

PROGRAMMATION OBJET TETRIS

Glenn PLOUHINEC
Mamadou DIALLO
GROUPE 303E
RAPPORT PROJET 1

1- Introduction :

L'objectif de ce projet est de réaliser un jeu Tetris en programmation objet JAVA. Tetris étant un jeu d'arcade dont le principe est d'aligner le maximum de pièce qui descendent afin de former des lignes horizontales pleines qui seront ensuite supprimées , Le joueur ne peut stopper les pièces qui descendent mais peut par contre les déplacer vers la gauche, la droite, ou accélérer le déplacement vers le bas ou encore faire pivoter les pièces. Une partie est perdue lorsqu'une nouvelle pièce ne peut être ajoutée dans la grille.

Afin de réaliser le jeu , nous devons programmer un ensemble de classe (Cellule ,Pièce ,Fabrique et Plateau) qui sont imposées par le sujet pour décomposer les différentes fonctionnalités du jeu .

2- Description Synthétique des Classes

Cellule

1-Objectif

Une cellule est comme une case composant le plateau de jeu, et les pièces contrôlées par le joueur. Nous lui avons attribué des coordonnées réelles x et y, ainsi qu'une couleur col définie simplement comme caractère. On peut alors considérer une cellule comme un type de point dans un repère orthonormé.

2-Interface

-Méthode getX()

Cette méthode permet de retourner l'abscisse de notre cellule.

-Méthode getY()

Cette méthode permet de retourner l'ordonnée de notre cellule.

-Méthode getC()

Cette méthode permet de retourner la couleur de notre cellule.

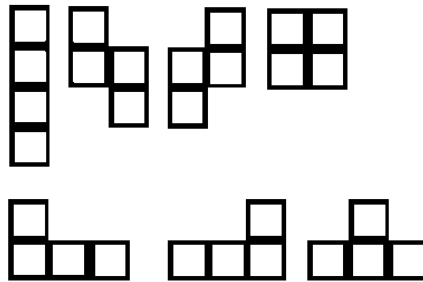
3-Structure De Données

Pour réaliser notre classe Cellule nous avons utilisé la structure d'un point c'est à dire avec deux coordonnées x (abscisse) et y (ordonnée) et en plus de cela nous avons rajoutée une couleur a ce point.

Piece

1-Objectif

Une pièce est simplement un ensemble de 4 cellules que le joueur peut déplacer pour former la notion de puzzle qu'est le principe du jeu. Il existe 7 types de pièces différentes, chaque pièce possède une couleur unique qui la caractérise



source : <http://mchammerbro.blogspot.fr/2012/03/classic-game-of-month-march-2012.html>

2-Interface

-Méthode getCelluleUn()

Cette méthode permet de retourner notre première cellule.

-Méthode getCelluleDeux()

Cette méthode permet de retourner notre deuxième cellule.

-Méthode getCelluleTrois()

Cette méthode permet de retourner notre troisième cellule.

-Méthode getCelluleQuatre()

Cette méthode permet de retourner notre quatrième cellule.

-Méthode versLeBas()

Cette méthode permet de retourner une pièce correspondant au déplacement vers le bas pour ce faire on conserve l'abscisse de toutes les cellules et on ajoute +1 aux ordonnées des 4 cellules.

-Méthode versLaDroite()

Cette méthode permet de retourner une pièce correspondant au déplacement vers la droite pour ce faire on conserve l'ordonnée de toutes les cellules et on ajoute +1 aux abscisses des 4 cellules.

-Méthode versLaGauche()

Cette méthode permet de retourner une pièce correspondant au déplacement vers la gauche pour ce faire on conserve l'ordonnée de toutes les cellules et on ajoute -1 aux abscisses des 4 cellules.

3-Structure De Données

Pour réaliser notre classe Pièce , nous n'avons pas utilisé de structure particulière , nous avons juste utilisé quatre attributs et chaque attribut stock une cellule qui n'est rien d'autre qu'un point et une couleur.

Fabrique

1-Objectif

Une Fabrique crée les différentes pièces énoncées plus tôt, de façon aléatoire dans l'interface du jeu. C'est-à-dire qu'un choix aléatoire est fait pour générer l'une des 7 pièces dans l'interface graphique.

2-Interface

-Méthode creerPiece()

Cette méthode permet de retourner une pièce au hasard , pour ce faire , nous prenons un chiffre au hasard en 1 et 7 et en fonction de ce chiffre nous retournons la pièce qui convient .

-Méthode creerL()

Cette méthode crée la pièce L qui peut apparaître n'importe où en haut du plateau.

-Méthode creerLInverse()

Cette méthode crée la pièce L inversé qui peut apparaître n'importe où en haut du plateau.

-Méthode creerBiais()

Cette méthode crée la pièce Biais qui peut apparaître n'importe où en haut du plateau.

-Méthode creerBiaisInverse()

Cette méthode crée la pièce Biais Inversé qui peut apparaître n'importe où en haut du plateau.

