МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Балтийский государственный технический университет «ВОЕНМЕХ» им. Д.Ф. Устинова»
(БГТУ «ВОЕНМЕХ» им. Д.Ф. Устинова)

Кафедра О7 Информационные системы и программная инженерия
шифр наименование кафедры, по которой выполняется работа

Дисциплина Визуальное программирование
наименование дисциплины

ПРАКТИЧЕСКАЯ РАБОТА № 5

Объектно-ориентированный подход к созданию графических сцен

Язык: C++ (вместе с Qt)

Вариант №10

ОБУЧАЮЩИЙСЯ

группы О726Б

Махов Н.М.
подпись фамилия и инициалы

дата сдачи

ПРОВЕРИЛ

преподаватель каф. О7
ученая степень, ученое звание, должность

Устиновский Г.С.
подпись фамилия и инициалы

Оценка / балльная оценка

дата проверки

г. Санкт-Петербург
2024 г.

СОДЕРЖАНИЕ

1 Цель работы и постановка задачи

1.1 Цель работы

Научиться применять объектно-ориентированный подход при создании графических сцен и использовать динамическое создание, удаление и перемещение различных графических объектов.

1.2 Постановка задачи

В данной работе необходимо написать программу в соответствие с индивидуальным вариантом. Все варианты описывают набор объектов для графической сцены. Если в варианте не сказано иного, то должны использоваться объекты трёх разных классов, определяющих отображение объекта и реакцию на события.

Часть объектов являются стандартными элементами управления, в вариантах даны названия виджетов Qt.

1.3 Вариативная часть задания

Вариант №10.

Написать программу, позволяющую разместить на графической сцене произвольное число объектов следующих трёх типов:

1) текст, предварительно введённый пользователем (для каждого объекта свой);

2) квадрат одного из трёх цветов: синий кадетский, тёмный лосось, помидор по выбору пользователя;

3) изображение одного из трёх греческих богов: Зевса, Посейдона или Аида.

Должно быть предусмотрено перемещение добавляемых объектов путём перетаскивания их левой кнопкой мыши.

Также должен быть представлен список из всех добавленных элементов, с возможностью выбора и удаления любого из них, а также изменения бога, текста или цвета.

С помощью фильтра событий, должна быть добавлена возможность выбирать объект в списке при нажатии на него правой кнопкой мыши на сцене.

