**ABP架构**

#### 客户应用

这些是通过HTTP API（API控制器，[OData](https://aspnetboilerplate.com/Pages/Documents/OData-Integration)控制器，甚至可能是GraphQL端点）将应用程序用作服务的远程客户端。远程客户端可以是SPA（单页面应用程序），移动应用程序或第三方消费者。可以在此应用程序内完成[本地化](https://aspnetboilerplate.com/Pages/Documents/Localization)和 [导航](https://aspnetboilerplate.com/Pages/Documents/Navigation)。

#### 表达层

ASP.NET [Core] MVC（模型 - 视图 - 控制器）可以被认为是表示层。它可以是物理层（通过HTTP API使用应用程序）或逻辑层（直接注入和使用[应用程序服务](https://aspnetboilerplate.com/Pages/Documents/Application-Services)）。在任何一种情况下，它都可以包括 [本地化](https://aspnetboilerplate.com/Pages/Documents/Localization)，[导航](https://aspnetboilerplate.com/Pages/Documents/Navigation)， [对象映射](https://aspnetboilerplate.com/Pages/Documents/Object-To-Object-Mapping)， [缓存](https://aspnetboilerplate.com/Pages/Documents/Caching)，[配置管理](https://aspnetboilerplate.com/Pages/Documents/Setting-Management)，[审计日志记录](https://aspnetboilerplate.com/Pages/Documents/Audit-Logging)等。它还涉及 [授权](https://aspnetboilerplate.com/Pages/Documents/Authorization)，[会话](https://aspnetboilerplate.com/Pages/Documents/Abp-Session)， [功能](https://aspnetboilerplate.com/Pages/Documents/Feature-Management)（用于 [多租户](https://aspnetboilerplate.com/Pages/Documents/Multi-Tenancy)应用程序）和[异常处理](https://aspnetboilerplate.com/Pages/Documents/Handling-Exceptions)。

#### 分布式服务层

该层用于通过REST，OData，GraphQL等远程API提供应用程序/域功能......它们不包含业务逻辑，只是将HTTP请求转换为域交互，或者可以使用应用程序服务来委派操作。该层通常包括[授权](https://aspnetboilerplate.com/Pages/Documents/Authorization)，[缓存](https://aspnetboilerplate.com/Pages/Documents/Caching)， [审计日志](https://aspnetboilerplate.com/Pages/Documents/Audit-Logging)，[对象映射](https://aspnetboilerplate.com/Pages/Documents/Object-To-Object-Mapping)，[异常处理](https://aspnetboilerplate.com/Pages/Documents/Handling-Exceptions)，[会话](https://aspnetboilerplate.com/Pages/Documents/Abp-Session)等...

#### 应用层

应用层主要包括[应用服务的](https://aspnetboilerplate.com/Pages/Documents/Application-Services)使用领域层和领域对象（[域服务](https://aspnetboilerplate.com/Pages/Documents/Domain-Services)， [实体](https://aspnetboilerplate.com/Pages/Documents/Entities) ...）来执行请求的应用程序的功能。它使用[数据传输对象](https://aspnetboilerplate.com/Pages/Documents/Data-Transfer-Objects)从表示或分布式服务层获取数据并将数据返回。它还可以处理 [授权](https://aspnetboilerplate.com/Pages/Documents/Authorization)，[缓存](https://aspnetboilerplate.com/Pages/Documents/Caching)，[审计日志](https://aspnetboilerplate.com/Pages/Documents/Audit-Logging)，[对象映射](https://aspnetboilerplate.com/Pages/Documents/Object-To-Object-Mapping)，[会话](https://aspnetboilerplate.com/Pages/Documents/Abp-Session)等...

#### 域层

这是实现我们的域逻辑的主要层。它包括 [实体](https://aspnetboilerplate.com/Pages/Documents/Entities)，[值对象](https://aspnetboilerplate.com/Pages/Documents/Value-Objects)和[域服务，](https://aspnetboilerplate.com/Pages/Documents/Domain-Services)以执行业务/域逻辑。它还可以包括[规范](https://aspnetboilerplate.com/Pages/Documents/Specifications)和触发[域事件](https://aspnetboilerplate.com/Pages/Documents/EventBus-Domain-Events)。它定义了存储库接口，以便从数据源（通常是DBMS）读取和保留实体。

#### 基础设施层

基础结构层使其他层工作：它实现存储库接口（例如使用[Entity Framework Core](https://aspnetboilerplate.com/Pages/Documents/Entity-Framework-Core)）以实际使用真实数据库。它还可能包括与供应商集成以[发送电子邮件](https://aspnetboilerplate.com/Pages/Documents/Email-Sending)等。这不是所有图层下面的严格图层，但实际上通过实现它们的抽象概念来支持其他图层。