Morgan Jully

Killian Mocret

Projet Informatique

SameGame

Tables de matières

I. Introduction

Le projet est un jeu de SameGame dont le but est de faire disparaitre tous les blocs du terrain de jeu. Celui-ci est sous la forme d'un rectangle. Les blocs sont divisés en trois types différents :

-Rouge

-Bleu

-Vert

Ces trois types peuvent former un groupe lorsqu'ils sont cote à cote (pas en diagonal). Un groupe est forcément composer du même type de blocs. Lorsque l'on clique sur un groupe, celui-ci disparait et les blocs qui se situaient au-dessus tombent en restant dans la même colonne. Les blocs change de colonne uniquement si l'une d'elle est complétement vide. A ce moment, les colonnes qui se trouver à sa droite se déplacent à gauche pour que les colonnes vides soit toute situer le plus à droite possible. Le jeu prend fin quand tous les blocs ont été éliminé ou que aucun groupe ne subsiste (blocs seuls ou non relier entre eux).

II. Description

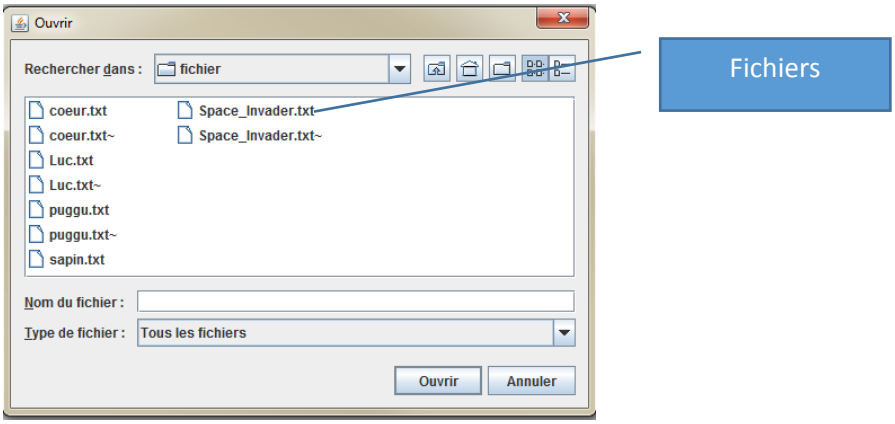
Le projet s'ouvre sur un menu contenant 2 boutons :

-Aléatoire

-Ouvrir un Fichier

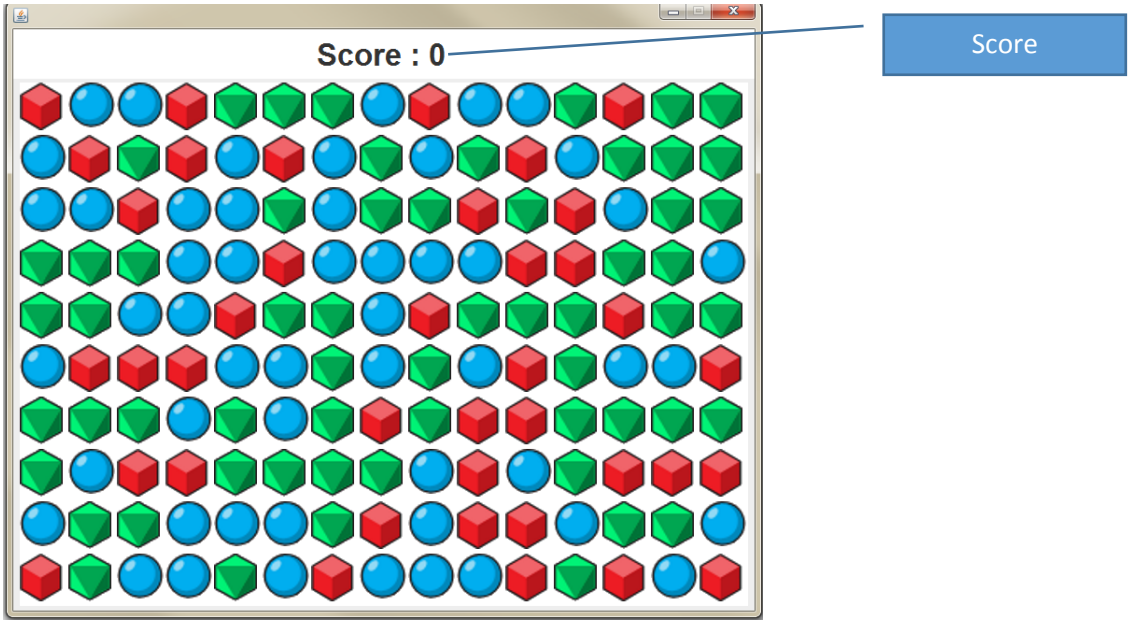
Le bouton « Aléatoire» ferme la fenêtre et en ouvre une autre avec un placement des blocs aléatoires.

