文件编号：NCI/SP-SE06-S01

NCI-编程规范

撰 写 人：张志勇
编写时间：2008-6-5
部门名称：研究中心
审 核 人：EGP
审核时间：2008-8-28

杭州新世纪信息技术股份有限公司
HANGZHOU NEW CENTURY INFORMATION TECHNOLOGY CO., LTD.

版本号:V1.0

修 订 记 录

*A – 增加 M –修改 D –删除

版本	日期	图或表或章节号	AMD	标题或简要描述	变更申请号	修订者	批准者
V1.0	2008-6-5		AMD			张志勇	李云水

目　　录

1. 目标

为了使软件开发过程有章可循，保证软件质量，加强开发管理。

本规范为一套编写高效可靠的 Java 代码的标准、约定和指南。它以安全可靠的软件工程原则为基础，使代码易于理解、维护和增强，提高生产效率。同时，将带来更大的一致性，使软件开发团队的效率明显提高。

2. 适用范围

本规范适用于采用J2EE规范的项目中，所有项目中的JAVA代码（含JSP,SERVLET,JAVABEAN）均应遵守这个规范。同时，也可作为其它项目的参考。

3. 参考资料（可选）

- CMMI Product Team 英文版 CMMI-DEV-v1.2

4. 注意事项

该规范分为强制执行和[可以参考] 和两部分。

5. 代码规范

5.1. JAVA 源代码规范

5.1.1. 命名

1. Package 的命名

Package 的名字应该都是由一个小写单词组成，例如：com.nci.modules。

此外，对于包名我们做如下约定：

- 工具函数类包名前缀为.util
- 服务类包名前缀为.service

实现类为.impl

- 展示类包名前缀为.view

构造器.builder

控件.ctrl

其它显示标签.display

- Ajax类包名前缀为.ajax
- Servlet类包名前缀为.servlet
- test case类包名前缀为.test

2. Class 的命名

Class 的名字必须由大写字母开头而其他字母都小写的单词组成，例如：DataFile或InfoParser。

3. Interface 的命名

Interface的命名可以采取一下2中命名规范。

- 以类名（Class Name）加后缀”IF”来区分，比如:实现类为BusinessDAO，接口可以命名为BusinessDAOIF
- 在实现该接口的类加后缀”Impl”来区分，比如: 接口为BusinessDAO，:实现类可以命名我BusinessDAOImpl

4. Class 变量的命名

- 变量的名字必须用一个小写字母开头。后面的单词用大写字母开头，例如：debug 或 inputFileSize。
- Static Final 变量的命名 ：Static Final 变量的名字应该都大写，并且指出完整含义，例如：MAX_UPLOAD_FILE_SIZE=1024。
- 参数的命名：参数的名字必须和变量的命名规范一致。
- 数组的命名 ：数组应该总是用下面的方式来命名：

```
byte[] buffer;
```

而不是：

```
byte buffer[];
```

- 方法的参数

使用有意义的参数命名，如果可能的话，使用和要赋值的字段一样的名字：

```
SetCounter(int size) {
{
    this.size = size;
}
```

下面为命名格式参照表:

标示符类型	命名约定	例子
包	1. 全部小写。 2. 标识符用点号分隔开来。为了使包的名字更易读。 3. 全局包的名字用你的机构的Internet保留域名开头 。	com.nci.应用名.子应用.类型。 如： com.nci.hr.hi.service
类，接口	1. 类的名字应该使用名词。 2. 每个单词第一个字母应该大写。 3. 避免使用单词的错写，除非它的缩写已经广为人知，如HTTP。	Class Hello ; Class HelloWorld ; Interface Apple ;

方法	1. 第一个单词一般是动词。
	2. 第一个字母是小写，但是中间单词的第一个字母是大写。
	3. 如果方法返回一个成员变量的值，方法名一般为get+成员变量名，如若返回的值是boolean变量，一般以is作为前缀。
	4. 如果方法修改一个成员变量的值，方法名一般为：set + 成员变量名。
		getName();
		setName();
		isFirst();

变量	1. 第一个字母小写，中间单词的第一个字母大写。 2. 不要用_或&作为第一个字母。 3. 尽量使用短而且具有意义的单词。 4. 单字符的变量名一般只用于生命期非常短暂的变量。i,j,k,m,n一般用于Integer；c,d,e一般用于characters。 5. 如果变量是集合，则变量名应用复数。	String myName; int i; int n; char c; int[] students; Object btNew;//(bt是Button的缩写)
常量	1. 所有常量名均全部大写，单词间以”_”隔开。	int MAX_NUM;

5.1.2. 代码格式

1. 文件头声明

源文件的头部需要一个 history 段，对于每次对源文件的重大改动，都需要在 history 段中注明。该段定义在 package 和 import 之间，例如：

```
/**
* <p>标题: WorkItemDo.java</p>
* <p>描述:
* 工作任务处理类
* Version 1.0
* </p>
* <p>版权: 2003 Hangzhou NCI System Engineering,  Ltd.</p>
* <p>公司: Hangzhou NCI System Engineering,  Ltd.</p>
*
* 修改记录
* 日期 作者 修改类型 描述
* 2004-08-26 张志勇 Created
* 2005-03-06 张志勇 修改 增加了若是回退操作，则突略所有缺省错误检查
* 2005-03-22 张志勇 修改 增加了返回时恢复原始参与者信息
```

