Практическая работа 3
Разработка модулей на основе спецификации

Цель занятия: Получить практический опыт разработки модулей на основе спецификации

Перечень оборудования и программного обеспечения

Персональный компьютер
Microsoft Office (Word, Visio)
Microsoft Visual Studio 2015

Краткие теоретические сведения

ADO.NET — это набор классов, предоставляющих службы доступа к данным программисту, работающему на платформе .NET Framework. ADO.NET имеет богатый набор компонентов для создания распределенных приложений, совместно использующих данные. Это неотъемлемая часть платформы .NET Framework, которая предоставляет доступ к реляционным данным, XML-данным и данным приложений.

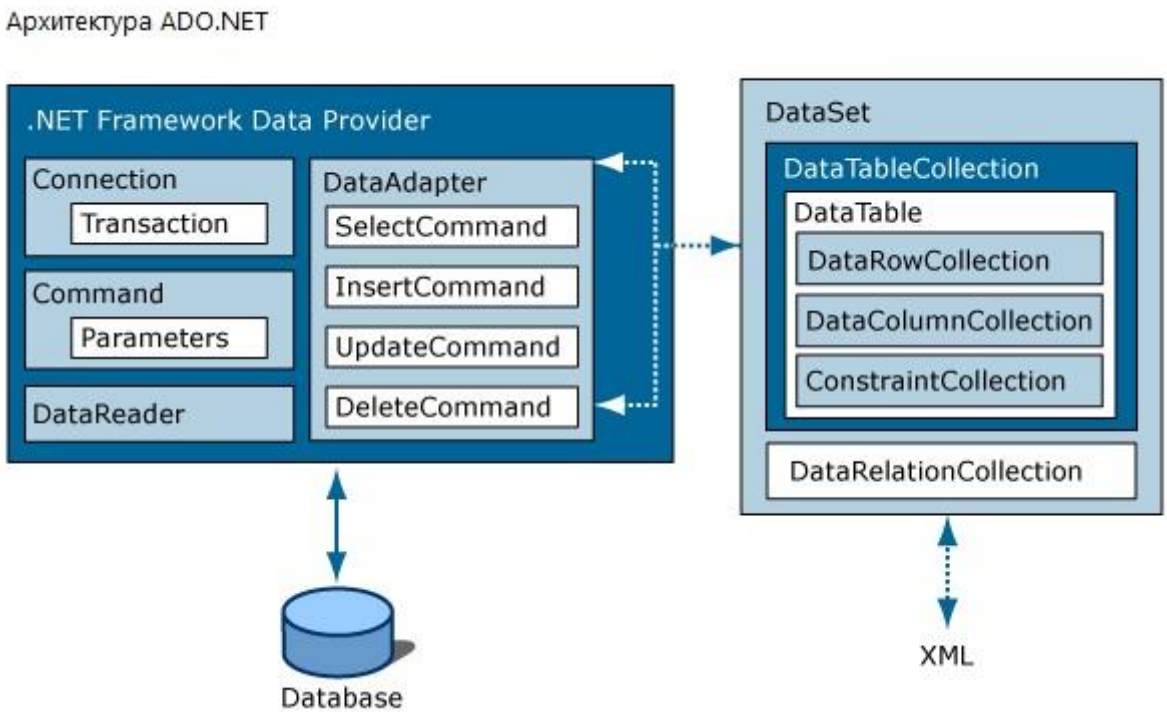
ADO.NET разделят доступ к данным и обработку данных на дискретные компоненты, которые могут использоваться отдельно или совместно. ADO.NET включает поставщиков данных .NET Framework для соединения с базой данных, выполнения команд и получения результатов. Эти результаты, помещенные в объект ADO.NET DataSet, обрабатываются непосредственно, чтобы они могли быть предоставлены пользователю нерегламентированным образом, объединенные с данными из многих источников или передаваемые между уровнями. Объект DataSet также может независимо использоваться поставщиком данных .NET Framework для управления локальными для приложения данными.

Для источников данных OLE DB используется пространство имен System.Data.OleDb.

В следующей таблице приведены четыре основных объекта, из которых состоит поставщик данных .NET Framework.

Object	Описание
Connection	Устанавливает соединение с конкретным источником данных. Базовым классом для всех объектов Connection является DbConnection.
Command	Выполняет команду в источнике данных. Обеспечивает доступность Parameters и может выполнять команды в области Transaction из Connection. Базовым классом для всех объектов Command является DbCommand.
DataReader	Считывает из источника данных однопроходный поток данных только для чтения. Базовым классом для всех

	объектов DataReader является DbDataReader.
DataAdapter	Заполняет DataSet и выполняет обновления в источнике данных. Базовым классом для всех объектов DataAdapter является DbDataAdapter.



