Choix d'implémentation

I) Phase 1

A) Organisation des fichiers du jeu

Les fichiers sources sont contenus dans un package général nommé game. Ce dernier contient des sous-packages qui structurent et organisent le jeu.

- game/display : contient les fichiers sources (AnimalColor.java, Display.java et DisplayTools.java) qui permettent l'affichage des tuiles et jetons colorés ou encore le choix des 4 tuiles et jetons.
- game/logic : contient les fichiers sources (Choice.java, GameLogic.java, Position.java, Scores.java) qui implémentent la logique du jeu : choix des tuiles et jeton pose des tuiles en fonction d'une tuile adjacente, interaction utilisateur.
- game/ main : Le fichier source Main.java qui inclut le programme principal du jeu.
- game/material : contient l'implémentation du matériel du jeu, tuiles, et environnement du joueur (composé de jetons et tuiles placés).
- game/player : ce package implémente les données d'une personne pour en faire un joueur.

Un fichier csv (Tuiles.csv) comprend toutes les tuiles du jeu.

B) Structure des éléments du jeu

Le matériel du jeu Cascadia a été implémenté de la manière suivante :

- Pour les tuiles, l'objet Tile les caractérise. Il englobe les animaux possibles (sous forme de liste) et l'unique place possible représenté de chaque par deux lettres. Pour l'affichage terminal les tuiles sont affichées en ASCII art.
- Tuiles de départ : HashMap qui implémentent des tuiles sans jetons dessus et LinkedHashMap,Position> qui permet de poser une tuile de départ à des positions précises (0,0) (0,1) et (0,2) . Cette même structure est utilisée pour implémenter l'environnement du joueur.
- Tuiles du jeu (basiques) : ArrayList comprend la liste des tuiles contenues dans le fichier csv.
- Dans le cas des jetons, le record token est présent, il représente les jetons avec le nom de l'animal et sa couleur. Dès lors qu'un jeton est posé dans l'environnement , la couleur en fonction de l'animal est mise en avant.
- Le sac du jeu pour mélanger les tuiles et jetons est représenté par la méthode shuffle de la classe Collections qui mélange les listes tuiles et jetons avant les choix des joueurs.
- L'environnement du joueur est représenté par cette structure LinkedHashMap,Position> qui permet de garantir l'ordre d'insertion des tuiles et jetons avec leur position dans la grille.
- En ce qui concerne la grille, elle est implémentée par une liste de liste de String
- Le calcul de score se fait par parcours en profondeur avec la recherche des groupes d'animaux avec stockage des positions déjà visitées.

C) Difficultés rencontrées

La conception de la première partie du projet s'est heurtée à plusieurs problèmes :

- L'affichage du jeu : Problèmes pour l'affichage des tuiles et coloration des jetons.
- Exploitation du fichier csv : Différents chemins de recherche du fichier pour l'exécution normale et celle du fichier jar.
- Conversions données : Des variables se partagent les mêmes données créant des environnements pour tous les joueurs.

II) Phase 2

A) Organisation des fichiers du jeu

L'organisation des fichiers pour la phase 2 est la même que pour la phase 1 avec quelques ajouts pour implémenter la partie graphique.

- game/color : contient les fichiers sources (AnimalColor.java, GraphicAnimalColor.java) qui permettent l'affichage des jetons colorés (ligne de commande et version graphique).
- Game/graphic : contient les fichiers sources (DataGame.java, Fonts.java, GraphicGame.java, GraphicTileToken.java, Menu.java, WindowInfo.java) qui permettent l'implémentation graphique du jeu.

Le reste des sous-package ne diffère pas de la phase 1.

B) Structure des éléments graphique du jeu

La partie graphique, implémentée avec la bibliothèque d'interface graphique Zen, est structurée de la manière suivante :

- DataGame.java utilisé pour obtenir les données de la partie nécessaire à l'implémentation graphique.
- Fonts.java utilisé pour gérer les polices d'écriture utilisées sur les différentes fenêtres graphiques.
- GraphicGame.java utilisé pour gérer la logique derrière les affichages graphiques.
- GraphicTileToken.java utilisé pour gérer les évènements sur la fenêtre graphique.
- Menu.java utilisé pour les menus de sélection du jeu.
- WindowInfo.java utilisé pour gérer les interactions textuelles avec l'utilisateur.

C) Difficultés rencontrées

La conception de la première partie du projet s'est heurtée à plusieurs problèmes :

- Mise en place et compréhension de la bibliothèque graphique Zen.
- Gestion des évènements dans l'interface.
- Gestions de la taille de la fenêtre.

D) Principaux changements suite à la soutenance beta

Suite à la soutenance beta, nous avons pu modifier certains points de notre projet. Notamment sur l'utilisation d'enum. Effectivement, avant la soutenance on utilisait des String pour implémenter les habitats et les animaux. Le fait d'utiliser des enum permet d'éviter les valeurs incohérentes, que le code soit plus lisible et que la modification et l'ajout de valeur soit plus simple.