Sprint 3 Séance de TP

Projet : LO3BA
(ACL-ISN 2025 Mines Rabat)

1 Backlog Sprint 3

Objectif global

Rendre le jeu plus dynamique et complet : intégration des ennemis, gestion des collisions avancées, détection de fin de partie, amélioration du HUD et ajout de mécanismes de progression (vies, score persistant, niveau actuel).

1.1 User Stories

ID	En tant que...	Je veux...	Afin de...	Priorité
US9	Joueur	Rencontrer des ennemis mobiles (ex. cactus animés, oiseaux)	Avoir des obstacles dynamiques à éviter	Haute
US10	Joueur	Subir des dégâts en touchant un ennemi	Ressentir la tension du jeu	Haute
US11	Joueur	Voir mon nombre de vies diminuer et le jeu se terminer à 0 vie	Savoir quand jai perdu	Haute
US12	Joueur	Voir le score, les vies et le niveau actuel dans le HUD	Suivre ma progression en temps réel	Haute
US13	Joueur	Recommencer le niveau ou le jeu après une défaite	Pouvoir retenter sans tout recommencer	Moyenne
US14	Développeur	Ajouter une classe `Enemy` héritant de `Entity`	Gérer différents types dennemis de façon modulaire	Haute
US15	Développeur	Implémenter un système de collisions avancées (AABB, gestion des rebonds)	Détecter précisément les interactions joueur/objets	Haute
US16	Développeur	Mettre à jour le diagramme UML avec `Enemy`, `CollisionManager`, `GameState`	Maintenir une architecture claire	Moyenne
US17	Développeur	Créer un écran de *Game Over* avec option de reprise	Gérer proprement la fin de partie	Moyenne

2 Conception (Prévision UML Sprint 3)

Nouvelles classes prévues

Classe	Rôle principal	Détails
`Enemy` (héritée de `Entity`)	Représente un ennemi mobile	Comportement IA simple (mouvement linéaire, patrouille), collision avec joueur.
`CollisionManager`	Gère toutes les détections de collision	Méthodes statiques ou singleton, vérification AABB, gestion des tags (player, enemy, collectible).
`GameState`	Suit létat global du jeu	Vie du joueur, score, niveau actuel, état (PLAYING, PAUSED, GAME_OVER).
`GameOverScreen`	Affiche lécran de fin	Boutons ń Recommencer ż et ń Quitter ż, affichage du score final.
`HUDController` (amélioration)	Gère linterface complète	Affichage du score, vies (icônes cur), niveau actuel, mise à jour en temps réel.

Relations UML simplifiées
Voir image : /docs/uml/LO3BASprint3UML.png

3 Répartition des tâches (Sprint 3)

Membre	Rôle / Tâches principales
Yassine 1	Implémentation de la classe `Enemy` (mouvement, animation, IA simple) + intégration dans `Level`.
Yassine 2	Développement du `CollisionManager` (AABB, gestion des tags) et tests de collision joueur/ennemi/objets.
Oualid	Création du système de vies, `GameState`, écran `GameOverScreen` + logique de reprise.
Wadie	Amélioration du `HUDController` (vies, niveau, score), mise à jour UML, documentation et préparation du tag Git `v3.0`.

4 Livrables de la séance

— Backlog Sprint 3 (ce fichier)
— Conception UML mise à jour (`/docs/uml/LO3BASprint3UML.png`)
— Répartition des tâches (tableau ci-dessus)
— Système dennemis fonctionnel (au moins 2 types : sol + aérien)
— Gestion des collisions avancées et perte de vie
— Écran de *Game Over* avec reprise
— HUD complet (score, vies, niveau actuel)
— Tag Git `v3.0` à la fin du sprint

5 Prochaine étape (préparation Sprint 4)

— Ajout de power-ups (ex. invincibilité, double saut)
— Système de sauvegarde/chargement de progression
— Effets sonores et musique de fond
— Optimisation des performances (pooling dobjets)
— Préparation de la démo finale et du rapport de projet