-Méthode creerTe()

Cette méthode crée la pièce Té qui peut apparaître n'importe où en haut du plateau.

-Méthode creerBatonVertical()

Cette méthode crée la pièce Bâton Verticale qui peut apparaître n'importe où en haut du plateau.

-Méthode creerCarre()

Cette méthode crée la pièce Carre qui peut apparaître n'importe où en haut du plateau .

-Méthode rotationPiece(Piece p)

Cette méthode assure la rotation des pièce , pour ce faire dans un premier temps ,elle identifie la pièce ainsi que sa position et avec ces informations , nous retournons une pièce qui correspond à la pièce tournée.

3-Structure De Données

Pour réaliser notre classe Fabrique , nous n'avons pas de structure car cette classe à un constructeur vide.

Plateau

1-Objectif

Un Plateau est un ensemble de cellules (de couleur noires) qui composent le jeu, il est de forme rectangulaire et semblable à une grille, sur laquelle on peut placer des pièces.

2-Interface

Méthode accepter(Piece p)

Cette méthode teste si une pièce peut être accepté ou pas dans notre plateau , elle renvoie true (vrai) si la pièce peut être accepté.

Méthode ajouter(Piece p)

Cette méthode permet d'ajouter une pièce dans notre plateau pour ce faire elle récupère les coordonnées de la pièce et avec ces informations la pièce est ajouté dans le plateau.

Méthode retirer(Piece p)

Cette méthode permet de retirer une pièce du plateau pour ce faire , elle récupère les coordonnées de la pièce et transforme la pièce en pièce morte.

Méthode getCellule(int k, int l)

Cette méthode renvoie une cellule du plateau en fonction des paramètres rentrés.

Méthode supprimerLigne()

Cette méthode teste si une ligne de notre plateau est pleine et lorsque cela est vrai la ligne est alors supprimée et tout le bloc du dessus descendant d'un cran .

3-Structure De Données

La classe Plateau utilise une structure en forme de matrice 10x20 qui représente une grille de Cellule ou chaque cellule de la grille est soit vide ou occupé par les cellules d'une pièce.

Jeu

1-Objectif

La Classe nous Permet de réaliser notre interface graphique , il récupère toutes les couleurs des cellules de notre Plateau et les affiche.

2-Interface

Méthode jouer()

La méthode jouer est le coeur de notre Jeu car elle fait appelle a presque toute les méthode de notre projet c'est aussi elle qui permet le déplacement de la pièce vers le bas chaque 0.3 seconde.

Méthode update(Graphics g)

Cette méthode crée ou efface un graphique et ensuite remplit le graphique de couleur grace a la méthode paint().

Méthode paint()

La méthode paint() permet de convertir chaque cellule de notre plateau en petit rectangle de couleur , la couleur étant définie par le caractère de la cellule.

Méthode keyPressed()

Cette méthode convertit les touche directionnel de notre clavier en action de déplacement (gauche , bas et droite) ou haut pour la rotation.

3-Structure De Données

Pour cette classe nous n'avons pas de structure particulière mais par contre nous utilisons un graphe de la même taille que notre plateau que nous remplissons à l'aide de notre plateau.

3-Relation Entres Classes

Entre Cellule et Pièce

Une Pièce étant une instance de la classe Piece composé d'un ensemble de Cellules (4 Cellules) alors pour implémenter notre classe Piece , nous avons eu a utiliser la classe Cellule pour pouvoir créer chaque cellule de notre pièce .

