

Міністерство освіти і науки України
Криворізький коледж Національного авіаційного університету

Курс лекцій
з предмету
***“Об’єктно-орієнтоване
програмування”***

Спеціальність 5.05010301

м. Кривий Ріг

Лекция №

Тема: Работа с базами данных в приложениях *Windows Forms*

Вопросы, рассматриваемые на лекции

- Основы ADO .NET
- класс DataTable;
- класс DataColumns;
- класс DataRows;
- методы и свойства;
- примеры задач.

Отсоединенная модель программирования

- В ADO .NET используется модель доступа – доступ к отсоединенным данным. При этом соединение устанавливается лишь на то время, которое необходимо для проведения определенной операции над базой данных.
- Возникает возможность поддержки большего количества параллельно работающих пользователей за счет постепенного увеличения количества клиентских компьютеров. Это преимущество имеет особенно большое значение при создании Web-приложений.



Концепция доступа к данным в ADO .NET основана на использовании двух компонентов:

- **НАБОРА ДАННЫХ** (представляется объектом класса DataSet) со стороны клиента. Это локальное временное хранилище данных;
- **ПРОВАЙДЕРА ДАННЫХ** (представляется объектом класса DataProvider). Это посредник, обеспечивающий взаимодействие приложения и базы данных со стороны базы данных (в распределенных приложениях – со стороны сервера).

Провайдеры данных ADO.NET

- Несмотря на подчеркнутое значение отсоединенной модели программирования, для извлечения, обновления, вставки и удаления данных необходимо подключиться к физической базе данных. Программное обеспечение ADO.NET для подсоединения и взаимодействия с физической базой данных называется **провайдером** данных ADO.NET.
- **Провайдер данных (*data provider*)** — это управляемый код .NET, который эквивалентен провайдеру OLEDB или драйверу ODBC.
- Провайдер данных состоит из нескольких объектов, которые реализуют необходимую функциональность в соответствии с определениями своих классов и интерфейсов.

Провайдеры данных ADO.NET

Наиболее часто используются следующие провайдеры:

- SQL Server .NET Data Provider – предназначен для работы с базами данных Microsoft SQL Server;
- OLE DB.NET Data Provider – предназначен для работы с источниками данных SQL Server, Oracle, Access;
- ODBC.NET Data Provider.– обеспечивает доступ к источникам данных через их ODBC-драйвера;
- ORACLE.NET Data Provider – предназначен для работы с базой данных Oracle.

Провайдеры данных и пространства имен

Провайдер данных	Пространство имен провайдера
Microsoft SQL Server 7.0 и выше	System.Data.SqlClient
Oracle 8.1.6 и выше	System.Data.OracleClient
Поддержка SqlXml в SQL Server	System.Data.SqlXml
Любой источник данных ODBC	System.Data.ODBC
Любой источник данных OleDb	System.Data.OleDb

Провайдеры данных и пространства имен

Основные объекты, входящие в состав провайдера, - это Connection, Command, DataAdapter и DataReader.

Connection используется для установления соединения с источником данных, а также для управления транзакциями.

Command позволяет манипулировать данными источника, а также выполнять хранимые процедуры. При этом могут использоваться параметры для передачи данных в обоих направлениях.

DataAdapter служит связующим звеном между DataSet и источником данных. Он использует Command для выполнения команд SQL как для заполнения DataSet данными, так и для обратной передачи измененных клиентом данных к источнику. Для выполнения этих функций объект имеет 4 метода: SelectCommand, InsertCommand, UpdateCommand и Deletecommand.

DataReader представляет однонаправленный поток данных от источника только на чтение. Если приложение клиента не модифицирует данные и не требуется произвольная выборка данных, а достаточно их однократного просмотра, то использование DataReader вместо DataSet позволит сохранить ресурсы RAM и CPU, а также поднять быстродействие приложения.

Создание подключения к БД

1 шаг – создание объекта подключения

Объект подключения источника данных наследуется от класса `DbConnection` и получает уже готовую логику, реализованную в базовых классах провайдеров (`OleDbConnection`, `SqlConnection`, `OracleConnection` и т. д.).

