

# 01 零基础1小时学编程，极简教程，简到奔溃dnc+vue

零基础1小时学编程，极简教程，简到奔溃

这个极简教程以电商平台系统作为演示，零基础1小时极速入门，挑战不可能的任务，马上开车，Are you Ready?

一、先安装相关软件，请看第1个文档安装步骤

[https://github.com/AskyEdu/Askv/tree/master/01\\_doc](https://github.com/AskyEdu/Askv/tree/master/01_doc)

1、Visual Studio 2017开发工具

2、MySql/MariaDB开源数据库

3、MySql/MariaDB开源管理工具HeidiSql

二、dnc + vue零基础1小时学编程，极简教程，简到奔溃

dnc是微软 新一代编程技术.NET Core / DotNet Core的缩写，

传统.NET / DotNet与新一代dnc的区别：[赛亚人与超级赛亚人的区别](#)

dnc是微软面向微服务、云原生应用打造的新一代编程平台，可部署到Linux、Docker，开发效率更高、性能更强、更轻量级

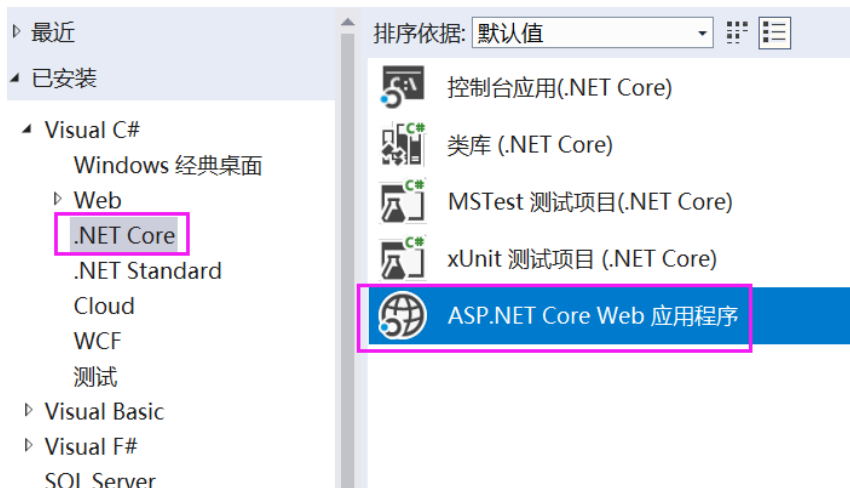
国内使用dnc的公司有微软、腾讯、网易、龙珠直播、同程旅游、新东方、博客园、途虎养车、NEO开源区块链、aelf开源区块链……等

vue是非常流行的开源javascript前端框架，前后端分离开发模式，后台工程师写dnc接口代码，前端工程师写vue代码和页面html、css代码

VS是Visual Studio的缩写，下面有些概念、技术术语，一开始不理解没有关系，先按教程走一遍，再看第2遍时自然就理解了，有些工具类库的固定用法，直接记住即可

