SameGame

Sommaire :

Introduction……..3p

Description des fonctionnalités……..3p
-Menu……..3p
-Paramètres……4p
-Jeu……6p
-Fin……..8p

Présentation de la structure du programme……9p-11p

Explication des données d'une partie en cours……12p

Conclusion personnelle……13p

Introduction :

SameGame est un jeu constitué d'une grille avec des blocs de trois types différents dispersés aléatoirement ou bien placés par le joueur par le biais d'un fichier. Le but est de vider la grille en faisant le plus grand Score. Pour cela, il vous suffit juste de réunir deux blocs du même types au minimum afin de vider la grille.

Description des fonctionnalités :

-Menu :

Sur le menu, nous pouvons voir trois boutons.
Le premier bouton « Jouer », permet de lancer jeu avec une grille aléatoire ou importée par le joueur si il l'a décidé dans les paramètres.

Le deuxième bouton « Paramètres », correspond aux paramètres de la partie.

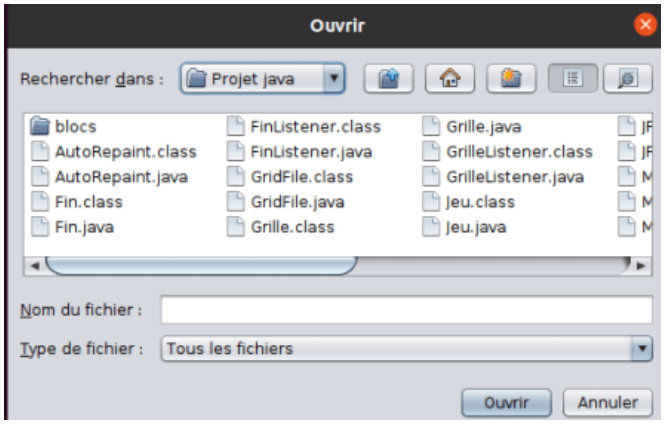
Le troisième bouton « Quitter », permet de quitter le programme. –

-Paramètres :

Dans les paramètres, on peut voir quatre boutons, deux au centre de la fenêtre puis deux autres en bas de la fenêtre.
Les deux boutons au centre « Random » et « Importer grille » sont exclusif l'un l'autre, c'est-à-dire que lorsque vous cliquez sur l'un d'eux, il se colore en jaune, et l'autre se colore en violet, ce qui permet à l'utilisateur de savoir quelle option est choisie. Le bouton « Random » permet de générer une grille aléatoire.
PS : Le bouton « Random » a été renommé en « Aléatoire ».

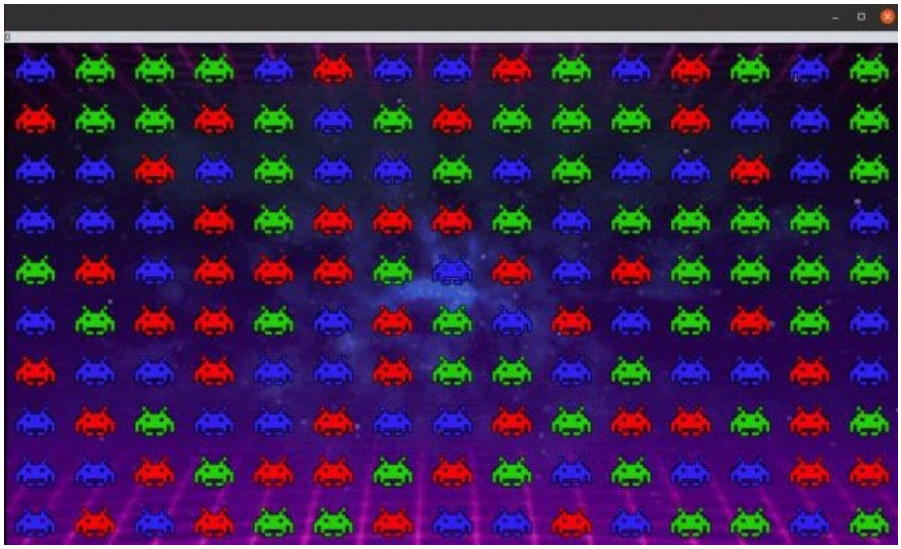
Le bouton « Importer grille », est un peu spécial car il ouvre une fenêtre de sélection de fichier qui permet d'importer sa propre grille, mais le bouton « Importer grille » se colore en jaune uniquement si la grille en question a bien pu être récupérée dans le fichier, sinon les deux boutons restent inchangés et la dernière option est gardée en mémoire. Par exemple, si vous aviez importé une grille qui avait fonctionné et que le bouton « Importer grille » s'était coloré en jaune, si vous recliquez dessus, vous aurez la possibilité de changer de fichier, mais si ce dernier n'est pas exploitable ou est introuvable, vous garderez la grille précédemment importée.

