Федеральное агентство связи

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

Кафедра прикладной математики и кибернетики

Отчет по лабораторной работе

по дисциплине «Визуальное программирование и человеко-машинное взаимодействие»

Лабораторная работа №7

Выполнил: студент 3 курса группы ИП-811 Мироненко К. А

Проверил: доцент кафедры ПМиК Мерзлякова Е. Ю.

Оглавление

[1. Постановка задачи 3](#_Toc52384053)

[2. Примеры работы программы 4](#_Toc52384054)

[*Приложение* Листинг 6](#_Toc52384055)

# Постановка задачи

**Цель:** научиться работать с графикой в Qt.

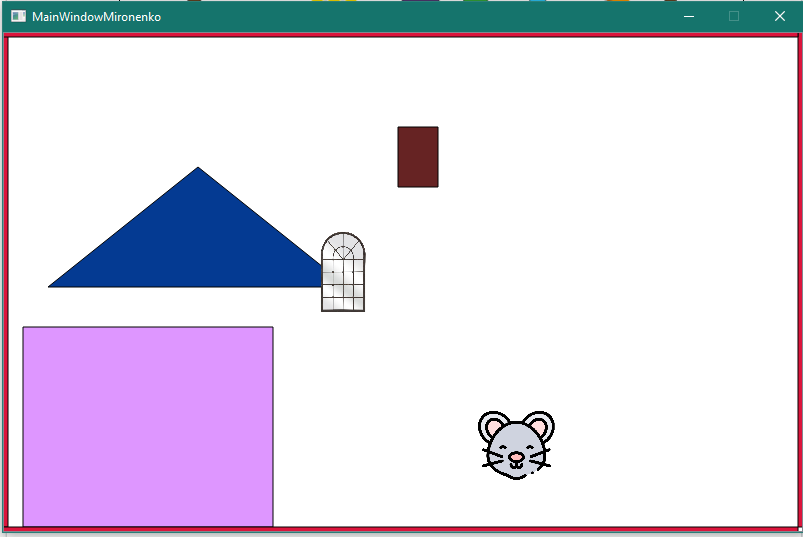
**Задание:**

1. Создать графическую сцену.
2. Поместить на сцену различные элементы для составления картинки по варианту. Обязательно использовать и геометрические фигуры, и картинки. Они должны перемещаться с помощью мыши.
3. Ограничить края сцены «стенами» в виде каких-либо элементов.
4. Поместить на сцену движущийся элемент по заданию. Он должен перемещаться с заданной скоростью, сталкиваться со «стенами» и фигурами на сцене. Используйте таймер и функцию обнаружения столкновений.

**Вариант:**

2) Дом и движущаяся мышь

# Примеры работы программы



(Начальное положение элементов)



(Положение элементов после составления картинки)

# *Приложение* Листинг

**main.cpp**

#include "mainwindowmironenko.h"

#include <QApplication>

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

{

QApplication a(argc, argv);

MainWindowMironenko w;

w.setFixedSize(800,500);

w.show();

return a.exec();

}

**mainwindow.h**

#ifndef MAINWINDOWMIRONENKO\_H

#define MAINWINDOWMIRONENKO\_H

#include <QMainWindow>

#include "graphicsmironenko.h"

namespace Ui {

class MainWindowMironenko;

}

class MainWindowMironenko : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindowMironenko(QWidget \*parent = nullptr);

~MainWindowMironenko();

private:

Ui::MainWindowMironenko \*ui;

};

#endif // MAINWINDOWMIRONENKO\_H

**mainwindow.cpp**

#include "mainwindowmironenko.h"

#include "ui\_mainwindowmironenko.h"

MainWindowMironenko::MainWindowMironenko(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindowMironenko)

{

ui->setupUi(this);

GraphicsMironenko\* scene = new GraphicsMironenko;

ui->graphicsView->setScene(scene);

}

MainWindowMironenko::~MainWindowMironenko()

{

delete ui;

}

**graphicsmironenko.h**

#ifndef GRAPHICSMIRONENKO\_H

#define GRAPHICSMIRONENKO\_H

#include <QWidget>

#include <QGraphicsScene>

#include <QGraphicsItem>

#include <QGraphicsEllipseItem>

#include <QTime>

#include <QTimer>

class GraphicsMironenko : public QGraphicsScene

{

Q\_OBJECT

public:

GraphicsMironenko(QObject\* parent= nullptr);

QGraphicsItem\* itemCollidesWith(QGraphicsItem\* item);

void Init();

private:

QGraphicsRectItem\* walls[4];

QGraphicsPixmapItem\* mouse;

int speed;

double dx, dy;

public slots:

void MoveMouse();

};

#endif // GRAPHICSMIRONENKO\_H

**graphicsmironenko.cpp**

#include "graphicsmironenko.h"

void GraphicsMironenko::Init(){

QPolygon poligoneTube;

poligoneTube << QPoint(0, 0) << QPoint(40, 0)<<QPoint(40, 60)<<QPoint(0, 60);

QGraphicsPolygonItem\* tube = addPolygon(poligoneTube, QPen(Qt::black), QBrush(QColor(102, 35, 35)));

tube->setPos(400, 100);

tube->setFlags(QGraphicsItem::ItemIsMovable);

tube->setData(0, "House");

QPolygon poligoneHouse;

poligoneHouse << QPoint(0, 0) << QPoint(250, 0)<<QPoint(250, 200)<<QPoint(0, 200);

QGraphicsPolygonItem\* house = addPolygon(poligoneHouse, QPen(Qt::black), QBrush(QColor(222, 150, 255)));

house->setPos(25, 300);

house->setFlags(QGraphicsItem::ItemIsMovable);

house->setData(0, "House");

QPolygon poligoneRoof;

poligoneRoof << QPoint(-150,60) << QPoint(0,-60) << QPoint(150,60) ;

QGraphicsPolygonItem\* roof = addPolygon(poligoneRoof, QPen(Qt::black), QBrush(QColor(4,58,146)));

roof->setPos(200, 200);

roof->setFlags(QGraphicsItem::ItemIsMovable);

roof->setData(0, "House");

QPixmap newImage;

if (!newImage.load(QStringLiteral(":/images/window.png"))) {

return;

}

QGraphicsPixmapItem\* window[2];

for(int i=0; i<2; i++){

window[i] = addPixmap(newImage);

window[i]->setPos(300,200);

window[i]->setScale(0.15);

window[i]->setData(0, "House");

window[i]->setFlags(QGraphicsItem::ItemIsMovable);

};

if (!newImage.load(QStringLiteral(":/images/mouse.png"))) {

return;

}

mouse = addPixmap(newImage);

mouse->setScale(0.15);

mouse->setPos(500, 400);

mouse->setData(0, "mouse");

walls[0] = addRect(QRectF(0,0,10,500),QPen(Qt::black), QBrush(QColor(220, 20, 60)));

walls[1] = addRect(QRectF(800,0,10,500), QPen(Qt::black), QBrush(QColor(220, 20, 60)));

walls[2] = addRect(QRectF(0,0,800,10), QPen(Qt::black), QBrush(QColor(220, 20, 60)));

walls[3] = addRect(QRectF(0,500,800,10), QPen(Qt::black), QBrush(QColor(220, 20, 60)));

for(int i=0; i<4; i++)

walls[i]->setData(0,"Wall");

}

void GraphicsMironenko::MoveMouse(){

QTransform transform = mouse->transform();

transform.translate(dx,dy);

mouse->setTransform(transform);

QGraphicsItem\* barrier = itemCollidesWith(mouse);

transform=mouse->transform();

if(barrier){

if(barrier->data(0)=="Wall" || barrier->data(0)=="House"){

dx=-dx;

dy=-dy;

}

}

}

GraphicsMironenko::GraphicsMironenko(QObject\* parent):QGraphicsScene (parent)

{

Init();

dx = 1;

dy = 1;

speed = 1;

qsrand(QTime(0,0,0).secsTo(QTime::currentTime()));

QTimer\* timer = new QTimer(this);

connect(timer, SIGNAL(timeout()), this, SLOT(MoveMouse()));

timer->start(10);

}

QGraphicsItem \* GraphicsMironenko::itemCollidesWith(QGraphicsItem \*item)

{

QList<QGraphicsItem \*> collisions = collidingItems(item);

foreach (QGraphicsItem \*it, collisions){

if(it == item)

continue;

return it;

}

return nullptr;

}