**ava 技术管理规范**

**代码风格**

**a. 强制**

1. 必须使用4个空格缩进，禁止使用Tab
2. 每行代码长度不超过120个字符
3. 类名必须使用大驼峰命名法(PascalCase)
4. 方法名和变量名必须使用小驼峰命名法(camelCase)
5. 常量名必须全部大写，单词间用下划线分隔
6. 必须使用大括号包裹所有控制语句体，即使只有一行代码
7. 必须为所有public类和方法编写Javadoc注释
8. 必须使用@Override注解来标明方法重写
9. 禁止使用魔法数字，必须定义为常量
10. 必须使用try-with-resources处理所有可关闭资源

**b. 推荐**

1. 推荐将大括号放在行尾而非新行
2. 推荐每行只声明一个变量
3. 推荐使用final修饰不会被修改的局部变量和参数
4. 推荐使用静态导入仅用于常量类
5. 推荐将复杂的条件表达式提取为布尔方法
6. 推荐使用枚举代替整数常量
7. 推荐使用空行分隔逻辑相关的代码块
8. 推荐将方法长度控制在50行以内
9. 推荐将类长度控制在500行以内
10. 推荐使用Optional而不是返回null

**c. 允许**

1. 允许在简单的getter/setter方法中使用单行形式
2. 允许在简单的lambda表达式中省略大括号
3. 允许在测试代码中使用较短的变量名
4. 允许在内部工具类中使用较宽松的命名规范
5. 允许在特定领域使用领域特定缩写

**异常处理**

**a. 强制**

1. 必须捕获特定异常而非通用的Exception
2. 必须在日志中记录异常堆栈信息
3. 禁止在catch块中忽略异常(空的catch块)
4. 必须为自定义异常提供有意义的描述信息
5. 必须在finally块中释放资源

**b. 推荐**

1. 推荐使用特定异常而非通用异常
2. 推荐在方法签名中声明可能抛出的检查异常
3. 推荐将异常转换为适合当前抽象层的异常类型
4. 推荐使用异常链保存原始异常信息

**并发编程**

**a. 强制**

1. 必须为共享可变数据添加适当的同步机制
2. 必须使用线程安全的集合类或进行适当同步
3. 禁止在锁内调用可能阻塞的外部方法

**b. 推荐**

1. 推荐使用java.util.concurrent包中的高级并发工具
2. 推荐使用不可变对象来简化并发编程
3. 推荐使用ExecutorService而非直接创建线程

**c. 允许**

1. 允许在性能关键路径使用双重检查锁定模式
2. 允许在特定场景使用volatile代替锁

**其他最佳实践**

**a. 强制**

1. 必须使用equals()方法比较对象而非==
2. 必须重写equals()时同时重写hashCode()
3. 必须为集合接口声明泛型类型

**b. 推荐**

1. 推荐使用构建器模式创建复杂对象
2. 推荐优先使用接口而非具体实现类
3. 推荐使用工厂方法而非直接构造函数
4. 推荐使用组合而非继承

**c. 允许**

1. 允许在性能敏感代码中使用原生类型数组