Le bouton Ouvrir un fichier lui ouvre une petite fenêtre nous permettant de choisir un fichier qui contient des placements prédéfinit pour les blocs.

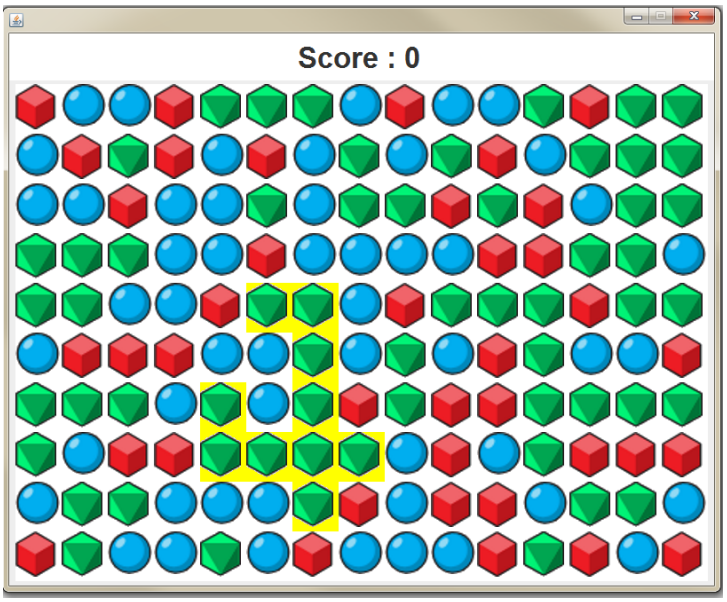


Après l'avoir sélectionné, les 2 fenêtres ce ferme et laisse place au terrain de jeu avec les blocs choisit par le fichier.

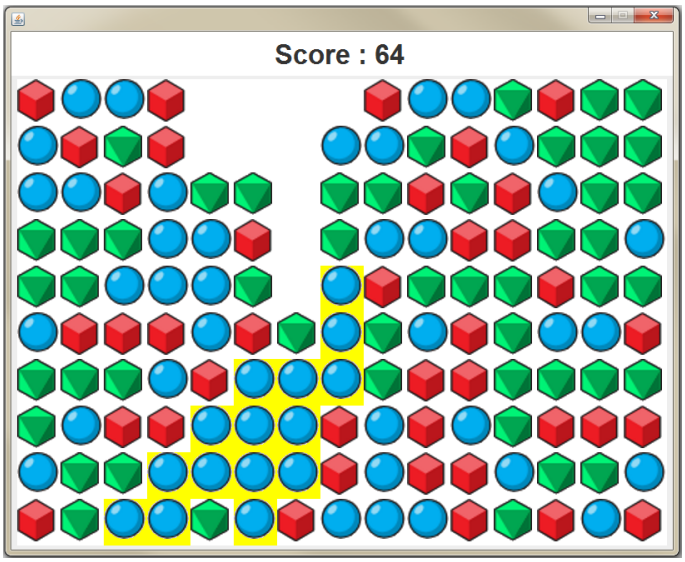
Le terrain de jeu est composé de 3 types de blocs de différentes couleurs et du score du joueur en haut.



