SSAO:
Rapport de Projet
Deuxième Soutenance

Groupe: **Les Rejetés**
Ouwéis MOOLNA, Edouard EISENFISZ, Lucas PINOT, Simon QUESNEY

02 Mai 2018

Table des matières

1 Introduction du Rapport de deuxième soutenance

SSAO sera un jeu de tir en coopération. Ce sera une jeu à la première personne, et les joueurs pourront y jouer de 1 à 4 personnes.
Les joueurs devront évoluer dans un environnement et affronter différents types d'ennemis pour atteindre certains objectifs. Ces objectifs seront de différentes natures et nous les détaillerons dans ce cahier des charges. L'environnement sera un monde mêlant le style Fantastique et le style Steampunk.
En effet, le monde en lui-même sera dans le style Steampunk, mais plusieurs caractéristiques des mondes Fantastiques seront présents, comme la magie. Les joueurs auraient des armes propres à leur personnage, mais pourraient en obtenir d'autres, ou avoir leurs statistiques améliorées s'ils trouvent des objets cachés dans les différentes cartes où ils évolueraient.
Les joueurs devront veiller à ne pas toucher leurs partenaires, le tir allié étant activé, car ceci pourrait les tuer et bloquerait leur progression dans le jeu ou le niveau. Effectivement, certains objectifs nécessiteront la présence de tous les membres de l'instance pour pouvoir être réalisé.
Les mondes du jeux évolueront au cours du temps. Par exemple, au début, les joueurs n'auront que peu de choix pour réaliser les objectifs, comme devoir combattre les ennemis pour atteindre un objectif ou terminer le niveau. Mais au fur et à mesure qu'il avanceront, les choix se diversifieront, par exemple, ils pourront combattre les ennemis, passer discrètement sans se faire repérer, ou même prendre d'autres chemins détournés leur proposant d'autres choix pour passer.

2 Équipe

— **Edouard EISENFISZ**, A.K.A "FireFox"
J'ai toujours la pêche et une coupe entre le feu vivant et un nuage.
J'ai déjà fait un projet l'année dernière, et j'ai grandement apprécié. Je suis un grand fan des jeux Bethesda, et des jeux en coopération, comme Warhammer : Vermineville ou Dishonored. J'ai déjà utilisé Blender et Unity.
L'idée de ce projet étant le fruit d'un partage de nos envies et de nos idées respectives, nous avons tous été à l'origine de l'idée de ce projet.

— **Simon QUESNEY**, **Chef de Projet**, A.K.A "Qui-Gon-Jinn"
"Le dernier Star Wars c'est pas très très bien pour un Star Wars"
FAN des jeux Bethesda (j'aime les bugs), notamment les jeux solo (Dishonored 1 et 2, Skyrim, Fallout), et autres jeux solo comme Bioshock ou The Witcher.
Je compte m'améliorer en C# dans ce projet en abordant des sujets complexes comme l'IA ou la mise en place de la furtivité tel un Dishonored.

— **Lucas PINOT**, A.K.A "PublicStaticVoid"
Je suis moi aussi un fan des jeux Bethesda, plus particulièrement de la saga des The Elder Scrolls. Ayant déja fait un jeu l'année dernière en projet d'ISN en python, je compte faire beaucoup mieux cette année en passant de la 2D à la 3D.

— **Ouwéis MOOLNA**, A.K.A "Bûcheron"
Je suis le seul qui n'ai jamais joué à un jeu Bethesda, sauf le dernier DOOM, bien que j'ai beaucoup entendu parler de leurs jeux. J'ai cependant joué à Bioshock et plus particulièrement Infinite que j'ai beaucoup apprécié et qui nous a aussi influencé dans certains choix pour notre jeu.
Avec ce projet j'espère progresser dans le C# mais aussi apprendre à utiliser de nouveaux outils comme Unity.

3 Distribution

Tâches	Ouwéis	Lucas	Edouard	Simon
Fonctionnalités obligatoires				
Contrôle basiques :	Chargé	Suppléant		
Level Design :			Suppléant	Chargé
IA, et mouvement des ennemis :		Suppléant		Chargé
Graphisme :		Suppléant	Chargé	
Sons :	Chargé			Suppléant
Contrôles avancés :		Chargé		Suppléant
Rapports de Projets	Chargé		Suppléant	
Site :			Suppléant	Chargé
Réseau :	Suppléant		Chargé	
Procédure d'installation :	Suppléant	Chargé		
Fonctionnalités Bonus				
Animation :	Chargé	Suppléant		
Menu :		Chargé	Suppléant	
Mode Furtif :		Suppléant		Chargé
Boss :	Suppléant	Chargé		
Code de triche :	Suppléant		Chargé	
Versus :			Chargé	Suppléant
Canon à ennemis :		Suppléant	Chargé	

4 Progression

module	**oral** 1	**Soutenance 1**	**oral 2**	**Etat actuel**	**final**
Contrôle de base	75%	En avance	100%	A l'heure	100%
Level design	33.33%	A l'heure	66.66%	A l'heure	100%
IA, et mouvement des ennemis	22.22%	A l'heure	56.65%	A l'heure	100%
Graphisme	25%	A l'heure	60%	A l'heure	100%
Animation	20%	A l'heure	60%	A l'heure	100%
Menu	0%	_	50%	A l'heure	100%
Sons	20%	A l'heure	60%	En retard	100%
Site	10%	A l'heure	65%	A l'heure	100%
Réseau	40%	A l'heure	70%	A l'heure	100%
Procédure d'installation	0%	_	0%	_	100%
Mode Furtif :	0.0001%	_	42.123%	En avance	100%
Versus :	20%	A l'heure	60%	A l'heure	100%
Contrôles avancés	20%	A l'heure	50%	A l'heure	100%
Canons à ennemis :	0%	_	30%	-	100%
Boss	0%	_	20%	-	100%
Code de Triche	20%	-	50%	-	100%

5 Le Projet

5.1 Les Contrôles

5.1.1 Contrôle Basique

Les contrôles de base sont les déplacements du personnage, que ce soit pour avancer, reculer, aller à droite ou à gauche ou bien sauter. Mais aussi le déplacement de la caméra par rapport au curseur. Les contrôles géreront les claviers et souris mais aussi les manettes.

5.1.2 Contrôle Avancé

Les contrôles avancés sont tous les contrôles concernant les attaques mais aussi les lancers de sort. Ils seront comme pour les contrôles de bases implémentés aussi bien sur clavier et souris que sur manettes.

5.2 Ennemis

5.2.1 Les Types d'Ennemis

Les ennemis de vague :

Les ennemis de vague seront les ennemis de base. Ils apparaîtront en groupe et sous forme de vagues. Ce seront les ennemis les plus faibles du jeu sans compétence spéciale en dehors de leurs attaques de base.

5.2.2 L'IA ennemie

L'IA des ennemis en vague :

L'IA la plus simple ici qui va seulement avoir pour but de trouver le joueur le plus proche et de l'attaquer, pour ce faire elle va connaître sa position au préalable et juste avancer en sa direction, dépendant des ennemis elle va soit le cibler à distance soit l'attaquer au corps-à-corps.

