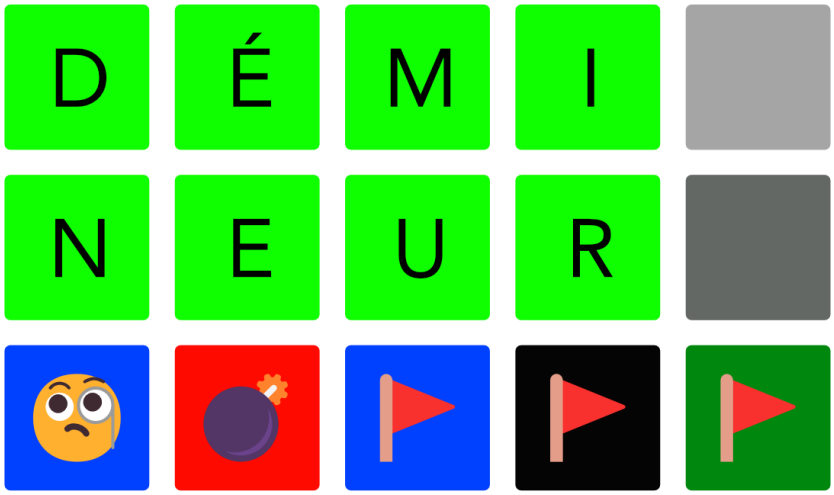
Rapport SAÉ: DÉMINEUR



Rapport

Lien vers le pdf: rapport.pdf

Sommaire du rapport

Membres

- Yvan FOUCHER (Groupe 5)

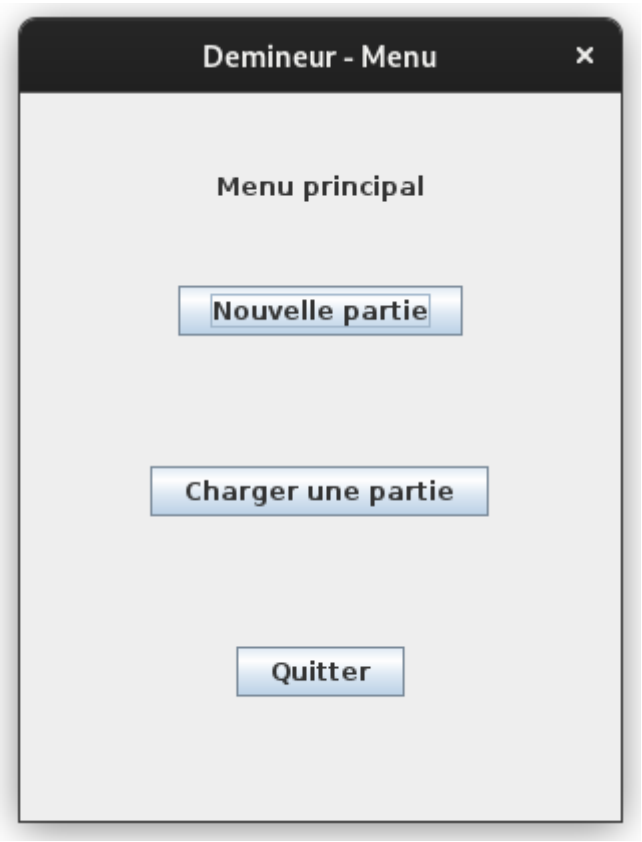
Introduction du sujet

Le but du projet est de coder le célèbre jeu du démineur en Java en ajoutant la possibilité de sauvegarder sa partie. L'utilisateur doit pouvoir signaler les bombes en utilisant des drapeaux (1 clic droit sur une case) ou dire quand il a un doute sur une case avec le smiley 🧐 (qui s'affiche après 2 clic droit sur une case).

