МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

 «БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра ИИТ

ЛАБОРАТОРНАЯ РАБОТА №3

По дисциплине: «ОСИСП»

Выполнил:

Студент ФЭИС

3-го курса, группы ПО-5

Лозейко П. А.

Проверила:

Дряпко А.В.

Брест 2021

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

*Задание:*

Доработать программу, разработанную в лабораторной работе No1-2, 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 биб-

лиотеку helper\_class.dll. В основном приложении осуществить загрузку реализованного класса

во время компиляциии..

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.

**about.h**

#ifndef ABOUT\_H

#define ABOUT\_H

#include "about\_global.h"

extern "C" ABOUT\_EXPORT void about();

#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

**generator.h**

#ifndef GENERATOR\_H

#define GENERATOR\_H

#include <QVector>

#include <QPair>

#include <QStack>

#include <ctime>

QVector<QVector<int> > generate(int width, int height);

void markCell(QVector<QVector<int> > &lens, QStack<QPair<int, int> >& stack, QVector<QVector<int> > field);

#endif // GENERATOR\_H

**helper.h**

#ifndef HELPER\_H

#define HELPER\_H

#include "helper\_global.h"

#include <QString>

extern "C" HELPER\_EXPORT QString set\_window\_name();

extern "C" HELPER\_EXPORT QString set\_name\_About();

extern "C" HELPER\_EXPORT QString set\_window\_btn\_name();

#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

#else

# define HELPER\_EXPORT Q\_DECL\_IMPORT

#endif

#endif // HELPER\_GLOBAL\_H

**maze.h**

#ifndef MAZE\_H

#define MAZE\_H

#include <QMainWindow>

#include <QPaintEvent>

#include <QKeyEvent>

#include <QWidget>

#include <QComboBox>

#include <QPainter>

#include <time.h>

#include <QLibrary>

#include <QString>

#include <QtWidgets>

#include <QMessageBox>

#include "generator.h"

#include "interface.h"

const int BLOCK\_SIZE=15;

const int MARGIN=22;

const int AREA\_ROW=31;

const int AREA\_COL=31;

enum moved

{

UP,

DOWN,

LEFT,

RIGHT,

};

QT\_BEGIN\_NAMESPACE

namespace Ui { class Maze; }

QT\_END\_NAMESPACE

class Maze : public QMainWindow

{

Q\_OBJECT

public:

Maze(QWidget \*parent = nullptr);

virtual void paintEvent(QPaintEvent \*event);

virtual void keyPressEvent(QKeyEvent \*event);

~Maze();

int place\_x;

int place\_y;

int place\_finish\_x, place\_finish\_y;

QVector<QVector<int> > field;

public:

void SetPlugins();

void InitGame();

void InitPause();

void PauseResumeGame();

bool IsGameOver();

void InitMenu();

void gener();

public slots:

void on\_btn\_exit\_clicked();

private slots:

void SnakeUpdate(int key);

void aboutQt();

void applyPlugin(int index);

void on\_btn\_start\_clicked();

private:

Ui::Maze \*ui;

Ui::Maze \*game\_win; // окно для игры

bool flag\_game = false;

bool isPause;

QList<QPoint> wall;

QList<QPoint> man;

moved dir;

QVector< Interface\* > mPlugins;

int countLevels = 0;

};

#endif // MAZE\_H

**game.cpp**

#include "maze.h"

#include "ui\_maze.h"

void Maze::InitGame()

{

field = generate(30, 30);

game\_win->setupUi(this);

game\_win->centralwidget->setVisible(false);

typedef QString (\*Helper\_set\_name\_window)();

// установка имени окна приложения

QLibrary \*helper\_set = new QLibrary("Helper");

Helper\_set\_name\_window set\_window\_name\_ = (Helper\_set\_name\_window)helper\_set->resolve("set\_window\_name");

