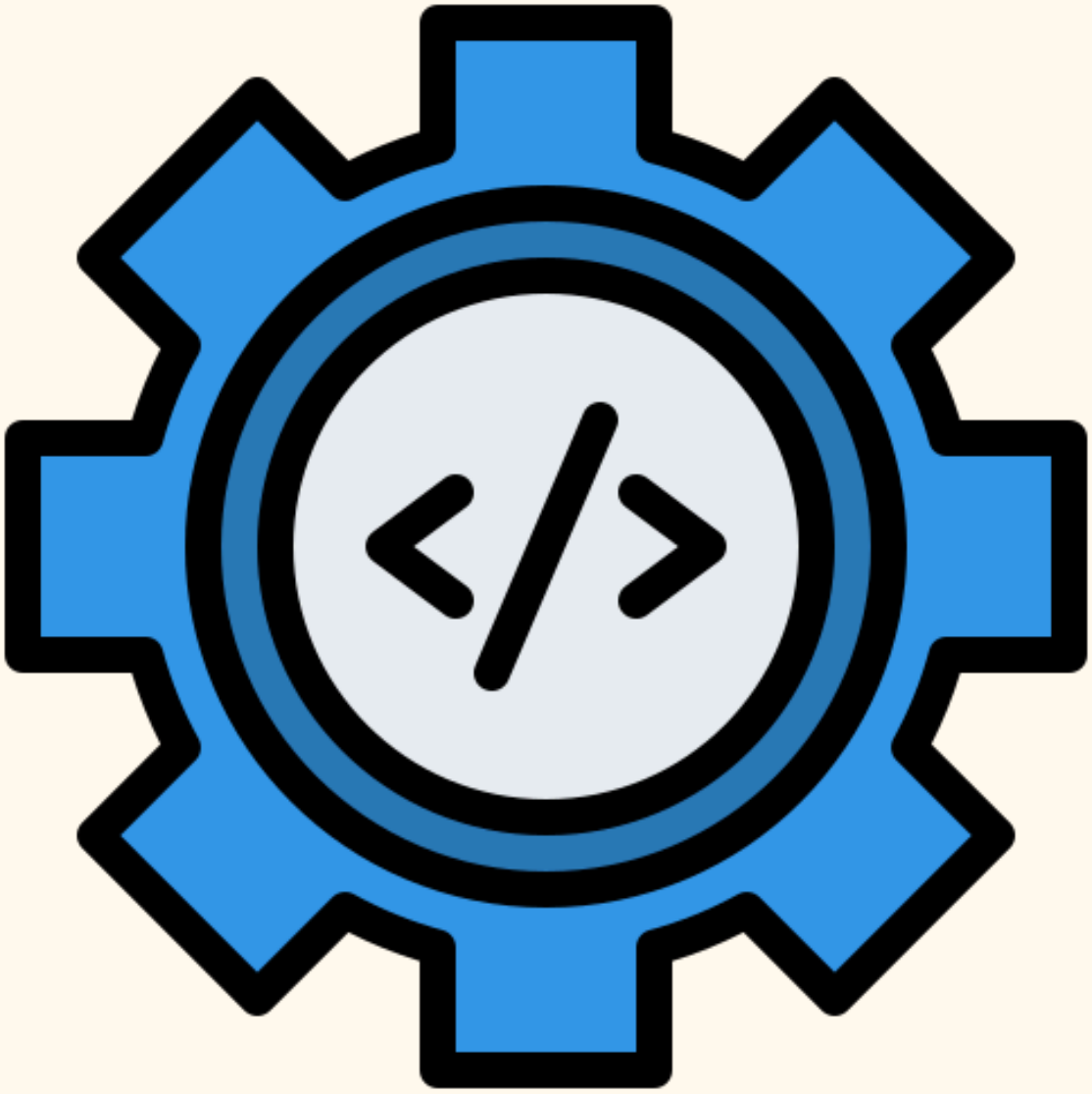
EPITECH.
L'ECOLE DE L'EXPERTISE INFORMATIQUE



COMMENT IMPLÉMENTER DE NOUVEAUX FICHIERS DE CONFIGURATIONS

Projet RayTracer : Marius Pain, Landry Gigant, Thomas Boué & Aubane Nourry

Introduction

L'intégration de nouveaux fichiers de configuration pour le projet RayTracer d'Epitech requiert une approche méthodique et structurée, étroitement alignée avec une architecture spécifique afin de garantir sa validité lors de la récupération des différentes composantes du fichier. Dans ce document, nous explorerons les étapes essentielles pour implémenter de nouveaux fichiers de configuration dans le cadre du projet RayTracer. Nous mettrons en lumière les différentes composantes nécessaires et les conventions établies dans le but de garantir la compatibilité de ces fichiers de configurations avec notre projet, tout en assurant une grande liberté lors de la création de ces fichiers.

