1. 访问数据库
2. MS SQLServer

**public** **class** MSSQLHelper

**{**

**public** string mConnString **=** ""**;**

**public** MSSQLHelper**(**String pConnString**)**

**{**

mConnString **=** pConnString**;**

**}**

#region 执行简单更新、删除语句

// 执行SQL语句，返回影响的记录数

**public** **int** ExecuteSql**(**string SQLString**,** params SqlParameter**[]** cmdParms**)**

**{**

using **(**SqlConnection conn **=** **new** SqlConnection**(**mConnString**))**

**{**

conn**.**Open**();**

SqlCommand cmd **=** **new** SqlCommand**();**

cmd**.**Connection **=** conn**;**

cmd**.**CommandText **=** SQLString**;**

cmd**.**CommandType **=** CommandType**.**Text**;**

**if** **(**cmdParms **!=** **null)**

**{**

foreach **(**SqlParameter lParm **in** cmdParms**)**

cmd**.**Parameters**.**Add**(**lParm**);**

**}**

**return** cmd**.**ExecuteNonQuery**();**

**}**

**}**

#endregion

#region 执行简单查询语句

// 执行查询语句，返回DataSet

**public** DataSet Query**(**string SQLString**,** params SqlParameter**[]** cmdParms**)**

**{**

using **(**SqlConnection conn **=** **new** SqlConnection**(**mConnString**))**

**{**

conn**.**Open**();**

SqlCommand cmd **=** **new** SqlCommand**();**

cmd**.**Connection **=** conn**;**

cmd**.**CommandText **=** SQLString**;**

cmd**.**CommandType **=** CommandType**.**Text**;**

SqlDataAdapter da **=** **new** SqlDataAdapter**(**cmd**);**

DataSet ds **=** **new** DataSet**();**

da**.**Fill**(**ds**,** "ds"**);**

**return** ds**;**

**}**

**}**

#endregion

#region 执行复杂数据库事务

/// <summary>

/// 有序执行

/// </summary>

/// <param name="SQLStringList">多条SQL</param>

**public** **void** ExecuteSqlTran\_sort**(**Dictionary**<**string**,** object**>** SQLStringList**)**

**{**

using **(**SqlConnection conn **=** **new** SqlConnection**(**mConnString**))**

**{**

conn**.**Open**();**

SqlTransaction trans **=** conn**.**BeginTransaction**();**

SqlCommand cmd **=** **new** SqlCommand**();**

**try**

**{**

foreach **(**KeyValuePair**<**string**,** object**>** item **in** SQLStringList**)**

**{**

cmd**.**Connection **=** conn**;**

cmd**.**CommandText **=** item**.**Key**.**ToString**();** **;**

cmd**.**CommandType **=** CommandType**.**Text**;**

SqlParameter**[]** cmdParms **=** **(**SqlParameter**[])**item**.**Value**;**

**if** **(**cmdParms **!=** **null)**

**{**

foreach **(**SqlParameter lParm **in** cmdParms**)**

cmd**.**Parameters**.**Add**(**lParm**);**

**}**

cmd**.**ExecuteNonQuery**();**

cmd**.**Parameters**.**Clear**();**

**}**

trans**.**Commit**();**

**}**

**catch** **(**Exception ex**)**

**{**

trans**.**Rollback**();**

**throw** ex**;**

**}**

**}**

**}**

#endregion

**}**

1. 通用数据库帮助类

**public** class DBHelper

**{**

**private** static MSSQLHelper mMSSQLHelper **=** **new** MSSQLHelper**();**

**public** static EXESQLRET GetDataTable**(**string sql**,** **ref** DataTable dt**)**

**{**

**try**

**{**

DataSet ds **=** **new** DataSet**();**

ds **=** mMSSQLHelper**.**Query**(**sql**);**

**if** **(**ds **!=** **null** **&&** ds**.**Tables**.**Count **>** 0 **&&** ds**.**Tables**[**0**].**Rows**.**Count **>** 0**)**

