Visual Studio 中的容器工具

2019/03/20 •

本文内容

Visual Studio 中用于使用容器进行开发的工具易于使用，并大大简化生成、调试和部署容器化应用程序的过程。 可以将容器用于单个项目，也可以将容器业务流程与 Docker Compose、Service Fabric 或 Kubernetes 结合使用，以便使用容器中的多个服务。

先决条件

- Docker Desktop
- 安装了“Web 开发”、“Azure 工具”工作负载和/或“.NET Core 跨平台开发”工作负载的 Visual Studio 2019
- 用于使用 .NET Core 进行开发的 .NET Core 开发工具。
- 若要发布到 Azure 容器注册表，需要 Azure 订阅。 注册免费试用版。

Visual Studio 中的 Docker 支持

Docker 支持适用于 ASP.NET 项目、ASP.NET Core 项目，以及 .NET Core 和 .NET Framework 控制台项目。

Visual Studio 中的 Docker 支持因版本而异，以响应客户需求。 可以向项目添加两个级别的 Docker 支持，并且受支持的选项因项目类型和 Visual Studio 版本而异。 借助某些受支持的项目类型，如果只想将容器用于单个项目，而不使用业务流程，则可以通过添加 Docker 支持来完成。 下一级别是容器业务流程支持，该支持可为所选的特定业务流程协调程序添加相应的支持文件。

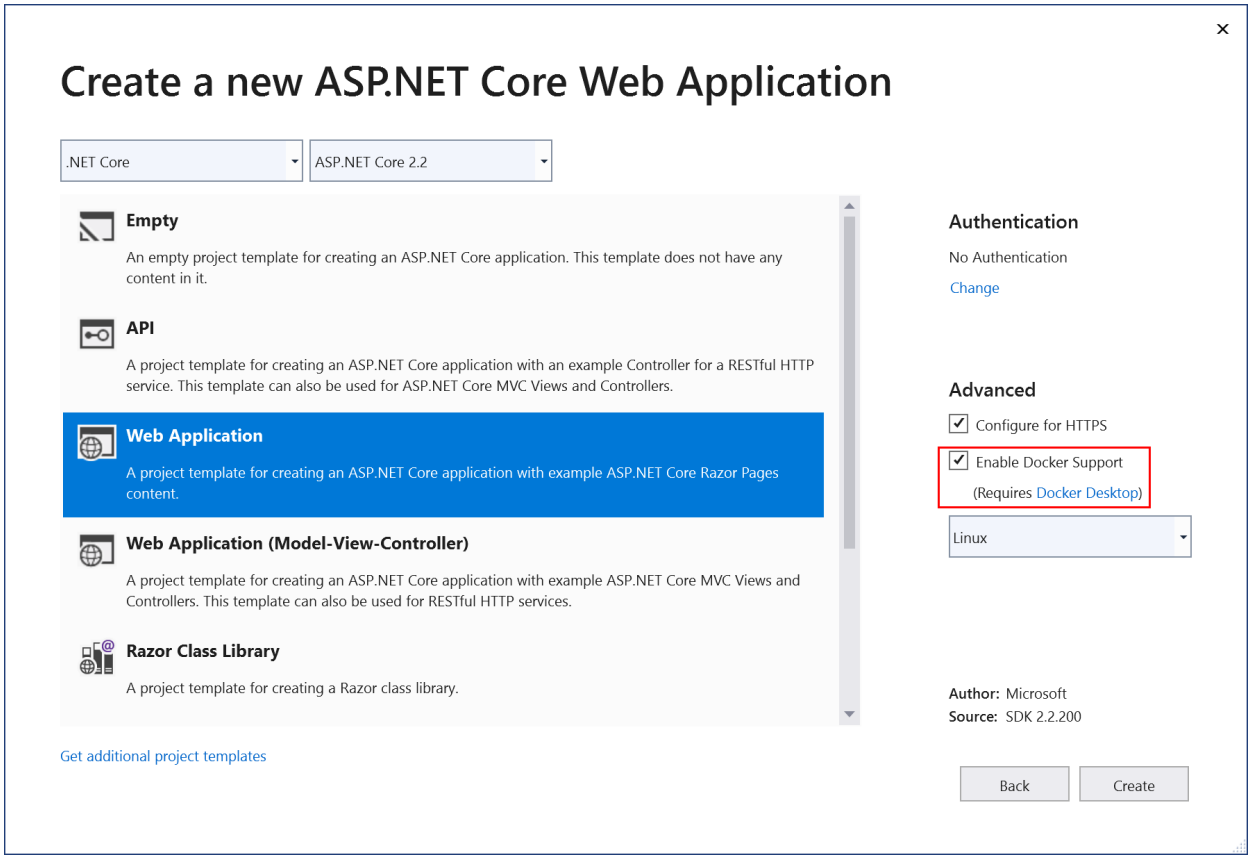
借助 Visual Studio 2019，可以将 Docker Compose、Kubernetes 和 Service Fabric 用作容器业务流程服务。

