МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт информационных технологий и управления в технических системах

Кафедра «Информационные системы»

**ОТЧЕТ**

о выполнении лабораторной работы № 2

по дисциплине “Кроссплатформенное Программирование”

Выполнил: ст. гр. ИС/б-20-2-о

Белик Г. М.

Проверил: доц. каф.

«Информационные системы»

Строганов В. А.

Севастополь

2023

**ЛАБОРАТОРНАЯ РАБОТА №2**

**«Исследование принципов работы платформы QT и среды разработки QT Creator. Исследование способов использования сигналов и слотов в QT-приложениях»**

**Цель работы**

Исследовать принцип работы механизма сигналов и слотов фреймворка Qt. Приобрести практические навыки применения сигналов и слотов при разработке Qt-приложений.

**Задачи**

1. Изучить принципы работы механизма сигналов и слотов в Qt, способы соединения сигналов и слотов.
2. Создать проект Qt Gui Application.
3. Создать класс-наследник класса QLabel, добавив собственный сигнал, который будет посылаться, когда значение QLabel равно числу, большему десяти.
4. Разместить на форме виджеты QLineEdit, QPushButton, два виджета QPlainTextEdit и виджет созданного на шаге 3 наследника QLabel.
5. Обеспечить изменение названия заголовка окна приложения на значение, введенное в QLineEdit при нажатии на кнопку.
6. Создать собственный слот для MainWindow, который будет копировать текст из первого QPlainTextEdit во второй, заменяя все символы ‘a’ на ‘\*’.
7. Подключить слот, созданный на предыдущем этапе к textChanged сигналу первого QPlainTextEdit, таким образом обеспечив автоматическое копирование.
8. Создать собственный слот для MainWindow, который будет выводить количество ‘\*’ во втором QPlainTextEdit в QLabel.
9. Подключить слот, созданный на предыдущем этапе к textChanged сигналу второго QPlainTextEdit, таким образом обеспечив автоматическое подсчет количества символов ‘\*’.
10. Подключить слот setDisabled первого QPlainTextEdit к сигналу, созданному на шаге 3.3, тем самым обеспечив запрет на дальнейший ввод (setDisabled cлот) при вводе более десяти символов ‘a’.
11. Выполнить экспериментальное исследование полученного приложения, выполняя ввод тестовых последовательностей с различным количеством символов ‘a’ и различным их положением во вводимой строке: в начале, в середине и в конце.

**Ход работы**

При запуске программы появляется окно MainWindow

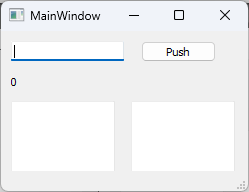


Рисунок 1 – Старт программы

Оставим поле ввода пустым и нажмем кнопку “Push”. Результат изменения названия окна приложения представлен на рисунке 2.

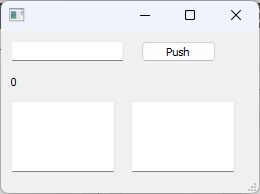


Рисунок 2 – Результат изменения окна приложения

Проверим замену символов “а” на “\*” путем ввода различных последовательностей символов в поле. Данный процесс представлен на рисунке 3.

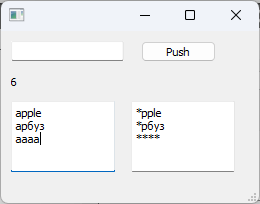


Рисунок 3 – Процесс замены символа “a” на “\*” и блокировка текстового поля

Проверим блокировку первого поля при достижении количества символов “\*” больше 10 во втором поле. Результат проверки изображен на рисунке 4.

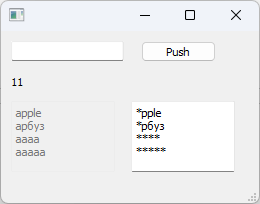


Рисунок 4 – Результат блокировки поля ввода при достижении количества \* больше 10

**Текст программы:**

**main.cpp**

#include "mainwindow.h"

#include <QApplication>

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

{

QApplication a(*argc*, argv);

MainWindow w;

w.show();

return a.exec();

}

**mainwindow.h**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QMessageBox>

#include <mylabel.h>

namespace **Ui** {

class **MainWindow**;

}

class **MainWindow** : public QMainWindow

{

Q\_OBJECT

public:

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

~***MainWindow***();

private:

Ui::MainWindow \*ui;

private slots:

void **changeTitle**();

void **copyText**();

void **countOfStars**();

void **blockPlainTextEdit**();

};

#endif

**mainwindow.cpp**

#include "mainwindow.h"

#include "ui\_mainwindow.h"

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

QMainWindow(parent),

ui(new Ui::MainWindow)

{

ui->setupUi(this);

connect(ui->pushButton, SIGNAL(clicked()), this, SLOT(changeTitle()));

connect(ui->plainTextEdit, SIGNAL(textChanged()), this, SLOT(copyText()));

connect(ui->plainTextEdit\_2, SIGNAL(textChanged()), this, SLOT(countOfStars()));

connect(ui->label, SIGNAL(ownSignal()), this, SLOT(blockPlainTextEdit()));

}

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

{

delete ui;

}

void MainWindow::**changeTitle**()

{

QString newTitle= ui->lineEdit->text();

if(!newTitle.isEmpty())

{

this->setWindowTitle(newTitle);

}

else

{

this->setWindowTitle(" ");

}

}

void MainWindow::**copyText**()

{

QString text = ui->plainTextEdit->toPlainText();

text.replace("a","\*");

text.replace("а","\*");

ui->plainTextEdit\_2->setPlainText(text);

}

void MainWindow::**countOfStars**()

{

int count = ui->plainTextEdit\_2->toPlainText().count("\*");

ui->label->setCount(count);

}

void MainWindow::**blockPlainTextEdit**()

{

ui->plainTextEdit->setDisabled(true);

}

**mylabel.h**

#ifndef MYLABEL\_H

#define MYLABEL\_H

#include <QLabel>

class **MyLabel** : public QLabel

{

Q\_OBJECT

private:

int value = 0;

const int limit = 10;

public:

**MyLabel**(QWidget \*parent = 0);

public slots:

void **setCount**(int);

signals:

void **ownSignal**();

};

#endif

**mylabel.cpp**

#include "mylabel.h"

MyLabel::**MyLabel**(QWidget \*parent)

: QLabel(parent)

{

}

void MyLabel::**setCount**(int value)

{

if (value > limit)

{

emit ownSignal();

}

this->setText(QString::number(value));

}

**Выводы**

В ходе лабораторной работы были исследованы способы использования сигналов и слотов в QT-приложениях. Сигналы и слоты – фундаментальный механизм Qt, позволяющий связывать объекты друг с другом, они намного удобнее, чем функции обратного вызова, предлагаемые другими библиотеками. Преимуществом можно является то, что функция всегда вызывается с корректными аргументами. Также привязка слотов происходит при помощи функции connect, что позволяет избавиться от жесткой привязки обратного вызова с вызывающей его функцией.