Examen Unity C#

Flipper

Examinateur:

Mickaël Mancini
m.mancini@rubika-edu.com
mickael.mancini@gmail.com

Objectif:

Réaliser un Flipper sous Unity3D

Contraintes:

Moteur de jeu : Unity 3D

- L'examen doit être réalisé avec le moteur Unity 3D, version spécifiée : Unity 6 LTS.
- Utilisation de C# comme langage de programmation pour tous les scripts.

Physique du jeu :

- Utilisation du moteur physique de Unity pour la gestion des collisions et interactions.
- La balle de flipper doit réagir de manière réaliste aux éléments du décor (palettes, bumpers, trous, etc.).
- Les matériaux physiques (Physic Materials) doivent être configurés pour gérer la friction et le rebond de la balle.

Systèmes de jeu à implémenter :

- **Palettes de flipper (Flippers)** : Doivent être contrôlables via des touches du clavier (ex : touche gauche/droite ou flèches).
- **Bumpers et obstacles** : Des bumpers doivent renvoyer la balle lorsqu'elle entre en collision avec eux, avec une force configurable.
- **Gestion des scores** : Le jeu doit comporter un système de scoring, qui augmente lorsque la balle touche certains objets comme les bumpers.
- **Gestion des vies** : Mise en place d'un système de vies. La balle doit être remise en jeu lorsqu'elle tombe dans la zone de perte, avec réduction du nombre de vies.

Interface utilisateur (UI) :

- Affichage du score en temps réel à l'écran.
- Affichage du nombre de vies restantes.
- Interface de fin de jeu avec le score final et une option de redémarrage.

Limitation de ressources externes :

- Les étudiants ne peuvent pas utiliser des scripts ou des assets externes non autorisés.
- Les assets graphiques (images, textures, sons) peuvent être fournis ou générés rapidement via Unity (objets 3D simples ou primitives).

Code propre et structuré :

- Le code doit être bien structuré.
- Respect des principes de programmation orientée objet (POO) : utilisation des classes, méthodes, et encapsulation.

Bonus et extensions (optionnel) :

- Ajout d'effets visuels ou sonores pour les collisions avec les bumpers et les obstacles.
- Amélioration de la gestion de la caméra pour suivre la balle.
- Ajout d'un système de multiplicateur de score ou de bonus.

Consigne:

- Le jeu doit comporter un menu permettant de redémarrer et de quitter.
- Le projet zippé, le build zippé ainsi qu'un gif nommé "NOM Prenom" d'une taille inférieure à 50 mo et moins de 1000 frame devront se trouver dans le dossier nommé "NOM Prenom"
- Un fichier Read Me explicitant les difficultés rencontrées (crash, bugs, etc…), les challenges surmontés, les contrôles et les certaines subtilités que l'examinateur pourrait omètre devra se trouver également dans le dossier de rendu.
- Le non-respect de la nomenclature entraîne une perte de points.

Critère de notation:

- Organisation et Présentation du Projet
- Fonctionnalité des Mécaniques de Jeu
- Qualité et Structure du Code
- Expérience Utilisateur et Contrôles
- Rendu Visuel et Ressenti du Jeu
- Fonctionnalités Supplémentaires et Créativité
- Résilience aux Bugs et Playtesting

Conseil:

- Pour enregistrer le GIF vous pouvez utiliser le package Unity "Recoder" ou installer un logiciel tier comme ScreenToGif
- Gardez votre hierarchy organisé en créant des liens de parenté
- Vous avez le droit d'utiliser des packs d'assets graphiques, mais il faut indiquer dans le Read Me où vous les avez trouvés.
- Vous n'avez pas le droit d'utiliser Chat GPT, Github copilot, etc… pour générer du code.
- Vous n'avez pas accès à internet autrement que pour accéder à la doc unity et l'asset store, autrement dit, vous n'avez pas accès à Youtube ou des sites de tutoriels.

