Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное   
образовательное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

ИНСТИТУТ ЭЛЕКТРОННЫХ И ИНФОРМАЦИОННЫХ СИСТЕМ

КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

ЛАБОРАТОРНАЯ РАБОТА №5. БАЗЫ ДАННЫХ

Отчет по лабораторной работе №5 по учебной дисциплине «Алгоритмические языки и программирование»

по специальности 09.03.01 Информатика и вычислительная техника

|  |  |
| --- | --- |
|  | Руководитель  / И. Ю. Кулаков  «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |
|  | Студент группы 3091  / Р. А. Михайлов  «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

* 1. **Постановка задачи** – Напишите код программы и графический интерфейс, для работы с базой данных
  2. **Наименование задачи** – Лабораторная работа №5. Базы данных
  3. **Словесное описание**:

Напишите код программы и графический интерфейс, для работы с базой данных.

Создайте программу, которая помогает команде разработки управлять задачами и проектами. Каждая задача может содержать описание, статус выполнения, дедлайн и ответственного члена команды. Программа должна предоставлять возможность добавления новых задач, просмотра задач по ответственному, сортировки задач по статусу и дедлайну.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Объект программы | Имя этого объекта в программе | Тип данных | Диапазоны представления | Простая/  структура | Вход/  выход/  Константа |
| Что должно поступать на вход или выходить из программы, что хранится и тд. | Имя этого объекта | Указываем тип данных, то есть это | Указываем ограничения от пользователя:  Диапазон  Они должны быть меньше или равны тем что вышло в коде |  | если не входное, не выходное и не константа то оставляем пустым |

**Внешние спецификации данных (от лица пользователя)**

* 1. **Внешние спецификации функций программы**

В программе должна быть реализована работа с базой данных, помогающая команде разработчиков управлять задачами и проектами.

Должен быть реализован функционал:

– Добавление записи;

– Удаление записи;

– Удаление всех записей;

– Удаление таблицы;

– Редактирование записи;

– Вывод БД;

– Вывод по статусу;

– Вывод по ответственному;

– Подключение к таблице;

– Подключение к БД.

* 1. **Математическая формулировка задачи**

Введены ограничения на срок сдачи (минимальный год – 2024), так как логически правильно заносить предстоящие задачи, а не те, что уже выполнены или просрочены.

Введены ограничение на длину текста в полях «Название», «Описание», «Статус», «Ответственный» – максимально 200 символов, обосновано тем, что максимальный размер текста достаточен.

* 1. **Спецификации интерфейса**

Для работы с базой данных было использовано пространство имен SqlClient.

Модульное тестирование реализовано при помощи фреймворка CppUnitTest и встроенное тестирование, при помощи процедур и функций, проводимое как при запуске программы, так и при нажатии отдельной кнопки.

Плюсы интерфейса заключаются в том, что реализован контролируемый ввод, при попытке ввода некорректных данных срабатывает функция контроля ввода.

Реализована обработка повторного нажатия ключевых кнопок, что предотвращает некорректный вывод данных.

* 1. **Внешние данные тестирования**

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | | | T01 |
| Заголовок | | | Проверка выхода из приложения |
| Предусловие | | | Приложение не запущено |
| Ожидаемый результат | | | Успешный выход из приложения |
| Шаг | Действие | Предполагаемый результат | Pass/Fail |
| 1 | Запуск main.exe | Открытие главного окна приложения | Pass |
| 2 | Нажатие на кнопку «Выход» | Отображение окна «Вы точно хотите выйти?» | Pass |
| 3 | Нажатие на кнопку «Да» | Выход из приложения | Pass |
| Результат | | | Pass |

* 1. **Пример работающей программы** (руководство пользователя)

Показывает скриншоты программы, если реализация может быть непонятной, то пишем руководство.

1. **Проектирование задачи**

Реализовать приложение с графическим интерфейсом, использую библиотеку Windows Forms, для работы с базой данных. Модульное тестирование, используя фреймворк CppUnitTest и встроенное тестирование, при помощи процедур и функций, проводимое как при запуске программы, так и при нажатии отдельной кнопки.

* 1. **Наименование программы** – DBROM.exe
  2. **Уточненное словесное описание задачи**

Напишите код программы и графический интерфейс, для работы с базой данных.

Создайте программу, которая помогает команде разработки управлять задачами и проектами. Каждая задача может содержать описание, статус выполнения, дедлайн и ответственного члена команды. Программа должна предоставлять возможность добавления новых задач, просмотра задач по ответственному, сортировки задач по статусу и дедлайну.

* 1. **Выбор метода решения поставленной задачи**

Описываем методику вашего решения, обычно описываются уже существующие методы.

* 1. **Уточненные глобальные данные программы и пользовательские типы**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Объект программы | Имя этого объекта в программе | Характеристики | | | Как используется в программе |
| Тип данных | Диапазоны представления | Прост/структура/ константа |
| В каком файле содержится | Как называется |  | Диапазон типа данных, должен быть больше или равен ограничениям пользователя |  | описание |

* 1. **Декомпозиция функций**

Описывается каждая функция и процедура в программе

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Назначение | Имя | Параметры | | Процедура/ функция | Тестируется/  не тестируется |
| Вход:  тип | Выход: тип |
| Зачем она |  | Данные на входе | Данные на выходе | Функция возвращает значения  Процедура ничего не возвращает | Можно ли ее тестировать, укажите номер теста или пункта из 2.6 |
| Подключение к БД | Connect\_to\_DB | - | void | Процедура | Не тестируется |
| Создание новой записи | Insert\_data | String^ label, String^ description, String^ status, String^ admin, String^ deadline | void | Процедура | Не тестируется |
| Чтение одного ряда из таблицы | Read\_single\_row | DataGridView^ dgw, String^ record | void | Процедура | Не тестируется |
| Вывод полученной таблицы | Outpour\_table | DataGridView^ dgw, String^ query | void | Процедура | Не тестируется |
| Процедура, запускающая действия при нажатии "Новая запись" | новаяЗаписьToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии "Удалить все записи" | удалитьВсеЗаписиToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии "Вывод таблицы" | выводТабToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии "Вывод таблицы по статусу" | выводТаблицыПоСтатусуToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тустируется |
| Процедура запускающая действия при нажатии "А - Я" | аЯToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии "Я - А" | яАToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии "В прямом порядке" | вПрямомПорядкеToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии "В обратном порядке" | вОбратномПорядкеToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии "Удалить запись" | удалитьЗаписьToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии "Редактирование записи" | редактированиеЗаписиToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии "Информация" | информацияToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура подключения к БД при вызове конструктора | connection\_to\_constr | - | void | Процедура | Не тестируется |
| Процедура подключения в БД | connection\_to | - | void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии "Подключение к БД" | подключениеВБДToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии "Удалить таблицу" | удалитьТаблицуToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии "Пользовательский SQL-запрос" | пользовательскийSQLзапросToolStripMenuItem\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура получения ID записей | get\_id | - | void | Процедура | Не тестируется |
| Процедура удаления выбранной записи | delete\_rec | - | void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии кнопки "Отмена" | exit\_button\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии кнопки "Удалить" | delete\_button\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура проверки корректности введённой даты | check\_date | - | void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии "Добавить" | button1\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Функция получения данных из поля "Название" | get\_label | - | String^ | Функция | Не тестируется |
| Функция получения данных из поля "Описание" | get\_description | - | String^ | Функция | Не тестируется |
| Функция получения данных из поля "Статус" | Get\_status | - | String^ | Функция | Не тестируется |
| Функция получения данных из поля "Ответственный" | get\_admin | - | String^ | Функция | Не тестируется |
| Функция получения данных из поля "Срок сдачи" | Get\_deadline | - | String^ | Функция | Не тестируется |
| Процедура запускающая действия при нажатии "Отмена" | button2\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура проверки поля "Название" | check\_lenght1 | - | void | Процедура | Не тестируется |
| Процедура проверки поля "Описание" | check\_lenght2 | - | void | Процедура | Не тестируется |
| Процедура проверки поля "Статус" | check\_lenght3 | - | void | Процедура | Не тестируется |
| Процедура проверки поля "Ответственный" | check\_lenght4 | - | void | Процедура | Не тестируется |
| Процедура проверки поля "Срок сдачи" | textBox5\_TextChanged | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при изменении данных в поле "Название" | textBox1\_TextChanged | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при изменении данных в поле "Описание" | textBox2\_TextChanged | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при изменении данных в поле "Статус" | textBox3\_TextChanged | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при изменении данных в поле "Ответственный" | textBox4\_TextChanged | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура проверки корректности данных в поле "Срок сдачи" | check\_date | - | void | Процедура | Не тестируется |
| Процедура изменения выбранной записи | edit | - | void | Процедура | Не тестируется |
| Процедура получения ID записей | get\_id | - | void | Процедура | Не тестируется |
| Процедура получения данных выбранной записи | get\_curr\_data | - | void | Процедура | Не тестируется |
| Процедур запускающая действия при нажатии кнопки "Изменить" | rename\_button\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии кнопки "Отмена" | exit\_button\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при выборе изменяемого столбца | col\_comboBox\_SelectedIndexChanged | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при выборе изменяемой записи | id\_comboBox\_SelectedIndexChanged | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запускающая действия при изменении данных в поле "Введите новое значение" | new\_data\_TextChanged | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запуска службы SQL-сервера | start\_service | String^ serviceName | void | Процедура | Не тестируется |
| Процедура остановки службы SQL-сервера | stop\_serv | String^ serviceName | void | Процедура | Не тестируется |
| Процедура запуска остановки службы SQL-сервера | stop\_service | - | void | Процедура | Не тестируется |
| Функция получения значений из поля "Сервер" | Get\_DS | - | String^ | Процедура | Не тестируется |
| Функция получения значений из поля "База данных" | Get\_IC | - | String^ | Процедура | Не тестируется |
| Функция получения значений переменной tn | get\_table\_name | - | String^ | Процедура | Не тестируется |
| Процедура сохранения параметров текущего подключения | save\_curr\_to\_file | - | void | Процедура | Не тестируется |
| Процедура чтения параметров предыдущего подключения | get\_from\_file | - | void | Процедура | Не тестируется |
| Процедура запуска действий при нажатии кнопки "Подключиться" | connect\_button\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура запуска действий при нажатии кнопки "Отмена" | exit\_button\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура создания таблицы в БД | create\_table | (String^ t\_name | void | Процедура | Не тестируется |
| Процедура проверки существования таблицы с введённым названием | Check\_table | - | bool | Процедура | Не тестируется |
| Процедура присваивания глобальной переменной ds значений в поле "Сервер" | DS\_textBox\_TextChanged | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура присваивания глобальной переменной ic значений в поле "База данных" и включение скрытых полей | IC\_textBox\_TextChanged | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура проверки активности comboBox "Подключиться к существующей таблице" | enable\_check | - | void | Процедура | Не тестируется |
| Процедура отслеживания активности comboBox "Подключиться к существующей таблице" | exist\_checkBox\_CheckedChanged | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |
| Процедура ввода в список названий таблицы значения из текущей базы данных по специальным параметрам | table\_names | - | void | Процедура | Не тестируется |
| Процедура запускающая действия при нажатии кнопки "Запуск" | button1\_Click | System::Object^ sender, System::EventArgs^ e | System::Void | Процедура | Не тестируется |

