# SuperBoot设计文档

作者：张帅

日期：2017-09-01

## 目录

[SuperBoot设计文档 1](#_Toc493239235)

[目录 2](#_Toc493239236)

[前言 3](#_Toc493239237)

[项目介绍 3](#_Toc493239238)

[框架规范约定 4](#_Toc493239239)

[接口规范 4](#_Toc493239240)

[组织结构 5](#_Toc493239241)

[技术选型 5](#_Toc493239242)

[模块介绍 5](#_Toc493239243)

[环境搭建 5](#_Toc493239244)

[开发指南 5](#_Toc493239245)

[资源下载 5](#_Toc493239246)

## 前言

随着技术日新月异，新技术新平台不断出现，对现如今的开发人员来说选择快速高效的框架进行项目开发，既能提高产出，又能节约时间。对目前节奏越来越快的互联网大环境来说，这是个必然的趋势，也是唯一的选择，针对目前的平台技术难度和口碑。本人结合多年工作经验开发一套敏捷框架，无需开发即可实现服务注册、服务发现、负载均衡、服务网关、配置中心、API管理、分布式事务、支撑平台、集成框架等功能，帮助大家更加专注实际业务而无需关心平台。

## 项目介绍

SuperBoot框架是基于SpringCloud、SpringBoot、Vue的敏捷开发框架，框架开发初衷是为了方便快速开发项目，无需关心基础代码的编写，可以更专注于业务本身。框架实现基于JWT Token授权验证，实现单点登录SSO，服务鉴权，实现Redis数据缓存，在保证数据一致性的前提下提高接口响应速度。无缝集成MongoDB数据库，提供对非结构型数据存储，解决关系型数据库瓶颈问题。集成swagger框架，实现自动API测试及调试功能，解决程序员最反感的编写技术文档的问题。数据源基于Druid，提供更高性能及SQL监控。框架提供统一异常处理，统一响应结果，增加对JPA、Mongo的AOP拦截，由Snowflake ID自动生成赋值主键，数据实体无需开发均可由Idea自动生成。增删改查默认基于方法名称即可实现，无需写具体SQL。

## 框架规范约定

### 接口规范

#### API接口命名规范

API接口基于RESTful,传统推荐方式为CRUD模式（Create、Read、Update、Delete），这样从技术角度看起来很清楚，但是对于客户或者非开发人员来说识别难度过大，故项目推荐使用DDD模式【DDD 使用实体（Entity）和聚合（Aggregate）来描述业务对象，还定义了服务（Service）、值对象（Value Object）和仓库（Repository）】用以解决业务领域或 DDD 边界上下文问题。比如对于银行账号开说，账户就是一个以 【领域对象】以及这些对象所提供的 【业务操作为中心】。接口URL可以定义规则如下：

POST /account ——新开一个账户。

PUT /account/{accountId}/Invalid ——注销一个已有的账户。

PUT /account/{accountId}/reduce ——从账户里扣掉一些钱。

PUT /account/{accountId}/increase ——往账户里存入一些钱。

GET /account/{accountId} ——通过账户 ID 加载相应的账户信息。

GET /account/{accountId}/transactions ——列出账户的交易历史。

#### 数据字典设计规范

1. 数据字典创建的时候必须包含TS（时间戳）、DR（删除标识）字段，数据删除为了提供历史追溯功能，数据采取逻辑删除非物理删除，即将数据的DR标识改为1则标识为删除状态。
2. 数据库主键采用BigInt类型，在IDEA生成实体类的时候，会自动增加@ID注解，在数据进行保存的时候，框架AOP会自动拦截save方法，利用反射对主键字段赋值。
3. 数据存储使用分表分库模式，项目启用多数据源，base数据源存储公共部分，业务数据库统一使用ext数据源，各项目生成实体类均放在项目下的ext中，repository的数据库操作DAO也需要放在ext中，操作base库的均放在super-boot-dao公共模块中。

### 组织结构

### 技术选型

#### 后端技术

#### 前端技术

### 模块介绍

### 环境搭建

### 开发指南

### 资源下载