ASP.NET Core框架技术

敖 剑 范 华 主编

重庆大学电子音像出版社
http//www.cqup.com.cn

ASP.NET Core框架技术

[illegible] 主编

内容提要

本书主要讲解了.NET Core 介绍、ASP.NET Core 初识、Core MVC 筛选器与中间件、ASP.NET Core 路由与 Kestrel、Entity Framework Core、Identity Server 4 原理和实战、Consul 服务发现与注册、API 网关 Ocelot、远程过程调用 GRPC、ASP.NET Boilerplate 和 Linux 与 Docker 实战。通过本书的学习，学生能掌握 ASP.NET Core 框架的知识和实际应用。

版权信息

书名：ASP.NET Core 框架技术

主编：敖剑 范华

重庆大学电子音像出版社

责任编辑：田雨

地址：重庆市沙坪坝区大学城西路 21 号

ISBN 978-7-89446-563-4

出版时间：2022 年 1 月

邮编：401331

电话：023-88617016

字数：405 千字

定价：96.00 元

前 言

欢迎学习本系列课程（新形态教材），课程研发团队由教学经验丰富的一线老师、企业中工作经验丰富的行业专家及教育专家组成。

本系列课程作为新形态教材，广泛利用多媒体等新兴技术辅助教学，让学生寓教于乐，提高学习的兴趣和效率；同时，充分考虑了读者的阅读习惯和学习习惯，在编排上做了非常科学的安排：

- 本系列课程为作者团队花费了大量的人力、物力和财力倾力打造的新形态教材，全系采用“二维码链接配套资源”的新形态教材模式，每本教材都拥有视频教学资源、评估试题等配套教学资源，可以通过嵌入到教材中的二维码轻松查看配套资源，让学习变得高效、有趣又轻松。

- 本书包含学习目标、课程内容、总结、作业等，这个编排结构可以让读者更加轻松、高效地学习。一来可以提高理论的应用能力，二来可以巩固所学的理论知识，锻炼读者自己解决问题和举一反三的能力。

- 课程内容中含有实战案例，让读者在提高应用能力的同时获得实战经验，真正体现了学以致用的指导方针。

- 采用图文结合的编排方式，宽松的版式让读者可以轻松阅读。

本系列课程由大量的老师及专家给予支持和帮助，由于参与本系列课程研发的人数太多，在这里没有一一列出他们的名字，在此由衷地感谢他们！本课程中使用的图例和片段仅用于教学示范和讲解，不做其他商业用途。在编写过程中，有一些图例和片段无法确定作者与出处，在此也向他们深表感谢，并请原作者与出版社或主编本人联系。同时希望读者和同行人士多提宝贵意见和建议。

本系列课程适合教学使用，也适合自学使用。

编　者

2021 年 8 月 10 日

评估试题参考答案

案例资源

目录

第1章 .NET Core 介绍

教学资源

学习目标

了解什么是.NET Core

了解.NET Core 与.NET Framework 区别

了解.NET Core 与 Mono 区别

掌握.NET Core 的环境安装

1.1 .NET Core 简介

1.1.1 什么是.NET Core

.NET Core 是开放源代码通用开发平台，由 Microsoft 和 .NET 社区在 GitHub 上共同维护。它跨平台（支持 Windows、macOS 和 Linux），并且可用于生成设备、云和物联网（IoT）应用程序。

1. .NET Core 特性

- 跨平台：可在 Windows、mac OS 和 Linux 操作系统上运行。
- 跨体系结构：在多个体系结构（包括 x64、x86 和 ARM）上以相同的行为运行代码。
- 命令行工具：包括可用于本地开发和持续集成方案中的易于使用的命令行工具。
- 部署灵活：可以包含在应用或已安装的并行（用户或系统范围安装）中。 可搭配 Docker 容器使用。
- 兼容性强：.NET Core 通过 .NET Standard 与 .NET Framework、Xamarin 和 Mono 兼容。
- 开放源代码：.NET Core 平台是开放源代码，使用 MIT 和 Apache 2 许可证。.NET Core 是一个.NET Foundation 项目。
- 由 Microsoft 支持：.NET Core 由 Microsoft 依据.NET Core 支持提供支持。

2. 语言

可以使用 C#、Visual Basic 和 F# 语言编写，适用于 .NET Core 的应用程序和库。这些语言可在你喜欢的文本编辑器或集成开发环境 (IDE) 中使用，包括：

- Visual Studio
- Visual Studio Code
- Sublime Text
- Vim

这种集成部分由 OmniSharp 和 Ionide 项目的参与者提供。

3. 框架

在 .NET Core 之上建立了多个框架：

- ASP.NET Core

- Windows 10 通用 Windows 平台 (UWP)
- Tizen

1.1.2 开源

.NET Core 是开源的（MIT 许可证），由 Microsoft 于 2014 年提供给 .NET Foundation。它现在是最活跃的 .NET Foundation 项目之一。个人和公司可出于个人、学术或商业等目的对其进行使用。许多公司将 .NET Core 用作应用、工具、新平台和托管服务的一部分。其中某些公司在 GitHub 上为 .NET Core 做出了巨大贡献，并作为 .NET Foundation Technical Steering Group（.NET Foundation 技术控制组）的成员为产品方向提供指导。

1.1.3 与 .NET Framework 对比

.NET 由 Microsoft 于 2000 年首次发布，而后发展至今。近 20 年以来，.NET Framework 一直是 Microsoft 出品的主要 .NET 实现。

