



EPITA

PROJET JEU VIDÉO SUP S2

---

## Rapport de soutenance 2

---



*Projet : Best Of Ten*

*Groupe : Less An Year*

Jules RAITIERE-DELSUPEXHE  
Mathieu EVEN  
Mathéo CRESPEL  
Lucas BESNARD

Janvier 2022 - Juin 2022

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Les membres du groupe . . . . .	2
1.2	Préambule du projet . . . . .	2
<b>2</b>	<b>Description des fonctionnalités du jeu</b>	<b>3</b>
2.1	Origine du projet . . . . .	3
2.2	Règles du jeu . . . . .	3
<b>3</b>	<b>Avancement du projet</b>	<b>4</b>
3.1	Multijoueur . . . . .	5
3.1.1	Synchronisation . . . . .	6
3.2	Intelligence artificielle . . . . .	7
3.2.1	Fonctionnement . . . . .	7
3.2.2	Problèmes rencontrés . . . . .	8
3.2.3	Implémentation . . . . .	9
3.2.4	Mise à jour des zones . . . . .	9
3.2.5	Implémentation des comportements d'IA . . . . .	9
3.3	Interface utilisateur . . . . .	9
3.4	Création des parcours . . . . .	12
3.5	Décors . . . . .	14
3.6	Modélisation 3D . . . . .	16
3.7	Communication . . . . .	17
3.8	Choix de l'apparence du joueur . . . . .	18
3.9	Système des 3 manches . . . . .	18
3.10	Interaction avec les objets . . . . .	19
<b>4</b>	<b>Récit de la réalisation</b>	<b>19</b>
4.1	Nos peines . . . . .	20
4.2	Nos joies . . . . .	20
<b>5</b>	<b>Avenir du projet</b>	<b>20</b>
5.1	Apparence du joueur . . . . .	20
5.2	Création des parcours et des décors . . . . .	20
5.3	Interface utilisateur . . . . .	21
5.4	Mode solo . . . . .	21
5.5	Bonus . . . . .	21
5.6	Site web . . . . .	21
<b>6</b>	<b>Conclusion</b>	<b>22</b>

## 1 Introduction

Le groupe « Less An Year » est fier de vous présenter ce rapport de soutenance destiné à présenter ce qui a été réalisé depuis notre première soutenance.

Ce document contient : une description générale du jeu, de ses origines, de ses règles ainsi qu'un aperçu des fonctionnalités, une description de l'avancement du projet, notre ressenti global sur le déroulement du projet qu'il soit positif ou négatif et notre planning sur les tâches à venir.

### 1.1 Les membres du groupe

Notre groupe est composé de 4 personnes : Jules RAITIERE-DELSUPEXHE, Mathieu EVEN, Mathéo CRESPEL et Lucas BESNARD.

- Jules : Organisé et assidu, il sera le chef de groupe du projet. Il apportera rigueur et clarté à la répartition des tâches. En tant que responsable, nous comptons sur sa logique et sa bonne perception des problèmes pour faire avancer le projet dans la technique comme dans l'organisation.
- Mathieu : Ayant réalisé quelques projets informatiques, il avait déjà une idée du déroulement des projets en général. Il va donc pouvoir apporter son expérience au groupe. Nous pourrons compter sur sa persévérance pour atteindre les objectifs fixés.
- Mathéo : Adorant les vieilles consoles et les vieux jeux, il a déjà fait de nombreux projets sur ces dernières, lui donnant une expérience sur la réalisation de jeux vidéo. Même si ce projet est peu semblable à ce qu'il a pu faire auparavant, il saura apporter son calme, ses idées et ses conseils pour le mener à bien.
- Lucas : Novice en informatique, il n'avait pas touché à une seule ligne de code avant d'intégrer EPITA. Cependant, sa détermination et sa motivation lui ont permis de faire une progression exponentielle. Nous comptons sur sa rigueur et sa capacité d'apprentissage pour tirer le projet vers le haut.

### 1.2 Préambule du projet

Nous sommes en train de réaliser un jeu de plateforme uniquement disponible pour Windows. Aujourd'hui, de nombreux jeux nécessitent des heures de pratique pour prendre en main tous les éléments et espérer pouvoir battre des adversaires. Ils nécessitent du temps, des dizaines de Gigaoctets de mémoire et un ordinateur puissant.

De notre côté, nous produisons un jeu accessible à tous et rapide à comprendre. Cela est possible grâce à des parties de détente entre amis ou avec des

inconnus pendant une durée de 10 minutes environ. Le joueur pourra se divertir grâce à 2 modes de jeu différents : contre la montre et multijoueur. Il évoluera dans des cartes représentant différents environnements, son objectif sera de franchir un enchaînement d'obstacles le plus rapidement possible. Chaque partie sera divisée en 3 manches et contiendra toujours 10 joueurs (IA comprise). C'est un jeu compétitif avec un seul gagnant : celui-ci devra obtenir le meilleur temps face à ses adversaires lors des différentes épreuves.

## 2 Description des fonctionnalités du jeu

### 2.1 Origine du projet

L'idée de Best Of Ten vient des couloirs de EPITA, de nombreux étudiants jouent entre eux, aux intercours, à un jeu de plateforme de même type : Stumble Guys qui est un clone mobile de Fall Guys. Les élèves s'amusent entre eux et rigolent, et c'est ce que l'on recherche dans la conception de notre jeu : du fun et de l'amusement. Beaucoup de jeux auxquels nous avons joué nous ont inspirés pour les graphismes en ombrage de celluloïd (*Cell Shading*), notamment Fortnite : le *Cell Shading* est une technique graphique qui consiste à représenter des décors de la réalité avec des graphismes cartoons. Après plusieurs idées ajoutées ou écartées, notre jeu a pris forme avec des originalités telles que des objets intégrés au décor avec lesquels on peut interagir ou encore des bonus récupérables sur le parcours.

