Équipe : LANDOLSI Ines / MARGUERIT Sophia

Section 1. Introduction

Contexte et Objectifs

Dans le cadre de ce projet, nous avons développé un moteur de jeu 2D extensible en utilisant la bibliothèque LibGDX. Le projet, intitulé ***KittyLost***, est un jeu de plateforme où le joueur contrôle un petit chat perdu.

Gameplay

L'histoire suit un petit chat qui s'est égaré loin de chez lui et doit retrouver son chemin. Le joueur devra guider le chat à travers un environnement dangereux, rempli de pièges tels que des pics, des points d'eau ou des obstacles. Chaque pas rapproche le chat de son foyer, mais il faudra éviter les dangers mortels pour survivre à cette aventure périlleuse.
Les défis du jeu incluent :

- Traverser des plateformes avec agilité.
- Éviter des pièges mortels comme des pics, des points d'eau ou des sables mouvants.
- Maintenir une bonne gestion des mouvements et des sauts pour surmonter les obstacles et esquiver les pièges.

Le jeu ne propose pas de checkpoints, ce qui rend chaque erreur cruciale. Cela fait partie de la dynamique du jeu, un petit jeu plateform 2D simple et mignon mais énervant. Le but est de tester la persévérance du joueur et d'encourager une approche méthodique pour atteindre la fin de l'aventure. Le score maximal est fixé à **100 poissons**, un objectif conçu pour les joueurs les plus persévérants.

Objectifs du Projet

Les principaux objectifs incluent :

1. Créer un moteur de jeu capable de charger et d'interpréter des cartes créées avec l'outil *Tiled*.
2. Implémenter un moteur physique simple pour gérer les collisions et les interactions avec l'environnement.
3. Fournir une interface intuitive pour ajouter de nouveaux contenus (pièges, plateformes, objets) directement via *Tiled* sans modifier le code source.

Section 2. Présentation du Projet

Technologies et Outils Utilisés

LibGDX : Bibliothèque de développement pour le rendu graphique, la gestion des entrées et des interactions physiques.

Tiled : Logiciel utilisé pour concevoir les cartes et organiser les obstacles, pièges et plateformes.

IDE IntelliJ IDEA : Environnement de développement intégré pour écrire et compiler le code facilement.

Git : Utilisé pour la gestion des versions et la collaboration.

Photoshop: Utilisé pour créer les tilesets personnalisés (background).

Fonctionnalités Implémentées

1. **Système de Menu** :
   - Écran d'accueil avec une image illustrant le jeu.
   - Lancement de la partie via un clic sur l'image.
2. **Moteur de Jeu** :
   - Lecture et affichage de la carte Tiled (*long_map.tmx*).
   - Gestion des collisions avec des plateformes et des pièges définis dans Tiled via des calques objet.
   - Détection des interactions joueur-piège pour déclencher l'écran "*Game Over*".
3. **Gameplay** :

Déplacements du personnage :

- Déplacement horizontal (droite (D)/gauche(Q)) avec gestion de la gravité.
- Système de saut contrôlé (Z).
- Réinitialisation de la position du joueur après un "*Game Over*".

Collecte d'objets :

- Collecte de poissons en tant que score

Système de mort :

- Réinitialisation du personnage après un Game Over.

Configuration et Ajout de Contenu avec Tiled

KittyLost offre une grande flexibilité pour les créateurs de contenu. L'ajout de nouvelles cartes ou d'éléments de jeu se fait facilement via Tiled. Voici les étapes pour configurer et étendre le jeu :

Pour ajouter de nouvelles cartes et éléments :

1. **Création des cartes** : Utilisez *Tiled* pour définir :
    - Les couches de tuiles (par exemple, le décor des plateforms, le background).
    - Une couche d'objets nommée collision pour définir les zones solides où le joueur peut marcher.
    - Une couche d'objets nommée pièges pour définir les zones dangereuses entraînant un game over.
    - Une couche d'objets nommée poisson pour définir les emplacements des poissons collectables..
    - Une couche d'objet pour définir la zone marquant la fin de la carte et déclenchant le message de victoire.