**{**

// 存在数据

dt **=** ds**.**Tables**[**0**];**

**return** EXESQLRET**.**SUCCESS**;**

**}**

**else**

**{**

// 不存在数据

**return** EXESQLRET**.**NODATA**;**

**}**

**}**

**catch** **(**Exception ex**)**

**{**

**return** EXESQLRET**.**ERROR**;**

**}**

**}**

**public** static EXESQLRET ExecuteSQL**(**string sql**)**

**{**

**try**

**{**

int ret **=** mMSSQLHelper**.**ExecuteSql**(**sql**);**

**if** **(**ret **>** 0**)**

**{**

**return** EXESQLRET**.**SUCCESS**;**

**}**

**else** **if** **(**ret **==** 0**)**

**{**

**return** EXESQLRET**.**NODATA**;**

**}**

**else**

**{**

**return** EXESQLRET**.**ERROR**;**

**}**

**}**

**catch** **(**Exception ex**)**

**{**

**return** EXESQLRET**.**ERROR**;**

**}**

**}**

**}**

1. Sql执行结果枚举

**public** enum EXESQLRET

**{**

SUCCESS **=** 1

**,**

NODATA **=** 2

**,**

ERROR **=** 0

**}**

1. 加密解密
2. DES

**public** **class** DES

**{**

//注意了，是8个字符，64位

**const** string KEY\_64 **=** "NTCHHLAQ"**;**

**const** string IV\_64 **=** "NTCH2018"**;**

/// <summary>

/// 64位加密

/// </summary>

/// <param name="data"></param>

/// <returns></returns>

**static** **public** string Encode**(**string data**)**

**{**

**byte[]** byKey **=** System**.**Text**.**ASCIIEncoding**.**ASCII**.**GetBytes**(**KEY\_64**);**

**byte[]** byIV **=** System**.**Text**.**ASCIIEncoding**.**ASCII**.**GetBytes**(**IV\_64**);**

DESCryptoServiceProvider cryptoProvider **=** **new** DESCryptoServiceProvider**();**

**int** i **=** cryptoProvider**.**KeySize**;**

MemoryStream ms **=** **new** MemoryStream**();**

CryptoStream cst **=** **new** CryptoStream**(**ms**,** cryptoProvider**.**CreateEncryptor**(**byKey**,** byIV**),** CryptoStreamMode**.**Write**);**

StreamWriter sw **=** **new** StreamWriter**(**cst**);**

sw**.**Write**(**data**);**

sw**.**Flush**();**

cst**.**FlushFinalBlock**();**

sw**.**Flush**();**

**return** Convert**.**ToBase64String**(**ms**.**GetBuffer**(),** 0**,** **(int)**ms**.**Length**);**

**}**

/// <summary>

/// 64位解密

/// </summary>

/// <param name="data"></param>

/// <returns></returns>

**static** **public** string Decode**(**string data**)**

**{**

**byte[]** byKey **=** System**.**Text**.**ASCIIEncoding**.**ASCII**.**GetBytes**(**KEY\_64**);**

**byte[]** byIV **=** System**.**Text**.**ASCIIEncoding**.**ASCII**.**GetBytes**(**IV\_64**);**

**byte[]** byEnc**;**

**try**

**{**

byEnc **=** Convert**.**FromBase64String**(**data**);**

**}**

**catch**

**{**

**return** **null;**

**}**

DESCryptoServiceProvider cryptoProvider **=** **new** DESCryptoServiceProvider**();**

MemoryStream ms **=** **new** MemoryStream**(**byEnc**);**

CryptoStream cst **=** **new** CryptoStream**(**ms**,** cryptoProvider**.**CreateDecryptor**(**byKey**,** byIV**),** CryptoStreamMode**.**Read**);**

