|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |   Институт Искусственного Интеллекта | |
| Кафедра программного обеспечения систем радиоэлектронной аппаратуры | |
|  |
|  |
|  |
| **КУРСОВАЯ РАБОТА** | |

по дисциплине \_\_\_\_\_\_«Методы и стандарты программирования» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
 (наименование дисциплины)

**Тема курсовой работы:** Разработка игры «НЛО»

**Студент группы** \_КМБО-05-20, Гудков Всеволод Алексеевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
 (учебная группа, фамилия, имя, отчество студента) (подпись студента)

**Руководитель курсовой работы** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **(**Должность, звание, ученая степень) (подпись руководителя)

**Консультант**. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
 (Должность, звание, ученая степень) (подпись руководителя)

Работа представлена к защите «\_\_\_\_» \_\_\_\_\_\_\_\_\_20\_\_г.

Допущен к защите «\_\_\_\_» \_\_\_\_\_\_\_\_\_20\_\_г.

Москва 2022

**Оглавление**

Оглавление2

Введение3

Постановка задачи3

Описание программы3

1. Общие сведения3

1.1. Наименование программы3

1.2. Функциональное назначение программы3

1.3. Программное обеспечение, необходимые для функционирования программы3

1.4. Язык программирования, на котором написана программа3

2. Описание структуры программы3

2.1. Алгоритмы работы программы4

2.2. Технические средства4

2.3. Вызов программы5

3. Руководство пользователя5

3.1. Сборка программы5

3.2. Запуск программы5

3.3. Процесс игры5

4. Рисунки6

Заключение10

Список использованных источников11

Приложение12

Введение

Задачей курсовой работы является разработка и реализация проекта «НЛО» с графическим интерфейсом, с использованием программного интерфейса Qt creator. В ходе работы развиваются навыки программирования, анализа требований и разработки алгоритмов, критического и абстрактного мышления и командной работы.

Постановка задачи

Данная курсовая работа предусмотрена учебным планом по специальности и выполняется в третьем семестре. Целью работы является закрепление и практическое применение навыков, полученных вследствие посещений курса «Методы и стандарты программирования», а также расширение знаний в области работы с графическими интерфейсами и закрепление навыков самостоятельной разработки программного обеспечения от начала до конца.

Описание программы

1. Общие сведения
   1. Наименование программы

Название разрабатываемой программы – игра «НЛО».

* 1. Функциональное назначение программы

Игра «НЛО» предназначена для интерактивного компьютерного развлечения. Так как у программы монохромный, минималистичный графический интерфейс, контролируемый эксклюзивно мышкой и некоторыми клавишами клавиатуры (стрелками), что делает программу простой в осваивании и удобной в использовании.

* 1. Программное обеспечение, необходимое для функционирования программы

Операционная система Windows 10, 4 451КБ свободного места на жёстком диске и Qt creator версии 4.3 и выше.

* 1. Язык программирования, на котором написана программа

Программа и все её классы написаны на языке C++, с использованием основных библиотек Qt creator. Для реализации курсовой работы использовался Visual Studio Code– редактор кода с функциями IDE. Данный инструмент не является полноценной средой разработки, но он удобен в использовании и его функциональность была достаточна для реализации программы.

1. Описание структуры программы:

Исходные файлы данной программы можно найти по ссылке:

https://github.com/

2.1 Алгоритмы работы программы, структура программы, описание функций составных частей и связей между ними

Алгоритм разделен между шестью классами: «mainwindow», «Aboutwindow», «Introductionwindow», «ufo», «ufo\_killer» и «ufo\_people». При запуске игры создается объект класса «ufo\_killer» и «ufo\_people».

Класс «ufo\_people» отвечает за создание инопланетных существ, а класс «ufo» за передвижение нашего корабля и его создания ufo или НЛО. Класс «ufo\_killer» отвечает за поведение нашей «биомассы» и ее создания.

Классы «Aboutwindow», «Introductionwindow» отвечают за вывод сообщений пунктов “help” с графикой и музыкальным сопровождением. Класс «mainwindow» отвечает за отрисовку графического интерфейса и обработки информации, вводимой игроком. За игровое поле.

