Treasure Hunt

UbiHard

4 Mars 2021

Table des matières

1 Introduction

Nous sommes fière du travail que nous avons réalisé pour cette première soutenance. En effet, presque tous les objectifs que nous nous étions fixés sont remplis, ainsi que certaines tâches sur lesquelles nous sommes en avance. Nous avons suivi notre cahier des charges, malgré quelques petites modifications réalisées pendant la programmation. Notamment, nous avons décidé de créer deux coffres distincts.
Tout d'abord, nous avons rencontrés de nombreux problèmes avec Git, surtout dus à des merge conflict. Nous nous sommes adaptés et avons trouvé plusieurs solutions, comme par exemple créer différentes scènes pour chacun d'entre nous, afin d'éviter les conflits.
Ensuite, nous avons également mis un peu de temps à nous adaptés à Unity, car nous n'avons jamais fait de projet avec des affichages visuels aussi avancés.
Enfin, il nous reste encore de nombreuses tâches à accomplir, comme le design des niveaux, l'implémentation des ennemis, des objets et des sons du jeux, et nous avons du retard sur certains points de notre planification Malgré cela, nous sommes déjà fiers du travail que nous avons accompli, sachant que c'était pour nous la première fois que nous utilisions l'outil Unity et que nous travaillions sur un projet.

2 Pour cette soutenance

Depuis le rendu de notre cahier des charges, nous avons avancés dans la conception de notre jeu sur plusieurs aspects. Nous avions prévu dans notre cahier des charges les tâches à accomplir pour la première soutenance. Voici ce qu'il en est :

Tâche	Responsable	État
Joueur mouvement	Nicolas Schmitt	Terminé
Joueur autres actions	Nicolas Schmitt	Terminé sauf mort
Joueur sons	Joan Zasempa	Presque tous terminés
Ennemi 1 implémentation	Nicolas Schmitt	Terminé
Ennemi 1 mouvement	Nicolas Schmitt	Terminé
Ennemi 1 sons	Joan Zasempa	Pas terminé
Ennemi 1 I.A	Quentin Sefrin	Pas terminé
Ennemi 2	Joan Zasempa	Pas terminé
Création niveau test	Nicolas Schmitt	Terminé
Piège Oneshot	Nicolas Wittwe	Terminé
H.U.D vie	Nicolas Wittwe	Terminé
H.U.D objets	Nicolas Wittwe	Terminé
Menu principal	Quentin Sefrin	Terminé
Menu solo	Nicolas Wittwe	Terminé
Site Web	Nicolas Wittwe	Déployé
Coffre	Nicolas Wittwe	Terminé
Levier	Nicolas Wittwe	Terminé
Flèche	Nicolas Schmitt	Pas terminé
Balle	Nicolas Schmitt	Terminé
Porte	Nicolas Wittwe	Terminé
Consommable 1	Joan Zasempa	Terminé
Consommable 2	Joan Zasempa	Terminé
Consommable 3	Joan Zasempa	Pas de résurrection
Multijoueur	Quentin Sefrin	Implémenté, mais bugué

2.1 Le site Web

Notre site Web est déjà en place. Il est en langage HTML et CSS. Il n'est pas encore complet, mais présente notre projet dans son ensemble. Il est composé de quatre pages. La première présente le jeu aux joueurs intéressés. Nous y ajouterons avec le temps des images du jeu et un trailer. La deuxième permettra d'installer correctement le jeu quand le téléchargement sera mis en place. Dans la troisième page, on présentera plus en détail notre projet, avec par exemple notre cahier des charges, ou nos rapports de soutenances. On ajoutera également une vidéo ou des images du niveau présenté à chaque soutenance. Enfin, la quatrième page présentera l'équipe du projet.

Son adresse est https://ubihard-projet.netlify.app/

2.2 Le Personnage

Nous avons implémenté le personnage avec certaines de ses fonctionnalités. Il peut, pour l'instant :

— Se déplacer horizontalement avec les touches Q et D. Pour cela, on utilise la méthode Input.GetButtonDown ("Horizontal"), qui renvoie -1 si Q est pressé, 1 si D est pressé et 0 si les deux sont pressés. Cela permet ensuite, avec une méthode Move() créée, de déplacer le personnage en fonction de sa vitesse et de sa direction.

