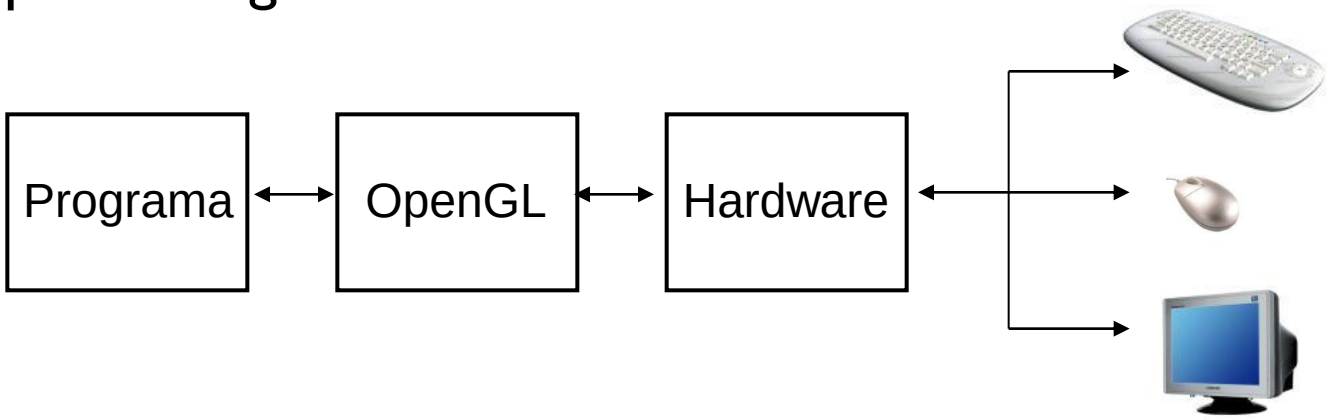
Introdução à Programação em OpenGL

Prof. Márcio Bueno
{cgtarde,cgnoite}@marciobueno.com

OPENGL (*Open Graphical Library*)

- OpenGL é uma interface de software (API – *Aplication Program Interface*) para aceleração da programação de dispositivos gráficos



- Extremamente portável e rápida, possibilitando a criação de imagens com excelente qualidade
- O gerenciamento de janelas e a interação com o usuário devem ser implementadas para cada ambiente