QString windowTitle = set\_window\_name\_();

delete helper\_set;

this->setWindowTitle(windowTitle);

man.clear();

wall.clear();

gener();

srand(time(0));

isPause=false;

}

void Maze::gener() {

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

for (int j = 0; j < field.at(i).size(); j++) {

int cur = field.at(i).at(j);

if (cur == 0) {qMakePair(i, j); wall.push\_back(QPoint(i, j));}

if (cur == 4) {qMakePair(i, j); place\_finish\_x = i, place\_finish\_y = j; qDebug() << place\_finish\_x << place\_finish\_y;}

else if (cur == 3) {qMakePair(i, j); man.push\_back(QPoint(i, j));}

}

}

}

void Maze::paintEvent(QPaintEvent \*event)

{

if(flag\_game) {

Q\_UNUSED(event);

QPainter painter(this);

painter.setBrush(Qt::black);

painter.drawRect(MARGIN,MARGIN,AREA\_COL\*BLOCK\_SIZE,AREA\_ROW\*BLOCK\_SIZE);

painter.setBrush(Qt::blue);

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

painter.drawRect(MARGIN+wall[i].x()\*BLOCK\_SIZE,MARGIN+wall[i].y()\*BLOCK\_SIZE,BLOCK\_SIZE,BLOCK\_SIZE);

painter.setBrush(Qt::red);

painter.drawRect(MARGIN+man[0].x()\*BLOCK\_SIZE,MARGIN+man[0].y()\*BLOCK\_SIZE,BLOCK\_SIZE,BLOCK\_SIZE);

}

}

void Maze::keyPressEvent(QKeyEvent \*event)

{

if(flag\_game) {

switch(event->key())

{

case Qt::Key\_Up:

SnakeUpdate(Qt::Key\_Up);

break;

case Qt::Key\_Down:

SnakeUpdate(Qt::Key\_Down);

break;

case Qt::Key\_Left:

SnakeUpdate(Qt::Key\_Left);

break;

case Qt::Key\_Right:

SnakeUpdate(Qt::Key\_Right);

break;

case Qt::Key\_Escape:

PauseResumeGame();

break;

default:

break;

}

}

}

bool Maze::IsGameOver()

{

int x=man.front().x();

int y=man.front().y();

if(x<0||x>AREA\_COL-1||y<0||y>AREA\_ROW-1)

return true;

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

if(wall[i]==man.front())

return true;

return false;

}

void Maze::SnakeUpdate(int key)

{

if(key == Qt::Key\_Up) man.push\_front(QPoint(man.front().x(),man.front().y()-1));

if(key == Qt::Key\_Down) man.push\_front(QPoint(man.front().x(),man.front().y()+1));

if(key == Qt::Key\_Left) man.push\_front(QPoint(man.front().x()-1,man.front().y()));

if(key == Qt::Key\_Right) man.push\_front(QPoint(man.front().x()+1,man.front().y()));

if(IsGameOver())

{

man.clear();

man.push\_back(QPoint(place\_x, place\_y));

}

else {

man.pop\_back();

place\_x = man.front().x();

place\_y = man.front().y();

qDebug() << place\_x << place\_y;

update();

}

if(place\_x == place\_finish\_x && place\_y == place\_finish\_y) {

QMessageBox::information(this, "Информация", "Вы прошли уровень");

flag\_game = false;

InitMenu();

}

}

**maze.cpp**

#include "maze.h"

#include "ui\_maze.h"

Maze::Maze(QWidget \*parent): QMainWindow(parent), ui(new Ui::Maze)

{

typedef QString (\*Helper\_set\_name\_window)();

typedef QString (\*Helper\_set\_name\_about)();

typedef QString (\*Helper\_set\_name\_btn)();

ui->setupUi(this);

setFixedSize(QSize(MARGIN\*2+(AREA\_COL+1)\*BLOCK\_SIZE,MARGIN\*2+AREA\_ROW\*BLOCK\_SIZE));

SetPlugins();