— Sauter à l'aide de la touche ESPACE. Avec la méthode Input.GetButtonDown("Jump") et si le personnage n'est pas au sol, on applique une force positive verticale au personnage. Pour détecter s'il est au sol, on utilise un point placé aux pieds du personnage, et on vérifie s'il est en contact avec le sol.

— S'accroupir avec la touche S, ce qui réduit sa taille et sa vitesse, lui permettant d'esquiver les flèches et de passer dans des endroits étroits. Le personnage possède un circle collider 2D sur sa moitié inférieure et un box collider 2D sur sa moitié

supérieure. Quand il s'accroupit, la box collider 2D est désactivée, ce qui réduit la zone de collision du personnage.

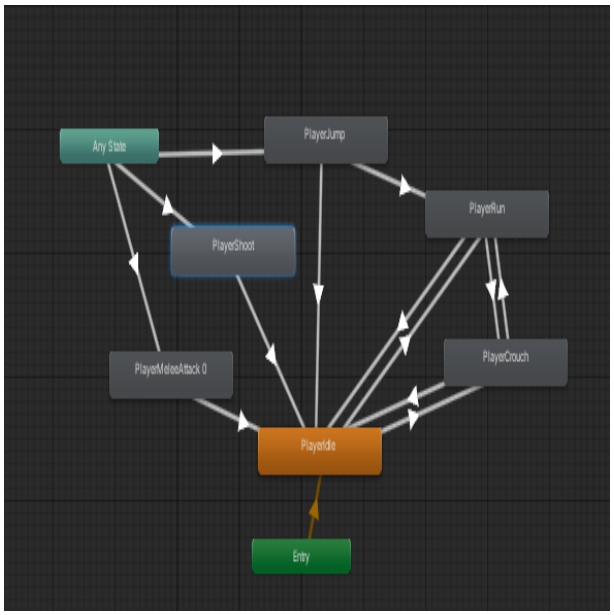
— Attaquer au corps à corps avec clic gauche. Un point est défini sur l'arme du personnage. On calcule à l'aide de physics 2D, au moment de l'attaque, si un ennemi est présent dans un rayon autour de ce point. Si un ennemi est présent, la méthode TakeDamage() de l'ennemi est appelée. Un cooldown a été mis en place, afin de laisser l'animation du personnage se dérouler et de ne pas le rendre surpuissant face aux ennemis.

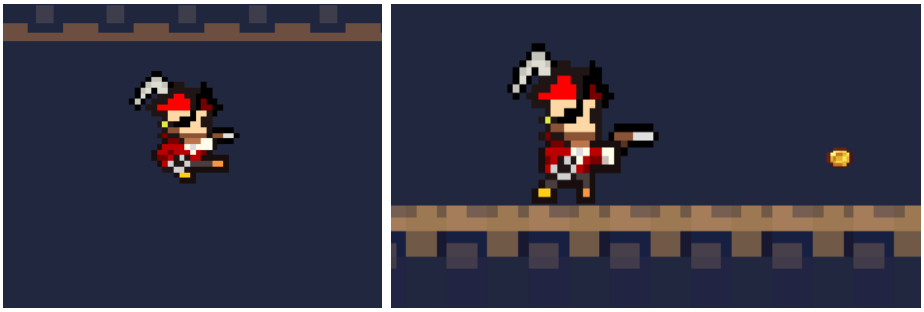
— Attaquer à distance avec clic droit. Un point est défini sur l'arme du personnage, la balle tirée par le joueur part de ce point. l'objet balle est préfabriqué et instancié au moment du tir. La balle avance en ligne droite et quand elle rencontre un obstacle, elle se détruit. Si l'obstacle est une ennemi, la méthode TakeDamage() de l'ennemi est appelée. Un cooldown sera mis en place plus tard, afin de laisser l'animation du personnage se dérouler et de ne pas le rendre surpuissant face aux ennemis. De plus, pour être plus réaliste, un pistolet pirate ne tire pas en rafale.