Создание объектов для разных провайдеров:

```
OleDbConnection oledbconn = new OleDbConnection();  
SqlConnection sqlconn = new SqlConnection();  
OdbcConnection odbconn = new OdbcConnection();  
OracleConnection oracleconn = new OracleConnection();
```

`SqlConnection.ConnectionString`

[http://msdn.microsoft.com/ru-ru/library/system.data.sqlclient.sqlconnection.connectionstring\(v=vs.110\).aspx](http://msdn.microsoft.com/ru-ru/library/system.data.sqlclient.sqlconnection.connectionstring(v=vs.110).aspx)

Свойства SqlConnection

Имя	Описание
<u>ClientConnectionId</u>	Идентификатор соединения последней попытки подключения, независимо от того, успешно ли выполнена попытка или завершилась ошибкой.
<u>ConnectionString</u>	Получает или задает строку, используемую для открытия базы данных SQL Server.
<u>ConnectionTimeout</u>	Получает время ожидания при попытке установки подключения, по истечении которого попытка подключения завершается и создается ошибка.
<u>Database</u>	Получает имя текущей базы данных или базы данных, которая будет использоваться после открытия подключения.
<u>DataSource</u>	Получает имя экземпляра SQL Server, к которому осуществляется подключение.
<u>PacketSize</u>	Получает размер сетевых пакетов (в байтах), используемых при взаимодействии с экземпляром SQL Server.
<u>ServerVersion</u>	Получает строку, содержащую версию экземпляра SQL Server, к которому подключается клиент.
<u>State</u>	Отображает состояние SqlConnection во время последней сетевой операции, выполненной по подключению.
<u>StatisticsEnabled</u>	Когда задано значение true, разрешает сбор статистических сведений для текущего подключения.
<u>WorkstationId</u>	Получает строку, определяющую клиента базы данных.

Свойства OleDbConnection

Имя	Описание
<u>Connection String</u>	Возвращает или задает строку, используемую для открытия базы данных.
<u>Connection Timeout</u>	Получает время ожидания при попытке установки подключения, по истечении которого попытка подключения завершается и создается ошибка.
<u>Container</u>	Возвращает контейнер <u>IContainer</u> , содержащий компонент <u>Component</u> .
<u>Database</u>	Получает имя текущей базы данных или базы данных, которая будет использоваться после открытия подключения
<u>DataSource</u>	Получает имя сервера или имя файла источника данных.
<u>Provider</u>	Получает имя поставщика OLE DB, указанное в выражении "Provider= " строки подключения.
<u>ServerVersion</u>	Получает строку, содержащую номер версии сервера, к которому подключается клиент.
<u>Site</u>	Получает или задает экземпляр <u>ISite</u> для компонента <u>Component</u> .
<u>State</u>	Получает текущее состояние подключения.

Значения свойства State (перечисление ConnectionState)

Имя члена	Описание
Broken	Подключение к источнику данных разорвано. Это может произойти только после открытия подключения. Подключение в этом режиме может быть закрыто, а затем повторно открыто. (Это значение зарезервировано для будущих версий продукта.)
Closed	Подключение закрыто.
Connecting	Объект подключения подключается к источнику данных.
Executing	Объект подключения выполняет команду. (Это значение зарезервировано для будущих версий продукта.)
Fetching	Объект подключения получает данные. (Это значение зарезервировано для будущих версий продукта.)
Open	Подключение открыто.

Пример:

```
if (connect.State == ConnectionState.Open)  
connect.Close();
```

Строка подключения

2 шаг – задание строки подключения

Для того, чтобы открыть подключение необходимо указать, какая информация необходима для выполнения этой задачи, например, имя сервера, идентификатор пользователя, пароль и т. Д.

Поскольку каждому целевому источнику подключения может понадобиться особый набор информации, позволяющей ADO.NET подключиться к источнику данных, выбирают гибкий механизм указания всех параметров через строку подключения **ConnectionString**.

