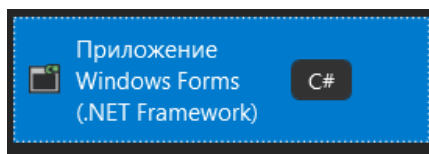
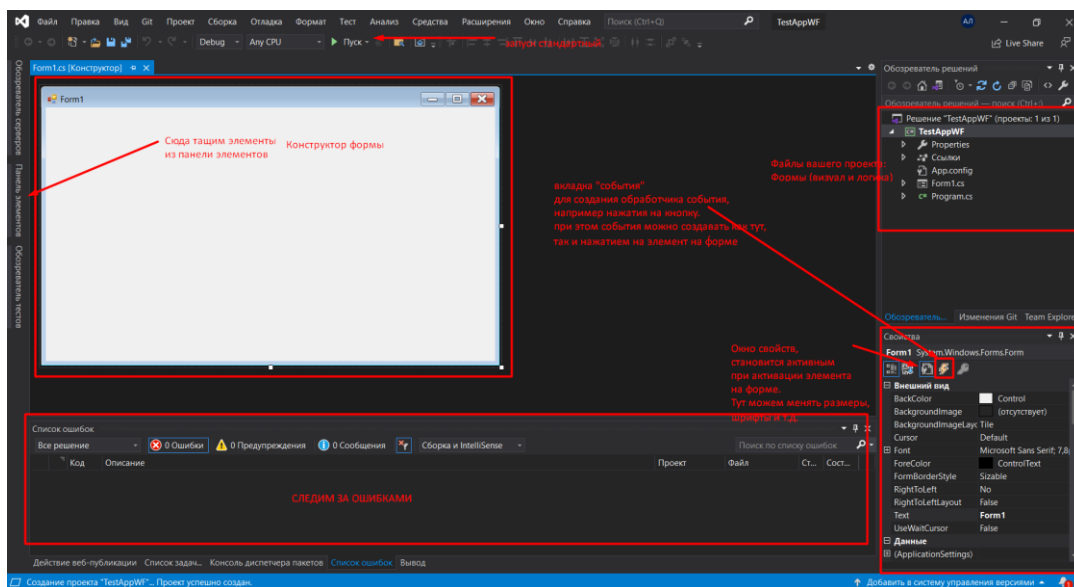


МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО РАБОТЕ С WINDOWS FORMS И БД

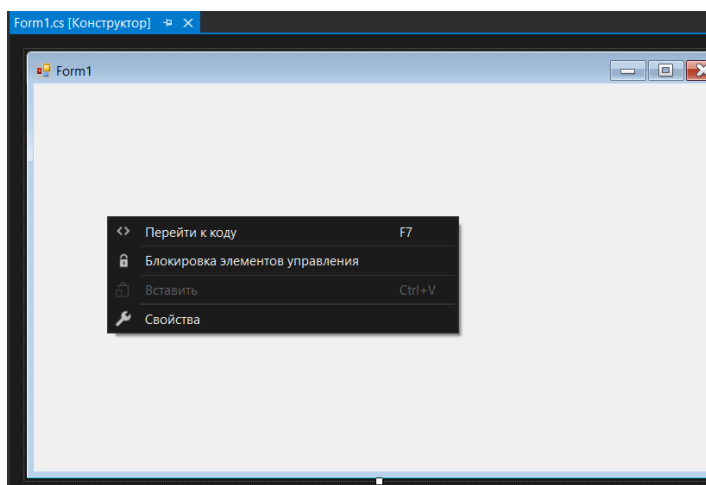
1. Создаем приложение в MS Visual Studio типа **Windows Forms (.NET Framework)**, обязательно даем **нормальное название!**



