



EPITA

PROJET JEU VIDÉO SUP S2

Rapport de soutenance 1



Projet : Best Of Ten

Groupe : Less An Year

Jules RAITIERE-DELSUPEXHE

Mathieu EVEN

Mathéo CRESPEL

Lucas BESNARD

Janvier 2022 - Juin 2022

Table des matières

1	Introduction	2
1.1	Les membres du groupe	2
1.2	Préambule du projet	2
2	Description des fonctionnalités du jeu	3
2.1	Origine du projet	3
2.2	Règles du jeu	3
3	Modifications du cahier des charges	4
4	Avancement du projet	5
4.1	Déplacements et vue à la troisième personne	6
4.2	Multijoueur	8
4.3	Intelligence artificielle	11
4.3.1	Acquisition de connaissances	11
4.3.2	Fonctionnement	12
4.3.3	Mode « obstacle »	13
4.3.4	Implémentation	13
4.4	Animations	14
4.5	Interface utilisateur	15
4.6	Création du parcours	16
4.7	Modélisation 3D	18
4.8	Décors	18
4.9	Communication	20
5	Récit de la réalisation	20
5.1	Nos peines	20
5.2	Nos joies	20
6	Avenir du projet	21
6.1	Apparence du joueur	21
6.2	Création de parcours	21
6.3	Interface utilisateur	21
6.4	Mode solo	22
6.5	Bonus	22
6.6	Utilisation des décors	22
7	Conclusion	22

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 la remise de notre cahier des charges.

Ce document contient : une description générale du jeu, de ses origines, de ses règles ainsi qu'un aperçu des fonctionnalités, une reprise de notre cahier des charges faisant suite à quelques modifications, 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 allons 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 souhaitons produire un jeu accessible à tous et rapide à comprendre. Cela sera 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 divisé 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 intervalles, 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 Modifications du cahier des charges

Nous maintenons l'ensemble des informations fourni dans le cahier des charges excepté la répartition des tâches qui a légèrement changé. En effet, après ces premières semaines de développement, les rôles se sont parfois inversés entre responsable et suppléant. Dans d'autres tâches comme la communication, c'est finalement Jules qui s'occupe du compte Instagram et Mathéo qui s'occupe du site web alors que ce devait être Mathieu. En outre, nous n'avions pas pensé à une partie non négligeable du projet : la modélisation 3D. C'est Jules qui en est le responsable et Mathéo son suppléant. Vous pouvez trouver ci-dessous le tableau corrigé.

	Mathieu	Jules	Mathéo	Lucas
Multijoueur	R	S		
Déplacements et caméra		R		S
Intelligence Artificielle			R	S
Interface utilisateur	R			S
Animations (Victoire, défaite et classement)			S	R
Modélisation 3D		R	S	
Parcours			S	R
Système des 3 manches et réapparition	R	S		
Sons	S		R	
Mode solo avec fantôme	S		R	
Bonus et utilisation du décor	S	R		
Communication (Instagram et site web)		S	R	

R : Responsable

S : Suppléant

FIGURE 1 – Répartition des tâches

La liste des tâches ayant évolué, le planning subit lui aussi quelques modifications. Les délais indiqués dans le diagramme de Gantt du cahier des charges sont inchangés, ce sont simplement des ajouts.

