1.

 Razor 视图文件的 @page 指令。 @page 使文件转换为一个 MVC 操作 ，这意味着它将直接处理请求，而无需通过控制器处理。 @page 必须是页面上的第一个 Razor 指令。

@page

@using RazorPagesIntro.Pages

@model IndexModel2

<h2>Separate page model</h2> <p> @Model.Message </p>

Pages/Index2.cshtml.cs 页面模型：

C#复制

using Microsoft.AspNetCore.Mvc.RazorPages;using System;

namespace RazorPagesIntro.Pages

{

public class IndexModel2 : PageModel

{

public string Message { get; private set; } = "PageModel in C#";

public void OnGet()

{

Message += $" Server time is { DateTime.Now }";

}

}

}

下表显示了 Razor 页面路径及匹配的 URL：

| **文件名和路径** | **匹配的 URL** |
| --- | --- |
| /Pages/Index.cshtml | / 或 /Index |
| /Pages/Contact.cshtml | /Contact |
| /Pages/Store/Contact.cshtml | /Store/Contact |
| /Pages/Store/Index.cshtml | /Store 或 /Store/Index |

2

解析方法：

1、 首 先 Nuget 安装： Microsoft.Extensions.Configuration 和

Microsoft.Extensions.Configuration.Json

2、 然后 using Microsoft.Extensions.Configuration;

3、 下面的代码就把 Logging 下的 LogLevel 下的 Default 值读取出来

var builder =

new ConfigurationBuilder().SetBasePath(Directory.GetCurrentDirectory())//SetBasePath 设

置配置文件所在路径

.AddJsonFile("appsettings.json");

var configRoot = builder.Build();

var value =

configRoot.GetSection("Logging").GetSection("LogLevel").GetSection("Default").Value;

System.Console.WriteLine(value);

3.

Install-Package MySql.Data.EntityFrameworkCore -Version 8.0.12

暂时用Pomelo.EntityFrameworkCore.MySql替代

Microsoft.Extensions.Configuration

Microsoft.Extensions.Configuration.Json

1. using Microsoft.AspNetCore.Hosting;

var contentRootPath = \_IHostingEnvironment.ContentRootPath; //网 站 根 目 录 的 物 理 路 径 ）

\_IHostingEnvironment.WebRootPathwwwroot 文件 夹 的 物 理 路 径

1. https://github.com/NLog/NLog.web/wi 日志
2. Use 方法可使管道短路（即不调用 next 请求委托）

Map创建管道分支。 基于给定请求路径的匹配项来创建请求管道分支

Run:管道末端的中间件使用，最后一步

**注意注册的顺序**

app.Map("/test", applicateionBuilder => {

applicateionBuilder.Run(async context =>

{

await context.Response.WriteAsync("Map multiple segments.");

});

});// 输入路径https://localhost:44375/test 的时候会调用

app.Run(async requestHandler => {

await requestHandler.Response.WriteAsync("app.Run async requestHandler 直接结束");

});

app.Use(async (requestHandler,next) => {

await requestHandler.Response.WriteAsync("执行下一步");

Await Next();

});

7.可使用 @inject 指令将服务注入到视图中。

@inject StatisticsService StatsService

<!DOCTYPE html>

<html>

<head> <title>To Do Items</title> </head>

**<body> Total Items: @StatsService.GetCount()**

**</body>**

**</html>**

1. **tagHelper**
2. [Partial](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.rendering.htmlhelperpartialextensions.partial) 和 [RenderPartial](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.rendering.htmlhelperpartialextensions.renderpartial) 分别是 PartialAsync 和 RenderPartialAsync 的同步等效项。 但不建议使用同步等效项，因为可能会出现死锁的情况。 未来版本将不包含同步方法。

实例化分部视图时，它会获得父视图的 ViewData 字典的副本。 在分部视图内对数据所做的更新不会保存到父视图中。 在分部视图中的 ViewData 更改会在分部视图返回时丢失。

<partial name="/Views/Folder/\_ViewName.cshtml" />

@await Html.PartialAsync("/Views/Folder/\_ViewName.cshtml")

