Rapport de projet – Quoridor

Nathan Amorison et Igor Dujardin

BA1 Sciences Informatiques année 2019-2020

Table des matières :

1. Introduction:  
1.1 Défis et Organisation au sein du groupe

2. Choix Personnels :  
2.1 Prototype  
2.2 Création de Board  
2.3 Pawn et Possibilitie  
2.4 Walls  
2.6 Interface Graphique  
2.7 Drag & Drop  
2.8 Tests Unitaires

3 Points forts et points faibles  
3.1 Points faibles  
3.2 Points forts

4 Bugs et erreurs  
4.1 Bugs résolus  
4.2 Bugs non résolus  
5 Apports positifs et négatifs  
5.1 Apports positifs  
5.2 Apports négatifs  
6 Guide Utilisateur

1. Introduction:  
1.1 Défis et Organisation au sein du groupe

Arrivés tous les deux à la faculté des sciences en janvier suite à une réorientation, ce projet est vite apparu comme une occasion parfaite pour s’imprégner des méthodes de travail de la faculté et rattraper nos lacunes du quadrimestre manqué.

Dès la première séance, après avoir pris connaissance des consignes et des règles du jeu, nous avons établi un planning simple et précis des tâches à réaliser. Ainsi la première étape de notre projet serait la réalisation d’un prototype en Python, prototype que nous avons programmé ensemble au fur et à mesure que nous rattrapions le cours de programmation et algorithmique 1.

Une fois le cap du prototype passé, nous avons découpé le programme selon ses différentes fonctionnalités et nous les avons réparties entre nous. Ainsi Nathan s’occuperait de l’IA, des pions et des murs alors que je me chargerais de l’interface graphique, du plateau ainsi que des règles de déplacement des pions et de placement des murs.

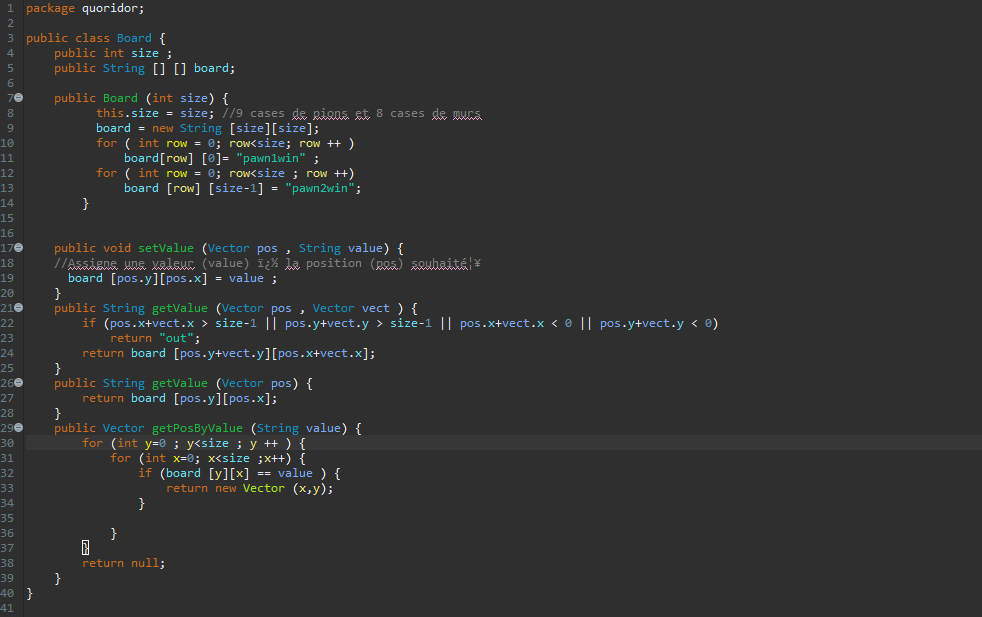
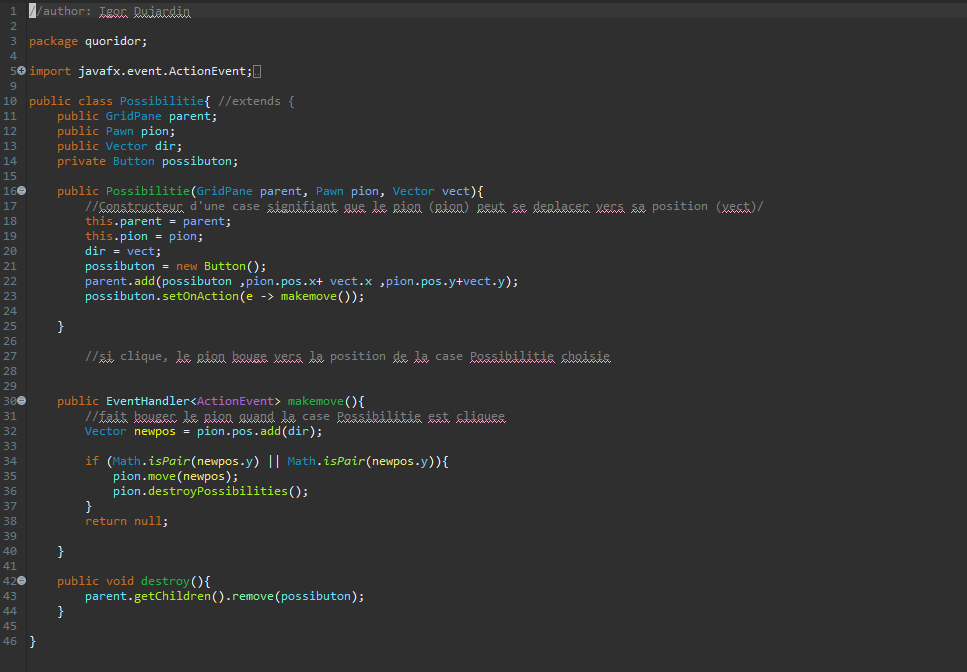
Nous jugions la communication au sein du groupe primordiale, ainsi nous travaillions souvent en tandem via Discord (confinement oblige) et prenions chacun le temps d’expliquer à l’autre chaque nouvelle fonctionnalité. Afin de simplifier la manipulation des fichiers d’un ordinateur à l’autre nous avons utilisé Git.  
  
