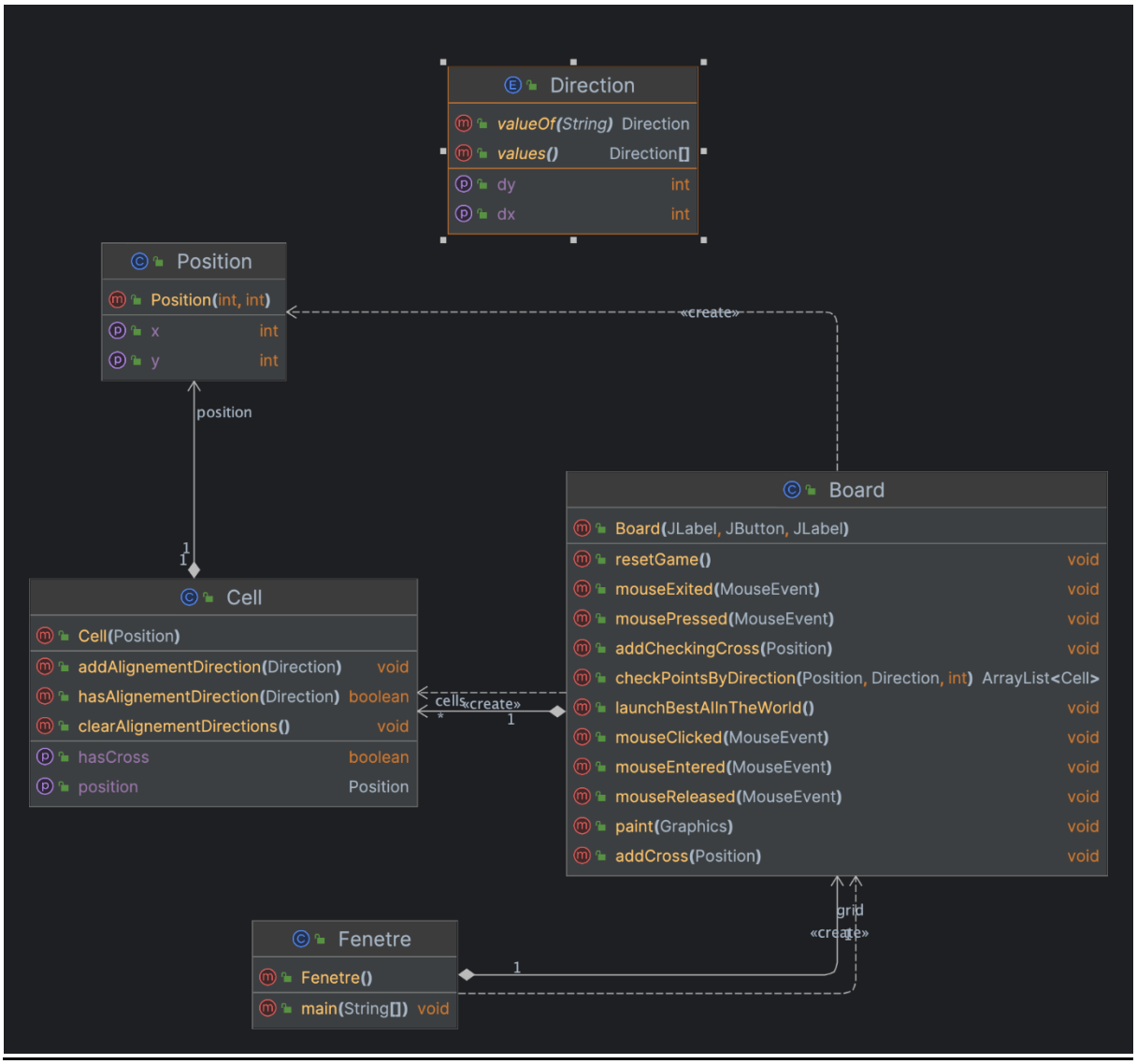
Bachelet Julie

Belbari Meriem

Documentation développeur

Architecture



La classe Fenêtre instancie un objet Board. Dans le constructeur de Board, un tableau à deux dimensions est créé, chaque élément du tableau étant une instance de la classe Cell. Un objet Cell contient des informations sur sa position dans le jeu. Lorsqu'un joueur clique sur le

jeu, les méthodes de Board sont utilisées pour vérifier si l'ajout d'un élément au jeu est possible en fonction de la position de la Cellule ciblée.

Construction du jeu

Création du plateau de jeu :

Pour créer notre plateau de jeu nous avons tout d'abord défini un nombre de lignes et colonnes que nous avons fixé à 25 qui nous ont permis par la suite de créer une grille. Chaque intersection de grille devient alors une cellule. Chaque cellule a des caractéristiques qui nous servirons par la suite à placer des points où des lignes. Chaque cellule contient un booléen qui nous indique si elle contient un point ou non. Pour les points initiaux de la grille, nous avons changé ce booléen en True à la main pour chaque coordonnées de cellule. Par la suite une boucle va permettre d'ajouter un point sur notre interface graphique grâce à la méthode fillOval, pour chaque cellule qui contient un True.

Et voilà ! Notre plateau est prêt, nous pouvons commencer à jouer. Enfin presque. Maintenant il faut ajouter des règles !