*/

2. import顺序

import包按以下顺序：

- jdk标准包
- java扩展包（例如servlet，javamail，jce等）
- 使用的外部库的包（例如xml parser）
- 使用的项目的公共包
- 使用的模块的其他包

每一类import后面加一个换行。

例如：

```
import java.io.*;
import java.util.*;

import javax.servlet.*;
import javax.mail.*;

import org.apache.xml.*;

import com.nci.*;
import com.nci.util.*;
import com.nci.framework.*;
```

3. 代码块书写格式

可以选择以下任意一种代码块的书写方式：

```
if (true){
    //body
}或
if (true)
{
    //body
}
```

建议使用第一种书写方式。如果是修改他人的代码，必须使用代码原来的书写方式。 对于代码块过长，超过1屏以上，}后面要说明属于那个代码块，例如：

```
if (i > 100)
{
    //too many lines more than one screen
```

```
}// if (i > 100)
```

4. 关于缩进

缩进使用4个连续空格，不要在源文件中保存tab字符， 请注意调整所用的IDE工具，打开将tab转换为空格功能。

5. 页宽

页宽应该设置为80字符. 源代码一般不会超过这个宽度，并导致无法完整显示，但这一设置也可以灵活调整. 在任何情况下，超长的语句应该在一个逗号或者一个操作符后折行. 一条语句折行后，应该比原来的语句再缩进4个空格。

6. 操作符

操作符左右各用一个空格分隔。

例如：

```
int a = b;
if (a > 0);
```

7. SQL语句

代码中书写的sql语句要求sql关键字全部大写，表名和字段名小写。例如： SELECT user_id, name FROM account WHERE user_id > ? AND depart = ? ORDER BY name

8. 类和方法定义

类定义或方法定义过长需要换行书写，例如：

```
public class CounterSet extends Observable implements Cloneable
    private PortletSet getPortlets( Portlets portlets,
        RunData rundata,
        boolean application,
        boolean applicationsOnly ) {
    }
```

5.1.3. 表达式和语句

以下部分为可选项，可以根据项目的实际情况来约定，而不是个人来约定

1. 每行应该只有一条语句。
2. if-else,if-elseif语句，任何情况下，都应该有“{”，“}”，格式如下：

```
if (condition) {
    //语句块;
} else {
    //语句块;
}
```

3. for语句格式如下：

```
for (int i = 0, j = array.length; i < array.length; i++, j--) {
}
```

如果语句为空：

```
for (int i = 0, j = array.length; i < array.length; i++);
```

4. while语句格式如下：

```
while (condition) {
} ;
```

5. do-while语句格式如下：

```
while (condition) {
};
do {
} while (condition);
```

6. switch语句，每个switch里都应包含default子语句,格式如下：

```
switch (number) {
 case RED:{
     //语句块;
     break;
 }
 case GREEN:{
     //语句块;
     break;
 }
 default:
     //语句块;
     break;
}
```

7. try-catch语句格式如下：

```
try {
     number = Integer.parseInt(value);
 } catch (NumberFormatException e) {
 }
```

5.1.4. 注释

public 和 protected的成员变量和方法必须写javadoc注释。超过1句以上的注释使用中文书写。对于代码多于10行的private方法也要写javadoc注释。

对于代码中的逻辑分支或循环条件需要书写注释，例如：

```
if (some condition) {
     //符合某个条件，应该这样处理
 }else{
     //否则应该那样处理
 }
```

5.1.5. 其它

1. 关于属性

类中的属性不能定义为public变量直接存取，而是定义成protect变量并编写get/set方法，例如：

```
protect String myName;
public String getMyName() {
    return myName;
}


public void setMyName(String myName){
    this.myName=myName;
}
```

5.1.6. 代码样例

```
* <p>公司: Hangzhou NCI System Engineering,  Ltd.</p>
*
* 修改记录
* 日期 作者 修改类型 描述
* 2004-08-26 张志勇 Created
* 2005-04-18 张志勇 修改 修改了基数
*/
public class ClassName extends SuperClass {
    private static final int CONSTANT= 0;
    /* This comment may span multiple lines. */
    private static int staticField= 0;
    // This comment may span only this line
    private String field= "zero";
    // TASK: refactor
    public int foo(int parameter) {
        int result=0;

        abstractMethod();
        /*修改前代码
        result= 42*hashCode();
        */

        //2005-04-18 张志勇 修改
```

```
        int local= 50*hashCode();
        //修改结束

        staticMethod();
        result=bar(local) + 24;

        return result;
    }
}
```

5.1.7. 注意事项

1. 避免使用不必要的嵌套

   过多的嵌套会使你的代码复杂化，减弱可读性。

```
Public class test {
    String add (){
        Int c=(a=a+b)+b; //过于复杂
        Return c
    }
}
```

