

# Rapport de Projet – Développement d'un Moteur de Jeu 2D avec LibGDX

Titre du projet : snake\_jeux

## Équipe et Contributions

ALMLIHAN Taimaa – Développement du moteur de jeu, intégration de Tiled, architecture générale.

BAH Maimouna – Développement du gameplay, gestion du serpent, collisions et score.

CISSE Mohamed – Développement des interfaces, gestion des niveaux, menus et sons.

## Section 1. Introduction

Ce projet s'inscrit dans le cadre du module de Programmation et conception orientée objet en Licence 3. L'objectif principal est de concevoir un moteur de jeu 2D extensible en utilisant LibGDX et l'éditeur de cartes Tiled. Le projet prend la forme d'un jeu de type Snake afin de démontrer une architecture propre, modulaire et évolutive.

L'objectif est de permettre l'ajout de contenu (cartes, obstacles, fruits, position de départ du serpent) directement depuis Tiled, sans modifier le cœur du code Java. La séparation claire entre la logique du moteur et les données du jeu constitue un point central du projet.

## Section 2. Présentation du projet

### Technologies et Outils Utilisés

LibGDX : utilisé comme moteur de jeu pour la gestion du rendu, des entrées utilisateur et du cycle de vie du jeu.

Tiled : utilisé pour la création et la configuration des cartes du jeu.

Java : langage principal de développement.

Visual Studio Code : IDE utilisé pour le développement.

Git et GitHub : gestion de versions et collaboration.

Système Linux via WSL pour l'exécution.

### Fonctionnalités implémentées

Chargement dynamique des cartes depuis des fichiers .tmx.

Gestion des obstacles, bordures et fruits via des calques Tiled.

Déplacement et croissance du serpent.

Système de score, victoire et défaite.

Gestion de plusieurs niveaux.

Interface de menus, sélection de niveaux et pause.

Gestion du son et de la musique.

## Configuration et Ajout de Contenu avec Tiled

L'ajout de nouvelles cartes se fait directement via l'éditeur Tiled. Chaque carte contient plusieurs calques : bordures pour les limites, obstacles pour les collisions, fruits pour les objets collectables et un calque d'objets définissant la position de départ du serpent.

Le moteur charge les cartes via la classe TiledGameMap, qui interprète automatiquement les différents éléments définis dans Tiled.

## Compilation et exécution

**Prérequis :** Java JDK 8 ou supérieur, LibGDX configuré, environnement Java fonctionnel.

**Étapes de compilation :** compilation via Gradle ou directement depuis l'IDE.

**Lancement du jeu :** exécution de la classe DesktopLauncher.

Lien GitHub : <https://github.com/Moh-CISSE/CISSE2> .

## Section 3. Présentation technique du projet et contributions

### Architecture Générale du moteur de jeu

Le moteur repose sur une architecture basée sur le système de Screens de LibGDX. La classe SnakeGame gère la navigation entre les écrans (Menu, Sélection de niveau, Jeu). GameScreen contient la boucle de jeu principale, tandis que TiledGameMap gère le chargement et l'interprétation des cartes.

### Diagramme de classes et architecture orientée objet

Entity : classe mère contenant position, taille et gestion des collisions.

Border : représente les limites de la map et provoque un Game Over.

Collectible : interface des objets ramassables comme les fruits.

Collidable : interface définissant les objets pouvant entrer en collision.

Direction : enum UP, DOWN, LEFT, RIGHT.

GameScreen : coeur du jeu, gère gameplay, score, pause, victoire et défaite.

LevelSelectScreen : menu de sélection des niveaux.

MenuScreen : menu principal unique.

ObjectFactory : prévue pour créer les objets automatiquement.

Obstacle : élément bloquant provoquant un Game Over.

