

第04期-使用CLI命令行创建分层架构

2020年7月19日 8:48

安装 ABP CLI 工具

```
dotnet tool install -g Volo.Abp.Cli  
dotnet tool update -g Volo.Abp.Cli
```

使用 .NET Core CLI 安装和使用 .NET Core 全局工具

使用 ABP CLI 使用模板创建一个默认项目

```
abp new BookStore -t app -d mongodb -u angular --separate-identity-server
```

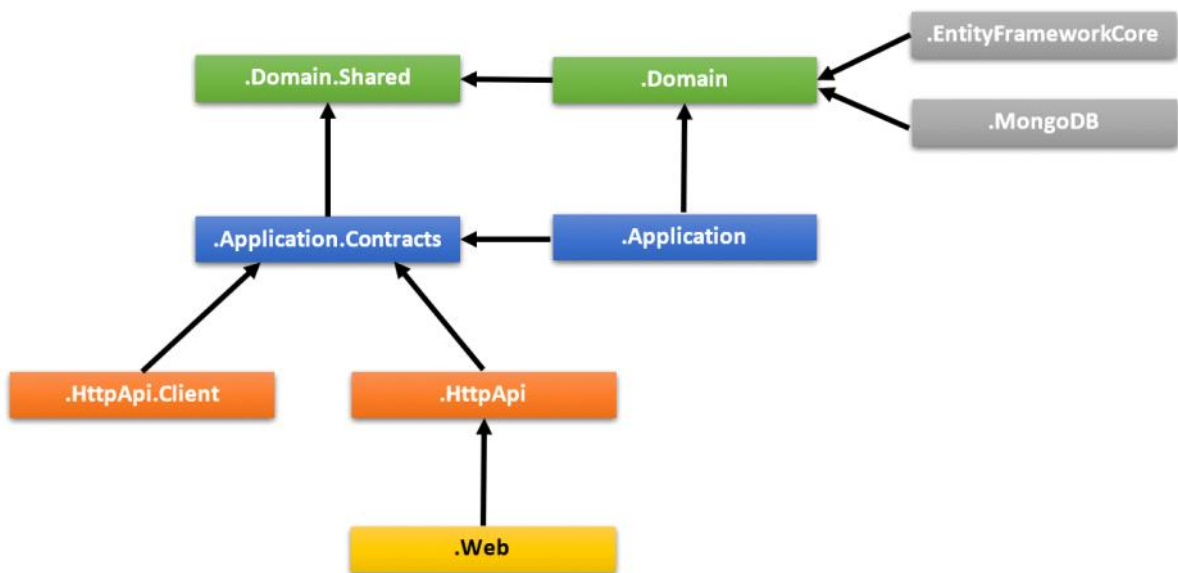
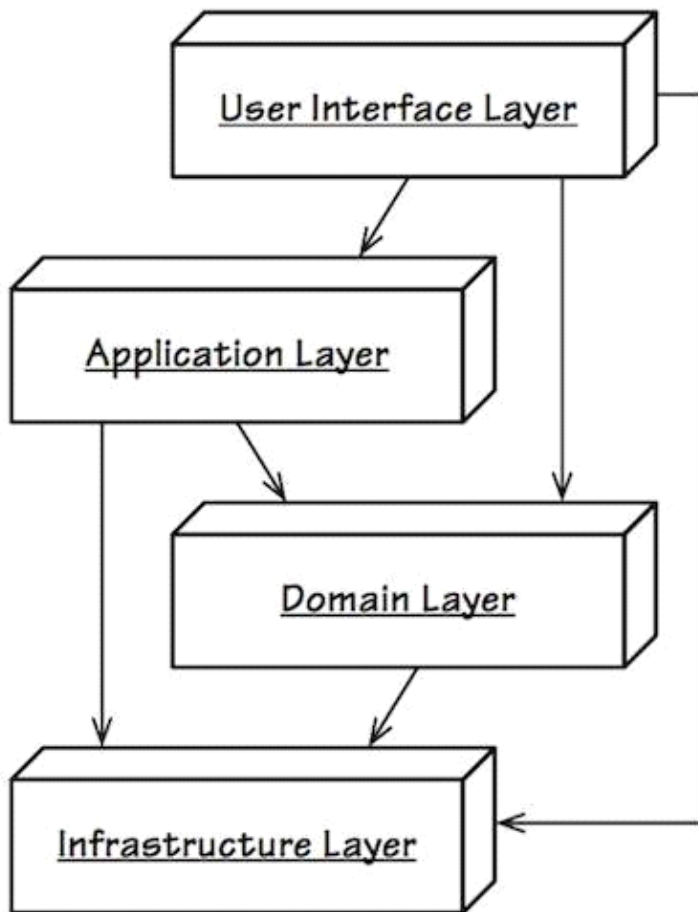
-t 参数指定 启动模板 名称. app 是一个启动模板名称,包含了预安装并且配置好的ABP模块。

