TP N°2 : Description d'un graphe de scène et liaison avec l'application

L'objectif de ce TP est de créer un langage permettant de décrire un graphe de scène (une structure de données représentant une scène géométrique 3D) ainsi que d'écrire le chargeur qui lui est associé.

1. Contexte du TP

a. La notion de graphe de scènes

Un graphe de scène est une structure de données représentant l'organisation d'une géométrie représentant une scène 3D. Dans un premier temps, ce graphe peut être vu comme un arbre (possédant donc une racine unique) pour lequel les nœuds internes sont des transformations géométriques (facteur d'échelle, translation, rotation…) et les feuilles sont des géométries. Les nœuds internes représentent des opérations (transformations et autres) qui sont appliquées aux nœuds fils. L'appellation graphe de scène au lieu d'arbre de scène vient du fait que des sous arbres peuvent être partagés au sein de la structure globale, créant ainsi un graphe acyclique orienté.

Dans ce TP, une structure pré-codée de graphe de scène vous est fournie. Il s'agira alors de créer un langage permettant de décrire le graphe et, dans un second temps, d'analyser ce langage pour construire le graphe de scène décrit.

b. Préparation de l'environnement

Dans le répertoire *share/esir2/cin*, vous trouverez un répertoire nommé *tp_cin_scenegraph* contenant les éléments utiles à ce TP :

- le répertoire *jogl* : contient Java OpenGL i.e. une interface Java vers OpenGL. Le répertoire contient plusieurs fichiers jar (les bibliothèques Java utiles à jogl) ainsi que plusieurs DLL (les bibliothèques natives permettant à jogl d'accéder à l'OpenGL natif installé sur un système Windows). Attention, si vous souhaitez utiliser jogl sur un système type Linux ou MacOS, il faudra vous rendre sur le site de jogl et télécharger la version propre à votre système.
- *antlr-3.2.jar* : un jar contenant la bibliothèque associée à ANTLR.
- *TP_CIN.zip* : une archive au format ZIP contenant le code source Java associé à ce TP. Cette archive contient l'implémentation d'un graphe de scènes ainsi qu'une application test.

L'implémentation d'un graphe de scène qui vous est fournie comporte les nœuds suivants : Group, Rotation, Translation, Cone, Cube, Cylinder, Sphere, Teapot, Tetrahedron, Torus. Cette implémentation vous est fournie dans le paquetage *Lib3d.GlSceneGraph*.

c. Elements pour le rendu de TP

Ce TP est un TP projet. Vous aurez un langage à spécifier et vous devrez fournir un chargeur Java construisant un graphe de scène à partir de sa description. Le rendu de TP devra comporter un volet de spécification du langage (description de la grammaire avec la sémantique associée). D'autre part, il devra expliquer votre démarche de conception du chargeur et sa connexion à la structure de graphe de scène qui vous est fournie.

2. Vers un langage de description de graphe de scène

Dans un premier temps, nous allons décrire un arbre de scène. Autrement dit, nous ne gérerons pas le partage des sous arbres qui sera traité dans les questions suivantes. Nous allons créer un langage permettant de décrire la structure d'un graphe de scène composé des nœuds présents dans le graphe de scène qui vous est fourni (décrits à la fin de la section 1.b). Ce langage devra accepter la description de la structure arborescente du graphe ainsi que la description des différents attributs liés à un nœud donné (le rayon d'une sphère, le vecteur de translation etc…). Dans un premier temps, vous ne gèrerez pas les matériaux associés aux objets mais simplement leur géométrie.

Question 1 : Travaillez sur une spécification du langage : les besoins, sa syntaxe et les manières de décrire les éléments. Essayez de penser ce langage afin qu'il soit facile à utiliser.

Question 2 : Ecrivez la grammaire du langage que vous avez spécifié et testez là sous ANTLR. A des fins de validation, vous concevrez plusieurs fichiers exemples bien formés et mal formés.

Question 3 : Décorez la grammaire afin de construire le graphe de scène lors de l'analyse du texte fourni en entrée.

Question 4 : Intégrez votre grammaire au sein de l'application exemple qui vous est fournie. L'application chargera donc un fichier exemple et permettra de visualiser le graphe de scène décrit par le fichier.

Question 5 : Modifiez votre grammaire afin de pouvoir associer un matériau aux objets géométriques (colorimétrie et éventuelle texture associée).

Question 6 : Intégrez le langage ainsi modifié au sein de votre application.

Question 7 : Afin de pouvoir décrire un graphe de scène et non plus un arbre de scène, il faut être à même de partager des sous arbres. Pour ce faire, vous allez modifier votre grammaire afin que cette dernière accepte le nommage des nœuds et la réutilisation d'un nœud nommé aux endroits appropriés. Cela permettra de nommer un sous arbre d'une part et de réutiliser le sous arbre ainsi nommé d'autre part. Modifiez votre grammaire et testez le bon fonctionnement de vos modifications.

Question 8 : Nous souhaitons pouvoir décrire des trajectoires associées aux objets géométriques. Ces trajectoires seront décrites par l'intermédiaire de fonctions (implicitement paramétrées par une date exprimée en secondes) retournant des vecteurs 3D ou des scalaires. Spécifiez ce langage et fournissez la grammaire associée (NB : vous pouvez éventuellement vous appuyer sur le langage décrit au premier TP). Dans votre rapport, vous prendrez soin de discuter des éventuelles limites de ce langage.

Question 9 : Modifiez votre grammaire de description de graphe de scène afin que ces fonctions puissent être utilisées dans la description afin de fournir le comportement des nœuds de translation et des nœuds de rotation associés à votre langage de description.

Question 10 (subsidiaire) : Intégrer ces modifications dans votre chargeur afin que ce dernier puisse animer les scènes.