.NET Core 和 .NET Framework 的主要差异在于：

- 应用模型——.NET Core 并非支持全部 .NET Framework 应用模型。 具体而言，它不支持 ASP.NET Web 窗体和 ASP.NET MVC，但支持 ASP.NET Core MVC。自 .NET Core 3.0 起，.NET Core 还支持 WPF 和 Windows 窗体（仅限在 Windows 上）。
- API——.NET Core 包含 .NET Framework 基类库的一个大型子集，但具有不同的组成要素（程序集名称不同；类型上公开的成员在关键用例中不同）。 在某些情况下，这些差异要求进行更改，以将源移植到 .NET Core 中。 有关详细信息，请参阅 .NET 可移植性分析器。 .NET Core 实施 .NET Standard API 规范。
- 子系统——.NET Core 实现 .NET Framework 中子系统的子级，目的是实现更简单的实现和编程模型。 例如，不支持代码访问安全性 (CAS)，但支持反射。
- 平台——.NET Framework 支持 Windows 和 Windows Server，而 NET Core 还支持 macOS 和 Linux。
- 开源——.NET Core 是开源的，而 .NET Framework 的只读子集是开源的。
- 虽然 .NET Core 是唯一的且与 .NET Framework 和其他 .NET 实现大不相同，但可使用源或二进制共享技术在这些实现之间轻松共享代码。
- 由于 .NET Core 支持并行安装及其运行时完全独立于.NETFramework，它可以安装在装有.NET Framework 的计算机上，不会出现任何问题。

1.1.4 与 Mono 比较

Mono 是 .NET 的原始跨平台。它一开始是用作 .NET Framework 的开源替代项，之后随着 iOS 和 Android 设备变得流行，过渡到面向移动设备。可将它看作是.NET Framework 的社区克隆。Mono 项目团队依赖于 Microsoft 发布的开放.NET 标准（尤其是 ECMA 335）来实现兼容性。

.NET Core 和 Mono 的主要差异在于：

- 应用模型——Mono 通过 Xamarin 产品支持部分 .NET Framework 应用模型（例如 Windows 窗体）和其他移动开发辅助模型（例如 Xamarin.iOS），而.NET Core 不支持 Xamarin。
- API——Mono 使用相同程序集名称和组成要素支持 .NET Framework API 的大型子集。
- 平台——Mono 支持很多平台和 CPU。
- 开源——Mono 和 .NET Core 两者都使用 MIT 许可证，且都属于 .NET Foundation 项目。
- 焦点——最近几年，Mono 的主要焦点是移动平台，而 .NETCore 的焦点是云和桌面工作负载。

1.1.5 未来动向

已宣布 .NET 5 将成为 .NET Core 的下一版本且表示平台实现统一。 此项目旨在以几项关键方法来改进.NET：

- 生成可随处使用且提供统一运行时行为和开发人员体验的单一.NET 运行时和框架。
- 通过充分利用 .NET Core、.NET Framework、Xamarin 和 Mono 来扩展 .NET 的功能。
- 根据单个基本代码构建开发人员（Microsoft 和社区）可处理且协同扩展，同时可改善所有方案的产品。

目前最新版本是.NET Core 3.0。新功能包括通过 Windows Presentation Foundation (WPF) 和 Windows 窗体提供的 Windows 桌面支持、使用 Blazor 进行完整堆栈 C# Web 开发、SignalR 和 Azure SignalR 服务的新增强功能、C# 8 的新增 C# 语言功能等等。下小节介绍.NET Core3.0 常用的新增功能。

1.1.6 ASP.NET Core 3.0 的新增功能

1. Blazor

Blazor 是一个使用 .NET 生成交互式客户端 Web UI 的框架。

使用 C# 代替 JavaScript 来创建信息丰富的交互式 UI。

共享使用 .NET 编写的服务器端和客户端应用逻辑。

将 UI 呈现为 HTML 和 CSS，以支持众多浏览器，其中包括移动浏览器。

2. Blazor 框架支持的方案

可重用的 UI 组件。（Razor 组件）

客户端路由。

组件布局。

对依赖项注入的支持。

窗体和验证。

用 Razor 类库构建组件库。

JavaScript 互操作。

3. Blazor 服务器

Blazor 将组件呈现逻辑从 UI 更新的应用方式中分离出来。 Blazor 服务器在 ASP.NET Core 应用中添加了对在服务器上托管 Razor 组件的支持。可通过 SignalR 连接处理 UI 更新。 ASP.NET Core 3.0 支持 Blazor Server。

4. Blazor WebAssembly（预览版）

使用基于 WebAssembly 的 .NET 运行时直接在浏览器中运行 Blazor 应用。 Blazor WebAssembly 处于预览版阶段，ASP.NET Core 3.0 不提供支持。 ASP.NET Core 的未来版本将支持 Blazor WebAssembly。

5. Razor 组件

Blazor 应用是基于组件构建的。 组件是自包含的用户界面 (UI) 块，例如页面、对话框或窗体。组件是定义 UI 呈现逻辑和客户端事件处理程序的普通 .NET 类。 无须 JavaScript 即可创建丰富的交互式 Web 应用。

通常使用 Razor 语法（HTML 和 C# 的自然混合）创建 Blazor 中的组件。 Razor 组件与 Razor Pages 和 MVC 视图类似，因为它们都使用 Razor。 与基于请求，响应模型的页面和视图不同，组件专门用于处理 UI 构成。