

```

#include <GL/glut.h>
#include <stdlib.h>

#define MUNDO 3.0f

void display (void);
void reshape (int width, int height);
void inicializar (void);
void tecladoEspecial (int key, int cx, int cy);
void teclado (unsigned char key, int cx, int cy);

float rotaX, rotaY, zoom;
int proyeccion;

int main (int argc, char **argv)
{
    glutInitDisplayMode(GLUT_RGBA | GLUT_DOUBLE | GLUT_DEPTH);
    //glutInitDisplayMode(GLUT_RGBA | GLUT_DOUBLE);

    glutInitWindowPosition(50, 50);
    glutInitWindowSize(500.0, 500.0);
    glutInit(&argc, argv);
    glutCreateWindow("Carita que gira");
    inicializar();

    //Registro de funciones de Callback
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glutSpecialFunc(tecladoEspecial);
    glutKeyboardFunc(teclado);
    glutMainLoop();

    return 0;
}

//===== display =====

void display (void)
{
    static GLfloat rojo[] = {1.0, 0.0, 0.0};
    static GLfloat verde_oscuro[] = {0.0, 0.5, 0.0};
    static GLfloat carne[] = {0.8, 0.5, 0.5};

    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    //glClear(GL_COLOR_BUFFER_BIT);

    glPushMatrix();
    glTranslatef(0.0, 0.0, -4.0);

    glPushMatrix();
    /*Ajuste de la camara */
    glTranslatef(0.0, 0.0, zoom);
    glRotatef(rotaX, 1.0, 0.0, 0.0);
    glRotatef(rotaY, 0.0, 1.0, 0.0);

    /*cara*/
    glPushMatrix();
    glScalef(0.75, 1.0, 0.75);
    glColor3fv(carne);
    glutSolidSphere(1.0, 50, 50);
    glPopMatrix();

    /*sombbrero*/
    glPushMatrix();
    glTranslatef(0.0, 0.5, 0.0);
    glRotatef(-90.0, 1.0, 0.0, 0.0);
    glColor3fv(verde_oscuro);
    glutSolidCone(0.75, 1.0, 10, 10);
    glPopMatrix();

    /*ojos*/
    glPushMatrix();

```

```

        glPointSize(5.0); //Tamaño del punto en pixeles
        glColor3f(0.0, 0.0, 0.0);
        glBegin(GL_POINTS);
            glVertex3f(0.2, 0.1, 0.75);
            glVertex3f(-0.2, 0.1, 0.75);
        glEnd();
        glPopMatrix();

        /*nariz*/
        glPushMatrix();
            glBegin(GL_TRIANGLES);
                glVertex3f(0.0, 0.0, 0.75);
                glVertex3f(-0.1, -0.3, 0.75);
                glVertex3f(0.1, -0.3, 0.75);
            glEnd();
        glPopMatrix();

        /*boca*/
        glPushMatrix();
            glColor3fv(rojo);
            glLineWidth(3); //Grueso de la linea en pixeles
            glBegin(GL_LINE_STRIP);
                glVertex3f(-0.3, -0.3, 0.65);
                glVertex3f(-0.2, -0.4, 0.69);
                glVertex3f(0.2, -0.4, 0.69);
                glVertex3f(0.3, -0.3, 0.65);
            glEnd();
        glPopMatrix();

        glPopMatrix(); //La de la camara

        glPopMatrix(); //La traslacion en -z

    glutSwapBuffers();
}

//===== Reshape =====
void reshape (int ancho, int alto)
{
    float aspectRadio;

    glViewport(0, 0, ancho, alto);

    if (alto == 0) {
        alto = 1;
    }
    aspectRadio = (float) ancho / (float) alto;

    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();

    gluPerspective(45, aspectRadio, 2.0, 8.0);
    /*
    if (ancho <= alto) {
        glOrtho(-MUNDO, MUNDO, -MUNDO / aspectRadio, MUNDO / aspectRadio, -MUNDO, MUNDO);
    } else {
        glOrtho(-MUNDO * aspectRadio, MUNDO * aspectRadio, -MUNDO, MUNDO, -MUNDO, MUNDO);
    }
    */

    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
}

//===== Inicializa =====
void inicializar (void) {
    glClearColor(0.0, 0.0, 0.0, 0.0);
    rotaX = 0;
    rotaY = 0;
    zoom = 0;
    proyeccion = 1;
}

```

```

        zoom = 0.0f;
        //Prendemos el buffer de profundidad
        glEnable(GL_DEPTH_TEST);
    }

//===== tecladoEspecial =====
void tecladoEspecial(int key, int cx, int cy) {

    switch(key) {

        case GLUT_KEY_UP:
            rotaX -= 10.0;
            break;

        case GLUT_KEY_DOWN:
            rotaX += 10.0;
            break;

        case GLUT_KEY_LEFT:
            rotaY -= 10.0;
            break;

        case GLUT_KEY_RIGHT:
            rotaY += 10.0;
            break;

        case GLUT_KEY_PAGE_UP:
            zoom += 0.1f;
            break;

        case GLUT_KEY_PAGE_DOWN:
            zoom -= 0.1f;
            break;
    }
    glutPostRedisplay();
}

//===== teclado =====
void teclado (unsigned char key, int cx, int cy) {

    switch (key) {
        case 27:
            exit(0);
            break;

        default: break;
    }

    glutPostRedisplay();
}

```