ПІД’ЄДНАНИЙ РЕЖИМ

Основные параметры строки соединения

Параметр	Описание
Provider (Поставщик)	Свойство применяется для установки или возврата имени поставщика для соединения, используется только для объектов OleDbConnection
Connection Timeout или Connect Timeout (Время ожидания связи)	Длительность времени ожидания связи с сервером перед завершением попытки и генерацией исключения в секундах. По умолчанию 15
Initial Catalog (Исходный каталог)	Имя базы данных
Data Source (Источник данных)	Имя используемого SQL-сервера, когда установлено соединение, или имя файла базы данных Microsoft Access
Password (Пароль)	Пользовательский пароль для учетной записи SQL Server
User ID (Пользовательский ID)	Пользовательское имя для учетной записи SQL Server
Integrated Security или Trusted Connection (Интегрированная безопасность, или Доверительное соединение)	Параметр, который определяет, является ли соединение защищенным. True, False и SSPI - возможные значения (SSPI - эквивалент True)
Persist Security Info (Удержание защитной информации)	Когда установлено False, нуждающаяся в защите информация, такая как пароль, не возвращается как часть соединения, если связь установлена или когда-либо была установленной. Выставление этого свойства в True может быть рискованным в плане безопасности.

Приклади SQL Server ConnectionString

```
"Persist Security Info=False;Integrated Security=true;
Initial Catalog=DBName;Server=ServerName"
```

```
"Persist Security Info=False;Trusted_Connection=True;
database=DBName; server=(local)"
```

```
"Persist Security Info=False;User
ID=*****;Password=*****;Initial
Catalog=DBName;Server=MySqlServer\MSSQL1"
```

Приклади синтаксу для інших провайдерів - https://docs.microsoft.com/ru-ru/dotnet/framework/data/adonet/connection-string-syntax

DBConnection

- Представляє підєднання до БД.
- **Простір імен:** System.Data.Common
  Збірка: System.Data (в System.Data.dll)
  Ієрархія успадкування

- System.Object
  - System.MarshalByRefObject
    - System.ComponentModel.Component
      - System.Data.Common.DbConnection
        - System.Data.EntityClient.EntityConnection
        - System.Data.Odbc.OdbcConnection
        - System.Data.OleDb.OleDbConnection
        - System.Data.OracleClient.OracleConnection
        - System.Data.SqlClient.SqlConnection

Властивості

Назва	Опис
ConnectionString	Возвращает или задает строку, используемую для открытия подключения.
ConnectionTimeout	Получает время ожидания при установлении подключения, по истечении которого попытка завершается и создается ошибка.
Database	Получает имя текущей базы данных после открытия подключения или имя базы данных, указанный в строке подключения, перед открытием подключения.
DataSource	Возвращает имя сервера базы данных, к которой требуется подключиться.
Events	Возвращает список обработчиков событий, которые прикреплены к этому ComponentВозвращает список обработчиков событий, которые прикреплены к этому Component.(Наследуется от Component.)
ServerVersion	Возвращает строку, представляющую версию сервера, к которому подключен объект.
State	Возвращает строку, описывающую состояние

Метод

Назва	Опис
BeginDbTransaction()	Начинает транзакцию базы данных.
ChangeDatabase(String)	Изменяет текущую базу данных для открытого подключения.
Close()	Закрывает соединение с базой данных. Рекомендуется использовать этот метод для закрытия любого открытого подключения.
CreateCommand()	Создает и возвращает DbCommand объект, связанный с текущим соединением.
Dispose(Boolean)	Освобождает неуправляемые ресурсы, используемые объектом Component, а при необходимости освобождает также управляемые ресурсы.(Наследуется от Component.)
EnlistTransaction(Transaction)	Присоединяет указанную транзакцию.

Назва	Опис
Finalize()	Освобождает неуправляемые ресурсы и выполняет другие операции очистки, перед тем как объект Component будет удален при сборке мусора.(Наследуется от Component.)
GetSchema(String, String[])	Возвращает сведения схемы для источника данных этого объекта DbConnection используя указанную строку для имени схемы и указанный массив строк для значений ограничений.
OnStateChange()	Вызывает событие StateChange.
Open()	Открывает подключение к базе данных с параметрами, указанными в ConnectionString.
OpenAsync()	Асинхронная версия Open, который открывает подключение к базе данных с параметрами, указанными в ConnectionString. Этот метод вызывает виртуальный метод OpenAsync с CancellationToken.None.

