## 同济大学异构与可信计算实验室

## 源程序编码规范

## （C/JAVA/JSP源程序编码规范）

1文件名命名

用1-3文名词英文单词书写，首字符大写，将文件看做一个包，把处理一大类功能的多个函数(即同一个主题)放在此文件中，例如：CommincationOpetationMode.c

2 源程序文件的创建要求

每个源文件的最开始部分必须包含如下格式的注释：

**以JSP文件为例，如下：**

<! --

文件名称: 本文件的名称

创建日期: 创建文件的日期

开发者姓名: 开发者的姓名（可以多个）

功能: 该文件的简要功能说明

-- !>

**以c语言为例如如下：**

/\*

文件名称: 本文件的名称

创建日期: 创建文件的日期

开发者姓名: 开发者的姓名（可以多个）

功能: 该文件的简要功能说明

\*/

3 源程序内部变量定义

**（1）一般变量的定义**

* 提取变量的含义，以“名词”或“名词性短句”翻译成英语作为变量名，一般用1至3个单词为宜；
* 变量名首字符为小写，其他单词的首字符为大写；
* 在变量名前冠予变量数据类型说明：**（可选）**

**int** 变量前面加i；

**byte** 变量前面加by；

**string** 变量前面加s；

**char** 变量前面加c；

**long** 变量前面加l；

**float** 变量前面加f；

**double** 变量前面加d；

**boolean** 变量前面加b；

**file** 变量前面加fl；

**ArrayList** 变量前面加al；

**MyFont** 变量前面加mf；(MyFont是我们自定义的一个字体类)

**ResultSet** 变量前面加rs；

**FileInputStream** 变量前面加fi；

**FileOutputStream**变量前面加fo；

**数组变量在上述基础上再加a；**

* 变量定义按a-z顺序排列，而非先用先书写；
* 示例：一个变量既是函数的参数,又是整型数组变量,则最后变量前面应加mai。

**（2）全局作用域的外部变量的定义**

变量名前加ev\_, 后带一般变量的命名法。

如： ev\_iBodyWeight

**（3）文件内作用域的变量的定义**

变量名前加fv\_, 后带一般变量的命名法。

如： fv\_iBodyWeight

4类的定义和设计

**(1) 类设计一般原则**

* 提取相应领域或过程中的**客观**实体作为对象，抽象成类；
* 用首大写字母的1~3个名词性英文单词，从含义上命名成类，例如TongjiStudent；
* 类不宜太大或太小，一般小于10个成员属性变量，小于10个成员行为函数；
* 尽量做到类对应的功能独立，即高内聚，松耦合；
* 在整个软件中，同一功能，即便是类似功能，只能用类定义一次；
* 尽量做到层次化，尽量做到继承和重用；
* 在类的设计时，尽量做到只需要做少量的修改或不修改，便可适应需求的变化，或适应性维护；
* 尽量做到类的友好设计，使得可移植到类似的软件系统中去，体现软件设计和开发经验的积累。

**(2) 成员变量（属性）设计原则**

* 成员变量命名参考上面第3点，即“源程序内部变量定义”
* 成员变量的用途用中文解释之， 特别要解释public 变量；
* 对于数值变量说明其取值范围和边界；
* 对于标志变量说明标志的含义；
* 对于控制变量解释取值和控制的动作；
* 变量为数据结构时，要做详细解释和说明

**(3) 成员函数（方法）设计原则**

* 成员函数编码总体上可参考下面第6点，即“函数编码规则”；
* 成员函数的功能一定要加以解释；
* 尽量不要设计重载函数，不同参数用函数名加以区别；
* 函数参数和返回值，以及涉及到的外部文件要求解释一下；
* 对函数涉及的关键算法有必要加以解释；
* 重要的函数有必要画出程序流程图；

**(4) 类设计的特殊要求**

* 基类和公共类要优先设计和实现；
* 拒绝重复，主动复用，体现一个变化，仅一处修改。也就是说，一个用户需求或软件环境或配置改变了，只允许在一个类中作修改，不允许改多处；
* 关键类的生命周期必须描述出来，也就是说，解释如何使用该类。在Rose中可用StateDiagram 来画；
* 同一子目录中的多个类，以及和其他目录中的类，必须画出关联关系图，体现继承，使用，泛化等关系；

