A l'attention de
Monsieur **Dominique YOLIN**

Rapport de projet

MineCube

C++, 2ème semestre ING2, 2022

Blaise Arthur, Juif Alexandre, Julien Théo, Richard Julien, Teste Lucas

Diagramme des classes **3**

Fonctionnalités **4**

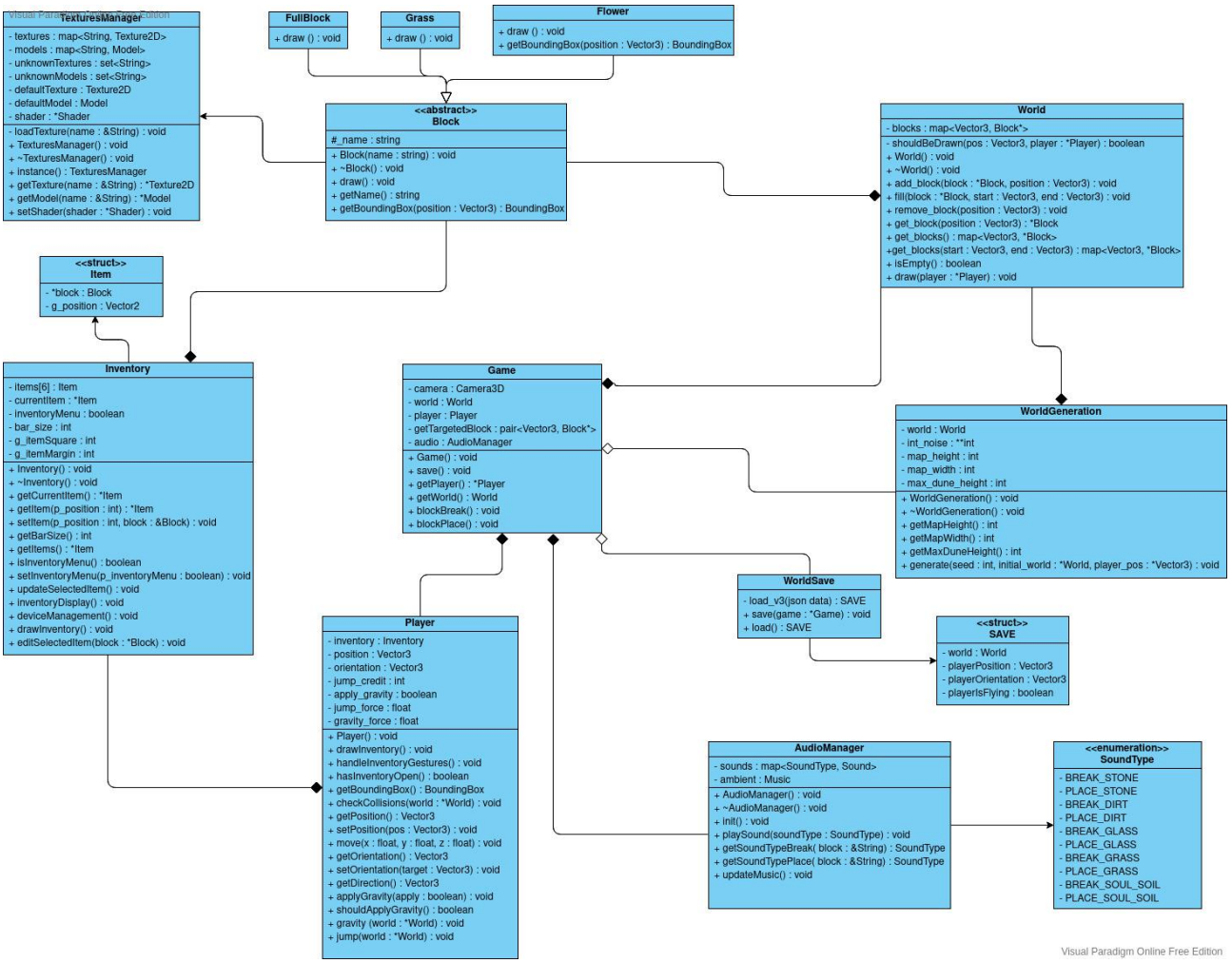
Gestion de projet **5**

Outils utilisés 5

Répartition des tâches 5

Gestion de projet 6

1. Diagramme des classes



2. Fonctionnalités

Lorsque le jeu est lancé pour la première fois, un monde est généré et le joueur apparaît au milieu avec une barre en bas au centre de l'écran indiquant les blocs disponibles dans l'inventaire du joueur.
Une fois dans le monde, le joueur peut faire les actions suivantes :

- Sauter
- Avancer, reculer
- Bouger la caméra à l'aide de la souris
- Choisir un bloc parmi ceux disponibles dans son inventaire
- Changer la place des blocs dans l'inventaire
- Casser un bloc
- Placer un bloc
- Passer en mode vol (non soumis à la gravité)
- Modifier son inventaire
- Son de jeu (et lorsqu'un bloc est cassé/placé)
- Générer un monde (différentes couches de blocs)
- Journée (nuit et jour avec mouvement dynamique du soleil)
- Brouillard sur les blocs trop loin pour la vision du joueur
- Sauvegarder la partie en cours
- Optimisation de performances pour obtenir des FPS acceptables pour jouer
- Apparition des blocs en fonction de la position du joueur

Pour la gestion du monde nous avons utilisé deux librairies, la première étant JSON nous permettant de sauvegarder les propriétés du monde dans le fichier "worldsave.json" lorsque le joueur quitte le jeu. Ainsi, le joueur pourra reprendre sa partie là où il l'a arrêtée.
La deuxième librairie nommée PerlinNoise permet de générer de manière pseudo-aléatoire le relief du monde généré quand il n'y a pas de sauvegarde du monde existante.

