一、Java编程规范

1. 命名规范

命名规范使得程序更易理解，可读性更强。并且能够提供函数和标识符的信息。

文件命名规范

java程序使用如下的文件名后缀：

文件类型 后缀

Java 源文件 .java

Java 字节码文件 .class

对系统的文件命名方式有待于根据实际业务决定。

包命名规范

包名应该唯一，它的前缀可以为任何小写的ASCII字符但必须是顶级域名，目前包括com, edu, gov, mil, net, org,或者ISO标准3166,1981中两个字符的国别代码。包名接下来的部分按照公司内部的命名规范，这些规范指出了某种目录名，主要包括部门，项目，机器，或者登录名。

类命名规范

类名应该是名词，并且是大小写混合的。首字母要大写。尽量保证类名简单并且描述性强。避免使用只取单词首字母的简写或者单词的缩写形式，除非缩写形式比单词的完整形式更常用（例如：URL或者HTML）。

文件名必须和public的类名保持一致，注意大小写（JBuilder等一些编译器可以忽略大小写，要特别注意）.

接口命名规范

接口命名方式与类命名方式相同。

方法命名规范

方法名应该为动词，并且是大小写混合的。首字母要小写，方法名的第

二个单词的第一个字母大写。

变量命名规范

变量，以及所有的类实例应为首字母小写的大小写混合形式。变量名的第二个单词的首字母大写。变量名的首字母不能为下划线或者$符。

变量名应该尽可能的短小，但要有意义。变量名应该便于记忆，也就是说变量名应该尽可能的做到见名知意。除了暂时使用的变量外（一般用于循环变量），应该避免使用只有一个字母的变量名。对于临时变量一般说来：i，j，k，m，n代表整型变量。c，d，e代表字符型变量。

常量命名规范

声明为类常量的变量或者ANSI常量应该全部为大写字母，并且每个单词间用下划线“\_”隔开。为了便于调试，应避免使用ANSI常量。

2、注释规范

Java 提供了两种类型的注释：程序注释和文档注释。程序注释是由分隔符/\*…\*/，和// 隔开的部分，这些注释和C++ 中的注释一样。文档注释（即“doc 注释”）是Java 独有的。由分隔符/\*\*…\*/隔开。使用javadoc工具能够将文档注释抽取出来形成HTML文件。程序注释主要是对程序的某部分具体实现方式的注释。文档注释是对程序的描述性注释，主要是提供给不需要了解程序具体实现的开发者使用。注释应该是代码的概括性描述，提供不易直接从代码中得到的信息，并且只包含对阅读和理解程序有用的信息。例如注释中包含相应的包如何编译或者在哪个目录下，而不应该包括这个包驻留在哪儿的信息。注释中可以描述一些精妙的算法和一些不易理解的设计思想，但应该避免那些在程序代码中很清楚的表达出来的信息。尽可能的避免过时的信息。错误的注释比没有注释更有害。经常性的注释有时也反映出代码质量的低下。

程序注释

程序注释有四种格式：块注释格式，单行注释，跟随注释，行尾注释

块注释格式

块注释主要用于描述：文件、方法、数据结构和算法。一般在文件或者方法定义的之前使用。也可以用在方法定义里面，如果块注释放在函数或者方法定义里，它必须与它所描述的代码具有相同的缩进形式。

块注释应该用一个空行开头，以便于代码部分区分开来。

单行注释

比较短的注释可以放在一行中，但必须与它所跟随的代码有相同的缩进。如果注释不可以放在一行，那么必须按照块注释的格式来写。单行的注释会被解释为一空行。

跟随注释

非常短的注释可以和它所描述的代码放在同一行。但要保证代码和注释之间有足够的间隔。在同一块代码中不止一个这样的注释时它们应该对齐。

行尾注释

注释标记?//”能够注释一行或者该行由“//”开始直到行尾的部分。行尾注释不能用在多行文本注释中。但它可以将多行代码注释

文档注释 文档注释描述了Java类，接口，构造函数，方法和属性。每个文档注释放在文档注释符/\*\*…\*/中。每个类、接口或者成员只在声明的地方之前有一个文档注释。

注意：最外层的类或者接口的文档注释不用缩进。但它的成员的文档注释与成员的声明具有相同的缩进格式。

文档注释从第二行开始与第一行相比缩进一个字符。即注释的每一行的星号要对齐。

如果关于类、接口、变量或者方法的注释信息不是为文档信息提供的，就应该使用块注释或者单行注释（这两类注释应该使用在声明之后）。关于类实现方法的详细信息应该放在类语句后的块注释中，而不应该放在类的文档注释中。

文档注释方法或者构造函数的定义块内。Java会将文档注释后的第一个声明于文档注释关联起来。

方法类型为protected、public的必须提供方法注释

3、构造方法规范

工具类不允许有public或default构造方法。这可以避免不必要的创建实例。

一个类里面的所有public方法都是static的，则这个类就可以作为工具类。其它类使用工具类的方法时，无需创建其实例，直接使用其方法即可。

例：在sysframework包中存在着的一个典型的工具类的例子：

com.sinosoft.sysframework.common.util.StringUtils，它的所有public方法都是static的，里面只有一个private构造方法。

