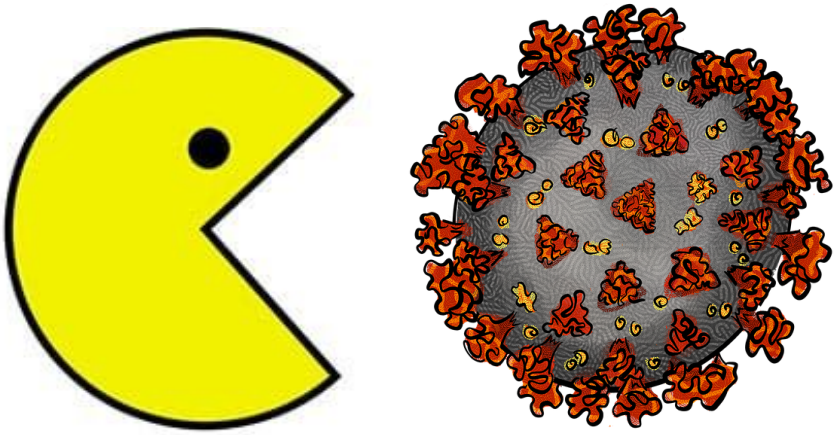
# Rapport de projet POO :

# Pac-Man édition coronavirus



*1**

Julien GIRAUD, Mérien GHALI
L3 Informatique - 2019 / 2020
Université Lyon 1

*1 : Images non contractuelle*

# Présentation du projet

## Préambule

Nous avons décidé de revisiter le Pac-Man tout en restant très fidèle au jeu original. Les changements majeurs sont au niveau de la map, des niveaux et des icônes. Les éléments qui n'ont pas changés sont le menu, les contrôles clavier, les sons (modulo quelques notes), les scores, la gestion des vies et les éléments far du jeu : les boules, les superboules, les fruits bonus, les fantômes et le passage magique sur les côtés de la map.

Vous noterez que le fantôme orange a été atteint par le coronavirus, d'où sa couleur verte. Il a probablement contaminé ses amis, raison de plus pour les craindre !


Figure 1.

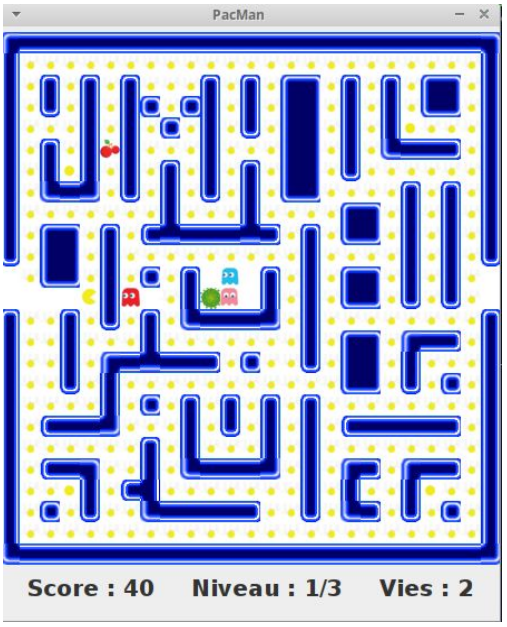



Figure 2.

En lançant le jeu, un menu apparaît contenant deux boutons, en cliquant sur *Lancer le jeu*, une nouvelle fenêtre apparaît (voir figure 2 ) avec un son d'introduction, cette fenêtre affiche la map, le score, le niveau ainsi que le nombre de vies.

En mangeant une superBoule, les icônes des fantômes ainsi que le Convid-19 changent. Les fantômes sont alors vulnérables, plus on en mange plus on gagne de points. Un fantôme mangé se retrouve dans le spawn avec une icône spéciale (voir figure 3).

