

# **PROJET TUTORÉ**

**GAUDRU Rémy**

**&**

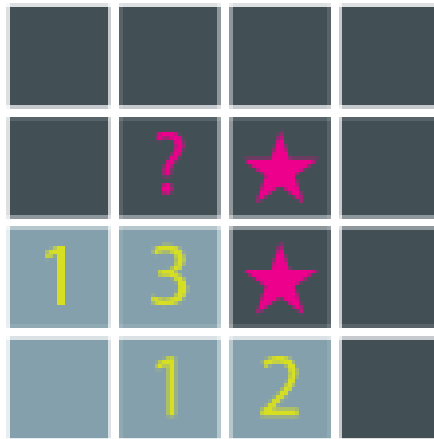
**Jordan DEMARTIN**

## SOMMAIRE

<u>I – INTRODUCTION.....</u>	<u>3</u>
<u>II – DESCRIPTIONS DES FONCTIONNALITES.....</u>	<u>4</u>
<u>III – PRESENTATION DE LA STRUCTURE INTERNE DU PROGRAMME.....</u>	<u>7</u>
<u>IV – ALGORITHME POUR REVELER PLUSIEURS CASES.....</u>	<u>9</u>
<u>V – MECANISME DE SAUVEGARDE.....</u>	<u>10</u>
<u>VI – CONCLUSIONS PERSONNELLES.....</u>	<u>11</u>

## I – INTRODUCTION

Ce projet tutoré, lancé en avril 2018, consiste à réaliser en langage orienté objet (ici, le java) un petit jeu connu de tous, le démineur.



Celui-ci est un jeu de réflexion dont le but est de découvrir parmi une grille toutes les cases ne possédant pas de mine.

Avant de commencer une partie, l'utilisateur a le choix du nombre de lignes et de colonnes que possèdera la grille (entre 4 et 30). De même, il peut choisir le nombre de bombes/mines qu'il y aura (entre 0 et autant que la grille le permette).

Une fois en jeu, le joueur peut :

- Faire un clic gauche sur une case afin de révéler la case (bombe ou non)
- Faire un clic droit sur une case afin d'y dessiner une étoile pour marquer qu'il y a une mine/bombe, ou bien un point d'interrogation afin d'exprimer un soupçon.

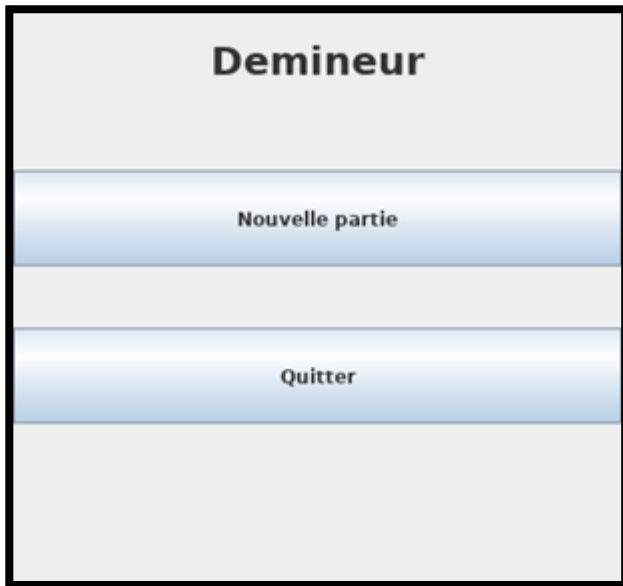
Durant la partie le joueur doit être informé du nombre de bombe(s) restante(s) dans la grille et doit pouvoir sauvegarder la partie en la quittant afin de la reprendre plus tard.

## II - DESCRIPTION DES FONCTIONNALITES

### a) Menu

Lorsque le programme est lancé, 2 choix de menu sont possibles.

