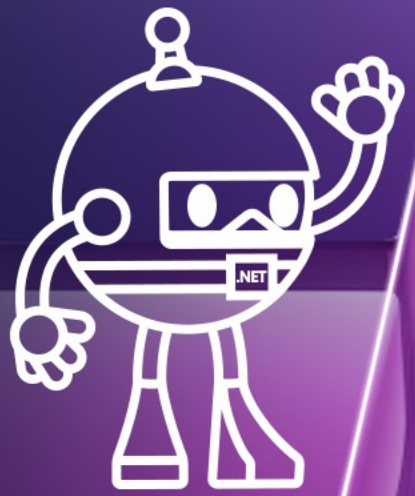
.NET Conf China 2023

2023/12/16
09:30 - 18:00

中国・北京



.NET中文社区

中国 · 北京

.NET Conf China 2023

从 Java 8 到 .NET 8，团队升级工作汇报

肖伟宇 2023/12/16

目录

- Why，为什么要从 Java 到 .NET
- How，升级过程是怎么操作的
- Result，团队付出了什么代价，获得了哪些收益
- 思考与总结

Why

为什么要从 Java 到 .NET

背景

技术背景

Java 8
SpringBoot 2.x
微服务架构

业务背景

产品功能交付/迭代变慢的速度越来越快

变更BUG不可控，问题频出

系统负载能力弱，流量小高峰被打崩

团队背景

- 团队成员.NET 零经验
- 初级、中级、高级工程师全覆盖

你以为的决策过程

	框架落地成本	团队学习成本	收益	ORM	风险
Java 8	极低	无	无	MyBatis Plus	无
Java 21	中	低	新特性 更好的性能 虚拟线程	JPA	低，编程语言不切换
.NET 8	中	低	Linq async/await 更好的性能	EntityFrameworkCore	低，有成功的落地经验
Go	高	高	更好的性能 并行计算能力	GORM	高，团队缺少Go大牛

实际的决策过程

与产品

- 产品： “为什么功能迭代越来越慢”
- 我： “我们的技术实现与实际业务偏离越来越大了 “
- 产品： “该怎么解决”
- 我： “业务设计与实现需要重构，原则是保持技术实现与产品模型一致性”

与后端

- 后端： “旧代码维护起来太痛苦了”
- 我： “这里有一个重新开始的机会，但得用新框架”
- 后端： “换技术栈风险高”
- 我： “语法与Java很像，框架开箱即用，一小时上手”

最底层的决策逻辑

如何保持可持续的快速迭代?

保持技术实现与业务建模的一致性

使用领域驱动设计战术框架

最底层的决策逻辑

领域驱动设计落地需要充血模型

EntityFrameworkCore vs JPA（Hibernate）vs MyBatis Plus

系统要支持事件的最终一致性，以支持健壮的领域事件实现

CAP vs Seata

新框架需要足够快速和简单地上手，尽可能降低技术投入

C# vs Java，语法相似

开箱即用的领域驱动设计战术框架

新框架需要对测试友好

成熟的单元/集成测试框架

代码驱动的环境与数据初始化

有落地的实践经验

成功的SaaS系统落地经验

最底层的决策逻辑

.NET 8 是最合适的选择

How

升级过程是怎么操作的

第一个项目

1天搭建工程（集成框架/CICD）

1小时上手培训

5天项目开发

4天集成测试

心态过程

开始前，本能地恐惧 → 写第一个模块时，熟悉又陌生 → 第一个项目交付时，真香

遇到的困难

心态的困难

1.对新事物的本能畏惧

2.新语言和框架不熟练

项目上的困难

1.Json序列化，需要使用Newtonsoft.Json更宽松的序列化

2.MQ消息与Java消息格式兼容

3.雪花算法Id生成器（让GPT翻译一个版本）

团队外部的质疑

异步IO（async/await），Java也有

强大的的ORM，Java也有

自动化测试支持，Java也有

“交付型项目，客户会在意编程语言”

“为什么不用Go？”

