SAÉ 2.1
Démineur

Introduction

Description des fonctionnalités du programme

Présentation de la structure du programme

Sauvegarde d'une partie

Révélation de plusieurs cases

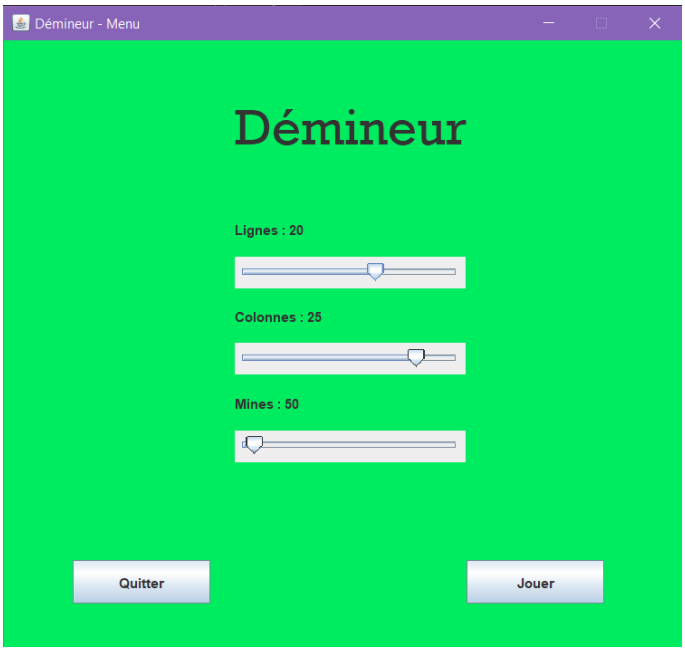
Conclusion

Introduction :

Le démineur est un solitaire où des mines sont placées dans une grille. L'objectif du joueur est de révéler toutes les cases où il n'y a pas de mine, soit de déminer le terrain. Pour l'aider dans sa réfléxion, une case révélée indique le nombre de mines dans les cases adjacentes, et le joueur peut placer un marqueur sur les cases qu'il pense minées. Ce projet vise à programmer ce jeu en java.

Description des fonctionnalités du programme :

Tout d'abord un menu permet de choisir le nombre de lignes, de colonnes et de mines que le joueur veut pour sa partie :

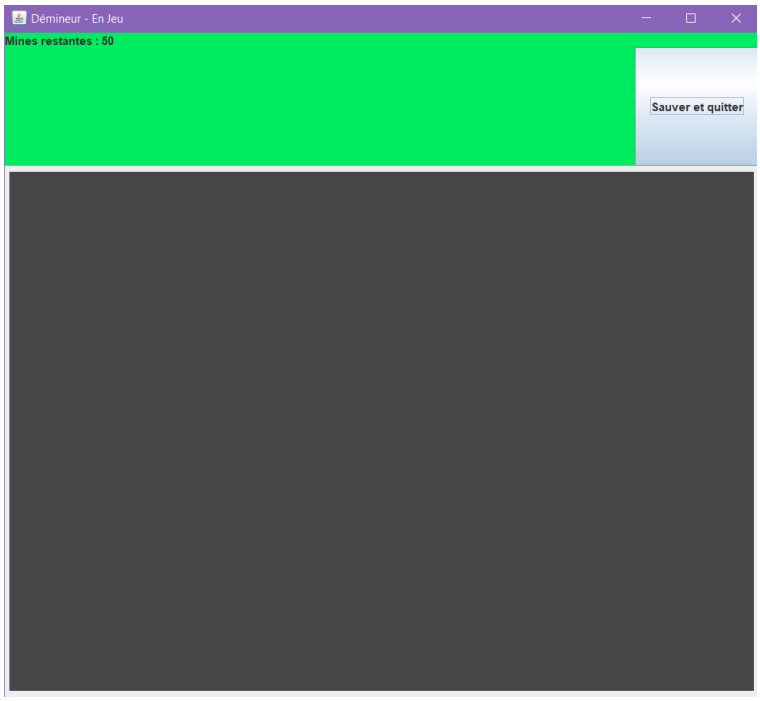


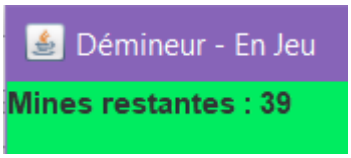
Les valeurs sont initialisées de façon à avoir une partie d'un niveau résonnable.

