2019-2020

PS5-PROJET

Citadelles

ÉQUIPE JTHS :

Julien N'DIAYE
Titouan LE MAO
Hamza AYOUB
Sana DIBE

SOMMAIRE

SYNTHÈSE DU PROJET

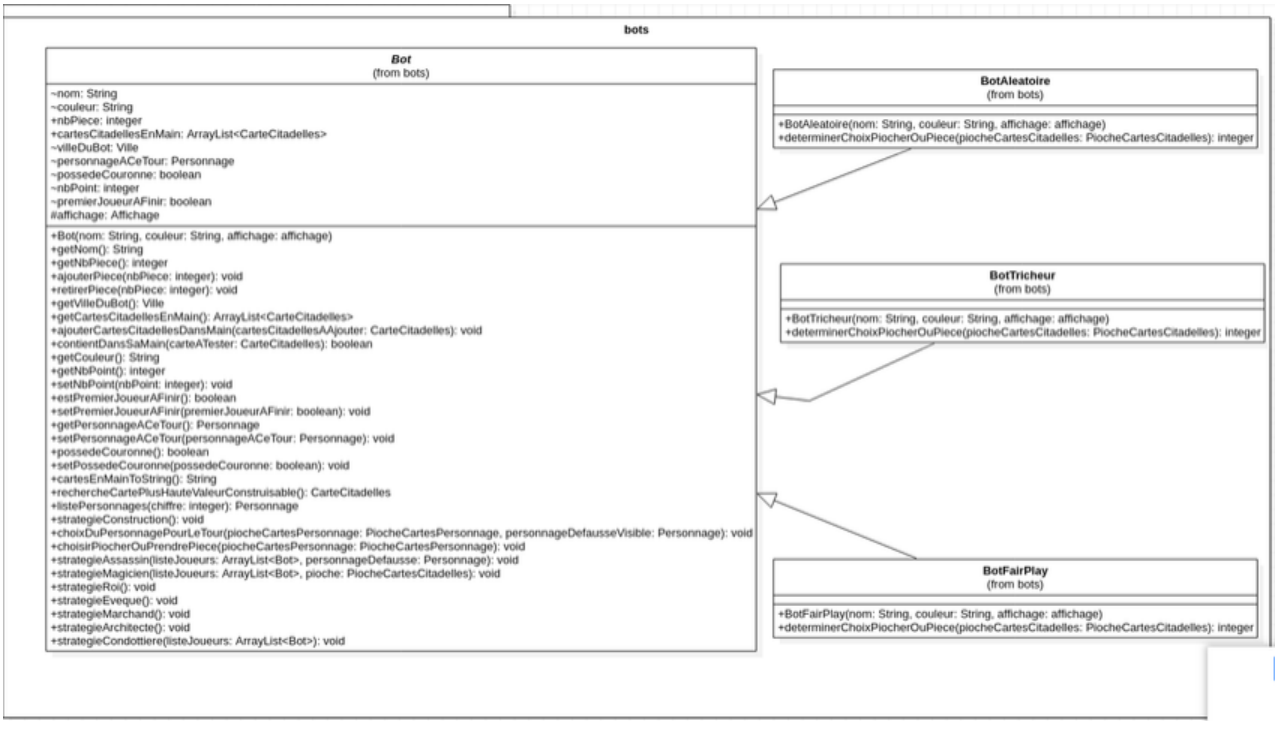
À travers ce projet, nous avons implémenté citadelles, un jeu de société, sur java pour en avoir une version informatisée.
Le jeu n'est pas jouable entre humain, seulement des bots s'affrontent reprenant soit des concepts aléatoires ou bien des concepts humains par l'intermédiaire de décisions réfléchies pour les bots les plus perfectionnés.
Notre version informatisée de citadelles respecte toutes les règles de citadelles, étant conforme au jeu et à la réalité elle y glisse quelques bots tricheurs par ci par là, ceux étant bien évidemment retirable mais beaucoup plus fort que les autres !
Citadelles est un jeu non résolu, c'est-à-dire qu'il n'existe aucune façon de prédire à partir de n'importe quelle position dans la partie l'issue de cette dernière si les 2 joueurs, ici bots jouent à la perfection. Cela nous à conduit à faire des Test bench pour en déduire quelle stratégie était la meilleure notamment pour la gestion de sa main et du choix du personnage. En effet notre projet nous a permis d'optimiser certaines stratégies de citadelles c'est pour cela que nous avons confiance en notre projet tout en gardant un esprit critique sur celui-ci.