Строка подключения - это набор разделенных точкой с запятой пар атрибут-значений, определяющих, как следует устанавливать подключение к источнику данных. Хотя строки подключения имеют тенденцию выглядеть одинаково, возможные и обязательные атрибуты различаются в зависимости от поставщика данных и источника данных.

Строка подключения

Перечень некоторых атрибутов строк подключения для установления соединений:

- ✓ Provider - имя провайдера (только для OLE DB);
- ✓ Driver - ODBC-драйвер (только для ODBC);
- ✓ Connection Timeout - время в секундах для попыток установки соединения (по умолчанию 15 сек.)
- ✓ InitialCatalog - имя базы данных в SQLServer;
- ✓ Data Source - имя сервера или путь к базе данных Access;
- ✓ Password - пароль;
- ✓ User ID - логин;
- ✓ Workstation ID - имя рабочей станции или компьютера
- ✓ Integrated Security - true, если выбирается Windows Authentication, false - если SQL Authentication, sspi – эквивалентно true.

Строка подключения

Например, строка соединения при использовании провайдера OLE DB и подключении к базе данных lab.mdb будет выглядеть так:

```
string str = "Provider = Microsoft.Jet.OLEDB.4.0; DataSource = lab.mdb";
```

Теперь свойстваConnectionString объекта подключения oledbconn можно задать необходимое значение:

```
oledbconn.ConnectionString = str;
```

Эти действия можно выполнить с помощью одной строки, например:

```
oledbconn.ConnectionString = "Provider = Microsoft.Jet.OLEDB.4.0;  
DataSource = lab.mdb ";
```

[http://msdn.microsoft.com/ru-ru/library/system.data.sqlclient.sqlconnection.connectionstring\(v=vs.110\).aspx](http://msdn.microsoft.com/ru-ru/library/system.data.sqlclient.sqlconnection.connectionstring(v=vs.110).aspx)

Методы SqlConnection

Имя	Описание
<u>BeginTransaction</u>	Начинает транзакцию базы данных.
<u>ChangeDatabase</u>	Изменяет текущую базу данных для открытого подключения SqlConnection.
<u>ChangePassword</u>	Заменяет пароль SQL Server для пользователя, указанного в строке подключения, заданным новым паролем.
<u>ClearAllPools</u>	Очищает пул подключений.
<u>ClearPool</u>	Очищает пул подключений, связанный с заданным подключением.
<u>Close</u>	Закрытие подключения к базе данных. Рекомендуется использовать этот метод для закрытия любого открытого подключения.
<u>CreateCommand</u>	Создает и возвращает объект <u>SqlCommand</u> , связанный с SqlConnection.
<u>Dispose()</u>	Освобождает все ресурсы, используемые объектом <u>Component</u> .

Методы SqlConnection

Имя	Описание
<u>EnlistDistributedTransaction</u>	Выполняет присоединение указанной транзакции как распределенной транзакции.
<u>EnlistTransaction</u>	Выполняет присоединение указанной транзакции как распределенной транзакции.
<u>GetSchema()</u>	Возвращает информацию схемы для источника данных этого объекта SqlConnection. Дополнительные сведения о схеме см. <u>Коллекции схем SQL Server</u> .
<u>GetSchema</u>	Возвращает сведения схемы для источника данных этого объекта SqlConnection, используя указанную строку для имени схемы.
<u>Open</u>	Открывает подключение к базе данных со значениями свойств, определяемыми объектом <u>ConnectionString</u> .
<u>ResetStatistics</u>	Если сбор статистики разрешен, все значения сбрасываются в нуль.
<u>RetrieveStatistics</u>	Возвращает коллекцию пар имя-значение статистических данных на момент вызова метода.