Диаграмма классов

MainWindow

+aboutInfo()

+createMenu()

+createActions()

+Introduction()

-

-

-

-

-

…

1

1

1

ufo

1

1

1

ufo\_killer

1

1

IntroductionWindow

1

0…n

AboutWindow

ufo\_people

2.2 Технические средства

Для нормального функционирования программы необходим IBM-совместимый компьютер под управлением операционной системы Windows 10, Qt creator, монитор с любым разрешением экрана (рекомендуется разрешение не меньше 640x480), компьютерная мышь и около 4 500 КБ свободного места на жёстком диске для установки и работы программы.

2.3 **Вызов программы**

Программа запускается переходом в компилятор qt creator и открытием UFO\_Rescue\_spaceship. Для успешного запуска программы необходимо, чтобы библиотеки «Qt5core.dll», «Qt5Gui.dll», «Qt5Multimedia.dll» и «Qt5Widgets.dll» либо лежали в той же папке, что и бинарный UFO\_Rescue\_spaceship.exe файл, либо полный путь к ней лежал в переменной окружения PATH.

1. Руководство пользователя

3.1 Сборка программы

Перед сборкой у вас должны быть установлены следующие программы:

* Cmake
* Visual Studio 15 2017 года и новее.
* Qt creator version 4.8 и новее.

Для сборки убедитесь, что у вас имеются эти программы, далее вам необходимо прописать команду cmake и указать путь до папки с исходными файлами.

3.2 Запуск программы

Программа запускается переходом в компилятор qt creator и открытием UFO\_Rescue\_spaceship. Для успешного запуска программы необходимо, чтобы библиотеки «Qt5core.dll», «Qt5Gui.dll», «Qt5Multimedia.dll» и «Qt5Widgets.dll» либо лежали в той же папке, что и бинарный UFO\_Rescue\_spaceship.exe файл, либо полный путь к ней лежал в переменной окружения PATH.

3.3 Процесс игры

Запустив приложение, перед пользователем появляется окно размера 600x640 с графическим интерфейсом (Рисунок 1). Кнопка start отвечает за начало игры. Кнопки верх, вниз, вправо, влево, выполняют функции «вперед/назад», «поворот». Кнопка «пробел» ставит игру на паузу при повторном нажатии игра возобновляется. Управление производится в основном клавиатурой. После того как вы нажимаете на кнопку «start» «биомасса» начинает следовать за вами, ваша цель уходить от нее используя кнопки управления. Когда биомасса вас догонит игра заканчивается. По ходу игры на игровом поле будут появляться пришельцы, которых нужно собирать на ваш корабль.

Для окончания игры достаточно нажать на крестик в правом верхнем углу окна с графическим интерфейсом, закрыв таким образом программу.

