

Weaver Software

中国上海耀华支路39弄9号泛微软件大厦

邮政编码：200122

电话：+86 21 50942228

传真：+86 21 50942278

e-cology开发代码技术规范

**仅限阅读 请勿传播**

当您阅读本方案时，即表示您同意不传播本方案的所有内容

**版权说明**

本手册以及相关的计算机软件程序仅用于为泛微协同管理平台最终用户提供信息，上海泛微网络科技股份有限公司有权。本手册是上海泛微网络科技股份有限公司的专有信息，并受中华人民共和国版权法和国际公约的保护。

未得到上海泛微网络科技股份有限公司明确的书面许可，不得为任何目的、以任何形式或手段辅助或传播本手册的任何部分内容。

本手册内容若有变动，恕不另行通知。本手册中例子中所引用的公司、人名和数据若非特殊说明，均属虚构。

本手册以及本手册所提及的任何产品的使用均受适应的最终用户许可协议限制。本手册由上海泛微网络科技股份有限公司制作。本手册中提及的所有商标、商标名称、服务标志及徽标均归其各自公司所有。

目录

[前 言 5](#_Toc441499575)

[第1章 页面样式规范 5](#_Toc441499576)

[一、 标准规范 5](#_Toc441499577)

[二、 E-cology8.0新增规范 7](#_Toc441499578)

[2.1、关于topTitle上的使用： 7](#_Toc441499579)

[2.2、高级搜索中的按钮样式应用： 8](#_Toc441499580)

[2.3、关于布局和表单基本信息所用的样式问题： 9](#_Toc441499581)

[2.4、关于弹出框中的按钮的样式应用： 9](#_Toc441499582)

[2.5、关于弹出框时，弹出层页面需要做的一些特殊处理： 10](#_Toc441499583)

[2.6、关于二选一checkbox框的美化使用 10](#_Toc441499584)

[2.7、关于页面中间的添加和删除按钮的样式： 11](#_Toc441499585)

[第2章 代码规范 12](#_Toc441499586)

[一、 缩进 12](#_Toc441499587)

[二、 长度 12](#_Toc441499588)

[三、 间隔 12](#_Toc441499589)

[四、 对齐 13](#_Toc441499590)

[五、 注释 14](#_Toc441499591)

[5.1、基本原则 14](#_Toc441499592)

[5.2、JavaDoc 14](#_Toc441499593)

[5.3、文件注释 14](#_Toc441499594)

[5.4、类、接口注释 15](#_Toc441499595)

[5.5、方法注释 15](#_Toc441499596)

[5.6、注释参考表 16](#_Toc441499597)

[5.7、其他注释 18](#_Toc441499598)

[六、 命名 18](#_Toc441499599)

[6.1、基本原则 18](#_Toc441499600)

[6.2、文件、包 19](#_Toc441499601)

[6.3、类、接口 19](#_Toc441499602)

[七、 字段 20](#_Toc441499603)

[7.1、常量 20](#_Toc441499604)

[7.2、变量和参数 20](#_Toc441499605)

[7.3、集合 20](#_Toc441499606)

[7.4、其他 20](#_Toc441499607)

[八、 方法 20](#_Toc441499608)

[8.1、异常 21](#_Toc441499609)

[8.2、命名约定表 21](#_Toc441499610)

[九、 声明 23](#_Toc441499611)

[9.1、基本原则 23](#_Toc441499612)

[9.2、包 23](#_Toc441499613)

[9.3、方法 24](#_Toc441499614)

[十、 类与接口 24](#_Toc441499615)

[10.1、基本原则 24](#_Toc441499616)

[10.2继承与组合 24](#_Toc441499617)

[十一、 方法 25](#_Toc441499618)

[11.1、基本原则 25](#_Toc441499619)

[11.2、参数和返回值 26](#_Toc441499620)

[十二、 表达式与语句 26](#_Toc441499621)

[12.1、基本原则 26](#_Toc441499622)

[12.2、循环语句 26](#_Toc441499623)

[十三、 错误与异常 27](#_Toc441499624)

[13.1、基本原则 27](#_Toc441499625)

[13.2、异常的捕捉与处理 28](#_Toc441499626)

