ISC微服务

Java编码规范说明书

版本：1.0

编制人：王孝坤

目录

[ISC微服务 1](#_Toc5969165)

[Java编码规范说明书 1](#_Toc5969166)

[1. 引言 3](#_Toc5969167)

[2. 文件组织-源文件 3](#_Toc5969168)

[2.1.文件名 3](#_Toc5969169)

[2.2.文件编码 3](#_Toc5969170)

[2.3.源文件 3](#_Toc5969171)

[2.4.许可证或版权信息 3](#_Toc5969172)

[2.5. package语句 3](#_Toc5969173)

[2.6. import语句 4](#_Toc5969174)

[2.7.类声明 4](#_Toc5969175)

[3. 命名约定 4](#_Toc5969176)

[3.1.包名 4](#_Toc5969177)

[3.2.类名 4](#_Toc5969178)

[3.3.方法名 4](#_Toc5969179)

[3.4.常量名 4](#_Toc5969180)

[3.5.非常量字段名 4](#_Toc5969181)

[3.6.参数名 4](#_Toc5969182)

[3.7.局部变量名 5](#_Toc5969183)

[4. 编码约束 5](#_Toc5969184)

[4.1.行数限定 5](#_Toc5969185)

[4.2.循环判断逻辑限定 5](#_Toc5969186)

[5. 注释 5](#_Toc5969187)

[5.1.文件注释 5](#_Toc5969188)

[5.2.类的注释 5](#_Toc5969189)

[5.3.方法的注释 6](#_Toc5969190)

[6. 分隔和缩进 6](#_Toc5969191)

[6.1.使用空行 6](#_Toc5969192)

[6.2.使用空格符 6](#_Toc5969193)

[6.3.使用换行符 7](#_Toc5969194)

[6.4.使用小括号 7](#_Toc5969195)

[6.5.使用花括号 7](#_Toc5969196)

[6.6.使用缩进 7](#_Toc5969197)

[7. 异常处理 7](#_Toc5969198)

[7.1.异常信息传递 7](#_Toc5969199)

[7.2.异常封装 7](#_Toc5969200)

# 引言

本文是一套面向ISC微服务框架Java开发人员所应遵循的开发规范。

俗话说国有国法，行有行规，软件工程里有明确的软件管理流程规范以及代码编写规范。程序是由开发语言组织起来的，遵循好的业界公认的开发规范的程序代码能提高代码的可读性，更能提高代码的延续性和可扩展性。正是基于提高团队协作效率，让ISC微服务框架java程序能遵循统一的业界标准，并提高延续性，可扩展性和开发质量，编写了此java开发规范。

遵循此开发规范希望能达到以下目的：

* 代码风格一致
* 更高的代码可读性和代码质量
* 更容易定位问题和排除故障
* 提高程序的可扩展性和延续性
* 在有人员变动情况下，新人更容易接手

本规范是初版，只适用于一般情况的通用规范，并不能覆盖所有的情况

# 文件组织-源文件

## 2.1.文件名

源文件以其最顶层的类名来命名，大小写敏感，文件扩展名为.java

## 2.2.文件编码

源文件编码格式为UTF-8。

## 2.3.源文件

一个源文件包含(按顺序地包含)：

* 许可证或版权信息
* package语句
* import语句
* 一个顶级类(只有一个)

## 2.4.许可证或版权信息

文件包含许可证或版权信息，那么它应当被放在文件最前面并注明以下信息：版权: Copyright 2007-2016 ISC All Rights Reserved.

## 2.5. package语句

package语句不换行，(即package语句写在一行里)

## 2.6. import语句

import不要使用通配符，即不要出现类似这样的import语句：import java.util.\*;import语句不换行

## 2.7.类声明

有一个顶级类声明。每个顶级类都在一个与它同名的源文件中(当然，还包含.java后缀)。类的成员顺序对易学性有很大的影响，但这也不存在唯一的通用法则。不同的类对成员的排序可能是不同的。 最重要的一点，每个类应该以某种逻辑去排序它的成员，维护者应该要能解释这种排序逻辑。比如， 新的方法不能总是习惯性地添加到类的结尾，因为这样就是按时间顺序而非某种逻辑来排序的。当一个类有多个构造函数，或是多个同名方法，这些函数/方法应该按顺序出现在一起，中间不要放进其它函数/方法。

# 命名约定

## 3.1.包名

包名全部小写，连续的单词只是简单地连接起来，不使用下划线。类名都以UpperCamelCase风格编写。

## 3.2.类名

类名通常是名词或名词短语，接口名称有时可能是形容词或形容词短语。现在还没有特定的规则或行之有效的约定来命名注解类型。测试类的命名以它要测试的类的名称开始，以Test结束。例如，HashTest或HashIntegrationTest。