Fonctionnalités du programme

Un magnifique menu

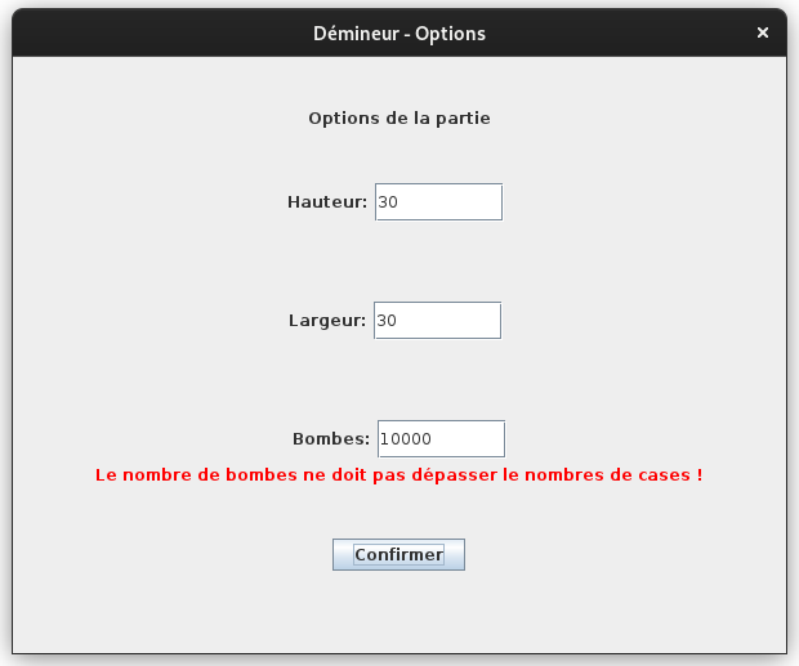
Au lancement du jeu, le joueur peut choisir de créer une nouvelle partie ou d'en charger une sauvegardée.



Paramètres de la partie

Après avoir choisit de créer une nouvelle partie, le joueur peut choisir le nombres de carreaux en hauteur, en largeur et le nombre de bombes sur la grille.

Un petit message d'erreur peut ~~l'insulter~~ le prévenir s'il a fait une erreur dans un des champs (comme par exemple: nombre de lignes, colonnes trop petit, ou bien le nombre trop élévé de bombes).



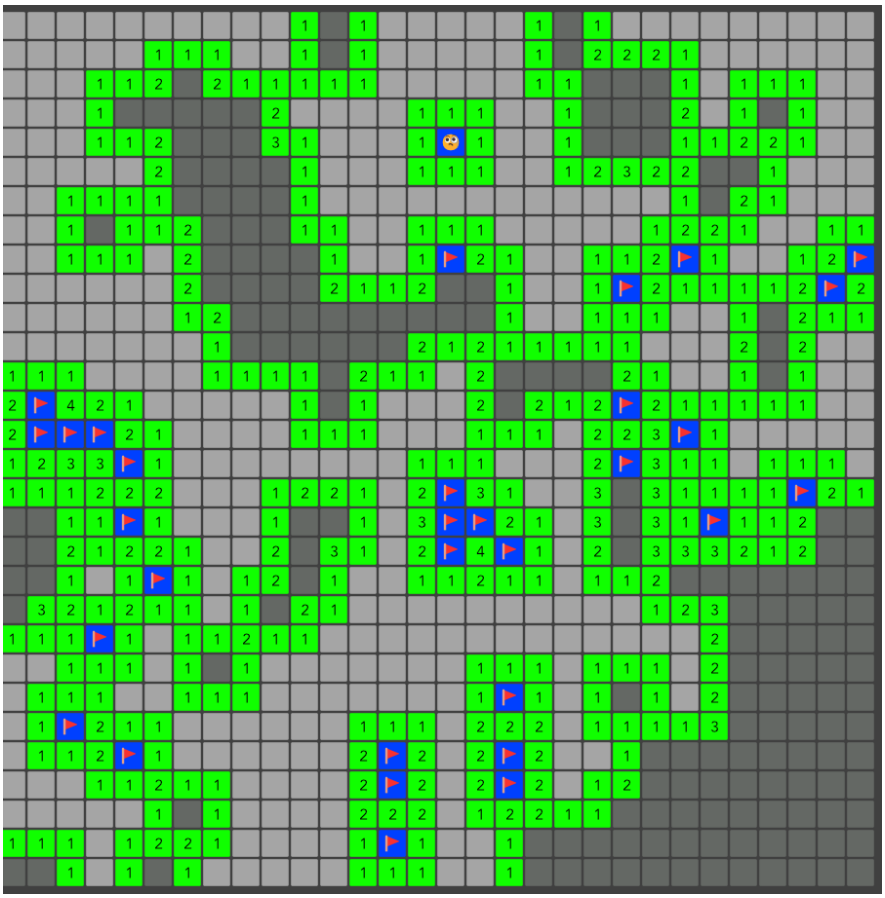
En partie

Les cases

Les cases sont des images de `30px` x `30px` désignées avec Affinity Designer. Les icones proviennet des émojis sous Windows 11.

