

## 02\_零基础30分钟入门demo示例dnc+vue【AskYedu.com】

《AskY极简教程：零基础1小时学编程》开源教程

从零开始全程演示，如何开发一个大型互联网系统，开源教程 + 开源代码 + 开源解决方案

本教程从零基础入门，1小时学完前3节后，可学会基本编程思路，后面的教程是在这基础上的深入进阶，每个小节学习时间30分钟左右，由浅入深，循序渐进，从完全不懂编程到逐渐掌握编程技能

AskYedu.com 首届dnc开源峰会 dncNew.com 开源社区QQ群 618093978

- 01、安装开发工具、开源数据库
- 02、零基础30分钟入门demo示例dnc + vue
- 03、开发用户注册、登录、在线状态模块
- 04、本机vbox虚拟机安装Linux CentOS系统
- 05、XShell连接Linux基本操作
- 06、Linux离线安装dnc运行环境
- 07、VS发布生成dnc部署包
- 08、部署dnc到Linux + 守护进程systemd
- 09、nginx负载均衡 + 多台Linux Web服务器
- 10、Linux搭建Redis Cluster集群
- 11、dnc + Redis 零基础30分钟上手
- 12、C#编程语言基本语法
- 13、Linux基本操作命令
- 14、vim基本操作
- 15、AskY开源工具类库Nuget
- 16、dnc + 开源数据库Tidb
- 17、dnc + 开源数据库PostgreSQL
- 18、dnc + 极简分片分库 + MySQL/PostgreSQL
- 19、dnc + RabbitMQ 消息队列
- 20、dnc + kafka 消息队列
- 21、dnc + ElasticSearch 搜索引擎
- 22、dnc + Docker 容器
- 23、微服务架构 dnc on Linux/Docker

---

## 02、零基础30分钟入门demo示例dnc + vue

dnc = .NET Core、dotnet Core 简写

dnc是微软新一代主力编程平台，开源、免费、跨平台、轻量级、高性能，可部署到Linux、Docker、k8s等环境，适合开发微服务、云原生、大型互联网应用、全开源解决方案

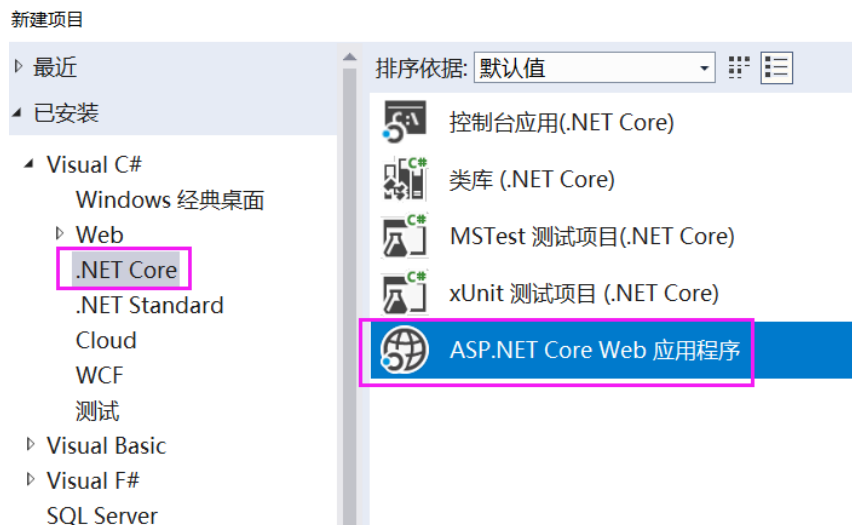
vue是流行的javascript前端开发框架

dv = dnc + vue 前后端分离开发，dnc工程师写后台接口，vue工程师写前端代码，分工合作

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 ASP.NET Core Web 应用程序 - Demo2Core



