Telecom Nancy

Université de Lorraine

Rapport de projet C

NOM DU JEU

Auteurs :
Nicolas Bédrine
Raphaël Moulet
Vincent Albert

23 mai 2015

Table des matières

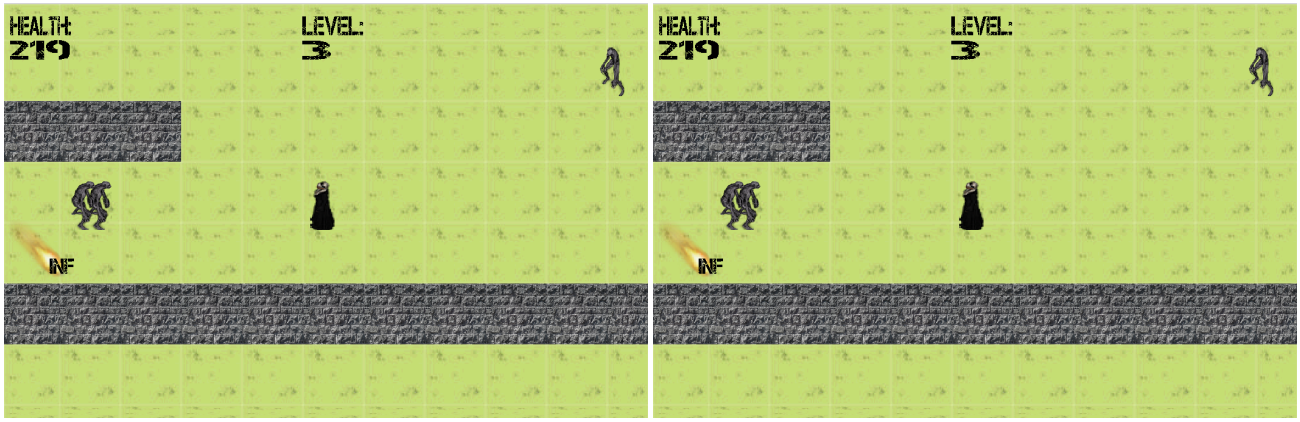

FIGURE 1 – Snapshots

1 Présentation du sujet

1.1 Un rapide résumé

Le jeu que nous nous sommes proposés de réaliser est un beat'em'all et tower defense s'inspirant du jeu BoxHead. Le principe général est de survivre sur une carte à plusieurs vagues successives d'ennemis. Pour se faire, le joueur incarne un personnage en vue à la troisième personne qui peut se déplacer, attaquer à distance avec des sorts et construire des défenses.

1.2 Une licence pour les protéger tous

Le logiciel et l'ensemble des sources sont sous licence CC BY-NC-SA 3.0 FR.
Le partage et l'adaptation du logiciel sont permises à condition de ne pas l'utiliser commercialement, d'indiquer les changements effectués, de référencer l'œuvre originale et de conserver la même licence lors d'une diffusion ultérieure.

1.3 Le cahier des charges

Afin de gérer au mieux le projet, nous nous proposons d'établir un cahier des charges minimal dont les caractéristiques se retrouveront dans toute version finale du projet. En plus de cela, nous établirons une liste non ordonnée de fonctionnalités qu'il semble de prime abord intéressant d'implémenter. Leur mise en place découlera de leur difficulté ainsi que du temps disponible. Cahier des charges minimal :

- Affichage graphique de la partie (joueur, carte et ennemis)
- Lancement d'une partie sur une carte unique
- Interagir avec le clavier pour se déplacer dans l'environnement
- Gestion de l'interaction avec l'environnement (collision, ...)
- Utilisation d'une arme pour se défendre
- Création d'un dispositif de défense
- Implémentation de l'IA des ennemis (repérer, poursuivre le joueur et éviter les obstacles)
- Implémentation de l'IA générale de la partie qui gérera la création d'ennemis au cours du temps
- Menu d'accueil

Fonctionnalités optionnelles :

- Mode coopératif à deux en réseau ou sur le même clavier
- Mode deathmatch entre deux joueurs
- Création/achat de nouvelles armes
- Création de systèmes défensifs supplémentaires
- Mini mode aventure

— Difficulté croissante des ennemis dans le temps (nouveaux ennemis plus rapide, plus résistants, ...)
— Menu pause
— Choix de la difficulté générale
— Capacités spéciales à usage limité
— Ajoute de cartes supplémentaires
— Éditeur de cartes