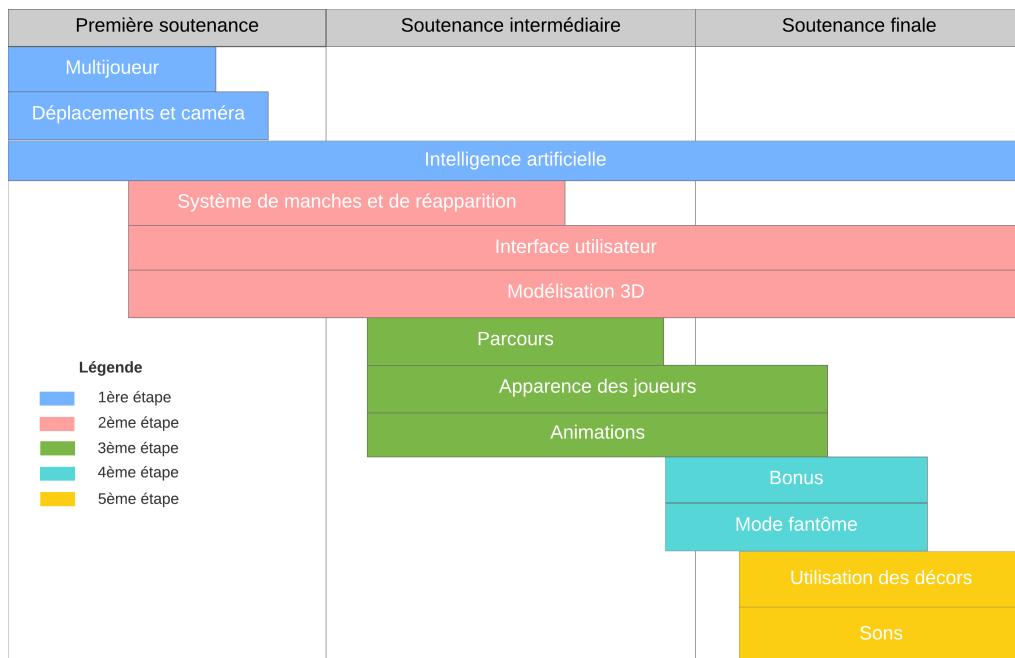


FIGURE 2 – Diagramme de Gantt

4 Avancement du projet

Dans son état actuel, le jeu se déroule de la manière suivante : on se connecte au réseau puis on arrive sur l'écran d'accueil où sont disposés trois boutons ainsi qu'un champ de saisie. Les boutons permettent soit 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 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, on affiche la liste des joueurs présents, 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, la 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 allant à gauche, reculer ou aller à droite à l'aide des touches ZQSD. Il peut aussi sauter grâce à la barre espace ou courir en maintenant la touche majuscule gauche. Nous pouvons aussi bouger la caméra autour du personnage avec la souris. Tous ces mouvements sont accompagnés des animations correspondantes.

Pour en revenir à la scène, c'est un décor de forêt entouré de montagnes

avec une rivière. Le parcours, délimité par des barrières, 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). Les intelligences artificielles, comme les joueurs, cherchent à finir le parcours au bout duquel se trouve l'arche d'arrivée qui lance une danse au personnage lorsqu'elle est franchie. Pendant toute la partie, on peut cliquer sur Échap ou P pour afficher le menu pause qui nous permet de quitter la partie pour revenir à l'écran d'accueil.

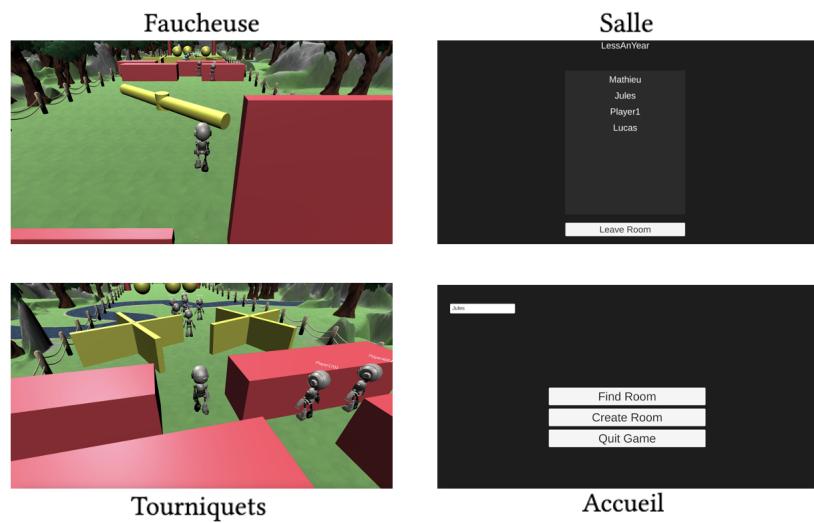


FIGURE 3 – Menus et obstacles

4.1 Déplacements et vue à la troisième personne

Comme nous l'avions explicité dans le cahier des charges, l'un de nos premiers objectifs a été d'avoir un personnage qui puisse se déplacer que ce soit en avant, en arrière, à gauche, à droite, mais aussi qui puisse sauter. Un autre de ces objectifs que nous avons développé en même temps est la vue troisième personne orientable grâce à la souris.

Jules et Mathieu ont rapidement découvert un pack tout-en-un qui comprenait la gestion des déplacements, le saut, la vue troisième personne et même des animations de personnages ! Il s'agit de l'asset « Starter Assets - Third Person Character Controller » directement fourni par Unity sur le *Unity Asset Store*. Ceci a ajouté un dossier complet à notre projet, nous n'avions plus qu'à glisser un Prebab sur la scène et c'était fonctionnel. Les touches étaient gérés grâce au nouvel Input System de Unity et la vue grâce à Cinemachine. C'était exactement

ce dont nous avions besoin donc nous l'avons conservé et continuer à développer le multijoueur ainsi que l'ébauche d'un premier parcours grâce à celui-ci. Toute l'équipe était satisfaite car c'était un bon début, nous avions des déplacements et une vue troisième personne parfaitement fonctionnel.

