

宁波DOTNET社区(NBDNC)线下交流分享会

MAY 2018

1. DotNet开发框架ABP介绍
2. Azure云介绍及演示

DOTNET开发框架ABP 介绍

BY 王伯承

目录

1. 自我介绍
2. ABP框架介绍
3. ABP实践经验分享
4. ABP的成功案例

I. 自我介绍

- 王伯承(personball), 2010年起从事dotnet开发, 至今八年多 ;
- 前五年在宁波工作, 经历过八六人才网、土拨鼠、浙江大道集团 ;
- 2015年8月到上海, 跟随国内最早ABP开发框架的引入者——罗阳铭 (<http://www.cnblogs.com/mienreal/>), 先后就职“要买车”和“健康724”。积累丰富的ABP框架实践经验, 并根据实践经验开源ABP扩展库github.com/personball/abplus ;
- 个人博客 : <http://personball.com>

2.ABP框架介绍



企业级开发框架需要什么？

基础框架组件的功能：

- 1、基础框架组件独立、通用，可用于多个不同项目。类似于daxnet的Apworks框架。
- 2、对项目实现模块化开发提供了支持，每个模块有独立的EF DbContext，可单独指定数据库。
- 3、对DDD的技术实现进行了封装，让项目以极精简的代码，专注于业务领域。
- 4、多租户支持，每个租户的数据自动隔离，业务模块开发者不需要手动操作TenantId。
- 5、集成ASP.NET Identity，实现登录认证、功能权限授权&验证、角色和用户管理。
- 6、集成Log4Net，实现日志记录。
- 7、集成AutoMapper，实现Dto类与实体类的双向自动转换。
- 8、实现UnitOfWork模式，为应用层和仓储层的(会写数据库的)方法自动实现数据库事务。
- 9、可通过ApplicationService的方法自动建立相应的WebApi方法，ajax可直接调用，不需要写ApiController和Action。
- 10、调用ApplicationService的方法时，自动验证权限和参数有效性(用相应的Attribute标注)。
- 11、继承自FullAuditedEntity基类的领域实体，会自动实现软删除（在数据库中用IsDeleted字段进行标注）。
- 12、实现一系列扩展方法，简化编码。

2.1 ABP是什么？

- ABP是“ASP.NET Boilerplate Project (ASP.NET样板项目)”的简称。
- 开源时间：2014年5月4日发布在 Github
- 整个2017年作者全职做ABP框架后，版本升得飞快

时间	Commits	Releases	版本号
2017年3月7日	-	-	V1.5.0
2017年4月7日	3023	124	V2.0.0
2018年4月16日	4612	163	V3.5.0

2.1 ABP是什么？

- ASP.NET Boilerplate 是一个用最佳实践和流行技术开发现代WEB应用程序的新起点.
- 它旨在成为一个通用的WEB应用程序框架和项目模板。
- ASP.NET Boilerplate 基于DDD的经典分层架构思想，实现了众多DDD的概念（但没有实现所有DDD的概念）。
- 官网：<https://www.aspnetboilerplate.com>
- github.com/aspnetboilerplate

 Watch ▾

717

 Unstar

4,503

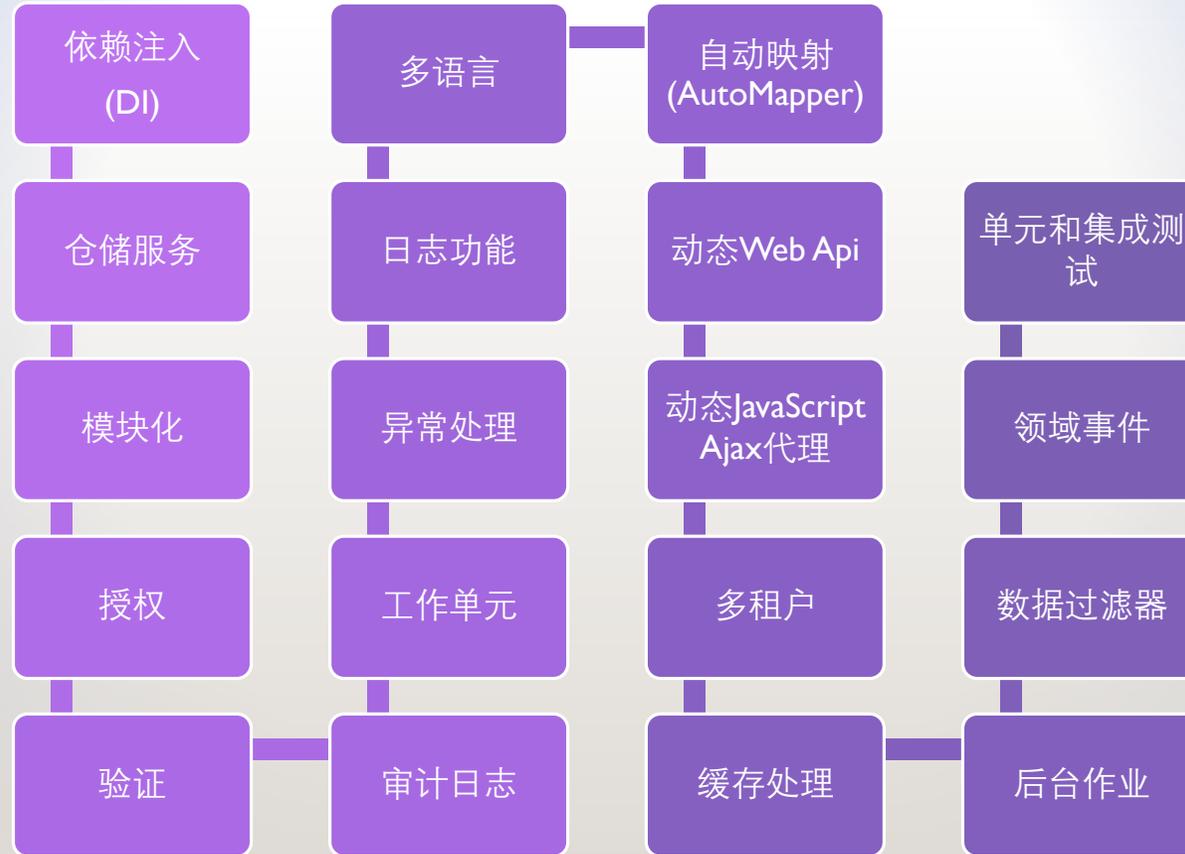
 Fork

2,013

2.2ABP框架的理念

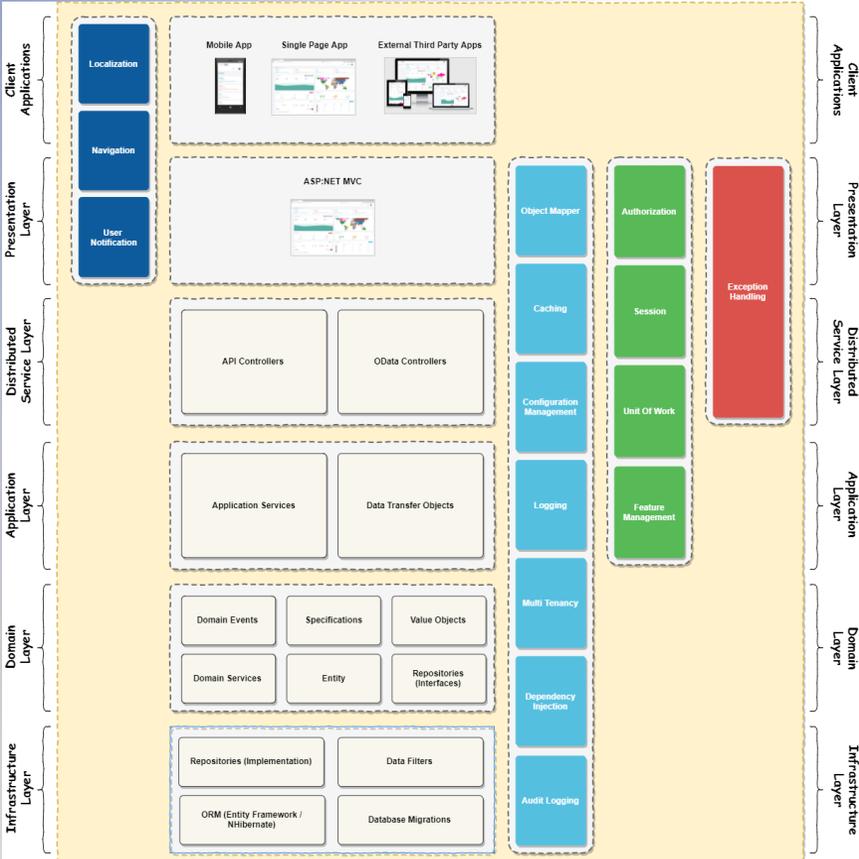


2.3 ABP提供的现成机制



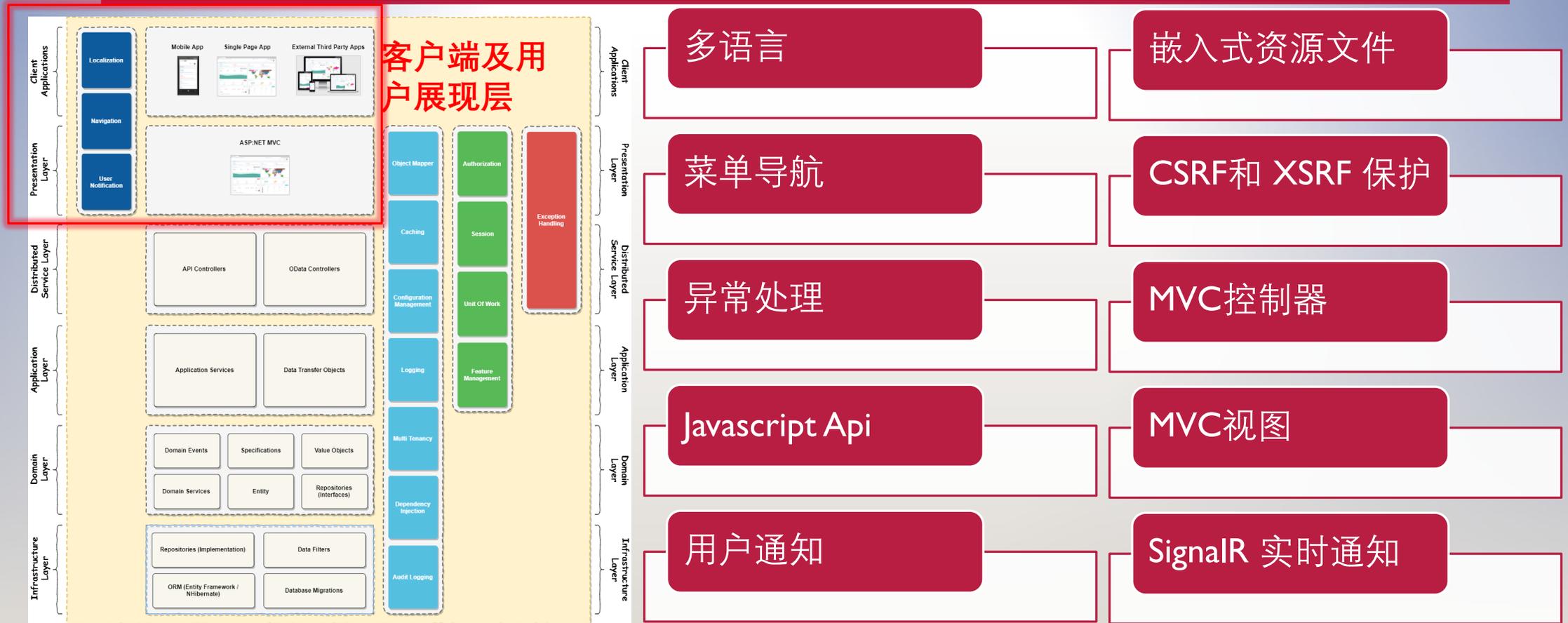
DRY :
Don't Repeat Yourself !