Result

团队付出了什么代价，获得了哪些收益

必然的代价

业务模型重构

开发效率提升

- 起始阶段不低于原来的迭代速度
- 成熟阶段有显著的迭代速度提升
- 避免了“模型与实现不一致”的潜在成本

团队变化

领域驱动设计的战术实践成为现实（专注业务）

1.基于事件驱动的建模设计实现

2.建模设计与代码实现一致性

更明确的代码组织方式（代码的可维护性）

1.充血模型，负责模型自身的业务逻辑

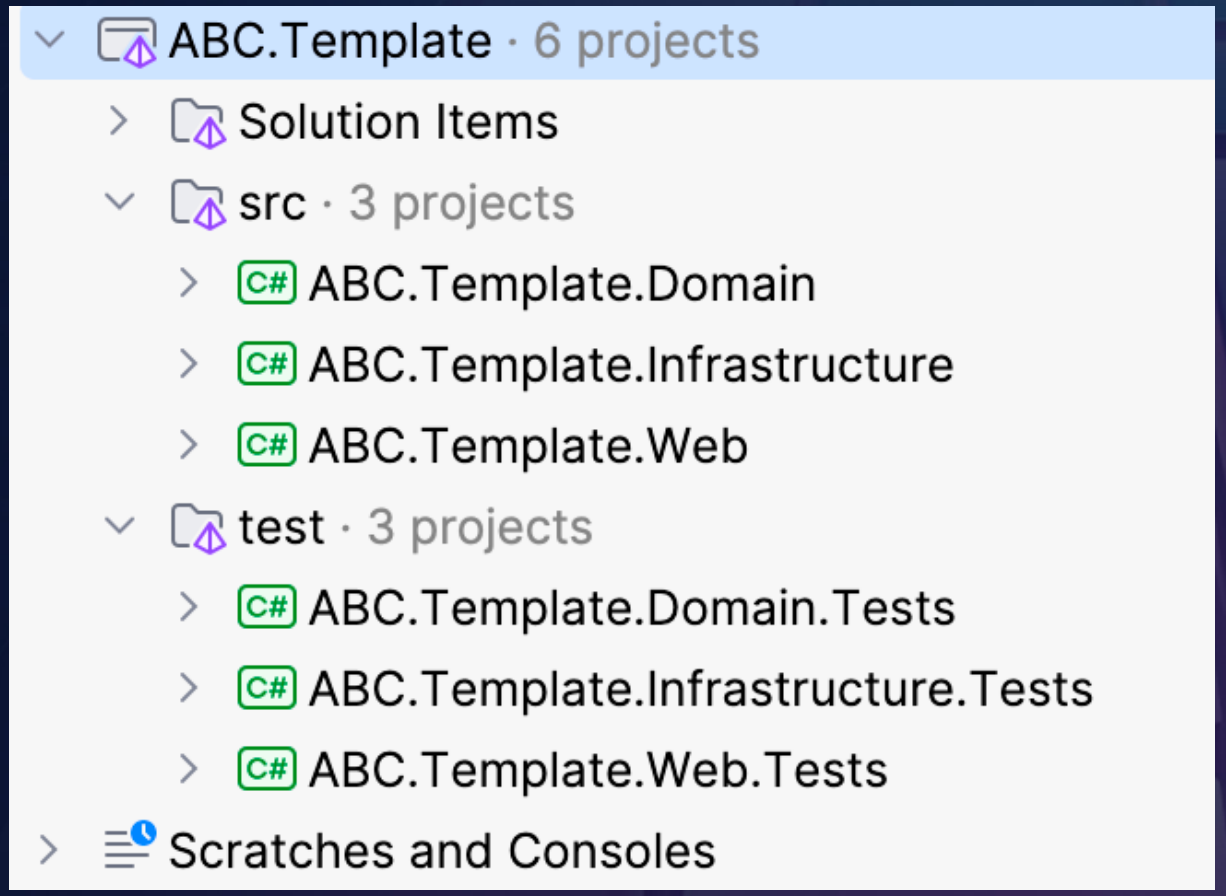
2.CommandHandler，负责接收指令，操作模型

3.EventHandler，负责处理事件逻辑

4.Controller，负责与UI/外部服务交互

5.Query，负责从数据源检索数据

6.领域层/基础设施/应用层


ABC.Template · 6 projects
Solution Items
src · 3 projects
ABC.Template.Domain
ABC.Template.Infrastructure
ABC.Template.Web
test · 3 projects
ABC.Template.Domain.Tests
ABC.Template.Infrastructure.Tests
ABC.Template.Web.Tests
Scratches and Consoles


.NET
.NET 8
.NET

- ABC.Template.Web
  - Dependencies
  - Properties
  - wwwroot
  - Application
    - Commands
    - DomainEventHandlers
    - Hubs
    - IntegrationEventHandlers
    - Jobs
    - Mappers
    - Queries
  - Controllers
    - DemoController.cs
    - OrderController.cs
  - Extensions
  - Migrations
  - appsettings.json
  - appsettings.Development.json
  - Dockerfile
  - GlobalUsings.cs
  - Program.cs

架构成果

- 多语言混合架构
- 分布式链路追踪（skywalking）
- 上下文传递系统
- 全链路灰度系统

性能与资源

更低的资源占用（更高的资源利用率）

内存：300MB vs 1500MB

更好的性能（极高的下限）

启动速度 10s vs 100s

异步IO，更好的吞吐能力

自动化测试收益

框架组件自动化测试全覆盖 + 业务接口自动化测试全覆盖 = 形成测试用例资产的持续积累

思考与总结

成功的核心原因

有一群心态开放且能够独立思考判断的队友

对新事物有好奇心

有独立的思考判断能力

.NET 的生态足够的成熟

要用的都有

人之所以为人，就是因为学会了使用工具

选择合适的工具

下限很高，
上限更高

让平凡人做
不平凡事

出发点（目的）很重要

语言之争

团队进步

可衡量的结果很重要

高层的信任很重要

持续提高开发者幸福感

感谢

FireUG 技术社区

做些好玩的程序员下饭小视频

引用资源

netcorepal/netcorepal-cloud-framework

基于 ASP.NET Core 的领域驱动设计微服务架构实现方案

https://github.com/netcorepal/netcorepal-cloud-framework

netcorepal/netcorepal-cloud-template
基于netcorepal-cloud-framework的项目模板工具
https://github.com/netcorepal/netcorepal-cloud-template