# 前 言

本规范的目的是使泛微技术开发部门以标准的、规范的方式设计和编码。通过建立编码规范，以使每个开发人员养成良好的编码风格和习惯；并以此形成开发小组编码约定，提高程序的可靠性、可读性、可修改性、可维护性和一致性等，增进团队间的交流，并保证ECOLOGY产品的质量。本规范适用于ECOLOGY产品及针对客户的项目的设计、开发以及维护、升级等。

1. 页面样式规范

## 标准规范

1、对于所有的宽度或者高度等设置（包括padding，margin等），除0之外，所有的度量单位都必须加单位（如29px）结尾；如：

|  |
| --- |
| 正确：padding:2px 0 2px 20px;  错误：padding:2 0 2 20; |

2、对于background的写法，不能增加rgba(0,0,0)属性，该属性是css3标准，ie8暂不支持

|  |
| --- |
| 正确：background:url(listbox/open2.png) no-repeat scroll 0 50%;  错误：background:url(listbox/open2.png) no-repeat scroll 0 50% rgba(0,0,0); |

3、对于class名字，大小写敏感。即使用的样式名必须和定义的样式名保持一致。

定义：button.Calendar{background-image:url(general/calendar.png)}

引用：

|  |
| --- |
| 正确：class="Calendar"  错误：class="calendar" |

4、对于非块级元素，使用background时必须指定图片的宽度和高度，或者指定元素的宽度和高度；

5、对于td元素，使用跨列时不能指定display为block或者inline-block，不然会导致跨列失效；

6、所有的document.all修改为$GetEle方法调用，有id的元素可以使用jQuery获取（建议使用jQuery来操作dom对象）；

如document.all("test")，修改为$GetEle("test")

7、对于设定元素的高度与宽度时，建议全部改用jQuery的方法进行设定，如果使用原生的，那么必须加px或者使用%值。

|  |
| --- |
| 错误：document.getElementById("divContent").style.height = 800;  正确：document.getElementById("divContent").style.height = "800px";  正确：jQuery("#divContent").height(800); |

8、对于ckeditor在IE8下报“存储空间不足”错误的解决方案：

* 修改ckeditor.js，将样式解析屏蔽；（统一处理）
* 需要在使用ckditor的界面增加如下代码：

|  |
| --- |
| <!--[if lte IE 8]>  <link href="/wui/common/js/ckeditor/skins/kama/editor.css" type=text/css rel=STYLESHEET>  <![endif]--> |

9、ie的专有属性如onpropertychange不能应用，请使用\_listener属性代替；

10、css中expression表达式不能使用，需使用js替换；

11、对于设定元素的高度，请注意在ie盒子模型（quicks模式）中，设定的高度（宽度）包括padding的高度（宽度），但在标准盒子模型中，设定的高度（宽度）不包括padding的高度（宽度）；如：

|  |
| --- |
| .a{  height:100px;  padding-top:10px;  }  <div class="a"></div>  quicks模式下div的高度就是100px；  标准模式下div的高度就是110px；  如果本意设置的高度要包括padding的高度为100px，那么请把height设置为90px； |

12、对于需要以100%显示的元素，请增加属性height:100%，以确保所有的浏览器都能正常显示。

## E-cology8.0新增规范

### 2.1、关于topTitle上的使用：

|  |
| --- |
| **红色部分的样式不要修改，按照这个格式来**  1、按照如下格式定义好toptitle的内容：  <table id="topTitle" cellpadding="0" cellspacing="0">  <tr>  <td>  </td>  <td class="rightSearchSpan" style="text-align:right;">  <%**if**(canEdit){ %>  <input type=button class="e8\_btn\_top" onclick="onSave();" value="<%=SystemEnv.getHtmlLabelName(86, user.getLanguage())%>"></input>  <%} %>  <%**if**(HrmUserVarify.checkUserRight("DocSubCategoryAdd:add", user)){ %>  <input type=button class="e8\_btn\_top" onclick="openDialog3(<%=id %>);" value="<%=SystemEnv.getHtmlLabelName(82,user.getLanguage())+SystemEnv.getHtmlLabelName(66,user.getLanguage())%>"></input>  <%} %>  <input type="text" class="searchInput" name="flowTitle" value="<%=qname %>"/>  <span id="advancedSearch" class="advancedSearch">高级搜索</span>  <span title="菜单" class="cornerMenu"></span>  </td>  </tr>  </table>  <div class="advancedSearchDiv" id="advancedSearchDiv" style="display:none;" >  ...  </div>  2、页面加载完成时调用如下方法  jQuery("#topTitle").topMenuTitle({searchFn:onBtnSearchClick});  jQuery("#hoverBtnSpan").hoverBtn();  其中：  searchFn参数是提供全文搜索用的方法，传入自己定义的方法名称即可；  高级搜索根据情况需要就添加，不需要就不要添加；  如果有高级搜索条件，那么所有的条件都请放置在  <div class="advancedSearchDiv" id="advancedSearchDiv" style="display:none;" >  ...  </div>  中 |