Chaque icone a une signification différente.


Bombe non trouvée
Doute
Drapeau
Drapeau mal placé
Drapeau bien placé

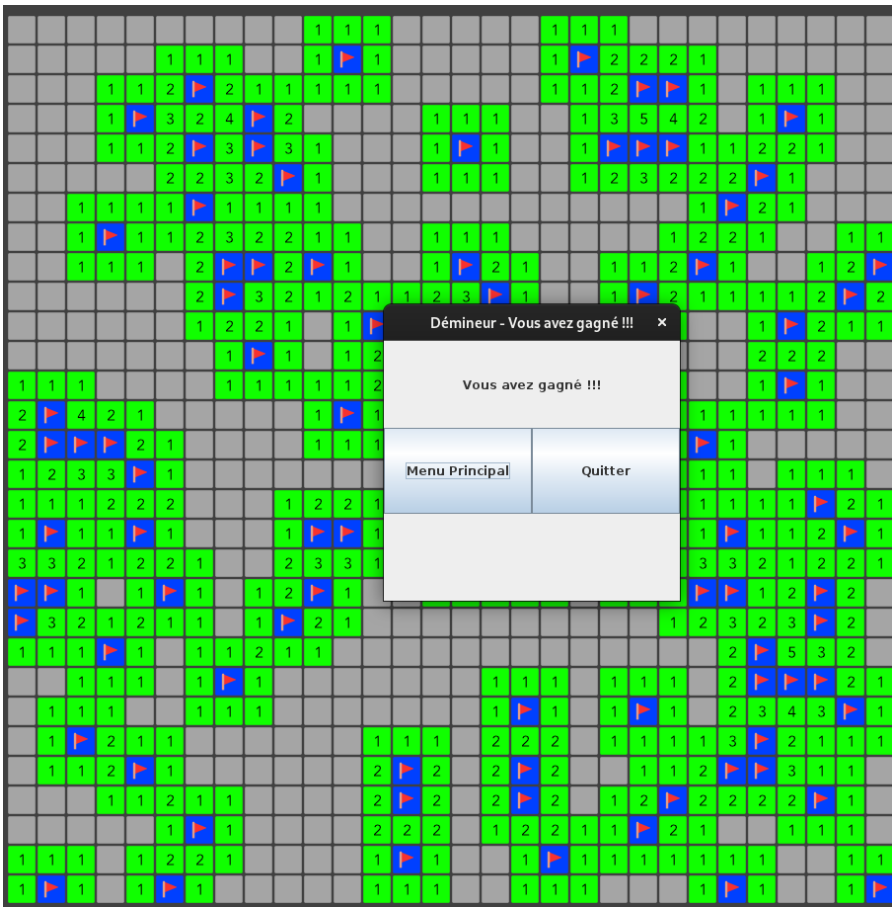


Le joueur peut sauvegarder et quitter la partie en plein milieu s'il le désire. La partie pourra être rechargée au prochain lancement du jeu.

