服务端工程结构规范

1. 使用的语言

所有服务端工程均采用 Java 或者 JVM 支持的其他语言（如 Groovy，Scala，Kotlin 等）进行开发，除非经过讨论认为有必要引入非 JVM 体系语言。JVM 体系语言能够互相操作，从而不会造成集成问题。

2. 基本框架

为了加快开发速度，使后端分布式应用能够快速健壮的发挥实际功能，采用 Spring Boot 和 Spring Cloud 进行集成开发。涉及到 H5 页面的，采用前后端完全分离的方式进行，从而使后端的业务逻辑实现最大程度的复用。系统间通信采用 JSON 方式进行；微服务之间通过 Feign 接口进行数据交换，从而让远程过程调用对业务透明。

3. 项目名称

项目名称的模板形如 iwanvi-${project.name}-${module.name}，例如 iwanvi-free-book-pay, iwanvi-free-book-comment, iwanvi-zwsc-pay 等等结构，其中 project.name 也需要符合 AA-BB-XX 的形式，长度不限，但是必须反应工程名称的实际功能。

4. 项目目录结构

为了不造成各项目之间的混乱，规定所有服务端工程采用 MAVEN 构建工具，目录结构以 maven 的标准目录结构进行；如果采用其他语言，则需要添加相应的语言源码包，如采用 kotlin 进行开发，则需要增加 src/main/kotlin，src/test/kotlin 两个相应的源码包；package 在老项目中，各种命名的方式都有，新工程将统一成如下形式：

com.iwanvi.${project.short.name}.${module.short.name}

其中 project.short.name 针对每一个项目是唯一的，目前罗列如下：

免费电子书： freebook

中文书城：zwsc

大数据：fishworm

module.short.name 根据项目下的模块进行实际的命名，但是必须要清晰反映实际的模块功能。

子 package 根据实际情况进行命名即可，要求是准确反映包下文件的功能。

5. 开发工具

JDK：1.8 8u172

Tomcat: 9.0.8

IDE: IDEA，社区版和旗舰版均可

质量控制工具：PMD+FINDBUGS+CHECKSTYLE

6. 协同问题

版本控制管理工具：GIT

Nexus 私服：默认情况下，升级协同开发 SDK 的时候，需要升级版本号