5 常数标识符

* 全大写，每个单词间用下横线连接。

1. 函数编码规则
2. **命名**

* 提取函数功能语义，以**“谓宾”**形式翻译成英语短句作为函数名；
* 名称以2-5个单词为宜，每个单词首字符大写。例如findTwoRoot()；
* 函数名第1个字母小写，以便区别类的命名；

1. **参数**

* 输入参数是变量，其名称的定义遵循内部变量的定义，但前面加m ；
* 输出参数是变量，其名称的定义遵循内部变量的定义，但前面加o ；
* 输入、输出参数的个数以**3-8**个为宜；
* 返回值为函数是否成功执行的状态。0为成功，非零（－1）为不成功。

1. **函数的结构**

* 建议单入口单出口，例如C语言函数内，只有一个return语句；
* 必须有异常处理，例如open(File,…)之后，要判断打开文件是否出错；

1. **函数的注释**

在函数的开始部分，应对函数的功能作简要的注释。对函数的输入参数、输出参数的含义作简要的说明，并注释说明该函数的主要流程。格式如下：

/\*---

功能: 该文件的简要功能说明。

输入参数：所有输入参数含义的说明。

主要流程：如果必要的话，说明该函数的流程。

---\*/

7 { }对的使用

* 同一对{ }位于上下行的同一列的位置；
* 在一对{ }内书写语句书写时，后移一个Tab列位置，建议Tab设置为4个空格位置；
* 前一对{}和后一对{}相隔一个Tab列位置；
* 一个函数最多包括8对{ }。因为太多{ }对，程序难以阅读，逻辑层次很不清楚，建议可以改写该函数成两个或多个函数。

8 注释规定

* 注释尽量使用中文。注释应简洁，增强程序的可读性。
* 在第一个“{”的上一行开始注释，举例如下：

[功能简述]，[参数含义]：[接口]：[最新作者]：[最新修改日期]：

* 对**全局变量**的使用时，应详细注释。
* 对**系统资源**的使用和存取时，要详细注释。
* **异常处理**时，要注释原因和处理方法。
* **调用其它函数**说明出处，即//from \*\*\*bean, \*\*\*.java
* 对**判断条件**、**循环语句**等应注释说明其用途。例如：

if (oStudent.osName != null ) **//**学生名为空吗？

{ **//**不空，则

Do Something; **//**做该做的事情

} **//e**ndif oStudent.osName

1. SQL语句的规范

* 每个SQL语句中的**关键字**一律**大写**；
* 对于任何的SQL语句不论复杂与否，每个子句应单放一行，例如：

**SELECT \* FROM table WHERE column\_1 = ‘Value’ ORDER BY column\_1**

应写为：

**SELECT \***

**FROM table**

**WHERE column\_1 = ‘value’**

**ORDER BY column\_1**

* 若SQL语句的WHERE子句条件比较复杂，尽量多用括号来分隔。

10 JSP的设计规则

* JSP编程是WEB编程和单机Application编程的主要区别；
* JSP编程主要处理可视化用户界面上和WEB网络通讯上的功能和任务，类封装成Java Bean后，主要用于处理不可视化的功能；JSP将WEB界面，以及Java Bean串起来，完成一个业务流程的处理；
* 一个不同的页面显示，则用一个jsp文件书写和设计；
* 所有的JSP文件同样构成树形层次结构，子目录，子子目录的设计和功能划分，以及类的层次结构相对应；
* 根据JSP处理功能的含义，用首大写字母的3~5个动宾性英文单词，从含义上命名，后缀为.jsp；
* 优先提取公共和独立的JSP处理模块，组成CommonModule子目录，率先设计和实现；
* 单个JSP的处理过程要求用程序流程图的方式粗细恰当的画出来，在rational rose中，特别用Text Note工具在开始处指出使用了哪些Bean，该描述作为相应JSP的子目录。