> **备注**
>
> 如果你使用完整的 .NET Framework 控制台项目模板，则在创建项目后，支持的选项是“添加容器业务流程协调程序支持”，它包括使用 Service Fabric 或 Docker Compose 的选项 。 对于没有业务流程的单个项目，无法在项目创建时添加支持，也无法添加 Docker 支持 。

在 Visual Studio 2019 版本 16.4 及更高版本中，提供了“容器”窗口，你可用它来查看正在运行的容器，浏览可用的映像，查看环境变量、日志和端口映射，检查文件系统，附加调试器，或者在容器环境中打开终端窗口 。 请 Visual Studio 中查看和诊断容器和映像。

添加 Docker 支持

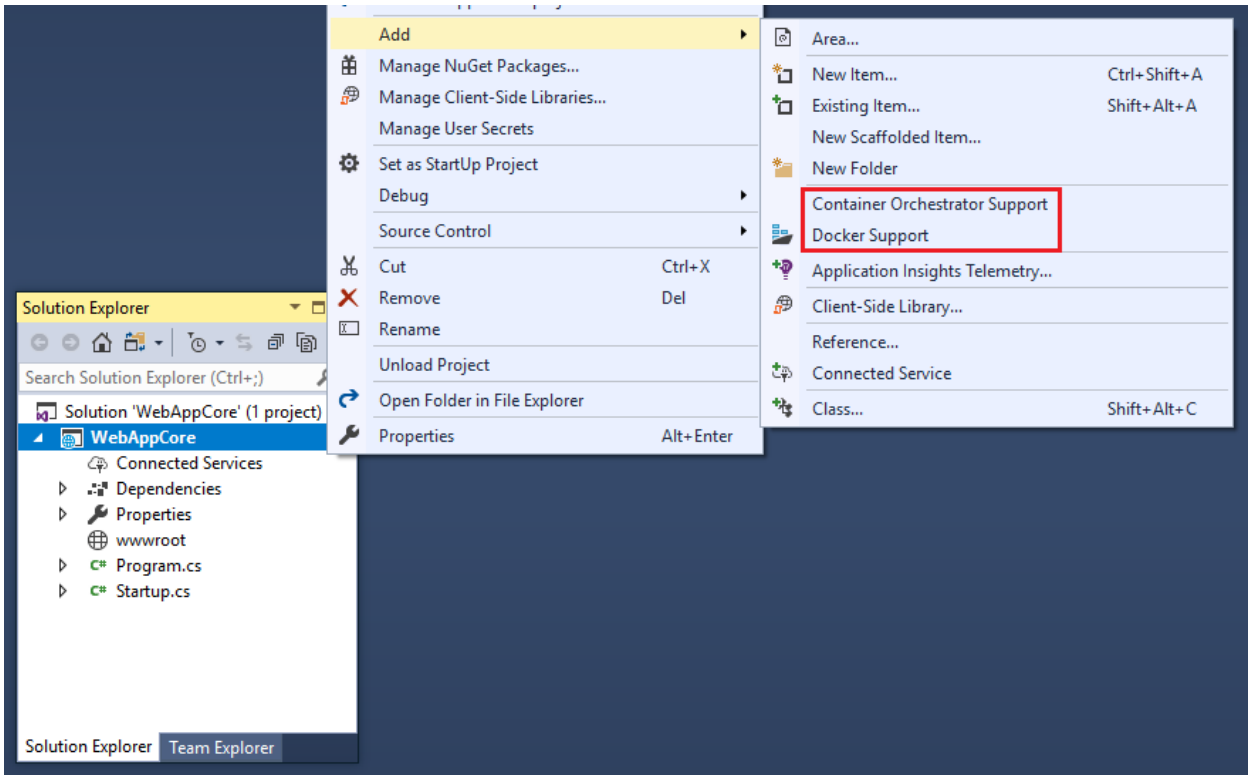
可以通过在创建新项目时选择“启用 Docker 支持”来在项目创建期间启用 Docker 支持，如以下屏幕截图所示 ：



