

Dapper

.NetFramework 下使用Dapper

安装Dapper包(例子用的是.netFrame4.5,所以安装对应的1.5版本)

```
Install-Package Dapper -Version 1.50.0
```

创建 `SQLHelper` 类并提供一个静态方法返回 `DbConnection` 对象

```
using System;
using System.Collections.Generic;
using System.Data.Common;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Dapper4NetFramework
{
    public class SQLHelper
    {
        private static readonly string providerName =
            System.Configuration.ConfigurationManager.ConnectionStrings["connectStr"].ProviderName;

        private static readonly string connectStr =
            System.Configuration.ConfigurationManager.ConnectionStrings["connectStr"].ConnectionString;

        private static readonly DbProviderFactory dbFactory =
            DbProviderFactories.GetFactory(providerName);

        public static DbConnection GetDbConnection()
        {
            DbConnection dbConnection = dbFactory.CreateConnection();
            dbConnection.ConnectionString = connectStr;
            return dbConnection;
        }
    }
}
```

连接SQLserver

- 引用 `System.Configuration.dll`

配置 `App.config`

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <connectionStrings>
    <add name="connectStr" providerName="System.Data.SqlClient"
connectionString="Data Source=192.168.182.128;Initial Catalog=BookStore;User
ID=sa;pwd=*****;Min Pool Size=1"/>
  </connectionStrings>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />
  </startup>
</configuration>
```

操作数据库

引用命名Dapper空间

```
using Dapper;
```

查询语句

定义一个拥有与 sql 查询语句字段名相同的名属性的类

```
public class book
{
    /// <summary>
    ///
    /// </summary>
    public string Id { get; set; }
    /// <summary>
    ///
    /// </summary>
    public string Title { get; set; }
    /// <summary>
    ///
    /// </summary>
    public string PubTime { get; set; }
    /// <summary>
    ///
    /// </summary>
    public string Price { get; set; }
    /// <summary>
    ///
    /// </summary>
    public string Page { get; set; }
}
```

使用 DbConnection 的扩展方法 QueryAsync (异步方法)或 Query 方法执行sql查询，会返回与之泛型匹配的实例

```
using Dapper;
```

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Dapper4NetFramework
{
    class Program
    {
        static async Task Main(string[] args)
        {
            using (System.Data.Common.DbConnection dbConnection =
SQLHelper.GetDbConnection())
            {
                IEnumerable<book> enumerable = await
dbConnection.QueryAsync<book>(@"SELECT Id,

                Title,

                PubTime,

                Price,

                Page,

                publisher

                FROM

                T_Books");

                foreach (var item in enumerable)
                {
                    Console.WriteLine($"[Id]:{item.Id}\r\n[Title]:
{item.Title}\r\n[PubTime]:{item.PubTime}\r\n[Price]:{item.Price}\r\n[Page]:
{item.Page}\r\n[publisher]:{item.PubTime}\r\n");
                }
                Console.ReadLine();
            }
        }
    }
}

```

执行结果

C:\Users\he_zhw\source\repos\Dapper4NetFramework\bin\Debug\Dapper4NetFramework.exe

```

[Id]:1
[Title]:444
[PubTime]:2021-10-25 21:40:11
[Price]:5
[Page]:444
[publisher]:2021-10-25 21:40:11

[Id]:2
[Title]:sda
[PubTime]:2021-10-25 21:45:26
[Price]:6
[Page]:66
[publisher]:2021-10-25 21:45:26

```

连接Mysql

安装Mysql 驱动包(注意看包的依赖项是否符合当前 Framework 版本)

```
Install-Package MySql.Data -Version 6.7.9.0
```