Une fois un niveau gagné, on passe au suivant (voir figure 4). Le niveau affiché change, ainsi que le fruit bonus et l'IA des fantômes. En conséquence, la difficulté augmente.
À la fin d'une partie on revient au menu. Il est alors possible de consulter le tableau des scores, une nouvelle fenêtre apparaît (voir figure 5). On peut ensuite revenir au menu avec un bouton, puis rejouer.

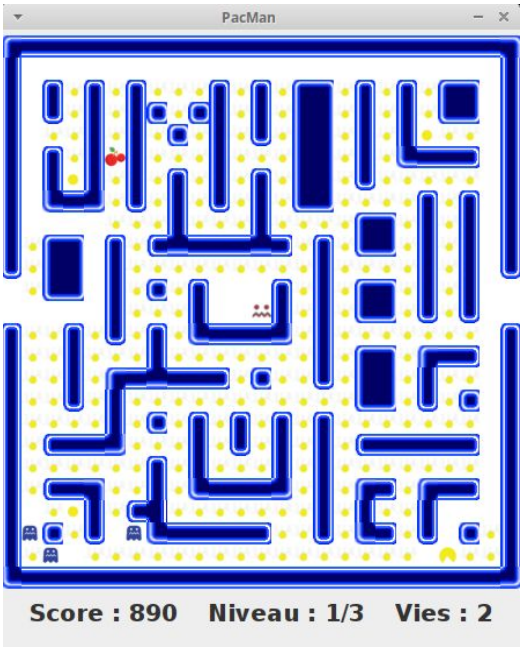



Figure 3.

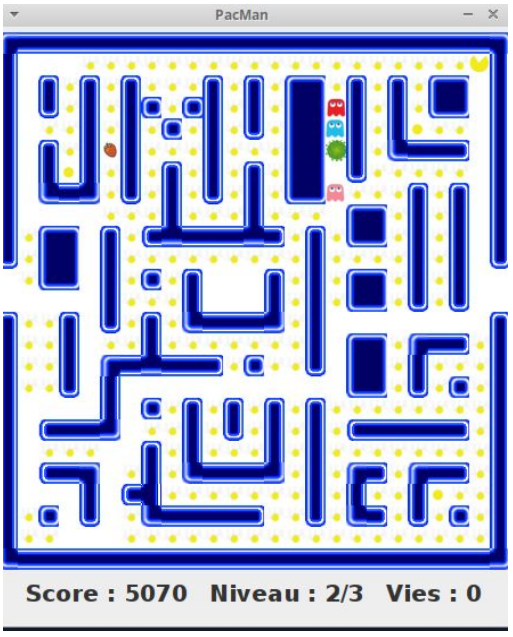



Figure 4.

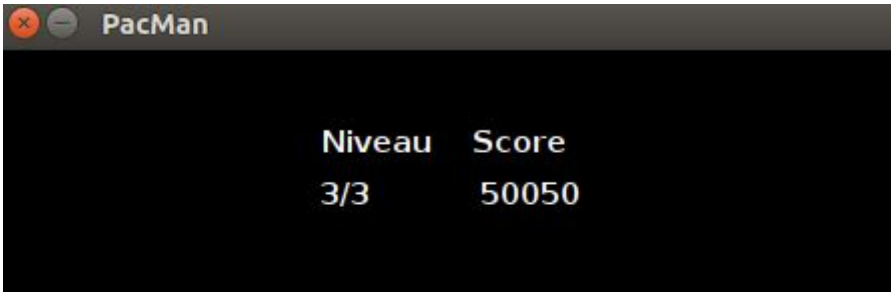



Figure 5.

Le record actuel est de 50 050 pour environ 25 minutes de jeu, la compétition est ouverte !

