Arktos

Skylab

Rapport de soutenance n°1

Raphaël Gonon Julie Durandeau Mathis Villemin Grégoire Lefaure

7 mars 2022

Introduction

Notre histoire se déroule dans un univers parallèle appelé Arktos, gouverné par la magie. Ses habitants ont acquis des pouvoirs en apprenant à maîtriser les énergies émanant des étoiles, et plus particulièrement d'une constellation bien connue de tous : la Grande Ourse.

« Tout allait bien à Arktos, un monde où toutes les espèces vivaient en harmonie, où chacun était en accord avec la nature et où tout le monde se servait de la magie... jusqu'au jour où la Grande Ourse s'est mystérieusement éteinte. C'est alors que commença le drame. Tous les habitants perdirent un à un leurs pouvoirs. Pour tenter de trouver une solution à ce problème de taille, les hautes instances d'Arktos nommèrent un nouveau leader : Sonis. Les jours passèrent, mais ces derniers ne parvinrent pas à trouver une solution viable... Les dirigeants d'Arktos eux-même perdaient leurs pouvoirs les uns après les autres et n'étaient plus en mesure de maintenir l'ordre. C'est alors qu'Arktos tomba dans un sombre chaos, et la loi du talion reprit le dessus sur l'équilibre magique...

Mais c'était sans compter sur un alchimiste, dénommé Nuri qui, lorsque tout espoir semblait vain, se présenta face au leader d'Arktos. Nuri était toujours en pleine possession de ses pouvoirs magiques. Une lueur d'espoir se présentait enfin ! De plus, Nuri savait qu'il n'était pas le seul. En effet tout au long de son périple pour atteindre les hautes instances de l'univers, il avait rencontré plusieurs autres divergents, des personnes qui, comme lui, pouvaient toujours utiliser la magie.

Sur ces mots, Sonis lança un appel à tous les citoyens pour rassembler les quelques divergents de l'univers. Après de nombreuses semaines, sept divergents se présentèrent, prêts à tout pour sauver le monde. Dès lors commença une lutte acharnée pour rallumer la Grande Ourse... »

Arktos est un jeu de plateforme retraçant la quête de nos aventuriers souhaitants redonner son éclat à la Grande Ourse. Notre jeu sera composé de deux modes : solo et multijoueur. Dans les deux modes les joueurs suivront le même fil conducteur, à savoir des quêtes liées à l'histoire d'Arktos. Il sera développé en 2D sur Unity, à l'aide du langage de programmation C#.

Le joueur sera plongé dans différents environnements où il devra survivre à des hordes d'ennemis et résoudre des énigmes pour pouvoir avancer dans le jeu. Ce dernier sera composé d'un total de sept environnements différents, chacun témoignant de l'avancée de nos héros dans leur quête commune.

L'intelligence artificielle sera utilisée dans le déplacement et les actions des ennemis. L'objectif est de faire en sorte que ces derniers s'adaptent au niveau des joueurs et de rendre l'expérience utilisateur la plus agréable possible tout en frustrant ce dernier pour qu'il ait envie de continue à jouer.

Skylab se compose de quatre élèves de SUP en cycle francophone : Raphaël Gonon (E1), Julie Durandeau (E1), Mathis Villemin (E2) et Grégoire Lefaure (E1).

Dans ce cahier des charges, vous trouverez plus de détails concernant le jeu en lui-même et notre organisation pour mener à bien notre projet.

Table des matières

1 Présentation Générale

1.1 Le Projet

L'objectif de ce projet est de mener à bien le développement de Arktos, un jeu de plateforme dont l'objectif principal est de résoudre différentes énigmes tout en survivant aux attaques des ennemis. Arktos regroupe deux modes de jeu : la campagne, au sein de laquelle il vous faudra résoudre un total de 7 niveaux pour réussir à compléter l'objectif principal, auxquels viendront s'ajouter de nombreux easter eggs et des quêtes subsidiaire, et le mode multijoueurs qui sera sensiblement similaire avec certaines fonctionnalités de jeu qui seront disponibles dès le lancement de celui ci afin d'améliorer la jouabilité. Chaque niveau suivra le modèle suivant : le joueur devra résoudre une énigme (quête principale de chaque niveau) afin de passer au niveau suivant et cela en survivant aux hordes de monstres qui tenteront de l'en empêcher.

Dans ces deux modes de jeu, il sera possible d'incarner 8 personnages, chacun avec des capacités, des attaques et des mécaniques de gameplay différentes. Au commencement du mode campagne le joueur n'aura accès qu'à un seul personnage, l'alchimiste Yuri. Les autres seront débloquables en progressant dans la résolution des niveaux. Pour déverrouiller le huitième personnage il faudra résoudre une énigme englobant tout l'univers que nous avons créé, en s'appuyant sur des indices cachés dans le jeu et dans le code source de notre site web. Comme énoncé dans notre cahier des charges, nous avons l'intention de mettre en place un CTF [1] sur notre site web. La résolution de ce dernier permettra d'obtenir un indice et de débloquer un personnage. Par ailleurs il sera possible de jouer avec la gravité, et d'inverser cette dernière afin d'accéder à des endroits inaccessibles. Cette mécanique sera au cœur de la résolution des énigmes, des easter eggs et permettra au joueur d'échapper aux monstres les plus faibles.

