COMPTE RENDU DE PROJET TURORE

FELIUS Robin VANDERVALLE Basile

INFO 3A

TABLE DES MATIERES

I] Sujet 2
II] Implémentation 2
II. 1 – Classes 2
II. 2 – Jeu 4
II. 3 – IA des fantomes 6
III] Extensions 6
III. 1 – Menu 6
III. 2 – Choix des parametres du jeu 8
La Carte 8
La difficulté 8

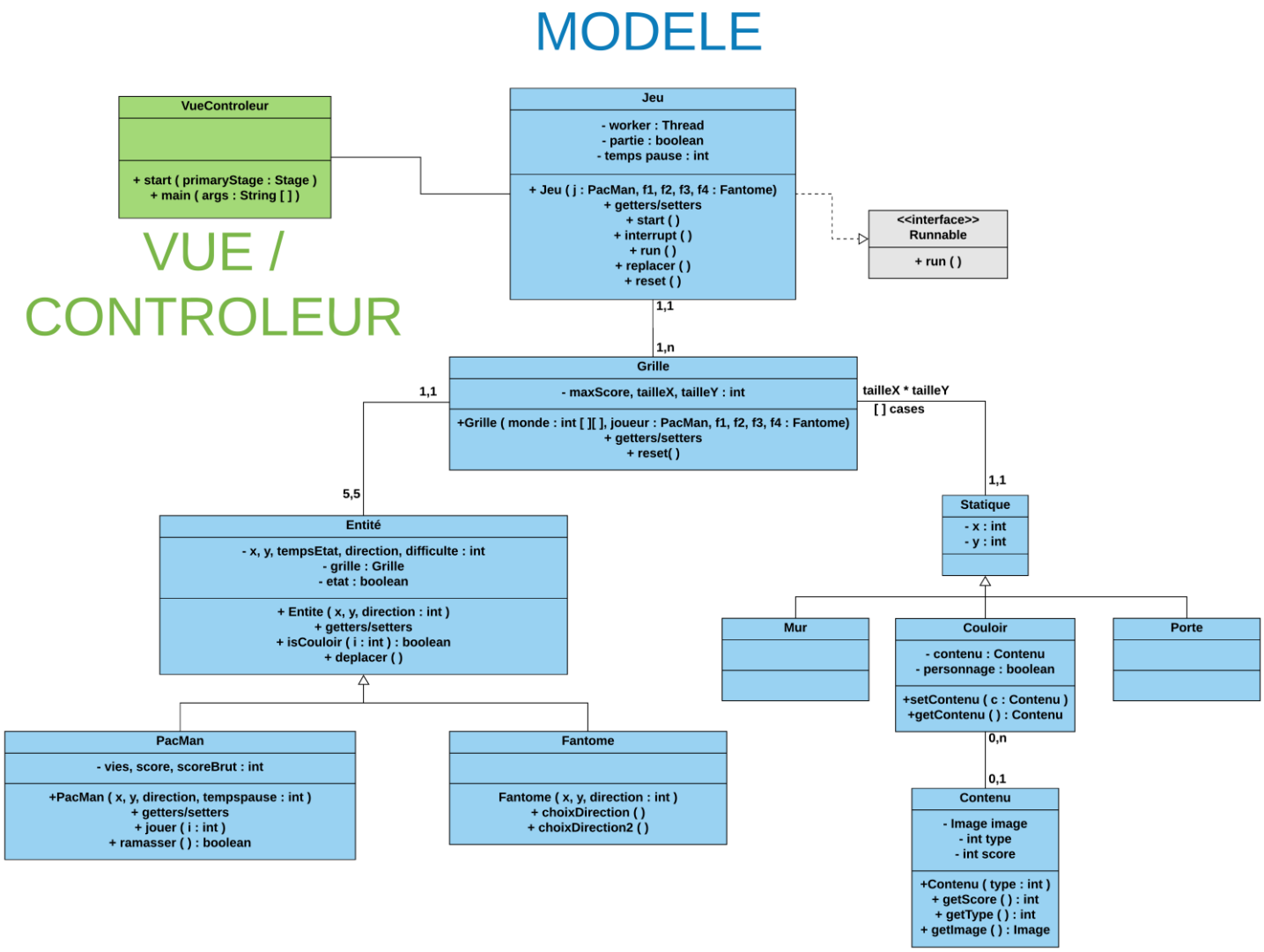
I] SUJET

Ce projet encadré avait pour but de créer une application graphique mettant en place le jeu arcade Pac-Man. Cette application, développée sous Java grâce à JavaFx, permet au joueur d'incarner le héros Pac Man, pour le faire se déplacer à l'aides des touches du clavier {z (haut), q(gauche), s(Bas), d(droite)} sur une carte ressemblant à un labyrinthe, avec des objets à ramasser (pacgum, super pacgum, sablier) et des ennemis (fantômes) à éviter. Nous nous sommes efforcés de développer notre application JavaFx en respectant le pattern Modèle-Vue-Contrôleur et en implémentant le jeu de manière la plus générique possible, pour permettre l'ajout rapide de fonctionnalités et le débogage.

II] IMPLEMENTATION

II. 1 – CLASSES

Nous avons, au début de notre projet, et avant l'implémentation, établi un diagramme de classes de la méthode UML pour nous aider lors de l'implémentation. Nous l'avons complété tout au long du projet dès que c'était nécessaire. Ce diagramme est présenté sur la figure 1, en page suivante.

