

Sommaire :

[1. Introduction 2](#_Toc68376680)

[2. Fonctionnalités du jeu 2](#_Toc68376681)

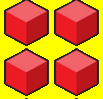
[3. Identification des boules 4](#_Toc68376682)

[4. Structures du programme 5](#_Toc68376683)

[5. Conclusion 6](#_Toc68376684)

**Kévin METRI Bastien LEBLET**

# Introduction

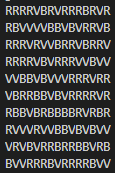
Le SameGame est un jeu en grille (15 par 10 dans notre cas) constitué de 3 boules de couleurs différentes : rouge, bleue et vert. Le but est de vider la grille jusqu’à qu’on ne puisse plus faire de groupes de boules de même couleur. Lorsqu’on survole un groupe de boules similaires, il est mis en évidence en fond jaune.

La stratégie la plus logique dans ce jeu est de faire des groupes de boules le plus grand possible car plus le groupe de boules est grand plus on gagne des points (le nombre de points est équivalent à (boules-1) ^2).

# Fonctionnalités du jeu

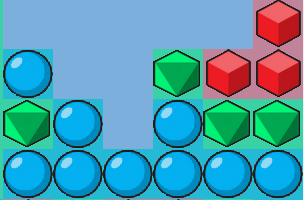
Le jeu commence avec le Main qui lance la classe Menu sui consiste à lancer la fenêtre qui affiche le menu du jeu

C’est la classe ActionMenu qui gère toutes les actions disponibles sur le Menu. On a utilisé l’interface MouseListener qui nous a permis de faire les actions au clique grâce aux coordonnées des boutons par rapport à la fenêtre. Lorsqu’on choisit « Grille Aléatoire » cela lance la méthode aléa de la classe TabAlea qui permet d’initialiser la grille aléatoire de boules. On utilise un objet de la classe Random qui choisit un nombre aléatoire entre 0 et 3 et on l’applique à une variable integer. Lorsque cette variable est égale à 0 il place un R dans le tableau, un B lorsqu’elle est égale à 1 et un V lorsqu’elle est égale à 2. Cette méthode renvoie donc le tableau initialisé avec des couleurs aléatoires comme ci-dessous



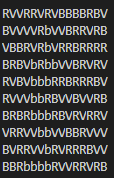
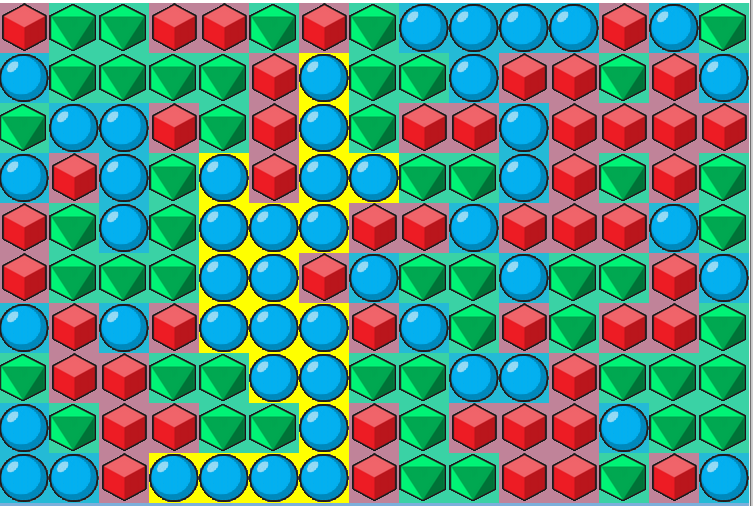
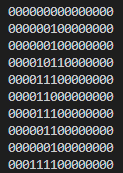
Au contraire la méthode choix de la classe TabChoix permet de choisir la grille que vous voulez à partir d’un fichier .txt (seulement cette extension grâce *à FileNameExtensionFilter* utilisé dans la méthode récupération de la classe RecupFichier qui renvoie le chemin du fichier à la classe TabChoix*)* si ce fichier est constitué de caractères R B V en 15 par 10( si ce fichier n’est pas en 15 par 10 cela renvoie une erreur). La méthode récupère le fichier, le décortique ligne par ligne et décortique la ligne caractère par caractères. On met alors les caractères un par un dans le tableau.