Lorsque le joueur change le nombre de lignes ou de colonnes, le nombre maximal de mines est modifié.

Le Bouton Quitter arrête le programme.

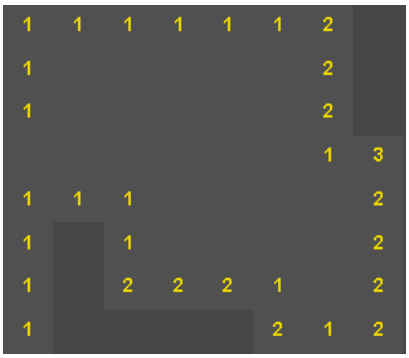
Lorsqu’on lance la partie, la grille de jeu apparaît :

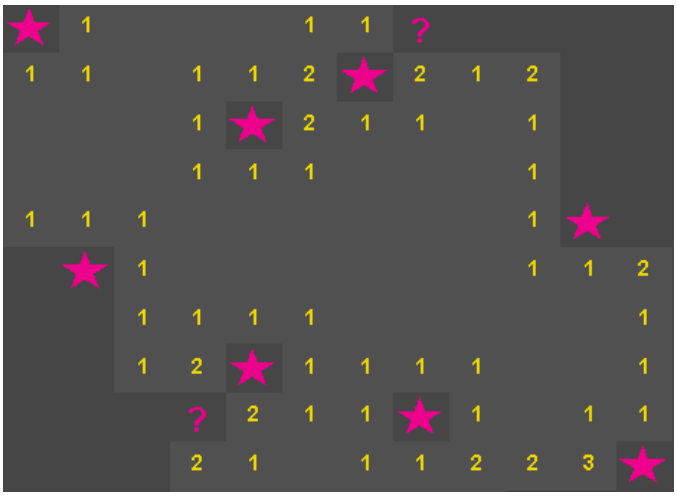




En haut, l’utilisateur peut voir le nombre de mines qu’il doit encore trouver.

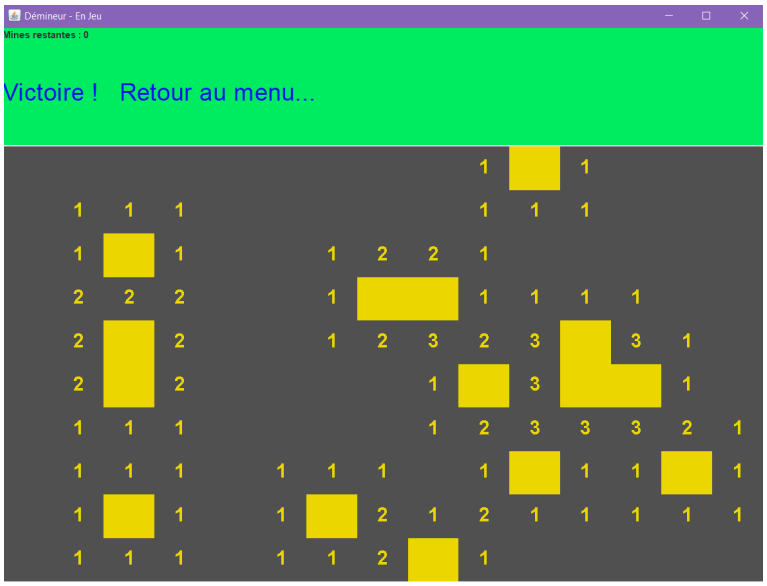
Le Joueur peut révéler une case avec un click gauche. Lorsque la case est révélée, soit elle est minée et l’utilisateur perd la partie, soit elle ne l’est pas et elle affiche le nombre de mines adjacentes.



