



# Projet DEV4 – C++

---

## Rapport de la version GUI de Tetris en C++

Auteurs : Angielczyk Marcel 60453 & Derboven Téo 60073

### Introduction

Le présent rapport est destiné à fournir des explications sur notre jeu Tetris en interface graphique développé en langage C++ à l'aide du framework QT.

### Changements apportés après la défense console

Nous avons premièrement apporté un changement à notre remise console.

Dans notre implémentation du pattern MVC nous n'avions pas de réelle classe pour la vue ainsi que le controller mais de simples fichiers sources contenant des fonctions.

Nous avons donc créé une classe controller et une classe vue pour notre jeu en version console.

Nous avons également changé l'emplacement du tetromino courant, celui-ci ne se trouve plus dans la classe Grid mais dans Game, ce qui rend la classe Grid plus cohérente.

De plus, nous n'avons plus de classe Bag, puisque nous trouvions qu'elle était inutile si l'on souhaitait respecter la séparation des responsabilités. Nous avons directement implémenté ses fonctionnalités dans la classe Game.

Et finalement nous avons ajouté une méthode **time** dans notre classe Game qui doit être appelée à chaque unité de temps par le **GameControlleur** du gui à l'aide d'un signal.

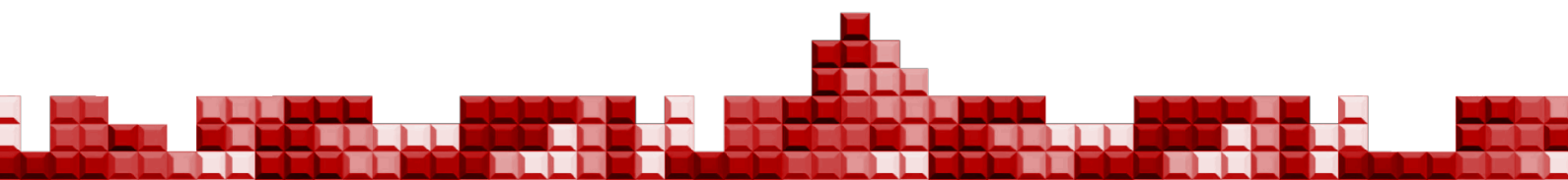
Celle-ci permet au **Game** de faire descendre le tetromino automatiquement et ainsi assurer le déroulement du jeu.

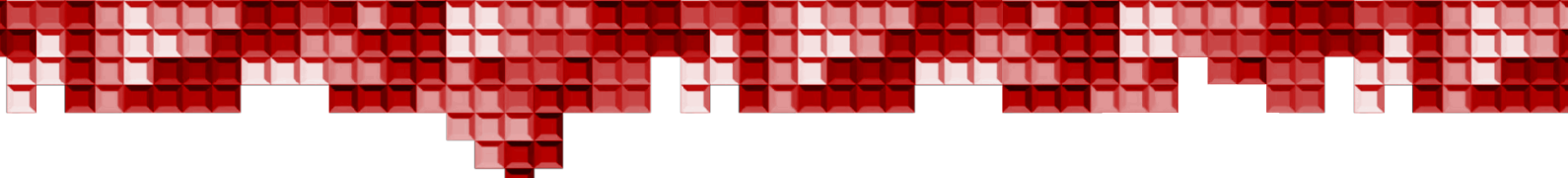
### Les problèmes/solutions rencontrés

Lorsque nous avons commencé à nous renseigner pour réaliser la version gui de notre projet, il nous a été assez compliqué de nous familiariser avec l'outil QT Design qu'on trouvait assez difficile à prendre en main.

C'est pourquoi nous n'avons finalement pas utilisé directement l'éditeur d'interface mis en place par QT Design.

Nous avons préféré mettre en place tous nos éléments d'interface directement en code. Nous trouvons cela bien plus familier dû au fait que les éléments de QT partagent globalement la





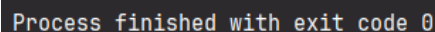
même logique que les éléments de Javafx, sachant que nous avons l'avons appris dans notre cours d'ATLIR4.

