés il y a 4 min ou moins. Enfin, si l'un des membres rencontrait une difficulté ou ne comprenait pas quelque chose, il faisait appel à l'autre qui l'aidait comme il le pouvait.

0.5 Tableau récapitulatif du nombre d'heures de travail

Ce tableau comptabilise le nombre d'heures de chacun passées à travailler sur le projet. La charge de travail de ce projet est de l'ordre de 25h par personne.

Étapes	Lucie	Lola
Conception		
Structure	3h	0h
Fonctions auxiliaires	3h	2h
Implémentation		
Argument	2h	2h
Parcourt	3h	0h
Name	4h	0h
Size	0h	4h
Date	0h	3h
Regex	1h	0h
Mime	0h	4h
Dir	2h	0h
CatchTheCat	1h	0h
Perm	0h	4h
Test		
Passage des tests	3h	2h
Rapport		
Rédaction Rapport	0.5h	2.5h
Relecture	0.5h	0.5h
Total		
	23h	24h

TABLE 0.1 – Charge de travail

0.6 Conclusion

Ce projet a permis aux membres du groupe de pratiquer le langage de programmation C qui avait été découvert lors du second semestre de la première année à TELECOM Nancy. Dans l'ensemble, nous avons apprécié le projet car nous y ont trouvé une grande satisfaction lors de la réalisation des options. La totalité des options exigées ont été implémentées ainsi que deux extensions : "-color" et "-perm". Notre code a été refactorisé à plusieurs reprises afin d'être le plus lisible et rapide possible. Au total, près de 50h ont été consacrées dans la réalisation de ce projet.