Voici la fenêtre de sélection de fichier qui apparaitra en cliquant sur « Importer grille » :



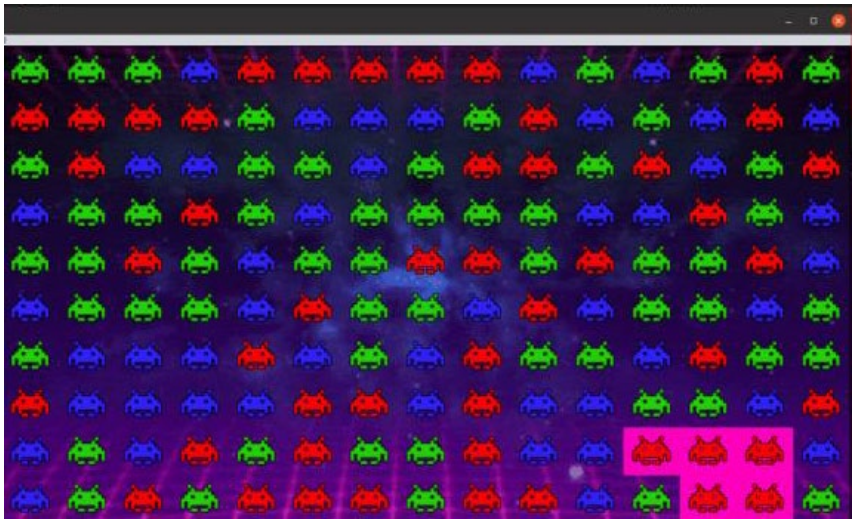
Enfin, on peut voir deux boutons en bas de la fenêtre, le bouton de gauche « Retour » permettant de revenir au menu principal et le bouton de droite « Jouer » permettant de lancer la partie.

-Jeu :



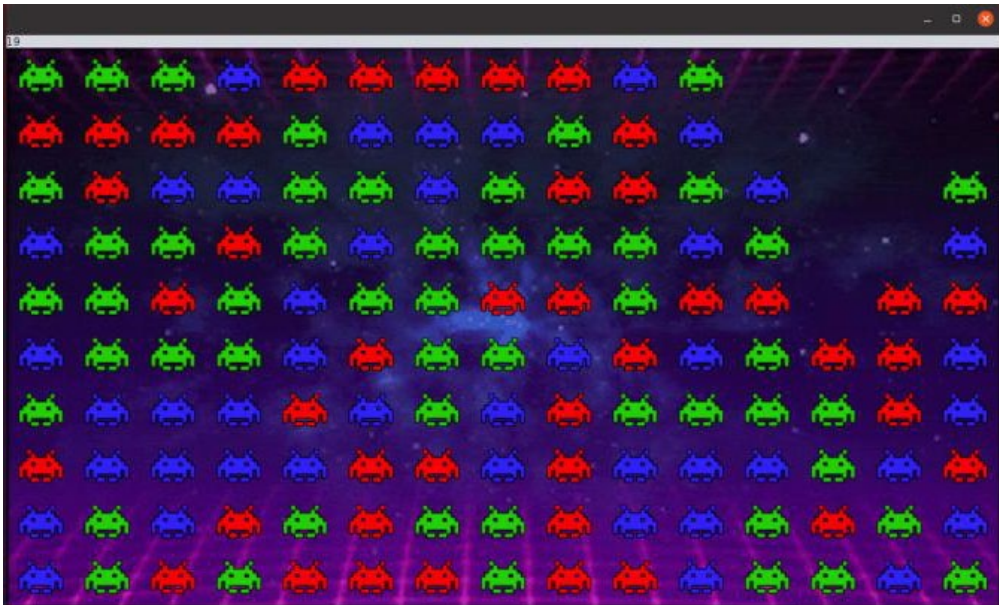
Après avoir appuyer sur le bouton « Jouer », une grille de 10 lignes par 15 colonnes apparait avec des aliens qui se sont échapper de Space Invaders.
Ces aliens, se distingue par trois couleurs différentes le bleu, le vert ainsi que le rouge. Chaque couleur correspond à une race précise d'aliens. Par exemple les rouges ne peuvent pas être mélanger avec les bleus.

