INFO-H-200 : Programmation orientée objet
Projet

Professeur : Hugues Bersini
http://cs.ulb.ac.be/public/teaching/infoh200

Année académique 2017-2018

---

1 Sujet

On vous demande de réaliser un jeu du type Hack and Slash / Dungeon Crawler en Java s'accompagnant d'une GUI. Le délivrable devra présenter une interface graphique et être implémenté en orienté objet. Par ailleurs, votre code devra mettre en pratique l'ensemble des concepts OO vus durant les séances de travaux pratiques.

Un jeu de type Dungeon Crawler peut se décrire comme suit : il s'agit d'une sorte de jeu de rôle dans lequel un héro parcourt un labyrinthe (a.k.a. Le donjon) en tuant des ennemis et en ramassant les objets qu'il peut trouver ou ceux résultant du décès d'un ennemi.

FIGURE 1 – Exemple : The Wizard `https://www.youtube.com/watch?v=XvnUZ_kHcaw`

De nombreux exemples peuvent être cités et nombreuses sont les variations existant autour de cette thématique. Nous nous contentons ici (Voir Figure 1) de vous fournir un petit exemple simple illustrant le principe sur un cas implémenté en 2D sur une carte vue du haut, soit ce que nous attendons précisémment pour ce projet.

2 Cartes

Le plateau de jeu devra être d'une taille paramétrable par l'utilisateur, par exemple 50x50. Chaque plateau peut soit être généré aléatoirement, en plaçant différents objets, ennemis et portes sur la carte au chargement, soit en lisant la description de ceux-ci directement depuis des fichiers. Le joueur pourra ainsi se déplacer en suivant quatre directions : Nord, Sud, Ouest et Est. Nous vous laissons choisir le mode d'enchaînement des cartes parmi les trois propositions suivantes :

- Carte statique : une fois l'ensemble des monstres tués, une nouvelle carte est générée. (e.g : Survival)
- Labyrinthe : plusieurs cartes existent et sont liées par un ensemble de portes. Ces cartes sont mémorisées pour assurer la cohérence des passages d'une pièce à l'autre. (e.g : Pokemon ou Hack and Slash classique)
- Fenêtrage : la carte sera d'une taille bien plus importante, par exemple 10.000 x 10.000 mais le joueur ne pourra visualiser qu'une partie de 50x50 dans une même fenêtre. Le déplacement de celui-ci fera se déplacer la fenêtre. (e.g : Command and Conquer)

Le personnage pourra donc se déplacer, ramasser des objets, tuer des PNJ, etc...

3 Objets / Inventaires

Différents objets doivent être placés sur la carte ou peuvent être récupérés au sol après avoir éliminé un ennemi. On vous demande de pouvoir gérer un inventaire d'une taille fixe. Les objets pourront donc être ramassés, utilisés ou jetés.

4 Armes

Dans sa version la plus simple, le personnage aura une seule attaque à sa disposition. Une épée ou pour un mage, une attaque directe et non à distance.

5 Bonus et améliorations

D'autres fonctionnalités doivent être implémentées, quelques exemples sont fournis ci-dessous. Attention, un bonus ne sera validé comme apport supplémentaire qu'à la condition qu'il implique un effort de dévelopement. Par exemple, proposer différentes potions 100HP, 200HP,... ne compte pas comme bonus car il s'agit, non d'un nouvel objet avec un comportement différent, mais un objet avec une valeur d'attribut modifée.

- Différents objets : potions, armes, boucliers, vies,...
- Des objets particuliers permettant d'accroitre la taille de l'inventaire...
- Différents effets : attaques à distance, damage over time, area damage, ralentissement (i.e : Snare),...
- Des emplacements de la carte pourraient soumettre l'utilisateur à un ralentissement de déplacement.
- Les ennemis pourraient se déplacer selon un schéma prédéfini, attaquer le joueur ou le fuir.
- Téléportation, retour dans le passé,...
- Sauvegarde, chargement
- Multi-joueur

Le principe du jeu est simple et nous n'offrons qu'un seul sujet, libre à vous de choisir les fonctions que vous désirez implémenter !

