Projet de PI4 2023

Groupe ED1A :
Laïdouni Mohamed, Boudelaa Fares, Khemakhem Ayman,
Friedmann Eliot, Ben Atia Necerine

Janvier - Mai 2023

1 Présentation du sujet

Le sujet, proposé par Enrica Duchi, consiste à réaliser un jeu inspiré du DoodleJump. Le DoodleJump est un jeu vidéo à un joueur, où le but est de monter le plus haut possible en sautant de plate-forme en plate-forme et en collectant des points.

2 Parties du cahier des charges traitées

Nous avons traité tout le contenu du cahier des charges minimal, c'est-à-dire implémenter les principales fonctions du DoodleJump :

- Le fait de sauter d'une plate-forme à l'autre.
- Le fait de pouvoir prendre un jetpack ou un hélicoptère pour se faire transporter plus haut.
- Le fait de pouvoir cumuler des points.
- L'utilisation de la bibliothèque Swing pour l'interface graphique.

3 Généralisations

Comme suggéré dans le sujet, nous avons ajouté nombreux éléments à ce jeu :

1. Ajout de différents types de plate-formes :
   - Plate-formes de base, fixes.
   - Plate-formes mobiles, qui bougent à l'horizontale et offrent un saut plus haut.
2. Ajout de divers items :
   (a) Des monstres :
      - Toucher un monstre nous fait mourrir.
      - En revanche, sauter sur un monstre le tue et nous fait rebondir.
      - Tirer sur un monstre le tue.
      - Il existe différents types de monstres.

   (b) Des projectiles :
      - On peut tirer des projetctiles depuis notre personnage.
      - Les projectiles servent à tuer les monstres.

   (c) Des pièces :
      - Toucher une pièce augmente notre nombre de pièces.
      - Les pièces sont récoltées pour augmenter le score.
3. Possibilité de passer d'un côté à l'autre de l'écran.
4. Possibilité de mettre le jeu en pause.
5. Possibilité de jouer avec de l'inertie.
6. Ajout d'une difficulté et de différents niveaux :
   - Une difficulté croissante qui intervient sur le type des plate-formes, sur l'écart entre celles-ci, sur la vitesse des plate-formes mobiles et sur le nombre de monstres.
   - Quatres niveaux de jeu, qui interviennent sur l'accroissement de la difficulté (au niveau 4, le jeu devient rapidement difficile).
7. Possibilité de jouer avec différents "skins", créés avec des thèmes.
8. Un système de paramétrage :
   - Choix du niveau.
   - Choix du skin.
   - Choix du jeu avec ou sans inertie.
9. Possibilité de jouer à deux :

- C'est un mode course qui se termine dès que l'un des joueurs à perdu.
- Dans ce mode de jeu, le score et les pièces ne sont pas comptées.
  - Soit sur le même ordinateur.
  - Soit en réseaux (mode multi-joueurs).

10. Ajout d'un identifiant pour le joueur :
    - Cet identifiant est local.
    - Il est créé si le joueur n'a jamais joué de parties, sinon on reprend le même.
    - Chaque joueur possède donc un unique identifiant.
11. Ajout d'un nom pour le joueur :
    - Le nom peut-être changé à chaque partie.
    - Le nom n'influe pas sur l'identifiant du joueur.
    - Le nom par défaut est celui utilisé lors de la dernière partie, ou "Mizer".
    - Le nom apparaît au-dessus du personnage pendant la partie, permettant de différencier les joueurs lors des parties à deux.
12. Un système de classement et d'historique de parties :
    (a) L'historique de parties :
    - L'historique est local.
    - Il sauvegarde l'identifiant, le nom et le score du joueur à chaque parties dans un fichier "history.csv".
    - Il est utilisé pour connaître les données du joueur à la dernière partie, ainsi que son meilleur score.

    (b) Le classement :
    - Le classement est global.
    - Il sauvegarde l'identifiant, le nom et le score de tous les joueurs pour chacunes de leur parties dans un fichier "classement.csv".
    - Il permet d'afficher les 10 meilleurs scores historiques, tout joueur confondu.