### 2.2 Règles du jeu

Chaque partie se déroulera sur 3 manches. 10 joueurs seront présents pendant les 3 manches. Ces manches seront des courses d'obstacles ou des courses spéciales (avec des fonctionnalités en plus ou en moins par rapport aux courses normales).

**Le but est d'arriver le plus rapidement possible à la ligne d'arrivée sur les 3 manches :**

- À la fin d'une manche, le joueur se voit attribuer des points en fonction de sa place d'arrivée.
- À la fin d'une partie, les points obtenus par le joueur sur les 3 manches sont additionnés pour donner le score final du joueur. Ce dernier est placé dans un classement qui donnera le vainqueur de la partie.

Toutes les règles sont permises pour arriver le plus rapidement possible jusqu'à la ligne d'arrivée. Certaines règles peuvent être appliquées en fonction du type

de manche (course d'obstacles ou course spéciale) ou bien en fonction du type de terrain (jungle, désert, etc.).

### 3 Avancement du projet

Aujourd'hui, le jeu se déroule de la manière suivante : on se connecte à Internet, on lance le jeu puis on arrive sur l'écran d'accueil. Celui-ci a pour arrière-plan un aperçu de nos environnements dans lesquels les parties pourront se dérouler. Sur cet écran d'accueil sont disposés quatre boutons ainsi qu'un champ de saisie. Les boutons permettent de choisir l'apparence de son joueur, de rejoindre une salle, d'en créer une ou de quitter le jeu :

- Le champ de saisie permet d'affecter un nom à notre joueur, après la première ouverture du jeu, le nom choisi sera sauvegardé et sera réassigné à chaque nouveau lancement du jeu.
- Lorsque l'on veut changer d'apparence, on clique sur le bouton à cet effet qui nous redirige vers une scène où sont disposés deux personnages, on peut donc choisir celui avec lequel on souhaite jouer.
- Lorsque l'on clique sur le bouton pour créer une salle, un champ de saisie est affiché afin de saisir le nom de la salle puis on entre dans la salle et la liste des joueurs est affichée. Il y a un bouton pour revenir au menu principal ainsi qu'un bouton pour lancer la partie.
- Lorsque que l'on souhaite rejoindre une salle, la liste des salles actives est affichée et rejoignable.

Lorsque l'on est prêt, le créateur de la salle peut lancer la partie, notre personnage apparaît donc dans une scène à côté des autres joueurs humains et des intelligences artificielles. Notre personnage peut marcher : en avançant, reculant, en allant à gauche ou à droite grâce aux touches ZQSD. Il peut aussi sauter grâce à la barre espace ou courir en maintenant la touche majuscule gauche appuyée. Nous pouvons aussi bouger la caméra autour du personnage avec la souris. Tous ces mouvements sont accompagnés des animations correspondantes.

Le monde dans lequel nous sommes peut-être différent selon la partie, l'environnement peut changer : forêt, désert, glacier, ville et prochainement bien d'autres. Le parcours, peu importe l'environnement, comporte plusieurs obstacles, certains réagissent lorsqu'ils entrent en contact avec le joueur (tourniquets, boules roulantes) tandis que d'autres bougent automatiquement et repoussent le joueur lorsque les deux rentrent en collision (faucheuse, murs mouvants). En haut à droite de l'écran se situe notre temps sur le parcours et notre place dans la course se situe en haut à droite. Les intelligences artificielles, comme

les joueurs, cherchent à finir le parcours au bout duquel se trouve l'arche d'arrivée. Pendant toute la partie, on peut cliquer sur « Échap » pour afficher le menu pause qui nous permet de quitter la partie pour revenir à l'écran d'accueil. Lorsque la ligne d'arrivée est franchie, le personnage danse pour célébrer sa victoire et son chronomètre s'arrête pour lui indiquer son temps final.

### 3.1 Multijoueur

Pour le bon fonctionnement de notre jeu, le multijoueur est important, nous pensions l'avoir terminé, mais celui-ci s'avère plus difficile que prévu...

Lors de l'implémentation initiale de celui-ci, Mathieu et Jules se sont formés sur YouTube à l'aide de nombreuses vidéos sur le sujet. Après une première expérience décevante avec Mirror, nous avons finalement choisi d'utiliser Photon, un pack Unity disponible sur l'Asset Store. Bien qu'il soit plus difficile à prendre en main que Mirror, il est beaucoup plus complet et permet plus de choses, c'est pour cela qu'on l'a choisi. Il prend en charge les serveurs dédiés qui nous permettent de jouer ensemble dès l'instant où l'on est connecté à Internet.

Ce système s'articule autour de plusieurs scripts, notamment pour la connexion au serveur et pour la création des salles. Toutes les fonctions sont déjà prédéfinies grâce à Photon, il a donc été facile d'implémenter le multijoueur en mettant bout à bout ces fonctions. Le fonctionnement est le suivant : lorsque le jeu est lancé, la fonction *PhotonNetwork.ConnectUsingSettings* est appelée pour se connecter au réseau, puis la fonction *PhotonNetwork.JoinLobby* est appelée en arrière-plan pour rejoindre le menu principal. Ensuite, il a fallu mettre en place la création de salles. La fonction *PhotonNetwork.CreateRoom* permet de le faire simplement. On a maintenant chaque joueur connecté au même serveur et donc un multijoueur fonctionnel.

```
void Start()
{
    Debug.Log("Connecting to Master");
    PhotonNetwork.ConnectUsingSettings();
}

public override void OnConnectedToMaster()
{
    Debug.Log("Connected to Master");
    PhotonNetwork.JoinLobby();
    PhotonNetwork.AutomaticallySyncScene = true;
}

public void CreateRoom()
{
    if (string.IsNullOrEmpty(roomNameInputField.text))
    {
        return;
    }
    PhotonNetwork.CreateRoom(roomNameInputField.text);
    MenuManager.Instance.OpenMenu("loading");
}
```

FIGURE 1 – Différentes fonctions de Photon

### 3.1.1 Synchronisation

Notre projet étant de plus en plus avancé, il nous est parfois compliqué de résoudre certains problèmes. L'un d'entre eux concernant le multijoueur est la synchronisation de l'ensemble des obstacles et des joueurs. Cette synchronisation est capitale pour notre jeu : elle permet à l'ensemble des joueurs de voir, d'entendre et d'interagir avec les mondes qu'ils traversent de la même manière. Il est donc important de la mettre en place le mieux possible.

