ЛАБОРАТОРНАЯ РАБОТА №3

«ИССЛЕДОВАНИЕ СПОСОБОВ ПОСТРОЕНИЯ ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ НА БАЗЕ QTWIDGETS»

1. Цель работы

Изучить основные методики создания графического пользовательского интерфейса с использованием виджетов QtWidgets. Приобрести навыки разработки интерфейса пользователя для приложений на основе фреймворка Qt.

2. Постановка задачи

2.1. Создать проект Qt Gui Application.

2.2. Разработать графический интерфейс приложения согласно варианту задания.

2.3. Используя механизм сигналов и слотов, добавить функционал согласно варианту задания.

3. Вариант задания

|  |  |  |  |
| --- | --- | --- | --- |
| № Варианта | Тип формы | Функционал | Метод разработки |
| 13 | 3 | 1 | Qt designer |

Тип форм: Форма отзыва (имя (Line Edit), e-mail (Line Edit), оценка (Radio Button), текст отзыва (Text Edit));

Функционал: 1) Валидация (Наличие текста в полях, соответствие типам вводимых данных).

4. Ход работы

Был разработан графический интерфейс приложения согласно варианту задания. Макет интерфейса в Qt Designer представлен на рисунке 1. На рисунке 2 изображено собранное приложение.