4 Représentation du projet

4.1 Représentation des fichiers

Le code constituant notre projet est divisé entre 4 répertoires :

- gameobjects/: contenant tous les fichiers responsables des parties.

  - GameObject : (Abstraite) Représente les différents types d'objets d'une partie, par des coordonnées et des dimensions.
  - Personnage : Représente un personnage. C'est l'objet principal du jeu, un GameObjet avec une vitesse en x et en y.
  - Plateforme : (Abstraite) Représente une plate-forme. C'est un GameObjet avec un saut.
  - PlateformeBase.
  - MovingPlateforme : C'est une plate-forme avec un vitesse en x et un saut plus important.
  - Projectile : Représente un projectile. C'est un GameObject avec une vitesse en x et en y.
  - Monstre : Représente un monstre. C'est un GameObject avec une vitesse en x, une santé et un identifiant.
  - Items : (Abstraite) Représente un item. C'est un GameObjet avec un saut.
  - Fusee et Helicoptere : Représente des items qui apportent un saut plus haut.
  - Coins : Représente une pièce. C'est un item.
  - Joueur : Représente un joueur avec comme attributs son identifiant, son nom, son score et son personnnage.
  - Terrain : Représente le terrain d'une partie dans son intégralité. Cette classe fait le lien entre les différents composants d'une partie.

- gui/ : contenant les fichiers responsables de l'affichage, ainsi que les images et le fichier App.
  - App : Lance le jeu.
  - Vue : Composant (JPanel) qui se charge de l'affichage des éléments.
  - Etat : Représente un état du jeu.
    * MenuDemarrer : Représente l'accueil.
    * MenuSetting : Représente le menu de paramétrage.
    * MenuClassement : Représente le classement (graphiquement).
    * MenuLancement : Représente le menu de paramétrage de début de partie.
    * Game : Représente une partie.
    * MenuFin : Représente l'écran de fin de partie.
  - images/ : Contient toutes les ressources images du jeu.

- leaderboard/ : contentant les fichiers responsables de la sauvegarde des scores.

  - leaderboard : (Abstraite) Représente un gestionnaire de fichier ".csv" pour le classement ou l'historique, avec une méthode pour lire et ajouter une ligne à un fichier.
  - classement : Un LeaderBoard qui gère le fichier "classement.csv".
  - history : Un LeaderBoard qui gère le fichier "history.csv".
  - classement.csv
  - history.csv

- multiplayer/ : contenant les fichiers responsables du jeu en réseaux.
  - JoueurConnecte : classe qui gere le coté du client, contient un socket et les methodes pour envoyer et recevoir les données.
  - Serveur: classe qui gere le coté du "host", contient un serveursocket les methodes pour envoyer et recevoir les données.
  - ThreadMouvement : un Runnable pour gere le thread du multi-joueur.