2.4ABP的分层架构

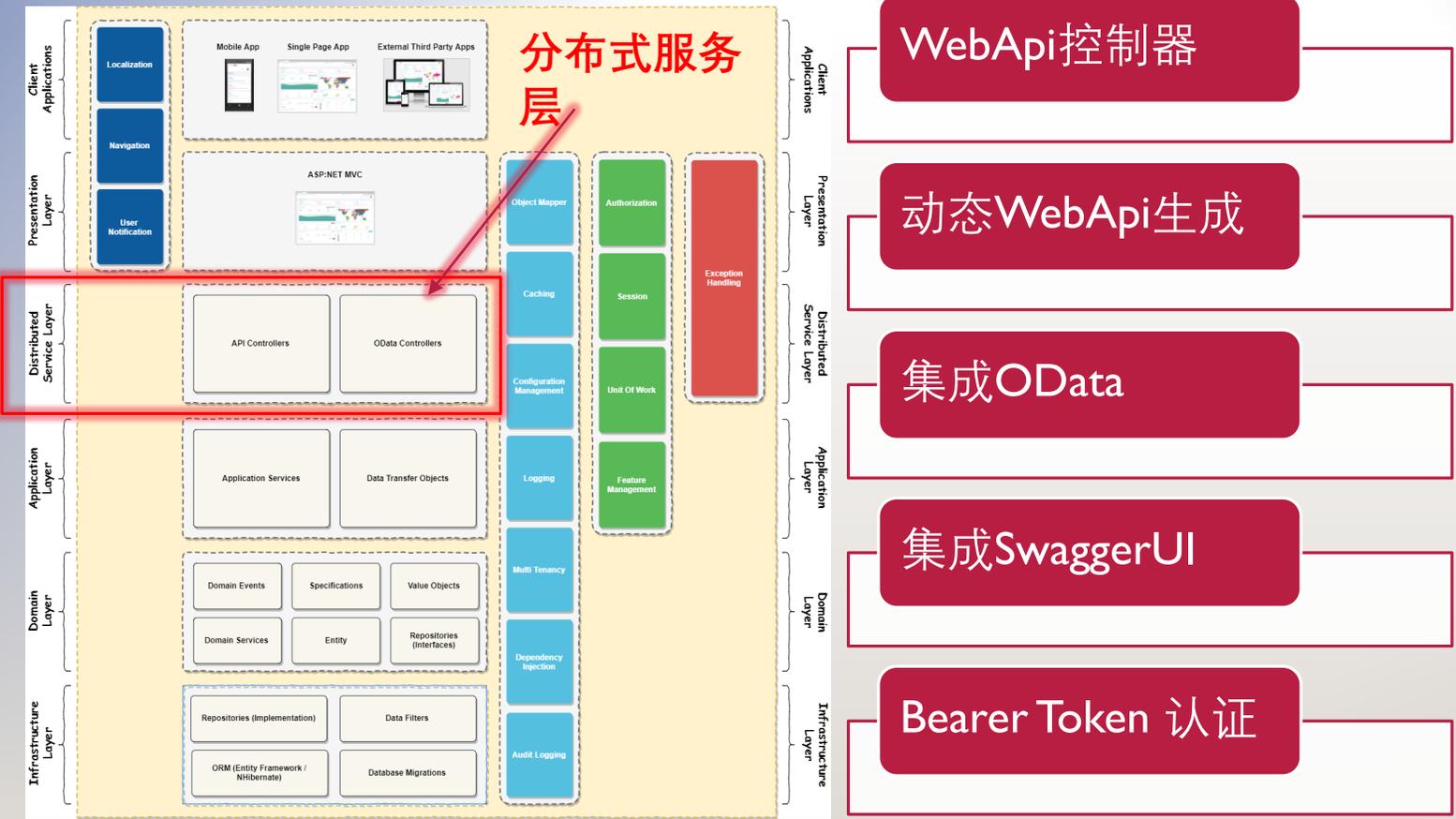


Layers	
Presentation	View Models (Javascript) Views (HTML/CSS) Localization
Web	Web API Controllers MVC Controllers
Application	Application Services DTOs DTO Mappers
Domain	Entities Value Objects Repositories Business Classes Domain Services Unit of Work Domain Events
Infrastructure	ORM DB Migrations
Others	
Server Side	Dependency Injection Logging