Методы SqlConnection

3 шаг – открытие базы данных (для OleDb)

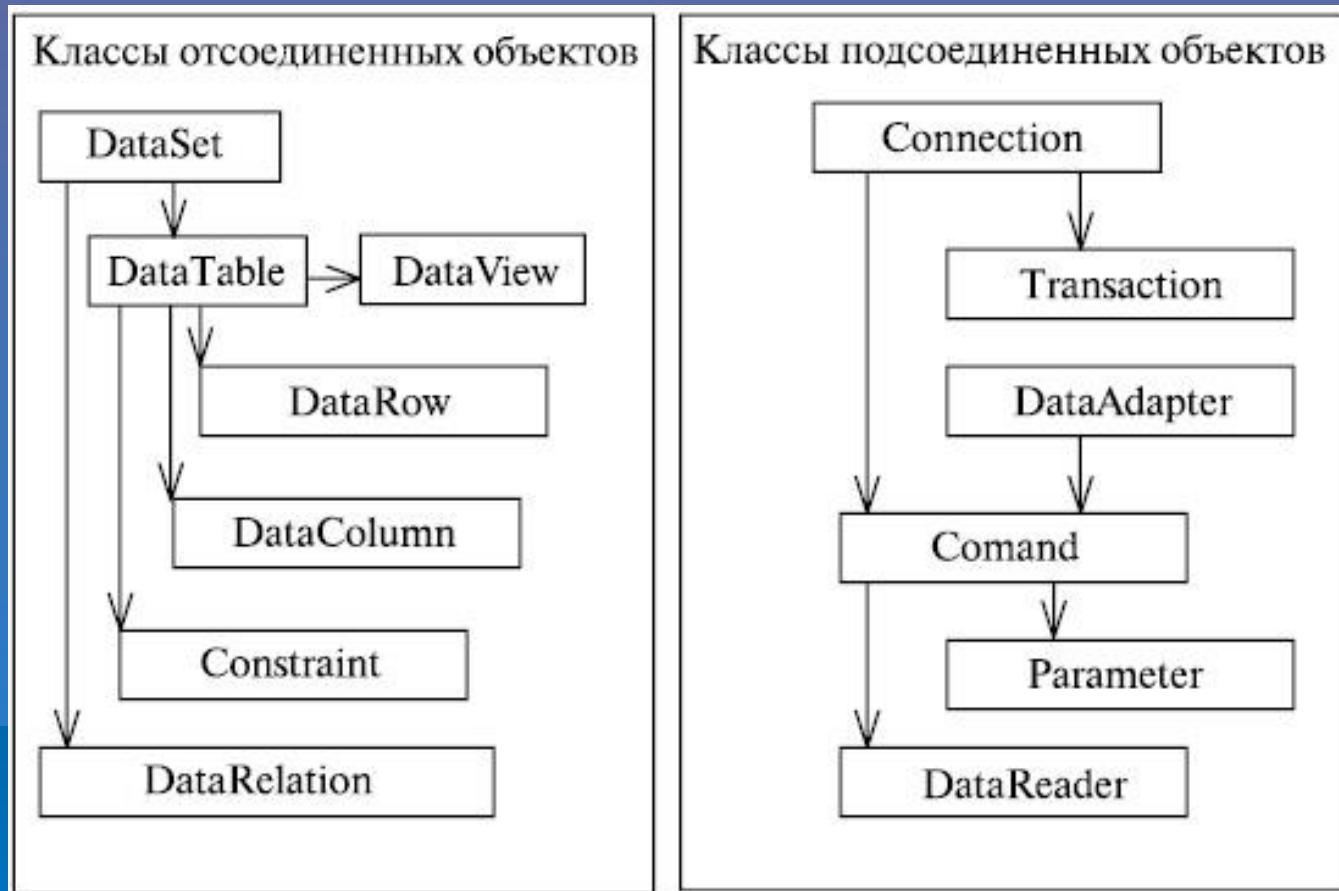
```
OleDbConnection oledbconn = new OleDbConnection();
oledbconn.ConnectionString=@"Provider=Microsoft.Jet.OLEDB.4.0;
DataSource=c:\1\lab.mdb";
try
{
oledbconn.Open();
MessageBox.Show( "З'єднання відчинено" );
// Виконання корисних дій
oledbconn.Close();
MessageBox.Show( " З'єднання зачинено " )
}
catch(System.Data.OleDb.OleDbException e1)
// Виникнення OleDbException
{
MessageBox.Show("З'єднання відчинено "+e1.Message);
}
catch (System.InvalidOperationException e2)
// Помилка відкриття бази даних
{
MessageBox.Show("З'єднання не відчинено " + e2.Message);
}}
```