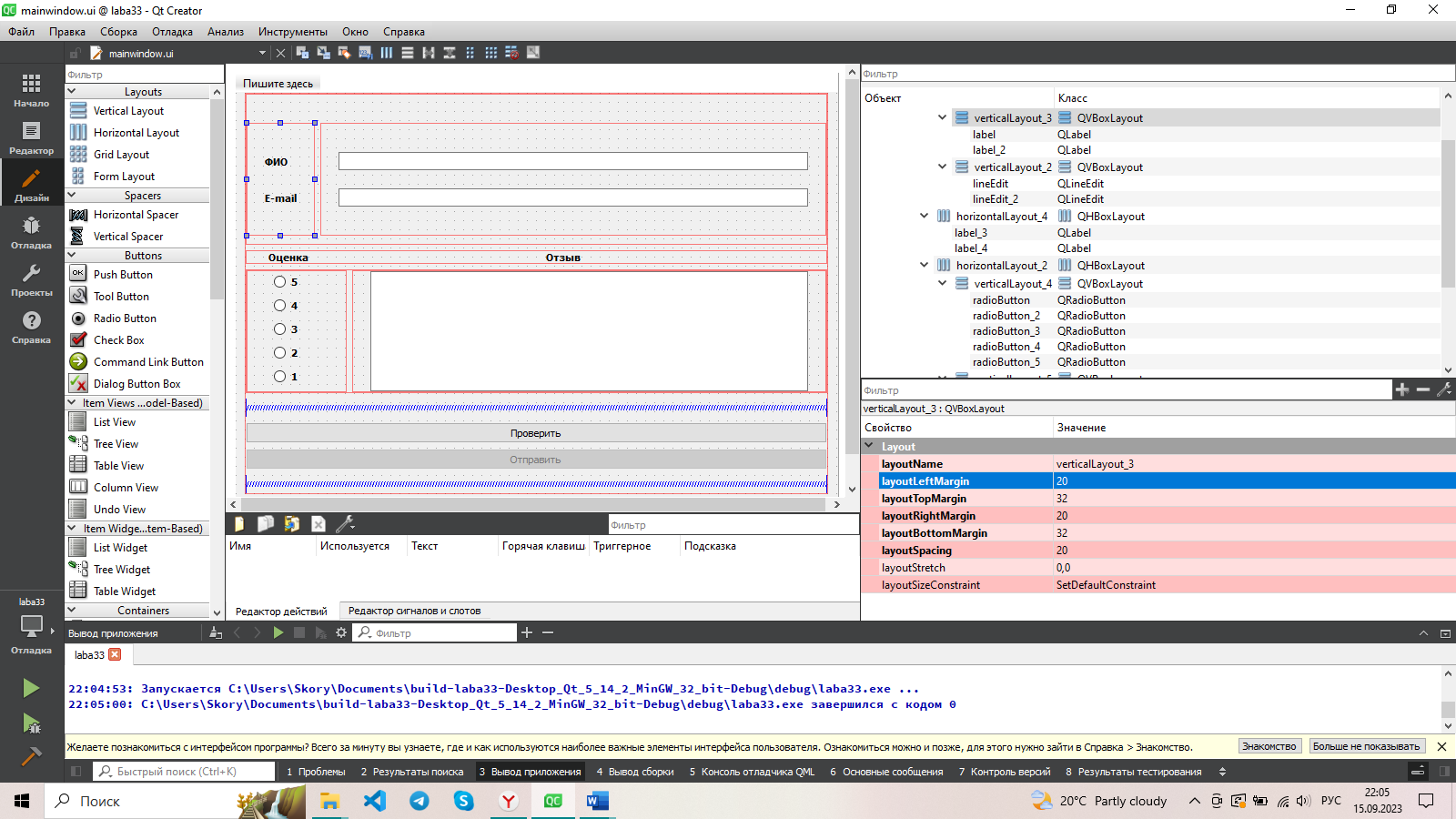


Рисунок 1 - Макет интерфейса в Qt Designer

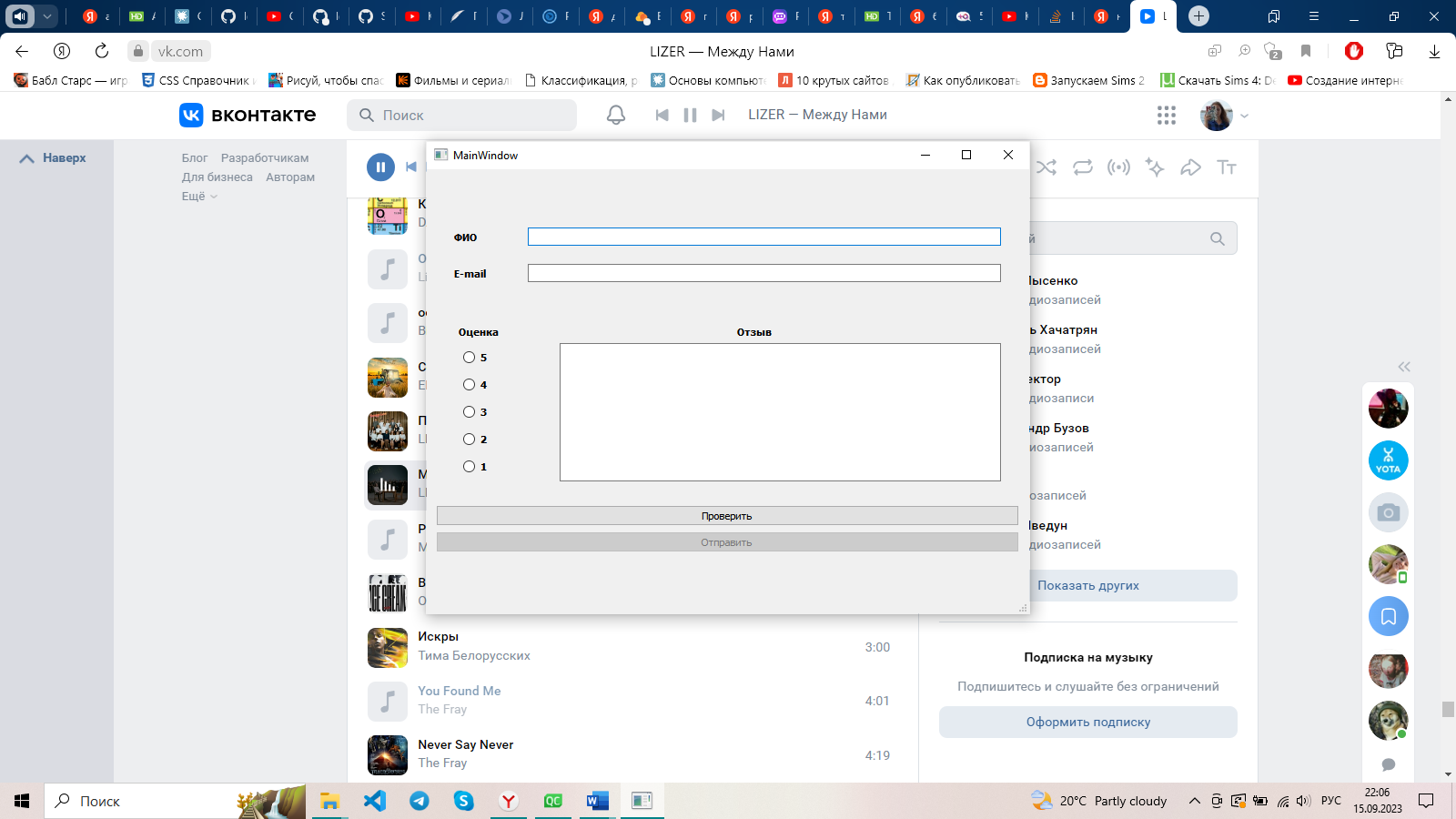


Рисунок 2 – Окно приложения

С использованием механизма сигналов и слотов была добавлена валидация форм. В листинге 1 представлен полный код файла mainwindow.cpp, в котором функция validate() отвечает за валидацию всех форм.

Листинг 1 – Код mainwindow.cpp

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <QLineEdit>

#include <QPushButton>

#include <QRegExpValidator>

#include <QHBoxLayout>

#include <QButtonGroup>

MainWindow::**MainWindow**(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

connect(ui->pushButton, SIGNAL(clicked()), this,

SLOT(validate()));

}

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

{

delete ui;

}

void MainWindow::**validate**()

{

//Задание формы ФИО

QRegExp nameRegExp("^[А-Яа-яЁё]+\\s[А-Яа-яЁё]+\\s[А-Яа-яЁё]+$");

//берем содержимое строки ФИО

QString isText = ui->lineEdit\_2->text(); //слово из нашей строки для ввода

// сохраняем значение в переменной

bool d = nameRegExp.exactMatch(isText);

//Задание формы e-mail

QRegExp emailRegExp("^[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,}$");

//берем содержимое строки e-mail

isText = ui->lineEdit->text();

// сохраняем значение в переменной

bool a = emailRegExp.exactMatch(isText);

//сохраняем в переменной то, что вытащим из textEdit

QString is = ui->textEdit->toPlainText();

//Создание группы из радио-батонов

QButtonGroup \*buttonGroup = new QButtonGroup(this);

buttonGroup->addButton(ui->radioButton);

buttonGroup->addButton(ui->radioButton\_2);

buttonGroup->addButton(ui->radioButton\_3);

buttonGroup->addButton(ui->radioButton\_4);

buttonGroup->addButton(ui->radioButton\_5);

//если будут несоответствия, то вылетают сообщения и кнопка блюрится

if (d==false || a==false || is=="" || buttonGroup->checkedButton() == nullptr){

QMessageBox msgBox;

if(d==false){

msgBox.setText("Ваше ФИО не соответствует стандарту Нурсултана Керимбаева");

msgBox.setStandardButtons(QMessageBox::Ok);

msgBox.*exec*();

ui->pushButton\_1->setDisabled(true);

