**一、术语**

1、术语class可表示一个普通类，枚举类，接口或是annotation类型(@interface)。

2、术语comment只用来指代实现的注释(implementation comments)，我们不使用“documentation comments”一词，而是用Javadoc。

**二：源文件**

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、特殊转义序列

对于具有特殊转义序列的任何字符(\b, \t, \n, \f, \r, \“, \‘及\)，我们使用它的转义序列，而不是相应的八进制(比如\012)或Unicode(比如\u000a)转义。

**三：源文件结构**

 一个源文件包含(按顺序地)：

1.许可证或版权信息(如有需要)：若有，放在文件最前面。

2.package语句

3.import语句：①不要使用通配符；②不要换行；③顺序和间距：import语句可分为以下几组，按照这个顺序，每组由一个空行分隔：

A、所有的静态导入独立成组

B、com.google imports(仅当这个源文件是在com.google包下)

C、第三方的包。每个顶级包为一组，字典序。例如：android, com, junit, org, sun

D、java imports

E、javax imports

组内不空行，按字典序排列。

**四：类声明**

1、只有一个顶级类声明

2、类成员顺序: 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。

3、重载：永不分离。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。

**五：大括号**

1、大括号与if, else, for, do, while语句一起使用，即使只有一条语句(或是空)，也应该把大括号写上。

2、一个空的块状结构里什么也不包含，大括号可以简洁地写成{}，不需要换行。

3、每当开始一个新的块，缩进增加2个空格，当块结束时，缩进返回先前的缩进级别。缩进级别适用于代码和注释。

4、一行一个语句。

5、某些需自动换行：一般情况下，一行长代码为了避免超出列限制(80或100个字符)而被分为多行，我们称之为自动换行(line-wrapping)。

6、用小括号来限定组。

**六：具体结构**

1、枚举类：枚举常量间用逗号隔开，换行可选。

2、变量声明:①每次只声明一个变量；②需要时才声明，并尽快进行初始化；

3、数组：①数组初始化可写成块状结构；②中括号是类型的一部分：String[] args，而非String args[]；

4、Switch：

①switch块的大括号内是一个或多个语句组。每个语句组包含一个或多个switch标签(case FOO:或default:)，后面跟着一条或多条语句。

②switch块中的内容缩进为2个空格。

③在一个switch块内，每个语句组要么通过break, continue, return或抛出异常来终止，要么通过一条注释来说明程序将继续执行到下一个语句组，任何能表达这个意思的注释都是OK的(典型的是用// fall through)。这个特殊的注释并不需要在最后一个语句组(一般是default)中出现。

④每个switch语句都包含一个default语句组，即使它什么代码也不包含。

5、注解：注解紧跟在文档块后面，应用于类、方法和构造函数，一个注解独占一行。单个的注解可以和签名的第一行出现在同一行。应用于字段的注解紧随文档块出现，应用于字段的多个注解允许与字段出现在同一行。

6、注释：

块注释与其周围的代码在同一缩进级别。它们可以是/\* ... \*/风格，也可以是// ...风格。对于多行的/\* ... \*/注释，后续行必须从\*开始，并且与前一行的\*对齐。在写多行注释时，如果你希望在必要时能重新换行(即注释像段落风格一样)，那么使用/\* ... \*/。

**七：命名约定**

1、标识符只能使用ASCII字母和数字，因此每个有效的标识符名称都能匹配正则表达式\w+。

2、标识符类型的规则：

①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。

②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。类名都以UpperCamelCase风格编写。

③方法名通常是动词或动词短语。方法名都以lowerCamelCase风格编写。④每个常量都是一个静态final字段，但不是所有静态final字段都是常量。常量名命名模式为CONSTANT\_CASE，全部字母大写，用下划线分隔单词。

⑤非常量字段名以lowerCamelCase风格编写。

⑥参数名以lowerCamelCase风格编写。 ⑦局部变量名以lowerCamelCase风格编写。⑧类型变量可用以下两种风格之一进行命名：单个的大写字母，后面可以跟一个数字(如：E, T, X, T2)。•以类命名方式(5.2.2节)，后面加个大写的T(如：RequestT, FooBarT)。

3、驼峰式命名法(CamelCase)：驼峰式命名法分大驼峰式命名法(UpperCamelCase)和小驼峰式命名法(lowerCamelCase)。 有时，我们有不只一种合理的方式将一个英语词组转换成驼峰形式，如缩略语或不寻常的结构(例如"IPv6"或"iOS")。

**八：编程实践**

1、只要是合法的，就把@Override注解给用上。

2、捕获的异常。

3、使用类名调用静态的类成员，而不是具体某个对象或表达式。

4、Finalizers: 禁用。

**九：Javadoc**

1、一般形式：Javadoc块的基本格式如下所示：

/\*\*

 \* Multiple lines of Javadoc textare written here,

 \* wrapped normally...

 \*/

public int method(String p1) { ... }

或者是以下单行形式：

/\*\* An especially short bit of Javadoc. \*/

2、空行(即，只包含最左侧星号的行)会出现在段落之间和Javadoc标记(@XXX)之前(如果有的话)。 除了第一个段落，每个段落第一个单词前都有标签<p>，并且它和第一个单词间没有空格。

3、标准的Javadoc标记按以下顺序出现：@param,@return, @throws, @deprecated, 前面这4种标记如果出现，描述都不能为空。当描述无法在一行中容纳，连续行需要至少再缩进4个空格。

4、至少在每个public类及它的每个public和protected成员处使用Javadoc。

对此的读取，更加清楚的认识到代码的注释与编码的规范，认识到自己水平的有限，空间发展之广，java学习要慎重、细心。