3. Gestion de projet

1. Outils utilisés

- **GitHub** : Comme tout projet nécessitant un travail de groupe, nous avons utilisé GitHub afin de centraliser notre travail. Nous avons utilisé principalement GitHub pour gérer les branches de développement et la branche principale mais nous avons aussi utilisé la partie “questions (issue)” de GitHub afin de savoir en permanence les fonctionnalités développées par les membres du groupe.

- **Google Docs** : Pour la rédaction de ce rapport nous avons l’outil de collaboration proposé par Google. Cet outil nous a permis de travailler simultanément pour la rédaction du rapport.

- **Discord** : L’outil principal de notre projet a été Discord. Nous avons créé un serveur avec les membres du groupe que que nous avons décomposé en 3 parties :
    - Informations : Cette partie correspondait à un canal avec uniquement les informations essentielles du projet. Les informations essentielles étaient les points attendus par le professeur pour ce projet. L’évolution de la “To-do list” qui nous a permis de s’assigner une nouvelle tâche lorsque nous finissions la précédente.
    - Discussion : Principalement utilisé pour discuter et programmer des séances de travail communes. Nous mettions dans cette discussion les problèmes rencontrés pour que le créateur de la fonctionnalité puisse résoudre le problème. Nous mettions aussi nos résultats et nos appels à l’aide lorsqu’une personne bloquait sur une fonctionnalité.
    - Git : Ce canal a été utilisé uniquement par le bot GitHub qui envoyait un message pour chaque mise à jour du code sur GitHub (nouvelle branche, commit, pull request, etc.). Ce canal nous a permis de connaître l’avancement des autres et de lire leur code une fois qu’il a été fusionné sur la branche principale (afin de connaître entièrement le projet).

2. Répartition des tâches

Voici le détail des fonctionnalités développées par les membres du groupe :

- Visuel et type de bloc : Lucas/Arthur
- Collisions : Théo/Arthur
- Déplacement du joueur : Arthur
- Casser/placer des blocs : Alexandre
- Sauvegarde du monde : Arthur
- Inventaire et changement de place des objets de l’inventaire : Théo
- Menu d’inventaire (choix des blocs placés dans la barre d’inventaire) : Julien

- Gestion des blocs affichés en fonction de la position du joueur (optimisation) : Arthur
- Mipmapping : Arthur
- Sauts et gravité : Théo
- Mode vol (non soumis à la gravité) : Arthur
- Sauvegarde de la position du regard et du mode vol : Arthur
- Brouillard : Arthur
- Ciel (Soleil dynamique et nuages) : Théo
- Génération du monde (différentes couches et relief du monde) : Théo
- Cycle jour/nuit : Arthur
- Nom du bloc sélectionné et informations de déplacement : Arthur
- Musique et bruitages : Arthur
- Diagramme de classe : Théo

3. Gestion de projet

Pour mener à bien ce projet nous avons décidé au préalable d'une organisation liée à la méthode Agile.

Dans un premier temps nous avons décidé d'une "to-do list". Ce détail des fonctionnalités et du travail à faire nous a permis de toujours avoir quelque chose à faire et donc de ne jamais stagner sur le développement du projet. Cette liste a évolué en fonction de nos idées, et de la division des trop grosses fonctionnalités en plusieurs petites fonctionnalités. La première répartition des tâches étaient liées au objectifs principaux du projet c'est-à-dire au remplissages des conditions pour une notation maximale (double héritage, polymorphisme, etc.).

Dans un second temps nous avons programmé des séances de travail régulières afin de développer le projet tous ensemble et de faciliter les échanges au sein du groupe. Une fois les tâches obligatoires effectuées pour un membre du groupe, il y avait deux situations possibles :

- Un autre membre du groupe nécessitait de l'aide sur le développement de sa fonctionnalité et donc la suite du travail était de l'aider au développement de son code.
- Les autres membres développaient leurs fonctionnalités sans avoir particulièrement besoin d'aide. Dans ce cas, la suite du travail était de s'attribuer (via les "issues" GitHub) la fonctionnalité qui lui semblait la plus intéressante à développer et de la développer.

Lors du développement d'une fonctionnalité, nous avons décidé de partir sur des commentaires Doxygen pour commenter nos fonctions car cela permet d'avoir une description rapide et efficace de la fonction lorsque c'est quelqu'un d'autre qui l'a développé.

Dans un dernier temps lorsque l'échéance du projet était assez proche nous avons décidé de peaufiner notre travail en testant différentes circonstances et en ajoutant des fonctionnalités très rapides mais qui rendent l'expérience du jeu beaucoup plus immersive (tel que la musique et les bruitages). Nous avons simultanément fait ce rapport et le diagramme de classe qui peut être amené à évoluer car ce projet nous a passionné et nous avons envie de développer d'autres fonctionnalités sans se limiter à une date limite.