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

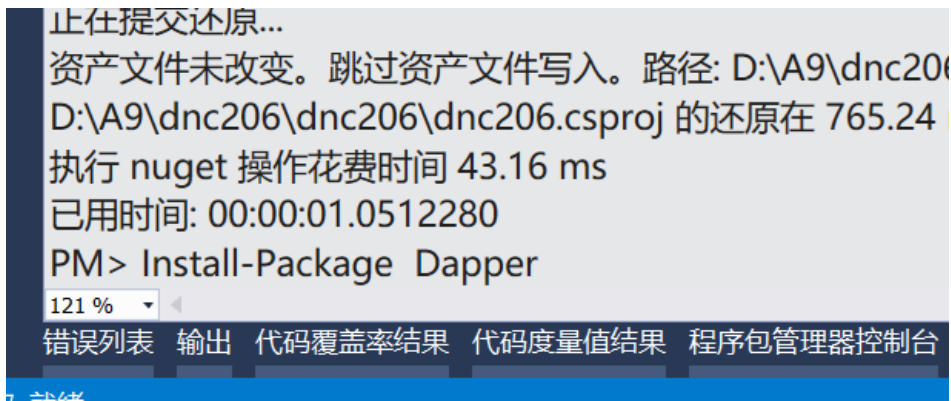
Install-Package AskyCore

Install-Package MySqlConnector

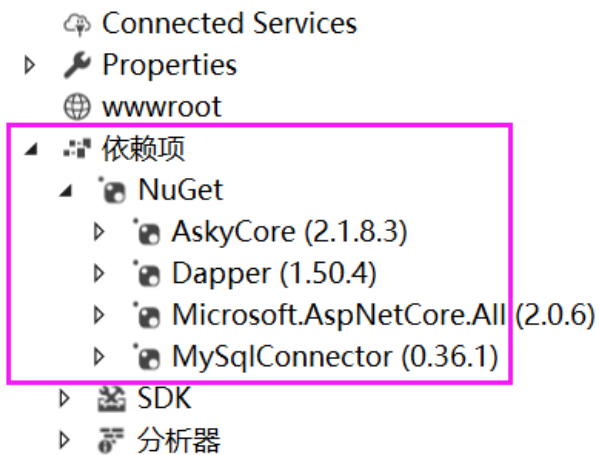
Install-Package Dapper



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



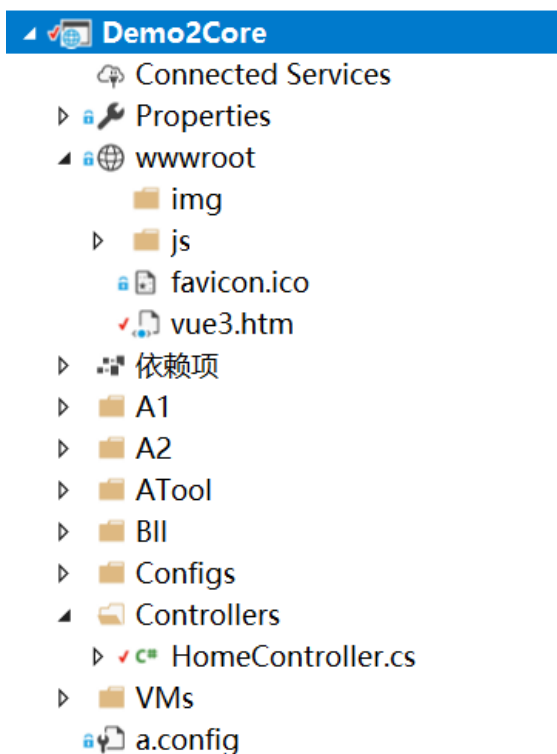
如果找不到【程序包管理器控制台】这个窗口，可从视图菜单 -> 其他窗口 -> 打开【程序包管理器控制台】