}

else{

if(a==false){

msgBox.setText("Ваш e-mail не соответствует стандарту Нурсултана Керимбаева");

msgBox.setStandardButtons(QMessageBox::Ok);

msgBox.*exec*();

ui->pushButton\_1->setDisabled(true);

}

else{

if(is==""){

msgBox.setText("Ваш отзыв не соответствует стандарту Нурсултана Керимбаева");

msgBox.setStandardButtons(QMessageBox::Ok);

msgBox.*exec*();

ui->pushButton\_1->setDisabled(true);

}

else{

msgBox.setText("Выберете оценку от 1 до 5 стандарту Нурсултана Керимбаева");

msgBox.setStandardButtons(QMessageBox::Ok);

msgBox.*exec*();

ui->pushButton\_1->setDisabled(true);

}

}

}

}

else{

//если всё ок, то разблюрить кнопку "Отправить"

ui->pushButton\_1->setDisabled(false);

}

}

Валидация форм была проверена с помощью разработанных примеров, приведенных на рисунках 3-.

На рисунке 3 видно, что, если поле ФИО пустое, пользователю выскакивает сообщение о необходимости заполнить поле. А далее, на рисунке 4, видно, что недостаточно просто заполнить форму – необходимо сделать это корректно. Данное поле допускает лишь три буквенных выражения на русском языке, разделенных двумя пробелами соответственно.

