|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ  **09.03.03  Название направления**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 9 |

**Программирование с использованием Qt**

**Дисциплина:** Объектно-ориентированное программирование

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-24 Б |  |  | Д.О Кошенков |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | О.А Веселовская |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

**1** Создание простого приложения На рисунке 5 показан внешний вид простого приложения, в котором предлагается ввести возраст с использованием одного из трех вариантов ввода: 1) непосредственного ввода числа, 2) посредством стрелок (элемент типа QSpinBox), последовательно увеличивающих или уменьшающих значение, 3) с помощью специального ползунка (слайдера – элемент типа QSlider).

**Задание** Замените в программе схему выравнивания QHBoxLayout на QVBoxLayout и зафиксируйте результат.

**Файл Mainwindow.h**  
#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

QT\_BEGIN\_NAMESPACE

namespace Ui { class MainWindow; }

QT\_END\_NAMESPACE

class MainWindow : public QMainWindow{

Q\_OBJECT

public:

MainWindow(QWidget \*parent = nullptr);

~MainWindow();

private slots:

void on\_pushButton\_clicked();

private:

Ui::MainWindow \*ui;

};

#endif // MAINWINDOW\_H

**Файл main.cpp**

#include "mainwindow.h"

#include <QApplication>

#include <QLocale>

#include <QTranslator>

#include <QDebug>

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

{

QApplication a(argc, argv);

QTranslator translator;

const QStringList uiLanguages = QLocale::system().uiLanguages();

for (const QString &locale : uiLanguages) {

const QString baseName = "t1\_" + QLocale(locale).name();

if (translator.load(":/i18n/" + baseName)) {

a.installTranslator(&translator);

break;

}

}

MainWindow w;

w.show();

return a.exec();

}  
  
**файл mainwindow.cpp**

#include "mainwindow.h"

#include "./ui\_mainwindow.h"

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow){

ui->setupUi(this);

}

MainWindow::~MainWindow(){

delete ui;

}

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

qDebug() <<"Программа завершена корректно";

}

Рисунок 1 - Интерфейс

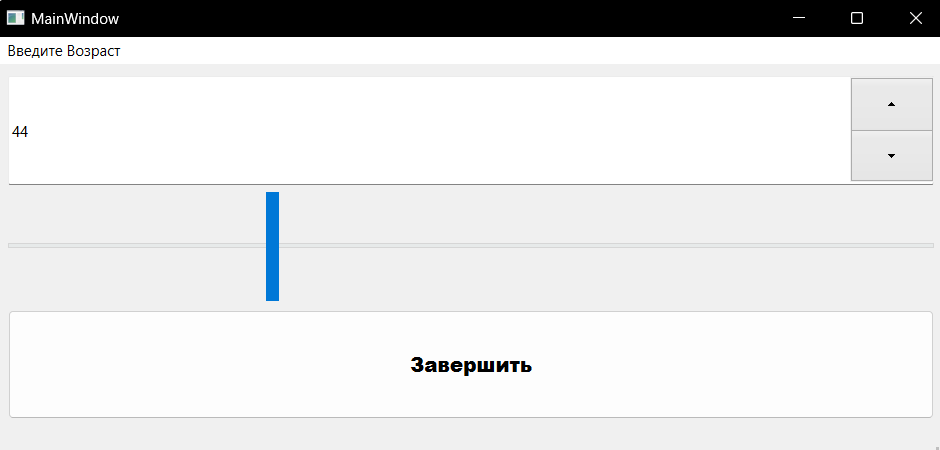
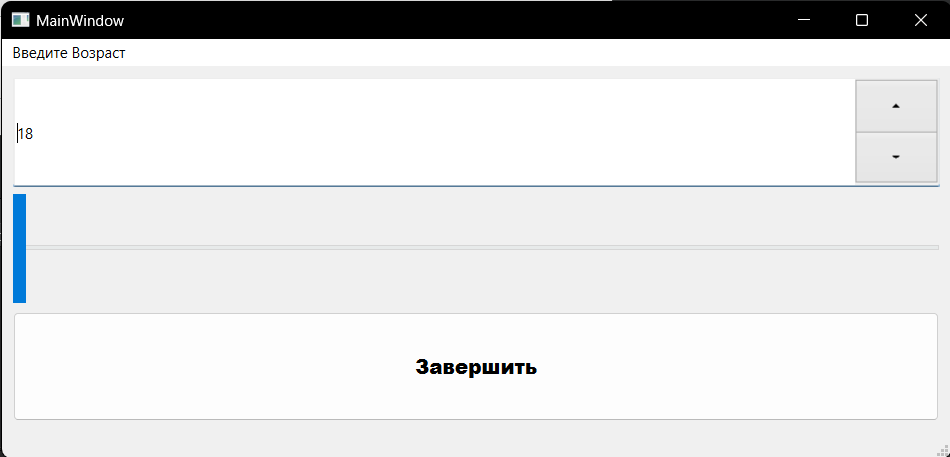


