**Java代码规范**



项目名称： 表白墙网站开发

专业班级： 软件工程1903

小组成员：吴联想、王义博、郑航舰

指导教师： 杨枨老师

二零二一年十二月三日

JAVA代码规范

(一) 命名规约

【规范】类名使用UpperCamelCase 风格，必须遵从驼峰形式

【规范】方法名、参数名、成员变量、局部变量都统一使用lowerCamelCase 风格，必须遵从驼峰形式。

【规范】抽象类命名使用 Abstract 或 Base 开头 ； 异常类命名使用 Exception 结尾 ； 测试类命名以它要测试的类的名称开始，以 Test 结尾。

【规范】不允许任何魔法值（ 即未经定义的常量 ） 直接出现在代码中。

【规范】各层命名规约：

Service / DAO 层方法命名规约

1 ） 获取单个对象的方法用 get 做前缀。

2 ） 获取多个对象的方法用 list 做前缀（习惯：getXXXList）。

3 ） 获取统计值的方法用 count 做前缀。

4 ） 插入的方法用 save（ 推荐 ） 或 insert 做前缀。

5 ） 删除的方法用 remove（ 推荐 ） 或 delete 做前缀。

6 ） 修改的方法用 update 做前缀(或modify)。

(三) 格式规约

【规范】缩进只使用TAB键，不能使用空格键；方法体的开始、类的定义数据说明代码都要采用缩进方式

【规范】不要为了节省空间而把多个语句写在同一行

【规范】在 if/else/for/while/do 语句中必须使用大括号，即使只有一行代码

【规范】除常用方法(如 getXxx/isXxx)等外，不要在条件判断中执行其它复杂的语句，将复杂逻辑判断的结果赋值给一个有意义的布尔变量名，以提高可读性。

(四) OOP规约

【强制】避免通过一个类的对象引用访问此类的静态变量或静态方法，无谓增加编译器解析成本，直接用类名来访问即可。

【强制】所有的覆写方法，必须加@Override注解。

反例：getObject()与get0bject()的问题。一个是字母的O，一个是数字的0，加@Override可以准确判断是否覆盖成功。另外，如果在抽象类中对方法签名进行修改，其实现类会马上编译报错。

【推荐】当一个类有多个构造方法，或者多个同名方法，这些方法应该按顺序放置在一起，便于阅读。

【推荐】 类内方法定义顺序依次是：公有方法或保护方法 > 私有方法 > getter/setter方法。

（四）集合处理

【参考】合理利用好集合的有序性(sort)和稳定性(order)，避免集合的无序性(unsort)和不稳定性(unorder)带来的负面影响。

(五) 并发处理

【强制】获取单例对象要线程安全。在单例对象里面做操作也要保证线程安全。

说明：资源驱动类、工具类、单例工厂类都需要注意。

【强制】线程资源必须通过线程池提供，不允许在应用中自行显式创建线程。

(六) 控制语句

【强制】在if/else/for/while/do语句中必须使用大括号，即使只有一行代码，避免使用下面的形式：if (condition) statements;

【推荐】推荐尽量少用else， if-else的方式 ，如果使用要if-else if-else方式表达逻辑，【强制】请勿超过3层，超过请使用状态设计模式。

【推荐】除常用方法（如getXxx/isXxx）等外，不要在条件判断中执行复杂的语句，以提高可读性。

【推荐】循环体中的语句要考量性能，定义对象、变量操作尽量移至循环体外处理。

【推荐】接口入参保护，这种场景常见的是用于做批量操作的接口。

【参考】方法中需要进行参数校验的场景：

1） 调用频次低的方法。

2） 执行时间开销很大的方法，参数校验时间几乎可以忽略不计，但如果因为参数错误导致中间执行回退，或者错误，那得不偿失。

3） 需要极高稳定性和可用性的方法。

【参考】方法中不需要参数校验的场景：

1） 极有可能被循环调用的方法，不建议对参数进行校验。但在方法说明里必须注明外部参数检查。

2） 底层的方法调用频度都比较高，一般不校验。毕竟是像纯净水过滤的最后一道，参数错误不太可能到底层才会暴露问题。一般DAO层与Service层都在同一个应用中，部署在同一台服务器中，所以DAO的参数校验，可以省略。

(八) 注释规约

【规范】类、类属性、类方法的注释必须使用 Javadoc 规范，使用/\*\*内容\*/格式，不得使用 //xxx 方式。

【规范】所有的抽象方法(包括接口中的方法)必须要用 Javadoc 注释、除了返回值、参数、 异常说明外，还必须指出该方法做什么事情，实现什么功能。

说明:对子类的实现要求，或者调用注意事项，请一并说明。

【风格】方法内部单行注释，在被注释语句上方另起一行，使用//注释。方法内部多行注释使用/\* \*/注释，注意与代码对齐。

【规范】所有的枚举类型字段必须要有注释，说明每个数据项的用途。

【规范】代码修改的同时，注释也要进行相应的修改，尤其是参数、返回值、异常、核心逻辑 等的修改。

【规范】注释掉的代码尽量要配合说明，而不是简单的注释掉。

(八) 其它

【强制】在使用正则表达式时，利用好其预编译功能，可以有效加快正则匹配速度。

说明：不要在方法体内定义：Pattern pattern =Pattern.compile(规则);

【推荐】任何数据结构的使用都应限制大小。

说明：这点很难完全做到，但很多次的故障都是因为数据结构自增长，结果造成内存被吃光。

【推荐】对于“明确停止使用的代码和配置”，如方法、变量、类、配置文件、动态配置属性等要坚决从程序中清理出去，避免造成过多垃圾。清理这类垃圾代码是技术气场，不要有这样的观念：“不做不错，多做多错”。