**Как подключить и использовать PostgreSQL в C#**

18 января

782 прочитали



Постгре

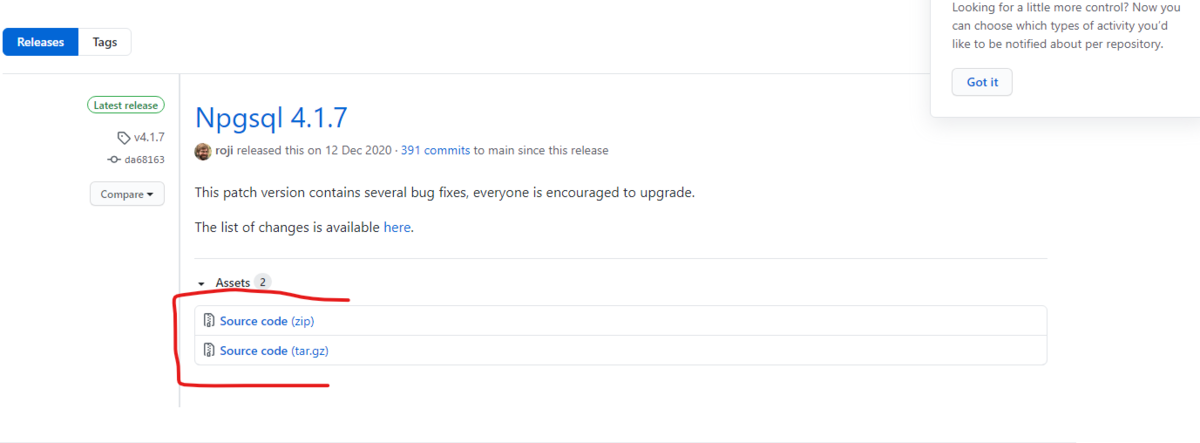
Подкл**ючение**

Для подключения PostgreSQL к C# требуется усатновить Npgsql.dll, для чего возможно два пути:

1. Установка при помощи NuGetPkgManager. Скачивается и устанавливается пакет Npgsql
2. Самостоятельная сборка Npgsql.dll. И мы воспользуемся именно этим методом

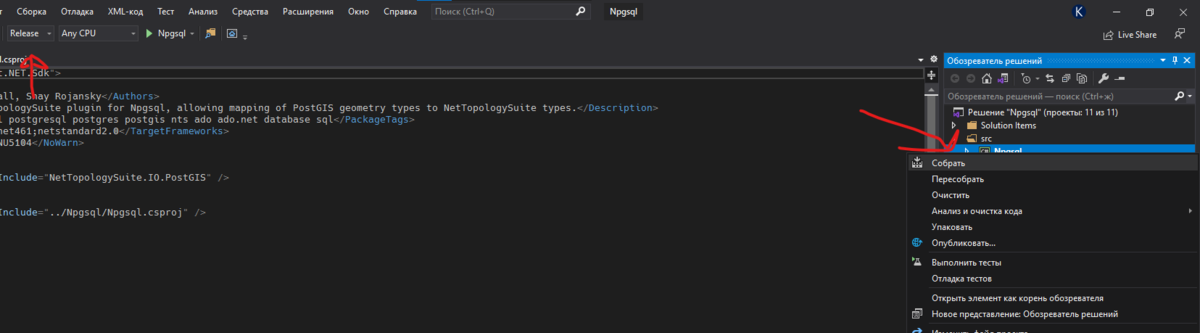
Для чего необходимо:

Скачать с сайта Github реппозиторий: [Npgsql](https://github.com/npgsql/npgsql" \t "_blank)



Как качать

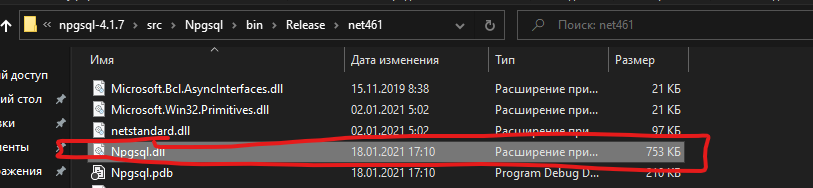
*Распаковать, открыть через Visual Studio и произвести сборку проекта Npgsql из папки src, установив сверху не Debug, а Release*



Сборка

В результате описанных выше процедур в папке: *npgsql-4.1.7\src\Npgsql\bin\Release\net461\*

окажется файл *Npgsql.dll*



Собранная либа

После чего этот файл необходимо скопировать в директорию с исходными кодами проекта, в котором будет использоваться соединение с PostgreSQL и в Обозревателе решений Visual Studio нажать на ссылки->добавить ссылку в открывшемся окне нажимем Обзор и выбираем нашу Npgsql.dll  
Подключаем в требуемом месте заголовок using Npgsql;  
Довольствуемся результатом

**Использование**

**Подключение к базе данных**

Чтобы начать использовать Npgsql нам понадобится сервер Postgre(Очевидно) и некоторая информация о подключении к серверу.  
Подключение к серверу формируется при помощи так называемых ConnectionString. Для PostgreSQL имеющих следующий формат:  
string connString = "Host=myserver;Username=mylogin;Password=mypass;Database=mydatabase";  
Вместо myserver необходимо указать адрес нашего сервера, например: localhost, 127.0.0.1, 192.168.0.12 и т.п. Вместо mylogin указываем наш логин.  
Вместо mydatabase указываем нашу базу данных  
Далее необходимо подсоединится к базе данных при помощи следующей команды:

*NpgsqlConnection nc = new NpgsqlConnection(connString);*  
*try{*  
*//Открываем соединение.*  
*nc.Open();*  
*}*  
*catch(Exception ex){*  
*//Код обработки ошибок*  
*}*

Команда открытия соединения обёрнута в try-catch блок не просто так, а потому что в npgsql метод Open() не возвращает ничего об открытии соединения, поэтому остаётся надеяться на самые обычные Exception'ы. Однако есть выход, проверить состояние поля FullState{get;}, содержащего перечисление ConnectionState, и если у нас состояние Broken или Closed, то соединение не открылось и надо что-то менять в своей жизни.  
Пример кода:

NpgsqlConnection nc = new NpgsqlConnection(connString);  
nc.Open();  
if(nc.FullState == ConnectionState.Broken || nc.FullState == Connection.Closed){  
//Тут меняем что-то в своей жизни, но я обычно выбрасываю исключение, чтобы не искать в коде миллион лет, что сломалось.  
}

**Выполнение запросов к базе данных**

Для выполнения команд используется объект класса NpgsqlCommand следующим образом:

NpgsqlCommand npgc = new NpgsqlCommand("ЗАПРОС", nc);  
int rows\_changed=npgc.ExecuteNonQuery();//Если запрос не возвращает таблицу  
NpgsqlDataReader ndr = npgc.ExecuteReader()//Если запрос возвращает таблицу

Далее необходимо обработать полученные результаты следующим образом:

NpgsqlDataReader reader = command.ExecuteReader();  
if (reader.HasRows)//Если пришли результаты  
{  
while (reader.Read())//Пока есть записи  
{for (int i = 0; i < width; i++)  
{  
if (reader.GetFieldType(i).ToString() == "System.Int32")//Проверяем тип следующей записи  
{  
ResultAsStringArray[i, tmp] = reader.GetInt32(i).ToString();//Получаем запись и переводим её в строку  
}  
}  
}  
}

Данный пример показан лишь для демонстрации, весь остальной код - пишите сами.