Alibaba Group
阿里巴巴集团

码出高效

阿里巴巴Java开发手册 1.4.0

Alibaba Java Coding Guidelines

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

前言

《阿里巴巴 Java 开发手册》是阿里巴巴集团技术团队的集体智慧结晶和经验总结，经历了多次大规模一线实战的检验及不断完善，系统化地整理成册，回馈给广大开发者。现代软件行业的高速发展对开发者的综合素质要求越来越高，因为不仅是编程知识点，其它维度的知识点也会影响到软件的最终交付质量。比如：数据库的表结构和索引设计缺陷可能带来软件上的架构缺陷或性能风险；工程结构混乱导致后续维护艰难；没有鉴权的漏洞代码易被黑客攻击等等。所以本手册以 Java 开发者为中心视角，划分为编程规约、异常日志、单元测试、安全规约、MySQL 数据库、工程结构、设计规约七个维度，再根据内容特征，细分成若干二级子目录。根据约束力强弱及故障敏感性，规约依次分为强制、推荐、参考三大类。对于规约条目的延伸信息中，“说明”对规约做了适当扩展和解释；“正例”提倡什么样的编码和实现方式；“反例”说明需要提防的雷区，以及真实的错误案例。

本手册的旨在**码出高效，码出质量**。现代软件架构的复杂性需要协同开发完成，如何高效地协同呢？无规矩不成方圆，无规范难以协同，比如，制订交通法规表面上是要限制行车权，实际上是保障公众的人身安全，试想如果没有限速，没有红绿灯，谁还敢上路行驶。对软件来说，适当的规范和标准绝不是消灭代码内容的创造性、优雅性，而是限制过度个性化，以一种普遍认可的统一方式一起做事，提升协作效率，降低沟通成本。代码的字里行间流淌的是软件系统的血液，质量的提升是尽可能少踩坑，杜绝踩重复的坑，切实提升系统稳定性，码出质量。

考虑到可以零距离地与众多开发同学进行互动，决定未来在线维护《手册》内容，此 1.4.0 的 PDF 版本，是最为详尽的版本，新增设计规约大章节，并增加若干条目；我们已经在 2017 杭州云栖大会上发布了阿里巴巴 Java 开发规约插件（点此下载），阿里云效（一站式企业协同研发云）也集成了代码规约扫描引擎。最后，《**码出高效**——阿里巴巴 Java 开发手册详解》即将出版，敬请关注。

目录

（注：浏览时请使用 PDF 左侧导航栏）

Java 开发手册

版本号	制定团队	更新日期	备注
1.4.0	阿里巴巴集团技术团队	**2018.5.20**	增加设计规约（详尽版）

一、编程规约

（一）命名风格

1. 【强制】代码中的命名均不能以下划线或美元符号开始，也不能以下划线或美元符号结束。

   反例：`_name / __name / $name / name_ / name$ / name__`

2. 【强制】代码中的命名严禁使用拼音与英文混合的方式，更不允许直接使用中文的方式。

   说明：正确的英文拼写和语法可以让阅读者易于理解，避免歧义。注意，即使纯拼音命名方式也要避免采用。

   正例：`alibaba / taobao / youku / hangzhou` 等国际通用的名称，可视同英文。

   反例：`DaZhePromotion [打折] / getPingfenByName() [评分] / int 某变量 = 3`

3. 【强制】类名使用 `UpperCamelCase` 风格，但以下情形例外：`DO / BO / DTO / VO / AO / PO / UID` 等。

   正例：`MarcoPolo / UserDO / XmlService / TcpUdpDeal / TaPromotion`

   反例：`macroPolo / UserDo / XMLService / TCPUDPDeal / TAPromotion`

4. 【强制】方法名、参数名、成员变量、局部变量都统一使用 `lowerCamelCase` 风格，必须遵从驼峰形式。

   正例： `localValue / getHttpMessage() / inputUserId`

5. 【强制】常量命名全部大写，单词间用下划线隔开，力求语义表达完整清楚，不要嫌名字长。

   正例：`MAX_STOCK_COUNT`

   反例：`MAX_COUNT`

6. 【强制】抽象类命名使用 `Abstract` 或 `Base` 开头；异常类命名使用 `Exception` 结尾；测试类命名以它要测试的类的名称开始，以 `Test` 结尾。

7. 【强制】类型与中括号紧挨相连来表示数组。

   正例：定义整形数组 int[] arrayDemo;

   反例：在 main 参数中，使用 String args[]来定义。

8. 【强制】`POJO` 类中布尔类型的变量，都不要加 `is` 前缀，否则部分框架解析会引起序列化错误。

   反例：定义为基本数据类型 `Boolean isDeleted` 的属性，它的方法也是 `isDeleted()`，RPC

框架在反向解析的时候，“误以为”对应的属性名称是 deleted，导致属性获取不到，进而抛出异常。

9. 【强制】包名统一使用小写，点分隔符之间有且仅有一个自然语义的英语单词。包名统一使用单数形式，但是类名如果有复数含义，类名可以使用复数形式。

正例：应用工具类包名为 com.alibaba.ai.util、类名为 MessageUtils（此规则参考 spring 的框架结构）

10. 【强制】杜绝完全不规范的缩写，避免望文不知义。

反例：AbstractClass“缩写”命名成 AbsClass；condition“缩写”命名成 condi，此类随意缩写严重降低了代码的可阅读性。

11. 【推荐】为了达到代码自解释的目标，任何自定义编程元素在命名时，使用尽量完整的单词组合来表达其意。

正例：在 JDK 中，表达原子更新的类名为：AtomicReferenceFieldUpdater。

反例：变量 int a 的随意命名方式。

12. 【推荐】如果模块、接口、类、方法使用了设计模式，在命名时需体现出具体模式。

说明：将设计模式体现在名字中，有利于阅读者快速理解架构设计理念。

正例：
```
public class OrderFactory;
public class LoginProxy;
public class ResourceObserver;
```

13. 【推荐】接口类中的方法和属性不要加任何修饰符号（public 也不要加），保持代码的简洁性，并加上有效的 Javadoc 注释。尽量不要在接口里定义变量，如果一定要定义变量，肯定是与接口方法相关，并且是整个应用的基础常量。

正例：接口方法签名 `void commit();`

接口基础常量 `String COMPANY = "alibaba";`

反例：接口方法定义 `public abstract void f();`

说明：JDK8 中接口允许有默认实现，那么这个 default 方法，是对所有实现类都有价值的默认实现。

14. 接口和实现类的命名有两套规则：

1）【强制】对于 Service 和 DAO 类，基于 SOA 的理念，暴露出来的服务一定是接口，内部的实现类用 Impl 的后缀与接口区别。

正例：CacheServiceImpl 实现 CacheService 接口。

2）【推荐】如果是形容能力的接口名称，取对应的形容词为接口名（通常是–able 的形式）。

正例：AbstractTranslator 实现 Translatable 接口。