**Рисунки**

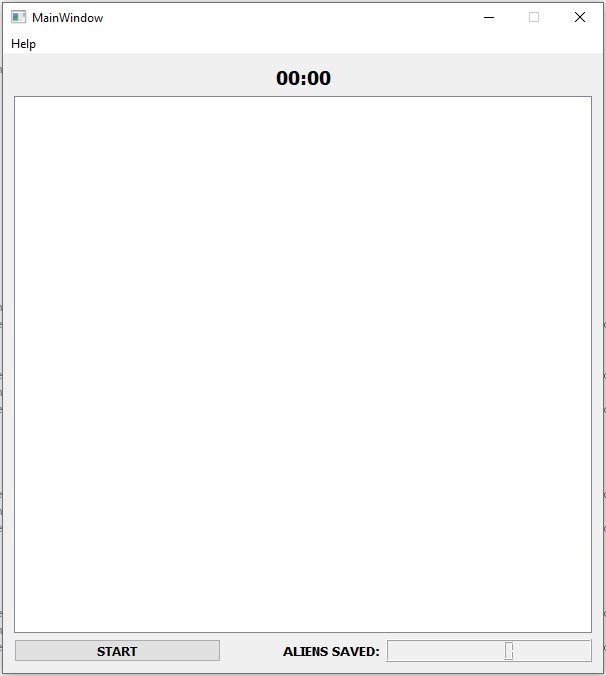
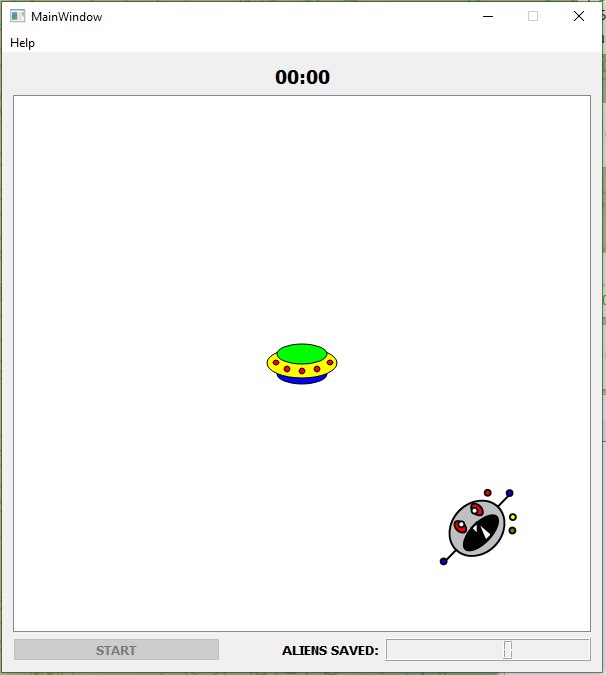
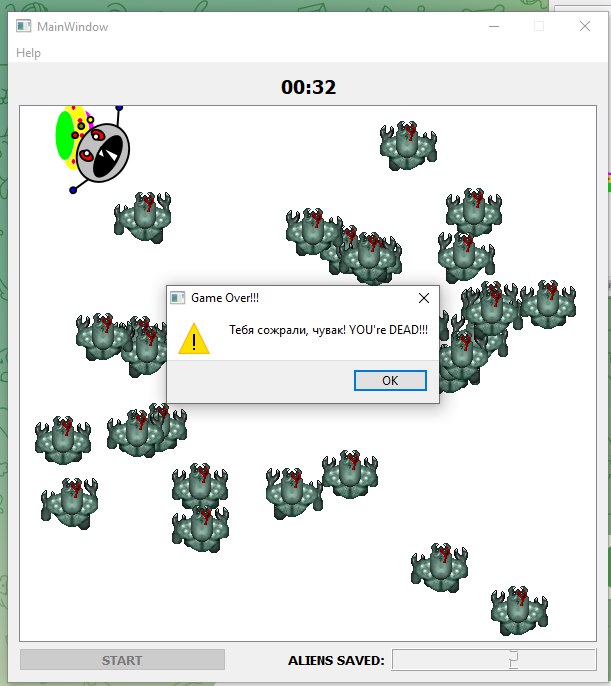


Рис.1. Исходный вид программы при запуске.

Рис. 2. Вид программы при, нажатий старт.Рис. 3. Вид программы спустя какое-то время и смерть.

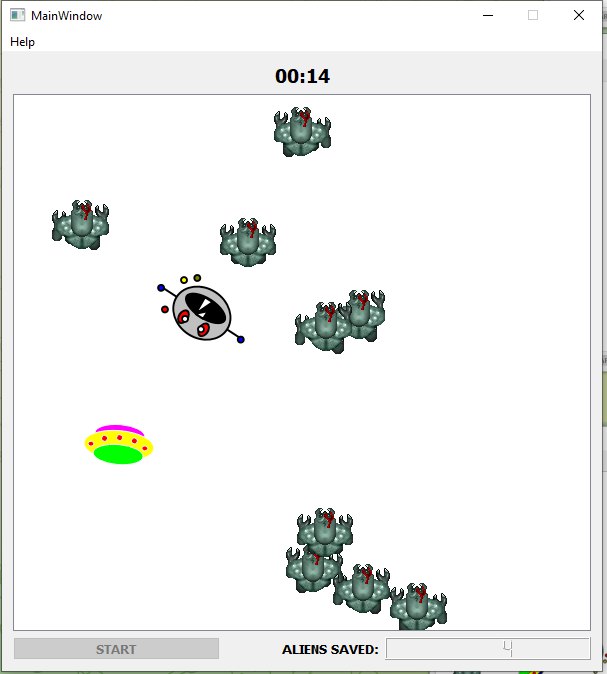


Рис. 4. Вид программы при паузе и собранных 4 пришельцев.

