Rapport de projet : To the moons

Mesquita Quentin, Goulin Jolan

Sommaire

Présentation

To the moons est un jeu codé en C++ en utilisant l'OpenGL. C'est un jeu basé sur le principe du jeu *Temple Run*.

Dans ce jeu la Terre, n'ayant que la Lune pour amie, se sent seule. Elle décide donc de partir à l'aventure chercher de nouveaux copains satellites. Ce périple l'amène sur une gigantesque route arc en ciel où vivent de nombreuses lunes qu'elle peut récolter. Mais la route possède sa propre conscience et voit l'arrivée de la Terre d'un très mauvais œil ! Celle-ci devra donc faire très attention, car le chemin est rempli de nombreux trous et d'obstacles qui chercheront à la détruire (littéralement).
Le joueur contrôle la Terre et doit collecter le plus de lunes et esquiver les pièges.

Mode d'emploi

Commandes de déplacement :

Q : déplacement à gauche
D : déplacement à droite
Z : saut

Commandes de caméra :

C : Changement de mode caméra (tps/fps)
L : Blocage de la caméra
Attraper et bouger l'écran avec la souris : rotation de caméra (limité à l'axe vertical en vue fps)
Molette de la souris/flèches haut/bas : Zoom / Dézoom

Commandes de jeu :

E : Lancer/relancer le jeu
P : Mettre le jeu en pause / annuler la pause
A : Sauvegarder le jeu
W : Charger la partie sauvegardée
R : Ressusciter et être temporellement immortel (Pour les pauvres dev / testeurs fatigués)

Note : après avoir chargé une sauvegarde, le jeu se met automatiquement en pause. Il faut donc presser la touche **P** pour que le jeu reprenne.

Architecture logicielle :

Afin de créer notre architecture logicielle nous nous somme basé sur celle vue en TP.

Le dossier **Asset** contient les fichiers images et textes nécessaires au fonctionnement du projet.

Le dossier **third-party** contient les librairies glm et stb_image.

Le dossier **Projet** contient les shaders utilisés dans le projet ainsi que projet.cpp le fichier principal.

Le dossier **glimac** contient les fichiers qui nous ont été mis à disposition pour réaliser ce projet, ainsi que d'autres fichiers que nous avons créés afin de réaliser ce projet sous version .hpp et .cpp :

Game object : Fichier contenant la structure personnage qui contient la majorité des informations nécessaires aux jeu.

Map : Fichier contenant les fonction nécessaires à l'initialisation, la génération de la map et les collision de la Terre avec les objets

TrackballCamera : Fichier contenant la gestion de la caméra

PersoDraw : Fichier contenant la gestion de l'affichage des objets cubiques

BoxDraw : Fichier contenant la gestion de l'affichage des objets sphériques

Le fichier projet.cpp utilise ces différents éléments pour créer le jeu et gère lui-même des tâches simples et la gestion des événements liés aux commandes du joueur.

Gestion de la map :

Nous utilisons afin de créer le parcours du jeu une approche différente de celle proposée dans le sujet. En effet, notre parcours est représenté par une grille 1x10 sous la forme d'un tableau de 10 entiers. Ces entiers représentent des types de case (case neutre, case à obstacle, à pièce ou à trou). Quand le joueur avance sur le parcours, le tableau se met à jour en supprimant les éléments dépassés par le joueur et en ajoutant de nouveaux éléments générés aléatoirement parmi les quatre possibilités

Ayant une carte initiale très simple qui se reconstruit au fil du temps, il nous a semblé peu utile de la lire depuis un fichier texte.

Design pattern :

Nous avons choisi dans ce projet d'utiliser le design pattern State. La caméra est définie par des états : un état bloqué/libre et un état fps/tps. Ces états influencent toutes les actions de la caméra.

Extensions du projet :

Le parcours est généré aléatoirement au fur et à mesure de la partie et la vitesse augmente lentement (mais de manière exponentielle).

Il est possible de sauvegarder la partie en appuyant sur *a* et de la charger en appuyant sur *w*.

Côté technique :

Technique	**Utilisé (O/N)**	**Présent dans**
Classes	Oui	BoxDraw.hpp, PersoDraw.hpp, TrackballCamera.hpp, GameObjects.hpp
Template	Non	/
Polymorphisme	Non	/
Encapsulation	Oui	Répartion du code dans les fichiers décrite ci-dessus
STL	Oui	Usage de vector dans : Map.cpp, Project.cpp, GameObject.cpp
Espace de nommage	Oui	Usage du namespace glimac dans tous les fichiers ajoutés a glimac
Exceptions	Oui	PersoDraw.cpp, GameObject.cpp
Messages d'erreur	Oui	PersoDraw.cpp, GameObject.cpp
Assert	Oui	GameObject.cpp
Fonctions lambdas	Non	/
Héritage	Non	/
Constexpr	Non	/
Doxygen	Non	/

Gestion du projet :

Nous avons tous deux commencé le projet seul et avons eu un certain nombre de complications qui nous ont conduit à nous réunir afin de former ce groupe de projet.

Jolan :
J’ai essayé d’utiliser windows pour coder ce projet et j'ai passé plusieurs jours a essayer de paramétrer mon environnement de travail pour y travailler. Je n’ai pas réussi notamment à cause de problèmes avec SDL. J’ai ensuite installé un dual boot linux et paramétrer les librairies nécessaires sur ce dernier. Ceci m’a fait perdre un temps considérable.

Quentin:
Pour ma part, je me suis lancé dans un projet un peu trop ambitieux pour mes seules compétences. Si j’ai voulu faire le projet tout seul au début, c’était pour en apprendre le maximum et vraiment mettre les mains dans le camboui. J’ai tenté de créer un système de jeu très proche de celui du Temple Run original, à base de blocs pré-construits qu’on met les uns à la suite des autres de manières aléatoires. Malheureusement, même après plusieurs jours passés à travailler dessus, je n’ai jamais réussi à les afficher correctement.

Suite à cela nous avons décidé de continuer sur la base du projet de Jolan afin de le continuer et avons travaillé en utilisant une méthode agile.
Nous avons donc travaillé par objectifs incrémentaux, ce qui nous a permis de nous adapter aux difficultées rencontrées le long du chemin.

Satisfaction

Finalement, nous sommes plutôt contents du résultat final : le jeu fonctionne correctement et peut même être amusant.

Mais nous aurions aimé rajouter d'autres choses au projet, comme la présence d’un menu, de l’affichage du score et une musique de fond. Malheureusement, TTF et Mixer n’ont jamais été reconnus par le code, malgré nos nombreuses prières.