2.1.1、高级搜索中底部按钮列的td请按照以下格式书写

|  |
| --- |
| <wea:group context="" attributes="{'groupDisplay':'none'}">  <wea:item type="toolbar">  <input type="button" onclick="doRefresh(this)" value="<%=SystemEnv.getHtmlLabelName(197,user.getLanguage())%>" class="e8\_btn\_submit"/>  <input type="button" value="<%=SystemEnv.getHtmlLabelName(2022,user.getLanguage())%>" class="e8\_btn\_cancel" onclick="resetCondtion();"/>  <input type="button" value="<%=SystemEnv.getHtmlLabelName(31129,user.getLanguage())%>" class="e8\_btn\_cancel" id="cancel"/>  </wea:item>  </wea:group> |

### 2.2、高级搜索中的按钮样式应用：

取消或者重置按钮，请使用样式e8\_btn\_cancel，其他按钮请使用样式e8\_btn\_submit

如：

|  |
| --- |
| <input type="button" onclick="jQuery('#isentrydetail').val('0');checkSubmit();" class="zd\_btn\_submit" id="zd\_btn\_submit" value="保存">  <input type="button" onclick="parentWin.closeDialog();" class="zd\_btn\_cancle" id="zd\_btn\_cancle" value="取消"> |

### 2.3、关于布局和表单基本信息所用的样式问题：

* 表单基本信息页的样式请使用ViewForm，如：

|  |
| --- |
| 这种页面请使用<table class="ViewForm"></table>  另外，需要白色背景的td，请添加Field样式，如<td class="Field"></td> |

* 非表单基本信息布局用的样式请使用LayoutTable，如：

|  |
| --- |
| 上述这种仅仅是布局用的table，请使用<table class="LayoutTable"></table> |

### 2.4、关于弹出框中的按钮的样式应用：

取消或者关闭按钮，请使用样式zd\_btn\_cancel，其他按钮请使用样式zd\_btn\_submit

如：

|  |
| --- |
| <input class="e8\_btn\_submit" value="搜索" type="submit">  <input value="重置" class="e8\_btn\_cancel" onclick="resetCondtion();" type="button">  <input value="取消" class="e8\_btn\_cancel" id="cancel" type="button"> |

### 2.5、关于弹出框时，弹出层页面需要做的一些特殊处理：

为了固定底牌按钮的位置，需要添加一些特殊标志来进行计算

具体格式如下(**红色部分的格式固定不变**)：

|  |
| --- |
| <body>  <%**if**("1".equals(isDialog)){ %>  <div class="zDialog\_div\_content">  <%} %>  //具体内容部分  <%**if**("1".equals(isDialog)){ %>  </div>  <div id="zDialog\_div\_bottom" class="zDialog\_div\_bottom">  <wea:layout needImportDefaultJsAndCss="false">  <wea:group context="" attributes="{\"groupDisplay\":\"none\"}">  <wea:item type="toolbar">  <input type="button" accessKey=2 id=btnclose value="<%=SystemEnv.getHtmlLabelName(309,user.getLanguage())%>" id="zd\_btn\_cancel" class="zd\_btn\_cancle" onclick="dialog.close()">  </wea:item>  </wea:group>  </wea:layout></div>  <script type="text/javascript">  resizeDialog(document);  </script>  <%} %>  </BODY> |

