

Utiliser les extensions OpenGL

Christophe [Groove] Riccio

www.g-truc.net

Création : 25 octobre 2003

Révision : 6 octobre 2006

Sommaire

Introduction

1. A propos des extensions
 2. Utilisation des extensions
 - 2.1. Conditions
 - 2.2. Détections des extensions
 - 2.3. Fonctions des extensions
 - 2.4. Constante des extensions
 3. Exemple
- Conclusion

Introduction

Les évolutions d'OpenGL sont réalisées à partir d'extensions avant d'être intégré à la spécification d'OpenGL. Dans ce tutorial, nous allons étudier les extensions en commençant par une explication de ce que sont les extensions OpenGL puis comment il est possible de les utiliser et enfin nous finirons pas un exemple.

L'utilisation des extensions est impérative sous système Windows (4, 5, 6) pour profiter des nouvelles caractéristiques. En effet, Microsoft a arrêté son implémentation à OpenGL 1.1. Pour utiliser la possibilité OpenGL 2.1, il faut donc utiliser OpenGL 1.1 puis charger manuellement des extensions.

Une autre utilisation des extensions consistent à utiliser des possibilités plus avancées d'OpenGL que ce la carte graphique propose. Par exemple, les vertex buffer objects ont été intégrés dans les spécifications d'OpenGL 1.5 hors le nVidia TNT qui n'est pas une carte OpenGL 1.5, support les vertex buffer objects via l'extension GL_ARB_vertex_buffer_object.

1. A propos des extensions

Les extensions OpenGL sont créées par IARB (Architecture Review Board) qui regroupe de grands noms de l'infographie tel que nVidia, ATI, Silicon Graphics, Apple, ...

Pour créer une extension, il n'est pas obligatoire que l'ensemble des acteurs de ARB se mettent d'accords, un fabricant seul peut créer une extension cependant cette extension n'aura pas le même statu qu'une extension approuvé par ARB.

Ainsi on retourne une notion de statu de l'extension au saint de son nom qui prendra toujours la forme suivante : GL_STATU_nom_extension.

Voici quelques exemples de valeur possible pour STATU:

- ARB : Extension approuvé par ARB
- EXT : Extension approuvé par un grand nombre d'acteurs du conseil ARB (entre la 1/2 et 3/4)
- NV : Extension propriétaire de nVidia.
- ATI : Extension propriétaire de ATI.
- APPLE : Extension propriétaire de Apple.
- SGIS : Extension spécialisé de Silicon Graphics.
- HP : Extension propriétaire de HP.

Seule les extensions ARB sont des extensions officielles d'OpenGL. Bien souvent les extensions EXT

finissent par être promu ARB, voir même les extensions propriétaires comme l'extension `GL_NV_point_sprite` qui a été promu `GL_ARB_point_sprite` avec OpenGL 1.5. Les extensions propriétaires peuvent être utilisées par d'autres entreprises que le propriétaire lui-même, ainsi ATI utilise pour ses GPU de la série R300, l'extension `GL_NV_point_sprite`.

2. Utilisation des extensions

2.1. Conditions

Tout d'abord, il faut s'assurer que les pilotes de la carte graphique de votre ordinateur sont bien installés. En effet, les extensions OpenGL utilisent directement le code des fonctions contenu dans la partie OpenGL des pilotes. De plus, pour utiliser les dernières extensions OpenGL, il vous faut la dernière version des pilotes proposés par le constructeur de votre carte graphique. Avoir des pilotes à jour est souvent un moyen de réduire les problèmes notamment sur les fonctionnalités récentes.

2.2. Détections des extensions

Avant d'utiliser une extension, il est préférable de vérifier que l'extension que vous souhaitez utiliser est bien supportée par votre carte graphique.

La liste des extensions est donnée par le paramètre `GL_EXTENSIONS` passé à la fonction `glGetString`.

```
const GLubyte *extensions = glGetString(GL_EXTENSIONS);
```

Les extensions sont séparées par des espaces dans la chaîne de caractère obtenue.

2.3. Fonctions des extensions

Pour récupérer une fonction d'extension, il faut utiliser des fonctions propriétaires du système d'exploitation qui récupèrent l'adresse de la fonction dans la bibliothèque dynamique (dll / so) du pilote OpenGL:

- `wglGetProcAddress` sous Windows

- `glXGetProcAddress` sous X11 (Linux, Irix, Solaris, ...)

Tout système d'exploitation supportant OpenGL dispose de moyens pour utiliser les extensions OpenGL, bien que la procédure puisse être plus complexe, comme sur MacOS X.

Prenons par exemple la fonction `glSecondaryColor3fEXT` de l'extension `GL_EXT_secondary_color`, devant être déclaré ainsi :

```
typedef void (APIENTRY * PFNGLSECONDARYCOLOR3FEXTPROC) (GLfloat red, GLfloat green, GLfloat blue);
```

Déclaration de pointeur sur fonction `glSecondaryColor3fEXT` :

```
PFNGLSECONDARYCOLOR3FEXTPROC glSecondaryColor3fEXT = NULL;
```

Obtention de l'adresse de la fonction `glSecondaryColor3fEXT` :

```
glSecondaryColor3fEXT = (PFNGLSECONDARYCOLOR3FEXTPROC) wglGetProcAddress("glSecondaryColor3fEXT");
```

Utilisation de la fonction `glSecondaryColor3fEXT` :

```
glSecondaryColor3fEXT(1.f, 1.f, 1.f);
```

Cet appel de fonction est relativement équivalent à :

```
glColor3f (1.f, 1.f, 1.f);
```

2.4. Constantes des extensions

Une extension n'équivaut pas à une fonction. Dans une extension, il peut y avoir une ou plusieurs fonctions, voir aucune ainsi qu'un certain nombre de constantes que l'on trouve sous la forme de *#define* en C/C++.

L'extension GL_ARB_texture_env_dot3, intégré dans OpenGL 1.3, fait partie des extensions qui ne dispose que de *#define* :

```
#define GL_DOT3_RGB_ARB          0x86AE
#define GL_DOT3_RGBA_ARB       0x86AF
```

A la question : "D'où sorte ces valeurs ?" il existe une réponse simple :
<http://oss.sgi.com/projects/ogl-sample/registry>.

Cette page référence toutes les extensions OpenGL et indique comment elles doivent être implémentées, ce qui inclut les valeurs des *#define* et les déclarations des fonctions.

Exemple

Certains concepts sont encore un peu flous ? Cet exemple est là pour retirer la brume.

L'exemple a été réalisé avec Visual C++ 6 et utilise GLUT. Il met en oeuvre l'application de l'extension GL_EXT_secondary_color pour colorer un cube. Le rendu est identique à un rendu utilisant *glcolor**, cependant si vous utilisez simultanément ces deux formes de colorisations vous comprendrez le rôle de l'extension GL_EXT_secondary_color.

A télécharger sur <http://www.g-truc.net/articles/ogl-ext.zip>

Conclusion

Pour plus d'informations, je vous invite à consulter la dernière spécification d'OpenGL sur le site www.opengl.org ainsi que la page <http://oss.sgi.com/projects/ogl-sample/registry>

J'espère que vous avez apprécié ce tutorial. Je vous invite à me faire part de vos remarques par email www.g-truc.net/contact.html pour améliorer ce document ou si vous souhaitez un complément d'informations sur ce sujet.