Ajout des points pour le joueur :

Lorsque l'on clique sur une intersection de grille, on aimerait positionner un point. Pour cela nous avons utilisé mouseClicked pour récupérer les coordonnées de la cellule choisie par l'utilisateur et éventuellement y ajouter un point. En effet dans ce jeu nous ne pouvons placer un point sur la grille seulement si celui-ci permet de faire un alignement de 5 points à l'horizontal, à la verticale ou en diagonale. Nous avons donc créé des contraintes qui nous permettent de définir si l'on peut ou non placer le point dans la cellule. Le principe de ces contraintes est de vérifier pour chaque direction (par ex à l'horizontale) et dans les deux sens (par ex gauche droite pour horizontale), si les cellules voisines ont un point ou non. Si un voisin est trouvé, on vérifie à son tour ses voisins mais cette fois ci seulement dans la direction de ce voisin. Si 4 voisins sont trouvés dans la même direction, alors la cellule

cliquée par l'utilisateur prend la valeur True pour l'indicateur de point. Pour finir, on refait tourner la fonction Paint qui va permettre de rajouter notre point à la grille.

Ajout des lignes :

Pour l'ajout des lignes, nous avons créé une ArrayList qui va comporter des listes de 5 points. Chaque liste correspondra à une ligne à tracer. Lorsque l'on clique sur une cellule et que l'on vérifie les cellules voisines, celles-ci sont ajoutées à une liste, ainsi que la cellule sur laquelle nous avons cliqué. Si la liste comporte 5 éléments, alors cette liste est ajouté dans la ArrayList. Une fois qu'une liste est ajouté dans la ArrayList, on utilise repaint() qui va appeler une nouvelle fois la méthode paint qui contient notre méthode pour ajouter des lignes sur la grille. Dans notre méthode pain, nous prenons chaque liste de la ArrrayList que nous allons trier, puis on récupère le premier et le dernier élément de la liste trier pour tracer notre ligne à l'aide de la méthode drawLine.

Notre jeu est enfin prêt pour jouer ! Il comporte toutes les fonctionnalités nécessaire pour le déroulement d'une partie. Pourtant certaines fonctionnalités supplémentaires seraient appréciables pour améliorer notre morpion solitaire, les voici :

Les fonctionnalités

Le bouton Recommencer :

Le bouton est créé à l'aide de JButton et permet de nettoyer complètement le plateau de jeu en ne laissant que la grille et les points de départ. Pour cela nous passons à False les indicateurs de points et de ligne pour chaque cellule de la grille, puis nous rajoutons les points de départ comme précédemment. Pour finir on appelle repaint() pour redessiner notre interface graphique avec nos nouvelles données. Nous passons aussi le score à 0. Et voilà, prêt à recommencer une nouvelle partie !

Le score et le bestscore :

Sur notre interface graphique nous avons affiché le score du joueur. Il est initialisé à 0 et s'incrémente de 1 à chaque ligne tracée. Le score se réinitialise lorsqu'on clique sur le bouton recommencer. Le bestscore enregistre le score lorsque l'on clique sur recommencer, si celui si est supérieur au bestscore actuel.

Une génération de solution aléatoire automatique :

Pour notre solution automatique, nous avons créé une méthode qui va ajouter des points et des lignes automatiquement jusqu'à ce que ce ne soit plus possible. Pour cela nous avons tout d'abord initialisé une variable continueAI à true. Tant que continueAI = true, une boucle permet de vérifier pour chaque cellule, s'il est possible ou non d'y placer un point en fonction des contraintes. S'il est possible d'ajouter un point, alors on place cette cellule dans une liste et on continue la boucle pour trouver tous les points possibles. Si la liste n'est pas vide, alors on prend au hasard une cellule de la liste, on lui met associe un point et une ligne et on recommence à chercher de nouveaux points. Une fois que la liste est vide, continueAI passe à false et la boucle s'arrête.

Les fonctionnalités non explorées

Bien que notre jeu soit parfaitement fonctionnel, nous aurions aimé rajouter quelques fonctionnalités après réflexion comme par exemple :

- Un bouton « cancel » qui permettrait d'annuler le dernier point ajouté
- La recherche NMCS
- Multithreading
- Une interface plus élaborée

Organisation du travail

Pour ce projet nous n'avons pas réparti le travail entre nous avec différentes taches à faire à chaque fois, nous avons plutôt organisé des sessions de travail en vrai où en vision pour travailler et avancer ensemble. Nous avons en général travaillé sur une machine en réfléchissant ensemble. Nous avons réalisé notre projet sur Eclipse et nous n'avons pas tout de suite créé de répertoire sur github puisque nous avancions toujours ensemble. Nous l'avons donc créé à la fin de notre projet, puis nous nous somme repartis le travail pour réaliser la javadoc, la documentation utilisateur et la documentation développeur.

Les difficultés rencontrées

Tout d'abord nous avons eu du mal comprendre en détail les règles du jeu et nous avions commencé l'élaboration du projet avec les mauvaises conditions. Nous avons dû revenir dessus plus tard lorsque nous nous en sommes rendu compte. Ensuite nous n'avion jamais réalisé d'interface graphique. Nous avons du apprendre avec ce projet comment faire et nous avons découvert de nombreuses fonctionnalités. Pour finir nous avons passé beaucoup de temps sur l'élaboration de la grille et le placement des points, nous avons rencontré un problème au niveau des coordonnés et de la taille de la fenêtre.