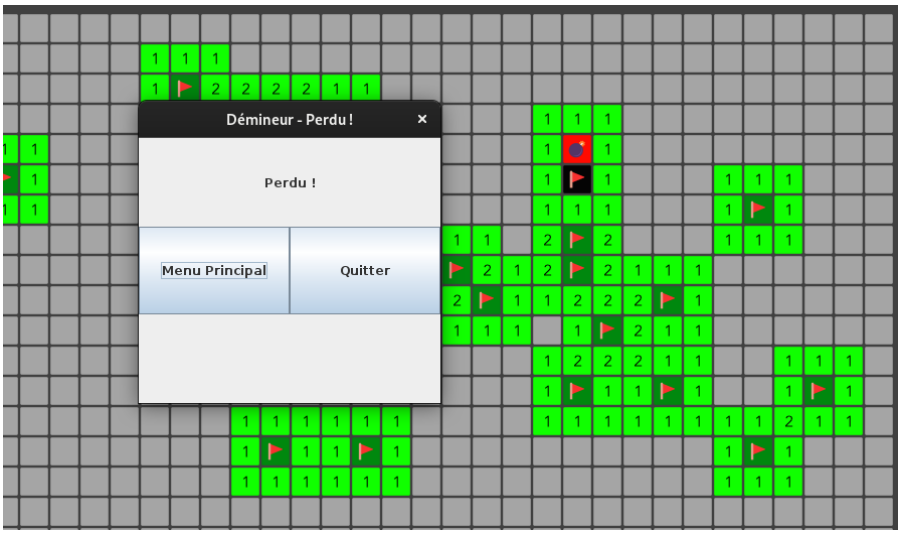
Victoire (ou défaite)

En cas de victoire un magnifique écran vient proposer au joueur de relancer une partie ou de quitter le jeu.

Le joueur peut tout de même contempler la grille en déplaçant le message.



En cas de défaite, l'utilisateur peut voir les drapeaux qu'il avait bien placé mais aussi les drapeaux mal placés et les bombes restantes.



Structure du programme

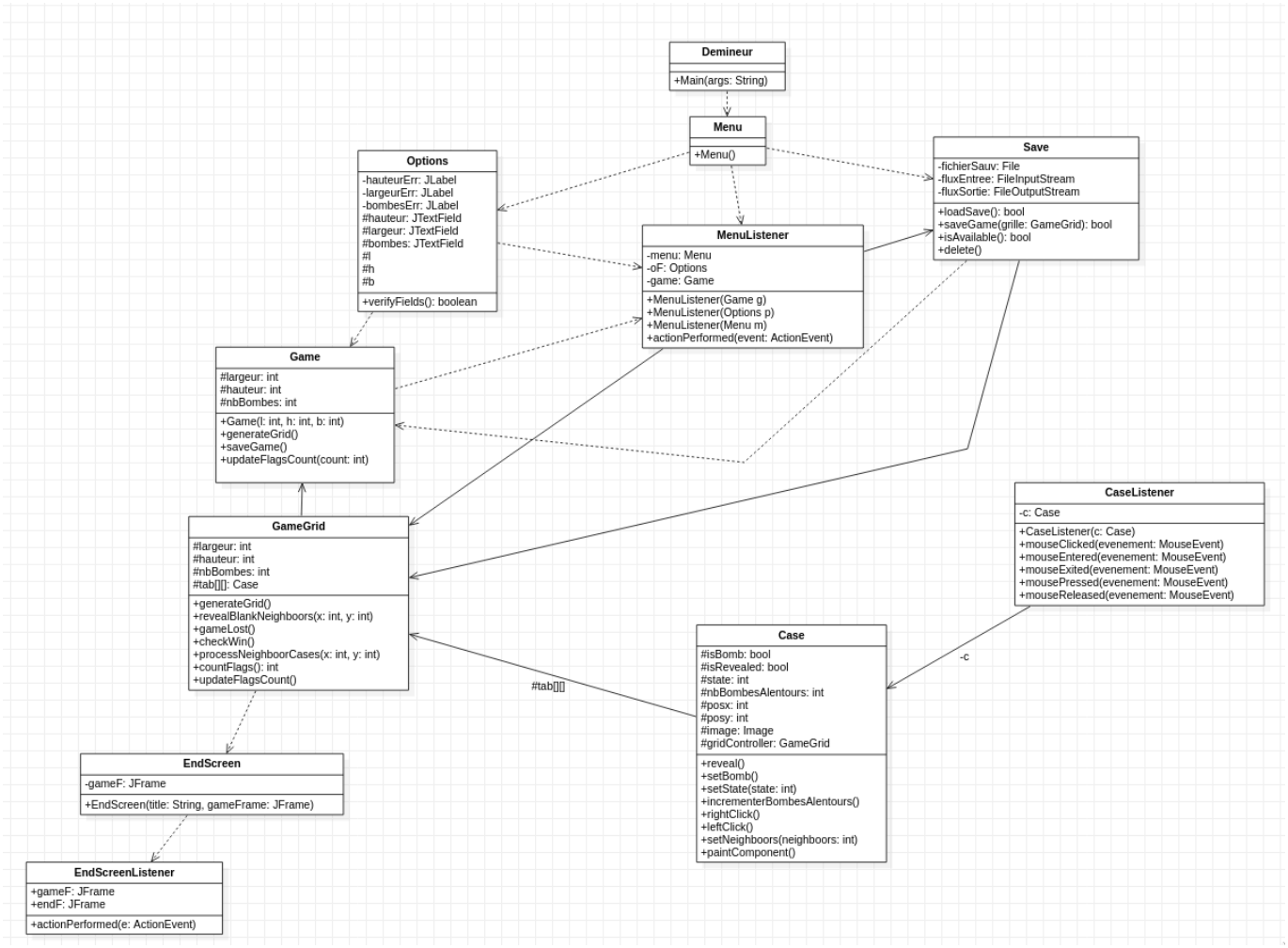
Le programme est reparti en plusieurs classes qui permettent d'effectuer les différentes actions tout au long d'une partie.