2 Реализация

2.1 Содержание файла main.cpp

```
#include "customitem.h"

int main(int argc, char *argv[]) {
    QApplication app(argc, argv);

    QWidget mainWindow;
    QVBoxLayout *layout = new QVBoxLayout(&mainWindow);

    QGraphicsScene scene;
    QGraphicsView view(&scene);
    view.setRenderHint(QPainter::Antialiasing);
    scene.setSceneRect(0, 0, 800, 600);
    layout->addWidget(&view);

    QListWidget *itemList = new QListWidget();
    layout->addWidget(itemList);

    QPushButton *addButton = new QPushButton("Добавить
объект");
    QPushButton *removeButton = new QPushButton("Удалить
объект");
    QPushButton *editButton = new QPushButton("Редактировать
объект");
    layout->addWidget(addButton);
    layout->addWidget(removeButton);
    layout->addWidget(editButton);
```

```
    QObject::connect(addButton, &QPushButton::clicked, [&]()
{
        bool ok;
        QString type = QInputDialog::getItem(nullptr, "Тип
объекта", "Выберите тип объекта:", {"Текст", "Квадрат",
"Изображение"}, 0, false, &ok);
        if (!ok) return;

        if (type == "Текст") {
            QString text = QInputDialog::getText(nullptr,
"Введите текст", "Текст для объекта:", QLineEdit::Normal, "",
&ok);
            if (ok && !text.isEmpty()) {
                CustomItem *item = new CustomItem(text);
                item->setPos(0, scene.items().count() * 60);
                scene.addItem(item);
                itemList->addItem(new QListWidgetItem(text));
            }
        } else if (type == "Квадрат") {
            QColor color = QColorDialog::getColor(Qt::green,
nullptr, "Выберите цвет квадрата");
            if (color.isValid()) {
                CustomItem *item = new CustomItem(color);
                item->setPos(100,   scene.items().count()   *
60);
                scene.addItem(item);
                itemList->addItem(new
QListWidgetItem("Квадрат"));
            }
```

```
        } else if (type == "Изображение") {
            QString fileName = QFileDialog::getOpenFileName(nullptr, "Выберите изображение", FILEPATH, "Images (*.png *.jpg *.bmp)");
            if (!fileName.isEmpty()) {
                CustomItem *item = new CustomItem(fileName, true);
                item->setPos(200, scene.items().count() * 60);
                scene.addItem(item);
                itemList->addItem(new QListWidgetItem("Изображение"));
            }
        }
    });

    QObject::connect(removeButton, &QPushButton::clicked, [&]() {
        QList<QGraphicsItem*> selectedItems = scene.selectedItems();
        for (QGraphicsItem* item : selectedItems) {
            scene.removeItem(item);
            delete item; // Освобождаем память
        }
        // Удаление из списка
        QList<QListWidgetItem*> selectedListItems = itemList->selectedItems();
        for (QListWidgetItem* listItem : selectedListItems) {
            delete listItem;
```

```
        }
    });
    QObject::connect(editButton, &QPushButton::clicked, [&]()
{
        QListWidgetItem*     currentItem     =     itemList-
>currentItem();
        if (!currentItem) return;

        int index = itemList->row(currentItem);
        if (index < scene.items().count()) {
            QGraphicsItem*           graphicsItem           =
scene.items().at(index);
            if          (auto          customItem          =
dynamic_cast<CustomItem*>(graphicsItem)) {
                if (customItem->isText) {
                    QString             newText             =
QInputDialog::getText(nullptr, "Редактировать текст", "Новый
текст:", QLineEdit::Normal, customItem->text);
                    if (!newText.isEmpty()) {
                        customItem->text = newText;
                        currentItem->setText(newText);
                        customItem->update();
                    }
                }
                else if (customItem->isColor) {
                    QColor             newColor             =
QColorDialog::getColor(customItem->color, nullptr, "Выберите
новый цвет");
                    if (newColor.isValid()) {
```

```
                        customItem->color = newColor;
                        customItem->update();
                    }
                }
                view.update();
            }
        }
    });


    QObject::connect(itemList,     &QListWidget::itemClicked,
[&](QListWidgetItem* item) {
        int index = itemList->row(item);
        if (index < scene.items().count()) {
            QGraphicsItem*           graphicsItem           =
scene.items().at(index);
            graphicsItem->setSelected(true);
        }
    });

    mainWindow.setWindowTitle("Динамическая сцена");
    mainWindow.resize(800, 600);
    mainWindow.show();

    return app.exec();
}
```

