La Rochelle Université

PROJET JEU

Reseaux transmission

Créer par:

Hector BIDAN

hector.bidan@gmail.com

www.hector-bidan.fr

Introduction

Le but du jeu consiste à bloquer les joueurs adverses en les coinçant entre deux joueurs de votre équipe, mais tout l'intérêt de ce projet se trouve dans la création de celui-ci. Le réel objectif est donc de créer un jeu en réseau suivant le protocole TCP, qui comprends les principales étapes suivantes:

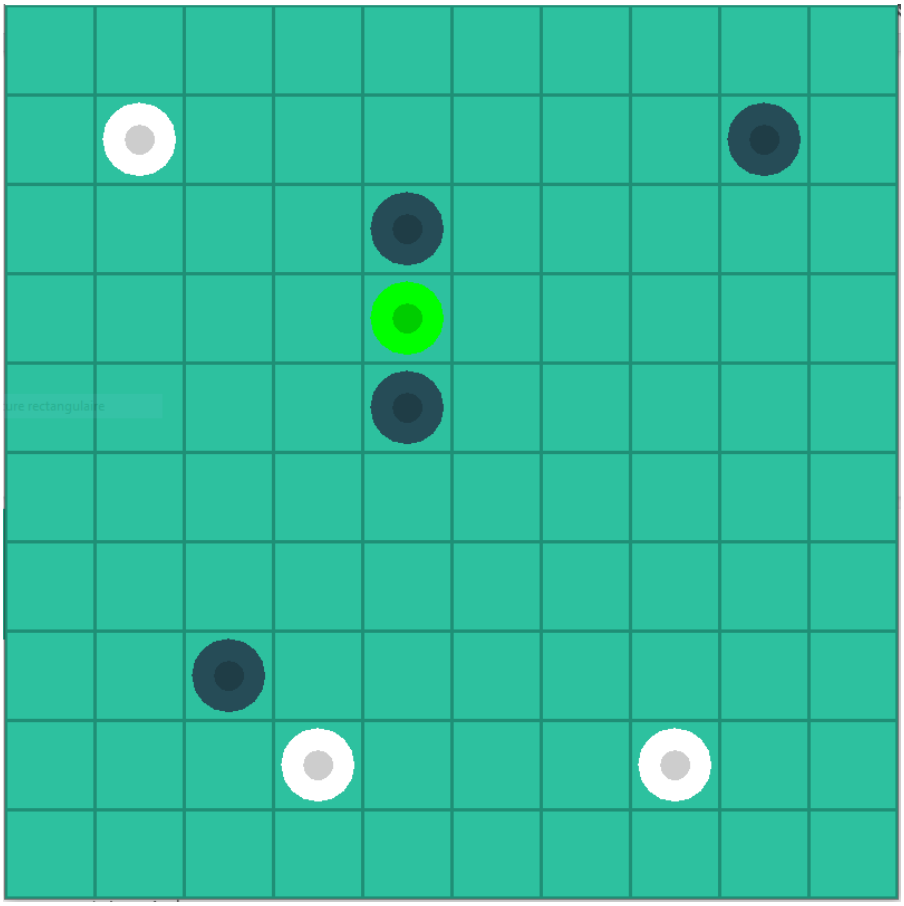
Le serveur s'ouvre et attend la connexion des clients. Lorsqu'un joueur se connecte, le serveur envoie les informations nécessaires pour que le joueur puisse rejoindre la partie en cours.

Chaque fois qu'un joueur veut modifier sa position, il va envoyer cette demande au serveur qui doit la valider lui répondre pour que ce joueur puisse, ou non, bouger.

Si un joueur se retrouve bloqué entre deux joueurs adverses, ce joueur voit son pion en rouge et se retrouve bloqué, les membres de l'équipe adverse le voient en vert.

Lorsqu'un joueur exécute un mouvement, se déconnecte ou se fait éliminer, il faut actualiser tous les clients.