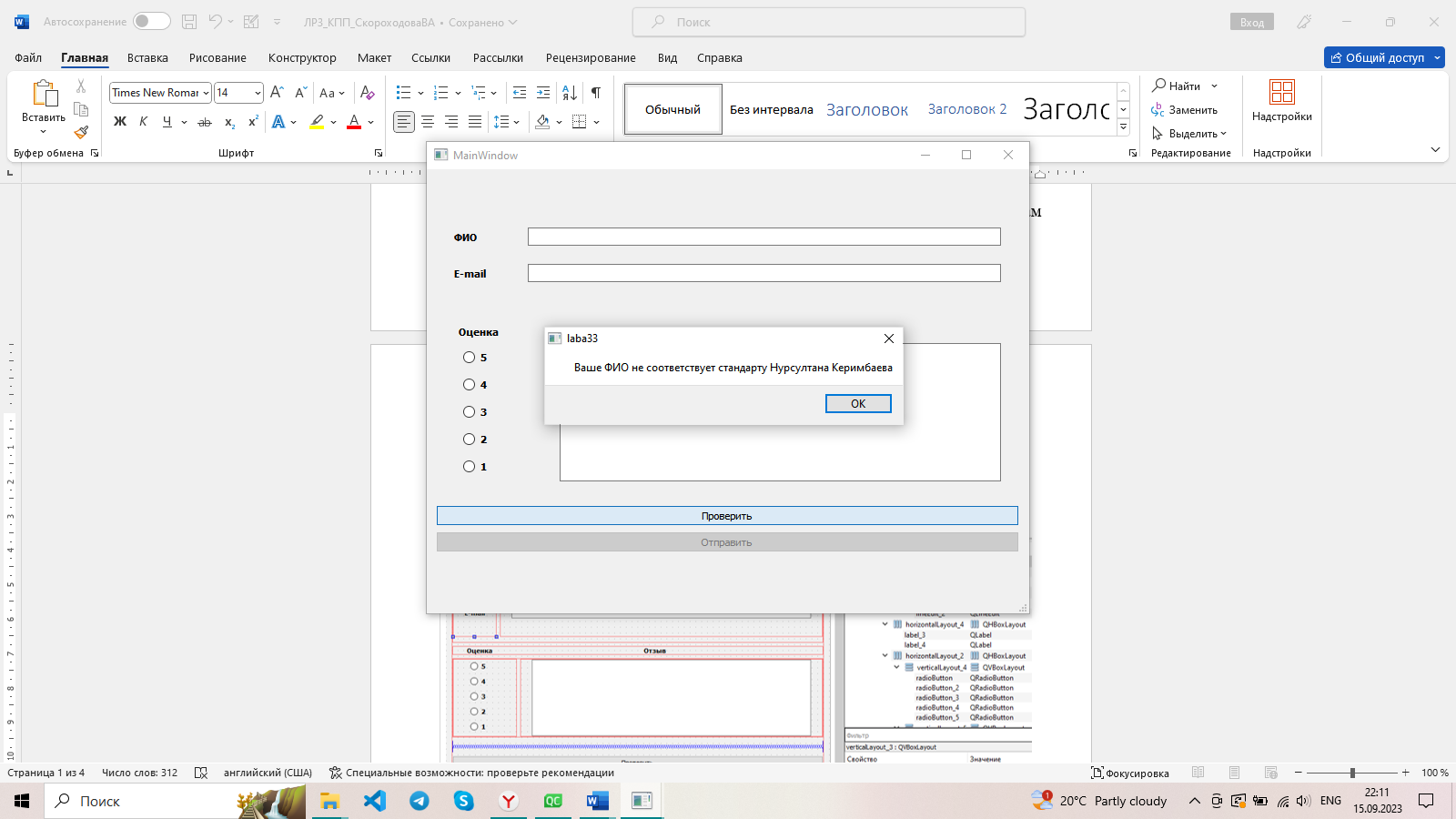


Рисунок 3 – Работа валидации при пустом поле ФИО

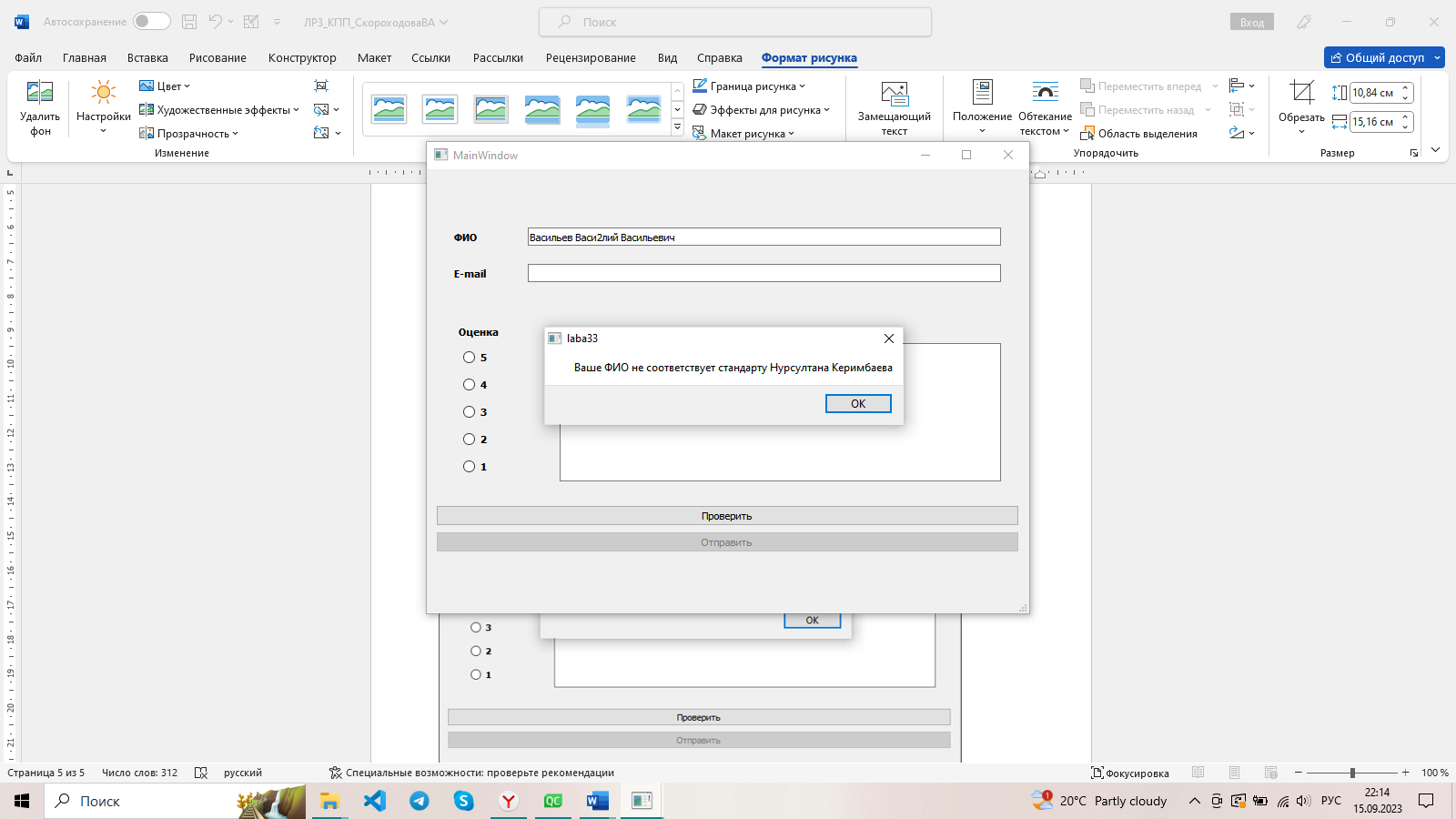


Рисунок 4 – Работа валидации при некорректно заполненном поле ФИО

Когда форма ФИО наконец заполнена корректно, валидация переходит к полю E-mail. Как и с полем ФИО, не допускается пустой формы (рисунок 5).

