UNIVERSITE TOULOUSE III PAUL SABATIER
Faculté Sciences et Ingénierie

L2 CUPGE 2019/2020

Projet de Structure de Données

Objectifs ...

L'objectif de ce projet est de se familiariser à l'écriture d'un programme à plusieurs avec partage des tâches tant au niveau de la programmation que de la documentation.

Il s'agit de construire et de produire un programme (fichiers source et un exécutable testé et opérationnel avec la documentation), qui permet de gérer un parc de ressources physiques ainsi que les prêts de ces ressources. Tout cela sachant qu'une personne peut demander ou prêter plusieurs ressources et qu'une ressource à un propriétaire et ne peut être empruntée que par une personne à un moment donné. Une ressource est créée par une personne.

Le développement de ce programme se fera obligatoirement sous environnement Linux en langage C.

Projet de Structure de Données
... par Peresson Arthur & Saint-Maxin Théo

Sommaire

Organisation du programme

Nous avons voulu ici, essayer de trier les fonctions dans différents répertoires de façon à ce que tous soit bien organisé. Le but étant de s'y retrouver facilement.

Pour cela, voici une présentation de nos répertoires et de nos sous-répertoires :

… include

Ce répertoire consiste à accueillir tous nos fichiers « .h » contenant les déclarations de toutes nos fonctions, qui sont créées dans notre répertoire source.
Nous avons porté fortement notre attention sur leur documentation afin de comprendre ce qu'elles font et pouvoir les utiliser en toute simplicité.

… lib

Ce répertoire regroupe les librairies de chaque répertoire source afin de permettre au programme d'effectuer la compilation séparée grâce à des Makefile.

… doc

Ce répertoire regroupe tous ce qui concerne la documentation de notre programme.
On y retrouve un carnet de bord que nous avons rempli au fur et à mesure du projet pour garder un suivi de celui-ci.
De même, le dossier de configuration de la documentation Doxygen avec son exécutable y sont présent.
Puis enfin, le fichier « Configuration.txt » qui regroupe des informations qui servent surtout pour nous rappeler comment télécharger certaines librairies, ou même comment utiliser Doxygen.

… data

Ce répertoire regroupe tous les fichiers sauvegardant les données des utilisateurs et de leurs comptes associés, des objets créés ainsi que des prêts en cours.
Chaque fichier est sauvegardé sous le format JSON et chaque fichier JSON ne représente qu'une entité, donc soit une personne, soit un compte, soit un objet ou soit un prêt.
Tous cela sachant qu'un compte est sauvegardé comme ceci : « nom_d'utilisateur.json ».
Puis qu'une personne, un objet ou un prêt sera sauvegardé comme ceci : « ID.json », avec ID qui est composé de 8 chiffres, commençant par 1 si c'est un objet, 2 pour une personne ou 3 pour un prêt permettant ainsi de les distinguer facilement.

… src

Dans ce répertoire source, nous avons tous ce qui va faire fonctionner notre programme. Toutes les fonctions sont écrites dans les fichiers « .c » de chaque sous-répertoire. Chaque sous-répertoire contient un fichier « .mk » et un Makefile associé permettant de générer sa librairie. Ils contiennent aussi un main permettant de tester particulièrement des fonctions de son « .c ». De plus nous pouvons aussi trouver des scripts en Shell et des fichiers html qui contribuent au fonctionnement de notre programme.
Voici une explication de chacun de ces sous-répertoires :

- Entrees -

Ce sous-répertoire vise à contrôler les saisies de l'utilisateur au clavier, et être sûr qu'il va rentrer ce qu'on attend de lui.

- Objets –

Ce sous-répertoire permet de créer, initialiser, et contrôler tous ce qui touche aux objets. De plus il contient des fonctions essentielles à notre programme pour les manipuler, tels que les fonctions de sauvegarde ou encore ses getters et setters.

- Personnes -

Ce sous-répertoire permet de créer, initialiser, et contrôler tous ce qui touche aux personnes. De plus il contient des fonctions essentielles à notre programme pour les manipuler, tels que les fonctions de sauvegarde ou encore ses getters et setters.