## 3.3.方法名

方法名都以lowerCamelCase风格编写。方法名通常是动词或动词短语。

## 3.4.常量名

常量名命名模式为CONSTANT\_CASE，全部字母大写，用下划线分隔单词。

## 3.5.非常量字段名

非常量字段名以lowerCamelCase风格编写。

## 3.6.参数名

参数名以lowerCamelCase风格编写。参数应该避免用单个字符命名。

## 3.7.局部变量名

局部变量名以lowerCamelCase风格编写，比起其它类型的名称，局部变量名可以有更为宽松的缩写。虽然缩写更宽松，但还是要避免用单字符进行命名，除了临时变量和循环变量。即使局部变量是final和不可改变的，也不应该把它示为常量，自然也不能用常量的规则去命名它。

# 编码约束

## 4.1.行数限定

一个java程序文件中的代码行数不能超过2000行，一个方法不能超过200行，超过200行考虑方法封装。

## 4.2.循环判断逻辑限定

* If else嵌套不得超过3层，否则将非常难于理解和测试，超过可考虑函数封装；
* 方法（构造器）参数在5个以内 ，太多的方法（构造器）参数影响代码可读性。考虑用值对象代替这些参数或重新设计。

# 注释

注释模板请参考模板文件（导入eclipse即可）：

../style/formatter.xml

## 5.1.文件注释

如下：

/\*\*

\*

\* Copyright: Copyright 2007-2016 ISC All Rights Reserved.

\* **@Title**: DataSynQuartz.java

\* **@Package** com.midea.isc.admin.controller

\* **@Description**: 下拉值服务

\* **@author** wangxk7

\* **@date** 2019-4-12 15:54:47

\* **@lastModifier** wangxk7

\*/

## 5.2.类的注释

如下：

/\*\*

\*

\* **@Description** 物料查询Dao

\* **@author** wangxk7

\* **@date** 2019-4-12 15:54:47

\*/

**public** **interface** MatierialBillingDao

{

List<MatierialVO> queryBMatierialList();

}

## 5.3.方法的注释

如下

/\*\*

\*

\* **@Description** 查询物料信息列表

\* **@author** wangxk7

\* **@date** 2019-4-12 15:54:47

\* **@return** 物料信息列表

\* **@lastModifier** wangxk7

\*/

List<MatierialVO> queryBMatierialList();

# 分隔和缩进

## 6.1.使用空行

在下述情况下使用单行的空白行来分隔：

* 在方法之间
* 在方法内部代码的逻辑段落小节之间
* 在方法中声明局部变量之后，具体的Java语句之前
* 在注释行之前
* 在同一个源文件中定义的多个类/接口之间使用双行的空白行来分隔

## 6.2.使用空格符

除正常的成分之间以空格符分隔名（如数据类型和变量名之间），在下述情况下也应使用一个空格符来分隔：

* 运算符和运算符之间，如：　c = a + b；
* 在参数列表中的逗号后面，如： void m1（int year， int month） {}
* 在if，for，while待语句之后　如： if （a > b） {}

## 6.3.使用换行符

不建议在一行中写多条语句，一条语句的长度一般超过了80个字符时，应该换行; 当一行表达式不能在一行内显示，请按下列顺序要求拆行：

* 在"("或"="符号后拆行
* 在","拆行
* 在一个操作符后拆行

## 6.4.使用小括号

可以使用小括号来强行规定运算顺序

## 6.5.使用花括号

{这个独占一行，开始位置另起一行，这样和}形成对齐。

## 6.6.使用缩进

在下述情况下应用缩进

* 1）类中的成分
* 2）方法体或语句块中的成分
* 3）换行时的非起始行

**具体的缩进请参考模板文件：../style/codetemplates.xml**

# 异常处理

## 7.1.异常信息传递

重新抛出的异常必须保留原来的异常，即throw new NewException("message", e);而不能写成throw new NewException("message")。在所有异常被捕获且没有重新抛出的地方必须写日志。如果属于正常异常的空异常处理块必须注释说明原因，否则不允许空的catch块。

## 7.2.异常封装

框架尽量捕获低级异常并封装成高级异常重新抛出，隐藏低级异常的细节，方便系统能够更好的跟踪运行情况。如果一个层要抛出多个异常，那么所有自定义异常必须统一继承一个父类异常,这样上层可以通过父类异常捕获。