МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М. А. БОНЧ-БРУЕВИЧА» (СПбГУТ)

# **Лабораторная работа № 6**

# **Создание проекта с графическим интерфейсом с использованием библиотеки Qt**

Руководитель,

старший преподаватель Петрова О.Б.

подпись, дата

Исполнитель,

группа ИКПИ-33 К о н ь к о в М . Д .

подпись, дата

**Условия работы:**

**В рамках лабораторной работы необходимо разработать приложение, обладающее графическим интерфейсом с использованием библиотеки Qt. Приложение должно включать:**

* Два текстовых однострочных редактора (QLineEdit), расположенные горизонтально, а также поле для вывода результата.
* Возможность выполнения следующих операций:
  + clear — очистка содержимого конкретного редактора.
  + clearAll — очистка содержимого всех редакторов.
  + reverse — реверсирование содержимого редактора.
  + reverseAll — реверсирование содержимого всех редакторов.
  + swap — обмен содержимым редакторов.
  + copy — копирование содержимого из одного редактора в другой.
  + concat — объединение содержимого двух редакторов.

### 2. Описание классов с их составляющими:

### Класс MainWindow

**Описание:**  
MainWindow — основной класс приложения, унаследованный от QMainWindow. Он содержит элементы интерфейса и методы для обработки пользовательских действий.

**Составляющие класса:**

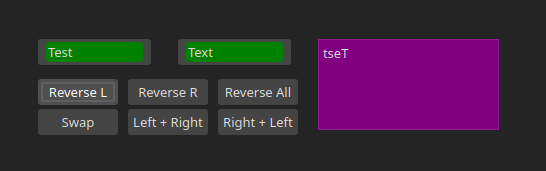
1. **Графические компоненты (элементы интерфейса):**
   * QLineEdit \*lineEditLeft;  
     Поле ввода для левого текста (левый операнд).
   * QLineEdit \*lineEditRight;  
     Поле ввода для правого текста (правый операнд).
   * QTextEdit \*lineEditResult;  
     Поле вывода результатов операций.
2. **Кнопки операций:**
   * QPushButton \*btnReverseLeft;  
     Кнопка для реверса текста из левого поля.
   * QPushButton \*btnReverseRight;  
     Кнопка для реверса текста из правого поля.
   * QPushButton \*btnReverseAll;  
     Кнопка для реверса текстов из обоих полей.
   * QPushButton \*btnSwap;  
     Кнопка для обмена содержимым между левым и правым полями.
   * QPushButton \*btnConcatLeft;  
     Кнопка для объединения текста (левое поле + правое поле).
   * QPushButton \*btnConcatRight;  
     Кнопка для объединения текста (правое поле + левое поле).
3. **Дополнительные свойства:**
   * Плейсхолдеры для полей ввода:
     + lineEditLeft: "Type the L-line".
     + lineEditRight: "Type the R-line".
     + lineEditResult: "Here is a result line".
4. **Обработчики событий (слоты):**
   * onReverseLeftClicked() — реверс текста в левом поле.
   * onReverseRightClicked() — реверс текста в правом поле.
   * onReverseAllClicked() — реверс текста в обоих полях с объединением результата.
   * onSwapClicked() — обмен содержимым между левым и правым полями.
   * onConcatLeftClicked() — объединение текста (левое поле + правое поле).
   * onConcatRightClicked() — объединение текста (правое поле + левое поле).

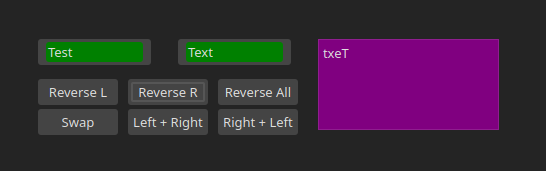
### Макет приложения

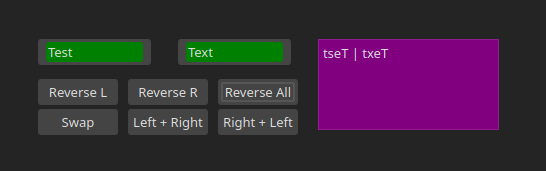
**Описание:**  
Используется визуальный редактор Qt Designer для создания графического интерфейса.  
Основной макет включает:

