Rapport TP4 : Le jeu du Chomp

DE JESUS MILITAR Dylan 2619

15 mars 2021

1 Introduction

Le principal objectif de ce projet était la réalisation d'un premier projet modularisé sous forme d'un jeu du Chomp.
Le jeu consiste en une partie entre deux joueurs, chaque joueur à la possibilité de manger n'importe quelle partie de la tablette considérée, afin de la réduire. La première case contenant un poison qui s'il est mangé entraîne la perte du joueur responsable.
En plus de cette notion de modularisation, ce projet m'a permis de travailler d'autres notions comme :

- Les types structurés

- Les types énumérés

- L'approfondissement de l'utilisation de la librairie graphique ncurses

- La compilation bash à l'aide de fichier

2 Exercice 1

Voir fichier Chomp.c

3 Exercice 2

Voir fichier Chomp.c

4 Exercice 3

Voir fichier Chomp.c

5 Exercic 4

5.1 Découpage

J'ai opté pour le découpage suivant :

- Coup

- Joueur

- Tablette

- Position

- Clic

- AffichageGraphique

- Regles

J'ai commencé par créé un module par types, ainsi que les fonctions qui leur sont liées puis j'ai remarqué trois grand états principaux, l'aspect graphique, la gestion de 'l'évènement clic par l'utilisateur, ainsi que les règles du jeu, c'est à dire l'état d'une partie ainsi que la légalité d'un coup

5.2 Compilation bash

J'ai commencé par la compilation et l'obtention des fichiers objets de chaque module puis celui du Main :

```
gcc -ansi -Wall -c Coup.c
gcc -ansi -Wall -c Joueur.c
gcc -ansi -Wall -c Position.c
gcc -ansi -Wall -c Tablette.c
gcc -ansi -Wall -c AffichageGraphique.c
gcc -ansi -Wall -c Regles.c
gcc -ansi -Wall -c Clic.c
gcc -ansi -Wall -c Main.c
```

Puis par la lecture de ces fichiers objets, puis obtenu en un executable nommé Projet :

gcc Main.o Coup.o Joueur.o Position.o Tablette.o AffichageGraphique.o Regles.o Clic.o -o Projet -lncurses

Pour compiler le projet il suffit simplement d'executer ces deux commande :

./Compilation.sh (compilation)

./Projet (execution)

5.3 Comparaison

Avantages Fichier unique :

Tout se trouve au même endroit il est ainsi plus facile de trouver une erreur à la compilation, ou de chercher une erreur de segmentation.
Aucun problème de conflit d'inclusion ou de nom de fonctions.
Il n'y a pas de problème de nombre importants de fichiers

Modularisé :

La lisibilité et l'entretien du programme sont facilités, on pourra en effet ne changer que certaines parties localisées, s'il y a un problème dans l'affichage graphique on n'aura besoin que de changer cette partie
La réutilisation des différents modules du projet favorise également ce concept de modularisation.
Il est possible de compiler localement.
Il est plus aisé de séparer la programmation de la documentation.

Inconvénients Fichier unique :

La lisibilité peut-être extrêmement mauvaise à partir d'une certaine quantité de code

Modularisé :

Il est possible de se retrouver avec un nombre extrêmement important de fichiers
Si le découpage est mauvais il y a un fort risque de conflits d'inclusions entre fichiers

6 Difficulté rencontrée

Concernant les difficultés rencontrées lors de ce projet, je retiens la difficulté de la recherche d'un bon découpage concernant un projet, il n'est en effet pas simple de réussir à bien lier les différentes parties entre elles.
De plus, j'ai été dans l'incapacité de réussir à bien aligner les différentes barres du rectangle dessiné graphiquement, toutes les solutions ont été testées, mais le fait d'ajouter un caractère ° représentant le poison de la tablette, empêche cet alignement

7 Conclusion

Pour conclure, ce projet m'a permis de réaliser pour la première fois de manière correcte un projet modularisé représentant un jeu graphique avec gestion du clic.