Tutorial 3 : L'illumination

Je repars du 2ème tutorial sur l'initialisation d'OpenGL. Il vous suffit de rajouter une variable globale et de modifier quelques fonctions afin de pouvoir tester l'annimation sous OpenGL avec les matrices de transformation.

```
// variable globale en dessous des includes
int change = 0; // pour changer la forme à afficher
```

Nous allons aussi modifier la fonction d'initialisation pour gérer les lumières :

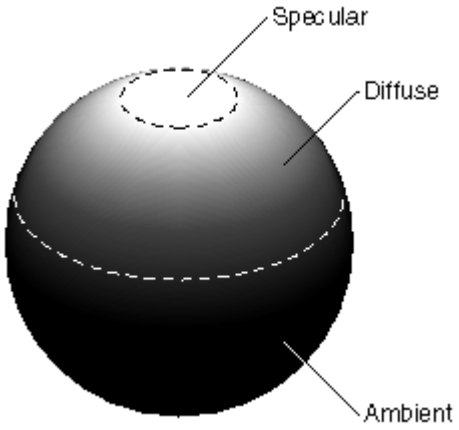
```
glEnable(GL_LIGHTING) ; // active le calcul de l'éclairage
glEnable(GL_LIGHT0) ; // allume la lampe 0

glEnable(GL_NORMALIZE); // normalisation des normales automatiques

float DiffuseProperties[] = {0.9f, 0.9f, 0.9f, 1.0f}; // propriétés diffuses de la lumière (couleur)
float positionProperties[] = {0.0f, 50.0f, 0.0f, 10.0f}; // position de la lumière

// on applique les propriétés à la lumière souhaitée
glLightfv( GL_LIGHT0, GL_DIFFUSE, DiffuseProperties); // GL_AMBIENT, GL_DIFFUSE,
GL_SPECULAR GL_EMISSION GL_SHININESS
glLightfv( GL_LIGHT0, GL_POSITION, positionProperties); // , GL_POSITION,
GL_SPOT_DIRECTION, GL_SPOT_EXPONENT, GL_SPOT_CUTOFF
```

Différences entre les modes de lumière :



Les lumières d'OpenGL sont toujours sous la forme de GL_LIGHTi ou 'i' va de 0 à 7 (normalement). Mais si vous voulez connaître le nombre de lumières que votre carte graphique peut afficher, vous avez une fonction de GLUT qui vous le donne.

```
glGetIntegerv(GL_MAX_LIGHTS, &nb_lights); // donne le nb de lumière dans nb_lights
```

Maintenant, on va modifier la fonction qui gère les touches spéciales du clavier. Ceci afin de déterminer la forme à afficher.

```
void GestionClavierSpe(int key, int x, int y) {
switch(key)
{
//touches spéciales clavier
case GLUT_KEY_UP:
change = 0;
break;
case GLUT_KEY_DOWN:
change = 1;
break;
case GLUT_KEY_RIGHT:
change = 2;
break;
case GLUT_KEY_LEFT:
change = 3;
break;
}
}
```

Enfin, nous allons modifier la fonction d'affichage :

```
void myDisplay() {

// vide le Z-Buffer et le Buffer de couleur
glClear(GL_DEPTH_BUFFER_BIT | GL_COLOR_BUFFER_BIT);

static unsigned long tps = GetTickCount();

static float variation = 0.0f; // variable qui va faire varier la transformation
variation+=(GetTickCount()-tps)*100.0f/1000.0f; // variation de 100.0f en 1sec
tps=GetTickCount(); // re-init du temps

glPushMatrix(); // Empile la matrice de transformation
glTranslatef(0.0f, // translation sur l'axe X
0.0f, // translation sur l'axe Y
10.0f); // translation sur l'axe Z
glRotatef(variation,
1.0f, // rotation sur l'axe X
1.0f, // rotation sur l'axe Y
1.0f); // rotation sur l'axe Z

if (change == 0)
{
glBegin(GL_TRIANGLES); // mode de polygones à afficher
glNormal3d(0.0,0.0,-1.0); // normale de la face
// couleurs et points des vertexes à afficher
glColor4f(1.0f,1.0f,1.0f,1.0f); glVertex3f(0.0f, 1.0f, 0.0f);
glColor4f(1.0f,1.0f,1.0f,1.0f); glVertex3f(-1.0f, -1.0f, 0.0f);
glColor4f(1.0f,1.0f,1.0f,1.0f); glVertex3f(1.0f, -1.0f, 0.0f);
```

```
glEnd();
}
if (change == 1)
glutSolidSphere(1.5f,30.0f,30.0f); // affiche une sphère
if (change == 2)
glutSolidCube(1.5f); // affiche un cube
if (change == 3)
glutSolidTorus(0.5f, 1.5f, 30.0f, 30.0f); // affiche un anneau
glPopMatrix(); // Dépile la matrice de transformation

glutSwapBuffers(); // permute la surface primaire avec la secondaire
}
```

Pour que les lumières fassent leurs effets sur les objets de la scène, ceux-ci doivent avoir des normales. Car les rayons lumineux se réfléchissent par rapport aux normales des objets.
De plus, ici, je vous montre des formes de base que GLUT contient. La sphère, le torus ou le cube, vous les trouverez dans glut.h.