



TECH

CAHIER DES CHARGES

ASSET ENGINE



INTRODUCTION

Nom du projet : **Asset Engine**
N° du Projet : **3**
Nom d'UE : **Développement avancé**
N° d'UE : **1**
Date : **20 octobre 2025**
Classe : **GTECH 3 Moteur Lyon**

MODALITÉS

Groupes de 2 étudiants à former avant le début du projet.

CONTEXTE ET DESCRIPTION DU PROJET

Le projet Asset Engine a pour objectif de concevoir et d'implémenter un système d'archivage binaire permettant de regrouper et de gérer des ressources numériques (fichiers, données, assets divers) au sein d'un seul et même fichier.

L'enjeu est de comprendre comment structurer, écrire et relire un format binaire personnalisé en C++, sans dépendre de bibliothèques externes autres que celles fournies par Windows et la bibliothèque standard.

Ce travail vous amènera à penser votre propre format, à définir les conventions de stockage, à gérer la sérialisation et la désérialisation des données, et à concevoir une petite interface en ligne de commande permettant de manipuler vos archives.

OBJECTIFS PÉDAGOGIQUES

- Apprendre à concevoir un format de fichier binaire (en-tête, table des contenus, données, métadonnées).
- Comprendre les mécanismes de lecture et d'écriture bas-niveau (fichiers binaires, alignement mémoire, offsets).
- Mettre en œuvre la sérialisation (conversion de structures en flux de bytes) et la désérialisation (lecture et reconstruction en mémoire).

- Gérer l'intégrité des données (vérifications, CRC, validation d'offsets).
- Concevoir une API simple et cohérente pour lire et écrire des archives.
- Créer un outil exécutable permettant d'emballer, d'extraire ou de lister les ressources.
- Documenter le format et justifier vos choix de conception.

THÉORIE ET EXERCICES

Cette première phase est consacrée à la compréhension des notions fondamentales liées à la manipulation de données binaires et à la conception de formats structurés.

Contenu de la semaine

- Introduction aux formats binaires (structure, en-tête, tables de contenus, blocs de données).
- Lecture et écriture de fichiers en mode binaire.
- Organisation interne des données : offsets, alignements, tailles, encodage des chaînes.
- Notions de sérialisation et de désérialisation.
- Contrôle d'intégrité et validation (CRC, versioning, compatibilité).
- Conception d'une API claire pour l'accès aux données sérialisées.
- Exercices guidés pour s'approprier les opérations de base.

L'objectif est que chaque étudiant soit à l'aise avec la manipulation de fichiers binaires, la lecture/écriture de structures, et la compréhension des contraintes de bas niveau avant d'entamer la phase de production.

CONCEPTION ET DÉVELOPPEMENT DE L'ASSET ENGINE

Pendant cette semaine, vous concevrez puis réaliserez votre propre moteur d'assets sous la forme d'un outil capable de regrouper différents fichiers dans une archive binaire unique, lisible et validable.

Travail attendu

- Définir la structure interne de votre format (entête, table d'index, blocs de données, etc.).

- Implémenter un système permettant d'ajouter, stocker et relire différents types de données.
- Assurer la cohérence et la validité des informations contenues dans l'archive (notamment grâce au CRC32 ou autres plus robustes).
- Implémenter un système de « purge » pour ne pas réécrire l'entièreté de l'archive pour l'ajout et la suppression d'un fichier.
- Implémenter un modèle de chiffrement symétrique des données à l'aide d'une clé privée
- Pour les plus avancées : implémenter une librairie zlib pour compresser les données.
- Proposer une interface en ligne de commande permettant de :
 - o Créer une archive à partir d'un dossier ou d'une liste de fichiers
 - o Afficher son contenu
 - o Extraire des ressources spécifiques
 - o Vérifier l'intégrité du fichier
 - o Intégrer une commande -help
- Ecrire un exemple en fichier .BAT
- Documenter le fonctionnement global du format et le processus de lecture/écriture.

Chaque équipe devra donc concevoir un format cohérent, évolutif et robuste, accompagné d'un outil fonctionnel et d'une documentation claire.

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...

LIVRABLES

En fin de semaine de projet, chaque groupe devra remettre :

- Le code source complet du projet.
- Un projet qui compile un exécutable de l'outil d'archivage en Debug et Release.

- Un document de spécification décrivant la structure interne du format binaire.
- Un ou plusieurs exemples d'archives de test.

CONTRAINTES

- Utilisation du Solution Generator à inclure dans le dépôt
- Le projet doit être intégralement réalisé en C++.
- Aucune librairie externe n'est autorisée (sauf API système Windows, STL et zlib).
- L'archive doit être autonome et lisible sans outil tiers.
- Tous les chemins et chaînes doivent être encodés en ASCII.
- Le format doit être pensé pour pouvoir évoluer dans le temps (numéro de version, champs réservés, etc.).
- Une attention particulière doit être portée à la stabilité du code et à la gestion des erreurs.

CONSEILS

- Avant de coder, documentez votre format sur papier : structure du fichier, ordre des blocs, offsets, alignements.
- Testez régulièrement avec de petits fichiers avant de passer à des jeux de données complets.
- Gardez une trace de vos décisions techniques : elles seront utiles dans le rapport final.
- Pensez la sérialisation comme un protocole : ce qui est écrit doit pouvoir être relu sans ambiguïté.
- Privilégiez la clarté à la complexité : un format simple, cohérent et bien documenté vaut mieux qu'un format ambitieux mais instable.

SAVOIR-ÊTRE

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

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



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 », « La fille du pont » et « À contretemps ».

Publication d'un recueil de nouvelles « Venelle » sur Amazon.

Intervenant chez Gaming Campus depuis 2021.



ROADMAP

DESCRIPTION

Veillez 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	24 octobre 2025	Constitution des groupes et configuration du repo GitHub de l'école
2	Projet final	31 octobre 2025 avant 8h	Rendu des projets sur le compte du GitHub de l'école
3	Revue de code	31 octobre 2025	Les étudiants doivent assister à toutes les revues de code 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.

Description	Barème
Qualité technique du code (lisibilité, structure, bonnes pratiques, absence de bugs et fuites mémoire)	/10
Fonctionnalités implémentées (complétude, conformité aux objectifs fixés)	/10
Présentation orale (clarté du discours, démonstration, capacité à répondre aux questions)	/10
Total	/30

Remarques

Les rendus doivent impérativement être disponibles pour le jury avant la date butoir depuis le compte GitHub de l'école.

La soutenance ou la revue de code 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.