1、VS2017 文件菜单 -> 新建.NET Core 2.0 【空项目】，项目命名为Demo2Core

新建项目





2、程序包管理器控制台，下拉列表选择默认项目为Demo2Core项目，安装Nuget包  
复制下面的命令到PM>后面

Install-Package Microsoft.AspNetCore.All

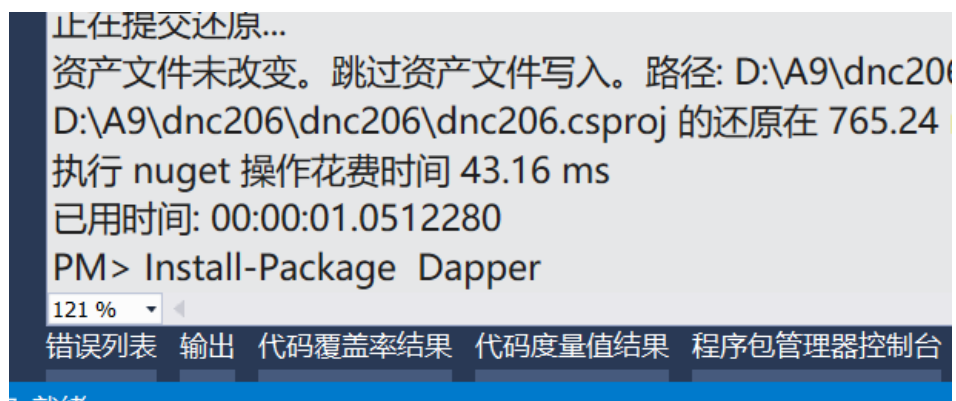
Install-Package AskyCore

Install-Package MySqlConnector

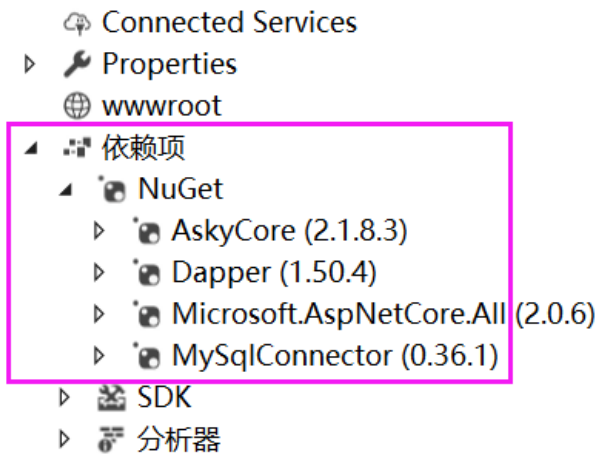
Install-Package Dapper



注意最后一行会停在Install-Package Dapper，必须再按一次回车安装



安装Nuget包之后，查看VS右侧依赖项，应该看到已安装了4个nuget包



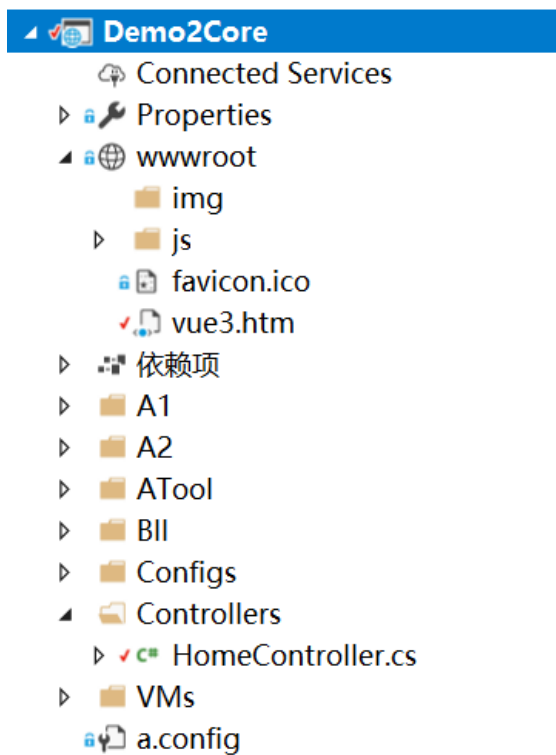
如果找不到【程序包管理器控制台】这个窗口，可从视图菜单 -> 其他窗口 -> 打开【程序包管理器控制台】



3、下载示例代码、体验dnc + vue

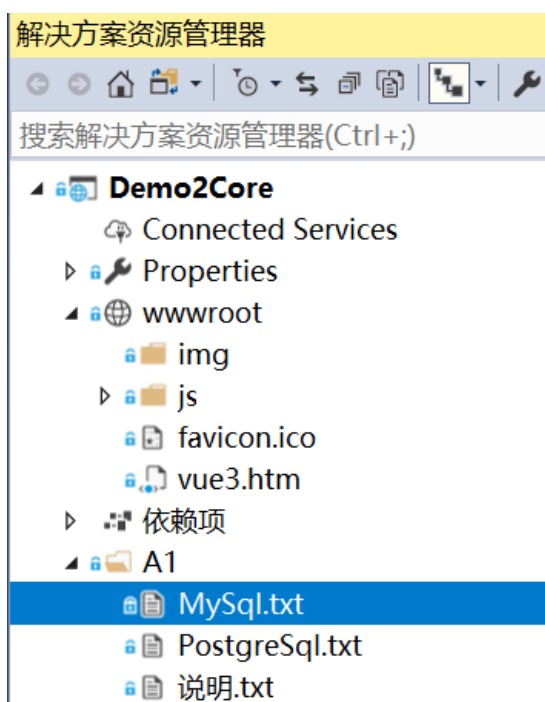
1) 下载示例代码 [https://github.com/AskvEdu/Askv/raw/master/02\\_demoCode/01\\_Demo2Core.zip](https://github.com/AskvEdu/Askv/raw/master/02_demoCode/01_Demo2Core.zip)