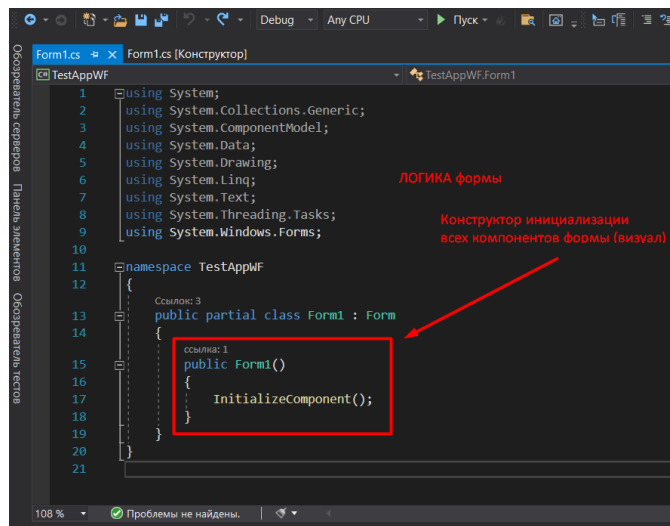
2. Ознакомьтесь с рабочими областями. Форма состоит из конструктора (то, что видит пользователь), и логики вашей формы



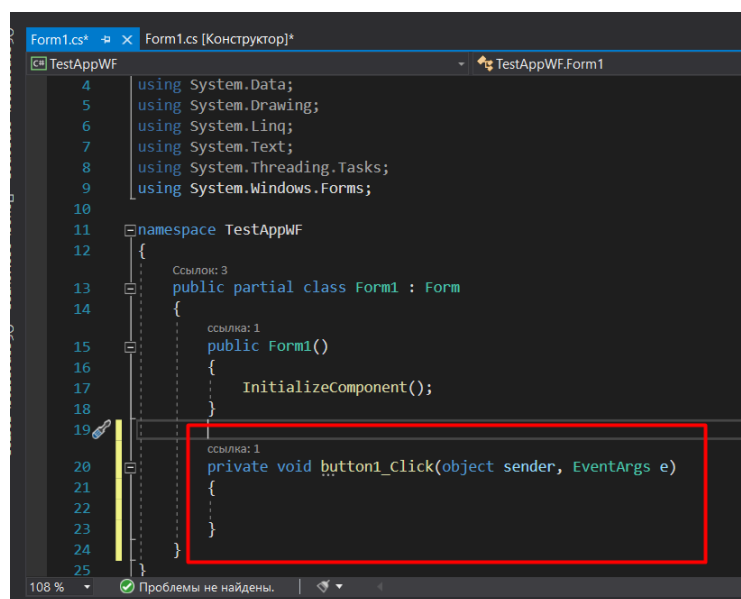
3. ПКМ по форме и выбрать «Перейти к коду», для перехода к логике формы



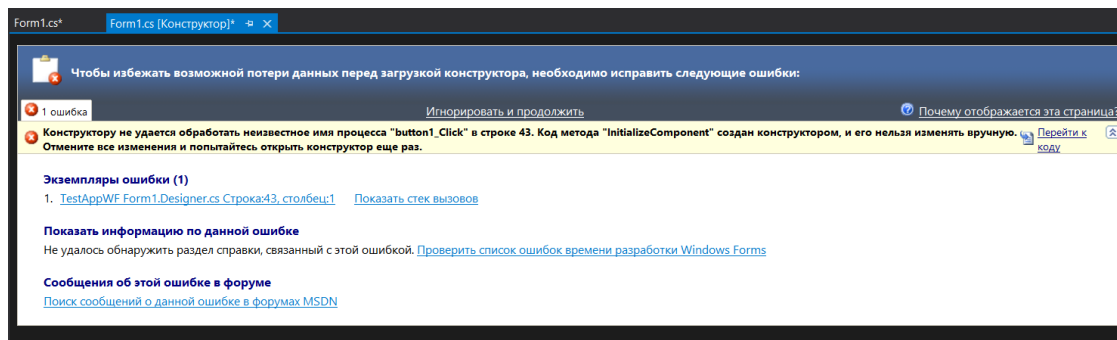
4. Логика формы прописывается в файле .cs



5. Не нужно ничего удалять с этой страницы, изначальный код генерируется автоматически
6. При создании обработчика нажатия на кнопку создается (автоматически!) событие