Créer un nouveau fichier de configuration

Un fichier de configuration comprend de nombreux éléments qui, une fois récupérés et traités par le programme seront intégrés à une scène puis à l'aide d'un raycasting une image de la scène sera généré. Nous allons donc voir comment vous pouvez créer une scène personnalisée à l'aide d'un fichier de configuration.

1. LibConfig

Vous devrez dans un premier temps créer télécharger la bibliothèque libconfig (pour le C++), si cela n'est pas fait référencer vous au document : HowToInstallLibConfig . Une fois la bibliothèque installée vous devriez pouvoir compiler le projet RayTracer avec la commande : `make` .

Puis, pour le format du fichier, celui-ci utilise le format créé pour l'utilisation de la libconfig, il vous faudra donc le lire et l'appliquer lors de la création de votre fichier de configuration afin de vous éviter des messages d'erreur tel que :

- "`The configuration file is invalid`"
- "`Invalid setting type in the configuration file`"

Pour cela, rendez-vous sur le manuel de libconfig.

2. Les différents éléments

Afin d'implémenter une scène qui pourra être transformée en image vous devrez d'abord créer plusieurs types d'éléments tels que : la caméra (requis), les lumières (requis), les formes (optionnelles) et les transformations (optionnelles).

a. La Caméra

Pour mettre en place un caméra (le point de vu d'où sera généré votre scène), vous devez créer un objet nommé "camera" qui contient les éléments suivants :

- "resolution" : Cet objet permet de donner la résolution que l'on souhaite pour l'image générée. Cet objet contient les champs :
  - "width" : Nombre entier qui correspond à la largeur de la résolution.
  - "height" : Nombre entier qui correspond à la hauteur de la résolution.
- "position" : Cet objet permet de donner la position que l'on souhaite pour notre caméra. Cet objet contient les champs :
  - "x" : Nombre entier qui correspond à la position en x de la caméra.

    - “y” : Nombre entier qui correspond à la position en y de la caméra.
    - “z” : Nombre entier qui correspond à la position en z de la caméra.
- “rotation” : Cet objet permet de donner la rotation que l’on souhaite pour notre caméra. Cet objet contient les champs :
    - “x” : Nombre entier qui correspond à la rotation en x de la caméra.
    - “y” : Nombre entier qui correspond à la rotation en y de la caméra.
    - “z” : Nombre entier qui correspond à la rotation en z de la caméra.
- “fieldOfView” : Nombre décimal qui correspond au champ de vue de la caméra en degrés.

b. Les lumières

Pour mettre en place les lumières présente dans la scène, vous devez créer un objet nommé “lights” qui contient les éléments suivants :

- “ambiant” : Nombre décimal qui correspond à l’intensité de la lumière ambiante (0 correspondant à pas de lumière ambiante et 1 à un lumière ambiante maximale).
- “diffuse” : Nombre décimal qui correspond à l’intensité des lumière diffuse (0 correspondant pas de lumière et 1 à un lumière diffuse maximale).
- “ambiantColor” : Cet objet permet de donner la couleur que l’on souhaite pour notre lumière ambiante (s’il n’est pas présent alors la lumière ambiante est blanche). Cet objet contient les champs :
    - “r” : Nombre entier (entre 0 et 255) qui correspond à la valeur de la composante rouge de la lumière.
    - “g” : Nombre entier (entre 0 et 255) qui correspond à la valeur de la composante verte de la lumière.
    - “b” : Nombre entier (entre 0 et 255) qui correspond à la valeur de la composante bleu de la lumière.
- “directional” : Cet objet permet de créer différentes lumières directionnelles à d’un vecteur. Cet objet contient les champs :

- “name” : Chaîne de caractère correspondant au nom de la lumière.
- “x” : Nombre entier qui correspond à la valeur en x du vecteur de la lumière.
- “y” : Nombre entier qui correspond à la valeur en y du vecteur de la lumière.
- “z” : Nombre entier qui correspond à la valeur en z du vecteur de la lumière.
- “color” : Objet correspondant à la couleur de la lumière directionnelle (si absent alors la lumière sera blanche). Cet objet contient les champs :
  - “r” : Nombre entier (entre 0 et 255) qui correspond à la valeur de la composante rouge de la lumière.
  - “g” : Nombre entier (entre 0 et 255) qui correspond à la valeur de la composante verte de la lumière.
  - “b” : Nombre entier (entre 0 et 255) qui correspond à la valeur de la composante bleu de la lumière.

c. Les formes (les primitives)

Pour mettre en place les primitives présentes dans la scène, vous devez créer un objet nommé “primitives” qui contiendra plusieurs listes d’un certain type de primitive (exemple : une liste “spheres” pour les sphères, une liste “planes” pour les plans, ect …). Chaque liste est composé d’objets (ceux-ci ne possèdent pas de nom) et ces objets contiennent chacun les champs suivant :

- “name” : Chaîne de caractère correspondant au nom de la primitive.
- “x” : Nombre entier qui correspond à la position en x de la primitive.
- “y” : Nombre entier qui correspond à la position en y de la primitive.
- “z” : Nombre entier qui correspond à la position en z de la primitive.

