Rapport de projet

Conception agile de projets informatiques et génie logiciel

Université Lumière Lyon 2

Année Universitaire 2022-2023

Projet Balle au prisonnier

Alexandre Maurin (5212218)

Anthony Graissot (2191889)

Haroun Feniza (5213136)

Introduction

« Balle au prisonnier » est un projet que nous avons réalisé en trinôme* dans le cadre de l'unité d'enseignement Conception agile de projets informatiques et génie logiciel.

**Il est à rappeler que nous somme en trinôme car le groupe de td est composé d'un nombre impair d'étudiants*

Principe

Le jeu se compose d'un terrain (field) ainsi que de deux joueurs contrôlés par l'utilisateur(s) à l'aide des touches zqsd,a et ↑↓←→, espace; ainsi que des joueurs contrôlés par l'ordinateur.

Le but est d'éliminer tous les adversaires en les touchant à l'aide de la balle pour l'emporter, il est donc important d'essayer d'esquiver la balle quand elle arrive vers vous.

Il est possible de toucher plusieurs personnes avec le ballon : il n'y a pas de prison, une fois touché le joueur est directement éliminé.

Vous pouvez faire une passe à vos coéquipiers, il n'est pas possible d'éliminer ses coéquipiers, vous ne pouvez pas vous éliminez vous-même, la vitesse de la balle dépend de la force du joueur et la balle rebondie sur les murs.

La partie s'arrête quand il n'y a plus aucun représentant d'une même équipe en vie.

Le jeu peut être mis en pause à l'aide de la touche Echap ou de la touche P.

Architecture Et Structure

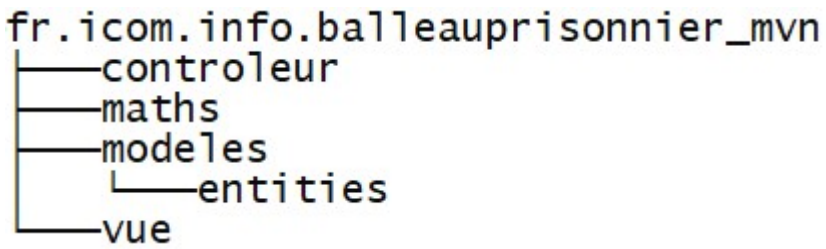
Architecture principale

Le projet suit une architecture de type Modèle-vue-contrôleur séparée à l'aide de packages.

Packages

Le projet se dissèque en 5 packages :

- fr.icom.info.balleauprisonnier_mvn.controleur
- fr.icom.info.balleauprisonnier_mvn.modeles
- fr.icom.info.balleauprisonnier_mvn.modeles.entities
- fr.icom.info.balleauprisonnier_mvn.vue
- fr.icom.info.balleauprisonnier_mvn.maths



3 packages principaux *modèle*, *vue* et *contrôleur*, un sous-package de *modèle* pour les entités et un package de classes utilitaire *maths*.

Classes

Il y a au total 11 classes :

–Modele–

- Field :: Le terrain du jeu, initialise les éléments qui le composent

–entities–

- Entity :: classe abstraite commune à toutes les entités
- Player :: classe représentant le modèle d'un joueur
- PlayerIA :: classe qui étend la classe Player, représentant un joueur contrôlé par l'ordinateur
- Projectile :: classe représentant le modèle du projectile (la balle)
- Sprite :: modèle d'un sprite, utilisé par les joueurs et afficher dans la vue

–vue–

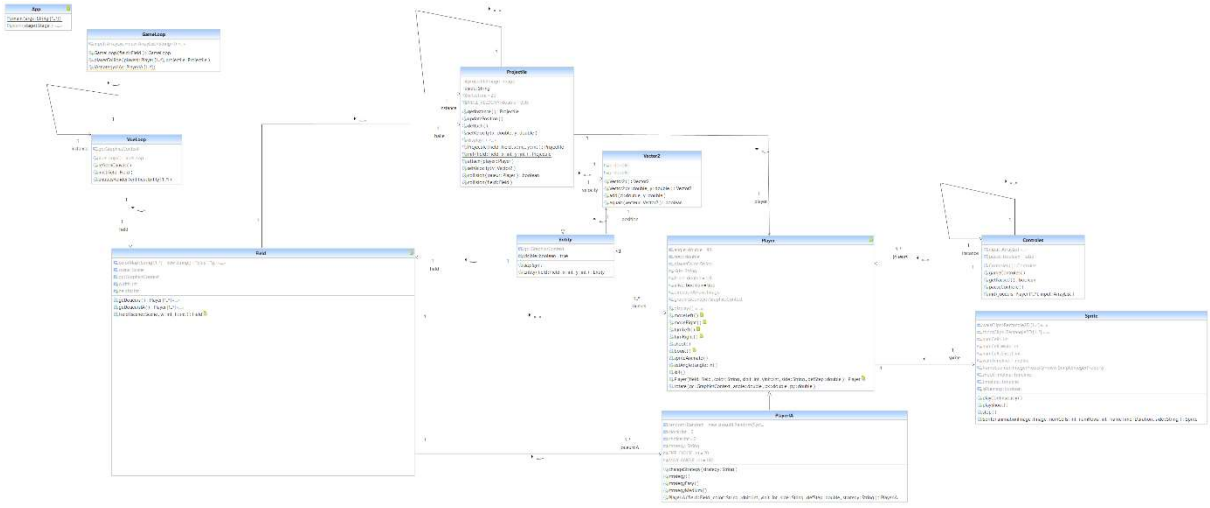
- App :: classe principale de notre Application. N'appartient pas forcément à la vue mais ne s'occupe que de charger les éléments javafx

- Affichage :: classe contenant les méthodes d'affichage des éléments du jeu.

–controleur–

- Controles :: classe gérant les touches clavier
- GameLoop :: boucle principale du jeu, fait appel à la vue

Diagramme de classe



Le diagramme représentant les classes, disposé de la façon la plus compréhensible et claire que nous avons pu

Patterns utilisés

Nous nous sommes servis de plusieurs singletons :

- La classe Controles
- La classe Affichage
- La classe Projectile

Ces choix se justifient car nous voulions n'avoir qu'une seule instance de Controles, un seul Affichage, une seule balle. On aurait pu faire de même pour Field car on a qu'un seul terrain, mais on a préféré ne pas abuser du singleton.

Nous avons également une fabrique abstraite par classe plutôt que par interface (plus simple)

- La classe abstraite Entity

Ce choix se justifie par le fait que nous avions eu besoin de plusieurs objets très similaires ayant tous une base commune. Ces classes sont Player, PlayerIA et Projectile.

PlayerIA et une extension de la classe Player qui est lui-même une extension de la classe Entity.

Nous avons aussi eu l'idée de séparer PlayerIA en différentes sous-classes représentant les différents niveaux de difficulté. Mais cela aurait permis un changement de difficulté en jeu moins dynamique.

Conclusion

Le projet a été mené à bien, le jeu fonctionne et la structure ainsi que la qualité du code nous satisfait. Nous avons fait quelques concessions sur les éléments à intégrer au jeu, après rétrospection il n'y a pas autant d'éléments que nous l'aurions voulu et le jeu reste plutôt rustre. Mais dans l'ensemble le cahier des charges a été respecté à l'exception de l'affichage du score, tous les autres éléments du barème ont été intégrés.

Le résultat est en adéquation avec nos attentes.