En ce qui concerne le jeu, chaque bot dirige une ville en essayant de l'améliorer au fil de la partie. Chaque bot possède un nombre de pièce qu'il se fait attribuer au début par une banque et peut en réclamer à chaque début de tour. Il peut aussi choisir un personnage parmi une liste de personnage disponible à chaque début de tour tout comme choisir une carte citadelles s'il ne prend pas de pièce d'or. Une carte citadelle représente potentiellement un futur quartier qu'un bot pourra construire dans sa ville. À 8 quartiers distincts construit dans une ville, le jeu effectue la fin de son dernier et ultime tour, à l'issue de celui-ci est déterminé le gagnant.
Sur ce projet nous avons volontairement décidé de la non mise en place d'une banque. En effet le jeu citadelles étant un jeu de construction, nous n'avons pas trouvé utile après de multiple Test bench de laisser les stratégies d'économisation aux bots. Ainsi chacun construit le meilleur quartier dès qu'il peut. Du haut de ses 5 bots implémentable max, jamais la somme de toutes leurs économies ne dépasse 30 pièces ce qui respecte bien les règles du jeu.

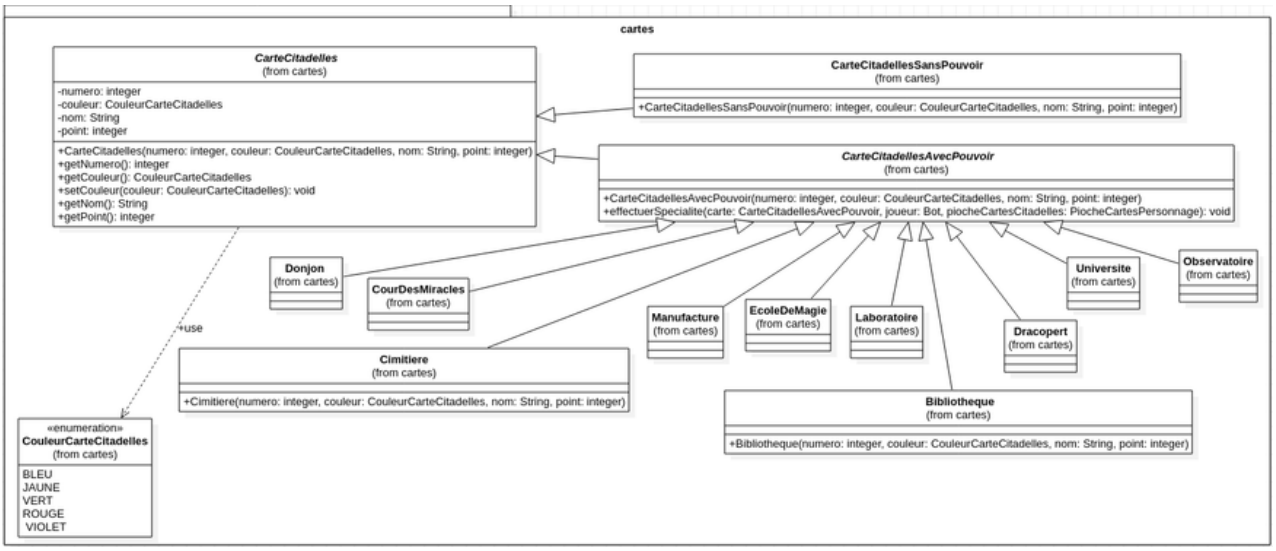
CRITIQUE DU CODE

Lorsque nous avons codé notre projet, nous étions relativement inexpérimenté et partis sur de mauvaises bases. S'en est suivit de multiples réorganisations du code. Donc un point négatif est que notre code contient quelques redondances car nous nous sommes plus axés sur sa structure que sur son contenu.