2.2 Содержание файла customitem.cpp

```
#include "customitem.h"
#include <QFile>
```

```
#include <QPainter>

CustomItem::CustomItem(QString text)
    :     text(text),     isText(true),     isColor(false),
isImage(false) {
    setFlag(QGraphicsItem::ItemIsMovable);
    setFlag(QGraphicsItem::ItemIsSelectable);
}

CustomItem::CustomItem(QColor color)
    :     color(color),     isText(false),     isColor(true),
isImage(false) {
    setFlag(QGraphicsItem::ItemIsMovable);
    setFlag(QGraphicsItem::ItemIsSelectable);
}

CustomItem::CustomItem(QString imagePath, bool isImage)
    : isText(false), isColor(false), isImage(true) {
    if (QFile::exists(imagePath)) {
        image.load(imagePath);
        image = image.scaled(100, 100, Qt::KeepAspectRatio,
Qt::SmoothTransformation);
    } else {
        qWarning() << "Файл не найден:" << imagePath;
    }
    setFlag(QGraphicsItem::ItemIsMovable);
    setFlag(QGraphicsItem::ItemIsSelectable);
}
QRectF CustomItem::boundingRect() const {
```

```
    if (isText) {
        return QRectF(-5, -5, 110, 60);
    } else if (isColor) {
        return QRectF(-10, -10, 70, 70);
    } else if (isImage) {
        return QRectF(-10, -10, image.width() + 20,
image.height() + 20);
    }
    return QRectF();
}

void CustomItem::paint(QPainter *painter, const
QStyleOptionGraphicsItem *option, QWidget *widget) {
    Q_UNUSED(option);
    Q_UNUSED(widget);
    if (isText) {
        painter->drawText(boundingRect(), Qt::AlignCenter,
text);
    } else if (isColor) {
        painter->fillRect(boundingRect().adjusted(5, 5, -5, -
5), color);
    } else if (isImage) {
        painter->drawImage(0, 0, image);
    }

    // Рисуем выделение с закругленными углами
    if (isSelected()) {
        QPen pen(Qt::cyan, 2, Qt::DashDotDotLine);
        painter->setPen(pen);
```

```
        painter->setBrush(Qt::NoBrush);
        painter->drawRoundedRect(boundingRect(), 10, 10);
    }
}
```

2.3 Содержание файла customitem.h

```
#ifndef CUSTOMITEM_H
#define CUSTOMITEM_H

#include <QGraphicsItem>
#include <QPainter>
#include <QImage>
#include <QColor>
#include <QApplication>
#include <QGraphicsView>
#include <QGraphicsScene>
#include <QInputDialog>
#include <QColor>
#include <QColorDialog>
#include <QFileDialog>
#include <QPushButton>
#include <QVBoxLayout>
#include <QWidget>
#include <QListWidget>
#include <QListWidgetItem>
#include <QMessageBox>

#define                                                    FILEPATH
"/Users/nikitamakhov/Documents/pr5Ready/icons"
```

```
class CustomItem : public QGraphicsItem {
public:
    CustomItem(QString imagePath, bool isImage);
    CustomItem(QString text);            // Для текста
    CustomItem(QColor color);            // Для цвета

    QRectF boundingRect() const override;
    void        paint(QPainter        *painter,        const
QStyleOptionGraphicsItem *option, QWidget *widget) override;

    QString text;
    QColor color;
    QImage image;
    bool isText;
    bool isColor;
    bool isImage;
};

#endif // CUSTOMITEM_H
```

3 Демонстрация работы программы

При запуске программы пользователю показывается главное окно, показанное на рисунке 1. Пользователь может добавить три типа объектов.

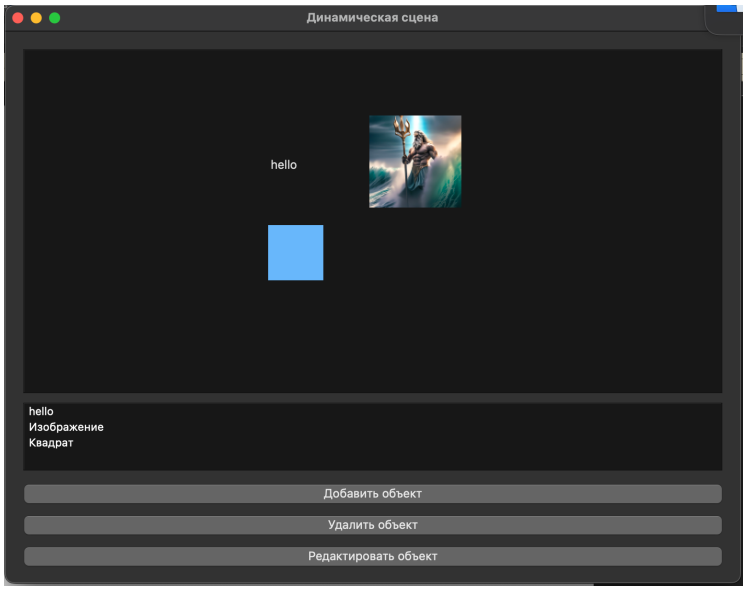

Рисунок 1 — Запуск программы и объекты разных типов

Можно добавить текстовое поле, заполнить его произвольным текстом и менять положение, после добавления на динамическую сцену можно отредактировать, результат представлен на рисунке 2.