* 1. **Блок-схема**

Блок-схемы в Приложении A.

Блок-схемы в Приложении Б.

Блок-схемы в Приложении В.

Блок-схемы в Приложении Г.

Блок-схемы в Приложении Д.

* 1. **Алгоритмизация**

Листинг файла MainWindow.h в Приложение Е.

Листинг файла ConnectionWindow.h в Приложение Ж.

Листинг файла DeleteRecord.h в Приложение И.

Листинг файла EditRecord.h в Приложение К.

Листинг файла NewRecord.h в Приложение Л.

Листинг файла MainWindow.cpp в Приложение М.

Листинг файла UnitTest.cpp в Приложение Н.

Листинг файла Functions.h в Приложение О.

Приложение А

Блок-схема

Приложение Б

Блок-схема

Приложение В

Блок-схема

Приложение Г

Блок-схема

Приложение Д

Блок-схема

Приложение Е

Листинг файла MainWindow.h

/\*

АЯиП Лабораторная работа #5

Заголовочный файл проекта DBROM

Михайлов Роман, группа 3091, 1 курс

01.04.2024 - 17.04.2024

\*/

#pragma once

#include "ConnectionWindow.h"

#include "NewRecord.h"

#include "EditRecord.h"

#include "DeleteRecord.h"

#include "ConnectionWindow.h"

namespace DBROM {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::Data::SqlClient;

using namespace Microsoft::VisualBasic;

using namespace System::IO;

using namespace System::Text;

using namespace Microsoft::VisualBasic;

/\*using namespace iTextSharp;

using namespace iTextSharp::text;

using namespace iTextSharp::text::pdf;\*/

/// <summary>

/// Сводка для Visual

/// </summary>

public ref class MainWindow : public System::Windows::Forms::Form

{

public:

MainWindow(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

this->Show();

connection\_to\_constr();

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~MainWindow()

{

cw->stop\_service();

if (components)

{

delete components;

}

}

protected:

private: SqlConnection^ connection; // Строка подключения

private: String^ data\_source = ""; // Название сервера

private: String^ initial\_catalog = ""; // Название базы данных

private: String^ t\_name = ""; // Имя таблицы

private: ConnectionWindow^ cw = gcnew ConnectionWindow(false); // Окно подключения

private: System::Windows::Forms::MenuStrip^ menuStrip1;

private: System::Windows::Forms::ToolStripMenuItem^ редактированиеToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ запросToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ выводВФайлToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ новаяЗаписьToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ удалитьЗаписьToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ удалитьВсеЗаписиToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ выводТаблицыПоОтветственномуToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ выводТаблицыПоСтатусуToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ выводТаблицыПоСрокуСдачиToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ выводТабToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ вПрямомПорядкеToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ вОбратномПорядкеToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ аЯToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ яАToolStripMenuItem;

private: System::Windows::Forms::DataGridView^ dataGridView1;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column1;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column2;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column3;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column4;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column5;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column6;

private: System::Windows::Forms::ToolStripMenuItem^ редактированиеЗаписиToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ подключениеToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ информацияToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ подключениеВБДToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ удалитьТаблицуToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ пользовательскийSQLзапросToolStripMenuItem;

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(MainWindow::typeid));

this->menuStrip1 = (gcnew System::Windows::Forms::MenuStrip());

this->подключениеToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->информацияToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->подключениеВБДToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->редактированиеToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->новаяЗаписьToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->удалитьЗаписьToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->удалитьВсеЗаписиToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->редактированиеЗаписиToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->удалитьТаблицуToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->запросToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->выводТабToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->выводТаблицыПоСтатусуToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->выводТаблицыПоСрокуСдачиToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->вПрямомПорядкеToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->вОбратномПорядкеToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->выводТаблицыПоОтветственномуToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->аЯToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->яАToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->выводВФайлToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->dataGridView1 = (gcnew System::Windows::Forms::DataGridView());

this->Column1 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column2 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column3 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column4 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column5 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column6 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->пользовательскийSQLзапросToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->menuStrip1->SuspendLayout();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView1))->BeginInit();

this->SuspendLayout();

//

// menuStrip1

//

this->menuStrip1->GripMargin = System::Windows::Forms::Padding(2, 2, 0, 2);

this->menuStrip1->ImageScalingSize = System::Drawing::Size(24, 24);

this->menuStrip1->Items->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(3) {

this->подключениеToolStripMenuItem,

this->редактированиеToolStripMenuItem, this->запросToolStripMenuItem

});

this->menuStrip1->Location = System::Drawing::Point(0, 0);

this->menuStrip1->Name = L"menuStrip1";

this->menuStrip1->Size = System::Drawing::Size(1898, 33);

this->menuStrip1->TabIndex = 19;

this->menuStrip1->Text = L"menuStrip1";

//

// подключениеToolStripMenuItem

//

this->подключениеToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(2) {

this->информацияToolStripMenuItem,

this->подключениеВБДToolStripMenuItem

});

this->подключениеToolStripMenuItem->Name = L"подключениеToolStripMenuItem";

this->подключениеToolStripMenuItem->Size = System::Drawing::Size(143, 29);

this->подключениеToolStripMenuItem->Text = L"Подключение";

//

// информацияToolStripMenuItem

//

this->информацияToolStripMenuItem->Name = L"информацияToolStripMenuItem";

this->информацияToolStripMenuItem->Size = System::Drawing::Size(271, 34);

this->информацияToolStripMenuItem->Text = L"Информация";

this->информацияToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::информацияToolStripMenuItem\_Click);

//

// подключениеВБДToolStripMenuItem

//

this->подключениеВБДToolStripMenuItem->Name = L"подключениеВБДToolStripMenuItem";

this->подключениеВБДToolStripMenuItem->Size = System::Drawing::Size(271, 34);

this->подключениеВБДToolStripMenuItem->Text = L"Подключение в БД";

this->подключениеВБДToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::подключениеВБДToolStripMenuItem\_Click);

//

// редактированиеToolStripMenuItem

//

this->редактированиеToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(5) {

this->новаяЗаписьToolStripMenuItem,

this->удалитьЗаписьToolStripMenuItem, this->удалитьВсеЗаписиToolStripMenuItem, this->редактированиеЗаписиToolStripMenuItem, this->удалитьТаблицуToolStripMenuItem

});

this->редактированиеToolStripMenuItem->Name = L"редактированиеToolStripMenuItem";

this->редактированиеToolStripMenuItem->Size = System::Drawing::Size(162, 29);

this->редактированиеToolStripMenuItem->Text = L"Редактирование";

//

// новаяЗаписьToolStripMenuItem

//

this->новаяЗаписьToolStripMenuItem->Name = L"новаяЗаписьToolStripMenuItem";

this->новаяЗаписьToolStripMenuItem->Size = System::Drawing::Size(308, 34);

this->новаяЗаписьToolStripMenuItem->Text = L"Новая запись";

this->новаяЗаписьToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::новаяЗаписьToolStripMenuItem\_Click);

//

// удалитьЗаписьToolStripMenuItem

//

this->удалитьЗаписьToolStripMenuItem->Name = L"удалитьЗаписьToolStripMenuItem";

this->удалитьЗаписьToolStripMenuItem->Size = System::Drawing::Size(308, 34);

this->удалитьЗаписьToolStripMenuItem->Text = L"Удалить запись";

this->удалитьЗаписьToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::удалитьЗаписьToolStripMenuItem\_Click);

//

// удалитьВсеЗаписиToolStripMenuItem

//

this->удалитьВсеЗаписиToolStripMenuItem->Name = L"удалитьВсеЗаписиToolStripMenuItem";

this->удалитьВсеЗаписиToolStripMenuItem->Size = System::Drawing::Size(308, 34);

this->удалитьВсеЗаписиToolStripMenuItem->Text = L"Удалить все записи";

this->удалитьВсеЗаписиToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::удалитьВсеЗаписиToolStripMenuItem\_Click);

//

// редактированиеЗаписиToolStripMenuItem

//

this->редактированиеЗаписиToolStripMenuItem->Name = L"редактированиеЗаписиToolStripMenuItem";

this->редактированиеЗаписиToolStripMenuItem->Size = System::Drawing::Size(308, 34);

this->редактированиеЗаписиToolStripMenuItem->Text = L"Редактирование записи";

this->редактированиеЗаписиToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::редактированиеЗаписиToolStripMenuItem\_Click);