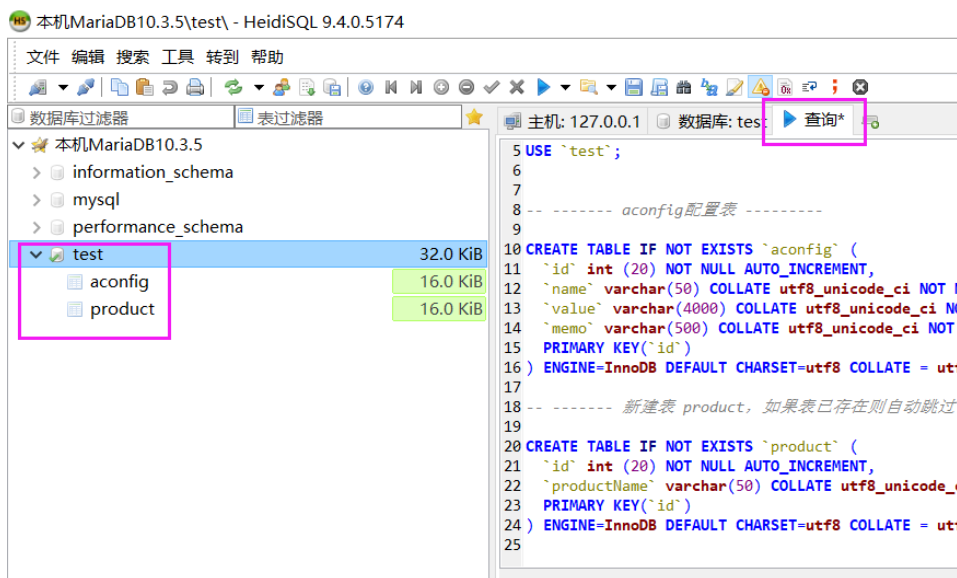
解压替换到本项目的根目录，用VS2017查看时，目录结构应该是这样：



2) 参考第1个软件安装文档，用HeidiSql工具连接本机MariaDB数据库，点击【查询】

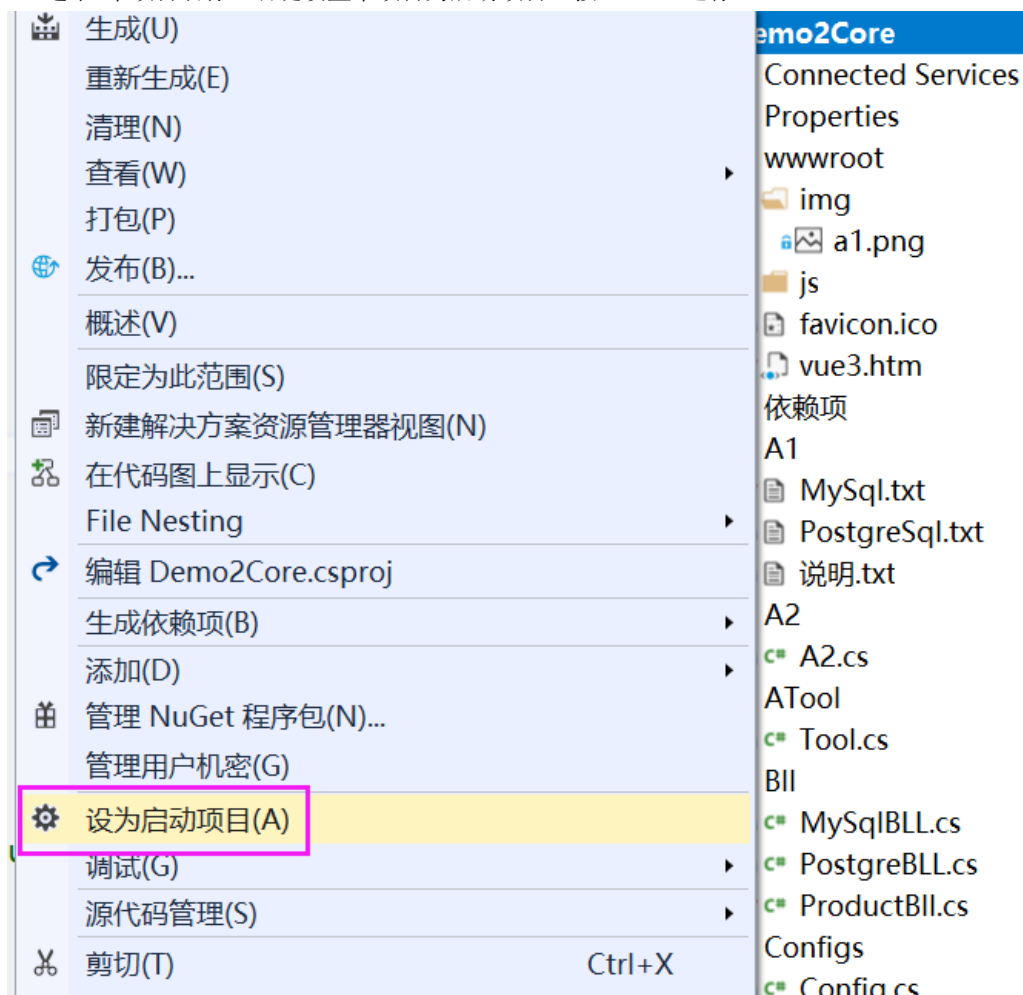
用VS打开 \A1\MySql.txt，复制里面的sql语句到HeidiSql【查询】窗口，按F9执行sql语句来创建数据库、创建表 -> 左侧test数据库，按F5刷新，查看创建了aconfig、product两个表