修饰符规范

按照Java语言规范，修饰符按如下顺序组织：

public

protected

private

abstract

static

final

transient

volatile

synchronized

native

strictfp

变量声明

每行定义变量数目每行定义的变量数目必须有且只有一个。

变量初始化

在声明局部变量的时候就要初始化变量。

变量定义位置

在for循环里的循环变量可以在for语句里面定义。

for (int i = 0; i < maxLoops; i++) { ... }

注意：应避免局部变量屏蔽了外层变量的作用范围。也就是说不要在内部块中声明一个与外部块某个变量同名的变量。

数组的定义

数组的[]应该放在类型名的后面，而不是变量名的后面。

类和接口声明

类和接口的声明应该遵循以下规范：

A.在方法名和参数列表的圆括号以及括号后的第一个参数间都没有空格。

B.开括号“{”必须与声明语句放在同一行。

C.闭括号“}”必须与声明语句有相同的缩进格式。

D.如果类或者接口实现内容为空，则可以将“}”放在“{”后面。

E.方法之间要用一个空行隔开。

4、语句规范

不允许出现空语句的情况，典型的错误案例是：

A.一行里只有一个分号

B.一个语句后面连续出现几个分号

不允许出现无意义语句，如 policyNo = policyNo;

简单语句

每行最多包含一个语句。

组合语句

组合语句使用大括号括起来的一串语句。

A.号中的语句比组合语句多一级缩进。

B.号“{”应该放在组合语句前的语句末尾。闭括号“}”应该放在新的一行并与组合语句开始前的第一个语句有相同的缩进。

C.语句是控制语句的一部分时，所有的语句都要用大括号围起来，即使只有一个语句也要用括号，例如在if-else或者for语句中。这样避免在添加语句时忘记添加括号而导致程序产生bug。

return语句

return语句在有返回值时不需要使用圆括号，除非使用圆括号在某些特定的情况下能够提高代码的可读性。

注意：在if语句中必须使用大括号，避免使用下面的形式。

if ( condition) //禁止使用! 这种方式忽略了大括号{}，容易导致错误。

switch语句

使用格式如下：

switch ( condition) {

case ABC:

statements;

/\* 不包括break语句 \*/

case DEF:

statements;

break;

case XYZ:

statements;

break;

default:

statements;

break;

}

A.每一个case语句都要包含break语句，如果不包含应该说明正常情况下应在什么时候退出switch语句。

B.每一个switch语句都要包含default情况，default语句后的break语句不是多余的，它能避免在以后添加其他的case语句时而可能导致的错误。

C.一般switch语句中condition的值是整型数。

for语句

for语句使用格式如下：

for (int i = 0; i < maxLoops; i++) {

...

}

try-catch语句

try-catch语句使用格式如下：

try {

statements;

} catch (ExceptionClass e) {

statements;

}

try-catch语句后也可以跟随finally。无论try块内的语句是否成功执行，finally块内的语句都会执行。

使用格式如下：

try {

statements;

} catch (ExceptionClass e) {

statements;

} finally {

statements;

}

如果catch块中除throw 异常外没有任何处理，则不应该写该try/catch块。

5、空白使用规范

可使用编辑工具（如Eclipse）的代码格式化功能实现，注意设置。使用无代码格式化功能的编辑器的，必须严格按照此规范编写。

空行使用规范

空行的正确使用能够提高源代码的可读性，使得程序的逻辑关系更清楚。

两行空行的使用

在下列情况下使用两行空行：

A.在一个文件的两部分（section）代码之间；

B.在两个类或者接口的定义之间。

单行空行的使用

在下列情况下使用单行空行：

A.在两个方法定义之间；

B.在一个方法中局部变量的变量定义语句和方法的第一个其他语句之间；

C.在一个语句块或者单行注释之前；

D.为了提高程序可读性，在一个方法的两个逻辑段之间。

空格使用规范

在下列情况下使用空格：

A.关键字后跟随圆括号时要在两者间添加空格；

B.函数调用时，函数名和圆括号间要添加括号；

注意：在定义方法时的方法名和圆括号之间不加空格，这样便于区分方法调用和方法定义。

C.参数列表中的逗号后面要添加空格；

D.除了成员运算符（.）以外，所有的二元运算符应该用空格和操作数分开；在一元运算符（如：++、--、负数-）和操作数之间不用空格；

E.for语句中的表达式要用空格隔开；

F.强制类型转换后面要用空格隔开；

6、缩进规范

可使用编辑工具（如Eclipse）的代码格式化功能实现，注意设置。使用无代码格式化功能的编辑器的，必须严格按照此规范编写。

每一级缩进都要再上一级的基础上缩进4个字符。

不允许使用TAB键来进行缩进。

代码长度

检查点 最大字符数或行数

可执行语句的最大行数 120

Java源文件最大行数 2000

方法、构造方法最多字符数 120

匿名内部类的最大行数 120

参数的最大个数 7

7、JAVA源文件

每个Java 源文件都包含一个单一的公共类或接口。若私有类和接口与一个公共类相关联，

可以将它们和公共类放入同一个源文件。公共类必须是这个文件中的第一个类或接口。

Java 源文件还遵循以下规则：

- 开头注释

- 包和引入语句

- 类和接口声明

所有的源文件都应该在开头有一个C 语言风格的注释，其中列出类名、版本信息、日期和

版权声明。

包和引入语句

在多数Java 源文件中，第一个非注释行是包语句。在它之后可以跟引入语句。例如：

package java.awt;