> **备注**
>
> 对于 .NET Framework 项目（而不是 .NET Core），只有 Windows 容器可用。

可以通过在“解决方案资源管理器” 中选择“添加” > “Docker 支持” 来向现有项目添加 Docker 支持。 Add > Docker Support 和 Add > Container Orchestrator Support 命令位

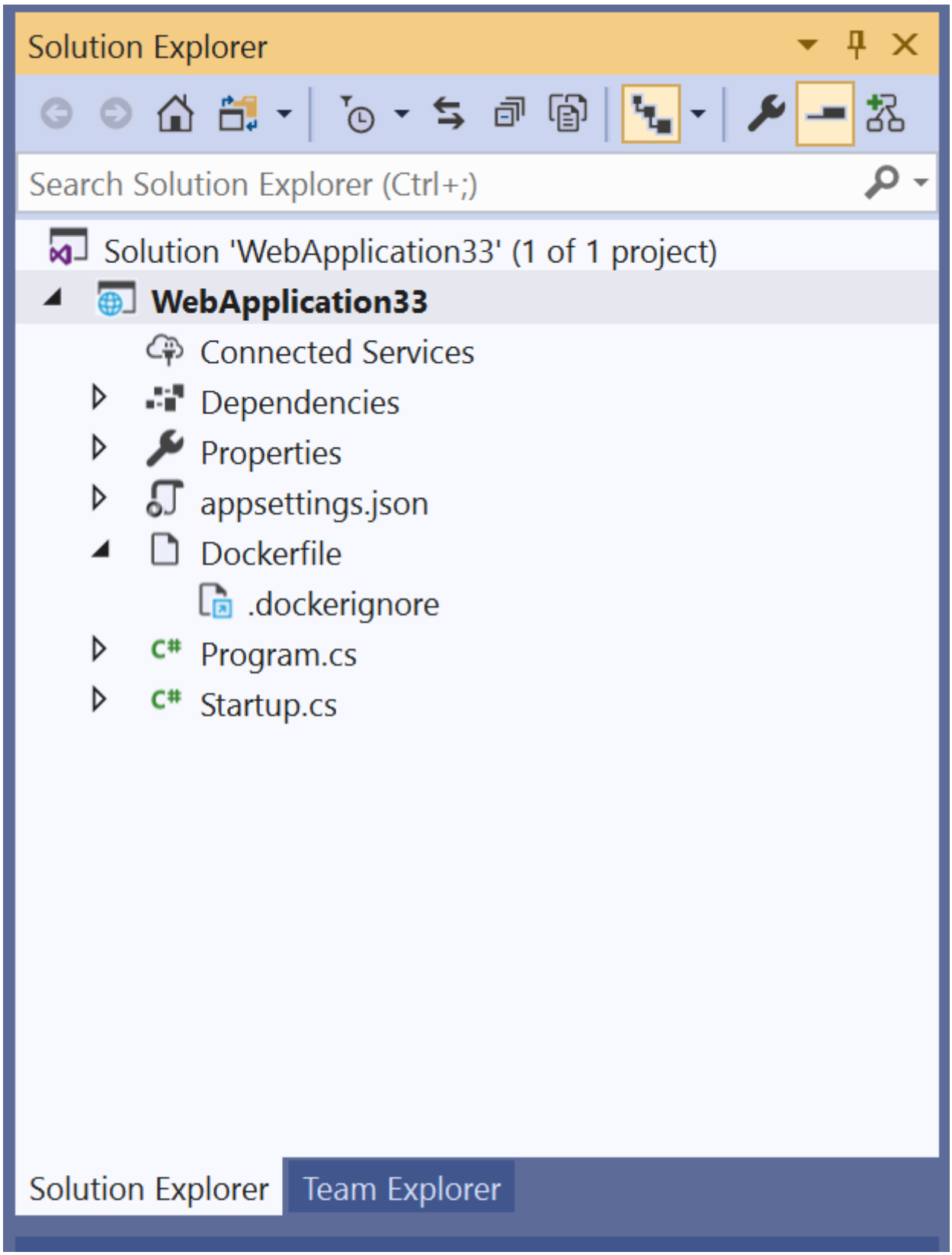
于“解决方案资源管理器” 中 ASP.NET Core 项目的项目节点的右键单击菜单（或上下文菜单）上，如以下屏幕截图所示：