Заключение

В результате выполнения курсовой работы с использованием полученных знаний, была реализована «НЛО». Для этого были задействованы навыки объектно-ориентированного программирования на языке C++ и использования программного интерфейса QT creator. Готовый продукт удовлетворяет требованиям технического задания.

Список использованных источников

1. <https://vladnews.ru/2017-03-18/106392/vnezemnoy_razum>
2. [https://ru.bmstu.wiki/Qt\_](https://ru.bmstu.wiki/Qt_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F))
3. <https://doc.qt.io/>

**Приложение**

**Текст программы**

Программные файлы:

Заголовочные файлы:

- mainwindow.h;

Файлы реализации:

- main.cpp;

- mainwindow.cpp;

- menuclass.cpp;

- ufo.cpp;

- ufo\_killer.cpp

- ufo\_people.cpp

**Содержимое файлов mainwindow.h и mainwindow.cpp:**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QMessageBox>

#include <QGraphicsScene>

#include <QGraphicsItem>

#include <QShortcut>

#include <QDebug>

#include <QTime>

#include <QTimer>

#include <QElapsedTimer>

#include <QMediaPlaylist>

#include <QMediaPlayer>

#include <QMenu>

#include <QAction>

#include "ufo.h"

#include "ufo\_killer.h"

#include "ufo\_people.h"

#include "menuclass.h"

#define START\_GAME 0

#define STOP\_GAME 1

namespace Ui {

class MainWindow;

}

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = 0/\*nullptr\*/);

~***MainWindow***();

void **createMenu**();

void **createActions**();

public slots:

void **aboutInfo**();

void **Introduction**();

private:

Ui::MainWindow \*ui;

QGraphicsScene \*scene;

Ufo \*ufo;

QAction \*aboutAction;

QAction \*introAction;

QMenu \*fileMenu;

QTimer \*timer;

QTimer \*timerUfoPeopleCreator;

QTime m\_time;

QTimer m\_timer;

QElapsedTimer m\_eTimer;

QList<QGraphicsItem \*> ufo\_peoples;

double count;

Ufo\_killer \*ufo\_killer;

QShortcut \*pauseKey;

int gameState;

private slots:

void **slotDeleteUfoPeople**(QGraphicsItem \*item);

void **slotCreateUfoPeople**();

void **on\_pushButton\_clicked**();

void **slotGameOver**();

void **slotPause**();

void **updateTime**();

};

#endif

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

{

ui -> setupUi(this);

this -> resize(600, 640);

this -> setFixedSize(600, 640);

createActions();

createMenu();

scene = new QGraphicsScene();

ui -> graphicsView -> setScene(scene);

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

ui -> graphicsView -> setVerticalScrollBarPolicy(Qt::ScrollBarAlwaysOff);

ui -> graphicsView -> setHorizontalScrollBarPolicy(Qt::ScrollBarAlwaysOff);

scene -> setSceneRect(-250, -250, 500, 500);

timer = new QTimer();

timerUfoPeopleCreator = new QTimer();

gameState = STOP\_GAME;

pauseKey = new QShortcut(this);

pauseKey -> setKey(Qt::Key\_Space);

connect(pauseKey, &QShortcut::activated, this, &MainWindow::slotPause);

/\*

connect(timer, &QTimer::timeout, ufo, &Ufo::slotGameTimer);

timer -> start(1000/100);

timerUfoPeopleCreator = new QTimer();

connect(timerUfoPeopleCreator, &QTimer::timeout, this, &MainWindow::slotCreateUfoPeople);

timerUfoPeopleCreator -> start(1000);

connect(ufo, &Ufo::signalCheckItem, this, &MainWindow::slotDeleteUfoPeople);

\*/

}

MainWindow::~***MainWindow***()

{

delete ui;

}

void MainWindow::**createMenu**()

{

fileMenu = menuBar()->addMenu(tr("&Help"));

fileMenu -> addAction(aboutAction);

fileMenu -> addAction(introAction);

}

void MainWindow::**createActions**()

