

|  |
| --- |
| 武汉易瑞信息技术股份有限公司  2017年3月 |
| 总部：武汉市东西湖区临空港大道融园国际23层  武汉研发分部：湖北省武汉市武昌区中北路长城汇T2,16层1601 |
| 电话: 027-87278766  传真: 86-10-5874-9001  网址：Http://www.isoftstone.com |



文件编号：201504-ETP-04-05

正 本

e-Ray开发手册V1.0

修订记录：

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 修订人 | 修订日期 | 修订描述 |
| V1.0 | 胡黄驰 | 2017-07-28 | 创建 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目 录

[1. 编程规约 1](#_Toc489015251)

[1.1. 命名风格 1](#_Toc489015252)

[1.2. 常量定义 2](#_Toc489015253)

[1.3. 代码格式 2](#_Toc489015254)

[1.4. OOP规约 3](#_Toc489015255)

[1.5. 控制语句 4](#_Toc489015256)

[1.6. 注释规约 5](#_Toc489015257)

[2. 异常日志 6](#_Toc489015258)

[2.1. 异常处理 6](#_Toc489015259)

# 编程规约

## 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/$Object/name\_/name$/Object$

2、【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

3、【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。

4、【强制】中括号是数组类型的一部分，数组定义如下：String[] args;

反例：使用String args[]的方式来定义。

5、【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.eray.open.util、类名为MessageUtils

6、【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。

7、【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。

8、【参考】Service/DAO层方法命名规约

1） 获取单个对象的方法用get/selectOne做前缀，或者selectXxxXxByPrimaryKey。

2） 获取多个对象的方法用List做后缀。

3） 获取统计值的方法用Count做后缀。

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。

9、【参考】对于变量不同类型的命名规则：

1）String：str开头

2）Boolean/boolean：bl开头

3）Integer/int：i开头

4）Long/long：l*（L的小写）*开头

5）Double/double：d开头

6）Float/float：f开头

7）Char：char开头

8）Date：Date结尾

9）集合类型：集合接口结尾，比如：xxxxxMap、xxxxxList等。

## 常量定义

1、【强制】不允许任何未经定义的常量直接出现在代码中。

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。

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。

## 代码格式

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；表示终止的右大括号后必须换行。

2、【强制】 左小括号和字符之间不出现空格；同样，右小括号和字符之间也不出现空格。对于运算符左右两边各加一个空格。

正例：if (a == b) c = a + b；

反例：if (空格a==b空格) c=a+b

3、【强制】if/for/while/switch/do等保留字与括号之间都必须加空格。

4、【强制】任何二目、三目运算符的左右两边都需要加一个空格。 说明：运算符包括赋值运算符=、逻辑运算符&&、加减乘除符号等。

5、【强制】方法参数在定义和传入时，多个参数逗号后边必须加空格。

正例：下例中实参的"a",后边必须要有一个空格。 method("a", "b", "c");

6、【强制】IDE的text file encoding设置为UTF-8

7、JSP/JS/CSS/JAVA，禁止使用IDE的自动格式化功能（Ctrl+Shift+F），JSP/JS代码结构清晰。

## OOP规约

1、【强制】避免通过一个类的对象引用访问此类的静态变量或静态方法，无谓增加编译器解析成本，直接用类名来访问即可。

2、【强制】所有的覆写方法，必须加@Override注解。

3、【强制】不能使用过时的类或方法。

4、【强制】Object的equals方法容易抛空指针异常，应使用常量或确定有值的对象来调用equals。

正例： "test".equals(object);

反例： object.equals("test");

5、【强制】所有的相同类型的包装类对象之间值的比较，全部使用equals方法比较。

6、【强制】构造方法里面禁止加入任何业务逻辑，如果有初始化逻辑，请放在init方法中。

7、【强制】POJO类必须写toString方法。使用IDE的中工具：source> generate toString时，如果继承了另一个POJO类，注意在前面加一下super.toString。