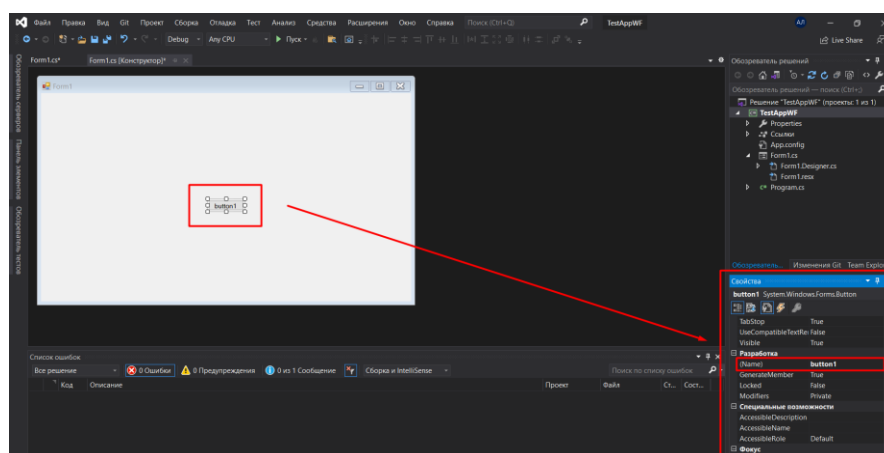


7. Весь код логики при нажатии на эту кнопку прописывается внутри именно этого блока
8. Нельзя удалять обработчики событий из логики, не удалив их из событий в конструкторе, это вызовет ошибки!



9. Именуем формы и элементы не «button1», «Form1», а нормально и очевидно по логике, для чего нужен этот элемент? Например, кнопка авторизации будет иметь название в коде: LoginButton. При этом каждое новое слово с заглавной буквы! Без пробелов и т.д.

10. Переименовать элемент для кода можно в конструкторе



11. Навигация между страницами:

<https://metanit.com/sharp/windowsforms/2.3.php>

```
private void button1_Click(object sender, EventArgs e)
{
    Form2 newForm = new Form2(this);
    newForm.Show();
}
```

Где Form2 – название вашей новой формы, той, к которой нужно перейти при нажатии на кнопку.

Сайт, которым вы будете пользоваться при возникающих вопросах:

<https://metanit.com/sharp/windowsforms/>

Форумы: github, cyberforum, stackoverflow, хабр и другие.

Учитесь искать информацию и формулировать свои вопросы, я буду рядом не всегда.

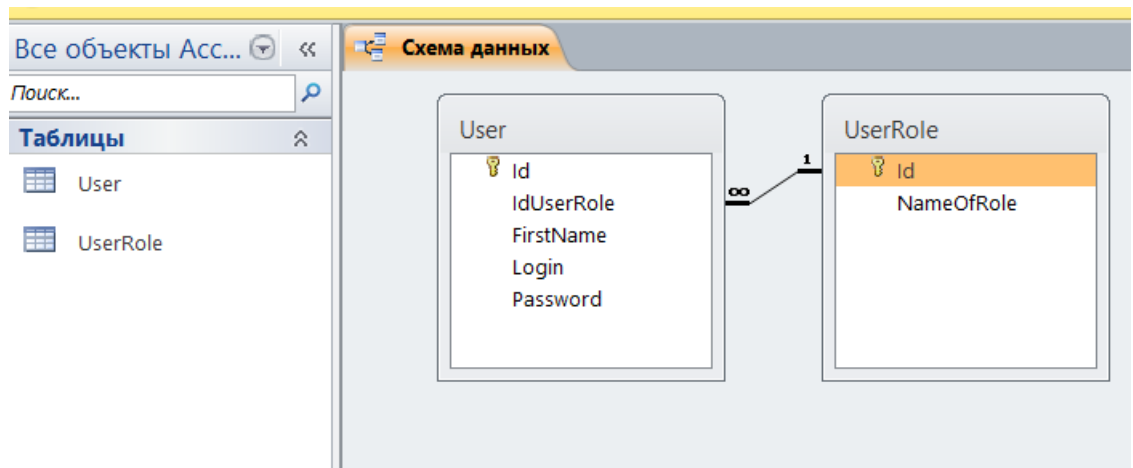
Лисавина Алёна Вадимовна

ВЗАИМОДЕЙСТВИЕ С БД

Доступная СУБД, которую вы можете использовать – MS Access.

