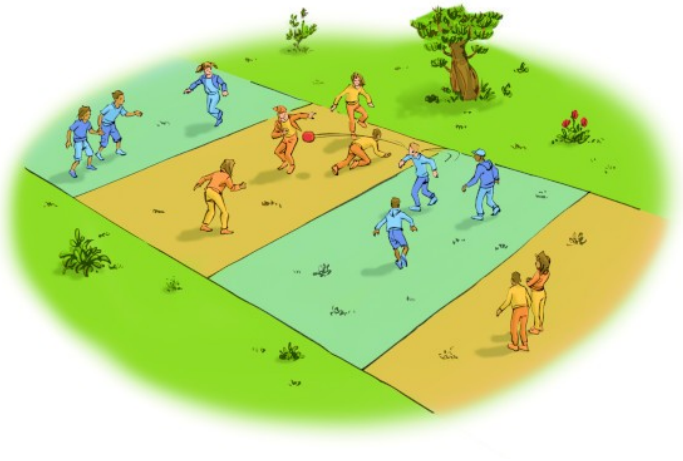
Mathieu ALLARD
Rabah AOUDJ

Le ballon prisonnier



La « balle au prisonnier » ou le « ballon prisonnier » est un jeu opposant deux équipes qui doivent se toucher en lançant des balles.
Dans le cadre du programme du cours de Conception Agile de Projet Informatique, nous avons dû retravailler et améliorer la structure d'un programme informatique en langage Java implémentant numériquement ce jeu. En effet, la structure des fichiers et des classes était complexe et elles étaient fortement inter-dépendantes les une des autres. Nous vous proposons donc de découvrir notre travail disponible sur le dépôt GitHub suivant : https://github.com/Ecla474/conceptionAgile.

Nous avons tout d'abord choisi de poursuivre ce programme en séparant ses différents éléments distincts dans des *packages* Java et des dossiers séparés. La compilation du projet a été effectué tout du long avec le compilateur Maven à l'aide des commandes `*mvn compile*` et `*mvn javafx:run*`. Nous avons donc partitionné le programme comme suit :

- App ;
- Field ;
- Joueur ;
- Projectile ;
- Sprite ;

1. App

L'unique classe, *package* et dossier « App » sert, comme dans le code fournis à initialiser la fenêtre du jeu et d'autre éléments essentiels de la bibliothèque *JavaFx* comme la racine de la scène.

2. Field

Ce *package* et classe, encore une fois unique dans son dossier, gère le terrain de jeu, ses composant et leurs interactions comme par exemple : les joueurs, les projectiles et leurs collisions. Nous avons choisi d'utiliser l'architecture M.V.C. dans cette classe avec les fonctions *controlleur()*, *vuTerrain()* et *vuScore()* qui s'occupent respectivement :

- d'appeler les contrôleurs des classes des éléments présents sur le terrain ;
- puis de les afficher en appelant les « vues » des classes des éléments du terrain ;
- et enfin d'afficher les scores des deux humains pouvant s'affronter sur ce jeu.

3. Joueur

Ce *package* et composé de 3 classes et d'une interface servant à utiliser un *Factory Pattern*.

Tout d'abord, la classe Rectangle est la classe mère de Player qui possède les données membres géométriques de base ainsi que les assesseur et mutateur qui leurs correspondent. Elle sert aussi à représenter la figure géométrique d'un rectangle pour des calculs de collision.
La classe Player implémente un joueur pouvant être contrôlé par un humain. Enfin elle est aussi la mère de la classe Bot qui implémente un joueur autonome. Ces joueurs peuvent être déterminé à leurs créations comme étant :

- statique ;
- en déplacement hasardeux ;
- en train de suivre le joueur humain
- ou encore en train de fuir les Projectiles.

Enfin, ces classes ont également des fonctions *vues()* ou *controleur*() afin de correspondre à l'architecture M.V.C.

4. Sprite

Ce *package* est composé de 3 classes : la classe abstraite « Sprite » qui sert à regrouper les nombreuses caractéristiques communes des deux autres classes spriteExplosion et spritePersonnage qui ont pour but de représenter respectivement l'explosion ayant lieu quand un projectile touche un joueur adverse et lors du déplacement d'un joueur.

