Java 技术管理规范文档

孙玮利 2022141461204

一、命名规范

a. 强制

1. 命名不得使用拼音、缩写、中文，不得以 _ 或 $ 开头/结尾。
2. 类名使用 UpperCamelCase，方法/变量用 lowerCamelCase。
3. 常量名用全大写+下划线，如 MAX_TIMEOUT_SECONDS。
4. 抽象类以 Abstract 或 Base 开头，异常类以 Exception 结尾。
5. 包名统一使用小写，采用有语义的英文单词。

b. 推荐

6. 变量命名体现语义（如 startTime，不写 st）。
7. 枚举类命名建议加 Enum 后缀，成员名大写+下划线。
8. 使用全名表达类能力或设计模式，如 EventFactory。
9. Service/DAO 实现类命名以 Impl 结尾。
10. 接口使用 -able 表示能力（如 Runnable、Serializable）。

c. 允许

11. 类名可根据上下文使用复数（如 Messages、Users），包名仍建议用单数。

二、代码格式与风格

a. 强制

12. 使用 4 个空格缩进，禁止使用 Tab。
13. 大括号 {} 写法必须标准，空块写为 {}。
14. 操作符、关键词之间保留一个空格，如 if (x == 1)。
15. 语句末尾必须加分号，即使 Java 允许省略。
16. IDE 编码统一使用 UTF-8，换行符使用 Unix 格式。
17. 方法参数多于 3 个建议换行对齐，避免单行过长。
18. Java 文件头必须标注作者、日期等信息。
19. 注释必须使用 Javadoc 风格 /** */ 进行描述类、方法。

b. 推荐

20. 同类构造方法和重载方法放在一起。
21. 类中方法顺序：公共方法 > 保护方法 > 私有方法 > getter/setter。
22. 方法长度建议不超过 80 行。
23. 不同逻辑之间用空行分隔。
24. 尽量不要对齐 = 来美化代码，避免维护困难。

c. 允许

25. 可使用 @formatter:off/on 控制特殊格式的代码块，但需备注说明。

三、OOP 设计规约

a. 强制

26. 所有覆盖方法必须加 @Override 注解。

27. 禁止在构造函数中编写业务逻辑，需放入 init()。
28. Object 的 equals 调用使用常量在前防止 NPE。
29. == 只能用于基本类型或同一引用比较，包装类需用 equals。
30. Float/Double 判断需使用误差范围或 BigDecimal。
31. 禁止用 BigDecimal(double) 构造函数。

b. 推荐

32. 所有 POJO 应重写 toString()。
33. POJO 不设默认值，避免误操作覆盖。
34. RPC/数据库交互禁止使用基本类型，使用包装类处理 null。
35. 构造函数应清晰展示依赖，不宜过长。

c. 允许

36. 可使用 Lombok（如 @Data）简化代码，但须谨慎控制生成的 equals/hashCode。

四、集合操作与泛型

a. 强制

37. 自定义对象做 Map Key 必须重写 equals 和 hashCode。
38. 使用 Collectors.toMap() 时必须传入合并函数，防止 key 冲突异常。
39. ArrayList.subList() 不可强转为 ArrayList。
40. 使用 toArray() 必须传入类型匹配的数组参数。

b. 推荐

41. 集合初始化时指定初始容量。
42. 遍历 Map 用 entrySet() 而非 keySet() 提高效率。
43. 使用 isEmpty() 判断集合是否为空，而非 size()==0。
44. 避免使用 Arrays.asList() 后直接修改集合。
45. foreach 中不要修改集合，应使用 iterator.remove()。
46. List 去重优先考虑转 Set。

c. 允许

47. 可以在性能不敏感场景中使用 Stream 操作，但需处理空指针与合并函数。

五、并发处理规范

a. 强制

48. 禁止直接 new Thread，应使用线程池创建线程。
49. 禁止使用 Executors.newXXX()，必须使用 ThreadPoolExecutor。
50. 使用 SimpleDateFormat 时，避免静态变量，可用 ThreadLocal 包装。
51. 必须显式释放 ThreadLocal 值，避免内存泄漏。
52. 自定义线程工厂时设置线程名，方便排查问题。

b. 推荐

53. 双重校验锁下的变量必须加 volatile。
54. 推荐使用 CountDownLatch、Semaphore、Future 等并发工具替代 sleep + flag。
55. 金融系统更新使用悲观锁（如：synchronized、数据库行锁）。

c. 允许

56. 可使用 volatile 解决可见性问题，但不可用于原子性控制。

六、安全与异常

a. 强制

57. 所有对外接口方法必须进行参数校验。
58. 禁止捕获 Throwable，只允许捕获 Exception 或自定义异常。
59. 异常处理必须记录日志，日志信息应包含异常堆栈。
60. 不允许捕获异常后不处理或只打印 e.printStackTrace()。