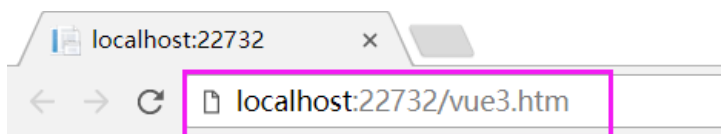


3) 修改a.config的MySQL数据库连接串，如果之前默认密码与文档一致，则不用修改连接串

4) 选中 本项目名称，右键设置本项目为启动项目，按Ctrl+F5运行



5) 复制/vue3.htm 到浏览器中，拼接地址后面，例如<http://localhost:22732/vue3.htm> 你的localhost后面端口号可能数字不一样，测试操作增删查改几个按钮



.NET Core 2.x demo1值  
【0】A.GetRootPath() 当前项目: D:\A7\VstsGit10  
【1】a.config配置节点Config.LogPath: D:\Log\  
【2】读写cookie值:  
【3】读写本地缓存值: k1缓存值  
【4】api接口/home/demo1?id=1&name=测试  
【5】vue+dnc示例/vue3.htm

<http://localhost:22732/vue3.htm> 你的localhost后面端口号可能数字不一样

## vue + dnc / .NET Core 2.x

Demo示例, 增删查改, vue调用dnc后台接口

id  name

pageIndex  pageSize

返回值: [{ "Id": 104, "ProductName": "奇才" }, { "Id":

数据列表: [{ "Id": 104, "ProductName": "奇才" }, { "Id

### Id ProductName

104 奇才

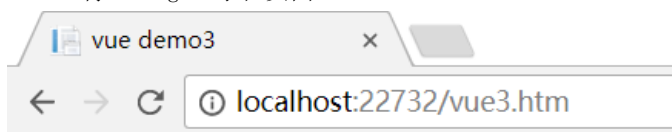
Delete

105 好

Delete

4、VS打开相关代码文件, 分析代码逻辑

1) vue将message显示在页面上



## vue + dnc / .NET Core 2.x

wwwroot/vue3.htm

```
<div id="a1">
  {{ message }}
</div>
```

wwwroot/js/demo3.js

```
//将message显示在页面上, el: '#a1'对应<div id="a1">中的id值a1
var a1 = new Vue({ el: '#a1', data: { message: 'vue + dnc / .NET Core 2.x' } });
```

