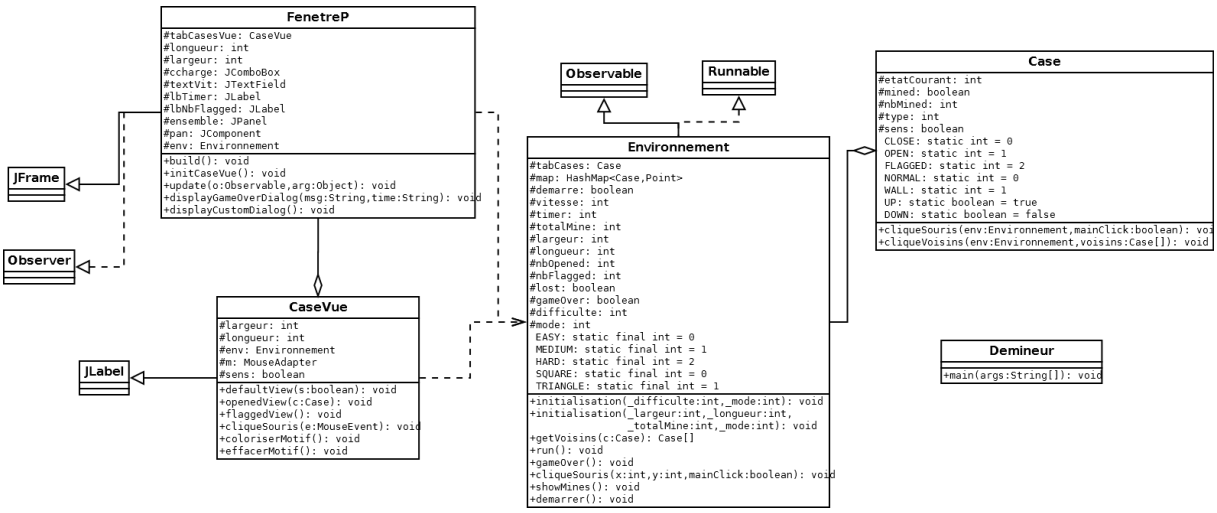
RAPPORT PROJET DÉMINEUR

SUJET :

Développer un jeu Démineur respectant les modèles MVC et Observer-Observable avec la librairie java Swing.

PRESENTATION GENERALE



LE MODELE

Le modèle est réparti sur 2 classes : Environnement et Case.

Environnement représente le modèle général. Il n'existe qu'une seule instance de cette classe lors de l'exécution. Cette classe rassemble le stockage de la grille mais aussi des données « méta » telle que : les dimensions du jeu, le nombre de mine, le nombre de case ouverte, la difficulté, le mode ou encore le temps de jeu. Elle abrite ainsi les processus métiers liés à l'initialisation de la grille, à la vérification de fin de partie mais aussi à la récupération des voisins de chaque case. Pour cela, les cases sont stockées dans un tableau à 2 dimensions et aussi dans une HashMap associant à chaque Case, un point (x, y). De plus cette classe hérite de la classe Observable, elle est en effet « surveillée » par la classe FenetreP de la vue. Ainsi elle exécute régulièrement les méthodes setChanged() et notifyObservers(). De plus elle implémente l'interface Runnable, sa

méthode run() se déroule ainsi : mettre à jour le temps de jeu, effectuer les vérifications de fin de partie, informer ses observers et attendre une seconde.

Case représente le modèle d'une case. Il existe autant d'instance de cette classe que de case affichées lors de l'exécution. Cette classe stocke l'état d'une case (fermée, ouverte, avec un drapeau), son type (normal ou mur), son sens (haut ou bas, utilisé pour le mode de jeu triangle), si elle est minée ou dans le cas contraire le nombre de ses voisines qui sont minées. Elle abrite le processus métier lié à l'interaction sur une mine (par exemple le clic à la souris) et la transmission aux cases voisines.

LA VUE

La vue est-elle répartie sur 2 classes : FenetreP et CaseVue.