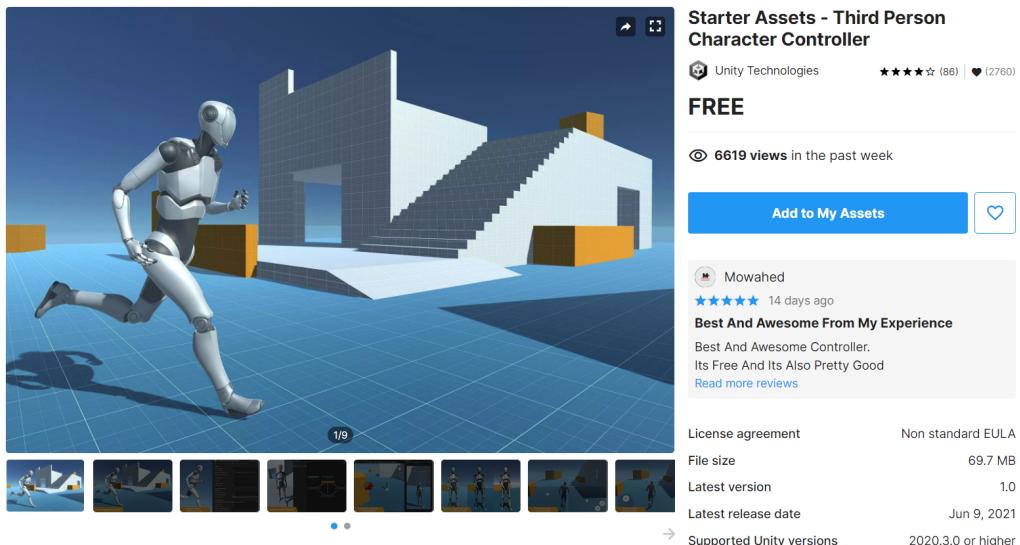


FIGURE 4 – Starter Assets - Third Person Character Controller

Malheureusement, la satisfaction a été de courte durée... Lors de l'ajout des premiers obstacles, des bugs sont apparus : les déplacements fonctionnait mais la caméra prenait des angles d'inclinaison étrange et semblait se heurter aux obstacles comme si elle était sur le parcours au même titre que le joueur. En essayant de résoudre ces problèmes lors d'une réunion de groupe hebdomadaire, nous avons rapidement conclu que nous ne comprenions pas comment fonctionnait l'asset et que cela allait être particulièrement handicapant pour la suite du projet. Lucas a alors décidé de se former sur le sujet, cela a été plutôt simple car il existe de nombreuses vidéos sur les déplacements d'un objet facilement fusionnable avec une vue troisième personne. Après quelques heures de formation, de compréhension et quelques recopies il faut l'avouer, Lucas a réussi à supprimer le dossier complexe de l'asset Unity pour simplement le remplacer par un prefab et deux scripts.

Découvrons globalement comment cette partie fonctionne. Il y a deux scripts : l'un pour le joueur, l'autre pour la caméra. Le prefab contient donc l'apparence d'un joueur (avec un rigidbody, un collider et son script) et une caméra (avec le composant caméra, ses dépendances et son script) comme vous pouvez le voir sur la figure ci-dessous.

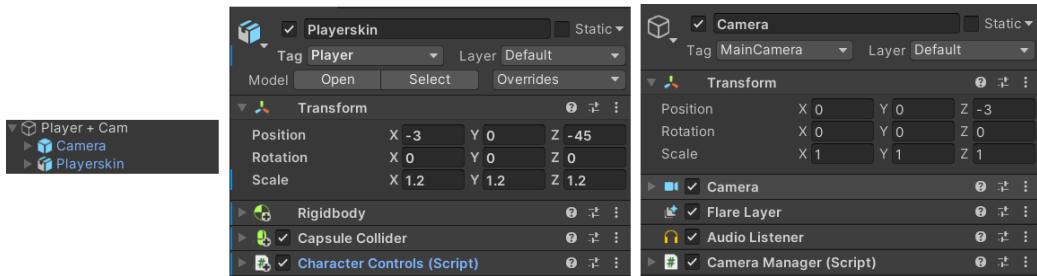


FIGURE 5 – Prefab du joueur et vue à la troisième personne

Pour les déplacements, on récupère l'état des touches via la fonction *Input.GetAxis* qui correspond aux touches ZQSD ainsi que les 4 flèches du clavier. On applique ensuite une force (*Rigidbody.Addforce*) sur le personnage en fonction de la touche pressé jusqu'à ce qu'il atteigne une certaine vitesse choisie. Le joueur peut marcher, mais a également la possibilité de courir en pressant la touche shift gauche, ainsi la force appliquée est multiplié par 1,8.

Pour le saut, on récupère l'état de la barre espace, si la barre est pressée une force verticale est exercé et le joueur se hisse en l'air. Grâce à la gravité, il retombe naturellement sur le sol.

Pour la vue troisième personne, il y a une caméra dont l'angle horizontal dépend de la position du curseur de la souris. Son rôle est de suivre à la trace une cible. Ainsi, elle récupère la position de sa cible en temps réel et se place à une certaine distance de celle-ci.