## Fonctionnalités du jeu

- Un menu qui permet de lancer et relancer les parties ou consulter le tableau des scores
- Une map agréable à jouer avec nos initiales en haut à gauche, des obstacles divers, une zone réservée aux fantômes et un passage magique sur les côtés comme dans le Pac-Man original
- Une barre d'informations avec le score en temps réel, le niveau et le nombre de vie restantes
- Les mêmes contrôles que dans le jeu original : on ne peut pas stopper Pac-Man d'un coup en tournant dans un mur, mais si on lui demande de tourner dans une direction il continuera d'avancer jusqu'à ce qu'il puisse tourner
- Le contrôle des fantômes est similaire au jeu original : ils ne peuvent pas faire demi-tour et ils doivent attendre une intersection pour changer de direction
- Il y a 3 niveaux qui se caractérisent par le niveau d'intelligence des fantômes : l'IA qui gère les déplacements des fantômes est d'abord aléatoire, puis "stupide" avec heuristique, et enfin intelligente avec heuristique (A* pour être précis)

## Spécificités du code

- Gestion pratique et POO / MVC des icônes : lorsqu'un élément possède plusieurs icônes (selon sa direction, sa vulnérabilité…) la vue utilise la fonction getState sur l'entité (getType pour les murs), un numéro est alors retourné pour indiquer l'icône à utiliser
- Optimisation de l'affichage : la vue utilise une sauvegarde de la carte pour détecter les différences de grille entre les tours afin d'actualiser que les cases qui changent
- Une gestion des entités complètement revue afin de gérer les problèmes de superposition des entités
- Un système générique qui utilise l'interface "Mangeable" pour gérer les entités que Pac-Man peut manger
- Ajout des notions de vie, de point de spawn, de manger, de score, de mort, de respawn, de niveau et de partie
- Gestion du bonus comme dans le jeu original
- Gestion des fantômes qui deviennent vulnérables quand Pac-Man mange une superboule
- Gestion POO de l'IA des fantômes : l'algorithme de A* est complètement indépendant de la grille et de nos classes, il prend en entrée une liste de points ce qui le rend très générique

| Estimation du temps passé à développer la fonctionnalité | | | |
|---|---|---|---|
| Très peu | Peu | Pas mal | Énormément |

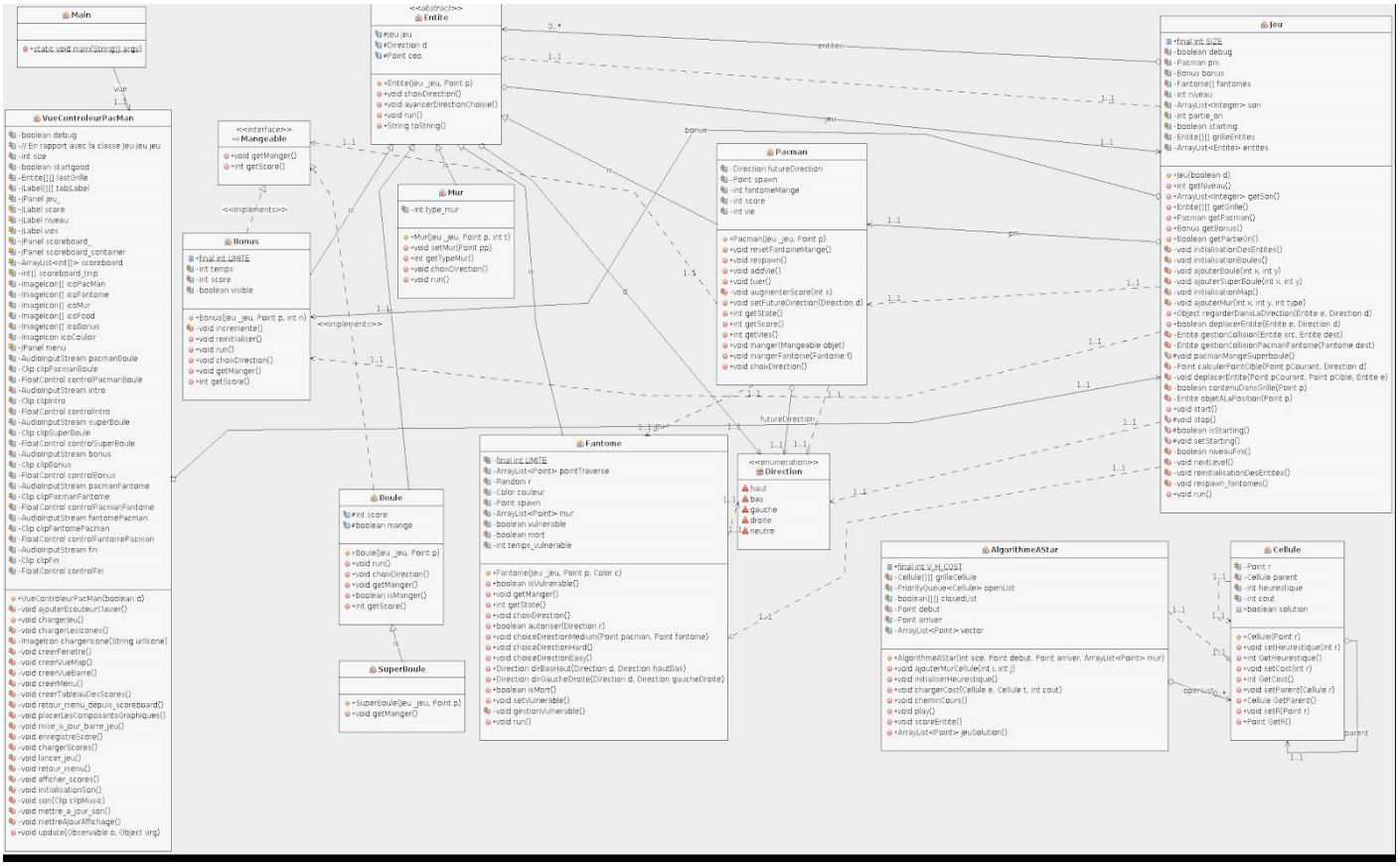
Au sein du binôme la répartition du développement était : Julien Giraud 55%, Mériem Ghali 45%.

