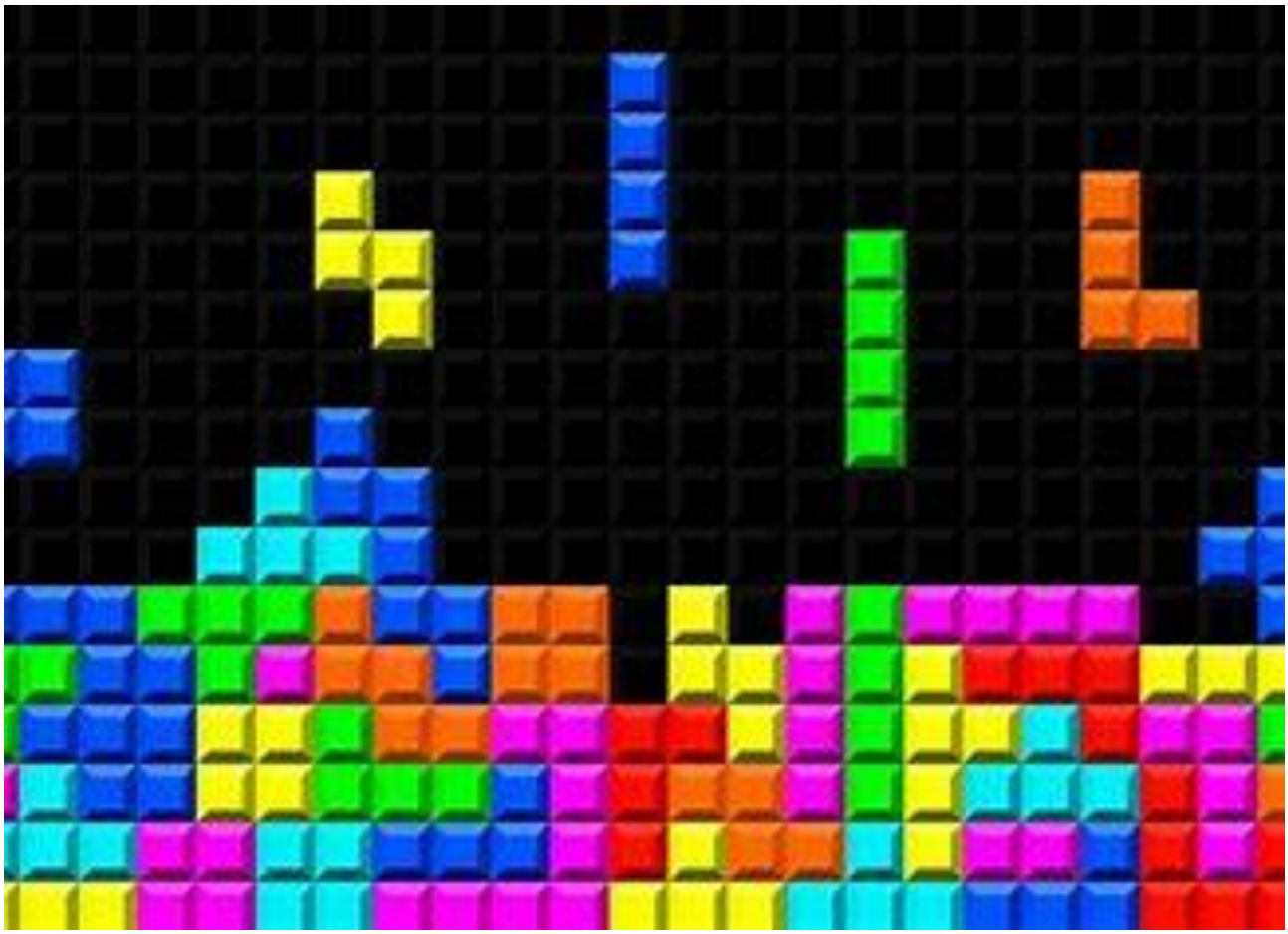
# Projet de Développement Logiciel

## Livrable des extensions du jeu Falling Blox

Année scolaire 2022-20203



Travail effectué par : NDE NOUMI Steve Darius

# Sommaire

# INTRODUCTION

Le Projet de Développement Logiciel Livrable des extensions du jeu Falling Blox est un travail réalisé par NDE NOUMI Steve Darius au cours de l'année scolaire 2022-2023. Ce projet consiste en l'ajout de neuf extensions au jeu Falling Blox, comprenant l'ajout de pièces manquantes, la détection et suppression de lignes complètes, la gestion du score, la possibilité d'échanger la pièce actuelle avec la pièce suivante, la modification de la vitesse de la gravité, l'utilisation des boutons physiques pour gérer le mouvement et la rotation des pièces, l'utilisation du clavier pour la rotation et le mouvement des pièces, ainsi que la gestion de la fin de partie. Dans ce rapport, nous allons décrire les extensions réalisées, expliquer leur fonctionnement, présenter les diagrammes UML des classes ajoutées et modifiées, et fournir les sources du projet.

## 1- Liste des extensions implémentées

Nous avons amélioré la version de base de notre jeu en ajoutant neuf extensions que nous avons réparties en trois classes, comprenant deux classes existantes et deux nouvelles classes. Les modifications apportées sont les suivantes :

- Ajout de pièces manquantes (TPiece, LPiece, JPiece, ZPiece, SPiece).
- Détection des lignes complètes dans le tas pour les supprimer et faire descendre les pièces plus haut afin de réduire la hauteur du tas.
- Ajout d'un système de score (par exemple, 10 points pour chaque ligne complète dans le tas).
- Possibilité pour le joueur d'échanger la pièce actuelle avec la pièce suivante.
- Modification de la vitesse de la gravité (augmentation de la vitesse après avoir complété un certain nombre de lignes, par exemple 10, 15, 20...).