2) vue3.htm页面, 输入name任意内容, 点击【添加】按钮

## Demo示例，增删查改，vue调用dnc后台接口

id  name

pageIndex  pageSize

wwwroot/vue3.htm

```
<div id="a2">
  <!-- vue + dnc 增删查改 -->
  id <input type="text" v-model="newProductId" style="width:80px" />
  name <input type="text" v-model="newProductName" style="width:80px" />
  <button @click="addProduct">添加</button>
```

wwwroot/js/demo3.js

```
//综合示例 vue 调用dnc后台接口
var a2 = new Vue({
  el: '#a2', //对应<div id="a2">中的id值a2
  data: {
    newProductId: 0, //对应v-model="newProductId"输入框
    newProductName: '', //对应v-model="newProductName"输入框
    result: '', //返回值
    productList: [], //产品列表数据
    pageIndex: 1, //默认值显示第1页
    pageSize: 10, //默认值每页10条
    isShowList: false //默认不显示列表数据
  },
```

vue接收页面name输入框newProductName的值，用post方式调用后台dnc接口/Home/AddProduct，插入一条数据，接收到后台接口的返回值response.data，判断vm.result.status == 1表示调用接口成功插入一条数据，然后调用a2.getProductList() 在页面上显示出新数据

```
name <input type="text" v-model="newProductName" style="width:80px" />
```

```
methods: {
  //插入一条数据
  addProduct: function () {
    var vm = this; //必须这样，后面的字段赋值才能在页面显示{{result}}
    console.log("【vm.newProductName】" + vm.newProductName);
    //流程：从页面输入框得到vm.newProductName，
    //用Post方式提交到后台接口 /Home/AddProduct，得到接口返回值response.data
    var params = new URLSearchParams();
    params.append('productName', vm.newProductName);
    axios.post('/Home/AddProduct', params) //axios是一个js工具类库，用于调用后台接口
      .then(function (response) {
        vm.result = response.data; //返回值，赋值给vm.result才能在页面显示，因为this作用域不同
        if (vm.result != null && vm.result.status == 1) {
          a2.getProductList(); //调用查询列表数据的方法，必须用a2调用，因为vm、this作用域不同
        }
      })
      .catch(function (error) { console.log("【操作失败】" + error); });
  },
```

3) 选择列表中的一条数据，输入它的id、name修改新值，点击【更新】按钮，进行数据更新



id  name      
pageIndex  pageSize

返回值 : { "status": 1, "data": 1, "msg": "成功" }

数据列表 : [ { "Id": 104, "ProductName": "奇才" }, { "Id": 108, "ProductName": "更新1" }, { "Id": 109, "ProductName": "更新2" } ]

**Id ProductName**

奇才

wwroot/vue3. html

```
<div id="a2">
  <!-- vue + dnc 增删查改 -->
  id <input type="text" v-model="newProductId" style="width:80px" />
  name <input type="text" v-model="newProductName" style="width:80px" />
  <button @click="addProduct">添加</button>
  <button @click="updateProduct">更新</button>
  <button @click="deleteProduct(newProductId)">删除</button> <br />
  pageIndex<input type="text" v-model="pageIndex" style="width:80px" />
  pageSize<input type="text" v-model="pageSize" style="width:80px" />
  <button @click="getProductList">查询</button>

  <br />返回值 : {{result}}
  <br />数据列表 : {{productList}}
```

wwroot/js/demo3. js

```
//更新一条数据
updateProduct: function () {
  var vm = this;
  var params = new URLSearchParams();
  params.append('id', vm.newProductId);
  params.append('productName', vm.newProductName);
  axios.post('/Home/updateProduct', params)
    .then(function (response) {
      //console.log("【response.data】 " + response.data); //返回值
      vm.result = response.data; //返回值
      //for循环找到这一条数据，更新ProductName
      for (var i = 0; i < vm.productList.length; i++) {
        if (vm.productList[i].Id == vm.newProductId) {
          vm.productList[i].ProductName = vm.newProductName;
        }
      }
    })
    .catch(function (error) { console.log("【操作失败】 " + error); });
},
```

4) 列表数据中选择一条，输入它的id，点击【删除】按钮，或者直接在列表数据中点击【Delete】按钮进行删除



id  name

pageIndex  pageSize

返回值: { "status": 1, "data": 1, "msg": "成功" }

数据列表: [ { "Id": 104, "ProductName": "奇才" }, { "Id": 108, "ProductName": "更新1" }, { "Id": 109, "Pr