5.3 Graphisme

5.3.1 Level Design

Il y aura différents niveaux adaptés à chaque type de missions et modes de jeux. En effet, le mode Versus ne demandera pas une grande zone de jeu, celle-ci sera basique et ne comportera sûrement que des obstacles grâce auxquels les joueurs pourront se protéger des tirs de leur adversaire. Les missions à objectif, seront créés autour de ces objectifs. Par exemple, si l'objectif est de protéger un point, ce point sera au centre de la carte. à L'inverse, une mission où les joueurs devront atteindre un point, la carte correspondante positionnera les joueurs aux points les plus éloignés du point à défendre.
Ces différents type de carte seront dans des environnements urbains jouant entre un style londonien et le style médiéval, et jouant sur des effets de machine à vapeur créant un ensemble appartenant à une architecture de type Steampunk.
Les cartes à objectifs, auront également des paliers séparés par des portes cassables. Il faudra casser celles-ci pour continuer. C'est cartes auront également des salles cachées avec des pièges, contenant des bonus facilitant la suite du niveau.

5.3.2 Apparence des joueurs

Les joueurs auront chacun une apparence unique, dans un design Steampunk, mais nettement différente des autres et surtout des ennemis. Leurs caractéristiques seront en fonction de leur rôle. Par exemple, le guerrier sera petit et gros car il sera un nain, tandis que le mage sera fin et grand. Leur accoutrement sera également, toujours dans le style Steampunk, adapté en fonction de leur classe. Pour l'instant nous n'avons qu'un humanoide lambda.

5.4 Sorts

5.4.1 Types de Sort

Les attaques de base et objets normaux :

Les attaques de base seront les attaques normales d'un personnage, l'utilisation d'une attaque de base n'entraîne pas de temps de rechargement, elle peut donc être utilisée à volonté et aussi souvent que le joueur le voudra.
Les attaques de base pourront être améliorées ou totalement modifiées par l'acquisition d'un objet normal, comme un fusil à pompe pour le Combattant ou bâton lanceur de flammes pour le Mage Guerrier.

Les sorts d'objets magiques ou technologiques (M/T) :

Les objets M/T seront acquis au fil de la partie, pendant chaque niveau, définitivement. L'acquisition de ces objets octroiera un nouveau sort au joueur l'ayant récupéré permettant une modification du gameplay du personnage.
N'importe quel personnage pourra récupérer n'importe quel objet mais en contrepartie cet objet devra obligatoirement être récupéré pour terminer un niveau ou réussir un objectif du niveau, obligeant les joueurs à choisir plus ou moins soigneusement le personnage qui recevra l'objet en question et à réussir un objectif intermédiaire du niveau qui sera l'acquisition de cet objet.
Le joueur ayant récupéré un objet durant un niveau recevra en conséquence moins d'expérience en échange de pièces d'or servant à améliorer le ou les objet(s) récupérés au cours de la partie.
Un objet utilisé aura un temps de rechargement avant de pouvoir être réutilisé à nouveau.
La bombe jetable peut par exemple être un objet T dont l'utilisation entraînera un temps de rechargement avant réutilisation.

5.4.2 Animations des sorts

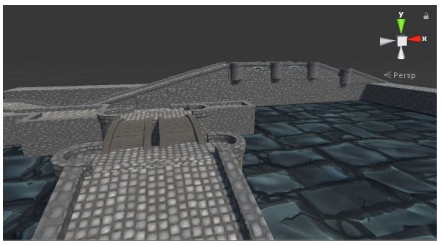
Les animations des sorts seront toutes les animations en rapport avec les sorts, et elles seront différentes en fonction du sort choisi, par exemple les sorts physiques n'auront pas la même animation que les sorts magiques. Mais il y aura aussi une animation sur le personnage quand les sorts auront fini d'être rechargés.

6 Présentation du travail réalisé

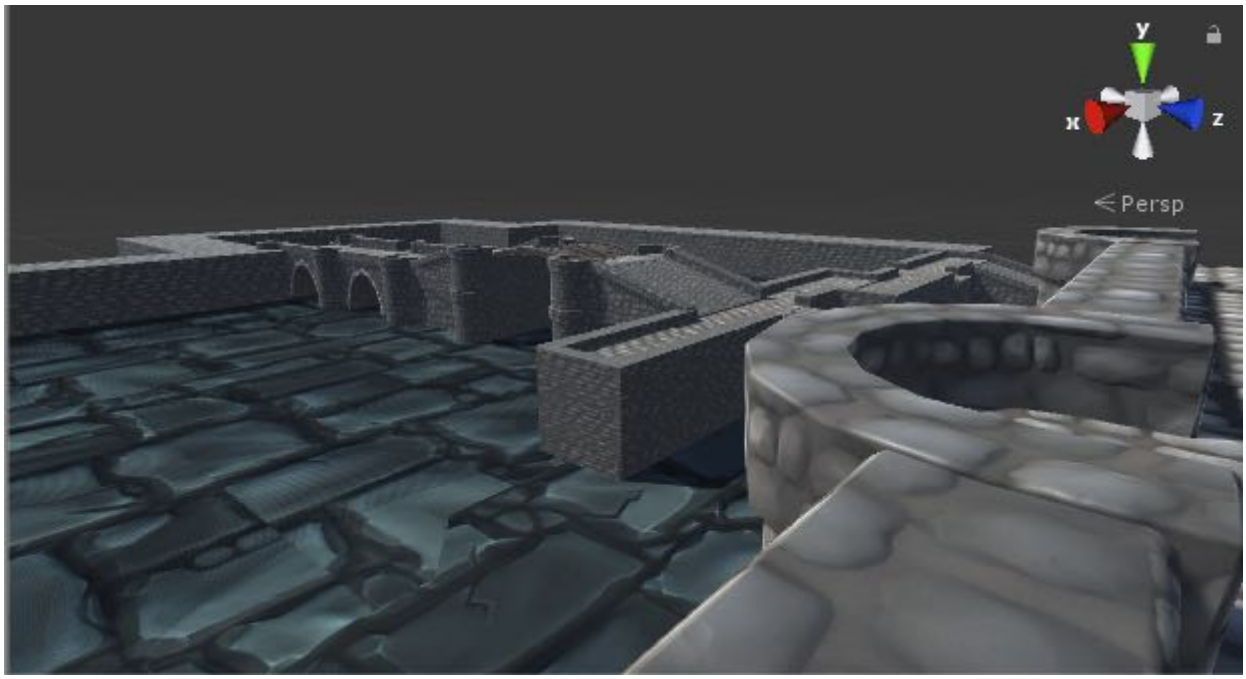
6.1 Edouard AKA FireFox

6.1.1 La Map

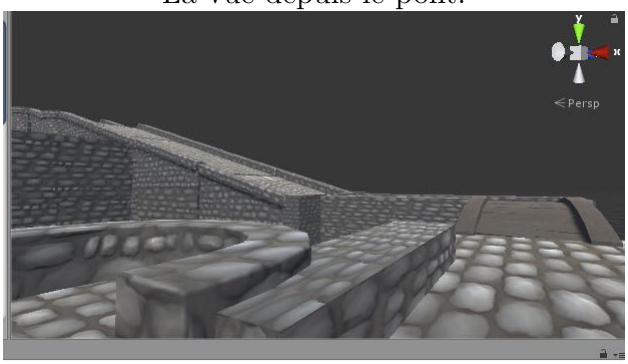
Nous nous sommmes lancé, avec l'aide de Simon, dans la création d'une nouvelle map d'un style différent. Nous avons voulu faire une map ou les joueurs avaient le choix entre se déplacer en hauteur, sur un chemin où il n'y aura pas beaucoup d'ennemi, mais qui sera rempli de piège, et un chemin sans piège, mais qui serait rempli d'ennemi. Nous avons eu beaucoup de problème avec le navmesh des passage en hauteur.



