Architecture

Nous avons décidé de découper notre projet en plusieurs packages pour mieux organiser les fonctionnalités.

1. Package card

Ce package regroupe toutes les cartes du jeu, à savoir :

• Les cartes décompte pour chaque animal.

• Les cartes variantes.

• Les cartes servant au calcul des scores pour les paysages.

Toutes ces cartes sont représentées par des records. Nous avons également mis en place une interface Card qui regroupe ces cartes. Cette interface contient une méthode commune, counterScore(), qui permet de calculer le score de chaque carte et de l'ajouter au score du joueur.

Nous avons également une classe ManagementCard, qui est une ArrayList<Card>. Cette classe regroupe toutes les cartes de score utilisées lors d'une partie.

2. Package model

Ce package stocke tous les records liés aux données du jeu. On y trouve :

• Les enums Landscape, Animal et Angle.

• Le record Pos, qui représente une position avec un couple (x, y).

L'enum Angle représente l'orientation d'un paysage. Il y a six valeurs possibles pour un angle :

• Ouest, Est, Nord-Ouest, Nord-Est, Sud-Est et Sud-Ouest.

Nous avons également ajouté une valeur spéciale, Angle.NULL, pour les tuiles à un seul paysage et les tuiles carrées. Cela nous permet d'éviter de stocker des valeurs null dans la structure HashMap<Landscape, Angle> de notre classe Tile.

3. Package board

Ce package regroupe toutes les fonctionnalités liées au plateau de jeu. Il contient :

• La classe Board, qui représente le plateau de jeu sous forme d'une structure HashMap<Pos, Tile>.

• La classe Tile, qui contient :

  • Une ArrayList<Animal> pour stocker les animaux présents sur la tuile.

  • Une HashMap<Landscape, Angle> pour stocker les paysages de la tuile.

  • Un objet FaunaToken pour représenter un éventuel jeton faune présent sur la tuile.

De plus, ce package contient le record FaunaToken, qui représente un jeton faune.

4. Package player

Ce package regroupe toutes les fonctionnalités liées aux joueurs :

• La classe Player, qui contient :

  • Un nom (représenté par un String).

  • Un score (représenté par un int).

  • Un environnement (représenté par un Board).

• La classe ManagementPlayer, qui contient une ArrayList<Player>. Cette liste regroupe tous les joueurs d'une partie, ce qui permet d'ajouter autant de joueurs que souhaité.

• La classe Success, qui représente les succès associés à chaque joueur. Cependant, cette fonctionnalité est encore incomplète (seule une partie a été implémentée).

5. Package graphicdisplay

Ce package contient les classes :

- Display, qui regroupe toutes les fonctions liées à l'affichage.

- Click, qui regroupe toutes les fonctions liées au click de la souris

-ManageGraphic, qui regroupe toute les fonction qui permettent le fonctionnement du mode graphique

6. Package game

Ce package contient les fonctionnalités permettant d'initialiser et de gérer le jeu. On y trouve :

• La classe Game, qui prend en paramètres deux listes :

  • Une liste allTiles contenant les 100 tuiles utilisées dans le jeu.

  • Une liste partTiles contenant uniquement les tuiles utilisées pour une partie (le nombre varie en fonction du nombre de joueurs).

  • Une ArrayList<FaunaToken> représentant tous les jetons faune disponibles au début du jeu.

  • Deux tableaux, choiceTiles et choiceFaunaToken, qui représentent les 4 choix proposés aux joueurs à chaque tour.

  • Un boolean isTileSquare, qui détermine le type des tuiles (carrées ou non) à l'initialisation du jeu.

• La classe Save, qui permet de sauvegarder et de charger une partie via un fichier texte. Cette classe prend en paramètres :

  • ManagementPlayers players, qui représente tous les joueurs d'une partie.

  • ManagementCards cards, qui regroupe les cartes utilisées.

  • Un entier round représentant le nombre de tours joués.

  • Un boolean isTileSquare pour indiquer le type des tuiles.

Dans cette classe, différentes méthodes permettent d'écrire les données de la partie dans un fichier texte, ainsi que de les récupérer pour restaurer une partie.

• La classe Interaction, qui permet d’interagir avec le joueur et de récupérer les valeurs qu’il rentre, cette classe prend en paramètre :

  • BufferedReader reader, qui permet de récupérer ce que rentre le joueur

Améliorations depuis la soutenance Bêta

Plusieurs améliorations ont été apportées depuis la soutenance bêta :

• Les paysages (Landscape) et animaux (Animal), qui étaient auparavant représentés par des String, sont maintenant des enums.

• Le tableau utilisé pour représenter le plateau de jeu, initialement de taille fixe, a été modifié pour devenir dynamique. Il s’agrandit automatiquement au fur et à mesure que les joueurs placent des tuiles.

Difficultés Rencontrées

Nous avons rencontré plusieurs difficultés, surtout au niveau de la gestion du zen et de l'affichage graphique, afin que nos tuiles restent bien centrées au milieu de l'écran. De plus, nous avons rencontré quelques problèmes au niveau des cartes de décompte, notamment celles de la buse.