Universidade Federal do Piauí
Centro de Ciências da Natureza
Departamento de Computação

Introdução ao OpenGL

Prof. Dr. Laurindo de Sousa Britto Neto

1

Introdução ao OpenGL

- **Definição**: OpenGL (*Open Graphics Library*) é uma API (*Application Programming Interface*) para interface com o hardware gráfico;
- **Arquitetura**: Projetado como uma máquina de estados;
- **Portabilidade**: Interface independente do hardware. Implementada em várias plataformas;

2

2

Bibliotecas

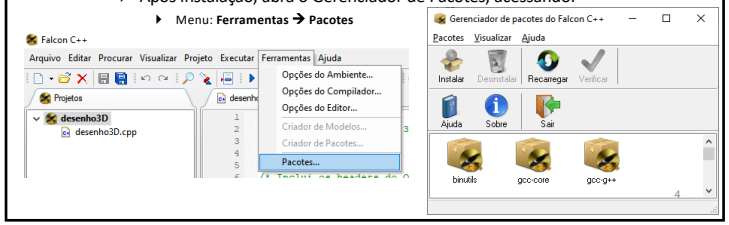
- ***OpenGL Utility Library* (GLU)**
  - Fornece diversas funções para auxiliar na montagem de matrizes de visualização e projeção, desenho de superfícies e imagens 3D.
  - Essa biblioteca é fornecida juntamente com as implementações do OpenGL e todas as rotinas dessa biblioteca tem o prefixo glu.
- ***OpenGL Utility Toolkit* (GLUT)**
  - Implementa uma API para manipular janelas com OpenGL;
  - Adequado para aprendizagem e para desenvolver aplicações de OpenGL simples (de pequeno a médio porte)
  - Alternativas ao GLUT: GTK+, SDL, QT, wxWidgets, GLFW etc.

3

3

Instalação do GLUT

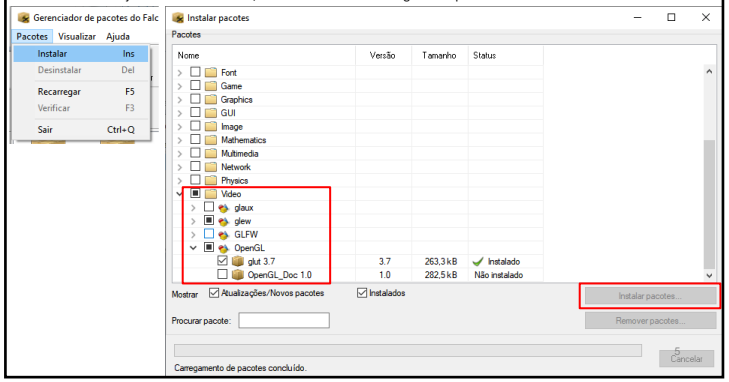
- Linux
  - # apt-get install libglut3-dev
  - # apt-get install freeglut3
- Windows
  - Falcon C++:
    - Download: http://falconcpp.sourceforge.net/
    - Após instalação, abra o Gerenciador de Pacotes, acessando:
      - Menu: **Ferramentas → Pacotes**



4

Instalação do GLUT no Falcon C++ (Windows)

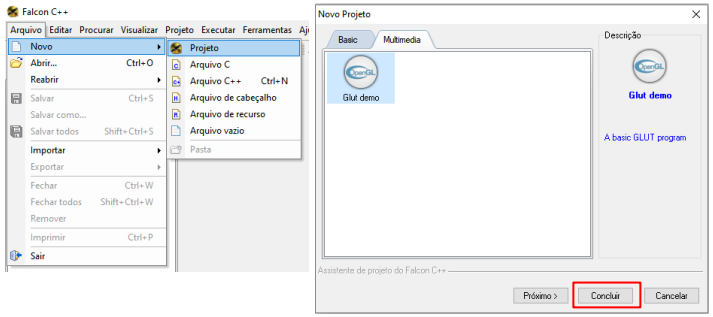
- No Gerenciador de Pacotes, acesse o menu: **Pacotes → Instalar**
  - Na janela Instalar Pacotes, selecione a biblioteca do glut e clique no botão instalar.



