Université de Bordeaux

Architecture Logicielle : rapport de Projet

Jeu de l'oie

Anthony CHARLY - Yassir DAOUDI

17 mai 2021

Table des matières

1 Introduction

Dans le rapport présent, est décrit le projet réalisé dans le cours d'Architecture Logicielle pour formaliser la situation suivante : Créer un jeu que nous avons selectionné avec une interface graphique, en utilisant pour cela le mécanisme d'extension du framework (abstract factory, visitor, decorator, composite, ...) que nous avons mis en place.

2 Structure du projet

2.1 Contexte

Le jeu de l'oie est un jeu de société de parcours où l'on déplace des pions en fonction des résultats de deux dés. Traditionnellement, ce jeu de hasard pur comprend plusieurs cases disposées en spirale enroulée vers l'intérieur et comportant un certain nombre de pièges. Le but est d'arriver le premier à la dernière case.

2.1.1 Design Pattern

Les design patterns, ou "patron de conception", sont des méthodes utilisées en développement logiciel. Ils permettent d'optimiser, de clarifier du code informatique et de le rendre plus robuste, en répondant principalement à une problématique posée par les choix de conceptions. Chaque design pattern que nous avons utilisés, décrit ci-dessous, répond à un problème.

Abstract factory : La fabrique abstraite nous fournit une interface pour la création de famille d'objets apparentés ou interdépendants. Cela nous a permis de créer des familles de produits complètes sans avoir à préciser leurs classes concrètes.Nos deux familles sont AgeFutureFactory et AgeMiddleFactory (réutilisation de l'abstract factory du framework)

Prototype : Le design pattern prototype a été mis en place pour pouvoir cloner l'état d'un joueur afin de l'utiliser dans le design pattern memento.

Decorator : Le decorateur nous a permis d'attacher dynamiquement des responsabilités à un objet. En effet, pour la création de Box (malusBoxFight , bonusBoxUnit, bonusBoxForwad etc ...) le décorateur rends notre implementation dynamique et plus statique , permettant d'étendre les fonctionnalités .Exemple, un joueur peut tomber sur case malusBoxUnit(malusBoxBack(Neutral)).

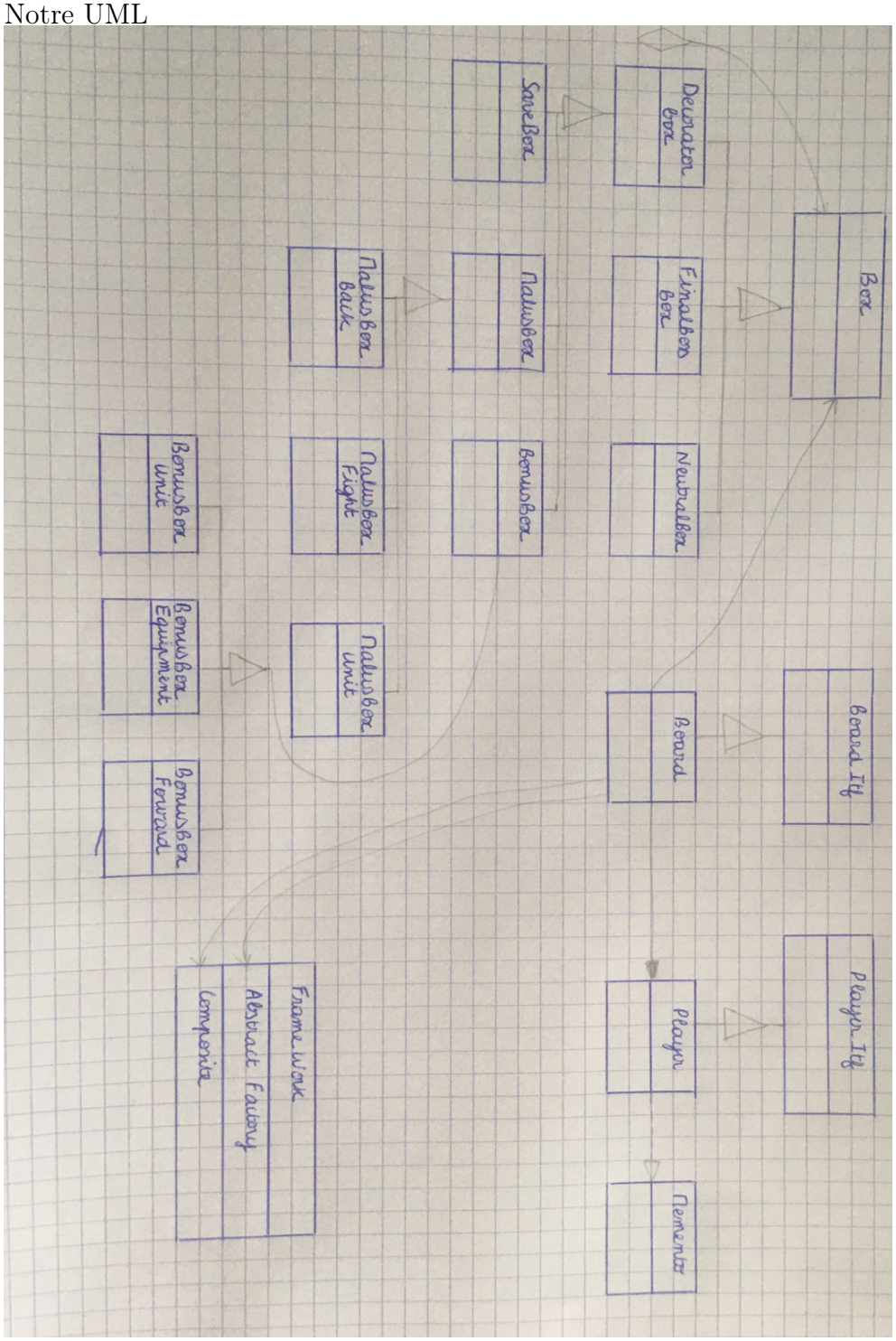
Composite : Le composite nous a permis de traiter de la même manière les objets individuels et les combinaisons de ceux-ci. Cela nous a simplifié la tâche pour l'utilisation des case Box (ajouter une unité ou enveler une unité), réutilisation du framework.

Memento : Le choix d'intégrer le design pattern memento a permis de résoudre le problème de restaurer l'état interne d'un objet en évitant de violer l'encapsulation. Notre jeu dispose d'une case spéciale qui une fois franchis, permet au joueur de restaurer l'état dans lequel il était lorsqu'il avait franchis la case spéciale une fois revenue à la case départ.

Singleton : Pour éviter plus instantion du board dans le main.

2.2 UML

Notre UML



3 Conclusion

Pour conclure, ce projet nous a permis de mettre en pratique le framework vue durant les tp, ainsi que son utilisation dans un nouveau projet. Les difficultés rencontrées dans la réalisation du jeu de l'oie n'a pas été du point de vue du framework (application des pattern du framework) mais beaucoup plus dans l'utilisation et la compréhention de JavaFx. En effet, nous voulions intégrer le design pattern builder dans notre projet mais par manque de maîtrise (Javafx), nous avons du faire machine arrière.

4 Contenu du projet

Vous trouverez dans le projet, notre implement , notre rapport en format pdf ainsi qu'une vidéo simulant une partie du jeu de l'oie(execution du code). Néanmoins nous n'avons pas réussit à vous fournir un jar executable.