Pour les timers ou cooldowns de notre projet, nous avons décidé de les incrémenter dans la fonction FixedUpdate() plutôt qu'Update(), afin que la puissance de l'ordinateur n'influence pas ceux-ci. Au niveau des animations du personnage, nous en avons 6 pour l'instant :

— Idle
— Run
— Jump
— Crounch
— MeleeAttack
— Shoot

Quand on lance le jeu, le joueur est d'abord en animation d'Idle, s'il se met a courir, il passe de l'animation d'Idle à l'animation de Run et inversement quand il arrête de courir. Plusieurs paramètres ont été créé dans l'animateur afin de gérer les transitions entre les différentes animations. L'animation Jump peut se lancer depuis n'importe quelle animation si le joueur saute, de même que les animations MeleeAttack et Shoot quand le joueur attaque. Une fois qu'elle est fini, on peut retourner vers l'animation de Run ou d'Idle en fonction de la vitesse du personnage.



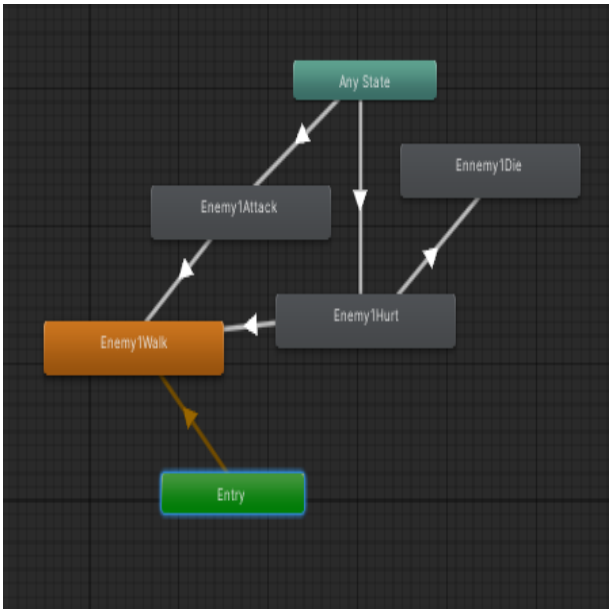


2.3 Les Ennemis

Pour cette première soutenance, nous avons créé un ennemi : le guerrier squelette. Il peut faire des allers-retours entre deux points et attaquer un joueur devant lui. De plus, il peut prendre des dégâts et mourir. L'ennemi possède pour l'instant 3 animations :

— Walk
— Attack
— Hurt
— Die

Quand la vie du squelette est <= 0 la fonction Die() de ce dernier est appelée, puis il ne reste plus qu'un tas d'os sur le terrain. Chaque fois la fonction TakeDamage() du squelette est appelée, l'animation Hurt est jouée.



2.4 Les pièges

La création des pièges est à jour par rapport à nos prévisions. Le piège mortel, qui enlève toute la vie du joueur, est en effet implémenter et fonctionnel. Pour cela, si le joueur est en contact avec le piège, il appel la méthode TakeDamege(int n), qui soustrait

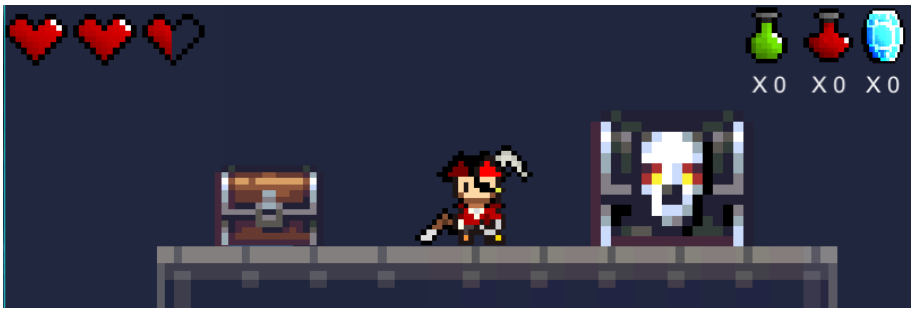
à la vie actuelle du joueur l'entier n en argument. Il prend la forme de lave ou d'eau.