2 Organisation

2.1 Les outils utilisés

Le projet a été codé en langage C sous systèmes de type Unix. L'affichage graphique a été réalisé à l'aide de la bibliothèque SDL qui est déjà utilisée dans de nombreux jeux. Les sprites sont libres de droits. Étant donné l'ampleur du projet et le nombre de participants, nous nous sommes convenus d'utiliser le gestionnaire de version git. Un dépôt public a été créé sur Github et est accessible à l'adresse suivante : `https://github.com/Neressea/projetC`

2.2 Le planning initial

Afin d'achever au mieux les objectifs que nous nous étions fixés, nous avons dressé un planning optimal des tâches à effectuer.

2.3 Le schéma des fichiers et des fonctionnalités

3 La phase de développement

3.1 Les attentes atteintes

L'ensemble du cahier des charges minimal a été implémenté. De plus, la difficulté est proportionnelle au niveau du joueur. En effet, le nombre d'ennemis dépend du niveau du joueur, et donc des ennemis qu'il a déjà vaincu. Nous avons aussi ajouté une attaque à usage limitée. Des items sont laissés par les ennemis vaincus pour régénérer les points de vie du héros ou son sort limité. Enfin, la vue a été adaptée pour que la carte est redimensionnable en cours de partie.

3.2 Ce qui reste inachevé

Malheureusement, nous n'avons pas réussi à mettre en pratique nos autres idées. Cependant, nous avons tout de même préparé l'implémentation d'autres fonctionnalités. Ainsi, même si l'éditeur de cartes n'a pas été réalisé, le code a été adapté afin de pouvoir charger des fichiers textes formatés pour faciliter son ajout par la suite.

De la même manière, bien que la vue soit prête pour la modification des touches, et qu'il existe une fonction permettant de modifier les contrôles utilisateurs, nous n'avons pas eu le temps de lier les deux.

3.3 Les problèmes rencontrés

3.3.1 Quelques problèmes de communication

En nous confrontant pour la première fois à un travail de groupe de cette ampleur, nous avons rencontré quelques difficultés, dont certaines communicationnelles. En effet, lors de la recherche de sprites, nous avons commencé par les boules de feu ; puis nous nous sommes concertés pour la recherche des boules de glace. Cependant à cause d'un quiproquo, nous avons temporairement fini avec des sprites de cornets de glace.

3.3.2 De la mémoire et des fuites

Après plusieurs longues séances de codage consécutives

3.3.3 Gestion événementielle

Nous n'arrivions pas dans un premier temps à gérer l'appuie de plusieurs touches en même temps. Nous avons alors changer notre Wait_Event par un Poll_Event. Ce dernier a comme avantage de garder en mémoire plusieurs évènements, contrairement à Wait_Event qui les gèrent un par un. Les évènements étant stockés dès le premier appel de la fonction Poll_Event, un switch(Poll_Event) n'est pas nécessaire. Nous avons ensuite décidé de créer un tableau de booléen contenant les états de chaque touche nécessaire au jeu. Il a donc fallu contrôler à la fois lorsqu'une touche est pressée, mais aussi lorsqu'elle ne l'est plus. Ce tableau nous a permis de gérer les évènements de manière plus lisible, mais aussi de gérer grâce à un if l'état de plusieurs touches à la fois, permettant de gérer l'appuie de plusieurs touches.

3.3.4 La vue et le modèle

Collisions et vue déplaçable.

Références et autres sources d'inspiration

`http://blog.hikoweb.net/index.php?post/2011/11/06/Exemple-de-rapport-en-LaTeX`
Source pour le template LaTeXayant servi à la rédaction de ce rapport.

`http://www.crazymonkeygames.com/Boxhead-2Play-Rooms.html`
Source d'inspiration du jeu.

`https://wiki.libsdl.org/`
Wiki de la SDL 2.0.

`http://gnurou.org/`
Nous nous sommes inspirés des algorithmes de ce site pour la gestion des événements.

`http://jeux.developpez.com/tutoriels/sdl-2/guide-migration/` *Guide de migration de passage de la SDL 1.2 à la SDL 2.0. Ce site nous a été utile étant donné que nous avions déjà programmé avec la SDL 1.2 auparavant.*

`https://creativecommons.org/licenses/by-nc-sa/3.0/fr/` *Licence s'appliquant au logiciel.*