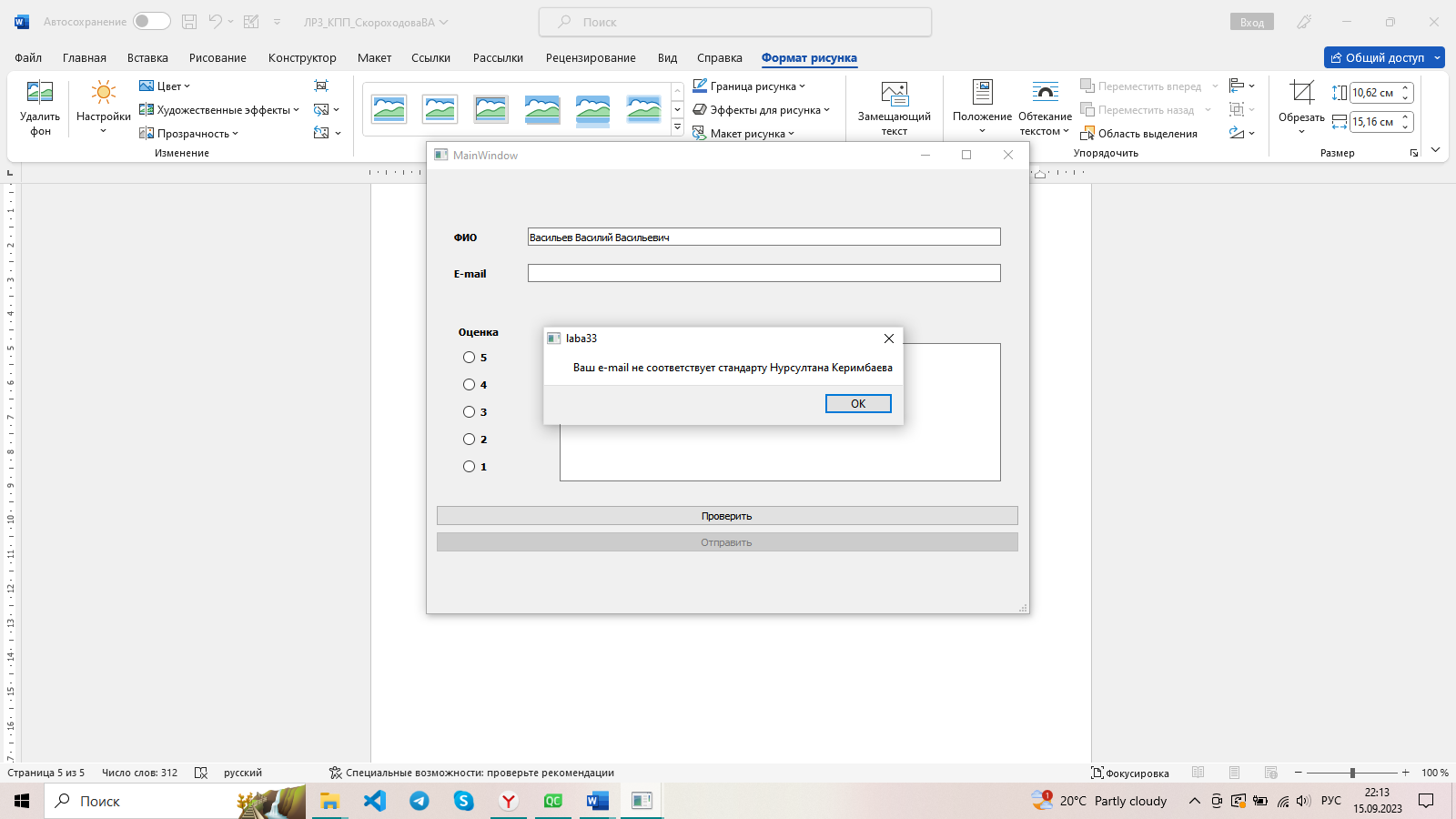


Рисунок 5 – Работа валидации при пустом поле E-mail

Форма должна быть заполнена согласно схеме: имя почтового ящика может быть любой длины и содержать как буквы, так и цифры, затем обязательно должен идти символ «@», а доменное имя содержит только буквы и должно быть разделено точкой. На рисунке 6 показана работа валидации при неправильном заполнении поля.