Рисунок 2 – работа программы

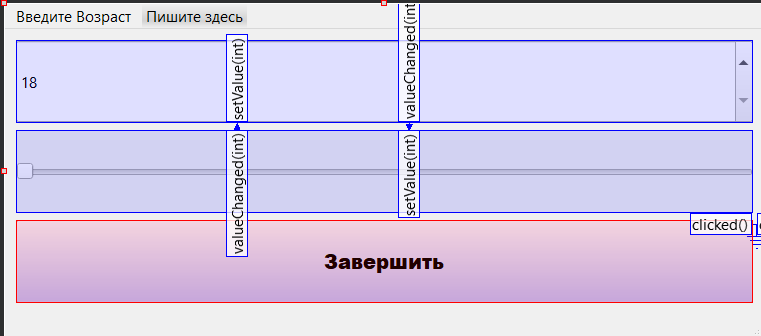


Рисунок 3 - слоты

2 Создание простого приложения в QtDesigner Рассмотрим разработку прототипов диалогов в QtDesigner. В предыдущем задании были показаны простейшие способы выравнивания виджетов. QtDesigner, помимо разнообразных виджетов, позволяет использовать так называемые менеджеры компоновки, работа с которыми и будет рассмотрена в задании.

**Задание** Измените тип разделителя с QSplitter(Qt::Horizontal); на QSplitter(Qt::Vertical); и зафиксируйте полученный результат.

