

# 第02期-使用CLI命令行运行项目源码

2019年12月28日 8:03

## 微服务体系结构

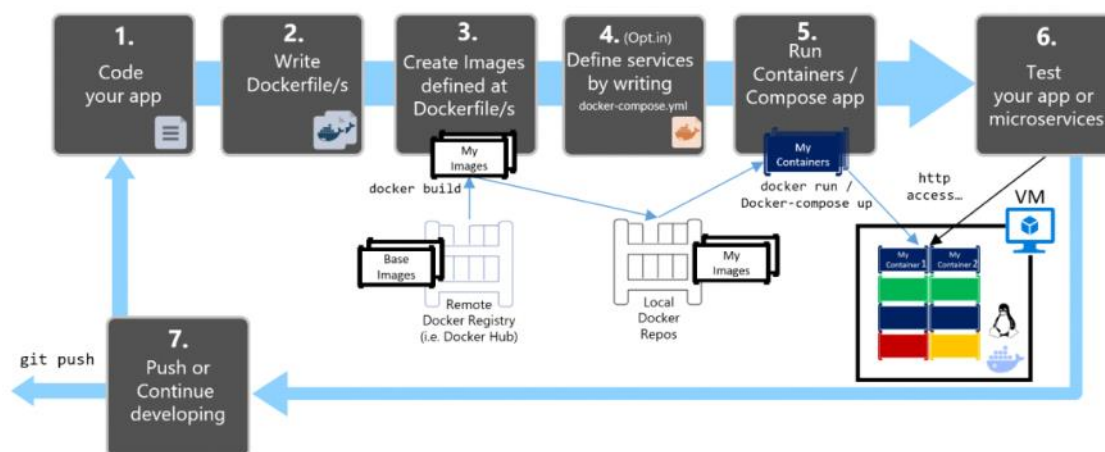
顾名思义，微服务体系结构是一种将服务器应用程序生成为一组小型服务的方法，主要面向后端，每个服务在自己的进程中运行，并使用 HTTP/HTTPS、WebSocket、gRPC 或 AMQP 等协议与其他进程进行通信，每个微服务自主开发，可领域驱动，独立部署，每个微服务拥有自己的逻辑和数据，可使用不同的数据库(SQL或者NoSQL)和不同的编程语言。

微服务应该有多大：松散耦合，方便开发、部署和缩放，它必须具有内部内聚，并且独立于其他服务。

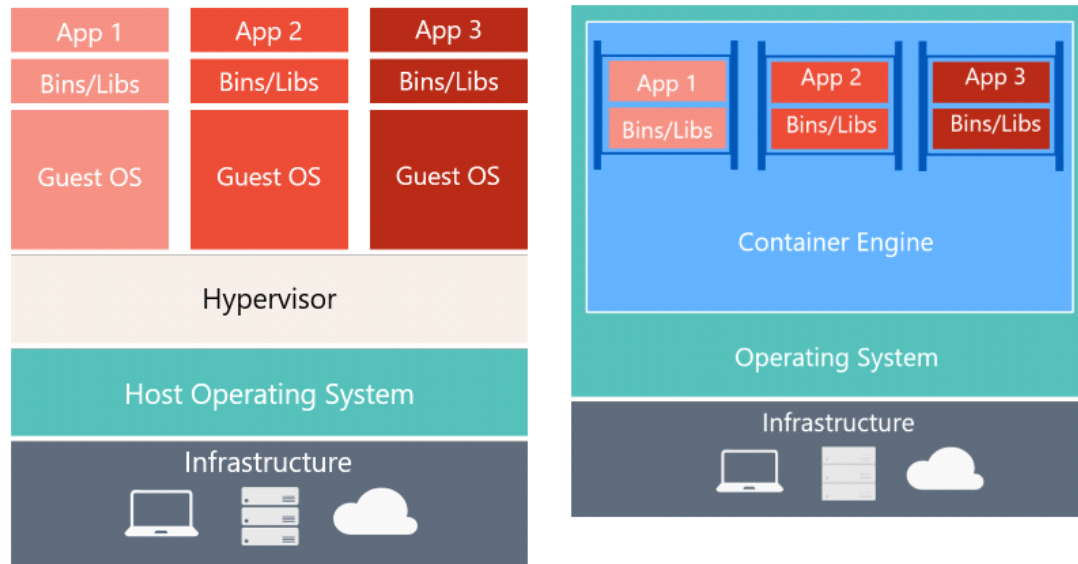
微服务优点：长期灵活性，微服务在复杂、大型和高度可缩放的系统中支持更好的可维护性，微服务的另外一大优势是，可以独立横向扩展。