//

// удалитьТаблицуToolStripMenuItem

//

this->удалитьТаблицуToolStripMenuItem->Name = L"удалитьТаблицуToolStripMenuItem";

this->удалитьТаблицуToolStripMenuItem->Size = System::Drawing::Size(308, 34);

this->удалитьТаблицуToolStripMenuItem->Text = L"Удалить таблицу";

this->удалитьТаблицуToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::удалитьТаблицуToolStripMenuItem\_Click);

//

// запросToolStripMenuItem

//

this->запросToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(5) {

this->выводТабToolStripMenuItem,

this->выводТаблицыПоСтатусуToolStripMenuItem, this->выводТаблицыПоСрокуСдачиToolStripMenuItem, this->выводТаблицыПоОтветственномуToolStripMenuItem,

this->пользовательскийSQLзапросToolStripMenuItem

});

this->запросToolStripMenuItem->Name = L"запросToolStripMenuItem";

this->запросToolStripMenuItem->Size = System::Drawing::Size(87, 29);

this->запросToolStripMenuItem->Text = L"Запрос";

//

// выводТабToolStripMenuItem

//

this->выводТабToolStripMenuItem->Name = L"выводТабToolStripMenuItem";

this->выводТабToolStripMenuItem->Size = System::Drawing::Size(404, 34);

this->выводТабToolStripMenuItem->Text = L"Вывод таблицы";

this->выводТабToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::выводТабToolStripMenuItem\_Click);

//

// выводТаблицыПоСтатусуToolStripMenuItem

//

this->выводТаблицыПоСтатусуToolStripMenuItem->Name = L"выводТаблицыПоСтатусуToolStripMenuItem";

this->выводТаблицыПоСтатусуToolStripMenuItem->Size = System::Drawing::Size(404, 34);

this->выводТаблицыПоСтатусуToolStripMenuItem->Text = L"Вывод таблицы по статусу";

this->выводТаблицыПоСтатусуToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::выводТаблицыПоСтатусуToolStripMenuItem\_Click);

//

// выводТаблицыПоСрокуСдачиToolStripMenuItem

//

this->выводТаблицыПоСрокуСдачиToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(2) {

this->вПрямомПорядкеToolStripMenuItem,

this->вОбратномПорядкеToolStripMenuItem

});

this->выводТаблицыПоСрокуСдачиToolStripMenuItem->Name = L"выводТаблицыПоСрокуСдачиToolStripMenuItem";

this->выводТаблицыПоСрокуСдачиToolStripMenuItem->Size = System::Drawing::Size(404, 34);

this->выводТаблицыПоСрокуСдачиToolStripMenuItem->Text = L"Вывод таблицы по сроку сдачи";

//

// вПрямомПорядкеToolStripMenuItem

//

this->вПрямомПорядкеToolStripMenuItem->Name = L"вПрямомПорядкеToolStripMenuItem";

this->вПрямомПорядкеToolStripMenuItem->Size = System::Drawing::Size(285, 34);

this->вПрямомПорядкеToolStripMenuItem->Text = L"В прямом порядке";

this->вПрямомПорядкеToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::вПрямомПорядкеToolStripMenuItem\_Click);

//

// вОбратномПорядкеToolStripMenuItem

//

this->вОбратномПорядкеToolStripMenuItem->Name = L"вОбратномПорядкеToolStripMenuItem";

this->вОбратномПорядкеToolStripMenuItem->Size = System::Drawing::Size(285, 34);

this->вОбратномПорядкеToolStripMenuItem->Text = L"В обратном порядке";

this->вОбратномПорядкеToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::вОбратномПорядкеToolStripMenuItem\_Click);

//

// выводТаблицыПоОтветственномуToolStripMenuItem

//

this->выводТаблицыПоОтветственномуToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(2) {

this->аЯToolStripMenuItem,

this->яАToolStripMenuItem

});

this->выводТаблицыПоОтветственномуToolStripMenuItem->Name = L"выводТаблицыПоОтветственномуToolStripMenuItem";

this->выводТаблицыПоОтветственномуToolStripMenuItem->Size = System::Drawing::Size(404, 34);

this->выводТаблицыПоОтветственномуToolStripMenuItem->Text = L"Вывод таблицы по ответственному";

//

// аЯToolStripMenuItem

//

this->аЯToolStripMenuItem->Name = L"аЯToolStripMenuItem";

this->аЯToolStripMenuItem->Size = System::Drawing::Size(154, 34);

this->аЯToolStripMenuItem->Text = L"А - Я";

this->аЯToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::аЯToolStripMenuItem\_Click);

//

// яАToolStripMenuItem

//

this->яАToolStripMenuItem->Name = L"яАToolStripMenuItem";

this->яАToolStripMenuItem->Size = System::Drawing::Size(154, 34);

this->яАToolStripMenuItem->Text = L"Я - А";

this->яАToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::яАToolStripMenuItem\_Click);

//

// выводВФайлToolStripMenuItem

//

this->выводВФайлToolStripMenuItem->Name = L"выводВФайлToolStripMenuItem";

this->выводВФайлToolStripMenuItem->Size = System::Drawing::Size(32, 19);

//

// dataGridView1

//

this->dataGridView1->AllowUserToAddRows = false;

this->dataGridView1->AllowUserToDeleteRows = false;

this->dataGridView1->AutoSizeColumnsMode = System::Windows::Forms::DataGridViewAutoSizeColumnsMode::Fill;

this->dataGridView1->ColumnHeadersHeightSizeMode = System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;

this->dataGridView1->Columns->AddRange(gcnew cli::array< System::Windows::Forms::DataGridViewColumn^ >(6) {

this->Column1,

this->Column2, this->Column3, this->Column4, this->Column5, this->Column6

});

this->dataGridView1->Dock = System::Windows::Forms::DockStyle::Fill;

this->dataGridView1->Location = System::Drawing::Point(0, 33);

this->dataGridView1->Name = L"dataGridView1";

this->dataGridView1->ReadOnly = true;

this->dataGridView1->RowHeadersWidth = 40;

this->dataGridView1->RowTemplate->Height = 28;

this->dataGridView1->Size = System::Drawing::Size(1898, 992);

this->dataGridView1->TabIndex = 16;

//

// Column1

//

this->Column1->FillWeight = 20;

this->Column1->HeaderText = L"ID";

this->Column1->MinimumWidth = 8;

this->Column1->Name = L"Column1";

this->Column1->ReadOnly = true;

//

// Column2

//

this->Column2->HeaderText = L"Название";

this->Column2->MinimumWidth = 8;

this->Column2->Name = L"Column2";

this->Column2->ReadOnly = true;

//

// Column3

//

this->Column3->HeaderText = L"Описание";

this->Column3->MinimumWidth = 8;

this->Column3->Name = L"Column3";

this->Column3->ReadOnly = true;

//

// Column4

//

this->Column4->HeaderText = L"Статус";

this->Column4->MinimumWidth = 8;

this->Column4->Name = L"Column4";

this->Column4->ReadOnly = true;

//

// Column5

//

this->Column5->HeaderText = L"Ответственный";

this->Column5->MinimumWidth = 8;

this->Column5->Name = L"Column5";

this->Column5->ReadOnly = true;

//

// Column6

//

this->Column6->HeaderText = L"Срок сдачи";

this->Column6->MinimumWidth = 8;

this->Column6->Name = L"Column6";

this->Column6->ReadOnly = true;

//

// пользовательскийSQLзапросToolStripMenuItem

//

this->пользовательскийSQLзапросToolStripMenuItem->Name = L"пользовательскийSQLзапросToolStripMenuItem";

this->пользовательскийSQLзапросToolStripMenuItem->Size = System::Drawing::Size(404, 34);

this->пользовательскийSQLзапросToolStripMenuItem->Text = L"Пользовательский SQL-запрос";

this->пользовательскийSQLзапросToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainWindow::пользовательскийSQLзапросToolStripMenuItem\_Click);

//

// MainWindow

//

this->AutoScaleDimensions = System::Drawing::SizeF(9, 20);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(1898, 1025);

this->Controls->Add(this->dataGridView1);

this->Controls->Add(this->menuStrip1);

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->MainMenuStrip = this->menuStrip1;

this->MaximumSize = System::Drawing::Size(2869, 1631);

this->MinimumSize = System::Drawing::Size(498, 491);

this->Name = L"MainWindow";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"DBROM";

this->WindowState = System::Windows::Forms::FormWindowState::Maximized;

this->menuStrip1->ResumeLayout(false);

this->menuStrip1->PerformLayout();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView1))->EndInit();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

//

// Подключеник к БД

//

private: void connect\_to\_DB()

{

SqlConnectionStringBuilder^ strbild = gcnew SqlConnectionStringBuilder();

strbild->DataSource = data\_source;

strbild->InitialCatalog = initial\_catalog;

strbild->IntegratedSecurity = true;

connection = gcnew SqlConnection(Convert::ToString(strbild));

}

//

// Процедура создания новой записи

//

private: void insert\_data(String^ label, String^ description, String^ status, String^ admin, String^ deadline)

{

connect\_to\_DB();

String^ command\_text = "INSERT INTO " + t\_name +"(label, description, status, admin, deadline) VALUES(@Label, @Description, @Status, @Admin, @Deadline)";

SqlCommand^ cmd = gcnew SqlCommand(command\_text, connection);

cmd->Parameters->AddWithValue("@Label", label);

cmd->Parameters->AddWithValue("@Description", description);

cmd->Parameters->AddWithValue("@Status", status);

cmd->Parameters->AddWithValue("@Admin", admin);

cmd->Parameters->AddWithValue("@Deadline", Convert::ToDateTime(deadline));

connection->Open();

cmd->ExecuteNonQuery();

connection->Close();

output\_table(dataGridView1, "SELECT \* FROM " + t\_name);

}

//

// Процедура чтения одного ряда из таблицы запроса

//

private: void read\_single\_row(DataGridView^ dgw, IDataRecord^ record)