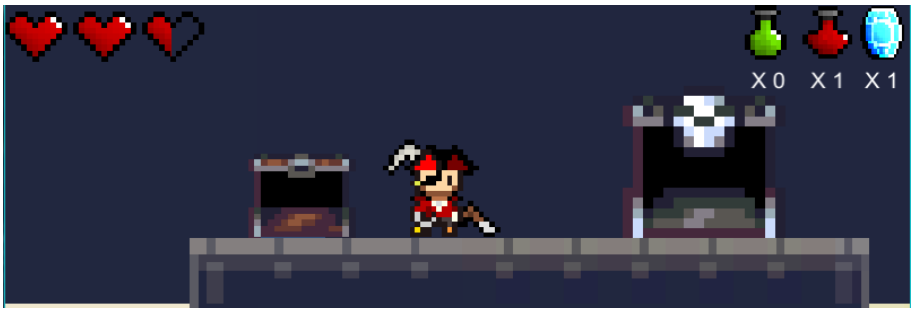
Nous avons implémenté un peu en avance les piques vers le haut. Ceux-ci infligent 1 de dégât soit un demi-cœur à tous les joueurs avec qui ils sont en contact. Ils fonctionnent donc de la même manière que le piège mortel. Cependant, un timer après avoir infligé des dégâts l'empêche de blesser à nouveau le joueur immédiatement. Autrement, il perdrait toute sa vie instantanément.

2.5 Les objets

Nous avons déjà terminer la création de plusieurs objets.

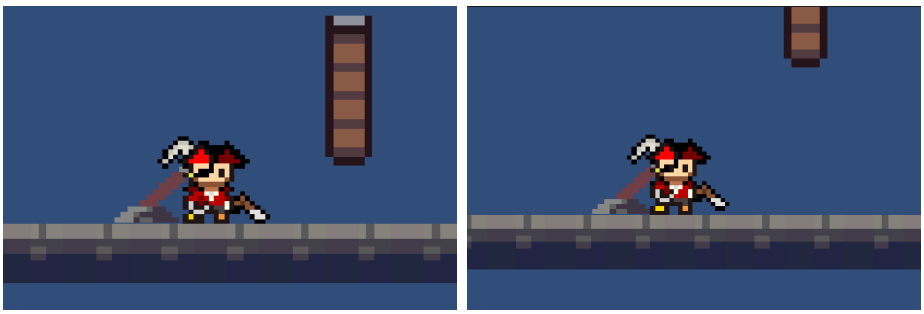
Le premier est le coffre. Nous avons créé un coffre qui s'ouvre quand un joueur à proximité utilise la touche d'interaction. Les coffres donnent au joueur un objet consommable. Au départ, nous avions prévu de ne créer qu'un seul coffre, et de modifier les consommables qu'il donnent en fonction du mode de jeu (solo ou multijoueur). Cependant, après réflexion, nous avons décidé de créer 2 coffres différents : le premier donne au joueur une potion aléatoire tandis que le deuxième lui donne toujours un totem de résurrection. Les coffres fonctionnent ainsi : quand un joueur est en contact avec un coffre fermé (détecté grâce à un box collider 2D), si le joueur interagit avec la touche E, alors le coffre donnera au joueur une potion au hasard. Le deuxième fonctionne de la même manière, mais donnera uniquement un totem de résurrection (il ne sera présent qu'en multijoueur).





Le deuxième est une porte. Pour l'afficher de manière cohérente dans un monde 2D, nous avons choisit une texture ressemblant à un mur. Ainsi, pour son ouverture, nous avons choisi de la faire "glisser" vers le haut ou vers le bas. Pour cela, nous avons défini 2 points : un d'état fermé, l'autre d'état ouvert. Nous appliquons ensuite une force vers ces points en fonctions de l'état de la porte et du levier.

Le troisième est un levier. Il permet d'ouvrir ou de fermer les portes. Nous avons prévu 2 types de levier, cependant un seul a été complètement réalisé pour cette soutenance. En effet, le deuxième type de levier n'est pas encore associé à l'objet porte. Nous avons créé le premier type de levier. Il change de sens quand le joueur interagit avec en étant à proximité. Le sens du levier détermine la direction de la porte, vers un état ouvert ou fermé. Pour l'instant, il est possible de stopper le mouvement d'une porte grâce au levier. Cela ne semble pas être un mécanisme gênant, nous l'avons donc conservé, mais il pourrait être retiré plus tard lors de l'équilibrage du jeu. Nous avons implémenté le deuxième type de levier celui avec un timer. Celui-ci reprend sa position initiale (ferme la porte) au bout d'un certain temps. Ce temps sera variable en fonction de l'endroit où sera placé le levier et sa porte. En effet, ils seront utilisés dans des énigmes où le joueur devra parcourir une certaine distance dans un temps limité.