Строка подключения

3 шаг – открытие базы данных (для Sql)

```
try
{
using (SqlConnection conn = new SqlConnection ( source ))
{
// Відкрити з'єднання
conn.Open () ;
// Зробити що-небудь корисне. Закрити з'єднання самостійно conn.Close () ;
}
}
catch (SqlException)
{
//
```

ADO .NET. Объектная модель



- Классы подсоединенных объектов обеспечивают установление соединения с базой данных и управление базой со стороны приложения;
- Классы отсоединенных объектов обеспечивают сохранение, использование и преобразование полученной от базы данных информации на стороне приложения.

Свойства класса DataSet

Имя	Описание
<u>CaseSensitive</u>	Возвращает или задает значение, определяющее, учитывается ли регистр при сравнении строк в объектах DataTable.
<u>Container</u>	Возвращает контейнер для компонента.
<u>DataSetName</u>	Возвращает или задает имя текущего DataSet.
<u>DefaultViewManager</u>	Получает новое представление данных класса DataSet для осуществления фильтрации, поиска или перехода с помощью настраиваемого класса DataViewManager.
<u>DesignMode</u>	Возвращает значение, показывающее, находится ли компонент в настоящий момент в режиме разработки.
<u>EnforceConstraints</u>	Возвращает или задает значение, определяющее соблюдение правил ограничения при попытке совершения операции обновления.
<u>Events</u>	Возвращает список обработчиков событий, которые прикреплены к этому компоненту.
<u>ExtendedProperties</u>	Возвращает коллекцию настраиваемых данных пользователя, связанных с DataSet.
<u>IsInitialized</u>	Получает значение, указывающее, инициализирован ли набор данных DataSet.
<u>Locale</u>	Возвращает или задает сведения о языке, используемые для сравнения строк таблицы.
<u>Namespace</u>	Возвращает или задает пространство имен класса DataSet.
<u>Prefix</u>	Возвращает или задает префикс XML, который является псевдонимом пространства имен класса DataSet.
<u>Relations</u>	Возвращает коллекцию соотношений, связывающих таблицы и позволяющих переходить от родительских таблиц к

Свойства класса DataSet

Примеры :

1) DataColumn parentColumn = dataSet.Tables["Suppliers"].Columns["SupplierID"];

2) dataSet.Relations.Add(relation);

3)

```
private void CopyDataSet(DataSet dataSet)
```

```
{
```

```
    DataSet copyDataSet;
```

```
    copyDataSet = dataSet.Copy();
```

```
}
```

Методы класса DataSet

Имя	Описание
<u>AcceptChanges</u>	Сохраняет все изменения, внесенные в класс DataSet после его загрузки или после последнего вызова метода <u>AcceptChanges</u> .
<u>BeginInit</u>	Начинает инициализацию класса DataSet, используемого в форме или другим компонентом. Инициализация происходит во время выполнения.
<u>Clear</u>	Удаляет из класса DataSet любые данные путем удаления всех строк во всех таблицах.
<u>Clone</u>	Копирует структуру класса DataSet, включая все схемы, соотношения и ограничения объекта <u>DataTable</u> . Данные не копируются.
<u>Copy</u>	Копирует структуру и данные для DataSet.
<u>CreateDataReader</u>	Возвращает объект <u>DataTableReader</u> с одним результирующим набором для каждой последовательности <u>DataTable</u> в той же последовательности, в которой таблицы отображаются в коллекции <u>Tables</u> .
<u>Dispose</u>	Освобождает все ресурсы, используемые объектом <u>MarshalByValueComponent</u> . (Унаследовано от <u>MarshalByValueComponent</u> .)
<u>EndInit</u>	Завершает инициализацию объекта DataSet, который используется в форме или другим компонентом. Инициализация происходит во время выполнения.
<u>Equals(Object)</u>	Определяет, равен ли заданный объект текущему объекту. (Унаследовано от <u>Object</u> .)
<u>Finalize</u>	Позволяет объекту попытаться освободить ресурсы и выполнить другие операции очистки, перед тем как объект будет утилизирован в процессе сборки мусора.
<u>GetChanges()</u>	Получает копию класса DataSet, содержащую все изменения

