ECE Steal Game

Laure Andouy - Ratsada Manivong - Elise Malard

Hugues Puiseux - Pierre Maïsterrena - Valentin Malomsokie

Projet de mineur de Création numérique

Table des matières

Introduction

Lors du premier semestre de la mineure des métiers de la Création Numérique, nous avons choisi de s'orienter vers la création d'un jeu vidéo comme tout le groupe était plutôt attiré par ce domaine. Après une séance de brainstorming, nous avons choisi le thème de notre jeu vidéo (qui partait d'une blague entre nous) : Le vol d'objet à l'ECE.

Règle du jeu :

But : Voler un maximum d'objet de l'ECE pour gagner un maximum de points et d'expériences sans vous faire remarquer par les surveillants. Vous aurez la possibilité de vous faire aider par d'autres élèves mais attention, certains peuvent vous dénoncer et vous faire perdre des points.

Mode Quête

Lancer une quête en allant parler à Toto le PNJ, pour qu'il vous donne une mission à réaliser. Si vous arrivez à satisfaire sa demande, vous gagnez de l'expérience et pourrez monter de niveau.

Mode Libre

Baladez-vous au sein du campus de l'ECE PARIS comme vous le souhaitez, et profitez-en pour voler les objets que vous voulez sans vous faire remarquer !

Après avoir établi les règles et fonctionnalités qu'on voulait implémenter dans notre jeu, nous nous sommes réparti les tâches selon les préférences de chacun.

Répartition des tâches :

Elise / Laure	Valentin	Rathsada	Pierre	Hugues
- Modélisation 3D du décor sur Unity - Gestion des collisions et des propriétés physiques	- Scripts : attraper + lancer objets - Mécanisme de points + zone - Bugfixes et QoL	- Création joueur, NPC, Canvas dialogue - Management des dialogues - Lancement quête (avec Valentin)	- Création des scripts et de l'UI des menus - Vision globale et mise en place du projet	- Composition de la musique d'ambiance - Intégration de tous les éléments sonores

Partie Pierre

1. Lancement du projet

Au début du projet, j'ai eu un rôle de chef de projet notamment en encadrant les sessions de brainstorming qui se sont révélées très riches et en évaluant et répartissant les tâches à effectuer en fonction des affinités de chacun. Je suis familier au milieu du jeu vidéo et je m'étais déjà intéressé à leur conception et à leur réalisation. C'est pourquoi j'ai pu dès le début, et avec le soutien de Valentin qui a également de nombreuses connaissances dans le domaine, évaluer les différents aspects du jeu (musique, code, graphismes, game design et scénario) afin de les regrouper dans un fichier Excel et de demander à chaque membre du groupe sur quelle partie, ils préfèrent travailler. Ensuite, en fonction des affects de chacun, j'ai proposé une répartition des tâches qui a été validée par l'ensemble du groupe.

	Description	Elise	Valentin	Rathsada	Laure	Hugues	Pierre
Recherche	Mettre tout en place avant de commencer (définir le thème et ambiance général du jeu pour adapter musique et modélisation, mettre en place un CDC précis)	TOUS					
GameDesign				x			v
Musique		x		x			
Sons					v		x
Code				v	v	x	
Modélisation	Blender 3D	v			v		
Scénario		v		v	v		x

Tableau des affinités de chacun en fonction des tâches à réaliser

2. Les premiers tutoriels

Je me suis ensuite concentré sur Unity avec Valentin, je me suis renseigné sur les premiers tutos afin de prendre en main le logiciel. J'ai trouvé le tutoriel « FPS Microgame » qui permettait de créer un FPS (First Person Shooter – Jeu de tir à la première personne) basique avec quelques fonctionnalités comme des ennemies, des tirs, des armes, un boss, etc. Finalement, nous avons décidé de partir de 0 pour notre jeu, mais ce tutoriel nous a beaucoup aidés pour nous familiariser avec Unity. J'ai donc suivi ce tutoriel ainsi que celui permettant de créer un système de quêtes basé sur des scripts (je n'ai pas eu besoin de coder, les scripts étant déjà faits, j'avais juste à les assigner aux bons objets.) ainsi que d'autres tutoriels de base qui m'ont appris à changer la couleur d'un objet, en rajouter un à la scène, des connaissances du base d'Unity.

3. L'UI et les menus

Je me suis ensuite concentré sur l'UI du jeu, je me suis intéressé aux différents scripts nécessaires à un menu principal : lancer le jeu, quitter le jeu et passer au menu des options, ainsi que les scripts du menu des options : mettre le jeu en plein écran ou non, changer le volume général du jeu et permettre au joueur de choisir la résolution du jeu en fonction des différentes résolutions disponible sur son ordinateur. J'ai également suivi des tutoriels sur la gestion des scènes afin de pouvoir passer du menu principal au jeu, du jeu à l'écran de game over, et de cet écran au menu principal. J'ai aussi suivi un tutoriel pour créer ses menus au sein du jeu (faire les boutons, personnaliser le texte, la barre de volume, etc.)

