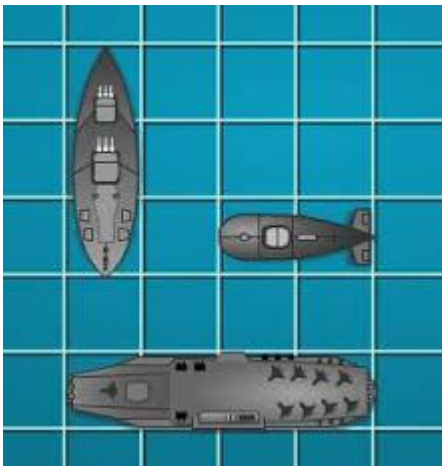
GIANG André

LEXA Corentin

LY Hue-Nam

RAPPORT DE PROJET DESIGN PATTERN



SUJET : la réalisation d'un jeu « Bataille naval » sur ordinateur. L'application est mono-utilisateur et le joueur est opposé à l'ordinateur. Le jeu doit disposer d'une interface graphique et offrir la possibilité de sauvegarder une partie et la reprendre plus tard au redémarrage de l'application.

M1-INFO 2018-2019 UNIVERSITE DE LORRAINE

INTRODUCTION

Le but de ce projet est la mise en pratique des connaissances vu en cours de design pattern. Le langage utilisé pour développer ce projet est JAVA.

Les design patterns appliqués dans ce projet sont :

- MVC.
- AbstractFactory.
- Strategy.
- DAO.
- Singleton.

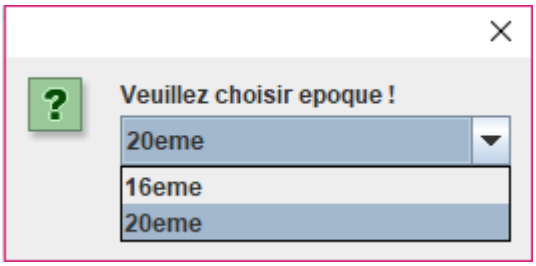
On a décidé d'utiliser deux IDE : Eclipse et Intellij.

Pour la mise en commun de nos travaux respectifs, on a utilisé le logiciel de versionnage GIT que vous pouvez consulter via le lien suivant pour le code source :
https://github.com/giang2u/DP_2018_BlackPearl.

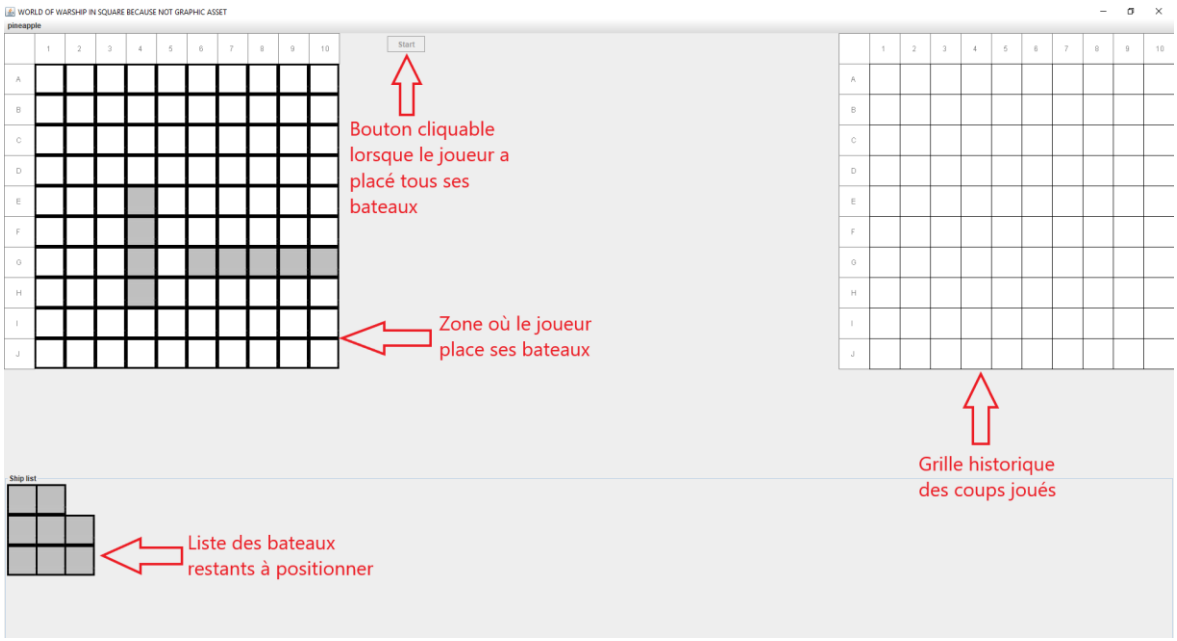
Lancement du Jeu :