Globalement, il est reparti comme ceci:

- Demineur
- Options
- Jeu
- Grille
- Case

Ceci peut être vu plus en détails grâce au diagramme de classe.

Diagramme de classe



Sauvegarde

Le fonctionnement de la sauvegarde est plutôt simple.

Les informations de la grille sont stockées en entiers dans le fichier `sauv.dat`.

Tout d'abord dans le fichier, on vient commencer par stocker le nombre de lignes (hauteur), le nombre de colonnes (largeur), puis le nombre de bombes posées. Ce qui donnerait en décimal pour une grille de 12x12 avec 20 bombes.

```
12
12
20
```

Ensuite on vient parcourir le tableau avec une boucle en enregistrant dans le fichier pour chaque cases, les valeurs suivantes:

Bombe

Valeur	Description
1	La case est une bombe
0	La case n'est pas une bombe

Révélée

Valeur	Description
1	La case est révelée
0	La case n'a pas été révélée

L'état de la case

Valeur	Description
0	Le joueur n'a pas effectué d'action sur la case
1	Le joueur a signalé une bombe sur la case
2	Le joueur a un doute sur la case

Voisins

Ensuite on vient stocker le nombre de bombes voisines que la case possède.

Exemple de valeurs décimales pour une case

```
1
0
1
4
```

La case est une bombe, elle n'a pas été révélée mais le joueur a émis un doute sur cette case. La case est entourée par 4 bombes.

Algorithme qui permet de révéler plusieurs cases

Lorsqu'une case ne contient pas de numéro et n'est pas une bombe, vient être cliquée par l'utilisateur, on fait appel à la fonction `revealNonBombNeighboors` qui vient simuler un clic droit pour chaque case qui n'est pas une bombe. En cas de clic sur une case qui contient un nombre on ne fera pas appel à la fonction et la case sera quand même révélée.

Conclusion personnelle

J'ai bien aimé ce projet car c'est un jeu que j'affectionne particulièrement pendant les heures de projet à l'IUT (quand on a pas de projets bien sûr). La partie algorithmique m'a bien plu sur ce projet. Je m'étais lancé un défi pour voir en combien de temps je pouvais faire ce projet et en 13 heures la plus grande partie du projet était faite. En 20 heures le projet a pu être fini entièrement jusqu'à la sauvegarde des parties.

Idées d'améliorations

- Une amélioration possible serait de découper les différentes parties dans plusieurs sous-répertoires pour permettre une meilleure lisibilité. Cela serait possible avec les `package` mais pour l'instant ce n'est pas encore au programme (à voir si j'ai le temps de le faire un jour quand même).
- Refaire la partie gestion de la sauvegarde pour essayer le plus possible de faire appel à des fonctions pour pouvoir mieux refactoriser le code.
- Supprimer la sauvegarde après que la partie soit gagnée ou perdue.
- Ajouter un bouton pour relancer une partie rapidement