**后端开发技术体系**

**版本：V1.0**

**陕西秦快互联科技有限公司**

**2017年05月**文档信息

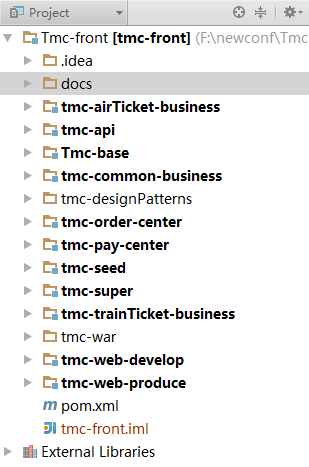
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **文档名称** | 后端开发技术体系 | | | |
| **适用范围** | ☑公司 | | | |
| **版本** | **变更说明** | **作者** | **日期** | **备注** |
| V1.0 | 初稿 | 韩后起 | 2017-05-22 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# 1、后端技术体系

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **技术内容** | **技术内容详解** | **备注** |
| **开发环境** | JDK1.8、MySql、Tomcat、Redis |  |
| **IDEA** | IDEA环境、常用快捷键、使用方式 |  |
| **Maven** |  |  |
| **Git** |  |  |
| **MySql** |  |  |
| **Redis** |  |  |
| **MyBatis** |  |  |
| **Spring** |  |  |
| **SpringMvc** |  |  |
| **JSON** | 掌握JSON格式及使用。 |  |

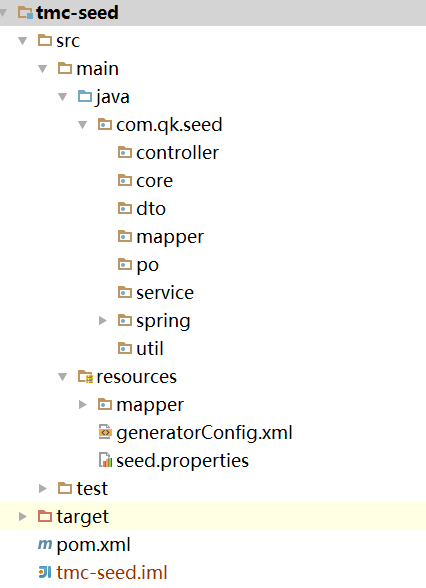
# 2、后端模块结构

## 2.1、模块划分如图（待重构后修改）



**注意：添加后端模块，需经审核后开启，不可私自创建模块**

## 2.2、模块包结构

****

## 2.3、模块说明

### 2.3.1、总体设计

可组装的功能块，我们称为一个复用构件，简称构件。

构件分两种，一种为功能构件，一种为配置构件。

#### 2.3.1.1、功能构件

原则上，功能构件应包含两个模块，一个接口模块，一个实现模块。  
  
1. 接口模块之包含 interface，不包含任何 implements。  
1. 原则上接口模块不允许依赖其他构件。  
1. 原则上不允许包含任何资源文件和测试代码。  
1. 接口模块应包含po。  
  
实现模块  
  
1. 实现模块主要提供对应接口的服务 implement 层。  
1. 实现模块主要组成部分有：Mapper、Service、Controller 三层。  
1. 原则Service层的每个方法，都必须提供构件测试，该层的测试覆盖率需要达 100%。

#### 2.3.1.2、配置构件

原则上，一个配置构件只含有一个配置模块，主要负责组装用途，只能包含配置性质的资源或代码

### 2.3.2、外放API模块(单独提出，可被内部集成，可外放)

模块规范：  
tmc-API  
单独部署，提供外放接口  
tmc-service-api  
作为我司内部项目直接集成使用

# 3、模块开发步骤（以seed模块介绍）

## 3.1、需求 -> 业务逻辑

(此处略去三百页……)

## 3.2、编写测试 -> 服务层设计、实体层设计、数据传输层设计

编写测试

1. 对每个业务逻辑编写服务测试  
2. 对关联的业务逻辑编写业务流测试  
3. 从服务层逻辑中，导出 po 或 dto 的设计

## 3.3、审查设计

测试编写完成后，提交测试进行审查。主要为服务层的接口与业务逻辑的一致性、完整性。

## 3.4、实现服务层(service)

1. 从测试生成服务层的类结构  
   2. 实现类结构中的每个方法  
   3. 需要操作数据时，分离出数据层

## 3.5、从服务层推导数据层(mapper)

根据编写服务层时候的需要，编写数据层

## 3.6、审查实现

完成服务、数据层后，提交审查。主要检查测试通过率，及各项内部代码是否符合代码规范与原则。

## 3.7、编写数据传输层(controller)

1. 对每个服务，编写 web api(controller)  
   2. 使用接口工具进行测试(postman)

# 编码

业务模块内部遵循分层原则：  
配置(spring)、  
数据传输层(controller)：只调用相关的service，不做任何逻辑处理  
服务层(service)  
数据层(mapper)  
转换(dto)、  
实体层(po)、  
核心部分(core)  
特定工具类(util)

## 4.1、接口规范

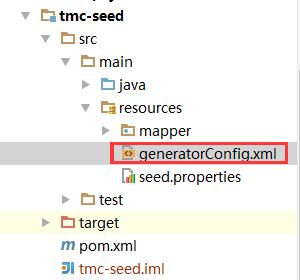
除 controller 层，所有参数必须具有确定性，不得是通用的 Map 或 JsonObject 类型  
2、除 controller 层，所有接口、方法不得带有隐含的前置条件  
3、controller 层尽可能使用确定性参数，避免接受/返回 Map/JsonObject 类型  
4、service必须以 interface / implement 方式编写，注解 @Service 应该放到 implement 之上  
5、service，controller层接口，注释清楚调用方法参数示例

6、mapper接口，注明

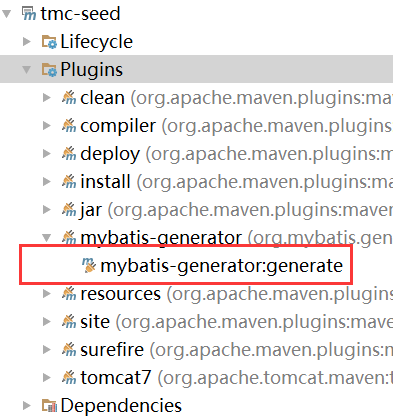
## 4.2、**方法命名**

查询方法前缀：query  
添加方法前缀：add  
更新方法前缀：update  
删除方法前缀：delete

## 4.3、po，mapper.xml，mapper自动生成



、



## 4.4、实例概述

**############# 详见tmc-seed模块 ##################**