Abyssal Ascension

Rapport de soutenance 1

BOUDJEMA Elyas - MONSELLIER Jonathan - VASSAUX Clément - XIA Frédéric

Table des matières

1. Introduction

1.1. Présentation du projet

Abyssal Ascension est un de plateforme 2D, type Metroidvania, inspiré de grands jeux tels que Hollow Knight et Ori. Une ambiance sombre, puis s'illuminant de plus en plus afin d'atteindre les sommets les plus hauts. Le but est d'explorer afin de trouver les quelques items permettant d'avancer dans le jeu. En effet, contrairement aux autres, il ne suffira pour jouer que d'items simples et peu nombreux, car la réflexion est la clé.

1.2. Présentation de Caféine

Nous sommes Caféine, un groupe qui tire son nom de notre motivation pour ce projet et de notre addiction à la caféine (pas le café par contre, c'est pas bon). Une équipe qui mélange des personnes qui ont déjà des jeux, d'autres qui ont de bonnes bases en Python et d'autres qui ont la logique nécessaire au développement du projet. Ce mélange nous permettra d'apprendre les uns des autres dans la conception de notre jeu.

2. Conception

2.1. World Design

Sans monde, il n'y a pas d'Abyssal Ascension. Il faut donc travailler sur le world building.
Pour cela, j'utilise le logiciel LDtk (Level Design ToolKit). Ce logiciel permet la conception de monde de manière simple et efficace.

Pour le début de la conception du monde, on a choisi de prendre un design (tileset) simple composé de sol en pierre et d'un fond violet (cf. Figure 2 et 3) qu'on a projeté sur un layer avec des règles pour pouvoir faciliter la conception des niveaux. (cf. figure 7)
De plus, chaque zone dans le monde a une dimension de 960x480px et chaque cases (tiles) dans la zone a une taille de 24px par 24px (cf. Figure 4). L'agencement des zones entre eux est également rendu très simple grâce à l'interface de la gestion du monde (cf. Figure 5 et 8), il suffit juste de prendre et glisser la zone à l'endroit qu'on souhaite.
Lors de la création des zones, on choisit donc un layer (couche, comme stone ou dirt) et on dessine le terrain que l'on souhaite obtenir) (cf. figure 9).

Ce logiciel permet d'exporter la map en png et JSON. Ce qui permet d'implémenter la map dans le code facilement.

2.2. Multijoueur

Un des piliers d'Abyssal Ascension est la coopération entre les joueurs. Nous avons implémenté une architecture client-serveur en LAN (Local Area Network) permettant une synchronisation fluide et cohérente entre deux instances du jeu tournant sur le même réseau local.

Pour synchroniser les données en python, j'utilise la bibliothèque native python `socket`. Le système tourne autour de 2 entités :

- Le serveur (host) : héberge la logique du jeu et distribue aux clients l'état du monde (positions de l'ensemble des joueurs, ennemis, statut de la partie…). Le serveur à toujours l'identifiant 0.
- Les clients : transmettent la position de leur joueur au serveur après l'application des inputs (entrées claviers / souris) permettant le déplacement du joueur. Chaque client à un identifiant unique, il commence avec un identifiant temporaire de -1 (indiquant qu'il n'est pas encore connecté au serveur), puis une fois connecté, le serveur lui

attribue un identifiant libre (par exemple 1 si ce client est le premier à se connecter sur le serveur).

(cf. figure 14)

L'échange des informations s'effectue via le format JSON. Ce format permet de structurer des données complexes (coordonnées, états, liste d'ennemis…) sous forme de chaînes de caractères lisibles, facilitant ainsi le débogage et la communication entre les instances.

Pour maintenir une expérience multijoueur fluide tout en limitant la surcharge du réseau, les données sont synchronisées toutes les 3 frames. Le jeu tournant à 60 fps, cela revient environ à une synchronisation toutes les 50ms.

2.3. Menus & Interface utilisateurs

Le menu principal a pour image de fond, une image sombre représentant l'ambiance du début du jeu (les abysses).
Le thème principal est également joué en arrière plan pour rendre le menu principal plus complet et vivant.

Pour faciliter la connexion avec des amis, on a décidé d'afficher l'adresse IP locale de la machine dans le menu principal. Cela permet de la partager et de pouvoir héberger puis rejoindre une partie multijoueur plus rapidement.

Trois modes de jeu sont proposés :

- Solo : Lancement immédiat d'une partie locale.
- Héberger (multijoueur) : Création d'un serveur et lancement immédiat de la partie hébergée.
- Rejoindre (multijoueur) : Redirige vers une interface de saisie.