Photon, l'utilitaire nous permettant d'implémenter le multijoueur de notre jeu, permet de mettre en place une synchronisation de manière très simple : il suffit, selon la documentation, d'utiliser un « component Photon View » (figure 2). Ce composant permet d'envoyer tous les détails d'un objet le possédant à un serveur et ce dernier renverra tous ces détails aux autres joueurs étant dans la même partie que l'objet. Ces détails sont sa position, sa rotation, sa taille et sa vitesse. Nous avons donc ajouté ce composant à chaque objet que nous souhaitions synchroniser.

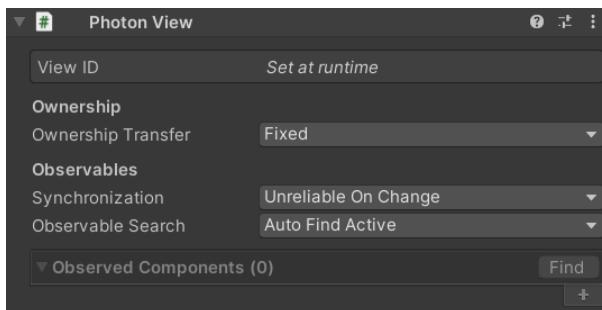


FIGURE 2 – Composant PhotonView

Cet ajout a été fait avant notre première soutenance et, sauf cas particuliers, aucun objet était synchronisé entre les joueurs (les cas particuliers étaient les objets réagissant au contact des joueurs comme les murs).

Nous avons donc recherché sur la documentation Photon et sur de nombreux sites et forums sans avoir trouvé de solutions. Nous avons testé un bon nombre de fois de changer certains paramètres, résultant uniquement à de faux espoirs. Puis, finalement, nous nous sommes rendus compte qu'il s'agissait d'un problème de « possession ».

Comme dit plus tôt, un objet possédant le composant « Photon View » envoie à un serveur des propriétés le concernant, mêmes propriétés qui seront transmises aux autres joueurs. Mais certaines données envoyées sont plus importantes que les autres. Il s'agit des données envoyées par le **possesseur** de l'objet. Si un objet est déplacé par un joueur ne « possédant » pas l'objet, il enverra quand

même les données au serveur. Mais si le joueur possédant l'objet déplace ce dernier, la position de l'objet sera la dernière envoyée au serveur par le possesseur et non par l'avant-dernier joueur l'ayant touché. Il nous a donc fallu transmettre la propriété d'un objet au joueur l'ayant touché pour la dernière fois. Nous avons donc changé la propriété « ownership transfer » du composant « Photon View » de « Fixed » à « Takeover ». Ainsi, tous les objets réagissant au contact des joueurs sont correctement synchronisés.

Il reste toutes fois les objets ne réagissant pas au joueur (comme les pendules). Comme chaque joueur charge une manche localement et que les obstacles font partie d'une manche, ils ne peuvent pas être synchronisés, même avec un composant « Photon View ». Il faut donc que l'on charge l'ensemble des obstacles uniquement lorsque le premier joueur entre dans la partie. Pour ce faire, nous avons utilisé la fonction *Instantiate()* de Photon, permettant de créer les obstacles en une seule fois par partie, les synchronisant.

Nous avons donc terminé d'implémenter la synchronisation dans notre jeu.

## 3.2 Intelligence artificielle

Il nous faut créer une intelligence artificielle (IA) « copiant » les actions des joueurs. Certaines auront des déplacements hasardeux, d'autres voudront du mal aux autres joueurs, d'autres encore iront finir le parcours le plus vite possible.

### 3.2.1 Fonctionnement

Pour commencer, l'IA ne doit pas emprunter le chemin le plus court vers la ligne d'arrivée, sinon, les joueurs n'auront aucune chance de gagner une partie. Il faut donc qu'elle suive un chemin moins précis. De plus, il ne faut pas que l'IA reste bloquée sur des obstacles du parcours. Il faut donc qu'elle puisse sauter ou bien éviter certains obstacles. Pour finir, comme les joueurs, les IA auront la possibilité de récupérer des bonus. Il faut donc que ces IA puissent les utiliser au meilleur moment.

Après avoir posé toutes ces caractéristiques, nous avons décidé d'implémenter un système de zones visibles seulement par les IA. Toutes les cartes posséderont des zones. Ces zones seront réparties sur l'entièreté des cartes, du début jusqu'à l'arrivée. Elles permettent aux IA de se diriger vers la ligne d'arrivée en passant à la fois les obstacles sans embûches et les joueurs leur voulant du mal.