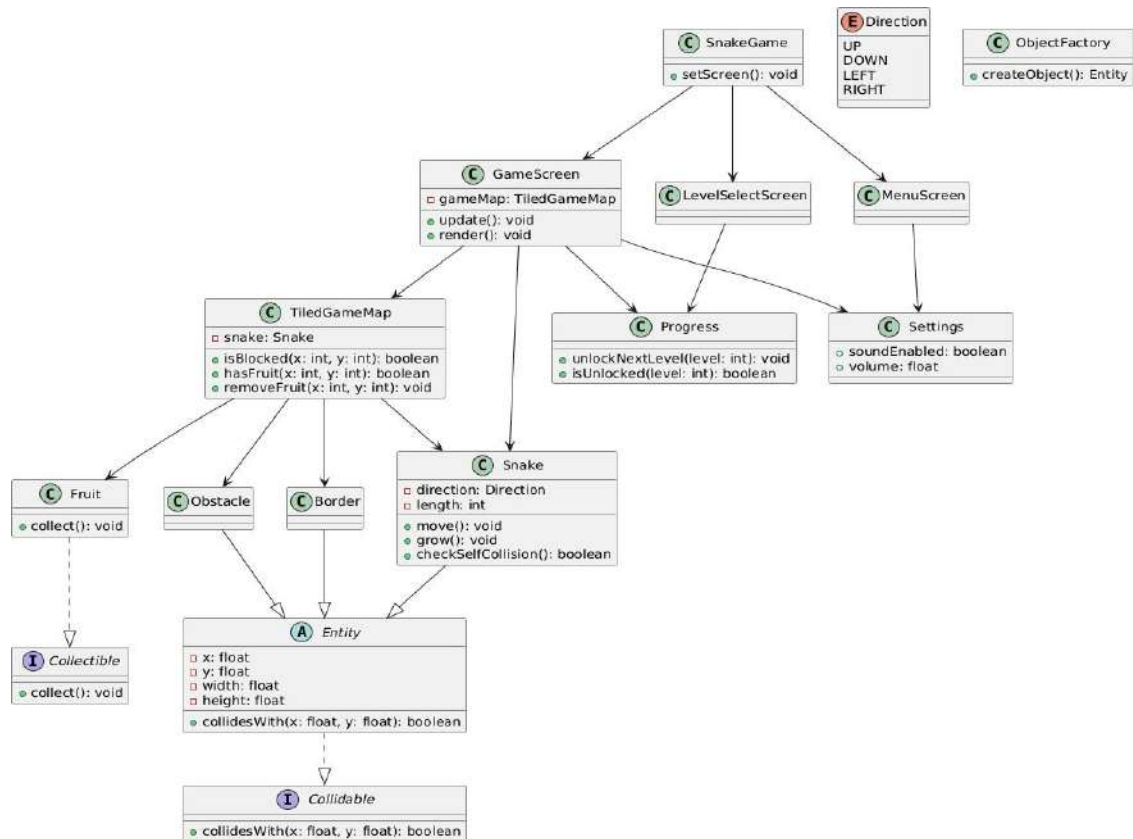
Progress : sauvegarde et déverrouille les niveaux.

Settings : paramètres globaux comme le son.

Snake : gère le déplacement, la croissance et les collisions.

SnakeGame : point d'entrée LibGDX.

TiledGameMap : charge les cartes TMX et crée les objets du jeu.



## Utiliser et étendre la librairie du moteur de jeu

Pour étendre le moteur, un développeur peut hériter de la classe **Entity** afin de créer de nouveaux objets du jeu. Les interfaces **Collidable** et **Collectible** permettent d'ajouter facilement de nouveaux comportements.

De nouveaux niveaux peuvent être ajoutés sans modifier le code Java, uniquement en créant de nouvelles cartes avec **Tiled**.

## Répartition des Tâches

ALMLIHAN Taimaa : architecture générale, moteur de jeu, intégration **Tiled**.

BAH Maimouna : logique du serpent, gameplay, collisions et score.

CISSE Mohamed : interfaces utilisateur, menus, niveaux et gestion sonore.

## Section 4. Conclusion et Perspectives

Ce projet a permis de concevoir un moteur de jeu 2D fonctionnel, modulaire et extensible. Les objectifs initiaux ont été atteints, notamment la séparation entre moteur et données.

Parmi les perspectives d'amélioration figurent l'ajout d'ennemis, de nouveaux types d'objets, ainsi qu'un enrichissement des paramètres configurables via **Tiled**.

# Annexes

Documentation LibGDX officielle.

Documentation de l'éditeur Tiled.

Tutoriels et ressources communautaires utilisés durant le développement.