- Administration -

Ce sous-répertoire permet de créer, initialiser, et contrôler tous ce qui touche aux comptes et à l'administration de notre programme. On peut y trouver quelques fonctions spécifiques comme les fonctions pour créer et chiffrer des mots de passe, une fonction pour supprimer son compte ou encore des fonctions pour l'administrateur pour par exemple, modifier les données des demandeurs / prêteurs.
De plus il contient des fonctions essentielles à notre programme pour les manipuler, tels que les fonctions de sauvegarde ou encore ses getters et setters.

- Prets -

Ce sous-répertoire permet de créer, initialiser, et contrôler tous ce qui touche aux prêts d'objets entre utilisateurs. De plus il contient des fonctions essentielles à notre programme pour les manipuler, tels que les fonctions de sauvegarde ou encore ses getters et setters.

- Temps -

Ce sous-répertoire est plus particulier que les autres. Il permet de gérer un temps, ce qu'on avait besoin pour faire un prêts car chaque objet a une durée de prêt, qui permet de récupérer son objet au bout d'un certain temps d'emprunt. Pour cela on a utilisé la librairie <time.h>, qui permet de manipuler un temps ou une durée très facilement.

- HTML -

Ce sous-répertoire vise à afficher à l'utilisateur ce qu'il a besoin. C'est à dire, lorsque l'utilisateur s'est connecté, il peut demander à voir ses objets, à voir sa liste d'emprunts ou encore sa liste d'objets prêtés. Nous avons donc opté pour les fichiers html. Nos fonctions écrivent dans un fichier html puis l'exécute, le fichier s'ouvre alors dans le navigateur et affiche comme convenu ce que l'utilisateur désire dans une mise en page que nous avons choisi nous-même. Beaucoup de script étaient nécessaire au fonctionnement de ces fonctions.

- Main -

Ce sous-répertoire est sûrement le plus ambitieux de tous car il contient le main du projet et donc aussi toute la partie graphique que nous avons choisi de mettre en place grâce à la SDL (dépassant ainsi la barre des 1000 lignes de code à lui seul). Nous avons donc toute la manipulation des évènements qui y est gérée ainsi que l'appel de chaque fonction. Nous savons que très tardivement que l'utilisation des « goto » est fortement déconseillé, nous ne pouvions donc pas nous permettre, en terme de temps, de refaire le main avec seulement des « while », donc notre main fonctionne avec les « goto ».

- Images -

Ce sous-répertoire contient toutes les images servant pour le sous-répertoire Main. Toutes ces images ont été créées de toute pièce par nos soins. Ainsi, lorsque l'utilisateur clic sur un bouton, nous avions juste à changer l'image. Cela modélise donc plutôt bien la gestion d'évènements comme sur un site web.

… pour plus de détails

Connectez-vous au compte Administrateur et appuyez sur « Doxygen »

Mode d’emploi du programme

Voici ici les points importants et à suivre pour le bon fonctionnement du programme. Bien-sûr, ce programme n’est valable que pour l’environnement Linux et est écrit en langage C.

… prérequis d’installation

Veuillez avant tous d’être à jour sur votre ordinateur, pour cela commencez par lancez dans le terminal ces lignes et de suivre les indications :

```
nom_utilisateur@pc:~$ sudo apt-get upgrade
nom_utilisateur@pc:~$ sudo apt-get update
```

Ensuite, entrez ces prochaines lignes pour pouvoir télécharger les librairies indispensables à l’utilisation de notre programme :

```
nom_utilisateur@pc:~$ sudo apt-get install libjson-c-dev
nom_utilisateur@pc:~$ sudo apt-get install libsdl2-dev
nom_utilisateur@pc:~$ sudo apt-get install libsdl2-ttf-dev
```

… lancement du programme

Placez-vous dans le dossier ShareThings puis lancez la commande :

```
nom_utilisateur@pc:~$ make
```

Bilan de notre travail

Nous allons à présent vous exposer ce que ce travail nous a apporté, les problèmes que nous avons pu rencontrer et les différents choix que nous avons fait lors de sa conception.