**Id ProductName**

奇才

更新1

wwwroot/js/demo3.js

//删除一条数据

```
deleteProduct: function (deleteId) {
```

```
    var vm = this;
```

```
    var params = new URLSearchParams();
```

```
    params.append('id', deleteId);
```

```
    axios.post('/Home/deleteProduct', params)
```

```
        .then(function (response) {
```

```
            vm.result = response.data; //返回值
```

```
            if (vm.result != null && vm.result.status == 1) {
```

```
                //for循环找到这一条数据进行删除, console.log记录浏览器调试日志, 方便排查问题
```

```
                for (var i = 0; i < vm.productList.length; i++) {
```

```
                    if (vm.productList[i].Id == deleteId) {
```

```
                        console.log("【vm.productList[i].Id】找到匹配元素" + vm.productList[i].Id);
```

```
                        vm.productList.splice(i, 1); //删除一个
```

```
                    }
```

```
                }
```

```
            }
```

```
        }).catch(function (error) { console.log("【操作失败】" + error); });
```

```
    },
```

5) 输入pageIndex显示第几页, pageSize每页显示几条数据, 点击【查询】按钮, 查出数据列表

pageIndex 1      pageSize 10      查询

返回值 : [ { "Id": 104, "ProductName": "奇才" }, { "Id": 108, "ProductName": "更新1" }, { "Id": 109, "P  
数据列表 : [ { "Id": 104, "ProductName": "奇才" }, {  
{ "Id": 108, "ProductName": "更新1" }, { "Id": 109, "

Id ProductName	
104 奇才	Delete
105 更新1	Delete

wwwroot/js/demo3.js

```
//查询列表
getProductList: function () {
    var vm = this;
    var params = new URLSearchParams();
    params.append('pageIndex', vm.pageIndex);
    params.append('pageSize', vm.pageSize);
    axios.post('/Home/getProductList', params)
        .then(function (response) {
            vm.result = response.data; //返回值
            vm.isShowList = true; //将默认隐藏的数据列表显示出来
            vm.productList = response.data;
            console.log("【vm.productList】" + response.data);
        }).catch(function (error) { console.log("【操作失败】" + error); });
},
```

6) 上面已经演示了基本的增删查改，在页面上输入值，点击按钮，vue调用后台dnc接口进行增删查改，下面演示dnc后台的C#代码

dnc / .NET Core / DotNet Core是编程平台，它的编程语言是C#

go是编程平台，它的编程语言也是go

java是编程平台，它的编程语言也是java

Controllers  
HomeController.cs  
VMs  
a.config

C#编程语言的文件后缀名是.cs

\Controllers\HomeController.cs

```
// /home/AddProduct?productName=%E6%96%B01
0 个引用 | mike, 2 小时前 | 1 名作者, 2 项更改
public async Task<string> AddProduct(string productName)
{
    if (productName.IsNullOrEmpty())
        return "productName不能为空";

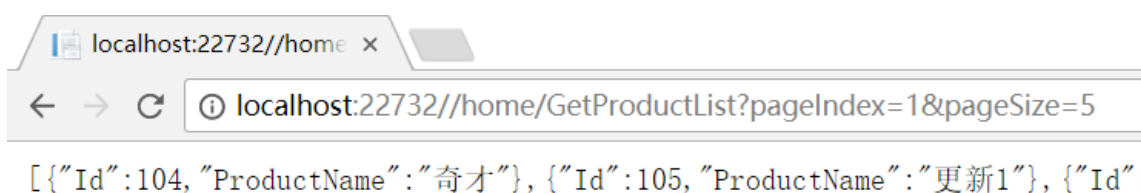
    return await ProductBll.AddProduct(productName);
}
```

将鼠标移到ProductBll.AddProduct(productName)的方法名AddProduct中，按F12进入这个方法的具体代码，它使用Dapper轻量级ORM工具执行一条sql语句，这条sql语句是向MySQL/MariaDB/TiDB开源数据库中插入一条数据，返回标准json值，可以鼠标移到Code.SuccessJson方法，按F12进入查看它的代码，HomeController.cs中其它方法类似，可打开查看

```
public class ProductBll
{
    /// <summary>
    /// 添加一个商品，Dapper + TiDB或MySQL或MariaDB插入，返回标准json
    /// </summary>
    1 个引用 | mike, 2 天前 | 1 名作者, 1 项更改
    public static async Task<string> AddProduct(string productName)
    {
        using (var conn = Db.Conn())
        {
            var sql = "insert into product (productName) values (@productName)";
            var amount = await conn.ExecuteAsync(sql, new { productName = productName });
            return (amount > 0) ? Code.SuccessJson(amount) : Code.FailJson("添加商品失败");
        }
    }
}
```