2. Choix Personnels :  
2.1 Prototype

Nous avons décidé de réaliser un prototype pour deux raisons majeures.

Tout d’abbord, afin d’expérimenter et d’optimiser, en effet le prototype nous aura servi à tester les différentes fonctionnalités que nous souhaitions retrouver dans le produit final, pour chaque fonctionnalité nous avons essayé plusieurs méthodes, puis les avons comparées afin de choisir la solution optimale. Ainsi c’est sur python que nous avons établi la base de notre Pathfinder, entre autres.

Ensuite, afin de rattraper nos lacunes du premier quadrimestre manqué et d’intégrer les concepts de la programmation en Python. La réalisation du prototype nous a été très utile afin de comprendre le cours de programmation et algorithmique 1 et d’en appliquer les concepts.

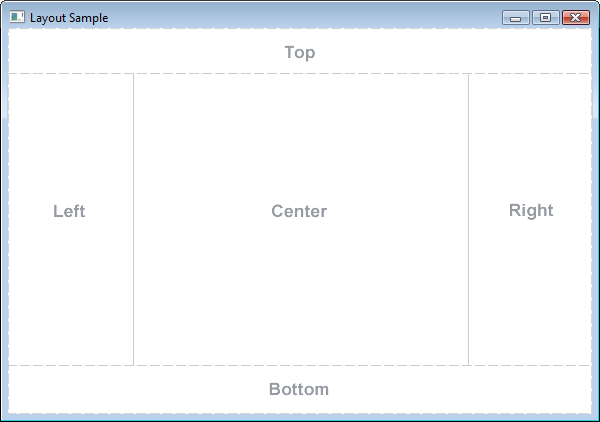
2.2 Création de Board

Nous avons pensé notre Board comme un tableau de strings, avec pour chaque case une value correspondant au numéro de colonne et de ligne. Les deux extrémités du Board on la mention “pawnXwin” , comme étant les conditions de victoire.  
2.3 Pawn et Possibilitie  


La classe Possibilitie a pour but de nous montrer les différents mouvements que le pion peut effectuer, c’est en effet une fonctionnalité essentielle, et c’est la seule manière que nous avons trouvé pour montrer au mieux les mouvements possibles.