DataSet - это центральный компонент, который используется для доступа к данным независимо от источника. Это объект содержит от одного и более объектов DataTable, состоящих из колонок и столбцов, а так же первичного ключа, внешнего ключа, ограничений информации, отношений и так далее.

Пример реализации:

Для соединения с базой данных MS Access 2007 (файл .accdb) в C# следует использовать класс OleDbConnection со следующими параметрами соединения:

provider= Microsoft.Ace.OLEDB.12.0;data source=databaseFile

Здесь databaseFile — абсолютный путь к файлу базы данных Access. Провайдер соединения должен иметь значение Microsoft.Ace.OLEDB.12.0.

Ход выполнения:

1. Для начала следует создать базу данных и заполнить базу содержимым.
2. Создайте проект "Приложение Windows. В конструкторе Windows Forms выберите форму.
3. Перетащите 3 элемента управления textbox, элемент управления button из панели элементов в свою форму.
4. Для элемента управления button создайте событие button1_Click.
5. Вставьте следующий код в обработчик событий.

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

```
        string sql = String.Concat("SELECT * FROM student WHERE Nzach=",
textBox1.Text);
        /* в строке указывается формат SQL-запроса*/
      string connectionString;
      connectionString = "Provider=Microsoft.Ace.OLEDB.12.0;" +
                 @"Data Source= C:\\Users\\Mary\\Desktop\\Database.accdb";
       /* в строке указывается файл для подключения*/

      OleDbConnection connection = new OleDbConnection(connectionString);
      connection.Open();

      OleDbCommand command = new OleDbCommand(sql, connection);
      OleDbDataReader dataReader = command.ExecuteReader();
      while (dataReader.Read())
      {
          textBox2.Text = textBox2.Text + dataReader["Fam"];
          textBox3.Text = textBox3.Text + dataReader["Im"];
          textBox4.Text = textBox4.Text + dataReader["Otch"];
      }
      // В цикле содержимое полей записи записывается в  textBox.Text
      dataReader.Close();
      connection.Close();
```

Задания

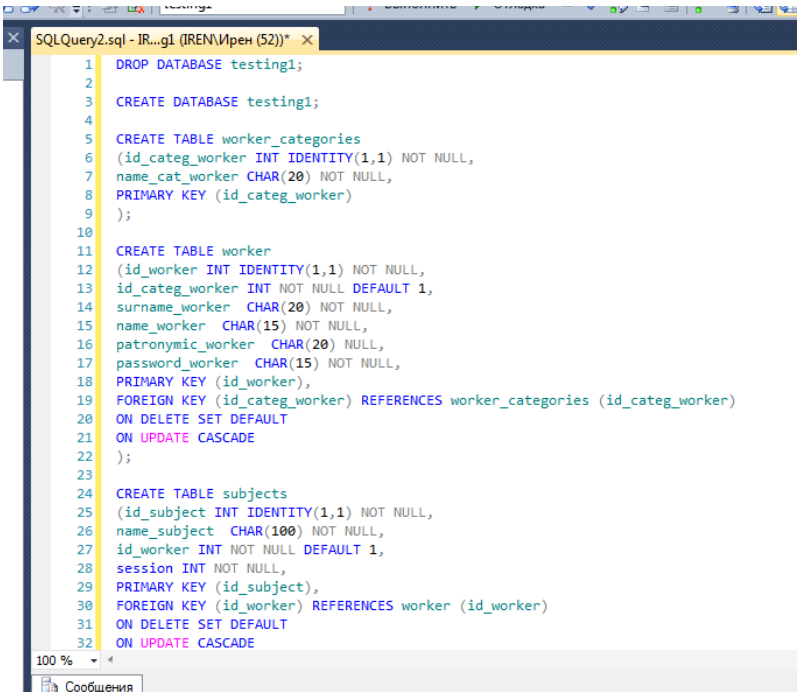
1 Изучить теоретические сведения и задание к работе

2 В соответствии с вариантом задания разработать интерфейс, организовать хранение данных в базе данных и создать модуль взаимодействия.

Порядок выполнения работы

Рассмотрим на примере приложения Тестирование.

