
# Chapitre3: Les bases d'OpenGL

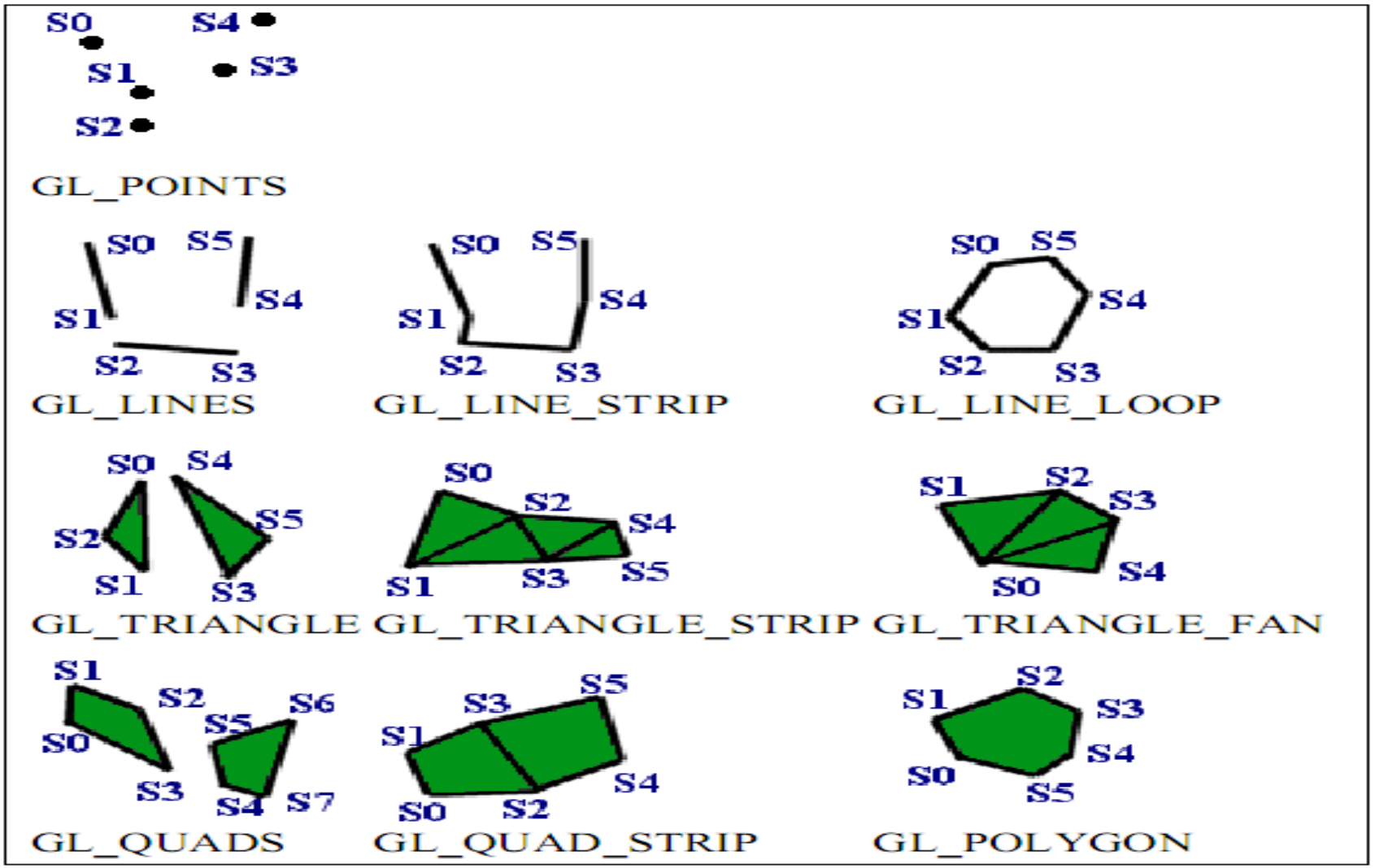
# Primitive de tracé opengl

- GL_POINTS Trace un point pour chacun des n vecteurs.
- GL_LINES Trace une série de lignes non connectées. Les segments sont tracés entre v0 et v1, v2 et v3,...etc. Si n est impair vn-1 est ignoré.
- GL_POLYGON Trace un polygone avec v0, v1,..,vn-1 comme vecteurs. n doit être au moins égal à 3 ou rien n'est dessiné. De plus, le polygone ne peut se croiser lui et doit être convexe à cause de limitations (dues aux algorithmes).
- GL_TRIANGLES Trace une série de triangles avec les vecteurs v0, v1 et v2, puis v3, v4 et v5, etc. Si n n'est pas un multiple de 3, les points restants sont ignorés.
- GL_LINE_STRIP Trace une ligne de v0 à v1, puis de v1 à v2 et ainsi de suite. Finalement de vn-2 à vn-1 pour un total de n-1 segments. Il n'y a pas de restrictions sur les vecteurs décrivant les tronçons de lignes, les lignes peuvent arbitrairement se croiser.

# Primitive de tracé opengl

- GL_LINE_LOOP Identique à GL_LINE_STRIP excepté qu'un segment final est dessiné de vn-1 à v0, pour fermer la boucle.
- GL_QUADS Dessine une série de quadrilatères avec les vecteurs v0, v1, v2, v3 et v4, v5, v6, v7 et ainsi de suite.
- GL_QUAD_STRIP Dessine une série de quadrilatères avec les vecteurs v0, v1, v3, v2 puis v2, v3, v5, v4 et ainsi de suite.
- GL_TRIANGLE_STRIP Dessine une série de triangles avec les vecteurs v0, v1, v2, puis v2,v1, v3, puis v2, v3, v4, etc. L'ordre permet de s'assurer que les triangles ont l'orientation correcte et l'a bande peut être utilisée pour former une partie de la surface.