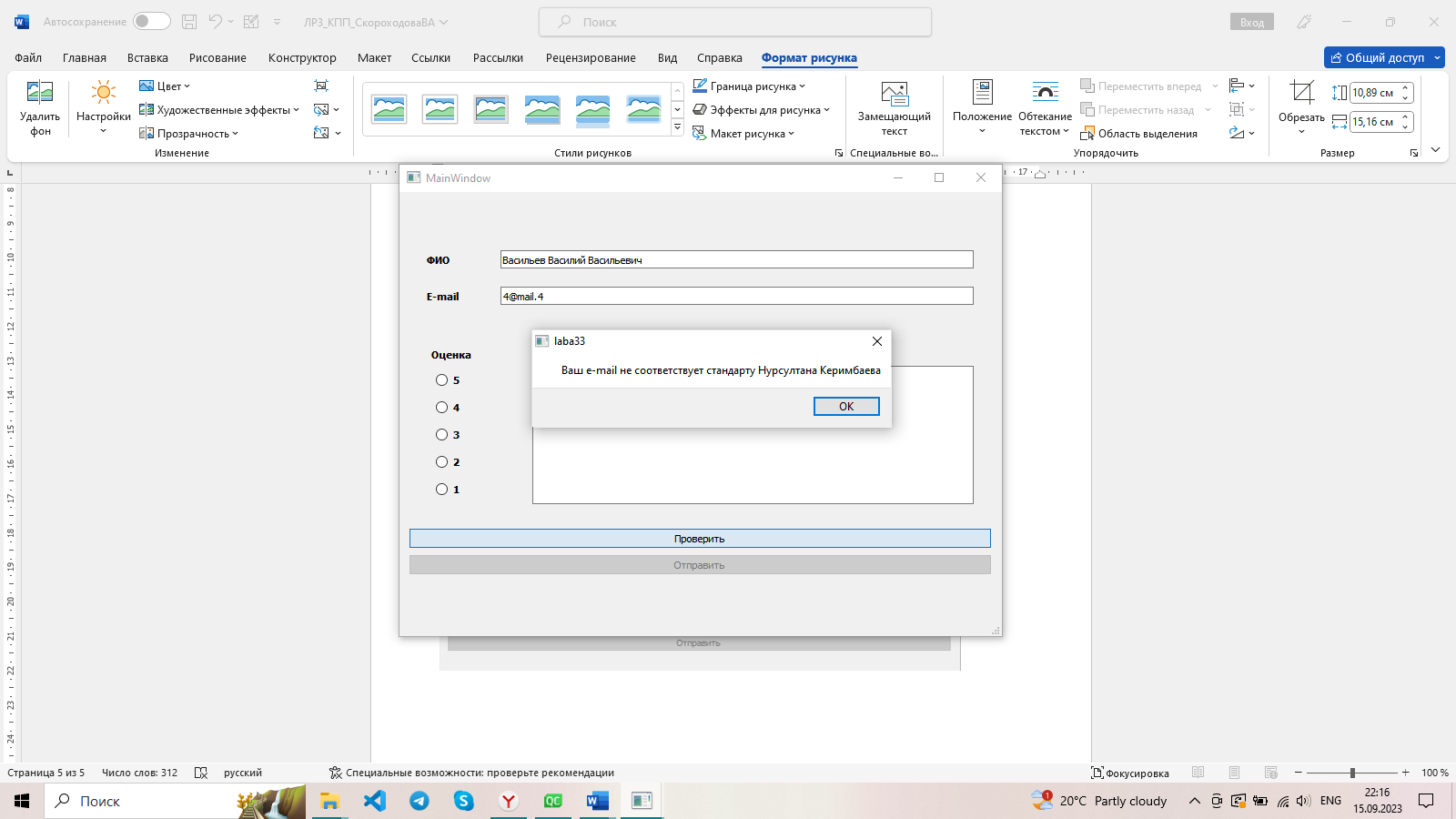


Рисунок 6 – Работа валидации при пустом поле E-mail

Когда почта и ФИО заполнены правильно, валидация переходит к полю отзыва. В этом поле могут содержаться любые символы, оно только не должно быть пустым.

Работа валидации поля отзыва изображена на рисунке 7.

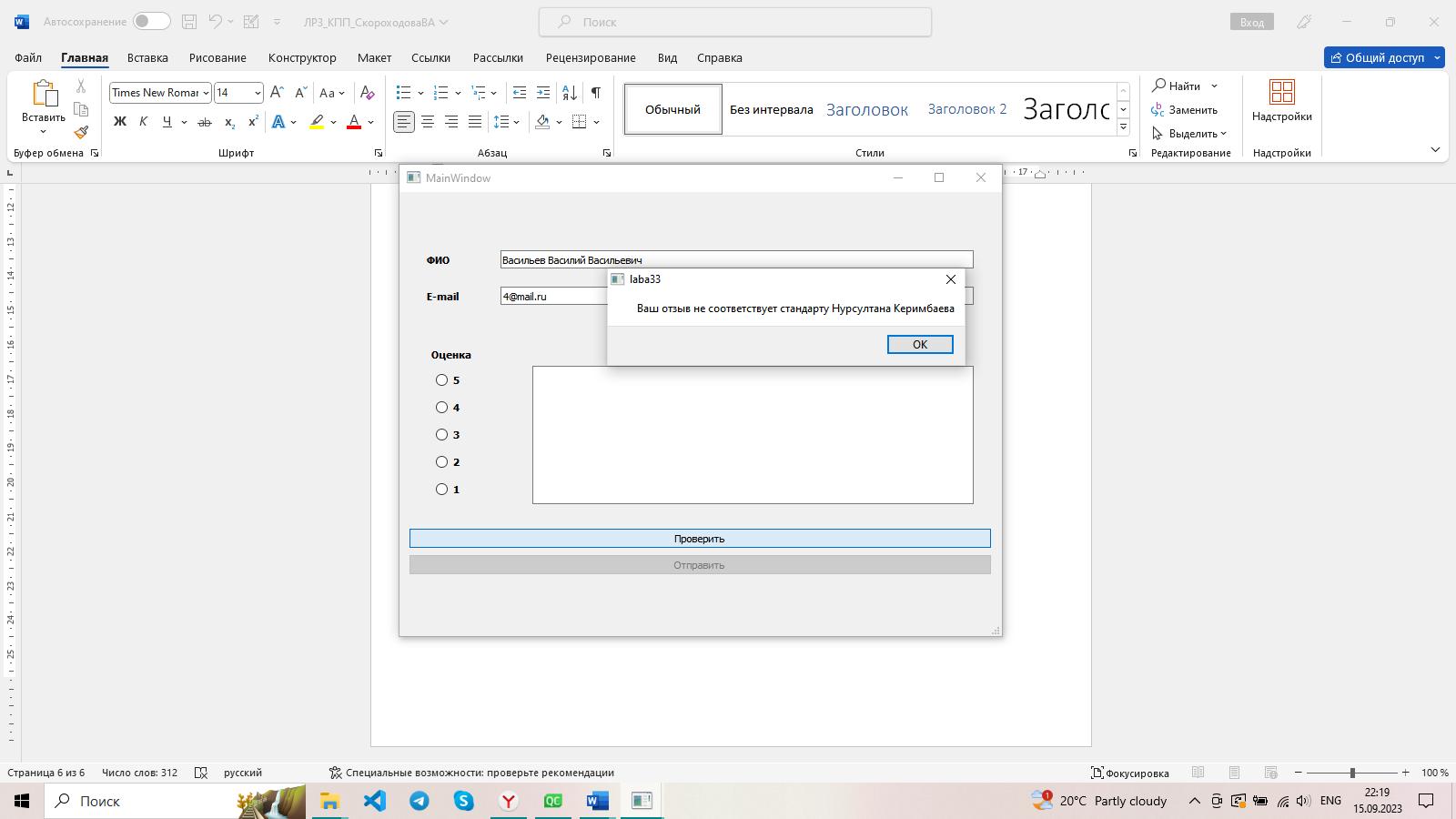


Рисунок 7 – Работа валидации при незаполненном поле отзыва

Если три предыдущих корректно заполнены, валидация переходит к радио-баттонам. Условие успешной валидации: выбор одного из пяти вариантов оценки.

На рисунке 8 показана работа валидации при пустом выборе из пяти оценок.

На рисунке 9 показано, что при успешной проверке и валидации всех форм, ранее недоступная кнопка «Отправить» становится доступной для клика.