Dans le menu "Rejoindre", le joueur saisit l'IP du serveur à rejoindre. Un indicateur visuel valide la saisie en temps réel selon le format standard (présence des points et chiffres compris entre 0 et 255).

Lorsque le joueur se retrouve en jeu (en mode solo ou multijoueur), il peut voir de nombreuses informations de débogage en haut à gauche de son écran. Ces informations correspondent au mode de jeu actuel, au nombre de joueurs connectés sur le serveur, à l'identifiant du joueur, au nombre de FPS, à la position et à la vélocité du joueur local, ainsi qu'à sa vie et à son statut (au sol / en l'air).

2.4. Textures & Animations

Pour tout ce qui touche à l'animation, nous avons choisi le style "pixel art". De plus, chaque animation et graphique a été dessiné à la main. Le site internet que nos designers utilisent actuellement est "Piskel". L'interface est rapide à prendre en main et très pratique.

Pour ce faire, il faut suivre plusieurs étapes. Sur le logiciel de création utilisé, on doit dans un premier temps dessiner ce que l'on appelle des "sprites" (cf. figure 10). Ils correspondent à une image unique. Via le logiciel, on va ensuite disposer plusieurs sprites à la suite et les faire défiler pour créer du mouvement (cf.figure 11). Cela permet alors d'importer chaque animation dans le jeu vidéo et de les lier à chaque action. Les graphismes présents sur l'écran d'accueil, les futures cinématiques ou encore le site internet ont également été créés via ces techniques.

Pour l'implémentation dans le jeu, il n'y aura rien de très compliqué. Lors de l'écriture du code, nous avons des variables globales qui définissent le statut du joueur (course / marche / chute…). Lorsque l'on veut déclencher une animation spécifique, il suffira de relier les sprites correspondants au bon moment en fonction de la du statut du joueur. Tant que la condition est remplie, on fait défiler les sprites correspondants. Le personnage avance donc dans le jeu et l'animation tourne en boucle jusqu'à ce que ce dernier s'arrête.

Pour la suite, on ajoute les sprites directement dans un fichier du jeu prévu pour l'animation afin de pouvoir les afficher selon certaines conditions. Les images sont enregistrées en fichier .PNG afin de faciliter leurs utilisations. Chaque image est renommée d'une façon précise : 0-Running-0 signifie dans l'ordre qu'il s'agit du personnage N°1; de l'animation de course (running); du sprite N°0. Cela permet de s'y retrouver très facilement lorsque nous sommes amenés à coder pour les animations. Il faut aller chercher la bonne image au bon endroit. (cf. figure 12)

Pour ce qui est du multijoueur, chaque personne connectée aura le même design de personnages, mais avec une couleur de cape différentes. D'autres petites modifications apparaissent également (cf. figure 13).

2.5. Son & Musique

Nous avions décidé de créer notre propre musique pour le jeu.
A l'aide de l'expérience que j'ai accumulée après plusieurs années de piano et de composition musicale, j'ai utilisé le logiciel gratuit MuseScore pour composer un thème principal pour le jeu.

L'objectif des musiques ambiantes du jeu est de créer une ambiance lourde et pesante. Pour ce faire j'ai utilisé des tonalités mineures (La mineur, Mi mineur…).

Pour le thème principal, j'ai utilisé le sol dièse mineur (G# minor) qui permet d'évoquer des choses lourdes et lourdes émotionnellement et le son d'un piano felt (cf. Figure 1 pour voir la partition).

Pour les effets sonores, nous les avons enregistrés nous même, en utilisant principalement notre voix pour simuler chaque son (notamment pour le saut et le dash). On a également enregistré nos propres bruits de pas pour les effets sonores de course et de marche.

2.6. Site Web

Afin d'accompagner le projet, nous avons conçu un site web composé de plusieurs rubriques et organisé par un style css simple et coloré. Nous avons choisis des couleurs plutôt sombres comme un bleu très foncé #0a0e27 ou encore un bleu marin #1e3a8a qui représentent bien les couleurs et l'ambiance de notre jeu.

Pour chaque rubrique nous avons créé des classes pour pouvoir manipuler leur style (dimensions/couleurs) séparément depuis le fichier css afin d'obtenir un fichier structuré.

Le site est hébergé par github pages depuis le repository principal du jeu dans le dossier 'docs' situé à la racine du projet.
A chaque nouveau commit, le site se met à jour automatiquement par les actions github (ensemble d'actions automatisées par github).

2.7. Contrôles

J'ai implémenté les contrôles de base pour le joueur.
Le joueur peut utiliser les touches Q / D ou les flèches gauche / droite pour se déplacer horizontalement.
Pour sauter, le joueur peut utiliser les touches espace ou flèche vers le haut.
Pour effectuer un dash, le joueur peut utiliser les touches shift gauche ou shift droite.