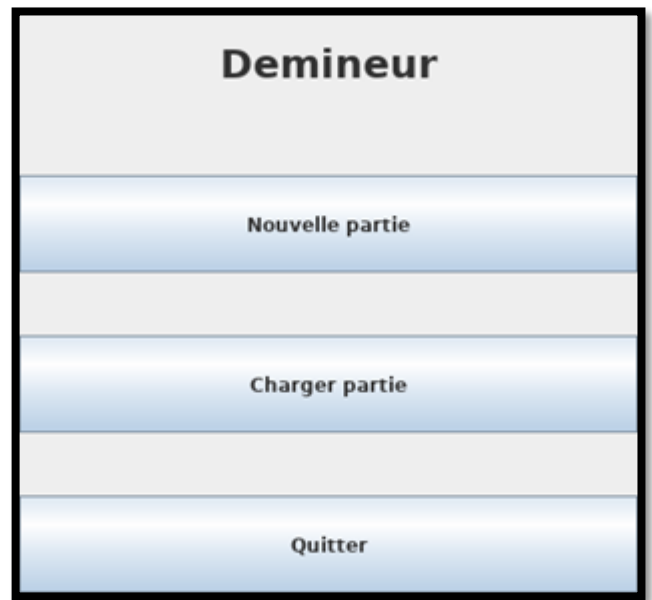
Dans le premier cas, aucune sauvegarde n'a été trouvée dans le dossier du jeu, alors il n'offre que 2 choix au joueur :



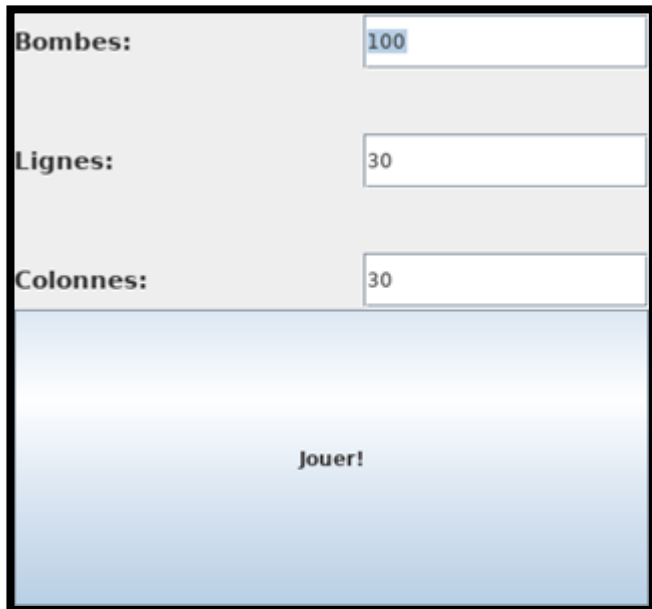
- **Commencer une nouvelle partie :** qui emmène le joueur dans un second menu afin de préparer la grille de jeu qu'il souhaite avant de lancer la partie.
- **Quitter :** qui permet de fermer la fenêtre et quitter le programme.

Dans le second cas, le programme a détecté un fichier de sauvegarde, alors un 3<sup>ème</sup> choix est proposé au joueur dans le menu :

- **Charger partie :** qui permet au joueur de reprendre une ancienne partie qui a été sauvegardée.



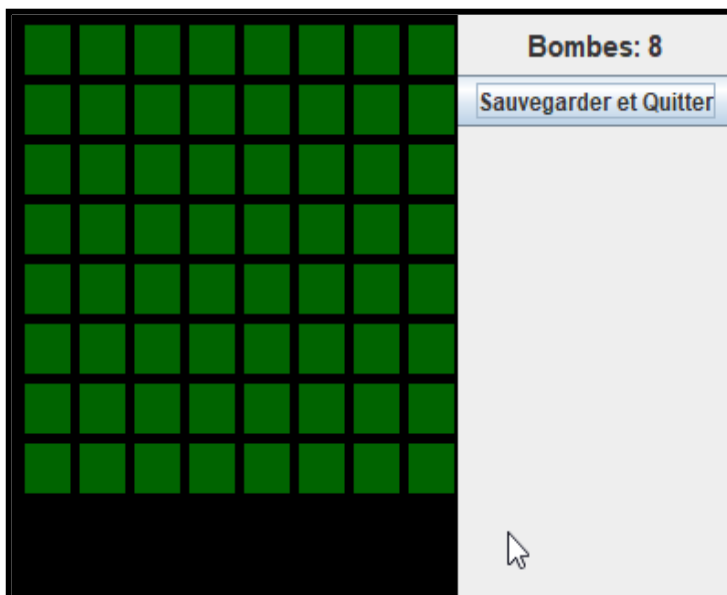
Le menu permettant au joueur de préparer sa grille est constitué ainsi :



- Un champ texte afin d'écrire le nombre de **bombe(s)** qu'il souhaite.
- Un champs texte afin d'écrire le nombre de **lignes** qu'il souhaite.
- Un champs texte afin d'écrire le nombre de **colonnes** qu'il souhaite.
- Un bouton « **Jouer !** » qui permet de lancer la partie avec les caractéristiques que le joueur a choisi pour sa grille.