2.4 Walls

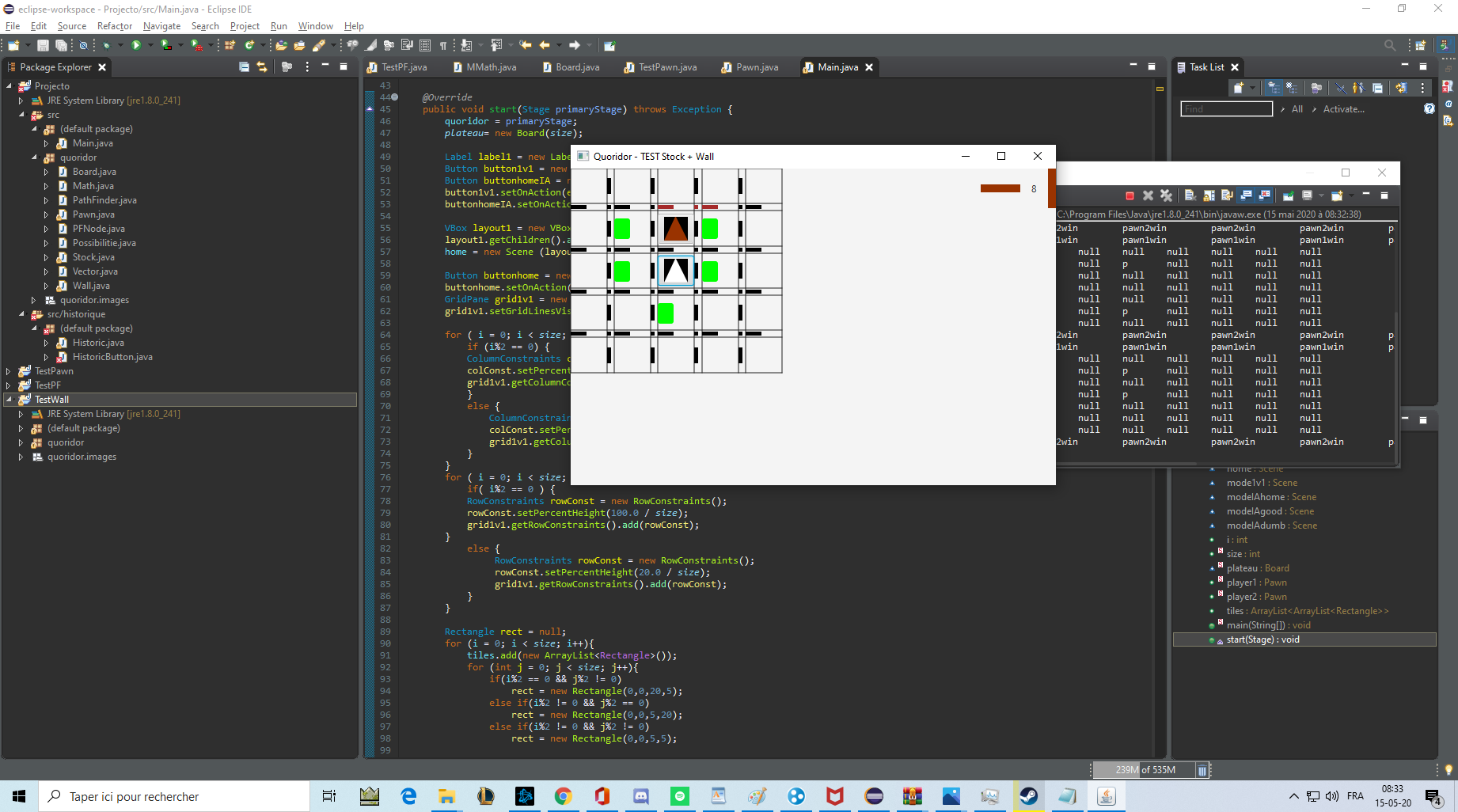
Les murs se placent facilement avec le Drag&Drop  
2.6 Interface Graphique

Nous avons utilisé comme demandé JavaFX pour l’interface graphique, la majeure partie de l’interface se trouve dans la méthode Main, elle consiste en une série de Scenes représentant chacune une partie différente du jeu (menu principal, terrain de jeu, sélection de l’IA,etc ). Nous avons choisi comme layout le BorderPane, car il correspond parfaitement au type d’interface souhaitée.  
   