<https://metanit.com/sharp/adonet/2.3.php>

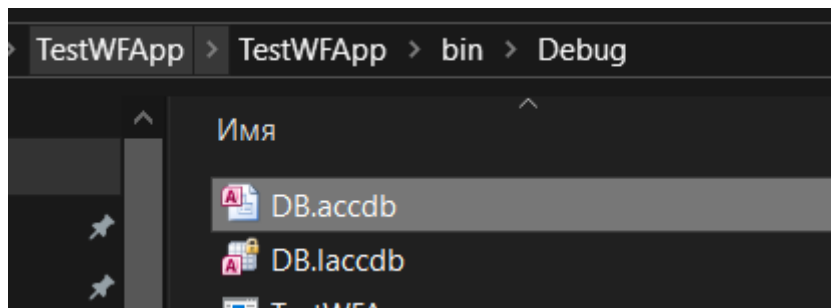
В моей тестовой БД всего 2 таблицы: пользователи и их роли.



1. Создание строки подключения к БД

```
public static string ConnectionString = @"Provider=Microsoft.ACE.OLEDB.12.0;Data Source = DB.accdb;";
```

2. Файл с БД «*.accdb» добавляем в папку проекта по пути:



3. Создание подключения: в библиотеки добавляем

```
using System.Data.OleDb;
```

А в логику добавляем подключение и открытие БД:

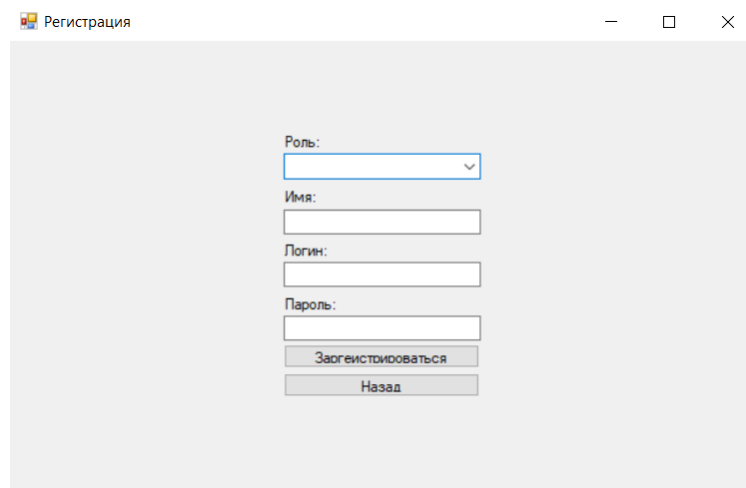
```
using (OleDbConnection connection = new OleDbConnection(connectionString))  
{  
    connection.Open();  
}
```

4. После этого можем осуществлять любые операции с БД:
добавление, удаление, редактирование, просмотр

Чтобы выполнить команду, необходимо применить один из методов OleDbCommand:

- ExecuteNonQuery: просто выполняет sql-выражение и возвращает количество измененных записей. Подходит для sql-выражений INSERT, UPDATE, DELETE.
- ExecuteReader: выполняет sql-выражение и возвращает строки из таблицы. Подходит для sql-выражения SELECT.
- ExecuteScalar: выполняет sql-выражение и возвращает одно скалярное значение, например, число. Подходит для sql-выражения SELECT в паре с одной из встроенных функций SQL, как например, Min, Max, Sum, Count.

