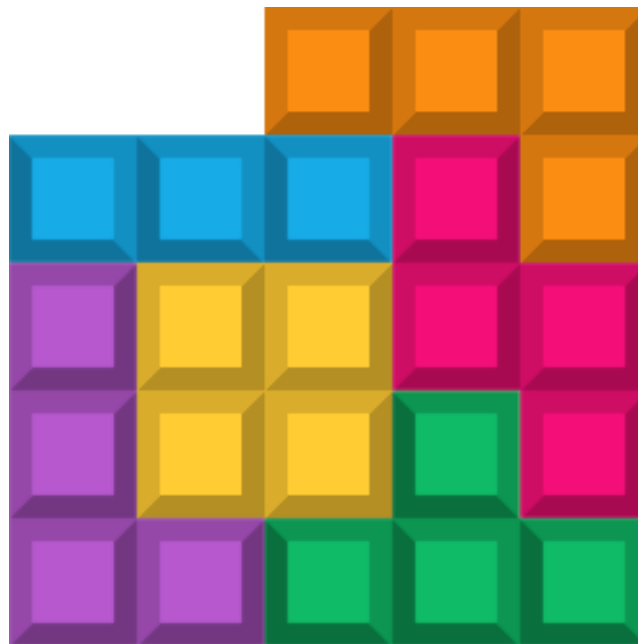


# Java Gaming Puzzle Stadium Ultimate

Projet de premier semestre en Méthode de conception



# Sommaire

|  |   |
|--|---|
| Patterns utilisés.....                                     | 3 |
| Bot.....   | 4 |
| 1. Les stratégies.....                                     | 4 |
| a. Explication de l'algorithme génétique (AgStrategy)..... | 4 |
| 2. Les mouvements.....                                     | 4 |
| 3. Utilisation réel.....                                   | 5 |
| 4. La classe Bot.....                                      | 5 |
| Le modèle.....   | 6 |
| 1. Les arrangements.....                                   | 6 |
| a. Arrangement.....  | 6 |
| b. ArrangementList.....                                    | 6 |
| 2. Le Playboard.....                                       | 6 |
| Comparaison par rapport à l'ancienne architecture.....     | 7 |

## Patterns utilisés

| Patterns | L'adapter       | Factory      | Composite | Command           |
|----------|-----------------|--------------|-----------|-------------------|
| Class    | ArrangementList | PieceFactory | Toutes    | Move<br>MoveToIHM |

| Patterns | Observer           | Strategy | TemplateMethod   | State |
|----------|--------------------|----------|------------------|-------|
| Class    | PlayBoard<br>Piece | Bot      | AbstractStrategy | Bot   |

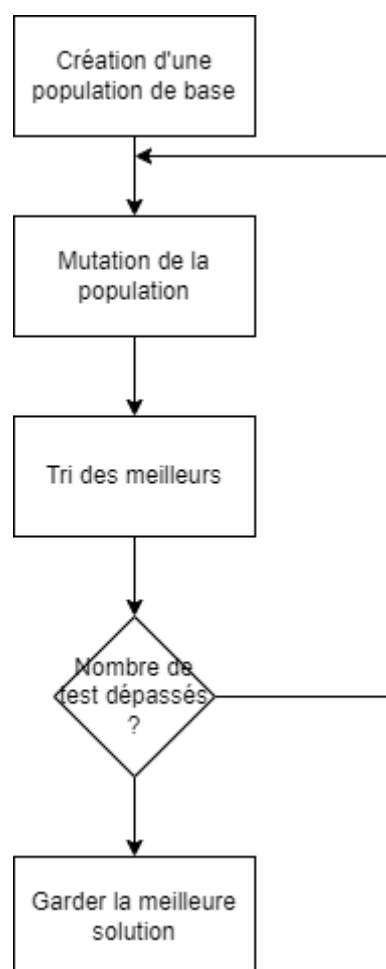
# Bot

## 1. Les stratégies

Il existe 2 stratégies :

- NoobStrategy qui consiste à créer des playboard random et garder le meilleur
- AgStrategy qui crée un nombre de playboard les fait muter et recommence ça un nombre de fois en ne gardant que les tant premiers à chaque itération

### a. Explication de l'algorithme génétique (AgStrategy)



## 2. Les mouvements

Pour stocker les mouvements on utilise la classe Move qui stocke l'id de la pièce, sa rotation et sa position nous permettant alors en faisant une liste de ces paramètres un playBoard complet avec une solution.

### **3. Utilisation réel**

Lorsque les bots sont utilisés ils sont mis dans un BotThread qui va les faire “Ticker” ( pour qu’ils puissent calculer) et lorsqu’ils ont fini de calculer le BotThread va alors lancer leurs IMapView respectable pour que celles-ci transmettent à la vue leurs mouvements.

### **4. La classe Bot**

La classe bot n’est qu’une encapsulation permettant de switcher entre les différentes stratégies et délègue les calculs à la stratégie en cours

# Le modèle

## 1. Les arrangements

### a. Arrangement

Chaque arrangement est un enregistrement d'une partie sous la forme

sizeX;sizeY;pieceCount;seed;username;score

Ex : 10;10;3;0;Shesh;10

### b. ArrangementList

La classe ArrangementList est un adaptateur permettant de transformer les arrangements en une JTable

## 2. Le Playboard

Créer un playboard avec une seed qui donne un résultat dépendant de la seed ce qui rend plus facile le stockage car un playboard = une seed, sa taille et son nombre de pièces

Permet de stocker les pièces sous forme d'un tableau d'id :

- 0 lorsqu'il n'y a pas de pièces
- un autre nombre lorsqu'une pièce occupe la case

## Comparaison par rapport à l'ancienne architecture

En sachant, dans le cas où l'application n'est pas à un but unique la méthode des Observer est plus pratique et plus modulable, mais dans le contexte où l'application ne peut être affichée qu'une seule fois notre méthode permet un accès restreint au modèle et permet de centraliser et clarifier les actions que la vue engendre.