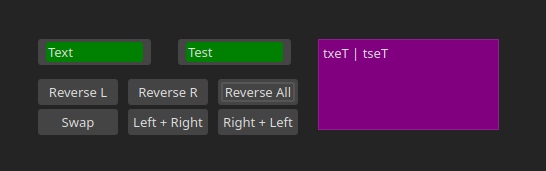
* Поля ввода (линии) для текста:
  + lineEditLeft и lineEditRight (левое и правое поле ввода).
  + lineEditResult для вывода результата.
* Ряд кнопок для выполнения операций (размещены ниже полей ввода).
* Используются свойства геометрии (размеры, положение) для настройки положения компонентов.

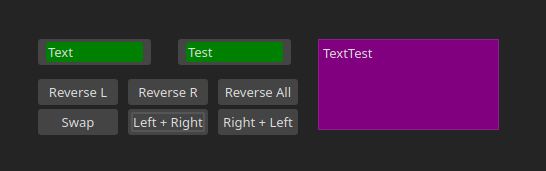
**Результат работы и проверки:**

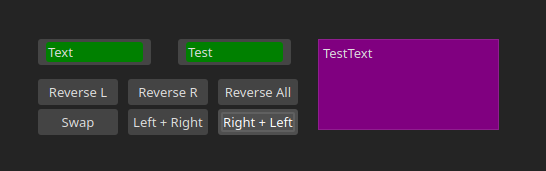
****

****

****

****

****

****

***Выше были протестированы всех шесть операций при одном запуске программы (реверсия левого слова, реверсия второго слова, реверсия обоих слов, перемена слов местами, конкатенация правого слова к левому, конкатенация левого слова к правому).***

**Листинг кода:**

**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.ui:**

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>MainWindow</class>

<widget class="QMainWindow" name="MainWindow">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>800</width>