{

dgw->Rows->Add(record->GetInt32(0), record->GetString(1), record->GetString(2), record->GetString(3), record->GetString(4), Convert::ToString(record->GetDateTime(5))->Remove(10));

}

//

// Процедура вывода полученной таблицы

//

private: void output\_table(DataGridView^ dgw, String^ query)

{

dgw->Rows->Clear();

connect\_to\_DB();

SqlCommand^ cmd = gcnew SqlCommand(query, connection);

connection->Open();

SqlDataReader^ reader = cmd->ExecuteReader();

while (reader->Read())

{

read\_single\_row(dgw, reader);

}

reader->Close();

connection->Close();

}

//

// Процедура запускающая действия при нажатии "Новая запись"

//

private: System::Void новаяЗаписьToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

NewRecord^ nr = gcnew NewRecord();

nr->ShowDialog(this);

if (nr->get\_label() != "" && nr->get\_description() != "" && nr->get\_status() != "" && nr->get\_admin() != "" && nr->get\_deadline() != "")

{

insert\_data(nr->get\_label(), nr->get\_description(), nr->get\_status(), nr->get\_admin(), nr->get\_deadline());

}

}

//

// Процедура запускающая действия при нажатии "Удалить все записи"

//

private: System::Void удалитьВсеЗаписиToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

System::Windows::Forms::DialogResult result = MessageBox::Show("Вы уверены, что хотите удалить все записи?", "Уведомление", MessageBoxButtons::YesNo, MessageBoxIcon::Question);

if (result == System::Windows::Forms::DialogResult::Yes)

{

connect\_to\_DB();

String^ command\_text = "DELETE " + t\_name;

SqlCommand^ cmd = gcnew SqlCommand(command\_text, connection);

connection->Open();

cmd->ExecuteNonQuery();

connection->Close();

}

}

//

// Процедура запускающая действия при нажатии "Вывод таблицы"

//

private: System::Void выводТабToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

output\_table(dataGridView1, "SELECT \* FROM " + t\_name);

}

//

// Процедура запускающая действия при нажатии "Вывод таблицы по статусу"

//

private: System::Void выводТаблицыПоСтатусуToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

output\_table(dataGridView1, "SELECT \* FROM " + t\_name + " ORDER BY status");

}

//

// Процедура запускающая действия при нажатии "А - Я"

//

private: System::Void аЯToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

output\_table(dataGridView1, "SELECT id, admin, label, description, status, deadline FROM " + t\_name + " ORDER BY admin ASC");

}

//

// Процедура запускающая действия при нажатии "Я - А"

//

private: System::Void яАToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

output\_table(dataGridView1, "SELECT id, admin, label, description, status, deadline FROM " + t\_name + " ORDER BY admin DESC");

}

//

// Процедура запускающая действия при нажатии "В прямом порядке"

//

private: System::Void вПрямомПорядкеToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

output\_table(dataGridView1, "SELECT \* FROM " + t\_name + " ORDER BY deadline ASC");

}

//

// Процедура запускающая действия при нажатии "В обратном порядке"

//

private: System::Void вОбратномПорядкеToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

output\_table(dataGridView1, "SELECT \* FROM " + t\_name + " ORDER BY deadline DESC");

}

//

// Процедура запускающая действия при нажатии "Удалить запись"

//

private: System::Void удалитьЗаписьToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

DeleteRecord^ dr = gcnew DeleteRecord(data\_source, initial\_catalog, t\_name);

dr->ShowDialog(this);

output\_table(dataGridView1, "SELECT \* FROM " + t\_name);

}

//

// Процедура запускающая действия при нажатии "Редактирование записи"

//

private: System::Void редактированиеЗаписиToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

EditRecord^ er = gcnew EditRecord(data\_source, initial\_catalog, t\_name);

er->ShowDialog(this);

output\_table(dataGridView1, "SELECT \* FROM " + t\_name);

}

//

// Процедура запускающая действия при нажатии "Информация"

//

private: System::Void информацияToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

MessageBox::Show("Для работы программы необходимо подключиться к SQL серверу. На сервере необходимо создать базу дынных и таблицу с произвольными названиями.\r\n\r\nДля корректной работы программы таблица базы данных должна удовлетворять следующим условиям:\r\n1. У колонок должны быть следующие названия, сначала идет имя колонки, в скобках тип данных: ID (int), label (varchar(200)), description (varchar(200)), status (varchar(200)), admin (varchar(200)), deadline (date)\r\n2. У колонки ID должен быть включён параметр \"Identity Specification\"\r\n3. У всех колонок должен быть отключён параметр \"Allow Nulls\"\r\n\r\nДля подключения к созданной БД выберите \"Подключение к БД\", в открывшемся окне введите название сервера и базы данных", "Информация о подключении к локальному SQL серверу", MessageBoxButtons::OK, MessageBoxIcon::Information);

}

//

// Процедура подключения к БД при вызове конструктора

//

private: void connection\_to\_constr()

{

ConnectionWindow^ cw = gcnew ConnectionWindow(false);

cw->ShowDialog(this);

data\_source = cw->get\_DS();

initial\_catalog = cw->get\_IC();

t\_name = cw->get\_table\_name();

if (cw->ex == "1")

{

cw->stop\_service();

this->Close();

}

else

{

if (data\_source == "" && initial\_catalog == "" && this->t\_name == "") this->Close();

else

{

output\_table(dataGridView1, "SELECT \* FROM " + t\_name);

if (data\_source != "" && initial\_catalog != "" && t\_name != "")

{

if (this->Text != "DBROM") this->Text = this->Text->Remove(5) + " — " + t\_name;

else this->Text = this->Text + " — " + t\_name;

}

}

}

}

//

// Процедура подключения в БД

//

private: void connection\_to()

{

ConnectionWindow^ cw = gcnew ConnectionWindow(true);

cw->ShowDialog(this);

data\_source = cw->get\_DS();

initial\_catalog = cw->get\_IC();

t\_name = cw->get\_table\_name();

if(cw->ex != "1")

{

if (this->Text != "DBROM") this->Text = this->Text->Remove(5) + " — " + t\_name;

else this->Text = this->Text + " — " + t\_name;

}

}

//

// Процедура запускающая действия при нажатии "Подключение к БД"

//

private: System::Void подключениеВБДToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

connection\_to();

if(t\_name != "")

output\_table(dataGridView1, "SELECT \* FROM " + t\_name);

}

//

// Процедура запускающая действия при нажатии "Удалить таблицу"

//

private: System::Void удалитьТаблицуToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

System::Windows::Forms::DialogResult result = MessageBox::Show("Вы уверены, что хотите удалить эту таблицу?", "Уведомление", MessageBoxButtons::YesNo, MessageBoxIcon::Question);

if (result == System::Windows::Forms::DialogResult::Yes)

{

dataGridView1->Rows->Clear();

connect\_to\_DB();

String^ query = "DROP TABLE " + t\_name;

SqlCommand^ cmd = gcnew SqlCommand(query, connection);

connection->Open();

cmd->ExecuteNonQuery();

connection->Close();

подключениеВБДToolStripMenuItem\_Click(sender, e);

}

}

//

//

//

//private: System::Void выводВФайлToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

//{

//

//}

//

// Процедура запускающая действия при нажатии "Пользовательский SQL-запрос"

//

private: System::Void пользовательскийSQLзапросToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

int x = this->Left + (this->Width / 2) - 200;

int y = this->Top + (this->Height / 2) - 100;

int check = 0;

String^ query = "";

while (check != 1)

{

query = Interaction::InputBox("Введите SQL-запрос", "SQL-запрос", query, x, y);

try

{

output\_table(dataGridView1, query);

check = 1;

}

catch (System::Exception^)

{

MessageBox::Show("Введён некорректный SQL-запрос", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

}

}

}

};

}

Приложение Ж

Листинг файла ConnectionWindow.h

/\*

АЯиП Лабораторная работа #5

Заголовочный файл проекта DBROM

Михайлов Роман, группа 3091, 1 курс

01.04.2024 - 17.04.2024

\*/

#pragma once

namespace DBROM {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace SqlClient;

using namespace System::Text;

using namespace System::IO;

using namespace System::ServiceProcess;

/// <summary>

/// Сводка для ConnectionWindow

/// </summary>

public ref class ConnectionWindow : public System::Windows::Forms::Form

{

public:

ConnectionWindow(bool not\_constr)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

get\_from\_file();

this->not\_constr = not\_constr;

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~ConnectionWindow()

{

if (components)

{

delete components;

}

}

private: String^ ds = ""; // Строка настройки подключения к БД DataSource

private: String^ ic = ""; // Строка настройки подключения к БД InitialCatalog

private: String^ tn = ""; // Имя таблицы к которой происходит подключение

private: String^ serv = ""; // Имя службы локального SQL сервера

public: String^ ex = ""; // Передаваемая переменная, обозначающая закрытие окна присоединения

private: bool not\_constr; // Переменная, обозначающая открытие окна не через конструктор

private: System::Windows::Forms::TextBox^ DS\_textBox;

private: System::Windows::Forms::TextBox^ IC\_textBox;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Button^ exit\_button;

private: System::Windows::Forms::Button^ rename\_button;

private: SqlConnection^ connection;

private: System::Windows::Forms::TextBox^ table\_name\_textBox;

private: System::Windows::Forms::ComboBox^ tables\_comboBox;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::TextBox^ service\_textBox;

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::CheckBox^ exist\_checkBox;

private: System::Windows::Forms::Label^ create\_label;