2. **Chargement des cartes** :
    - Les différents fichiers pour les graphismes sont dans le répertoire assets.
    - Le moteur charge automatiquement la carte via la classe GameScreen.

3. **Types d'éléments** :
    - **Rectangles** : Pour les plateformes, les poissons collectables et la zone de fin de carte.
    - **Polygones** : Pour les pièges (pics et eau).

Compilation et Exécution

Prérequis

- **Java** : JDK 8 ou supérieur..
- **LibGDX** : Version compatible avec l'environnement.

Étapes de Compilation

1.Récupérer le projet sur GitHub:

- Clonez le dépôt Git : git clone https://github.com/SophiaMgt/PCOO_project.git
- Ou télécharger le dossier zip directement sur GitHub

2. Lancement du jeu:

- Ouvrez le projet dans IntelliJ IDEA et exécutez la classe principale *KittyLostGame (Nécessite Java17)*
- Ou directement en ligne de commande en se passant à la racine du projet.:

  ❯ ./gradlew lwjgl3:run

3. Lancement d'une partie:

- Cliquez n'importe où sur l'image du Menu pour lancer une partie.

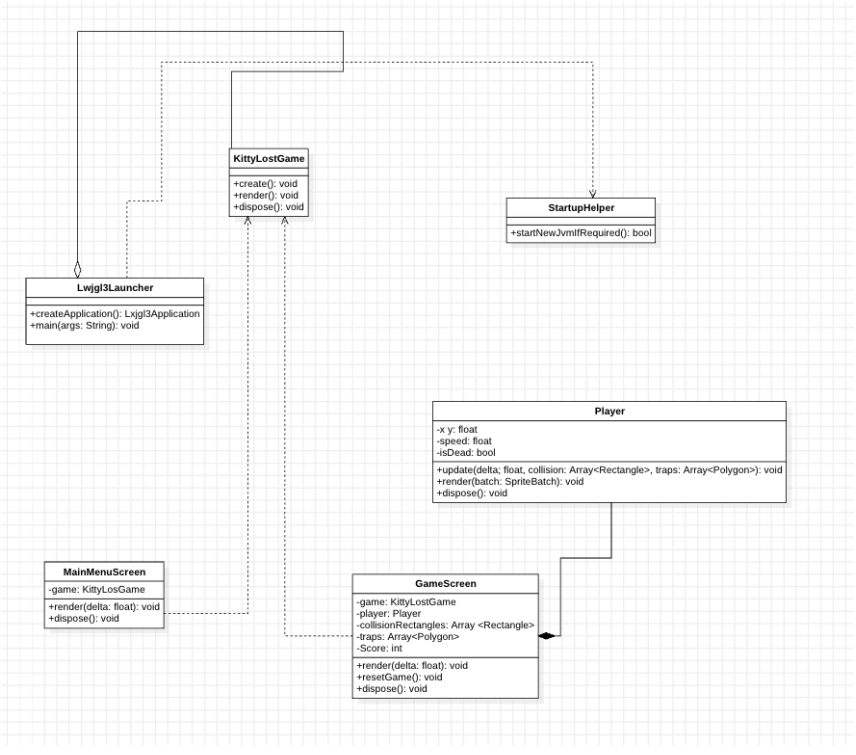
Section 3. Présentation Technique du Projet et Contributions

Architecture Générale du Moteur de Jeu

Le moteur est structuré en plusieurs modules :

1. **KittyLostGame** : Point d'entrée du jeu. Gère la navigation entre les écrans.
2. **MainMenuScreen** : Écran d'accueil affichant une image de fond interactive.
3. **GameScreen** : Écran principal où le gameplay se déroule. Il gère :
   - Le rendu de la carte.
   - Le système de collisions et d'interactions.
   - L'état *Game Over et Win* .
4. **Player** : Classe gérant le comportement du personnage (déplacements, sauts et collisions).