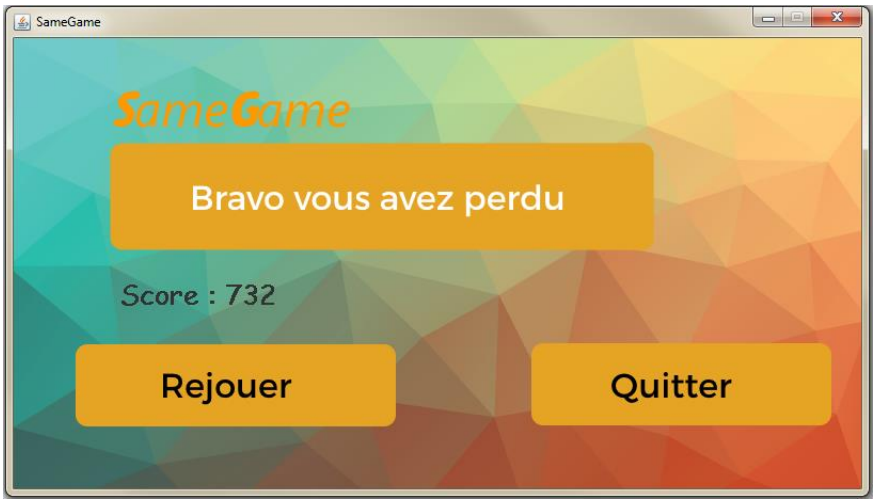
Lorsque le joueur survole un bloc le fond ce dernier ainsi que le groupe auquel il est appartient se modifie en jaune pour permettre la délimitation du groupe



Lorsque celui-ci clique sur un groupe, il disparait et les colonnes se réarrangent de manière à boucher les trous.



La partie continue comme cela jusqu'à ce que le joueur ait retiré tous les blocs ou qu'il ne reste aucun groupe. Lorsque cela arrive, le jeu prévient le joueur que la partie est finie, il lui suffit alors de cliquer sur la fenêtre pour être emmener vers l'écran de fin qui lui donne son score et le félicite de sa partie.



III. Structure du programme

Le programme commence avec la classe « Main ». Cette classe ouvre la fenêtre du menu (taille et location) en créant un objet de la classe Menu qui est composée de 2 boutons (dont les effets visibles sont spécifiés plus haut).

Un controller est ajouté sur cette fenêtre avec la classe ControllerMouseMenu qui gère les clics en fonction de la position de la souris.

Lorsque l'on clique sur un bouton, la classe « ControllerMouseMenu» qui utilise « MouseListener» est appelé, selon la position de la souris lors du clic 2 actions sont possibles :

- **Ouvrir un fichier** : le bouton invoque la classe « JFileChooser » ainsi que « PrintWriter ». JFileChooser ouvre une fenêtre de sélection de fichier dans le dossier « fichier ».

Lorsque le fichier est choisi par l'utilisateur, sont 'adresse ' est transmise à la classe « Lire ».

Celle-ci crée 2 tableaux multidimensionnels de même taille, le 1er contiendra des valeurs (int) qui symboliseront des couleurs et des états des blocs.

Le 2ème est un tableau d'objet qui contiendra des objets de différentes couleurs pour permettre la sélection des blocs.

De plus, elle crée aussi un GridLayout pour ordonner les Objets dans la grille du jeu.

La classe crée un flux de caractère et va lire caractère par caractère les lignes de fichier (.txt) sélectionné.

Elle lancera une double boucle for de la taille des tabeaux qui, selon ce qu'elle lira à chaque passage, elle attribuera dans le tableau de int une valeur (1 pour vert, 2 pour bleu et 3 pour rouge) et appellera la classe de la couleur voulue (Rouge, Vert ou Bleu) et l'attribuera au tableau d'objet Couleur.

Lorsque que la boucle ce termine, la classe «ControllerMouse » est appeler. C'est le cœur du jeu.

- **Aléatoire**: Le bouton invoque la classe «Frame».

Celle-ci crée comme la dernière les 2 même tableaux et dans une double boucle de la taille des tableaux, utilise la méthode « random » pour les remplir de manière aléatoire (hormis le coté hasardeux, même fonctionnement que l'autre class pour l'attribution des couleurs, une valeur dans le tableau int et attribution des classe Vert, Bleu ou Rouge au tableau d'objet Couleur).

Après la boucle, elle crée la fenêtre du jeu. Juste à la suite de cela, le bouton Aléatoire appelle elle aussi la classe «ControllerMouse ».

A partir de ce point-là, le programme se déroule de la même manière.

La classe «ControllerMouse » permet de contrôler toutes les actions de la souris ainsi que les actions liées à la gestion et aux calcules pour les variables du jeu.

Elle utilise les 2 tableaux crée précédemment ainsi que un 3ème qui contient la position des blocs des groupes survolé et 2 variable int pour le score et la taille du groupe.

La méthode « mouseEntered » sert lorsque la souris survole un bloc.