3、下载示例代码、体验dnc + vue

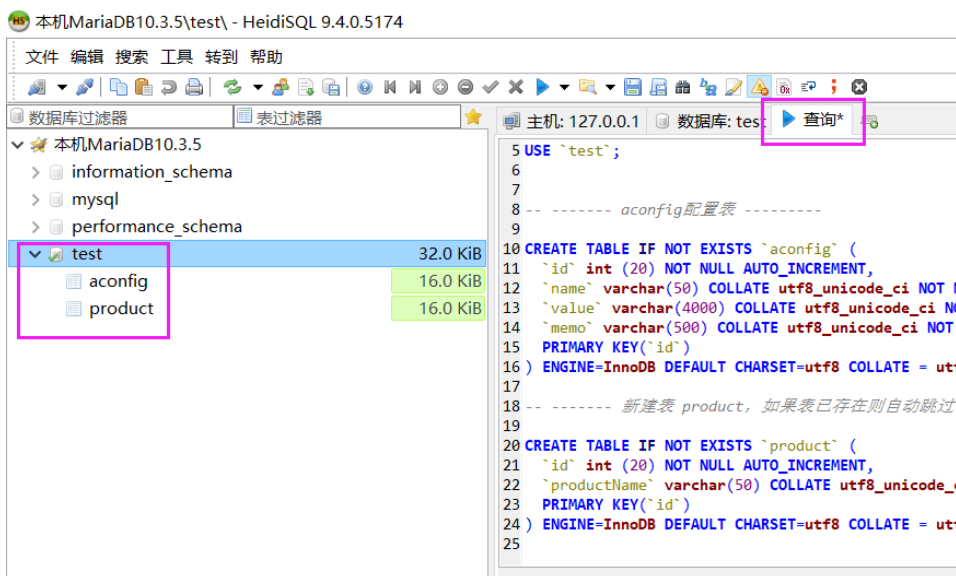
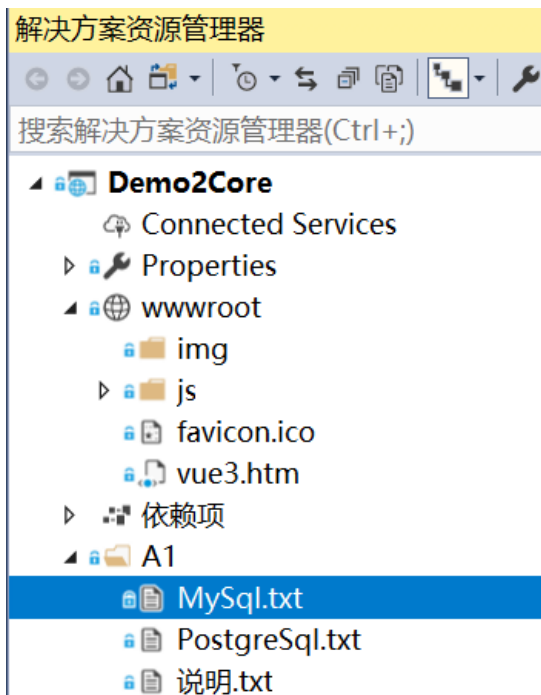
1) 下载示例代码 [https://github.com/AskvEdu/Askv/raw/master/02\\_demoCode/01\\_Demo2Core.zip](https://github.com/AskvEdu/Askv/raw/master/02_demoCode/01_Demo2Core.zip)