Les quatrièmes sont les consommables :

— Potion de soin inférieure : elle rend au joueur 1 point de vie. Cela modifie le H.U.D du joueur, au niveau des points de vie et du nombre de consommables

possédés.
— Potion de soin supérieure : elle agit de la même manière que la potion inférieure, mais rend 2 points de vie.
— Totem de résurrection : il ramène un joueur mort à la vie. Il n'a pas encore d'effet de résurrection, mais le joueur peut l'obtenir et l'utiliser aussi en solo, ce qui rend toute sa vie au joueur. Le système de résurrection sera implémenté avec le système de mort. A terme, il ne sera pas disponible en mode solo.

2.6 L'Interface

Nous avons déjà créé un menu en partie fonctionnel. Pour l'instant, il ne permet pas de modifier les options, ni de jouer à partir d'une sauvegarde. Il est cependant fonctionnel et permet de démarrer rune nouvelle partie, en solo ou en multijoueur. Il reste cependant à modifier afin de le rendre attrayant aux yeux des joueurs.

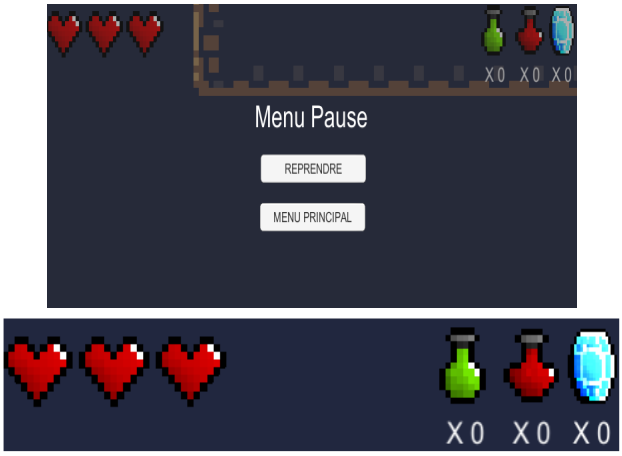
Nous avons implémenté le H.U.D solo. Dans celui-ci se trouvent la vie du joueur et son inventaire. Pour la vie nous avons fait un système de barre de vie avec un skin de cœur. Pour cela nous avons deux images superposées : une avec les cœur vide et une avec les cœurs rouges. Les cœur vide sont tous le temps affiche mais l'image cœur rouge s'affiche en fonction du pourcentage de vie du joueur. Nous avons défini les PV max du joueur à 6 (2 fois le nombre de cœur) pour que nous puissions avoir soit des cœurs pleins ou des demi-cœurs.

Pour l'inventaire nous avons créé 3 images qui représente chacune un objet et sous chacune d'elle un texte qui affiche son nombre. En solo il n y a pas de totem de résurrection mais nous l'avons quand même mis dans le H.U.D (pour faire des tests sur le coffre totem par exemple). Pour cela nous avons une variable entière pour chaque objet qui représente le nombre d'items que le joueur possède. Ces variables sont incrémentées dans leur méthode attitrée et sont converties en chaîne de caractère pour être affichées. Les méthodes sont ensuite utilisées dans d'autre objet qui donne les consommable. Par exemple les coffres font appel à ces méthodes pour incrémenter le nombre d'objet du

joueur. Bien sûr le nombre d'item de chaque objet est initialisé à 0 à chaque début de partie.

Nous avons également implémenté un menu pause solo qui permet au joueur en appuyant sur la touche « escape » d'ouvrir ce menu. Dans ce menu il y a deux boutons : le premier « Reprendre » permet de reprendre la partie. A noter également que quand le menu est ouvert on peut le refermer en appuyant une deuxième fois sur « escape ». Le deuxième « Retour au menu » permet de retourner au menu principal. Ce menu pause n'arrête pas le temps car c'est un jeu multijoueur mais il désactive le joueur pendant qu'il est ouvert.

De plus, nous sommes en avance sur ce point, car un nous avons déjà créé un menu pause en partie fonctionnel en multijoueur.