其它投入：服务太多管理难，对服务和基础结构的监视和运行状况检查，可缩放基础结构，多个级别的安全设计和实现，快速应用程序交付，DevOps 和 CI/CD 实践和基础结构。

## Inner-Loop development workflow for Docker apps



## 认识虚拟机与容器技术



## 系统需求

### Windows 平台

硬件要求：16GB内存，由于 Docker 社区版需要 Hyper-V 才能运行 Linux Docker Host，并且还要使用 SQL Server 容器和 Reids 容器，8GB可能太小。

软件要求：Windows 10 Pro 64 位 + 启用Hyper-V + Docker 社区版 + .NET Core 2.2 SDK + 可选 Visual Studio 2017 15.8 以上

<https://docs.docker.com/docker-for-windows/install>

### MAC 平台

硬件要求：16GB内存，Mac 上使用 Linux Docker 主机运行 VM，并且还要使用 SQL Server 容器和 Reids 容器，8GB可能太小。

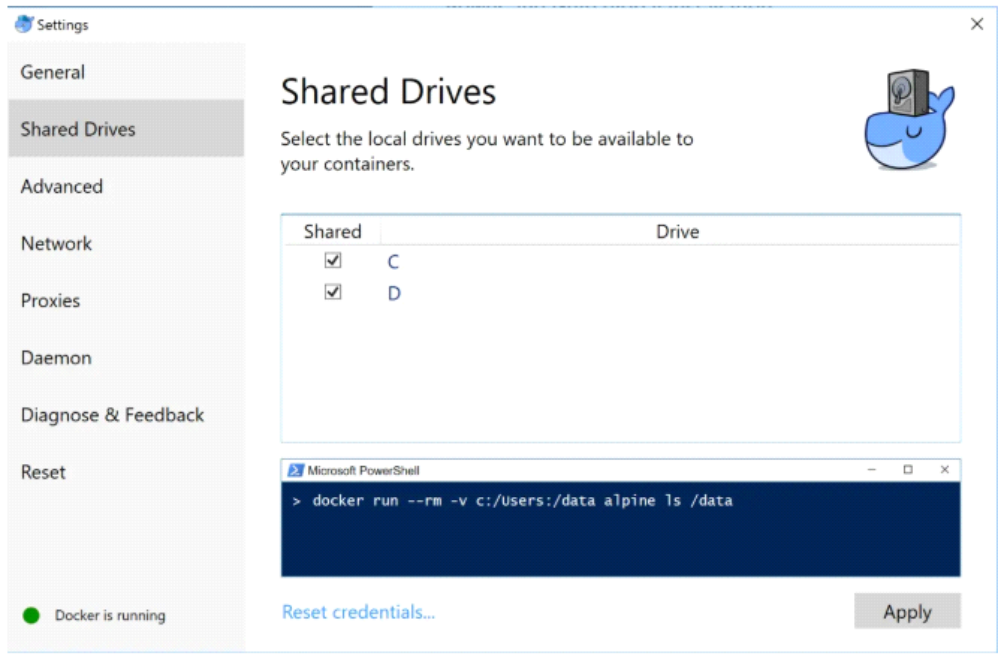
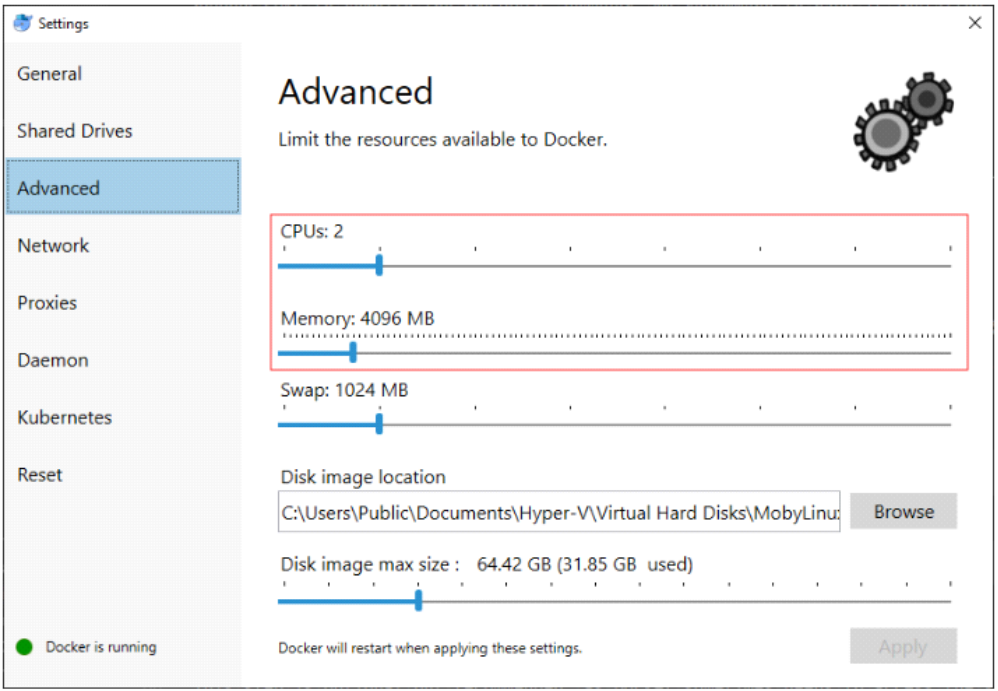
软件要求：OS X El Capitan 10.11 + 启用 Hyper-V + Docker 社区版 + .NET Core 2.2 SDK + 可选 Visual Studio for Mac