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(ConnectionWindow::typeid));

this->DS\_textBox = (gcnew System::Windows::Forms::TextBox());

this->IC\_textBox = (gcnew System::Windows::Forms::TextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->exit\_button = (gcnew System::Windows::Forms::Button());

this->rename\_button = (gcnew System::Windows::Forms::Button());

this->table\_name\_textBox = (gcnew System::Windows::Forms::TextBox());

this->tables\_comboBox = (gcnew System::Windows::Forms::ComboBox());

this->label3 = (gcnew System::Windows::Forms::Label());

this->service\_textBox = (gcnew System::Windows::Forms::TextBox());

this->button1 = (gcnew System::Windows::Forms::Button());

this->exist\_checkBox = (gcnew System::Windows::Forms::CheckBox());

this->create\_label = (gcnew System::Windows::Forms::Label());

this->SuspendLayout();

//

// DS\_textBox

//

this->DS\_textBox->Enabled = false;

this->DS\_textBox->Location = System::Drawing::Point(176, 77);

this->DS\_textBox->Name = L"DS\_textBox";

this->DS\_textBox->Size = System::Drawing::Size(363, 26);

this->DS\_textBox->TabIndex = 0;

this->DS\_textBox->TextChanged += gcnew System::EventHandler(this, &ConnectionWindow::DS\_textBox\_TextChanged);

//

// IC\_textBox

//

this->IC\_textBox->Enabled = false;

this->IC\_textBox->Location = System::Drawing::Point(176, 127);

this->IC\_textBox->Name = L"IC\_textBox";

this->IC\_textBox->Size = System::Drawing::Size(363, 26);

this->IC\_textBox->TabIndex = 1;

this->IC\_textBox->TextChanged += gcnew System::EventHandler(this, &ConnectionWindow::IC\_textBox\_TextChanged);

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(18, 77);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(69, 20);

this->label1->TabIndex = 2;

this->label1->Text = L"Сервер:";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(18, 127);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(110, 20);

this->label2->TabIndex = 3;

this->label2->Text = L"База данных:";

//

// exit\_button

//

this->exit\_button->Location = System::Drawing::Point(237, 297);

this->exit\_button->Name = L"exit\_button";

this->exit\_button->Size = System::Drawing::Size(145, 40);

this->exit\_button->TabIndex = 12;

this->exit\_button->Text = L"Отмена";

this->exit\_button->UseVisualStyleBackColor = true;

this->exit\_button->Click += gcnew System::EventHandler(this, &ConnectionWindow::exit\_button\_Click);

//

// rename\_button

//

this->rename\_button->Location = System::Drawing::Point(394, 297);

this->rename\_button->Name = L"rename\_button";

this->rename\_button->Size = System::Drawing::Size(145, 40);

this->rename\_button->TabIndex = 11;

this->rename\_button->Text = L"Подключиться";

this->rename\_button->UseVisualStyleBackColor = true;

this->rename\_button->Click += gcnew System::EventHandler(this, &ConnectionWindow::connect\_button\_Click);

//

// table\_name\_textBox

//

this->table\_name\_textBox->Enabled = false;

this->table\_name\_textBox->Location = System::Drawing::Point(219, 227);

this->table\_name\_textBox->Name = L"table\_name\_textBox";

this->table\_name\_textBox->Size = System::Drawing::Size(320, 26);

this->table\_name\_textBox->TabIndex = 18;

//

// tables\_comboBox

//

this->tables\_comboBox->Enabled = false;

this->tables\_comboBox->FormattingEnabled = false;

this->tables\_comboBox->Location = System::Drawing::Point(375, 177);

this->tables\_comboBox->Name = L"tables\_comboBox";

this->tables\_comboBox->Size = System::Drawing::Size(164, 28);

this->tables\_comboBox->TabIndex = 19;

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(18, 27);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(147, 20);

this->label3->TabIndex = 20;

this->label3->Text = L"Название службы:";

//

// service\_textBox

//

this->service\_textBox->Location = System::Drawing::Point(176, 27);

this->service\_textBox->Name = L"service\_textBox";

this->service\_textBox->Size = System::Drawing::Size(228, 26);

this->service\_textBox->TabIndex = 21;

//

// button1

//

this->button1->Location = System::Drawing::Point(410, 27);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(129, 33);

this->button1->TabIndex = 22;

this->button1->Text = L"Запуск";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &ConnectionWindow::button1\_Click);

//

// exist\_checkBox

//

this->exist\_checkBox->AutoSize = true;

this->exist\_checkBox->Location = System::Drawing::Point(22, 177);

this->exist\_checkBox->Name = L"exist\_checkBox";

this->exist\_checkBox->Size = System::Drawing::Size(351, 24);

this->exist\_checkBox->TabIndex = 23;

this->exist\_checkBox->Text = L"Подключиться к существующей таблице:";

this->exist\_checkBox->UseVisualStyleBackColor = true;

this->exist\_checkBox->Visible = not\_constr;

this->exist\_checkBox->CheckedChanged += gcnew System::EventHandler(this, &ConnectionWindow::exist\_checkBox\_CheckedChanged);

//

// create\_label

//

this->create\_label->AutoSize = true;

this->create\_label->Enabled = false;

this->create\_label->Location = System::Drawing::Point(18, 227);

this->create\_label->Name = L"create\_label";

this->create\_label->Size = System::Drawing::Size(195, 20);

this->create\_label->TabIndex = 24;

this->create\_label->Text = L"Создать новую таблицу:";

//

// ConnectionWindow

//

this->AutoScaleDimensions = System::Drawing::SizeF(9, 20);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(551, 362);

this->Controls->Add(this->create\_label);

this->Controls->Add(this->exist\_checkBox);

this->Controls->Add(this->button1);

this->Controls->Add(this->service\_textBox);

this->Controls->Add(this->label3);

this->Controls->Add(this->tables\_comboBox);

this->Controls->Add(this->table\_name\_textBox);

this->Controls->Add(this->exit\_button);

this->Controls->Add(this->rename\_button);

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->IC\_textBox);

this->Controls->Add(this->DS\_textBox);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedToolWindow;

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->Name = L"ConnectionWindow";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"Подключение";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

//

// Процедура запуска службы SQL-сервера

//

private: void start\_service(String^ serviceName)

{

ServiceController^ service = gcnew ServiceController(serviceName);

try

{

if (service->Status != ServiceControllerStatus::Running)

{

service->Start();

service->WaitForStatus(ServiceControllerStatus::Running, TimeSpan::FromMinutes(1));

}

}

catch (System::Exception^)

{

MessageBox::Show("Такой службы нет", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

this->service\_textBox->Text = "";

}

serv = this->service\_textBox->Text;

}

//

// Процедура остановки службы SQL-сервера

//

private: void stop\_serv(String^ serviceName)

{

ServiceController^ service = gcnew ServiceController(serviceName);

if (service->Status != ServiceControllerStatus::Stopped)

{

service->Stop();

service->WaitForStatus(ServiceControllerStatus::Stopped, TimeSpan::FromMinutes(1));

}

}

//

// Процедура запуска остановки службы SQL-сервера

//

public: void stop\_service()

{

stop\_serv(this->service\_textBox->Text);

}

//

// Функция получения значений из поля "Сервер"

//

public: String^ get\_DS()

{

return this->DS\_textBox->Text;

}

//

// Функция получения значений из поля "База данных"

//

public: String^ get\_IC()

{

return this->IC\_textBox->Text;

}

//

// Функция получения значений переменной tn

//

public: String^ get\_table\_name()

{

return tn;

}

//

// Процедура сохранения параметров текущего подключения

//

private: void save\_curr\_to\_file()

{

StreamWriter^ writer = gcnew StreamWriter("ConnectionProperties.dbrom");

writer->Write(this->service\_textBox->Text + "\r\n" + this->DS\_textBox->Text + "\r\n" + this->IC\_textBox->Text);

writer->Close();

}

//

// Процедура чтения параметров предыдущего подключения

//

private: void get\_from\_file()

{

if (File::Exists("ConnectionProperties.dbrom"))

{

StreamReader^ reader = gcnew StreamReader("ConnectionProperties.dbrom");

String^ serv = reader->ReadLine();

String^ ds = reader->ReadLine();

String^ ic = reader->ReadLine();

reader->Close();

this->service\_textBox->Text = serv;

this->DS\_textBox->Text = ds;

this->IC\_textBox->Text = ic;

}

}

//

// Процедура запуска действий при нажатии кнопки "Подключиться"

//

private: System::Void connect\_button\_Click(System::Object^ sender, System::EventArgs^ e)

{

ServiceController^ service = gcnew ServiceController(this->service\_textBox->Text);

if (service->Status == ServiceControllerStatus::Running)

{

if (this->tables\_comboBox->SelectedIndex != -1 || this->table\_name\_textBox->Text != "")

{

save\_curr\_to\_file();

if (exist\_checkBox->Checked == true)

{

tn = Convert::ToString(tables\_comboBox->Items[tables\_comboBox->SelectedIndex]);

this->Close();

}

else if (exist\_checkBox->Checked == false)

{

tn = this->table\_name\_textBox->Text;

if (check\_table())

create\_table(table\_name\_textBox->Text);

}

}

else MessageBox::Show("Таблицы не выбрана", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

}

else MessageBox::Show("SQL служба не запущена", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

}

//

// Процедура запуска действий при нажатии кнопки "Отмена"

//

private: System::Void exit\_button\_Click(System::Object^ sender, System::EventArgs^ e)

{

this->ex = "1";

this->tables\_comboBox->Items->Add("");

this->tables\_comboBox->SelectedIndex = 0;

this->Close();

}

//

// Процедура подключения к БД

//

private: void connect\_to\_DB()

{

SqlConnectionStringBuilder^ strbild = gcnew SqlConnectionStringBuilder();

strbild->DataSource = ds;

strbild->InitialCatalog = ic;

strbild->IntegratedSecurity = true;

connection = gcnew SqlConnection(Convert::ToString(strbild));

}

//

// Процедура создания таблицы в БД

//

private: void create\_table(String^ t\_name)

{

connect\_to\_DB();

String^ command\_text = "CREATE TABLE " + t\_name + " (ID INT IDENTITY(1, 1) PRIMARY KEY NOT NULL, label varchar(200) NOT NULL, description varchar(200) NOT NULL, status varchar(200) NOT NULL, admin varchar(200) NOT NULL, deadline date NOT NULL)";

connection->Open();