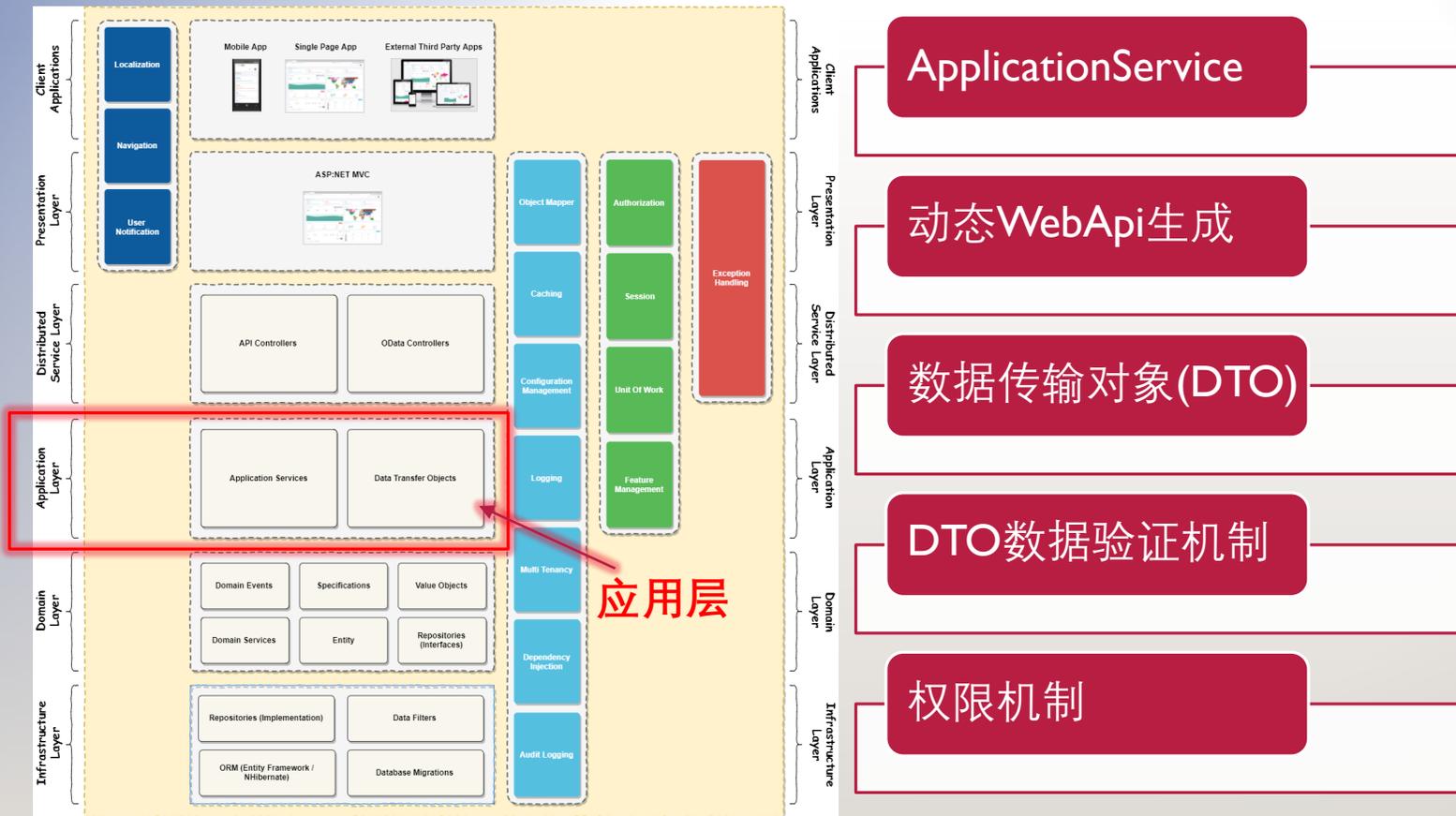
2.4ABP的分层架构



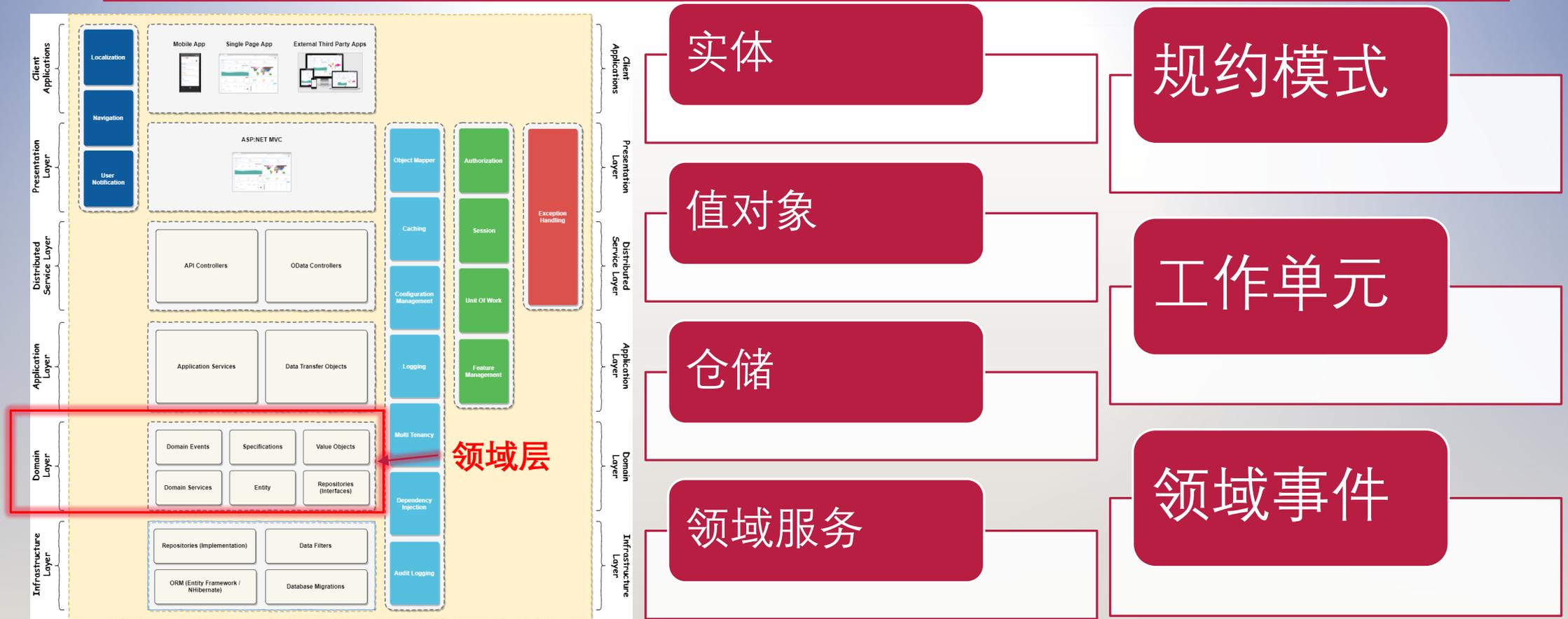
2.4ABP的分层架构



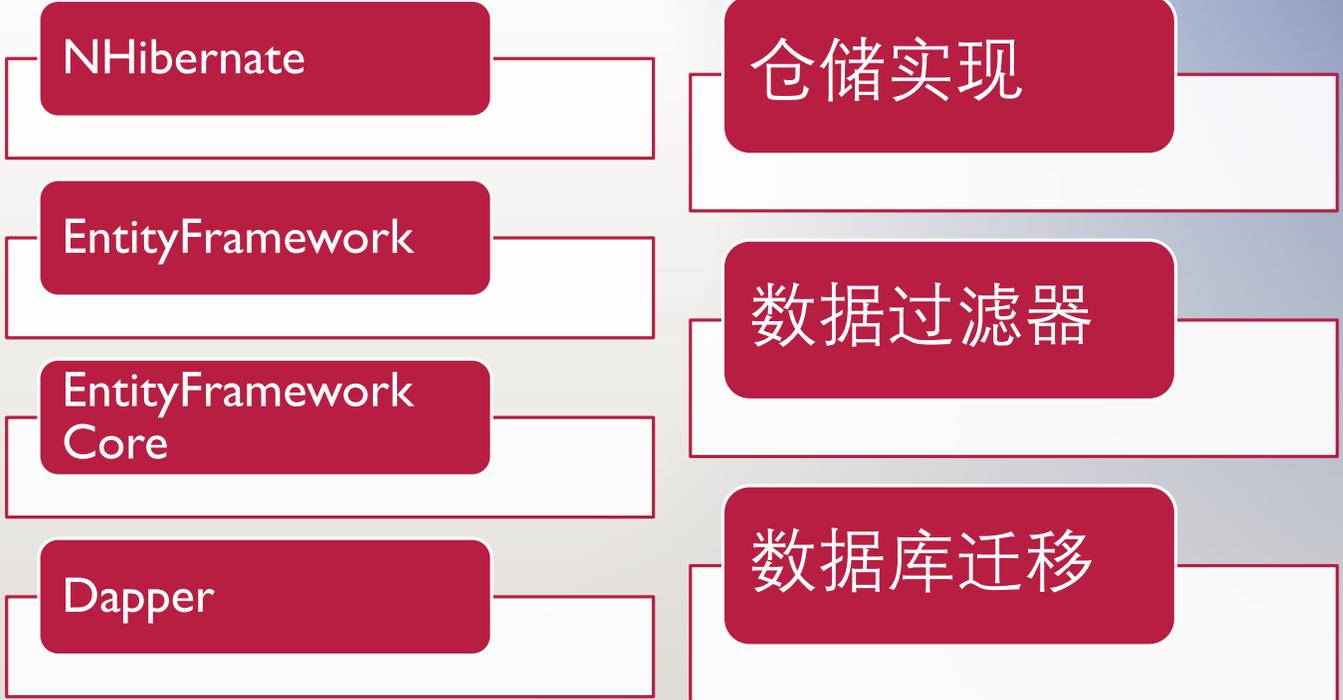
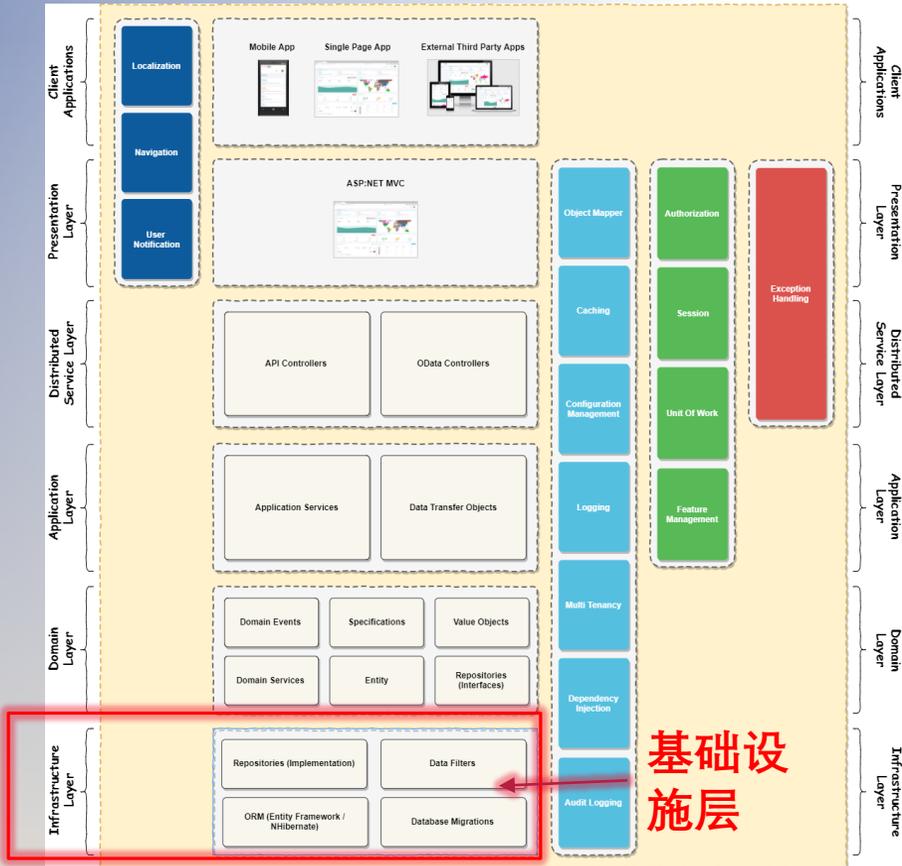
2.4ABP的分层架构



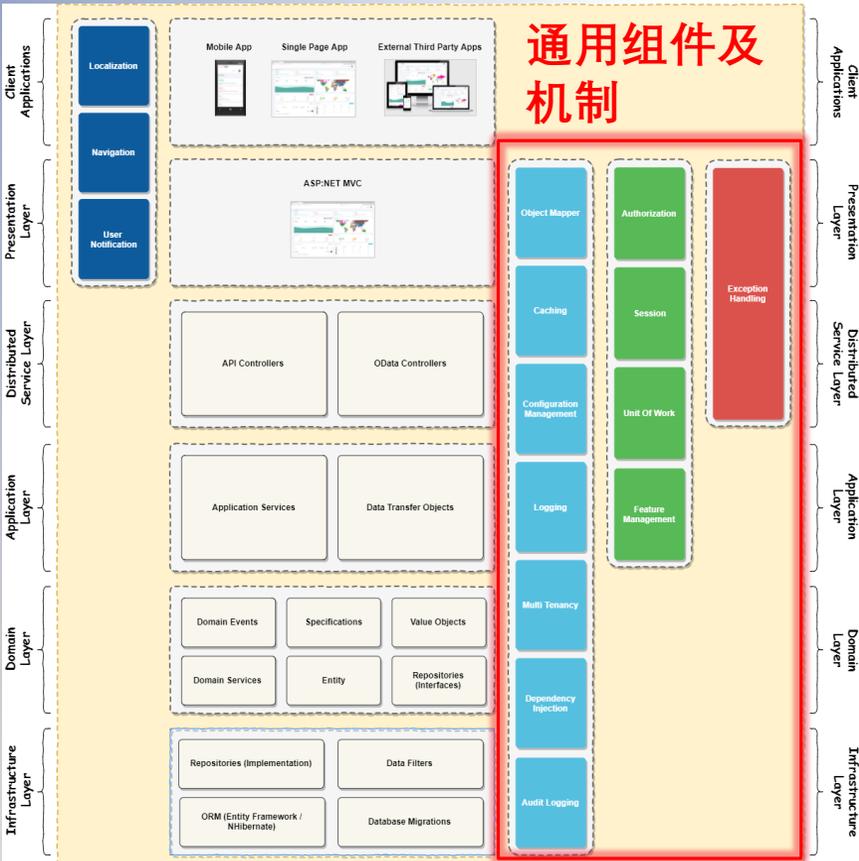
2.4ABP的分层架构



2.4ABP的分层架构

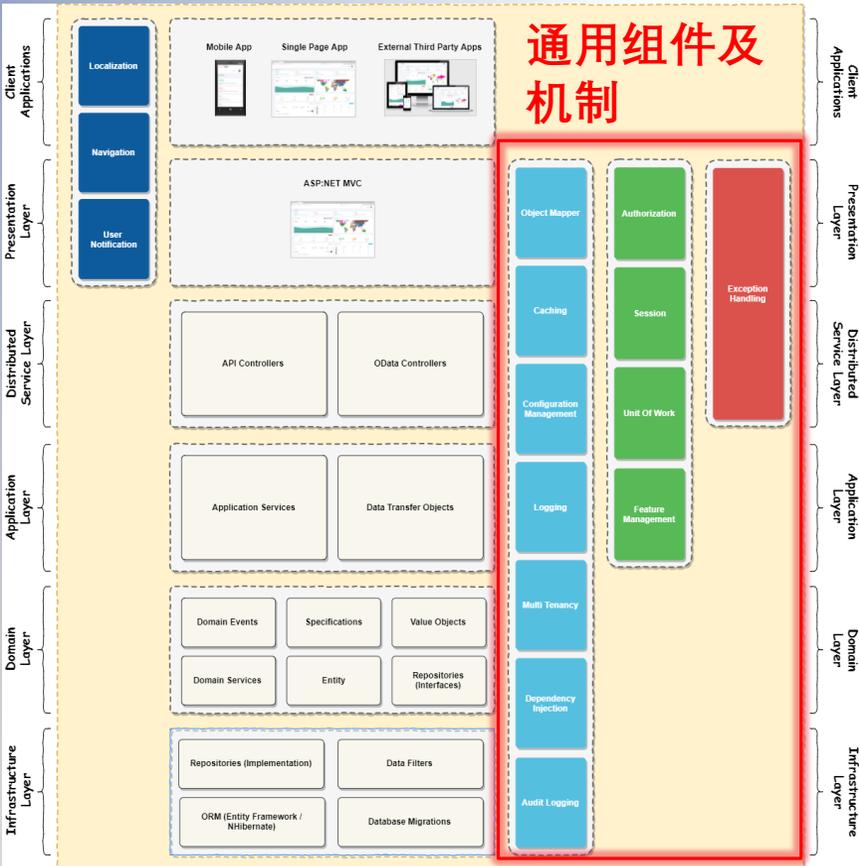


2.4ABP的分层架构

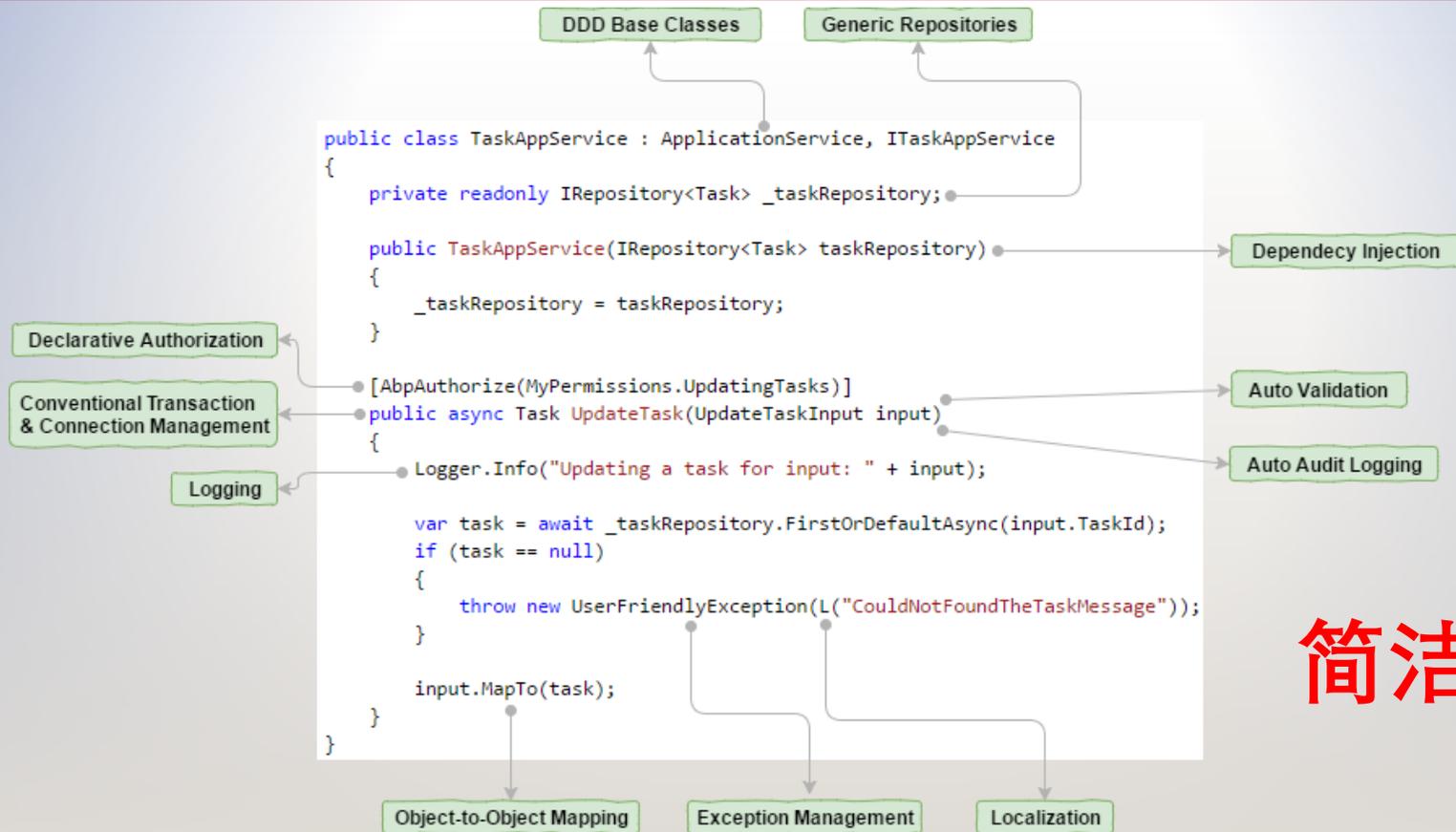


- 对象映射
- 缓存
- 配置管理
- 日志
- 多租户
- 依赖注入
- 异常处理
- 认证授权
- Session
- 工作单元
- Feature管理 (功能版本)
- 审计日志