La création de bot étant un axe principal pour la réalisation du code, nous nous sommes concentrés dessus. Leurs codes sont donc de bonne qualité. À contrario, étant pris par le temps, l'implémentation des cartes violettes s'est faite relativement rapidement et nous sommes convaincus que nous aurions pu optimiser le code d'une meilleure façon.
Le moteur de jeu quant à lui est très bien implémenté se découpant même en 2 classes.
Les personnages, les cartes citadelles et les pioches étant relativement simple à implémenter (appart le personnage Condottière) nous sommes confiants de leurs qualités.
Tous les tests des fonctions ont été réalisés (sauf sur les fonctions bidons ex : la fonction get). Mais où cela s'est compliqué, était au niveau des tests. En effet le jeu citadelle repose vachement sur l'aléatoire et cela fut dur pour nous de faire tous les bons tests adéquats en géré le côté aléatoire.
Pour ce qui est de l'utilisation des concepts objets cela a été bien réalisé tout comme la répartition des responsabilités, comme expliqué plus haut, nous avons concentré la majorité de nos efforts sur la réorganisation du code.
Voyez par exemple la structure des packages ci-dessous :

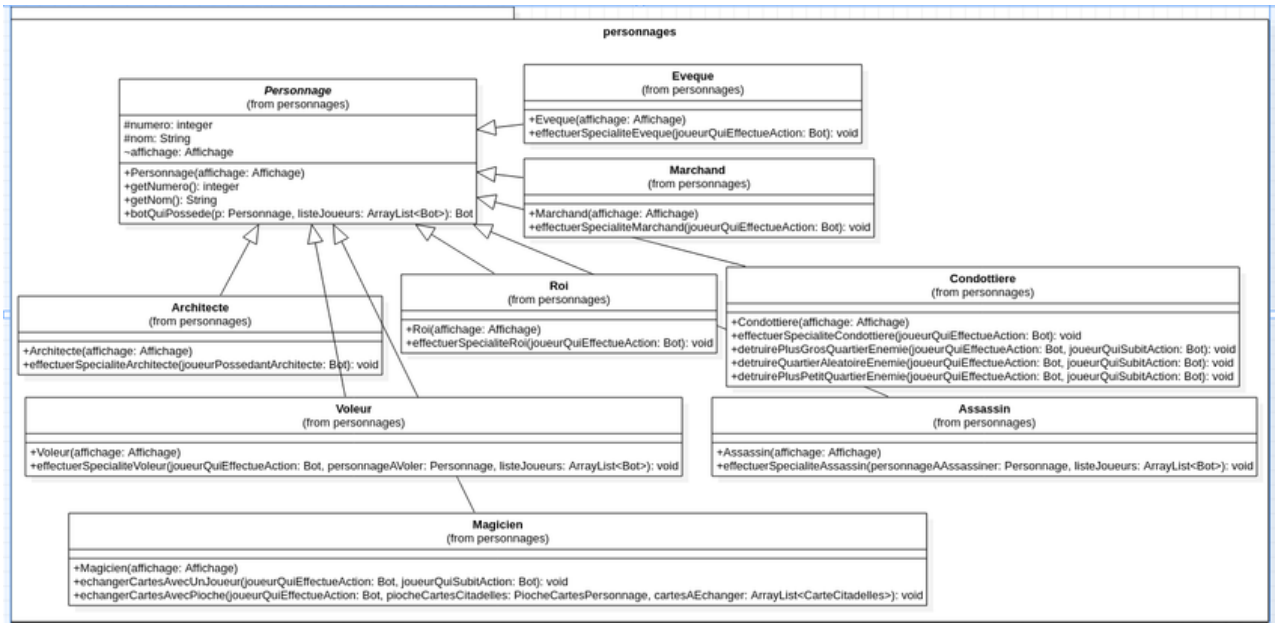
Package bots :