Avec ce type de layout, nous pouvons disposer de manière ergonomique les différents éléments visuels (plateau, stock de murs, boutons divers, etc).

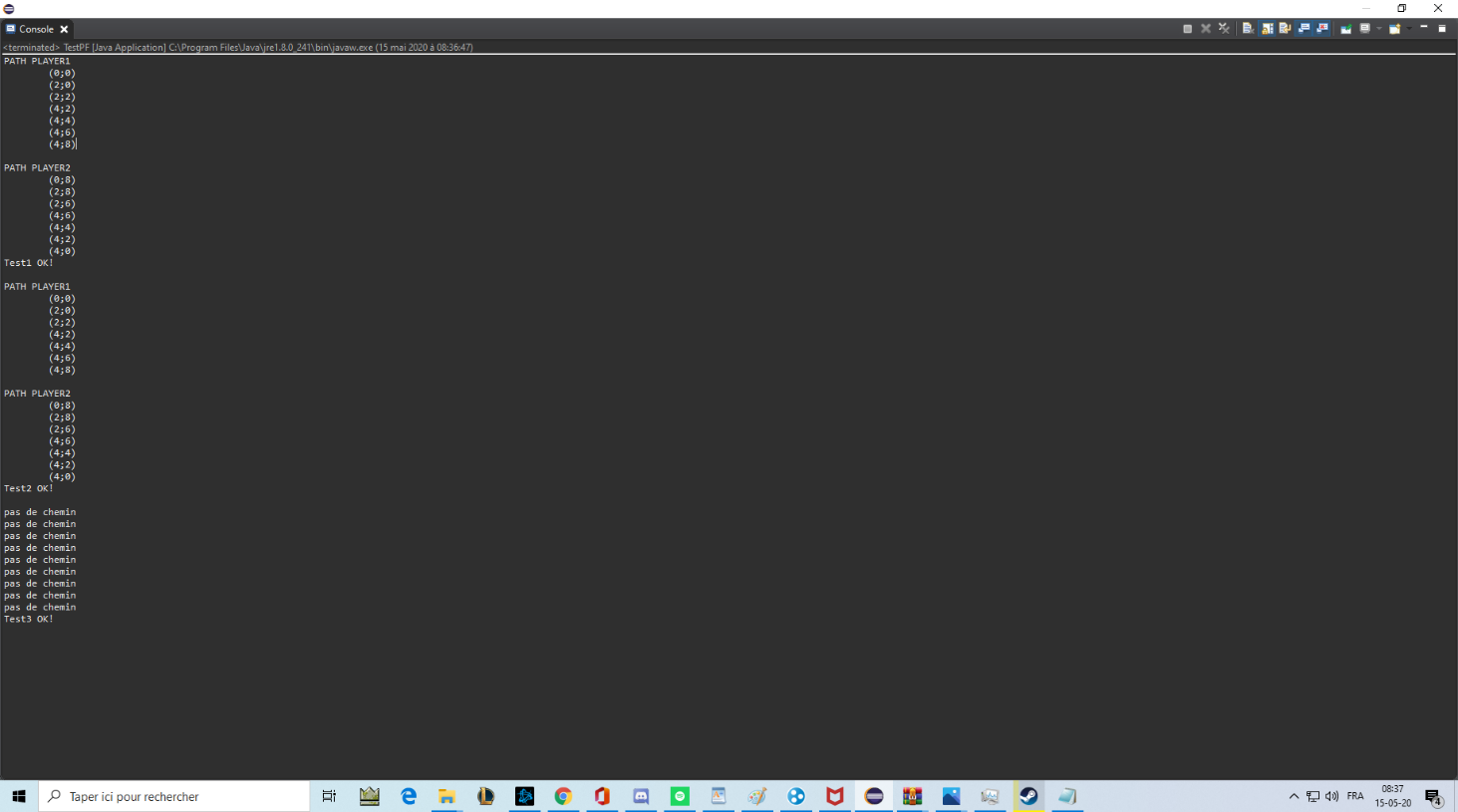
Pour adapter notre classe Board à l’interface graphique, le type gridpane était une évidence, nous avons pu aisément créer une grille aux dimensions identiques de la vraie.