D'autre part, nous travaillons sur la création de tout un univers pour Arktos, afin de renforcer l'expérience de notre joueur. Les graphismes nous permettront de plonger le joueur dans notre univers, un monde régit par la magie et plongé dans une période sombre de son histoire. De plus, les ambiances sonores et lumineuses accompagneront le joueur tout au long de son expérience, en variant en fonction des situations (musiques davantage dynamiques pendant les combats, moins de visibilité pour les scènes dans l'espace, etc...).

1.2 L'équipe Skylab

1.2.1 Raphaël Gonon

Bonjour, je m'appelle Raphaël et je suis en charge de la coordination pour ce projet. J'ai suivi un bac général avec la spécialité NSI, ce qui m'a permis de développer quelques projets dans le cadre scolaire. J'en garde une très bonne expérience, et j'espère pouvoir apprendre de mes erreurs pour gagner en efficacité dans ce nouveau projet car c'est la première fois que

1. Capture The Flag

j'entreprends un projet informatique d'une telle ampleur.

En termes de compétences techniques, j'avais jusque-là développé des projets en python à l'aide du module Pygame, ou des sites internet avec les outils HTML/CSS. Je ne connais pas encore bien Unity mais ce projet sera l'occasion parfaite pour m'y jeter à bras ouverts.

Par ailleurs j'ai une expérience dans la vente de textile qui, même si le marché et le public sont très différents, pourra nous être utile pour la partie commerciale de notre projet.

1.2.2 Julie Durandeau

Bonjour ! Je me présente, je m'appelle Julie. Je voudrais bien réussir mon projet, et faire un super jeu ! Je suis de la promo 2026, j'ai pris NSI en première où j'ai pu découvrir le monde du code, notamment avec python, et toujours aimé programmer. L'optique de faire un jeu vidéo en équipe me motive fortement, et d'autant plus avec cette équipe de choc ! Je remplirai dans la création du jeu le rôle de directrice artistique. J'espère pouvoir m'épanouir pendant cette expérience unique et que nous pourrons vous présenter un super projet !

1.2.3 Mathis Villemin

Bonjour ! Je suis Mathis Villemin. J'ai fait NSI et Maths jusqu'en terminale. Je tiens à dire que je suis heureux de commencer ce projet. Premièrement car j'apprécie l'équipe avec laquelle je vais évoluer mais aussi car j'attends beaucoup des problèmes que nous allons devoir résoudre en équipe. C'est une première pour moi car, bien qu'ayant mené à bien de gros projets d'informatique (pour la terminale), ces projets n'étaient pas en équipe, ce qui fait rentrer en compte une nouvelle variable. Pour ce projet, je serai responsable du gamedesign et plus généralement de tout ce qui est lié à l'expérience utilisateur. Je dois donc m'assurer que le jeu soit facile à comprendre autant pour ce qui concerne les règles que pour les mécaniques et les missions à accomplir. Je dois aussi rendre le jeu agréable pour le joueur. Bref, j'ai à la fois hâte de commencer ce projet et beaucoup d'attente venant de celui-ci !

1.2.4 Grégoire Lefaure

Hello ! Moi c'est Grégoire, je suis un épitéen depuis bientôt 5 mois. Ayant fait NSI en première et en terminale, cela a été pour moi l'occasion de faire de petits projets individuels en python et de découvrir de nouvelles choses.

En tant que gérant technique, j'espère que ce projet me permettra d'en apprendre d'avantage, aussi bien techniquement parlant avec la partie développement de notre jeu et la gestion des infrastructures (le site web, le repertoire git, etc...) qu'humainement parlant avec tout ce qui concerne le travail d'équipe. Cette dernière est une partie aussi importante que la partie technique dans ce projet.

On peut donc dire que j'ai de nombreuses attentes en ce qui concerne ce projet, et j'espère que cette expérience m'apportera beaucoup, aussi bien techniquement qu'humainement parlant.

1.3 Répartition des tâches

La répartition des tâches de ce projet s'est faite très facilement en fonction des envies et des points forts de chacun. Nous avons commencé par travailler sur des points nécessaires au fonctionnement du jeu afin d'obtenir le plus rapidement possible une scène jouable comportant les fonctionnalités voulues. C'est pourquoi au cours de cette première période, Grégoire s'est occupé de toute la structure des classes de nos personnages, du développement du changement de gravité et de la création d'une scène de test jouable. Mathis de son côté s'est appliqué à faire différents menus principaux, de pause, de réglages et de les intégrer à la scène test de Grégoire. D'autres part Raphaël s'est appuyé sur la scène test de Grégoire afin de paramétrer les fonctionnalités de multijoueur. Tout cela n'aurait pas été possible sans les réalisations artistiques de Julie pour la conception des personnages et de leurs animations ainsi que des images d'arrière-plan.