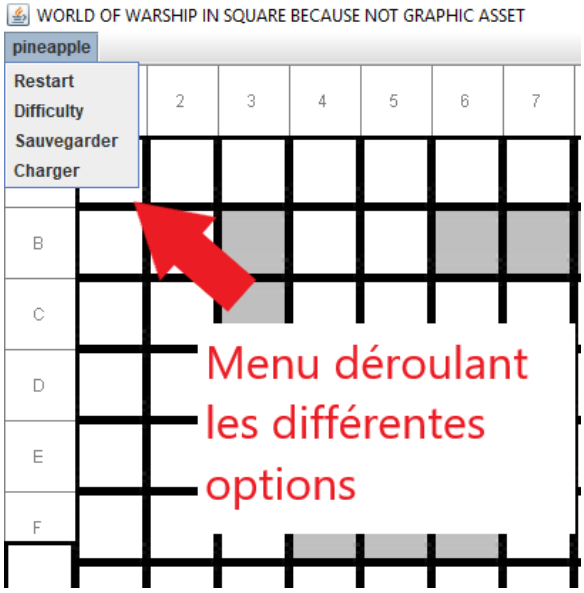
UTILISATION DU JEU

1) Sélectionner « époque » :



2) Interface de jeu :





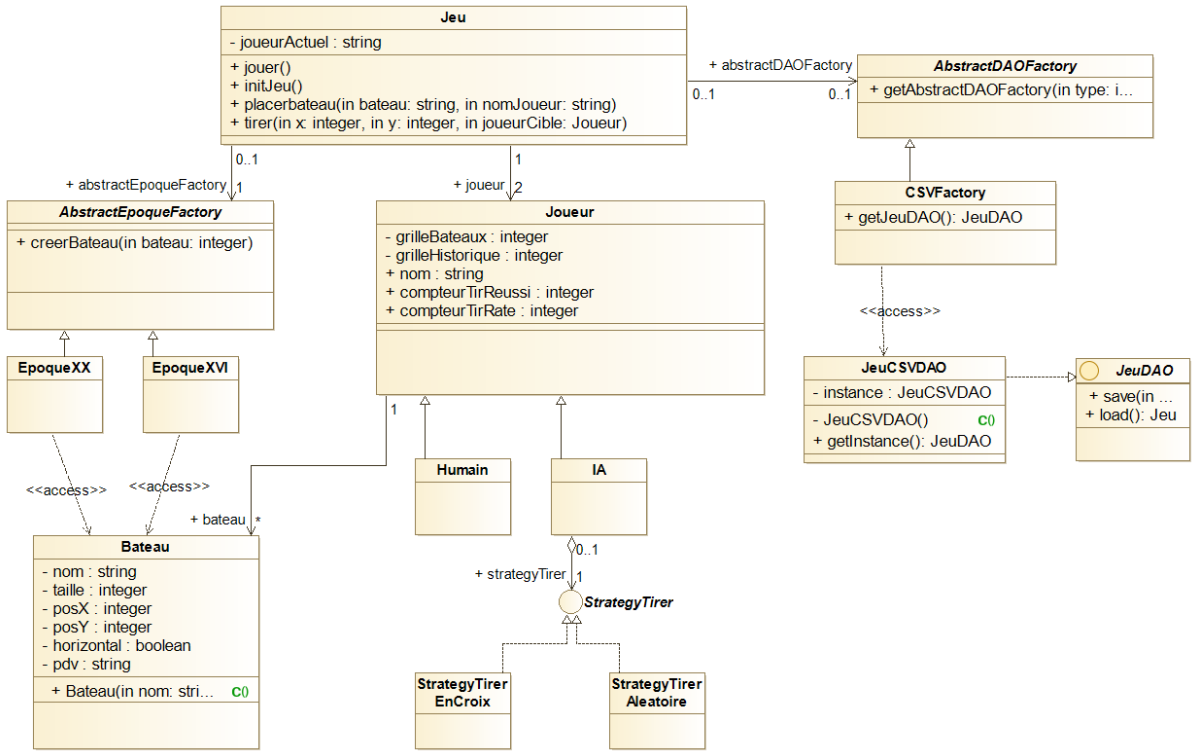
Lorsque le joueur positionne son bateau, il est horizontal, en cliquant à nouveau sur le bateau il pourra le placer en position vertical.