Par manque de temps et de talent artistique, nous avons choisi un design sobre et épuré pour les pions et les murs.  
2.7 Drag & Drop

Nous avons choisi d’implémenter une mécanique de drag & drop pour le placement des murs. En effet, c’est une fonctionnalité agréable pour l’utilisateur, et assez simple à coder. Notre prototype avait une forme primitive de drag and drop (on ne voyait le mur bouger lors du déplacement de la souris) mais nous nous sommes documentés sur les Event JavaFX afin d’implémenter cette fonctionnalité en Java.  
2.8 Tests unitaires



Nous avons réalisé une série de tests unitaires afin de tester nos différentes fonctionnalités. Ici nous testons la règle du face à face, l’affichage des possibilités et le placement des murs. Ceci était notre dernière étape avant l’assemblage final de notre jeu définitif.



Ici nous testons notre Pathfinder, le cœur de notre intelligence artificielle, le test nous permet de voir les déplacements de chaque pion dans une série de petits parcours prédéfinis.

Dans le premier parcours, les deux pions sont lâchés sur le board sans aucun mur et trouvent naturellement leur chemin.

Dans le second parcours, on place un mur en face des pions, ces derniers vont le contourner et continuer leur chemin.

Dans le dernier parcours, on érige un énorme mur tout le long du board, empêchant les pions d’arriver à leur objectif, le PathFinder nous informe qu’il n’y a pas de chemin possible, ce qui est tout à fait normal. On peut donc implémenter cette mécanique afin de bloquer les coup illégaux (par exemple : empêcher un joueur d’avoir un chemin pour gagner).

3 Points forts et points faibles  
3.1 Points faibles

Notre code est loin d’être parfait, nous le savons, ce projet fut un véritable défi pour nous deux, fraîchement arrivés à la faculté. Ainsi nous pensons que les défauts majeurs de notre projet sont les suivants :

-Un design assez basique

-Un code un peu brouillon pour l’interface graphique  
3.2 Points forts

Nous sommes cependant également très fiers de notre code, car nous jugeons qu’il brille par les points suivants :

-Facile à utiliser

-Facile à comprendre

4 Bugs et erreurs  
4.1 Bugs résolus  
La grande majorité de nos bugs sont survenus au moment de passer de l’interface statistique à l’interface graphique:

-Le drag & drop sélectionnait les deux murs en même temps et une fois relâché, ne posait aucun mur sur le tableau.

-A de très nombreuses reprises, les boutons de Possibilite refusaient de s’afficher sur le Board

-Le GridPane de l’interface graphique n’avait pas le bon nombre de colonnes

-Le placement des murs a cessé de fonctionner au moment de passer du test unitaire au jeu définitif, c’est le bug le plus coriace que nous ayons rencontré, il nous aura pris des jours à résoudre.  
4.2 Bugs non résolus

-Le Drag & Drop peut se montrer capricieux avec certains murs, nous avons tenté de rectifier ce bug mais n’avons pas réussi à en trouver les causes exactes.

-Nous sommes actuellement incapables de faire fonctionner nos IA , nous n’avons pas réussi à correctement implémenter le PathFinder au sein de l’IA

5 Apports positifs et négatifs  
5.1 Apports positifs  
Ce projet nous a apporté beaucoup, nous avons appris ensemble à gérer un projet en tandem, à communiquer, partager les fichiers via Git, etc.

Il va également de soi que ce projet nous a aidé dans la compréhension des cours de programmation et algorithmique 1 et 2, la réalisation du prototype nous a tout particulièrement aidé dans l’apprentissage de Python.  
5.2 Apports négatifs  
Nous avons pu expérimenter la forme frustrante de certains bugs extrêmement récurrents, même s’il est très satisfaisant de détruire ces mêmes bugs, le temps perdu à essayer de les corriger aurait pu nous permettre d’implémenter de nouvelles fonctionnalités.   
6 Guide Utilisateur

Une fois le jeu lancé, il suffit de sélectionner dans le menu principal le mode de jeu souhaité et de jouer la partie. L’utilisateur doit faire attention à bien