Таблица 3 - Ошибки SQL Server

Номер ошибки	Описание
17	Неверное имя сервера
4060	Неверное название базы данных
18456	Неверное имя пользователя или пароль

Таблица 4 Уровни ошибок SQL Server

Интервал возвращаемых значений	Описание	Действие
11-16	Ошибка, созданная пользователем	Пользователь должен повторно ввести верные данные
17-19	Ошибки программного обеспечения или оборудования	Пользователь может продолжать работу, но некоторые запросы будут недоступны. Соединение остается открытым
20-25	Ошибки программного обеспечения или оборудования	Сервер закрывает соединение. Пользователь должен открыть его снова

```
static string _connectString = "Persist Security Info = False;" +
            @"Integrated Security = True; Initial Catalog =
torg_firm ; Server = HP-LADA\LADASERV";
try
{
    SqlConnection _connect = new SqlConnection(_connectString);
    _connect.Open();

    //do smth...
    _connect.ChangeDatabase("test");

    //do smth...
    _connect.ChangeDatabase("torg_firm");

    //do smth...
        _connect.Close();
        _connect.Dispose();
 }
catch (Exception ex)
    {
                Console.WriteLine(ex.Message);
                throw;
     }
```

DBCommand

- Представляет инструкцию SQL или хранимую процедуру, выполняемую с источником данных. Предоставляет базовый класс для классов, определяемых базой данных, которые представляют команды.
- **Пространство имен:** System.Data.Common
  Сборка: System.Data (в System.Data.dll)

  Иерархия наследования
- System.Object
  System.MarshalByRefObject
  System.ComponentModel.Component
  System.Data.Common.DbCommand
  System.Data.EntityClient.EntityCommand
  System.Data.Odbc.OdbcCommand
  System.Data.OleDb.OleDbCommand
  System.Data.OracleClient.OracleCommand
  System.Data.SqlClient.SqlCommand

Властивості Command:

Connection - подключение к базе данных

CommandType - тип команды (запроса),

Text - Текстовая команда состоит из SQL-конструкции,

StoredProcedure -Текстовая команда состоит из названия хранимой процедуры.

TableDirect -Текстовая команда состоит из названия таблицы базы данных

CommandText - собственно сам текст запроса.

Властивості класа DBCommand

Имя	Описание
CommandText	Возвращает или задает текст команды для выполнения в источнике данных.
CommandTimeout	Возвращает или задает время ожидания перед завершением попытки выполнить команду и созданием ошибки.
CommandType	Указывает или определяет как CommandText интерпретируется свойство.
Connection	Возвращает или задает соединение DbConnection, используемое этой командойDbCommand.
DbParameterCollection	Возвращает коллекцию DbParameter объектов.
DbTransaction	Возвращает или задает DbTransaction внутри этого DbCommand выполняется объект.

Events	Возвращает список обработчиков событий, которые прикреплены к этому Component.(Наследуется от Component.)
Parameters	Возвращает коллекцию DbParameter объектов. Дополнительные сведения о параметрах см. в разделе Настройка параметров и типы данных параметров.
Transaction	Возвращает или задает DbTransaction внутри этого DbCommand выполняется объект.
UpdatedRowSource	Возвращает или задает способ применения результатов команды DataRow при использовании метода Update объекта DbDataAdapter.

Методи класу DBCommand

	Описание
Cancel()	Пытается отменить выполнение DbCommand.
CreateDbParameter()	Создает новый экземпляр объекта DbParameter.
CreateParameter()	Создает новый экземпляр объекта DbParameter.
Dispose()	Освобождает все ресурсы, занятые модулем Component.(Наследуется от Component.)
Equals(Object)	Определяет, равен ли заданный объект текущему объекту.(Наследуется от Object.)
Prepare()	Создает подготовленную (скомпилированную) версию команды в источнике данных.