6 Code de base et conseils

Un code de base implémentant, entre autre, une interface graphique simple vous est fournie comme exemple. Vous pouvez partir de celui-ci pour réaliser votre projet.

Il est essentiel de lire attentivement ce code, et de ne pas simplement copier-coller certaines parties. Certains aspects de ce code n'ont pas encore été étudiés durant les TP (comme le Partern Observeur par exemple). Ne paniquez pas si certains morceaux de codes vous semblent compliqués. N'hésitez pas à y insérer des instructions "print" pour mieux comprendre comment il s'exécute. N'hésitez pas non plus à utilisez votre moteur de recherche favori pour obtenir plus d'informations.

7 Consignes

On vous demande d'implémenter une application graphique 2D en Java en illustrant les concepts orientés objets vus durant les cours et aux séances de travaux pratiques (e.g : Héritage, Encapsulation, Exceptions, Interfaces, Généricité, Polymorphisme, Design patterns, Threads, etc...). Le projet peut être réalisé individuellement ou par groupe de **2 étudiants** au maximum. Vous n'êtes pas obligés de faire partie de la même série de TP pour former un groupe. La défense aura lieu en salle informatique et comportera, outre une démonstration de votre jeu, des questions individuelles. Chaque étudiant devra donc connaître et comprendre l'ensemble du programme présenté et sera interrogé sur sa conception. La démonstration pourra se faire soit sur les ordinateurs des salles, soit sur votre ordinateur personnel. Chaque groupe doit envoyer un mail à **gdejaege@ulb.ac.be** comprenant les noms, prénoms et emails de chacun des membres du groupe ainsi formé et ce avant le **23 mars 2018**. Le sujet de ce mail devra être précisément INFOH200_Projet afin de faciliter de traitement de l'afflux massif d'emails attendu durant cette période.

8 Délivrables

8.1 Rapport préliminaire

Un rapport de maximum 5 pages portant sur la phase d'analyse fonctionnelle, de conception et de modélisation vous sera demandé pour le **2 avril 2018**. Ce rapport devra contenir les éléments suivants :

1. Une description de votre architecture orientée objet accompagnée d'une explication sur les choix posés. Cette description comprendra un diagramme de classe ainsi qu'un diagramme de séquence de votre boucle principale.
2. Une description des différentes fonctionnalités qui seront implémentées. Au moins dix fonctionnalités de votre choix doivent être détaillées.

8.2 Rapport final

Le rapport final devra décrire l'ensemble du travail réalisé avec les éléments suivants :

1. Une description synthétique des fonctionnalités implémentées ainsi que le ou les noms du ou des auteurs de cette fonctionnalité.
2. Les diagrammes UML : Classe et Séquence (boucle de jeu principale)
3. Une description complète et justifiée de votre architecture finale en indiquant les Design Patterns qui ont été employés.

Le rapport final devra être envoyé par email à **gdejaege@ulb.ac.be** pour le **12 mai 2018** accompagné du code source complet et commenté. Le sujet de votre email devra également être INFOH200_Projet.

9 Évalutation

La défense du projet consistera en une brève démonstration de votre programme ainsi que quelques questions relatives à votre code. La défense aura lieu durant la semaine suivant la remise et se fera suivant un horaire à déterminer.

L'évaluation du projet portera les critères suivants :

— L'architecture du projet

— Le respect des règles de bonnes pratiques de la programmation orientée objet
— La maîtrise des concepts vus aux cours et aux travaux pratiques
— La qualité et la clarté de votre implémentation (e.g : structure claire, commentaires appropriés,...)
— Les possibilités de paramétrage de l'application
— Le contenu du rapport
— La pertinence de vos réponses aux questions lors de la défense.