2.7 Level Design

Au début du projet, il est inutile de designer les niveaux du jeu, il nous faut d'abord créer les éléments nécessaires à la création du monde. Cependant, nous avons créer un niveau de test afin de réaliser nos éléments. Il était prévu de ne créer qu'un seul niveau pour nos test, mais pour les rendre plus faciles, nous avons décidé de créer un niveau pour chacun d'entre nous.

Nous avons tout de même réaliser une scène spécial pour la présentation de la soutenance.

2.8 Le Multijoueur

Pour cette soutenance, le système multijoueur est assez basique. Il permet de créer une partie ou d'en créer une, sans plus d'options. Pour le moment, seul le déplacement

des joueurs est synchronisé, sans aucune animation. Pour cela, nous avons utilisé le package PUN 2, qui permet de se connecter sur un serveur en multijoueur avec la méthode ConnectUsingSettings(). Par la suite, dans le menu du jeu, le joueur décide s'il souhaite rejoindre une partie ou en créer une. Pour le moment, il est seulement possible de rejoindre une partie aléatoirement à l'aide de la méthode JoinRandomRoom, mais par la suite, nous permettrons au joueur de voir en avance une liste des différentes partie en cours, et de rejoindre celle qu'il souhaite. De même, pour créer une partie, le joueur n'est pas libre. La partie créée avec la méthode CreateRoom() a toujours le même nom et la même capacité de joueurs. Cependant, il sera possible par la suite de choisir le nom de la partie, ainsi que le nombre de joueurs maximum, compris entre 2 et 4. Durant la mise en place du multijoueur, il a fallu faire face à certains problèmes. Premièrement, en plus

2.9 Les Textures

Nous avions déjà commencer à rechercher des textures à utiliser dans notre jeu pendant l'écriture du cahier des charges. Nous ne savons pas encore toute les textures que nous allons utiliser, mais nous en avons déjà trouvées certaines. Nous avons déjà décidé des textures du personnage, de chaque ennemi, des objets, ainsi que des décors de certains niveaux.

2.10 Les Sons

Nous avions prévu d'implémenter quelques sons pour la première soutenance, cependant, nous n'avons pas pu ajouté tous les sons que nous voulions incorporer pour la première soutenance. Notamment, les sons de saut du joueur et d'utilisation des consommables ont été ajoutés au jeu. Pour cela, nous avons placé dans le script, on initialise une source audio, ainsi qu'un fichier audio. Ensuite, au moment d'effectuer une action, on utilise la méthode source.AudioOneShot(son) afin de jouer le son depuis la source. Il a donc fallu ajouter un composant Audio Source dans Unity. Toutefois, nous avons déjà trouvé certains sons que nous allons utilisés, notamment la musique d'introduction et les sons des squelettes.

3 Pour la prochaine soutenance

Pour la prochaine soutenance, nous aurons presque terminé notre projet. Voici ce que prévoit notre cahier des charges :

Tâche	Responsable	État prévu
Joueur sons	Nicolas Schmitt	Terminé
Ennemi 1 sons	Joan Zasempa	Terminé
Ennemi 2 implémentation	Joan Zasempa	Terminé
Ennemi 3 implémentation	Quentin Sefrin	Terminé
Ennemi 4 implémentation	Quentin Sefrin	Terminé
Ennemi 5 implémentation	Joan Zasempa	Terminé
Level design 'Lobby'	Nicolas Schmitt	Terminé
Level design Plage	Joan Zasempa	Terminé
Level design Forêt	Quentin Sefrin	Terminé
Level design Grotte	Nicolas Schmitt	Terminé
Level design Salle finale	Nicolas Wittwe	Terminé
Piques vers le haut	Nicolas Wittwe	Terminé
Piques vers le bas	Nicolas Wittwe	Terminé
Ralentissement	Nicolas Wittwe	Terminé
Menu multi	Quentin Sefrin	Terminé
Site Web	Nicolas Wittwe	Mise à jour
Échelle	Nicolas Wittwe	Terminé
Cible	Nicolas Wittwe	Terminé
Digicode	Nicolas Wittwe	Terminé
Bloc mobile	Nicolas Wittwe	Terminé
Flèche	Nicolas Schmitt	Terminé
Torche	Nicolas Wittwe	Terminé
Panneau	Nicolas Schmitt	Terminé
Portail	Nicolas Schmitt	Terminé
Consommable 3	Joan Zasempa	Terminé
Musique Lobby	Joan Zasempa	Terminé
Musique Plage	Joan Zasempa	Terminé
Musique Forêt	Quentin Sefrin	Terminé
Musique Grotte	Joan Zasempa	Terminé
Musique Fin	Joan Zasempa	Terminé
Bruits de l'environnement Lobby	Joan Zasempa	Terminé
Bruits de l'environnement Plage	Joan Zasempa	Terminé
Bruits de l'environnement Forêt	Quentin Sefrin	Terminé
Bruits de l'environnement Grotte	Joan Zasempa	Terminé
Bruits de l'environnement Fin	Joan Zasempa	Terminé
Sauvegardes	Quentin Sefrin	Terminé en solo
Multijoueur	Quentin Sefrin	Perso, objets, ennemis terminés