Pour la suite du projet, chacun va continuer d'avancer à travailler sur ses parties jusqu'à ce qu'elles soient terminées. Une fois ces dernières réalisées, nous travaillerons en fonction du tableau ci-dessous :

Tâches	Raphaël	Julie	Mathis	Grégoire
Site				★
Graphismes	☆	★		
Musiques	★	☆		
Menus		☆	★	
IA	★	★	★	★
Multijoueur	★	★	★	★
Combats		★	★	
Énigmes	★			★

TABLE 1 – Répartition des tâches

Dans le tableau ci dessus, les étoiles pleines indiquent les personnes qui sont en charges de la section concernée et les étoiles vides représentent leurs adjoints.

Nous avons fait le choix de travailler tous ensemble sur le développement de l'intelligence artificielle car nous considérons sa réalisation comme la tâche la plus complexe et la plus enrichissante. En effet, aucun d'entre nous n'a déjà essayé de concevoir une intelligence artificielle, bien que nous soyons tous sensibles à ce sujet.

2 État actuel du projet

2.1 Site web

Afin de rendre notre projet accessible à tous, nous avons mis en place un site web (`skylab-arktos.com`) pour partager les différentes versions de notre jeu ainsi que toutes les informations que nous avons estimées utiles à l'utilisateur. La partie concernant les différentes versions de notre jeu est pour l'instant assez vide puisqu'il n'y a pas encore de « jeu »à proprement parler, mais la section permettant d'obtenir plus d'informations est, quant à elle, remplie avec tout ce que nous estimons à ce jour nécessaire.

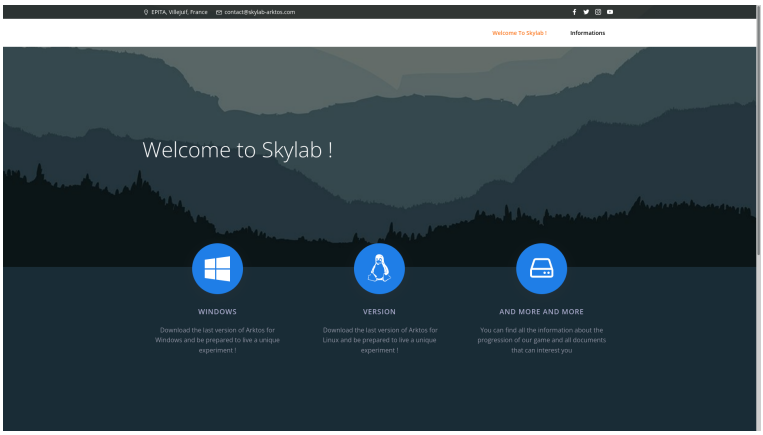


FIGURE 1 – Page d'accueil

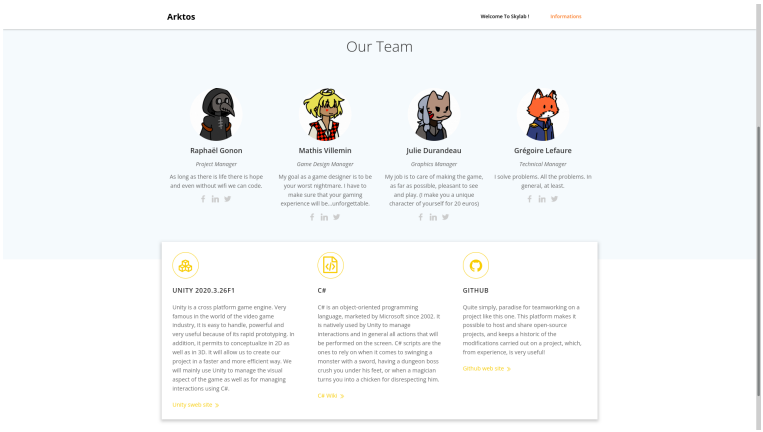


FIGURE 2 – Présentation de l'équipe et des outils utilisés

2.2 Structure du projet

La structure du projet suit l'architecture suivante :

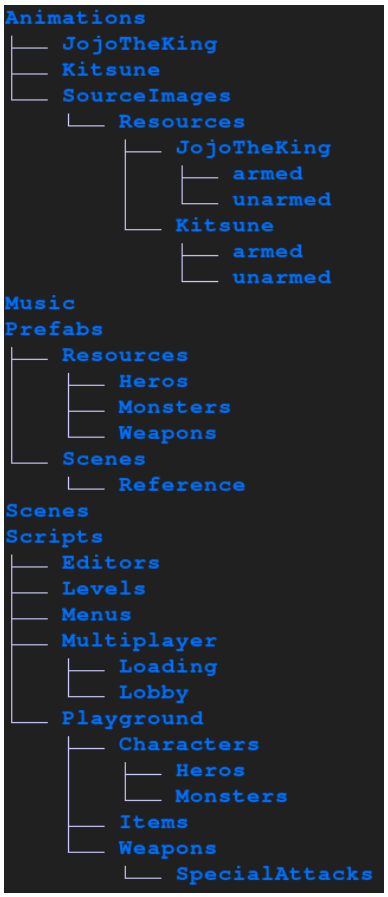


FIGURE 3 – Architecture du projet

Tout d'abord le dossier principal « Playground »regroupe les dossiers « Characters », « Items »et « Weapon ». Ces différents dossiers regroupent l'ensemble des classes permettant l'instanciation des objets de nos scènes.

