Rapport projet de programmation système 2016/2017

Groupe IN501A2
Gautier DELACOUR
Bérangère SUBERVIE
Maxime VALLERON

6 janvier 2017

Table des matières

Chapitre 1

Le projet

Ce projet a été réalisé dans le cadre de l'UE "Programmation Système" du semestre 5 de Licence d'Informatique de l'Université de Bordeaux. Le but était de développer des fonctionnalités destinées au développement d'un jeu de plateforme 2D à l'aide du langage de programmation C.

1.1 Le jeu

Nous avons disposé au départ de tous les fichiers nécessaires à la compilation d'un code permettant de créer un exécutable permettant de lancer le jeu. Celui-ci se présente comme une jeu de plateforme 2D, où il est possible d'ajouter ou d'enlever des objets à cette carte.

1.2 Sauvegarde, chargement, et l'utilitaire Maputil

La première partie du projet consistait à permettre, d'une part, la sauvegarde de cartes, et leur chargement, et d'autre part, la modification de ces cartes grâce à l'utilitaire Maputil.
Celui-ci devait permettre d'obtenir les différentes données utiles d'une carte (sa largeur, sa hauteur, son nombre d'objets), mais aussi de modifier les données de cette carte.

1.3 Gestion des temporisateurs

La deuxième partie du projet nous demandait d'avoir une gestion de signaux permettant par exemple la présence d'objets dynamiques tels que des bombes ou des mines.

Chapitre 2

Le code fourni et son organisation

2.1 Arborescence

L'arborescence du projet se trouve dans un un dossier **Projet**. Elle s'organise ainsi :

deps : dossier où sont créés les fichiers de dépendances (.d) lors de la compilation.

images : dossier où sont stockées toutes les images utilisées dans le jeu.

include : dossier où se trouvent tous les fichiers en-tête (.h) définissant la signatures des fonctions dans les différents fichiers du code.

lib : dossier où se trouve la bibliothèque *libgame.a* où sont codés la plupart des fonctions permettant de jouer.

maps : dossier où se trouvera les fichiers contenant les cartes sauvegardées. Il est pour l'instant inutilisé.

obj : dossier où sont créés les fichiers objet (.o) à partir des fichiers de code (.c) lors de la compilation.

sons : dossier où sont stockés tous les sons utilisés dans le jeu.

src : dossier où se trouve le fichier *mapio.c* et le fichier *tempo.c* (il manquait cependant le code de quelques fonctions).

util : dossier où se trouve le fichier objet.txt où sont décrits les différents objets du jeu.

2.2 Code

Du code était déjà fourni dans des fichiers de l'arborescence de base.

2.2.1 mapio.c

map_new : cette fonction alloue l'espace nécessaire pour créer une carte, puis construit une carte de base.

map_save : cette fonction devra sauvegarder une carte modifiée. Pour l'instant elle ne fait qu'afficher que **map_save** n'est pas implémentée.

map_load : cette fonction devra charger une carte modifiée depuis le fichier *saved.map* dans le dossier **maps**. Pour l'instant elle ne fait qu'afficher que **map_load** n'est pas implémentée.

2.2.2 tempo.c

get_time : cette fonction retourne le temps écoulé depuis le début de 2016 en microseconde.

time_init : cette fonction est appelée au démarrage, elle mettra en place la gestion des signaux et initialisera les variables utiles pour leur traitement. Pour l'instant, elle ne fait que retourner 0 pour montrer que le code n'est pas encore écrit.

time_set : cette fonction arme un temporisateur d'une durée "delay" en microseconde et déclenchera un événement prédéfini avec **sdl_push_event** grâce à la valeur "param".

2.2.3 libgame.a

Cette bibliothèque implémente toutes les autres fonctions décrites dans les fichiers en-tête du dossier **include**.

