Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

**Отчёт**

Лабораторная работа №3

По дисциплине ОСиСП

«Динамические библиотеки»

Выполнил

Студент ФЭИС

3-го курса, группы ПО-5

Бриштен Д.С.

Проверила Дряпко А.В.

Брест 2021

Цель работы: научиться разрабатывать и использовать динамические библиотеки (DLL) с использованием Qt.

Вариант 4

Задание. Доработать программу, разработанную в лабораторной работе №1-2, внеся следующие изменения:

1) Выбрать 3 вспомогательные функции и вынести их описание и реализацию в динамическую библиотеку helper.dll. В основном приложении осуществить загрузку реализованных функций во время работы программы (at run-time, с использованием объекта QLibrary) и их вызов.

2) Выбрать вспомогательный класс и вынести его описание и реализацию в динамическую библиотеку helper\_class.dll. В основном приложении осуществить загрузку реализованного класса во время компиляции (at compile-time).

3) Реализовать окно «О программе» в виде объекта динамической библиотеки about.dll с указанием автора программы, группы, курса и краткого описания разработанного приложения. Осуществить импорт указанной библиотеки и отображение соответствующего окна при выборе пункта меню «О программе».

4) Реализовать расширения для приложения, позволяющие изменять оформление пунктов меню (шрифт, размер, начертание и т.д.). Соответствующие изменения должны происходить при выборе специального пункта меню. Создать как минимум три расширения такого типа.

Файлы интерфейса

**about.h**

#ifndef ABOUT\_H

#define ABOUT\_H

#include "about\_global.h"

#include <QMessageBox>

extern "C" void **showAbout**(QWidget \*);

## #endif // ABOUT\_H

**about\_global.h**

#ifndef ABOUT\_GLOBAL\_H

#define ABOUT\_GLOBAL\_H

#include <QtCore/qglobal.h>

#if defined(ABOUT\_LIBRARY)

# define ABOUT\_EXPORT Q\_DECL\_EXPORT

#else

# define ABOUT\_EXPORT Q\_DECL\_IMPORT

#endif

## #endif // ABOUT\_GLOBAL\_H

**fontPlugin.h**

#ifndef FONTPLUGIN\_H

#define FONTPLUGIN\_H

#include <QFont>

#include <fontPluginInterface.h>

class **FontPlugin** : public QObject, public FontPluginInterface

{

Q\_OBJECT

Q\_INTERFACES(**FontPluginInterface**)

Q\_PLUGIN\_METADATA(**IID** "Pacman.FontPluginInterface" **FILE** "fontPlugin.json")

public:

virtual QFont ***getFont***();

};

## #endif // FONTPLUGIN\_H

**fontPluginInterface.h**

#ifndef FONTPLUGININTERFACE\_H

#define FONTPLUGININTERFACE\_H

#include "pluginInterface\_global.h"

#include <QFont>

class **FontPluginInterface**

{

public:

virtual QFont ***getFont***() = 0;

};

Q\_DECLARE\_INTERFACE(FontPluginInterface, "Sokoban.FontPluginInterface");

## #endif // FONTPLUGININTERFACE\_H

**pluginInterface\_global.h**

#ifndef PLUGININTERFACE\_GLOBAL\_H

#define PLUGININTERFACE\_GLOBAL\_H

#include <QtCore/qglobal.h>

#if defined(PLUGININTERFACE\_LIBRARY)

# define PLUGININTERFACE\_EXPORT Q\_DECL\_EXPORT

#else

# define PLUGININTERFACE\_EXPORT Q\_DECL\_IMPORT

#endif

## #endif // PLUGININTERFACE\_GLOBAL\_H

**helper.h**

#ifndef HELPER\_H

#define HELPER\_H

#include "helper\_global.h"

#include <QPainter>

extern "C" void **generatePointPixmap**();

QPixmap ballpix;

## #endif // HELPER\_H

**helper\_global.h**

#ifndef HELPER\_GLOBAL\_H

#define HELPER\_GLOBAL\_H

#include <QtCore/qglobal.h>

#if defined(HELPER\_LIBRARY)

# define HELPER\_EXPORT Q\_DECL\_EXPORT \_\_declspec(dllexport)

#else

# define HELPER\_EXPORT Q\_DECL\_IMPORT

#endif

## #endif // HELPER\_GLOBAL\_H

**helper\_class\_global.h**

#ifndef HELPER\_CLASS\_GLOBAL\_H

#define HELPER\_CLASS\_GLOBAL\_H

#include <QtCore/qglobal.h>

#if defined(HELPER\_CLASS\_LIBRARY)

