**做实验的先决条件：**

* 你已经将GitHub主页上的项目克隆到你的本地

将远程项目克隆到本地

git clone https://github.com/264dormitory/dotnet

建立本地的分支

git checkout -b 本地分支名称

* 你的电脑中有MySQL，并且已经打开了MySQL的服务（如果使用wampsever的话，直接打开wampserver就好）
* 你已经打开了后端的服务进程

在cmd中先查找到api.jar所在的位置，这里以桌面举例

cd Desktop

java -jar api.jar

**步骤一：**

文件的移植全部以HTML文件为中心进行移植，首先在相应页面的Controllers文件夹里面绑定对应的视图，例如：

// GET: 绑定登录注册页面的视图

public ActionResult LoginAndRegist()

{

return View();

}

然后鼠标选中LoginAndRegist之后点击鼠标右键->添加视图，之后创建一个和原来的Html页面同名的View视图即可。

建立视图成功之后，可以在View文件夹里面发现一个与Controller相同名称的文件夹，文件夹里面将会包含一个与刚刚创建的视图相同的以.cshtml结尾的文件。

**步骤二：**

鼠标双击刚刚建立的视图，将原有项目HTML代码粘贴到视图之中，即可完成项目的移植。

在移植HTML页面的时候应该注意将页面中所需要引入的CSS和JS文件也同时引入到ASP的项目中（复制、粘贴），引入之后应按照ASP里面所规定的方式来对文件的路径进行设置。

**步骤三：**

在引入了文件之后应测试引入的页面是否能正常显示，引入的JS、CSS、图片和字体等资源的路径是否正确。下面简单说明一下具体的引入资源的规范，以及如何对引入的资源是否正确加载进行测试。

**引入资源的规范：**

* 在引入CSS和JavaScript文件的时候，一律采用@Url.Content(“my\_url”)的方式来对资源的路径进行格式化。
* 当页面需要引入一些共有的页面元素的时候（例如每一页的页眉和页脚），一律将共有元素的HTML文件放在~/Views/pageModel的路径下面，并使用一下语句来对相应的页面模块进行引用@Html.Partial(“my\_page\_url”)。

**测试方法：**

在相应的View页面代码中，点击Ctrl + F5运行本页面，首先查看页面是否与原页面的展示效果相同，之后在浏览器界面按下F12，点击“网络”一栏，之后按下F5键刷新页面，便能看见你所引入的相应的文件是否正确地加载了进来。