TECH

CAHIER DES CHARGES

MOTEUR 3D

INTRODUCTION

Nom du projet :	**Moteur 3D**
N° du Projet :	**VII**
Nom d'UE :	**Développement avancé**
N° d'UE :	**II**
Date :	**3 février 2025**
Classe :	**GTECH 2 Lyon**

Modalités

Groupes de 3 étudiants à former avant le début du projet.

Contexte et description du projet

Développement d'un moteur 3D de jeux vidéo en langage C++ avec Visual Studio. Il sera développé avec l'API DirectX 12 sans utiliser d'autres librairies externes, ni vcpkg.

L'apprentissage se fera également sur la base du développement d'un jeu 3D de type « First-Person Spaceship Shooter » en temps-réel.

Specs

Vous devez implémenter dans votre moteur :

- DirectX12
    - L'initialisation de DirectX12
    - Le rendu 3D (Gameplay) et 2D (UI)
    - Un rendu simple de meshes procéduraux
    - Des shaders basiques
    - Un système de particules

    - Gestion des lumières (forward ou deferred pour les + avancés en utilisant des spot lights)

- Architecture
    - Un modèle ECS (entités, composants, systèmes)
    - Des managers et une machine à état générique
    - Des composants pour gérer les positions et rotations, les meshes, la physique, les collisions simples, les UI, les caméras, le code gameplay via des scripts c++...
    - Les contrôleurs Clavier et Souris
    - La création et l'utilisation de Templates
    - Une structure TRANSFORM (position, rotation, scaling avec des vecteurs, matrices et quaternions qui devront faire références aux éléments de DX12)

- Optimisations
    - Une méthode pour partitionner l'espace afin d'optimiser les collisions
    - Une méthode pour "clipper" les objets non visibles à l'écran
    - Un système flexible pour gérer le temps afin de produire des ralentis in-game
    - Une couche réseau avec les threads

<u>Vous devez implémenter dans votre jeu d'exemple :</u>

- Gameplay
    - Un niveau complet procédural en générant l'apparition des ennemis hostiles
    - Des obstacles à viser en plus des ennemis
    - Des projectiles
    - Une barre de vie
    - Un shader post processing pour modifier la saturation, la luminosité et le contraste du rendu final (activable ou désactivable via la touche F5)
    - Un ralenti soit durant la partie soit du durant une phase de transition

    - Un mode Replay (séquence enregistrée et rejouée après avoir perdu)
    - À vous de compléter votre gameplay pour obtenir un défi satisfaisant afin de mettre en avant les fonctionnalités du moteur

- UI
    - Un écran Splash ou Menu
    - Un score visible à l'écran pendant la partie

Consignes à respecter :

- Solution Generator 1.15
- Fonctions lambda interdites
- Mettre en place une nomenclature rigoureuse
- Pas de code dans le header à l'intérieur de la classe
- LOGs désactivables en Release

Informations complémentaires :

Le plus important est de réussir à finir le projet avant la date butoir en implémentant toutes les fonctionnalités imposées. Il est évident que la qualité du code est également au cœur de votre formation. Votre code doit avant tout être évolutif et pour y parvenir vous devez vous focaliser sur l'architecture de votre application, la nomenclature employée, la gestion de la mémoire, le temps d'exécution des instructions, l'absence de bugs, la mise en place d'une méthode de débogage...

Savoir-être

À l'issue de ce projet, les étudiants seront capables de :

- Travailler efficacement en groupe
- Planifier le bon déroulement du projet
- Respecter les délais
- Évaluer sa propre contribution
- Reconnaître la contribution des autres

Ressources

Documentation technique DirectX12 dans le dossier RESSOURCES :
https://drive.google.com/file/d/12w_eL3UkSJlriurCOa-AnAkRQsMqx7JY

Travail préparatoire

Documentation technique DirectX12 dans le dossier RESSOURCES :