FenetreP représente la vue générale. Il n'existe qu'une seule instance de cette classe lors de l'exécution. Cette classe, héritant de la classe Swing JFrame, correspond à la fenêtre principale de l'application. C'est ici que sont définis tous les éléments visuels de l'application : la barre de menu, les différents boutons, le panneau principal stockant la grille de CaseVue dans un tableau à deux dimensions. Implémentant l'interface Observer, la classe FenetreP référence l'instance Environnement et exécute la méthode update() qui réinitialise la grille en cas de changement de mode/difficulté ou de fin de partie, met à jour l'affichage des CaseVue ou initialise une boite de dialogue en cas de fin de partie.

CaseVue représente la vue d'une case. Pour chaque instance de la classe Case, il existe une instance CaseVue correspondante. Cette classe comprend simplement des méthodes définissant l'affichage en fonction de l'état de la case.

LE CONTRÔLEUR

Tout d'abord la classe Demineur joue seulement le rôle de point d'entrée de l'application avec sa méthode main. Elle se contente d'instancier un Environnement et une FenetreP et de démarrer le thread d'exécution de l'Environnement.

Ensuite le rôle du contrôleur est joué par des mouseAdapter et des mouseListener définis de façon anonyme dans le code des classes de la vue. Ils permettent de récupérer le clic sur chaque élément de la vue et d'appeler la méthode du modèle correspondante.

SPÉCIFICATIONS

Tout d'abord notre jeu implémente 3 modes de difficulté : facile, moyen et difficile qui modifient les dimensions et le nombre de mines, ainsi que 2 modes de jeu : carré et triangle qui modifient la forme des cases et de la grille.

MODE CARRE

- Facile : grille de 10*10 cases avec 10 mines
- Moyen : grille de 18*18 cases avec 40 mines
- Difficile : grille de 18*33 cases avec 99 mines

MODE TRIANGLE

- Facile : triangle isocèle avec 9 cases à la base et 5 cases de hauteur avec 5 mines
- Moyen : triangle isocèle avec 15 cases à la base et 8 cases de hauteur avec 10 mines
- Difficile : triangle isocèle avec 27 cases à la base et 14 cases de hauteur avec 15 mines

De plus le mode carré comprend une difficulté personnalisée où l'utilisateur peut choisir les dimensions de la grille et le nombre de mines.

L'utilisateur peut d'un clic gauche ouvrir une case fermée ou d'un clic droit placer un drapeau pour marquer la position d'une case. Un compteur affiche le nombre de drapeau placés sur le nombre total de mines.

Cliquer sur une case ouverte, ayant par exemple 2 cases voisines minées et dont 2 de ses voisines sont marquées d'un drapeau, permet d'ouvrir automatiquement les autres cases.

Ainsi, à l'instar du vrai jeu démineur, les drapeaux ne sont qu'un outil permettant simplement de marqué les case minées afin de ne pas les ouvrir et le joueur n'est nullement obligé de les utiliser.

La fin d'une partie a donc lieu si toutes les cases non minées sont ouvertes dans le cas d'une victoire ou si une case minée est ouverte dans le cas d'une défaite. Une fenêtre de dialogue affiche alors le temps de jeu et « victoire » ou « défaite ».

La répartition des mines se fait de façon aléatoire : tant que toutes les mines n'ont pas été placées, on parcourt la grille et pour chaque case on « jette un dé » ayant moins de 10% de chance d'y placer une mine. La répartition semble ainsi proche du vrai démineur et est différente à chaque partie.

DIFFICULTÉES RENCONTRÉES

Le mode de jeux en triangle n'est pas complet. Nous avons fait le modèle, c'est-à-dire la génération de la grille, précisant le sens du triangle pour chaque case (pointe vers le haut ou vers le bas) et la recherche des voisins permettant d'implémenter le mode triangle. Cependant nous n'avons pas réussi à implémenter une vue satisfaisante.

Nous avons réussi à dessiner un triangle au centre de chaque CaseVue. Nous avions ensuite l'idée de décaler les cases en modifiant les méthodes getX( )et getY() afin de superposer les cases,

permettant ainsi d'avoir une grille de triangle collés. Nous avons essayé d'hériter les classes JPanel ou encore Polygon afin de donner une forme triangulaire aux cases mais sans succès.

Nous avons donc laissé des traces de nos essais en commentaire dans le code et nous avons laissé la possibilité de jouer au mode triangle même si la représentation carrée des cases ne permet pas de visualiser les 12 voisines de chaque case.