Pour éliminer ces aliens, il faut qu'ils soient regroupés au minimum par deux. Si cela est possible, ils auront un fond rose lorsque l'on passera la souris au-dessus et ils indiqueront aussi s'il y a un membre de leur race qui les touches, comme indiqué ci-dessous :

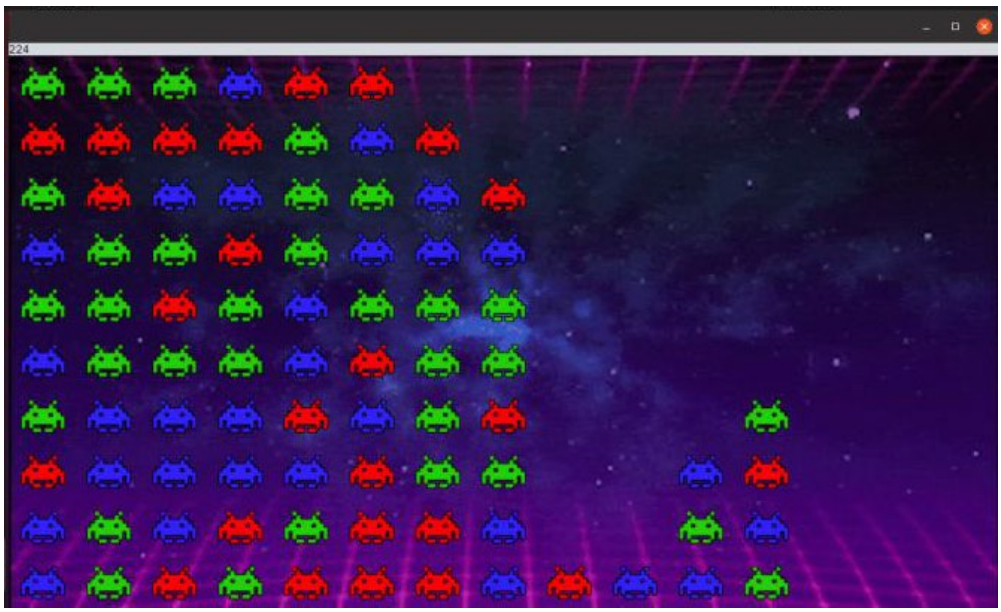


Pour éliminer un groupe d'aliens il suffit de cliquer dessus.
Plus le nombre d'alien éliminé en une fois est grand plus vous gagnez de points.

Ainsi lorsque vous éliminé les aliens chutent de haut en bas de manière à combler les trous.