{

aboutAction = new QAction(tr("&About «UFO Rescue spaceship»"), this);

connect(aboutAction, SIGNAL(triggered()/\*clicked()\*//\*activated()\*/), this, SLOT( aboutInfo() ));

introAction = new QAction(tr("&Introductions. Help Contents"), this);

connect(introAction, SIGNAL(triggered()/\*clicked()\*//\*activated()\*/), this, SLOT( Introduction() ));

}

void MainWindow::**aboutInfo**()

{

AboutWindow \*info = new AboutWindow();

info -> *setVisible*(true);

}

void MainWindow::**Introduction**()

{

IntroductionWindow \*info = new IntroductionWindow();

info -> *setVisible*(true);

}

void MainWindow::**updateTime**()

{

m\_time=m\_time.addMSecs(m\_eTimer.restart());

ui->label\_2->setText(m\_time.toString("mm:ss"));

}

void MainWindow::**slotDeleteUfoPeople**(QGraphicsItem \*item)

{

foreach(QGraphicsItem \*ufopeople, ufo\_peoples)

{

if(ufopeople == item)

{

scene -> removeItem(ufopeople);

ufo\_peoples.removeOne(ufopeople);

delete ufopeople;

ui -> lcdNumber -> display(count++);

QMediaPlayer \*m\_player = new QMediaPlayer(this); QMediaPlaylist \*m\_playlist = new QMediaPlaylist(m\_player);

m\_player -> setPlaylist(m\_playlist);

m\_playlist -> addMedia(QUrl("qrc:/sound/schpok.wav"));

m\_playlist -> setPlaybackMode(QMediaPlaylist::CurrentItemOnce);

m\_player -> play();

}

}

}

void MainWindow::**slotCreateUfoPeople**()

{

qDebug() << "-> ufo create";

Ufo\_people \*ufopeople = new Ufo\_people();

scene -> addItem(ufopeople);

ufopeople -> setPos((qrand() % (251))\*((qrand()%2 == 1)?1 : -1),

(qrand() % (251))\*((qrand()%2 == 1)?1 : -1));

ufopeople -> setZValue(-1);

ufo\_peoples.append(ufopeople);

}

void MainWindow::**on\_pushButton\_clicked**()

{

qDebug() << "-> Start the Game...";

count = 0;

ui -> lcdNumber -> display(count);

ufo = new Ufo();

scene -> addItem(ufo);

ufo -> setPos(0,0);

m\_time = QTime(0,0);

connect(&m\_timer, SIGNAL(timeout()), this, SLOT(updateTime()));

m\_timer.start(1000);

m\_eTimer.start();

ufo\_killer = new Ufo\_killer(ufo);

scene -> addItem(ufo\_killer);

ufo\_killer -> setPos(240, 240);

connect(ufo\_killer, &Ufo\_killer::signalGameOver, this, &MainWindow::slotGameOver);

connect(timer, &QTimer::timeout, ufo, &Ufo::slotGameTimer);

timer -> start(1000 / 100);

connect(timerUfoPeopleCreator, &QTimer::timeout, this, &MainWindow::slotCreateUfoPeople);

timerUfoPeopleCreator -> start(1000);

connect(ufo, &Ufo::signalCheckItem, this, &MainWindow::slotDeleteUfoPeople);

ui -> pushButton -> setEnabled(false);

gameState = START\_GAME;

}

void MainWindow::**slotGameOver**()

{

qDebug() << "-> Game Over!...";

timer -> stop();

timerUfoPeopleCreator -> stop();

m\_timer.stop();

QMediaPlayer \*m\_player = new QMediaPlayer(this);

QMediaPlaylist \*m\_playlist = new QMediaPlaylist(m\_player);

m\_player -> setPlaylist(m\_playlist);

m\_playlist -> addMedia(QUrl("qrc:/sound/razgovor\_marsian.wav"));

m\_playlist -> setPlaybackMode(QMediaPlaylist::CurrentItemOnce);

m\_player -> play();

