Relatório

Nomes dos integrantes: Bernardo Morais, Pedro Parentoni, Maguirrichard Oliveira

O projeto, feito no complilador mingw, está organizado em um “.pro”, que é o arquivo responsável por carregar as informações no programa QT, e 4 pastas com os arquivos do programa, sendo elas:

- Headers, contendo os arquivos camera.h, glwidget.h, light.h, mainwindow.h, material.h, trackball.h
- Sources, contendo os arquivos camera.cpp, glwidget.cpp, light.cpp, main.cpp, mainwindows.cpp, material.cpp, trackball.cpp
- Forms, contendo o arquivo mainwindow.ui
- Resources, contendo o arquivo resources.qrc

Dentro de cada arquivo, em sua respectiva pasta, possui elementos fundamentais para a execução correta do programa, sendo as classes responsáveis pela manipulação dos dados que serão carregados, informação da tela principal do programa, configurações do projeto, etc.

As classes, distribuídas dentro de cada arquivo “.cpp” mencionados anteriormente, são:

camera.cpp :

Camera(): Informações sobre a câmera

glwidget.cpp :

explicit GLWidget(QWidget *parent): Construtor padrão do que irá inicializar algumas variáveis padrões do programa.

virtual ~GLWidget(): Destrutor do programa que destrói todos os recursos indesejados na memória ao fim da execução.

void initializeGL(): Inicializa algumas variáveis sempre que um novo objeto é escolhido.

void paintGL(): Renderiza todas as informações salvas do objeto.

void resizeGL(int w, int h): Efetua o redimensionamento da matriz de projeção sempre que o objeto sofre alguma interferência de zoom provido do mouse.

void browseFormFile():

void toggleBackgroundColor():

void readOFFFile(const QString &fileName): Lê o objeto que sera renderizado na tela.

void genNormals(): Responsável por calcular os vetores normais de cada vértice do objeto.

void genTextCoordsCylinder(): Responsável por calcular as coordenadas das texturas do objeto.

void genTangents(): Responsável por calcular as tangentes do objeto.

void createShaders(): Responsável pela criação de cada um dos shaders.

void destroyShaders(): Responsável pela destruição dos shaders utilizados no programa.

void createVBOs(): Cria os Vertex Buffer Objects, que são responsáveis pelo objeto já pronto para ser renderizado criado de uma maneira otimizada.

void destroyVBOs(): Remove qualquer VBO existente de outros objetos.

void keyPressEvent(QKeyEvent *event): Responsável pelos sinais do teclado, que são utilizados para a troca de Shaders.

void mouseMoveEvent(QMouseEvent *event): Cuida do movimento do mouse em relação a janela.

void mousePressEvent(QMouseEvent *event): Cuida do pressionar no mouse.

void mouseReleaseEvent(QMouseEvent *event): Cuida do mouse liberado.

void wheelEvent(QWheelEvent *event): Cuida do scroll do mouse.

void animate():

light.cpp :

Light(): Informações sobre a luz no objeto

main.cpp

main(int argc, char *argv[]): Inicia a aplicação.

mainwindow.cpp :

MainWindow(QWidget *parent): Responsável pela janela principal do programa.

~MainWindow(): Responsável por deletar a janela principal do programa.

material.cpp :

Material(): Informações sobre a difusão natural do objeto.

trackball.cpp :

trackBall(): Classe que transforma as informações do mouse em valores e altera a matriz