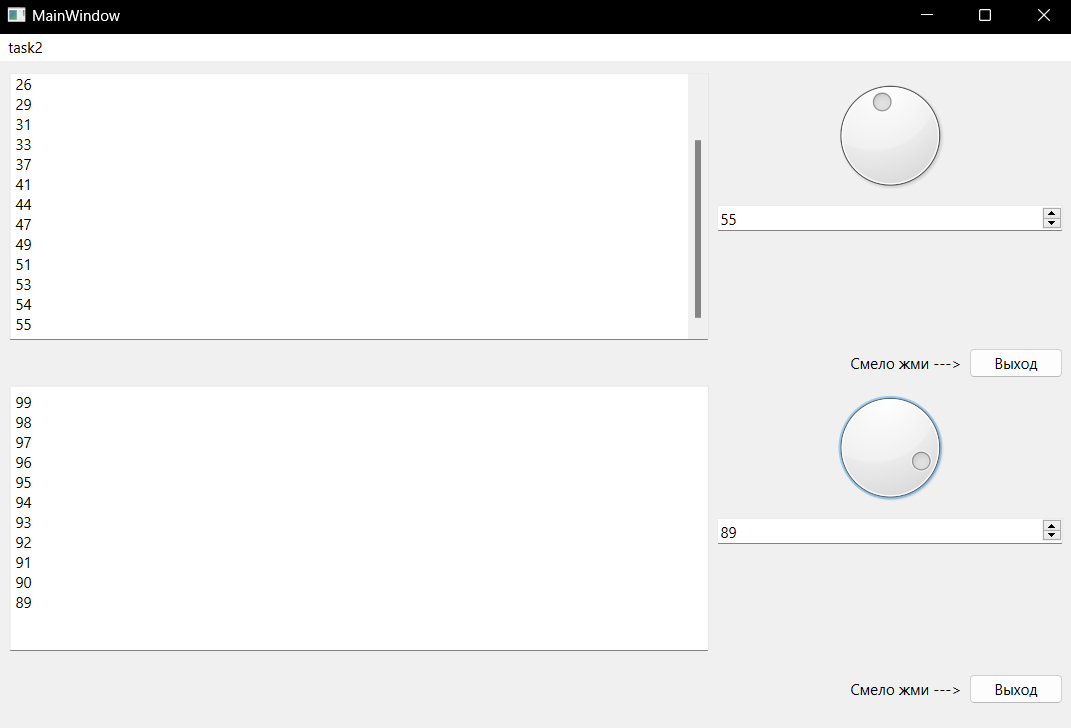


Рисунок 4 – интерфейс и работа программы

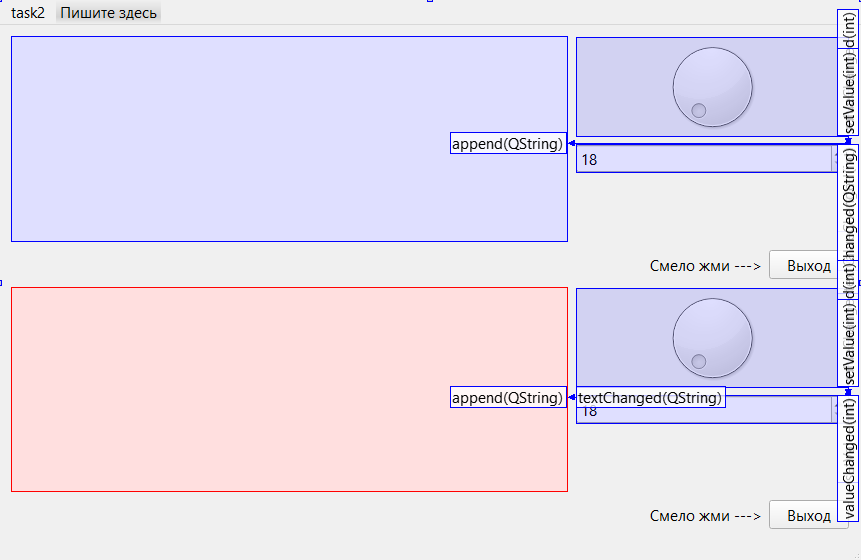


Рисунок 5 – слоты

**Файл mainwindow.h**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

QT\_BEGIN\_NAMESPACE

namespace Ui { class MainWindow; }

QT\_END\_NAMESPACE

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

MainWindow(QWidget \*parent = nullptr);

~MainWindow();

private slots:

void on\_pushButton\_clicked();

void on\_pushButton\_3\_clicked();

private:

Ui::MainWindow \*ui;

};

#endif // MAINWINDOW\_H

**Файл main.cpp**

#include "mainwindow.h"

#include <QApplication>

#include <QLocale>

#include <QTranslator>

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

{

QApplication a(argc, argv);

QTranslator translator;

const QStringList uiLanguages = QLocale::system().uiLanguages();

for (const QString &locale : uiLanguages) {

const QString baseName = "t2\_" + QLocale(locale).name();

if (translator.load(":/i18n/" + baseName)) {

a.installTranslator(&translator);

break;

}

}

MainWindow w;

w.show();

return a.exec();

}

Файл mainwindow.cpp

#include "mainwindow.h"