Menu Principal de ECE-Steal

4. Vision globale du jeu

Pendant l'ensemble de mon travail, je me suis également renseigné sur le travail des autres membres de l'équipe afin de pouvoir avoir une idée de ce qu'ils faisaient, me permettant ainsi d'avoir une vision plus globale du projet et de la création d'un jeu vidéo en général. Je me suis notamment beaucoup intéressé au travail de Valentin, qui a acquis de solides compétences sur Unity et en particulier sur les scripts qui permettent d'allier le code et les éléments du jeu.

5. Conclusion

Pour conclure, j'ai donc beaucoup apprécié ce projet, il m'a permis de découvrir les différents aspects de la création de jeux vidéo, mais il m'a aussi fait rendre compte que je préfère travailler dans la conception de jeux vidéo plus que dans leur réalisation en elle-même. Je préférerai donc avoir une expérience proche d'un game designer ou d'un manager plus que d'un développeur ou d'un UI designer. Cette prise de conscience s'est faite grâce à ce projet et c'est pourquoi je suis très content d'avoir pu y participer.

Partie Elise – Laure

Afin de réaliser la modélisation 3D de notre jeu, soit le décor et environnement de notre jeu : ECE Steal, nous avons procédé de la manière suivante :

1. Définition du plan

Nous avons pris plusieurs mesures à E4 (tailles des murs et des portes, espacement des tables, emplacement des objets). Pour perfectionner notre modélisation nous avons pris plusieurs photos de E4 sous différents angles., Ces photos nous ont servi d'exemple pour notre modélisation. Nous avons ainsi réalisé un schéma et défini les éléments que nous voulions incorporer au décor du jeu.

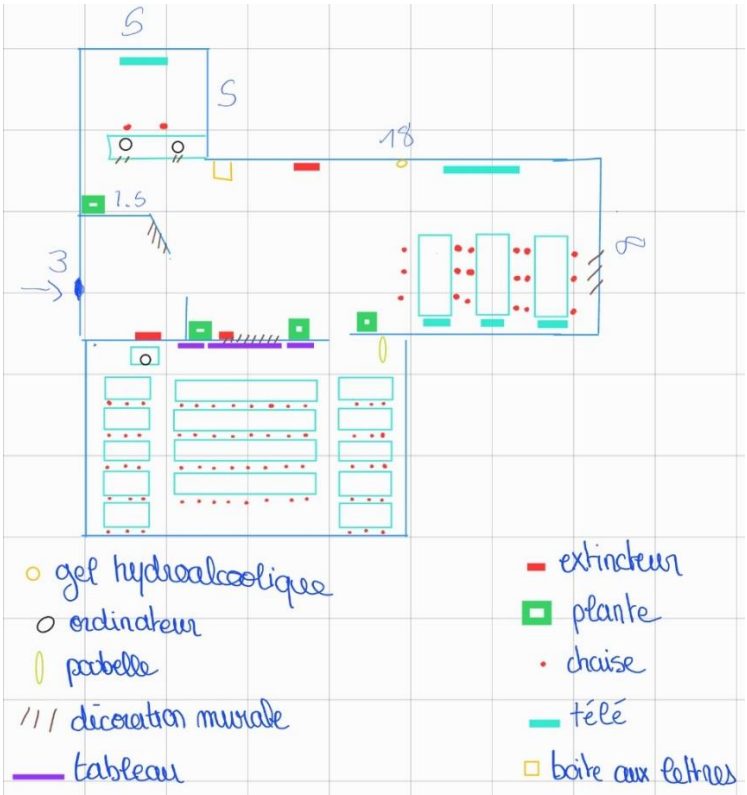

Figure 1: Plan du hall de E4 réalisé à partir de photos et de mesures

Nous voulions dans un premier temps modéliser chaque objet sur Blender. Après réflexion nous avons préféré utiliser la bibliothèque 3DWarehouse conseillée par l'un de nos professeurs. Avec l'aide de Rathsada, nous avons choisi quels objets nous voulions importer en premier après avoir créer les murs et le sol. Dans un premier temps, nous avons récupéré des *model.dae* de la bibliothèque pour modéliser des chaises et des tables.

2. Modélisation sur Unity

La prochaine étape était d'importer nos *models* dans Unity, mais pour se faire il nous fallait un terrain de jeu. Nous avons donc suivi les tutoriels proposés par Unity afin de nous familiariser avec les outils. Ces tutoriels nous ont permis de réaliser un sol encadré de 4 murs que nous avons adapté au plan de E4. Les couleurs des murs (*materials*), les propriétés physiques (*box colliders*) et les dimensions ont été entièrement géré sur Unity. Cependant, les objets étaient particulièrement difficiles à positionner et à dimensionner car chaque objet possède sa propre échelle et son propre repère.