SqlCommand^ cmd = gcnew SqlCommand(command\_text, connection);

SqlDataReader^ reader = cmd->ExecuteReader();

connection->Close();

this->Close();

}

//

// Процедура проверки существования таблицы с введённым названием

//

private: bool check\_table()

{

if (DS\_textBox->Text != "" && IC\_textBox->Text != "")

{

String^ query = "IF EXISTS (SELECT 1 FROM sys.tables WHERE name = '" + this->table\_name\_textBox->Text + "') SELECT '1' as prove ELSE SELECT '0' as prove";

connect\_to\_DB();

SqlCommand^ cmd = gcnew SqlCommand(query, connection);

connection->Open();

SqlDataReader^ reader = cmd->ExecuteReader();

String^ check = "";

while (reader->Read()) check = reader["prove"]->ToString();

connection->Close();

if (check == "1")

{

MessageBox::Show("Таблица с таким именем уже существует", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

return false;

}

else if (check == "0") return true;

}

}

//

// Процедура присваивания глобальной переменной ds значений в поле "Сервер"

//

private: System::Void DS\_textBox\_TextChanged(System::Object^ sender, System::EventArgs^ e)

{

ds = this->DS\_textBox->Text;

}

//

// Процедура присваивания глобальной переменной ic значений в поле "База данных" и включение скрытых полей

//

private: System::Void IC\_textBox\_TextChanged(System::Object^ sender, System::EventArgs^ e)

{

ic = this->IC\_textBox->Text;

if (this->service\_textBox->Text != "")

{

ServiceController^ service = gcnew ServiceController(this->service\_textBox->Text);

if (service->Status == ServiceControllerStatus::Running)

{

if (this->IC\_textBox->Text != "" && this->DS\_textBox->Text != "")

{

this->exist\_checkBox->Visible = true;

this->tables\_comboBox->Enabled = true;

this->table\_name\_textBox->Enabled = false;

this->create\_label->Enabled = false;

}

else

{

this->exist\_checkBox->Visible = false;

this->tables\_comboBox->Enabled = false;

this->table\_name\_textBox->Enabled = true;

this->create\_label->Enabled = true;

}

}

}

}

//

// Процедура проверки активности comboBox "Подключиться к существующей таблице"

//

private: void enable\_check()

{

if (exist\_checkBox->Checked == true)

{

this->create\_label->Enabled = false;

this->table\_name\_textBox->Enabled = false;

this->tables\_comboBox->Enabled = true;

}

else

{

this->create\_label->Enabled = true;

this->table\_name\_textBox->Enabled = true;

this->tables\_comboBox->Enabled = false;

}

}

//

// Процедура отслеживания активности comboBox "Подключиться к существующей таблице"

//

private: System::Void exist\_checkBox\_CheckedChanged(System::Object^ sender, System::EventArgs^ e)

{

enable\_check();

this->tables\_comboBox->Items->Clear();

table\_names();

}

//

// Процедура ввода в список названий таблицы значения из текущей базы данных по специальным параметрам

//

private: void table\_names()

{

connect\_to\_DB();

String^ query = "select distinct table\_name from information\_schema.columns where column\_name in ('label','description','status','admin','deadline')";

connection->Open();

SqlCommand^ cmd = gcnew SqlCommand(query, connection);

SqlDataReader^ reader = cmd->ExecuteReader();

String^ name\_temp = "";

while (reader->Read())

{

name\_temp = reader["table\_name"]->ToString();

tables\_comboBox->Items->Add(String::Format("{0}", name\_temp));

}

connection->Close();

}

//

// Процедура запускающая действия при нажатии кнопки "Запуск"

//

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e)

{

start\_service(this->service\_textBox->Text);

if (this->service\_textBox->Text != "" && serv == this->service\_textBox->Text)

{

ServiceController^ service = gcnew ServiceController(this->service\_textBox->Text);

if (service->Status == ServiceControllerStatus::Running)

{

this->DS\_textBox->Enabled = true;

this->IC\_textBox->Enabled = true;

if(DS\_textBox->Text != "" && IC\_textBox->Text != "") this->create\_label->Enabled = true;

save\_curr\_to\_file();

}

else

{

this->DS\_textBox->Enabled = false;

this->IC\_textBox->Enabled = false;

}

}

if (this->DS\_textBox->Text != "" && this->IC\_textBox->Text != "" && this->service\_textBox->Text != "")

{

this->exist\_checkBox->Visible = true;

this->create\_label->Enabled = true;

this->table\_name\_textBox->Enabled = true;

}

else

{

this->tables\_comboBox->Enabled = false;

this->exist\_checkBox->Visible = false;

this->create\_label->Enabled = false;

this->table\_name\_textBox->Enabled = false;

}

}

};

}

Приложение И

Листинг файла DeleteRecord.h

/\*

АЯиП Лабораторная работа #5

Заголовочный файл проекта DBROM

Михайлов Роман, группа 3091, 1 курс

01.04.2024 - 17.04.2024

\*/

#pragma once

namespace DBROM {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::Data::SqlClient;

/// <summary>

/// Сводка для DeleteRecord

/// </summary>

public ref class DeleteRecord : public System::Windows::Forms::Form

{

public:

DeleteRecord(String^ ds, String^ ic, String^ tn)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

this->ds = ds;

this->ic = ic;

this->tn = tn;

get\_id();

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~DeleteRecord()

{

if (components)

{

delete components;

}

}

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

private: SqlConnection^ connection; // Строка подключения

private: String^ ds = ""; // Название сервера

private: String^ ic = ""; // Название базы данных

private: String^ tn = ""; // Имя таблицы

private: System::Windows::Forms::ComboBox^ id\_comboBox;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Button^ exit\_button;

private: System::Windows::Forms::Button^ delete\_button;

System::ComponentModel::Container^ components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

this->id\_comboBox = (gcnew System::Windows::Forms::ComboBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->exit\_button = (gcnew System::Windows::Forms::Button());

this->delete\_button = (gcnew System::Windows::Forms::Button());

this->SuspendLayout();

//

// id\_comboBox

//

this->id\_comboBox->FormattingEnabled = true;

this->id\_comboBox->Location = System::Drawing::Point(313, 13);

this->id\_comboBox->Name = L"id\_comboBox";

this->id\_comboBox->Size = System::Drawing::Size(138, 28);

this->id\_comboBox->TabIndex = 0;

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(12, 16);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(295, 20);

this->label1->TabIndex = 1;

this->label1->Text = L"Выбирите ID записи для её удаления";

//

// exit\_button

//

this->exit\_button->Location = System::Drawing::Point(149, 65);

this->exit\_button->Name = L"exit\_button";

this->exit\_button->Size = System::Drawing::Size(145, 40);

this->exit\_button->TabIndex = 12;

this->exit\_button->Text = L"Отмена";

this->exit\_button->UseVisualStyleBackColor = true;

this->exit\_button->Click += gcnew System::EventHandler(this, &DeleteRecord::exit\_button\_Click);

//

// delete\_button

//

this->delete\_button->Location = System::Drawing::Point(306, 65);

this->delete\_button->Name = L"delete\_button";

this->delete\_button->Size = System::Drawing::Size(145, 40);

this->delete\_button->TabIndex = 11;

this->delete\_button->Text = L"Удалить";

this->delete\_button->UseVisualStyleBackColor = true;

this->delete\_button->Click += gcnew System::EventHandler(this, &DeleteRecord::delete\_button\_Click);

//

// DeleteRecord

//

this->AutoScaleDimensions = System::Drawing::SizeF(9, 20);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(463, 117);

this->Controls->Add(this->exit\_button);

this->Controls->Add(this->delete\_button);

this->Controls->Add(this->label1);

this->Controls->Add(this->id\_comboBox);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedToolWindow;

this->Name = L"DeleteRecord";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"Удаление записи";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

//

// Процедура подключения к БД

//

private: void connect\_to\_DB()

{

SqlConnectionStringBuilder^ strbild = gcnew SqlConnectionStringBuilder();

strbild->DataSource = ds;

strbild->InitialCatalog = ic;

strbild->IntegratedSecurity = true;

connection = gcnew SqlConnection(Convert::ToString(strbild));

}

//

// Процедура получения ID записей

//

private: void get\_id()

{

connect\_to\_DB();

String^ query = "SELECT ID FROM " + tn;

connection->Open();

SqlCommand^ cmd = gcnew SqlCommand(query, connection);

SqlDataReader^ reader = cmd->ExecuteReader();

String^ id\_temp = "";

while (reader->Read())

{

id\_temp = reader["ID"]->ToString();

id\_comboBox->Items->Add(String::Format("{0}", id\_temp));

}

connection->Close();

}

//

// Процедура удаления выбранной записи

//

private: void delete\_rec()

{

String^ col\_name = "";

String^ id = Convert::ToString(id\_comboBox->Items[id\_comboBox->SelectedIndex]);

String^ command\_text = "DELETE FROM " + tn + " WHERE ID = " + id;

connect\_to\_DB();

SqlCommand^ cmd = gcnew SqlCommand(command\_text, connection);

connection->Open();

cmd->ExecuteNonQuery();

connection->Close();

}

//

// Процедура запускающая действия при нажатии кнопки "Отмена"

//

private: System::Void exit\_button\_Click(System::Object^ sender, System::EventArgs^ e)

{

this->Close();

}

//

// Процедура запускающая действия при нажатии кнопки "Удалить"

//

private: System::Void delete\_button\_Click(System::Object^ sender, System::EventArgs^ e)

{

System::Windows::Forms::DialogResult result = MessageBox::Show("Вы уверены, что хотите удалить эту та?", "Уведомление", MessageBoxButtons::YesNo, MessageBoxIcon::Question);

if (result == System::Windows::Forms::DialogResult::Yes)

{

delete\_rec();

}

this->Close();

}

};