Le dossier « Characters »regroupe les dossiers « Hero »et « Monsters »ainsi que le fichier « Characters », qui contient la classe mère dont vont hériter chaque classe du dossier « Hero », ainsi que les monstres instanciés dans le dossier « Monsters »et le fichier CharacterController2D.cs », utilisé pour définir les mouvements du player.

Le dossier « Item »contient pour l'instant seulement le fichier Item.cs qui gère l'ensemble des interactions avec des objets spéciaux tels que des potions de vie et autres atouts que nous pouvons accumuler dans le gameplay. De plus, ce sera dans ce dossier que nous créerons l'objet pour ajouter un inventaire à notre personnage.

Le dossier « Weapon »centralise le dossier SpecialAttacks et les fichiers correspondants aux attaques principales de chacun de nos personnages (nos objets Hero). Le dossier contient lui les fichiers correspondants a l'attaque special de nos personnages.

D'autre part, le dossier « Multijoueur »regroupe les dossiers « Loading »et « Lobby »et le fichier SpawnPlayer.cs. Le fichier SpawnPlayer.cs gère le spawn des joueurs pour la partie multijoueur en créant une instance de notre objet sur le serveur. Le dossier Loading recueille le

fichier ConnectToServer.cs qui comme son nom l'indique lorsque la scène se lance, établit la connexion avec le serveur et lance la scène de lobby pour créer ou rejoindre une salle de jeu.

Le dossier « Menu »réunit l'ensemble des scripts et scènes nécessaires au bon fonctionnement des menus et de la mise en relation des différentes scènes entre elles. Le dossier « Level »quant à lui contient les scripts permettant de lancer notre scène de test.

L'ensemble des dossiers en dehors du dossier « Scripts »contiennent les fichiers utilisés pour Unity tels que les prefabs, les fichiers d'animation, etc...

De plus, dans un soucis de clarté, nous avons mis en place un système de Readme, présent dans chaque sous dossier de « Scripts »afin de détailler l'utilité des fichiers présents dans ce sous-dossier, ainsi qu'une architecture détaillée des fichiers présents dans ce sous-dossier. Dans le même but, nous essayons de commenter au maximum notre code [2] afin que nos partenaires puissent réutiliser et facilement adapter ce dernier à leurs besoins.

En ce qui concerne les autres dossiers, ils possèdent une architecture plus simple car ils contiennent souvent moins de fichiers, bien qu'ils ne soient pas moins importants. En bref, nous pouvons voir le dossier « Prefabs »qui regroupent tous les objets Unity que nous avons besoin de stocker ; le dossier « Scene », qui comme son nom l'indique, regroupe les différentes scènes de notre jeu ; le dossier « Animations »dans lequel sont organisés les différentes animations de nos personnages ; et enfin le dossier « Music »qui va regrouper la bande son de notre jeu.

2. Sans tomber dans l'exagération bien entendu

2.3 Testing Scene

FIGURE 4 – Capture d'écran de notre scène de test

La Testing Scene est la scène où nous avons fait l'ensemble de nos tests jusqu'à présent. Elle nous a servi de scène de référence pour le développement des menus et du multijoueurs. Notre objectif avec cette scène était de créer une première impression avec toutes les fonctionnalités qui seraient présentes dans l'ensemble des futurs niveaux. Ainsi, dès lors que cette dernière sera achevée, nous pourrons créer les 7 niveaux qui composeront notre jeu. De ce fait, cette scène regroupera l'ensemble des éléments de décors singuliers à l'univers d'Arktos. Les différentes box permettent d'illustrer au mieux les divers déplacements du personnage. Il est possible dans le jeu de se déplacer horizontalement, de sauter et d'inverser la gravité de notre personnage. Cette dernière fonctionnalité permet lors d'une partie multijoueur de créer de nouvelles énigmes basées sur la coordination et la communication, puisqu'un inversement de gravité n'est pas personnel, il s'applique à tout le monde. De plus, les mouvements horizontaux et l'inversion de gravité sont animés. Enfin, c'est dans cette scène que se retrouvent nos personnages lors d'une session multijoueur.

2.4 Graphismes - Bande Son

Pour la partie graphique, nous avons actuellement deux personnages jouables, les designs de nos menus ainsi que des images de background pour nos scènes et une première bande son. Notre premier personnage est notre personnage de cœur car il représente notre bien aimé ancien directeur Monsieur Joël Courtois. Nous souhaitions faire un personnage à son effigie pour qu'il soit en permanence avec nous et pour ne pas oublier ses enseignements et sa vision du monde ainsi que du travail. Ce personnage sera le personnage caché énoncé précédemment, qui sera débloquable sous certaines conditions, en commençant par la résolution du CTF sur notre site internet. Vous le reconnaitrez probablement [3] dans la figure 5.

FIGURE 5 – Monsieur Joel Courtois

Notre second personnage est appelé Kitsune, c'est un renard possédant des pouvoirs magiques. Pour ce dernier, nous avons commencé à développer les animations de ses attaques dont vous pouvez voir les frames ci-dessous.