- Murs, sols et plafonds : nous avons utilisé les objets 3D "box" et les avons regroupés par famille, dans une d'arborescence parmi les assets du *Level 1*.

Figure 2: Arborescence du Level 1

- *Materials* : Nous avons créé des matériaux pour la texture/couleur de nos éléments 3D. Ainsi ou peut apercevoir sur la photo ci-dessous des *materials* pour les murs et pour les objets.

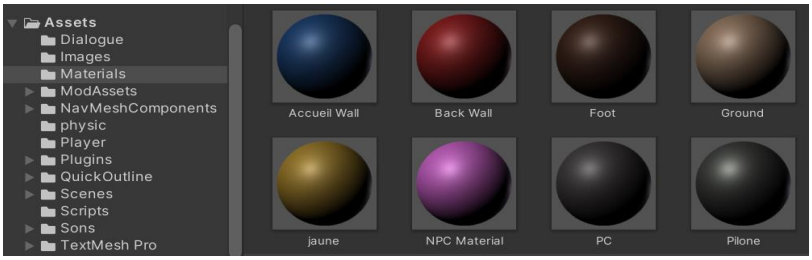

Figure 3: Materials utilisés pour notre projet

- *Box colliders* : pour interagir avec les objets, c'est-à-dire pouvoir les attraper, les lancer et ne pas traverser les objets, nous avons ajouté des *box colliders* sur tous les objets n'en possédant pas initialement. Le sol, les murs et le plafond en avaient à la bonne dimension par défaut : c'est l'une des propriétés de certains objets 3D Unity. Les objets importés par format .dae n'avaient pas de box colliders et le joueur passait au travers. Nous avons donc ajouté un élément à chaque objet et l'avons dimensionné en fonction des proportions de l'objet. Sur la photo si dessous le collider apparait en vert sur la table :

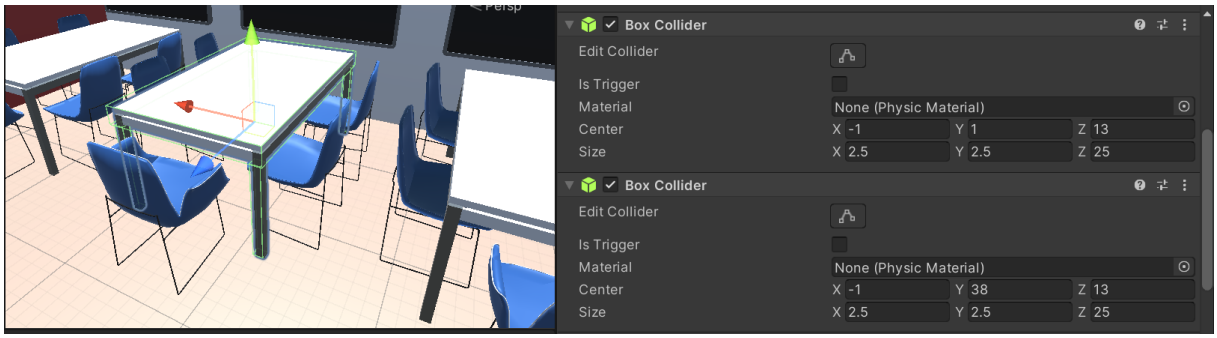

Figure 4: Box colliders d'une table

- Objets : pour perfectionner notre décor nous avons ajouté une vingtaine d'objets tels que des tables, chaises, télés, un sac, des plantes, une poubelle etc. A chacun de ces objets nous avons ajouté des couleurs et des colliders pour les rendre plus réaliste.

3. Features supplémentaires

Initialement, il était prévu d'ajouter une salle de classe au décor du jeu. Malheureusement, le manque de temps et de maîtrise de Unity ne nous a pas permis de le faire. Nous avons donc décidé de ne garder que le hall d'entrée de E4, ne s'agissant de toute manière que d'une version "bêta" servant principalement à tester nos scénarios, notre joueur et le jeu de manière générale. L'ajout d'autres salles n'était pas en soit plus difficile : il s'agissait simplement de rajouter des murs, des sols et de nouveaux objets. Nous aurions également dû rédiger des scripts permettant d'ouvrir les portes en plus de ceux permettant d'interagir avec les objets.
Le rendu final de la modélisation 3D est disponible dans une vidéo que Elise et Pierre ont réalisé, il est possible de voir le projet sous plusieurs angles

Figure 5: modélisation 3D sur Unity

Figure 6: photo du hall de E4

4. Conclusion / retour sur expérience

En ce qui nous concerne (Elise et Laure), nous avons beaucoup aimé travailler sur ce projet. En effet, nous avons appris à maîtriser plusieurs outils comme Blender et Unity, et la modélisation était très satisfaisante car nous pouvions voir le résultat prendre vie à l'écran. Aussi, nous avions toujours été curieuse quant à l'univers qui entoure la fabrication des jeux vidéo mais n'en connaissions pas grand-chose. Cette mineure nous a donc permis d'élargir nos horizons et d'en apprendre plus sur certains domaines de la création numérique. Cependant, nous ne pensons pas poursuivre dans cette voie pour l'instant. En effet, il s'agit d'une partie du développement informatique qui nous attire moins. Comme projet, c'était toutefois très formateur et intéressant.