Au départ, les joueurs ont un chemin obligatoire puis doivent prendre un grand pont qui leur permet de voirle reste de la zone. Il y aura dans ce cas des ennemis et également des pièges mais leur nombre sera réduit pour laisser au groupe la possibilité de faire son choix après.

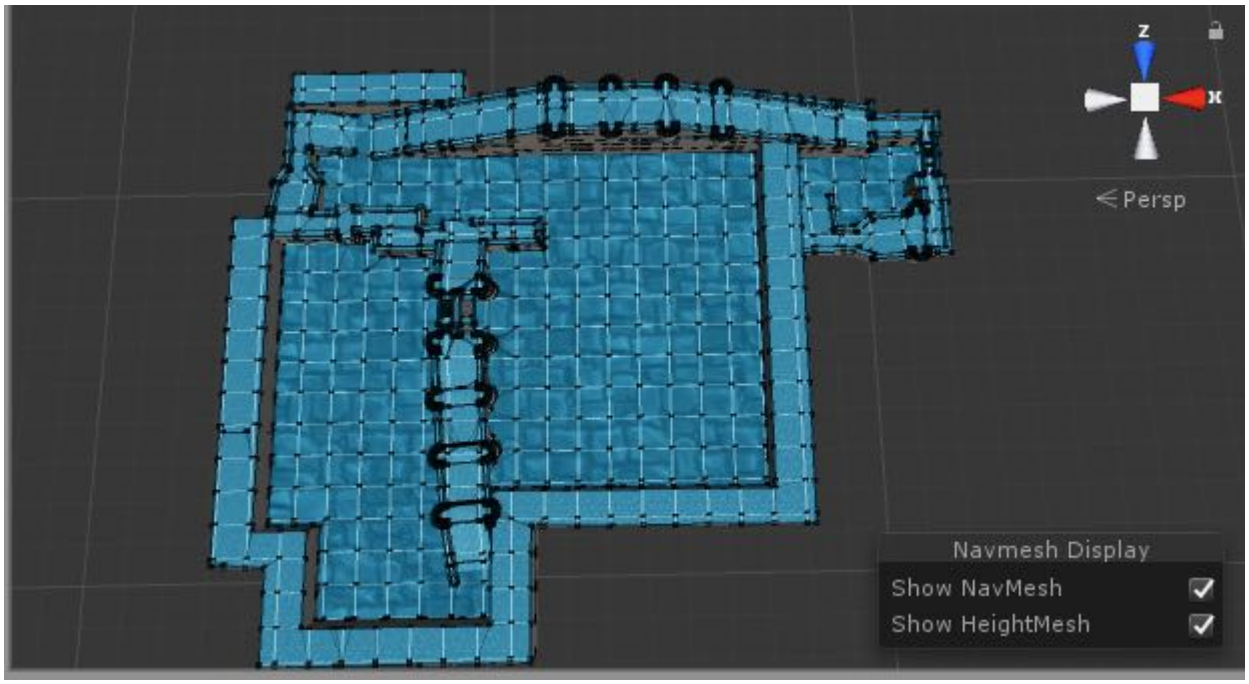


La vue depuis le pont.

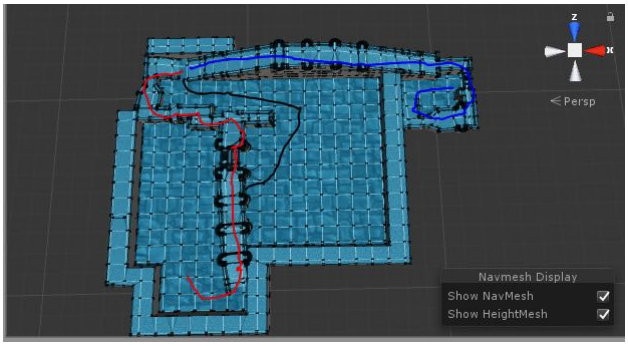


Le chemin de départ.

Nous avons eu quelque problème pour ce qui est du navmesh. En effet, pour permettre que l'on puisse passer sous les ponts, Nous avons du faire de nombreux test en changeant les paramètre, ou en modifiant drastiquement l'aspect de la map que je faisais.

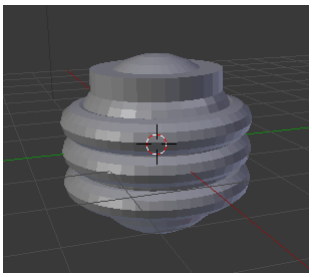


Dans l'idée les chemins de la map sont comme dans l'image ci-dessous.Le chemin bleu est le chemin de départ. Le chemin rouge est le chemin en hauteur rempli de piège et le chemin noir est le chemin avec une horde d'ennemi.

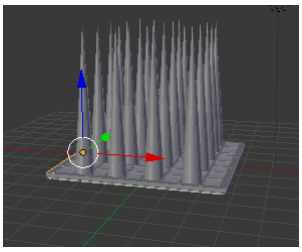


Pour ce qui est des pièges, nous avons créés différents modèles. Nous avons associé le script des portes aux portes elle-même, que nous avons ainsi pu placer sur la première map.

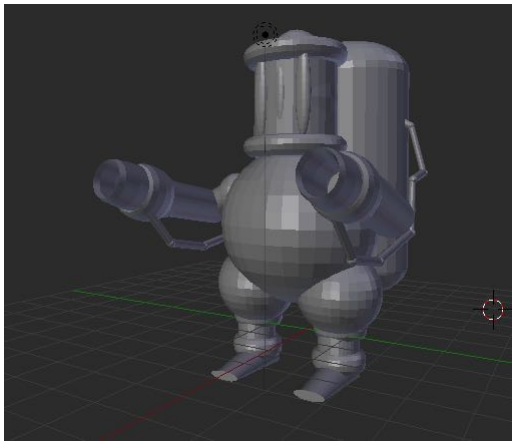
Nous avons également fait un modèle pour une bombe. Les scripts et le modèle étant fait, il ne manque qu'a les intégrer dans les scènes. Elles sont destructibles.



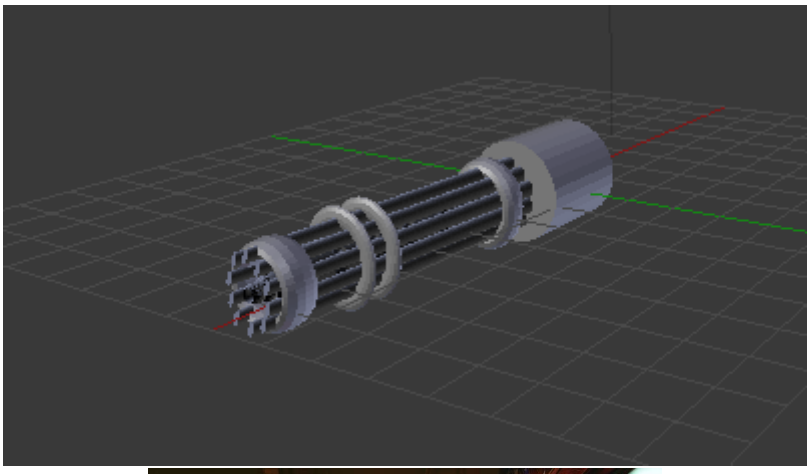
Nous avons également fait un script pour des pièges à piques ainsi qu'un modèle pour ceux-ci.