Lorsque cela se produit, la méthode crée une double boucle (toute les doubles boucle sont de la même taille), vérifie la valeur dans le tableau de int du blocs et si elle ne vaut pas -2 (la valeur -2 est attribué aux blocs vide, donc plus utilisables). Apres cette vérification, elle colore en jaune le fond du bloc et invoque la méthode « checkGroupe » (Cette méthode sera décrite en détail plus tard).

Après cela, une nouvelle double boucle intervient pour vérifier les valeurs de tableau groupe. Si elles valent autre chose que -2 ou -1 (le tableau groupe est remplie par défaut de la valeur -1, après l'intervention de « checkGroupe », ce chiffre représente les blocs qui ne font pas partie du groupe) les fonds blocs des concerner sont peint en jaune.

La méthode « mouseExited » intervient lorsque la souris ne survole plus le bloc.

Les fonds des blocs sont repeint en blanc et le tableau groupe est remplie de nouveau de -1.

La méthode « isInGroup » sert à savoir le bloc indiqué a déjà été ajouté au tableau groupe ou non. Elle renvoie un booléen.

La méthode « mouseReleased » agi après un clic du joueur, elle a pour but de :

-colorier en blanc les blocs appartenant au groupe du bloc cliqué.

-passer les valeurs du groupe dans selon les tableaux (-1 pour groupe, -2 pour tableau de int).

-invoquer la méthode « reorganiseColonne » pour faire chuter les blocs encore jouable pour la colonne donné en argument.

-gère le score selon l'algorithme prévu.

-fait appelle à la méthode « checkToutеColonneVide » qui vérifie pour chaque colonne si elle vide. Si c'est le cas et que la colonne concernée n'est pas la dernière, elle appelle la méthode « copieColonne » qui via une boucle, copie la colonne de droite sur celle-ci et la peint en blanc.

-vérifie si la partie est terminée en utilisant la méthode « checkJeuFin » qui vérifie s'il reste un groupe sur le terrain. Si la partie est terminée, la fenêtre se ferme après un clic et laisse place à une autre fenêtre (Objet de classe MenuFin) pour afficher le score et féliciter le joueur. Il peut choisir de quitter le jeu ou de recommencer un partie qui a pour effet de créer un objet de classe Menu.

Enfin, le tableau d'objet est créé à partir de la classe « Couleur » qui attribue une valeur au variable de couleur et utilise la méthode « repaint » qui comme son nom l'indique repeint le cercle d'une couleur. Les classe Vert, Rouge et Bleu hérite de la classe Couleur ce qui permet par substitution de les stocker dans le tableau à 2 dimension d'objet de la classe Couleur.