7) 实际的团队分工中，后台工程师一般只负责开发后台接口，前端html页面、js、vue代码等都是前端工程师负责开发，你可以直接测试后台接口，不需要前端页面和js、vue代码，例如复制HomeController.cs中的绿色注释Url，在浏览器中拼接到地址后面访问，可方便地测试后台接口代码

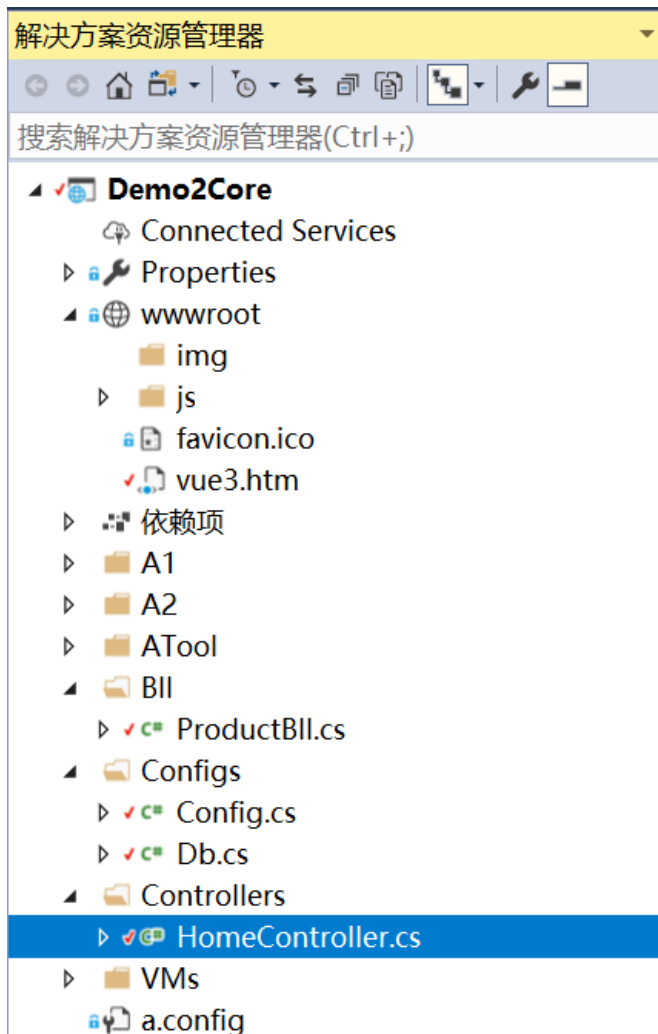
```
// /home/GetProductList?pageIndex=1&pageSize=5
0 个引用 | mike, 4 天前 | 1 名作者, 1 项更改
public async Task<string> GetProductList(int pageIndex, int pageSize)
{
    return await ProductBll.GetProductList(pageIndex, pageSize);
}
```



5、恭喜！到这一步，你已经神奇地在1小时内，从零基础开始，学会了最重要的【编程套路】与示例代码，建议打开各

个代码文件，再仔细看一遍，加深理解，从HomeController作为入口开始看。

整体思路：从页面输入框接收数据 -> 提交到js、vue代码 -> vue代码调用后台接口 -> 后台接口执行Sql语句，向数据库提交指令 -> 数据库进行数据的增删查改



三、dnc + vue 后续打怪升级、极简教程、还是简到崩溃！

- 1、如何设计用户登录注册模块？如何进行权限控制？如何增加a.config配置节点、数据库缓存表配置节点、如何读写Cookie、读写本地缓存？
- 2、如何设计比较复杂的电商平台？
- 3、电商平台系统上线运行后，客户订单越来越多，用户访问量越来越大，系统快扛不住了，怎么办？如何提升性能？
- 4、中小型公司是否只能照搬BAT的复杂系统架构，不断招聘大量工程师才能开发出大型系统？有没有性价比更高的高性能技术方案？作为老板，你的利润是否都被迫投入到无限扩张的研发成本中了？
- 5、如何开发一个高性能、水平无限扩展、支撑亿万级用户、百亿级订单的系统？

后续更高深的技术教程，正在抓紧准备中，敬请期待~

#### 【版权申明】

基于MIT开源协议，无需报备，欢迎转载，但必须保留来源链接及官方QQ群，否则追究法律责任

来源：<https://github.com/AskvEdu>

.NET新时代开源社区QQ群：618093978

C#区块链编程学院QQ群：546200985