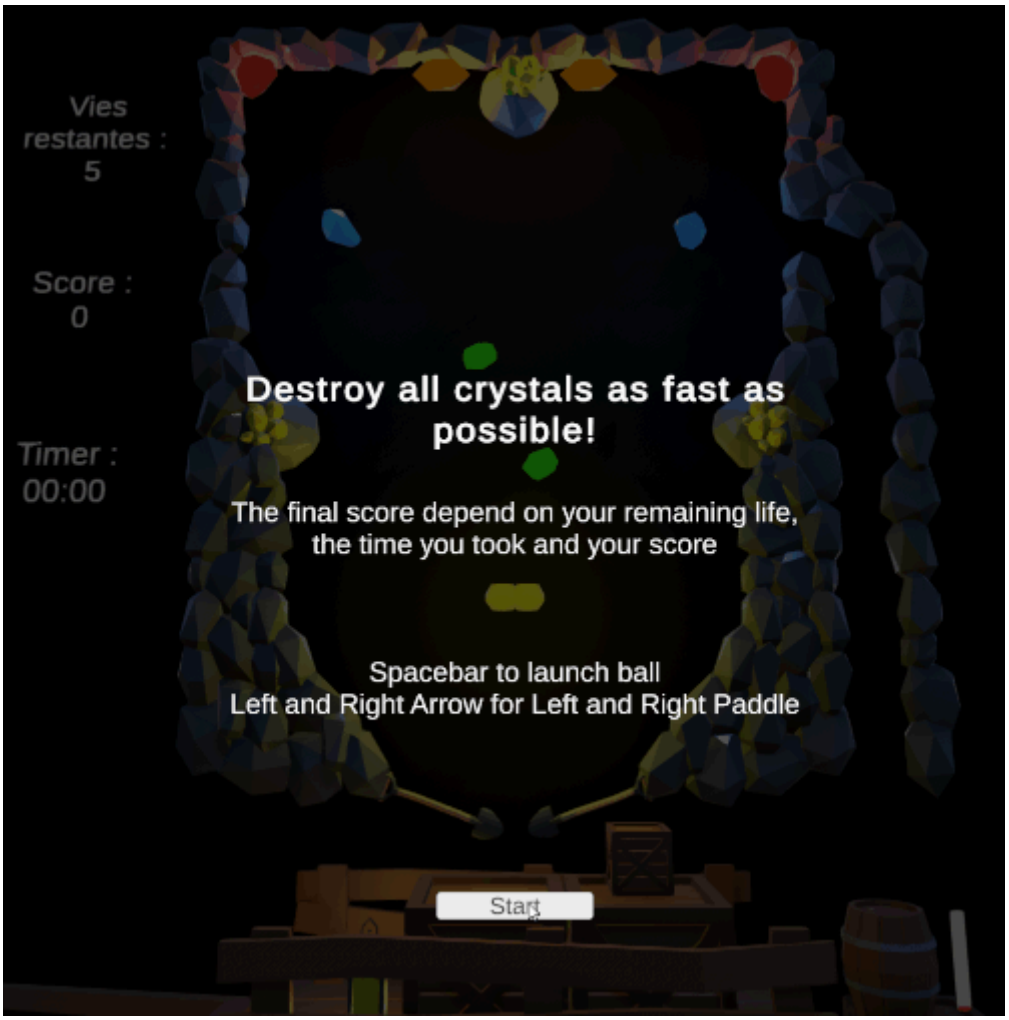
Référence:

Pinball Spire
https://www.youtube.com/watch?v=qTgcE7MH0I8

XENOTILT
https://www.youtube.com/watch?v=bfoaN9pCcqs

Pinbleton Park
https://store.steampowered.com/app/3323970/Pinbleton_Park/?l=french