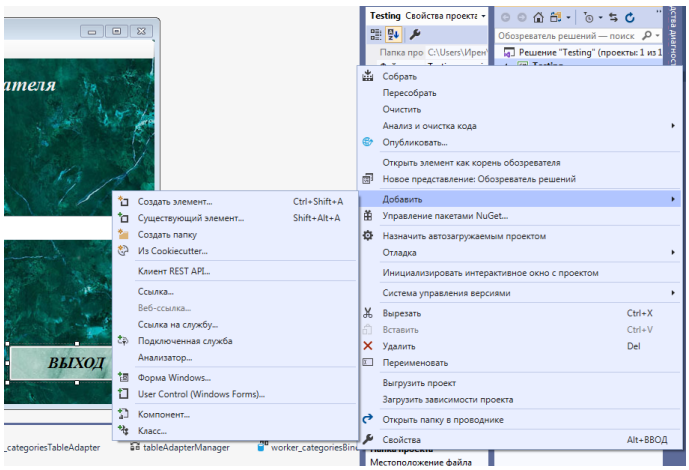
Создадим базу данных testing в SQL Server Management Studio.



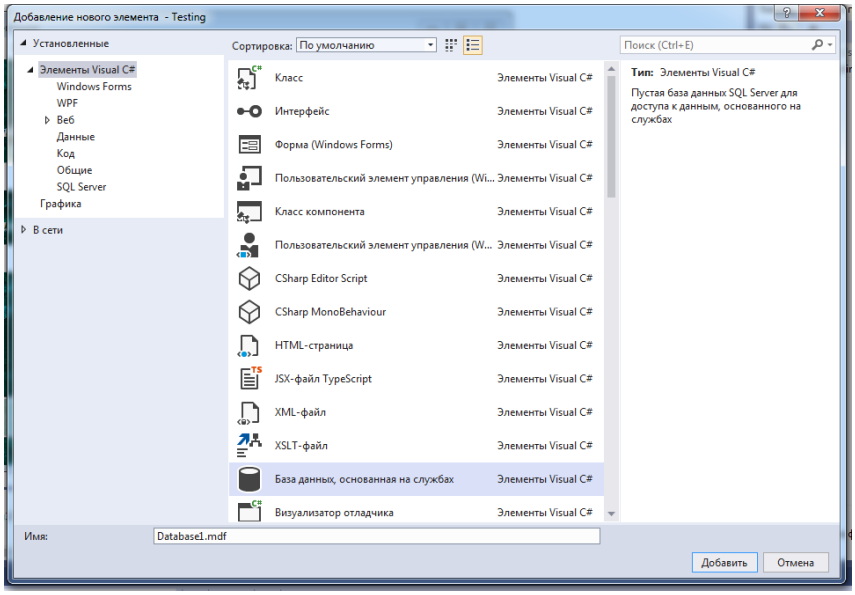
В результате анализа предметной области приложение имеет следующие внешние сущности: руководитель, сотрудник, преподаватель, студент, имеющие различные уровни доступа и привилегии. Добавим администратора базы данных, имеющего самый высокий уровень доступа, и гостя, имеющего доступ только к тестам без регистрации, для желающих проверить свои знания.

```
INSERT INTO dbo.worker_categories (name_cat_worker)
VALUES ('Администратор');
INSERT INTO dbo.worker_categories (name_cat_worker)
VALUES ('Сотрудник');
INSERT INTO dbo.worker_categories (name_cat_worker)
VALUES ('Преподаватель');
INSERT INTO dbo.worker_categories (name_cat_worker)
VALUES ('Студент');
INSERT INTO dbo.worker_categories (name_cat_worker)
VALUES ('Гость');
INSERT INTO dbo.worker_categories (name_cat_worker)
VALUES ('Руководитель');
```

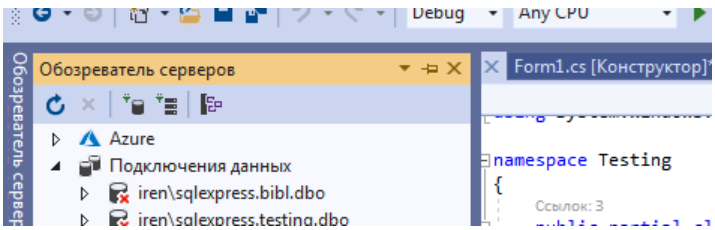
Для работы программы необходимо подключить базу данных к приложению. Нажмем ПКМ но проекте и выберем Добавить, а затем Компонент.