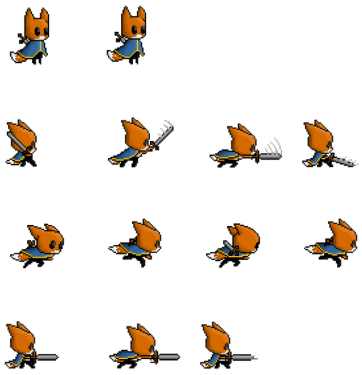


FIGURE 6 – Frame du Kitsune

Enfin nous avons déjà réalisé une bande son pour notre jeu, reprenant les ambiances graphiques et cherchant à compléter l'immersion du joueur. Cette derniere est disponible a sur notre github a l'adresse suivante : `https://github.com/Gregoire-perso/arktos/blob/main/Unity_project/Assets/Music/Arktos_track_1.mp3`

3. On l'espère en tous cas

2.5 Multijoueurs

La conception du multijoueur s'est décomposé en 2 étapes majeures : La première a été de se documenter sur l'ensemble des services, librairies disponibles permettant la réalisation d'un jeu multijoueur. Les noms qui revenaient le plus souvent étaient Photon et Mirror. Nous avons finalement opter pour Photon, car la documentation de Photon est très claire et compréhensible facilitant ainsi la résolution de potentiellement problème rencontré lors du développement, deuxieme car étant l'un des assets les plus utilisés cela nous offrait encore plus de ressources pour résoudre nos problèmes avec les forums de questions et tutos quant au développement d'un jeux multijoueur.
La seconde a été de réaliser la partie multijoueur de notre jeu. Pour y parvenir nous avons commencé par créer un serveur photon gratuit sur leur site internet et de créer une scène de chargement dans Unity permettant de se connecter à ce dernier. En termes de code, ce dernier est très minimaliste car il s'appuie sur les fonctions prédéfinies de la bibliothèque de Photon. Cette scène est lancée après avoir choisi la section multijoueur dans le menu principal. Une fois la scène de chargement terminée (et le joueur connecté au serveur) le joueur se retrouve dans un lobby ou il est invité à renseigner le nom du room a creer ou une à rejoindre. A noter qu'il n'est pas possible de rejoindre une room qui n'a pas déjà été créé. Lorsqu'un joueur renseigner le nom d'une room a creer cette dernier est creer est ajouter dans la liste des rooms disponibles. Dans le cas où le joueur essaye de renseigner le nom d'une room inexistante il ne se passe rien, nous pensons d'ores et déjà à implémenter un message d'erreur pour en informer le joueur. Ensuite nous avons créé un script pour le point de spawn des joueurs et créé les instances des personnages incarnés par nos personnages sur le serveur. Une fois que nos joueurs se retrouvent dans la scène de jeu, leurs positions et les animations de ces derniers sont synchronisées à l'aide des composants issues de l'asset de Photon.
Par ailleurs Photon étant a déjà beaucoup de fonctionnalités installées par défaut facilitant grandement le développement du multijoueur. Par exemple, nous pensions devoir nous charger de lister l'ensemble des rooms actuellement créés et disponibles pour éviter que les joueurs ne puissent créer deux rooms avec le même nom. Quelle ne fut pas notre surprise lorsque nous nous rendîmes compte que Photon gérait déjà dans ce cas.

2.6 Menu

La création des menus a été pensée et revue à plusieurs reprises au fur et à mesure de l'évolution du projet. En effet, le menu est la première scène que trouvera le joueur en découvrant le jeu. Il est donc indispensable qu'il soit clair et simple d'utilisation. Pour ce faire, nous utilisons la règle des trois clics. Cela veut dire que le joueur doit être capable de trouver tout ce qu'il veut en maximum trois clics, sans avoir toutes les informations possibles sur une seule page. Pour ce faire nous avons utilisé différentes pages pour trier les menus (par exemple, les pages play, setting et pause). Nous avons aussi utilisé des options d'Unity tels que les dropdown qui sont très pratiques lorsqu'on désire proposer plusieurs choix qui se ressemblent (tels que le choix du

personnage). Le détail des menus est pour l'instant très simple. Il existe trois grandes fenêtres de menu :

1. Menu Principal :
   — Play Permet de choisir le mode de jeu (campagne ou multijoueur)
   — Setting Permet d'atteindre les réglages du jeu
   — Exit Permet de quitter le jeu.(On n'est pas des monstres, on laisse les gens partir)
   — Choix personnage : (En cours de développement) Permettra à terme de choisir son personnage avant de commencer une partie.
2. Menu Pause :
   — Resume & Restart : Permettent respectivement de continuer et recommencer la partie
   — Setting : Permet d'atteindre les réglages du jeu
   — Exit : Permet de quitter le jeu.
3. Menu Setting :
   — Volume : Permet de gérer le volume sonore du jeu (En cours de développement, seul l'aspect graphique de cette option est disponible) Permettra à terme de mettre par défaut le jeu en plein écran ou de sélectionner la taille de la fenêtre.

Voici un petit aperçu des différents menus que vous allez rencontrer dans notre jeu.

