BSI TP0: Exemple minimal OpenGL

Loïc Simon

September 30, 2015

L'objectif de ce TP est d'appréhender les bases de l'API d'OpenGL moderne pour être capable d'initier des rendus 3D. Dans ce TP et les suivants nous utiliserons la librairie GLFW pour créer un contexte OpenGL Core Profile et gérer les interactions utilisateur.

1 Instructions préliminaires

1.1 Installation locale de GLFW et autres

Pour vous faire gagnez du temps et de l'espace disque, GLFW et d'autres librairies sont déjà installées dans un dossier commun. Pour que leur installation soit effective pour vous, vous devez toutefois effectuer les instructions suivantes:

```
cd $HOME
ln -s /home/public/simonl/Synthese/local_install .
```

1.2 Récupération du code et premiers tests

Pour réaliser les exercices suivants, vous devez cloner le répertoire git mis a disposition par votre enseignant:

```
  cd pathtoworkdir  # à adapter (par exemple $HOME/Bureau/TPSI)
  git clone http://www.ecole.ensicaen.fr/\~simonl/gits/GLLabs.git/
  cd GLLabs
  mkdir build
  cd build
  cmake ..
  make
```

Une fois la compilation terminée, vous disposez d'un exécutable que vous pouvez tester:

```
  ./fromScratch/myOwnOpenGLProg
```

Dans ce TP vous travaillerez uniquement sur le fichier

`fromScratch/myOwnOpenGLProg.cpp`.

Dans son état original, l'exécutable créé, affiche un message d'encouragement sur la console.

1.3 Structure du répertoire

Le répertoire est constitué de plusieurs dossiers ayant chacun leur utilité:

- `resources` est destiné a stocker les ressources de type GLSL, textures, maillages.
- `include` contient un unique header `lightGlAPI.hpp` qui implémente l'API de la sur-couche d'OpenGL utilisée dans les TPs suivants (inutile ici).
- `utils` contient des routines de plus bas niveau, utilisées dans l'API précédente. Ces fonctionnalités proviennent pour la plupart du tutoriel: `http://www.opengl-tutorial.org/`. Ces routines ne seront normalement pas utilisées directement.
- `fromScratch` contient le code nécessaire pour ce premier TP. D'autres répertoire du même type seront ajoutés pour les autres TPs.

1.4 Gestion de git

Git est un outil de suivi de version. Si vous voulez éviter tout souci, il vous faut indexer votre travail régulièrement. Pour ce faire, vous pourrez utiliser la commande:

```
1  git commit -am "A nice log message" # Adapter le message en conséquence.
```

De plus, Git permet de partager son travail en créant des répertoires publics. Pour les besoins du chargé de TP, vous allez créer un tel répertoire et le maintenir à jour avec votre répertoire local de travail.

```
1  ## Create the public repository
2  cd ~/public_html
3  mkdir gits
4  git clone --bare pathtoworkdir gits/GLLabs.git
5  ## Tricky handling of refs updates for http
6  cd gits/GLLabs.git/hooks
7  git update-server-info
8  mv post-update.sample post-update
9  chmod 700 post-update
10 ## test
11 cd /tmp
12 git clone  http://www.ecole.ensicaen.fr/~YOURLOGIN/gits/GLLabs.git # Adapt YOURLOGIN
13 ## Verify that /tmp/GLLabs is uptodate
```

Enfin pour maintenir votre répertoire public synchrone avec votre répertoire de travail vous utiliserez la séquence suivante:

```
1  cd pathtoworkdir
2  ## Give a nice nickname to your public repo (only once)
3  git remote add mypublic ~/public_html/gits/GLLabs.git
4  ## Make sure everything was commited
5  git commit -am "A nice log message"
6  ## Push new stuff to the server
7  git push mypublic master:master
8  ## test
9  cd /tmp/GLLabs
10 git pull
11 ## Verify that /tmp/GLLabs is uptodate
```

1.5 Utilisateurs de QTcreator

Pour les utilisateurs de QTcreator, il faut en premier lieu ouvrir le fichier `CMakeLists.txt` situé a la racine de votre dossier de travail. Suivez ensuite les instructions proposées.

De plus pour que les ressources (fichier GLSL, textures) soient bien accessible lors de l'exécution du programme, il est nécessaire de configurer le chemin a partir duquel l'exécutable sera lancer. Ce paramètre est accessible a partir de l'onglet `Projects` (sur le panneau gauche). Choisissez ensuite `Builds&Run/Run/Working directory`.

Attention: certaines versions de QTcreator sont bugguées et nécessitent d'effacer le fichier de configuration CMakeLists.txt.user avant chaque réouverture du projet. Ce fichier sera donc généré à nouveau.

2 Exercice

Pour ce TP, vous devez implementer l'intégralité d'une application openGL+GLFW, permettant dans un premier temps d'afficher un triangle.

Pour ce faire, vous devrez suivre les instructions fournies lors des premières séances de cours. Vous pouvez évidemment utiliser toute documentation en ligne supplémentaire, notamment:

- La documentation de glfw, glm, ...
- Les tutoriaux pointés par la page foad du cours.

D'un point de vue pratique, vous devrez compléter le fichier `myOwnOpenGLProg.cpp` et la paire de fichiers GLSL, `myOwnGLSLProg.v.glsl` et `myOwnGLSLProg.f.glsl`.

3 Compte-rendu du TP

Ce TP vise a vous donner une opportunité d'implémenter une application simple mais **complete**. Il n'y a pas de retour attendu autre que l'expression palpable d'une satisfaction extrême sur vos visages. Merci de laisser paraître votre enthousiasme!