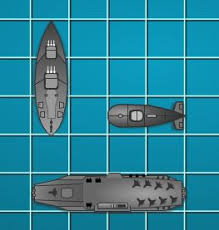
GIANG Andre

LEXA Corentin

LE 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-INFO2018-2019 UNIVESITE LORRAINE

**INTRODUCTION**

Le but de ce projet est mis en pratique des connaissances sur le design pattern. Le langage utilise pour développer ce projet est le JAVA.

Les design patterns appliquer dans ce projet sont :

* MVC.
* AbstracFactory.
* Strategy.
* DAO.
* Singleton.

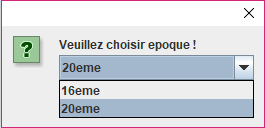
On a décidé d’utiliser deux IDE : Eclipse et Intellij.

Pour mise en commune notre parti, on a utilisé le logiciel de version GIT que vous pouvez consulter via le lien suivant pour le code source : <https://github.com/giang2u/DP_2018_BlackPearl>.

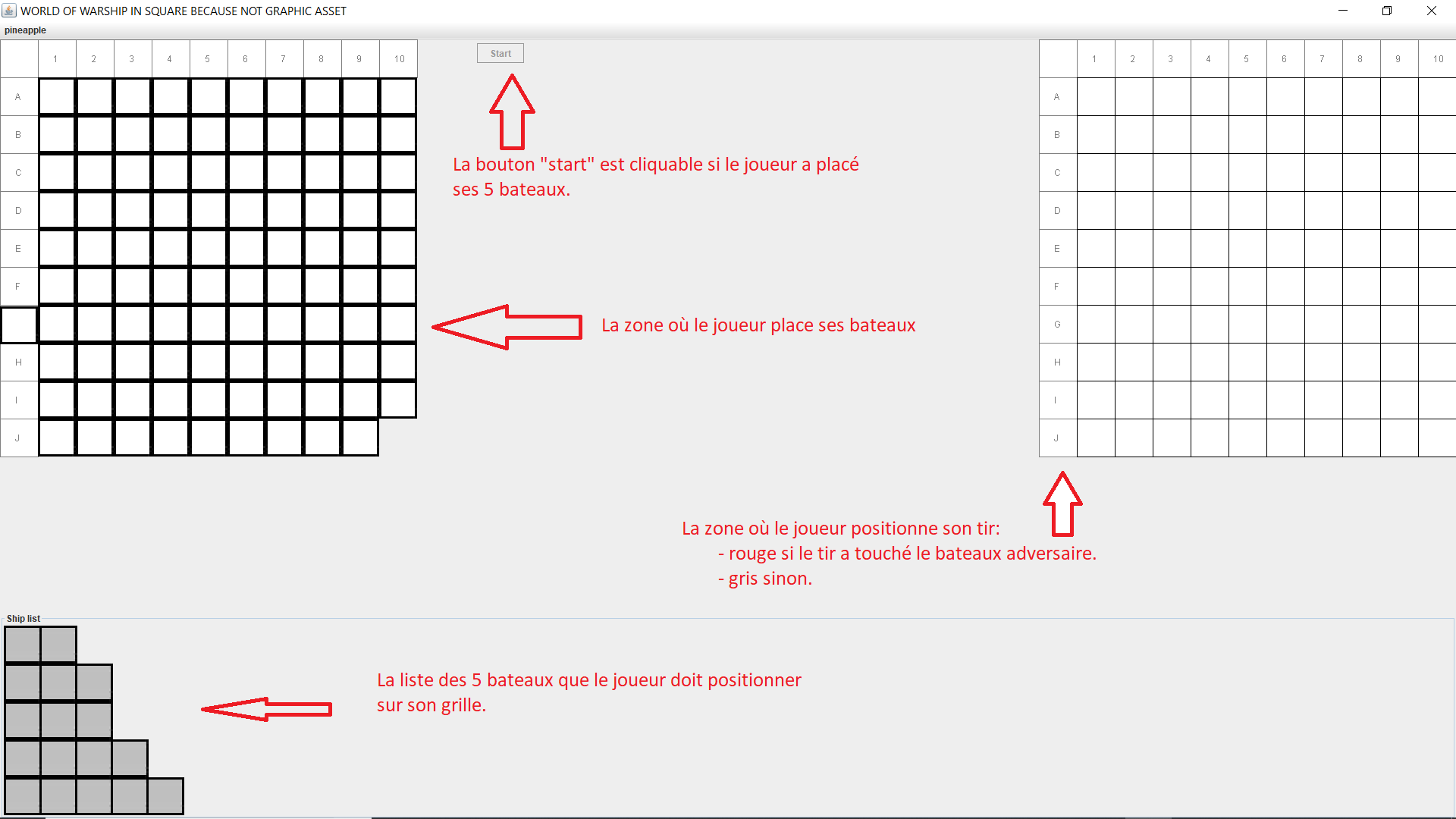
Lancement du Jeu :