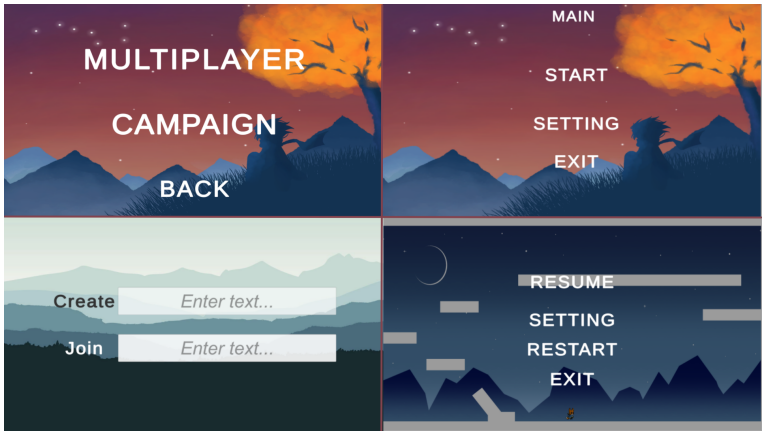


FIGURE 7 – Menus de jeu

2.7 Gestion des versions via Git

Comme précisé dans le cahier des charges, nous utilisons le logiciel Git afin de gérer les différentes versions de notre projet. Nous avons donc commencé par initialiser un répertoire Git [4]. Bien que cela ne fasse que quelques semaines que nous avons commencé notre projet, notre compréhension et prise en main de cet outil a beaucoup évolué. Il y a quelques semaines, aucun d'entre n'avait utilisé git dans le cadre d'un aussi gros projet, notre utilisation de cet outil se limitait à des repos personnels que nous utilisions pour des petits projets ou pour travailler et rendre nos TPs de programmation. Le travail à plusieurs sur un même projet a donc été une

4. accessible ici : `www.github.com/Gregoire-perso/arktos`

nouvelle étape pour nous dans notre apprentissage en tant qu'étudiants en informatique. Afin d'organiser notre code et de le fusionner plus facilement, nous utilisons le système de branches qui est encore nouveau pour nous, bien que nous ayons appris à le maîtriser au cours de ces dernières semaines.

En ce qui concerne notre organisation, nous essayons de séparer au maximum nos tâches afin de pouvoir travailler en parallèle. Pour cela, nous utilisons différentes branches (la plupart du temps, nous avons une branche par membre du groupe, mais il est possible qu'elles soient plus nombreuses dans certains cas). Lorsque l'un d'entre nous a terminé la fonctionnalité qu'il était en train de développer, nous regroupons sa branche avec la branche principale, afin que cette fonctionnalité soit disponible sur la branche correspondant à une version toujours fonctionnelle de notre projet. Bien entendu, lors de ces fusions, nous avons dû faire face à plusieurs problèmes, mais nous aborderons ce point là dans une autre partie de ce rapport.

Pour finir, en ce qui concerne le type d'utilisation de git, nous avons tenu à limiter au maximum l'utilisation de git via une interface graphique. Ce choix d'utiliser le moins possible une interface graphique, au profit d'une utilisation en lignes de commande, nous a permis d'apprendre de nombreuses choses sur le fonctionnement de git. Nous avons fait ce choix car, bien que plus complexe à prendre en main et à utiliser, il nous permet d'apprendre plus vite, et ces compétences nous seront aussi beaucoup plus utiles dans notre future vie d'epitéen, puisque l'utilisation de git en lignes de commande est un passage obligatoire dans notre scolarité. Alors, autant maîtriser cet outil au plus vite !

Cela a par ailleurs été l'occasion, comme vous pouvez le voir sur la figure suivante, d'explorer les options de git log afin de pouvoir travailler sur une représentation graphique de notre répertoire claire et concise regroupant toutes les informations importantes dont nous pouvons avoir besoin.

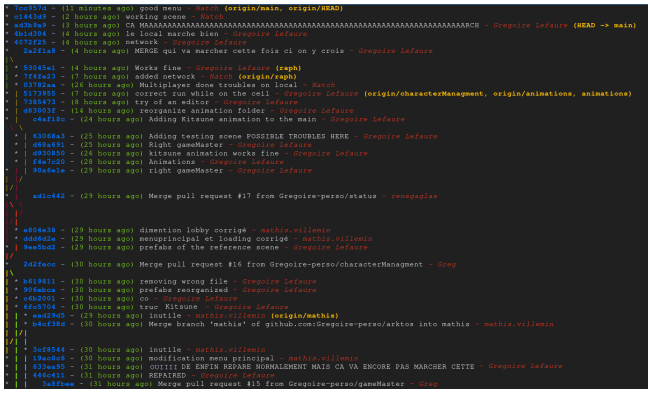


FIGURE 8 – Représentation des derniers commits

2.8 Compatibilité

Pour finir cette partie, nous avons réussi à créer un jeu cross-platform, disponible sur Windows, Linux et MacOS. Faire un jeu accessible à tous était pour nous un point très important, et Unity nous a permis de fournir un jeu cross-platform de manière simple.

