Une image contenant texte, horloge, clipart

Description générée automatiquement

HADID Hocine et CHEMIN Pierre | 01/01/2022

Dirigé par **RABAT Cyril et DELISLE Pierre**

INFO 601/602 – Jeu multijoueur en réseau

Table des Matières

[1. Introduction 2](#_Toc128474534)

[2. Editeur de monde 2](#_Toc128474535)

[A. La représentation d’un monde 2](#_Toc128474536)

[3. L’aspect serveur 3](#_Toc128474537)

[4. L’aspect client 3](#_Toc128474538)

[5. Organisation du travail 3](#_Toc128474539)

1. Introduction

Nous avons entrepris la conception d’un projet ambitieux appelé Starlyze, qui consiste à développer un jeu de plateforme multijoueur en ligne. L’objectif est de créer un monde virtuel dans lequel chaque joueur peut contrôler un personnage et interagir avec d’autres joueurs de la même partie. Le jeu est conçu pour être immersif, passionnant et stimulant pour les joueurs.

Le concept de base du jeu est simple : chaque joueur peut créer son propre monde et y jouer avec d’autres joueurs. Cela signifie qu’il n’y aura pas de monde prédéfinies, et chaque joueur aura la liberté de créer son propre monde avec ses propres niveaux, décors, obstacles et ennemis. Ainsi, chaque partie de Starlyze sera unique et offrira une expérience de jeu différente à chaque fois.

Le projet est divisé en trois parties principales : l’éditeur, le serveur et le client. L’éditeur permet aux joueur de créer leur propre monde et de le personnaliser selon leurs préférences. Les joueurs peuvent ajouter des objets, des niveaux et des décors à leur monde. Le serveur est la partie du jeu qui gère la mise en relation des joueurs et la gestion de la partie. Il est responsable de la création de parties, de la gestion des joueurs et de la communication avec les joueurs. Le serveur est conçu afin d’être performant et fiable, afin d’offrir une expérience de jeu fluide et sans interruption. Enfin, le client est l’application qui permet aux joueurs de visualiser et de jouer au jeu. Il est conçu pour être facile à utiliser et à comprendre, avec une interface utilisateur intuitive et ergonomique. Le client affiche la partie en temps réel et permet aux joueurs de contrôler leur personnage.

De plus, les applications « serveur » et « client » communiquent tout au long de la partie via différents protocoles afin de garantir la fluidité de l’expérience de jeu et une interaction en temps réel entre les joueurs.

En somme, Starlyze est un projet conséquent qui vise à offrir une expérience de jeu unique et immersive à tous les joueurs. Avec sa conception intuitive et son gameplay stimulant, ce jeu promet d’être une référence dans le monde des jeux de plateforme multijoueur en ligne.

1. Editeur de monde

Dans cette partie nous allons aborder l’application permettant d’éditer les mondes et niveaux du jeu. Il faut effectivement d’abord créer son monde et ses niveaux avant de pouvoir jouer. Nous verrons comment nous avons géré la sauvegarde des mondes, l’édition de ces derniers et comment leurs niveaux sont représentés dans l’éditeur ainsi que son fonctionnement.

1. La représentation d’un monde

Nous aborderons ici la représentation d’un monde et sa gestion uniquement du côté de l’éditeur.

Tout d’abord pour ce qui est de la représentation d’un monde. Il est composé d’une multitude de grille que l’on appellera niveau ou « level ». Un niveau est tout simplement une grille de 60x20, c’est-à-dire une matrice de 60 cases de largeur et de 20 cases de hauteur. Un monde proprement construit et complet est composé de plusieurs niveaux dont seulement un possède la « porte d’entrée » à savoir là où les joueurs apparaissent dans le monde et un autre niveau qui détient la « porte de sortie », c’est-à-dire la fin du monde, là où le joueur doit se rendre pour gagner.

1. **Les éléments d’un niveau**

Chaque niveau possède des éléments mais pour identifier ce que représente chaque élément et pour pouvoir les afficher correctement

1. **Le monde dans l’éditeur**

Pour représenter notre monde dans l’éditeur nous avons créé une structure intitulée « game\_level » qui possède plusieurs paramètres :

* ***width* :** un entier représentant la largeur du niveau
* ***height* :** un entier représentant la hauteur du niveau
* ***elements\_map* :** une matrice en deux dimensions contenant des pointeurs vers les éléments du niveau

Le fait de représenter le niveau sous forme d’une matrice de pointeur nous permet de faciliter la gestion mémoire des éléments en évitant de multiples copies ou encore des oublis de libération de mémoire lors de la suppression ou modification d’un élément. En effet, par exemple, pour ajouter un élément il suffit que chaque case occupé par cet élément pointe vers lui. C’est pourquoi pour le supprimer ou le modifier on ne modifie que le pointeur de l’élément pour que cela soit effectif sur toutes les cases. Lors de la suppression toutes les cases occupées par l’élément sont mises à **NULL** et on libère la mémoire occupée par l’élément.

1. **Le monde dans un fichier**

Pour pouvoir sauvegarder au mieux un monde dans un fichier, il faut que ce dernier soit structuré. Par conséquent, le fichier possède des tables d’adresses qui sauvegarde la position de la représentation des niveaux dans le fichier. On retrouve également des tables de vides qui permettent de savoir où on peut insérer un niveau sans problème.

On rappelle que pour un monde il faut un fichier qui sera composé au minimum d’une table d’adresses, d’une table de vide et d’un niveau. Il ne peut pas y avoir 2 mondes dans le même fichier mais seulement un.

Ensuite, pour minimiser les appels systèmes « read », « write » et « lseek », il a fallut réfléchir à une structure et une méthode spécifique pour enregistrer un niveau. C’est pour cela qu’un élément du niveau n’a pas la même représentation dans le fichier que dans l’éditeur. Dans l’éditeur, un élément est spécifié avec sa position X, sa position Y, son type, sa spécificité alors que dans un fichier nous enregistrons seulement son type suivit de sa spécificité. De plus, les éléments sont enregistré en parcourant chaque case de la matrice représentant le level. C’est-à-dire que l’on enregistre en premier l’élément à la case {0,0}, ensuite la case {0,1} et pour finir à la dernière case {m,n} où m représente le nombre de ligne et n le nombre de colonne. S’il n’y a aucun élément dans la case alors on écrit des 0 pour le type et la spécificité.

Avec cette représentation, il est maintenant facile d’aller lire où écrire un élément spécifique pour le modifier ou le supprimer sans avoir à réécrire tout le niveau dans le fichier, ce qui permet de limiter le nombre d’appel à « read » et « write ».

1. L’aspect serveur
2. L’aspect client
3. Organisation du travail