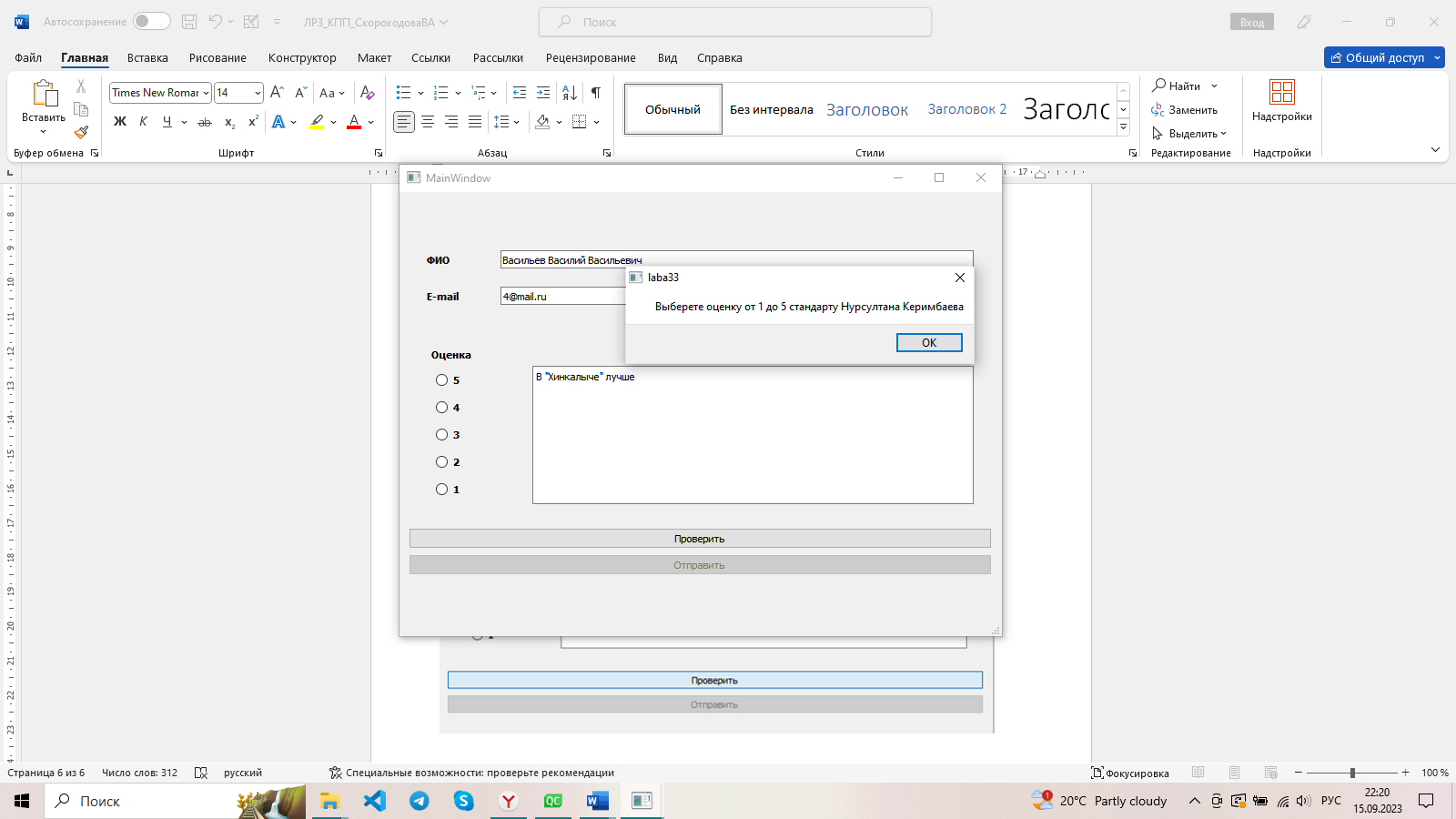


Рисунок 8 – Работа валидации при пустых радио-баттонах

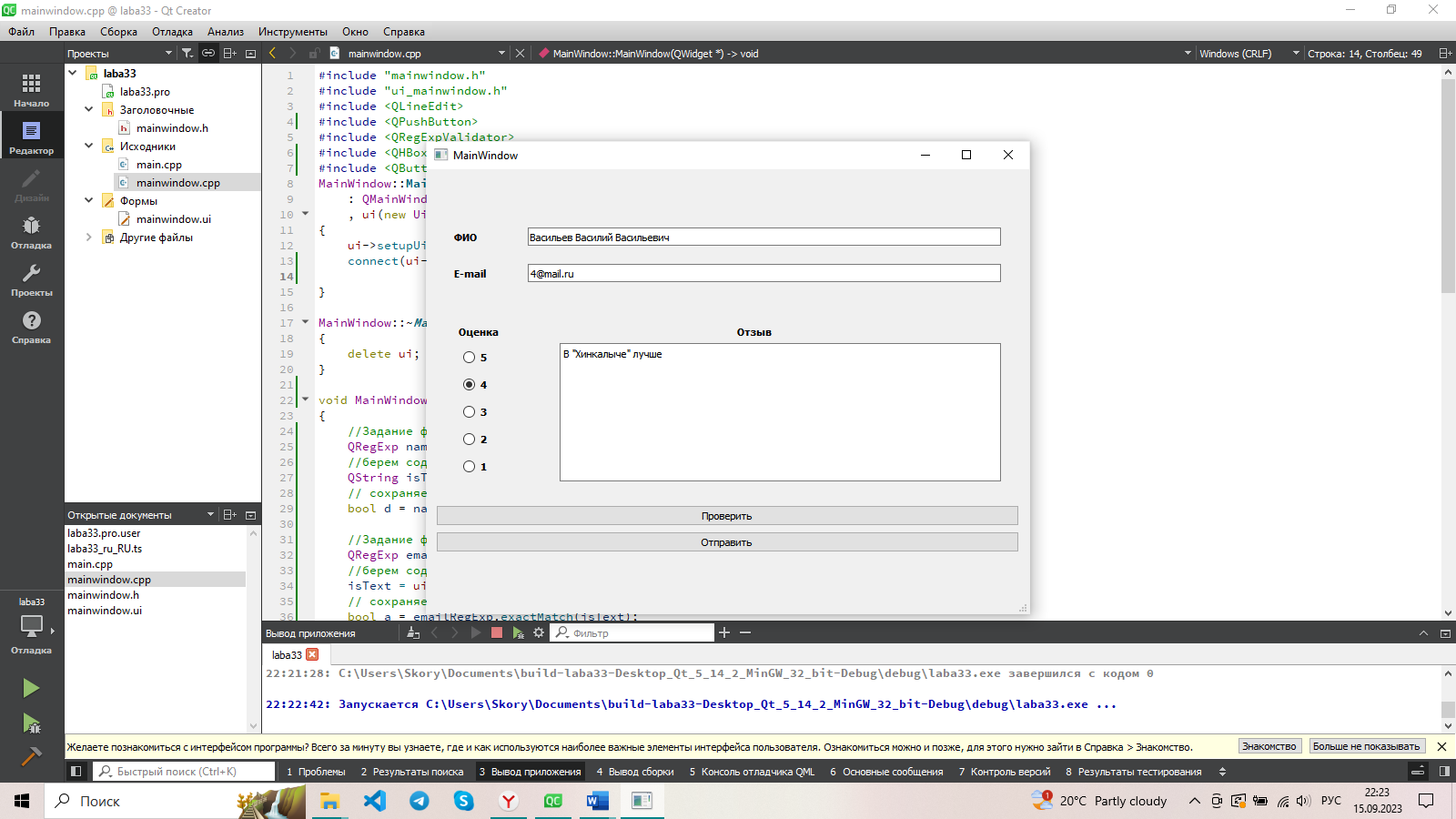


Рисунок 9 – Открытие кнопки «Отправить» при успешном прохождении проверки

ВЫВОДЫ

В ходе выполнения лабораторной работы были изучены основные методики создания графического пользовательского интерфейса с использованием виджетов QtWidgets. Было создано приложение с помощью Qt Designer в соответствии с заданием по варианту, а также разработан функционал, а именно валидация форм. В ходе разработки программного кода проводилось тестирование функции валидации, а по результатам тестов установлено, что функция работает корректно.

Контрольные вопросы:

5.1. Что такое виджет?

5.1. В Qt виджет - это базовый элемент пользовательского интерфейса, который отображается на экране и с которым пользователи могут взаимодействовать. Виджеты могут представлять собой кнопки, текстовые поля, списки, таблицы и другие элементы интерфейса.

5.2. Что такое схема размещения, для чего она нужна?

5.2. Схема размещения (Layout) в Qt служит для организации и управления размещением виджетов внутри других виджетов. Она определяет, как виджеты будут располагаться и как они будут масштабироваться при изменении размеров родительского виджета. Схемы размещения позволяют легко создавать гибкие и адаптивные пользовательские интерфейсы.

5.3. Какие есть виды схем размещения, в чем их различие?

QVBoxLayout: Схема размещения вертикального списка, в котором виджеты располагаются один под другим по вертикали.

QHBoxLayout: Схема размещения горизонтального списка, в котором виджеты располагаются горизонтально, один рядом с другим.

QGridLayout: Схема размещения в виде сетки, в которой виджеты располагаются в ячейках таблицы.

QFormLayout: Схема размещения для формы, в которой виджеты меток и полей ввода располагаются в столбцах.