<height>600</height>

</rect>

</property>

<property name="windowTitle">

<string>MainWindow</string>

</property>

<widget class="QWidget" name="centralwidget">

<widget class="QLineEdit" name="lineEditLeft">

<property name="geometry">

<rect>

<x>80</x>

<y>140</y>

<width>113</width>

<height>26</height>

</rect>

</property>

<property name="styleSheet">

<string notr="true">background-color: green</string>

</property>

<property name="text">

<string/>

</property>

</widget>

<widget class="QLineEdit" name="lineEditRight">

<property name="geometry">

<rect>

<x>220</x>

<y>140</y>

<width>113</width>

<height>26</height>

</rect>

</property>

<property name="styleSheet">

<string notr="true">background-color: green</string>

</property>

<property name="text">

<string/>

</property>

</widget>

<widget class="QTextEdit" name="lineEditResult">

<property name="geometry">

<rect>

<x>360</x>

<y>140</y>

<width>181</width>

<height>91</height>

</rect>

</property>

<property name="styleSheet">

<string notr="true">background-color: purple</string>

</property>

</widget>

<widget class="QPushButton" name="btnReverseLeft">

<property name="geometry">

<rect>

<x>80</x>

<y>180</y>

<width>80</width>

<height>26</height>

</rect>

</property>

<property name="text">

<string>Reverse L</string>

</property>

</widget>

<widget class="QPushButton" name="btnSwap">

<property name="geometry">

<rect>

<x>80</x>

<y>210</y>

<width>80</width>

<height>26</height>

</rect>

</property>

<property name="text">

<string>Swap</string>

</property>

</widget>

<widget class="QPushButton" name="btnReverseRight">

<property name="geometry">

<rect>

<x>170</x>

<y>180</y>

<width>80</width>

<height>26</height>

</rect>

</property>

<property name="text">

<string>Reverse R</string>

</property>

</widget>

<widget class="QPushButton" name="btnConcatLeft">

<property name="geometry">

<rect>

<x>170</x>

<y>210</y>

<width>80</width>

<height>26</height>

</rect>

</property>

<property name="text">

<string>Left + Right</string>

</property>

</widget>

<widget class="QPushButton" name="btnReverseAll">

<property name="geometry">

<rect>

<x>260</x>

<y>180</y>

<width>80</width>

<height>26</height>

</rect>

</property>

<property name="text">

<string>Reverse All</string>

</property>

</widget>

<widget class="QPushButton" name="btnConcatRight">

<property name="geometry">

<rect>

<x>260</x>

<y>210</y>

<width>80</width>

<height>26</height>

</rect>

</property>

<property name="text">

<string>Right + Left</string>

</property>

</widget>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>800</width>

<height>23</height>

</rect>

</property>

</widget>

<widget class="QStatusBar" name="statusbar"/>

</widget>

<resources/>

<connections/>

</ui>

**mainwindow.h:**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QString>

QT\_BEGIN\_NAMESPACE

namespace Ui { class MainWindow; }

QT\_END\_NAMESPACE

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = nullptr);

~MainWindow();

private slots:

void onReverseLeftClicked();

void onReverseRightClicked();

void onReverseAllClicked();

void onSwapClicked();

void onConcatLeftClicked();

void onConcatRightClicked();

private:

Ui::MainWindow \*ui;

};

#endif // MAINWINDOW\_H

**mainwindow.cpp:**

##include "mainwindow.h"

#include "./ui\_mainwindow.h"

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

// Добавляем placeholder text для LineEdit

ui->lineEditLeft->setPlaceholderText("Type the L-line");

ui->lineEditRight->setPlaceholderText("Type the R-line");

ui->lineEditResult->setPlaceholderText("Here is a result line");

// Соединяем кнопки с соответствующими обработчиками

connect(ui->btnReverseLeft, &QPushButton::clicked, this, &MainWindow::onReverseLeftClicked);

connect(ui->btnReverseRight, &QPushButton::clicked, this, &MainWindow::onReverseRightClicked);

connect(ui->btnReverseAll, &QPushButton::clicked, this, &MainWindow::onReverseAllClicked);

connect(ui->btnSwap, &QPushButton::clicked, this, &MainWindow::onSwapClicked);

connect(ui->btnConcatLeft, &QPushButton::clicked, this, &MainWindow::onConcatLeftClicked);

connect(ui->btnConcatRight, &QPushButton::clicked, this, &MainWindow::onConcatRightClicked);

}

MainWindow::~MainWindow()

{

delete ui;

}

// Реализация обработчиков

void MainWindow::onReverseLeftClicked()

{

QString text = ui->lineEditLeft->text();

std::reverse(text.begin(), text.end());

ui->lineEditResult->setText(text);

}

void MainWindow::onReverseRightClicked()

{

QString text = ui->lineEditRight->text();

std::reverse(text.begin(), text.end());

ui->lineEditResult->setText(text);

}

void MainWindow::onReverseAllClicked()

{

QString leftText = ui->lineEditLeft->text();

QString rightText = ui->lineEditRight->text();

std::reverse(leftText.begin(), leftText.end());

std::reverse(rightText.begin(), rightText.end());

ui->lineEditResult->setText(leftText + " | " + rightText);

}

void MainWindow::onSwapClicked()

{

QString leftText = ui->lineEditLeft->text();

QString rightText = ui->lineEditRight->text();

ui->lineEditLeft->setText(rightText);

ui->lineEditRight->setText(leftText);

}

void MainWindow::onConcatLeftClicked()

{

QString result = ui->lineEditLeft->text() + ui->lineEditRight->text();

ui->lineEditResult->setText(result);

}

void MainWindow::onConcatRightClicked()

{

QString result = ui->lineEditRight->text() + ui->lineEditLeft->text();

ui->lineEditResult->setText(result);

}

**Компиляция и запуск:**

Для сборки проекта будем использовать cmake (хорошо работает с Qt), после чего будем запускать программу. Алгоритм компиляции и запуска:

1. **cd build** — перед сборкой необходимо переместиться в ./build;

2. **cmake ..** — создание необходимых файлов сборки;

3. **cmake --build .** — последующая сборка после изменений, когда файлы уже были сгенерированы;

4. **./Lab6** — запуск скомпилированной программы.