5. Annexes

Photo du jeu*

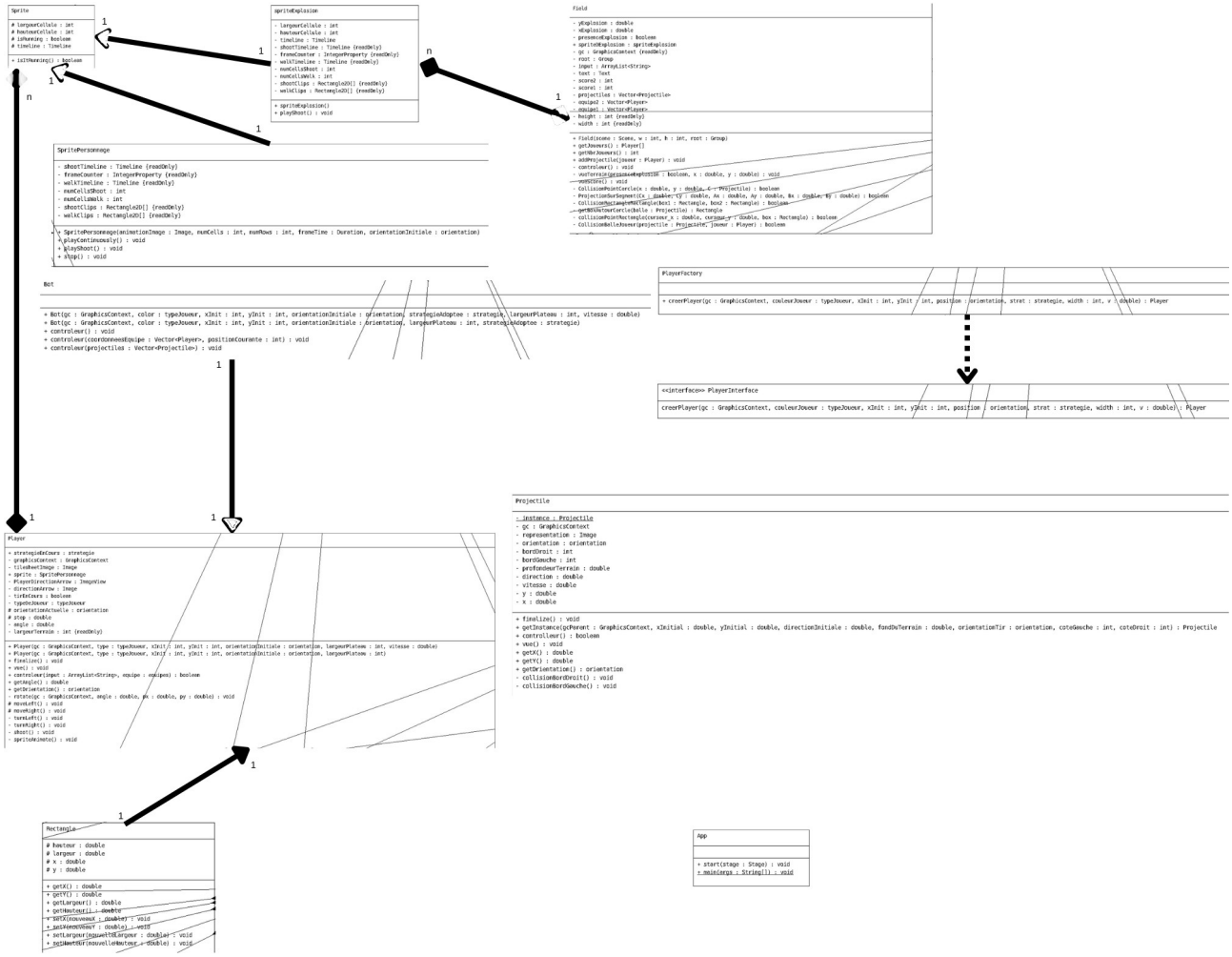


Diagramme des classes simplifié du projet