En conclusion, malgré un premier échec et la tentation de la simplicité, nous avons maintenant le contrôle et nous comprenons le fonctionnement des déplacements de nos joueurs et leur vue à la troisième personne. De plus, cette partie est complètement implémentée dans notre jeu. Grâce à ces péripéties, nous avons compris qu'il est essentiel de comprendre ce que l'on fait et que l'utilisation d'assets facile à prendre en main est souvent une mauvaise idée.

4.2 Multijoueur

Pour le bon fonctionnement de notre jeu, le multijoueur était également l'une de nos priorités. Nous avons donc cherché un moyen simple de l'implémenter. Plusieurs solutions étaient disponibles sur Internet, plus ou moins simples à comprendre et à implémenter. Mathieu et Jules se sont donc formés sur YouTube à l'aide de nombreuses vidéos sur le sujet. Dans un premier temps, notre choix s'est penché sur Mirror : une bibliothèque de Unity qui rend la mise en réseau facile et concise, le client et le serveur utilisent le même code. Nous avons donc

très rapidement compris comment cela fonctionnait et, à l'aide des tutoriels, son implémentation a été simple et efficace.



FIGURE 6 – Mirror

Voyons comment cela fonctionne, il y a deux scripts : un Network manager directement fourni avec Mirror qui permet la connexion au réseau et un autre pour désactiver les éléments des autres joueurs. Il nous a fallu ajouter ces scripts à la préfab du joueur pour que chaque joueur puisse se connecter correctement. Cependant, après avoir tout mis en place, c'est-à-dire la connexion au réseau et la synchronisation entre les joueurs sur la partie, nous nous sommes rendu compte que Mirror ne prenait pas en charge de serveurs dédiés, c'est à dire que l'on pouvait se connecter à plusieurs uniquement en passant par le réseau local.

Mathieu a alors dû se renseigner sur le sujet, malgré le fait qu'il y avait peu d'informations. Après de nombreux essais pour établir une connexion sur un serveur, lors d'une réunion de groupe, nous avons décidé de changer de système de mise en réseau. Nous avons alors dû recommencer de zéro notre projet pour repartir sur une bonne base. Le système le plus utilisé était Photon, un pack Unity disponible sur l'Asset Store pour le multijoueur. Bien qu'il soit plus difficile à prendre en main, 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 en fonction de notre région et permet donc de se rejoindre depuis Internet sur des salles créées par les joueurs eux-mêmes.



FIGURE 7 – Photon

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 8 – Différentes fonctions de Photon

Cependant nous avons quand même rencontré quelques difficultés. En effet, l'implémentation du multijoueur implique une interface utilisateur complète pour gérer cela dans le jeu. Mathieu en est le responsable et a eu quelques problèmes notamment lorsqu'il a réalisé notre menu pause. Lorsque nous quittions la partie pour revenir au menu principal, la connexion au serveur était difficile. Il a eu du mal à comprendre pourquoi, mais à force de persister il a réussi à résoudre le problème. La synchronisation des joueurs a aussi posé problème, en effet la manière de synchroniser était différente que sur Mirror et il nous a alors fallu du temps pour comprendre son fonctionnement. Celui-ci est pourtant simple, il suffit d'ajouter le composant *PhotonView* qui permet donc de synchroniser tous les éléments qui le possède lors de la connexion à un serveur. Le multijoueur a tout de même pu être implémenté correctement dans son intégralité.

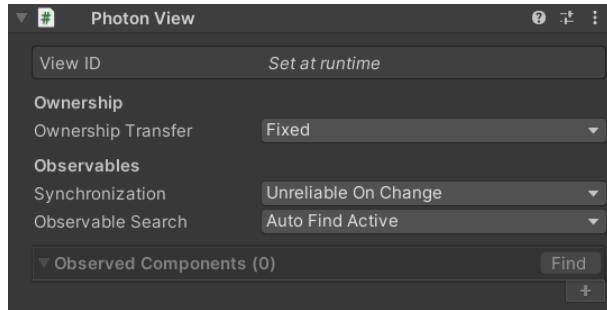


FIGURE 9 – Composant PhotonView

4.3 Intelligence artificielle

L'un des principes parmi les plus importants de notre jeu réside dans son nom : *Best Of Ten*. Comme expliqué dans la description des fonctionnalités du jeu dans la sous-section 2.2, chaque partie comporte obligatoirement 10 joueurs, d'où le *Ten*. Ainsi, lorsqu'un joueur entre dans une salle pour jouer à une partie, il est obligé d'attendre que sa salle comporte exactement 10 joueurs avant de commencer. Pour éviter des temps d'attente trop long, il nous est nécessaire d'implémenter un compte à rebours qui débute une partie lorsque ce dernier s'arrête. Par conséquence, pour respecter la règle des 10 joueurs, nous devons implémenter une intelligence artificielle (IA). Cette IA permettra de simuler un vrai joueur et sera donc capable de reproduire tous les déplacements d'un joueur humain, d'utiliser des objets et des bonus et surtout, elle aura autant de chances de gagner qu'un joueur humain.