QMessageBox::warning(this, "Game Over!!!", "Тебя сожрали, чувак! YOU're DEAD!!!");

disconnect(timerUfoPeopleCreator, &QTimer::timeout, this, &MainWindow::slotCreateUfoPeople);

disconnect(ufo, &Ufo::signalCheckItem, this, &MainWindow::slotDeleteUfoPeople);

disconnect(ufo\_killer, &Ufo\_killer::signalGameOver, this, &MainWindow::slotGameOver);

ufo\_killer -> deleteLater();

ufo -> deleteLater();

m\_player -> stop();

foreach (QGraphicsItem \*ufopeople, ufo\_peoples)

{

scene -> removeItem(ufopeople);

ufo\_peoples.removeOne(ufopeople);

delete ufopeople;

}

ui -> pushButton -> setEnabled(true);

gameState = STOP\_GAME;

}

void MainWindow::**slotPause**()

{

qDebug() << "-> Pause...";

if(gameState == START\_GAME)

{

if(timer -> isActive())

{

timer -> stop();

m\_timer.stop();

timerUfoPeopleCreator -> stop();

ufo\_killer -> pause();

}

else

{

timer -> start(1000/100);

m\_timer.start(1000);

m\_eTimer.start();

timerUfoPeopleCreator -> start(1000);

ufo\_killer -> pause();

}

}

}

**Содержимое файлов ufo.cpp и ufo.h:**

#include "ufo.h"

Ufo::**Ufo**(QObject \*parent) : QObject(parent), QGraphicsItem()

{

angle = 0;

move = 1;

count\_move = 0;

setRotation(angle);

m\_player = new QMediaPlayer(this);

m\_playlist = new QMediaPlaylist(m\_player);

m\_player -> setPlaylist(m\_playlist);

m\_playlist -> addMedia(QUrl("qrc:/sound/polyot\_korablya.wav"));

m\_playlist -> setPlaybackMode(QMediaPlaylist::CurrentItemOnce);

}

Ufo::~***Ufo***()

{

}

QRectF Ufo::***boundingRect***() const

{

return QRectF(-40/\*x\*/,-50/\*y\*/,100/\*width\*/,70/\*height\*/);

}

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

{

if(move == 0){

painter->setPen(QPen(Qt::white,1));

painter->setBrush(Qt::magenta);

painter->drawEllipse(-25, 1, 50, 20);

}

else if(move == 1){

painter->setPen(QPen(Qt::black,1));

painter->setBrush(Qt::blue);

painter->drawEllipse(-25, 1, 50, 20);

}

else if(move == 2){

painter->setBrush(Qt::darkBlue);

painter->drawEllipse(-25, 1, 50, 20);

}

painter->setBrush(Qt::yellow);

painter->drawEllipse(-35/\*X\*/, -15/\*Y\*/, 70/\*width\*/, 30/\*height\*/);

painter->setBrush(Qt::green);

painter->drawEllipse(-25, -19, 50, 20);

painter->setBrush(Qt::red);

painter->drawEllipse(-29, -3, 6, 5);

painter->drawEllipse(25, -3, 6, 5);

painter->drawEllipse(12, 3, 6, 6);

painter->drawEllipse(-3, 5, 6, 6);

painter->drawEllipse(-18, 3, 6, 6);

Q\_UNUSED(option);

Q\_UNUSED(widget);

}

void Ufo::**slotGameTimer**()

{

if (GetAsyncKeyState(VK\_LEFT) ||

GetAsyncKeyState(VK\_RIGHT) ||

GetAsyncKeyState(VK\_UP) ||

GetAsyncKeyState(VK\_DOWN))

{

m\_player -> play();

if(GetAsyncKeyState(VK\_LEFT)){

angle -= 5;

setRotation(angle);

}

if(GetAsyncKeyState(VK\_RIGHT)){

angle += 5;

setRotation(angle);

}

if(GetAsyncKeyState(VK\_UP)){

setPos(mapToParent(0, -2));

}

if(GetAsyncKeyState(VK\_DOWN)){

setPos(mapToParent(0, 2));

}

count\_move++;

if(count\_move == 4){

move = 2;

update(QRectF(-40,-50,100,70));

}

else if(count\_move == 8){

move = 1;

update(QRectF(-40,-50,100,70));

}

else if(count\_move == 12){

move = 0;

update(QRectF(-40,-50,100,70));

}

else if(count\_move == 16){

move = 1;

update(QRectF(-40,-50,100,70));

count\_move = 0;