Enfin, quand une colonne est complètement éliminée, les colonnes situées à droite de celle-ci seront déplacées vers la gauche, comme ci-dessous :

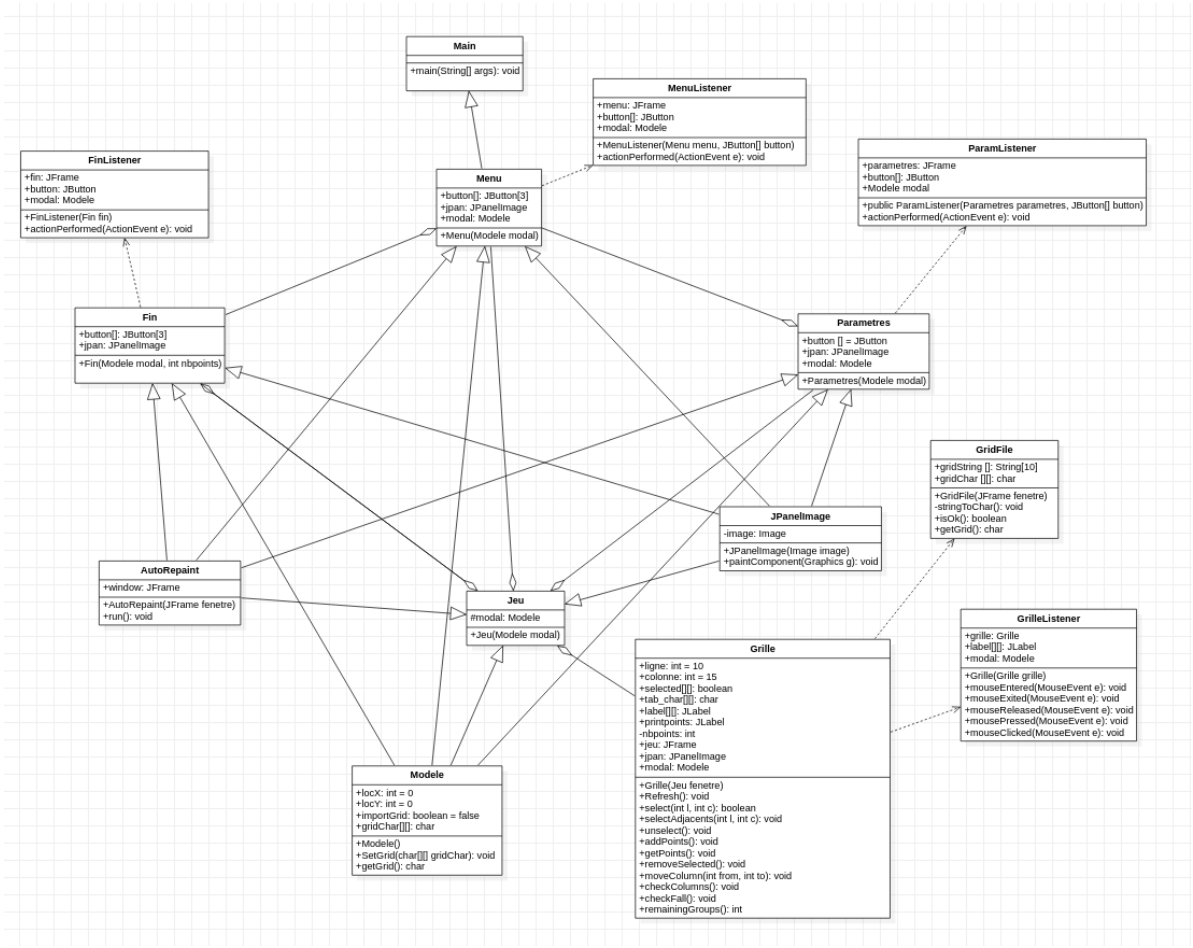


La partie dure jusqu'à ce qu'il n'y ai plus aucun groupe d'aliens.