ExecuteNonQuery()	Выполняет инструкцию SQL для объекта соединения.
ExecuteNonQueryAsync()	Это асинхронная версия ExecuteNonQuery. Поставщикам следует переопределять соответствующую реализацию. При необходимости можно проигнорировать токен отмены.
ExecuteReader()	Выполняет CommandText от Connection, и возвращает DbDataReader.
ExecuteReaderAsync()	Асинхронная версия ExecuteReader, который выполняет CommandText от Connection и возвращает DbDataReader. Вызывает ExecuteDbDataReaderAsync с CancellationToken.None.
ExecuteScalar()	Выполняет запрос и возвращает первый столбец первой строки в наборе результатов, возвращаемых запросом. Все столбцы и строки игнорируются.
ExecuteScalarAsync()	Асинхронная версия ExecuteScalar, который выполняет запрос и возвращает первый столбец первой строки в наборе результатов, возвращаемых запросом. Все столбцы и строки игнорируются. Вызывает ExecuteScalarAsync с CancellationToken.None.

```
SqlConnection _connect = new
SqlConnection(_connectString);
_connect.Open();

SqlCommand _createQuery = new SqlCommand(
          "Create database test",_connect);
_createQuery.ExecuteNonQuery();

_connect.ChangeDatabase("test");

_createQuery.CommandText = "create table Tab2(" +
          "pole1 int not  null primary key, " +
                    "pole2 varchar(20))";
_createQuery.ExecuteNonQuery();

_connect.ChangeDatabase("torg_firm");
```

```
SqlCommand _selectQuery = new SqlCommand(
          "Insert into tovar " +
                    "values('AAA', 20, 10, 1 )",
_connect);
_selectQuery.ExecuteNonQuery();

_selectQuery.CommandText = "select count(*) from tovar";
int _counTovarRow=(int)_selectQuery.ExecuteScalar();
Int32.TryParse(_selectQuery.ExecuteScalar().ToString(),
out int _counTovarRow1);

_createQuery.CommandText = "CREATE PROC Proc1 AS UPDATE
tovar SET price = price * 1.2";
                try {
                    _createQuery.ExecuteNonQuery();
                }
                catch { }
```

```
_selectQuery.CommandType =
System.Data.CommandType.StoredProcedure;

_selectQuery.CommandText = "Proc1";
_selectQuery.ExecuteNonQuery();

_connect.Close();
_connect.Dispose();
```

```
myCommand.CommandText = "SELECT COUNT (*) FROM Туры";
string KolichestvoTurov = Convert.ToString(myCommand.ExecuteScalar());

myCommand.CommandText = "SELECT MAX (Цена) FROM Туры";

string MaxPrice = Convert.ToString(myCommand.ExecuteScalar());
myCommand.CommandText = "SELECT MIN (Цена) FROM Туры";

string MinPrice = Convert.ToString(myCommand.ExecuteScalar());
myCommand.CommandText = "SELECT AVG (Цена) FROM Туры";

string AvgPrice = Convert.ToString(myCommand.ExecuteScalar());
conn.Close();

Console.WriteLine("Количество туров: " + KolichestvoTurov + "\nСамый
дорогой тур, цена в руб. : " + MaxPrice + “ \nСамый дешевый тур, цена в
руб.: " +MinPrice +"\nСредняя цена туров: " + AvgPrice);
```

```
string connectionString = @"Data Source=.\ Serv_name; AttachDbFilename="+
                        @" Pass\file.mdf”+
      ";Integrated Security=True;Connect Timeout=30;User Instance=True";

string commandText = "SELECT * FROM Tab1";

SqlConnection conn = new SqlConnection(connectionString);

conn.Open();

SqlCommand myCommand = conn.CreateCommand();

myCommand.CommandText = "SELECT * FROM Tab1";
SqlDataReader dataReader = myCommand.ExecuteReader();
while (dataReader.Read())
{
      Console.WriteLine(dataReader["Pole1"]);
}

dataReader.Close();
conn.Close();
```

DbDataReader

Считывает однопроходные потоки строк из источника данных.
Пространство имен: System.Data.Common
Сборка: System.Data (в System.Data.dll)

Иерархия наследования
System.Object
  System.MarshalByRefObject
    System.Data.Common.DbDataReader
      System.Data.DataTableReader
      System.Data.EntityClient.EntityDataReader
      System.Data.Odbc.OdbcDataReader
      System.Data.OleDb.OleDbDataReader
      System.Data.OracleClient.OracleDataReader
      System.Data.SqlClient.SqlDataReader

Властивості DbDataReader

Имя	Описание
Depth	Возвращает значение, показывающее глубину вложенности для текущей строки.
FieldCount	Возвращает число столбцов в текущей строке.
HasRows	Возвращает значение, указывающее на то, что в модуле чтения данных DbDataReadersодержится одна или несколько строк.
IsClosed	Возвращает значение, указывающее, закрыт ли модуль чтения DbDataReader.
Item[Int32]	Возвращает значение указанного столбца как экземпляр Object.
Item[String]	Возвращает значение указанного столбца как экземпляр Object.
RecordsAffected	Возвращает число строк, которые были изменены, вставлены или удалены инструкцией SQL.
VisibleFieldCount	Возвращает число не скрытых полей в DbDataReader.

Методи DbDataReader

Описание	
Close()	Закрывает объект DbDataReader.
Dispose()	Освобождает все ресурсы, используемые текущим экземпляром класса DbDataReader.
IsDBNull(Int32)	Возвращает значение, указывающее, содержатся ли в столбце несуществующие или отсутствующие значения.
IsDBNullAsync(Int32)	Асинхронная версия IsDBNull, которая возвращает значение, которое указывает, содержит ли столбец несуществующие или отсутствующие значения.
NextResult()	Перемещает модуль чтения данных к следующему результату при чтении результатов из пакета инструкций.
NextResultAsync()	Асинхронная версия NextResult, который перемещает средство чтения к следующему результату при считывании результатов пакета инструкций. Вызывает NextResultAsync с CancellationToken.None.
Read()	Перемещает модуль чтения к следующей записи в результирующем наборе.
ReadAsync()	Асинхронная версия Read, который перемещает модуль чтения к следующей записи в результирующем наборе. Этот метод вызывает ReadAsync с CancellationToken.None.
ToString()	Возвращает строку, представляющую текущий объект.(Наследуется от Object.)