<https://docs.docker.com/docker-for-mac/install>

## 在 Windows 平台上运行源码

### 配置 Docker 容器

内存 4096MB + CPU2



共享驱动器：从 CLI 进行构建时，这实际上不是必需的，但在从 Visual Studio 进行构建以访问要构建的代码时，则必须这样做。

### 配置网络防火墙

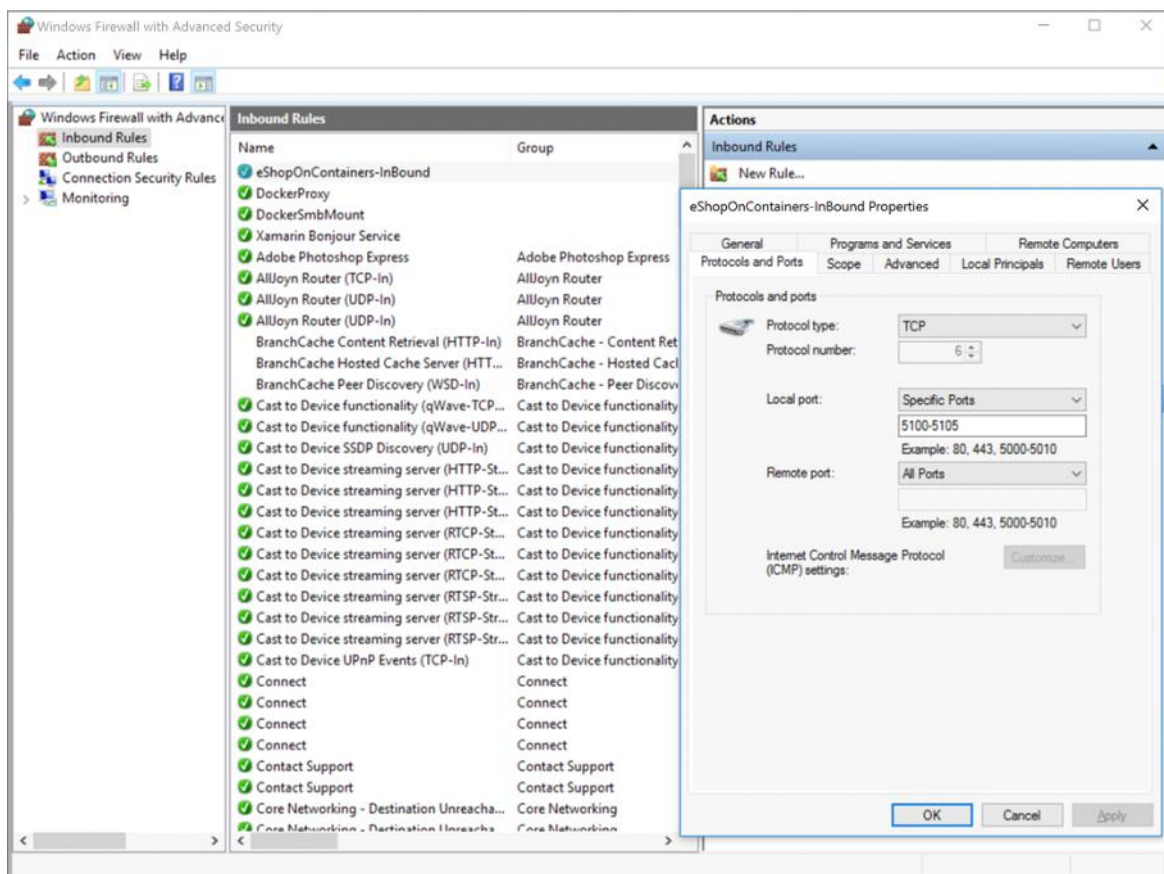
使用脚本打开防火墙：add-firewall-rules-for-sts-auth-thru-docker.ps1