Le jeu s’arrête lorsqu’une des deux joueurs n’a plus de bateau sur sa grille.

CONCEPTION

Diagramme de classes :

MODEL



Ce diagramme montre la partie model de notre conception MVC. Elle contient aussi les autres design patterns évoqués précédemment.

- ABSTRACT FACTORY : ce design pattern est utilisé pour implémenter l'époque, la vie des bateaux dépend du choix de l'époque fait par l'utilisateur au début de la partie.

- STRATEGY : ce design pattern est utilisé pour implémenter les tirs de l'ordinateur. L'ordinateur peut tirer de façon aléatoire ou en croix.

- DAO : ce design pattern est utilisé pour gérer la sauvegarde de la partie en cours.

Pour faciliter l'arborescence des fichiers du projet, nous avons créé les packages tels que :

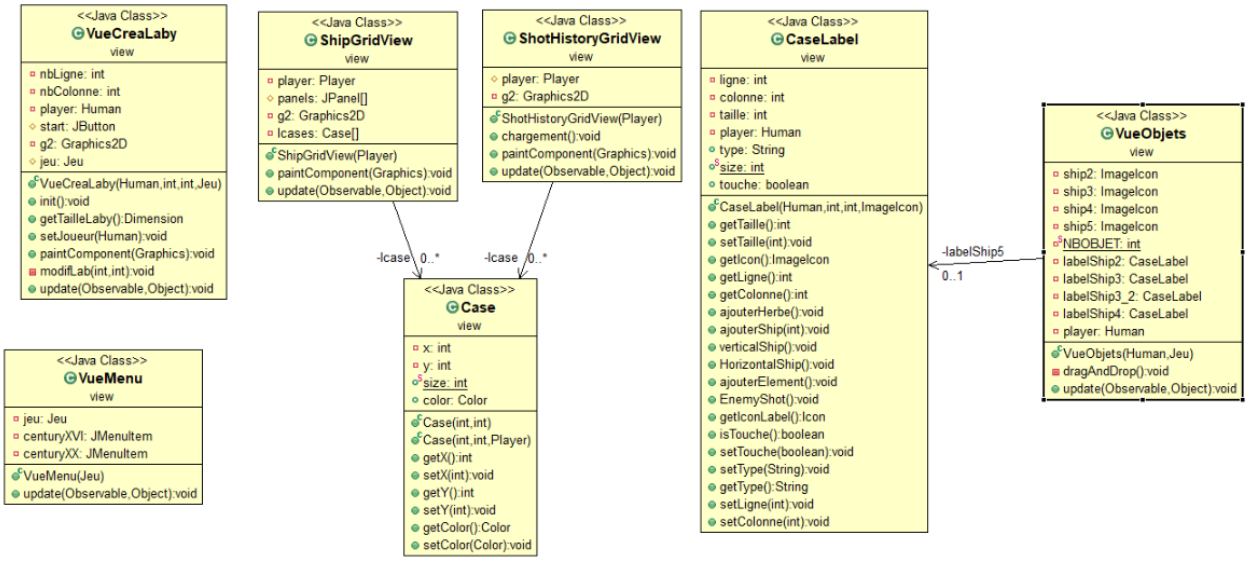
ship : contient toutes les implantations des bateaux.

epoch : contient les implantations de l'époque.

player : contient les implantations des joueurs.

player/strategy : contient les implantations des tirs de l'ordinateur.