### 2.6、关于二选一checkbox框的美化使用

|  |
| --- |
| 效果图如下：    具体使用如下：  引入如下css和js：  <link href="/js/checkbox/jquery.tzCheckbox.css" type=text/css rel=STYLESHEET>  <script language=javascript src="/js/checkbox/jquery.tzCheckbox.js"></script>  调用以下代码：  jQuery(document).ready(function(){  jQuery("input[type=checkbox]").each(function(){  if(jQuery(this).attr("tzCheckbox")=="true"){  jQuery(this).tzCheckbox({labels:['','']});  }  });  });  //checkbox框需要增加红色部分的属性  <INPUT type="checkbox" tzCheckbox="true" class=InputStyle name="norepeatedname" value="1" |

### 2.7、关于页面中间的添加和删除按钮的样式：

|  |
| --- |
| 添加按钮样式：class="addbtn"，删除按钮样式：class="delbtn"  <input class="addbtn" accesskey="A" onclick="addNewRow();" title="添加" type="button">  <input class="delbtn" accesskey="E" onclick="deleteSelectedRow();" title="删除" type="button"> |

1. 代码规范

## 缩进

子功能块当在其父功能块后缩进。当功能块过多而导致缩进过深时当将子功能块提取出来做为子方法。 代码中以TAB（4个字符）缩进，在编辑器中请将TAB设置为以空格替代，否则在不同编辑器或设置下会导致TAB长度不等而影响整个程序代码的格式。

## 长度

为便于阅读和理解，单个方法的有效代码长度当尽量控制在150行以内（不包括注释行），当一个功能模块过大时往往造成阅读困难，因此当使用子方法等将相应功能抽取出来，这也有利于提高代码的重用度。

单个类也不宜过大，当出现此类情况时当将相应功能的代码重构到其他类中，通过组合等方式来调用，要求单个类的长度包括注释行不超过2000行。 尽量避免使用大类和长方法。

## 间隔

类、方法及功能块间等应以空行相隔，以增加可读性，但不得有无规则的大片空行。 操作符两端应当各空一个字符以增加可读性。 相应独立的功能模块之间可使用注释行间隔，并标明相应内容。

## 对齐

关系密切的行应对齐，对齐包括类型、修饰、名称、参数等各部分对齐。

连续赋值时当对齐操作符。

当方法参数过多时当在每个参数后（逗号后）换行并对齐。

当控制或循环中的条件比较长时当换行（操作符前）、对齐并注释各条件。

变量定义最好通过添加空格形成对齐，同一类型的变量应放在一起。如下例所示：

对齐示例

|  |
| --- |
| //变量对齐-----------------------------------------------  int count = 100;  int length = 0;  String strUserName = null;  Integer[] flowCode = new Integer(2); //流程编码数组 |

|  |
| --- |
| //参数对齐----------------------------------------------  public Connection getConnection(String url,  String userName,  String password)  throws SQLException,IOException{  } |

|  |
| --- |
| //换行对齐----------------------------------------------  public final static String SQL\_SELECT\_WORKFLOW = “SELECT \* “  + “ FROM WORKFLOW\_REQUESTBASE WHERE ID = ”  + workID; |

|  |
| --- |
| //条件对齐----------------------------------------------  if( Condition1 //当条件一  && Condition2 //并且条件二  || Condition3){ //或者条件三 |

## 注释

### 5.1、基本原则

* 注释应该增加代码的清晰度。代码注释的目的是要使代码更易于被其他开发人员等理解。
* 避免使用装饰性内容。
* 保持注释的简洁。
* 注释信息不仅要包括代码的功能，还应给出原因。
* 除变量定义等较短语句的注释可用行尾注释外，其他注释当避免使用行尾注释。

### 5.2、JavaDoc

对类、方法、变量等的注释需要符合JavaDoc规范，对类、方法都应详细说明其功能、条件、参数等，并使用良好的HTML标记格式化注释，以使生成的JavaDoc易阅读和理解。 类注释中当包含版本和作者信息 。

### 5.3、文件注释

在每个文件头部必须包含版权信息等。

示例：

|  |
| --- |
| /\*  \* Created on 2014-05-18  \* Copyright (c) 2001-2014 泛微软件  \* 泛微协同商务系统，版权所有。  \*  \*/ |