}

Приложение К

Листинг файла EditRecord.h

/\*

АЯиП Лабораторная работа #5

Заголовочный файл проекта DBROM

Михайлов Роман, группа 3091, 1 курс

01.04.2024 - 17.04.2024

\*/

#pragma once

namespace DBROM {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::Data::SqlClient;

/// <summary>

/// Сводка для EditRecord

/// </summary>

public ref class EditRecord : public System::Windows::Forms::Form

{

public:

EditRecord(String^ ds, String^ ic, String^ tn)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

this->ds = ds;

this->ic = ic;

this->tn = tn;

get\_id();

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~EditRecord()

{

if (components)

{

delete components;

}

}

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

private: SqlConnection^ connection; // Строка подключения

private: String^ ds = ""; // Название сервера

private: String^ ic = ""; // Название базы данных

private: String^ tn = ""; // Имя таблицы

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::ComboBox^ id\_comboBox;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::ComboBox^ col\_comboBox;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::TextBox^ current\_data;

private: System::Windows::Forms::TextBox^ new\_data;

private: System::Windows::Forms::Button^ rename\_button;

private: System::Windows::Forms::Button^ exit\_button;

System::ComponentModel::Container^ components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

this->label1 = (gcnew System::Windows::Forms::Label());

this->id\_comboBox = (gcnew System::Windows::Forms::ComboBox());

this->label2 = (gcnew System::Windows::Forms::Label());

this->col\_comboBox = (gcnew System::Windows::Forms::ComboBox());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->current\_data = (gcnew System::Windows::Forms::TextBox());

this->new\_data = (gcnew System::Windows::Forms::TextBox());

this->rename\_button = (gcnew System::Windows::Forms::Button());

this->exit\_button = (gcnew System::Windows::Forms::Button());

this->SuspendLayout();

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(12, 18);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(392, 20);

this->label1->TabIndex = 0;

this->label1->Text = L"Выбирите ID записи, которую Вы хотите изменить";

//

// id\_comboBox

//

this->id\_comboBox->FormattingEnabled = true;

this->id\_comboBox->Location = System::Drawing::Point(410, 15);

this->id\_comboBox->Name = L"id\_comboBox";

this->id\_comboBox->Size = System::Drawing::Size(116, 28);

this->id\_comboBox->TabIndex = 1;

this->id\_comboBox->SelectedIndexChanged += gcnew System::EventHandler(this, &EditRecord::id\_comboBox\_SelectedIndexChanged);

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(12, 61);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(269, 20);

this->label2->TabIndex = 2;

this->label2->Text = L"Выбирите колонку для изменения";

//

// col\_comboBox

//

this->col\_comboBox->FormattingEnabled = true;

this->col\_comboBox->Items->AddRange(gcnew cli::array< System::Object^ >(5) {

L"Название", L"Описание", L"Статус", L"Ответсвенный",

L"Срок сдачи"

});

this->col\_comboBox->Location = System::Drawing::Point(287, 58);

this->col\_comboBox->Name = L"col\_comboBox";

this->col\_comboBox->Size = System::Drawing::Size(239, 28);

this->col\_comboBox->TabIndex = 3;

this->col\_comboBox->SelectedIndexChanged += gcnew System::EventHandler(this, &EditRecord::col\_comboBox\_SelectedIndexChanged);

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(12, 104);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(148, 20);

this->label3->TabIndex = 4;

this->label3->Text = L"Текущее значение";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Location = System::Drawing::Point(12, 147);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(200, 20);

this->label4->TabIndex = 5;

this->label4->Text = L"Введите новое значение";

//

// current\_data

//

this->current\_data->Location = System::Drawing::Point(166, 104);

this->current\_data->Name = L"current\_data";

this->current\_data->ReadOnly = true;

this->current\_data->ScrollBars = System::Windows::Forms::ScrollBars::Vertical;

this->current\_data->Size = System::Drawing::Size(361, 26);

this->current\_data->TabIndex = 6;

//

// new\_data

//

this->new\_data->Location = System::Drawing::Point(218, 147);

this->new\_data->Name = L"new\_data";

this->new\_data->Size = System::Drawing::Size(309, 26);

this->new\_data->TabIndex = 7;

this->new\_data->TextChanged += gcnew System::EventHandler(this, &EditRecord::new\_data\_TextChanged);

//

// rename\_button

//

this->rename\_button->Location = System::Drawing::Point(382, 203);

this->rename\_button->Name = L"rename\_button";

this->rename\_button->Size = System::Drawing::Size(145, 40);

this->rename\_button->TabIndex = 9;

this->rename\_button->Text = L"Изменить";

this->rename\_button->UseVisualStyleBackColor = true;

this->rename\_button->Click += gcnew System::EventHandler(this, &EditRecord::rename\_button\_Click);

//

// exit\_button

//

this->exit\_button->Location = System::Drawing::Point(225, 203);

this->exit\_button->Name = L"exit\_button";

this->exit\_button->Size = System::Drawing::Size(145, 40);

this->exit\_button->TabIndex = 10;

this->exit\_button->Text = L"Отмена";

this->exit\_button->UseVisualStyleBackColor = true;

this->exit\_button->Click += gcnew System::EventHandler(this, &EditRecord::exit\_button\_Click);

//

// EditRecord

//

this->AutoScaleDimensions = System::Drawing::SizeF(9, 20);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(539, 262);

this->Controls->Add(this->exit\_button);

this->Controls->Add(this->rename\_button);

this->Controls->Add(this->new\_data);

this->Controls->Add(this->current\_data);

this->Controls->Add(this->label4);

this->Controls->Add(this->label3);

this->Controls->Add(this->col\_comboBox);

this->Controls->Add(this->label2);

this->Controls->Add(this->id\_comboBox);

this->Controls->Add(this->label1);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedToolWindow;

this->Name = L"EditRecord";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"Редактировать запись";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

//

// Процедура проверки корректности данных в поле "Срок сдачи"

//

private: void check\_date()

{

if (this->new\_data->Text->Length > 10) this->new\_data->Text = this->new\_data->Text->Remove(10);

int j = 0;

for (int i = 0; i < this->new\_data->Text->Length; i++) if (this->new\_data->Text[i] == '.') j++;

if(j > 2) this->new\_data->Text = this->new\_data->Text->Remove(this->new\_data->Text->LastIndexOf("."));

if (this->new\_data->Text->Length > 9)

{

DateTime date;

if (DateTime::TryParse(this->new\_data->Text, date)) {}

else MessageBox::Show("Заполните поле срока сдачи корректно", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

}

}

//

// Процедура подключения к БД

//

private: void connect\_to\_DB()

{

SqlConnectionStringBuilder^ strbild = gcnew SqlConnectionStringBuilder();

strbild->DataSource = ds;

strbild->InitialCatalog = ic;

strbild->IntegratedSecurity = true;

connection = gcnew SqlConnection(Convert::ToString(strbild));

}

//

// Процедура изменения выбранной записи

//

private: void edit()

{

String^ col\_name = "";

switch (col\_comboBox->SelectedIndex)

{

case 0: col\_name = "label"; break;

case 1: col\_name = "description"; break;

case 2: col\_name = "status"; break;

case 3: col\_name = "admin"; break;

case 4: col\_name = "deadline"; break;

}

String^ id = Convert::ToString(id\_comboBox->Items[id\_comboBox->SelectedIndex]);

String^ comamnd\_text = "UPDATE " + tn + " SET " + col\_name + " = '" + new\_data->Text + "' WHERE ID = " + id;

connect\_to\_DB();

SqlCommand^ cmd = gcnew SqlCommand(comamnd\_text, connection);

connection->Open();

cmd->ExecuteNonQuery();

connection->Close();

}

//

// Процедура получения ID записей

//

private: void get\_id()

{

connect\_to\_DB();

String^ query = "SELECT ID FROM " + tn;

connection->Open();

SqlCommand^ cmd = gcnew SqlCommand(query, connection);

SqlDataReader^ reader = cmd->ExecuteReader();

String^ id\_temp = "";

while (reader->Read())

{

id\_temp = reader["ID"]->ToString();

id\_comboBox->Items->Add(String::Format("{0}", id\_temp));

}

connection->Close();

}

//

// Процедура получения данных выбранной записи

//

private: void get\_curr\_data()

{

String^ col\_name = "";

switch (col\_comboBox->SelectedIndex)

{

case 0: col\_name = "label"; break;

case 1: col\_name = "description"; break;

case 2: col\_name = "status"; break;

case 3: col\_name = "admin"; break;

case 4: col\_name = "deadline"; break;

}

String^ id = Convert::ToString(id\_comboBox->Items[id\_comboBox->SelectedIndex]);

String^ query = "SELECT " + col\_name + " FROM " + tn + " WHERE ID = " + id;

connect\_to\_DB();

connection->Open();

SqlCommand^ cmd = gcnew SqlCommand(query, connection);

SqlDataReader^ reader = cmd->ExecuteReader();

if(col\_comboBox->SelectedIndex != 4) while(reader->Read()) current\_data->Text = reader[col\_name]->ToString();

else while (reader->Read()) current\_data->Text = (reader[col\_name]->ToString())->Remove(10);

connection->Close();

}

//

// Процедур запускающая действия при нажатии кнопки "Изменить"

//

private: System::Void rename\_button\_Click(System::Object^ sender, System::EventArgs^ e)

{

edit();

this->Close();

}

//

// Процедура запускающая действия при нажатии кнопки "Отмена"

//

private: System::Void exit\_button\_Click(System::Object^ sender, System::EventArgs^ e)

{

this->Close();

}

//

// Процедура запускающая действия при выборе изменяемого столбца

//

private: System::Void col\_comboBox\_SelectedIndexChanged(System::Object^ sender, System::EventArgs^ e)

{

if(id\_comboBox->SelectedIndex != -1 && col\_comboBox->SelectedIndex != -1) get\_curr\_data();

}

//

// Процедура запускающая действия при выборе изменяемой записи

//

private: System::Void id\_comboBox\_SelectedIndexChanged(System::Object^ sender, System::EventArgs^ e)