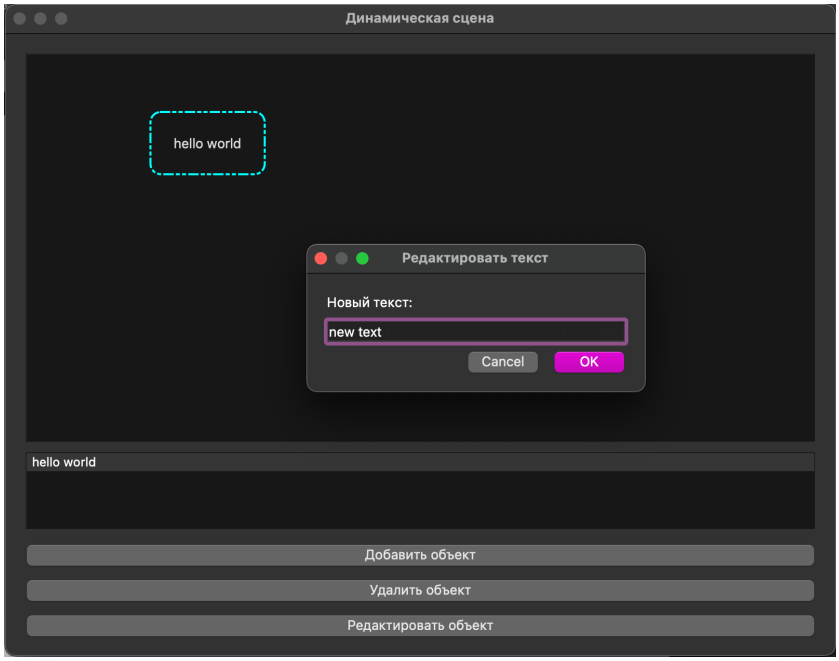

Рисунок 2 — Редактирование объекта типа текст

Можно добавить объект типа квадрат, выбрать его цвет, при необходимости изменить, результат представлен на рисунке 3.

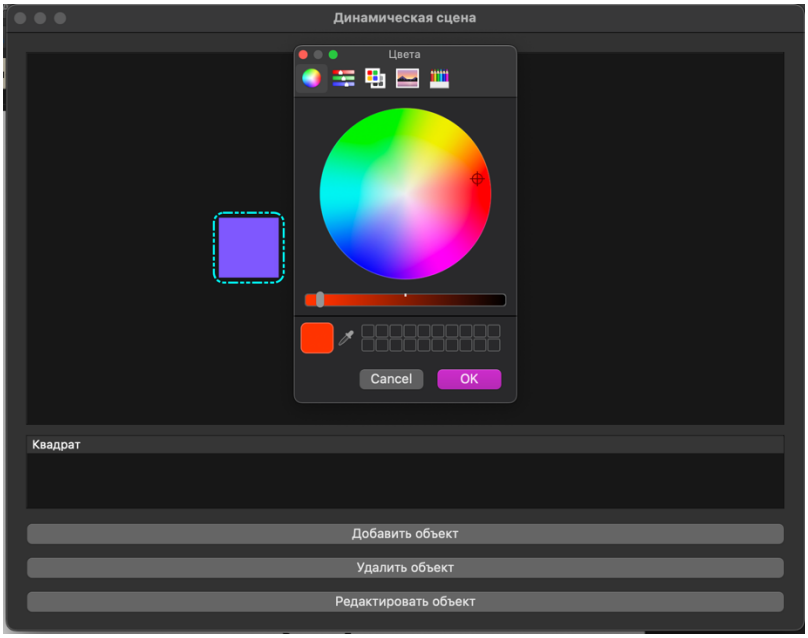

Рисунок 3 — Редактирование объектов типа квадрат

Можно добавить изображения богов: Аид, Посейдон, Зевс. Результат представлен на рисунке 4.