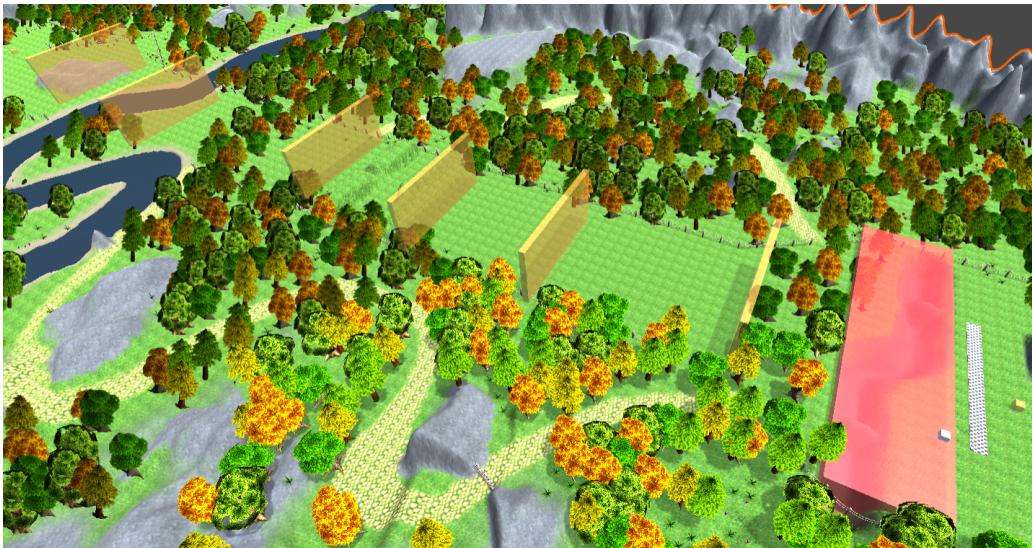


FIGURE 3 – Zones transparentes visibles uniquement par l’IA

### 3.2.2 Problèmes rencontrés

Avant la première soutenance, Mathéo a implémenté une intelligence artificielle se dirigeant vers un bloc en particulier. Cette IA n’était vouée qu’à certains tests que nous avons enlevés à présent. Lors du développement, nous avons rencontré quelques problèmes et même certains nous empêchant de faire bouger notre IA.

Un problème majeur que nous avons rencontré était les déplacements incontrôlés de notre IA vers une des zones. Nous utilisions l’utilitaire « Nav mesh » de Unity, permettant de déplacer un objet dans une scène en lui indiquant uniquement son point de destination et le tout en une seule ligne de code. Le revers de la médaille était le fait que les IA pouvaient se déplacer partout dans la scène, et donc à des endroits soit trop avantageux, soit pas assez. De plus, concernant les sauts des IA, ces derniers nous ont posés de nombreux problèmes. Il nous était impossible de bouger notre IA à la fois horizontalement et verticalement à cause du composant « Nav mesh ». Il fallait donc arrêter le programme permettant de faire bouger l’IA pour la faire sauter, impliquant l’utilisation de plus de variables et donc plus de mémoire pour un simple saut.

Nous sommes donc repartis de zéro et nous nous sommes servis du composant « Rigidbody » pour déplacer notre IA. Aujourd’hui, notre IA peut se déplacer vers une zone dans toutes les directions et elle peut sauter dès l’instant qu’elle se situe dans une zone obstacles. Son implémentation n’est pour l’instant pas complexe puisque celle-ci va être différente en fonction des parties.

### 3.2.3 Implémentation

Comme dit plus haut, nous avons utilisé le composant « Rigidbody » pour déplacer notre IA. Ce composant est utilisé par Unity pour simuler des forces sur les objets le possédant. Ainsi, pour déplacer l'IA, nous utilisons :

- La fonction *Rigidbody.MovePosition* : Cette fonction nous permet de déplacer l'IA vers une position donnée (un check point par exemple). Cette fonction « téléporte » notre IA vers cette position. Il nous faut donc calculer toute une série de points intermédiaires entre la position initiale de l'IA et son point d'arrivée pour produire des mouvements fluides. Nous avons utilisé la fonction *Slerp* donnant le point intermédiaire le plus proche. La fonction *MovePosition* s'occupe donc des déplacements horizontaux.
- La variable *Rigidbody.velocity* du Rigidbody de notre IA : Cette variable permet de simuler une vitesse pour l'IA par rapport à un vecteur donné. Elle nous permet ici de simuler un saut comme étant une force tirant l'IA vers le haut. Elle s'occupe donc des déplacements verticaux.

### 3.2.4 Mise à jour des zones

Nous avons implémenté des Zones de zones. Ces zones permettent de répondre à un de nos problèmes : la possibilité d'emprunter plusieurs chemins sur une carte. Si nous avions implémenté notre IA sans ces zones, lorsque ces dernières empruntaient des chemins annexes, elles ne pouvaient pas retrouver le chemin central ou bien elles se perdaient dans le décor. Lorsqu'une IA entre dans une de ces zones de zones, elle choisit au hasard un des check points dans ces zones pour s'y déplacer, et ce ainsi de suite jusqu'à la fin du parcours.

### 3.2.5 Implémentation des comportements d'IA

Notre première implémentation de comportements consiste en une attente entre plusieurs sauts lorsqu'une IA se trouve dans une zone obstacle. L'IA attend pendant un temps aléatoire pour simuler une réflexion et perfectionner des sauts. Pour la dernière soutenance, l'IA possédera tous les comportements énoncés dans notre premier cahier des charges.

## 3.3 Interface utilisateur

Comme évoqué précédemment et dans le cahier des charges, Mathieu est le responsable de l'interface utilisateur, il s'en est donc occupé et a même pris de l'avance sur les objectifs fixés pour la seconde soutenance. Tout d'abord, définissons ce qu'est l'interface utilisateur (UI). Pour faire simple, c'est tout ce qui

est conçu dans un dispositif d'information avec lequel une personne peut interagir. Il peut s'agir d'affichage, de claviers, d'une souris et de l'apparence d'un bureau d'ordinateur avec différents boutons. Elle est très importante pour mettre en forme le projet et rendre compréhensible le jeu.

Mathieu a d'abord dû s'occuper de mettre en forme la partie multi-joueur. Pour ce faire, il a créé différents menus. Les scripts *Menu* et *Menu Manager* ont alors été créés servant respectivement à initialiser le menu, à l'ouvrir et le fermer. Les menus sont les suivants : *Title*, *CreateRoom*, *FindRoom* et *Room*. Un menu s'active lorsqu'un bouton est pressé, à ce bouton est associé une fonction qui permet d'effectuer la tâche désirée. Le menu *Title* est actif dès le lancement du jeu et représente donc le menu principal qui permet de rentrer son pseudo et ouvrir les différents menus. Le menu *CreateRoom* permet de créer une salle en entrant le nom de celle-ci dans l'emplacement prévu à cet effet. Le menu *FindRoom* quant à lui permet de rejoindre une salle déjà existante. Pour finir, lorsqu'on a créé une pièce, on arrive dans le menu *Room* où sont affichés les différents joueurs présents dans la salle et le nom de celle-ci. C'est d'ici qu'on peut lancer la partie en cliquant sur le bouton *Play*.