Partie Valentin

1. Lancement du projet : Tutoriels et Documentation

Je suis parti dès le début du projet sur la conception des scripts. J'avais envie d'apprendre Unity, et les scripts sont probablement la partie la plus importante du moteur, puisqu'ils incarnent l'aspect fonctionnel. J'ai essayé dès le départ d'implémenter le système permettant d'attraper des objets. Ne sachant pas vraiment comment faire, j'ai effectué des recherches sur les forums dédiés, certaines fructueuses, d'autres, moins. A partir du moment où je me sentais tourner en rond, je me suis dit que tout simplement, il vaudrait mieux commencer par le commencement : la documentation officielle. Alors c'est ce que j'ai fait. En quelques jours, j'avais lu quasiment toute la doc, ce qui m'a permis de non seulement comprendre comment les éléments interagissaient dans Unity, mais aussi un peu plus en profondeur comment le moteur fonctionne de manière générale (plusieurs appels à Update différents, plus ou moins fréquents, gourmands en ressources, etc.) ainsi que les possibilités qu'offre Unity et ses fonctions de base. Cela m'a permis d'aboutir à la solution que j'ai mise en place pour le système d'objets.

2. Technique : mécanismes pour attraper les objets

Elle consiste tout d'abord à appliquer un script que j'ai appelé sobrement Item, non fonctionnel, aux objets que l'on veut prendre, contenant simplement des informations sur cet objet et des éléments s'y rattachant. Ensuite, j'ai appliqué un script à l'objet-joueur que j'ai nommé PlayerInteractions, qui contiendra beaucoup d'aspects fonctionnels concernant le joueur. A ce stade, il contient surtout un mécanisme permettant la détection d'objets en face de la caméra. Pour ce faire, ma lecture de la documentation m'a bien aidé, car j'aurais eu un peu de mal à savoir qu'il fallait se servir de l'appel à Physics.Raycast, spécifiquement dans le type d'update FixedUpdate. Cette fonction trace une ligne imaginaire de longueur spécifiée (que j'ai couplée à la portée de bras du joueur), dans la direction spécifiée (prendre la transformée forward de la caméra du joueur), peut permettre de récupérer le premier objet équipé de Colliders touché, et, dernier point important, ne tester que les objets appartenant à un Layer particulier. J'ai donc créé un Layer spécifique aux objets attrapables dans notre projet. Afin de tester le mécanisme complet, j'ai créé un objet simple auquel j'ai attaché ces Colliders. En testant avec 2 objets l'un derrière l'autre je me suis rendu compte qu'il pouvait y avoir des inconsistences dans la détection. J'ai essayé de remédier au problème dans le code avec l'utilisation d'une valeur buffer (d'où tous les if apparaissant à ce niveau dans le code). La correction fut peu concluante, mais je n'ai pas vraiment passé beaucoup plus de temps dessus. Autre chose que je peux ajouter, j'ai tenté d'intégrer un package Unity nommé « Simple Outlines », permettant de tracer le contour d'objets. J'ai voulu le rendre fonctionnel quand le joueur regarde un objet à sa portée, pour lui indiquer quel objet il regarde. Finalement, le mécanisme fonctionne, mais

je n'ai pas trouvé comment changer, dans le script Item (qui instancie l'Outline de chaque objet), la couleur de cet Outline, ce qui fait que l'on peut observer une couleur blanche, très peu visible, que j'aurais préféré rouge par exemple, et un peu plus épaisse.

3. Technique : mécanismes pour lancer les objets

Mon deuxième travail a été d'implémenter la fonctionnalité du lancer d'objets, afin de rendre le jeu un peu plus dynamique. Pour ce faire, la plupart des interactions que je vais mentionner se passent dans le script PlayerInteractions. Première étape, il faut pouvoir « stocker », et « déstocker » les objets ramassés, sans problèmes de pointeurs ou de valeurs nulles. J'ai donc mis en place un système simple de tableau, avec taille maximale et un entier servant d'index actuel du dernier objet. Incrément de l'index quand on ramasse un objet, décrément quand on le lance.