3 Problèmes & Solutions envisagées

3.1 Git

Comme rapidement évoqué dans la section Gestion des versions via Git, notre faible expérience avec cet outil nous a valu quelques surprises lors de la fusion de nos branches respectives. A de nombreuses reprises, nous avons eu des merge conflicts que nous avons dû régler. Certains étaient simples, et donc facilement résolus, mais d'autres étaient beaucoup plus complexes, en particulier la fusion de scènes ou de prefabs unity, qui sont des fichiers en YAML [5]. Ces derniers fichiers étant très gros et particulièrement peu lisibles pour un humain, résoudre les conflits à la main aurait été un enfer. C'est ce qui nous a fait découvrir, après de nombreuses recherches, un outil développé par Unity permettant de fusionner proprement ce type de fichier : UnityYAMLMerge.

Outre l'utilisation d'outils externes afin de nous faciliter la tâche lors de la réunion de nos différentes versions de notre projet, nous avons aussi au cours du temps eu de moins en moins affaire à ce type de conflits pénibles à régler. En effet, grâce à une organisation plus claire et intuitive de l'architecture de notre projet, ainsi qu'à une meilleure répartition des tâches de notre part, nous avons réussi à diminuer ce type de conflits. Malheureusement, nous ne sommes pas des « perfect programmer [6] », et nous savons que nous allons continuer à faire face à ce type de problèmes. Mais une meilleure maîtrise d'Unity et de ses subtilités, ainsi que de meilleures pratiques de code nous permettront de diminuer ce type de problèmes.

5. Le YAML est un format de représentation de données
6. programmeurs parfaits

3.2 Multijoueur

Bien que sa réalisation semblait se dérouler sans encombre, le multijoueur à dévoilé certains bugs au cours de la phase de test. Tout d'abord lors de la scène de connexion, si le joueur a une perte de connexion la requête de connexion au serveur échoue et n'est pas relancer, donc le joueur se retrouvera bloqué dans la scène de chargement. Nous n'avons pas encore fixé ce problème mais nous avons pensé à rajouter un timeout pour résoudre ce dernier. Un autre problème est survenu lorsque le joueur ayant créé la room quittait cette dernière, cependant ce bug ne survenait que lorsque que nous lancions notre jeu dans le mode de développement qui désactivait certaines fonctionnalités automatiques de Photon.

De plus d'un point de vue sécurité nous n'avons pas encore créé de statut privé ou publique pour nos rooms, cela étant n'importe qui peut rejoindre une room sans connaître son créateur, ce qui peut laisser place à une très mauvaise expérience de jeu et une porte d'entrée à tous tentative d'attaques informatiques ... Par ailleurs lors de l'implémentation de multijoueur un conflit c'est révélé entre la création des objets de nos personnages via le serveur et la création des objets dans la scène en locale. Nous avons finalement trouvé de solution pour faire vivre les deux modes de jeux en même temps, cela en créant deux scènes similaires mais avec un mode de spawn du joueur différent. En effet pour la partie serveur le joueur est instancié sur le serveur la oú dans le mode campagne ce dernier est créé en local.

En outre, le problème majeur qui se pose actuellement dans la partie multijoueur est une importante latence dont nous ne parvenons pas encore à trouver l'origine. Nous avons des pistes de recherche comme forcer la redirection de la connexion vers un serveur plus proche de notre position mais rien ne nous assure que cela résolve notre problème. Une chose est sur nous faisons tout notre possible pour résoudre ce projet qui est une véritable entrave au bon déroulement du jeu.

Pour finir nous ne sommes pas encore parvenu à synchroniser le changement de gravité pour nos deux personnages. En effet nous avons dû désynchroniser les déplacements de nos personnages pour le multijoueur, mais cela a également entraîné une désynchronisation du changement de gravité qui n'était pas voulu. Nous savons quelle est la source du problème mais nous n'avons pas encore eu le temps de traiter ce dernier.

3.3 Différences de point de vue

Au cours de ce projet, nous avons eu aussi affaire à des problèmes de conception. En effet, même si nous étions tous d'accord sur l'idée principale du projet et la direction que nous voulions lui donner, la concrétisation de nos idées étaient parfois différentes selon les points de vue. Malgré tout, la bonne entente au sein de Skylab et le fait que chacun soit à l'écoute des idées des autres nous a permis d'en discuter à plusieurs reprises lors de réunions afin de trouver une concrétisation de nos idées qui convenait à tous. De plus, nous pouvons communiquer et poser des questions facilement aux autres membres de notre groupe lorsque nous en avons le besoin grâce à un serveur discord qui nous permet de rester en lien même lorsqu'il nous est

impossible de nous voir en présentiel.

