RAPPORT

Lounès KEBDI Yanis FRANCOIS

Le jeu que nous avons développé est basé sur le modèle architectural MVC (Modèle-Vue-Contrôleur), ce qui a permis une organisation claire du code. Nous avons choisi de travailler ensemble tout au long du projet,
Nous avons travaillé ensemble en présentiel tout au long du projet, car on habite près l'un de l'autre. Cela nous a permis d'adopter une approche collaborative efficace, où nous avons pu immédiatement discuter et résoudre les problèmes au fur et à mesure qu'ils survenaient. Ce mode de travail exclusivement en présentiel, en duo, a grandement facilité notre coordination, nous aidant à ne pas se perdre dans la compréhension du code de l'autre.

Le Modèle

Le modèle, élément central de notre architecture, gère l'état du jeu et la logique métier. Il est composé de plusieurs classes, telles que **Train**, **Action** et **Butin**, chacune jouant un rôle crucial dans la dynamique du jeu. Le modèle traite les actions comme les déplacements et les braquages, en maintenant les états nécessaires.

La Vue

La vue, implémentée via la classe **Tvue** héritant de **JPanel**, est responsable de l'affichage des éléments graphiques. Nous avons rencontré des défis majeurs avec l'utilisation de **JPanel**, notamment des erreurs d'affichage et des problèmes de superposition d'images, qui ont nécessité des recherches approfondies pour une bonne maîtrise. Et un gros travaille sur les coordonnées pour avoir un affichage modulable selon la taille de la fenêtre.

Le Contrôleur

Le contrôleur, géré par la classe **Controleur**, sert d'intermédiaire entre la vue et le modèle. Il traite les interactions de l'utilisateur, comme les clics sur les boutons, pour mettre à jour le modèle selon les actions effectuées dans l'interface.

Difficultés rencontrées

La prise en main des éléments de l'interface graphique Java, tels que JFrame et JPanel, a constitué une part significative de nos défis initiaux. En particulier, l'intégration de ces

composants dans un environnement de jeu interactif et dynamique nous a demandé un effort considérable d'apprentissage et de débogage pour assurer une expérience utilisateur fluide.

En outre, la conception initiale de notre jeu était principalement orientée vers un seul joueur, ce qui a posé des problèmes lorsque nous avons étendu le jeu pour supporter plusieurs joueurs. Adapter le jeu à un mode multijoueur nous a forcés à repenser notre architecture et nos mécanismes de gestion d'état pour permettre à plusieurs joueurs de participer simultanément sans interférence.

Un autre obstacle majeur a été de séparer clairement les deux phases de jeu : la planification et l'action. Nous avons dû développer une méthode pour gérer ces phases de manière à ce que les actions planifiées par les joueurs soient exécutées dans l'ordre correct et au bon moment, tout en maintenant la synchronisation et la cohérence entre les joueurs. La solution a impliqué l'utilisation de structures de données complexes et la mise en œuvre de contrôles de flux spécifiques pour s'assurer que chaque phase se déroule sans chevauchement et respecte les règles du jeu.

Choix Artistique

Avant même de commencer le développement, nous étions déterminés à créer un jeu dans l'univers du RER plutôt que dans celui du Far West. Cette décision s'est naturellement imposée à nous, habitant dans le 77 et passant près de 5 heures dans les transports chaque jour. Cet environnement quotidien nous a inspiré et nous a semblé être un cadre parfait pour apporter une touche personnelle et originale à notre projet. En intégrant des éléments familiers de nos trajets quotidiens.

Conclusion

Ce projet a été une opportunité pour approfondir nos connaissances en programmation orientée objet et en gestion de l'interface utilisateur Java, tout en renforçant notre capacité à travailler en équipe sur des problématiques complexes. En réfléchissant en amont à l'organisation de notre code, nous avons également pu développer nos connaissances dans le domaine du génie logiciel.