说明：在方法执行抛出异常时，可以直接调用POJO的toString()方法打印其属性值，便于排查问题。

8、【推荐】当一个类有多个构造方法，或者多个同名方法，这些方法应该按顺序放置在一起，便于阅读。

9、【推荐】 类内方法定义顺序依次是：公有方法或保护方法 > 私有方法 > getter/setter方法。

10、【推荐】setter方法中，参数名称与类成员变量名称一致，this.成员名 = 参数名。在getter/setter方法中，不要增加业务逻辑，增加排查问题的难度。

11、【推荐】循环体内，字符串的连接方式，使用StringBuilder的append方法进行扩展。

12、【推荐】慎用Object的clone方法来拷贝对象。

## 控制语句

1、【强制】在一个switch块内，每个case要么通过break/return等来终止，要么注释说明程序将继续执行到哪一个case为止；在一个switch块内，都必须包含一个default语句并且放在最后，即使它什么代码也没有。

2、 【强制】在if/else/for/while/do语句中必须使用大括号。即使只有一行代码，避免使用单行的形式：if (condition) statements;

3、【推荐】循环体中的语句要考量性能，以下操作尽量移至循环体外处理，如定义对象、变量、获取数据库连接，进行不必要的try-catch操作（这个try-catch是否可以移至循环体外）。

## 注释规约

1、【强制】类、类属性、类方法的注释必须使用Javadoc规范，使用/\*\*内容\*/格式，不得使用//xxx方式。

说明：在IDE编辑窗口中，Javadoc方式会提示相关注释，生成Javadoc可以正确输出相应注释；在IDE中，工程调用方法时，不进入方法即可悬浮提示方法、参数、返回值的意义，提高阅读效率。

2、【强制】所有的方法（包括接口中的方法）必须要用Javadoc注释、除了返回值、参数、异常说明外，还必须指出该方法做什么事情，实现什么功能。 说明：对子类的实现要求，或者调用注意事项，请一并说明。

3. 【强制】所有的类都必须添加创建者和创建日期，必要时还需要写入修改人、修改日期和修改原因。在其它创建者的类中新建方法、变量、修改代码块也需要添加创建者和创建日期，特别是修改代码块时需要描述修改原因。

4. 【强制】方法内部单行注释，在被注释语句上方另起一行，使用//注释。方法内部多行注释使用/\* \*/注释，注意与代码对齐。

5. 【强制】所有的枚举类型字段必须要有注释，说明每个数据项的用途。

6、【推荐】代码修改的同时，注释也要进行相应的修改，尤其是参数、返回值、异常、核心逻辑等的修改。

7、【参考】合理处理注释掉的代码。在上方详细说明，而不是简单的注释掉。如果无用，则删除。

说明：代码被注释掉有两种可能性：1）后续会恢复此段代码逻辑。2）永久不用。前者如果没有备注信息，难以知晓注释动机。后者建议直接删掉

8、【参考】对于注释的要求：第一、能够准确反应设计思想和代码逻辑；第二、能够描述业务含义，使别的程序员能够迅速了解到代码背后的信息。

9、【参考】代码在实现时有逻辑或性能隐患而暂时无法处理时 建议使用FIXME、TODO标签进行标示。

# 异常日志

## 异常处理

1、【强制】异常不要用来做流程控制，条件控制，因为异常的处理效率比条件分支低。

2、【强制】对大段代码进行try-catch，这是不负责任的表现。catch时请分清稳定代码和非稳定代码，稳定代码指的是无论如何不会出错的代码。对于非稳定代码的catch尽可能进行区分异常类型，再做对应的异常处理。

3、【强制】有try块放到了事务代码中，catch异常后，如果需要回滚事务，一定要注意手动回滚事务。

4、【强制】捕获异常是为了处理它，不要捕获了却什么都不处理而抛弃之，如果不想处理它，请将该异常抛给它的调用者。最外层的业务使用者，必须处理异常，将其转化为用户可以理解的内容。

5、【强制】不能在finally块中使用return。

6、【推荐】方法的返回值可以为null，不强制返回空集合，或者空对象等，必须添加注释充分说明什么情况下会返回null值。