// установка имени окна приложения

QLibrary \*helper\_set = new QLibrary("Helper");

Helper\_set\_name\_window set\_window\_name\_ = (Helper\_set\_name\_window)helper\_set->resolve("set\_window\_name");

QString windowTitle = set\_window\_name\_();

delete helper\_set;

this->setWindowTitle(windowTitle);

// установка имени кнопки

QLibrary \*helper\_set\_n\_btn = new QLibrary("Helper");

Helper\_set\_name\_btn set\_btn\_name\_ = (Helper\_set\_name\_btn)helper\_set\_n\_btn->resolve("set\_window\_btn\_name");

ui->btn\_start->setText(set\_btn\_name\_());

delete helper\_set\_n\_btn;

// установка имени About

QLibrary \*helper\_set\_n\_about = new QLibrary("Helper");

Helper\_set\_name\_about set\_window\_name\_first = (Helper\_set\_name\_about)helper\_set\_n\_about->resolve("set\_name\_About");

QMenu\* menu = new QMenu(set\_window\_name\_first());

delete helper\_set\_n\_about;

ui->menubar->addMenu(menu);

menu->addAction("about.dll", this,SLOT(aboutQt()), Qt::CTRL + Qt::Key\_Q);

}

void Maze::InitMenu() {

typedef QString (\*Helper\_set\_name\_window)();

typedef QString (\*Helper\_set\_name\_about)();

typedef QString (\*Helper\_set\_name\_btn)();

ui->setupUi(this);

setFixedSize(QSize(MARGIN\*2+(AREA\_COL+1)\*BLOCK\_SIZE,MARGIN\*2+AREA\_ROW\*BLOCK\_SIZE));

connect(ui->btn\_start, SIGNAL(clicked()), this, SLOT(on\_btn\_start\_clicked()));

connect(ui->btn\_exit, SIGNAL(clicked()), this, SLOT(on\_btn\_exit\_clicked()));

SetPlugins();

// установка имени окна приложения

QLibrary \*helper\_set = new QLibrary("Helper");

Helper\_set\_name\_window set\_window\_name\_ = (Helper\_set\_name\_window)helper\_set->resolve("set\_window\_name");

QString windowTitle = set\_window\_name\_();

delete helper\_set;

this->setWindowTitle(windowTitle);

// установка имени кнопки

QLibrary \*helper\_set\_n\_btn = new QLibrary("Helper");

Helper\_set\_name\_btn set\_btn\_name\_ = (Helper\_set\_name\_btn)helper\_set\_n\_btn->resolve("set\_window\_btn\_name");

ui->btn\_start->setText(set\_btn\_name\_());

delete helper\_set\_n\_btn;

// установка имени About

QLibrary \*helper\_set\_n\_about = new QLibrary("Helper");

Helper\_set\_name\_about set\_window\_name\_first = (Helper\_set\_name\_about)helper\_set\_n\_about->resolve("set\_name\_About");

QMenu\* menu = new QMenu(set\_window\_name\_first());

delete helper\_set\_n\_about;

ui->menubar->addMenu(menu);

menu->addAction("about.dll", this, SLOT(aboutQt()), Qt::CTRL + Qt::Key\_Q);

}

void Maze::PauseResumeGame()

{

if(!isPause)

{

isPause=!isPause;

InitPause();

}

else isPause=!isPause;

}

void Maze::InitPause()

{

QMessageBox::StandardButton pause;

pause = QMessageBox::question(this, "Pause", "Продолжить?",

QMessageBox::Yes|QMessageBox::No);

if (pause == QMessageBox::Yes) {

PauseResumeGame();

}

else {

flag\_game = false;

InitMenu();

}

}

void Maze::on\_btn\_start\_clicked()

{

ui->centralwidget->setVisible(false);

flag\_game = true;

InitGame();

}

Вывод: научился разрабатывать и использовать динамические библиотеки (DLL) с использованием Qt.