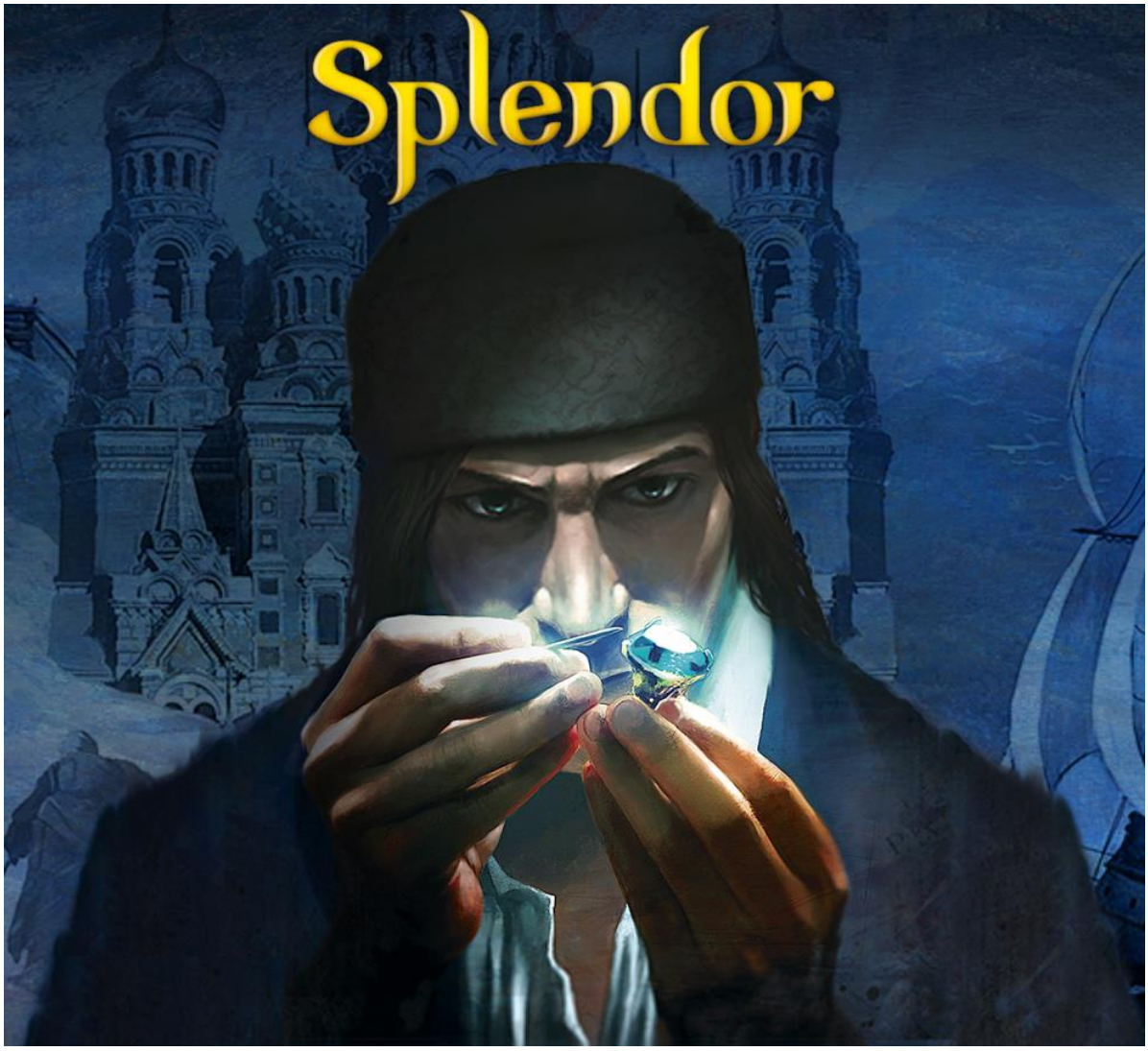
Licence 3 informatique 2021-2022

DAKKAK Inès
AHAMMAD Sadib

PROGRAMMATION EN JAVA MANUEL DÉVELOPPEUR : PROJET SPLENDOR



Introduction

Un objectif important de ce projet a été de coder de façon la plus optimale le jeu Splendor. Voyons dans ce manuel quelle architecture nous avons utilisé pour ce projet.

L'implémentation

Pour l'implémentation, nous avons décidé de lister chaque objet du jeu, et d'en faire un record ou une classe.
Pour des raisons de lisibilités nous les avons placés dans différents packages.

Les differents packages

Card

Nous avons défini les cartes par leur niveau, les points de prestige qu'elles apportent, la couleur bonus et leur prix sous forme de Tokens (hashmap).

