**Вопросы к лабораторной работе №8**

1. Что такое Binding в WPF? Какие атрибуты мы можем указывать и для чего?

Для определения привязки используется выражение типа:

|  |
| --- |
| {Binding ElementName=Имя\_объекта-источника,  Path=Свойство\_объекта-источника} |

То есть в данном случае у нас элемент TextBox является источником, а TextBlock - приемником привязки. Свойство Text элемента TextBlock привязывается к свойству Text элемента TextBox. В итоге при осуществлении ввода в текстовое поле синхронно будут происходить изменения в текстовом блоке.

2. Что такое ADO.NET? Какие существуют режимы работы?

ADO.NET — это набор классов, предоставляющих службы доступа к данным программистам, которые используют платформу .NET Framework.

При использовании **подключенного уровня (connected layer)**, кодовая база явно подключается к соответствующему хранилищу данных и отключается от него. При таком способе использования ADO.NET обычно происходит взаимодействие с хранилищем данных с помощью объектов подключения, объектов команд и объектов чтения данных.

**Автономный уровень (disconnected layer)**, позволяет работать с набором объектов DataTable (содержащихся в DataSet), который представляет на стороне клиента копию внешних данных. При получении DataSet с помощью соответствующего объекта адаптера данных подключение открывается и закрывается автоматически. Понятно, что этот подход помогает быстро освобождать подключения для других вызовов и повышает масштабируемость систем.

После выпуска .NET 3.5 SP1 в ADO.NET появилась поддержка нового API, которая называется **Entity Framework** (сокращенно EF). Технология EF показывает, что многие низкоуровневые детали работы с базами данных (например, сложные SQL-запросы) скрыты от программиста и отрабатываются за него при генерации соответствующего LINQ-запроса (например, LINQ с Entities).

3. Объясните назначение класса SqlConnection.

Объект SqlConnection представляет уникальный доступ в источнику данных SQL Server.

4. Объясните назначение класса SqlDataAdapter

Для работы с БД SQL Server пакет Microsoft.Data.SqlClient как раз предоставляет реализацию DataAdapter в виде класса **SqlDataAdapter**.

Для получения данных через объект SqlDataAdapter необходимо организовать подключение к БД и выполнить команду SELECT. Есть несколько способов создания SqlDataAdapter:

5. Объясните назначение класса SqlException. Чем данный класс отключается от класса-родителя Exception?

Исключение, которое возникает, когда SQL Server возвращает предупреждение или ошибку. Этот класс не наследуется.

SQLException наследуется от Exception и является, таким образом, checked exception. Кроме стандартных данных, которые передаются в Exception, SQLException всегда содержит дополнительные данные о ошибке: getMessage() возвращает сообщение об ошибке. getSqlState() возвращает стандартный код состояния запроса. Говоря по простому — каждый раз, когда SQL база данных выполняет запрос, она присваивает этому запросу некий статус: успешно; успешно, но с предупреждениями; совсем не успешно итд.

6. Объясните назначение класса SqlCommand.

**Класс** **SqlCommand** используется для выполнения T-**SQL**, **SQL**-операторов и хранимых процедур в базе данных **SQL** Server. Этот **класс** не может быть унаследован, поскольку он является закрытым **классом**. **SqlCommand** - это запечатанный **класс**, который наследует **класс** DbCommand и интерфейс ICloneable.

7. В чем разница между методами ExecuteNonQuery, ExecuteReader и ExecuteScalar?

* ExecuteScalar обычно используется, когда запрос возвращает единственное значение. Если он возвращает больше, то результатом является первый столбец первой строки. Примером может быть SELECT @@IDENTITY AS 'Identity'.
* ExecuteReader используется для любого результирующего набора с несколькими строками / столбцами (например,SELECT col1, col2 from sometable).
* ExecuteNonQuery обычно используется для операторов SQL без результатов (например, UPDATE, INSERT и т. д.).

8. Объясните назначение класса SqlDataReader.

**Класс** **SqlDataReader** пространства имён Microsoft.Data.SqlClient предназначен для получения данных из базы. Как следует из названия, объекты данного **класса** только читают информацию, но не изменяют данные в базе.

9. Объясните назначение класса SqlParameter.

Для определения параметров используется объект **SqlParameter**. Этот объект имеет ряд конструкторов, наиболее используемые из них:

* **SqlParameter()**
* **SqlParameter(String, Object)**: первый параметр конструктора передает имя, а второй - значение параметра
* **SqlParameter(String, SqlDbType)**: первый параметр конструктора передает имя параметра, а второй - его тип в виде объекта **SqlDbType**
* **SqlParameter(String, SqlDbType, Int32)**: первый параметр конструктора передает имя параметра, второй - его тип в виде объекта **SqlDbType**, а третий - размер
* **SqlParameter(String, SqlDbType, Int32, String)**: конструктор последовательно принимает значения для имя параметра, его типа, размера и имени стобца в таблице

10.Объясните назначение класса SqlTransaction.

Предоставляет транзакцию Transact-SQL, которая должна быть выполнена над базой данных SQL Server. Этот класс не наследуется.

11.Объясните назначение классов DataSet, DataTable, DataColumn, DataRow, DataRelation

Источник данных DataSet Основным объектом ADO является источник данных, представленный объектом DataSet. DataSet состоит из объектов типа DataTable и объектов DataRelation. В коде к ним можно обращаться как к свойствам объекта DataSet, то есть, используя точечную нотацию. Свойство Tables возвращает объект типа DataTableCollection, который содержит все объекты DataTable используемой базы данных.

Таблицы и поля (объекты DataTable и DataColumn)

Объекты DataTable используются для представления одной из таблиц базы данных в DataSet. В свою очередь, DataTable составляется из объектов DataColumn. DataColumn - это блок для создания схемы DataTable. Каждый объект DataColumn имеет свойство DataType, которое определяет тип данных, содержащихся в каждом объекте DataColumn. Например, можно ограничить тип данных до целых, строковых и десятичных чисел. Поскольку данные, содержащиеся в DataTable, обычно переносятся обратно в исходный источник данных, необходимо согласовывать тип данных с источником.

Объекты DataRelation

Объект DataSet имеет также свойство Relations, возвращающее коллекцию DataRelationCollection, которая в свою очередь состоит из объектов DataRelation. Каждый объект DataRelation выражает отношение между двумя таблицами (сами таблицы связаны по какому-либо полю (столбцу)). Следовательно, эта связь осуществляется через объект DataColumn.

Строки (объект DataRow)

Коллекция Rows объекта DataTable возвращает набор строк (записей) заданной таблицы. Эта коллекция используется для изучения результатов запроса к базе данных. Мы можем обращаться к записям таблицы как к элементам простого массива.