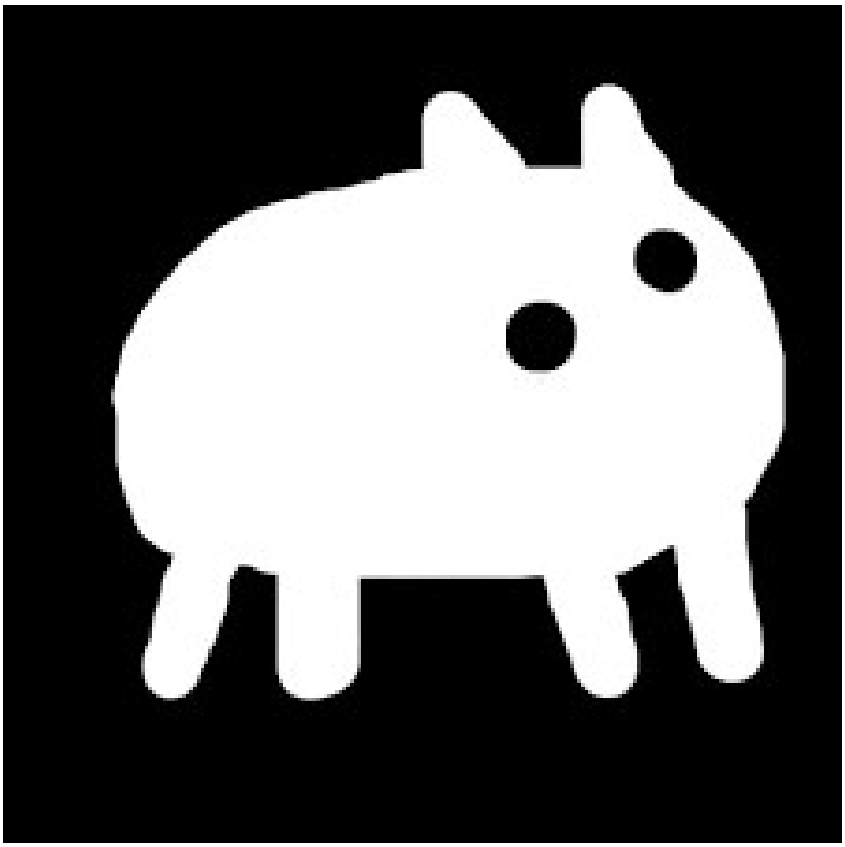
RAPPORT : BABA IS YOU



BARBÉ Romain ESIPE-INFO1
ROBERT Eric

Sommaire :

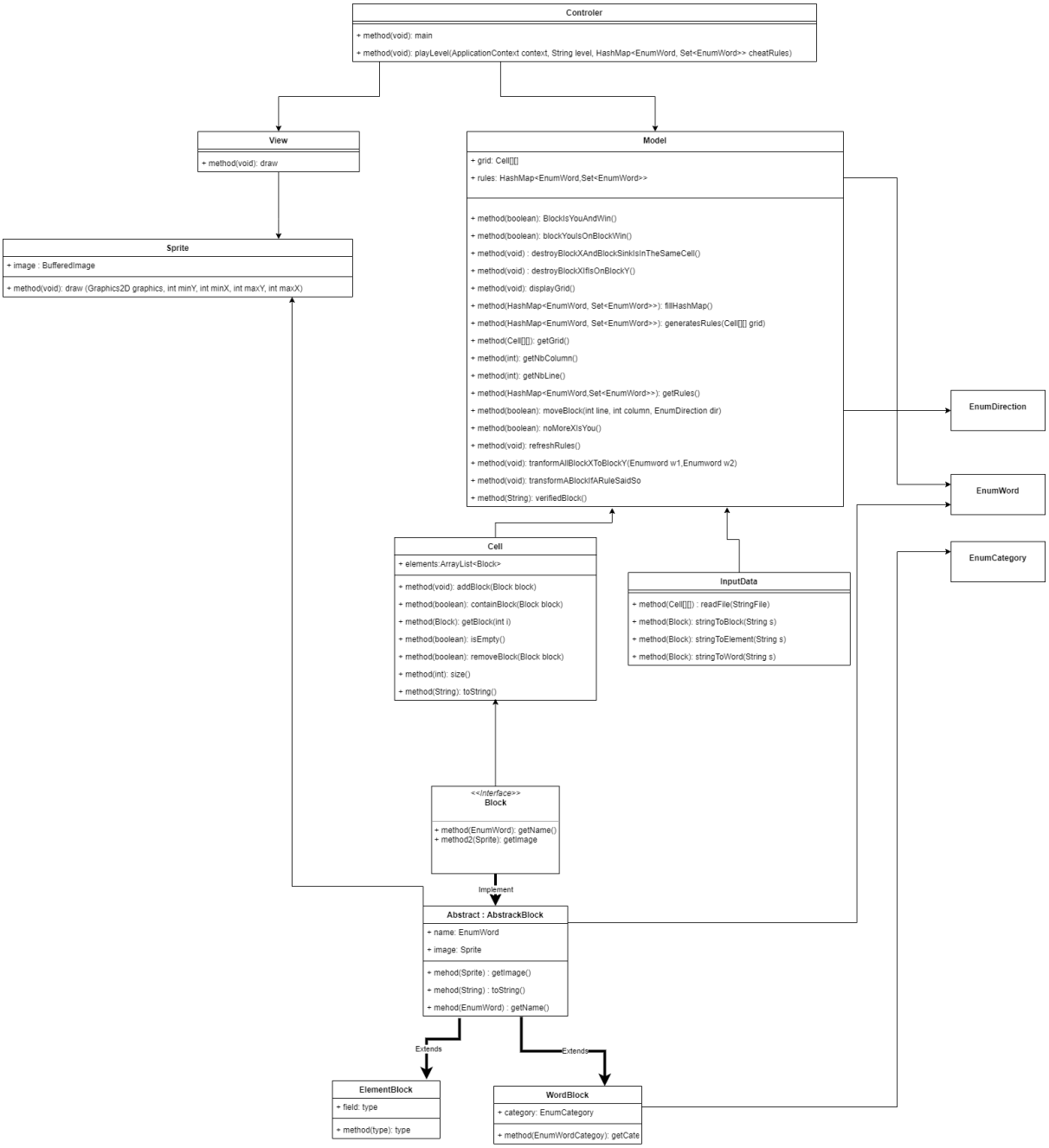
Introduction :

