Java语言技术管理文档

1. 命名规范

1.【强制】：使用英文或拼音命名，但不可在同一个命名中混用。不可使用纯拼音缩写的方式或无明确定义的英文缩写方式。

2.【强制】：类名采用大驼峰命名法。

3.【强制】方法名、参数名、成员变量、局部变量采用小驼峰命名法。

4.【强制】：常量命名一律使用大写字母，可使用下划线作为单词间的分隔。

5.【强制】：包名命名一律使用小写字母，可使用点分隔符作为单词间的分隔。

6.【推荐】：命名时尽量完整表达该元素的功用，避免使用含糊不明的命名。

7.【允许】：命名Service层方法时，用get作为获取单个元素方法的前缀；list作为获取多个元素方法的前缀；add/insert作为增加元素方法的前缀；delete作为删除元素方法的前缀；update作为修改元素方法的前缀

1. 常量定义

1.【强制】：所有常量在使用前必须先定义，不能在代码中直接出现未定义的常量。

2.【推荐】：按照常量功能将常量分开维护，便于分类管理。

1. 代码格式

1.【强制】：大括号使用规定：大括号内为空，则写成{}即可，不需要换行；如果是非空代码块则：1）左大括号前不换行 2）左大括号后换行3）右大括号前换行4）右大括号后还有else等代码则不换行；表示终止的右大括号后必须换行。

2.【强制】：缩进采用4空格缩进。

3.【强制】：任何运算符的左右两侧都需要加空格

4.【强制】：左右小括号与括号内字符间、与括号外关键字间都需要添加空格。

5.【强制】：定义和调用方法时，方法参数之间的逗号后需要添加空格。

5.【强制】：注释双斜线与注释内容间有一个空格。

6.【强制】：单行字符总数不能超过100个，若超出则需要换行。

7.【推荐】：在两个拥有独自自洽逻辑的语句块之间可以用单行空行隔开。

1. OOP规约

1.【强制】：直接用类名访问该类的静态变量或方法，而非通过一个类的对象来访问。

2.【强制】：所有覆写方法，必须添加@Override注解

3.【强制】：使用equals方法来比较相同类型的包装类对象

4.【强制】：定义DO/DTO/VO等POJO类时，不要设定属性默认值

5.【强制】：构造方法内不添加任何业务逻辑

6.【推荐】：类内方法定义次序为：public > private > getter/setter

7.【推荐】：在getter、setter方法中，不添加任何业务逻辑，参数名称与类成员名称保持一致。

8.【推荐】：字符串拼接使用append方法。

1. 并发处理

1.【强制】：创建线程或线程池时需要指定有意义的线程名称。

2.【强制】：线程资源必须通过线程池提供。

3.【强制】：获取单例对象需要保证线程安全，其中的方法也要保证线程安全。

4.【强制】：对多个资源、数据库表、对象同时加锁时，需要保持一致的加锁顺序，避免死锁问题。

5.【强制】：尽可能使加锁的代码块的工作量更小。

1. 控制语句

1.【强制】：使用控制语句时必须加大括号，即使语句内只有一行代码。

2.【强制】：每个switch/case块的最后都需要一个default语句，且每个case要么通过break/return等来终止，要么注释说明程序将继续执行到哪一个case为止。

3.【推荐】：if/else if/else、for循环语句尽量不要超过3层。

4.【推荐】：定义变量、对象、连接数据库等操作尽量移动到循环体外执行。

5.【推荐】：条件判断语句不要过于复杂。

1. 注释规范

1.【强制】：类、类属性、类方法的注释必须使用Javadoc规范，使用 /\*\*内容\*/ 格式，不得使用单行双斜线 // xxx方式。

2.【强制】：所有类都必须在开头注释创建者与创建日期

3.【强制】：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。

4.【强制】：方法内单行注释，使用 // ，多行注释，使用 /\* \*/

5.【推荐】：注释修改要与代码修改同步。

6.【允许】：每次注释掉代码段时，需要在最上方说明注释原因，或者直接删除代码段。

7.【允许】：代码注释力求以最少的字数表达最完整的代码段功能，避免不必要的注释，以便于阅读、修改。

1. 其他

1.【强制】初始化数据结构时，必须指定大小，避免其无限增长。

2.【推荐】及时清理不再使用的代码段或配置信息。

3.【强制】源文件以其最顶层的类命名，使用UTF-8编码

4.【强制】使用集合转数组时，必须使用集合的toArray方法