配置App.config

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <connectionStrings>
    <!--MySQL连接串配置-->
    <add name="connectStr" providerName="MySql.Data.MySqlClient"
connectionString="Data Source=192.168.182.128;Initial Catalog=bookstores;User
ID=root;pwd=root;Min Pool Size=1;"></add>
  </connectionStrings>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />
  </startup>
  <system.data>
    <DbProviderFactories>
      <remove invariant="MySql.Data.MySqlClient" />
      <add name="MySQL Data Provider" invariant="MySql.Data.MySqlClient"
description=".Net Framework Data Provider for MySQL"
type="MySql.Data.MySqlClient.MySqlClientFactory, MySql.Data, Version=6.7.9.0,
Culture=neutral, PublicKeyToken=c5687fc88969c44d" />
    </DbProviderFactories>
  </system.data>
</configuration>
```

其他用法同 SQLserver

连接oracle(使用Oracle.ManagedDataAccess.dll)

安装驱动(注意看包的依赖项是否符合当前 Framework 版本)

```
Install-Package Oracle.ManagedDataAccess -Version 19.13.0
```

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <configSections>
    <section name="oracle.manageddataaccess.client"
type="OracleInternal.Common.ODPMSectionHandler, Oracle.ManagedDataAccess,
Version=4.122.19.1, Culture=neutral, PublicKeyToken=89b483f429c47342" />
  </configSections>
  <connectionStrings>
    <!--Oracle连接串配置-->
    <add name="connectStr" providerName="Oracle.ManagedDataAccess.Client"
connectionString="User Id=BookStores;Password=oracle;Data Source=(DESCRIPTION=
(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=192.168.182.128)(PORT=1521)))
(CONNECT_DATA=(SERVICE_NAME=orc1)));Min Pool Size=1"></add>
  </connectionStrings>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />
  </startup>
</configuration>
```

```

</startup>
<system.data>
  <DbProviderFactories>
    <remove invariant="Oracle.ManagedDataAccess.Client" />
    <add name="ODP.NET, Managed Driver"
invariant="Oracle.ManagedDataAccess.Client" description="Oracle Data Provider
for .NET, Managed Driver"
type="Oracle.ManagedDataAccess.Client.OracleClientFactory,
Oracle.ManagedDataAccess, Version=4.122.19.1, Culture=neutral,
PublicKeyToken=89b483f429c47342" />
  </DbProviderFactories>
</system.data>
<runtime>
  <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
    <dependentAssembly>
      <publisherPolicy apply="no" />
      <assemblyIdentity name="Oracle.ManagedDataAccess"
publicKeyToken="89b483f429c47342" culture="neutral" />
      <bindingRedirect oldVersion="4.121.0.0 - 4.65535.65535.65535"
newVersion="4.122.19.1" />
    </dependentAssembly>
  </assemblyBinding>
</runtime>
<oracle.manageddataaccess.client>
  <version number="*">
    <dataSources>
      <dataSource alias="SampleDataSource" descriptor="(DESCRIPTION=
(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=192.168.182.128)(PORT=1521)))
(CONNECT_DATA=(SERVICE_NAME=orcl)))" />
    </dataSources>
    <!--<settings>
      <setting name="TNS_ADMIN"
value="D:\app\he_zhw\product\11.2.0\client_1\network\admin"/>
    </settings-->
  </version>
</oracle.manageddataaccess.client>
</configuration>

```

oracle.manageddataaccess.client 节点介绍

- dataSource 可以作为连接串中数据库源的别名，及连接串中 Data Source 的值可替换成alias对应的值

```

<!--Oracle连接串配置-->
  <add name="connectStr" providerName="Oracle.ManagedDataAccess.Client"
connectionString="User Id=BookStores;Password=oracle;Data
Source=SampleDataSource;Min Pool Size=1"></add>

```


- 若要使用oracle客户端托管,则将 settings 节点解开注释
 - TNS_ADMIN 的value值就是oracle客户端安装位置中 TNS 文件 tnsnames.ora 所在的文件
 - 然后连接串中的就可以使用 TNS 文件中配置的别名 ORCLNew

```
<!--Oracle连接串配置-->
    <add name="connectStr"
providerName="Oracle.ManagedDataAccess.Client" connectionString="User
Id=BookStores;Password=oracle;Data Source=ORCLNew;Min Pool Size=1">
</add>
```