**UTILISATION DU JEU**

1. Sélectionner « époque » :

****

1. Interface de jeu :



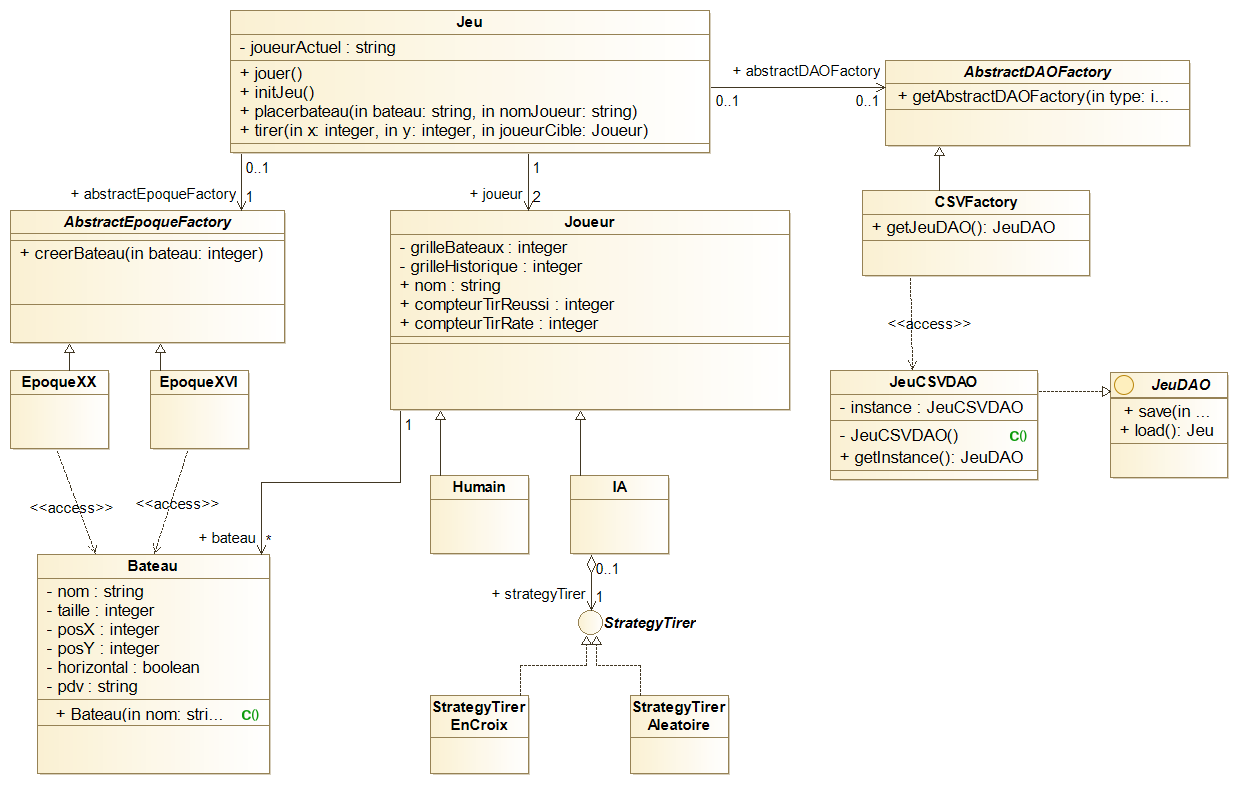
Lorsque le joueur positionne son bateau, il est horizontal, clique que à nouveau sur le bateau pour le mettre en vertical.

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

**CONCEPTION**

Diagramme de class :

MODEL



Ce diagramme est la partie model que le conception MVC. Elle contient aussi les autre design pattern qu’on a présenté dans l’introduction.

* ABSTRAC FACTORY : ce design pattern est utilisé pour implémenter l’époque, les vies des bateaux dépendent alors le choix de l’époque que l’utilisateur a sélectionné 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 de façon en croix.
* DAO : ce design pattern est utilisé pour gérer la sauvegarde du jeu.

Pour faciliter l’arborescence des fichiers de projet, nous avons créé les packages tel que :