# Analyse

## Utilisation de l'UML

Nous avons commencé par faire un diagramme de classe pour mieux visualiser le code de départ, ainsi mieux comprendre le fonctionnement et de quoi le projet se compose. Ensuite nous avons modifié le diagramme au fur et à mesure de nos modification du projet, jusqu'à obtenir le diagramme de classe final. Cette méthode nous a aidé à avancer plus rapidement et plus efficacement sur le projet.

## Diagramme de classes du projet

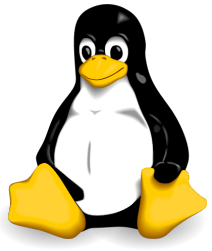



[Image en taille réelle ici]

# Bilan

## Outils utilisés





Le projet a été développé sous Linux en utilisant Java 8 et Netbeans comme IDE. Pour une meilleure gestion du partage du projet en binôme nous avons utilisé la forge de l'université (GitLab) et pour éviter les conflits nous avons travaillé sur des branches avec GIT.

## Difficultés rencontrées

Avec la période du confinement nous avons rencontrées quelques problèmes de communication comme : l'explication des parties de code, l'expression les idées… mais nous avons réussi à résoudre ces problème en utilisant Discord pour communiquer ainsi que faire des partages d'écran, afin de mieux exprimer nos idées et expliquer les parties de code par exemple.

## Conclusion

Nous avons pu faire le tour de nombreux éléments en rapport avec ce module de Java / UML :

- Une expérience complète sur le langage Java avec toutes les notions vues en cours
    - En back-end : typage, classes normales, classes abstraites, interfaces, héritage, polymorphisme, événements, multi-threading
    - En front-end : un bon tour de Java Swing et de ses composants, notamment JPanel, JLabel, JComponent, GridLayout, BorderLayout, FlowLayout, notions de rafraîchissement graphique (updateUI, repaint)
- Une idée précise du fonctionnement d'un MVC en Java
- Une expérience concrète d'utilisation de l'UML pour visualiser un projet et aller plus vite dans l'écriture du code des classes