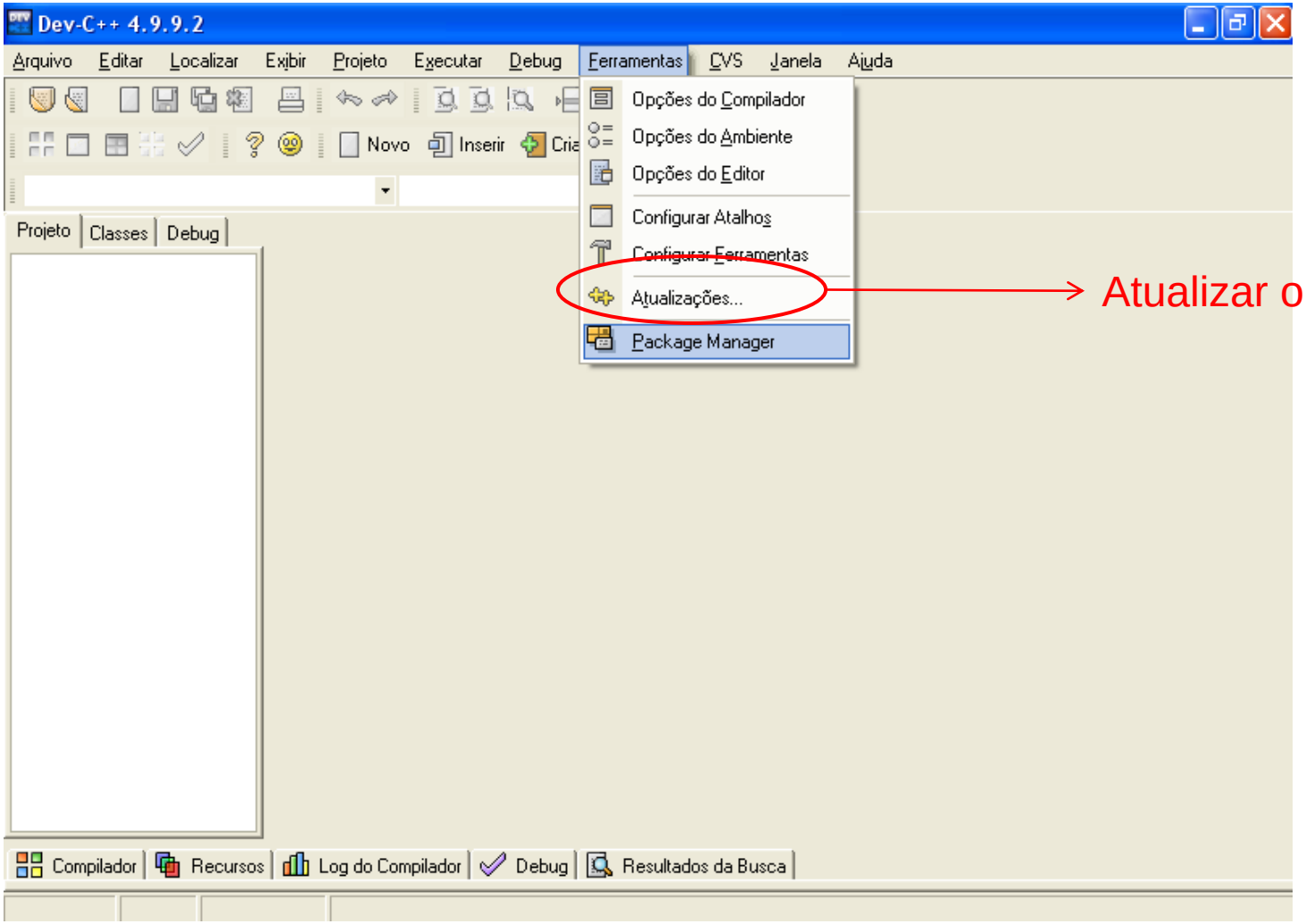
OPENGL com Windows

- Os comandos em OpenGL são disponibilizados através das DLLs (*Dynamic Link Library*) e seus respectivos arquivos *header* e *library*
  - opengl32.dll: gl.h e opengl32.lib
  - glu32.dll: glu.h e glu32.lib
  - glaux.dll: glaux.h e glaux.lib
  - glut.dll: glut.h e glut.lib
- Os arquivos .h são incluídos em código fonte enquanto os arquivos .lib devem ser incluídos no projeto

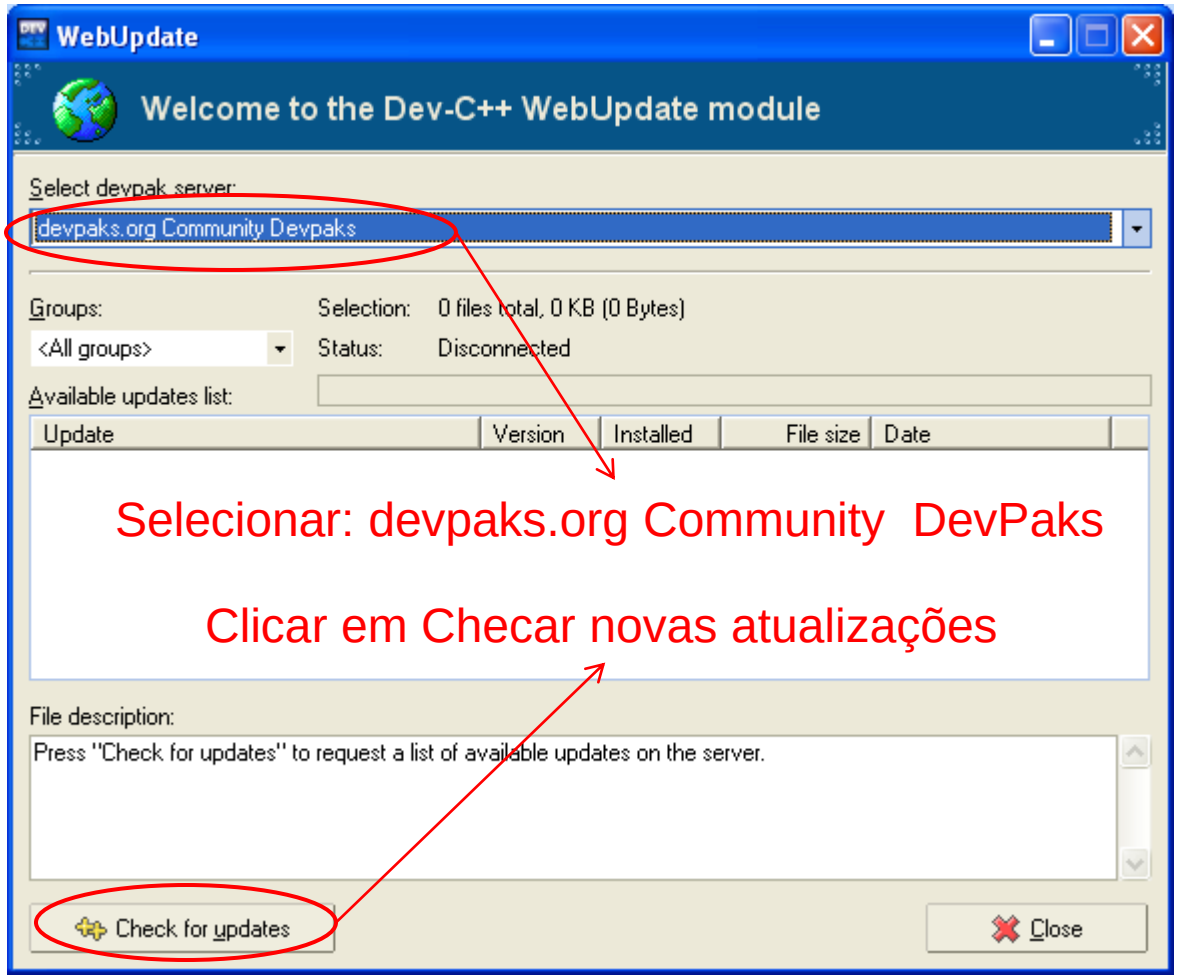
OPENGL com Windows

- Componentes Oficiais
  - GL: contém as funções padrão do OpenGL definidas pela *OpenGL Architeture Review Board* e são caracterizadas pelo prefixo **gl**
  - GLU: A biblioteca de utilitários contém contém funções com o prefixo **glu** para desenho de esferas, cubos, discos, cilindros etc.
- Componentes não Oficiais
  - GLUT: conjunto de ferramentas utilitárias do OpenGL (*OpenGL Utility Toolkit*)
  - Um sistema de gerenciamento de janelas. Todas as suas rotinas começam com o prefixo **glut**

