排版规范

1. 规则

1.1. \*程序块要采用缩进风格编写，缩进的空格数为 4个。

说明：对于由开发工具自动生成的代码可以有不一致。

1.2. \*分界符（如大括号‘ { ’和‘ } ’）应各独占一行并且位于同一列，同时与引用它们的语

句左对齐。在函数体的开始、类和接口的定义、以及 if、for、do、while、switch、case语

句中的程序都要采用如上的缩进方式。

1.3. \*较长的语句、表达式或参数（ >80字符）要分成多行书写，长表达式要在低优先级操作

符处划分新行，操作符放在新行之首，划分出的新行要进行适当的缩进，使排版整齐，

语句可读。

1.4. \*不允许把多个短语句写在一行中，即一行只写一条语句

1.5. \*if, for, do, while, case, switch, default 等语句自占一行，且 if, for, do, while 等语句的执行语句无论多少都要加括号 {} 。

1.6. \*相对独立的程序块之间、变量说明之后必须加空行。

1.7. \*对齐只使用空格键，不使用 TAB 键。

说明：以免用不同的编辑器阅读程序时，因 TAB 键所设置的空格数目不同而造成程序布局不

整齐。 JBuilder 、UltraEdit 等编辑环境，支持行首 TAB 替换成空格，应将该选项打开

1.8. \*在两个以上的关键字、变量、常量进行对等操作时，它们之间的操作符之前、之后或

者前后要加空格；进行非对等操作时，如果是关系密切的立即操作符（如 .），后不应加

空格

2. 建议

类属性和类方法不要交叉放置，不同存取范围的属性或者方法也尽量不要交叉放置。

注释规范

1. 规则

1.1. 一般情况下，源程序有效注释量必须在 30％以上。

说明：注释的原则是有助于对程序的阅读理解，在该加的地方都加了，注释不宜太多也不能

太少，注释语言必须准确、易懂、简洁。可以用注释统计工具来统计。

.2. 包的注释：包的注释写入一名为 package.html 的HTML 格式说明文件放入当前路径。

说明：方便 JavaDoc收集

1.3. 包的注释内容：简述本包的作用、详细描述本包的内容、产品模块名称和版本、公司版

权。

说明：在详细描述中应该说明这个包的作用以及在整个项目中的位置。

1.4. 文件注释：文件注释写入文件头部，包名之前的位置。

说明：注意以 /\* 开始避免被 JavaDoc 收集

1.5. 文件注释内容：版权说明、描述信息、生成日期、修改历史。

1.6. 类和接口的注释：该注释放在 package 关键字之后， class 或者 interface 关键字之前。

说明：方便 JavaDoc收集。

1.7. 类和接口的注释内容：类的注释主要是一句话功能简述、功能详细描述。

说明：可根据需要列出：版本号、生成日期、作者、内容、功能、与其它类的关系等。 如果

一个类存在 Bug，请如实说明这些 Bug。

1.8. 类属性、公有和保护方法注释：写在类属性、公有和保护方法上面。

1.9. 成员变量注释内容：成员变量的意义、目的、功能，可能被用到的地方。

1.10. 公有和保护方法注释内容：列出方法的一句话功能简述、功能详细描述、输入参数、输

出参数、返回值、违例等。

1.11. 对于方法内部用 throw语句抛出的异常，必须在方法的注释中标明，对于所调用的其他方

法所抛出的异常，选择主要的在注释中说明。 对于非 RuntimeException，即throws子句声

明会抛出的异常，必须在方法的注释中标明。

说明：异常注释用 @exception或@throws表示，在 JavaDoc中两者等价，但推荐用 @exception

标注 Runtime 异常， @throws标注非 Runtime 异常。异常的注释必须说明该异常的含义及什么条件

下抛出该异常。

1.12. \*注释应与其描述的代码相近，对代码的注释应放在其上方或右方（对单条语句的注

释）相邻位置，不可放在下面，如放于上方则需与其上面的代码用空行隔开

1.13. \*注释与所描述内容进行同样的缩排。

说明：可使程序排版整齐，并方便注释的阅读与理解。

1.14. \*将注释与其上面的代码用空行隔开

1.15. \*对变量的定义和分支语句（条件分支、循环语句等）必须编写注释。

说明：这些语句往往是程序实现某一特定功能的关键，对于维护人员来说，良好的注释帮助

更好的理解程序，有时甚至优于看设计文档。

1.16. \*对于 switch语句下的 case语句，如果因为特殊情况需要处理完一个 case后进入下一个 case

处理，必须在该 case语句处理完、下一个 case语句前加上明确的注释。

说明：这样比较清楚程序编写者的意图，有效防止无故遗漏 break语句。

1.17. \*边写代码边注释，修改代码同时修改相应的注释，以保证注释与代码的一致性。不再

有用的注释要删除。

1.18. \*注释的内容要清楚、明了，含义准确，防止注释二义性。

说明：错误的注释不但无益反而有害。

1.19. \*避免在注释中使用缩写，特别是不常用缩写。

说明：在使用缩写时或之前，应对缩写进行必要的说明。

2. 建议

2.1. \*避免在一行代码或表达式的中间插入注释。

说明：除非必要，不应在代码或表达中间插入注释，否则容易使代码可理解性变差。

2.2. \*通过对函数或过程、变量、结构等正确的命名以及合理地组织代码的结构，使代码成

为自注释的。

说明：清晰准确的函数、变量等的命名，可增加代码可读性，并减少不必要的注释。

2.3. \*在代码的功能、意图层次上进行注释，提供有用、额外的信息。

说明：注释的目的是解释代码的目的、功能和采用的方法，提供代码以外的信息，帮助读者

理解代码，防止没必要的重复注释信息。

示例：如下注释意义不大。

// 如果 receiveFlag 为真

if (receiveFlag)

而如下的注释则给出了额外有用的信息。

// 如果从连结收到消息 if (receiveFlag)

2.4. \*在程序块的结束行右方加注释标记，以表明某程序块的结束。

说明：当代码段较长，特别是多重嵌套时，这样做可以使代码更清晰，更便于阅读。

示例： 参见如下例子。

2.5. \*注释应考虑程序易读及外观排版的因素，使用的语言若是中、英兼有的，建议多使用

中文，除非能用非常流利准确的英文表达。

说明：注释语言不统一，影响程序易读性和外观排版，出于维护的考虑，建议使用中文。

2.6. 方法内的单行注释使用 //。

说明：调试程序的时候可以方便的使用 /\* 。。。 \*/ 注释掉一长段程序。

2.7. 注释尽量使用中文注释和中文标点。方法和类描述的第一句话尽量使用简洁明了的话概

括一下功能，然后加以句号。接下来的部分可以详细描述。

说明： JavaDoc工具收集简介的时候使用选取第一句话。

2.8. 顺序实现流程的说明使用 1、2、3、4在每个实现步骤部分的代码前面进行注释。

示例：如下是对设置属性的流程注释

//1 、 判断输入参数是否有效。 。。。。。 // 2 、设置本地变量。 。。。。。。

2.9. 一些复杂的代码需要说明。

示例：这里主要是对闰年算法的说明。