- GL_TRIANGLE_FAN Similaire à GL_TRIANGLE_STRIP excepté que les triangles sont v0,v1, v2, puis v0, v2, v3, puis v0, v3, v4, et ainsi de suite. . Tous les triangles ont v0 en commun.

# Résumé


S0
S4
S1
S3
S2
GL_POINTS
S0
S5
S4
S1
S2
S3
GL_LINES
S0
S5
S4
S1
S2
S3
GL_LINE_STRIP
S0
S5
S4
S1
S2
S3
GL_LINE_LOOP
S0
S4
S5
S2
S1
S3
GL_TRIANGLE
S0
S2
S4
S1
S3
S5
GL_TRIANGLE_STRIP
S1
S2
S3
S4
S0
GL_TRIANGLE_FAN
S1
S2
S6
S0
S5
S3
S4
S7
GL_QUADS
S3
S5
S1
S4
S0
S2
GL_QUAD_STRIP
S2
S1
S3
S4
S0
S5
GL_POLYGON

# Exemples

**Example #1: Basic Glut Library**

```
#include <stdio.h>
#include <GL/glut.h>

void Display(void) { }

int main(void) {

// Create main Window
glutCreateWindow("This is the window title");

// Set Call Back Window
glutDisplayFunc(Display);

// Window Message Loop
glutMainLoop();

return 0;
}
```

# Exemples

**Example #2: Clear black color**

```
#include <stdio.h>
#include <GL/glut.h>

void Display(void)
        {
        glClear(GL_COLOR_BUFFER_BIT);
        glFlush();
        }

int main(void)
        {
        // Create main Window
        glutCreateWindow("This is the window title");

        // Set Call Back Window
        glutDisplayFunc(Display);

        // Window Message Loop
        glutMainLoop();
        return 0;
        }
```

# Exemples

**Example #3: Change Clear Color to Red**

```
#include <stdio.h>
#include <GL/glut.h>

void Display(void)
        {
        glClear(GL_COLOR_BUFFER_BIT);
        glFlush();
        }

int main(void)
        {
        glutCreateWindow("This is the window title");
        glutDisplayFunc(Display);
        glClearColor(0,0,1,0);
        glutMainLoop();
        return 0;
        }
```