En plus des différents menus évoqués, Mathieu a mis en place le choix du pseudo. Pour ce faire, il a fallu utiliser un *Input Field*, c'est-à-dire un champ de texte où l'on peut entrer le nom de notre choix. Ce nom a dû être stocké pour être placer au-dessus de la tête du joueur lors de la partie. Un script a alors été créé pour gérer tout ça et il est appliqué à chaque joueur.

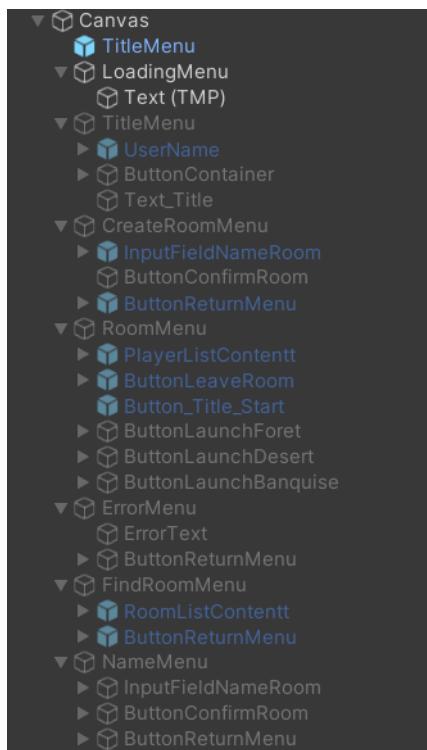


FIGURE 4 – Liste des canvas affichés au fur et à mesure

Une fois la partie lancée, on ne pouvait pas quitter la partie, Mathieu a alors mis en place un menu Pause qui permet de mettre en pause notre partie et de quitter la partie en cours.

Depuis la première soutenance, une fois que l’interface utilisateur était fonctionnelle et nous permettait d’accéder à une partie, il a fallu l’étoffer et trouver l’identité graphique de notre jeu. Celui-ci nous emmène dans différents univers comme évoqué auparavant, c’est pour cela qu’en arrière-plan Mathieu a animé une caméra pour montrer un aperçu de ces décors depuis le menu principal. Pour ce faire, il a dû apprendre à créer des animations afin de bouger la caméra au sein des décors miniatures qu’il a créées spécialement pour ça. De plus, pour mettre en accord l’environnement du jeu et les différents boutons, Mathieu a dû importer un asset avec différents choix de boutons et de design de menus. Ainsi, il a pu finir de mettre au goût du jeu l’UI tout en gardant les mêmes fonctionnalités que les précédents boutons à quelques exceptions près.

Depuis la dernière soutenance, on peut maintenant voir la place qu’occupe chaque joueur dans le jeu et son temps effectué sur le parcours. Deux scripts ont été créés à cet effet. Le script *TimerScript* et *CheckpointSystem* qui servent respectivement à calculer le temps de chaque joueur et à calculer la place de chaque joueur sur la partie. Pour cette dernière, il nous a fallu trouver un moyen

efficace de le faire et nous avons opté pour une solution qui consiste à calculer à intervalle régulier la distance entre chaque joueur et la ligne d'arrivée. Une fois ces distances ajoutées à une liste, il suffit de trier cette liste pour avoir le classement. Par la suite, Mathieu a donc pu afficher le tableau des scores au sein du menu pause où on peut voir la place, le temps et le nom du joueur. Le menu pause a donc évolué aussi, notamment avec l'ajout d'un onglet *Settings* où l'on peut choisir la résolution d'écran et régler le volume sonore du jeu.



FIGURE 5 – Menu Pause



FIGURE 6 – Menu Principal

### 3.4 Crédit des parcours

Nous avons commencé la création des parcours avant la première soutenance et cette tâche importante continue depuis et nous emmènera jusqu'à la dernière soutenance. C'est Lucas qui a commencé en premier la création de notre parcours forêt, il a trouvé un asset qui nous a beaucoup aidé, ce qui nous a poussé

à développer cette partie un peu plus tôt qu'initialement prévu sur le planning. Cependant, cet aspect de notre projet est voué à se développer jusqu'à la fin au travers de la création d'une multitude de parcours qui apportera une grande diversité à notre jeu.

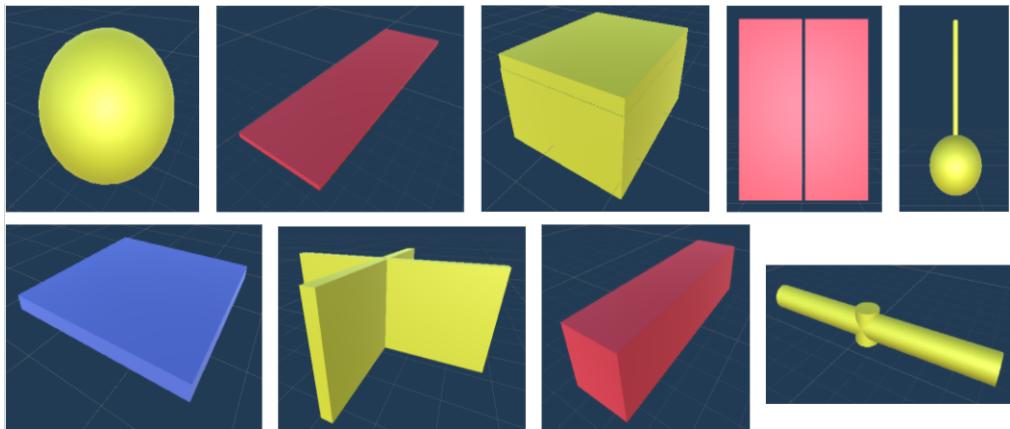


FIGURE 7 – Prefabs d'obstacles

Comme vous pouvez le voir ci-dessus, nous avons différents obstacles utilisables très facilement. La plupart d'entre eux proviennent de l'asset « Obstacle Course Pack » publié par AisuKaze Studio. Certains ont été adaptés pour notre utilisation tandis que d'autres, comme le tourniquet, ont été fait par Lucas. Ces obstacles peuvent être divisés en deux catégories : ceux utilisant la physique du moteur Unity n'ayant besoin daucun script et ceux étant en mouvement nécessitant un script pour les faire bouger.