# define HELPER\_CLASS\_EXPORT Q\_DECL\_EXPORT

#else

# define HELPER\_CLASS\_EXPORT Q\_DECL\_IMPORT

#endif

## #endif // HELPER\_CLASS\_GLOBAL\_H

**textdrawing.h**

#ifndef TEXTDRAWING\_H

#define TEXTDRAWING\_H

#include <QPainter>

#include <QGraphicsItem>

#include <QGraphicsScene>

class **Textdrawing** : public QGraphicsItem

{

public:

**Textdrawing**();

QRectF ***boundingRect***() const;

void **SetOver**(bool over);

void ***paint***(QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget);

bool over,playing;

int x,y,w,h,score;

};

## #endif // TEXTDRAWING\_H

**ball.h**

#ifndef BALL\_H

#define BALL\_H

#include <QPainter>

#include <QGraphicsItem>

#include <QGraphicsScene>

#include <QLibrary>

#include <QDebug>

typedef void (\*lib\_func)();

class **Ball** : public QGraphicsItem

{

public:

**Ball**();

QRectF ***boundingRect***() const;

void ***paint***(QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget);

void **drawballs**(QPainter \*painter);

void **setpoints**(QVector<QPoint> points);

void **changeGeometry**() { prepareGeometryChange(); }

int ballx,bally;

int ballw,ballh;

QPixmap ballpix;

QRectF rec;

QVector<QPoint> points;

lib\_func f;

};

## #endif // BALL\_H

**window.h**

#ifndef WINDOW\_H

#define WINDOW\_H

#include <QDialog>

#include <QtCore>

#include <QtGui>

#include <QMenuBar>

#include <QPainter>

#include "pacman.h"

#include "map.h"

#include "ball.h"

#include "ghost.h"

#include "powerball.h"

#include "textdrawing.h"

#include "fontPluginInterface.h"

typedef void (\*about)(QWidget \*painter);

namespace **Ui** {

class **window**;

}

class **window** : public QDialog

{

Q\_OBJECT

public:

explicit **window**(QWidget \*parent = 0);

void **pacman\_move**();

void **ghostsmove**();

void **ghostsmove1**();

void **ghostsmove2**();

void **moveghostsinrect2**();

void **moveghostsinrect1**();

void **moveghostsinrect**();

void **checklost**();

void **delay**();

void **start\_Game**();

void **end\_Game**();

void **loadPlugins**();

Pacman \*pacman;

Ghost \*ghost;

Ghost \*ghost1;

Ghost \*ghost2;

Map \*pac\_map;

Ball \*ball;

PowerBall \*powerball;

Textdrawing \*text;

about f2;

int pacx,pacy,direction,nextdirection;

int gosx,gosy,ghostdir,nextghostdir;

int gosx1,gosy1,ghostdir1,nextghostdir1;

int gosx2,gosy2,ghostdir2,nextghostdir2;

bool moving,ghostmoving,ghostmoving1,ghostmoving2;

bool scared,scared1,scared2;

int score,state3,state2,state1,state;

bool start,delayb;

bool ghoststart,ghoststart1,ghoststart2;

bool playing;

QVector<QPoint> ballpoints;

QVector<QPoint> Powerballpoints;

QThread \*sleeper;

~***window***();

public slots:

void **updater**();

void **ghostupdater**();

void **About**();

void **changeAppFont**(QFont font);

protected:

void ***keyPressEvent***(QKeyEvent \*event);

private:

Ui::window \*ui;

QGraphicsScene \*scene;

QTimer \*timer;

QTimer \*ghoststimer;

QMenuBar \*menuBar;

QMenu \*style;

};

## #endif // WINDOW\_H

Файлы реализации

**about.cpp**

#include "about.h"

void **showAbout**(QWidget \*parent)

{

QMessageBox::about(*parent*, "about program", "Pacman D.S.Brishten");

## }

**fontPlugin.cpp**

#include "fontPlugin.h"

QFont FontPlugin::***getFont***()

{

QFont font;

return font.defaultFamily();

## }

**helper.cpp**

#include "helper.h"

void **generatePointPixmap**()

{

QRect bounds = QRect(0, 0, 3, 3);

QPainter painter;

ballpix = QPixmap(bounds.size());

ballpix.fill(Qt::transparent);

painter.begin(*&ballpix*);

painter.setRenderHint(QPainter::Antialiasing);

painter.setPen(Qt::NoPen);

painter.setBrush(Qt::yellow);

painter.drawEllipse(0, 0, 3, 3);