Nous avons cependant un problème qui nous a donné du fil à retordre le dernier jour de développement. L'application fonctionnait correctement mais lorsqu'on la fermait, un code d'erreur survenait. Celui-ci stipulait une possible mauvaise gestion de la mémoire ( **exemple de code d'erreur -1073740940 (0xC0000374)** ).

Cependant nous avons bien passé en revue tout le code, nous avons certe des cas de mauvaise gestion de la mémoire, mais nous les avons réglés. Or, pendant un bon moment le code d'erreur nous a résisté.

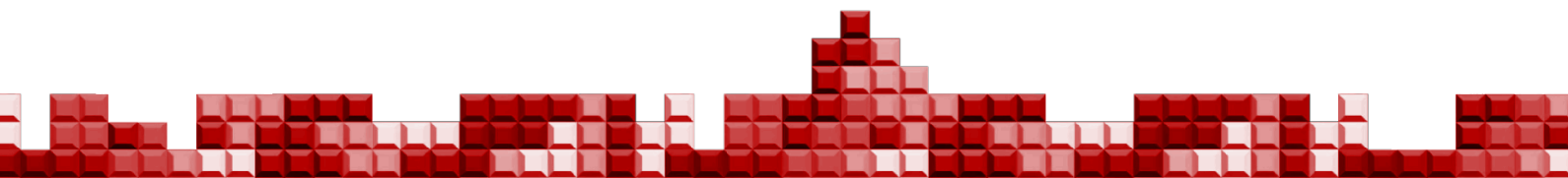
Il s'est avéré qu'il restait un endroit où le problème persistait. Il s'agissait de la création des vues et des contrôleurs dans notre main. Initialement, nous les avons créés directement sur la pile dans le main, mais vu que ces classes descendaient d'un QObject (pour l'utilisation des signaux) l'application QT essayait de libérer leurs adresses à la fin du programme. Du fait que ce n'étaient pas des objets dynamiques, une erreur survient lors de la tentative de libération de ces éléments.

Nous avons donc essayé de les transformer en objets dynamiques et nous avons enfin pu dire au revoir au code d'erreur lors de la fermeture de notre programme.