### 5.4、类、接口注释

在类、接口定义之前当对其进行注释，包括类、接口的目的、作用、功能、继承于何种父类，实现的接口、实现的算法、使用方法、示例程序等。

示例：

|  |
| --- |
| /\*\*  \* Title: 数据库链接  \* Company: 泛微软件  \* @author: weaver  \* @version: 1.0  \* create date : 2012-10-23  \* modify log:  \*  \*  \* Description: 数据库链接实现Connection，增加了关于使用次数，空闲时间，被调用时间等属性，其它部分和JDBC提供者  \* 提供的Connection完全相同，请参照java.sql 来解释其方法。  \* 所有创建的链接存放在链接池中，在客户端链接数据库时从链接池中取出链接，客户端在完成数据库操作后将链接返回链接池。  \*  \*  \*/ |

### 5.5、方法注释

依据标准JavaDoc规范对方法进行注释，以明确该方法功能、作用、各参数含义以及返回值等。复杂的算法用/\*\*/在方法内注解出。 参数注释时当注明其取值范围等 ,返回值当注释出失败、错误、异常时的返回情况。 异常当注释出什么情况、什么时候、什么条件下会引发什么样的异常

方法注释示例

|  |
| --- |
| /\*\*  \* 执行查询。  \* 该方法调用Statement的executeQuery(sql)方法并返回ResultSet 结果集。  \* @param sql 标准的SQL语句  \* @return ResultSet结果集，若查询失败则返回null  \* @throws SQLException 当查询数据库时可能引发此异常  \*/  public ResultSet executeQuery(String sql) throws SQLException  {  //Statement和SQL语句都不能为空  if(null != stmt && !StringUtil.isEmpty(sql)){  //返回查询执行结果  return stmt.executeQuery(sql);  }  return null;  }//end executeQuery() |

### 5.6、注释参考表

注释参考表

|  |  |
| --- | --- |
| **项目** | **注释内容** |
| 参数 | 参数类型  参数用来做什么  约束或前提条件  示例 |
| 字段/属性 | 字段描述  注释所有使用的不变量  示例  并行事件  可见性决策 |
| 类 | 类的目的  已知的问题  类的开发/维护历史、版本  注释出采用的不变量  并行策略 |
| 接口 | 目的它应如何被使用以及如何不被使用 |
| 局部变量 | 用处/目的 |
| 成员函数注释 | 成员函数做什么以及它为什么做这个  哪些参数必须传递给一个成员函数  成员函数返回什么  已知的问题  任何由某个成员函数抛出的异常  包含任何修改代码的历史  如何在适当情况下调用成员函数的例子  适用的前提条件和后置条件 |

### 5.7、其他注释

应对重要的变量加以注释，以说明其含义等。

应对不易理解的分支条件表达式加注释。不易理解的循环，应说明出口条件。过长的方法实现，应将其语句按实现的功能分段加以概括性说明。 对于异常处理当注明正常情况及异常情况或者条件，并说明当异常发生时程序当如何处理。

## 命名

### 6.1、基本原则

规范的命名能使程序更易阅读，从而更易于理解。它们也可以提供一些标识功能方面的信息，有助于更好的理解代码和应用。因此最好使用一些有实际意义的英文或缩写来命名。

* 使用可以准确说明变量/字段/类/接口/包等的完整的英文描述符。例如，采用类似 firstName，listAllUsers 或 CorporateCustomer 这样的名字，严禁使用汉语拼音及不相关单词命名，虽然Java支持Unicode命名，但本规范规定对包、类、接口、方法、变量、字段等不得使用汉字等进行命名。
* 采用该领域的术语。如果用户称他们的“客户” (clients) 为“顾客” 。(customers)，那么就采用术语 Customer 来命名这个类，而不用 Client。
* 采用大小写混合，提高名字的可读性。一般应该采用小写字母，但是类和接口的名字的首字母，以及任何中间单词的首字母应该大写。包名全部小写。
* 尽量少用缩写，但如果一定要使用，当使用公共缩写和习惯缩写等，如实（implement）可缩写成impl，经理（manager）可缩写成mgr等。
* 避免使用长名字（最好不超过 25 个字母）。
* 避免使用相似或者仅在大小写上有区别的名字。
* 避免使用数字，但可用2代替to，用4代替for等，如：go2Jsp。

### 6.2、文件、包

* 文件名当与其类严格相同，所有单词首字母大写。
* 包名一般以项目或模块名命名，少用缩写和长名，一律小写。
* 基本包：weaver，所有包、文件都从属于此包。
* 包名按如下规则组成： weaver.[模块名].[子模块名].