Pour commencer, les obstacles utilisant la physique du moteur Unity sont les plus simples : il y a la sphère en haut à droite de la figure 7 qui se met en mouvement lorsqu'un joueur la pousse, le tourniquet qui se met à tourner en contact avec un joueur, les murs qui tombent dès qu'un joueur les traverse, etc. Ces obstacles ont tous trois composants : un *Mesh Renderer*, un *Collider* et un *Rigidbody*. Ce sont ces deux derniers qui permettent d'interagir avec le joueur et d'utiliser les phénomènes physiques que l'on connaît dans le monde réel.

Ensuite, il y a les obstacles étant en permanence en mouvement, ceux-ci nécessitent un script C# pour qu'ils puissent bouger, contrairement à la précédente catégorie, ils n'ont pas d'interaction avec le joueur en dehors du fait de le repousser. On peut par exemple citer la faucheuse en bas à gauche de la figure 7 qui tourne sur elle-même ou bien les murs vivants se déplaçant de gauche à droite.

Cet asset est très important pour notre jeu et nous a fourni des éléments essentiels que l'on a pu comprendre grâce à la documentation Unity. Les modi-

fications que l'on a faites et l'ajout d'obstacles ne nous a pas posé de problèmes particuliers.

Cependant, ces obstacles ne sont visuellement pas adaptés à nos environnements et nos décors. C'est donc Jules qui a pu adapter l'apparence de ces obstacles, grâce à des modèles 3D.

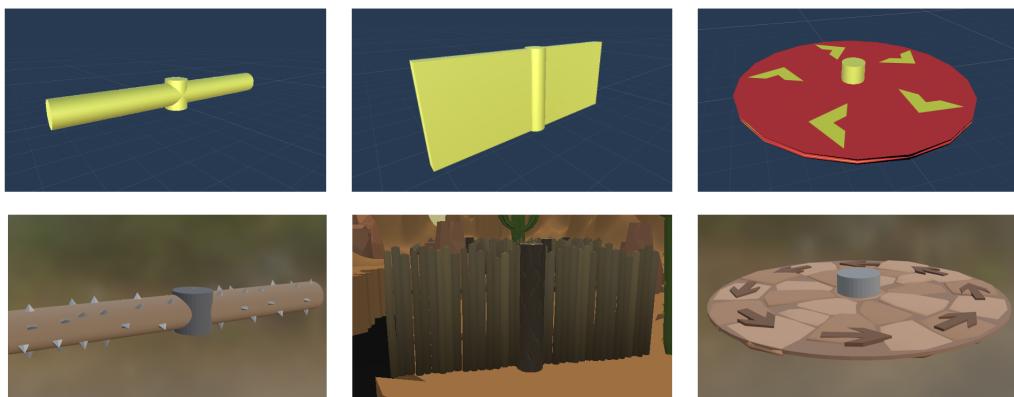


FIGURE 8 – Obstacles adaptés aux décors

Nous avons aussi ajouté des chemins alternatifs afin de rendre chaque partie unique. Grâce à cette collection d'obstacles, le travail de création de parcours consiste à les mettre bout à bout dans un ordre qui permettent au joueur d'avoir une expérience à la fois amusante et ni facile ni trop difficile. Cependant, cette tâche est difficile et nous prend beaucoup de temps. Nous avons tout de même choisi de réaliser nos parcours « À la main » plutôt que procéduralement car ceux-ci sont tous très différents et afin de correspondre aux différents décors, ce ne sont jamais exactement les mêmes obstacles qui sont utilisés.

### 3.5 Décor

Nous avons quatre décors déjà réalisés : forêt, désert, glacier et ville. La forêt et le glacier ont été réalisé par Lucas grâce à l'outil *Terrain* fourni par Unity. Il permet de créer un terrain et de complètement le personnaliser grâce à différents onglets permettant de gérer le relief, les textures et l'intégration des arbres. Le désert ainsi que la ville proviennent d'assets que Jules a importé sur le projet.