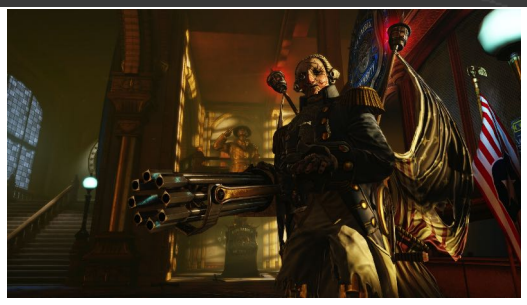


Ce piège est caché dans la structure, le principe étant qu'il met 3 secondes à s'activer, si le joueur se fait toucher par les piques, il perdra des points de vie. Les piques obtrueront le passage et il sera nécessaire de les détruire pour avancer. Pour faciliter le déplacement et pour les repérer, les pointes des piques sortiront du sol et seront rouges. Les pièges feront des dégats quelque soit les ennemis.
J'ai également avancé sur les modèles des ennemis. En effet, nous avons créé un petit ennemi lance-flamme. Nous nous sommes inspiré du lance-flamme que l'on trouve dans Bioshock Infinite. Nous avons déjâ un script pour un ennemi lance-flamme.



Nous avons également fait un modèle de gatling pour le prochain ennemi qui sera inspiré lui d'un autre ennemi de Bioshock Infinite.





Nous comptons avoir fini les modèles des ennemis, des boss et des Joueurs. Nous comptons également faire une troisième map qui sera en vague. Par exemple, les joueurs arriveront au sommet de la tour, et par exemple, actionneront un levier et des ennemis arriveront, puis au bout d'un certain nombre d'ennemi éliminé, un boss arriverait.

6.2 Simon AKA Qui-Gon-Jinn

6.2.1 IA des ennemis

Pour la première soutenance, j'avais mis en place la détection basique de l'ennemi (champ de vision), et la base de l'outil de navigation de unity pour les ennemis (mavmesh), ainsi qu'un chemin de retour dès que l'ennemi n'est plus en vue. Pour cette soutenance, j'ai considérablement amélioré la détection de l'ennemi. En effet, maintenant l'ennemi peut détecter plusieurs joueurs, ce qui n'était pas pris en compte avant, et ainsi, adapte l'IA au multijoueur. Les joueurs étant tous taggés avec le tag "Player" je peux utiliser la commande "GetGameObjectsWithTag("Player") pour détecter tous les joueurs sur la scène et renvoyer un tableau de tous ces joueurs (un maximum de 4 joueurs).

```
NavMeshHit hit;

if (!isFocused)
{
    foreach (GameObject player in Players)
    {
        if (player.activeSelf)
        {
            if (Vector3.Angle(player.transform.position - transform.position, transform.forward) < _fov && (player.transform.position - transform.position).magnitude < 3f
                && !nav.Raycast(player.transform.position, out hit) || player.GetComponent<AudioSource>().minDistance > (player.transform.position - transform.position).magnitude)
            {
                Player = player.transform;
                audio = player.GetComponent<AudioSource>();
                isFocused = true;
                break;
            }
            if ((player.transform.position - transform.position).magnitude < (Player.position - transform.position).magnitude)
            {
                Player = player.transform;
                audio = player.GetComponent<AudioSource>();
            }

        }
    }

}
```

Par exemple ce bout de code définit que si l'ennemi ne détecte pas de joueur ou qu'il ne détecte plus sa précédente cible, il cherchera un nouveau joueur, si un autre joueur est détecté, il changera de cible. Sinon, il va surveiller la cible la plus proche. Par ailleurs, la détection d'un personnage isolé a elle aussi été ameliorée. Maintenant, l'ennemi peut détecter un joueur au son. si l'ennemi entend le son du joueur faiblement, il deviendra suspicieux (augmentation de l'angle de détection, changement d'animation), et si il détecte fortement le son du joueur (sprint, trop proche) le joueur sera détecté, même hors du champ de vision.

```
if (hint || ((targetDir).magnitude < 3f && !nav.Raycast(Player.position, out hit) && angle < _fov || audio.minDistance > targetDir.magnitude))
{
    hint = false;
    isFocused = true;
    _time = 0f;
    nav.isStopped = false;
    anim.SetBool("walking",false);
    anim.SetBool("suspicious", false);
    anim.SetBool("detected", true);
    nav.SetDestination(Player.position);
}
```

Ici on a hint (dont nous parlerons plus tard), la détection en fonction du champ de vision, et la détection par le son. J'ai rendu la détection du son possible en changeant directement les valeurs de distance minimale et maximale de la source audio diffusant les bruits de pas dans le script qui controler le personnage, ainsi, la détection sera directement liée au son que nous entendons (pas de détection entre deux pas). Par ailleurs, l'ennemi si il ne détecte plus le joueur, ne va plus retourner immédiatement à sa place d'origine, en effet il suffisait de se placer dans son angle mort pour qu'il retourne a sa place. maintenant, si le joueur est a une très courte distance de l'ennemi et qu'il n'est plus détecté, l'ennemi ne perdra plus sa trace, et si le joueur est a distance de l'ennemi et qu'il n'est plus détecté

(par exemple après avoir tourné au coin d'une rue) l'ennemi va aller au dernier endroit où le jouer a été détecté et regarder autour de lui quelques instants avant de revenir à son point de départ, comme le montre le bout de code ci-dessous.

```
if ((nav.destination - transform.position).magnitude <= 0.2f)
{
    anim.SetBool("detected", false);
    if (anim.GetBool("walking"))
    {
        if (_time > 3f)
        {
            anim.SetBool("suspicious", false);
            _fov = 60f;
            _time = 0f;
        }
        anim.SetBool("walking",false);
        nav.isStopped = true;
        transform.rotation = initialDir;
    }
    else
    {
        anim.SetBool("suspicious",true);
        if (_time > 3f)
        {
            anim.SetBool("suspicious", false);
            anim.SetBool("walking", true);
            nav.SetDestination(initialPos);
            _time = 0f;
        }
    }
    if (anim.GetBool("suspicious"))
    {
        _time += Time.deltaTime;
    }
}
```

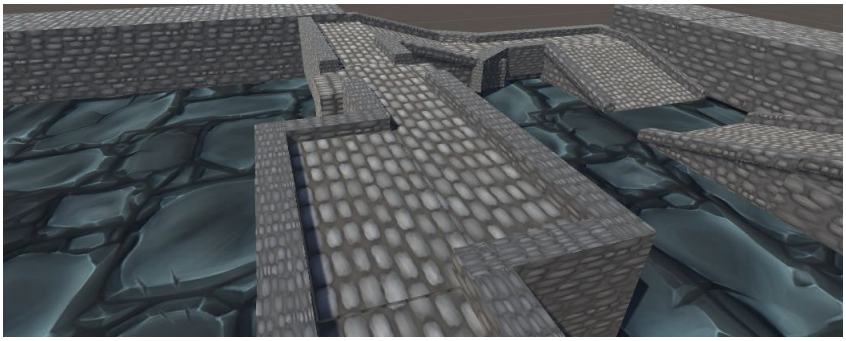
Reparlons maintenant de la variable "Hint" vue précédemment, elle permet a l'ennemi, si il se fait attaquer de loin, de le faire se déplacer vers la source du tir afin de vérifier la présence ou non d'un joueur. l'activation de Hint est gérée directement par le script du sort avec la collision et en détectant le tag "enemy". Ces améliorations de la détection permettent un meilleur traitement de la furtivité, ainsi qu'une augmentation de la difficulté du jeu, en effet, si le joueur court, il se fera détecter d'assez loin. ces améliorations vont de pair avec la partie furtivité que je dirige aussi, qui sera nécessaire pour un personnage furtif, ou même pour finir des niveaux plus vite ou sans se battre. Par ailleurs, avec l'implantation des dégats, les ennemis ont maintenant la possibilité de mourir (mort animée).