2.4ABP的分层架构



2.4ABP的分层架构

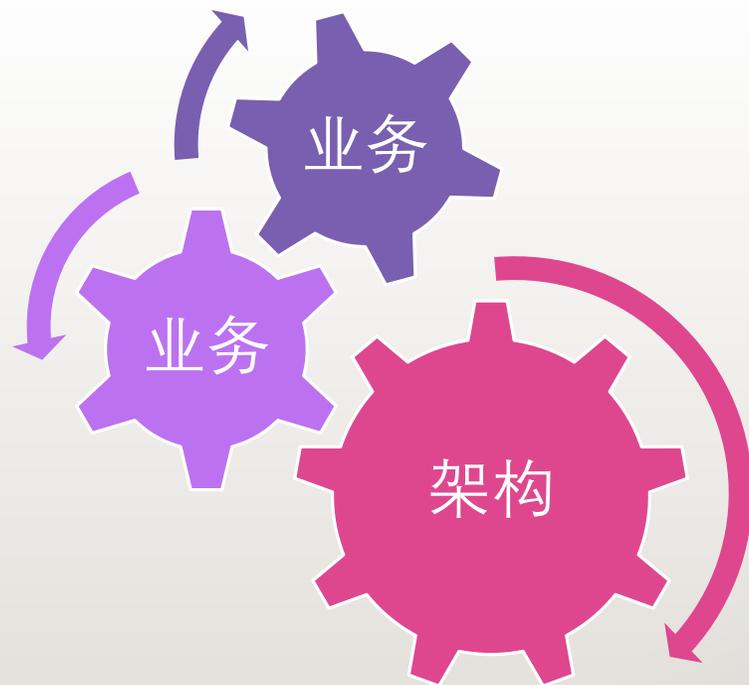


简洁！

2.5 ABP几个常见名词区分

- **ABP**（基础框架，通用，自由开源）
- **Module Zero**（提供了**ABP**所有基础概念的默认实现，用户、角色、权限、配置、语言、审计日志、**BackgroundJob**，特别是各种机制的默认存储实现）
- **Aspnet Zero**（作者用以支持**ABP**框架开发的，重点投入前端界面开发的收费版企业级启动项目模板）

3.ABP实践经验分享



3.1 架构方面——针对ABP进行了消息队列机制的扩展



基于Rebus.rabbitmq

重点在于：

- 1.在不同宿主中，我们可以统一跑一个框架，用同一份业务代码。
- 2.通过不同形式的部署（Web或Services）分别承担适合各自场景的任务，以灵活设计同步或异步的逻辑处理方案。

3.2原724项目介绍

事件

消息

可监控

可度量

容错、
重试

自建电商
系统

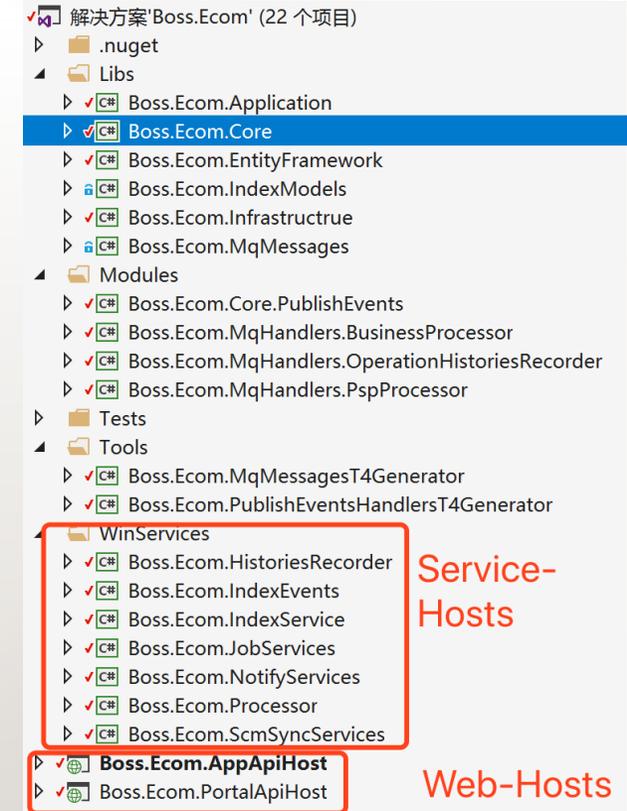
Ecom

自建供应
链系统

Scm

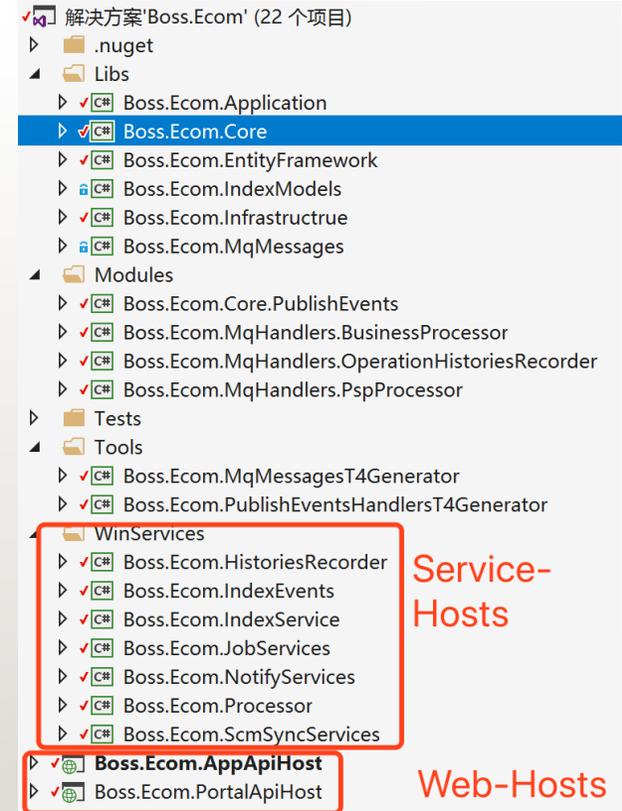
3.2原724项目介绍

Libs	Boss.Ecom.Application
	Boss.Ecom.Core
	Boss.Ecom.EntityFramework
	Boss.Ecom.IndexModels
	Boss.Ecom.Infrastructure
	Boss.Ecom.MqMessages



3.2原724项目介绍

Modules	Boss.Ecom.Core.PublishEvents
	Boss.Ecom.MqHandlers.BusinessProcessor
	Boss.Ecom.MqHandlers.OperationHistoriesRecorder
	Boss.Ecom.MqHandlers.PspProcessor



3.2原724项目介绍

Tools

Boss.Ecom.MqMessagesT4Generator

Boss.Ecom.PublishEventsHandlersT4Generator

解决方案'Boss.Ecom' (22 个项目)

- .nuget
- Libs
 - Boss.Ecom.Application
 - Boss.Ecom.Core**
 - Boss.Ecom.EntityFramework
 - Boss.Ecom.IndexModels
 - Boss.Ecom.Infrastructure
 - Boss.Ecom.MqMessages
- Modules
 - Boss.Ecom.Core.PublishEvents
 - Boss.Ecom.MqHandlers.BusinessProcessor
 - Boss.Ecom.MqHandlers.OperationHistoriesRecorder
 - Boss.Ecom.MqHandlers.PspProcessor
- Tests
- Tools
 - Boss.Ecom.MqMessagesT4Generator
 - Boss.Ecom.PublishEventsHandlersT4Generator
 - WinServices
 - Boss.Ecom.HistoriesRecorder
 - Boss.Ecom.IndexEvents
 - Boss.Ecom.IndexService
 - Boss.Ecom.JobServices
 - Boss.Ecom.NotifyServices
 - Boss.Ecom.Processor
 - Boss.Ecom.ScmSyncServices
 - Boss.Ecom.AppApiHost**
 - Boss.Ecom.PortalApiHost**

Service-Hosts

Web-Hosts

3.2原724项目介绍

WinServices

Boss.Ecom.HistoriesRecorder

Boss.Ecom.IndexEvents

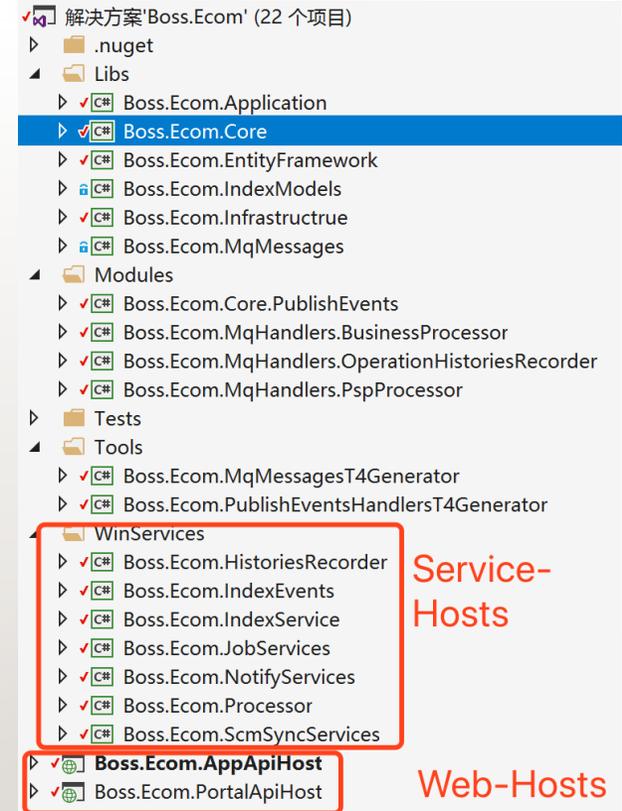
Boss.Ecom.IndexService

Boss.Ecom.JobServices

Boss.Ecom.NotifyServices

Boss.Ecom.Processor

Boss.Ecom.ScmSyncServices

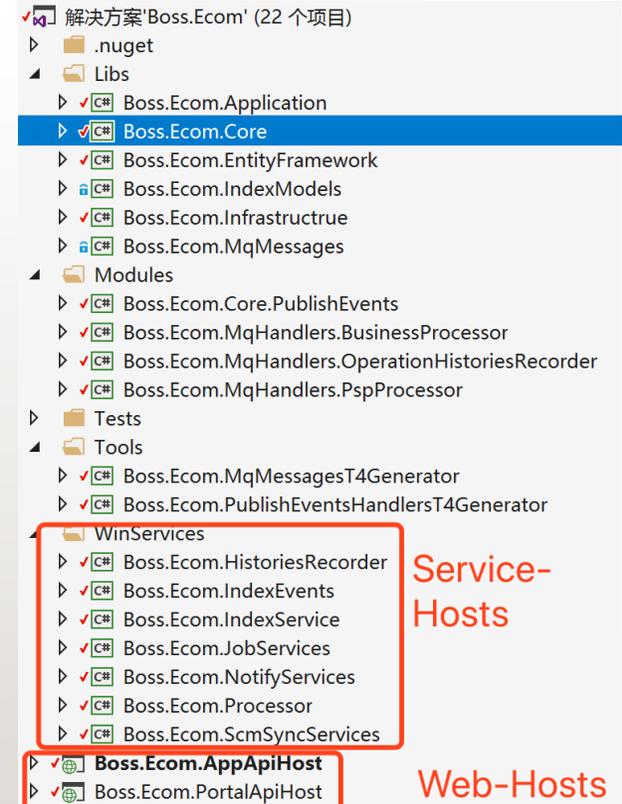


3.2原724项目介绍

Web(ApiHost)