@await Html.PartialAsync("\_PartialName", viewModel)

1. ViewComponent 视图组件。不参与控制器生命周期，这意味着不能在视图组件中使用[筛选器](https://docs.microsoft.com/zh-cn/aspnet/core/mvc/controllers/filters?view=aspnetcore-2.1)

可通过以下任一方法创建视图组件类：

* 从 ViewComponent 派生
* 使用 [ViewComponent] 属性修饰类，或者从具有 [ViewComponent] 属性的类派生
* 创建名称以 ViewComponent 后缀结尾的类

1. [Area("Products")]
2. [Route("[controller]")]

[HttpGet("Destination", Name = "Destination\_Route")]

12

public class SystemDateTime : IDateTime {

public DateTime Now { get { return DateTime.Now; } } }

services.AddTransient<IDateTime, SystemDateTime>();

public IActionResult About([FromServices] IDateTime dateTime) {

ViewData["Message"] = "Currently on the server the time is " + dateTime.Now; return View();

}

public class SampleWebSettings {

public string Title { get; set; }

public int Updates { get; set; }

}

services.Configure<SampleWebSettings>(Configuration);

public class SettingsController : Controller {

private readonly SampleWebSettings \_settings;

public SettingsController(IOptions<SampleWebSettings> settingsOptions)

{

\_settings = settingsOptions.Value;

}

public IActionResult Index() {

ViewData["Title"] = \_settings.Title;

ViewData["Updates"] = \_settings.Updates;

return View();

}

}

13.

@Html.ActionLink("Go to Services Home Page", "Index", "Home", new { area = "Services" })

<a asp-area="Services" asp-controller="Home" asp-action="Index">Go to Services Home Page</a>

14[ActionResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.actionresult?view=aspnetcore-2.1)=》StatusCodeResult=》：

[Microsoft.AspNetCore.Mvc.BadRequestResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.badrequestresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.ConflictResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.conflictresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.NoContentResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.nocontentresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.NotFoundResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.notfoundresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.OkResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.okresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.UnauthorizedResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.unauthorizedresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.UnprocessableEntityResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.unprocessableentityresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.UnsupportedMediaTypeResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.unsupportedmediatyperesult?view=aspnetcore-2.1)

[System.Web.Http.ConflictResult](https://docs.microsoft.com/zh-cn/dotnet/api/system.web.http.conflictresult?view=aspnetcore-2.1)

[System.Web.Http.InternalServerErrorResult](https://docs.microsoft.com/zh-cn/dotnet/api/system.web.http.internalservererrorresult?view=aspnetcore-2.1)

[Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)=》ActionResult

Derived

[Microsoft.AspNetCore.Mvc.ChallengeResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.challengeresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.ContentResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.contentresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.EmptyResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.emptyresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.FileResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.fileresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.ForbidResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.forbidresult?view=aspnetcore-2.1)

IActionResult

Derived

[Microsoft.AspNetCore.Mvc.ActionResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.actionresult?view=aspnetcore-2.1)

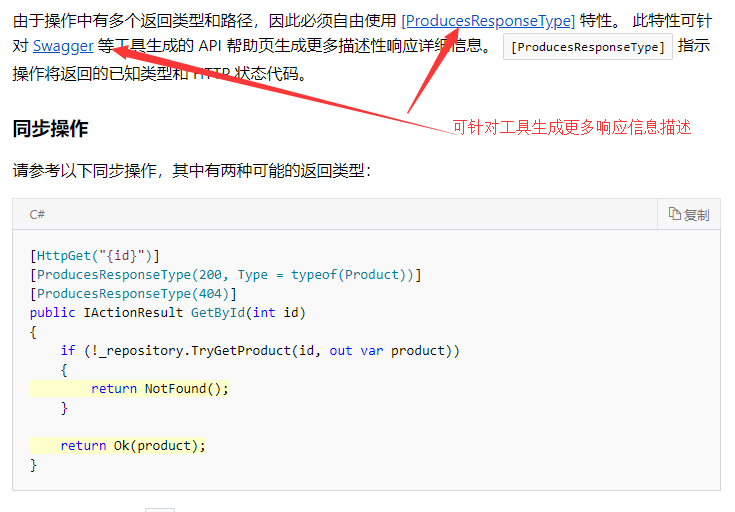
[Microsoft.AspNetCore.Mvc.RedirectResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.redirectresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.RedirectToActionResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.redirecttoactionresult?view=aspnetcore-2.1)