# Exemples

**Example #4: Set a white point at origin**

```
#include <stdio.h>
#include <GL/glut.h>
void Display(void)
        {
        glClear(GL_COLOR_BUFFER_BIT);
        glBegin(GL_POINTS);
        glColor3f(1.0f,1.0f,1.0f);
    glVertex3f(0,0,0);
        glEnd();

        glFlush();
        }

int main(void)
        {
        glutCreateWindow("This is the window title");
        glutDisplayFunc(Display);
        //glPointSize(10);
        glutMainLoop();
        return 0;
}
```

# Exemples

**Example #5: Set points at corners of screen**

```
#include <stdio.h>
#include <GL/glut.h>
void Display(void)
        {
        glClear(GL_COLOR_BUFFER_BIT);

        glBegin(GL_POINTS);
                glVertex3f(0,0,0);
                glVertex3f(1,1,0);
                glVertex3f(-1,1,0);
                glVertex3f(1,-1,0);
                glVertex3f(-1,-1,0);
        glEnd();
        glFlush();
        }
int main(void)
        {
        glutCreateWindow("This is the window title");
        glutDisplayFunc(Display);
        // Set Point Size to 10 from default 1
        glPointSize(10);
        glutMainLoop();
        return 0;
        }
```

# Exemples

## Example #6: Draw 10 Circle of points

```
#include <stdio.h>
#include <math.h>
#include <GL/glut.h>
void Display(void)
                    {
                    glClear(GL_COLOR_BUFFER_BIT);
                    glBegin(GL_POINTS);
               glVertex3f(cos(2*3.14159*0/1000.0),sin(2*3.14159*500/0.0),0);
                    glVertex3f(cos(2*3.14159*100/1000.0),sin(2*3.14159*100/1000.0),0);
                    glVertex3f(cos(2*3.14159*200/1000.0),sin(2*3.14159*200/1000.0),0);
                    glVertex3f(cos(2*3.14159*300/1000.0),sin(2*3.14159*300/1000.0),0);
               glVertex3f(cos(2*3.14159*400/1000.0),sin(2*3.14159*400/1000.0),0);
               glVertex3f(cos(2*3.14159*500/1000.0),sin(2*3.14159*500/1000.0),0);
                    glVertex3f(cos(2*3.14159*600/1000.0),sin(2*3.14159*600/1000.0),0);
                    glVertex3f(cos(2*3.14159*700/1000.0),sin(2*3.14159*700/1000.0),0);
                    glVertex3f(cos(2*3.14159*800/1000.0),sin(2*3.14159*800/1000.0),0);
               glVertex3f(cos(2*3.14159*900/1000.0),sin(2*3.14159*900/1000.0),0);
               glVertex3f(cos(2*3.14159*1000/1000.0),sin(2*3.14159*1000/1000.0),0);
                    glEnd();

                    glFlush();
                    }
int main(void)
                    {
                    glutCreateWindow("This is the window title");
                    glutDisplayFunc(Display);
    glPointSize(4);
                    glutMainLoop();
                    return 0;
                    }
```

# Exemples

## Example #7: Draw a Line

```
#include <stdio.h>
#include <math.h>
#include <GL/glut.h>

void Display(void)
        {
        glClear(GL_COLOR_BUFFER_BIT);

        glBegin(GL_LINES);
        glVertex3f(-1,0,0);
        glVertex3f(+1,0,0);
        glEnd();

        glFlush();
        }

int main(void)
        {
        glutCreateWindow("This is the window title");
        glutDisplayFunc(Display);
        glutMainLoop();
        return 0;
        }
```