#include "./ui\_mainwindow.h"

#include <QMessageBox>

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

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

if(QMessageBox::question(this, "What?", "close?", QMessageBox::Yes | QMessageBox::No) == QMessageBox::Yes){

QApplication::quit();

}

else show();

}

void MainWindow::on\_pushButton\_3\_clicked()

{

if(QMessageBox::question(this, "What?", "close?", QMessageBox::Yes | QMessageBox::No) == QMessageBox::Yes){

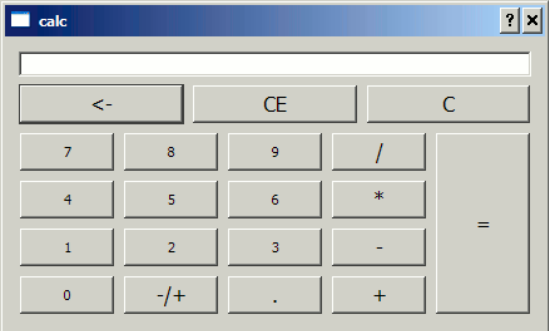
QApplication::quit();

}

else show();

}

3.Разработка калькулятора   
Исходные данные Рассмотрим создание более сложного приложения, наглядно иллюстрирующего возможности библиотеки Qt в части компактного кода программы и динамического создания элементов управления. Разработаем калькулятор, внешний вид которого представлен на рисунке



**Задание** Добавьте кнопки, выполняющие: бинарные операции x y , logy x (по аналогии с операциями +,-,/,\*), а также унарные sin(x) и cos(x) (по аналогии с операцией -/+) и разместите этот ряд кнопок вертикально, слева от цифровых кнопок с использованием нового объекта выравнивания (Layout).

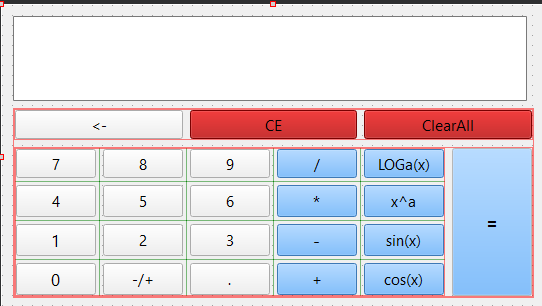


Рисунок 6 – создал интерфейс



Рисунок 7 – работа программы

**Файл mainwindow.h**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

QT\_BEGIN\_NAMESPACE

namespace Ui { class MainWindow; }

QT\_END\_NAMESPACE

class MainWindow : public QMainWindow{

Q\_OBJECT

public:

MainWindow(QWidget \*parent = nullptr);

~MainWindow();

private:

Ui::MainWindow \*ui;

private slots:

void digitsNumbers();

void on\_dot\_but\_clicked();

void operations();

void math\_operations();

void on\_ClearAll\_but\_clicked();

void on\_Total\_but\_clicked();

};

#endif // MAINWINDOW\_H

**Файл main.cpp**

#include "mainwindow.h"

#include <QApplication>

#include <QLocale>

#include <QTranslator>

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

{

QApplication a(argc, argv);

QTranslator translator;

const QStringList uiLanguages = QLocale::system().uiLanguages();

for (const QString &locale : uiLanguages) {

const QString baseName = "t3\_" + QLocale(locale).name();

if (translator.load(":/i18n/" + baseName)) {

a.installTranslator(&translator);

break;

}

}

MainWindow w;

w.show();

return a.exec();

}

**Файл mainwindow.cpp**

#include "mainwindow.h"

#include "./ui\_mainwindow.h"

#include <QtCore/qmath.h>

double num\_first=0;

void backspace(QString& text) {

if (text.isEmpty()) {

// Если строка пустая, ничего не делаем

return;