## b) Le jeu

Lorsque le joueur lance une nouvelle partie, celui-ci arrive sur cet écran. Il est composé en 2 parties :



- A gauche, la grille avec les caractéristiques qu'il a choisies
- A droite, le nombre de bombe(s) restante(s). Ce compteur est décrémenter à chaque marqueur étoile que le joueur place. Ainsi qu'un bouton « Sauvegarder et Quitter » qui permet au joueur de sauvegarder sa grille et revenir au menu principal.

Une case non révélée est de couleur verte.

Lorsque le joueur clique sur une case, celle-ci se révèle et devient de couleur blanche et affiche le nombre de bombe(s) qu'elle possède autour d'elle ou rien si elle n'en a pas. En cas de clique sur une case avec une étoile, par sécurité, la case ne se révèle pas.

Un premier clic droit sur une case non révélée colore la case en orange et affiche une étoile. Un second clic droit permet d'afficher cette fois-ci un point d'interrogation sur un fond jaune.

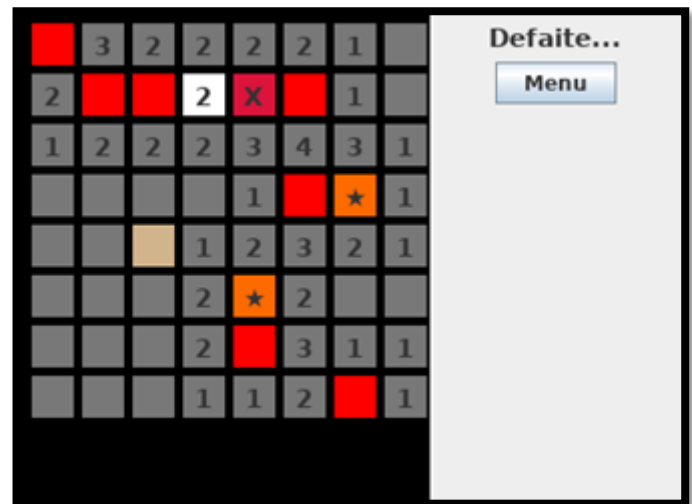
La défaite est détectée dès que le joueur appuie sur case qui possédait une mine. Toutes les cases sont alors révélées.

Les cases minées sont coloriées en rouge sauf celles qui étaient marquées d'une étoile, et une croix est dessinée sur la case responsable de la défaite.

S'il y avait un marqueur étoile mais que la case n'était pas une bombe alors la case est colorée en beige.

Les cases déjà révélées restent blanche, et les non révélées deviennent grises. Ainsi le joueur peut contempler sa grille avant de quitter.

Le message « Défaite... » est alors affiché à la place du nombre de bombe(s).



La victoire est détectée si toutes les cases sans bombes sont retournées, il s'affiche alors le message « Victoire ! » à la place du nombre de bombe(s).

Le joueur peut alors admirer sa grille et revenir au menu afin de recommencer une nouvelle partie.



- BoutonInfoJeu : Cette classe permet de gérer les actions du bouton contenu dans la classe InfoJeu, ici elle permet généralement de quitter, et dans certains cas de lancer la sauvegarde de la partie.
- ActionFenetre : Cette classe permet d' « écouter » les actions de la fenêtre, ici elle permet de lancer la sauvegarde si celle-ci se ferme.
- Menu : Cette classe permet de créer une fenêtre avec des boutons pour que le joueur puisse lancer une nouvelle partie ou continuer une partie sauvegardée.
- BoutonMenu : Cette classe permet de gérer les actions des boutons contenus dans la classe Menu.
- Main : Ceci est l'amorce du programme, c'est lui qui lance le jeu en appelant la classe Menu.