Ainsi, il nous faut créer une 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.

4.3.1 Acquisition de connaissances

Pour implémenter cette IA, Lucas et Mathéo sont partis de zéro. Ils ne savaient pas comment déplacer des entités sur une carte et donc ils ne savaient encore moins comment adapter leurs actions au terrain et aux autres joueurs se déplaçant autour. Pour pallier cela, Mathéo a trouvé une formation gratuite fournie par Unity permettant d'apprendre les bases du développement des intelligences artificielles sur le moteur de jeu. Parallèlement, Lucas et Mathéo ont fréquemment échangé leurs idées de développement et d'implémentation.

À la suite de cette formation, ils se sont rendu compte que l'ensemble de son contenu concernait des sujets très généraux et donc elle n'allait pas leur servir pour construire l'IA. Ils se sont donc concertés afin de créer une IA dont

le fonctionnement est détaillé juste en dessous.

4.3.2 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.

Nous avons défini plusieurs types de zones, servant à élargir le panel d'actions possibles par les IA :

- Les deux premiers types sont la zone de départ et la zone d'arrivée, objectif à atteindre de l'IA. Ces zones n'ont pour l'instant pas d'utilités majeures.
- Le deuxième type de zone est la zone checkpoint (block vertical en orange transparent sur la figure 10). C'est grâce à cette zone que les IA pourront réapparaître en cas de chutes et c'est aussi un but intermédiaire de destination pour les IA.
- Le troisième type de zone est le plus important : les zones de passage. Ces zones permettent à l'IA d'avancer vers la prochaine zone checkpoint ou bien l'arrivée. Lorsqu'une IA entrera dans une de ces zones, elle recevra des informations sur le chemin et la direction à emprunter pour aller jusqu'à son but.
- Le quatrième type de zone est la zone obstacles (block horizontal en rouge transparent sur la figure 10). Elle sera présente dès l'instant qu'un obstacle sera présent sur la carte. Lorsqu'une IA entrera dans une de ces zones, leur mode « obstacle » va être activé.
- Le dernier type de zone est la zone décor qui sera la moins utilisée. Cette zone permet aux IA y entrant de prendre des objets du décor pour les lancer sur les joueurs.

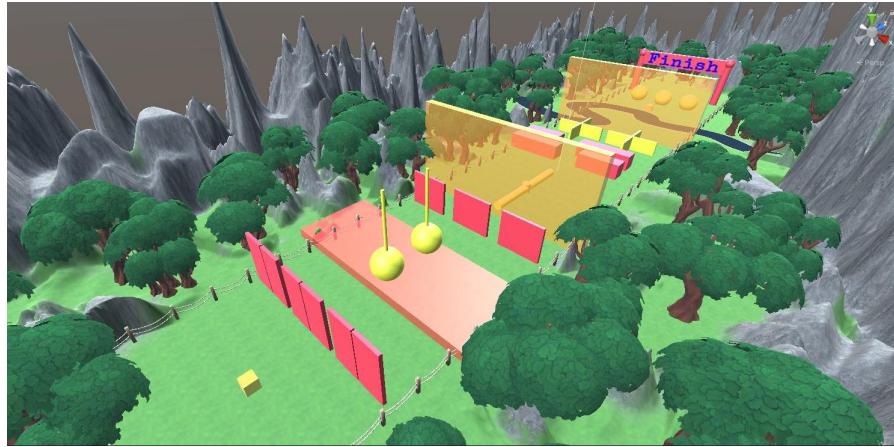


FIGURE 10 – Zones transparentes visibles uniquement par l’IA

Ici, peu importe la zone où se trouve l’IA, elle ira forcément là où la dernière zone de passage qu’elle a rencontré lui a dit d’aller.

4.3.3 Mode « obstacle »

Concernant le mode « obstacle », celui-ci permet de choisir des actions à réaliser par les IA lorsqu’elles entrent dans une zone obstacle :

- La première action possible est « **JUMP** ». Cette action permet à l’IA de franchir tous les obstacles qu’elle rencontre pendant un court laps de temps.
- La deuxième action possible est « **WAIT** » faisant attendre l’IA pendant une courte durée.
- La troisième possible est « **USE** ». Cette action permet aux IA d’utiliser leurs bonus intelligemment.
- Enfin, la dernière action possible est « **PUSH** ». Cette action permet à l’IA de pousser intentionnellement les joueurs autour d’elle.

4.3.4 Implémentation

Après avoir défini toutes ces zones, nous sommes donc passés à l’implémentation de cette IA. Aujourd’hui, Mathéo a fait en sorte que les IA, lorsqu’elles apparaissent, aillent directement vers un bloc particulier. De plus, comme toutes les zones ont été créées, il ne reste plus qu’à Lucas et Mathéo de fusionner cette IA avec le reste du travail.