QHBoxLayout и QVBoxLayout являются простыми, однако они могут быть вложены друг в друга для создания более сложных размещений.

5.4. Сколько виджетов может содержать другой виджет? Сколько схем размещения?

5.4. Виджет может содержать любое количество других виджетов.

Что касается схем размещения, то обычно каждый виджет имеет одну схему размещения, в которой он может содержаться. Однако вы можете использовать вложенные схемы размещения, чтобы создать более сложные макеты, включающие различные виды схем размещения внутри других схем.

5.5. Что такое окно?

5.5. В контексте Qt окно — это верхний уровень виджета, который представляет собой отдельное окно приложения. Окно обычно имеет заголовок, панель инструментов, панель меню и область содержимого, где располагаются другие виджеты и элементы интерфейса.

5.6. Как программно добавить на схему размещения виджет? Как добавить другую схему размещения?

Для программного добавления виджета на схему размещения в Qt вы можете использовать методы, такие как addWidget() или insertWidget(). Например:

QVBoxLayout \*layout = new QVBoxLayout(); // Создание вертикальной схемы размещения

QWidget \*widget = new QWidget(); // Создание виджета

QPushButton \*button = new QPushButton("Button"); // Создание кнопки

layout->addWidget(button); // Добавление кнопки на схему размещения

widget->setLayout(layout); // Установка схемы размещения для виджета

Для добавления другой схемы размещения на схему размещения вы можете использовать метод addLayout(). Например:

QVBoxLayout \*mainLayout = new QVBoxLayout(); // Основная схема размещения

QHBoxLayout \*subLayout = new QHBoxLayout(); // Вложенная схема размещения

QWidget \*widget = new QWidget(); // Создание виджета

subLayout->addWidget(button1);

subLayout->addWidget(button2);

mainLayout->addLayout(subLayout);

widget->setLayout(mainLayout);