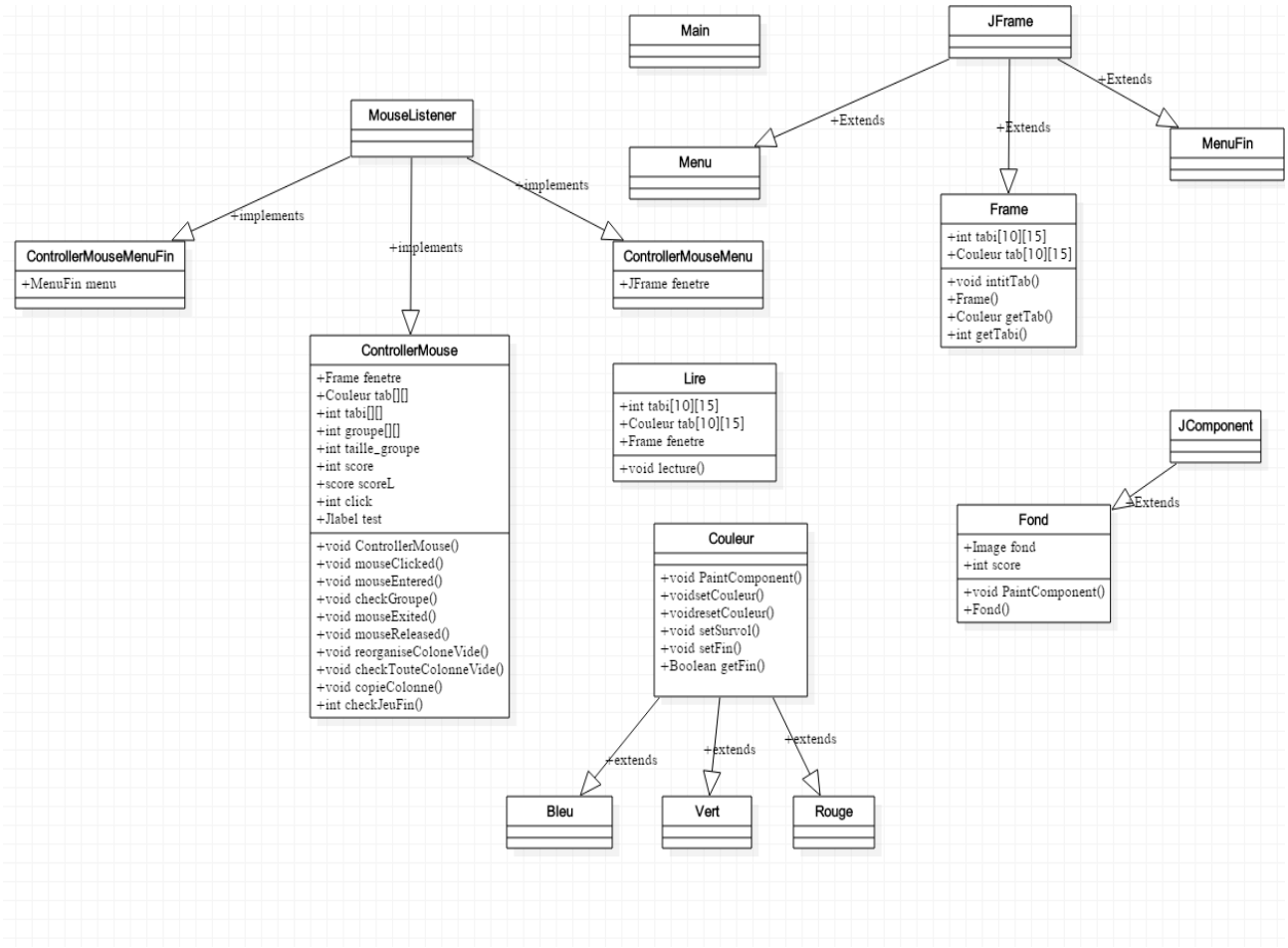
L'algorithme de sélection :

Il est contenu dans la méthode « checkGroupe ».

La méthode vérifie que ligne a bien une valeur plausible dans le jeu, que le bloc juste à coté (à sa gauche) est de la même couleur et qu'il n'est pas vide puis vérifie qu'il n'a pas déjà été ajouté via la méthode « isInGroup ». Si ce n'est pas le cas, le tableau groupe prend la valeur du tableau de int de ce bloc, le compteur de la taille de groupe s'incrémente et la méthode est rappelé avec la même colonne mais une valeur de ligne inferieur de 1. Si la ligne n'est pas supérieure à 0, cette partie de la récursivité s'arrête

Une 2ème fonction elle fait la même chose que la 1ère mais avec les blocs de droite. Elle se rappelle avec une valeur de ligne supérieure de 1. La récursivité s'arrête lorsque la valeur de la ligne est égale à 15.

Les même calcule s'opère avec les colonnes mais avec leur taille est leurs valeurs.



IV. Conclusion personnelle

Mocret Killian :

Ce projet m'aura permis de m'améliorer dans le codage du Java, de mieux comprendre son fonctionnement ainsi que ces spécificité. Il m'a permis de me conforter dans l'idée qu'il vaut mieux faire un projet étapes pas étapes que de faire toutes les chose facile en 1er puis après aviser. De plus il m'a montré que penser à la forme finale du programme avec toutes ses fonctionnalités permet de ne pas devoir modifier son code manière drastique lorsque l'on veut y greffer une nouvelle partie. Pour finir il m'a entrainé à lire le code de quelqu'un d'autre et de devoir le comprendre pour pouvoir adapter son code à celui déjà existant.

Jully Morgan : Pendant ce projet j'ai pu mettre œuvre les savoirs acquis pendant les TP d'APL. J'ai pu me rendre compte que je n'avais pas bien compris certaines fonctionnalités du langage java et donc cela m'a permis de réparer ceci. De plus durant ce projet j'ai été en binôme (ce qui n'était pas le cas avec le Blocus) et donc cela m'a appris à faire un projet à plusieurs (répartitions des tâches etc …).