# Java语言技术管理文档

## 强制

1. 一个Java文件只能有一个public类，并且这个类名必须和文件名一样。
2. 类、方法、变量都需要写注释，简单说明用途。
3. 每行最多写100个字符，单行字符太多影响观感也不方便修改。
4. 统一用两个字符缩进，不许使用tab键缩进，以免复制代码时格式混乱。
5. 类名用大写驼峰法（如StudentInfo），方法和变量名用小写驼峰法（如getName）。
6. 常量全用大写加下划线（如static final MAX\_SCORE）。
7. 不使用import xxx.\*，需要使用什么类就具体导入该类。
8. 异常处理不能只写catch（Exception e），要写具体的异常类型。
9. 所有的if/for/while都要加{}，哪怕只有一行也不能省。
10. 不能使用中文、拼音当变量名，比如int shuzi = 1是不允许的。
11. 方法名和变量名都要浅显易懂，表明用途。
12. 不允许直接使用System.out.println()打印调试信息，要使用日志系统。
13. 写类的时候访问修饰符要写清楚（public、private、protected）。
14. 方法必须写清楚返回值类型和参数类型。
15. 重写方法必须加上@Override，否则容易出错还不提示。
16. 函数参数不能直接传null，要先判断或者用Optional。
17. 不要使用循环拼接字符，建议使用StringBuilder。
18. catch内必须有内容，处理或写日志。
19. 项目必须使用UTF-8编码，以免乱码。

## 推荐

1. 每个类中的顺序最好是：常量、变量、构造方法、公共方法、私有方法。
2. 方法尽量控制在30行之内，若太长考虑拆分成多个方法。
3. 多人合作一个项目时建议使用Maven或Gradle管理依赖。
4. 尽量使用final修饰不想被修改的变量、方法和类。
5. 多线程时使用synchronized 或 Concurrent 包下的类。
6. 尽量写单元测试，尤其是项目中写了工具类、接口逻辑之后。
7. 如果可以，使用Optional<T> 替代 null 值的返回。
8. 使用枚举（enum）替代定义一堆int常量的情况，更加美观整洁。
9. 构建复杂对象时建议使用Builder模式，更加简洁。
10. 数据类可以用 record（Java 16+）或者简洁结构体形式。
11. 尽量使用 try-with-resources 自动关闭流、连接等资源。
12. 可以使用Stream API简化集合的遍历处理。
13. 项目中建议引入Checkstyle 或 SonarLint 检查代码风格。
14. 单元测试时可以使用AssertJ 让断言更清晰。
15. 推荐使用构造函数注入，而不是在类里面写一堆 setter 方法，这样对象创建更明确，也更容易测试。
16. 推荐每个类只实现一个功能，实现高内聚低耦合。
17. 推荐方法参数不超过 4 个，超过可考虑封装成对象。

## 允许

1. 可以使用内部类和匿名类，但不能层层嵌套过深，导致结构过于复杂。
2. 可以使用switch表达式来化简代码。
3. 可以使用Lombok 减少 get/set/构造方法样板代码，但团队要统一风格。
4. 可以使用反射（Class.forName()），但不能滥用，以免增加调试难度。
5. 可以写一些工具类作为静态方法调用（如 Utils.print()），但不要滥用静态方法当 OOP 写法。
6. 可以适当使用 Lambda 表达式，比如 (a, b) -> a + b，但不建议太过复杂，代码看得懂最重要。
7. 可以用链式调用（method chaining），比如 builder.setA().setB().build();，但不能写出太长的链。