3.1 Le site Web

Le contenu du site sera mis à jour, et son visuel pourra être modifié. Vous pourrez trouver les rapports de soutenances ainsi que les sources (textures, audios, ...) de notre jeu.

3.2 Le Personnage

Il nous reste à implémenter l'animation et l'écran de mort, ainsi que de régler les bugs des personnages en multijoueurs. nous mettrons aussi en place le système de spectateur des joueurs morts.

3.3 Les Ennemis

Nous allons, pour la prochaine soutenance, implémenté entièrement tous les ennemis restants, c'est à dire l'archer squelette, le loup, la chauve-souris et le golem. Nous allons également travailler sur leurs I.A, quitte à dévier du cahier des charges, afin de les rendre plus intéressants.

3.4 Les pièges

Il reste ici à ajouter les piques vers le bas, ainsi que leur actions quand une entité passe en dessous, ainsi que la zone empêchant le joueur de sauter.

3.5 Les objets

Il nous reste encore pas mal d'objets à implémenter. Notamment l'échelle qui permet au joueur de grimper, le digicode et la cible pour déverrouiller une porte, le portail qui téléportera le joueur dans des salles bonus. Nous avons prévu de rajouter un coffre, le coffre final, qui permettra de terminer le jeu (il n'a pas été prévu dans le cahier des charges).

3.6 L'Interface

Au niveau de l'interface, il ne nous reste qu'à gérer l'affichage de la vie en multijoueur, ainsi que d'améliorer les graphismes des menus. Nous allons également ajouté la possibilité de modifier les options, c'est à dire le niveau du son et éventuellement la taille de la fenêtre de jeu.

3.7 Level Design

Nous allons implémenter tous les niveaux dans leurs grandes lignes. En effet, nous continuerons à les améliorer jusqu'à la fin du projet.

3.8 Le Multijoueur

Nous allons devoir régler le problème de synchronisation des objets entre les joueurs. Par exemple, pour le moment, l'ouverture d'une porte avec un levier n'est effectué que pour le joueur l'ayant déclenché. Nous devrons de plus vérifier le fonctionnement de chaque objet en multijoueur. Enfin, il faudra ajouter la possibilité au joueur de choisir une partie.

3.9 Les Textures

Nous devons encore trouver les textures pour les niveaux du lobby et de la plage, ainsi que trouver plus de décors pour la forêt.

3.10 Les Sons

Nous devons encore trouver et implémenter la majorité des sons, ainsi qu'une musique qui suivra le joueur durant le jeu.

4 Conclusion

Pour cette première soutenance, nous avions prévues un peu trop de tâches, ainsi nous ne sommes pas à jour sur tous les points dans l'avancement de notre projet. En effet, nous avons sous-estimé les difficultés d'implémentation des ennemis et du multijoueur. Cependant, le but de la première soutenance était d'implémenter les bases de notre jeu, ce qui est le cas, nous avons créer tous les éléments de bases de notre projet. Nous avons donc réussi à prendre en main plusieurs aspects de l'éditeur Unity, et nous serons plus efficaces durant la deuxième période de la création de notre projet. De plus, malgré quelques retards, nous sommes tout de même en avance sur la création des pièges et des objets, nous sommes donc plutôt confiants quant à la suite du projet.