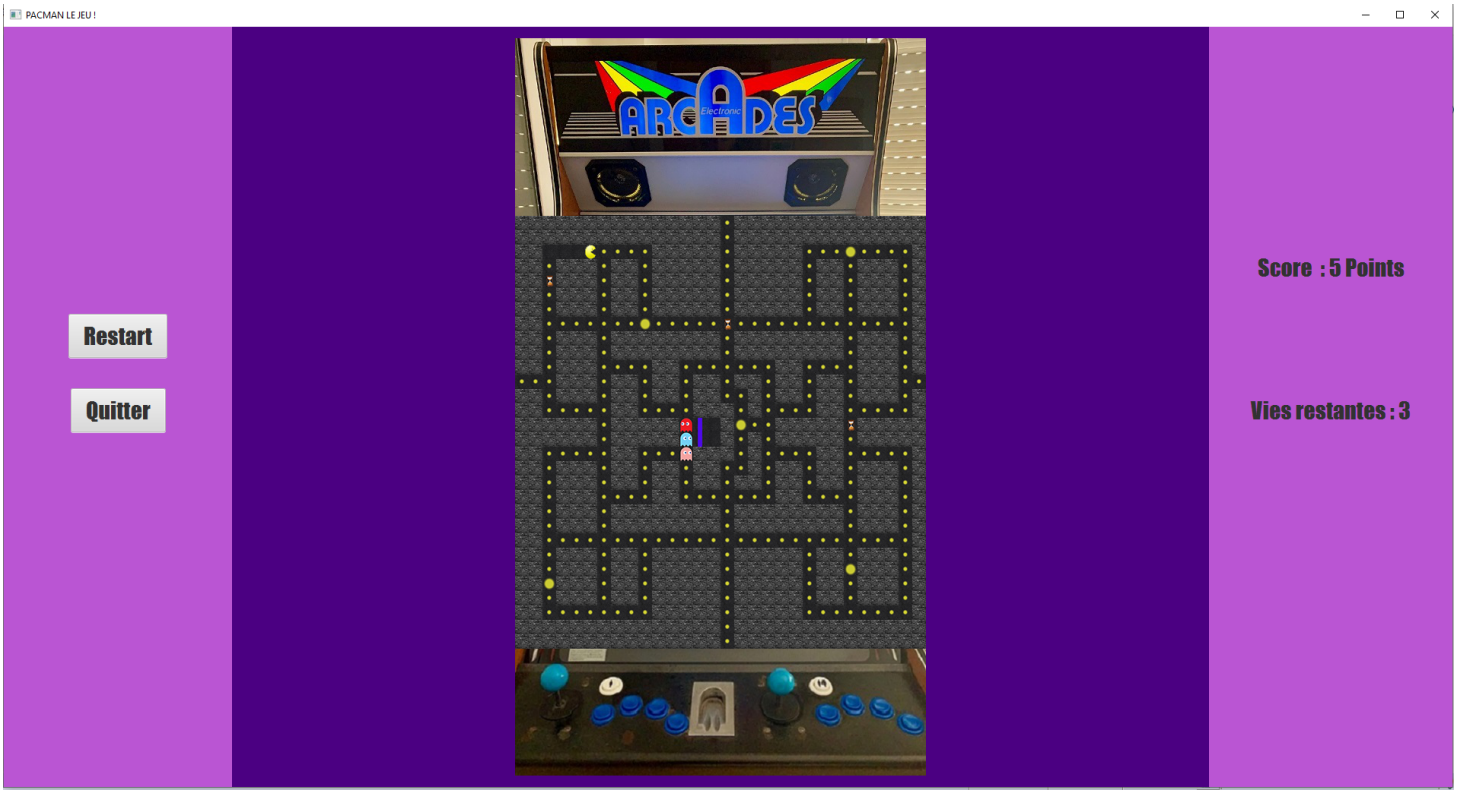


| Figure 1 : Diagramme de Classes

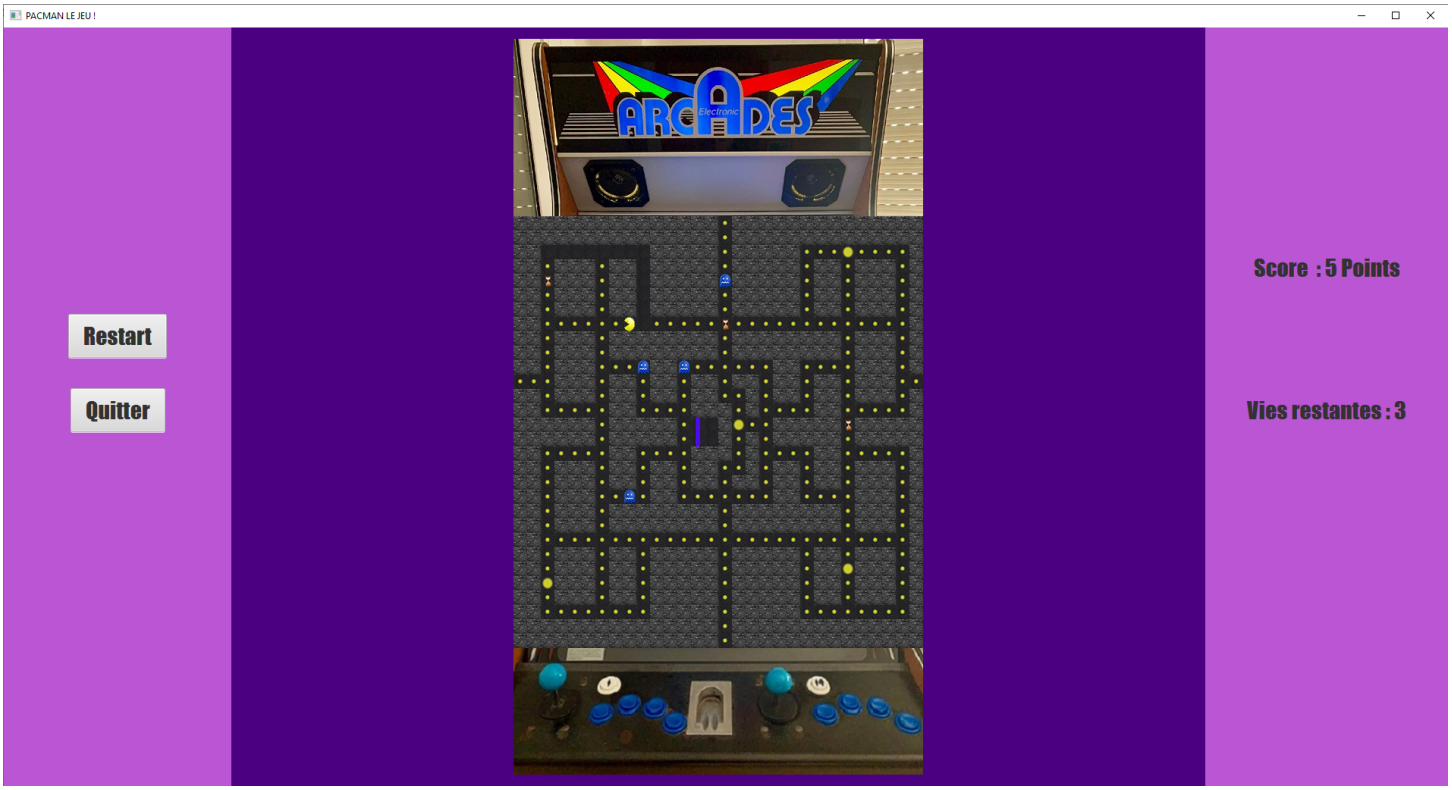
Nous avons une classe VueControleur dans laquelle les méthodes de la vue et du contrôleur sont séparées (respectant donc l'énoncé), ainsi qu'un package Modele, dans lequel toutes les classes inhérentes au modèle sont stockées. La classe jeu qui implémente l'interface Runnable, permet le déroulement du jeu sur un nouveau Thread. Le Jeu est composé d'un joueur : PacMan, quatre Fantômes, et une ou plusieurs Grilles de jeu, composées d'éléments statiques, comportant ou non un Contenu.

