GMINT317 - Moteurs de jeux – TP3
Communication client-serveur

Rémi Ronfard  *remi.ronfard@inria.fr*  *https://team.inria.fr/imagine/team/*

Objectifs

Le but de ce TP est de mettre en place des communications client/serveur entre plusieurs fenetres gérées par le moteur de jeu. Ce sera aussi l'occasion de créer des animations de particules et des effets graphiques permettant de simuler le passage des saisons.

Bonus :

- Paralléliser et accélérer les calculs de particules.
- Simuler la création d'une rivière

Gestionnaire de version

Au plus tard la semaine prochaine (minuit), vous devrez rendre un compte rendu de ce TP sur votre espace GIT, ainsi que votre code source.

Mettre en place des communications client/server

La semaine dernière, vous avez mis en place une décomposition de votre application en plusieurs sous-fenêtres. Nous allons réutiliser cette approche pour étudier la communication client/serveur. Afin de simplifier l'implémentation, nous resterons sur la même application.

Pour chaque fenêtre, nous allons intégrer un thread séparé avec la fonction : ***`QThread`***.
Une fenêtre sera utilisée comme serveur (avec la fonction ***QTcpServer***) et les autres fenêtres seront simplement client, en utilisant la classe Qt : ***QTcpSocket***.

Le but est de transférer un message à chaque « application » cliente, en fonction d'un temps écoulé (environ 5 minutes) sur l'application serveur ; pour cela vous devrez utiliser la fonction ***QTimer***.

Nous allons simuler pour chaque fenêtre une période différente de l'année (les 4 saisons). Le message devra ainsi contenir la saison courante de la fenêtre.

Puis, faites circuler les différentes saisons entre les fenêtres.

Vous devez créer une fonction qui initialise le serveur et réaliser un ***connect*** entre un signal : ***newConnection()*** et un slot que vous devez écrire : ***sendSeason()***.

Dans votre header, rajouter le *tag* ***slots*** au niveau de votre fonction ***sendSeason()***. Cette fonction devra envoyer un message à intervalles réguliers à vos différents clients.

Pensez à prendre le temps de faire un commit et un push sur git.

Simuler les changements de saisons

Nos saisons seront caractérisées par différents comportements.

L'été – une lumière vive, une couleur vive au sol
L'automne – pluie et une couleur rouge/orange au sol
L'hiver - chute de neige et une couleur blanche au sol
Printemps - des couleurs vives au sol

Pour vos fenêtres, vous allez devoir ajouter deux nouvelles données :

Une structure pour gérer la pluie, neige (il s'agit de particules)
Et une pour chaque sommet de notre terrain et un attribut de couleur.

Pour la chute des particules, vous allez devoir les habiller à l'aide de ***glPoint***.

Attention : la pluie ne traverse pas le terrain, elle disparaît.

Pensez à prendre le temps de faire un commit et un push sur git.

Compte rendu

Présentez toutes vos nouvelles fonctionnalités
Expliquez les points que vous n'êtes pas arrivé à réaliser et pourquoi.
Expliquez votre démarche de développement.
Présentez votre structure de données.
Expliquez comment vous vous y prendriez pour les parties bonus.

Bonus

- Paralléliser/ accélerer les traitements à l'aide de directive (pragma) OMP
- Proposer une approche pour accumuler les particules, et les transformer en surface (rivière, tas de neige)
- Localiser les effets climatiques : Afficher un rendu différent pour chaque fenêtre, mais gérer les mêmes conditions météo.