如：

|  |
| --- |
| weaver.workflow  weaver.workflow.request |

**不得将类直接定义在基本包下，所有项目中的类、接口等都当定义在各自的项目和模块包中。**

### 6.3、类、接口

所有单词首字母大写。使用能确切反应该类、接口含义、功能等的词。一般采用名词。 接口可带I前缀或able、ible、er等后缀。

## 字段

### 7.1、常量

采用完整的英文大写单词，在词与词之间用下划线连接，如：PRO\_ADDUSER。

### 7.2、变量和参数

对不易清楚识别出该变量类型的变量应使用类型缩写作其前缀，如字符串使用strXXX,boolean使用isXXX，hasXXX等等。除第一各个单词外其余单词首字母大写。

对私有实例变量可使用\_前缀，但在其存取方法中则应该将其前缀去掉。

### 7.3、集合

一个集合，例如数组和矢量，应采用复数命名来表示队列中存放的对象类型。命名应采用完整的英文描述符，名字中所有非开头的单词的第一个字母应大写，适当使用集合缩写前缀。如：

|  |
| --- |
| Vector vProducts = new Vector(); //产品向量  Array aryUsers = new Array(); //用户列表 |

### 7.4、其他

命名时应使用复数来表示它们代表多值。如：orderItems。

## 方法

方法的命名应采用完整的英文描述符，大小写混合使用：所有中间单词的第一个字母大写。方法名称的第一个单词常常采用一个有强烈动作色彩的动词。 取值类使用get前缀，设值类使用set前缀，判断类使用is(has)前缀。

例：

|  |
| --- |
| getName()  setSarry()  isLogon() |

方法参数建议顺序：(被操作者，操作内容，操作标志，其他.)

例：

|  |
| --- |
| public void replace(String sourceStr, //源字串  String oldStr, //被替换字串  String newStr){ //替换为字串  } |

### 8.1、异常

异常类名由表示该异常类型的单词和Exception组成，如ActionException。

异常实例一般使用e、ex等，在多个异常时使用该异常名或简写加E，Ex等组成，如：

SQLEx

ActionEx

### 8.2、命名约定表

|  |  |  |
| --- | --- | --- |
| **操作项** | **命名约定** | **示例** |
| 参数 | 使用传递值/对象的完整的英文描述符。 | userID |
| 字段/属性 | 字段采用完整的英文描述，第一个字母小写，任何中间单词的首字母大写。 | firstName |
| 布尔型的获取成员函数 | 所有的布尔型获取函数必须用单词 is（has）做前缀。 | isString()  hasMoney() |
| 类 | 采用完整的英文描述符，所有单词的第一个字母大写。 | Customer |
| 编译单元文件 | 使用类或接口的名字，或者如果文件中除了主类之外还有多个类时，加上前缀 java 来说明它是一个源码文件。 | Customer.java |
| 部件/组件 | 使用完整的英文描述来说明组件的用途，将组件类型使用匈牙利命名法则作其前缀。 | btnOK,cboTypeList |
| 构造函数 | 使用类名 | Customer() |
| 析构函数 | Java 没有析构函数，但一个对象在垃圾收集时，调用成员函数 finalize() 。 | finalize() |
| 异常类名 | 由表示该异常类型等的单词和Exception组成 | SQLException  ActionException |
| 异常实例名 | 通常采用字母 e 、ex表示异常。  多个异常时使用异常名或其简写加E、Ex等构成 | e  SQLEx |
| 静态常量字段（常量） | 全部采用大写字母，单词之间用下划线分隔。采用静态常量获取成员函数。 | DEFAULT\_NAME |
| 获取成员函数 | 被访问字段名的前面加上前缀 get。 | getUserName() |
| 接口 | 采用完整的英文描述符说明接口封装，所有单词的第一个字母大写。使用I前缀，其后使用able,. | IRunnable  IPrompter |