C’est alors que grâce à la classe Remplir\_Tab que l’on affiche les différentes boules. On déclare nos différentes boules en ImageIcon que l’on applique dans un JLabel. Lorsqu’une case du tableau est égale à la couleur, cela applique le JLabel à la grille de jeu. ON utilise R B V lorsque les boules ne sont pas survolées et r b v lorsqu’elle sont survolé. Cela nous permet de changer leurs valeurs dans le tableau temporairement dans le tableau pour pouvoir les remettre à leur état initial lorsqu’elles ne sont plus survolées. Lorsqu’on retire un groupe de boules elles sont remplacées par un “ ” dans le tableau ce qui les remplace par un fond bleu clair à l’affichage



Ce tableau est utilisé dans la classe centrale du jeu Grille.

# Identification des boules

L’identification de groupe dépend principalement de la méthode AutourCase de Grille qui prend en argument les coordonnées de la souris au mouvement grâce au méthode RecupX et RecupY qui renvoie les coordonnées à l’instant t. Cette méthode vérifie dans les cases adjacentes il y a dans le tableau une boule de même couleur. IL y a 4 if qui correspond à la vérification de la gauche, la droite, du haut et du bas de manières récursives. Lorsqu’il y a une boule de même couleur cela change sa valeur 1 dans un tableau prévu exclusivement pour le survol et le calcul des points (initialisés à 0). Et change sa lettre dans le tableau en le passant en minuscule ce qui change son image par celle en avec le fond jaune

Lorsqu’on quitte un groupe sans cliquer dessus, cela remet les lettres correspondantes en majuscules et on réinitialise le tableau de survol. Lorsqu’on clique sur un groupe cela le remplace par un “ ” dans le tableau ce qui le remplace par le fond bleu. Ensuite la méthode ChuteBoule vérifie si des boules ont en dessous d’eux “ ” et si c’est le cas il les intervertis dans le tableau ce qui donnent l’effet de chute des boules. Pour le décalage à gauche lorsqu’une colonne est vide, nous avons repris le même principe avec les méthodes VerifColonnes et BougeGauche. VerifColonnes vérifie la colonne que BoucheGauche lui envoie est vide et elle envoie si c’est le cas. Quand c’est le cas les boules prennent la place des cases tant qu’il y a colonnes vides à leur gauche. La méthode TailleGroupe come son nom l’indique compte le nombre de boules dans le groupe en comptant le nombre de 1 dans le tableau de survol. C’est ensuite la méthode Score qui fait le calcul du score en prenant le nombre de boules en compte. On réinitialise le nombre de boules lorsqu’on réinitialise le tableau de survol. Le score est ajouté à l’instant t à un JLabel lui-même dans JPanel situé en bas de la fenêtre. À chaque fois que l’on clique la méthode VerifFin inspiré de la méthode d’identifcation des groupes vérifie s’il n’est plus possible de faire de groupes et si c’est le cas la méthode Fin ferme la fenêtre actuelle et appelle la classe Fin qui affche la fenêtre de fin de jeu.

# Structures du programme

# Conclusion

**Kévin** : Ce projet m’a permis de mettre à l’épreuve mes compétences en Java acquises durant les TP mais aussi le travail en roupe. Ce projet touchait un peu tous les concepts mais le projet m’a fait les approfondir comme les évènements à la souris. Nous n’avons eu quelques problèmes sur la détection du groupe de blocs ce qui nous a bloqué une semaine mais après cela nous n’avons eu aucune grosse difficulté. Ce projet m’a surtout permis de voir mes progrès par rapport au projet de C qui m’a posé de grosses difficultés sur ma manière d’aborder les grosses notions des langages de programmation et le changement de méthode d’apprentissage que j’ai réalisé.