# 阿珍爱上了阿强

# 编码规范

## (一) 命名规约

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： \_nam / \_\_name / $Object / name\_  / name$ / Object$  
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。

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

　　正例： alibaba / taobao / youku / hangzhou 等国际通用的名称，可视同英文。  
3. 【强制】类名使用UpperCamelCase风格，必须遵从驼峰形式，但以下情形例外：（领域模型的相关命名）DO / BO / DTO / VO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】中括号是数组类型的一部分，数组定义如下：String[] args;

　　反例：请勿使用String args[]的方式来定义。  
8. 【强制】POJO类中布尔类型的变量，都不要加is，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 isSuccess；的属性，它的方法也是isSuccess()，RPC框架在反向解析的时候，“以为”对应的属性名称是success，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.open.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】如果使用到了设计模式，建议在类名中体现出具体模式。

　　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思想。

　　正例：public class OrderFactory;

　　　　　public class LoginProxy;

　　　　　public class ResourceObserver;  
12. 【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 f(); 接口基础常量表示：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。  
13. 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，取对应的形容词做接口名（通常是–able的形式）。

　　正例：AbstractTranslator实现 Translatable。  
14. 【参考】枚举类名建议带上Enum后缀，枚举成员名称需要全大写，单词间用下划线隔开。

　　说明：枚举其实就是特殊的常量类，且构造方法被默认强制是私有。

　　正例：枚举名字：DealStatusEnum，成员名称：SUCCESS / UNKOWN\_REASON。  
15. 【参考】各层命名规约：

　　 A) Service/DAO层方法命名规约

　　　　1） 获取单个对象的方法用get做前缀。

　　　　2） 获取多个对象的方法用list做前缀。

　　　　3） 获取统计值的方法用count做前缀。

　　　　4） 插入的方法用save（推荐）或insert做前缀。

　　　　5） 删除的方法用remove（推荐）或delete做前缀。

　　　　6） 修改的方法用update做前缀。

　　B) 领域模型命名规约

　　　　1） 数据对象：xxxDO，xxx即为数据表名。

　　　　2） 数据传输对象：xxxDTO，xxx为业务领域相关的名称。

　　　　3） 展示对象：xxxVO，xxx一般为网页名称。

　　　　4） POJO是DO/DTO/BO/VO的统称，禁止命名成xxxPOJO。

## (二) 常量定义

1. 【强制】不允许出现任何魔法值（即未经定义的常量）直接出现在代码中。

　　反例： String key="Id#taobao\_"+tradeId；

　　　　　cache.put(key, value);  
2. 【强制】long或者Long初始赋值时，必须使用大写的L，不能是小写的l，小写容易跟数字1混淆，造成误解。

　　说明：Long a = 2l; 写的是数字的21，还是Long型的2?  
3. 【推荐】不要使用一个常量类维护所有常量，应该按常量功能进行归类，分开维护。如：缓存相关的常量放在类：CacheConsts下；系统配置相关的常量放在类：ConfigConsts下。

　　说明：大而全的常量类，非得使用查找功能才能定位到修改的常量，不利于理解和维护。  
4. 【推荐】常量的复用层次有五层：跨应用共享常量、应用内共享常量、子工程内共享常量、包内共享常量、类内共享常量。

　　1） 跨应用共享常量：放置在二方库中，通常是client.jar中的constant目录下。

　　2） 应用内共享常量：放置在一方库的modules中的constant目录下。

　　　　反例：易懂变量也要统一定义成应用内共享常量，两位攻城师在两个类中分别定义 了表示“是”的变量：

　　　　　　 类A中：public static final String YES = "yes";

　　　　　　 类B中：public static final String YES = "y"; A.YES.equals(B.YES)，预期是true，但实际返回为false，导致产生线上问题。  
　　3） 子工程内部共享常量：即在当前子工程的constant目录下。

　　4） 包内共享常量：即在当前包下单独的constant目录下。

　　5） 类内共享常量：直接在类内部private static final定义。  
5. 【推荐】如果变量值仅在一个范围内变化用Enum类。如果还带有名称之外的延伸属性，必须使用Enum类，下面正例中的数字就是延伸信息，表示星期几。

　　正例：public Enum{ MONDAY(1), TUESDAY(2), WEDNESDAY(3), THURSDAY(4), FRIDAY(5), SATURDAY(6), SUNDAY(7);}

## (三) 格式规约

1. 【强制】大括号的使用约定。如果是大括号内为空，则简洁地写成{}即可，不需要换行；如果是非空代码块则：

　　1） 左大括号前不换行。

　　2） 左大括号后换行。

　　3） 右大括号前换行。

　　4） 右大括号后还有else等代码则不换行；表示终止右大括号后必须换行。  
2. 【强制】 左括号和后一个字符之间不出现空格；同样，右括号和前一个字符之间也不出现空格。详见第5条下方正例提示。  
3. 【强制】if/for/while/switch/do等保留字与左右括号之间都必须加空格。  
4. 【强制】任何运算符左右必须加一个空格。

　　说明：运算符包括赋值运算符=、逻辑运算符&&、加减乘除符号、三目运行符等。  
5. 【强制】缩进采用4个空格，禁止使用tab字符。  
说明： 如果使用 tab 缩进，必须设置 缩进，必须设置 缩进，必须设置 缩进，必须设置 缩进，必须设置 缩进，必须设置 1个 tab 为 4个空格。 IDEA 设置 tab 为 4个空格时， 请勿勾选 Use tab character ；而在 eclipse 中，必须勾选 insert spaces for tabs 。

## 要了解更多信息并获取 OneNote，请访问 [www.onenote.com](http://go.microsoft.com/fwlink/?LinkID=523891)。