- “color” : Objet correspondant à la couleur de la matière de la primitive. Cet objet contient les champs :
    - “r” : Nombre entier (entre 0 et 255) qui correspond à la valeur de la composante rouge de la lumière.
    - “g” : Nombre entier (entre 0 et 255) qui correspond à la valeur de la composante verte de la lumière.
    - “b” : Nombre entier (entre 0 et 255) qui correspond à la valeur de la composante bleu de la lumière.
- “material” : Chaîne de caractère correspondant au nom du matériaux utilisé pour construire la primitive (vous devrez choisir le matériaux parmis la liste de matériaux déjà implémenté en tant que plugins)

  <u>Le reste des champs dépendent de la primitives que vous souhaitez avoir dans votre scène, voici quelque exemple de champs supplémentaires :</u>

- Pour les sphères :
    - “r” : Nombre entier correspondant au rayon de la sphère qui sera créée.
- Pour les plans à partir d’un axe :
    - “axis” : Chaîne de caractères contenant soit “X”, “Y” ou “Z” qui correspond à l’axe du plan qui sera créé.
- Pour les plans à partir d’un vecteur :
    - “vector” : Objet correspondant au vecteur qui sera utilisé pour créer le plan. Cet objet contient les champs :
        - “x” : Nombre entier qui correspond à la valeur en x du vecteur du plan qui sera créé.
        - “y” : Nombre entier qui correspond à la valeur en y du vecteur du plan qui sera créé.
        - “z” : Nombre entier qui correspond à la valeur en z du vecteur du plan qui sera créé.

d. Les transformations

Pour mettre en place les transformations présente dans la scène, vous devez créer une list d'objet nommé "transformations" qui contiendra une liste d'objet (ceux-ci n'ont pas de nom), chaque objet contiendront les champs suivant :

- "type" : Chaîne de caractère qui définit le type de transformation à appliquer à la primitive cible.
- "target" : Chaîne de caractère qui définit le nom de la primitive cible qui subira la transformation.
  Le reste des champs dépendent de la transformation que vous souhaitez appliquer voici quelque exemple :
- Pour une translation :
  - "x" : Nombre entier qui correspond à la valeur en x du vecteur de la translation.
  - "y" : Nombre entier qui correspond à la valeur en y du vecteur de la translation.
  - "z" : Nombre entier qui correspond à la valeur en z du vecteur de la translation.
- Pour une rotation :
  - "x" : Nombre entier qui correspond à la valeur en x du vecteur de la rotation.
  - "y" : Nombre entier qui correspond à la valeur en y du vecteur de la rotation.
  - "z" : Nombre entier qui correspond à la valeur en z du vecteur de la rotation.
  - "angle" : Nombre entier qui correspond à la valeur de la rotation en degrés par rapport au vecteur de rotation.
- Pour une mise à l'échelle :
  - "multiplier" : Nombre décimal qui correspond au facteur de mise à l'échelle.

3. Les Erreurs

Lorsque vous aurez fini de créer votre fichier de configuration pour créer votre propre scène, vous voudrez sans doute le tester mais si jamais celui-ci ne marche pas du premier coup (s'il marche félicitation, vous êtes meilleurs). Sinon vous pouvez tomber sur un des type d'erreurs suivant :

- "`SceneMissingCameraException`" : Il n'y a pas de caméra renseigné dans le fichier de configuration
- "`SceneMissingLightException`" : Il n'y a pas de lumières renseigné dans le fichier de configuration
- "`SceneParseError`" : La libconfig n'arrive pas à traiter votre fichier de configuration, il y a une erreur syntaxique.
- "`SceneCameraException`" : Il manque un des champs nécessaires à la caméra (parmis résolution, position, rotation et fov).
- "`SceneLightException`" : Il manque un des champs nécessaire à vos lumières.
- "`SceneDuplicateNameException`" : Deux lumières ou deux primitives ont le même nom.
- "`SceneUnknownObjectException`" : Le type de primitive que vous essayez de créer n'existe pas ou alors vous essayé d'appliquer un transformation à un éléments qui n'existe pas.
- "`SceneInvalidTransformationException`" : Le type de transformation que vous essayer d'appliquer, n'est pas possible avec cette primitive.
- "`SceneUnknownTransformationException`" : Le type de transformation que vous essayez de réaliser n'existe pas.

4. Exemples

Vous pourrez trouver plusieurs exemples de fichier de configuration pour différentes scènes déjà créées dans le dossier “scènes”. Vous pourrez également trouver le résultat de l’image dans le dossier “screenshots”.

5. Conclusion

Vous avez désormais toutes les clefs en main pour implémenter un nouveau fichier de configuration pour notre raytracer, n’hésitez pas à faire une pull request pour ajouter votre fichier de configuration si vous souhaitez contribuer au projet. Nous vous remercions par avance de votre contribution.