II. 2 – JEU

Notre jeu consiste en une boucle tant que, qui s'arrête dès que la partie se termine (clic sur le bouton Quitter par le joueur, si le joueur a récupéré tous les objets récupérables ou s'il n'a plus de vie et qu'il est touché par un fantôme. Chaque tour de jeu consiste en 4 déplacements du joueur et de 3 ou 4 déplacements des fantômes selon si le joueur est sous l'effet d'une super pacgum ou non. Les déplacements gérés par la classe Entite dont héritent PacMan et Fantome. Ils récupèrent la direction actuelle de l'entité et, si le déplacement est possible (la case cible est un couloir) et si l'entité est un fantôme, il choisit une nouvelle direction, en fonction de son IA et de la difficulté choisie (cette extension sera explicitée plus loin dans ce rapport). Pour améliorer l'expérience du joueur, un contour de la grille de jeu, représentant une borne d'arcade vintage, ainsi qu'un environnement coloré avec un affichage du score et du nombre de vies restantes a été ajouté. Sont également présents des boutons permettant de recommencer la partie ou de quitter la partie et de retourner au menu. Une musique d'ambiance est présente tout au long de la partie (différente pour les trois niveaux de difficulté) et des sons sont joués si Pac Man mange un objet, un fantôme, ou s'il est touché par un fantôme. Plusieurs captures d'écran du jeu sont présentées ci-dessous :



| Figure 3 : Début de partie



| Figure 2 : Partie en cours (PacMan a mangé une super Pacgum)

