UNIVERSITÉ Clermont Auvergne

Rapport sur le Jeu de Taquin en C avec SDL2

Sulvac Faruk-Emre

December 27, 2023

Contents

1 Introduction

Le jeu de taquin est un puzzle classique qui consiste en une grille de tuiles numérotées disposées de manière désordonnée, à l'exception d'une tuile qui reste vide. L'objectif du jeu est de réorganiser les tuiles en déplaçant la tuile vide de manière à atteindre une disposition ordonnée des numéros.

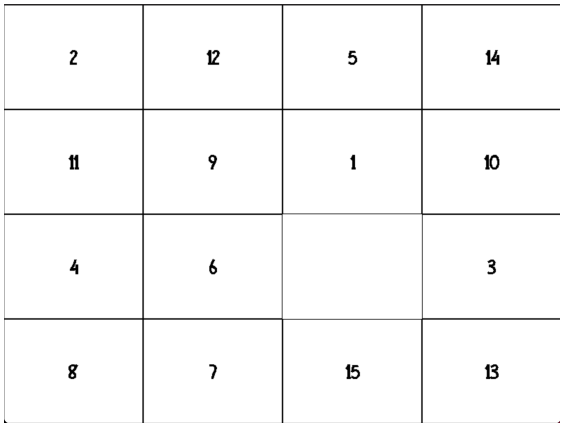

Figure 1: Exemple de jeu de taquin

Dans ce projet, nous avons implémenté le jeu de taquin en langage C en utilisant la bibliothèque SDL2 pour la gestion des graphismes et des événements. Ce rapport décrit le code source du jeu, la manière de le compiler et de l'exécuter, ainsi que les résultats obtenus.

2 Makefile

Voici une explication détaillée du makefile utilisé pour compiler un programme à partir d'un fichier source C (main.c) en utilisant le compilateur GCC.

Variables

```
TARGET = a.out
CC = gcc
CFLAGS = -I/opt/homebrew/include
LDFLAGS = -L/opt/homebrew/lib/ -lSDL2 -lSDL2_ttf -
   lSDL2_image -lSDL2main -lm -ldl
```

Liste des sources et des objets

```
SOURCES = main.c
OBJECTS = $(SOURCES:.c=.o)
```

Règles de compilation

```
$(TARGET): $(OBJECTS)
    $(CC) $(CFLAGS) -o $(TARGET) $(OBJECTS) $(LDFLAGS)
```

Règle générique de compilation des fichiers objets

```
%.o: %.c
    $(CC) $(CFLAGS) -c $< -o $@
```

Nettoyage du projet

```
clean:
    rm -f $(OBJECTS) $(TARGET)
```

En résumé, ce makefile automatise le processus de compilation en définissant des règles pour générer l'exécutable à partir du fichier source C, ainsi qu'une règle pour nettoyer les fichiers temporaires. Il utilise des variables pour rendre le makefile plus flexible et facile à maintenir. Les options de compilation et de liaison spécifiques à SDL2 sont également incluses.

3 Code Source

Ce programme en C utilise la bibliothèque SDL2 pour créer un jeu de puzzle de taquin (jeu de taquin numérique). Voici une explication détaillée du code :

Inclusions des bibliothèques :

- Les bibliothèques standard (`stdio.h`, `stdlib.h`, `time.h`) pour les fonctions d'entrée/sortie, la gestion de la mémoire et la génération de nombres aléatoires.
- La bibliothèque SDL2 (`SDL.h`) pour la gestion des fenêtres et du rendu graphique.
- La bibliothèque SDL_ttf (`SDL_ttf.h`) pour l'affichage de texte.

Constantes :

- `SCREEN_WIDTH` et `SCREEN_HEIGHT` définissent les dimensions de la fenêtre du jeu.
- `TILE_SIZE` spécifie la taille d'une tuile dans le puzzle.
- `BOARD_SIZE` indique la taille du plateau du puzzle (nombre de lignes et de colonnes).

Variables globales :

- `g_window` et `g_renderer` représentent la fenêtre et le renderer SDL.
- `g_font` est le pointeur vers la police de caractères utilisée pour afficher les numéros dans les tuiles.
- `g_puzzle` est une structure qui stocke l'état actuel du puzzle, y compris la disposition des tuiles et la position de la tuile vide.

Fonctions d'initialisation :

- `ft_initialize_font()` initialise la bibliothèque SDL_ttf et charge une police de caractères.
- `ft_initialize_puzzle()` initialise le plateau du puzzle avec des nombres ordonnés.
- `ft_shuffle_puzzle()` effectue des mouvements aléatoires pour mélanger le puzzle.

Fonction de dessin du puzzle (`ft_draw_puzzle()`):

- Utilise SDL pour dessiner le puzzle à l'écran, y compris les tuiles et les numéros.
- Utilise SDL_ttf pour afficher les numéros dans les tuiles.

Fonction de gestion des événements (`ft_handle_events()`):

- Gère les événements SDL, tels que les touches fléchées et la fermeture de la fenêtre.

Fonction de vérification de la résolution du puzzle (`ft_is_puzzle_solved()`)

- Vérifie si le puzzle est résolu en vérifiant si les numéros sont ordonnés et si la tuile vide est à la position finale.

Fonction de déplacement d'une tuile (`ft_move_tile()`):

- Déplace une tuile dans la direction spécifiée (haut, bas, gauche, droite).

Fonction de chargement du plateau depuis un fichier (`ft_load_board_from_f`

- Charge la configuration du plateau à partir d'un fichier texte.

Fonction principale (`main()`):

- Initialise SDL, crée la fenêtre et le renderer, charge la police, initialise le puzzle en fonction de l'option choisie.
- Utilise une boucle principale pour gérer les événements, dessiner le puzzle et vérifier s'il est résolu.

- Quitte le programme lorsque le puzzle est résolu ou que l'utilisateur ferme la fenêtre.

Ce programme offre deux options à l'utilisateur : charger un puzzle à partir d'un fichier ou générer un puzzle aléatoire. Il utilise les touches fléchées pour déplacer les tuiles et vérifie si le puzzle est résolu à chaque mouvement. Lorsqu'il est résolu, le programme affiche "Win" et se termine après un délai de 2 secondes.