Voici le résultat final du projet 😄:



Principe de communication

Nous avons un thread pour le client et un thread pour le serveur, ces deux-ci vont attendre l'envoi d'un message de l'autre. Ce message sera un entier correspondant à un code qui permettra de définir quelle action faire. *Par exemple, si le client veut demander au serveur de rafraîchir tous les clients, il enverra le code xx.*
Voici un tableau des codes que j'ai utilisé et les actions correspondantes:

Code	**Action**
254	Fermeture du client et le retirer du plateau
253	Fermeture du client
10	Rafraîchir tous les clients
11	Élimination d'un joueur
20	Demande de déplacement
21	Déplacement à droite
22	Déplacement à gauche
23	Déplacement en haut
24	Déplacement en bas

Fonctionnalités

Lors de la connexion d'un client, une vérification à lieu et voici les conditions nécessaire pour l'accepter:

- L'identifiant du joueur n'est pas déjà utilisé
- La position du joueur n'est pas déjà utilisée
- La position du joueur est bien sur la grille
- Son equipe existe (noir/blanc)

Vous pouvez aisément modifier la taille de la grille à l'aide de l'attribut gridSize, vous pouvez essayer de le modifier (par exemple pour une grille de 5x5 cases), toute la grille et la taille des joueurs s'adaptent. 👌

Vous pouvez également modifier la taille de la fenêtre (attribut size), En fonction de celle-ci, le jeu s'adapte également.

J'avais également fait en sorte que lorsqu'une équipe gagne, plus personne ne plus joueur et le serveur attends que chaque client ait fermé la fenêtre pour que le serveur s'arrête (pour qu'il y ait une fin au jeu), mais j'ai modifier le code pour que le jeu soit conforme au TEA.

Procédé & stratégie

Je voulais vraiment avoir un code propre tout le long du projet pour ne pas me perdre j'ai donc beaucoup commenté mon code dès le début et au fur et à mesure, j'ai créer un dossier TODO.txt ou je notais toutes les petites choses que je devrais/pourrait faire tout au long du projet ainsi que les codes de communication que j'ai utilisés.
J'ai également référencé les codes que j'ai utilisé au fur et à mesure pour la mem
Dans un premier temps, j'ai mis en place la communication entre le serveur et ses clients. Une fois cela fait, j'ai commencé à bien réfléchir à une stratégie pour réussir à terminer ce projet à temps:

1. Afficher le joueur à la bonne position et de la bonne couleur sur la grille
Cela me permet de bien comprendre comment fonctionne le tableau de byte représentant la grille (etat) et d'avoir une base visuelle de travail.

2. Actualiser le jeu pour tous les joueurs
En effet, lorsqu'un autre joueur se connecte, cela n'actualise pas son arrivée pour les joueurs déjà connectés. Il me sera ensuite plus facile par la suite de rafraîchir tous les joueurs.

3. Les déplacements
On commence à rentrer dans le vif du sujet, j'ai eu pas mal de difficultés sur ce point (principalement au niveau des collisions entre les joueurs et les bordures de grille), mais une fois terminé, nous avons un jeu qui commence à ressembler à quelque chose. Les joueurs peuvent se connecter et se balader librement sur la grille bien qu'il reste quelques bugs qui seront résolus lors de l'étape suivante.

4. Fin de partie
Il faut détecter à chaque mouvements (et à l'arrivée d'un nouveau joueur) si quelqu'un est bloqué. Si c'est le cas, le serveur prévient que le client a perdu et le bloque. la couleur du joueur pour les autres en fonction de si celui-ci est son allié ou non.

5. Finitions
Plus qu'à peaufiner le projet ! j'ai fait en sorte que la taille de la grille et de la fenêtre soit personnalisables, modifier les couleurs, la grille et les joueurs (histoire de rendre le jeu un peu plus joli 😎), optimisé et commenté un peu plus mon code, ajouté le readme etc…

Conclusion

J'ai pris beaucoup de temps à faire ce projet, en effet je n'y connaissais vraiment pas grand chose avant de vraiment m'y mettre (cela peut se voir à mes TPs), et j'ai par conséquent eu beaucoup de problèmes notamment par rapport à la communication entre le client et le serveur (logique vu que c'est l'intérêt du projet 💢). Cependant, une fois que j'ai bien compris comment cela fonctionnait, je suis allé bien plus rapidement que ce que je pensais. Le résultat de ce projet et le code source se retrouveront bientôt sur mon site (en cours de création) ainsi que sur mon github.