Maintenant il resterait a implémenter les dégats des ennemis, en développant leur modèles, en leur donnant des armes, pour qu'ils puissent potentiellement tuer un jouer et s'inscrire dans l'univers du jeu (steampunk). Pour la partie bonus de furtivité, il resterait a implémenter un troisième mode de déplacement plus furtif et/ou donner au ennemis des paternes de déplacements, afin de rendre plus dur l'évitement des ennemis.

6.2.2 Design du niveau

le premier niveau était un niveau uniquement horizontal avec de nombreux passages et obstacles, qui était assez adapté à la furtivité de par ses nombreux chemins et obstacles. cependant le deuxième niveau est orienté très différemment, en effet ce niveau est plus vertical, avec des ponts, mais aussi des pièges, et peu de chemins possibles sinon deux chemins. l'un des chemins, est rempli d'ennemis sans couverture, et l'autre chemin est rempli de pièges sans couverture. C'est un niveau qui est donc moins favorable a la furtivité, mais reste assez libre de choix. De par l'absence de protection, ce niveau favorise les

attaques a distance, et les ponts étant assez étroits, les attaques de zones (mage) peuvent être dévastatrices.



Cependant la partie des pièges de par le nombre d'ennemis limités et de l'impossibilité de se précipiter, la tactique est conseillée. Ce niveau s'annonce donc plus guerrier que le précédent tout en offrant une possibilité plus risquée et demandant tactique. Là où le premier niveau encourageait l'évitement des ennemis, et l'exploration, par sa pluralité de chemins, ce deuxième niveau s'oriente plus sur le combat et le risque que le premier. Il reste maintenant à faire un niveau de vague, plus carré que les précédents.

6.2.3 Site

Le site, toujours créé avec l'outil de GitHub, a été mis à jour et propose maintenant plusieurs pages.

ainsi qu'une interface un peu changée. pour l'instant le site ne contient qu'une description des membres et des liens vers GitHub, un lien de téléchargement de notre rapport. Le multi page permettras à terme de télécharger le jeu, de d'avoir une interface graphique attirante, et décrira notre jeu.

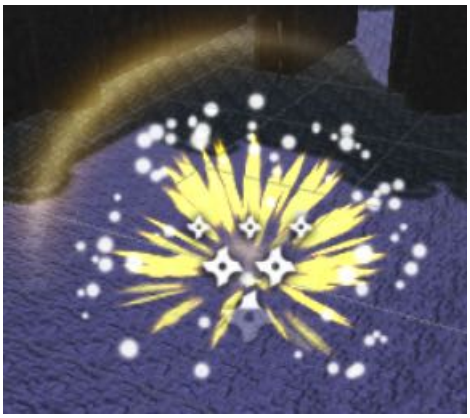
6.3 Lucas AKA PublicStaticVoid

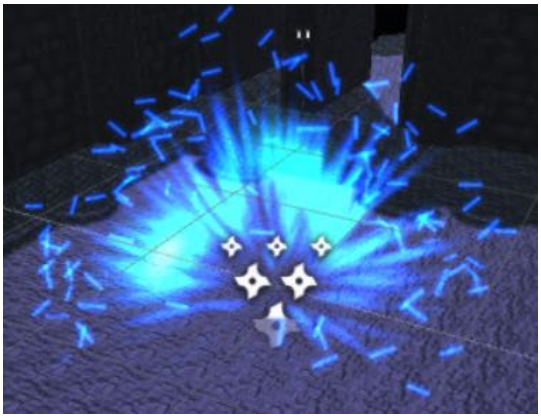
6.3.1 Les sorts

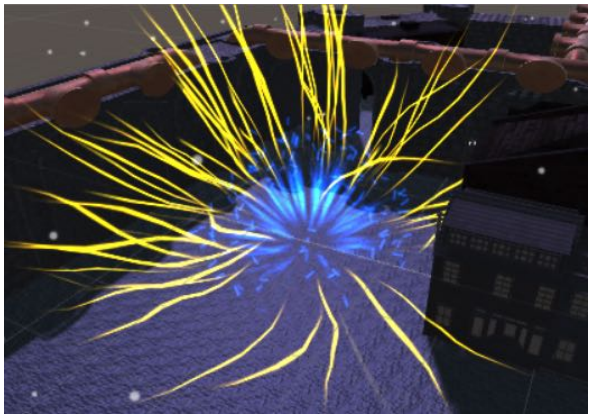
Lors de la dernière soutenance, je vous avait expliqué comment j'avais réussi à créer le premier sort du Mage, qui sera un des personnages jouables. Je vous avait également parlé du deuxième sort, il n'était pas encore fini à l'époque c'est pourquoi il a beaucoup évolué dans son aspect graphique. Depuis la dernière soutenance, j'ai implémenté 6 autres sorts : les deux restants pour le Mage et les quatre autres sont des sorts d'un nouveau personnage, le Guérisseur. Vous remarquerez sûrement que les deux personnages se ressemblent quant à l'apparence de leurs sorts mais les effets produits sur les ennemis et les alliés sont totalement différents. Comme je viens de le mentionner, j'ai aussi donné des effets aux sorts à l'aide de différents scripts que je vous expliquerai dans la suite du rapport, l'activation des sorts demande du Mana qui est l'énergie magique des deux personnages implémentés, chaque sort a un coût de Mana différent et ne peuvent pas être lancés si la réserve de Mana tombe à un niveau inférieur au coût du sort. Au début de la partie, le Mage et le Guérisseur ont chacun 100 points de vie et 100 points de Mana. Mon discours s'articulera en deux parties centrées sur l'animation des sorts et les effets des sorts du Mage et du Guérisseur.

