Projet C - **Snake**

Manuel développeur

Alice DELACHAT

Sommaire

Introduction

Le jeu Snake est un classique des jeux vidéo qui a connu un immense succès depuis sa création dans les années 1970.Il a été porté sur de nombreuses plateformes et est devenu un véritable symbole de l'industrie du jeu vidéo

Le but du jeu est simple : pour gagner le jeu, le joueur qui contrôle le serpent doit manger toutes les pommes.
En cas de collision avec le mur ou lui-même, le jeu est fini et le joueur a perdu.

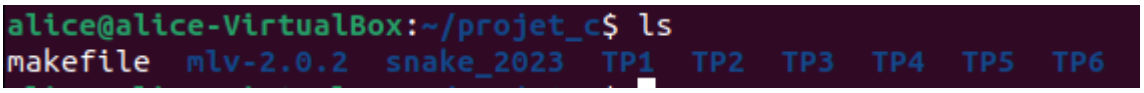
Dans le cadre de notre projet E3FR de programmation en C, nous devons recréer ce jeu.

Dans ce manuel développeur, vous pourrez découvrir le fonctionnement du code du jeu.

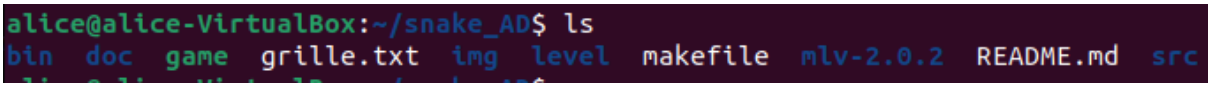
Vous y trouverez des explications sur la manière d'apporter des modifications ou d'ajouter de nouvelles fonctionnalités ainsi que toutes les fonctions, structures, énumérations et constantes y sont documentées. Ce guide vous offrira une vision complète du code du jeu.

Liste des fichiers du jeu

Vous pouvez y voir, avec la commande « ls », les différents dossiers des TPs qui nous ont permis de composer le jeu :



Le jeu final se situe dans le dossier *snake_AD*.



- On retrouve l'exécutable du jeu ***game***, généré avec la commande « make » ;
- Dans **/level**, il y a les différents niveaux proposés, sous forme de grilles, vous pourrez d'ailleurs en créer d'autres ;
- Dans le dossier **/src**, vous trouverez les fichiers sources de notre code :
  - ➢ **Game.c**, où l'on trouve tout le code nécessaire au bon fonctionnement du jeu ;
  - ➢ **Snake.c**, où l'on trouve tout le code lié au serpent et à ses fonctionnalités ;
  - ➢ **Grid.c**, où l'on trouve tout le code lié à la grille de jeu ;
  - ➢ **Snake.h**, où l'on trouve tout ce qui est utile pour générer le serpent (enum, struct, prototypes…) ;
  - ➢ **Grid.h**, qui nous permet de générer la grille (enum, struct, prototypes…) ;

Enfin, le fichier **README** est une brève explication du projet tandis que le fichier **Makefile** permet de générer les différents fichiers.

Par la suite, nous allons décrire le contenu de chaque fichier.

Game.c

Ce code permet le fonctionnement du jeu. Il contient la fonction main en plus des nombreux éléments indispensables au jeu.

Afficher_aide()
Cette fonction affiche un message d'aide sur la console lorsque l'option "-h" ou "--help" est utilisée.

Main()
La fonction principale ‘main()’ initialise les variables nécessaires au jeu, crée une fenêtre d'affichage et entre dans une boucle principale du jeu.
Ensuite, le code lit la grille à partir du fichier spécifié et initialise la fenêtre graphique du jeu.

Dans la boucle principale, la vitesse de déplacement du serpent est contrôlée en fonction de la variable "difficulty". Le serpent se déplace à chaque itération de la boucle.

Move_snake()
Cette fonction est appelée pour vérifier la case suivante et prendre les mesures appropriées. Par exemple, si le serpent heurte un mur, un écran de fin de jeu est affiché. Si le serpent mange un fruit, il grandit en ajoutant un segment à son corps.

La grille du jeu est dessinée à l'écran à chaque itération de la boucle.
Les touches du clavier sont détectées pour permettre au joueur de contrôler le serpent. Les touches directionnelles sont utilisées pour changer la direction du serpent. La boucle principale continue jusqu'à ce que la touche "Échap" soit enfoncée.

À la fin du jeu, la mémoire allouée pour la grille et le serpent est libérée, et la fenêtre d'affichage est fermée.

Le code **game.c** est donc responsable de la gestion des mécanismes de jeu, du contrôle du serpent et de l'interaction avec l'utilisateur via la fenêtre d'affichage.

Grid.c

Le code ‘**grid.c**’ est un fichier source qui contient des fonctions liées à la gestion de la grille de jeu du Snake.

La fonction "**draw_grid**" est utilisée pour dessiner la grille à l'écran en utilisant la bibliothèque MLV (Multi-Layer Visualization).

La fonction "**copy**" permet de copier une chaîne de caractères dans une autre.

La fonction "**fichier_get**" lit un fichier et obtient la grille correspondante, en enregistrant les données dans une structure "Grid".

La fonction "**debug**" est une fonction de débogage qui affiche la grille à l'écran.

La fonction "**compute_size**" calcule la taille des cases de la grille en fonction de la fenêtre.

Les fonctions "**tail_loss**", "**wall_loss**" et "**victory**" affichent différents écrans en cas de conditions spécifiques du jeu (perte de la queue du serpent, collision avec un mur, victoire).

La fonction "**move_snake**" permet de déplacer le serpent sur la grille et renvoie l'élément de la case suivante.