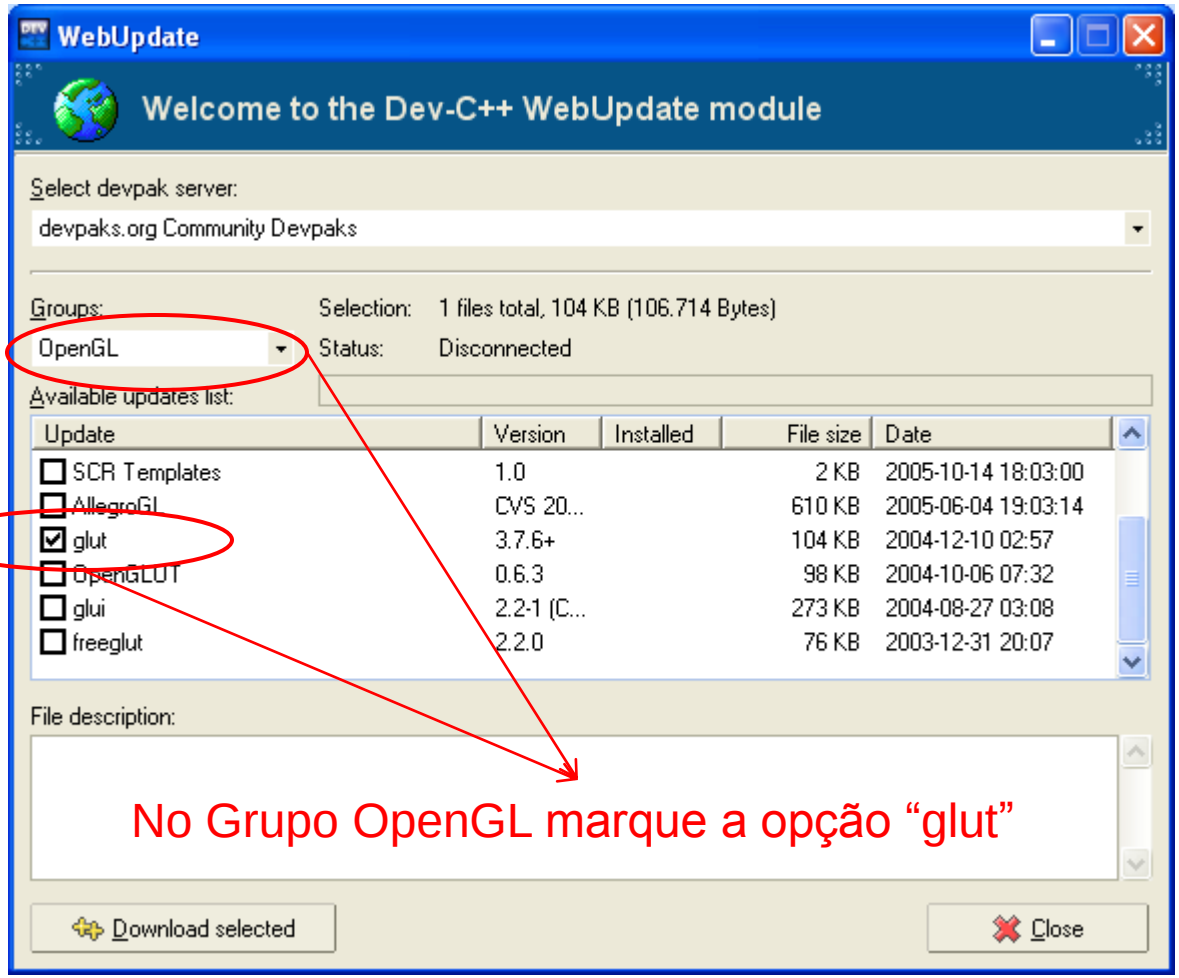
Configurando o Dev-C++ para OpenGL



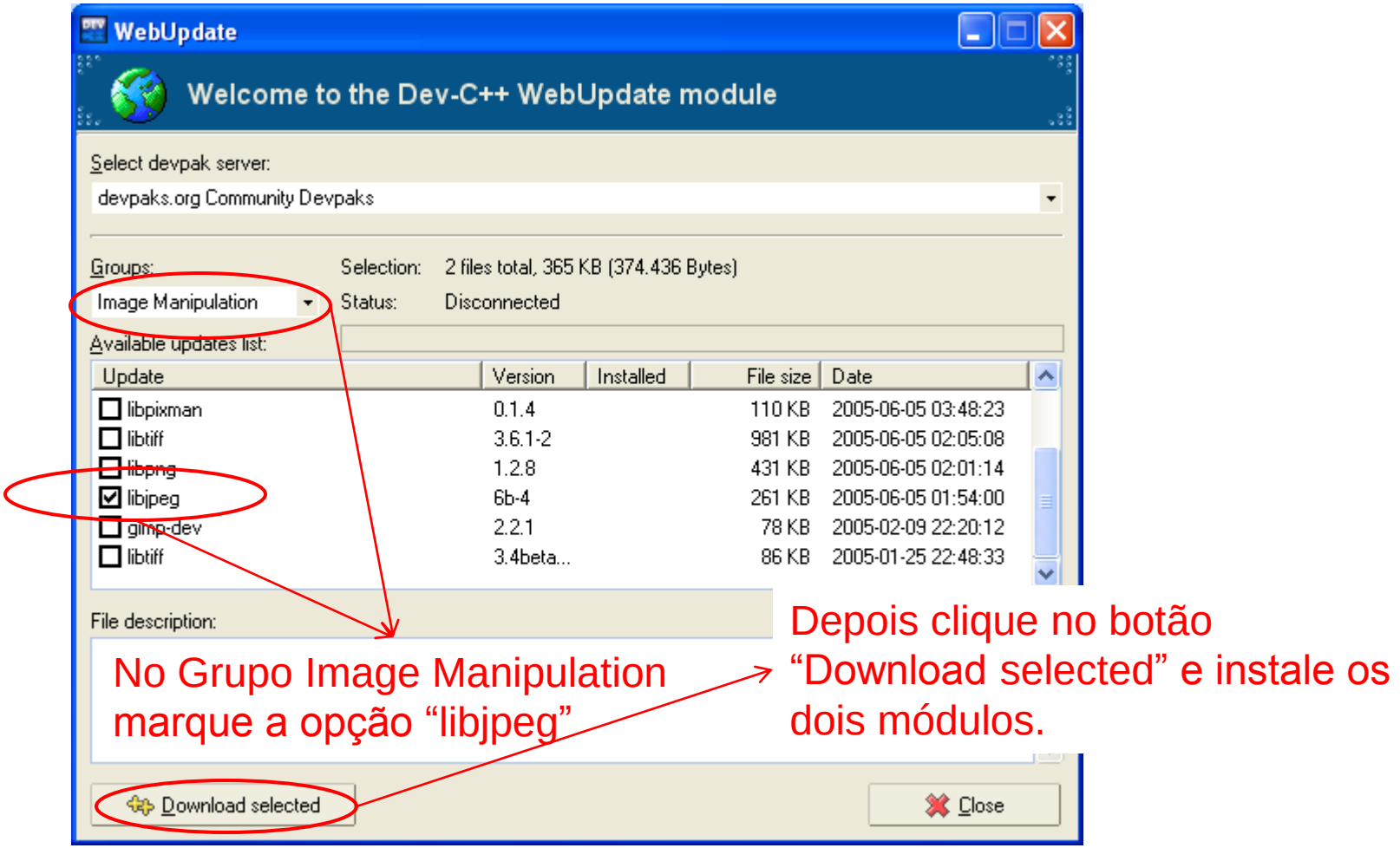
Configurando o Dev-C++ para OpenGL



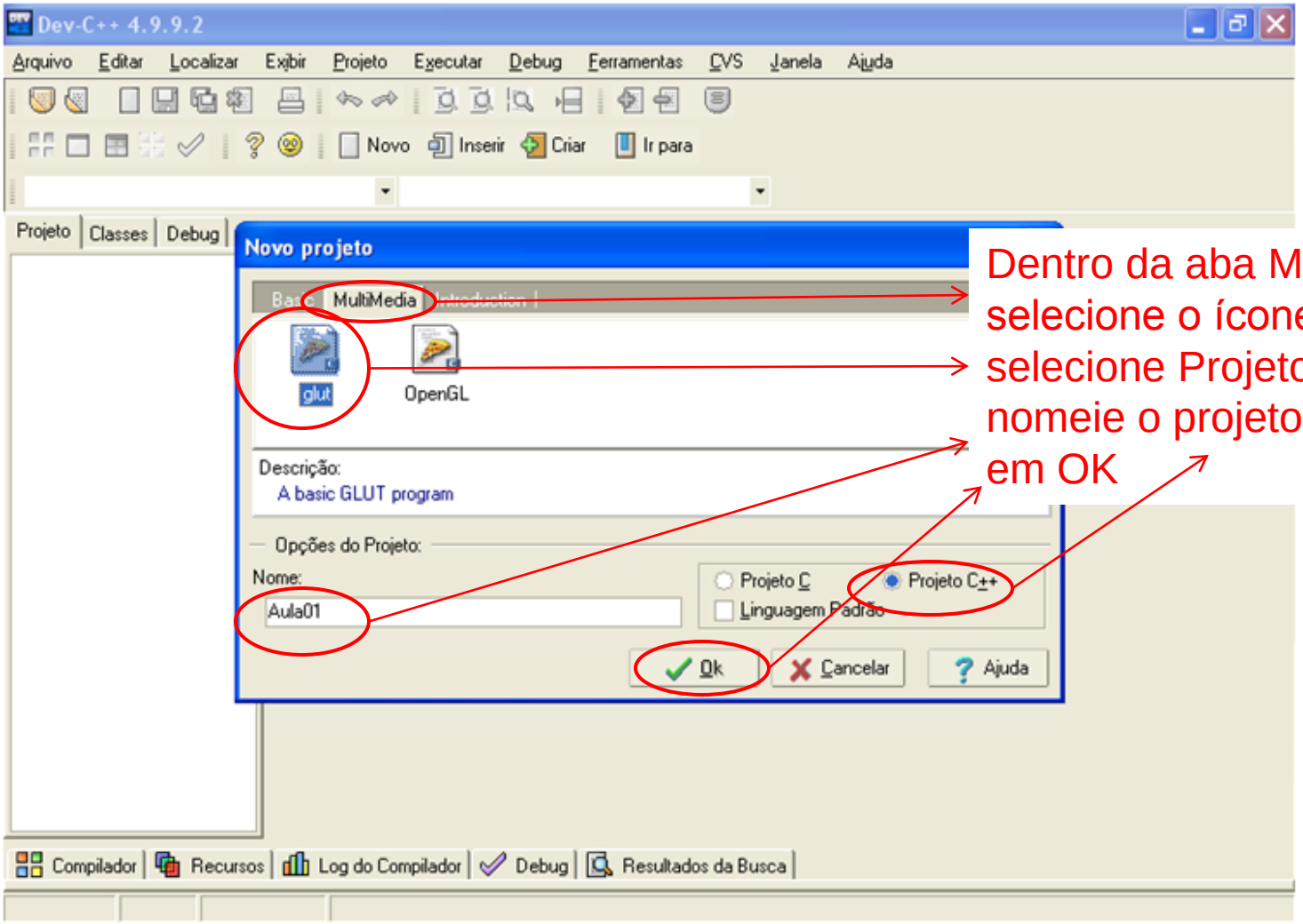
Configurando o Dev-C++ para OpenGL



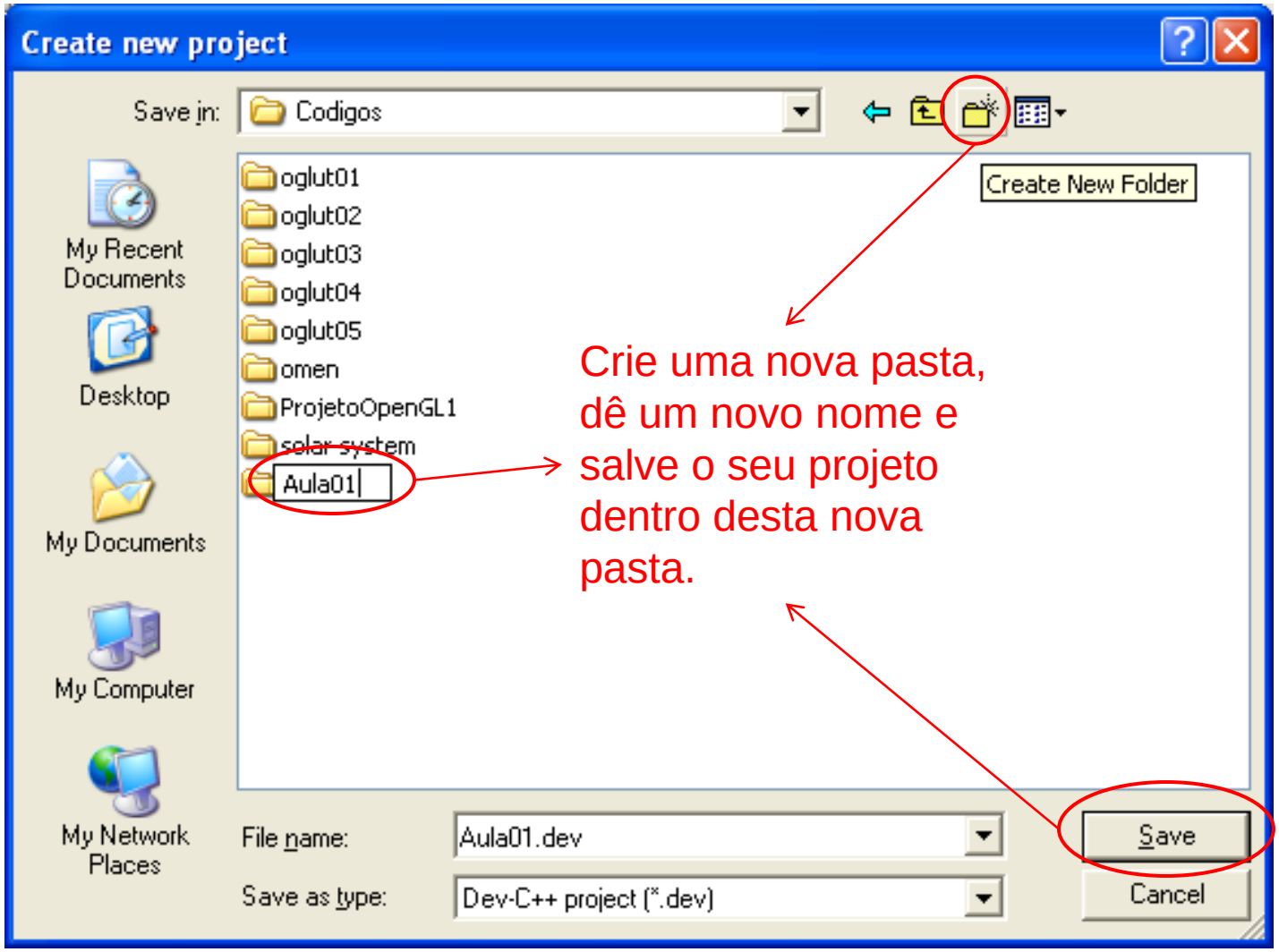
Configurando o Dev-C++ para OpenGL



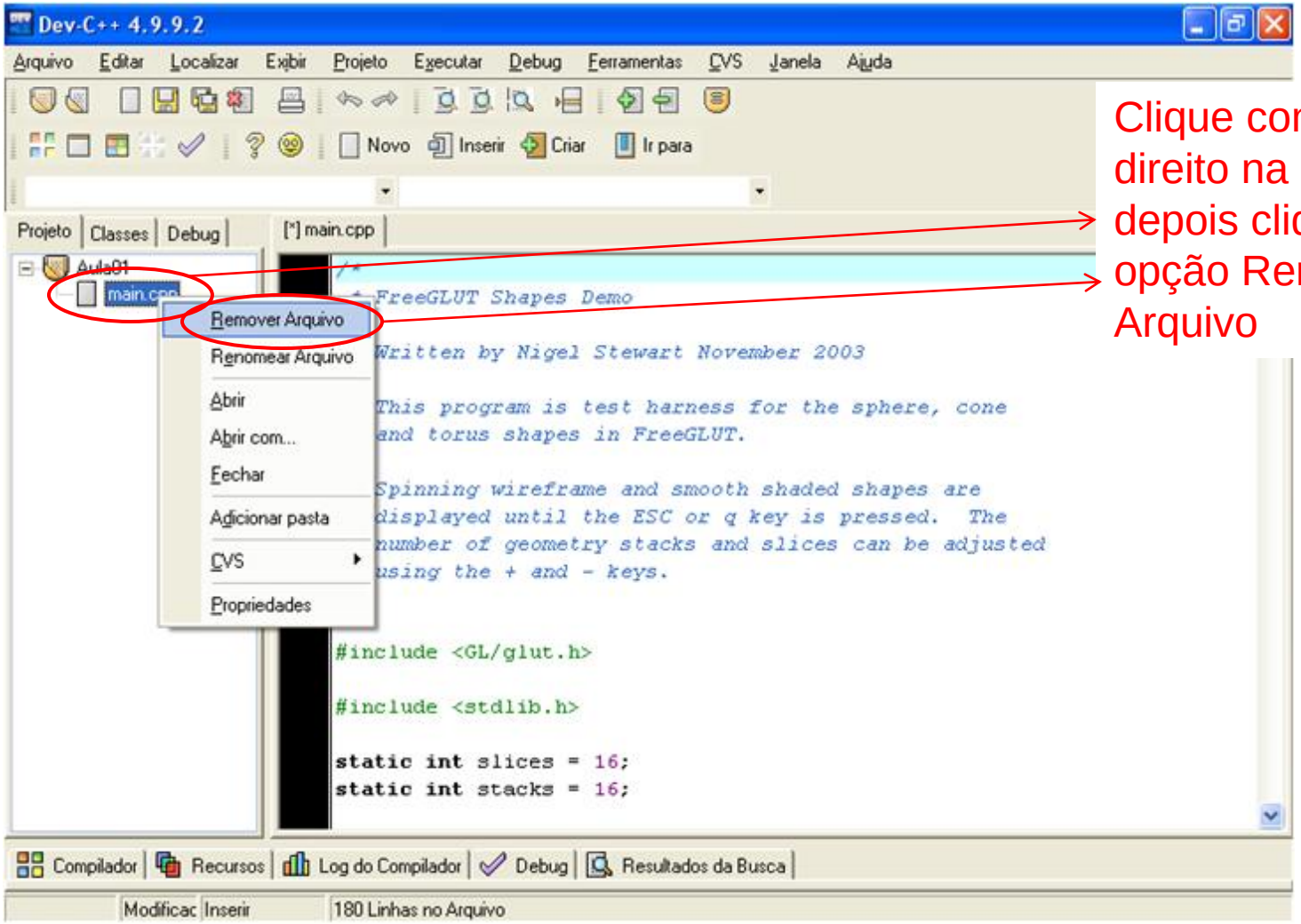
Criando um Projeto GLUT/OpenGL



Salvando o Projeto

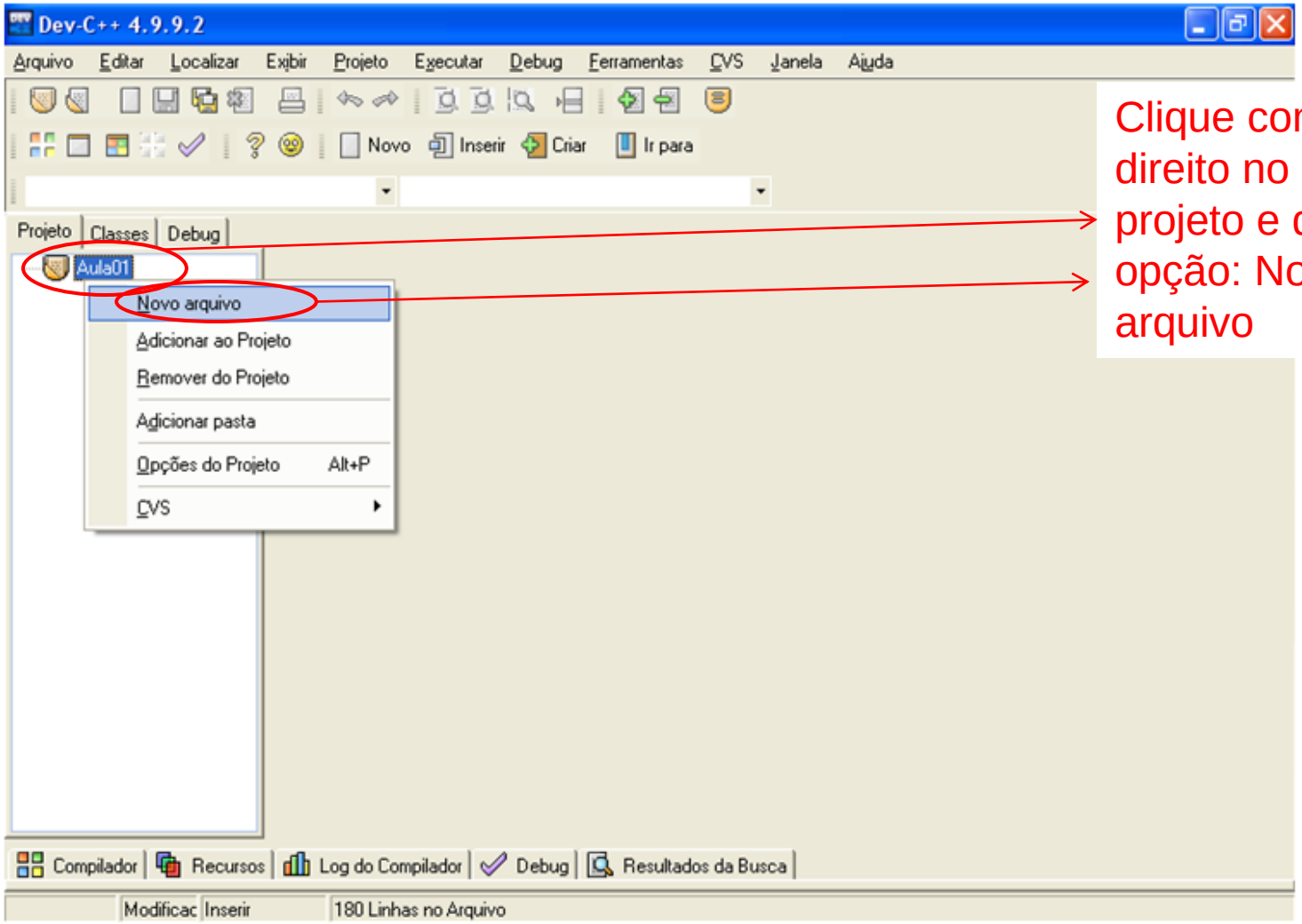


Removendo a main



Clique com o botão direito na main e depois clique na opção Remover Arquivo

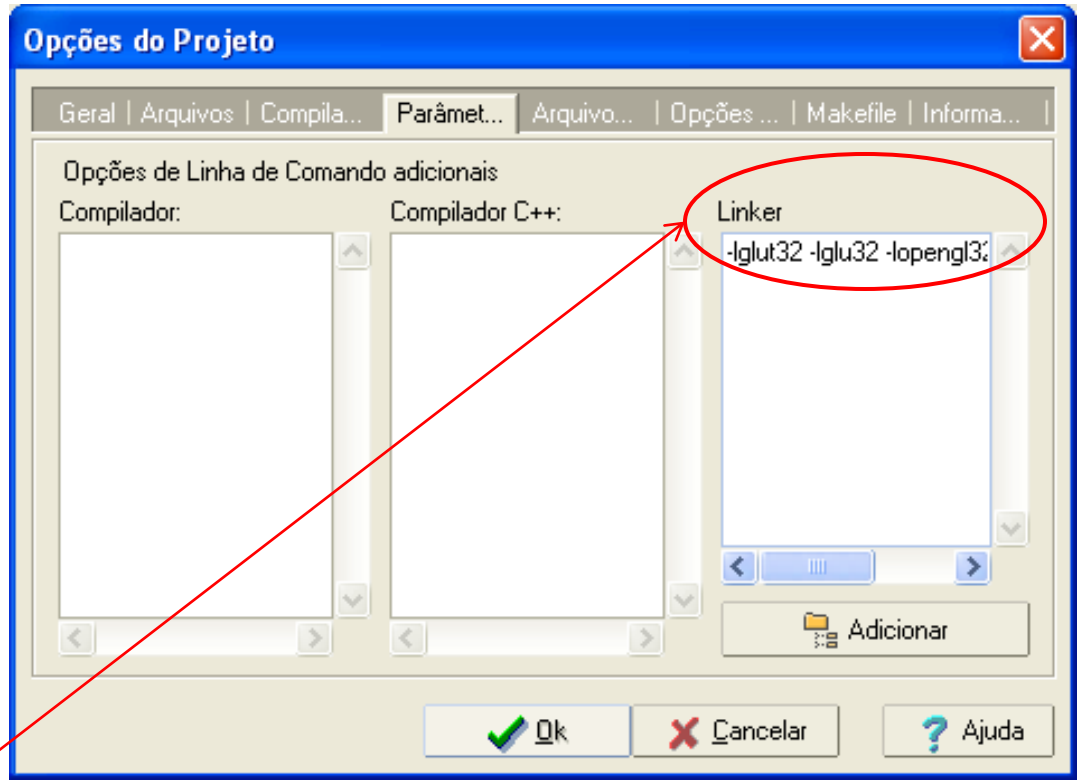
Criando um Novo Arquivo



Clique com o botão direito no nome do projeto e depois na opção: Novo arquivo

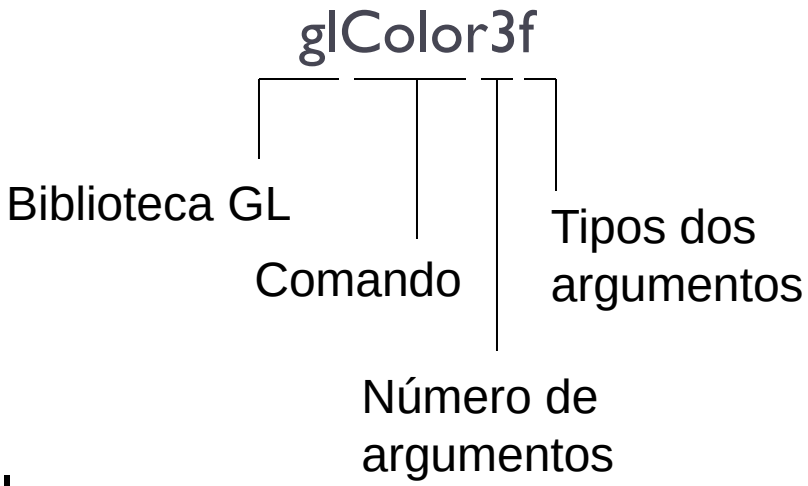