Boss.Ecom.AppApiHost

Boss.Ecom.PortalApiHost



3.2原724项目介绍

ELK

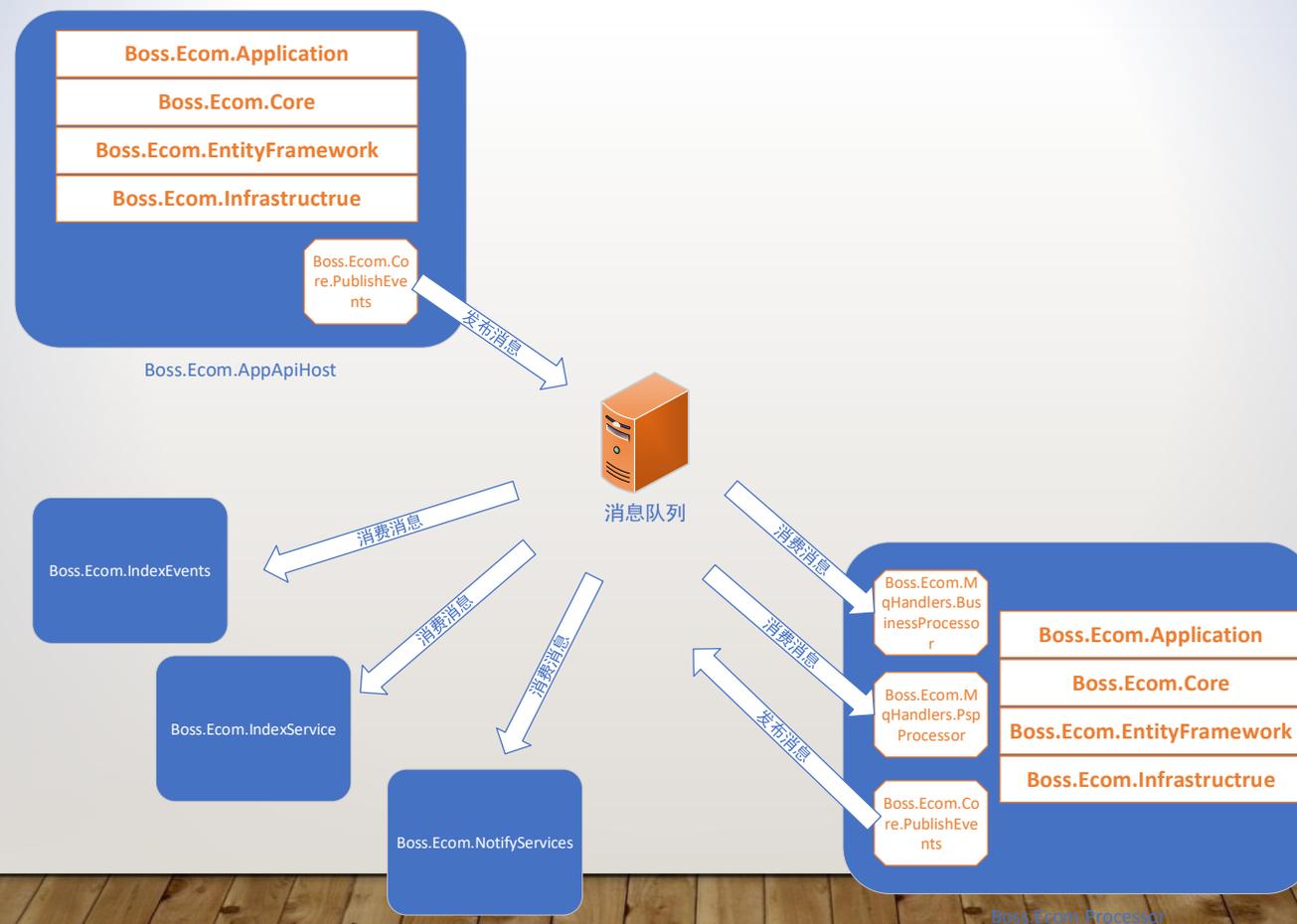
Abp审计日志收集，通过消息队列对接

应用日志收集，Nlog，filebeat

nginx访问日志收集，filebeat

H5异常日志收集，由js汇报，webapi中转到redis队列对接

3.2原724项目介绍



3.3 业务方面

业务逻辑代码放到
领域层



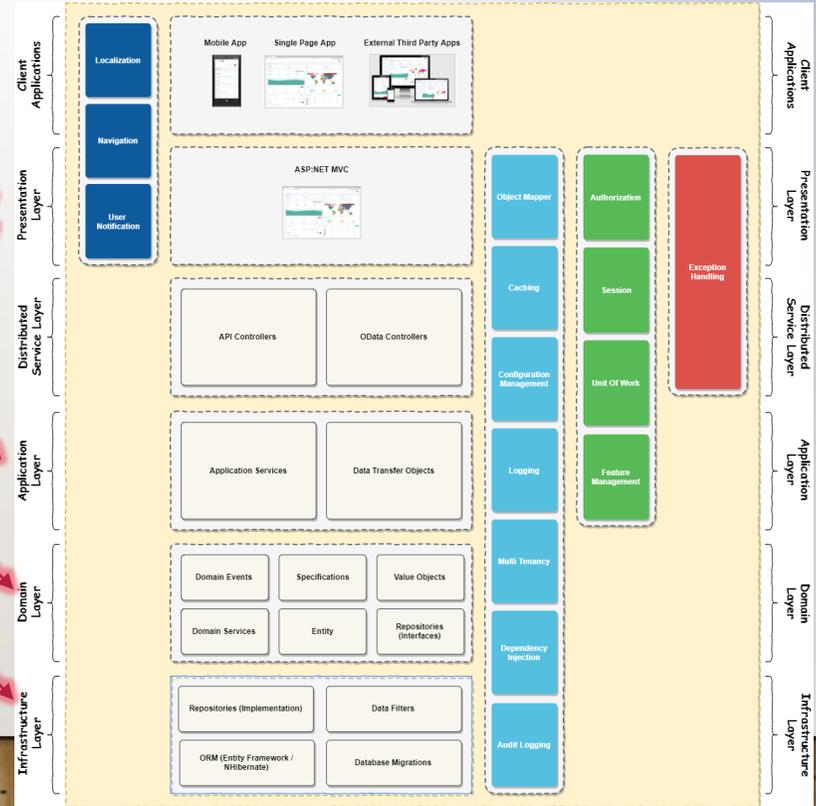
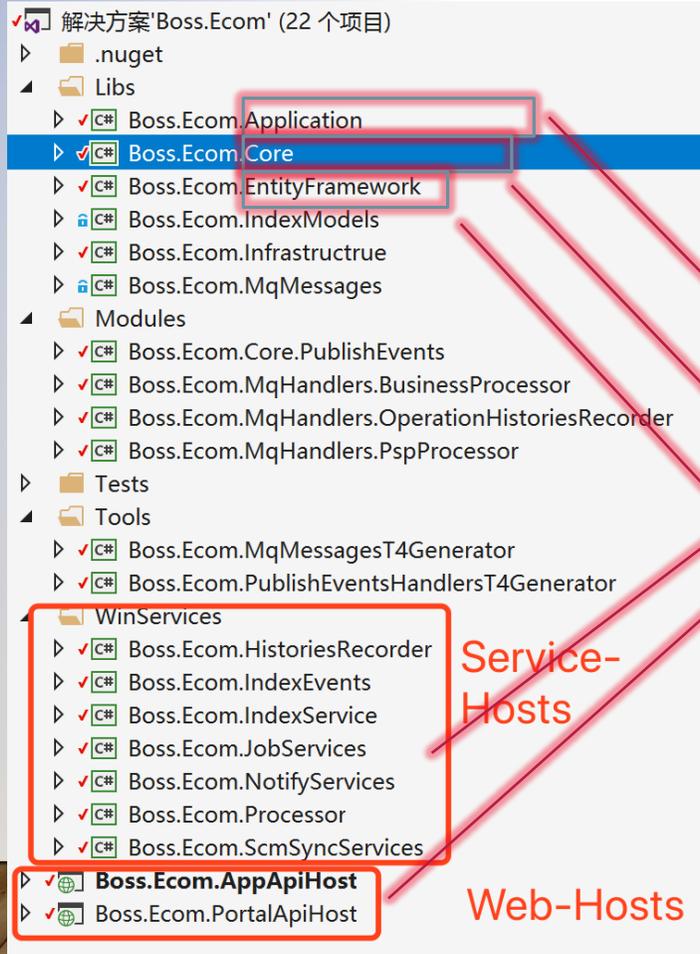
尽量给领域层一个
纯内存的执行环境



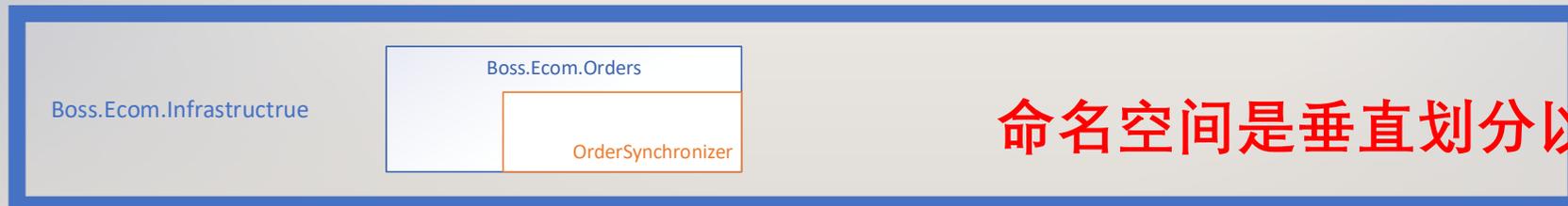
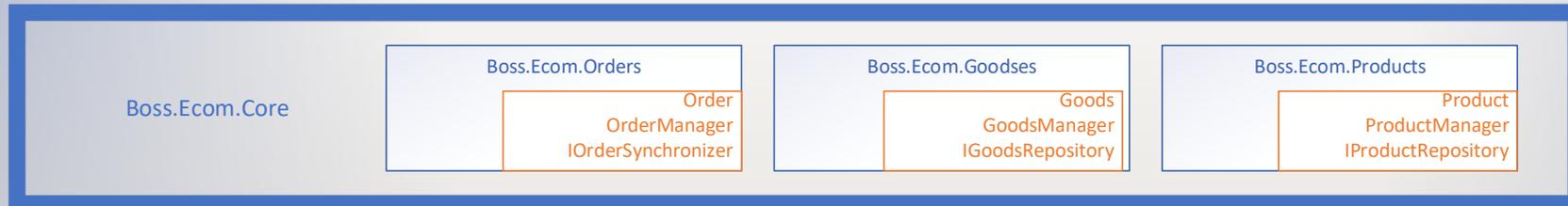
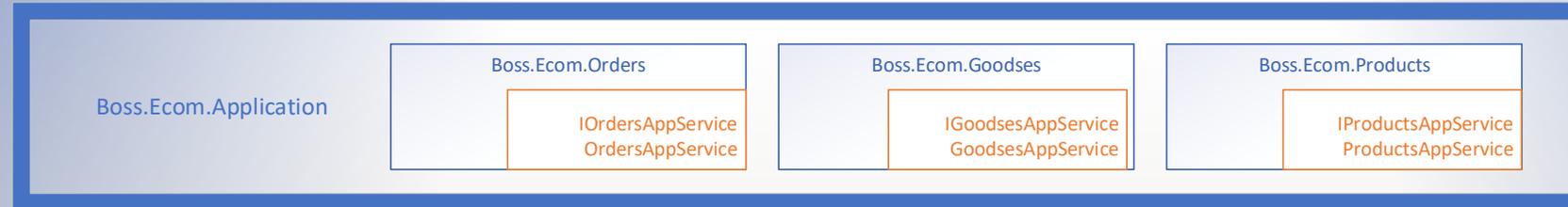
运用设计模式更好的
写出可维护可拓展的
业务代码（业务规则建模）