**cmakeLists.txt:**

cmake\_minimum\_required(VERSION 3.16)

project(Lab6 VERSION 0.1 LANGUAGES CXX)

set(CMAKE\_AUTOUIC ON)

set(CMAKE\_AUTOMOC ON)

set(CMAKE\_AUTORCC ON)

set(CMAKE\_CXX\_STANDARD 17)

set(CMAKE\_CXX\_STANDARD\_REQUIRED ON)

find\_package(QT NAMES Qt6 Qt5 REQUIRED COMPONENTS Widgets)

find\_package(Qt${QT\_VERSION\_MAJOR} REQUIRED COMPONENTS Widgets)

set(PROJECT\_SOURCES

main.cpp

mainwindow.cpp

mainwindow.h

mainwindow.ui

)

if(${QT\_VERSION\_MAJOR} GREATER\_EQUAL 6)

qt\_add\_executable(Lab6

MANUAL\_FINALIZATION

${PROJECT\_SOURCES}

)

# Define target properties for Android with Qt 6 as:

# set\_property(TARGET Lab6 APPEND PROPERTY QT\_ANDROID\_PACKAGE\_SOURCE\_DIR

# ${CMAKE\_CURRENT\_SOURCE\_DIR}/android)

# For more information, see https://doc.qt.io/qt-6/qt-add-executable.html#target-creation

else()

if(ANDROID)

add\_library(Lab6 SHARED

${PROJECT\_SOURCES}

)

# Define properties for Android with Qt 5 after find\_package() calls as:

# set(ANDROID\_PACKAGE\_SOURCE\_DIR "${CMAKE\_CURRENT\_SOURCE\_DIR}/android")

else()

add\_executable(Lab6

${PROJECT\_SOURCES}

)

endif()

endif()

target\_link\_libraries(Lab6 PRIVATE Qt${QT\_VERSION\_MAJOR}::Widgets)

# Qt for iOS sets MACOSX\_BUNDLE\_GUI\_IDENTIFIER automatically since Qt 6.1.

# If you are developing for iOS or macOS you should consider setting an

# explicit, fixed bundle identifier manually though.

if(${QT\_VERSION} VERSION\_LESS 6.1.0)

set(BUNDLE\_ID\_OPTION MACOSX\_BUNDLE\_GUI\_IDENTIFIER com.example.Lab6)

endif()

set\_target\_properties(Lab6 PROPERTIES

${BUNDLE\_ID\_OPTION}

MACOSX\_BUNDLE\_BUNDLE\_VERSION ${PROJECT\_VERSION}

MACOSX\_BUNDLE\_SHORT\_VERSION\_STRING ${PROJECT\_VERSION\_MAJOR}.${PROJECT\_VERSION\_MINOR}

MACOSX\_BUNDLE TRUE

WIN32\_EXECUTABLE TRUE

)

include(GNUInstallDirs)

install(TARGETS Lab6

BUNDLE DESTINATION .

LIBRARY DESTINATION ${CMAKE\_INSTALL\_LIBDIR}

RUNTIME DESTINATION ${CMAKE\_INSTALL\_BINDIR}

)

if(QT\_VERSION\_MAJOR EQUAL 6)

qt\_finalize\_executable(Lab6)

endif()

**Вывод:**

В ходе выполнения лабораторной работы было разработано приложение с графическим интерфейсом, реализованное на основе фреймворка Qt. Основной целью приложения является выполнение операций с текстовыми строками, введёнными пользователем.

В процессе работы:

1. Изучены основы работы с классами QMainWindow, QLineEdit, QPushButton, и системой сигналов и слотов.
2. С помощью Qt Designer создан графический интерфейс приложения, содержащий:
   * Два текстовых поля для ввода строк.
   * Поле для отображения результата операций.
   * Набор кнопок для выполнения операций: реверс строк, объединение строк в различных порядках и их обмен местами.
3. Реализована логика обработки пользовательских действий:
   * Написаны обработчики для каждой кнопки, работающие с текстовыми строками и отображающие результат в соответствующем поле.
   * Использован метод connect для связывания сигналов (нажатие кнопок) с соответствующими слотами.
4. Проведено тестирование работы приложения:
   * Успешно выполнено тестирование всех операций, включая ввод данных, обработку текста, отображение результатов, а также корректное поведение приложения при различных комбинациях действий.

Приложение продемонстрировало работоспособность и соответствие поставленным требованиям. Разработка показала, что Qt является мощным инструментом для создания приложений с графическим интерфейсом.