## 声明

### 9.1、基本原则

声明的基本原则是遵守Java语言规范，并遵从习惯用法。

### 9.2、包

在导入包时当完全限制代码所使用的类的名字,尽量少用通配符的方式，但导入一些通用包，或用到一个包下大部分类时，则可是使用通配符方式，如：

|  |
| --- |
| import weaver.workflow.request;  import java.util.\*; |

同一包中的类导入时当声明在一起，可由编辑器自动完成此功能。

重要的包当添加注释。

### 9.3、方法

良好的程序设计应该尽可能减小类与类之间耦合，所遵循的经验法则是：尽量限制成员函数的可见性。如果成员函数没必要公有 (public)，就定义为保护 (protected)；没必要保护 (protected)，就定义为私有 (private)。

## 类与接口

### 10.1、基本原则

类的划分粒度，不可太大，造成过于庞大的单个类，也不可太细，从而使类的继承太深。一般而言，一个类只做一件事；另一个原则是根据每个类的职责进行划分，比如用User来存放用户信息，而用UserDAO来对用户信息进行数据访问操作（比如存取数据库），用UserBroker来封装用户信息的业务操作等。

多个类中使用相同方法时将其方法提到一个接口中或使用抽象类，尽量提高重用度。

将不希望再被继承的类声明成final，例如某些实用类，但不要滥用final，否则会对系统的可扩展性造成影响。

将不希望被实例化的类的缺省构造方法声明成private。

### 10.2继承与组合

尽量使用组合来代替继承，一则可以使类的层次不至于过深，而且会使类与类，包与包之间的耦合度更小，更具可扩展性。

## 方法

### 11.1、基本原则

一个方法只完成一项功能，在定义系统的公用接口方法外的方法应尽可能的缩小其可见性。 避免用一个类是实例去访问其静态变量和方法。 避免在一个较长的方法里提供多个出口：

|  |
| --- |
| //不要使用这钟方式，当处理程序段很长时将很难找到出口点  if(condition){  return A;  }else{  return B;  }  //建议使用如下方式  String result = null;  if(condition){  result = A;  }else{  result = B;  }  return result; |

### 11.2、参数和返回值

避免过多的参数列表，尽量控制在5个以内，若需要传递多个参数时，当使用一个容纳这些参数的对象进行传递，以提高程序的可读性和可扩展性。

## 表达式与语句

### 12.1、基本原则

表达式和语句当清晰、简洁，易于阅读和理解，避免使用晦涩难懂的语句。 每行至多包含一条执行语句，过长当换行。 避免在构造方法中执行大量耗时的初始化工作，应当将这中工作延迟到被使用时再创建相应资源，如果不可避免，则当使用对象池和Cache等技术提高系统性能。 避免在一个语句中给多个变量赋相同的值。它很难读懂。

尽量在声明局部变量的同时初始化。唯一不这么做的理由是变量的初始值依赖于某些先前发生的计算。

一般而言，在含有多种运算符的表达式中使用圆括号来避免运算符优先级问题（即使运算符的优先级对你而言可能很清楚 ）。

### 12.2、循环语句

循环中必须有终止循环的条件或语句，避免死循环。

当在for语句的初始化或更新子句中使用逗号时，避免因使用三个以上变量，而导致复杂度提高。若需要，可以在for循环之前(为初始化子句)或for循环末尾(为更新子句)使用单独的语句。因为循环条件在每次循环中多会执行一次，故尽量避免在其中调用耗时或费资源的操作，比较一下两种循环的差异：

**//不推荐方式**

|  |
| --- |
| while(index < products.getCount()){  //每此都会执行一次getCount()方法，  //若此方法耗时则会影响执行效率  //而且可能带来同步问题，若有同步需求，请使用同步块或同步方法  } |

**//推荐方式**

//将操作结构保存在临时变量里，减少方法调用次数

|  |
| --- |
| final int count = products.getCount();  while(index < count){ |

## 错误与异常

### 13.1、基本原则

通常的思想是只对错误采用异常处理：逻辑和编程错误，设置错误，被破坏的数据，资源耗尽，等等。

通常的法则是系统在正常状态下以及无重载和硬件失效状态下，不应产生任何异常。

对于经常发生的可预计事件不要采用异常。

在本地进行安全性检查，而不是让用户去做。

### 13.2、异常的捕捉与处理

捕捉异常是为了处理它，不要捕捉了却什么都不处理而抛弃之，最低限度必须向控制台输出当前异常，如果你不想处理它，请将该异常抛给它的调用者，对每个捕捉到的异常都调用**writelog()**输出异常信息，避免因异常的湮没。