- Utilisation des boutons physiques pour gérer le mouvement et la rotation des pièces.
- Utilisation du clavier pour la rotation et le mouvement des pièces.
- Gestion de la fin de partie (au lieu de la levée actuelle d'une ArrayIndexOutOfBoundsException).

## 2- Explication des extenssions

- Ajout de pièces manquantes :

Les pièces manquantes (TPiece, LPiece, JPiece, ZPiece, SPiece) ont été ajoutées au jeu, mais le code pour leur création et leur rotation n'est pas inclus dans le code fourni.

- Détection et suppression de lignes complètes :

Le code implémente la fonctionnalité de suppression de lignes complètes dans le puits de jeu de Tetris, ainsi que la mise à jour du score en fonction du nombre de lignes supprimées. La méthode **cleanTas**() parcourt le puits de bas en haut, vérifie si chaque ligne est pleine en regardant si chaque cellule est occupée par une pièce, et supprime les lignes pleines en faisant descendre les pièces au-dessus de la ligne supprimée. La méthode **supprimeLigne**(i) décale chaque

ligne au-dessus de la ligne supprimée d'une rangée vers le bas en copiant les pièces de la rangée supérieure dans la rangée actuelle. Enfin, elle supprime toutes les pièces de la première ligne du puits en les remplaçant par null. La méthode **miseAJourScore**(**lignescore**) met à jour le score en fonction du nombre de lignes supprimées, en utilisant un switch-case pour calculer le nombre de points à ajouter.

- Gestion du score :

La mise à jour du score est gérée dans la fonction **miseAJourScore**(), qui prend en entrée le nombre de lignes supprimées lors de la dernière opération de nettoyage de tas. Le score est mis à jour en fonction du nombre de lignes supprimées, avec un nombre de points attribué pour chaque ligne supprimée.