4.4 Animations

Comme évoqué dans la sous-partie « Déplacements et vue à la troisième personne », c'est finalement Lucas qui s'est occupé de l'implémentation de celle-ci. Conformément à la répartition des tâches du cahier des charges, c'est aussi lui qui est en charge des animations. N'ayant aucune idée du fonctionnement des animations, il a repris une version du projet où nous utilisions l'asset « Starter Assets - Third Person Character Controller » qui gérait les animations du personnage.

Dans Unity, les animations des personnages sont gérés par un composant nommé *Animator* attribué à l'objet que l'on souhaite mettre en mouvement. Il a besoin d'un *Animator Controller* ainsi que de l'avatar du joueur à animer. Ce dernier contient les points d'articulations et diverses informations sur l'apparence du joueur à animer. L'*Animator Controller* permet de lier les différentes animations du personnage, il possède des paramètres accessibles et modifiable depuis les scripts C# permettant de déclencher les différentes animations.

Vous pouvez voir sur la figure 12 le contrôleur que nous utilisons. La case orange intitulé « Idle Walk Run Blend » est un *Blend Tree*, il permet de mélanger des animations en fonction d'une variable. Dans notre cas, le *Blend Tree* est lié à trois animations : repos, marche et course. Un nombre flottant variant entre 0 et 10, défini dans le script gérant les déplacements du joueur, permet d'avoir des animations visuellement cohérentes avec la vitesse du joueur. 3 animations (JumpStart, InAir et JumpLand) permettent de gérer le saut du joueur et l'animation Flair correspond à la danse de fin.

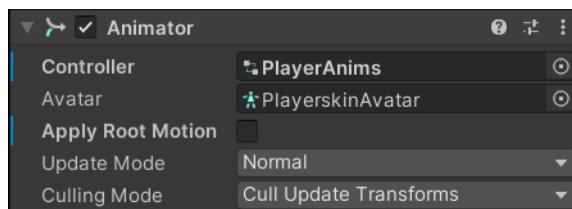


FIGURE 11 – Composant *Animator*

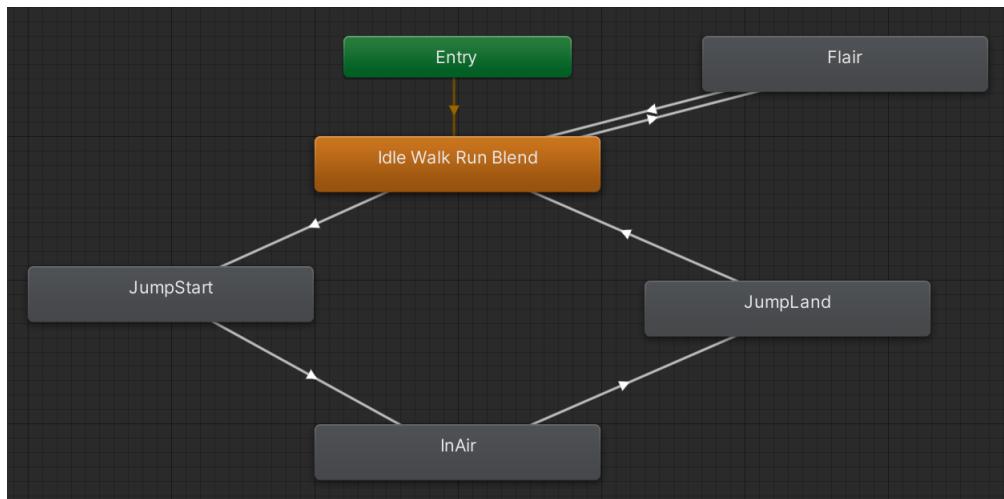


FIGURE 12 – *Animator Controller* utilisé dans notre jeu

Dans ce travail, Lucas s'est occupé de réaliser l'*Animator Controller* et de l'utiliser à travers les différentes variables dans le script de déplacement du joueur. Les animations en elles-mêmes proviennent du site internet [mixamo . com](http://mixamo.com) qui possède un large choix de mouvements disponible gratuitement. Le plus difficile a été la gestion du déclenchement des animations au bon moment dans le script. L'*Animator Controller* n'étant composé que d'animations déjà créées et de transitions avec des conditions entre celles-ci, son fonctionnement a été rapidement et facilement compris grâce à des tutoriels Youtube.

4.5 Interface utilisateur

Comme évoqué précédemment et dans le cahier des charges, Mathieu est le responsable de l'interface utilisateur, il s'en ai donc occupé avec l'aide de Jules et a même pris de l'avance sur les objectifs fixés pour la première 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 et Jules ont d'abord dû s'occuper de mettre en forme la partie multi-joueur. Pour ce faire, ils ont créé différents menus. Les scripts *Menu* et *Menu Manager* ont alors été créés servant respectivement à initialiser le menu et 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 *Launch Game*.

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 appliquer à chaque joueur.