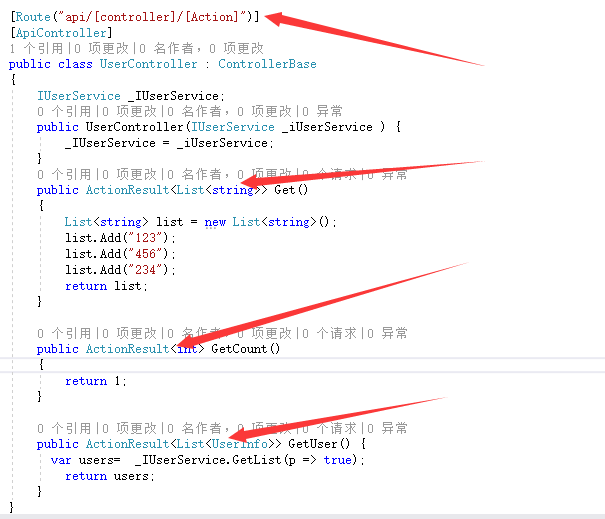
[Microsoft.AspNetCore.Mvc.RedirectToPageResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.redirecttopageresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.RedirectToRouteResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.redirecttorouteresult?view=aspnetcore-2.1)

[Microsoft.AspNetCore.Mvc.ViewFeatures.IKeepTempDataResult](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.viewfeatures.ikeeptempdataresult?view=aspnetcore-2.1)



1. ASP.NET Core 2.1 引入了面向 Web API 控制器操作的 [ActionResult<T>](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.mvc.actionresult-1) 返回类型

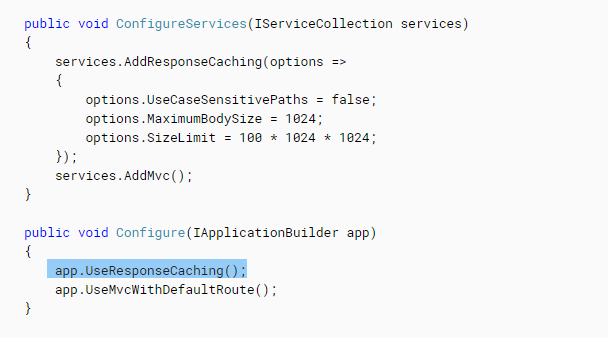


16.

* Transient  每次注入时，都重新 new 一个新的实体。《在去同一个请求中如果视图和action都注入了，那么这是了两个不同的实体》
* Scoped    每个 Request 都重新 new 一个新的实体。
* Singleton 程序启动后会 new 一个实体。也就是运行期间只会有一个实体。

1. [ResponseCache(Duration = 360)] 客户端缓存

服务端缓存：



1. IViewLocationExpander 扩展视图的位置

public class TemplateViewLocationExpander : IViewLocationExpander

{

public IEnumerable<string> ExpandViewLocations(ViewLocationExpanderContext context, IEnumerable<string> viewLocations)

{

var template = context.Values["template"] ?? "Default";

string[] locations = { "/Views/" + template + "/{1}/{0}.cshtml", "/Views/" + template + "/{0}.cshtml", "/Views/" + template + "/Shared/{0}.cshtml" };

return locations.Union(viewLocations);

}

public void PopulateValues(ViewLocationExpanderContext context)

{

context.Values["template"] = context.ActionContext.RouteData.Values["Template"]?.ToString() ?? "Default";

}

}

services.AddMvc(); //配置模版视图路径 services.Configure<RazorViewEngineOptions>(options => { options.ViewLocationExpanders.Add(new TemplateViewLocationExpander()); });