import java.awt.peer.CanvasPeer;

类和接口说明

下表描述了类和接口声明的各个部分以及它们出现的先后次序。参见"Java 源文件范例"中一

个包含注释的例子。

类/接口声明的各部分注解

A.类/接口文档注释(/\*\*……\*/) 该注释中所需包含的信息，参见"文档注释"

B.类和接口的声明

C.类/接口实现的注释(/\*……\*/)如果有必要的话该注释应包含任何有关整个类或接口的信

息，而这些信息又不适合作为类/接口文档注释。

D.类的(静态)变量首先是类的公共变量，随后是保护变量，再后是包一级别的变量(没有访

问修饰符，access modifier)，最后是私有变量。

E.实例变量首先是公共级别的，随后是保护级别的，再后是包一级别的(没有访问修饰符)，

最后是私有级别的。

F.构造器

G.方法这些方法应该按功能，而非作用域或访问权限，分组。例如，一个私有的类方法可

以置于两个公有的实例方法之间。其目的是为了更便于阅读和理解代码。

8、缩进排版

4 个空格常被作为缩进排版的一个单位。缩进的确切解释并未详细指定(空格vs. 制表符)。

一个制表符等于8 个空格(而非4 个)。

行长度

尽量避免一行的长度超过80 个字符，因为很多终端和工具不能很好处理之。

注意：用于文档中的例子应该使用更短的行长，长度一般不超过70 个字符。

换行

当一个表达式无法容纳在一行内时，可以依据如下一般规则断开之：

- 在一个逗号后面断开

- 在一个操作符前面断开

- 宁可选择较高级别(higher-level)的断开，而非较低级别(lower-level)的断开

- 新的一行应该与上一行同一级别表达式的开头处对齐

- 如果以上规则导致你的代码混乱或者使你的代码都堆挤在右边，那就代之以缩进8 个空

格。

9、注释

Java 程序有两类注释：实现注释(implementation comments)和文档注释(document comments)。

实现注释是那些在C++ 中见过的，使用/\*...\*/ 和//界定的注释。文档注释(被称为"doc

comments")是Java 独有的，并由/\*\*...\*/界定。文档注释可以通过javadoc 工具转换成HTML

文件。

实现注释用以注释代码或者实现细节。文档注释从实现自由(implementation-free)的角度描述

代码的规范。它可以被那些手头没有源码的开发人员读懂。

注释应被用来给出代码的总括，并提供代码自身没有提供的附加信息。注释应该仅包含与阅读和理解程序有关的信息。例如，相应的包如何被建立或位于哪个目录下之类的信息不应包

括在注释中。

在注释里，对设计决策中重要的或者不是显而易见的地方进行说明是可以的，但应避免提供

代码中己清晰表达出来的重复信息。多余的的注释很容易过时。通常应避免那些代码更新就

可能过时的注释。

尾端注释

极短的注释可以与它们所要描述的代码位于同一行，但是应该有足够的空白来分开代码和注

释。若有多个短注释出现于大段代码中，它们应该具有相同的缩进。

9、每行声明变量的数量

推荐一行一个声明，因为这样以利于写注释。亦即，

int level; // indentation level

int size; // size of table

要优于，

int level, size;

不要将不同类型变量的声明放在同一行，例如：

int foo, fooarray[]; //WRONG!

注意：上面的例子中，在类型和标识符之间放了一个空格，另一种被允许的替代方式是使用

制表符：

布局

只在代码块的开始处声明变量。（一个块是指任何被包含在大括号"{"和"}"中间的代码。）不

要在首次用到该变量时才声明之。这会把注意力不集中的程序员搞糊涂，同时会妨碍代码在

该作用域内的可移植性。

类和接口的声明

当编写类和接口是，应该遵守以下格式规则：

- 在方法名与其参数列表之前的左括号"("间不要有空格

- 左大括号"{"位于声明语句同行的末尾

- 右大括号"}"另起一行，与相应的声明语句对齐，除非是一个空语句，"}"应紧跟在"{"之后。

二、配置信息

操作系统：Windows 8及以上系统，编写软件：Eclipse java EE for Web Developers

版本信息：4.7.2

apache-maven-3.5.3 apache-tomcat-8.5.29 Git-2.17.0-64

jdk