-u 指定UI框架, 本例中是 angular。

--separate-identity-server 参数用于将Identity服务器应用程序与API主机应用程序分隔开. 如果未指定,你将只有一个端点.

使用 -d (或 --database-provider) 选项指定数据库提供程序。

默认创建一个 DDD 分层架构, 包括单元测试和集成测试, 界面使用 Razor 视图, 可选择数据库和所使用的 UI 方式, 数据库已为您配置 EF Core & SQLite 数据库。



修改数据库连接字符串

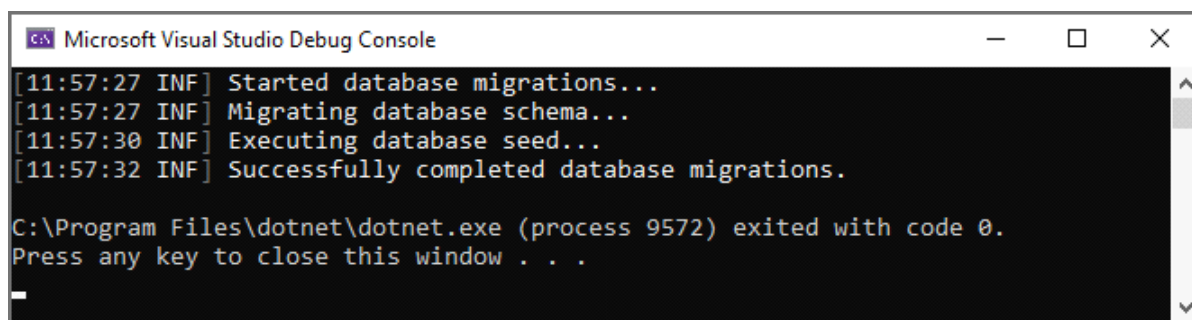
查看 .Web 项目下 appsettings.json 文件中的连接字符串

```
{
  "ConnectionStrings": {
```

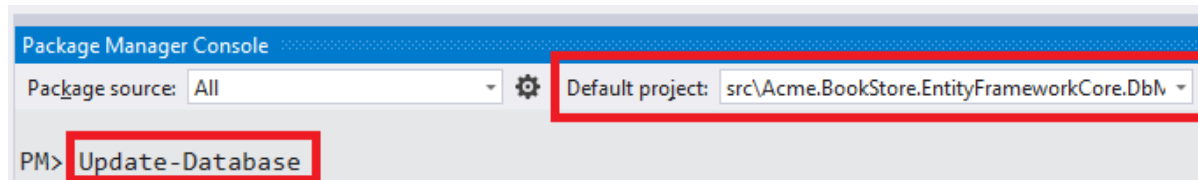
```
        "Default": "Data Source=(localdb)\\MSSQLLocalDB;Initial Catalog=BookStore"
    }
}
```

使用 DbMigrator 应用程序应用迁移

方案1: .DbMigrator 项目有自己的 appsettings.json, 因此, 如果你更改了上面的连接字符串, 则还应更改此字符串, 右键单击 .DbMigrator 项目并选择[设置为启动项目], 运行起来, 即可执行迁移流程。



方案2: 打开包管理器控制台, 选择 .EntityFrameworkCore.DbMigrations 项目作为默认项目并运行 Update-Database 命令。



