Rapport de Projet “Hex Light Bot”

Julien Trijean – Nathan Précigout

S2B

Date : 31/05/2017

Table des matières

1 Introduction 2
1.1 Contexte 2
2 Le programme réalisé 2
2.1 Contraintes 3
2.2 Objectifs pédagogiques 3
2.3 Phase de programmation 3
2.4 Phase d’exécution 4
3 Conception 4
3.1 Difficulté rencontré 5
4.1 Comment compiler 5
5. Bilan 6
5.1 Ce qui ne fonctionne pas 6
5.2 Ce qui fonctionne 6
5.3 Les améliorations 7
Conclusion 7

1 Introduction

Ce rapport, présente le projet Hex Light Bot ainsi que la réalisation de ce dernier.

1.1 Contexte

Ce projet nous a été proposé par l'IUT informatique de Bordeaux, comme projet de second semestre.

Livrables

- Un rapport (celui-ci)
- Code du projet fonctionnel

Technologies

Le projet a été réalisé en C++11 ainsi qu'avec la bibliothèque graphique SFML 2.0. Nous avons utilisé le logiciel QT Creator pour le code et le logiciel Photophiltre pour faire les sprites.

2 Le programme réalisé

Le jeu Lightbot est un jeu de puzzle éducatif qui va vous apprendre quelques bases de l'informatique

Le but et de déplacer un robot dans une grille, afin d'allumer des lumières. Pour cela vous allez disposer de différentes actions que vous pourrez mettre dans un « Main » ou dans une fonction afin de faire des boucles et ainsi résoudre les différents niveaux.

Le jeu comporte deux phases : programmation et exécution

Notre version « Hex Light Bot » est une adaptation de ce jeu avec pour différences majeurs que nous utilisons une grille hexagonale !

Le jeu est composé d'une grille composée elle-même de cases hexagonales. Les cases formeront toujours un carré.

Exemple : Grille 3x3 pu encore grille 6x6

Les cases peuvent avoir différentes valeurs qui sont :

- Cases vides (avec étages ou non)
- Cases lumières éteintes (avec étages ou non)
- Case simple (avec étages ou non)
- Case allumée (avec étages ou non)

2.1 Contraintes

Les contraintes qui nous ont été imposées sont les suivantes :

- Réaliser une adaptation du jeu avec une structure hexagonale
- Avoir un programme pour jouer au jeu qui sera fonctionnel
- Avoir un éditeur de terrains

2.2 Objectifs pédagogiques

- Mise en pratique des notions vues en M2103 Programmation Orientée Objet
- Développement de composants réutilisables (objets de l'interface, communs au jeu et à l'éditeur de terrains)
- Utilisation de la bibliothèque standard C++ (STL) et de la SFML
- Travail en collaboration et autonomie de gestion de projet

2.3 Phase de programmation

Le programme présenté ici commence par un menu afin de choisir entre jouer ou quitter le jeu

Le menu jouer permet de choisir entre différents niveaux de jeu.

Dans un premier temps le joueur compose un programme par glisser/déposer d'actions dans une fonction ou dans le main

Il exécute ensuite le programme

2.4 Phase d'exécution

Le robot effectue le parcours indiqué. Il s'arrête quand il n'y a plus d'actions à effectuer, ou si une action est impossible (avancer dans un mur).

Si l'objectif est atteint, l'utilisateur peut passer à un autre niveau, sinon il peut recommencer.

3 Conception

Notre projet contient 7 classes :

Appli

Gère les relations entre les classe et fait la boucle de lecture du jeu

Affichage

Elle permet l'affichage de tous nos sprites

Action

Cette classer permet de gérer les actions de notre jeu que le joueur pourra utiliser afin de terminer le niveau

Cases

Elle permet de gérer les cases qui sont dans la grille, elles ont toutes une valeur qui les définissent

MainProc

Permet de stocker toutes les actions du PROC

Proc

Permet de stocker toutes les actions

Terrain

Le terrain stocke toutes les cases

3.1 Difficulté rencontré

Le plus gros problème a été pour la gestion des cases en fonctions quelles soit paires ou impaires, ainsi que le types de la case.

Le prototype est le suivant :

```
void Affichage::affichageJeu(Terrain* damier, std::vector<int> liste_actions, std::vector<int> liste_Proc)
```

4 Origine des éléments

Les sprites ont été fait entièrement à la main avec le logiciel Photophiltre.

Le seul élément provenant d'une source extérieur est le logo « Hex Light Bot » qui a été créé avec Cooltext .com.

4.1 Comment compiler

Après avoir récupéré le dossier, il faut le dé zipper puis l'extraire dans votre répertoire personnel.

Afin de lancer le jeu vous devez être équipé de Qt Creator avec C++11 au moins. Vous devrez également disposer de la SFML 2.0.

Vous devrez changer le répertoire d'exécution du projet et le mettre là où sont rangés les sprites, normalement le dossier LightBot lui-même, afin de le compiler.

5. Bilan

Dans la réalisation de notre projet, certaines choses ont très bien fonctionnés, d'autres n'ont pas pu être réalisé par mauvaise gestion du temps.

5.1 Ce qui ne fonctionne pas

Nous n'avons pas réalisé d'éditeur de niveaux.

Nous n'avons pas affiché de sprites pour les cases.

Nous n'avons pas fait de sprites pour le robot.

Nous ne pouvons pas mettre d'action dans le main, nous pouvons uniquement mettre le P1.

Nous n'avons mis que 2 niveau jouable.

5.2 Ce qui fonctionne

La navigation entre les différentes interfaces de jeu

Nous pouvons jouer au jeu sans problèmes gênant.

La possibilité de quitter le jeu

La possibilité de choisir entre 2 niveaux

La possibilité de recommencer le niveau

5.3 Les améliorations

Les améliorations à apporter concernent principalement les choses que nous n'avons pas pu faire par mauvaise gestion de temps.

Nous aimerions faire un éditeur de jeu afin que le joueur puisse créer son niveau.

Nous aimerions faire un mode pas à pas pour l'exécution des actions.

Corriger quelques problèmes mineurs dans le jeu.

Conclusion

Pour conclure ce projet nous a vraiment permis de mettre en œuvre nos connaissances sur la programmation objet en C++

Cela les a aussi renforcés grâce à la recherche que nous avons dû faire pour résoudre les différents problèmes que nous avons rencontrés lors de la programmation de ce jeu.