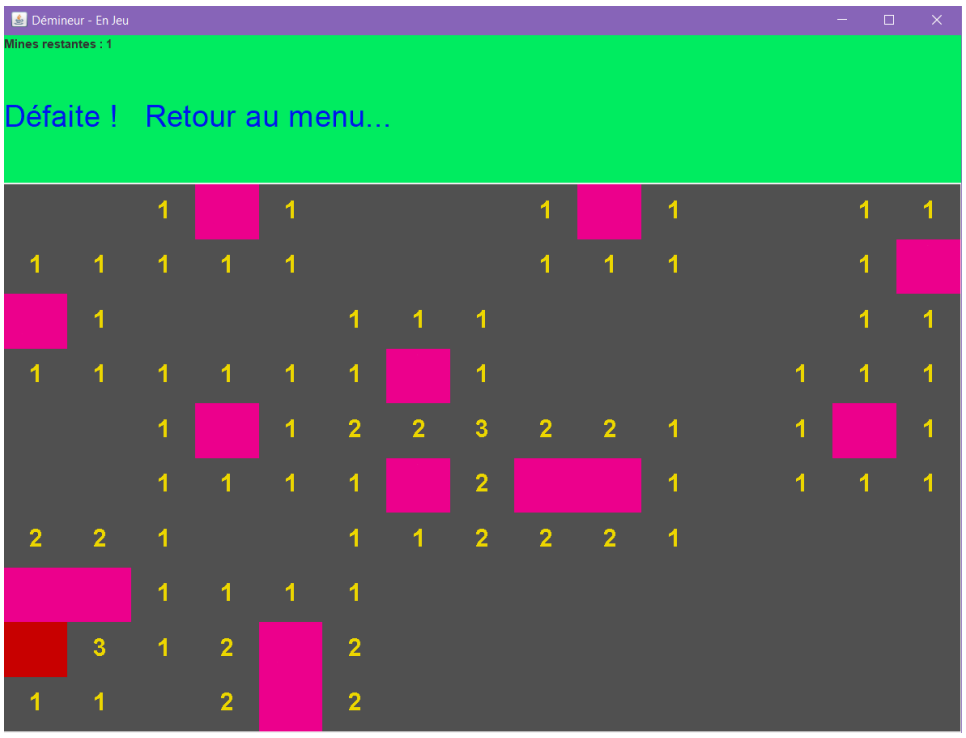


Il peut marquer d'une étoile une case qu'il croit minée avec un click droit. Cette étoile devient un point d'interogation, pour signifier un simple doute, avec un second click droit. Lorsqu'un marqueur est en place, le joueur ne peut pas révéler la case, afin d'éviter une défaite en cas de click non voulu. Un troisième click droit efface tout marquage et rend au joueur la possibilité de révéler la case.
Le bouton Sauver et Quitter permet de sauvegarder l'état de la partie et de quitter le programme.

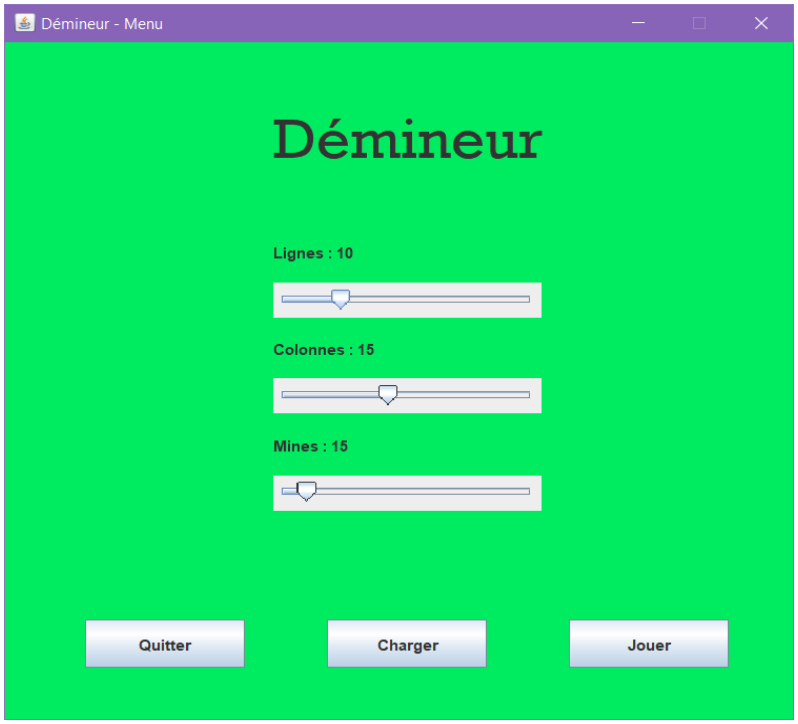
Lorsque toutes les cases non minées sont révélées, le joueur est notifié de sa victoire, les mines sont révélées, et le joueur est ramené au menu au bout de 7 secondes :



Si le joueur a cliqué sur une mine, il perd la partie. Il en est notifié et la grille est dévoilée. Les mines apparaissent et celle sur laquelle il a cliqué est peinte en rouge. Si des marqueurs sont encore présents sur la grille, cela signifie que la case n'était en réalité pas minée. Le joueur est également ramené au menu au bout de 7 secondes.