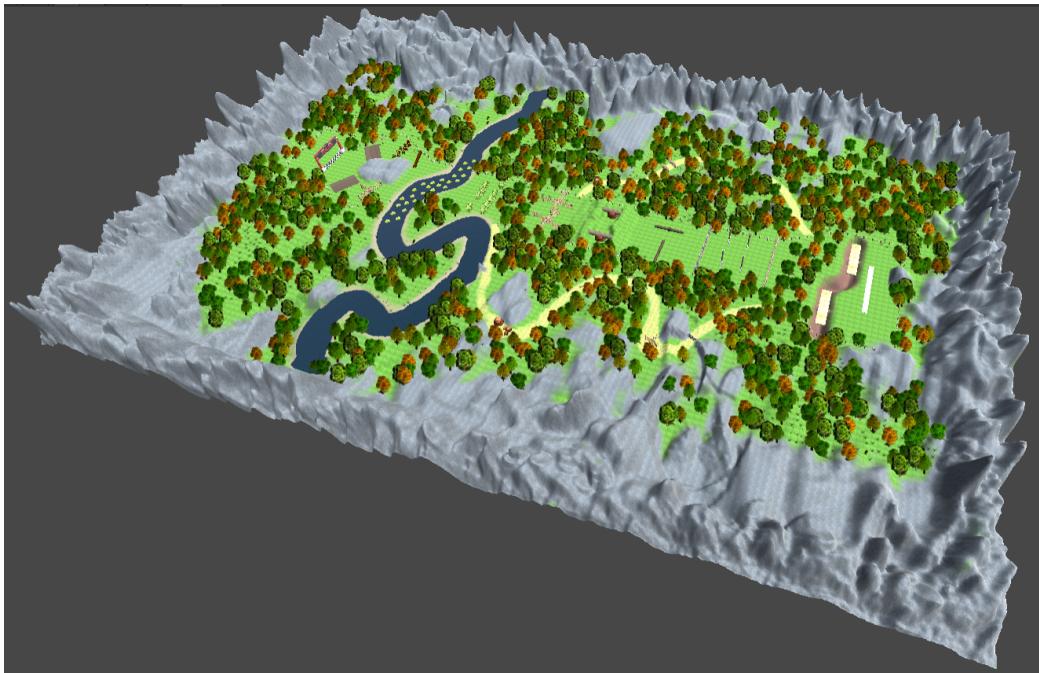


FIGURE 9 – Décor forêt



FIGURE 10 – Décor désert



FIGURE 11 – Décor glacier



FIGURE 12 – Décor urbain

### 3.6 Modélisation 3D

Au début du projet, nous n'avions aucune expérience dans le domaine de la modélisation 3D. C'est donc Jules qui s'est chargé de découvrir et d'apprendre cette compétence. Jules utilise le logiciel Blender, un logiciel gratuit et complet.

En parallèle, il a suivi et continue de suivre une formation sur YouTube de plus de 70 épisodes qui concerne la modélisation 3D, l'utilisation de Blender ainsi que les textures et le rendu final. Durant le développement du projet, Lucas a demandé à Jules plusieurs créations pour embellir le parcours forêt et il a aussi eu besoin d'objets spécifique pour faire son parcours désert. Certaines de ces créations sont présentes sur le jeu : les barrières, la rivière de la foret, l'arche de départ du désert, les arches d'arrivées de la forêt et du désert, les tourniquets, les buches piquantes, etc. Toutes ces créations permettent de rendre le jeu plus esthétique et d'intégrer les obstacles à l'univers du jeu.



FIGURE 13 – Barrières, buche, Faille et arches de départ et d'arrivée

### 3.7 Communication

Dès le début du projet, Jules a créé un compte Instagram sur lequel les avancées du projet sont régulièrement publiées. Afin de créer une communauté autour de notre projet, nous avons tous partagé ce compte sur nos réseaux personnels. Ce compte regroupe actuellement une centaine d'abonnés ce qui en fait l'un des projets de l'EPITA Rennes les plus suivis sur Instagram !

De plus, au début de la réalisation du jeu, Mathéo a commencé à créer le site internet du projet contenant une présentation de celui-ci, une bibliographie de tous les documents utilisés, un lien de téléchargement du jeu vidéo ainsi que la présentation des membres du groupe. Au fur et à mesure des soutenances, la page d'accueil se remplira de toutes les nouveautés ainsi que les bugs connus du jeu.

### 3.8 Choix de l'apparence du joueur

Pour une diversité des joueurs sur le terrain de jeu, Mathieu a mis en place un menu qui nous offre la possibilité de choisir entre 2 apparences de joueur différentes. Pour ce faire il a placé les deux personnages sur une scène puis les a animés à l'aide de tutoriels. C'est le script *SelectPlayer* qui gère tout ça. Chaque fois que l'on passe la souris sur l'un d'entre eux alors le *Collider* du joueur est détecté et le personnage est illuminé pour indiquer que l'on peut le sélectionner. Ainsi en cliquant dessus vous choisissez le personnage et vous êtes directement renvoyé au menu principal. Cet ajout a aussi été l'occasion pour Mathieu d'ajouter des sons à notre jeu chaque fois que l'on place la souris sur un personnage par exemple.



FIGURE 14 – Choix de l'apparence du joueur

### 3.9 Système des 3 manches

Le système des 3 manches par partie est une étape importante de notre jeu réalisé depuis la première soutenance. Lucas et Mathieu ont tout d'abord réussi à l'implémenter lorsqu'il n'y a qu'un seul joueur dans la partie. Cependant, lorsque Lucas et Mathieu ont tenté de synchroniser ceci avec le multijoueur, de nombreux problèmes sont apparus. Par exemple, attendre que l'ensemble des joueurs aient terminé la manche pour afficher le classement pendant quelques secondes puis, les déplacer dans une nouvelle scène qui se doit d'être la même pour tous les joueurs de la partie et que les mêmes joueurs se retrouvent dans cette nouvelle carte. Les 3 environnements sont choisis aléatoirement et il ne peut y avoir un même environnement joué 2 fois pendant la même partie. Après s'être confronté à cette liste non exhaustive de problèmes, ce système est finalement fonctionnel ! Lorsque des scènes intermédiaires affichant le classement seront créées et les erreurs corrigées, cette caractéristique phare de notre jeu sera terminé.

### 3.10 Interaction avec les objets

Comme dit plus tôt notre jeu consiste à faire un parcours d'obstacles le plus vite possible, mais ce serait ennuyant si on ne pouvait pas interagir entre joueur pour pouvoir ralentir la progression des autres sur le parcours. Il a fallut alors créer des objets que l'on peut porter et lancer sur nos adversaires. C'est Mathieu qui s'est chargé d'implémenter le script *PickableObject* qui nous permet de porter les objets et les lancer. Ce script doit par conséquent être rattaché à chaque objet que l'on veut pouvoir porter. Ainsi quand on se trouve proche de l'objet avec le script, la touche E nous permet de le porter, le clic droit de le reposer et le clic gauche de le lancer. Aujourd'hui le système est quasiment en place malgré quelques problèmes rencontrés lorsqu'on veut lancer l'objet devant nous.