Добавление данных (INSERT) на примере регистрации



The screenshot shows a web form titled "Регистрация" (Registration). It contains the following fields and controls:

- A dropdown menu labeled "Роль:" (Role).
- A text input field labeled "Имя:" (Name).
- A text input field labeled "Логин:" (Login).
- A text input field labeled "Пароль:" (Password).
- A button labeled "Зарегистрироваться" (Register).
- A button labeled "Назад" (Back).

```

private void RegButton_Click(object sender, EventArgs e)
{
    try
    {
        using (OleDbConnection connection = new OleDbConnection(Classes.ConnectionDB.ConnectionString))
        {
            connection.Open();

            string sqlExpression = "INSERT INTO [User] ([IdUserRole], [FirstName], [Login], [Password]) " +
                "VALUES (@Role, @Name, @Login, @Password);";
            OleDbCommand sqlCommand = new OleDbCommand(sqlExpression, connection);
            sqlCommand.Parameters.AddWithValue("@Role", RoleComboBox.SelectedIndex + 1);
            sqlCommand.Parameters.AddWithValue("@Name", NameTextBox.Text);
            sqlCommand.Parameters.AddWithValue("@Login", LoginTextBox.Text);
            sqlCommand.Parameters.AddWithValue("@Password", PasswordTextBox.Text);

            int number = sqlCommand.ExecuteNonQuery();
            if (number >= 1)
            {
                MessageBox.Show("Успешная регистрация!");

                this.Hide();
                LoginForm loginForm = new LoginForm();
                loginForm.ShowDialog();
            }
            else
            {
                MessageBox.Show("Ошибка регистрации!");
            }
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}

```

В sqlExpression обычный запрос к БД на вставку данных.

@ – это параметр, чтобы нельзя было навредить БД

Открываем подключение

Создаем команду OleDbCommand, передаем ей параметры – запрос и строку подключения

Добавляем указанные параметры, в данном случае – пример регистрации, поэтому данные я вытаскиваю из comboBox и textBox

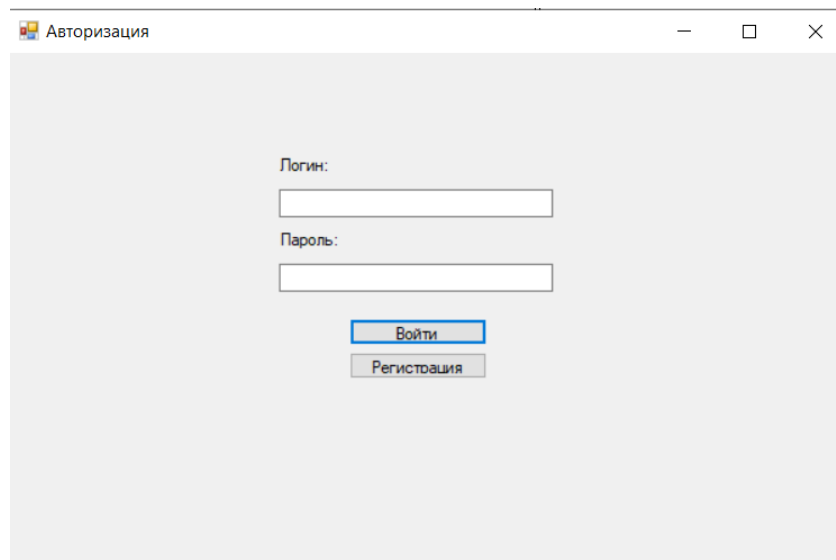
Выполняем команду и выводим результат

Все слова в SQL-запросе, которые являются ключевыми словами, вроде User, Login и т.д., берутся в скобки []

Обновление (редактирование) аналогично добавлению, отличие – ключевое слово UPDATE

