université de BORDEAUX

Empire of Lionel

Réalisé par :

Gabriel IFRIM
Arthur MACDONALD

Compte rendu de Projet Java
Cours d'approche object

Master 1 - Semestre 7 - Bordeaux

Contribution au projet :

Nous avons collaboré sur tous les aspects du projet, y compris le développement du backend (logique de l'application) et du frontend (interface utilisateur avec JavaFX). Les tâches n'ont pas été spécifiquement divisées, mais ont été abordées conjointement en fonction des besoins du projet à chaque étape.

Règles du jeu

Difficultés : Le jeu peut être lancé en différentes difficultés, influençant les ressources initiales.

Objectif : Le but du jeu est d'atteindre 1000 résidents.

Gestion des ressources : Les résidents consomment de la nourriture. Si la nourriture atteint 0, les résidents commencent à mourir petit à petit.

Construction et gestion : Construisez des bâtiments, assignez des travailleurs pour produire des ressources, et gérez vos ressources pour prospérer.

Stratégie : Enrichissez-vous en optimisant la production et la consommation des ressources, et en construisant les bâtiments nécessaires pour soutenir une population croissante.

Bilan du Projet

Nous avons réussi à implémenter toutes les fonctionnalités de base du jeu, y compris la gestion des bâtiments, des ressources, de la production et du temps. L'équilibrage du jeu a été ajusté pour assurer une expérience de jeu minimale viable.

Ce qui a été fait

- Implémentation des fonctionnalités de base du jeu (bâtiments, ressources, production, temps).
- Ajustement de l'équilibrage du jeu.
- Création d'une interface utilisateur (UI) simple en pixel art.
- Implémentation d'un système de production de ressources basé sur les travailleurs individuels.

Stratégies adoptées

- **Système de Production de Ressources Indépendant** : Chaque résident a un travail individuel et produit des ressources de manière indépendante, ce qui permet une gestion plus fine de la production et de la consommation.
- **Interface Utilisateur (UI) Simple** : Une UI minimaliste en pixel art a été choisie pour faciliter la compréhension et l'utilisation.

Design Patterns Utilisés

- **Singleton** : Garantir qu'une classe n'ait qu'une seule instance

Exemple :

`SceneManager.getInstance()` pour obtenir l'instance unique de SceneManager.

- **Factory** : Créer des objets sans spécifier la classe exacte de l'objet à créer.

Exemple :

Créer des objets sans spécifier la classe exacte de l'objet à créer.

`WorkFactory.createWork(WorkType.LUMBERJACK)` pour créer une instance de Lumberjack.

- **Observer** : Permettre à un objet (observateur) de recevoir des notifications de changement d'état d'un autre objet (sujet).

Exemple :

`GameTime` notifie les objets `Work` (implémentant `TimeObserver`) des changements de temps.

- **MVC** : Séparer les préoccupations de l'application en trois parties : le modèle (données), la vue (interface utilisateur) et le contrôleur (logique).

Modèle : Représente les données de l'application (par exemple, `Resident`, `Building`).

Vue : Représente l'interface utilisateur (par exemple, fichiers `FXML` comme `main.fxml`).

Contrôleur : Contient la logique de l'application et interagit avec le modèle et la vue.

Quelques contrôleurs et leurs rôles :

MapController :

Gère l'affichage et les interactions avec la carte du jeu. Permet de placer des bâtiments sur la carte et de mettre à jour l'affichage en fonction des actions de l'utilisateur.

ResourceController :

Gère l'affichage des ressources disponibles. Met à jour l'affichage des ressources en fonction des changements dans l'inventaire du joueur.

MenuController :

Gère les interactions avec le menu principal. Permet de démarrer le jeu, d'accéder aux paramètres ou de quitter l'application.

Ce qu'il reste à faire

- **Améliorations Bonus** : Implémentation de la gestion des déplacements des résidents et d'un système d'amélioration des bâtiments.

Difficultés rencontrées

- **JavaFX** : La création de l'UI en pixel art avec JavaFX s'est avérée complexe.

Améliorations possibles

- Amélioration de l'UI et de la gestion des threads.
- Ajustement supplémentaire de l'équilibrage du jeu.
- Ajout de la fonctionnalité d'upgrade des bâtiments pour qu'ils produisent plus de ressources. Nous avons commencé à implémenter cette fonctionnalité, mais elle n'a pas encore été ajoutée visuellement.
- Ajout d'une représentation visuelle des travailleurs se rendant à leur poste de travail et rentrant chez eux la nuit.

Comment jouer

1. **Utiliser Maven pour exécuter le projet.** Depuis le répertoire racine :
2. `cd elo/`
3. `mvn clean javafx:run`

Structure des Classes et Relations

L'image est également présente depuis la racine dans le dossier /output