# 开发规范

## 前言

一个项目的健壮性、可用性、可维护性，不在于使用最新的、最流行的框架，而在于是否从一而终保持代码的整洁；在于是否从一而终的保持代码及时重构；在于是否从一而终的保证代码的可读性。因此，为了保证代码的整洁，保证代码重构的质量，保证代码的持续可读性。特制订用于本项目的开发规范。本规范将涉及Java语言、数据库语言、JavaScript语言、CSS样式语言、HTML语言等语言的规范化（若以后会有其他语言加入到该项目，则进一步拓展）。

本项目的Java开发是基于IDEA进行的，因此Java规范（包括JSP）会与IDEA结合。而JavaScript、CSS、HTML会在sublime编辑器上进行，因此相关规范会与sublime关系较为紧密。数据库则统一规范化。

## Java规范

* 命名规范
* 代码中禁止出现无意义的名称（例：a1,a2…），任意一个位置出现的名称都应该有其明确的意义。
* 代码中出现的名称应该尽可能的含义准确，避免使用多义词。不要使用不是众所周知的缩写。明确是王道
* 避免使用编码。Java语言是强类型的，因此不要在命名中出现类型含义。禁止使用成员前缀。不要用前缀标出作用域，编辑器会告诉你使用的变量的作用域。接口类不要使用I开头，如果一定要添加的话，在实现类的末尾使用imp。
* 一律遵守驼峰式命名规则。

包名首字母小写。

类名首字母小写。类名应该是名字或名字短语，不应该是动词

方法名首字母小写。方法名应该是动词或动词短语。

* 函数
* 函数应该尽可能的短小。函数的缩进层级不应该多于一层或两层。
* 函数应该只做一件事。判断函数是否不知不止做了一件事，可以通过太函数是否能够再拆分出一个函数。而且新函数不仅是单纯地全是其实现，只做一件事的函数无法被合理地切分为多个区段
* 每个函数一个抽象层级
* 使用描述性的名称。这里有一个原则：如果每个例程都让你感到深合己意，那就是整洁代码。
* 参数个数要尽可能保持在三个以下，若超出三个以上，要考虑封装参数了
* 函数要么做什么事，要么回答什么事，不允许一个函数出现两者兼顾的情况
* Don’t repeat yourself! Don’t repeat yourself! Don’t repeat yourself!重要的事情说三遍！必须保证DRY原则
* 注释
* 唯一真正好的注释是想办法不去写注释
* 注释用来通基本信息
* 注释用来对意图进行解释
* 注释用来阐释一些比较晦涩的参数或返回值
* 注释用来警告其他程序员会出现某种后果
* TODO注释是一种程序员认为应该做，但由于某些原因还没有做的工作。
* 注释可以用来放大某种看来不合理之物的重要性。
* 公共API一定要提供良好的Javadoc
* 格式
  + - 所有的变量声明应该集中在类的开始
    - 变量声明结束和方法声明开始之间应间隔一行
    - 不同方法之间应该间隔一行
    - 代码块之间的缩进设置为4个空格符。其他参考统一的格式化文件
* 测试
  + - 保证每一个逻辑代码被测试覆盖。
    - 在编写不能通过的单元测试前，不可编写生产代码
    - 只可编写刚好无法通过的单元测试，不能编译也算不通过
    - 只可编写刚好足以通过当前失败测试的生产代码
    - 测试应该够快
    - 测试应该相互独立
    - 测试可以任何环境中重复通过
    - 测试应该有布尔值输出
    - 测试应及时编写