Une fois la partie lancé, on ne pouvait pas quitter la partie, Mathieu a alors mis en place un menu Pause qui permet de mettre en pause notre partie. Depuis celui-ci deux choix s'offrent à nous pour le moment, soit on reprend la partie en cliquant sur le bouton *Resume* ou alors on peut quitter la partie en cliquant sur le bouton *LEAVE ROOM* qui va nous renvoyer vers le menu principal.



FIGURE 13 – Menu pause et menu principal

4.6 Crédit du parcours

La création du parcours fait partie de la troisième étape définie dans le cahier des charges, elle devait commencer après cette première soutenance. Or, nous nous sommes rendu compte qu'une carte complètement plane sans obstacles n'était pas très intéressante. De plus, Lucas a trouvé un asset qui nous a beaucoup aidé ce qui nous a poussé à développer cette partie un peu plus tôt.

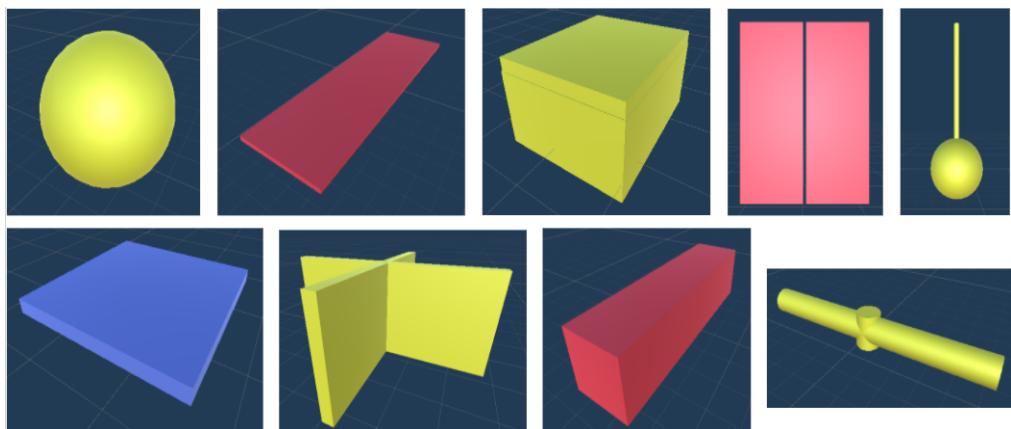


FIGURE 14 – Prefabs d’obstacles

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 d’aucun 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 14 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 14 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 modifications que l’on a faites et l’ajout d’obstacles ne nous a pas posé de problèmes particuliers.

4.7 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. Certaines de ces créations sont présentes sur le jeu : les barrières, dont la modélisation et la texture ont été faite par Jules, et la rivière, dont il a modélisé seulement la forme, car il n'a pas encore étudié les textures en mouvements. Par ailleurs, il a déjà modélisé la forme d'une arche d'arrivée, et il est actuellement en train de travailler les textures de cette dernière. Il a aussi créé le BOTCoin, qui sera utilisé comme bonus de temps directement récupérable sur le parcours.



FIGURE 15 – Barrières, arche d'arrivée et BOTCoin

4.8 Décors

Nous avons deux décors déjà réalisés : forêt et désert. Le premier a 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 second environnement (désert) provient d'une asset que Jules a importé sur le projet.