## **IV - ALGORITHME POUR REVELER PLUSIEURS CASES**

Une case vide est une case qui n'a aucune mine dans les 8 cases autour de celle-ci.

Quand l'utilisateur découvre une case vide, il ne sert à rien de le laisser cliquer sur toutes les cases autour de celle-ci sachant qu'il n'y a aucune mine, donc, pour accélérer le rythme de la partie, on révèle automatiquement les 8 cases autour d'une case vide et si, durant cette opération, on révèle une nouvelle case vide, alors on révèle pour elle aussi toutes ses voisines, et ainsi de suite.

Pour respecter ce principe, dans notre morceau de code qui révèle une case quand on clique dessus, nous avons appelé une méthode qui a pour effet de révéler toutes les cases autour d'une case donnée.

Dans cette méthode mentionner précédemment, quand nous révélons une case et que celle-ci s'avère être une case vide nous appelons à nouveau cette méthode en donnant cette fois la nouvelle case vide révéler.

Cette méthode va donc révéler, par propagation, l'ensemble des cases vides adjacentes.

## **V - MECANISME DE SAUVEGARDE**

Dans le démineur, quand l'utilisateur quitte le jeu et qu'une partie est en cours, la partie doit être sauvegardée.

Pour se faire, nous avons fait en sorte que les moyens de sortie du jeu déclenche automatiquement la sauvegarde.

Donc, en cas de fermeture de la fenêtre ou de l'utilisation du bouton permettant de quitter, une sauvegarde est réalisée.

Un flux d'écriture est ouvert et est transférer à la grille et aux cases pour que celle-ci enregistrent les informations essentielles permettant postérieurement de les recréer.

Pour la grille, on sauvegarde :

- Le nombre de bombes
- Le nombre de lignes
- Le nombre de colonne

Pour chaque case, on sauvegarde :

- La présence de bombe
- L'état de celle-ci (Révélée ou non)
- La façon dont elle est marquée. (★ ou ?)

Pour ce qui de la restauration de la partie, quand l'utilisateur le demande, le programme va créer une grille avec un nombre de bombes, un nombre de lignes et un nombre de colonnes correspondant à ce qu'il y'a dans le fichier de sauvegarde.

Ensuite, pour chaque case : le programme va lire dans le fichier de sauvegarde les trois informations qui la concerne.

Une fois que les informations des cases sont conformes à la sauvegarde, on remet en place la coloration et les flags de chacune avant de mettre à jour le nombre de bombes dans les informations de la partie.

## **VI - CONCLUSIONS PERSONNELLES**

### **Rémy Gaudru :**

Travailler sur ce nouveau projet tutoré a encore été un plaisir. En effet, depuis que nous avons commencé à apprendre le JAVA, j'ai eu hâte de découvrir le nouveau sujet qui nous sera proposé.

J'ai trouvé le démineur (beaucoup) plus simple à réaliser que le Taquin, en effet aucune difficulté n'est arrivé pendant le développement du jeu. Ainsi, avec mon camarade nous avons terminé ce projet en peu d'heure (-10h).

Pendant le développement du jeu, j'ai trouvé le JAVA particulièrement approprié à ce genre de projet. En effet, la réalisation d'une « application graphique » comme le jeu du démineur se fait très rapidement grâce aux outils mis à disposition par ce langage.

Pour conclure, j'ai vraiment pris du plaisir à réaliser ce jeu du démineur avec mon camarade, car, nous avons obtenu un résultat final très satisfaisant.

### **Jordan Demartin :**

J'ai personnellement beaucoup apprécié travailler sur ce démineur, particulièrement sur la sauvegarde et sur la récursivité car c'est toujours aussi plaisant de voir ses idées se concrétiser.

Je trouve que nous avons été plus efficace et rapide dans la réalisation du démineur par rapport à celle du taquin, nous avons pu rapidement terminer, en moins d'une dizaine d'heures, et avoir 2 semaines pour régler les divers bugs qui aurait pu se présenter.

En rendant ce projet je pense que nous avons respecté toutes les consignes et que nous avons produit un résultat dont je peux être fier.