Qt OpenGL - Gestion des extensions avec `QGLContext::getProcAddress()`

Sommaire

- [Introduction à OpenGL et Qt 5.4](#)
- [Support d'OpenGL dans Qt](#)
- [Qt OpenGL - Générer un terrain](#)
- [Qt OpenGL - Envoyer des données au processeur graphique](#)
- [Qt OpenGL - Utilisation du pipeline programmable](#)
- [Qt OpenGL - Ajouter des lumières et des textures](#)
- [Qt OpenGL - Réaliser un rendu offscreen](#)
- [Qt OpenGL - Overpainting : dessiner en 2D avec QPainter sur une scène 3D](#)
- [Qt OpenGL - Gestion des extensions avec `QGLContext::getProcAddress\(\)`](#)
- [Qt OpenGL - Annexes](#)

Les fonctionnalités offertes par les cartes graphiques sont très variables en fonction du constructeur et évoluent à un rythme différent d'OpenGL. Pour gérer cette grande diversité et pour faciliter l'intégration de nouvelles fonctionnalités, OpenGL utilise un système d'extensions : chaque nouvelle fonction est ajoutée dans une extension. Pour pouvoir l'utiliser, il faut donc : vérifier que le matériel supporte la fonctionnalité puis charger la fonction. Le lecteur se reportera au tutoriel Les extensions OpenGL pour avoir des explications détaillées sur la procédure générale à suivre, en particulier sur la syntaxe à utiliser pour déclarer les pointeurs de fonctions.

Il est possible d'utiliser une bibliothèque tierce qui permet de prendre en charge les extensions OpenGL, par exemple GLEW. Dans le cas où l'on souhaite utiliser peu d'extensions et que l'ajout d'une bibliothèque est trop lourd, Qt fournit la fonction `getProcAddress` pour récupérer une extension OpenGL :

```
PFNGLGENBUFFERSARBPROC glGenBuffers = (  
PFNGLGENBUFFERSARBPROC) getAddress("glGenBuffersARB");  
if (glGenBuffers)  
    // la fonction est disponible  
else  
    // la fonction n'est pas disponible
```

OpenGL, Qt