*tnsnames.ora - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
# (SERVER = DEDICATED)
# (SERVICE_NAME = test)
# )
# )
#
# You can modify the entry below for your own database.
# <data source alias> = Name to use in the connection string Data Source
# <hostname or IP> = name or IP of the database server machine
# <port> = database server machine port to use
# <database service name> = name of the database service on the server
```



```
ORCLNew =
(DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.182.128)(PORT = 1521))
  (CONNECT_DATA =
    (SERVICE_NAME = orcl)
  )
)
```

.net6下使用Dapper

创建 json 配置文件配置数据库连接

用到那个数据库就解开注释,并注释调其他连接配置

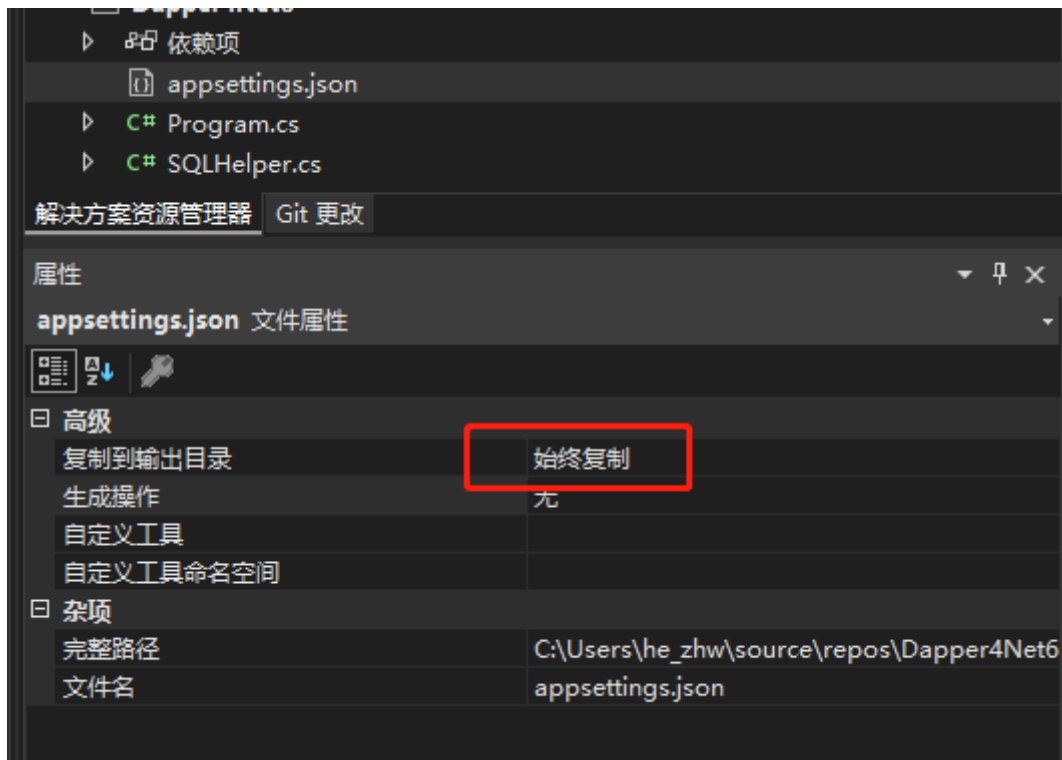
```
{
  ///oracle
  "ConnectionStrings": {
    "DbType": "Oracle",
    "DbProvider": "Oracle.ManagedDataAccess.Client.OracleClientFactory,
Oracle.ManagedDataAccess",
    "ConnectionString": "User Id=BookStores;Password=oracle;Data
Source=ORCLNew;Min Pool Size=1",
    "TNS_ADMIN": "D:\\app\\he_zhw\\product\\11.2.0\\client_1\\network\\admin"
  }

  //sqlserver
  //"ConnectionStrings": {
  //  "DbType": "SQLserver",
  //  "DbProvider": "System.Data.SqlClient.SqlClientFactory, System.Data",
```