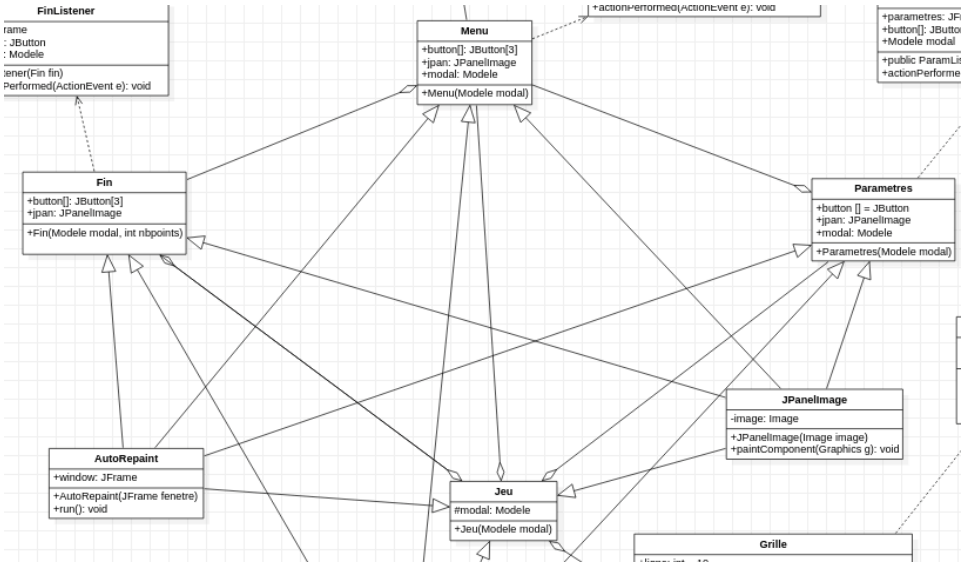
-Fin :

Une fois la partie terminée, le score du joueur est affiché au milieu de la fenêtre. Il y a trois boutons en bas de la fenêtre. Le bouton en bas à gauche permet de relancer une partie avec la même grille si elle avait été importée ou avec une grille aléatoire, le boutons en bas au milieu permet de revenir au menu principal et le bouton en bas à droite permet de quitter fermer le programme.

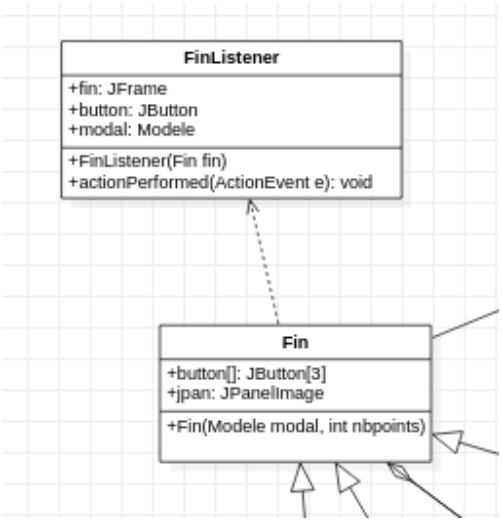
Présentation de la structure du programme :



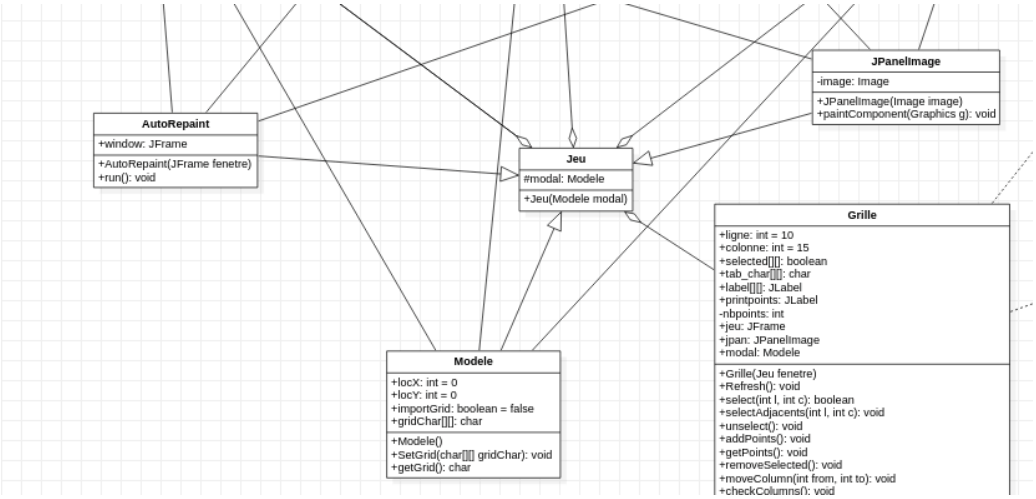
Voici, ci-dessus le diagramme de classe de notre jeu.
On peut voir au centre du diagramme un cycle d'agrégation entre les classes.