3. Réalisation

3.1. Fonctionnalités jouables actuelles

Pour cette première soutenance, le joueur peut lancer le jeu et choisir plusieurs options dans le menu d'accueil avec le thème musical principal du jeu. Il peut choisir entre créer une partie en solo, héberger une partie ou rejoindre une partie déjà hébergée ailleurs.

Les joueurs peuvent donc créer une partie avec un joueur serveur, et des joueurs clients. Dans la partie multijoueur, les joueurs peuvent se voir et se déplacer en utilisant des mouvements tels que, la course (gauche / droite), le saut, le double saut et le dash.

Chaque joueur possède également une variable représentant sa vie avec des fonctions permettants d'infliger des dégâts et de redonner de la vie (ces fonctions ne sont cependant pas encore implémentés dans les éléments du jeu, c'est à dire que les fonctions sont présentes et fonctionnelles mais ne sont jamais appelées pour le moment).

3.2. Problèmes & Solutions

Lors de la création des premiers designs du personnage principal, nous avons remarqué que le niveau de détails ne convenait pas à nos attentes pour le jeu. Nous avons donc pris la décision de passer d'animations de 32x32 pixels a 64x64. Ces animations étant dessinées à la main, nous avons alors choisi de garder un design simple et efficace afin de ne pas perdre trop de temps.

4. Synthèse des expériences individuelles

4.1. Elyas

J'ai dû réapprendre certaines compétences perdues au fil du temps comme le code html pour créer le site ce qui a fait perdre pas mal de temps dans la conception du site. J'ai également dû apprendre à utiliser github qui est très utile pour notre projet. Cela m'a permis de développer mes compétences basiques.
J'ai également décidé des touches optimales pour interagir avec le jeu afin de ne pas avoir de problèmes de confort en jeu (Q/D pour se déplacer, espace pour sauter, shift pour dash …)

4.2. Jonathan

J'ai pu grandement participer au début de la création du jeu. En effet, lors de mon année de terminale, j'ai créé un jeu vidéo dans le cadre d'un projet et j'ai donc une petite expérience sur le sujet. J'ai ainsi pu expliquer à mes camarades comment bien structurer leur code ou encore les étapes à suivre lors du développement.
Pour les aider davantage (hormis la création de décors et d'animations), j'ai appris à corriger les erreurs qui rendent notre code inopérant. Cela peut être qualifié de debugging. J'ai également appris le fonctionnement d'une map et des hitbox, et je compte approfondir la création d'animations.

4.3. Frédéric

En réalisant le début de ce projet jeu vidéo, et plus précisément sur la structure des niveaux, j'ai pu apprendre à utiliser des logiciels puissants pour la conception de monde comme par exemple LDtk. j'ai également pu approfondir la gestion des auto-layers pour automatiser des rendus de tuiles en fonction de certaines règles.
Ce début de projet m'a permis de penser comme un concepteur. Il ne suffit pas seulement de créer un beau décor, il fallait qu'il soit fonctionnel pour le joueur et techniquement compatible avec le code.

4.4. Clément

Après 2 mois à travailler sur le projet, j'ai appris énormément de choses sur la partie multijoueur des jeux vidéo en LAN (Local Area Network). En effet, j'ai dû apprendre les bases de la librairie python socket afin de transmettre des données entre deux machines du même réseau.
Ces 2 mois m'ont également fait travailler sur la théorie musicale, la composition de musique et la recherche de solutions pour obtenir des effets sonores cohérents fait par nous même.
Ce projet m'a également appris à chercher des solutions à tous les problèmes rencontrés lors du développement et à travailler en équipe afin de se mettre d'accord sur certaines fonctionnalités et idées que propose le groupe.

5. Prévision

5.1. Accord avec le planning

Tâches	Progression actuelle soutenance 1	Progression visée soutenance 1
World building	30%	30%
Character Design	40%	55%
Musique	40%	40%
IA	5%	30%
Contrôles joueurs	80%	80%
FX	40%	40%
Physique et collisions	80%	60%
Playtest	40%	20%
Interface / UI	50%	50%
Optimisation	40%	30%
Multijoueur	60%	25%
Système de sauvegarde	0%	30%
Site web	70%	15%

% = en retard
% = dans les temps
% = en avance

5.2. World Design

Pour le world design, on devra ajouter de nouvelles zones et également ajouter plus de détails aux zones actuelles.
Il va également falloir importer les designs dans le jeu et les implémenter correctement afin de remplacer le niveau de test qui est actuellement utilisé dans le jeu.