# Exemples

## Example #8: Draw Line Strip

```
#include <stdio.h>
#include <GL/glut.h>

void Display(void)
        {
        glClear(GL_COLOR_BUFFER_BIT);

        glBegin(GL_LINE_STRIP);
        glVertex3f(-1,0,0);
        glVertex3f(0,0,0);
        glVertex3f(0,1,0);
        glVertex3f(1,0,0);
        glEnd();

        glFlush();
        }

int main(void)
        {
        glutCreateWindow("This is the window title");
        glutDisplayFunc(Display);
        glutMainLoop();
        return 0;
        }
```

# Exemples

## Example #9: Draw Line loop

```
#include <stdio.h>
#include <GL/glut.h>

void Display(void)
        {
        glClear(GL_COLOR_BUFFER_BIT);

        glBegin(GL_LINE_LOOP);
        glVertex3f(-1,0,0);
        glVertex3f(0,0,0);
        glVertex3f(0,1,0);
        glEnd();

        glFlush();
        }

int main(void)
        {
        glutCreateWindow("This is the window title");
        glutDisplayFunc(Display);
        glutMainLoop();
        return 0;
        }
```

# Exemples

## Example #10: Draw a triangle Surface

```
#include <stdio.h>
#include <GL/glut.h>
void Display(void)
        {
        glClear(GL_COLOR_BUFFER_BIT);
        glBegin(GL_TRIANGLES);
                glColor3f(1,1,1);
                glVertex3f(0,0,0);
                glVertex3f(-1,0,0);
                glVertex3f(0,1,0);

                glColor3f(1,0,0);
                glVertex3f(0,0,0);
                glVertex3f(1,0,0);
                glVertex3f(0,-1,0);
        glEnd();
        glFlush();
        }
int main(void)
        {
        glutCreateWindow("This is the window title");
        glutDisplayFunc(Display);
        glutMainLoop();
        return 0;}
```

# Exemples

## Example #11: Draw a triangle strip

```
#include <stdio.h>
#include <math.h>
#include <GL/glut.h>

void Display(void)
        {
        glClear(GL_COLOR_BUFFER_BIT);
        glBegin(GL_TRIANGLE_STRIP);
        glVertex3f(0,0,0);
        glVertex3f(1,0,0);
        glVertex3f(0,1,0);
        glVertex3f(0.9,0.9,0);
        glVertex3f(1,-1,0);
        glEnd();

        glFlush();
        }
int main(void)
        {
        glutCreateWindow("This is the window title");
        glutDisplayFunc(Display);
        glutMainLoop();
        return 0;
        }
```

# Exemples

## Example #12: Draw a triangle fan

```
#include <stdio.h>
#include <math.h>
#include <GL/glut.h>

void Display(void)
        {
        glClear(GL_COLOR_BUFFER_BIT);
        glBegin(GL_TRIANGLE_FAN);
        glVertex3f(0,0,0);
        glVertex3f(1,0,0);
        glVertex3f(0,1,0);
        glVertex3f(-1,0,0);
        glVertex3f(-1,-1,0);
        glEnd();

        glFlush();
        }
int main(void)
        {
        glutCreateWindow("This is the window title");
        glutDisplayFunc(Display);
        glutMainLoop();
        return 0;
        }
```

# Exemples

**Example #13: Draw a polygon**

```
#include <stdio.h>
#include <math.h>
#include <GL/glut.h>

void Display(void)
        {
        glClear(GL_COLOR_BUFFER_BIT);
        glBegin(GL_POLYGON);
        glVertex3f(0,0,0);
        glVertex3f(1,0,0);
        glVertex3f(0,1,0);
        glVertex3f(-1,1/2.0,0);
        glVertex3f(0,-1,0);
        glEnd();

        glFlush();
        }
int main(void)
        {
        glutCreateWindow("This is the window title");
        glutDisplayFunc(Display);
        glutMainLoop();
        return 0;
        }
```