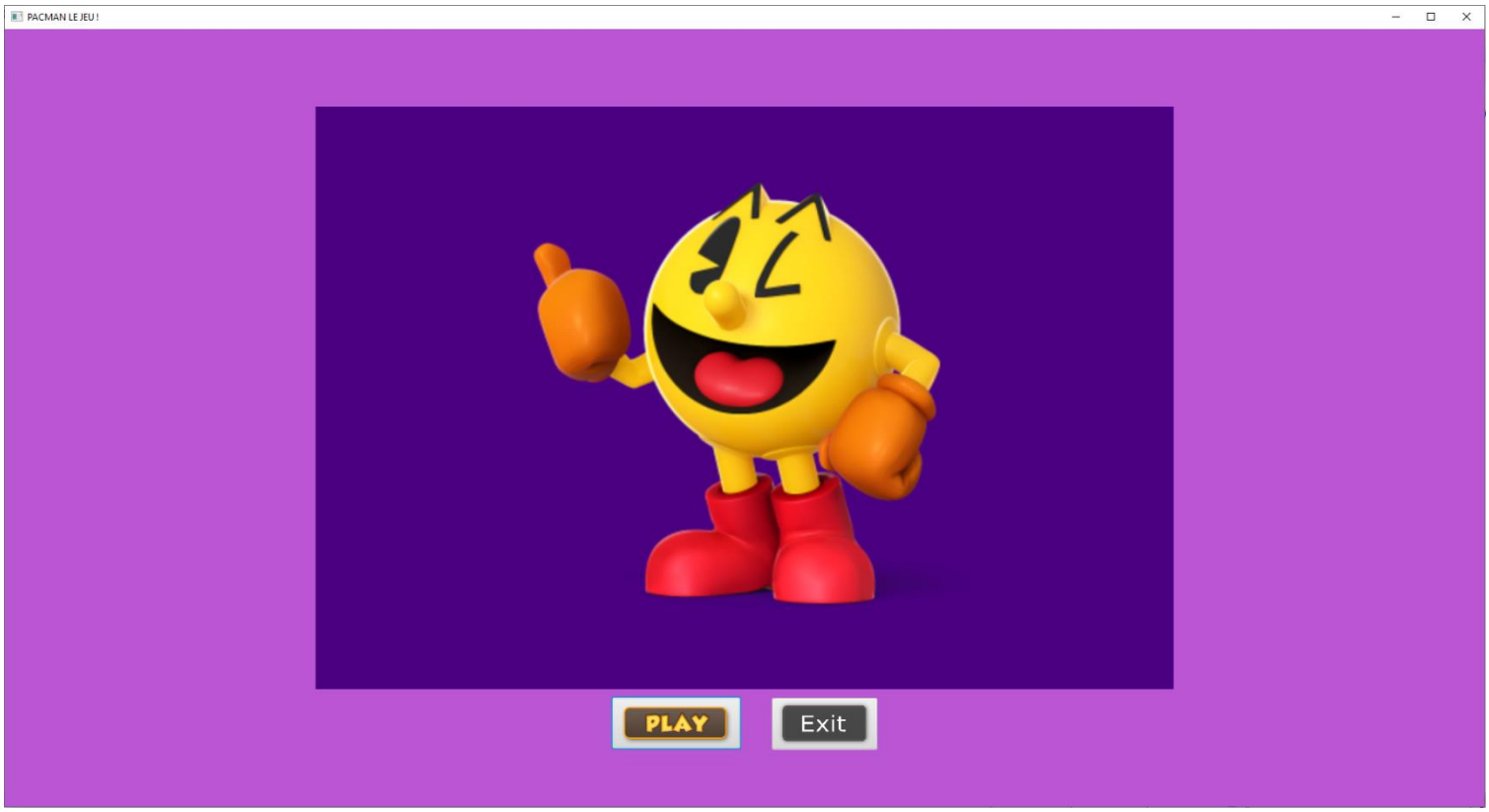
II. 3 – IA DES FANTOMES

Nous avons implémenté l'Intelligence Artificielle des fantômes de manière à ce qu'ils tentent de se rapprocher du joueur pour le mettre en difficulté. Cela donne lieu à une méthode choixDirection( ) qui choisit au hasard parmi les direction disponibles (où il n'y a pas de mur) ainsi qu'une autre méthode choixDirection2( ) qui choisit une direction disponible et qui permet de minimiser la distance avec le joueur. La méthode de déplacement des fantômes à proprement parler, alterne entre ces deux méthodes pour que les déplacements des fantômes ne soient pas toujours optimaux (sinon le joueur n'aurait aucune chance de gagner). La difficulté (extension explicitée par la suite) permet de décider de la proportion de choix optimaux et de choix aléatoires.