Chapitre 3

Gestion de cartes

La gestion de cartes nous demandait de gérer deux principaux points :

La sauvegarde et le chargement : une carte modifiée peut être enregistrée dans un fichier *saved.map*, enregistré dans le dossier **maps**. Ce meme fichier peut être utilisé pour charger cette carte éditée à la place de la carte de base.

Maputil : ce gestionnaire permet essentiellement la modification d'une carte enregistrée.

3.1 Sauvegarde et chargement d'une carte

Les fonctions de sauvegarde et de chargement se trouvent dans le fichier *mapio.c* dans le dossier **src**.

Sauvegarde : cette fonction permet de sauvegarder la largeur, la hauteur, les objets disponibles et l'emplacement des objets d'une carte dans le fichier *saved.map* dans le dossier **maps**.

Chargement : cette fonction permet de charger la carte décrite dans le fichier *saved.map* dans le dossier **maps** lors du lancement du jeu.

3.2 Maputil

Maputil est un utilitaire de manipulation de carte. Son code source se trouve dans le fichier *maputil.c* dans le répertoire **util**. Il permet de consulter des informations contenues dans un fichier *saved.map* et de modifier ces informations. Il contient les fonctions suivantes :

usage : cette fonction permet de vérifier que l'utilisateur a entré une commande valide. Sinon, elle arrête l'exécution.

optionsAlloc : cette fonction alloue un tableau contenant toutes les options possibles de maputil.

optionsFree : cette fonction libère la mémoire allouée par la fonction précédente ;

getWidth : cette fonction renvoie la largeur de la carte passée en paramètre.

getHeight : cette fonction renvoie la hauteur de la carte passée en paramètre.

getObjects : cette fonction renvoie le nombre d'objets d'une carte passée en paramètre.

setWidth : cette fonction change l'ancienne largeur d'une carte par une nouvelle.

setHeight : cette fonction change l'ancienne hauteur d'une carte par une nouvelle.

setObjects : cette fonction change l'ancienne liste d'objets d'une carte par une nouvelle.

pruneObjects : cette fonction supprime les objets inutilisés dans une carte de la liste d'objets de cette dernière.

traitementOption : cette fonction teste la correspondance entre l'option demandée et les options existantes, et appelle la ou les fonction(s) correspondante(s) si elle existe. Elle renvoie le nombre d'arguments utilisés.

Chapitre 4

Gestion des temporisateurs

La gestion des temporisateurs nous demandait de gérer à l'aide de signaux des événements, comme notamment l'explosion de bombes/mines dans le jeu.

4.1 Création d'un thread démon

Dans un premier temps, nous avons uniquement créé un thread qui gère la réception du signal SIGALRM dans la fontion **timer_init**. Il effectue une boucle infinie en appelant la fonction **sigsuspend**.

4.2 Implémentation simple d'une gestion de signaux

Dans un deuxième temps, nous avons effectué la gestion d'un et unique temporisateur (une explosion). Nous avons simplement rajouté un appel à la fonction *setitimer* dans la fonction **timer_set** et demandé l'affichage de l'événement sur la sortie standard dans la fonction **timer_init**.

4.3 Implémentation complète de la gestion de signaux et mise en service

Enfin, nous avons créé la gestion d'un échéancier d'événements dans la fonction **timer_set** afin de gérer plusieurs événements avec cette fonction.

Chapitre 5

Organisation de la charge de travail

Afin de répartir la charge de travail, chacun s'est occupé d'une ou plusieurs parties :

Gautier DELACOUR s'est occupé des fonctions de gestion et modification de la carte dans Maputil.

Bérangère SUBERVIE s'est occupé du traitement des options dans Maputil.

Maxime VALLERON s'est occupé des fonctions de sauvegarde et chargement dans le fichier *mapio.c* et de la gestion des temporisateurs.

Cependant, il est à noter que nous avons en réalité beaucoup travaillé ensemble sur toutes les parties du projet, et nous nous sommes corrigé nos erreurs les uns les autres.