## }

**textdrawing.cpp**

#include "textdrawing.h"

Textdrawing::**Textdrawing**()

{

over=false;

w=350;

h=50;

x=450/2-w/2;

y=480/2-h/2;

score=0;

playing=false;

}

QRectF Textdrawing::***boundingRect***() const

{

return QRect(x, y, w, h);

}

void Textdrawing::**SetOver**(bool over)

{

this->over=over;

}

void Textdrawing::***paint***(QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget)

{

QPen pen(Qt::blue);

painter->setPen(pen);

QFont font=painter->font() ;

QBrush brush(Qt::blue);

painter->setBrush(brush);

if(!playing){

font.setPointSize ( 18 );

painter->setFont(font);

if(over){

painter->drawText( x+90,y-70, "GAME OVER." );

painter->drawText( x+20,y+120, "PRESS ENTER TO PLAY." );

painter->drawText(*boundingRect*(),Qt::AlignCenter, "SCORE : "+QString::number(score) );

}else{

painter->drawText(*boundingRect*(),Qt::AlignCenter, "PRESS SPACE TO START");

}

}else{

font.setPointSize (13);

painter->setFont(font);

painter->drawText( 50,15, "SCORE : "+QString::number(score));

}

## }

**ball.cpp**

#include "ball.h"

Ball::**Ball**()

{

ballx=0;

bally=0;

ballw=5;

ballh=5;

//generatePointPixmap();

QLibrary lib("D:/5 semester/OSISP/Lab3/build-helper-Desktop\_Qt\_5\_15\_2\_MinGW\_32\_bit-Debug/release/helper.dll");

f = (lib\_func)lib.resolve("generatePointPixmap");

}

QRectF Ball::***boundingRect***() const

{

return QRect(0, 0, 450, 550);

}

void Ball::***paint***(QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget)

{

for(int i=0;i<points.size();i++){

painter->drawPixmap( points[i].x(), points[i].y(), ballpix );

}

}

void Ball::**drawballs**(QPainter \*painter)

{

for(int i=0;i<points.size();i++){

painter->drawPixmap( points[i].x(), points[i].y(), ballpix );

}

}

void Ball::**setpoints**(QVector<QPoint> points)

{

this->points.clear();

this->points=points;

## }

**window.cpp**

#include "window.h"

#include "ui\_window.h"

#include "textdrawing.h"

#include <QThread>

#include <QMenuBar>

#include <QMenu>

#include <QVBoxLayout>

#include <QFontDialog>

#include <QDebug>

window::**window**(QWidget \*parent) :

QDialog(*parent*),

ui(new Ui::window)

{

score=0;

gosx=450/2;

gosy=480/2;

gosx1=450/2;

gosy1=480/2;

gosx2=450/2;

gosy2=480/2;

pacx=410/2;

pacy=360;

ghostmoving=false;

ghostmoving1=false;

ghostmoving2=false;

scared=false;

scared1=false;

scared2=false;

start=false;

direction=0;

moving=false;

delayb=false;

playing=false;

ui->setupUi(this);

scene = new QGraphicsScene(this);

ui->graphicsView->setScene(*scene*);

ui->graphicsView->setRenderHint(QPainter::Antialiasing);

scene->setSceneRect(0,0,640,480);

ui->graphicsView->setSceneRect(scene->sceneRect());

pacman=new Pacman;

pacx=410/2;

pacy=410;

text=new Textdrawing;

text->over=false;

scene->addItem(*text*);

text->setZValue(7);

menuBar = new QMenuBar();

QAction \*aboutAction = menuBar->addAction(tr("О программе"));

connect(aboutAction, SIGNAL(triggered()), this, SLOT(About()));

style = new QMenu(tr("Стили"), this);

menuBar->addMenu(*style*);

QVBoxLayout \*mainLayout = new QVBoxLayout;

mainLayout->setMenuBar(*menuBar*);

setLayout(*mainLayout*);

loadPlugins();

}

void window::**changeAppFont**(QFont font)

{

QApplication::setFont(font);

}

void window::**loadPlugins**()

{

QDir dir(qApp->applicationDirPath() + "/plugins");

QStringList filter("\*.dll");

QPluginLoader loader;

foreach(QFileInfo info, dir.entryInfoList(filter))

{

loader.setFileName(info.absoluteFilePath());