6.3.2 Animation et effets des sorts du Mage

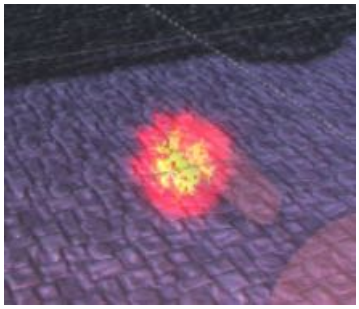
Donc comme je vous le disait, l'animation du deuxième sort du Mage est finie, le concept est, je le rappelle, que le joueur se téléporte dans la direction pointée par la souris sur une distance donnée et limitée, suite à ce déplacement, une explosion magique se produit qui infligera 20 points de dégât aux ennemis proches et la moitié aux alliés proches et demande 20 points de Mana. Cette explosion est en réalité composée de deux explosions (graphiquement uniquement), voici à quoi ressemble les deux différentes explosions et le résultat final :







Parlons maintenant des deux nouveaux sorts du Mage, ce sont les deux sorts les plus puissants et les plus impressionnants du jeu. Le troisième sort a été assez simple à créer mais sa création a pris beaucoup de temps car il est composé de trois différents GameObject, là où le premier sort n'en est composé que d'un et le deuxième sort, de deux (et le dernier de cinq mais je vais y revenir). Ce troisième sort consiste en l'envoi d'une boule d'énergie chaude et lente en ligne droite (qui se déplace grâce à un script appelé Mover que j'ai expliqué la dernière fois) et si elle touche un ennemi ou un joueur, il est instantanément mort et une colonne de flammes se crée dans une explosion qui dureront pas moins de dix secondes à l'emplacement où la boule a touché le malheureux élu, la colonne de feu infligeant 5 points de dégâts par seconde à tous les ennemis et alliés présents dans celle-ci et l'explosion inflige 2 points de dégât par seconde à tous ceux présents dans la zone de l'explosion. Ce sort coûte 60 points de Mana. Voici quelques visuels du sort, la boule d'énergie, l'explosion et la colonne de feu :

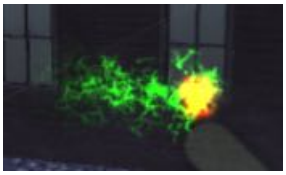


Et enfin, le dernier sort du Mage, le plus puissant, le plus impressionnant et le plus dur à réaliser : l'Astéroïde. En effet, l'ultime sort du Mage et composé de cinq GameObject, j'ai dû également créer un script Mover rien que pour le lancement de ce sort que j'ai appelé MeteorMover. C'est le sort qui m'a demandé le plus de temps à créer à ce jour car j'ai éprouvé quelques difficultés à le faire fonctionner. Le sort s'articule en trois étapes, la Canalisation, la chute de l'astéroïde et enfin l'Impact au sol à la suite de la chute. Lors de la Canalisation, le Mage et entouré d'une aura magique, il utilise tout son pouvoir pour créer l'astéroïde en altitude. En réalité, au début de la Canalisation, un GameObject, appelé MeteorLauncher, invisible dans le jeu est lancé à très grande vitesse pour que le sort puisse se lancer, que le joueur ne rate pas sa cible car les conséquences sur le joueur sont dramatiques lorsqu'il utilise sont ultime sort : le joueur doit utiliser 100 points de Mana et 50 points de vie pour utiliser le sort, durant la Canalisation il est immobilisé et ne peux plus lancer de sorts sachant que celle-ci dure 10 secondes. Donc le MeteorLauncher touche la cible et l'astéroïde est créé en altitude. Je vais faire ici un bref point pour expliquer pourquoi j'ai créé un MeteorMover car je viens de parler de création d'objet en altitude et non à l'emplacement de la cible. En effet, le Mover initial a été légèrement modifié depuis la dernière fois, au contact d'un ennemi ou d'un joueur, le Mover fait apparaître l'effet voulu à l'endroit où le contact à été établi, or pour l'astéroïde, on veut qu'il apparaisse au dessus de la cible, donc pour ne pas modifier le Mover qui est commun à plusieurs sorts, il faut en créer un autre spécifique à ce sort. L'astéroïde tombe, il traverse le sol (problème rencontré : la vitesse de l'astéroïde est trop grande pour que le collision soit

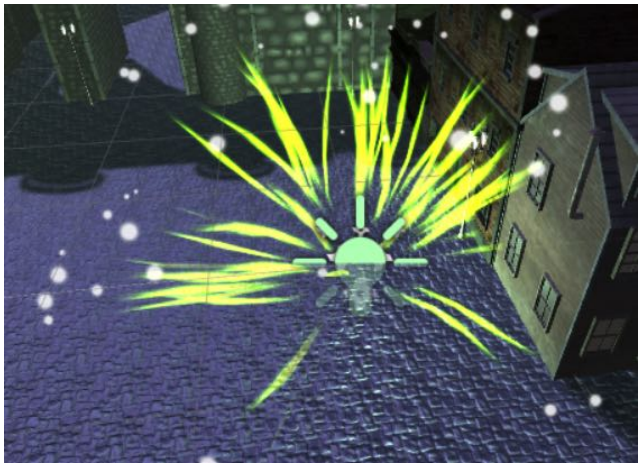
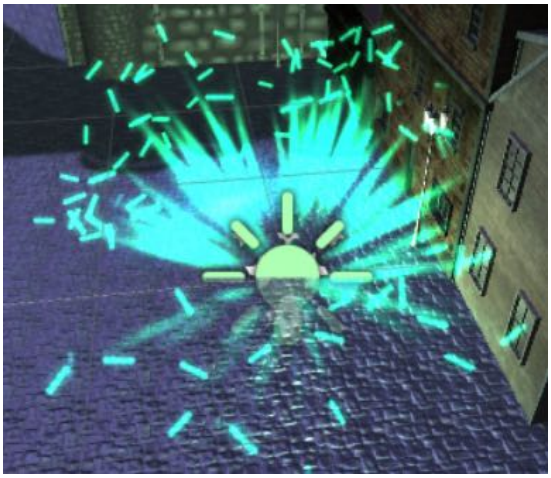
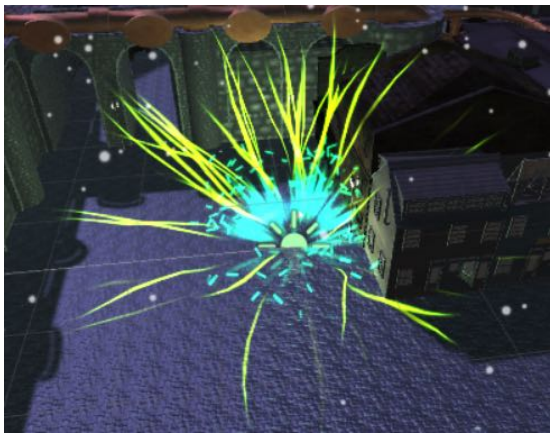
détectée, il fallait donc le ramener au niveau du sol lorsqu'il le dépasse) et une explosion d'énergie se crée, tout joueur ou ennemi dans la zone au centre de l'impact voit ses points de vie descendre à 1 point et dure 12 secondes, l'astéroïde reste sur le sol pendant toute la durée de la partie et tout personnage le touchant meurt(nouveau problème : avec le système habituel de détection de collision, l'explosion était lancée en boucle, j'ai donc créé un nouvel astéroïde qui ne déclenche pas d'explosion, j'ai supprimé le collider du premier qui ne sert plus à rien vu qu'il déclenche l'explosion, crée le nouvel astéroïde sur le sol et est supprimé dès qu'il est plus bas que le sol). Il est temps maintenant de donner quelques visuels de la canalisation, de l'astéroïde et de l'explosion d'énergie :