}

// Проверяем, есть ли точка в числе

bool has\_decimal\_point = text.contains('.');

if (has\_decimal\_point) {

// Если после точки только одна цифра, удаляем ее вместе с точкой

int decimal\_index = text.indexOf('.');

int num\_digits\_after\_decimal = text.length() - decimal\_index - 1;

if (num\_digits\_after\_decimal == 1) {

text.chop(2);

} else {

// Удаляем только последнюю цифру после точки

text.chop(1);

}

} else {

// Удаляем последнюю цифру

text.chop(1);

}

// Если после удаления осталась только точка, удаляем ее

if (text.endsWith('.')) {

text.chop(1);

}

}

double calculateLog(double a, double x) {

return qLn(x) / qLn(a);

}

double calculatePower(double x, double a) {

return qPow(x, a);

}

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow){

ui->setupUi(this);

connect(ui->nule\_but, SIGNAL(clicked()),this,SLOT(digitsNumbers()));

connect(ui->one\_but, SIGNAL(clicked()),this,SLOT(digitsNumbers()));

connect(ui->two\_but, SIGNAL(clicked()),this,SLOT(digitsNumbers()));

connect(ui->three\_but, SIGNAL(clicked()),this,SLOT(digitsNumbers()));

connect(ui->four\_but, SIGNAL(clicked()),this,SLOT(digitsNumbers()));

connect(ui->five\_but, SIGNAL(clicked()),this,SLOT(digitsNumbers()));

connect(ui->six\_but, SIGNAL(clicked()),this,SLOT(digitsNumbers()));

connect(ui->seven\_but, SIGNAL(clicked()),this,SLOT(digitsNumbers()));

connect(ui->eight\_but, SIGNAL(clicked()),this,SLOT(digitsNumbers()));

connect(ui->nine\_but, SIGNAL(clicked()),this,SLOT(digitsNumbers()));

connect(ui->Inverse\_but, SIGNAL(clicked()),this,SLOT(operations()));

connect(ui->bck\_but, SIGNAL(clicked()),this,SLOT(operations()));

connect(ui->ClearEm\_but, SIGNAL(clicked()),this,SLOT(operations()));

connect(ui->Plus\_but, SIGNAL(clicked()),this,SLOT(math\_operations()));

connect(ui->Minus\_but, SIGNAL(clicked()),this,SLOT(math\_operations()));

connect(ui->Sin\_but, SIGNAL(clicked()),this,SLOT(operations()));

connect(ui->Cos\_but, SIGNAL(clicked()),this,SLOT(operations()));

connect(ui->Log\_but, SIGNAL(clicked()),this,SLOT(math\_operations()));

connect(ui->Pow\_but, SIGNAL(clicked()),this,SLOT(math\_operations()));

connect(ui->Div\_but, SIGNAL(clicked()),this,SLOT(math\_operations()));

connect(ui->Mult\_but, SIGNAL(clicked()),this,SLOT(math\_operations()));

ui->Plus\_but->setCheckable(true);

ui->Minus\_but->setCheckable(true);

ui->Log\_but->setCheckable(true);

ui->Pow\_but->setCheckable(true);

ui->Mult\_but->setCheckable(true);

ui->Div\_but->setCheckable(true);

}

MainWindow::~MainWindow(){

delete ui;

}

void MainWindow::digitsNumbers(){

QPushButton \*button = (QPushButton\*)sender();

double allNums = 0;

QString num\_text;

allNums = (ui->textEdit->toPlainText() + button->text()).toDouble();

num\_text = QString::number(allNums, 'g', 15);

ui->textEdit->setText(num\_text);

}

void MainWindow::on\_dot\_but\_clicked(){

if(!ui->textEdit->toPlainText().contains('.'))

ui->textEdit->setText(ui->textEdit->toPlainText()+".");

}

void MainWindow::operations(){

QPushButton \*button = (QPushButton\*)sender();

double allNums = 0;