FontPluginInterface\* fontPlugin = qobject\_cast<FontPluginInterface\*>(*loader.instance()*);

if(fontPlugin)

{

QString name = loader.metaData().value("MetaData").toObject().value("fontName").toString();

QAction\* actionFontPlugin = style->addAction(name);

connect(actionFontPlugin, &QAction::triggered, this, [this, fontPlugin]{changeAppFont(fontPlugin->*getFont*());});

}

}

}

void window::**About**()

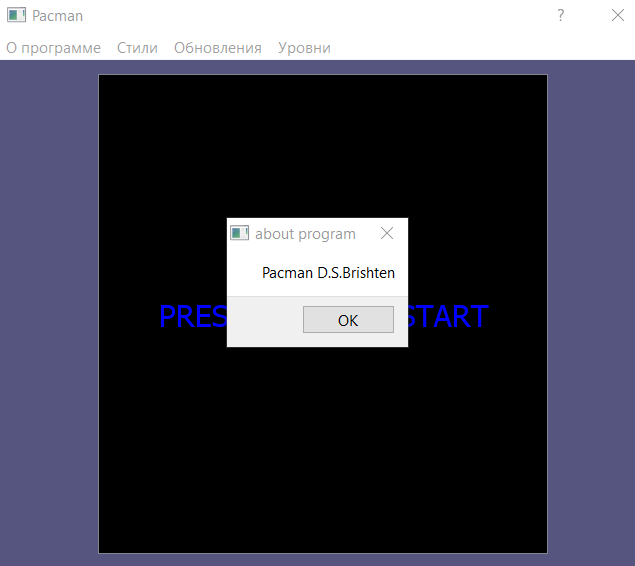
{

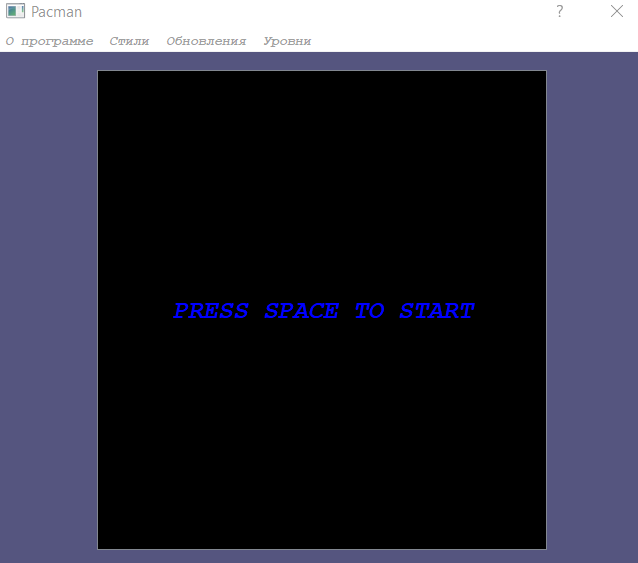
QLibrary lib2("D:/5 semester/OSISP/Lab3/build-about-Desktop\_Qt\_5\_15\_2\_MinGW\_32\_bit-Debug/debug/about.dll");

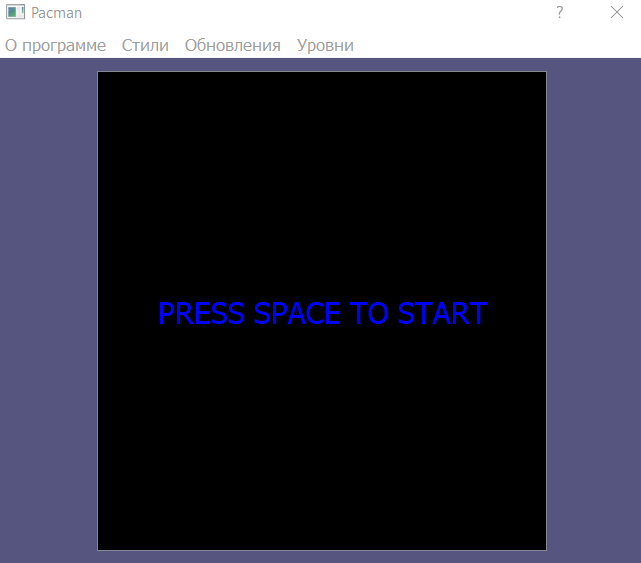
f2 = (about)lib2.resolve("showAbout");

f2(this);

## }







Вывод: Научился использовать динамические библиотеки Qt.