5

Configuração do GLUT no Falcon C++ (Windows)

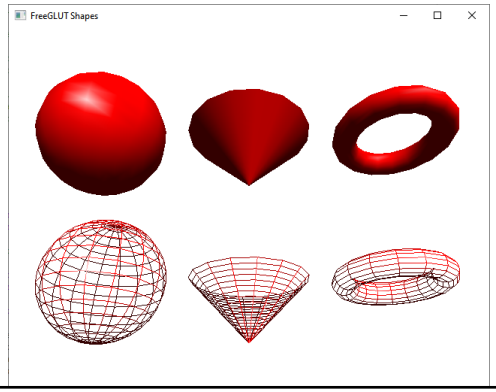
- De volta ao Falcon C++, acesse o menu: **Arquivo → Novo → Projeto**
  - Na janela Novo Projeto, acesse a aba **Multimedia** e clique no botão **Concluir**.



6

Compilação e Execução do GLUT no Falcon C++ (Windows)

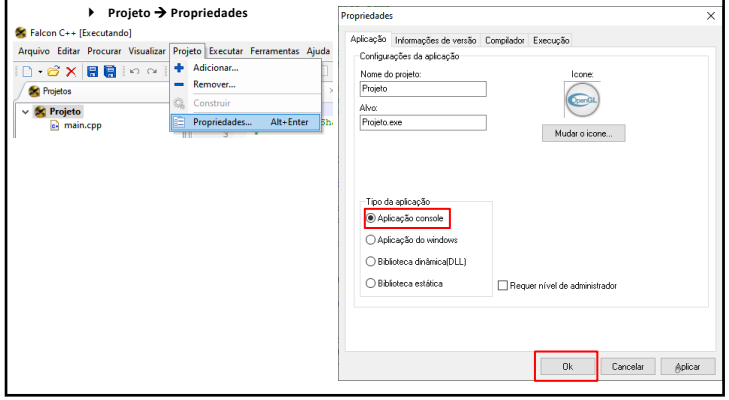
- O Falcon C++ vai criar o novo projeto com um código de exemplo
  - Tecle F9 para compilar e executar. Se a instalação ocorreu com sucesso, aparecerá janela GLUT:



7

Configurações Adicionais no Falcon C++ (Windows)

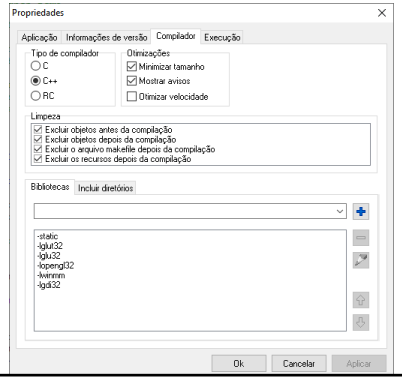
- Para adicionar o console aos programas GLUT, acesse o menu:
  - **Projeto → Propriedades**



8

Parâmetros do Linker no Falcon C++ (Windows)

- A criação do projeto adiciona automaticamente os parâmetros do Linker
  - -static -lglut32 -lglu32 -lopengl3 -lwinmm -lgdi32



9

9

Instalação do GLUT no Visual Studio Community (Windows)

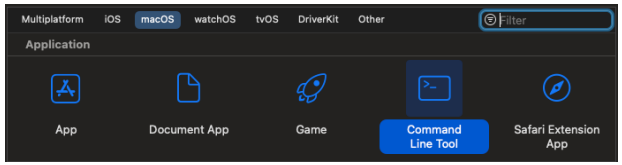
- Download: https://visualstudio.microsoft.com/pt-br/vs/community/
- Download do GLUT pré-compilado para windows: https://www.opengl.org/resources/libraries/glut/glut_downloads.php#windows
- Copie os arquivos nas pastas, conforme as instruções:
  - C:\Program Files\Microsoft Visual Studio\2019\Community\VC\Auxiliary\VS
    - **glut.h**: \include\GL\
    - **glut.lib** e **glut32.lib**:\lib\x64\ e \lib\x86\
  - C:\Windows\SysWOW64 e C:\Windows\System32
    - **glut.dll** e **glut32.dll**

10

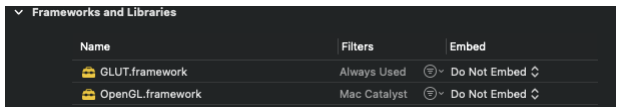
10

Instalação e Configuração no Xcode (macOS)

