|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 类别 | 内容 | | 说明 | 状态 | 正确示例 | 错误示例 |
| 格  式  规  范 | 缩进 | 缩进要求采用4个空格，禁止使用制表符（TAB） | 提高可读性 | 建议 | Public add1（）{  String sql=””;  } | Public add1（）{  String sql=””;  } |
| 空行 | 方法之间需要添加空行 | 提高可读性 | 建议 | public void add1() {  }  (空行)  public void add2() {  } | public void add1() {  }  Public void add2() {  } |
| 命名规范 | 变量命名规范 | 定义变量时，变量名最好与变量实际意思有关 | 提高可读性 | 规则 | String username= “abc”； | String aaa= “abc”； |
| 类命名 | 类的命名开头字母最好要大些 | Java的一般规则 | 规则 | public class Jibenxx{  } | Public class jibenxx{  } |
| 方法命名 | 方法的命名应采用含义对应的动词或词组； | 将提高代码的易读性。 | 建议 | Public void add（）{  } | public void aaa{  } |
| 常量命名 | 每个字母均采用大写格式，单词片断之间采用下划线“\_”连接，相同类别的常量应使用同一前缀。 | Java的一般规则。 | 建议 | static final int CPUTYPE\_INTEL\_CPU = 1;  static final int CPUTYPE\_AMD\_CPU = 2; |  |
|  | 注释 | 类注释通常放在类定义的前面（类实体以外）。 | 提高代码可读性， | 规则 | /\*\*  \* 示范方法的声明及方法体编写规范  \* @return float 缴费基数  \* @param year int 年度  \* @param employeeID java.lang.String 职工编号  \* @exception java.io.IOException 输入、输出例外  \*/ |  |
| 编写规范 | 表达式 | 在条件表达式中，对每个逻辑判断都使用()，即使没有逻辑上的需求。 | 增加代码的可读性和可维护性。 | 规范 | if((x == 0) || (y == 0)) { } | if(x == 0 || y == 0) { } |
| 流程控制 | 单个的循环、条件语句也要求加{}。 | 便于代码的修改、增删，有利于以后的代码维护工作。 | 建议 | if (a>b) {  a = b; } | if (a>b)   a = b; |
| 数据库 | 数据库连接、SDE连接必须关闭，包括在异常状态下。 | 连接不关闭，会造成打开的连接过多。无法获取新的连接。 | 规则 | Connection con = null;  try {  con = getConnection();  Statement stmt = con.createStatement();  // ....  } catch (SQLException sqle) {  throw sqle;  }  } finally {  if (stmt!=null) {  stmt.close();  }  if (con != null && !con.isClosed()) {  con.close(); // 正确  }  } | Connection con = null;  try {  con = getConnection();  Statement stmt = con.createStatement();  // ....  } finally {    } |