Bibliotecas Utilizadas

- Para visualizar as opções do projeto e verificar as bibliotecas utilizadas:
  - Abra o menu Projeto → Opções do Projeto
- Ao selecionar projeto glut os seguintes comandos do Linker são adicionados automaticamente: -lglut32 -lglu32 -lopengl32 -lwinmm -lgdi32



Sintaxe de Comandos em OPENGL

- GL



- GLU
  - Prefixo glu
- GLUT
  - Prefixo glut

Tipos de Dados OpenGL

Tipo de dado OpenGL	Representação interna	Tipo de dado C equivalente	Sufixo
GLbyte	8-bit integer	signed char	b
GLshort	16-bit integer	short	s
GLint, GLsizei	32-bit integer	int ou long	i
GLfloat, GLclampf	32-bit floating-point	float	f
GLdouble, GLclampd	64-bit floating-point	double	d
GLubyte, GLboolean	8-bit unsigned integer	unsigned char	ub
GLushort	16-bit unsigned integer	unsigned short	us
GLuint, GLenum, GLbitfield	32-bit unsigned integer	unsigned long ou unsigned int	ui
*v		ponteiro	bv, sv, iv, ...

PrimeiroPrograma.cpp

```
//*************************************************************
//
// PrimeiroPrograma.cpp
// Um programa OpenGL simples que abre uma janela GLUT
// e desenha um triângulo no centro
//
// Marcelo Cohen e Isabel H. Manssour
// Este código acompanha o livro
// "OpenGL - Uma Abordagem Prática e Objetiva"
//
//*************************************************************

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

PrimeiroPrograma.cpp

```
void Desenha(void) // Função callback de redesenho da janela de visualização
{
