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

/* For manual pages : sudo apt-get install opengl-4-man-doc

   glClearColor : man 3 glClearColor

   glClearColor - specify clear values for the color buffers

   void glClearColor(GLfloat red, GLfloat green, GLfloat blue, GLfloat alpha);

   red, green, blue, alpha : Specify the red, green, blue, and alpha values used when the color
buffers are cleared. The initial values are all 0

  glClearColor specifies the red, green, blue, and alpha values used by glClear() to clear the c
olor buffers
  Values specified by glClearColor are clamped to the range 0 1 */

/* glClear - clear buffers to preset values

   void glClear(GLbitfield mask);

     mask : Bitwise OR of masks that indicate the buffers to be cleared

     Three masks are GL_COLOR_BUFFER_BIT, GL_DEPTH_BUFFER_BIT, and GL_STE
NCIL_BUFFER_BIT.

   glClear sets the bitplane area of the window to values previously selected by glClearColor,
glClearDepth, and glClearStencil

   glClear takes a single argument that is the bitwise OR of several values indicating which bu
ffer is to be cleared

    The values are as follows:
      GL_COLOR_BUFFER_BIT   : Indicates the buffers currently enabled for color writing

      GL_DEPTH_BUFFER_BIT   : Indicates the depth buffer

      GL_STENCIL_BUFFER_BIT : Indicates the stencil buffer

   The value to which each buffer is cleared depends on the setting of the clear value for that b
uffer */

/* glFlush - force execution of GL commands in finite time

     void glFlush(void);

   Different GL implementations buffer commands in several different locations, including ne
twork buffers and the graphics accelerator itself

   glFlush empties all of these buffers, causing all issued commands to be executed as quickl
y as they are accepted by the actual rendering engine
```

```
   Though this execution may not be completed in any particular time period, it does complet
e in finite time

   Because any GL program might be executed over a network, or on an accelerator that buffe
rs commands, all programs should call glFlush whenever they count on having all of their pre
viously issued commands completed

   For example, call glFlush before waiting for user input that depends on the generated image

   glFlush can return at any time, it does not wait until the execution of all previously issued
GL commands is complete */

 void draw( void )  //Drawing funciton
 {
 //glClearColor(R,G,B
   glClearColor(0,1,0,1); //Background color
   glClear(GL_COLOR_BUFFER_BIT );
   glFlush(); //Draw order
 }

 int main(int argc, char **argv) //Main program
 {
   glutInit(&argc, argv);
   glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB ); //Simple buffer
   glutInitWindowSize (512, 384);
   glutInitWindowPosition (150, 150);
   glutCreateWindow ("Example OpenGL Window : Green Window");
   glutDisplayFunc(draw);//Call to the drawing function
   glutMainLoop();
   return 0;
 }

/* compile as :
   gcc -o GreenWindow GreenWindow.c -lglut -lGLU -lGL
   or
   g++ -o GreenWindow GreenWindow.c -lglut -lGLU -lGL

   Run as :
   ./GreenWindow
*/
```