Si le joueur a une partie sauvegardée, le menu affiche un bouton Charger, qui permet de reprendre sa partie :



Présentation de la structure du programme :

Voici le diagramme de classes représentant la structure du programme :

Lorsque le programme est lancé (à l'aide de la commander make run), c'est le main qui est executé. Il appelle alors FrameMenu qui affiche une fenêtre de menu. Ce menu possède 3 Jsliders qui sont suivits par SettingsListener et permettent à l'utilisateur de choisir les paramètres (lignes, colonnes, nombre de mines). Il y a un bouton quitter pour pour arrêter le programme, un bouton Jouer pour lancer une nouvelle partie et un bouton Charger si un fichier de sauvegarde existe. Le bouton Jouer est suivi par un NewGameListener et lance une fenêtre de jeu. Le bouton charger en fait de même mais donne à la fenêtre un objet SaveData qui contient les données de sauvegarde.

La fenêtre de jeu génère un objet Grille, qui genère lui même une bannière, qui sont toutes deux affichées sur la fenêtre. Il y a également un bouton Sauver et Quitter qui enregistre les informations de la partie dans Save.txt grâce à SaveManager avant d'arrêter le programme.

La bannière contient le Bouton Sauver et Quitter et un compteur de mines restantes à trouver.

La grille genère un tableau d'objets Case, qui sont suivies pas un ListenerCase. Lorsque l'utilisateur clique sur une case, elle est révélée. Si une ou plusieurs mines sont autour de cette case, un JComponent Entourage affiche sur la case ce nombre de mines alentours. Si l'utilisateur décide de marquer la case, soit un JComponent Etoile, soit un JComponent Doute s'ajoutent sur la case.

Tout au long de la partie, les conditions de victoire ou de défaite sont vérifiées. Lorsque la partie se termine, la Bannière annonce la victoire ou la défaite. Un MenuListener avec un Timer permet de retourner automatiquement à la fenêtre de menu.

Sauvegarde d'une partie :

Il existe deux moyens de sauvegarder une partie : cliquer sur le bouton prévu à cet effet, ou fermer la fenêtre de jeu lorsque la partie est en cours (on ne peut pas enregistrer une partie terminée).
La sauvegarde va créer un objet SaveManager qui va écrire dans Save.txt les infos sur la partie : Le nombre de lignes, de colonnes, de mines, et un nombre compris entre 0 et 4 pour chaque case pour décrire son état (méthode ToString de Case). 0 si la case est cachée et non minée, 1 si la case est minée, 2 si elle est marquée, 3 si elle est minée et marquée, 4 si elle est révélee.

Afin de charger la partie, un bouton Charger apparaît sur le menu lorsqu'un fichier Save.txt est trouvé. Si ce bouton est cliqué, un objet SaveData de la classe SaveManager est créé. Il correspond aux données de sauvegarde lues dans Save.txt. Après lecture, le bouton Charger disparaît et le fichier de sauvegarde est supprimé. Un deuxième constructeur de FrameJeu est utilisé pour recupérer l'objet SaveData. Un deuxième contructeur de Grille récupère cette sauvegarde et regénère un plateau de Cases à partir des ces informations. La partie est de nouveau jouable.

Révélation de plusieurs cases :

Afin de faciliter le déroulement de la partie, si le joueur révèle une case sans mine autour d'elle, les 8 cases alentours sont révélées. Par récursivité, les cases seront dévoilées jusqu'à atteindre une case avec une ou des mines autour d'elle. Pour ce faire, une méthode de grille, appelé lors que la révélation d'une case, va accomplir diverses vérifications. Elle va lire le tableau de case jusqu'à trouver la case qui a appelé la méthode. Elle va ensuite vérifier que cette case ne soit pas à une extrémité du plateau (tout en bas, tout à gauche, en haut à droite…). Enfin, elle va révéler les cases révélables autour de la case ayant appelé la méthode.

En un click on peut donc obtenir ce genre de grille :

Conclusion :

J’ai trouvé ce projet plutôt long dans le temps, éprouvant sur la fin, mais très enrichissant. J’ai appis beaucoup et me suis familiarisée avec le java. Le projet est loin d’être parfait, mais il est fonctionnel. Je pense le continuer malgré que la date de rendu soit passée, afin d’avoir une meilleure interface et un code plus optimisé (notamment dans la classe Grille). J’ai beaucoup aimé travailler sur ce projet. C’est la première fois que je menais un projet seule.