La fonction "**fruit_counte**r" compte le nombre de fruits présents sur la grille.

Les fonctions "**allocate_grid**" et "**free_grid**" sont utilisées pour allouer et libérer la mémoire associée à la grille.

En résumé, le fichier ‘**grid.c**’ contient des fonctions pour allouer et libérer une grille, dessiner la grille à l'écran, placer le serpent sur la grille, gérer le déplacement du serpent et détecter les collisions avec les éléments de la grille.

Grid.h

Le fichier ‘**grid.h**’ contient les déclarations des fonctions et des structures de données utilisées pour manipuler et afficher une grille de jeu.

Il commence par les directives du préprocesseur pour éviter les inclusions multiples.

Ensuite, une structure **‘Grid’** est définie pour représenter une grille de jeu. Elle contient un tableau à deux dimensions de caractères pour stocker les éléments de la grille, ainsi que les dimensions de la grille (‘nbc’ pour le nombre de colonnes et ‘nbl’ pour le nombre de lignes).

Une énumération **‘Element’** est définie pour représenter les différents éléments possibles dans la grille : ‘WALL’, ‘EMPTY’, ‘FRUIT’ et ‘SNAKE’.

Les déclarations des fonctions suivantes sont fournies :

- **Structure Grid** : contient un tableau de caractères pour représenter la grille, ainsi que les dimensions de la grille (nombre de colonnes et nombre de lignes).
- Énumération **Element** : représente les différents éléments pouvant être présents dans la grille, tels que le mur, la case vide, le fruit et le serpent.
- Fonction **allocate_grid** : alloue dynamiquement la mémoire pour une grille de taille spécifiée.
- Fonction **free_grid** : libère la mémoire allouée pour une grille.
- Fonction **debug** : affiche la grille à des fins de débogage.
- Fonction **compute_size** : calcule la taille d'un carré de la grille en fonction des dimensions de la fenêtre.
- Fonctions **draw_grid**, **compute_size_height** et **compute_size_width** : calculent les tailles des carrés de la grille pour les besoins d'affichage.
- Fonction **fichier_get** : charge les données de la grille à partir d'un fichier.
- Fonction **move_snake** : déplace le serpent sur la grille et retourne l'élément suivant.
- Fonctions **tail_loss**, **wall_loss** et **victory** : gèrent les différentes conditions de perte ou de victoire du jeu.
- Fonction **fruit_counter** : compte le nombre de fruits restants sur la grille.

En résumé, le fichier ‘**grid.h**’ définit les structures de données et les fonctions nécessaires pour manipuler et afficher une grille de jeu dans le cadre du jeu Snake.

Snake.c

Le code **snake.c** est un programme qui implémente le jeu du serpent. Voici un résumé condensé de ses fonctionnalités :

- La fonction "**new_snake()**" crée et retourne un nouveau serpent vide.
- La fonction "**add_segment()**" ajoute un segment au serpent à une position spécifiée.
- La fonction "**free_snake**()" libère la mémoire occupée par le serpent.
- La fonction "**crawl()**" déplace le serpent en fonction de sa direction actuelle et des limites de la grille.

Le code utilise une structure de données pour représenter le serpent, où chaque segment est représenté par une paire de coordonnées (x, y). Le serpent peut se déplacer dans quatre directions : haut, bas, gauche et droite.

Le programme permet de créer un serpent, d'ajouter des segments, de le faire ramper (crawling) et de libérer la mémoire occupée par le serpent.

Il constitue une partie importante du projet du jeu du serpent en gérant les fonctionnalités clés liées au serpent lui-même.

Snake.h

Le fichier "**snake.h**" contient les déclarations des structures et des fonctions nécessaires pour le jeu du serpent. Voici un résumé condensé des éléments présents dans ce fichier :

- La structure "**Coord**" représente les coordonnées sur la grille de jeu. Elle contient les coordonnées x et y ainsi qu'un pointeur vers la prochaine coordonnée.

- L'énumération "**direction**" définit les différentes directions possibles pour le serpent : en haut, en bas, à gauche et à droite.

- La structure "**Snake”** représente le serpent dans le jeu. Elle contient la direction du serpent, une liste de segments de coordonnées et la taille du serpent.

- La fonction "**add_segment**" permet d'ajouter un segment au serpent à une position spécifiée.

- La fonction "**free_snake**" libère la mémoire utilisée par le serpent.

- La fonction "**crawl**" fait avancer le serpent sur la grille en fonction d'un nombre de lignes et de colonnes spécifié.

- La fonction "**new_snake**" crée et initialise un nouveau serpent.

En résumé, le fichier "**snake.h**" définit les structures et les fonctions nécessaires pour manipuler et contrôler le serpent dans le jeu.

Makefile

Le fichier makefile permet au joueur de facilement compiler le jeu.

Niveaux

Il est facile de créer des nouveaux niveaux.Il suffit juste de baser sur les différentes grilles existantes dans le dossier ***/level***.

Bug

Un peu de latence dans le jeu et quelques warnings mais aucun bug n’est connu sur le projet pour le moment.

Conclusion

L'objectif de ce projet était de mettre en pratique les concepts fondamentaux de la programmation en C, tels que les structures de contrôle, les tableaux, les fonctions, etc., tout en créant un jeu amusant et interactif.

De plus, l'utilisation de la bibliothèque MLV nous a permis de découvrir et de bénéficier d'un ensemble d'outils graphiques pour faciliter le développement de l'interface du jeu qui tourne actuellement uniquement sur Linux (Debian ou Ubuntu).

Merci pour votre lecture et bon jeu.