3.3.1 理解命名空间的作用

程序集命名体现了水平分层的层次

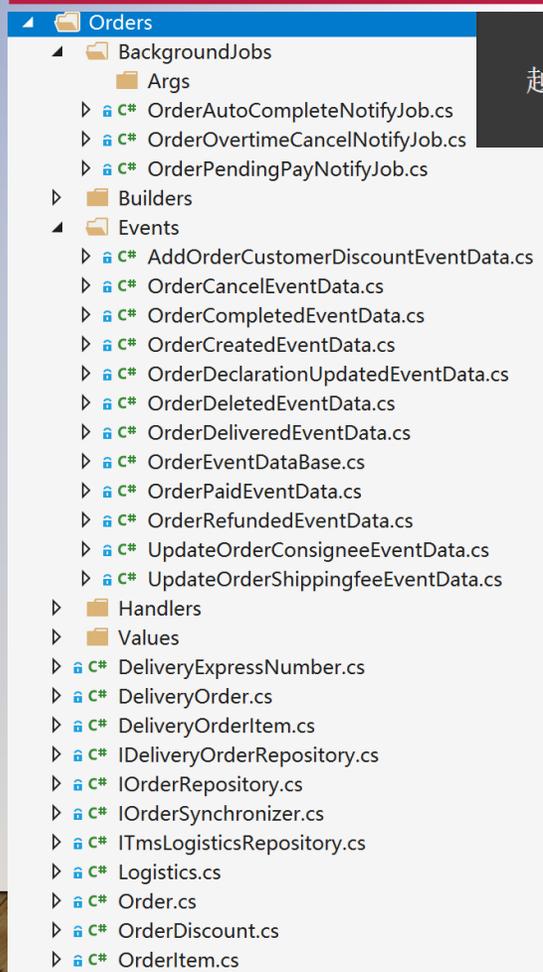


3.3.1 理解命名空间的作用



命名空间是垂直划分以组织代码

3.3.1 理解命名空间的作用



越是在上级空间中的对象，越具有可共享倾向（暂不考虑其他保护级别），越是往下级目录靠，其实就是越特殊

如果说，

程序集是水平划分代码（划分层次，程序集名称的最后一段代表其所在层次）

那么，

命名空间是垂直划分代码（划分概念，命名空间Personball.Demo.abc中的abc代表这部分代码所专注的领域和概念或者说是提供哪方面的功能）

这是正交的关系。

为了提供功能，必然需要跨越多个层次，在每个层次中都保证同样的目录结构（vs新建类或接口的所处命名空间），那么

同一个领域或概念在不同层次中是天然可见的，不需要特地用using语句引入命名空间

3.3.2 业务异常及错误码设计

```
using System;
using Abp;
using Abp.Logging;

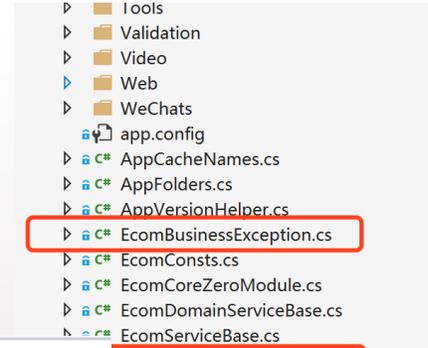
namespace Boss.Ecom
{
    99+ 引用 | personball, 255 天前 | 1 名作者, 2 项更改
    public class EcomBusinessException : AbpException, IHasErrorCode, IHasLogSeverity
    {
        99+ 引用 | personball, 364 天前 | 1 名作者, 1 项更改
        public EcomBusinessException(ErrorCode errorCode)
            : base(errorCode.ToString())
        {
            Code = (int)errorCode;
        }

        99+ 引用 | personball, 364 天前 | 1 名作者, 1 项更改
        public EcomBusinessException(ErrorCode errorCode, string message)
            : base(message)
        {
            Code = (int)errorCode;
        }

        0 个引用 | personball, 364 天前 | 1 名作者, 1 项更改
        public EcomBusinessException(ErrorCode errorCode, Exception innerException)
            : base(errorCode.ToString(), innerException)
        {
            Code = (int)errorCode;
        }

        0 个引用 | personball, 364 天前 | 1 名作者, 1 项更改
        public EcomBusinessException(ErrorCode errorCode, string message, Exception innerException)
            : base(message, innerException)
        {
            Code = (int)errorCode;
        }

        8 个引用 | personball, 364 天前 | 1 名作者, 1 项更改
        public int Code { get; set; }
        0 个引用 | personball, 255 天前 | 1 名作者, 1 项更改
        public LogSeverity Severity { get; set; } = LogSeverity.Warn;
    }
}
```



```
namespace Boss.Ecom
{
    99+ 引用 | lanyuanbo, 45 天前 | 8 名作者, 1 项更改
    public enum ErrorCode : int
    {
        通用错误异常 0-79

        账户注册登陆80-99

        #region 库存相关 100-110
        /// <summary>
        /// 库存不足
        /// </summary>
        InventoryNotEnough = 100,
        #endregion

        #region 优惠券相关 111-199
        /// <summary>
        /// 券码已被使用
        /// </summary>
        CouponUsed = 111,

        /// <summary>
        /// 券码无效
        /// </summary>
        CouponInvalid = 112,

        /// <summary>
        /// 超出领取时间
        /// </summary>
        CouponReceiveTime = 113,
    }
}
```

```
<!--ErrorCodes Start-->
<text name="UnknownException" value="未知异常"/>
<text name="ExistDuplicateItem" value="指定的对象存在重复"/>
<text name="ObjectIsNotActive" value="指定对象未激活"/>
<text name="CanNotBeDeletedSinceDependency" value="其他项依赖导致无法删除"/>
<text name="RequestParametersError" value="请求参数错误"/>
<text name="RequestHasOvertime" value="请求超时"/>
<text name="RequestSignError" value="请求签名错误"/>
<text name="ItemNotExists" value="对象不存在"/>
<text name="CanNotBeChangedSinceDependency" value="其他项依赖导致无法更改"/>
<text name="InventoryNotEnough" value="库存不足"/>
<text name="CouponUsed" value="券码已被使用"/>
<text name="CouponInvalid" value="券码无效"/>
<text name="CouponReceiveTime" value="超出领取时间"/>
<text name="CouponIsScrap" value="优惠券不能被领取"/>
<text name="CouponRepeat" value="不能重复领取"/>
<text name="NotCoupon" value="您还没有券"/>
<text name="CouponOverdue" value="优惠券过期了"/>
```

```
public override void PostInitialize()
{
    var errorInfoBuilder = IocManager.Resolve<IErrorInfoBuilder>();
    errorInfoBuilder.AddExceptionConverter(IocManager.Resolve<CustomExceptionErrorInfoConverter>());
}

EcomDbContextHelper.EntityFrameworkWarmUp();
}
```

- Libs\Boss.Ecom.Core\Orders\Order.cs (5)
- 194 : throw new EcomBusinessException(ErrorCode.OrderCannotDelete);
- 224 : throw new EcomBusinessException(ErrorCode.OrderCannotPaid);
- 250 : throw new EcomBusinessException(ErrorCode.OrderCannotComplete);
- 274 : throw new EcomBusinessException(ErrorCode.OrderCannotCancel);
- 309 : throw new EcomBusinessException(ErrorCode.OrderCannotRefund);

3.3.3 领域层设计

```
/// <summary>  
/// 分润账户余额（可提现余额）  
/// </summary>  
11 个引用 | personball, 281 天前 | 1 名作者, 3 项更改  
public int BalanceInCent { get; private set; }  
  
/// <summary>  
/// 冻结金额（提现中）  
/// </summary>  
8 个引用 | personball, 281 天前 | 1 名作者, 3 项更改  
public int FrozenInCent { get; private set; }
```

关闭setter

```
/// <summary> 余额增加  
2 个引用 | personball, 243 天前 | 2 名作者, 4 项更改  
public PspAccountHistory Increase(int amountInCent, string remark) ...  
  
/// <summary> 余额冻结  
1 个引用 | personball, 320 天前 | 1 名作者, 2 项更改  
public void Freeze(int amountInCent) ...  
  
/// <summary> 冻结解冻  
1 个引用 | personball, 320 天前 | 1 名作者, 2 项更改  
public void Unfreeze(int amountInCent) ...  
  
/// <summary> 冻结扣减  
1 个引用 | personball, 320 天前 | 1 名作者, 2 项更改  
public PspAccountHistory Decrease(int amountInCent, string remark) ...
```

添加methods

```
public PspAccountHistory Increase(int amountInCent, string remark)  
{  
    if (amountInCent < 0)  
    {  
        throw new ArgumentOutOfRangeException(nameof(amountInCent), $"nameof(amountInCent)");  
    }  
  
    if (amountInCent == 0)  
    {  
        return null;  
    }  
  
    BalanceInCent += amountInCent;  
  
    DomainEvents.Add(new Events.PspAccountIncreasedEventData  
    {  
        AccountId = this.Id,  
        AccountLevelId = this.AccountLevelId,  
        IncreasedAmountInCent = amountInCent,  
        OrganizationId = this.OrganizationId,  
        UserId = this.UserId,  
        IncreasedType = increasedType  
    });  
  
    return history;  
}
```

考虑事件

3.3.3 领域层设计

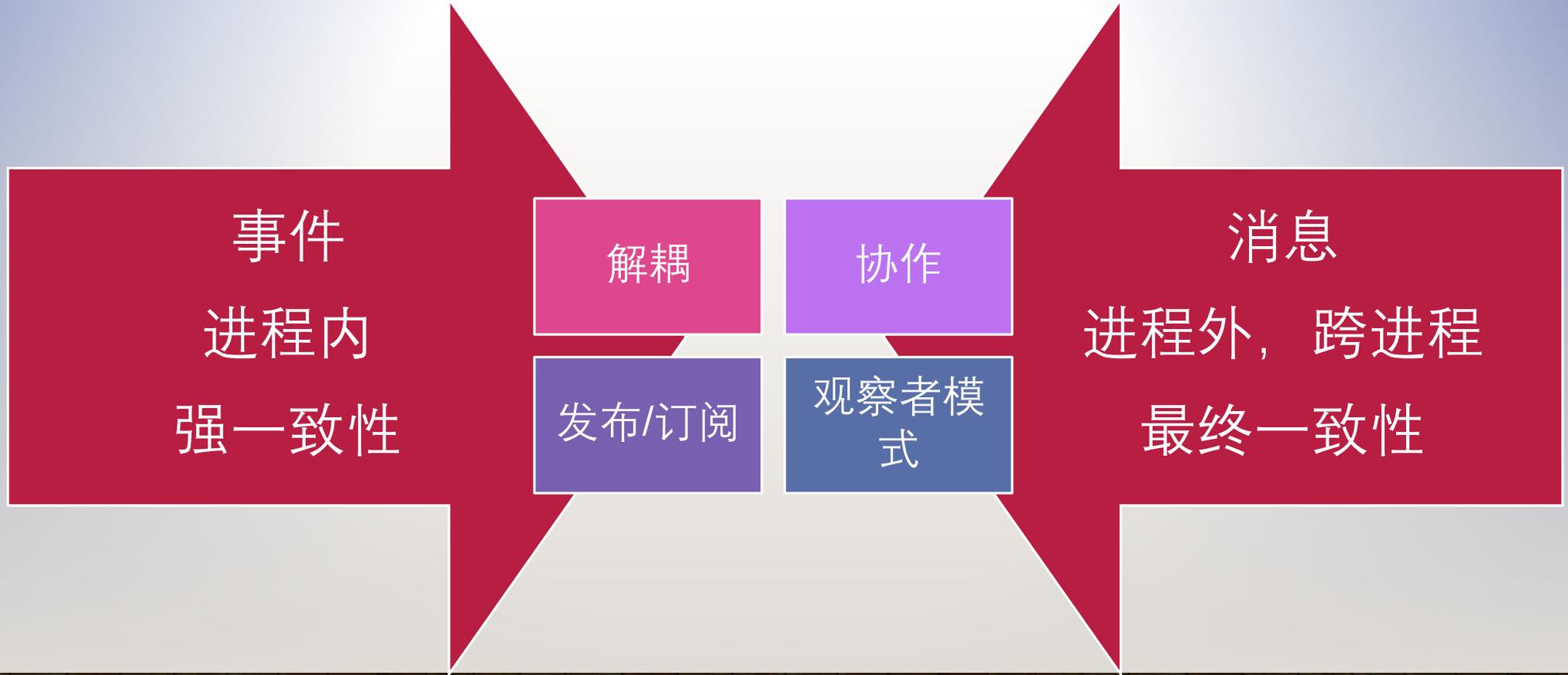
The image shows a Visual Studio IDE with two main windows. The left window displays the source code for `OrderManager.cs` in the `Boss.Ecom.Orders` namespace. The right window shows the Solution Explorer for the `Orders` project, highlighting the `Builders` folder.

```
OrderManager.cs*
Boss.Ecom.Core
Boss.Ecom.Orders.OrderManager
CreateOrder(OrderBasicInfo orderBasicInfo, bool isPreviewMode)
87
88
89 //构建builder
90 var orderBuilder = IoCManager.Instance.Resolve<DefaultOrderBuilder>(new
91 ...);
98
99 //构建上下文
100 var context =
101     new OrderBuilderContext(
102         new TaxCouponDecorator( //税优惠券
103             new SelectDefaultTaxCouponDecorator( //税优惠券取默认
104                 new TaxActivitiesDecorator( //税促销活动
105                     new ShippingFeeCouponDecorator( //运费优惠券
106                         new SelectDefaultShippingFeeCouponDecorator( //运费优惠券取默认
107                             new GoodsCouponsDecorator( //商品优惠券
108                                 new SelectDefaultGoodsCouponsDecorator( //商品优惠券取默认
109                                     new ShippingFeeActivityDecorator( //运费促销活动
110                                         new GoodsActivitiesDecorator( //商品促销活动
111                                             orderBuilder //订单基本信息构建
112                                         ), _freightCalculator
113                                     ), _couponRepository, _couponHistoryRepository
114                                 )
115                             )
116                         )
117                     ), _taxCalculator
118                 )
119             )
120         ), _promotionManager
121     );
122
123 ...
156 context.LoadUnexpiredActivities(canUseActivities);
157 ...
164
165 //设置预览模式
166 if (isPreviewMode)
167 {
168     context.SetPreviewMode(orderBasicInfo, needDefaultCoupons);
169 }
170 else
171 {
172     context.SetCreateNewMode(orderBasicInfo);
173 }
174
175 //构建订单
176 await context.Build();
177
```