6.3.3 Animation et effets des sorts du Guérisseur

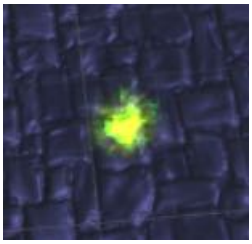
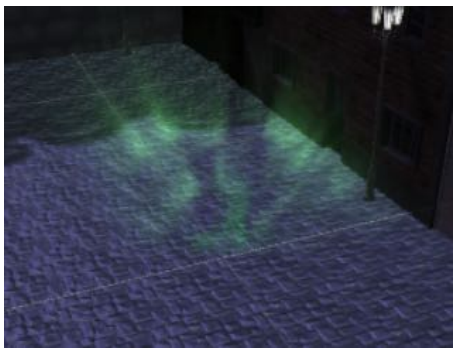
Le premier sort du Guérisseur est le même que celui du Mage sauf que les dégâts infligés aux ennemis sont de 5 au lieu de 10 et rendent la moitié de ces dégâts en points de vie au joueur s'il est touché au lieu de lui faire perde la moitié de ces dégâts. Ce sort coûte 5 points de Mana autant pour le Mage que pour le Guérisseur. Voici à quoi il ressemble :



Le deuxième sort du Guérisseur est le même que celui du Mage pour ce qui est de l'animation, pour ce qui est des effets, ils sont totalement différents sur les alliés : l'explosion infligera 10 points de dégât aux ennemis proches et redonnera en points de vie la moitié aux alliés proches et nécessitera 20 points de Mana pour être lancé. Cette explosion est en réalité composée de deux explosions (graphiquement uniquement), voici à quoi ressemble les deux différentes explosions et le résultat final :

Le troisième sort du Guérisseur consiste en l'envoi d'une boule d'énergie vitale et lente en ligne droite et si elle touche un ennemi ou un joueur, l'ennemi touché perd 50 points de vie et le joueur touché regagne 25 point de vie. Ce sort nécessitera 60 points de Mana pour être lancé. Voici quelques visuels du sort, la boule d'énergie et l'explosion :

Et enfin, le dernier sort du Guérisseur, le plus utile : la zone de soin. En effet, l'ultime sort du Guérisseur va rendre 5 points de vie par seconde à tous les joueurs présents dans la zone de soin et inflige 2.5 points de dégât aux ennemis présents dans la zone, le sort nécessite 100 points de Mana et le joueur ne peut plus lancer de sorts pendant la durée de celui-ci :

6.4 Menu

Nous avons aussi implémenté un menu. Il possède 4 options, un pour jouer en solo qui redirige vers la sélection de la classe, une autre option pour pouvoir lancer la partie multijoueur qui à été implémenté par Ouwéis, une troisième option qui redirige vers les options et permet de régler le volume pour l'instant, ainsi qu'une dernière option pour quitter le jeu.

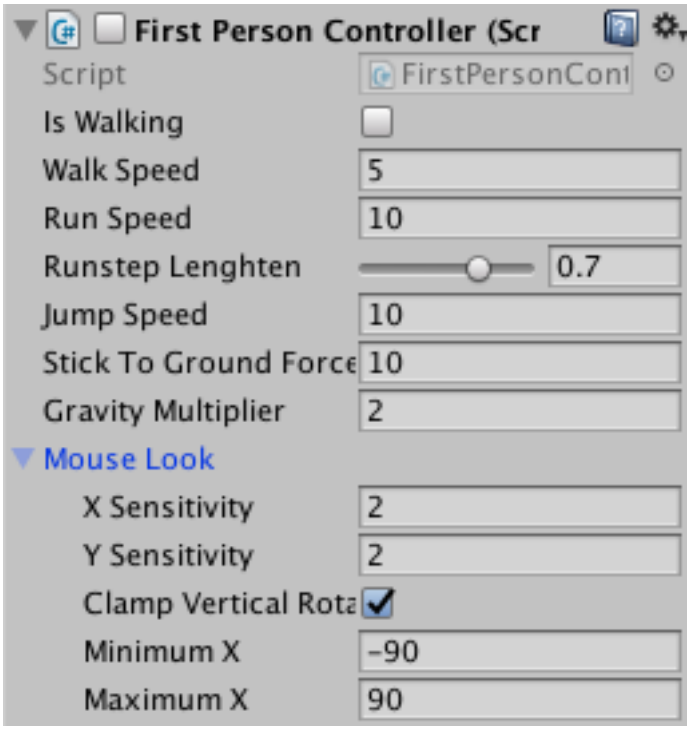
6.4.1 La suite des évènements :

Pour la prochaine soutenance, il faut que je termine l'animation du deuxième sort et que j'en fasse au moins 6 autres avec un objet spécial à récupérer sur la map. Je dois également faire une ébauche du menu pricipal du jeu et faire un boss de niveau. Et enfin je dois commencer à scripter mes sorts pour qu'il aient un réel impact sur les ennemis et les alliés.

6.5 Ouwéis AKA Bûcheron

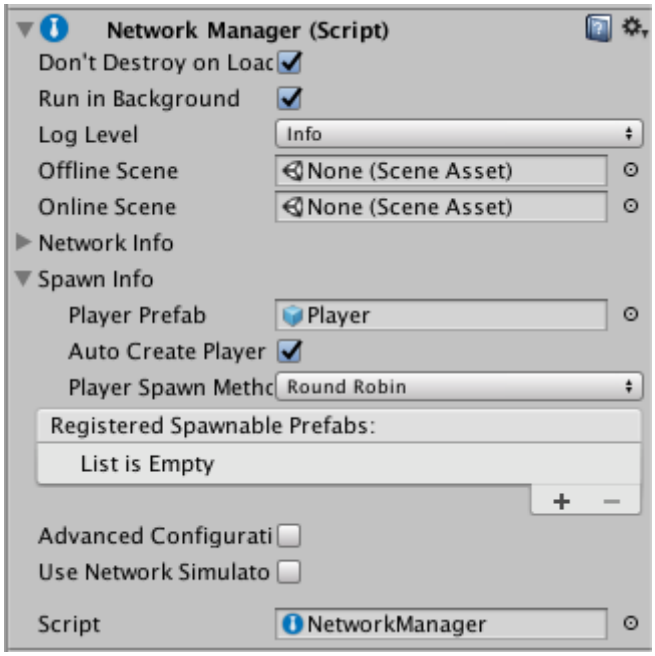
6.5.1 Contrôles Basiques

Les contrôles basiques ayant été implémentés pour la dernière soutenance grâce au script First Person Controller j'ai donc pu me concentrer sur d'autres partie du jeu, et en particulier le réseaux.