- O OpenGL e o GLUT já vem instalado.
  - Menu: **File → New → Project**



  - Aba **General:**

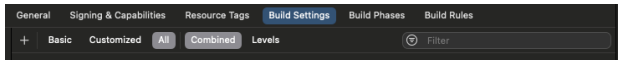


11

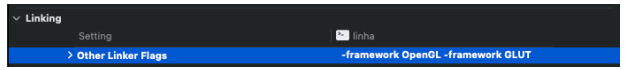
11

Configuração no Xcode (macOS)

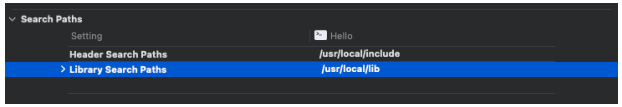
- Aba **Build Settings (All):**
  - **Filter**



  - Filter: **Flags**



  - Filter: **Paths**



12

12

Bibliotecas

```
#ifdef __APPLE__ // MacOS
    #define GL_SILENCE_DEPRECATION
    #include <GLUT/glut.h>
    #include <OpenGL/gl.h>
    #include <OpenGL/glu.h>
#else // Windows e Linux
    #include <GL/glut.h>
    #include <GL/gl.h>
    #include <GL/glu.h>
#endif
```

13

13

Compilando e Executando

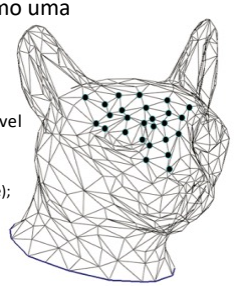
- Windows:
  - Falcon C++: Tecla [F9]
  - Visual Studio: Tecla [Ctrl + F5]
- MacOS:
  - Xcode: Tecla [Command + R]
- Linux:
  - g++ -o exemplo exemplo.cpp -lGL -lGLU –lglut

14

14

OpenGL

- Fornece apenas um conjunto limitado de primitivas geométricas, como pontos, linhas e polígonos;
- As geometrias 3Ds são representadas como uma coleção de triângulos;
  - Triângulos são simples o bastante para serem processados eficientemente pelo OpenGL;
  - Usando coleções de muitos triângulos é possível aproximar superfícies com formas complexas.
    - Ponto: vértice (vertex);
    - Linha (segmento de reta: dois vértices): aresta (edge);
    - Triângulo (três vértices): superfície (face);
    - Coleção de triângulos: malha (mesh).



15

15

Computação em OpenGL

- Determina a posição da tela para cada vértice de cada um dos triângulos;
- Descobre quais pontos da tela, chamados pixels, estão dentro de cada triângulo;
- Realiza alguns cálculos para determinar a cor desejada desse pixel.

16

16

Sintaxe em OpenGL

glVertex3f(float x, float y , float z)
- Prefixo: gl
- Comando: Vertex
- Número de valores especificados pelo tipo: 3
- Tipo específico dos parâmetros: f (float)

glVertex2sv(short v[2])
- Prefixo : gl
- Comando: Vertex
- Número de valores especificados pelo tipo: 2
- Tipo específico dos valores: s (short)
- Vetor: **v** indica que o parâmetro é um vetor

17

17

Resumindo a Sintaxe

- **gl** {nome do comando} {1,2,3,4} {b,s,i,f,d,ub,us,ui} {v}
  - b → Byte
  - s → Short
  - f → Float
  - d → Double
  - ub → uByte
  - us → uShort
  - ui → uInt

18

18

Primitivas Geométricas

- Desenha-se primitivas usando os comandos glBegin() e glEnd();

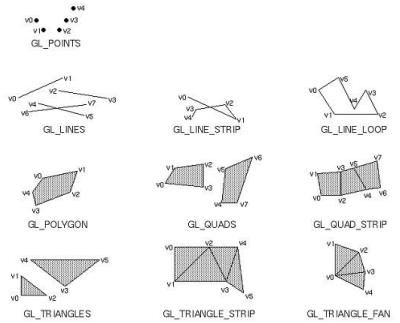
```
glBegin(GL_TRIANGLES);
  glVertex2f(100, 200);
  glVertex2f(0, 0);
  glVertex2f(200, 0);
glEnd();
```

19

19

Tipos de Primitivas (shapes)

- Download do Nate Robins Tutors:
  - https://user.xmission.com/~nate/tutors.html



20

20

Transformações Geométricas (transformation)