{

if (id\_comboBox->SelectedIndex != -1 && col\_comboBox->SelectedIndex != -1) get\_curr\_data();

}

//

// Процедура запускающая действия при изменении данных в поле "Введите новое значение"

//

private: System::Void new\_data\_TextChanged(System::Object^ sender, System::EventArgs^ e)

{

if (col\_comboBox->SelectedIndex == 4) check\_date();

else

{

if (this->new\_data->Text->Length > 200)

{

MessageBox::Show("Слишком длинный текст", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

this->new\_data->Text = this->new\_data->Text->Remove(200);

}

}

}

};

}

Приложение Л

Листинг файла NewRecord.h

/\*

АЯиП Лабораторная работа #5

Заголовочный файл проекта DBROM

Михайлов Роман, группа 3091, 1 курс

01.04.2024 - 17.04.2024

\*/

#pragma once

namespace DBROM {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Сводка для NewRecord

/// </summary>

public ref class NewRecord : public System::Windows::Forms::Form

{

public:

NewRecord(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~NewRecord()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::TextBox^ textBox2;

private: System::Windows::Forms::TextBox^ textBox3;

private: System::Windows::Forms::TextBox^ textBox4;

private: System::Windows::Forms::TextBox^ textBox5;

private: System::Windows::Forms::Button^ button2;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::Label^ label5;

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->button1 = (gcnew System::Windows::Forms::Button());

this->textBox2 = (gcnew System::Windows::Forms::TextBox());

this->textBox3 = (gcnew System::Windows::Forms::TextBox());

this->textBox4 = (gcnew System::Windows::Forms::TextBox());

this->textBox5 = (gcnew System::Windows::Forms::TextBox());

this->button2 = (gcnew System::Windows::Forms::Button());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->label5 = (gcnew System::Windows::Forms::Label());

this->SuspendLayout();

//

// textBox1

//

this->textBox1->Location = System::Drawing::Point(153, 25);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(301, 26);

this->textBox1->TabIndex = 0;

this->textBox1->TextChanged += gcnew System::EventHandler(this, &NewRecord::textBox1\_TextChanged);

//

// button1

//

this->button1->Location = System::Drawing::Point(309, 240);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(145, 40);

this->button1->TabIndex = 1;

this->button1->Text = L"Создать";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &NewRecord::button1\_Click);

//

// textBox2

//

this->textBox2->Location = System::Drawing::Point(153, 65);

this->textBox2->Name = L"textBox2";

this->textBox2->Size = System::Drawing::Size(301, 26);

this->textBox2->TabIndex = 2;

//

// textBox3

//

this->textBox3->Location = System::Drawing::Point(153, 108);

this->textBox3->Name = L"textBox3";

this->textBox3->Size = System::Drawing::Size(301, 26);

this->textBox3->TabIndex = 3;

//

// textBox4

//

this->textBox4->Location = System::Drawing::Point(153, 148);

this->textBox4->Name = L"textBox4";

this->textBox4->Size = System::Drawing::Size(301, 26);

this->textBox4->TabIndex = 4;

//

// textBox5

//

this->textBox5->Location = System::Drawing::Point(153, 188);

this->textBox5->Name = L"textBox5";

this->textBox5->Size = System::Drawing::Size(301, 26);

this->textBox5->TabIndex = 5;

this->textBox5->TextChanged += gcnew System::EventHandler(this, &NewRecord::textBox5\_TextChanged);

//

// button2

//

this->button2->Location = System::Drawing::Point(152, 240);

this->button2->Name = L"button2";

this->button2->Size = System::Drawing::Size(151, 40);

this->button2->TabIndex = 6;

this->button2->Text = L"Отмена";

this->button2->UseVisualStyleBackColor = true;

this->button2->Click += gcnew System::EventHandler(this, &NewRecord::button2\_Click);

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(18, 28);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(83, 20);

this->label1->TabIndex = 7;

this->label1->Text = L"Название";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(18, 68);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(83, 20);

this->label2->TabIndex = 8;

this->label2->Text = L"Описание";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(18, 108);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(62, 20);

this->label3->TabIndex = 9;

this->label3->Text = L"Статус";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Location = System::Drawing::Point(18, 148);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(130, 20);

this->label4->TabIndex = 10;

this->label4->Text = L"Ответственный";

//

// label5

//

this->label5->AutoSize = true;

this->label5->Location = System::Drawing::Point(18, 188);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(96, 20);

this->label5->TabIndex = 11;

this->label5->Text = L"Срок сдачи";

//

// NewRecord

//

this->AutoScaleDimensions = System::Drawing::SizeF(9, 20);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->BackColor = System::Drawing::SystemColors::Control;

this->ClientSize = System::Drawing::Size(467, 291);

this->Controls->Add(this->label5);

this->Controls->Add(this->label4);

this->Controls->Add(this->label3);

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->button2);

this->Controls->Add(this->textBox5);

this->Controls->Add(this->textBox4);

this->Controls->Add(this->textBox3);

this->Controls->Add(this->textBox2);

this->Controls->Add(this->button1);

this->Controls->Add(this->textBox1);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedToolWindow;

this->Name = L"NewRecord";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"Добавить новую запись";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

//

// Процедура проверки корректности введённой даты

//

private: void check\_date()

{

if (this->textBox5->Text->Length > 10) this->textBox5->Text = this->textBox5->Text->Remove(10);

int j = 0;

for (int i = 0; i < this->textBox5->Text->Length; i++) if (this->textBox5->Text[i] == '.') j++;

if(j > 2) this->textBox5->Text = this->textBox5->Text->Remove(this->textBox5->Text->LastIndexOf("."));

if (this->textBox5->Text->Length > 9)

{

DateTime date;

if (DateTime::TryParse(this->textBox5->Text, date))

{

if (Convert::ToInt32(this->textBox5->Text->Substring(6)) > 2023 && Convert::ToInt32(this->textBox5->Text->Substring(6)) < 10000) {}

else MessageBox::Show("Заполните поле срока сдачи корректно. Ограничение ввода года - от 2024 до 9999 ", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

}

else MessageBox::Show("Заполните поле срока сдачи корректно", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

}

}

//

// Процедура запускающая действия при нажатии "Добавить"

//

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e)

{

if (get\_label() != "" || get\_description() != "" || get\_status() != "" || get\_admin() != "" || get\_deadline() != "")

{

if (this->textBox5->Text->IndexOf(".") == 2 && this->textBox5->Text->LastIndexOf(".") == 5)

{

this->Close();

}

else

{

MessageBox::Show("Заполните поле срока сдачи корректно", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

}

}

else MessageBox::Show("Заполните все поля", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

}

//

// Функция получения данных из поля "Название"

//

public: String^ get\_label()

{

return this->textBox1->Text;

}

//

// Функция получения данных из поля "Описание"

//

public: String^ get\_description()

{

return this->textBox2->Text;

}

//

// Функция получения данных из поля "Статус"

//

public: String^ get\_status()

{

return this->textBox3->Text;

}

//

// Функция получения данных из поля "Ответственный"

//

public: String^ get\_admin()

{

return this->textBox4->Text;

}

//

// Функция получения данных из поля "Срок сдачи"

//

public: String^ get\_deadline()

{

return this->textBox5->Text;

}

//

// Процедура запускающая действия при нажатии "Отмена"

//

private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e)

{

this->Close();

}

//

// Процедура проверки поля "Название"

//

private: void check\_lenght1()

{

if (this->textBox1->Text->Length > 200)

{

MessageBox::Show("Слишком длинный текст", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

this->textBox1->Text = this->textBox1->Text->Remove(200);

}

}

//

// Процедура проверки поля "Описание"

//

private: void check\_lenght2()

{

if (this->textBox2->Text->Length > 200)

{

MessageBox::Show("Слишком длинный текст", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

this->textBox2->Text = this->textBox2->Text->Remove(200);

}

}

//

// Процедура проверки поля "Статус"

//

private: void check\_lenght3()

{

if (this->textBox3->Text->Length > 200)

{

MessageBox::Show("Слишком длинный текст", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

this->textBox3->Text = this->textBox3->Text->Remove(200);

}

}

//

// Процедура проверки поля "Ответственный"

//

private: void check\_lenght4()

{

if (this->textBox4->Text->Length > 200)

{

MessageBox::Show("Слишком длинный текст", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

this->textBox4->Text = this->textBox4->Text->Remove(200);

}

}

//

// Процедура проверки поля "Срок сдачи"

//

private: System::Void textBox5\_TextChanged(System::Object^ sender, System::EventArgs^ e)

{

check\_date();

}

//

// Процедура запускающая действия при изменении данных в поле "Название"

//

private: System::Void textBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e)

{

check\_lenght1();

}

//

// Процедура запускающая действия при изменении данных в поле "Описание"

//

private: System::Void textBox2\_TextChanged(System::Object^ sender, System::EventArgs^ e)

{

check\_lenght2();

}

//

// Процедура запускающая действия при изменении данных в поле "Статус"

//

private: System::Void textBox3\_TextChanged(System::Object^ sender, System::EventArgs^ e)

{

check\_lenght3();

}

//

// Процедура запускающая действия при изменении данных в поле "Ответственный"

//

private: System::Void textBox4\_TextChanged(System::Object^ sender, System::EventArgs^ e)

{

check\_lenght4();

}

};

}

Приложение М

Листинг файла MainWindow.cpp

/\*

АЯиП Лабораторная работа #5

Файл исходного кода проекта DBROM

Михайлов Роман, группа 3091, 1 курс

08.04.2024 - 17.04.2024

\*/

#include "MainWindow.h"

#include "Windows.h"

using namespace System;

using namespace System::Windows::Forms;

[STAThreadAttribute]

int main()

{

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

DBROM::MainWindow form;

Application::Run(% form);

}

Приложение Н

Листинг файла UnitTest.cpp

Приложение О

Листинг файла Functions.h