```
// "ConnectionString": "Data Source=192.168.182.128;Initial
Catalog=BookStore;User ID=sa;pwd=WSZ@19950319CSD;Min Pool Size=1"
//}

//mysql
//"ConnectionStrings": {
//"DbType": "MySQL",
// "DbProvider": "MySql.Data.MySqlClient.MySqlClientFactory, MySql.Data",
// "ConnectionString": "Data Source=192.168.182.128;Initial
Catalog=bookstores;User ID=root;pwd=root;Min Pool Size=1;;SslMode = none"
//}
}
```

注:配置文件最好是设置成始终复制到输出目录



安装配置文件相关的扩展包

Configuration 扩展包

```
Install-Package Microsoft.Extensions.Configuration
```

Json 配置扩展包

```
Install-Package Microsoft.Extensions.Configuration.Json
```

创建 SQLHelper 类

```
using Microsoft.Extensions.Configuration;
using System;
using System.Collections.Generic;
using System.Data.Common;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```

namespace Dapper4Net6
{
    public class SQLHelper
    {
        private static object obj=new object();

        private static string? providerName = null;

        private static string? connectStr = null;

        private static DbProviderFactory? dbFactory = null;
        static void InIt()
        {
            IConfigurationBuilder configurationBuilder = new
ConfigurationBuilder();
            configurationBuilder.AddJsonFile("appsettings.json");
            IConfigurationRoot configurationRoot = configurationBuilder.Build();
            connectStr =
configurationRoot["ConnectionStrings:ConnectionString"];
            providerName = configurationRoot["ConnectionStrings:DbProvider"];
            string dbType = configurationRoot["ConnectionStrings:DbType"];
            #region Oracle配置客户端连接
            if (dbType == "Oracle")
            {
                Oracle.ManagedDataAccess.Client.OracleConfiguration.TnsAdmin =
configurationRoot["ConnectionStrings:TNS_ADMIN"];
            }
            #endregion
            System.Data.Common.DbProviderFactories.RegisterFactory("DbProvider",
providerName);
            dbFactory =
System.Data.Common.DbProviderFactories.GetFactory("DbProvider");
        }
        public static DbConnection GetDbConnection()
        {
            if (dbFactory == null || providerName == null|| providerName==null)
            {
                lock (obj)//加锁防止多线程调用时重复调用InIt方法
                {
                    if (dbFactory == null || providerName == null ||
providerName == null)
                    {
                        InIt();
                    }
                }
            }
            DbConnection dbConnection = dbFactory.CreateConnection();
            dbConnection.ConnectionString = connectStr;
            return dbConnection;
        }
    }
}

```


安装Dapper包

```
Install-Package Dapper
```

连接SQLserver

安装 System.Data.SqlClient 包

```
Install-Package System.Data.SqlClient
```

连接MySQL

安装 MySQL 包

```
Install-Package MySql.Data
```

连接Oracle

安装oracle驱动包

```
Install-Package Oracle.ManagedDataAccess.Core
```

```
// See https://aka.ms/new-console-template for more information
using Dapper;
using Dapper4Net6;

using (System.Data.Common.DbConnection dbConnection =
    SQLHelper.GetDbConnection())
{
    IEnumerable<dynamic> enumerable = await dbConnection.QueryAsync("SELECT *
FROM [dbo].[T_Books]");
    foreach (var item in enumerable)
    {
        Console.WriteLine($"[Id]:{item.Id}\r\n[Title]:{item.Title}\r\n[PubTime]:
{item.PubTime}\r\n[Price]:{item.Price}\r\n[Page]:{item.Page}\r\n[publisher]:
{item.PubTime}\r\n");
    }
}

Console.ReadKey();
```

运行结果

```
[Id]:1  
[Title]:444  
[PubTime]:2021-10-25 21:40:11  
[Price]:5  
[Page]:444  
[publisher]:2021-10-25 21:40:11  
  
[Id]:2  
[Title]:sda  
[PubTime]:2021-10-25 21:45:26  
[Price]:6  
[Page]:66  
[publisher]:2021-10-25 21:45:26
```