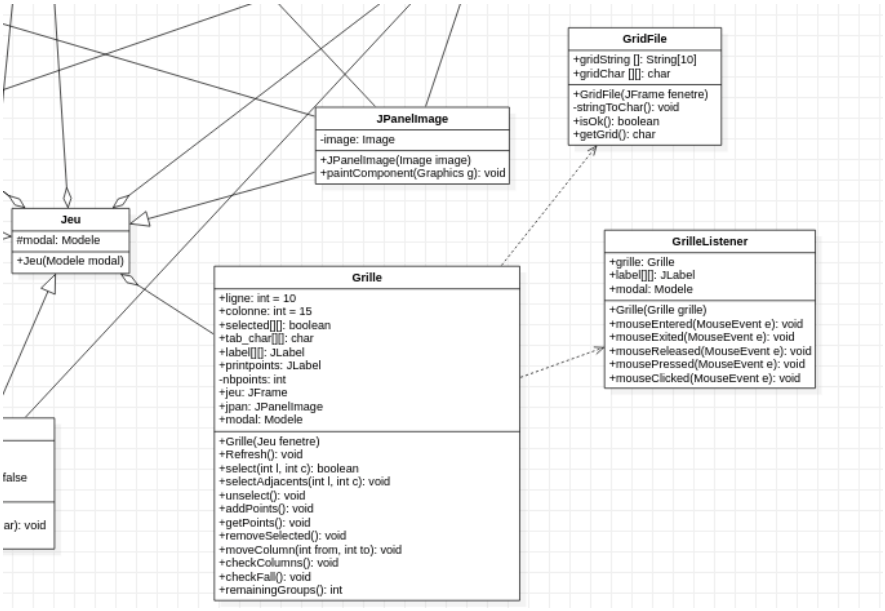
En effet, les classes Menu, Parametres, Jeu et Fin sont liées car lorsque que l'on appuyé sur les différents boutons qui ont des actions spécifiques qu'on verra juste après, on ouvre une autre fenêtre qui corresponds soit à paramètres soit à jeu etc. Ce cycle est vital dans notre jeu car il communique entre eut et permet de transverse des informations qui sont utilise dans d'autre classe.



Dans notre diagramme on peut constater que chaque classe qui représente une fenêtre possède une classe Listener (ci-dessus il s'agit du Listener de la classe Fin).
Ici, le Listener est une interface par classe, elle nous permet d'effectuer toutes les actions de la souris sur les boutons par exemple.



Ensuite, les classes AutoRepaint, Modele et JPanelImage sont des classes qui hérite des classes principales (Menu, Jeu, Parametres et Fin). Ces classes permettent la réalisation de l'esthétique du jeu ou dans le cas de Modele de gérer la grille importée.



Enfin, par la classe Jeu, on a fait une agrégation de la classe Grille qui possède elle-même deux interfaces de classe GridFile ainsi que GrilleListener.
La classe GridFile permet de choisir le fichier qu'on veut importer pour notre jeu. La classe GrilleListener est comme d'écrit précédemment la classe qui permet les actions de la souris sur la grille. Et pour finir, la classe Grille regroupe toutes les informations nécessaires pour créer une grille mais aussi toutes les vérifications afin d'attribuer des points mais aussi le déplacement de colonne ou/et de blocs lors d'une partie en cours.