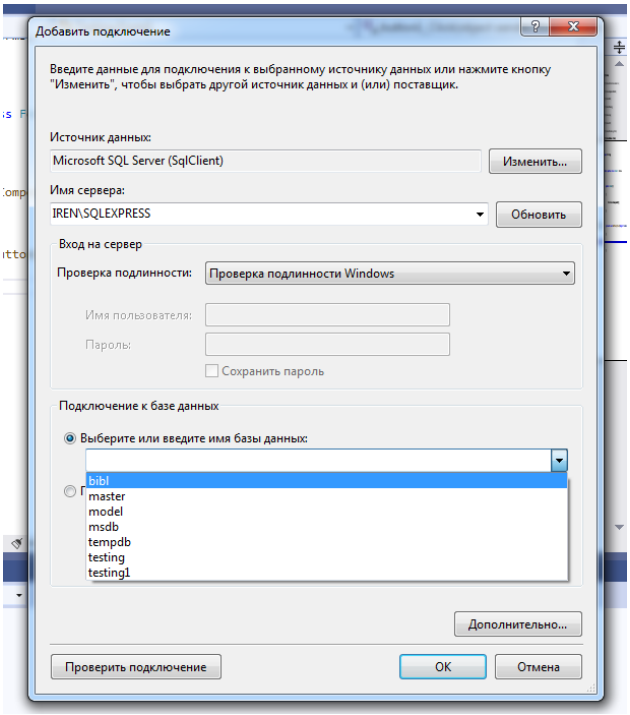
В открывшемся окне выбираем База данных, основанная на службах, а в разделе Имя пишем имя своей базы данных.



Во вкладке Обозреватель серверов нажмем значок Подключиться к базе данных

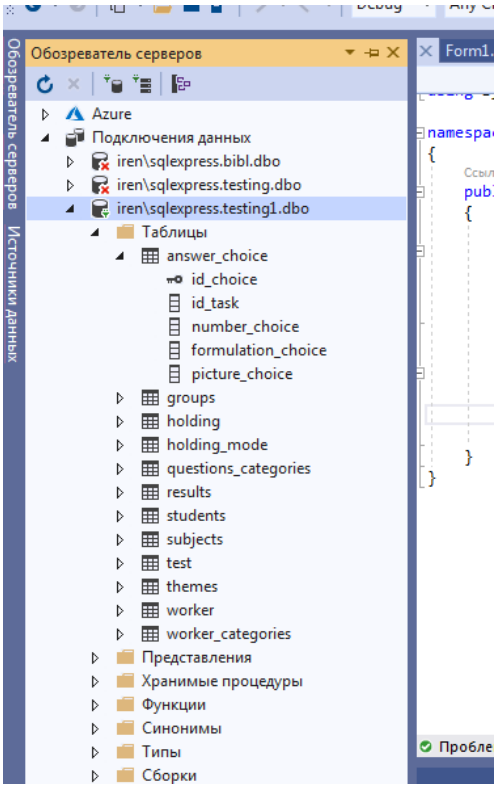


В окне **«Выбора источника»** выбираем **«Microsoft SQL Server»** (Можно подключить Access, тогда выбрать «Другое» и поставщик данных «Поставщик данных .NET Framework для OLE DB»):



При этом откроется диалог создания подключения, в нем выберем источник данных "Microsoft SQL Server…", В окне «Имя сервера» нажмем на стрелку, система сама найдет SQL сервер, аналогично с именем базы данных, которое можно выбрать из списка.

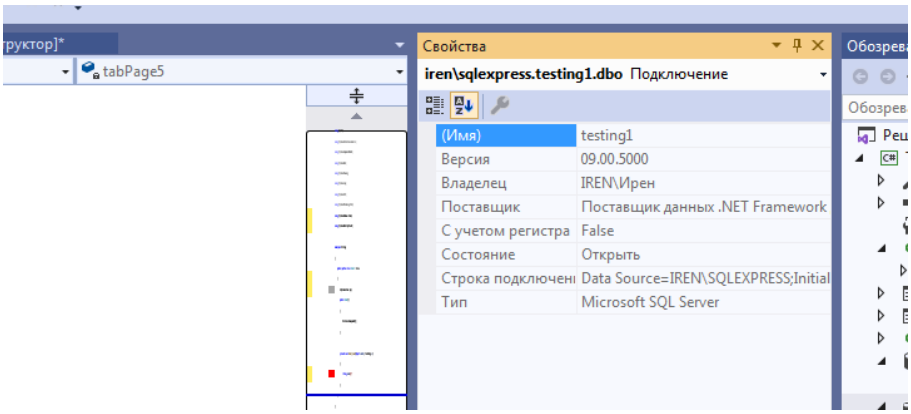
По окончанию установки связи с базой данных SQL сервер, во вкладке Обозреватель серверов можно просмотреть таблицы подключенной базы данных.