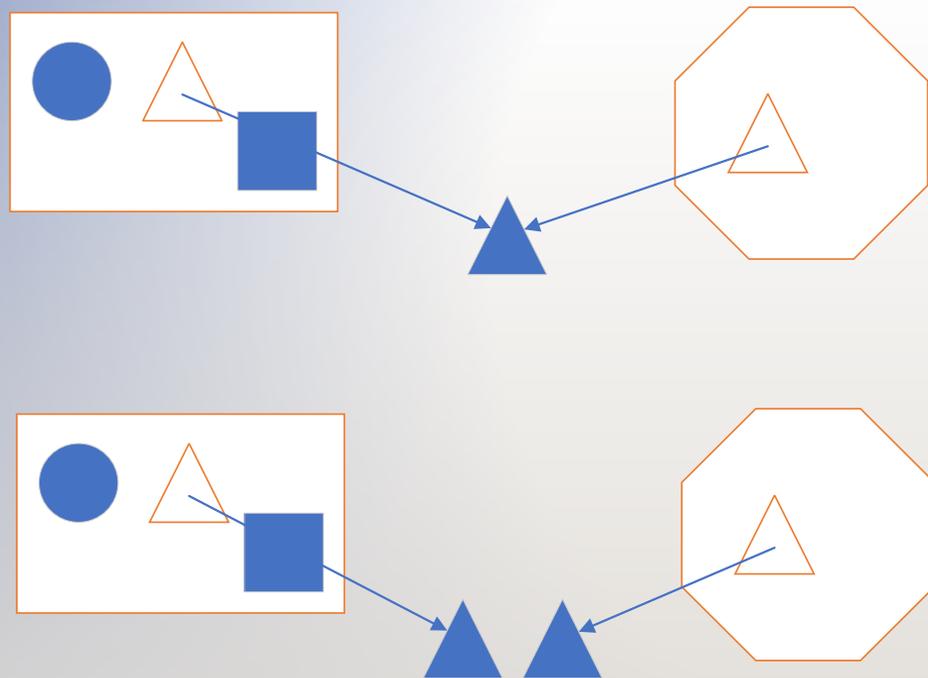
The Solution Explorer on the right shows the following structure for the `Orders` project:

- Orders
 - BackgroundJobs
 - Builders
 - Decorators
 - GoodsActivitiesDecorator.cs
 - GoodsCouponsDecorator.cs
 - SelectDefaultGoodsCouponsDecorator.cs
 - SelectDefaultShippingFeeCouponDecorator.cs
 - SelectDefaultTaxCouponDecorator.cs
 - ShippingFeeActivityDecorator.cs
 - ShippingFeeCouponDecorator.cs
 - TaxActivitiesDecorator.cs
 - TaxCouponDecorator.cs
 - CouponWithHistoryId.cs
 - DefaultOrderBuilder.cs
 - IOrderBuilder.cs
 - OrderBasicInfo.cs
 - OrderBuilderContext.cs
 - Events
 - Handlers
 - Values
 - DeliveryExpressNumber.cs
 - DeliveryOrder.cs
 - DeliveryOrderItem.cs
 - IDeliveryOrderRepository.cs
 - IOrderRepository.cs
 - IOrderSynchronizer.cs
 - ITmsLogisticsRepository.cs
 - Logistics.cs
 - Order.cs
 - OrderDiscount.cs
 - OrderItem.cs
 - OrderItemGoodsIdentifier.cs
 - OrderItemPackageItem.cs

3.3.4事件与消息



3.3.5 值对象的不可变性



引用类型的值对象必须具备不可变性，以防止多个实体持有同一个值对象引用时，其中一个实体修改了该对象但并未意识到影响了另一个实体就会造成意料之外的影响

必须关闭属性的set访问器！

```
12 个引用 | personball, 304 天前 | 1 名作者, 3 项更改
public class PspBuyerInfo : ValueObject<PspBuyerInfo>
{
    0 个引用 | personball, 311 天前 | 1 名作者, 1 项更改
    protected PspBuyerInfo() ...

    1 个引用 | personball, 304 天前 | 1 名作者, 1 项更改
    public PspBuyerInfo(long buyerAccountId, long buyerParentAccountId, int buyerOrganizationId) ...

    /// <summary> 买家账号
    9 个引用 | personball, 305 天前 | 1 名作者, 2 项更改
    public long BuyerAccountId { get; private set; }

    /// <summary> 买家所属组织
    3 个引用 | personball, 304 天前 | 1 名作者, 2 项更改
    public int BuyerOrganizationId { get; private set; }

    /// <summary> 推荐人
    2 个引用 | personball, 305 天前 | 1 名作者, 2 项更改
    public long BuyerParentAccountId { get; private set; }
}
```

4.ABP的成功案例



4. IABP的成功案例——积微物联(成都)



我们的服务

OUR SERVICES



积微钢铁

打造钢铁综合服务生态圈

智能交易、专业服务、一线品牌、一手资源、支付安全、价格透明



积微运网

全新的物流服务体验

加入积微运网, 运货更快捷, 找货更轻松、服务更贴心, 诚信有保证



积微循环

闲废物资交易一站式服务

致力于打造西南地区最专业的循环物资服务平台



我们'钛'在

打造全球闻名



积微钒网

不钒为您呈现

积微钒网, 成就“不钒”



仓储服务

您货品的全职管家

立足西南, 布局全国, 提供安全、便捷、透明的仓储服务



加工服务

货不出仓的一站式服务

产能在线, 精准加工, 专业化的定制服务

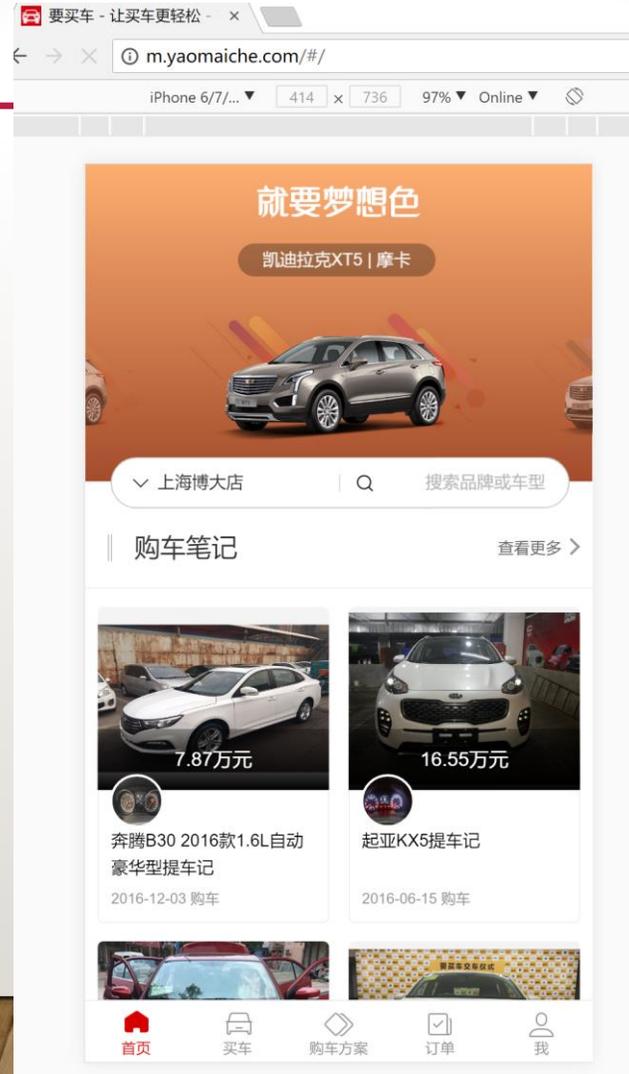


开启数据封

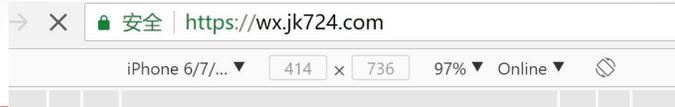
积百家观点,



4.2ABP的成功案例——要买车(上海)



4.3 ABP的成功案例——健康724(上海)



Visualize / 724电商平台接口统计

ServiceName: Boss.Ecom.AppApiHost.WebApi.Controllers.*

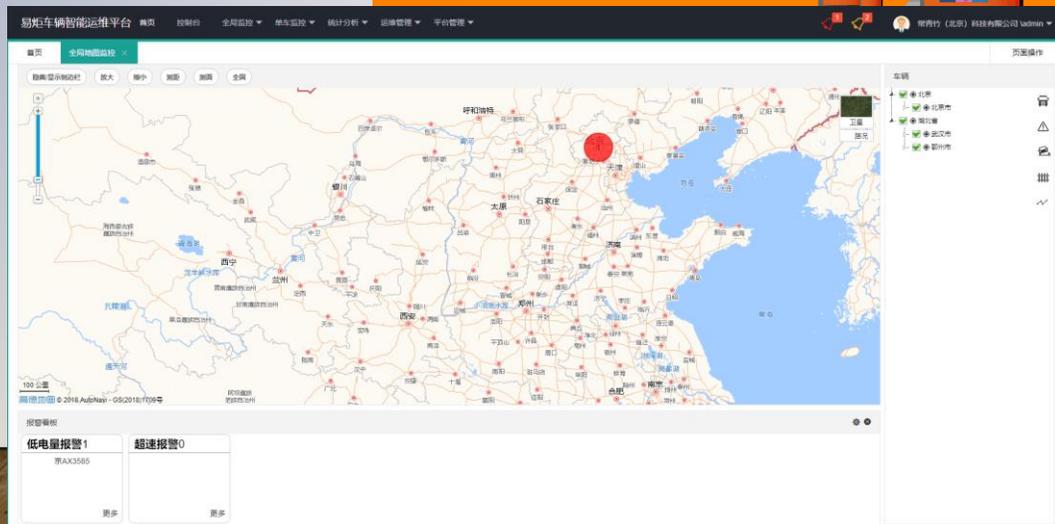
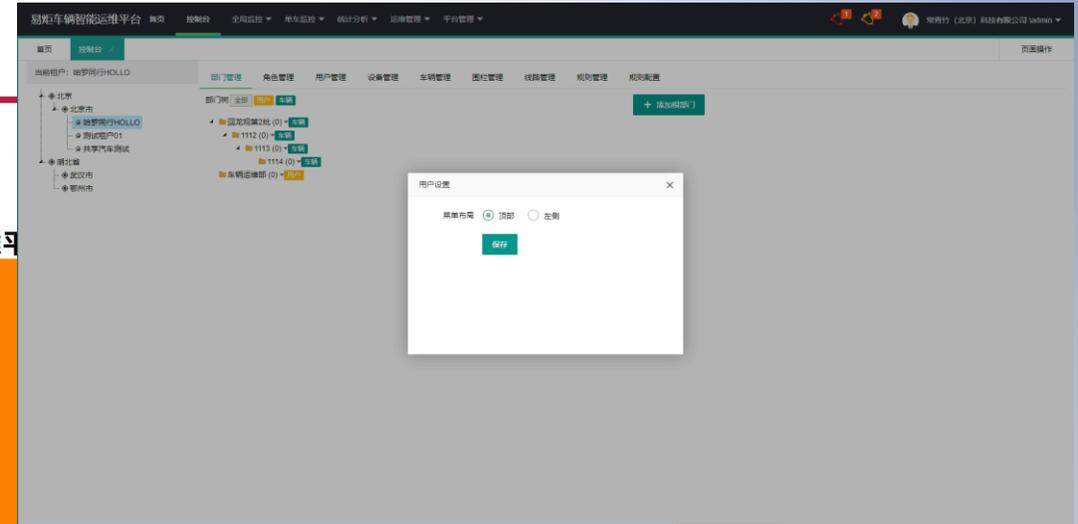
Filter: #this.keyword: '980' Add a filter