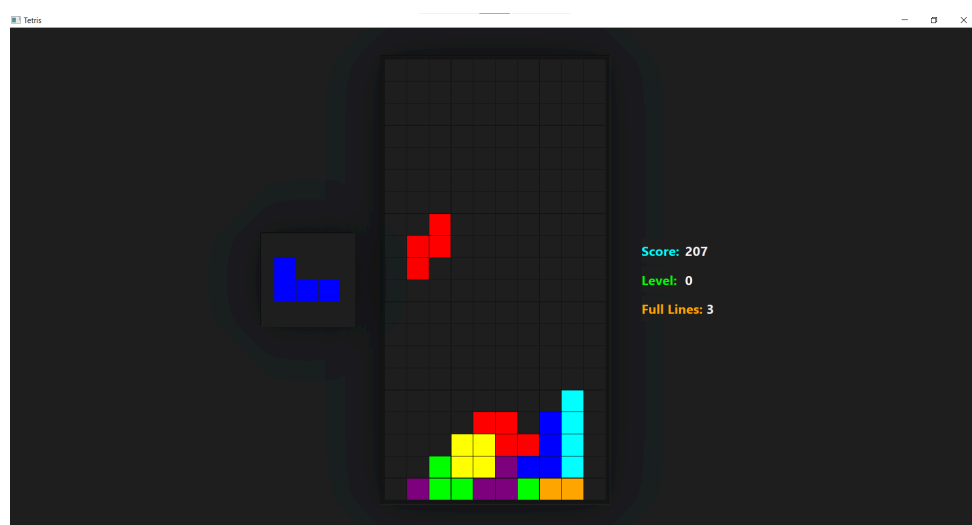
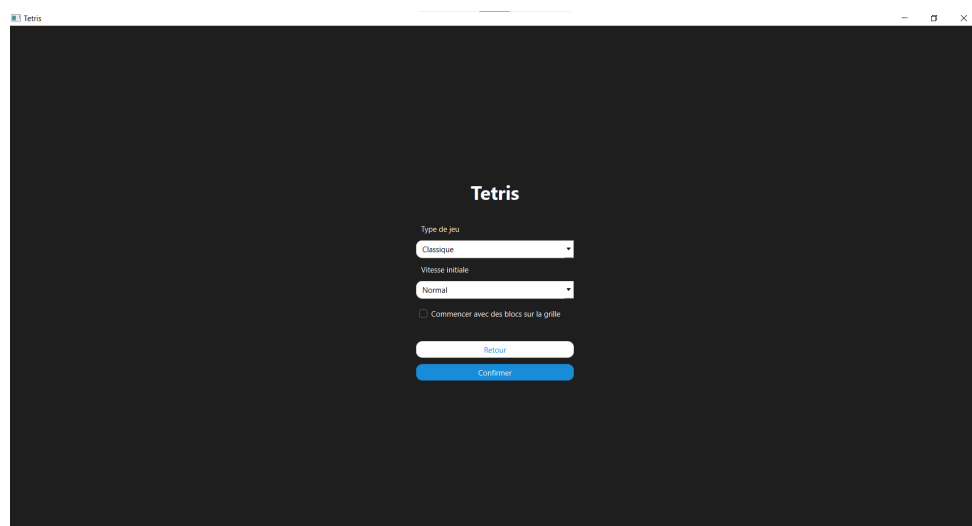
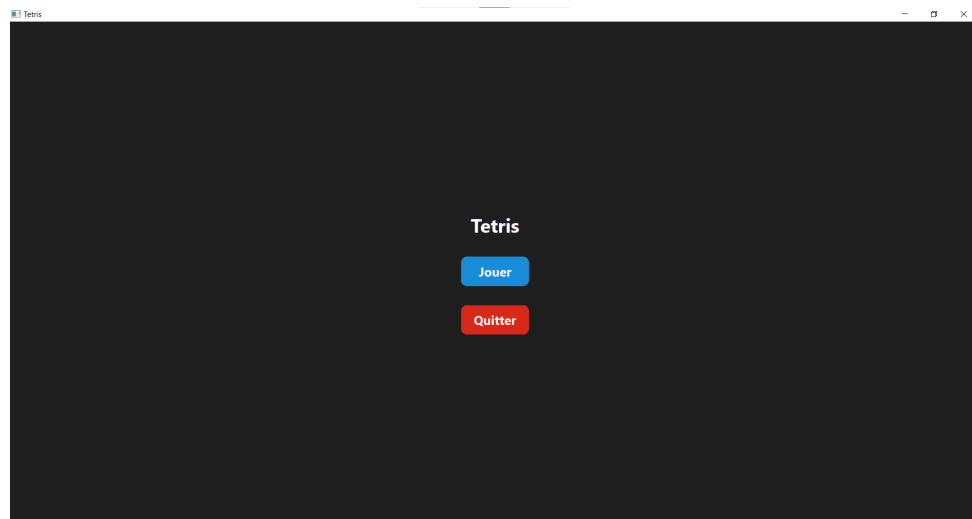


```
Process finished with exit code 0
```

*(C'est notre trophée cette capture)*



# Démonstration de l'interface graphique





## Temps consacré

Nous avons commencé à nous renseigner sur Qt durant nos vacances de mai.

Après avoir déterminé ce dont nous aurons besoin pour réaliser cette interface graphique, nous avons vraiment commencé à coder l'interface graphique le jeudi 9 mai.

Cependant dû à la familiarité avec Javafx nous avons été assez vite pour réaliser cette interface.

De plus, dû au fait que nous étions en congé, nous avons pu consacrer beaucoup de temps à ce projet. Nous passions entre 4 à 6h par jour à travailler sur le projet jusqu'au jour de la remise.

## Conclusion

A présent que ce projet du jeu tetris est terminé, nous pouvons le dire: nous sommes très contents du résultat final. Nous avons pu découvrir et utiliser un nouveau framework jusqu'ici inconnu pour nous.