5.3. IA

Pour l'IA, il faudra ajouter des ennemis avec un système de patrouille simple (gauche, droite sur une plateforme).
Lorsque l'ennemi sera à une certaine distance du joueur, il le pourchassera et attaquera automatiquement.

5.4. Multijoueur

Pour le multijoueur, il va falloir ajouter d'autres informations à transmettre en plus de la position des joueurs comme leur arme actuelle et leur état (en saut / en dash).
Il va aussi falloir implémenter un système de déconnexion (voulue, le joueur se déconnecte ou non voulue, le joueur perd la connexion).

5.5. Menus & Interface utilisateurs

Pour les menus et l'interface utilisateurs, il faudra retirer les informations de débogage une fois qu'elles seront inutiles.
Il faudra également créer un menu pause, une interface pour que le joueur ait un moyen d'accéder aux informations de son joueur.

5.6. Textures & Animations

L'objectif est de rendre chaque animation plus fluide en lui ajoutant 2 à 3 sprites. Des mouvements plus complexes étant prévus pour la suite du jeu, il faudra lier animations et hitbox pour rester cohérent. Il faudra aussi s'occuper des graphismes de fonds, (ex : menu pause, lors des dialogues et autres). Les graphismes des premiers ennemis sont également prévus; avec celui du premier boss si possible.

5.7. Son & Musique

Pour les sons et les musiques, il faudra ajouter plus de bruitages (effets sonores) pour de nouvelles actions comme l'attaque, la mort du joueur et les sons de l'interface (son joué sur un clic de bouton par exemple).
Il faudra également composer plusieurs autres musiques originales pour d'autres zones et contextes (dialogue, combat, animation…).

5.8. Site Web

Pour le site web, il faudra rajouter du contenu tel que les captures d'écrans du jeu, un lien d'installation et un guide d'installation.
Il faudra également le rendre plus élégant sur interfaces mobiles (en ce moment, l'interface est légèrement endommagé quand ouverte sur téléphone portable).

5.9. Contrôles

Pour les contrôles, il va falloir ajouter la possibilité de faire un “wall jump” (aller en direction du mur et utiliser la touche pour sauter), et la possibilité d’attaquer en ajoutant une nouvelle touche pour le combat. Pour la suite du jeu, il est prévu que les combats se basent sur différentes utilisations du même objet, ce qui inclut donc des combinaisons de touches ou actions à coder.

6. Conclusion

Notre travail depuis le début de ce projet s’est réalisé sans accroc et les objectifs fixés ont été respectés dans leur ensemble malgré la charge de travail.
Chacun continue de bien travailler au sein du groupe et l’aide est apportée dès que nécessaire si un des membres de l’équipe à des soucis durant son travail.
La charge de travail restante est tout de même très importante, mais l'avancée de la partie fonctionnelle et programmation va nous permettre de passer rapidement aux étapes suivantes afin de se rapprocher de l’objectif final du projet avant la date d’échéance.