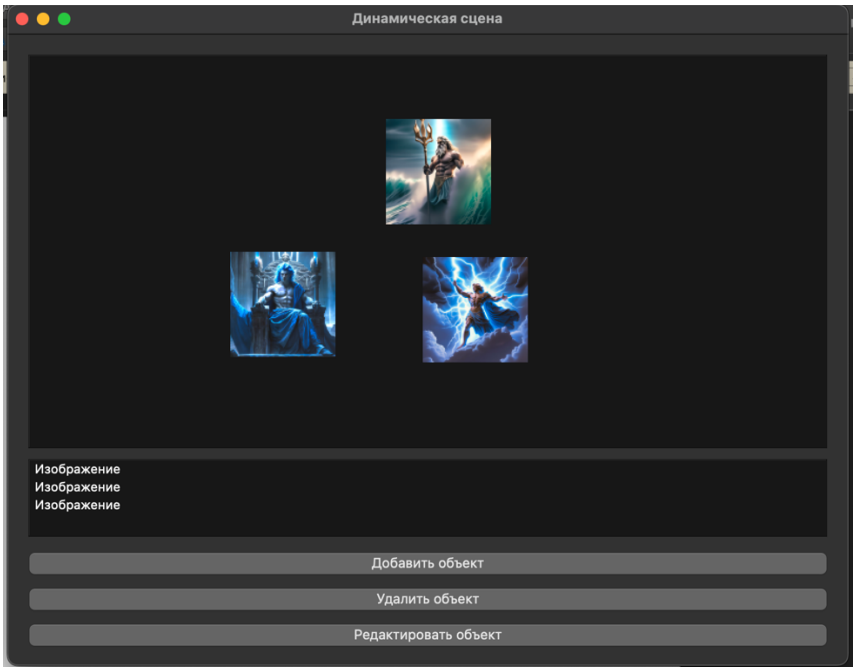

Рисунок 4 — Изображения богов

Объекты, можно выделять, выделять через список, двигать, удалять, результат удаления изображения представлен на рисунке 5.

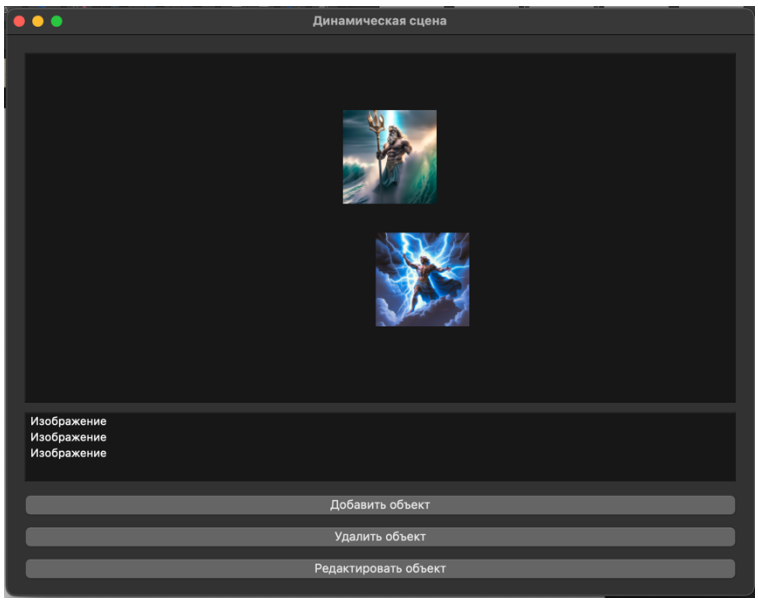

Рисунок 5 — Удаление объектов