}

}

QList<QGraphicsItem \*> foundItems = scene() -> items(QPolygonF()

<< mapToScene(0, 0)

<< mapToScene(-20, -20)

<< mapToScene(20, -20));

foreach (QGraphicsItem \*item, foundItems)

{

if (item == this) continue;

emit signalCheckItem(item);

}

if(this->x() - 10 < -250){

this->setX(-240);

}

if(this->x() + 10 > 250){

this->setX(240);

}

if(this->y() - 10 < -250){

this->setY(-240);

}

if(this->y() + 10 > 250){

this->setY(240);

}

}

#ifndef UFO\_H

#define UFO\_H

#include <QObject>

#include <QGraphicsItem>

#include <QPainter>

#include <QGraphicsScene>

#include <QMediaPlaylist>

#include <QMediaPlayer>

#include <windows.h>

class Ufo : public QObject, public QGraphicsItem

{

Q\_OBJECT

Q\_INTERFACES(QGraphicsItem)

public:

explicit Ufo(QObject \*parent = 0 /\*nullptr\*/);

~***Ufo***();

signals:

void **signalCheckItem**(QGraphicsItem \*item);

public slots:

void **slotGameTimer**();

protected:

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

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

private:

qreal angle;

int move;

int count\_move;

QMediaPlayer \*m\_player;

QMediaPlaylist \*m\_playlist;

};

#endif

**Содержимое файлов ufo\_killer.h :**

#ifndef UFO\_KILLER\_H

#define UFO\_KILLER\_H

#include <QGraphicsItem>

#include <QGraphicsScene>

#include <QObject>

#include <QTimer>

#include <QDebug>

#include <QPainter>

class Ufo\_killer : public QObject, public QGraphicsItem

{

Q\_OBJECT

Q\_INTERFACES(QGraphicsItem)

public:

explicit Ufo\_killer(QGraphicsItem \*target, QObject \*parent = 0/\*nullptr\*/);

~***Ufo\_killer***();

void **pause**();

signals:

void **signalGameOver**();

public slots:

protected:

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

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

private:

qreal angle;

int move;

int count\_move;

QTimer \*timer;

QGraphicsItem \*target;

private slots:

void **slotGameTimer**();

};

#endif

**Содержимое файлов ufo\_people.h :**

#ifndef UFO\_PEOPLE\_H

#define UFO\_PEOPLE\_H

#include <QObject>

#include <QGraphicsItem>

#include <QTimer>

#include <QPixmap>

#include <QPainter>

#include <QDebug>

class Ufo\_people : public QObject, public QGraphicsItem

{

Q\_OBJECT

Q\_INTERFACES(QGraphicsItem)

public:

explicit Ufo\_people(QObject \*parent = 0/\*nullptr\*/);

signals:

public slots:

private slots:

void **nextFrame**();

private:

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

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

private:

QTimer \*timer;

QPixmap \*spriteImage;

int currentFrame;

};

#endif

**Содержимое файлов menuclass.h :**

#ifndef MENUCLASS\_H

#define MENUCLASS\_H

#include <QWidget>

#include <QLabel>

#include <QVBoxLayout>

#include <QDebug>

#include <QMediaPlaylist>

#include <QMediaPlayer>

class AboutWindow : public QWidget

{

Q\_OBJECT

public:

explicit AboutWindow(QWidget \*parent = 0/\*nullptr\*/);

signals:

void **activated**(int i);

public slots:

private:

QPixmap \*aboutWindowPix;

QLabel \*aboutWindowLabel;

QLabel \*aboutWindowPixLabel;

};

class IntroductionWindow : public QWidget

{

Q\_OBJECT

public:

explicit IntroductionWindow(QWidget \*parent = 0/\*nullptr\*/);

signals:

void **activated**(int i);

public slots:

private:

QPixmap \*introductionWindowPix;

QLabel \*introductionWindowLabel;

QLabel \*introductionWindowPixLabel;

};

#endif

**Содержимое файлов main.cpp:**

#include "mainwindow.h"

#include <QApplication>

int main(int argc, char \*argv[])

{

QApplication a(argc, argv);

MainWindow w;

w.show();

return a.exec();}