Maintenant, pour le lancer en lui-même. Tout d'abord, il a fallu réfléchir à plusieurs choses : la force de lancer, la gravité que l'on utilise, l'endroit d'où l'objet est lancé, et enfin la direction de lancer. C'est là que j'ai rencontré quelques problèmes. Tout d'abord, pour la position initiale de l'objet lancé. Tous nos objets n'avaient de toute évidence pas leur position d'origine en leur centre volumique. Par exemple, je sais que nos tables, quand elles ont été modelées, avaient leur origine au bout d'un de leurs pieds. Mais d'autres objets, aucuns problèmes. Cette inconsistance fait qu'il est impossible de pouvoir prévoir le bon calcul pour la globalité de nos objets. Ainsi, le résultat est souvent assez médiocre : les tables sont bien lancées depuis une position proche de la caméra, mais c'est le pied qui se retrouve là où idéalement, il aurait fallu que ce soit son centre. Aucun problème pour les chaises, par contre. Maintenant que j'ai un peu plus d'expérience, je saurai qu'il faut vérifier ce genre de choses avant d'intégrer des objets.
Sur l'aspect technique, j'ai utilisé les propriétés de RigidBody (script appliqué à tous les objets affectés par la gravité), notamment AddForce, pour appliquer une force linéaire à l'objet, et j'ai aussi utilisé MoveRotation, pour ajouter un mouvement rotatif aléatoire, cf Ligne 152 de PlayerInteractions.
Pour ce qui est de la force de lancer et la gravité, j'ai simplement fait des essais pour voir ce qui conviendrait le mieux. J'ai diminué l'intensité de la gravité du moteur, car pour moi les objets retombaient trop vite.
Je regrette pour cette interaction le fait de ne pas pouvoir lancer un objet tout en bougeant (zqsd). J'ai passé des heures à rechercher une solution, à tester de potentiels correctifs, explorer des pistes, mais rien n'y a fait.

4. Autres travaux

Ce sont vraiment les deux grandes parties du projet que j'ai exécutées, celles qui m'ont pris plus de temps et de recherche. Dernière implémentation fonctionnelle, une zone permettant de gagner des points en y lançant des objets. Il a simplement fallu créer un objet dans la scène Unity contenant un Collider, que l'on définit comme Trigger. En y attachant un script, il est possible de définir une fonction appelée quand un objet (ses Colliders) y entre : OnTriggerEnter. On incrémente le score, en se basant sur le poids de l'objet, et on rafraichit l'UI du score. J'ai aussi défini une valeur is_triggered sur chaque objet que l'on met à true la première fois que l'objet rapporte des points, afin de ne pas créer de points à l'infini.

Au-delà de ça, j'ai notamment aidé Rathsada à créer un système de quêtes fonctionnel (bien que concrètement, on n'en a implémenté qu'une dans le prototype, et pas visible à l'utilisateur, mais la logique est là), et aussi Hugues sur des problèmes qu'il a eus pour implémenter le son. J'ai aussi essayé, quand j'en voyais, de régler des problèmes mineurs (rétablissement des touches de déplacement de qwerty à azerty, affinement des Colliders, ajout de RigidBody manquants, nettoyer les assets du projet pour diminuer les temps de chargement, etc.).

5. Conclusion personnelle

J'ai beaucoup appris durant ce projet. Je suis parti de rien, et j'ai maintenant le sentiment que je pourrais créer mon propre projet Unity et y faire quasiment tout ce que je veux. Pour moi, ce semestre a donc été une réussite, même si je regrette que notre projet n'ait pas pu être plus abouti, faute de temps. J'ai dû choisir d'ignorer des éléments qui me gênaient, car l'important était de se focaliser sur les fonctionnalités plutôt que les détails.

Partie Rathsada

En choisissant la mineure création du numérique, j'ai eu l'opportunité d'avoir un aperçu du monde de la création de jeux vidéo avec l'outil Unity3D. En effet, depuis longtemps je me posais la question de savoir si je voulais travailler dans le domaine des jeux des vidéos ou non, si j'étais seulement intéresser à y jouer ou si je voulais en créer et en faire mon métier. Je ne m'y intéressais que pour la partie ludique et non sur leur création. Je n'avais pas eu la chance d'en commencer un en partant de zéro.

1. Création du joueur et des PNJ

C'est pourquoi, j'ai choisi d'être dans l'équipe qui codait le jeu sur Unity3D. Malgré un début assez compliqué avec la prise en main de Unity3D et GIT (qui était plutôt ardue dû à un problème dont on ne comprenait pas la cause), j'ai pu commencer à créer le joueur « First Person Player » en orange avec ses déplacements (en QWSD -valeur par défaut – au début mais qui a été modifié en ZQSD par Valentin par la suite) et le mouvement de la caméra en bougeant la souris grâce à différents tutoriaux à l'aide de script « PlayerMovement.cs » et « MouseLook.cs ». Par ailleurs, au début de la création du Git, j'ai essayé de résoudre nos problèmes de git en tentant différentes approches mais la plus efficace a été de sauvegarder le projet localement, de le supprimer du dépôt git puis d'ajuster le .gitignore et de remettre le projet dans le dépôt.