… problèmes rencontrés

Premièrement, notre plus gros problème au cours de ce projet à été la gestion de la mémoire. Au final, nous pensions avoir réglé tous nos problèmes de mémoire, le programme fonctionnait très bien, sans aucun crash ni bug. Mais après avoir ajouté un objet dans chaque catégorie, nous nous somme rendu compte que les comptes ne peuvent pas avoir plus de 8 ou 9 objet sans faire buguer le programme et que même si le programme crash, les fonctions marchent quand même. On peut alors ajouter des objets ou en emprunter même si le programme va s'éteindre directement après. Nous nous en somme rendu compte trop tard pour réussir à corriger ce problème efficacement.

Deuxièmement, pour l'installation de la SDL, nous avons rencontré quelques problèmes ou le compilateur n'arrivait pas à trouver <sdl.h> et <ttf.h>, nous nous sommes alors rendu compte que les deux « .h » n'étaient au final même pas présent. De plus l'apprentissage à été assez long, même si SDL est assez facile d'utilisation et de compréhension, il y a beaucoup de points à assimiler, ce qui peut prolonger l'apprentissage. Pour naviguer dans les menus et retourner en arrière, nous avons utilisé les « goto », ce qui n'est pas vraiment recommandé, mais c'est le seul moyen que nous ayons trouvé (nous avons entendu parler de machine à état trop tard pour pouvoir faire autrement).

Troisièmement, nous avons rencontré des problèmes à cause de la gestion de la mémoire de la structure Objet, ce qui nous posait problème pour la bonne exécution des fonctions du répertoire HTML. Il a aussi fallu apprendre les bases du html, pour pouvoir afficher ce que nous voulions.

Finalement, quelques problèmes on été rencontrés avec JSON. Le premier problème a été de trouver un parseur permettant de lire un fichier json et de récupérer ses données. Après, il a fallu comprendre comment s'en servir, cela a été compliqué car il y a peu d'explications claires et précises sur internet. Nous avons finalement trouvé une très bonne vidéo sur Youtube qui nous a aidé à comprendre comment faire. Et dernièrement, nous n'avons pas réussi à utiliser json de manière approfondi, nous somme resté assez simple (une clé = un attribut).

… ressentis

Lors de ce projet, nous avons dû faire énormément de recherches concernant la SDL, JSON, html, etc … Tout cela nous a alors permi de découvrir de nouvelles choses, de comprendre et savoir comment les utiliser, et ce qu'il y a d'intéressant à les utiliser.

Nous sommes tous les deux ravis d'avoir pu en faire la connaissance et d'avoir pu les utiliser malgré toutes les difficultés rencontrées. Nous savons pertinemment que tous ça va beaucoup nous servir à l'avenir. Le mélange de matière entre le langage C, Shell et Java(json) fait qu'on comprend l'intérêt de les combiner pour nous aider dans certains aspects du code. On trouve cela extrêmement enrichissant et on conseil à tous de découvrir aussi différents langages et de voir ce qu'on peut faire avec.

En tout cas, ce projet était très intéressant autant pour coder que pour la gestion du travail de groupe. Nous avons essentiellement travaillé à deux, à part pour SDL et json que l'on s'est partagé pour gagné du temps, car il y avait beaucoup de choses à savoir à leur propos. Malgré cela, nous nous sommes expliqués mutuellement comment ils fonctionnaient, ce qui fait que nous avons compris plus facilement et plus rapidement leurs aspects.

Pour finir, nous sommes très content de ce projet. Il était très enrichissant, et a donc permis de combiner différentes matières pour le réaliser. Nous espérons que le programme vous plaira malgré qu'il plante lorsque vous possédez trop d'objets. Amusez-vous !

… pour plus de détails

Ouvrez le fichier « CarnetDeBord.txt » qui se situe dans le répertoire « doc » afin de voir les différentes étapes datées par lesquelles nous avons dû passer pour ce programme.