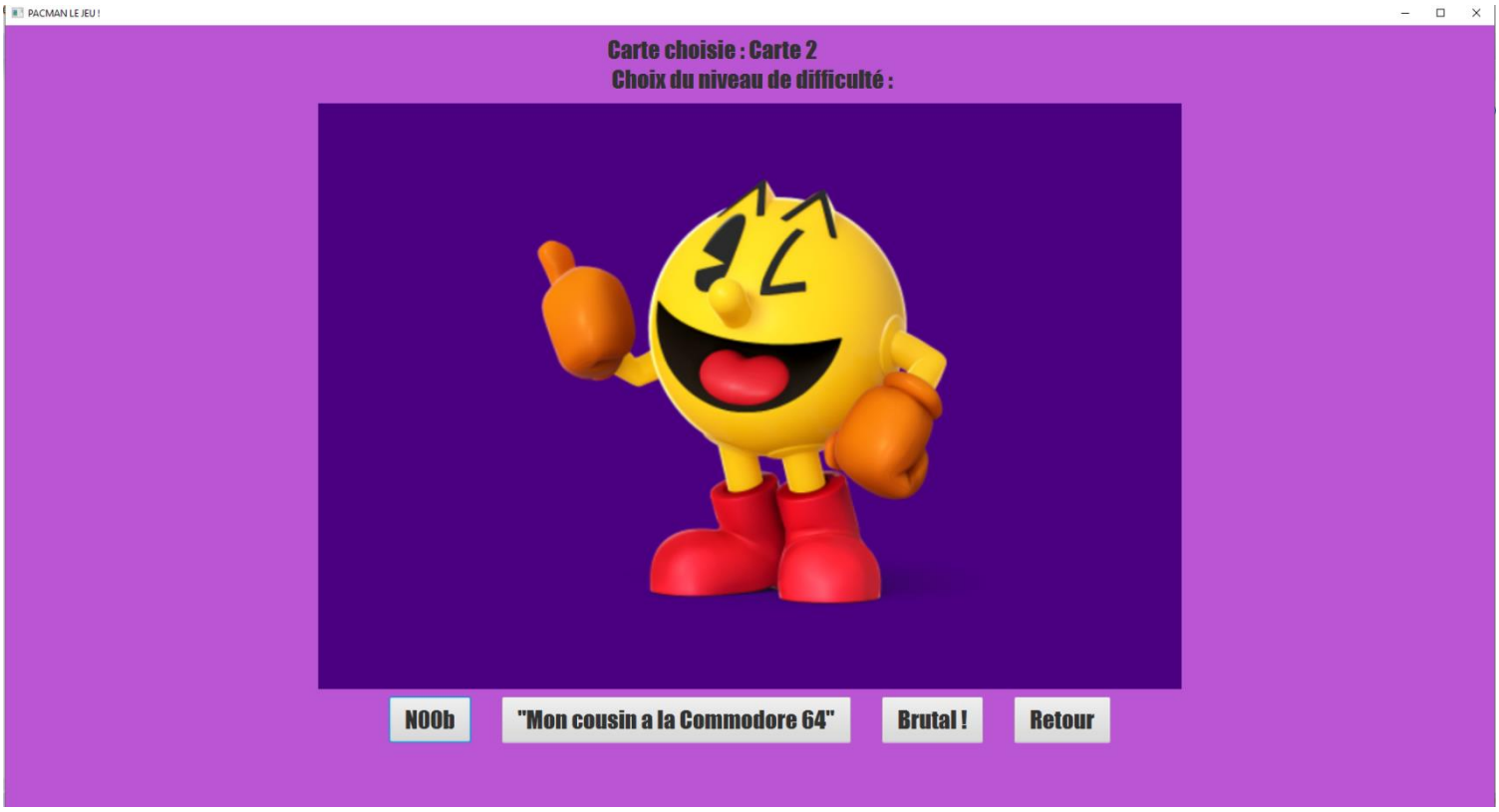
III] EXTENSIONS

III. 1 – MENU

L'une de nos extensions est l'ajout d'un menu permettant au joueur de choisir une carte (pour l'instant seules deux cartes ont été créées mais l'ajout de carte est assez simple et devrait être facilement implémentable par la suite …), ainsi que la difficulté de jeu, une autre de nos extensions, développée plus tard dans ce rapport. Des boutons sont pour cela utilisés, stockés dans des HBox, eux-mêmes disposés dans un BorderPane, qui constitue notre Scene de menu. Des exemples de captures d'écran sont présentés aux figures 2 et 3 en page suivante.



| Figure 5 : Menu (Accueil)



| Figure 4 : Menu (Choix de la difficulté)

III. 2 – CHOIX DES PARAMETRES DU JEU

LA CARTE

Nous avons donc réalisé notre vision de ce jeu, avec deux cartes dont la topologie et les objets collectables par le joueur sont différents. Le choix de la carte s'effectue depuis le menu, avant de commencer la partie.

LA DIFFICULTE

Un système de difficulté du jeu a également été ajouté, chacune des trois difficultés proposées modifiant la musique d'ambiance, la vitesse du jeu ainsi que l'IA des fantômes. En effet, plus la difficulté est élevée, plus le ratio de choix de déplacement optimaux par rapport aux déplacements aléatoires, est important.