Afin de représenter toutes les cartes, dans une classe CardInventory, nous avons créé une fonction getCardsFromFile qui lit un fichier CSV, et stock les informations des cartes dans une HashMap qui a pour clé le niveau de la carte et en valeur une ArrayList contenant toutes les cartes de ce niveau.

Noble

Les nobles sont définis par des entiers points de prestige, leur nom et un Tokens pour leur valeur, dans le record Noble.
De la même façon que pour les cartes, nous avons lu un fichier CSV contenant les informations des cartes nobles, mais que nous avons stockées dans une ArrayList cette fois-ci, dans la class NobleInventory.

Player

Nous avons défini le joueur par tous les objets qu'il possède (cartes réservées et achetées, points de prestige, bonus, jetons et nobles).
Dans ce package nous avons mis les classes relatives aux actions que le joueur peut effectuer (donc acheter des cartes, les réserver et recevoir la visite de nobles), que nous allons ensuite utiliser dans le package game.

Game

Ce package va inclure les classes nécessaires aux fonctionnement du jeux (les actions à réaliser, le contrôle de saisi pour la version terminale, le menu etc).
Avec les classes situées dans le package game, nous allons également afficher dans le terminal tout le plateau, ainsi que le joueur qui joue et les objets qu'il a en sa possession.

Token

Pour définir les tokens, nous avons décidé de faire une classe énumérée qui va définir les jetons comme des couleurs. Puis dans une classe Tokens, nous créons une HashMap qui regroupent les jetons par couleur et quantité.

Tray

Pour le plateau, nous avons une classe qui va permettre de rassembler sous forme de grille les cartes par niveau, les nobles et les jetons de le tas de jetons.

La partie graphique et la partie terminal

Quand on lance le programme, l'utilisateur peut choisir entre la version terminal et graphique en saisissant un nombre (1 ou 2).

Partie terminal.

La partie dans le terminal va être définie dans la classe Terminal dans le package game. Nous allons définir dedans les fonctions permettant l'affichage du jeu ainsi que l'utilisation des méthodes permettant de jouer au jeu (en récupérant les nombres saisies par l'utilisateur par exemple, qui vont être associés à des actions).

Partie graphique

Pour la partie graphique, nous avons défini toutes les classes utiles dans le packages game.
Ainsi, les classes Display, GridDisplay (permettre l'affichage de la grille de cartes sur le plateau, NobleDisplay (pour l'affichage des nobles), et PlayerDisplay (pour l'affichage des cartes du joueurs, de ses tokens et de ses cartes réservées et achetées).

Amélioration et correction apportée par la soutenance β

Les Tokens

L'élément le plus important que nous avons modifié est l'implémentation des tokens. En effet, nous avons créer une classe enum Token, qui représente un ensemble de couleurs. Afin de manipuler plus facilement les tokens, dans une classe Tokens, on crée une Hashmap qui prend en clé le Token (donc la couleur) et en valeur un Integer (la quantité de jeton correspondant à cette couleur).

Le plateau et modulation générale du projet

Dans notre ancien rendu, la classe Tray était une classe « God », nous avons donc séparer le joueur d'un côté et le plateau.

Utilisation des fichiers

Pour les nobles ainsi que les cartes, nous avons récupérer une liste d'information via un fichier CSV, et non codé en dur le nom des nobles comme dans l'ancienne version.

La règle des deux tokens de même couleur

Concernant la prise de jeton, nous l'avons contrôlée de cette manière : l'action de prendre 3 couleurs différentes est vérifiée : on demande au joueur de sélectionner à nouveau 3 jetons jusqu'à ce qu'ils soient de trois couleurs distinctes.

Fonctions en particulier

Pour la fonction qui nous permettait de distribuer les cartes sur le plateau, nous avons seulement mélangé les cartes (shuffle), puis on les distribue, au lieu de les parcourir.

Toutes les autres fonctions qui n'allaient pas de type get, celles qui renvoyaient des HashMap etc ont été supprimées avec la nouvelle implémentation des Tokens.

Piste d'amélioration

Pour la partie graphique, l'affichage fonctionne bien mais nous n'avons pas su faire/pas eu le temps de coordonner les actions avec l'affichage, nous ne pouvons donc pas jouer en graphique, juste afficher le jeu.
Ensuite pour la partie terminal, nous n'avons pas fait l'action de réserver à partir de l'inventaire et acheter cette carte.

Conclusion

Pour conclure, ce projet nous a permis de mettre en pratique de façon ludique tout ce que nous avons pu acquérir au cours de ce semestre en java.
Nous aurions voulu optimiser notre temps afin de finir la partie graphique.