7. Annexes

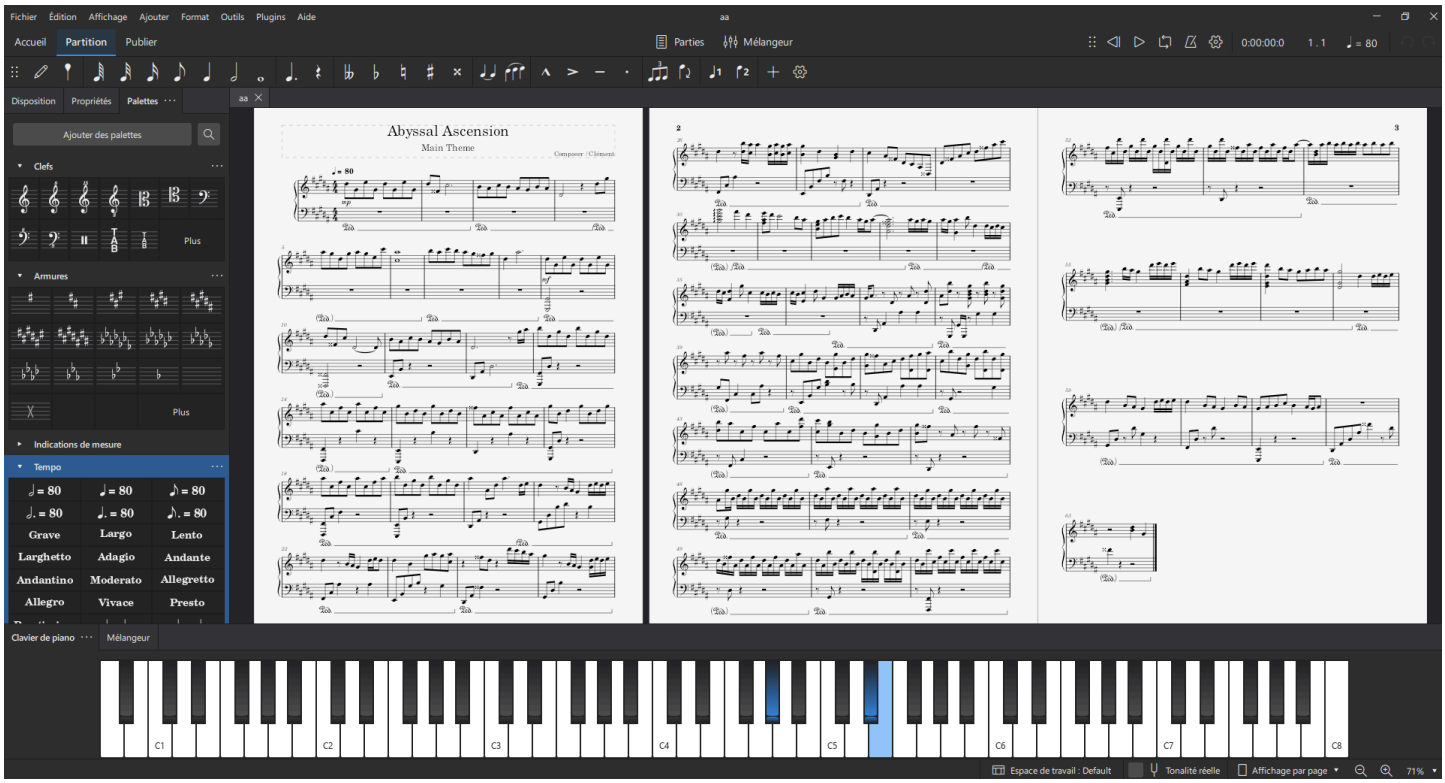

Figure 1 : Partition de musique du thème principal composée sur MuseScore.