# Exemples

## Example #14: Draw a Quad Strip

```
#include <stdio.h>
#include <math.h>
#include <GL/glut.h>

void Display(void)
        {
        glClear(GL_COLOR_BUFFER_BIT);
        glBegin(GL_QUAD_STRIP);
        glVertex3f(0,1,0);
        glVertex3f(0,0,0);
        glVertex3f(0.1,1,0);
        glVertex3f(0.1,0,0);
        glVertex3f(0.2,0.9,0);
        glVertex3f(0.2,-0.1,0);
        glEnd();
        glFlush();
        }
int main(void)
        {
        glutCreateWindow("This is the window title");
        glutDisplayFunc(Display);
        glutMainLoop();
        return 0;
        }
```

# Gestion des événements clavier

**void glutKeyboardFunc**
**(void (*fonct) (unsigned char key,int x,int y))**

Définit la fonction exécutée automatiquement par GLUT lorsqu'une touche ASCII du clavier est frappée. Au moment de son exécution, cette fonction recevra les paramètres key, x et y contenant respectivement le code ASCII de la touche de clavier et les positions instantanées en x et en y de la souris relativement à la fenêtre au moment de la frappe clavier.

TP : Clavier.c

# Gestion des événements clavier

- **void glutSpecialFunc(void (*fonct)(int key,int x,int y))**
  - Définit la fonction exécutée automatiquement par GLUT lorsqu'une touche de fonction ou une touche de curseur du clavier est frappée.
  - Au moment de son exécution, cette fonction recevra les paramètres key, x et y contenant respectivement le code de la touche de clavier (**GLUT_KEY_F1** à **GLUT_KEY_F12**, **GLUT_KEY_LEFT**, **GLUT_KEY_RIGHT**, **GLUT_KEY_UP**, **GLUT_KEY_DOWN**, **GLUT_KEY_PAGE_DOWN**, **GLUT_KEY_PAGE_UP**, **GLUT_KEY_HOME**, **GLUT_KEY_END**, **GLUT_KEY_INSERT**) et les positions instantanées en x et en y de la souris relativement à la fenêtre.

# Gestion des événements souris

- **void glutMouseFunc(void (*fonct)(int bouton,int etat,int x,int y))**

- La fonction glutMouseFunc établit la fonction de rappel de la souris pour la fenêtre courante.
- Lorsqu'un utilisateur appuie ou relâche un des boutons de la souris, chaque appui ou relâchement du bouton engendre un appel à la fonction de rappel de la souris.

- Le paramètre boutton peut prendre les valeurs : GLUT_LEFT_BUTTON ou GLUT_RIGHT_BUTTON.

# Gestion des événements souris

- **void glutMouseFunc(void (*fonct)(int bouton,int etat,int x,int y))**

- 'bouton' contient le nom du bouton qui a été pressé ou relâché (GLUT_LEFT_BUTTON, GLUT_RIGHT_BUTTON),
- 'etat' indique si le bouton a été pressé (GLUT_DOWN) ou relâché (GLUT_UP) et
- 'x' et 'y' contiennent les coordonnées de la souris au moment de l'événement.

- TP : Souris.c

# Gestion des événements souris

- **void glutMotionFunc(void (*fonct)(int x,int y));**

- Définit la fonction exécutée automatiquement par GLUT quand la souris se déplace devant la fenêtre avec un bouton appuyé.

  x et y sont les positions de la souris dans la fenêtre au moment de l'appel de la fonction.

- **void glutPassiveMotionFunc(void (*fonct)(int x,int y));**

- Définit la fonction exécutée automatiquement par GLUT quand la souris se déplace devant la fenêtre sans bouton appuyé. x et y sont les positions de la souris dans la fenêtre au moment de l'appel de la fonction.

# Fonctions utiles

- **void glutPostRedisplay(void);**
  - permet d'envoyer à la boucle principale un événement réclamant le réaffichage de la fenêtre.
- **void glTranslatef(float x,float y,float z) : c**ette fonction (dont les paramètres sont de type 'double') multiplie (à droite) la matrice active par une matrice de translation de vecteur (x,y,z).
- la translation est précédé par glLoadIdentity();