3.4 Difficultés graphiques

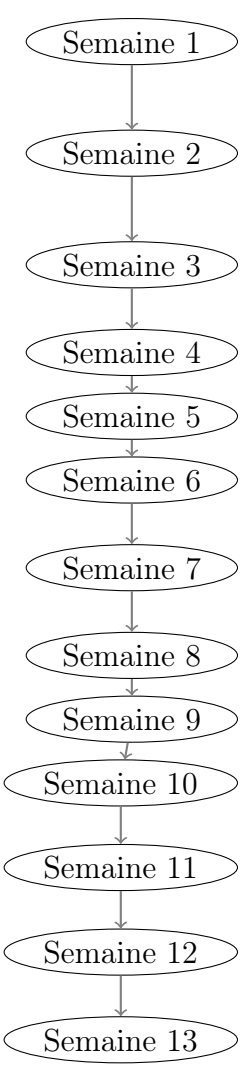
Les personnages du jeu sont en pixel-art. Cependant ce n'était pas le modèle initialement envisagé, mais le temps nécessaire pour dessiner chaque frame sans pixel-art prend beaucoup de temps. Nous avons donc testé la fonction intégrée à Unity pour l'animation de personnages 2D, mais celle-ci, bien que tout à fait fonctionnelle, donnait un rendu qui laissait à désirer, avec des déformations difficilement évitables lors des mouvements, avec un résultat peu convaincant. Nous avons donc décidé de nous pencher vers le pixel-art, quitte à dessiner chaque frame à la main, afin d'avoir un rendu agréable.

FIGURE 9 – Evolution du personnage Drow

4 Nos objectifs

4.1 Planning

Nous avons réalisé un planning commençant après la première soutenance allant jusqu'à la dernière soutenance afin d'avoir une vision claire de ce que nous avions à réaliser (tout en tenant compte de nos différentes obligations, telles que les partiels...) :



Développement des attaques des personnages - Création d'un choix de personnages pour le multijoueur

Création des animations de combats et test de ces derniers dans le multijoueurs

Mise en relation de l'ensemble de nos scripts et création des énigmes

Fin des fonctionnalitées majeures du multijoueur

Fin du design des personnages et des ambiances sonores

Début du développement de l'intelligence artificielle

Intégration des premiers résultats de l'intelligence artificielle dans le déplacement des monstres

Fin de l'intégration des personnages

Fin des niveaux de la campagne

Fin des dernières fonctionnalitées principales

Fin de l'intelligence artificielle et application de cette dernières aux monstres en scène de combats

Mise à jour du site web

Fin des features non achevés et temps pour la résolution des bugs de fin

4.2 Nous faciliter la vie

Nous avons découvert quelques jours avant la première soutenance la possibilité de créer nos propres éditeurs. En effet, Unity nous permet d'en créer, allant du plus basique au plus complexe. Nous avons donc comme projet de créer un éditeur nous permettant d'automatiser la création des différentes animations à partir de leurs différentes frames. Ainsi, nous pourrons rapidement créer de nouvelles animations, aussi bien pour des armes, des monstres, des personnages, ou tout autre type d'objet nécessitant un mouvement. Cela nous permettra aussi d'assurer une harmonisation des différentes animations.

4.3 Trouver des beta testeurs

Afin de développer un jeu le plus complet possible, nos simples tests ne suffiront pas. C'est pourquoi, afin qu'il y ait le moins de bug possibles, lorsque notre jeu sera suffisamment avancé pour être réellement jouable, nous engagerons des beta testeurs [7] dans le but de nous faire remonter les différents bugs présents.

5 Conclusion

Pour conclure, ce projet suscite toujours autant d'intérêt au sein du groupe. Bien que l'on soit dans les temps par rapport au cahier des charges. Le développement du jeu est très loin d'être fini. Cependant nous pensons avoir réalisé l'étape la plus importante et complexe qui est d'apprendre à utiliser, en commun, tous ces nouveaux outils. Maintenant que nous avons fait les premières étapes qui consistent à créer du code et des scènes à partir de rien. Il nous sera bien plus facile de continuer et d'améliorer notre jeu en ajoutant encore plus de fonctionnalités. Nous ne serons plus bloqués par le manque de connaissance du projet et pourrons avancer le développement du jeu à un rythme plus élevé.

Ce projet nous a beaucoup appris sur le plan technique comme sur le plan humain. Nous sommes fiers de vous présenter l'avancement de notre projet. Comme vous avez pu le comprendre, notre équipe à déjà un plan d'action jusqu'à la prochaine soutenance. Nous espérons que ce rapport vous aidera à comprendre la direction qu'a pris notre équipe dans le but de finir ce projet.

L'équipe Skylab

7. aussi appelés « amis »et « famille »

6 Source

Nous remercions toutes ces personnes qui nous ont aidé de manière directe ou indirecte à la réalisation de ce projet.

— TUTO UNITY FR : `https://www.youtube.com/channel/UCJRwb5W4ZzG43J5_dViL6Fw`
Pour la gestion du son.

— Blackthornprod : `https://www.youtube.com/channel/UC9Z1XWw1kmnvOOFsj6Bzy2g`
Pour la mise en place du multijoueur.

— Brackeys : `https://www.youtube.com/c/Brackeys`
Pour la création des menus.

— opengameart : `https://opengameart.org/`
Pour la music du multijoueur.

7 Documentation

— Documentation precise et detaillee d'Unity : `https://docs.unity3d.com/ScriptReference/`

— overleaf : `https://www.overleaf.com/learn`

— Photon : `https://doc-api.photonengine.com/en/pun/v1/general.html`

— GitHub : `https://docs.github.com/en`