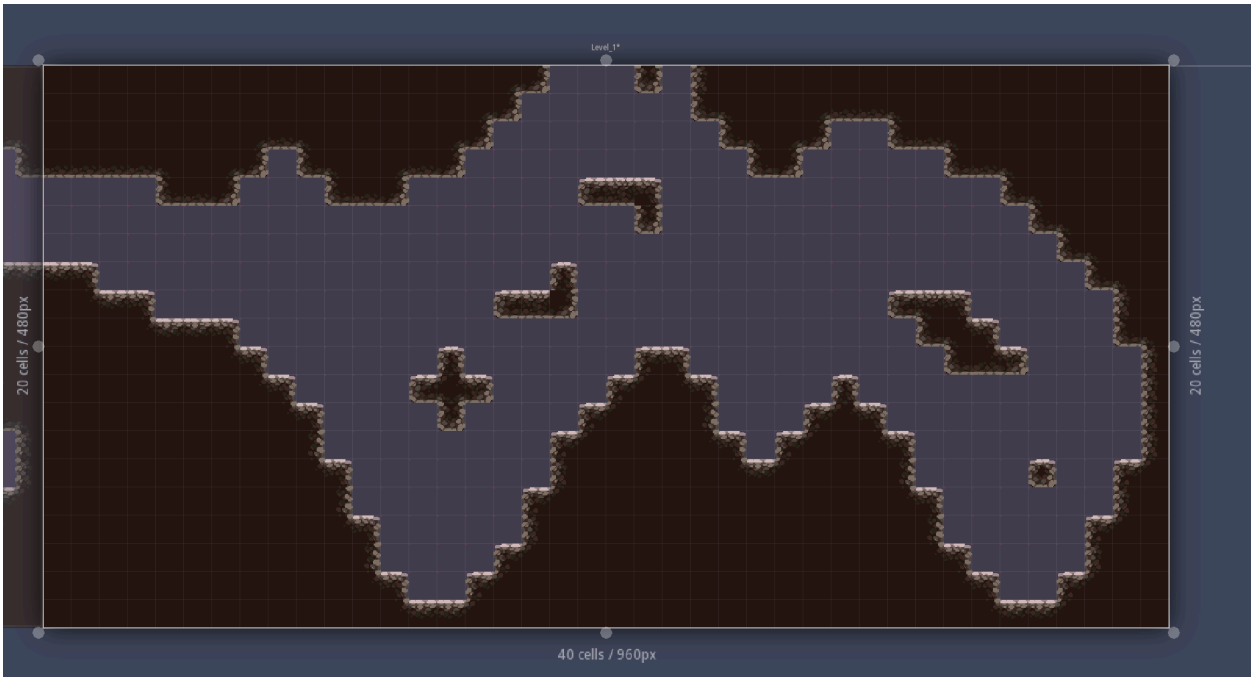

Figure 2 : Première zone jouable

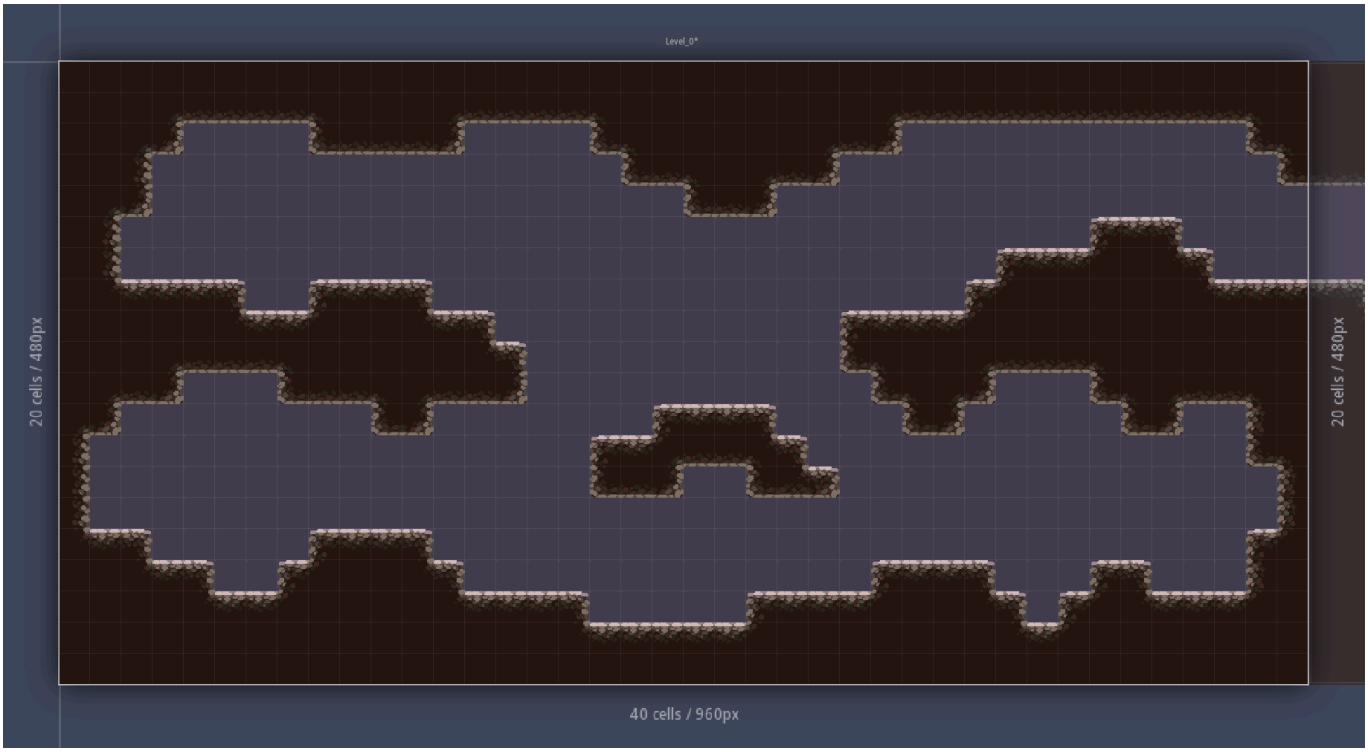

Figure 3 : 2e zone jouable

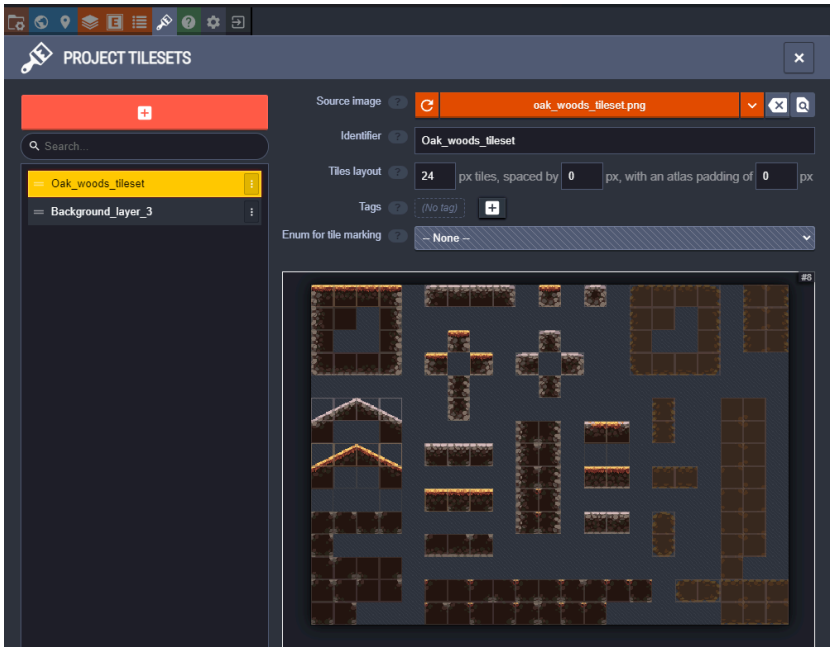

Figure 4 : Interface de configuration des tilesets sous LDtk

Figure 5 : Map globale du jeu sur LDtk.

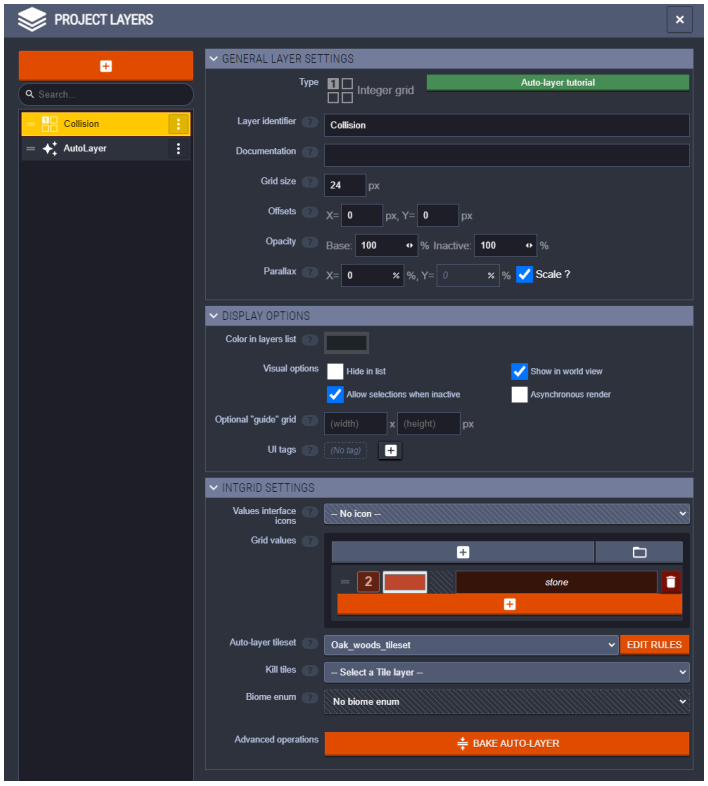

Figure 6 : Interface de gestion du World Building et des collisions