Прежде чем начать работать с этой базой данных, необходимо подключить следующие сборки:

```
using System.Data;
using System.Data.SqlClient;
```

Создадим объект класса SqlConnection для соединения с базой данных, но параметром конструктора класса является путь к ней, для этого необходим физический адрес подключенной базы данных. Чтоб его получить в обозревателе серверов кликнем ПКМ по названию БД и в контекстном меню откроем Свойства



Здесь скопируем данные раздела Строка подключения. Запишем их в строковую переменную, а затем создадим подключение

```
SqlConnection sqlc= new SqlConnection(conn);
```

Чтобы это подключение действовало для всех методов и событий модуля объявим в классе Form поле типа SqlConnection:

```
SqlConnection sqlc;
```

А в коде будем использовать это поле.

```
sqlc= new SqlConnection(conn);
```

После подключения к базе данных необходимо ее открыть с помощью метода sqlc.Open(), затем с использованием класса SqlCommand составить запрос к подключенной базе данных, после чего данные, которые хранятся в таблице, нужно считать, для этого потребуется метод ExecuteReader(), возвращающий объект SqlDataReader, который используется для чтения данных.

В цикле while (reader.Read()) в порядке следования столбцов получаем данные с помощью метода GetValue(), который возвращает данные в виде объекта типа object. После завершения работы с SqlDataReader надо его закрыть методом Close(). И пока один SqlDataReader не закрыт, другой объект SqlDataReader для одного и того же подключения мы использовать не сможем.

Для того, чтобы процесс подключения к базе данных, получения информации из нее не тормозил работу пользователя, лучше использовать потоки или асинхронные методы.

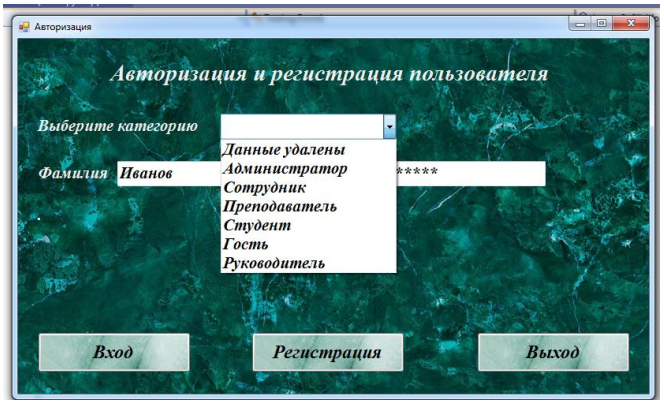
В методе private void Form1_Load(object sender, EventArgs e) добавляем слово async: private async void Form1_Load(object sender, EventArgs e). Внутри метода в командах добавляется await, например await sqlc.OpenAsync().

Для асинхронного чтения, во-первых, применяется метод ExecuteReaderAsync() класса SqlCommand, и во-вторых, метод ReadAsync() класса SqlDataReader:

После окончания работы с базой данных необходимо закрыть не только SqlDataReader, но и подключение к БД, лучше с проверкой условия наличия этого подключения и статуса:

```
if (sqlc != null && sqlc.State != ConnectionState.Closed) sqlc.Close();
```

В результате форма выглядит так



Содержание отчета

1. Название работы
2. Цель работы
3. Перечень оборудования и программного обеспечения
4. Порядок выполнения
5. Листинг отлаженного модуля в соответствии с вариантом задания.
6. Вывод.

Варианты заданий

Варианты заданий представлены в практической работе 1.

Используемая литература

1 Рудаков А.В. Технология разработки программных продуктов: учеб. пособие для студ. учреждений сред. проф. образования – 11-е изд., стер. – М.: Издательский центр «Академия», 2017. – 208 с.

2 Федорова Г.Н. Разработка, внедрение и адаптация программного обеспечения отраслевой направленности: учеб. пособие для студ. учреждений сред. проф. образования / Г.Н. Федорова.–: М.: КУРС : ИНФРА – М, 2017. – 334 с

3 Федорова Г.Н. Разработка, внедрение и адаптация программного обеспечения отраслевой направленности: Учебное пособие для СПО.- М.: КУРС, 2018. – 333 с.– ЭБС Знаниум

4 http://msdn.microsoft.com/ru-ru/library/67ef8sbd.aspx.