VUE



Dans le package Vue :

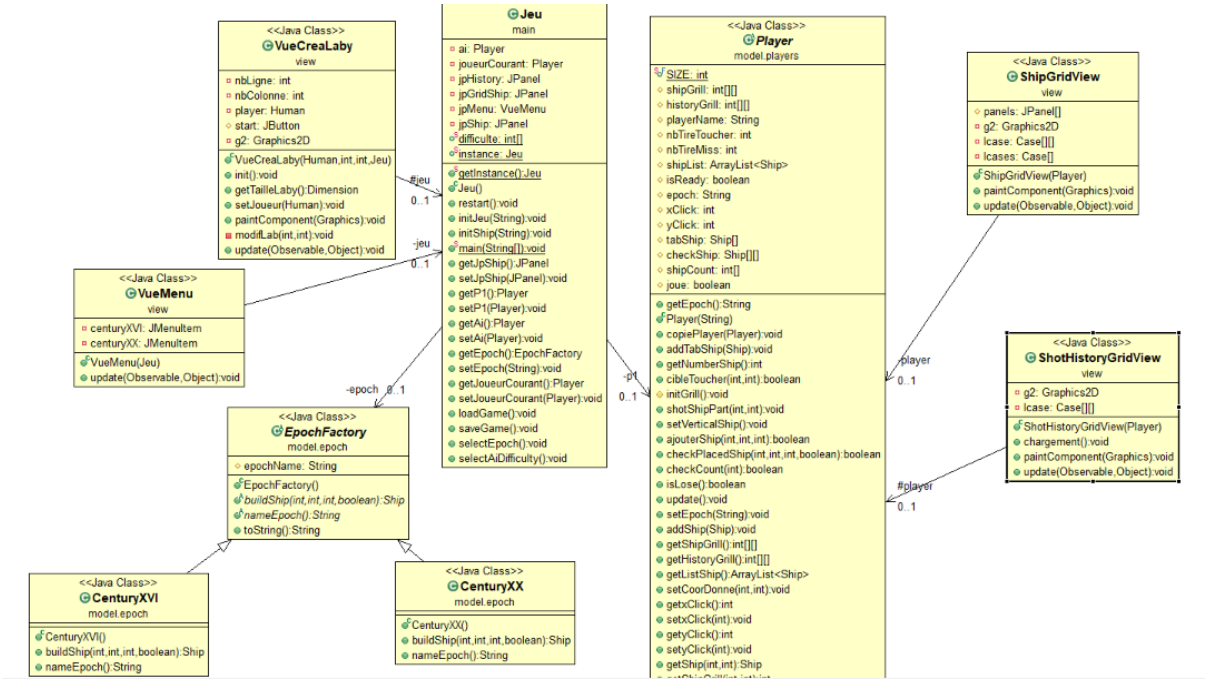
Les deux classes ShotHistoryGridView et VueCreaLaby sont les deux grilles de jeu contenant un ensemble de cases.

VueObjets contient des CaseLabel : c'est cette classe qui va implémenter l'interface graphique de vue des bateaux que le joueur doit poser.

VueCreaLaby : Elle est la classe qui contient la grille où le joueur va poser ses bateaux. Elle va permettre la mise à jour de ceux-ci dans la grille, et ne les affichera plus dans la VueObjets.

VueMenu : C'est un menu déroulant pour que l'utilisateur puisse sélectionner la difficulté, sauvegarder sa partie en cours ou encore charger une précédente partie.

La classe main Jeu :



Il n'y a pas de changement majeur du côté modèle depuis la première version. Mais on a ajouté des classes vues dans le but de mise en relation MVC.

Possibilité d'évolution

De nouvelles fonctionnalités peuvent être ajoutées telles que :

- Plus d'époques.
- Plus de caractéristiques sur les bateaux comme des boucliers, des niveaux de puissances de feu, etc.
- Une version distribuée (RMI).
- Ajouter de nouvelles stratégies de tir pour l'IA.
- Une meilleure charte graphique (images de bateaux pour chaque époque et état, une map plus jolie).
- Des variantes dans les règles de jeu (belges).