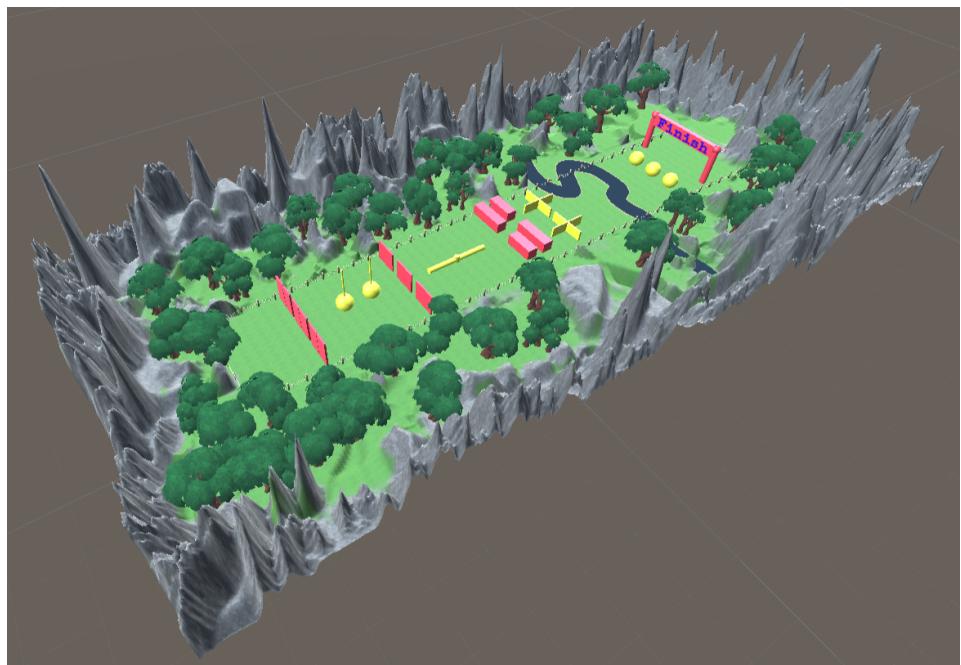


FIGURE 16 – Décor forêt

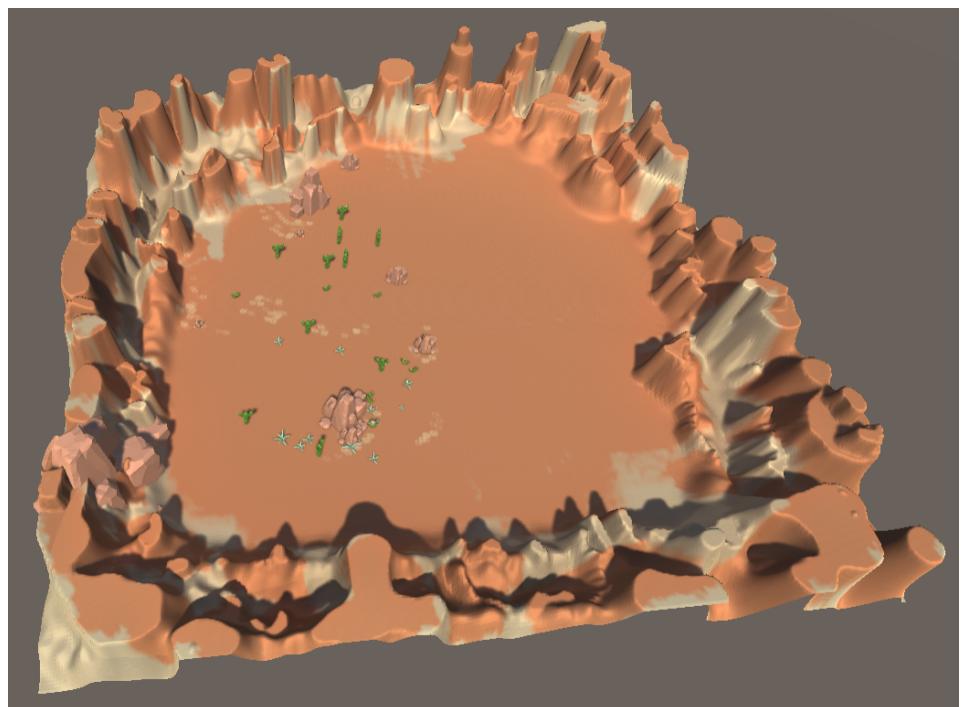


FIGURE 17 – Décor désert

4.9 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.

5 Récit de la réalisation

Depuis la validation du cahier des charges, notre projet a pris forme, 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 au moins 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.

5.1 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.

5.2 Nos joies

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.

6 Avenir du projet

6.1 Apparence du joueur

Lors de nos prochaines heures de développement, les apparences des joueurs sont amenées à changer. Nous avons prévu de créer de nouvelles peaux (skins) pour les joueurs grâce à la modélisation 3D ainsi que l'utilisation de logiciels pour créer les textures.

Pour la deuxième soutenance, nous avons prévu de créer au moins 2 skins qui seront disponibles lors d'un choix via l'interface utilisateur dans le menu d'accueil du jeu.

6.2 Création de parcours

Afin de diversifier l'expérience de jeu, de nouvelles cartes et de nouveaux obstacles vont faire leur apparition. Nous allons par exemple développer un environnement en ville où le joueur devra, en partie, esquiver la circulation routière.

Pour la seconde soutenance, nous souhaitons vous présenter au moins 2 parcours différents : la forêt déjà faite, le désert et nous espérons développer une carte dans des montagnes enneigées.

6.3 Interface utilisateur

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

Le fond noir présent sur chaque menu va changer par la suite pour être remplacé par quelque chose qui correspond mieux à l'image de notre jeu.

Les fonctionnalités comme le classement des joueurs et par conséquent la place et le temps de chaque joueur sera visible depuis le menu pause ainsi que la place et le temps individuel sur l'écran du joueur en question. Pour finir, l'ajout d'un choix de personnage va engendrer une modification de l'interface utilisateur avec de nouveaux boutons et de nouveaux affichages.

En conclusion, l'interface utilisateur n'est pas définitive et peut changer à tout moment, notamment à chaque ajout de fonctionnalité. Même si elle est déjà bien avancée, il reste à la rendre plus esthétique.

6.4 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.

6.5 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.

6.6 Utilisation des décors

Pour terminer notre jeu, nous aimerais rendre possible l'utilisation du décor, ce qui apporterait 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.

7 Conclusion

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 respecté notre planning et sommes prêts à implémenter toutes les nouvelles fonctionnalités pour la deuxième et la dernière soutenance.