官方推荐您使用方案1的方式。

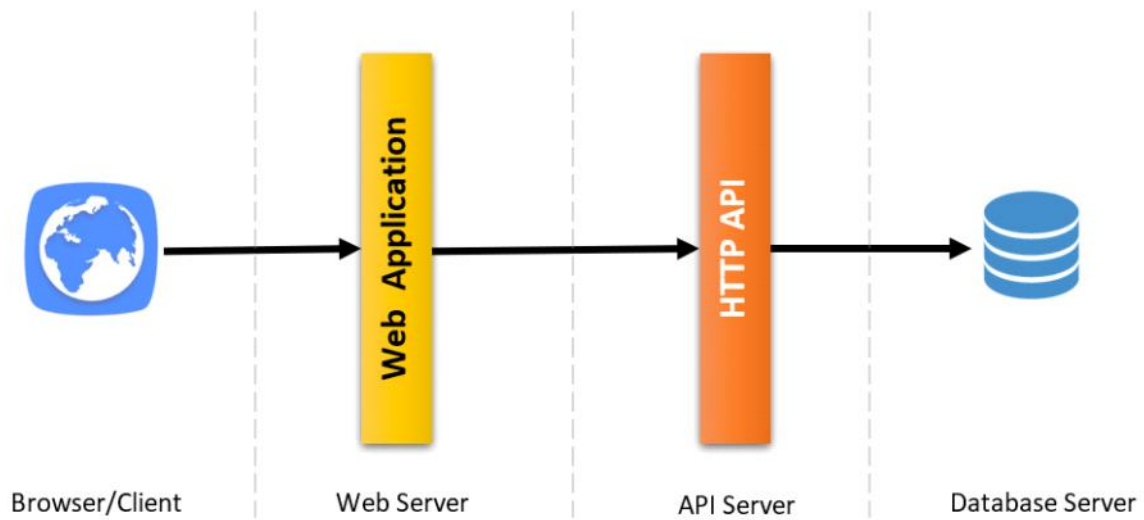
运行应用程序

最后确保 .Web 是启动项目, 运行应用程序后会在你的浏览器打开一个登录页面。

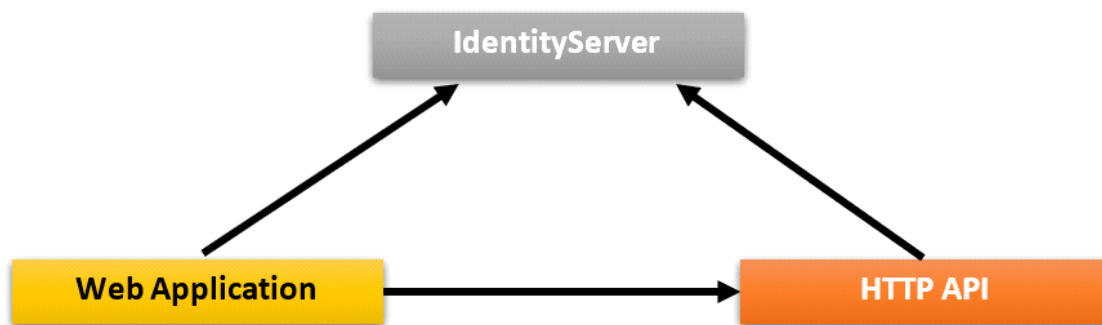
输入用户名 admin, 密码 1q2w3E* 登录到应用程序。

web 与 API 分层架构

如果你选择了 ASP.NET Core UI 并指定了 --tiered 选项, CLI 会创建分层解决方案, 分层结构的目的是将 Web 应用程序和 HTTP API 部署到不同的服务器。



如果使用 `--tiered` 选项，除此之外，我们会得到两个新的项目：`.IdentityServer` 和 `.HttpApi.Host`，有自己的数据库连接字符串。



参考资料

[从启动模板开始创建一个 DDD 项目](#)

[应用程序启动模板](#)

[ABP CLI 工具参数帮助文档](#)

[使用官方向导创建项目基架](#)