QString num\_str;

if(button->text() == "-/+"){

allNums = (ui->textEdit->toPlainText()).toDouble();

allNums \*= -1;

num\_str = QString::number(allNums, 'g', 15);

ui->textEdit->setText(num\_str);

}else if (button->text() == "<-"){

allNums = (ui->textEdit->toPlainText()).toDouble();

num\_str = QString::number(allNums, 'g', 15);

backspace(num\_str);

ui->textEdit->setText(num\_str);

}else if (button->text() == "CE"){

ui->textEdit->clear();

}else if (button->text() == "sin(x)"){

allNums = (ui->textEdit->toPlainText()).toDouble();

allNums = qSin(allNums);

num\_str = QString::number(allNums, 'g', 15);

ui->textEdit->setText(num\_str);

}else if (button->text() == "cos(x)"){

allNums = (ui->textEdit->toPlainText()).toDouble();

allNums = qCos(allNums);

num\_str = QString::number(allNums, 'g', 15);

ui->textEdit->setText(num\_str);

}

}

void MainWindow::math\_operations(){

QPushButton \*button = (QPushButton\*)sender();

num\_first = ui->textEdit->toPlainText().toDouble();

ui->textEdit->clear();

button->setChecked(true);

}

void MainWindow::on\_ClearAll\_but\_clicked(){

num\_first = 0;

ui->textEdit->clear();

ui->Plus\_but->setChecked(false);

ui->Minus\_but->setChecked(false);

ui->Log\_but->setChecked(false);

ui->Pow\_but->setChecked(false);

ui->Mult\_but->setChecked(false);

ui->Div\_but->setChecked(false);

}

void MainWindow::on\_Total\_but\_clicked(){

double textNum, num\_second = 0;

QString new\_Num;

num\_second = ui->textEdit->toPlainText().toDouble();

if (ui->Plus\_but->isChecked()){

textNum = num\_second + num\_first;

new\_Num = QString::number(textNum, 'g', 15);

ui->textEdit->setText(new\_Num);

ui->Plus\_but->setChecked(false);

}else if (ui->Minus\_but->isChecked()){

textNum = num\_first - num\_second;

new\_Num = QString::number(textNum, 'g', 15);

ui->textEdit->setText(new\_Num);

ui->Minus\_but->setChecked(false);

}else if (ui->Log\_but->isChecked()){

if (num\_second > 0 && num\_second != 1){

textNum =calculateLog(num\_second, num\_first);

new\_Num = QString::number(textNum, 'g', 15);

ui->textEdit->setText(new\_Num);

ui->Log\_but->setChecked(false);

}else

ui->textEdit->setText("ERROR");

}else if (ui->Pow\_but->isChecked()){

textNum = calculatePower(num\_second, num\_first);

new\_Num = QString::number(textNum, 'g', 15);

ui->textEdit->setText(new\_Num);

ui->Pow\_but->setChecked(false);

}else if (ui->Div\_but->isChecked()){

if (num\_second != 0){

textNum = num\_first/num\_second;

new\_Num = QString::number(textNum, 'g', 15);

ui->textEdit->setText(new\_Num);

ui->Div\_but->setChecked(false);

}else

ui->textEdit->setText("ERROR");

}else if (ui->Mult\_but->isChecked()){

textNum = num\_second \* num\_first;

new\_Num = QString::number(textNum, 'g', 15);

ui->textEdit->setText(new\_Num);

ui->Mult\_but->setChecked(false);

}

}

4.Простейшие элементы ввода-вывода

Некоторые средства для ввода и вывода текста Класс QString Для работы со строками в Qt используется класс QString. Основной особенностью этого класса является то, что внутреннее хранение и все операции над строками проводятся в кодировке UNICODE. Класс позволяет преобразовывать текст из различных кодировок строки в формат C и обратно. Реализуется операции склейки, добавления, сравнения, вырезания подстроки и пр.

