Stratego

Sami Slim, 56081
Gabriel Espinosa, 54720

Table des matières

1.Introduction

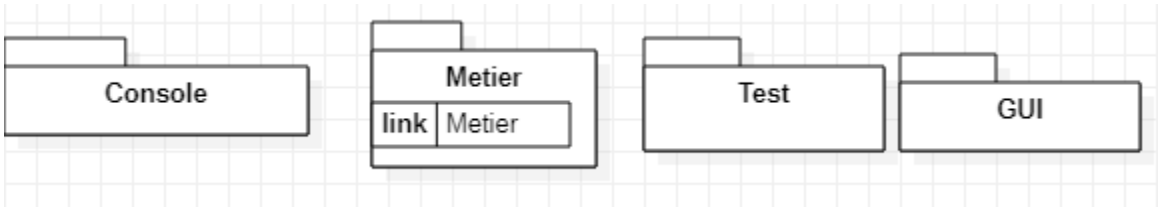
Ce projet organisé par les élèves Sami Slim et Gabriel Espinosa du groupe D111 pour la création du Stratego. Il a été encadrer avec les spécification abordé dans le document d'énoncé du projet[1] ainsi que les modalités[2] exigé par le professeur Marcelo Burda .

Le projet a été structuré et découpé en sous-système. Console : La version console du projet. Métier : Le model du projet, énumérant chaque élément visible et invisible du jeu. Test : Permet d'effectuer des tests dans le projet. GUI : Est la version graphique et interactif du projet.
Mais également implémenté les désignes patterns:
1.Modèle Vue Controller(MVC).
2.Observer observé, permet d'avoir une actualisation direct entre ce qui est fait graphiquement ainsi qu'avec les classes modèle en établissant une relation « d'inscription » entre elles.
3.Façade, isole le code dans une classe pour rendre propre et lisible.

[1] https://poesi.esi-bru.be/mod/resource/view.php?id=3023
[2] https://poesi.esi-bru.be/mod/resource/view.php?id=3069

2. Structure

La mise en place du découpage en séparant les package, permet d'apporter plus facilement la notion de Modèle Vue Controller et de réduire la dépendance entre les différente package.



a. Console

Permit d'afficher via la console de jouer une partie de Stratego et d'interagir.

b. Métier

Le model du Project. Qui contiendra pour l'instant toute les classes crées pour le modèle Métier.[3]

1) Entity

-Est la représentation d'un pion qui peut avoir un rôle définit dans l'énumération ROLE et peut être Flag,Spy,Bomb,etc.
-La méthode 'initEntity'qui permet d'initialisé l'attribut 'attack,distance,symbole' dépendant de son ROLE.
-L'attribut inGame , permet d'indiqué si il est dans la partie.
-L'attribut distance , est la représentation des mouvements possible du pion 0(immobile) à 10(Scout) .
-La méthode « dead() » fait 'mourir' le pion et change sont état « inGame ».

2) Case

La Case indique s'il contient un pion. Sinon son attribut est vide.

[3] DiagrammeClasse.svg

3) Board

-Son attribut est un tableau de Case auquel 3 état peuvent être possible.
1. La case d'un tableau fait référence à une' Case 'ayant une entité.
2. La case d'un tableau fait référence à une' Case' vide.
3.La case d'un tableau fait référence au vide (si c'est une case troue).
-La méthode 'move' permet de déplacé l'entité d'un endroit donné à un autre.
-La méthode 'destroy' enlevé l'entité à la position donnée.

4) Game

Initialise le jeu, donne quel joueur est courant ainsi que l'état du jeu.
-La méthode 'isEnd' vérifie si le jeu est finis(Uniquement si STATE est TURN).
- La méthode 'put' permet de mettre une entité dans une position donnée en début de partie (uniquement si STATE est START).
-Méthode 'turnPlayer' change le joueur courant(Uniquement si STATE est TURN).
-Méthode 'moveTo' permet de déplacer l'entité dans une position donnée(Uniquement si STATE est TURN).

5) Facade

Initialise la classe Game et lui sert de façade.

6) ROLE

L'énumération qui donne le rôle de chaque pion ainsi que son nombre total par joueur permet de contrôler la limitation de rôle.

c. Test

Les tests effectuer dans le projet pour le bon fonctionnement de celui-ci.

d. GUI

L'implémentation graphique du projet auquel l'utilisateur effectué des interactions hors commande.