Exposition de l'algorithme qui identifie les groupes :

Nous allons maintenant expliquer comment se déroule l'algorithme qui permet d'identifier un groupe, c'est à dire nos méthodes select(ligne, colonne) et selectAdjacent(ligne, colonne).
Pour commencer, nous allons recenser les informations que nous allons traiter dans cet algorithme :
- tab_char Un tableau de caractères en deux dimensions qui contient pour chaque case sa couleur symbolisée par une lettre ('R' : Rouge ; 'V' : Vert ; 'B' : Bleu ; ' ' : Vide)

- selected Un tableau de boolean en deux dimensions qui contient pour chaque case : la valeur true si elle est sélectionnée, et false si elle ne l'est pas

Pour ces deux tableaux les indices correspondent aux coordonnées de chaque case, par exemple pour la case sur la 1ére ligne et la 5éme colonne, sa couleur sera stockée dans tab_char[0][4] (-1 car les indices commencent à 0).
Maintenant que nous avons fait ce point, nous pouvons commencer, tout d'abord nous devons appeler select(ligne, colonne) qui va vérifier si la case en question n'est pas vide c'est à dire si tab_char[ligne][colonne] est different de ' ', si c'est le cas la méthode return false et s'arrête, car nous ne voulons pas que le joueur puisse sélectionner un groupe de cases vides.
Ensuite nous allons vérifier si la case au-dessus est du même groupe que notre case et si elle n'est pas déjà sélectionnée, et si ces conditions sont réunis, nous incrémentons la variable counter, nous positionnons la valeur du tableau selected de la case du dessus sur true, et nous appelons la méthode selectAdjacent sur cette dernière. Nous répétons cette opération pour la case du bas, de gauche, puis de droite. Ensuite nous vérifions si la variable counter est supérieur à 0, c'est à dire si la case choisie dans le select au départ à au moins 1 cases de même couleur adjacente à elle, si ce n'est pas le cas cela veut dire que la case est toute seule et qu'elle ne fait pas partie d'un groupe, et donc elle ne doit pas être sélectionnée au passage de la souris du joueur. Seulement si la variable counter est supérieur à 1 cela veut dire qu'elle fait bien partie d'un groupe et elle est donc mise en évidence (colorée en rose).

La méthode selectAdjacent est la même que la méthode select sauf qu'elle colore en rose immédiatement la case en question sans vérifier si elle est n'est pas vide et fait partie d'un groupe.
Ce qui fait que cela crée une récursivité de sélection jusqu'à ce que toutes les cases du groupe soient sélectionnées.

Le return de la méthode select() sert uniquement pour la méthode remainingGroups() qui permet de connaitre le nombre de groupes restants.

Conclusion personnelle :

Brice PANIZZI
C'est le premier projet que je programme en Java. Durant ce projet j'ai appris de nombreuses choses dont je n'avais même pas consilience ou que je ne savais pas faire auparavant. Au début je pensais que ce projet allait être long à réaliser. Mais au final j'ai trouvé ça tellement intéressant et instructif que je n'ai même pas vu le temps passer pour réaliser le projet. C'est aussi grâce à mon collègue qui a su m'expliquer et me diriger sur certaines choses à simplifier et épurer au maximum pour avoir un code le plus lisible et le plus compréhensible pour tous.

Paul LE CORRE
Cette fois ci nous avons directement commencé par le jeu au lieu du menu, et nous avons plus vu le menu plus comme un « bonus » qu'une nécessité c'est pourquoi tout nous a paru bien plus rapide que pour le projet en C où nous avions fait le jeu en dernier. Cela étant, il est vrai que nous avons commencé le projet bien plus tôt que pour celui en C, et c'est peut-être l'absence de ce sentiment de pression qui nous à faire sentir plus libre en termes de temps. J'ai trouvé le Java bien plus enclin à la création de jeu que le C même si j'avoue que je n'aimais pas du tout le Java au début, j'ai appris à découvrir ses points fort.