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：

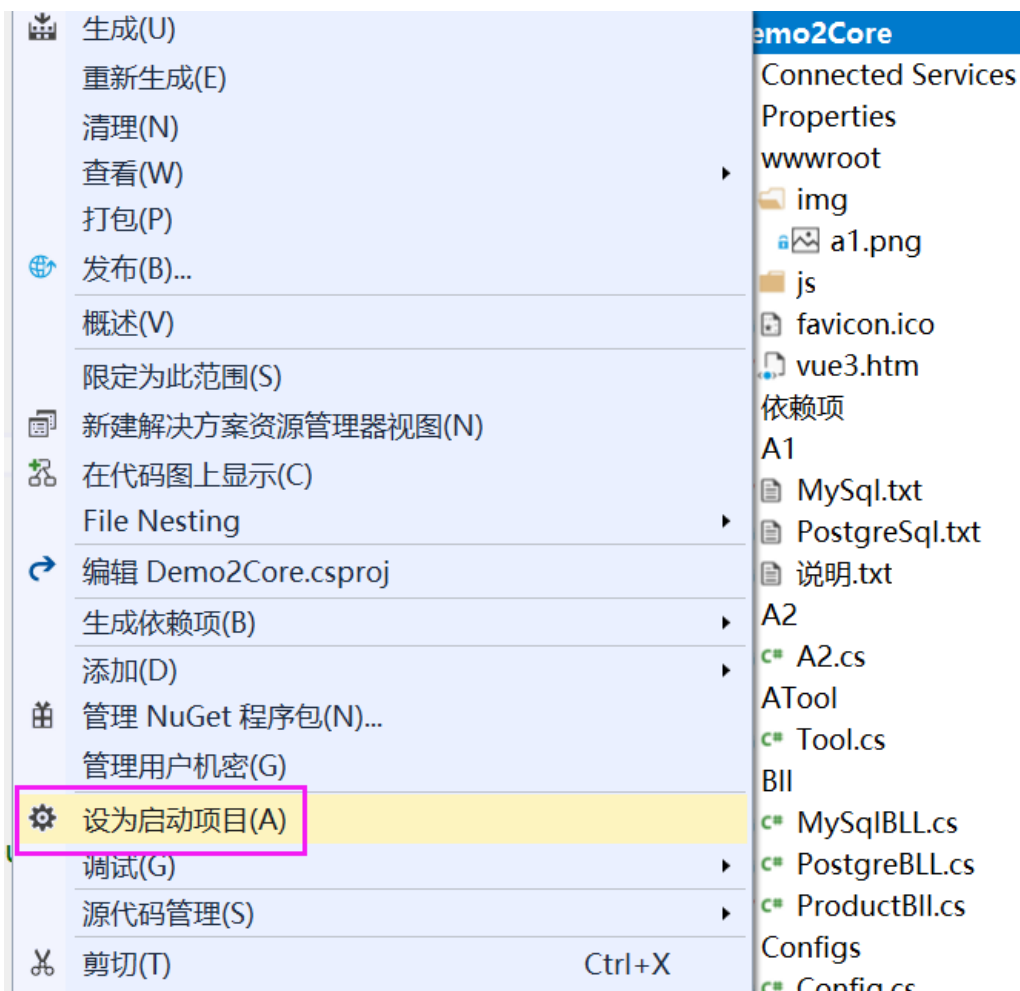


2) 参考第1个软件安装文档，用HeidiSql工具连接本机MariaDB数据库，点击【查询】

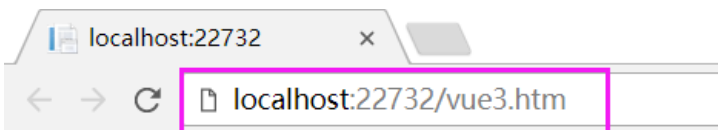
用VS打开 \A1\MySql.txt，复制里面的sql语句到HeidiSql【查询】窗口，按F9执行sql语句来创建数据库、创建表 -> 左侧test数据库，按F5刷新，查看创建了aconfig、product两个表



- 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，并拼接到地址后面，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"

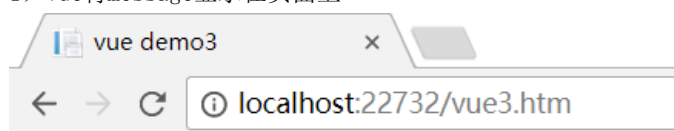
### Id ProductName

104 奇才

105 好

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
  },
  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); });
    },
  },
});
```

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



```

<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}}

```

wwwroot/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

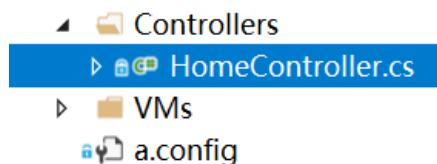
```
//查询列表
getProductList: function () {
    var vm = this;
    var params = new URLSearchParams();
    params.append('pageIndex', vm.pageIndex);
    params.append('pageSize', vm.pageSize);
    axios.post('/Home/getProductList', params)
        .then(function (response) {
            vm.result = response.data; //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



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";

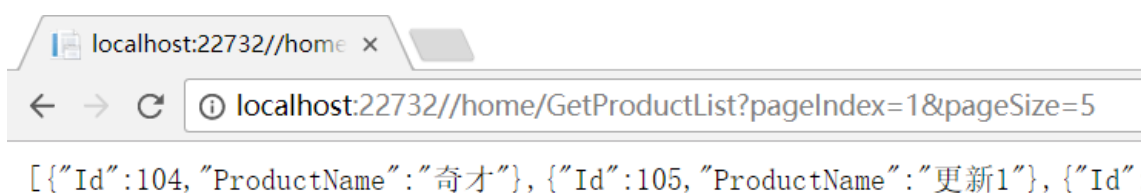
    return await ProductBll.AddProduct(productName);
}
```

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);
}
```



