# 三层架构规约

三层架构是为了让程序更好的做到“高内聚低耦合”，但只有三层架构是不够的，它存在着一些弊端或者说是不足。WCTF社区在编写时使用了一种更为详细的三层架构规约，在三层架构的基础上，对细节部分做了更完善的规定，并且这些规定可以在其他项目的复用。

## 各层分工

### Web层

1. Controller负责前台数据的组织，数据（逻辑可用性）校验，业务（可用性）校验交给Service层来处理。。
2. Controller中允许出现少量的业务逻辑来进行程序流程的组织，但务必要控制好量度。
3. 异常处理应该放在Controller层，但因为异常的特性，可以考虑使用过滤器等进行统一处理而不必像日志一样编码到各个Controlelr中。

### Service层

1. Service层是业务逻辑的主要实现部分，Controller和DAO层应尽量少包含业务逻辑。

### DAO层

1. DAO层与前端几乎没有耦合性，所以要尽可能的保证DAO中的方法的纯粹性。Service通过不同的DAO方法组合来达到需要完成的业务。
2. DAO层方法的返回值使用JDBC原生API的返回值。即查询语句返回ResultSet，更新语句返回整型。

## 细节规定

1. 面向接口编程不作为Service层的硬性规定。Service层在应用项目比较小时，可以选择直接写业务逻辑类，而不抽象业务逻辑接口。DAO层对接口 - 实现类的方式根据持久层框架来决定。比如MyBatis需要使用接口来定义DAO；Hibernate可以选择直接写实现类的方式来定义DAO。
2. 参数校验不会完全集中在Controller中，因为Controlelr不了解业务需求的数据格式。但Controller可以完成数据非空等校验。业务方面的数据格式校验放在Service层完成。
3. 规范Service层的返回值和参数列表。统一使用ApplicationDataSupport（应用数据流载体）来完成。具体细节参考第三节：应用数据流载体。

## 应用数据流载体

ApplicationDataSupport就像一个普通的实体对象，它被设计于存储参数数据、传递状态、发送响应码。而且它是JSON支持的，可以直接将其转为JSON对象。这使得它具有良好的跨编程语言性。

1. Controller层将参数全部封装到一个新创建的ApplicationDataSupport中，只需调用它的setParameter()方法即可。但有一个问题就是Controller并不知道Service需要哪些数据，因为Service层方法的参数列表都被规范成了ApplicationDataSupport。总不能把所有客户端传入的参数都原封不动传给Service吗？这就引出了另一个问题：基于注释编程。
2. Service层规定：业务层公共方法参数列表均为ApplicationDataSupport。所有业务层需要的参数都要写在文档注释中，按照一定的规范。@param data是ApplicationDataSupport的注释，下面是1~n个参数的文档注释写法：@param data$param1，@param data$param2 … @param data$paramn。所有“data$”后的字段都是Controller需要放入到ApplicationDataSupport中的参数。
3. **每个ApplicationDataSupport都有一个状态码，代表当前数据对象的状态，就像HTTP状态码一样。因为是JSON支持的，这样就可以从后端到前端均使用同一个对象来标识一个操作的不同状态。Service层应该根据不同的结果将Controller传入的ApplicationDataSupport设置上对应的状态码标识。**

# 状态码参考手册

|  |  |  |
| --- | --- | --- |
| 状态码 | 功能 | 描述 |
| 0 ~ 99 | 系统 | **------------------------------------------------------------** |
| 0 ~ 10 | 系统新增模块保留 | **-------------------------------** |
| 11 | 应用数据流载体数据异常 | 一般不会出现 |
| 12 | 登录状态异常 | 客户端传递的用户与Session中用户是否匹配 |
|  |  |  |
| 30 ~ 50 | 邮件发送模块 | **-------------------------------** |
| 30 | 邮件发送成功 |  |
| 31 | 邮件账号不正确 | 账号不正确 |
| 32 | 邮件发送失败 | 邮件服务器执行任务失败 |
|  |  |  |
| 100 ~ 300 | 用户模块 | ------------------------------------------------------------ |
| 100 ~ 119 | 用户模块保留 | **-------------------------------** |
|  |  |  |
| 120 ~ 149 | 用户登录模块 | **-------------------------------** |
| 120 | 用户登录成功 |  |
| 121 | 数据准备失败 | 用户登录需要的数据不完整或格式不正确 |
| 122 | 账号不存在 |  |
| 123 | 密码错误 |  |
| 124 | 验证码错误 |  |
| 125 | 数据库请求出现异常 |  |
|  |  |  |
| 150 ~ 179 | 用户注册模块 | **-------------------------------** |
| 150 | 注册成功 |  |
| 151 | 数据准备失败 |  |
| 152 | 账号已存在 |  |
| 153 | 数据库请求出现异常 |  |
|  |  |  |
| 180 ~ 199 | 用户修改信息模块 | **-------------------------------** |
| 180 | 修改成功 |  |
| 181 | 数据准备失败 |  |
| 182 | 数据库请求出现异常 |  |
|  |  |  |
| 200 ~ 239 | 用户修改密码 | **-------------------------------** |
| 200 | 修改成功 |  |
| 201 | 数据准备失败 |  |
| 202 | 原密码不正确 |  |
| 203 | 数据库请求出现异常 |  |
| 240 ~ 259 | 用户签到 | **-------------------------------** |
| 241 | 签到成功 |  |
| 242 | 数据准备失败 |  |
| 243 | 数据库请求出现异常 |  |
| 260 ~ 299 | 用户修改头像 | **-------------------------------** |
| 261 | 修改成功 |  |
| 262 | 数据准备失败 |  |
| 263 | 数据库请求出现异常 |  |
|  |  |  |
| 300 ~ 399 | 发贴模块 | ----------------------------------------- |
| 300 ~ 319 | 用户发贴 |  |
|  |  |  |
| 320 ~ 339 | 回贴 |  |
| 320 | 回贴成功 |  |
| 321 | 数据校验失败 |  |
| 322 | 数据准备失败 |  |
| 323 |  |  |
|  |  |  |
| 340 ~ 359 | 二级回贴 | **回复发贴回复** |
| 340 | 回复成功 |  |
| 341 | 数据准备失败 |  |
|  |  |  |
| 340 ~ 359 | 二级回贴 |  |
|  |  |  |
| 400 ~ 499 | 新闻模块 | ------------------------------------------------------------ |