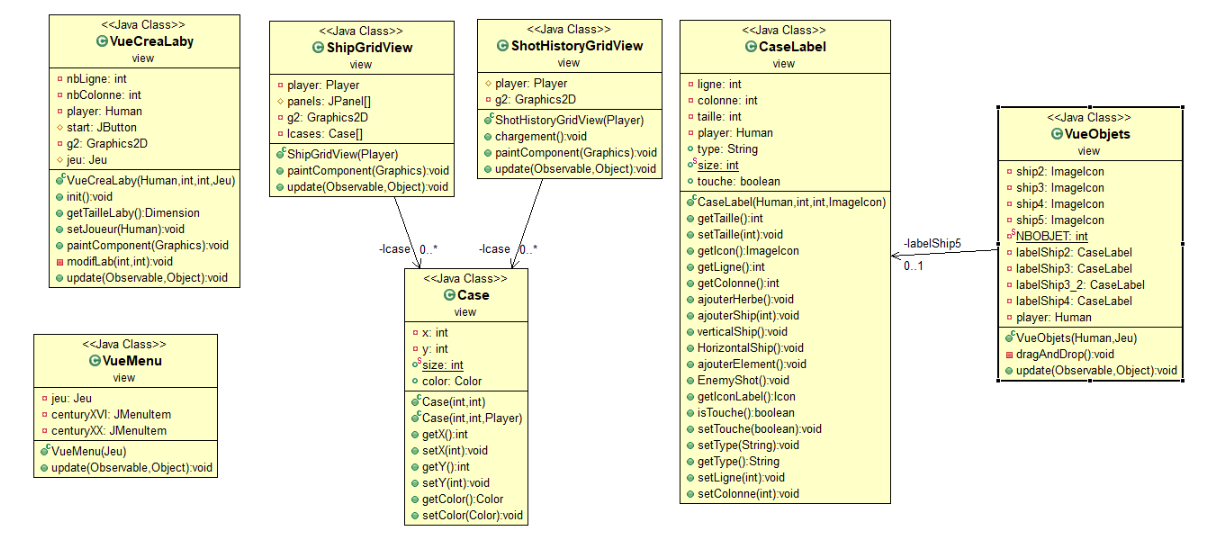
ship : contient tous les implantations des bateaux.

epoch : contient les implantations de l’ époque.

player : contient les implantations les joueurs.

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

**VUE**

****

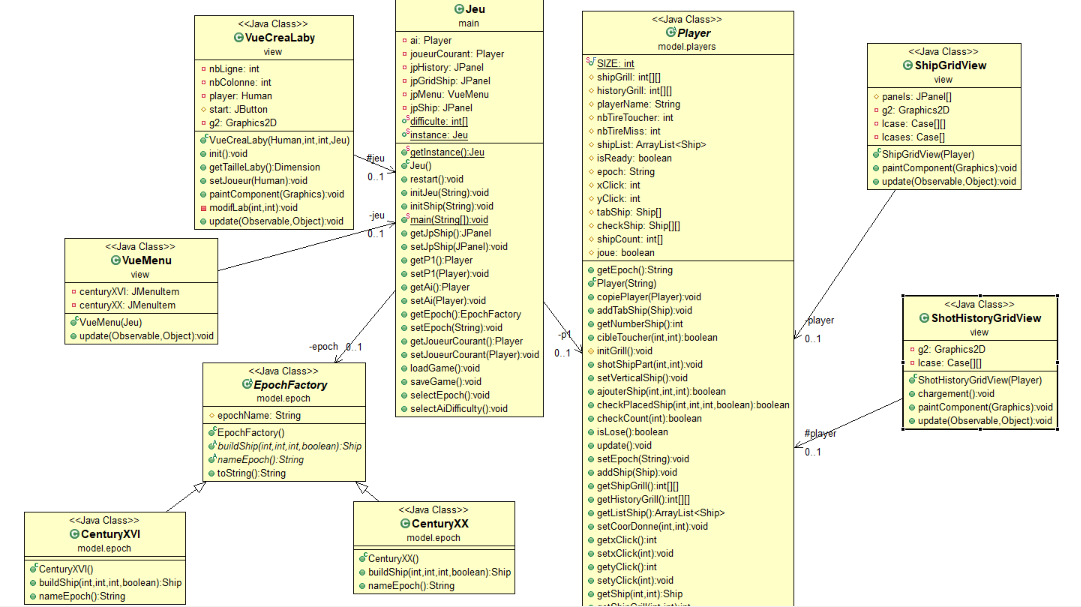
Dans le package Vue :

Les deux class ShotHistoryGridView, ShipGridView sont deux grilles de jeu contenir un ensemble de Case.

VueObjets contient la CaseLabel : c’est cette class qui va implémenter l’interface de vue de bateaux que le joueur doit poser dans la ShipGridView.

VueCreaLaby : est un ensemble des vues entrent ShipGridView et VueObjets c’est-à-dire, une mise à jour lorsque le joueur pose les bateaux dans la grille, elle n’affiche plus dans la VueObjets

VueMenu : pour implémenter le côté Menu qu’utilisateur va sélectionner la difficulté …

La class main Jeu : 

Il n’y a pas de changement majeur du côté modèle depuis la première version. Mais on a ajouté des class vue dans le but de mise en relation IHM.

**Possibilité d’évolution**

Si le temps nous permettre, on pourra ajouter

* Plus de l’époque.
* Plus de caractéristique sur les bateaux tel que : les boucliers.
* RMI.
* Ajouter une nouvelle stratégie de tire ordinateur : tire anticiper c’est-à-dire tirer 2 caisse plus loin par rapport aux premières tires.