Ce projet à pour but d'utiliser toutes nos connaissances acquises lors des cours de programmation objet Java afin de réaliser le jeu Baba is you.

Le but de ce jeu est de résoudre des énigmes sur un plateau de jeu en déplaçant le joueur pour qu'il atteigne un objectif. La spécificité de ce jeu est le fait que chaque interaction entre le joueur et un élément est dictée par la présence de règles disséminées sur le plateau de jeu.
Ces règles sont formées d'au moins 3 blocs de mot alignés horizontalement ou verticalement sous la forme "nom / opérateur / propriété" ou "non / opérateur / nom". Par exemple, si nous avons la règle "BABA / IS / YOU" sur le plateau de jeu, le joueur pourra déplacer tous les baba du plateau de jeu. De même, si nous avons la règle "WALL / IS / YOU" sur le plateau, le joueur contrôlera tous les murs du plateau. Dans la même logique, si le joueur possède "WALL / IS / STOP", les éléments que le joueur contrôle (les YOU) ne pourront pas passer à travers les murs.
De même, si la règle "WALL / IS / PUSH" est présente, les murs seront poussables par n'importe quel élément qui essaye de le pousser.
Par défaut, si un nom (WALL, BABA, etc…) n'a pas de propriété, tous les autres éléments du jeu pourront passer à travers ces éléments.

La finalité de ce projet a donc été de reproduire ce jeu ainsi que son comportement avec les connaissances que nous avions acquises au cours de ce semestre.

Organisation du programme :



Nous avons décidé de réaliser ce projet sous la forme du modèle MVC (Model - View - Controler) car c'est le design pattern avec lequel nous avions préalablement déjà travaillé et car nous trouvions que cette architecture était adaptée à ce projet.

Dans le package view :

La classe view est la classe permettant l'affichage global du jeu (le dessin des éléments du jeu).
La classe Sprite nous permet de stocker le sprite (l'image) associé à un élément du jeu de le dessiner.

Dans le package model :

La classe Model représente les données brutes du jeu, c'est-à-dire une grille de jeu représentée par un tableau 2D de Cell ainsi qu'une HashMap représentant nos règles (la clé de la map stocke tous les éléments différents du jeu et la valeur de la map stocke toutes les propriétés qui s'appliquent à sa clé).

La classe Cell nous permet de stocker des blocs car on peut avoir plusieurs blocs dans la même Cell.

AbstractBlock est une classe abstraite qui implémente l'interface Block qui permet de représenter un élément de jeu. Il est composé d'un nom et d'un Sprite (nom étant une énumération des éléments possibles). Ces blocs se déclinent en deux versions, les ElementBlock et les WordBlock représentant respectivement les bloc d'élément et les bloc de mot.

Dans le sous package elementList sont stockées toutes nos énumérations pour le jeu.

Dans le sous-package input, la classe InputData nous permet de récupérer les différents niveaux afin de les associer à un modèle.

Améliorations et corrections post-soutenance :

Outre le fait d'avoir rassemblé nos codes différents que nous avions le jour de la soutenance en un code fonctionnant correctement, la principale suggestion qui nous as été faite lors de la soutenance a été de modifier la structure dans laquelle nos règles sont stockées ; on les stockaient dans un simple HashSet contenant un ensemble de deux mots (nom/propriété ou nom/nom).
Suite à cette suggestion, nous avons opté, pour notre structure des règles, pour une HashMap de mot en clé avec un Set de mots en valeur. Ainsi, chaque mot (BABA, WALL, etc…) aura un ensemble de mots comme propriétés (YOU, PUSH, STOP, etc…).

Répartition des tâches :

Pour la répartition des tâches, voici ce que chaque membre du binôme a réalisé ;

Éric ROBERT :

- Mise en place et maintenance de l'architecture MVC
- Mise en place de la structure pour la grille de jeu (classe Cell.java)
- Réalisation de la Javadoc

Romain BARBÉ :

- Réalisation du contrôleur du jeu (=> main du jeu) (classe Controller.java)
- Ajout du fait de pouvoir ajouter des règles pour tricher (classe Controller.java)
- Mise en place de la représentation des blocs du jeu (classes Block.java, AbstractBlock.java, WordBlock.java, ElementBlock.java)
- Mise en place du modèle (classe Model.java) contenant en autre la méthode pour déplacer nos éléments du jeu
- Mise en place de la récupération et de la création des niveaux lors de la lecture de fichiers de niveaux (classe InputData.java)
- Mise en place de l'ensemble du graphique (les méthodes utilisant Zen5 permettant de dessiner le plateau de jeu ainsi que ses éléments) (classes Sprite.java et View.java)
- Création des ressources du jeu (dossier resource) (sprites + niveaux)
- Création du fichier build.xml

Les 2 membres du binôme :

- Mise en place des classes d'énumération du package model.elementList
- Réalisation des fichiers dev.pdf et user.pdf