Entre Piece et Fabrique

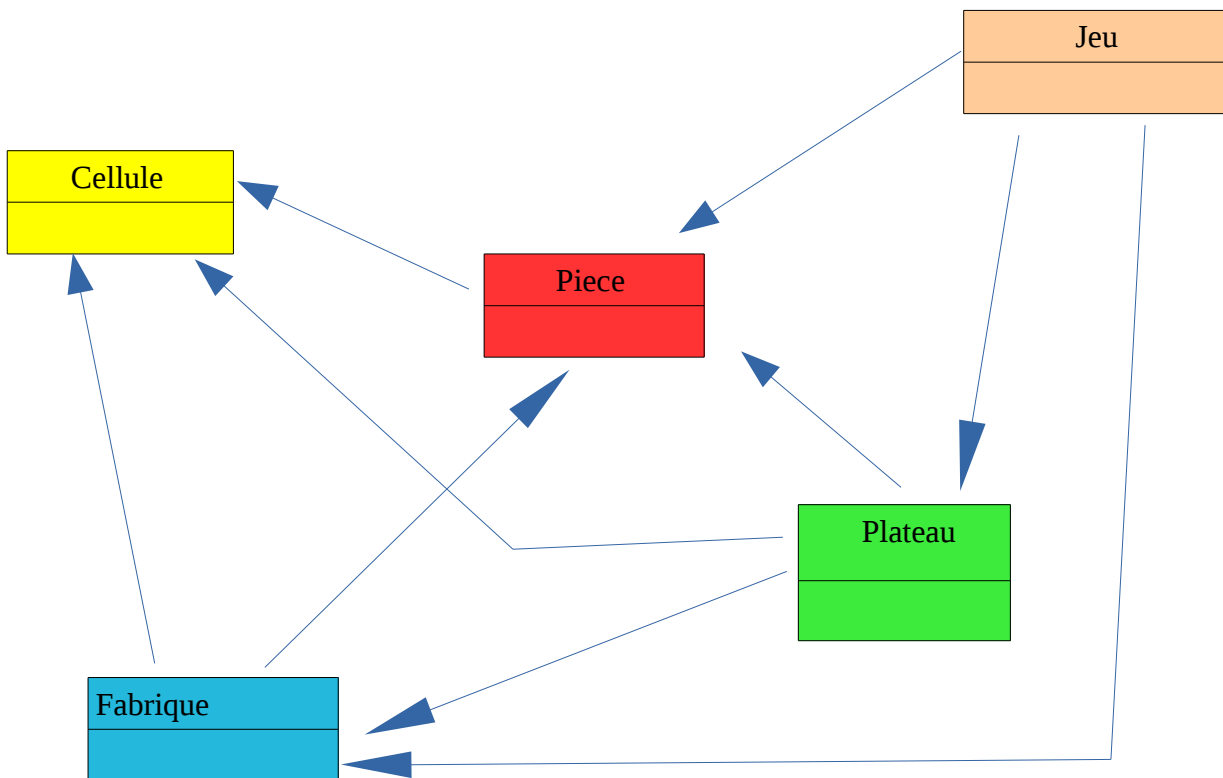
Une Fabrique étant une instance de la classe Fabrique permettant de créer les sept différentes pièces du jeu . Pour son implémentation , nous avons eu recours à la classe Piece .

Entre Plateau , Piece et Cellule

Notre Plateau étant une instance de la classe plateau représentant une grille contenant des pièces or vu qu'une pièce est composée de Cellule alors chaque petite case de notre grille sera de type cellule.

Entre Jeu , Plateau , Fabrique et Piece

Notre Jeu étant la classe principale de notre jeu , celle ci fait appelle à toutes les autres classe pour son fonctionnement.



4- Jeu de Test

Classe Cellules




Méthode	Justification	Résultat
getX()	abscisse	10
getY()	ordonnée	5.5
getC()	couleur	'b'

Classe Piece

Pièce p = new Piece(new Cellule(1,1,'a') ,new Cellule(1,2,'b') ,new Cellule(1,3,'c') ,new Cellule(1,4,'d'))

Méthode	Justification	Résultat
getCelluleUn()	Première cellule	Cellule(1,1,'a')
getCelluleDeux()	Deuxième cellule	Cellule(1,2,'b')
getCelluleTrois()	Troisième cellule	Cellule(1,3,'c')
getCelluleQuatre()	Quatrième cellule	Cellule(1,4,'d')
versLeBas()	Deplacement vers le bas , on augmente l'ordonnée	Piece(new Cellule(1,2,'a') , new Cellule(1,3,'b') , new Cellule(1,4,'c') , new Cellule(1,5,'d'))

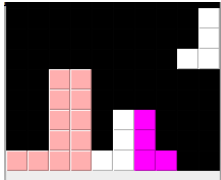
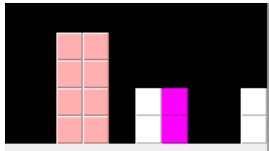
Classe Fabrique

Méthode	justification	Résultat
creerCarre()	Carre	
rotationPiece(p)	p = 	

Classe Plateau

Pièce p1 = new Piece(new Cellule(1,1,'a') ,new Cellule(1,2,'b') ,new Cellule(1,3,'c') ,new Cellule(1,4,'d'))

Pièce p2 = new Piece(new Cellule(-0.5,1,'a') ,new Cellule(1,2,'b') ,new Cellule(1,3,'c') ,new Cellule(1,4,'d'))

Méthode	Plateau	Résultat
accepter(p1)	$0 < 1 < 10$ et $4 < 20$	true
accepter(p2)	$-0.5 < 0$	false
supprimerLigne()	 La dernière ligne qui va se remplir	

5- Conclusion

Ce projet de jeu Tetris fut passionnant et très intéressant à réaliser . Ce projet nous a permis de passer en revue toutes les notions vu en cours et même d'appliquer d'autres notions qui n'ont pas été traitées en cours comme l'interface graphique , ce qui nous à permis à travers des recherches d'améliorer nos connaissances en programmation objet . Ce projet aussi étant le premier en programmation objet nous a permis de nous confronter aux difficultés de réalisation de la documentation et de l'organisation du code.

Pour les améliorations possible du jeu , on pourrait par exemple augmenter le nombre de pièce , on pourrait aussi ajouter dans donc fonctionnalités comme le niveau du jeu , aussi afficher le score dans la fenêtre graphique.