localhost:22732//home x

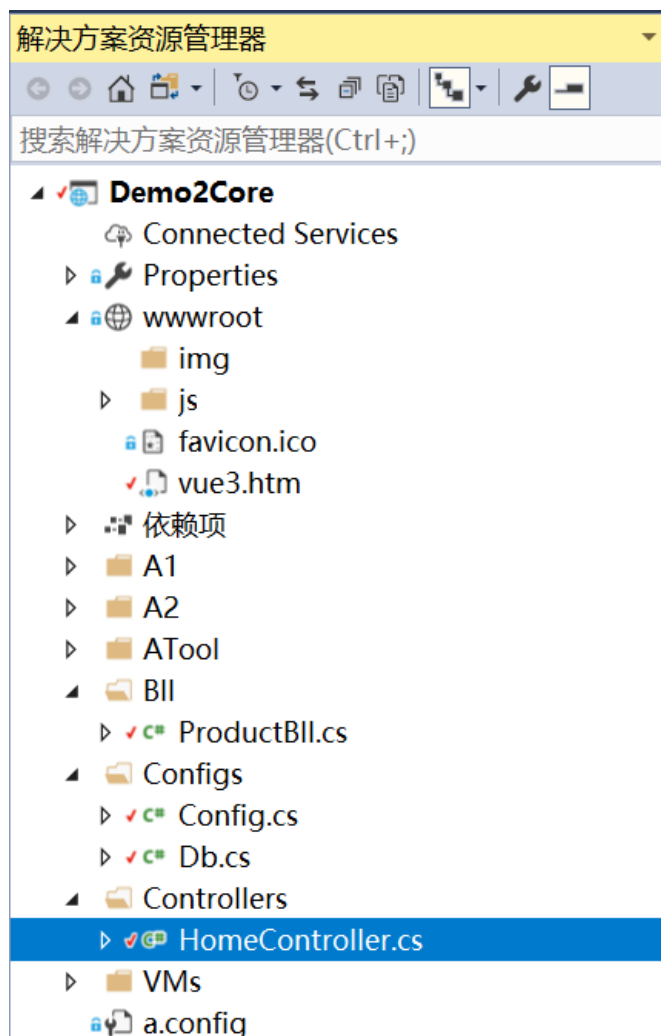
localhost:22732//home/GetProductList?pageIndex=1&pageSize=5

[{"Id":104,"ProductName":"奇才"}, {"Id":105,"ProductName":"更新1"}, {"Id":106,"ProductName":"更新2"}]

5、恭喜！到这一步，你已经神奇地从零基础开始，学会了最重要的【编程思维】与示例代码，建议打开各个代码文件，

再仔细看一看，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vue代码 -> vue代码调用后台dnc接口 -> 后台dnc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



dnc = .NET Core、dotnet Core 简写

dnc是微软新一代主力编程平台，开源、免费、跨平台、轻量级、高性能，可部署到Linux、Docker、k8s等环境，适合开发微服务、云原生、大型互联网应用、全开源解决方案

dnc国内公司案例

微软、腾讯、网易、同程旅游、龙珠直播、ThoughtWorks、新东方教育科技、中通快递、申通快递、青客白领公寓、途虎养车、博客园、视高盛景、如鹏网、行云创新、大连医卫、盛派网络、切尔思科技、斯诺物联、山海致远、neo.org开源区块链、aelf.io开源区块链……等公司

dnc各公司招聘职位: <http://cnblogs.com/dncNew/p/dnc.job.html>

本系列开源教程的后续章节，正在准备中，敬请期待~

- 1、如何实现SSO单点登录？如何设计复杂的电商平台？
- 2、电商平台上线运行后，订单越来越多，访问量越来越大，系统快扛不住了，怎么办？

- 3、研发团队500人以下的公司是否只能照搬BAT的复杂架构，招聘大量工程师才能开发出大型系统？
- 4、有没有性价比更高的技术方案？作为老板，你的利润是否快要赶不上不断上涨的研发成本？
- 5、如何开发一个高性能、水平扩展、十亿级到百亿级数据量、百万级并发访问的系统？

重要声明：这是从零开始的开源教程+开源代码，与任何培训机构无关，也不用于盈利目的

**【版权声明】**

本教程+代码 基于MIT开源协议，欢迎转载，但必须保留来源链接，否则追究法律责任

来源：<http://AskyEdu.com> 首届dnc开源峰会 <http://dncNew.com> 开源社区QQ群 618093978