    // Limpa a janela de visualização com a cor branca
    glClearColor(1,1,1,0);  glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(1,0,0);   // Define a cor de desenho: vermelho
    // Desenha um triângulo no centro da janela
    glBegin(GL_TRIANGLES);
            glVertex3f(-0.5,-0.5,0);
            glVertex3f( 0.0, 0.5,0);
            glVertex3f( 0.5,-0.5,0);
    glEnd();
    glFlush();   //Executa os comandos OpenGL
}
```

PrimeiroPrograma.cpp

```
// Função callback chamada para gerenciar eventos de teclas
void Teclado (unsigned char key, int x, int y) {
   if (key == 27)
         exit(0);
}
// Função responsável por inicializar parâmetros e variáveis
void Inicializa(void)  {
   // Define a janela de visualização 2D
   glMatrixMode(GL_PROJECTION);
   gluOrtho2D(-1.0,1.0,-1.0,1.0);
   glMatrixMode(GL_MODELVIEW);
}
```

PrimeiroPrograma.cpp

```
int main(void) { // Programa Principal
    // Define do modo de operação da GLUT
   glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
   glutInitWindowSize(400,400);  // tamanho em pixels da janela
   // Cria a janela passando como argumento o título da mesma
   glutCreateWindow("Primeiro Programa");
   glutDisplayFunc(Desenha);  // Registra função de redesenho
   glutKeyboardFunc (Teclado);  // Registra função de teclado
   Inicializa();   // Chama a função de inicializações
   // Inicia o processamento e aguarda interações do usuário
   glutMainLoop();
   return 0;
}
```

Atributos da Primitiva Line

- GL_LINES
  - Cada par de vértices entre glBegin() e glEnd() definem um segmento de reta
- GL_LINE_STRIP
  - Os vértices definem uma seqüência de segmentos de retas com a extremidade de um segmento iniciando no próximo segmento de reta
- GL_LINE_LOOP
  - Liga os segmentos de reta como em GL_LINE_STRIP mas, além disso, o último vértice é ligado ao primeiro
- Alterando a largura do segmento de reta
  - O comando glLineWidth(GLfloat width) fixa a largura do segmento de reta para exibição. O valor default é igual a 1

Cores e Preenchimento

- O comando glColor3f(GLfloat r, GLfloat g, GLfloa b) fixa a cor da linha
  - Exemplo: glColor3f(1.0,0.0,0.0)
- O atributo GL_POLYGON define que o polígono deve ser exibido com preenchimento.