FIGURE 15 – Tronc que l'on peut prendre

## 4 Récit de la réalisation

Depuis janvier, notre projet évolue constamment entraînant la mise en place d'une organisation particulière afin de faire avancer le projet. Pour répondre aux attentes et développer notre jeu, nous nous retrouvons en moyenne une fois par semaine afin de redistribuer certaines tâches. Ces rendez-vous sont aussi l'occasion de faire le point sur ce que chacun a fait de son côté, de s'expliquer les différentes parties pour être certain que chacun comprenne globalement ce qu'on fait les autres et de les aider sur les problèmes qu'ils peuvent rencontrer.

## 4.1 Nos peines

Ce projet n'est pas de tout repos, en effet il nécessite une grande part de notre temps de travail. Les problèmes rencontrés ainsi que la fusion des différentes tâches individuelles nous prennent une grande partie de notre concentration et nécessitent parfois beaucoup de recherches. Par exemple, pour le menu pause, le retour au menu comportait des problèmes de connexion qui ont coûté à Mathieu plus d'une semaine de travail consacré au jeu. La synchronisation des obstacles pour que les joueurs voient tous les mêmes mouvements des obstacles a aussi été la source de nombreuse recherche du côté de Mathéo. De plus, le logiciel Unity a récemment changé la gestion de la collaboration entre développeur en passant de Unity Collaborate à Plastic. Ce changement a considérablement ralenti notre travail pendant une semaine de mise en place et d'adaptation à ce nouveau système. En général, dû à l'avancée du projet, les problèmes que nous rencontrons sont de plus en plus durs à résoudre.

## 4.2 Nos joies

D'un autre côté, nous avons de plus en plus de compétences. Ce projet nous permet de développer nos capacités d'adaptation et nos connaissances dans plusieurs domaines insoupçonnés tels que la modélisation ou encore l'organisation et la pédagogie. Le travail en groupe nous permet de résoudre les problèmes plus rapidement en apportant de l'aide aux autres tout en leur expliquant ce qu'il ne va pas. Ensemble, nous faisons évoluer ce projet efficacement.

# 5 Avenir du projet

## 5.1 Apparence du joueur

Pour l'instant, nous n'avons le choix qu'entre deux apparences différentes. Nous comptons ajouter à cela au moins deux autres apparences afin d'augmenter la diversité dans les parties.

## 5.2 Création des parcours et des décors

Afin de diversifier l'expérience de jeu, de nouvelles cartes et de nouveaux obstacles vont faire leur apparition comme cela a été le cas entre la première soutenance et cette soutenance intermédiaire. Nous souhaitons, par exemple, développer un décor d'île où les joueurs devront prendre des moyens de locomotion disponibles aléatoirement afin de progresser sur la carte.

### 5.3 Interface utilisateur

L'interface utilisateur a bien été avancée. Nos deux dernières étapes seront :

- La création des tableaux de score afin d'avoir un récapitulatif des scores entre les manches ainsi que le classement général à la fin de la partie.
- L'ajout d'un écran de fin avec un affichage différent en fonction de si nous avons gagné ou perdu.

L'interface utilisateur va tout de même être améliorée pour la suite du projet. En effet, elle suit l'ajout de nouvelles fonctionnalités comme par exemple le mode solo, il va falloir mettre à jour cette interface pour rendre notre jeu plus agréable et utiliser ces différentes améliorations.

### 5.4 Mode solo

Pour la dernière soutenance, nous allons implémenter un nouveau mode de jeu : le mode solo avec fantôme.

Ce mode de jeu se déroulera comme sur une partie normale, à la seule différence que le joueur sera seul, comme son nom l'indique, contre un fantôme qui représentera le meilleur temps du joueur sur la carte où ils se trouvent. Ce mode de jeu implique donc la création d'un système de sauvegarde locale de l'ensemble des coordonnées du joueur pendant son meilleur temps lors d'une course. Ce mode de jeu implique aussi la création d'un nouveau skin fantôme unique.

### 5.5 Bonus

Avant la soutenance finale, nous souhaitons implémenter les bonus : des éléments récupérables directement sur le parcours. Ces bonus activeraient différents pouvoirs, tels qu'une augmentation de la vitesse de déplacements, de la hauteur de saut ou une diminution du temps final. Les prémisses de cette fonctionnalité ont commencées à être implémenter par Mathieu comme décrit précédemment. Cependant, nous souhaitons intégrer cela de manière plus discrète et pratique dans notre jeu, il y a aura différents objets qui auront des propriétés uniques afin d'apporter habileté et tactique à notre jeu. Par exemple, lancer un caillou sur quelqu'un pour l'étourdir et passer devant lui, ou utiliser une caisse afin d'accéder à un raccourci.

### 5.6 Site web

Notre site web est composé de 4 parties :

- Une page d'accueil permettant d'accéder aux autres pages

- Une page présentation du projet
- Une page de bibliographie ainsi qu'une présentation des membres
- Une page de téléchargement

## 6 Conclusion

Cela va maintenant faire 4 mois que la création du jeu *Best Of Ten* est lancé, nous avons acquis de nombreuses compétences et connaissances techniques dans la création de jeux vidéos à travers le moteur Unity. Au-delà de cet aspect, ce projet est également un grand pas en avant en terme d'autonomie, de travail en groupe et nous avons su mettre en place une démarche de projet en partant de rien.

Les conseils qui nous ont été donnés lors de la première soutenance nous ont aidé sur différents points auxquelles nous n'avions parfois pas pensé pour améliorer notre jeu, rendre celui-ci plus fluide et rendre son utilisation plus agréable. Nous remercions notre jury pour ces précieux éléments.

Ce projet se déroule à merveille dans l'ambiance de groupe. Effectivement, la cohésion de groupe est très efficace dans notre projet et nous permet de tirer vers le haut toute l'équipe en s'entraînant. Nous avons globalement respecté notre planning comme nous avions déjà réussi à le faire lors de la première soutenance et nous sommes prêts à implémenter toutes les nouvelles fonctionnalités pour la dernière soutenance.