- Possibilité pour le joueur d'échanger la pièce actuelle avec la pièce suivante :

Cette fonctionnalité est implémentée grâce à la méthode **echangePiece**() de la classe d'extensions. Elle permet au joueur d'échanger la pièce actuelle avec la pièce suivante tout en conservant les anciennes coordonnées de la pièce.

- Modification de la vitesse de la gravité :

La vitesse de la gravité est modifiée dans la classe gravite en fonction du nombre de lignes supprimées. La périodicité est réduite chaque fois que le joueur gagne 50 points supplémentaires.

- Utilisation du clavier pour la rotation et le mouvement des pièces :

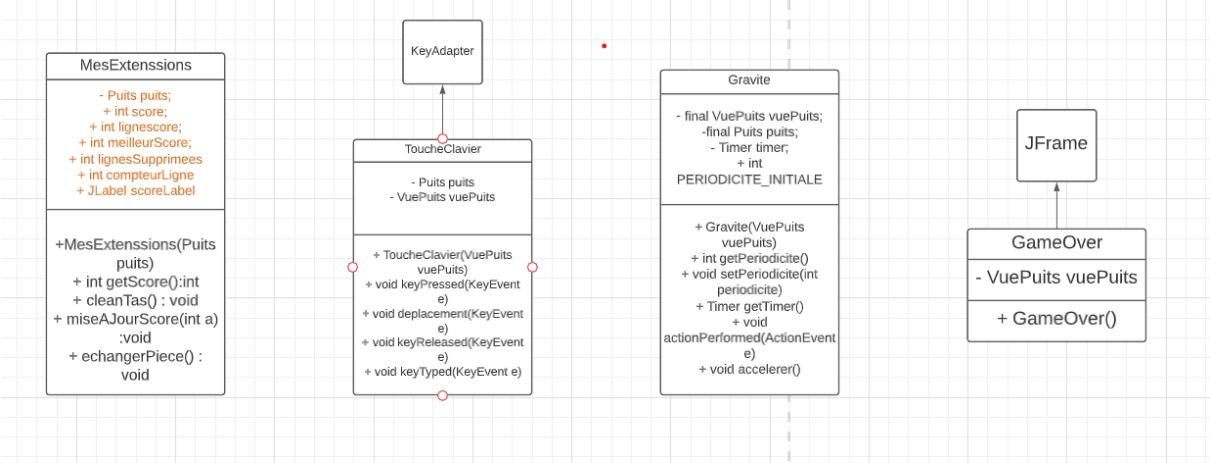
Les fonctionnalités de rotation et de mouvement des pièces sont mises à jour dans la classe touches clavier. Elle utilise un **keylistener** et est une sous-classe de **keyAdapter.**

- Gestion de la fin de partie :

La gestion de la fin de partie est implémentée dans la classe puit. Lorsqu'une collision est détectée, une exception est lancée, et une instance de la classe **GameOver** est créée pour signaler la fin du jeu.

## 3- Diagramme UML

Pour réaliser nos extensions nous avons implémenté et modifiée de nouvelles classe notamment les classes dont les diagrammes de classe suivent :




## Conclusion

En conclusion, ce projet de développement logiciel a été une excellente opportunité pour mettre en pratique les connaissances acquises en matière de programmation orientée objet. Nous avons ajouté neuf extensions intéressantes au jeu Falling Blox, ce qui a permis d'améliorer l'expérience utilisateur en rendant le jeu plus amusant et plus interactif. Les nouvelles fonctionnalités telles que la détection et la suppression des lignes complètes, la modification de la vitesse de la gravité, l'utilisation des boutons physiques pour gérer le mouvement et la rotation des pièces et la gestion de la fin de partie ont considérablement amélioré le gameplay. Nous avons également utilisé des diagrammes UML pour modéliser les classes ajoutées et modifiées, ce qui a permis une meilleure compréhension de la structure du code. En somme, ce projet a été une expérience enrichissante qui a permis de renforcer nos compétences en matière de programmation orientée objet.