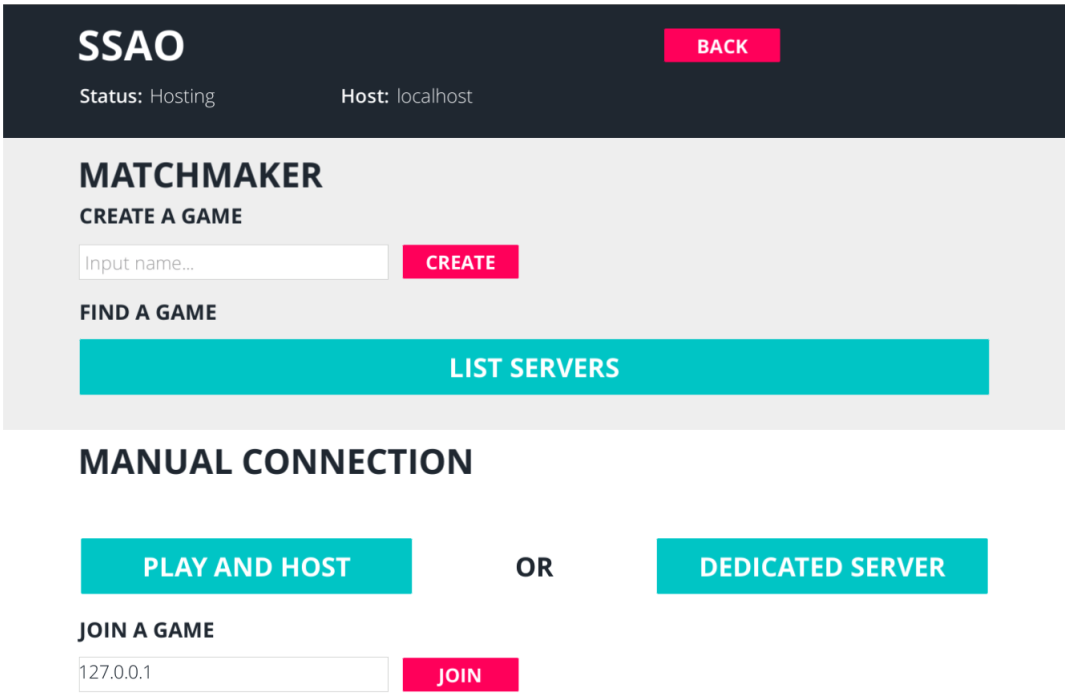


6.5.2 Réseau

Lors de la dernière soutenance nous utilisions un composant d'Unity, le Network Manager pour gérer la gestion du réseaux et en particulier la connection ainsi que la gestion du spawn des joueurs grâce au champ Player Prefab.

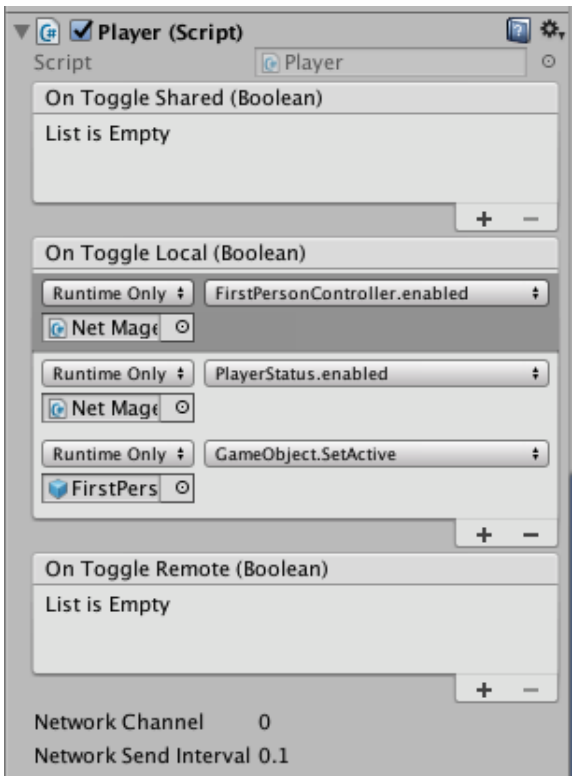


Mais le Network Manager possèdes aussi des inconvénients, en particulier le fait qu'il est lié à une seule scène ce qui rend le changement entre différentes map plus compliqué. Il a donc fallu nous tourner vers une autre solution. Après quelques recherches nous avons trouvé l'asset Network Lobby de Unity qui implémente toutes les fonctions du Network Manager tout en augmentant les possibilités.



Elle possède pour l'instant une interface basique mais elle nous permet de gérer tout ce qui est en rapport avec le réseaux.
De plus ce Lobby pour le multi joueur est accessible directement depuis le menu principal du jeu.
Elle aura à terme un design plus dans l'esprit de notre jeu ainsi que des fonctionnalités en plus.

Nous avons de plus améliorer le script qui s'occupe de gérer les différents joueurs.



De plus lors de la dernière soutenance la partie réseaux était séparée du projet elles sont desormais réunnis.

6.5.3 Animations Et Sons

Les ennemis et et les joueur possèdait déjà des animations lors de leurs déplacement. Ils possèdent désormais des animations lors de leurs morts. Ces différentes animations sont de plus implémentés en réseaux.

De plus les joueurs effectuent des sons lors de leurs différents mouvements. C'est à dire lors de leurs déplacement mais aussi lors des sauts.

Ces sons sont joués pendant leurs différentes animations et sont aussi partagés en réseaux. C'est à dire que les différents joueurs entendent les sons faits par les autres.

Nous avons aussi une bande son pour la première map qui est une musique libre de droit. Nous avons choisi la musique "Mr Gear' qui est une musique dans le style Steampunk et correspond donc parfaitement à l'ambiance notre jeu.

6.5.4 Travail à faire pour la prochaine soutenance

Pour la prochaine soutenance il faut que je termine la partie réseaux, les différentes animations. Il faut aussi que je rajoutes quelques annimations et ainsi que quelque sons. Mais aussi que j'aide à l'impélementation des boss et des codes de triches.

6.6 Logiciels Utilisés

Nous avons utilisés les logiciels suivant lors de la réalisation de notre projet :
- Unity : moteur de jeu 2D/3D permettant notamment grâce à des outils intuitifs (moteur physique, audio...) de réaliser des jeux en réseaux et des animations.

- Git et GitHub : Logiciel de gestion de version qui nous permettra de partager notre code entre nous et de le stocker sur GitHub.

- Visual Studio / Rider : Permettent la programmations des différents scripts C#

- Blender : Logiciel de modélisation, d'animation et de rendu en 3D.

6.7 Problèmes rencontrés

Comparé à la dernière soutenance nous avons eu beaucoup moins de problèmes avec GitHub parce que Ouwéis s'occupait des merges. Et nous avons aussi eu des problèmes avec l'implémentation du réseaux.

7 Annexes

7.1 Assets

Assets employés pour la map :
https ://assetstore.unity.com/packages/3d/environments/industrial/storage-building-50430
https ://assetstore.unity.com/packages/3d/environments/industrial/building-kit-1-68179
https ://assetstore.unity.com/packages/essentials/asset-packs/unity-particle-pack-73777
https ://assetstore.unity.com/packages/3d/props/exterior/victorian-pbr-streetlights-50828
https ://assetstore.unity.com/packages/3d/environments/urban/victorian-pie-shop-21541
https ://assetstore.unity.com/packages/3d/props/industrial/pipes-kit-64170
https ://assetstore.unity.com/packages/2d/textures-materials/floors/hand-painted-stone-texture-73949
https ://assetstore.unity.com/packages/2d/textures-materials/metals/yughues-free-pbr-metal-plates-35362

Assets employés pour les unités :
https ://assetstore.unity.com/packages/3d/characters/humanoids/strong-knight-83586
https ://assetstore.unity.com/packages/3d/characters/jeremy-base-male-character-33083
https ://assetstore.unity.com/packages/3d/environments/fantasy/modular-fantasy-bridges-99940

Asset employé pour les sorts :
https ://assetstore.unity.com/packages/vfx/particles/spells/ky-magic-effects-free-21927

Asset employé pour les textes du menu :
https ://assetstore.unity.com/packages/essentials/beta-projects/textmesh-pro-84126

Asset employé pour le lobby :
https ://assetstore.unity.com/packages/essentials/network-lobby-41836

8 Conclusion

Malgré bien des difficultés tant à la fois technique que de défis dû à nos compétences, nous sommes dans les temps pour ce qui est de notre avancement, nous avons même de l'avance dans certain domaine. Nous avons déjà de nombreuse idées et de solution pour continuer ce projet.