Методы класса DataSet

Имя	Описание
<u>Load(IDataReader, LoadOption, DataTable[])</u>	Заполняет набор данных DataSet значениями из источника данных с помощью предоставляемого объекта <u>IDataReader</u>
<u>MemberwiseClone</u>	Создает неполную копию текущего объекта Object. (Унаследовано от <u>Object</u> .)
<u>Merge(DataRow[])</u>	Осуществляет слияние массива объектов <u>DataRow</u> и текущего класса DataSet.
<u>Merge(DataSet)</u>	Осуществляет слияние указанного объекта DataSet и его схемы с текущим объектом DataSet.
<u>Merge(DataTable)</u>	Осуществляет слияние указанного объекта <u>DataTable</u> и его схемы с текущим объектом DataSet.
<u>RejectChanges</u>	Отменяет все изменения, внесенные в класс DataSet после его создания или после последнего вызова метода <u>DataSet.AcceptChanges</u> .
<u>Reset</u>	Очищает все таблицы и удаляет все связи, внешние ограничения и таблицы из DataSet. Для восстановления исходного состояния класса DataSet необходимо переопределить метод <u>Reset</u> в подклассах.

```
DataSet data=new DataSet("База_даних");
```

```
DataSet data=new DataSet();
```

```
DataColumn parentColumn = dataSet.Tables["Suppliers"].Columns["SupplierID"];
```


DataTable

Каждый объект DataTable представляет одну таблицу базы данных.

Таблица в каждый конкретный момент своего существования характеризуется:

- СХЕМОЙ таблицы,
- СОДЕРЖИМЫМ таблицы (информацией).

СХЕМА таблицы (структура объекта DataTable) определяется двумя наборами:

- множеством столбцов таблицы (набор DataColumnns, состоящий из множества объектов DataColumn),
- множеством ограничений таблицы (набор Constraints, состоящий из множества объектов Constraint).

Свойства класса DataTable

Имя	Описание
<u>ChildRelations</u>	Получает коллекцию дочерних отношений для объекта DataTable.
<u>Columns</u>	Получает коллекцию столбцов, принадлежащих данной таблице.
<u>DataSet</u>	Получает класс DataSet, к которому принадлежит данная таблица.
<u>DesignMode</u>	Возвращает значение, показывающее, находится ли компонент в настоящий момент в режиме разработки.
<u>Events</u>	Возвращает список обработчиков событий, которые прикреплены к этому компоненту.
<u>HasErrors</u>	Получает значение, указывающее наличие ошибок в строках таблиц класса DataSet, к которому принадлежат таблицы.
<u>IsInitialized</u>	Получает значение, указывающее, инициализирована ли таблица DataTable.
<u>Locale</u>	Возвращает или задает сведения о языке, используемые для сравнения строк таблицы.
<u>MinimumCapacity</u>	Возвращает или задает начальный размер таблицы.
<u>Namespace</u>	Возвращает или задает пространство имен для представления данных объекта DataTable в формате XML.
<u>ParentRelations</u>	Получает коллекцию родительских отношений для объекта DataTable.
<u>Prefix</u>	Возвращает или задает пространство имен для представления данных объекта DataTable в формате XML.
<u>PrimaryKey</u>	Возвращает или задает массив столбцов, которые являются столбцами первичного ключа для таблицы данных.