Ensuite, j'ai créé trois différents PNJ – Personnage Non Jouable – (NPC Jeff, NPC Toto et NPC Didi) en rose qui sont des prototypes avec une forme de « capsule » pour plus me concentrer sur les fonctionnalités de notre jeu que sur l'apparence du jeu. En effet, on voulait présenter au bout de un semestre un jeu qui serait jouable avec toutes les principales fonctionnalités qu'on voulait présenter, même si le design et l'animation n'est pas poussé.

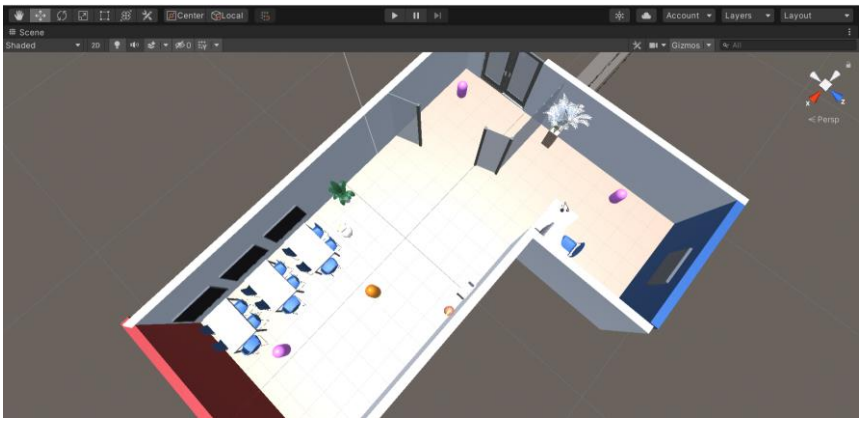

Figure 7 Vue du dessus avec en Orange : Joueur / Rose : PNJ

2. Création des dialogues

Après la création de ces PNJ, je me suis penché sur la mise en place d'un dialogue entre le joueur et ce dernier « DialogueManager.cs ». Cette partie m'a pris énormément de temps car je ne trouvais pas de tutoriel qui pouvait s'adapter à notre architecture qui partait de zéro. La plupart des tutoriaux que je trouvais, partaient d'un jeu déjà développé avec des « assets » ou une architecture précise qui était différente de la nôtre. N'ayant pas de grandes connaissances sur Unity3D, j'étais un peu perdue au début. Cependant, j'ai pu créer un premier prototype de dialogue entre le joueur et les PNJ et qui diffèrent selon les PNJ avec des réponses et un script de dialogues différents entre chaque PNJ. Le dialogue se déclenche lorsque le curseur du joueur est pointé sur un PNJ et que celui-ci appuie sur la E. Le choix de ses réponses disponible se fait en scrollant la molette de la souris, et valide sa réponse avec la touche Entrée. Une réponse différente est alors affichée sur l'écran selon le choix du joueur avant. On pourrait donc imaginer que cette fonctionnalité pourra s'étendre sur un management de script plus long et plus développé que 1 question et 1 réponse.

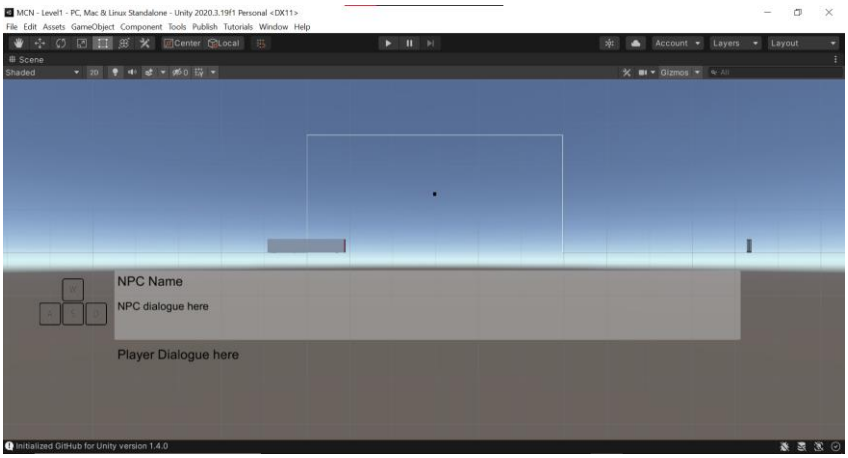

Figure 8 Canva

3. Lancement d'une quête

Ma dernière tâche a été d'implanter le lancement d'une quête lorsque le joueur choisit de lancer une quête avec le bon PNJ et bon dialogue. Avec la coopération de Valentin, le lancement d'une quête se fait avec le script « Quest.cs » et « PlayerInteraction.cs ». Lorsque le joueur va voir le bon PNJ pour lancer une quête, la quête se déclenche en arrière-plan et le joueur doit alors attraper 3 objets et les jeter pour compléter la quête. Dû à un court laps de temps, je n'ai pas pu ajouter d'autres types de quêtes.