Actions: Save Share Refresh Last 30 days Uses lucene query syntax

调用类型	调用方法	次数	平均时长	Percentile rank 100 of "时长"	Percentile rank 200 of "时长"	Percentile rank 500 of "时长"
Boss.Ecom.AppApiHost.WebApi.Controllers.UserController	SetIdCardAndName	43	1,444.047	0%	0%	0%
Boss.Ecom.AppApiHost.WebApi.Controllers.UserController	Update	360	322.378	0%	56.667%	97.115%
Boss.Ecom.AppApiHost.WebApi.Controllers.OrderController	GetOrderPreviewInfoBeforeAppVersion4_3_1	91	161.033	0%	93.64%	97.708%
Boss.Ecom.AppApiHost.WebApi.Controllers.CollegeArticleController	Delete	2	192.5	0%	2.119%	100%
Boss.Ecom.AppApiHost.WebApi.Controllers.PspOrganizationController	QueryMyMembersTest	1	39.476	0%	0%	0%
Boss.Ecom.AppApiHost.WebApi.Controllers.PspOrganizationController	QueryMyOrgMembersTest	2	6.1235	0%	0%	0%
Boss.Ecom.AppApiHost.WebApi.Controllers.CollegeVideoController	Delete	2	133.5	0%	100%	100%
Boss.Ecom.AppApiHost.WebApi.Controllers.MarketingActivityController	GetPaymentTopRankingList	372	293.831	0%	0%	99.415%
Boss.Ecom.AppApiHost.WebApi.Controllers.PspCloudStorageOutRecordController	GetPcsOutRecord	1	698	0%	0%	0%
Boss.Ecom.AppApiHost.WebApi.Controllers.PspCloudStorageTakeDeliveryRequestController	CreateTakeDeliveryRequest	2	440	0%	0%	15.306%
Boss.Ecom.AppApiHost.WebApi.Controllers.PspCloudStoragePurchaseRequestController	GetMyPurchaseRequests	4	153.5	0%	100%	100%
Boss.Ecom.AppApiHost.WebApi.Controllers.PaymentController	Pay	5,381	286.225	1.171%	31.964%	98.148%
Boss.Ecom.AppApiHost.WebApi.Controllers.PspOrganizationController	QueryMyMembers	1,102	2,986.867	1.26%	5.838%	17.208%
Boss.Ecom.AppApiHost.WebApi.Controllers.OrderController	CreateOrder	4,718	230.489	1.293%	46.183%	97.769%
Boss.Ecom.AppApiHost.WebApi.Controllers.CollegeVideoController	CreateOrUpdate	13	190.846	1.538%	54.751%	100%
Boss.Ecom.AppApiHost.WebApi.Controllers.OrderController	GetOrderPreviewInfo	15,304	140.684	1.712%	94.476%	99.115%
Boss.Ecom.AppApiHost.WebApi.Controllers.GoodsController	SaleCategorySearch	146,018	131.931	10.453%	99.344%	99.89%
Boss.Ecom.AppApiHost.WebApi.Controllers.HealthSelfRatedHistoriesController	Delete	46	132.109	20.788%	89.891%	100%
Boss.Ecom.AppApiHost.WebApi.Controllers.UserController	GetUserCenter	73,586	112.329	22.147%	98.408%	99.844%
Boss.Ecom.AppApiHost.WebApi.Controllers.FeedbackController	Post	4	109	23.026%	100%	100%
Boss.Ecom.AppApiHost.WebApi.Controllers.CollegeCourseController	GetCollegeCourse	3,411	153.75	24.89%	80.035%	99.077%
Boss.Ecom.AppApiHost.WebApi.Controllers.CollegeVideoController	GetCollegeVideoForHS	839	146.908	26.949%	80.731%	99.737%
Boss.Ecom.AppApiHost.WebApi.Controllers.PspOrganizationController	QueryDirectlyOrg	610	139.598	31.967%	85.656%	99.695%
Boss.Ecom.AppApiHost.WebApi.Controllers.HealthManagerController	GetHealthManagersByTag	6,257	161.781	37.75%	64.491%	98.549%
Boss.Ecom.AppApiHost.WebApi.Controllers.UserAddressController	DeleteUserAddress	21	96.048	40.659%	100%	100%
Boss.Ecom.AppApiHost.WebApi.Controllers.PspOrganizationController	QueryServiceTribe	221	269.552	42.986%	63.057%	80.215%
Boss.Ecom.AppApiHost.WebApi.Controllers.PspAchievementController	GetMyBalanceTransactionPagedListV2	18	90.889	45.96%	91.667%	100%
Boss.Ecom.AppApiHost.WebApi.Controllers.PspAchievementController	GetAllAchievementsStatistics	103	482.398	49.515%	70.838%	78.977%
Boss.Ecom.AppApiHost.WebApi.Controllers.PspCloudStorageOutRecordController	GetListPcsOutRecord	12	252.5	51.62%	55.182%	66.474%
Boss.Ecom.AppApiHost.WebApi.Controllers.PspCashOutRequestController	CreateRequest	13	83.923	55.769%	79.487%	100%

- 解决方案'Boss.Ecom' (22 个项目)
 - .nuget
 - Libs
 - Boss.Ecom.Application
 - Boss.Ecom.Core**
 - Boss.Ecom.EntityFramework
 - Boss.Ecom.IndexModels
 - Boss.Ecom.Infrastructure
 - Boss.Ecom.MqMessages
 - Modules
 - Boss.Ecom.Core.PublishEvents
 - Boss.Ecom.MqHandlers.BusinessProcessor
 - Boss.Ecom.MqHandlers.OperationHistoriesRecorder
 - Boss.Ecom.MqHandlers.PspProcessor
 - Tests
 - Tools
 - Boss.Ecom.MqMessagesT4Generator
 - Boss.Ecom.PublishEventsHandlersT4Generator
 - WinServices
 - Boss.Ecom.HistoriesRecorder
 - Boss.Ecom.IndexEvents
 - Boss.Ecom.IndexService
 - Boss.Ecom.JobServices
 - Boss.Ecom.NotifyServices
 - Boss.Ecom.Processor
 - Boss.Ecom.ScmSyncServices
- Service-Hosts**
- Web-Hosts
 - Boss.Ecom.AppApiHost
 - Boss.Ecom.PortalApiHost

4.4ABP的成功案例——易炬车辆智能运维平台(武汉)



```
using Abp.Authorization;
using Abp.Localization;

namespace Etorch.Framework.Authorization
{
    public class FrameworkAuthorizationProvider : AuthorizationProvider
    {
        public override void SetPermissions(IPermissionDefinitionContext context)
        {
            // 这里后期可以通过配置在webconf中定义
            var pages = context.GetPermissionOrNull(PPermissionNames.Pages) ?? context.CreatePermission(PPermissionNames.Pages, L("易炬车辆智能运维平台"));
            // 控制台
            var console = pages.CreateChild(PPermissionNames.Pages_ConsoleManage, L("控制台"));
            // 组织架构管理
            var organizationInits = console.CreateChild(PPermissionNames.Pages_ConsoleManage_OrganizationInits, L("组织架构管理"));
            organizationInits.CreateChild(PPermissionNames.Pages_ConsoleManage_OrganizationInits_Create, L("组织架构创建"));
            organizationInits.CreateChild(PPermissionNames.Pages_ConsoleManage_OrganizationInits_Delete, L("组织架构删除"));
            organizationInits.CreateChild(PPermissionNames.Pages_ConsoleManage_OrganizationInits_Update, L("组织架构更新"));
            // 角色管理
            var role = console.CreateChild(PPermissionNames.Pages_ConsoleManage_Role, L("角色管理"));
            role.CreateChild(PPermissionNames.Pages_ConsoleManage_Role_Create, L("角色创建"));
            role.CreateChild(PPermissionNames.Pages_ConsoleManage_Role_Delete, L("角色删除"));
            role.CreateChild(PPermissionNames.Pages_ConsoleManage_Role_Update, L("角色更新"));
            // 用户管理
            var user = console.CreateChild(PPermissionNames.Pages_ConsoleManage_User, L("用户管理"));
            user.CreateChild(PPermissionNames.Pages_ConsoleManage_User_Create, L("用户创建"));
            user.CreateChild(PPermissionNames.Pages_ConsoleManage_User_Delete, L("用户删除"));
            user.CreateChild(PPermissionNames.Pages_ConsoleManage_User_Update, L("用户更新"));
            // 设备管理
            var device = console.CreateChild(PPermissionNames.Pages_ConsoleManage_Device, L("设备管理"));
            device.CreateChild(PPermissionNames.Pages_ConsoleManage_Device_Create, L("设备创建"));
            device.CreateChild(PPermissionNames.Pages_ConsoleManage_Device_Delete, L("设备删除"));
            device.CreateChild(PPermissionNames.Pages_ConsoleManage_Device_Update, L("设备更新"));
            // 车辆管理
            var vehicle = console.CreateChild(PPermissionNames.Pages_ConsoleManage_Vehicle, L("车辆管理"));
            vehicle.CreateChild(PPermissionNames.Pages_ConsoleManage_Vehicle_Create, L("车辆创建"));
            vehicle.CreateChild(PPermissionNames.Pages_ConsoleManage_Vehicle_Delete, L("车辆删除"));
            vehicle.CreateChild(PPermissionNames.Pages_ConsoleManage_Vehicle_Update, L("车辆更新"));
            vehicle.CreateChild(PPermissionNames.Pages_ConsoleManage_Vehicle_Import, L("车辆导入"));
            vehicle.CreateChild(PPermissionNames.Pages_ConsoleManage_Vehicle_BindDevice, L("设备绑定"));
            vehicle.CreateChild(PPermissionNames.Pages_ConsoleManage_Vehicle_UnbindDevice, L("设备解绑"));
            // 统计管理
        }
    }
}
```

4.5ABP的成功案例——京师在线(北京)



4.6ABP的成功案例——WOOW.SITE(岳阳)

The image shows a browser window displaying the website woow.site. The page features a dark background with white text. On the left, the text reads: "商品防伪红包系统" (Product Anti-counterfeit Red Envelope System) and "一物一码，强制关注公众号后领红包，提升商品市场竞争力，更精准的微信活动营销" (One item, one code, mandatory follow of WeChat public account to receive red envelope, improving product market competitiveness, more precise WeChat activity marketing). On the right, there is a large red graphic with a QR code and the text "抢红包" (Grab Red Envelope). Below the QR code, it says "关注立" (Follow immediately) and "您购买的是正品" (What you buy is genuine). At the bottom of the graphic, it lists product details: "产品名称: 贵州茅台" (Product Name: Guizhou Maotai), "销售区域: 湖南" (Sales Area: Hunan), "生产日期: 2017-08-13" (Production Date: 2017-08-13), and "生产厂家: 茅台镇" (Manufacturer: Maotai Town). A small red envelope icon is visible at the bottom right of the graphic.

success - 国内版 Bing in x 一物一码防伪红包系统 x

woow.site

商品防伪红包系统

一物一码，强制关注公众号后领红包，提升商品市场竞争力，更精准的微信活动营销

抢

抢红包

关注立

您购买的是正品

产品名称: 贵州茅台

销售区域: 湖南

生产日期: 2017-08-13

生产厂家: 茅台镇

奖励由公众
需要您先关注

维护权益，支持正品

4.7 ABP的成功案例——MORE AND MORE



我们下周五上线一个在线教育类的项目。后端也是 abpcore + identity server4 来得及的话到时候也把地址给你

上午10:01:41

你们有正式使用ABP上生产的项目么?



[自动回复]您好, 我现在有事不在, 一会再和您联系。

我找点成功案例, 放PPT上宣传 😁



下午3:22:52



还没呢~ 最近的一个 还在内测~ 公网上无法访问的



最快 5月底 上线。

ABP是一个很好的起点！ 谢谢大家！

开源
便于学习

设计优良

DDD入门

项目
快速启动

集中精力开发
可维护可扩展
的业务

社区活跃



ABP

ABP架构设计交流群4
扫一扫二维码，入群聊。