Jandot Julien
Moteurs de Jeux

Compte rendu du TP 1 et du TP 2

Question 1 :

GeometryEngine est la classe qui va créer le cube à afficher, et le dessiner.

MainWidget est la classe qui va gérer la fenêtre d'affichage, avec les textures, les shaders, et les événements de la souris ou du clavier. C'est cette classe qui va définir ce que GeometryEngine va utiliser pour dessiner le cube.

Vshader correspond aux shaders appliqués aux arêtes, alors que fshader est appliqué à tous les pixels. Vshader est appliqué avant fshader.

Question 2 :

initCubeGeometry : cette fonction va servir à créer les arêtes et les faces du cube, puis à indiquer l'ordre des triangles à relier. Lors de l'initialisation, on associe une position de l'espace QVector3D, à une position dans la texture avec QVector2D.

drawCubeGeometry : cette fonction va récupérer les textures et les positions de l'objet, puis le dessiner, en utilisant les structures définies dans initCubeGeometry

Question 3 :

La mise à jour du terrain est contrôlée avec l'utilisation du timer et la fonction update. QTimer sert à controller, entre autres, la vitesse de mise à jour de l'affichage. En spécifiant la durée d'activation du timer, on va avoir un affichage plus ou moins fluide.

Une fois les connexions entre les fenêtres implémentées, on se rends compte qu'il est impossible de contrôler toutes les fenêtres depuis une fenêtre au hasard, mais qu'il faut une fenêtre mère.

Les principales difficultés rencontrées ont été sur la compréhension du code déjà existant, et sur la création de l'image plane texturée. Pour y remédier, je suis passé par une phase de développement sur papier avec des schémas.

Pour texturer les sommets en fonction de l'altitude, il aurait fallu borner toutes les altitudes entre 0 et 1, puis appliquer la valeur de ce coefficient sur 255, afin d'obtenir un niveau de gris à affecter à chaque sommet.