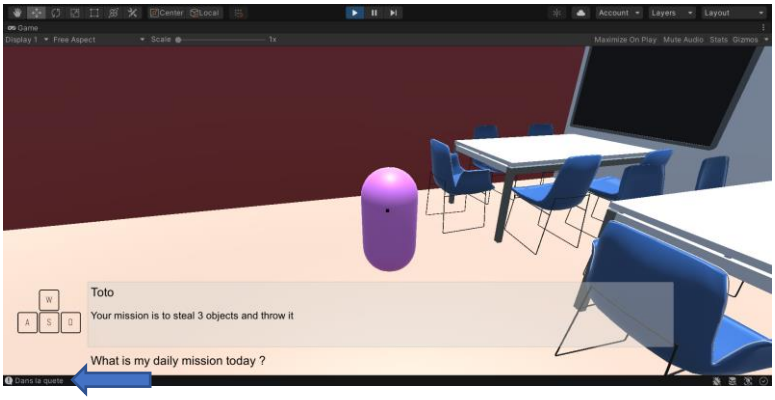

Figure Lancement d'une quête avec le PNJ Toto

4. Mon ressenti

Enfin, en réalisant ce mini-projet jeu, je me suis beaucoup amusée à le faire et surtout avec mon groupe auquel on s'entendait tous bien. Par ailleurs, j'ai pu avoir une idée plus précise sur la création d'un jeu vidéo en 3D en FPS (First Person Shooter – Vue en Première Personne) et qui je pense ne sera pas la voie professionnelle que je choisirais. Cependant, je suis fière du rendu que nous avons produit au bout de ce semestre de travail. Ce projet a donc été un moyen pour moi de me faire une première idée sur la création d'un jeu vidéo.

Partie Hugues

1. Lancement du projet

Dès le début du projet, j'ai souhaité m'intéresser à la gestion des différents sons pour notre jeu vidéo. Je suis un habitué des jeux vidéo et pour ma part je trouve que les ambiances sonores peuvent faire une réelle différence sur l'expérience vécue par le joueur. D'autre part, je m'intéresse beaucoup aux montages vidéo mais également à la composition et la conception, que ce soit de sons ou de musiques. J'ai donc été volontaire dès le début du projet pour m'occuper de cette partie. J'ai pu découvrir comment intégrer des sons dans un jeu vidéo et cela a été plus difficile que ce à quoi je m'attendais. La première chose à faire a été de créer un dossier de fichiers pour les sons dans Unity dans l'arborescence des fichiers du projet.

Pour commencer, j'ai passé beaucoup de temps sur internet à chercher des sons libres de droit. Je souhaitais trouver cinq sons pour différentes séquences ou moments dans le jeu : un son d'ambiance en arrière fond du jeu lors d'une partie, un autre pour une musique d'ambiance cette fois dans le menu, un autre pour le moment où le joueur ramasse un objet, un autre pour le moment où le joueur meurt, et enfin un autre pour le moment où le joueur passe d'un niveau au niveau supérieur. Pour des raisons de délai, seuls trois sons sont présents et utilisés dans notre jeu. J'ai réussi à travailler en parallèle de mes autres partenaires de projet, même si pour tester certains sons j'avais besoin que les scripts soient déjà présents dans notre jeu.

2. La musique d'ambiance

Pour la musique d'ambiance de fond durant une partie, je souhaitais avoir un fond sonore ainsi que des bruitages comme l'ouverture et la fermeture de porte, des chaises déplacées, des bruits de stylos, ainsi que des bruits de discussion. Le but était de récréer une ambiance sonore comme on peut l'entendre tous les jours dans les couloirs de l'école. Au début j'ai hésité à ajouter un fond sonore mais je me suis rendu compte que sinon, à certains moments, c'était un peu vide. Le simple fait d'avoir une musicalité joue beaucoup dans l'immersion et rend l'expérience de jeu beaucoup plus agréable. Je n'ai pas réussi à trouver une piste son toute faite avec tous les éléments que je voulais. J'ai donc cherché plusieurs pistes avec tous les différents bruits nécessaires à ma musique d'ambiance. Par la suite, j'ai utilisé le logiciel Adobe Audition 2020 pour mettre en parallèle mes différentes pistes sons et composer le son comme je l'avais pensé. Ensuite je l'ai intégré directement sur Unity. Pour cela, dans la scène du Level1, j'ai utilisé le GameObject FristPersonPlayer pour mettre un Component AudioSource là où je veux utiliser mon son d'ambiance. J'ai ensuite coché les options PlayAwake et Loop pour que mon son soit joué en boucle dès que le jeu se lance.