Pierre-Antoine CHALARON

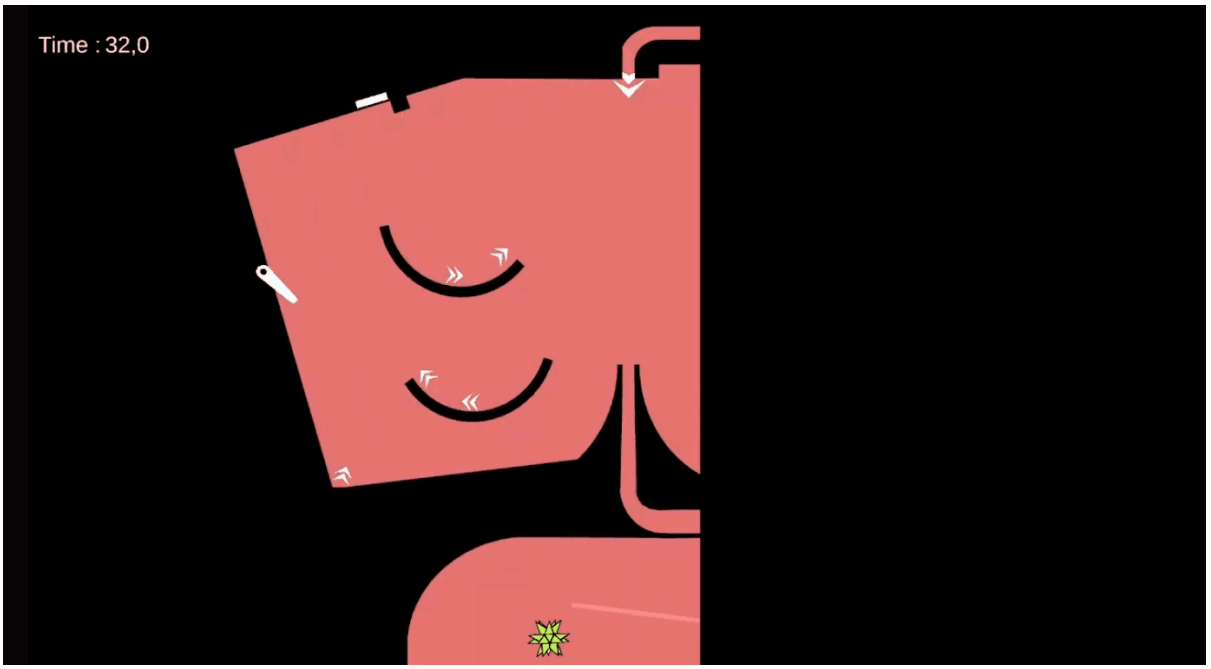


DURUY Charles

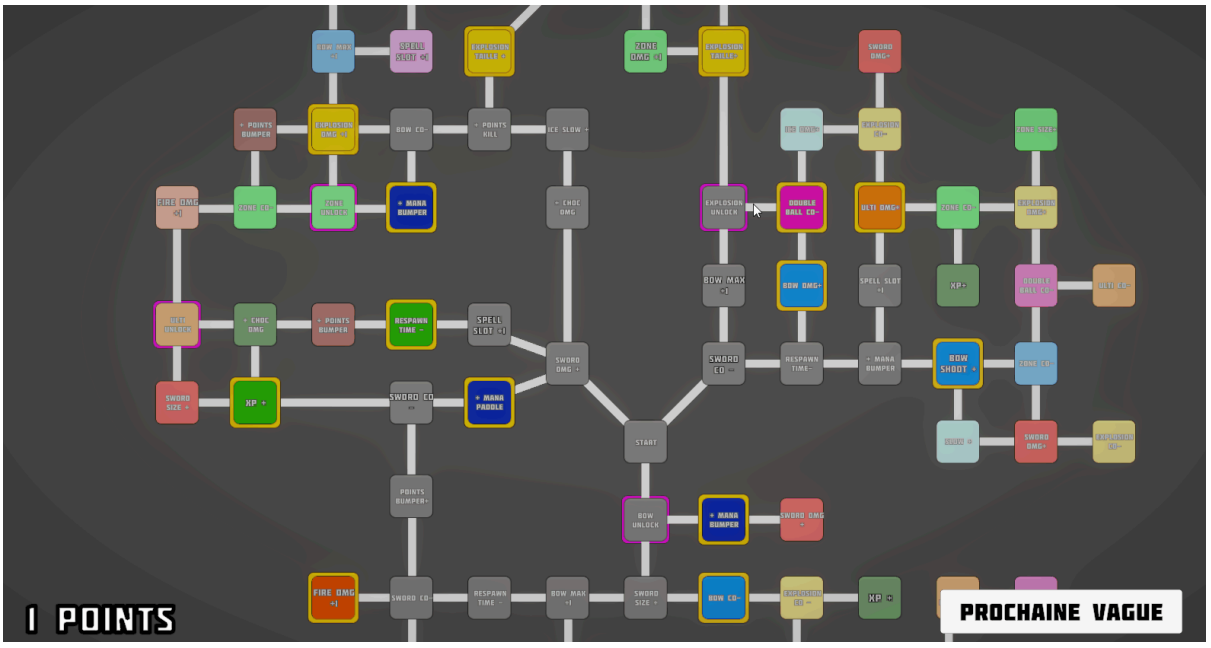
STRAPPAZZON Clément

HUYGHE Charles

DREYFUS Benjamin



LEVY Nathan



CINUS Nathan

LEZIN Eliott