Figure 7 : Interface de configuration des motifs pour le rendu automatique du décor

Figure 8 : Configuration des paramètres globaux du monde et du niveau dans LDtk

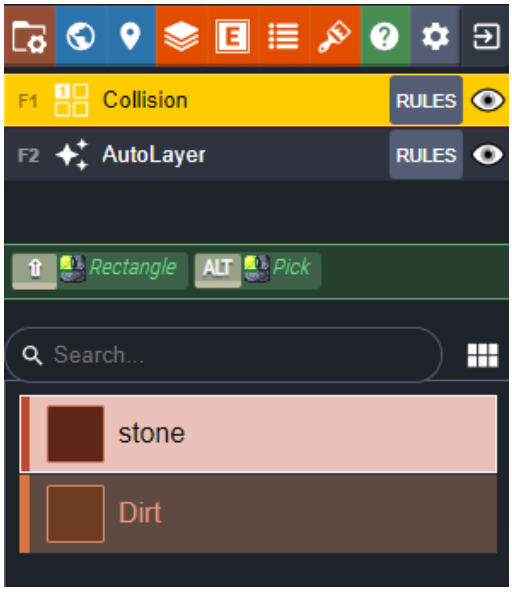

Figure 9 : Interface de sélection des matériaux pour le world building.

Figure 10 : Création d'un sprite pour le personnage principal.

Figure 11 : Création d'une animation de course en enchaînant rapidement plusieurs sprites afin de créer du mouvement.