- Créer une solution avec deux projets Desktop C++ (une lib pour le moteur et un exécutable pour le jeu)
- Le moteur doit linker avec DX et le jeu avec le moteur
- Savoir afficher un triangle à l'écran (avec un dégradé de 3 couleurs)
- Les termes à connaître par cœur :
    - RTV: render target view
    - DSV: depth stencil view
    - CBV: constant buffer view
    - SRV: shader resource view
    - VBV: vertex buffer view
    - IBV: index buffer view
    - PSO: pipeline state object

INTERVENANT

Nom	**Sylvain Seccia**
Titre	**Auteur & Développeur**
Email	**sylvain@seccia.com**
Site web	**https://www.seccia.com**
LinkedIn	**https://www.linkedin.com/in/sylvainseccia**

Biographie

Auteur et développeur de jeux vidéo et logiciels depuis la fin des années 90.

Développeur outils et framework chez Darkworks, Gameloft et Kubity à Paris de 2010 à 2017.

Conception et développement d'un logiciel de création de jeux d'aventure 2D no-code destiné aux artistes et scénaristes : seccia.dev

Création et distribution de nombreux jeux vidéo indépendants, notamment « Désiré » sorti en 2016, jeu multi-primé en France et à l'étranger.

Réalisation de court-métrages en noir et blanc, notamment « Un banc pour deux » et « La fille du pont ».

Intervenant chez Gaming Campus depuis 2021.

ROADMAP

Description

Veuillez trouver ci-dessous une description des livrables attendus ainsi que les dates d'échéance associées. Il est essentiel de respecter les échéances suivantes pour assurer une progression harmonieuse et structurée du projet. Chaque livrable représente une étape importante dans le processus de réalisation et permet d'évaluer l'avancement du travail. Les dates limites fixées doivent être rigoureusement respectées afin de garantir une évaluation équitable et de permettre un feedback constructif en temps opportun.

Jalons

Jalon	Livrables attendus	Date limite	Remarques
1	Mise en place du projet	10 février 2025 avant le début du coaching	Constitution des groupes et création du repo sur le compte GitHub de l'école
2	Version fonctionnelle	21 février 2025	Faire la démonstration d'un moteur fonctionnel
3	Projet final	7 mars 2025 avant 8h	Rendu du projet sur le compte du GitHub de l'école
4	Soutenance (fournir la présentation)	7 mars 2025	Les étudiants doivent assister à toutes les soutenances de leur classe et l'ordre de passage est annoncé le jour J

GRILLE D'ÉVALUATION

L'évaluation est conçue pour être holistique, prenant en compte non seulement le produit final, mais aussi le processus, les compétences acquises et les attitudes démontrées tout au long du projet.

Barème du projet

Semaine des fondamentaux	Semaine(s) du projet	Rendu final	Soutenance	NOTE
Assiduité individuelle	*Assiduité individuelle*	*Groupe*	*Groupe*	
/5	**/5**	**/10**	**-1, 0 ou +1***	**/20**

Barème de la notation d'assiduité individuelle

0	**Néant**
1	**Niveau inquiétant** (manque de travail, d'implication, de méthode, de concentration, de sérieux, de résultat, de compréhension, manque d'effort, manque de remise en question, projet à refaire entièrement)
2	**Niveau insuffisant** (en retard par rapport aux attendus du programme)
3	**Niveau scolaire** (manque d'analyse, de curiosité, d'analyse, de recherche, ne prend pas de recul)
4	**Profil conforme aux attendus du monde professionnel** (votre intervenant vous projette en entreprise comme junior, curiosité accrue, pousse les projets plus loin, remise en question permanente)
5	**Profil d'élite** (état d'esprit pour travailler dans de grandes entreprises)

Remarques

Les rendus doivent impérativement être disponibles pour le jury avant la date butoir depuis le compte GitHub de l'école.
La soutenance est évaluée lors de la présentation finale du projet. Les critères incluent la clarté, la cohérence, et la capacité à articuler et défendre les résultats du projet.

* la note globale ne peut être ni inférieure à 0 ni supérieure à 20