StreamReader sr **=** **new** StreamReader**(**cst**);**

**return** sr**.**ReadToEnd**();**

**}**

**}**

1. 前后台数据交互
2. 前台请求后台数据
3. Ajax-get

**function** AjaxGet**()** **{**

$**.**ajax**({**

"type" **:** "get"**,**

"url" **:** "/Car/HelloWorld"**,**

"success" **:** **function** **(**Data**,** StatusText**)** **{**

**alert(**"Data = " **+** Data**);**

**alert(**"StatusText = " **+** StatusText**);**

**},**

"error" **:** **function** **(**XMLHttpRequest**,** StatusText**)** **{**

**alert(**"XMLHttpRequest =" **+** XMLHttpRequest**);**

**alert(**"StatusText =" **+** StatusText**);**

**},**

"complete" **:** **function** **(**XMLHttpRequest**,** StatusText**)** **{**

**alert(**"XMLHttpRequest =" **+** XMLHttpRequest**);**

**alert(**"StatusText =" **+** StatusText**);**

**}**

**});**

**}**

**后台HelloWorld方法**

**public** class CarController **:** Controller

**{**

//

// GET: /Car/

**public** ActionResult Index**()**

**{**

**return** View**();**

**}**

**public** String HelloWorld**()**

**{**

**return** "{\"Return\":\"Hello World!\"}"**;**

**}**

**public** ActionResult Test**()**

**{**

**return** View**();**

**}**

**}**

1. DataTable转Json字符串
2. StringBuilder

**public** static String FromDataTable**(**DataTable pDT**)**

**{**

String lJsonStr **=** ""**;**

**if** **(**pDT **!=** **null** **&&** pDT**.**Rows**.**Count **>** 0**)**

**{**

lJsonStr **=** "["**;**

**foreach** **(**DataRow lDR **in** pDT**.**Rows**)**

**{**

lJsonStr **+=** "{"**;**

**foreach** **(**DataColumn lDC **in** pDT**.**Columns**)**

**{**

lJsonStr **+=** "\"" **+** lDC**.**ColumnName **+** "\":\"" **+** lDR**[**lDC**.**ColumnName**]** **+** "\","**;**

**}**

**if** **(**lJsonStr**.**EndsWith**(**","**))**

**{**

lJsonStr **=** lJsonStr**.**Substring**(**0**,** lJsonStr**.**Length **-** 1**);**

**}**

lJsonStr **+=** "},"**;**

**}**

**if** **(**lJsonStr**.**EndsWith**(**","**))**

**{**

lJsonStr **=** lJsonStr**.**Substring**(**0**,** lJsonStr**.**Length **-** 1**);**

**}**

lJsonStr **+=** "]"**;**

**}**

**return** lJsonStr**;**

**}**

1. 如何实现分页列表页面
2. 前台
3. 列表HTML

以【车辆列表】为例

<table class="table-bordered table-striped">

<tr>

<th style="width:100px;">序号</th>

<th style="width:200px;">车号</th>

</tr>

<tr>

<td>1</td>

<td>998963</td>

</tr>

<tr>

<td>2</td>

<td>98983</td>

</tr>

<tr>

<td>3</td>

<td>98967</td>

</tr>

<tr>

<td>4</td>

<td>98961</td>

</tr>

<tr>

<td>5</td>

<td>98957</td>

</tr>

<tr>

<td>6</td>

<td>8689</td>

</tr>

<tr>

<td>7</td>

<td>84595</td>

</tr>

<tr>

<td>8</td>

<td>8359</td>

</tr>

<tr>

<td>9</td>

<td>8356</td>

</tr>

<tr>

<td>10</td>

<td>8315</td>

</tr>

<tr>

<td>11</td>

<td>8135</td>

</tr>

</table>

1. 分页HTML

<nav aria-label="Page navigation example">

<ul class="pagination">

<li class="page-item"><a class="page-link" href="#">Previous</a></li>

<li class="page-item"><a class="page-link" href="#">1</a></li>

<li class="page-item"><a class="page-link" href="#">2</a></li>

<li class="page-item"><a class="page-link" href="#">3</a></li>

<li class="page-item"><a class="page-link" href="#">Next</a></li>

</ul>

</nav>

1. 数据库分页存储过程