GetBoolean(Int32)	Возвращает значение указанного столбца в виде логического значения.
GetByte(Int32)	Возвращает значение указанного столбца в виде байта.
GetBytes(Int32, Int64, Byte[], Int32, Int32)	Считывает поток байтов из указанного столбца.
GetChar(Int32)	Возвращает значение указанного столбца в виде одного символа.
GetChars(Int32, Int64, Char[], Int32, Int32)	Считывает поток символов из заданного столбца.
GetData(Int32)	Возвращает DbDataReader объекта для запрошенного порядкового номера столбца.
GetDataTypeName(Int32)	Возвращает имя типа данных заданного столбца.
GetDateTime(Int32)	Возвращает значение указанного столбца в виде объекта DateTime.
GetDbDataReader(Int32)	Возвращает объект DbDataReader для запрошенного порядкового номера столбца, который можно переопределить реализацией, зависящей от поставщика.
GetDecimal(Int32)	Возвращает значение указанного столбца в виде объекта Decimal.
GetDouble(Int32)	Возвращает значение заданного столбца в виде числа с плавающей запятой двойной точности.
GetEnumerator()	Возвращает перечислитель IEnumerator, который может использоваться для просмотра строк в модуле чтения данных.
GetFieldType(Int32)	Возвращает тип данных заданного столбца.
GetFloat(Int32)	Возвращает значение заданного столбца в виде числа с плавающей запятой одинарной точности.
GetGuid(Int32)	Возвращает значение заданного столбца в виде глобально-уникального идентификатора GUID.
GetInt16(Int32)	Возвращает значение указанного столбца в виде 16-разрядное целое число со знаком.
GetInt32(Int32)	Возвращает значение указанного столбца в виде 32-разрядное целое число со знаком.
GetInt64(Int32)	Возвращает значение указанного столбца в виде 64-разрядного целого числа со знаком.
GetName(Int32)	Возвращает имя столбца, если известен его порядковый номер (от нуля).
GetOrdinal(String)	Возвращает порядковый номер столбца, если известно его имя.
GetStream(Int32)	Получает данные в виде Stream.
GetString(Int32)	Возвращает значение указанного столбца как экземпляр String.
GetTextReader(Int32)	Получает данные в виде TextReader.

```
string _connectString;
 SqlConnection _connect;

_connect = new SqlConnection(_connectString);
_connect.Open();

SqlCommand _showData = new SqlCommand();
_showData.CommandType = System.Data.CommandType.Text;
_showData.CommandText = "Select * from "+_tableName;
showData.Connection = this._connect;
SqlDataReader _reader = _showData.ExecuteReader();
if (_reader.HasRows){
   while (_reader.Read()) {
         for (int i = 0; i < _reader.FieldCount; i++)
              Console.Write("{0}\t", _reader[i]);
         }
         Console.WriteLine("");
   }
connect.Close();
connect.Dispose();
```

```
DataTable dt = new DataTable();
dt.Load(_SelectCommand.ExecuteReader());
dataGridView1.DataSource = dt;
```

```
//Выводим данные в элемент listBox1
listBox1 .Items.Add("Pole_name1: " + Pole1  …);
while (dataReader.Read()){
    ListViewItem item = new ListViewItem(new string[]
        {
            Convert.ToString(dataReader[0]),
            Convert.ToString(dataReader[1]),
            Convert.ToString(dataReader[2]),
            Convert.ToString(dataReader[3])}
            );
        listView1.Items.Add(item);
        }
    }
```

```
conn = new SqlConnection();
conn.ConnectionString = @“/*write smth…*/";
conn.Open();
SqlCommand myCommand = conn.CreateCommand();
myCommand.CommandText = "SELECT * FROM Tab1";

dataReader = myCommand.ExecuteReader();
while (dataReader.Read()) {
	int Pole1 = dataReader.GetInt32(0);
	string Pole2= dataReader.GetString(1);
	string Pole3 = dataReader.GetString(2);
	string Pole4 = dataReader.GetString(3);
}
```

```
Console.WriteLine(dataReader[1]);
```

```
Console.WriteLine(dataReader[0]);
Console.WriteLine(dataReader[1]);
Console.WriteLine(dataReader[2]);
Console.WriteLine(dataReader[3]);
```

```
Console.WriteLine(Convert.ToString(dataReader[0]) + " “
      + Convert.ToString(dataReader[1]) + " "
      + Convert.ToString(dataReader[2]) +" "
      + Convert.ToString(dataReader[3]));
```