遇到错误

```
管理员: Windows PowerShell
Windows PowerShell
版权所有 (C) Microsoft Corporation。保留所有权利。

无法加载文件 C:\Users\Cloud\source\repos\eShopOnContainers\deploy\windows\add-firewall-rules-for-sts-auth-thru-docker.ps1，因为在此系统上禁止运行脚本。有关详细信息，请参阅 https://go.microsoft.com/fwlink/?LinkID=135170 中的 about_Execution_Policies。
+ CategoryInfo          : SecurityError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : UnauthorizedAccess
```

执行: Set-ExecutionPolicy RemoteSigned

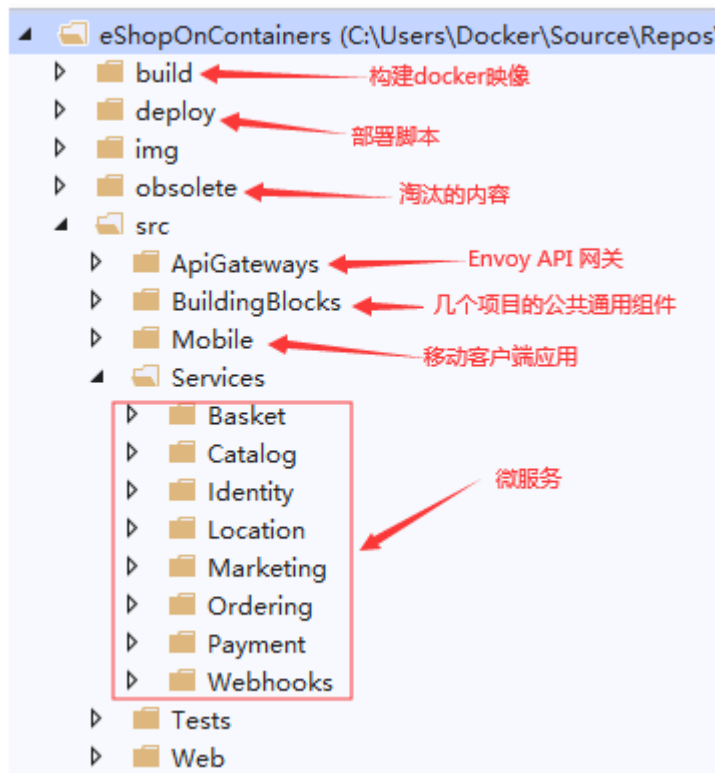


构建和部署源码

克隆项目到本地

git clone <https://github.com/dotnet-architecture/eShopOnContainers.git>

项目结构说明



## 构建生成应用程序

```
cd eShopOnContainers/src
docker-compose build
```

该过程需要10到30分钟才能完成，具体取决于系统速度。

## 部署到 Docker 主机容器

```
docker-compose up
```

导航到: <http://localhost:5107> 查看服务健康状况, 如果全为绿色启动 <http://localhost:5100>

## 探索应用程序

### Web apps

Web MVC: <http://localhost:5100>

Web SPA: <http://localhost:5104>

Web Status: <http://localhost:5107>

### Microservices

Catalog microservice: <http://localhost:5101> (Not secured)

Ordering microservice: <http://localhost:5102> (Requires login - Click on Authorize button)

Basket microservice: <http://localhost:5103> (Requires login - Click on Authorize button)

Identity microservice: <http://localhost:5105> (View "discovery document")

### Infrastructure

SQL Server (connect with SSMS to tcp:localhost,5433 with User Id=sa;Password=Pass@word; and explore databases:

Identity: Microsoft.eShopOnContainers.Service.IdentityDb  
Catalog: Microsoft.eShopOnContainers.Services.CatalogDb  
Marketing: Microsoft.eShopOnContainers.Services.MarketingDb  
Ordering: Microsoft.eShopOnContainers.Services.OrdeingDb  
Webhooks: Microsoft.eShopOnContainers.Services.WebhooksDb

Redis (Basket data): install and run redis-commander and explore in <http://localhost:8081/>  
RabbitMQ (Queue management): <http://10.0.75.1:15672/> (login with username=guest, password=guest)  
Seq (Logs collector): <http://10.0.75.1:5340>

<https://github.com/dotnet-architecture/eShopOnContainers/wiki/Windows-setup>

## 在 MAC 平台上运行源码

<https://github.com/dotnet-architecture/eShopOnContainers/wiki/Mac-setup>

## Docker 国内镜像

<https://www.cnblogs.com/noxy/p/11498507.html>

[Docker容器镜像常用命令](#)