4.2 Représentation graphique du projet

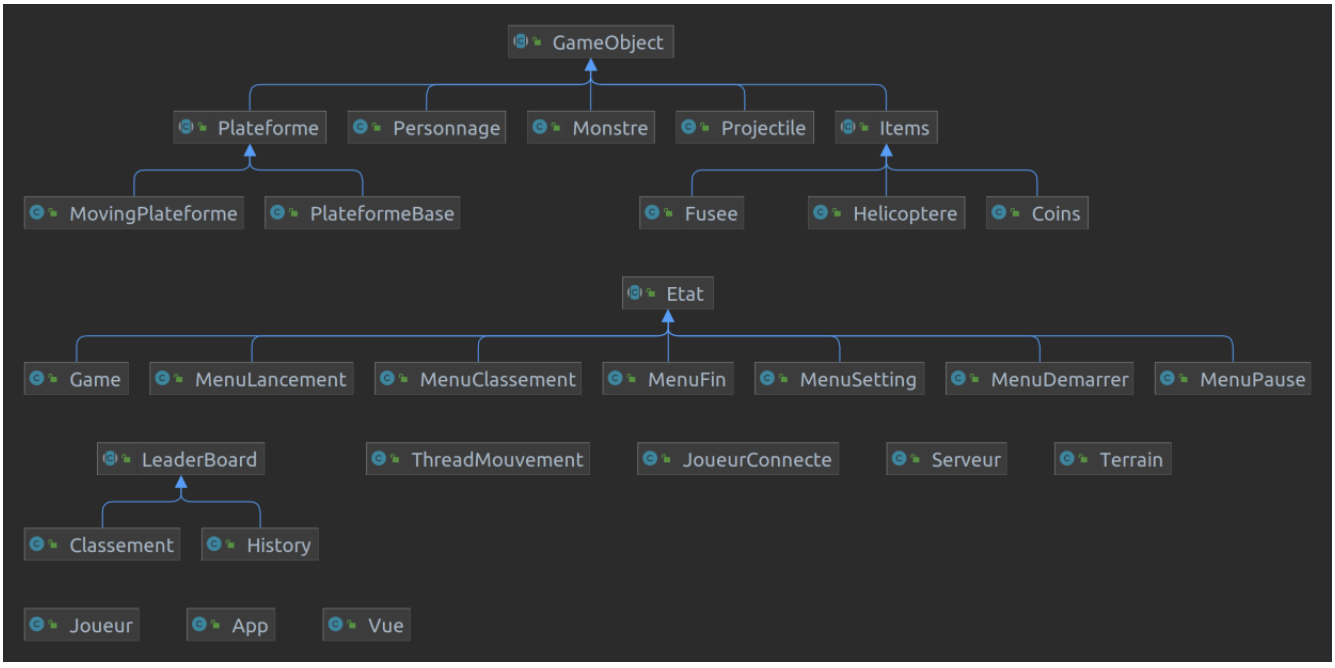

Figure 1: Liens d'héritage entre les différentes classes

5 Problèmes rencontrés

- Les principaux problèmes sont liés au mode multijoueur ou réseau, notamment un problème au niveau des threads. En effet, avec le système actuel, nous utilisons des JOptionPane pour "bloquer" le thread de Java Swing afin de nous assurer que le joueur client se connecte. Pour éviter cela, il serait possible de rajouter une variable booléenne volatile hasStarted dans la méthode run de la classe ThreadMouvement. De plus, il y a des problèmes de latence avec le jeu en cross-plateforme, par exemple, lorsque l'hôte est sur Linux et que le client est sur Windows. Cela entraîne une augmentation significative de la latence, rendant le jeu pratiquement injouable.

6 Pistes d'amélioration

Voici plusieurs idées pour améliorer notre jeu. Celles marquées par (*) sont des idées que l'on a essayé de mettre en place mais qui nous ont posé problème :

- Ajout d'un magasin : (*)
  - Le magasin serait une interface où l'on pourrait acheter et personnaliser les skins de son personnage, pour chaque pack de skin disponible.
  - À la base, les pièces (Coins.java) devaient servir à s'acheter des skins dans le magasin.
  - Par manque de temps, et à cause de la complexité de ce système d'achat, nous avons changer l'utilisation des pièces (comme dit en section 3.2).
- Ajout d'une sauvegarde des paramètres :
  - Les paramètres (niveau, skin, inertie) seraient sauvegardés dans un fichier, comme l'historique.
  - Idée que nous avons eu trop tardivement pour pouvoir la mettre en place.
- Ajout d'autres items de type "booster" :
  - ressorts
  - trampoline...
- Ajout d'autres types de plate-formes :
  - plate-formes qui se cassent lorsqu'on saute dessus et qui ne font pas rebondir.
  - plate-formes qui disparaissent après une utilisation.
  - plate-formes mobiles dans le sens vertical.

  - plate-formes qui, lorsque l'on saute sur l'une d'entre-elles, fait bouger toutes les plate-formes du même type.

- Possibilité d'avoir plusieurs vies :
  - Idée proposée par le sujet, mais non retenue pour garder une certaine difficultée du jeu.

- Ajout d'un (vrai) mode multi-joueurs en réseau : (*)
  - Le mode multi-joueurs actuellement disponible ne fonctionne qu'avec 2 joueurs. L'idée de base était de le faire fonctionner pour plusieurs joueurs. Mais suite à des problèmes (section 5), nous n'avons pas pu le mettre en place.