Свойства класса DataTable

Примеры :

```
1) private void ClearDataSet(DataSet dataSet)
{
    foreach(DataTable table in dataSet.Tables)
    {
        Console.WriteLine(table.TableName +
            "Rows.Count = " +
            table.Rows.Count.ToString());
    }
    dataSet.Clear();
}
```

```
2) private void PrintValues(DataTable table)
{
    foreach(DataRow row in table.Rows)
    {
        foreach(DataColumn column in table.Columns)
        { Console.WriteLine(row[column]); }
    }
}
```

```
3) private void GetTableNames(DataSet
dataSet)
{
    foreach(DataTable table in dataSet.Tables)
    {
        Console.WriteLine(table.TableName);
    }
}
```

```
4) table.Columns.Add(column);
```

```
5) DataSet ds = new DataSet();
    DataTable tablica = new
    DataTable("Products");
    ds.Tables.Add(tablica);
или
    DataSet ds = new DataSet();
    DataTable tablica =
    ds.Tables.Add("Products");
```

События класса DataTable

Имя	Описание
<u>ColumnChanged</u>	Происходит после изменения значения указанного объекта DataColumn в DataRow.
<u>ColumnChanging</u>	Происходит при изменении значения указанного объекта DataColumn и DataRow.
<u>Disposed</u>	Добавляет обработчик событий, чтобы прослушивать событие Disposed для компонента. (Унаследовано от MarshalByValueComponent.)
<u>Initialized</u>	Происходит после инициализации таблицы DataTable.
<u>RowChanged</u>	Происходит после успешного изменения DataRow.
<u>RowChanging</u>	Происходит при изменении объекта DataRow.
<u>RowDeleted</u>	Происходит после удаления строки таблицы.
<u>RowDeleting</u>	Происходит перед удалением строки таблицы.
<u>TableCleared</u>	Происходит после очистки DataTable.
<u>TableClearing</u>	Происходит, когда очищается таблица DataTable.
<u>TableNewRow</u>	Происходит, когда вставляется новая строка DataRow.

Класс DataColumn

Имя	Описание
<u>AllowDBNull</u>	Возвращает или задает значение, указывающее на допустимость нулевых значений в этом столбце для строк, принадлежащих таблице.
<u>AutoIncrement</u>	Возвращает или задает значение, показывающее, увеличивать ли автоматически значение столбца для новых строк, добавляемых в таблицу.
<u>AutoIncrementSeed</u>	Возвращает или задает начальное значение для столбца, свойство которого <u>AutoIncrement</u> установлено на true. Значение по умолчанию — 0.
<u>AutoIncrementStep</u>	Возвращает или задает инкремент для столбца, свойство <u>AutoIncrement</u> которого установлено на true.
<u>Caption</u>	Возвращает или задает заголовок для столбца.
<u>ColumnMapping</u>	Возвращает или задает <u>MappingType</u> столбца.
<u>ColumnName</u>	Возвращает или задает имя столбца в <u>DataColumnCollection</u> .
<u>Container</u>	Возвращает контейнер для компонента. (Унаследовано от <u>MarshalByValueComponent</u> .)
<u>DataType</u>	Возвращает или задает тип данных, хранимых в столбце.
<u>DateTimeMode</u>	Возвращает или задает DateTimeMode столбца.
<u>DefaultValue</u>	Возвращает или задает значение по умолчанию для столбца при создании новых строк.
<u>Expression</u>	Возвращает или задает выражение, используемое для фильтрации строк, расчета значений в столбце либо создания составного столбца.
<u>MaxLength</u>	Возвращает или задает максимальную длину текстового

Класс DataColumn

Имя	Описание
<u>Ordinal</u>	Возвращает позицию (с нуля) столбца в коллекции DataColumnCollection .
<u>ReadOnly</u>	Возвращает или задает значение, указывающее на допустимость изменения столбца после добавления строки в таблицу.
<u>Table</u>	Получает DataTable , к которому принадлежит столбец.
<u>Unique</u>	Возвращает или задает значение, показывающее, должны ли значения в каждой строке столбца быть уникальными.