Удаление аналогично добавлению, отличие – ключевое слово DELETE

Просмотр данных из таблиц (SELECT) на примере авторизации



```
ссылка:1
private void LoginButton_Click(object sender, EventArgs e)
{
    try
    {
        using (OleDbConnection connection = new OleDbConnection(Classes.ConnectionDB.ConnectionString))
        {
            connection.Open();

            string sqlExpression = "SELECT * FROM [User] WHERE [Login] = @Login AND [Password] = @Password;";
            OleDbCommand sqlCommand = new OleDbCommand(sqlExpression, connection);
            sqlCommand.Parameters.AddWithValue("@Login", LoginTextBox.Text);
            sqlCommand.Parameters.AddWithValue("@Password", PasswordTextBox.Text);

            using (OleDbDataReader dataReader = sqlCommand.ExecuteReader())
            {
                if (dataReader.Read())
                {
                    MessageBox.Show($"Здравствуйте, {dataReader["FirstName"]}");
                    this.Hide();
                    LkForm lkForm = new LkForm();
                    lkForm.ShowDialog();
                }
                else
                {
                    MessageBox.Show("Неверный логин/пароль!");
                }
            }
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}
```

Начало аналогично описанному выше

Добавляем новый using для чтения данных из БД

```
using (OleDbDataReader dataReader = sqlCommand.ExecuteReader())
```

И, если есть что читать – значит, что пользователь с таким логином и паролем есть в БД и мы даем доступ к ЛК

Имя (и другие столбцы) можно вытянуть названием столбца reader["FirstName"]

Обновление данных в таблицах (UPDATE)

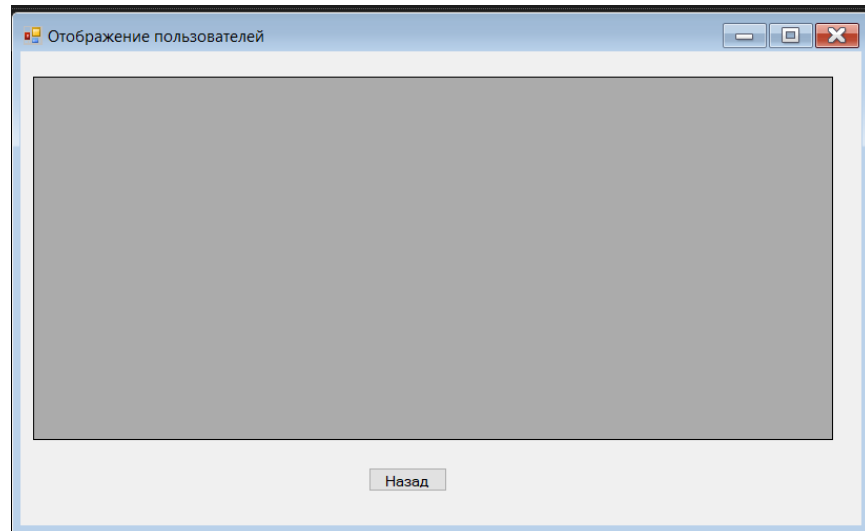
Аналогично INSERT, отличие – ключевое слово UPDATE

Удаление данных из таблиц (DELETE)

Аналогично INSERT, отличие – ключевое слово DELETE

Лисавина Алёна Вадимовна

Работа с выводом данных в DataGridView



```
ссылка: 1
public void FillDG()
{
    try
    {
        string sql = "SELECT [User].[Id], [UserRole].[NameOfRole] as [Роль], [User].[FirstName] as [Имя] " +
            "FROM [User] INNER JOIN [UserRole] ON [UserRole].[Id] = [User].[IdUserRole];";

        using (OleDbConnection connection = new OleDbConnection(Classes.ConnectionDB.ConnectionString))
        {
            connection.Open();

            OleDbDataAdapter sqlDataAdapter = new OleDbDataAdapter(sql, Classes.ConnectionDB.ConnectionString);
            DataSet ds = new DataSet();
            sqlDataAdapter.Fill(ds);
            UsersDataGridView.DataSource = ds.Tables[0];
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}
```

Обычный запрос к БД для получения данных из таблицы User и UserRole, столбцам задала псевдонимы для понятного вывода

Создание OleDbDataAdapter и DataSet— хранилища данных

Заполнение OleDbDataAdapter данными из DataSet

Присвоение источника данных DataGridView

Результат выглядит так:

