



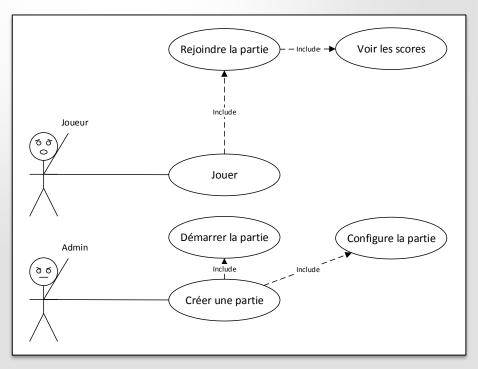
Sommaire

- Présentation du projet
- Cas d'utilisation
- Architecture générale
 - Protocole de communication réseau
 - Diagramme de séquence
 - Base de données
- Interface graphique du client
- Fonctionnalités abandonnées
- Difficultés rencontrées
- Développement restant
- Démo du jeu
- Conclusion

Présentation du projet

- Jeu de bataille 2D en vue de dessus
- Chaque joueur commande un tank
- Le but du jeu est de détruire l'adversaire
- De bonus sont disponibles sur la carte

Cas d'utilisation



Le joueur :

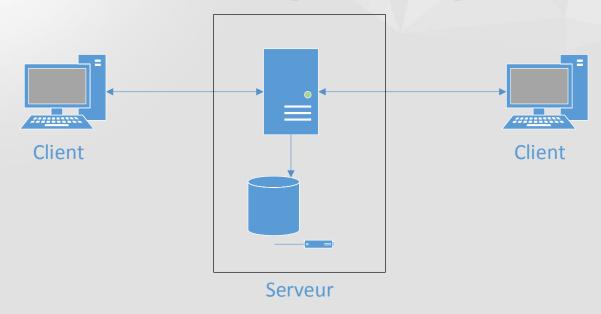
- Jouer
 - Rejoindre une partie
 - Voir les scores

L'admin:

- Créer une partie
 - Configurer la partie
 - Démarrer la partie

Architecture générale

- **►** Architecture client-serveur
- 2 clients nécessaires pour une partie



Protocole de communication réseau

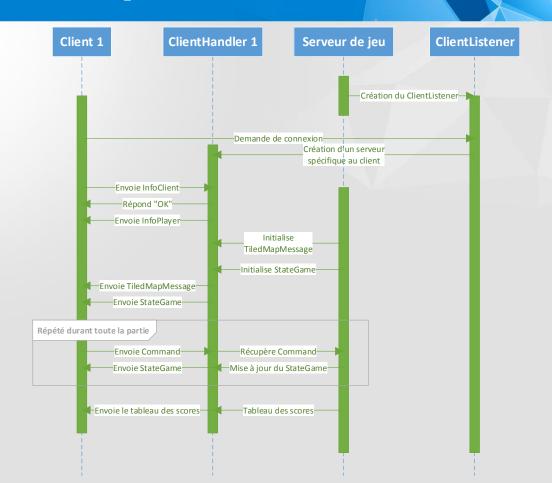
Initialisation :

Sérialisation de classes pour l'initialisation des clients.

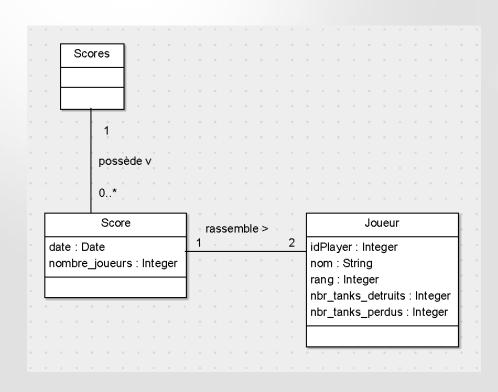
En cours de partie :

- 2 classes sérialisables
 - Une pour le serveur pour envoyer les mises à jours du jeu aux clients.
 - Une pour le client pour envoyer les commandes au serveur.
 - Utilisation de classes héritées pour spécifier la commande envoyée (déplacement, tir, utilisation d'un bonus, etc.).

Diagramme de séquence



Base de données



- Stock les scores de chaque joueurs
- Utilisation d'un fichier XML

Interface graphique du client



Fonctionnalités abandonnées

Le mode observateur

Le choix du tank en début de partie

Le choix de la carte

Les objets destructibles

Difficultés rencontrées

La gestion des collisions

Debug de la communication client-serveur

Merge entre client et serveur

Coordination de l'équipe

Développement restant

- Terminer la communication client-serveur
- Debug des collisions
- Implémentation des bonus restants
- Choix du tank en début de partie
- Relancement de la partie