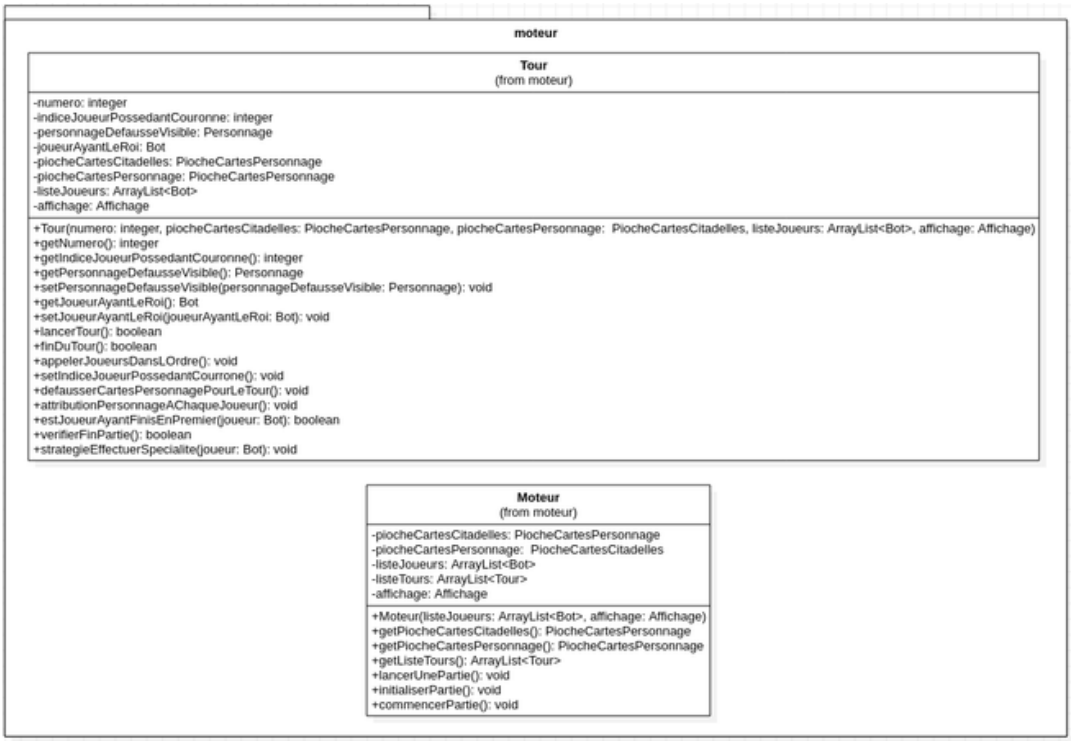
Package cartes :



Package personnages :



Package moteur :



Pourquoi notre projet est un bon / mauvais projet

Pourquoi notre projet est un bon projet :

Notre projet est un bon projet car nous avons respecté toutes les règles du jeu de base et avons même implémenté un type de bot censé représenter les personnes tricheuses ce qui ne nous paraissait pas hors sujet, bien que non demandé, cela constitue un plus nous le pensons non négligeable. Il est à noter que les bots tricheurs reprennent des concepts humains, cela ne veut pas dire qu'à la fin de la partie s'autoproclame vainqueur en trafiquant leur nombre de point car notamment au niveau du comptage les gens font attention mais au fil de la partie regardent les cartes de leur adversaires pour mieux pouvoir les contrer.
Comme un peu spoilé précédemment, il y a 3 types de bots :
- Les Bots Tricheur
- Les Bots FairPlay
- Les Bots Aléatoire

Remarque : les Bots Tricheur ne sont en réalité que la copie Bots Fairplay qui trichent, càd, ils sont codés avec la même structure et de la même façon pour les aspects du jeu qui ne sont pas améliorable par la tricherie (ex : si l'on peut construire 2 quartiers mais que l'on doit en choisir 1 pour un Tour, choisir de construire celui avec la plus grande valeur).

Pourquoi notre projet est un mauvais projet :

Notre projet peut etre considéré comme un mauvais projet car on avait pas bien distribué les taches entre nous au début et ca se voyait sur le code , en plus il y a quelques fonctions et méthodes qui pouvaient être plus optimisé et réduites .
Pour le découpage et la répartition des respensabilité on pense l'avoir bien fait mais malgré ca ils existent encore des redondances dans le code qui peuvent être évitées .

Rétrospective

Pour le prochain projet on envisage de bien répartire les taches afin d'éviter de perdre le temps à re-adapter le code chaque fois qu'on implémente de nouvelles fonctionnalités du jeu .
Il faut aussi bien créer les classes dès le début et réduire les méthodes car pour ce projet on l'a fait mais c'était à la fin et c'était dur à réaliser dc il faut prévoire que certaines méthodes vont étre utilisés plusieurs fois dans le code prochainement alors autant les bien créés .

Pourtant le travail d'équipe qui a été effectué lors de ce projet était très bien organisé et cela doit être conservé dans le prochain projet car ca permet de bien avancer en groupe et ne pas prendre du retard sur le projet .