3. Les autres sons

Pour le son d'ambiance dans la scène du menu du jeu, j'ai pris un son qui est celui du menu du jeu Minecraft. Pour l'intégrer dans Unity, cette fois-ci j'ai créé un GameObject, que j'ai appelé AudioManager, dans lequel j'en ai créé un autre, appelé Ambiance. Pour intégrer ce son, j'ai procédé

de la même façon que précédemment. J'ai mis un Component AudioSource, intégré mon son d'ambiance et coché les réglages PlayAwake et Loop.

Enfin, le son qui a pris l'essentiel de mon temps et pour lequel j'ai eu le plus de difficultés est le son joué au moment où le joueur ramasse un objet. Pour pouvoir intégrer et faire fonctionner ce son dans notre jeu, j'ai dû utiliser le travail de mes autres partenaires de projet. En effet, comparé aux autres sons, celui-ci se déclenchait selon une condition particulière. Pour y arriver, j'ai donc dû m'intéresser aux scripts dans Unity. La condition pour déclencher ce son était que le joueur appuie sur la touche E de son clavier, qui correspond à la touche pour ramasser un objet. J'ai commencé à créer un fichier pour gérer le son et mettre tout ce qu'il fallait pour pouvoir bien l'utiliser (méthodes, variables…). Par la suite, j'ai demandé de l'aide à Valentin pour cette étape. En effet, toutes les interactions entre le joueur et son environnement selon des conditions particulières étaient gérées depuis le script Interactions.cs et c'était lui qui l'avait codé. Il m'a donc expliqué son fonctionnement pour que je puisse ensuite mettre en place l'appel de la méthode pour déclencher le son au bon endroit du script.

```
void Interact(Item i)
{
    if(selector < bag_size - 1)
    {
        i.gameObject.GetComponent<Rigidbody>().constraints = RigidbodyConstraints.None;
        selector++;
        //Debug.Log("Picking up item "+selector);
        bag[selector] = i.gameObject;
        i.Interaction();
        Sound.PlaySound();
    }
```

Image : la méthode PlaySound utilisé sur la GameObject Sound pour déclencher le son selon la condition dans le if

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Catch_Object : MonoBehaviour
{
    [SerializeField] private AudioClip audioCatchObject = null;
    private AudioSource perso_AudioSource;



    // Start is called before the first frame update
    void Start()
    {
        perso_AudioSource = GetComponent<AudioSource>();
    }

    // Update is called once per frame
    void FixedUpdate()
    {

    }

    public void PlaySound()
    {
        perso_AudioSource.PlayOneShot(audioCatchObject);
    }
}
```

Image : fichier CatchObject.cs pour gérer l'implémentation et les réglages pour le son

En parallèle, il s'agit d'intégrer le son dans les fichiers de Unity au bon endroit. Comme pour les autres sons, j'ai mis un Component Audio Source dans le GameObject CatchObject lui-même dans FirstPersonPlayer. Ensuite je n'ai pas directement mis le son dans mon AudioSource, pour qu'il ne soit pas joué systématiquement mais dans une condition particulière. Avoir le Component permet de faire comprendre à Unity qu'il y a un son, mais c'est le script qui gère le moment où se déclenche le son.

Pour conclure, j'ai beaucoup apprécié de travailler sur ce projet et particulièrement sur la gestion du son dans le jeu vidéo. Etant joueur de jeux vidéo, j'étais intéressé de découvrir comment en concevoir et en mettre au point un.

Conclusion Générale

De manière générale, nous avons tous gagné à participer à la mineure Création Numérique ce semestre. Nous avons tous dorénavant une vision bien plus claire du processus de création actuel d'un jeu vidéo.

Certains d'entre nous ont aimé travailler sur ce projet, d'autres moins. Dans tous les cas, chacun en est ressorti avec une forme de fierté d'avoir réalisé quelque chose sous Unity, un standard aujourd'hui des éditeurs de jeux vidéo.

Nous n'étions pas forcément tous arrivés avec l'idée d'un jeu vidéo en tête, ni n'avions les mêmes objectifs, mais nous pouvons dire que nous sommes tous satisfaits, à différents niveaux. Nous avons su nous coordonner pour donner des rôles au sein de l'équipe qui plaisaient à chacun.

Nous avons effectué des recherches sur des domaines qui nous étaient inconnus jusqu'alors, commis des erreurs qui nous ont permis de gagner en expérience et en maturité.

Nous sommes tous d'accord pour se dire qu'avec les difficultés que nous avons rencontrées, ainsi que le peu de temps que nous avions en dehors de la mineure, nous sommes contents du résultat, mais aussi quelque peu frustrés. Nous aurions voulu pouvoir aller bien plus loin, bien plus vite sur ce projet. Cela nous aura ainsi permis d'avoir une meilleure appréciation du temps.

Ce projet aura donc été une expérience très révélatrice et riche en savoir pour chacun de nous, une expérience que nous pensons valorisante.