以下是一份基于Java语言的技术管理文档，参考Google、阿里巴巴等大厂规范，按强制(a)、推荐(b)、允许(c) 三个级别制定，共包含48条规则：

---

一、代码风格

1. 缩进：强制(a)使用4个空格，禁止使用Tab符。

2. 行宽：推荐(b)不超过120字符，超长时换行对齐。

3. 大括号：强制(a)K&R风格（左大括号不换行），如：

if (condition) { // 左括号同行

// ...

}

4. 空行：推荐(b)类内成员用1行空行分隔，方法间用2行空行分隔。

5. 空格：强制(a)运算符两侧、逗号后、类型转换后加空格，如：

int sum = a + b; // 运算符空格

method(arg1, arg2); // 逗号后空格

6. 包名：强制(a)全小写，无下划线（如 `com.example.project`）。

7. 类名/接口名：强制(a)大驼峰式（如 `UserService`）。

8. 方法名/变量名：强制(a)小驼峰式（如 `getUserId()`）。

9. 常量名：强制(a)全大写+下划线（如 `MAX\_COUNT`）。

二、编程规范

10. 访问权限：强制(a)成员变量私有化（`private`），通过方法访问。

11. `final`使用：推荐(b)常量、不可变类、工具类方法声明为 `final`。

12. 异常处理：

- 强制(a)禁止捕获`Exception`基类（需捕获具体异常）。

- 推荐(b)使用Try-With-Resources管理资源：

try (FileInputStream fis = new FileInputStream(file)) { ... }

13. `equals()`与`hashCode()`：强制(a)两者必须同时重写。

14. `toString()`：推荐(b)为实体类重写`toString()`。

15. 字符串拼接：强制(a)循环内用`StringBuilder`，禁用`+`。

16. 类型推断：推荐(b)使用`var`（Java 10+）简化局部变量声明。

17. 枚举比较：强制(a)用`==`代替`equals()`比较枚举值。

18. `Optional`：推荐(b)避免返回`null`，用`Optional`包装。

三、并发安全

19. 线程安全：

- 强制(a)共享变量用`volatile`或原子类（如`AtomicInteger`）。

- 推荐(b)优先使用`java.util.concurrent`包代替`synchronized`。

20. 锁粒度：推荐(b)减小同步代码块范围。

21. 线程池：强制(a)禁止手动创建线程，必须使用线程池。

---

四、OOP与设计

22. 继承：推荐(b)组合优于继承（除非是`is-a`关系）。

23. 接口与抽象类：

- 强制(a)接口只定义方法，抽象类可包含实现。

- 推荐(b)接口方法默认用`default`实现。

24. 工具类：强制(a)私有构造器+`final`类（防止继承/实例化）。

25. 构造器：推荐(b)超过4个参数的类使用Builder模式。

五、代码质量

26. 注释：

- 强制(a)Javadoc必须包含`@param`、`@return`、`@throws`。

- 推荐(b)废弃代码用`@Deprecated`标注并说明替代方案。

27. 魔法值：强制(a)禁止使用未解释的字面量（需定义常量）。

28. 代码重复：强制(a)DRY原则（Don't Repeat Yourself）。

29. 方法长度：推荐(b)不超过50行，超长需拆分。

30. 圈复杂度：推荐(b)方法圈复杂度≤10（使用Sonar检测）。

六、工程与工具

31. 包结构：推荐(b)按功能分包（如`com.example.user.service`）。

32. 依赖管理：强制(a)禁止循环依赖（使用ArchUnit检测）。

33. 日志：

- 强制(a)禁用`System.out.println()`，用`SLF4J`。

- 推荐(b)日志级别区分：`DEBUG`调试、`INFO`业务流程、`ERROR`异常。

34. 测试：

- 强制(a)核心逻辑必须有单元测试（覆盖率≥70%）。

- 推荐(b)测试类名以`Test`结尾（如 `UserServiceTest`）。

七、允许的例外(c)

35. `switch`穿透：允许(c)在`switch-case`中显式注释`// fall through`。

36. 静态导入：允许(c)导入静态常量（如`Math.PI`）或工具类方法。

37. 可变参数：允许(c)在工具类方法中使用（如 `String.format()`）。

38. `@SuppressWarnings`：允许(c)在确定无风险时抑制警告（需附原因注释）。

八、其他

39. `equals()`：强制(a)先检查`==`，再检查`instanceof`，最后字段比较。

40. 序列化：推荐(b)实现`Serializable`的类显式声明`serialVersionUID`。

41. 资源关闭：强制(a)I/O操作必须在`finally`块或Try-With-Resources中关闭。

42. 日期处理：强制(a)禁用`java.util.Date`，用`java.time.\*`（Java 8+）。

43. 集合初始化：推荐(b)用Guava或集合工具类初始化：

List<String> list = ImmutableList.of("a", "b"); // Guava

Map<String, Integer> map = Map.of("key", 1); // Java 9+

44. Lambda：推荐(b)替代匿名类，但避免超过3行的复杂Lambda。

45. `Optional`链式调用：允许(c)用`orElse()`/`orElseGet()`避免`null`。

46. 泛型：强制(a)禁止使用原生类型（如 `List` → `List<String>`）。

47. 注解顺序：推荐(b)按 `@Override` → `@Deprecated` → `@SuppressWarnings` 排列。

48. 代码格式化：强制(a)提交前用IDE或Spotless统一格式化。

执行建议：

强制(a)规则纳入CI流水线（Checkstyle/PMD/SonarQube）自动拦截。

推荐(b)规则在Code Review中人工检查，允许合理例外。

新项目严格遵循，存量代码渐进式重构