Diagramme UML



Utiliser et Étendre la Librairie du Moteur de Jeu

1. **Ajout d'un nouveau type de personnage** :
   - Héritez de la classe Player pour ajouter des comportements spécifiques.
2. **Ajout d'un nouvel obstacle** :
   - Ajoutez un nouvel objet dans Tiled sur une couche spécifique (par exemple, pièges).
   - Étendez la logique de collision dans la méthode update de la classe Player.

Répartition des Tâches

Ines :

- Développement du moteur physique.
- Développement du gameplay.
- Développement de la carte Tiled (Création plateformes et pièges).
- Gestion des objets collectables (poissons) et association à un score.
- Création du tileset background.
- Participation au développement global du code.
- Test de la praticité de la carte.

Sophia :

- Développement du moteur physique.
- Développement du gameplay.
- Développement de la carte Tiled.
- Développement des "interfaces utilisateur" (Menu, GameOver, Win).
- Gestion des collisions/ interactions avec les plateformes.
- Gestion de la mort du personnage et du score "Dead".
- Participation au développement global du code.

Section 4. Conclusion et Perspectives

Bilan du Projet

Le projet **KittyLost** a permis de réaliser un moteur de jeu 2D fonctionnel et extensible, répondant aux objectifs pédagogiques fixés. La collaboration a permis d'intégrer des fonctionnalités avancées tout en assurant la stabilité du moteur.

Défis rencontrés :

- Gestion précise des collisions dans un environnement dynamique.
- Optimisation de la fluidité du gameplay avec une caméra centrée.

Perspectives

Gameplay:

- **Ajout d'ennemis avec des comportements spécifiques** :

Intégrer des ennemis avec des stratégies différentes ajouterait de la difficulté au jeu. Ces ennemis pourraient surveiller certaines zones ou poursuivre le joueur lorsqu'il entre dans leur champ de vision.

- **Conditions météorologiques** :

Simuler des phénomènes météorologiques tels que le vent ou la neige pourrait influencer la jouabilité. Par exemple, modifier la trajectoire des sauts ou ralentir les déplacements à cause du vent ou rendre certaines surfaces glissantes à cause de la neige.

- **Système de checkpoints et transitions entre niveaux** :

Ajouter des checkpoints permettrait de sauvegarder la progression, rendant le jeu plus simple. Les transitions fluides entre niveaux offriraient une meilleure continuité et encouragerait les joueurs à explorer davantage.

Amélioration des graphismes

- **Animation du personnage** :

Donner vie au chat grâce à des animations fluides pour ses déplacements, ses sauts et ses interactions avec l'environnement.

- **Choix d'avatars** :

Permettre au joueur de sélectionner parmi différents chats.

- **Ajout d'effets sonores** :

Une musique d'ambiance adaptée (calme ou tendue selon les niveaux) ainsi que des effets sonores dynamiques (sons de pas, bruitages des pièges, collecte de poissons) enrichiraient l'environnement sonore du jeu.

3. Nouvelles fonctionnalités

- **Classement des scores avec saisie du pseudonyme** :

Introduire un système de classement permettrait aux joueurs de comparer leurs performances, encourageant une compétition. L'ajout d'une interface pour entrer un pseudonyme renforcerait cet aspect communautaire.

- **Développement de niveaux progressifs en difficulté** :

Proposer une série de niveaux avec une courbe de difficulté croissante maintiendrait l'intérêt du joueur tout en testant progressivement ses compétences. Cela pourrait inclure des pièges plus complexes, des plateformes mouvantes ou des énigmes légères.

Annexes

- Documentation LibGDX : https://libgdx.com/wiki/start/setup
- Tutoriel Tiled : https://doc.mapeditor.org/en/stable/manual/introduction/
- Tileset (carte) : www.kenney.nl
- Base des mages : généré avec IA : https://designer.microsoft.com

  UML : généré avec StarUML