- Transformações Geométricas
  - Translação

    glTranslatef(GLfloat x, GLfloat y, GLfloat z)
  - Escala

    glScalef(GLfloat x, GLfloat y, GLfloat z)
  - Rotação

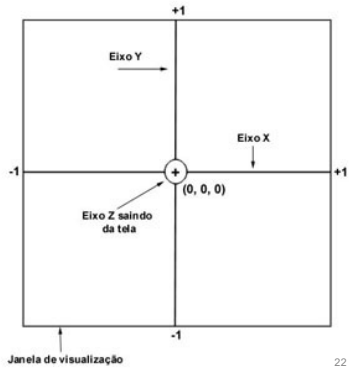
    glRotatef(GLfloat angulo, GLfloat x, GLfloat y, GLfloat z)
    - Ângulo é dado em graus e (x, y, z) é o eixo da rotação

21

21

Projeção Padrão

- Projeção Ortográfica



22

22

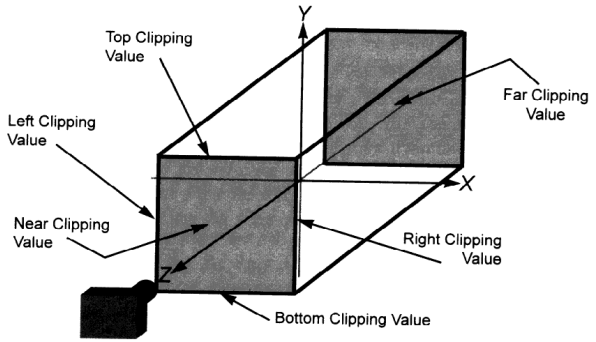
Viewport e Projeção

- **glViewport(0, 0, w, h):** define a área da janela que será desenhada;
- **glOrtho(left,right,bottom,top,near,far)**: usada para definir qual projeção ortográfica será utilizada para determinar o volume de visualização;
- **glMatrixMode(GL_PROJECTION):** avisa ao OpenGL que todas as operações de transformação futuras irão afetar a "câmera" (ou observador);
- **glMatrixMode(GL_MODELVIEW):** avisa ao OpenGL que todas as operações de transformação futuras irão afetar os modelos da cena (o que é desenhado);
- **glLoadIdentity():** faz com que a matriz corrente seja inicializada com a matriz identidade (nenhuma transformação é acumulada), ou seja, reinicia o sistema de coordenadas;

23

23

Exemplo de Projeção Ortográfica (janela)



24

24

Desenhando Círculos (1/3)

- Matriz de rotação 2D:

$$\begin{bmatrix} \cos\theta & sen\theta \\ -sen\theta & \cos\theta \end{bmatrix}$$

$$\begin{bmatrix} x' & y' \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} \cos\theta & sen\theta \\ -sen\theta & \cos\theta \end{bmatrix}$$

$$\begin{cases} x' = x\cos\theta - y\sin\theta \\ y' = x\sin\theta + y\cos\theta \end{cases}$$

25

25

Desenhando Círculos (2/3)

$$\begin{cases} x' = x\cos\theta - y\sin\theta \\ y' = x\sin\theta + y\cos\theta \end{cases}$$

Precisamos do centro e do raio?

se centro = (0,0) **então** x = 0;

se raio > 0 **então** y = raio;

$$\begin{cases} x' = -y\sin\theta \\ y' = y\cos\theta \end{cases}$$

26

26

Desenhando Círculos (3/3) (janela)

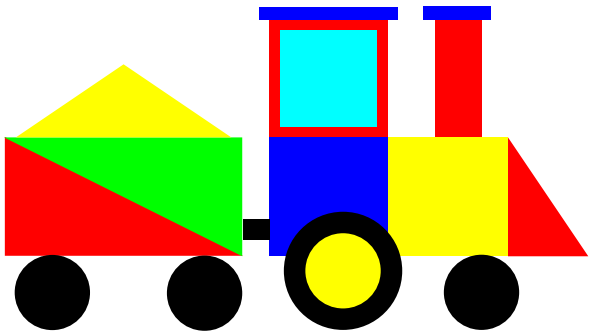
```
void desenhaCirculo(float raio, int num_linhas,bool preenchido){
    int i;
    GLfloat angulo;

    angulo = (2 * M_PI) / num_linhas;

    if(preenchido) glBegin(GL_TRIANGLE_FAN);
    else glBegin(GL_LINE_LOOP);

    for (i = 1; i <= num_linhas; i++){
        glVertex2f(-raio*sin(i * angulo) , raio*cos(i * angulo));
    }
    glEnd();
}
```

27

27

Tarefa

1) Desenhe o Trem 2D em OpenGL



28

28