**Задание** Разработать приложение, имеющее строку ввода данных, кнопку запуска преобразования и текстовое поле, предназначенное только для отображения информации. При этом не использовать QtDesigner! Любой текст строки ввода должен отображаться в текстовом поле сразу после завершения ввода. В начале строки должна быть вставлена пометка «input:». При нажатии кнопки преобразования строка ввода должна быть преобразована либо в верхний регистр, либо в нижний противоположно тому, что производилось при предыдущем нажатии кнопки.

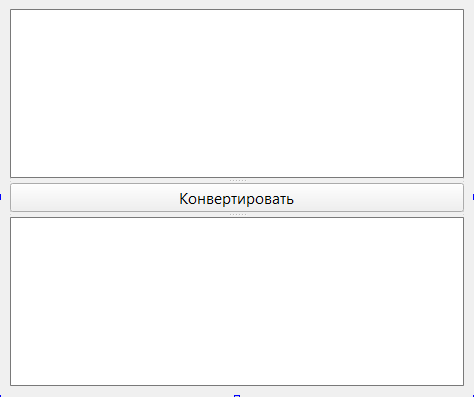


Рисунок 8 - интерфейс

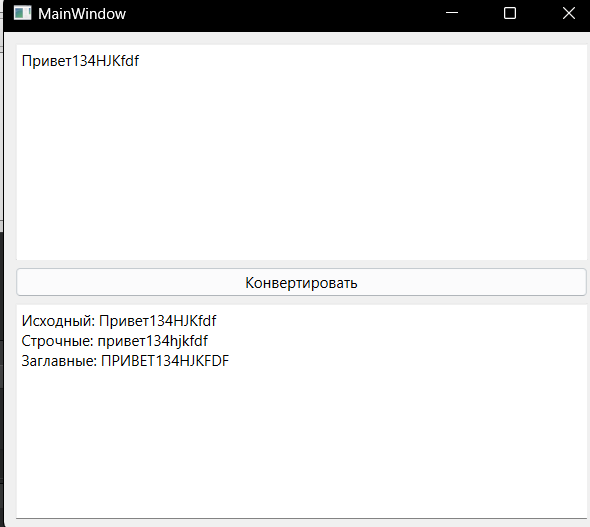


Рисунок 9 – работа программы

**Файл mainwindow.h**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

QT\_BEGIN\_NAMESPACE

namespace Ui { class MainWindow; }

QT\_END\_NAMESPACE

class MainWindow : public QMainWindow{

Q\_OBJECT

public:

MainWindow(QWidget \*parent = nullptr);

~MainWindow();

private slots:

void on\_pushButton\_clicked();

private:

Ui::MainWindow \*ui;

};

#endif // MAINWINDOW\_H

**Файл main.cpp**

#include "mainwindow.h"

#include <QApplication>

#include <QLocale>

#include <QTranslator>

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

{

QApplication a(argc, argv);

QTranslator translator;

const QStringList uiLanguages = QLocale::system().uiLanguages();

for (const QString &locale : uiLanguages) {

const QString baseName = "t4\_" + QLocale(locale).name();

if (translator.load(":/i18n/" + baseName)) {

a.installTranslator(&translator);

break;

}

}

MainWindow w;

w.show();

return a.exec();

}

**Файл mainwindow.cpp**

#include "mainwindow.h"

#include "./ui\_mainwindow.h"

short num = 0;

QString this\_word = "";

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~MainWindow()

{

delete ui;

}

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

QString str;

str = ui->input->toPlainText();

if (str != this\_word){

this\_word = str;

num = 0;

}

if (num == 0){

ui->output->setText("Исходный: "+ str);

}

num++;

if (num == 1){

ui->output->append("Строчные: " + str.toLower());

}else if (num == 2)

ui->output->append("Заглавные: " + str.toUpper());

}

Вывод:разработал базовые интерфейсы, рассмотрел механизм выравнивания