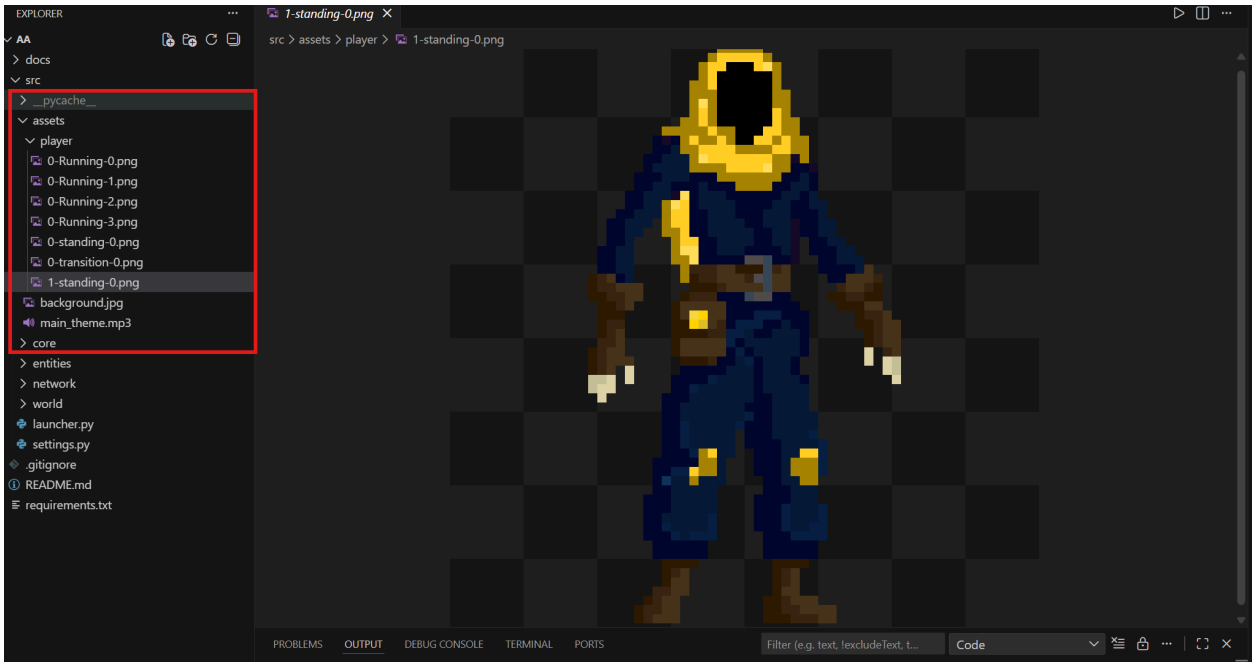

Figure 12 : Vue de l'organisation des fichiers pour les animations dans l'éditeur.

Figure 13 : Player 1 Player 2

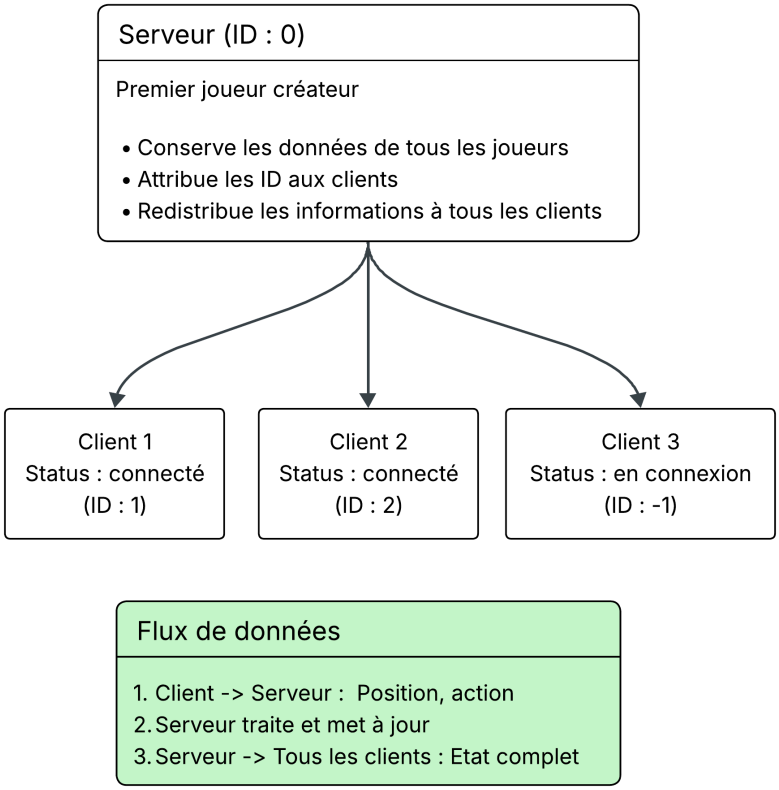

Figure 14 : Schéma représentatif du système multijoueur client-serveur