当添加或启用 Docker 支持时，Visual Studio 会向项目添加以下各项：

- Dockerfile 文件
- .dockerignore 文件
- 对 Microsoft.VisualStudio.Azure.Containers.Tools.Targets 的 NuGet 包引用

添加 Docker 支持后，解决方案如下所示：

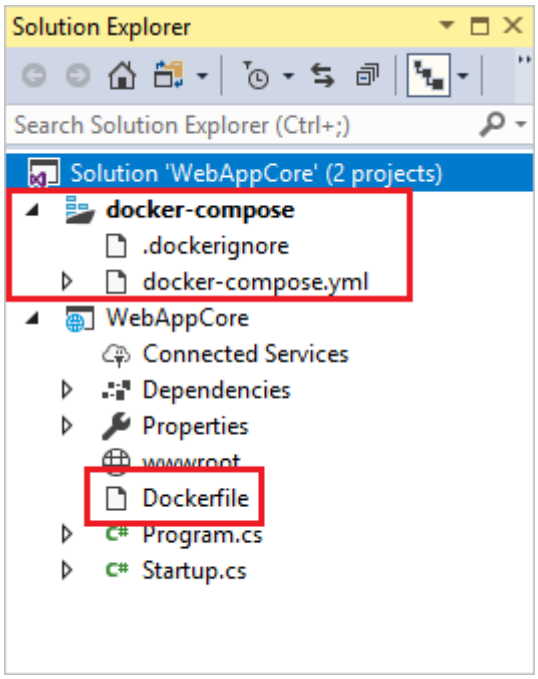


Docker Compose 支持

当你想要使用 Docker Compose 撰写多容器解决方案时，请向项目添加容器业务流程支持。 这样就可以同时运行和调试一组容器（整个解决方案或项目组）（如果已在同一个 docker-compose.yml 文件中定义这些容器）。

若要使用 Docker Compose 添加容器业务流程支持，请右键单击“解决方案资源管理器”中的解决方案或项目节点，然后选择“添加”>“容器业务流程支持” 。 然后，选择“Docker Compose” 以管理容器。

向项目添加容器业务流程支持后，会看到添加到项目的 Dockerfile （如果尚无）以及添加到“解决方案资源管理器” 中的某个解决方案的 docker-compose 文件夹，如下所示：



如果 docker-compose.yml 已存在，Visual Studio 只需向其添加配置代码所需的行。

对要使用 Docker Compose 控制的其他项目重复该过程。

Kubernetes 支持

借助 Kubernetes 支持，可以在本地项目和 Azure Kubernetes 服务 (AKS) 中运行的 Kubernetes 群集之间启用连接，从而使用 Visual Studio 修改和调试 AKS 中运行的服务。 此服务由 Azure Dev Spaces 提供。 通过 Azure Dev Spaces，还可以设置名为 dev spaces 的 Kubernetes 服务的单独分支用于开发目的，因此可以有效地将生产服务与开发中的工作版本隔离，并使每个修改完全不同。

若要向项目添加 Kubernetes 支持，请在添加容器业务流程支持时选择“Kubernetes/Helm” 。 多个文件添加到项目，其中包括 azds.yaml （用于配置 Azure Dev Spaces）和 Helm 图表（用于描述 Kubernetes 服务的结构）。

Service Fabric 支持

借助 Visual Studio 中的 Service Fabric 工具，可以开发和调试 Azure Service Fabric、进行本地运行和调试并部署到 Azure。

Visual Studio 2019 支持使用 Windows 容器和 Service Fabric 业务流程来开发容器化微服务。

有关详细教程，请参阅教程：将 Windows 容器中的 .NET 应用程序部署到 Azure Service Fabric。

有关 Azure Service Fabric 的详细信息，请参阅 Service Fabric。

持续交付和持续集成 (CI/CD)

Visual Studio 与 Azure Pipelines 轻松集成，以便自动完成服务代码和配置更改的持续集成和交付。 若要开始使用，请参阅创建第一个管道。

有关 Service Fabric 的信息，请参阅教程：使用 Azure DevOps Projects 将 ASP.NET Core 应用部署到 Azure Service Fabric。

有关 Kubernetes 的信息，请参阅将 Docker 容器应用部署到 Azure Kubernetes 服务。

后续步骤

有关服务实现以及将 Visual Studio 工具用于容器的更多详细信息，请阅读以下文章：

Debugging apps in a local Docker container（在本地 Docker 容器中调试应用）

使用 Visual Studio 将 ASP.NET 容器部署到容器注册表