2. 避免在同一行声明不同类型的多个变量

   这样可以使程序更加清晰，避免混乱

```
private int index, index1[];
```

   正确的应该如此：

```
private int index;

private int index1[];
```

3. 在每一行里写一条语句

   这条规则不包括 for 语句：比如：’for (int i = 0; i < 10; i++) x--;’可以增加代码的可读性。

```
public class OSPL {
    int method (int a, int b) {
        int i = a + b; return i; // 可读性不强
    }
```

   正确的：

```
public class OSPLFixed {
    int method (int a, int b) {

        int result=0;

        result= a + b;

        return result;

}
```

4. Statement 在 finally 中切记要关闭

```
Statement stmt=null;

try{

    stmt=初始化

}catch(Exception e){

}finally{

    if(stmt!=null)try{stmt.close();}catch(Exception ex){}

}
```

5. 使用 StringBuffer 拼装字符串

使用 String 连接字符串会造成多次初始化对象，影响性能

```
String sql=”select * from ”+tableName+” where ”+条件;
```

正确的：

```
StringBuffer sql=new StringBuffer(100).append(“select * from ”).

    append(tableName).

    append(“ where ”).append(条件);
```

5.2. jsp/html 代码规范

1. jsp/html描述注释

jsp/html页面顶部必须存在一个基本描述注释，包含功能描述、参数列表和历史修改信息，例如：

```
<%--
/**************************************************
**************************************************/
* <p>标题: file_download.jsp </p>
```

```
* <p>描述:
* 下载文件
* </p>
* <p>版权: 2005 Hangzhou NCI System Engineering,  Ltd.</p>
* <p>公司: Hangzhou NCI System Engineering,  Ltd.</p>
*
* 修改记录
* 日期 作者 修改类型 描述
* 2004-08-26 张志勇 Created
* @version 1.0
*
/**************************************************
**************************************************/
--%>
```

2. jsp头格式

jsp头部一般需要遵循以下格式：

```
<%@ page contentType="text/html;charset=GBK" %>
<%@ page import="java.io.*" %> // jdk标准包
<%@ page import="javax.mail.*" %> // java扩展包
<%@ page import="org.apache.xml.*" %> //使用的外部库的包
<%@ page import="com.sunrise..*" %> //使用的项目的公共包
<%@ page import=" com.sunrise.applications.*" %> //使用的模块的其他包
<%@ include file="some.jsp" %> //include其他的jsp
<%
response.setHeader("Pragma","No-cache");
response.setHeader("Cache-Control","no-cache");
response.setHeader("Expires","0");
%> //一般 jsp 都需要防止缓存
```

3. html格式

- html头一般需要遵循以下格式：

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=GBK">
<title>some title</title>
<link rel="stylesheet" href="some.css" type="text/css">
<script language="javascript">
    //some javascript
</script>
</head>
```

注意：必须指定一个有意义的<title>，严禁出现”Untitled”或”未命名”之类的<title>。

- 所有html标签使用小写
- html页面一般需要设置一个背景色（一般是#FFFFFF）。
- html语法校验

所有的jsp/html页面需要能够使用DreamWeaver正确打开（即html语法正确，没有错误的标记）。

- 注释

一般不使用html注释，除非是有必要让最终用户看到的内容。对于包含JSP代码的html块，必须使用JSP注释。对于没有必要的注释，在发行版本中必须移除。

- form属于域的maxlength

对于text类型的输入域，必须根据数据库字段的长度设置相应的maxlength，例如：数据库类型是VARCHAR(64)，那么maxlength是32（因为中文浏览器对于中文也认为是一个字符）。

5.3. 数据模型规范

1. 命名：数据模型的名字必须由全部大写字母组成，名称前应以应用、子应用作为前辍；应用、子应用、名称应通过“_“分隔开。例如：HR_HI_PERSIONINFO。

5.4. 功能模型规范

1. 命名：功能模型的名字必须由全部大写字母组成，名称前应以应用、子应用作为前辍；应用、子应用、名称应通过“_“分隔开。例如：HR_HI_PERSIONLIST。
2. 存储文件名：功能模型对应的xml文件应以小写字每命名，使用应用、子应用名作为子目录。如员工管理功能模型文件名为：hr_hi_personadmin.xml,功能模型存放子目录为：/hr/hi
3. 注册：功能模型注册时，路径使用斜杠(/)分隔符

5.5. 代码模型规范

1. 命名：代码模型的名字必须由全部大写字母组成，名称前应以应用、子应用作为前辍；应用、子应用、名称应通过“_“分隔开。例如：性别代码，HR_HI_SEXTYPE。
2. 存储：数据量较少的简单类型代码应统一在代码表中存储，如T_JFW_CODE_DATA。

5.6. 其它规范

1. 代码归档格式：最后提交的代码必须不依赖任何IDE，而需要可以使用ant完成所有的编译工作。一般提交的代码目录格式如下：

\
|_bin（存放输出的文件class文件）
|_lib（使用的库）
|_src（源代码）

|_docs（文档）

|_jsps（页面文件）

|_build.xml（ant的build文件）

|_changes.log（代码版本和修改的日志）

2. 限制session的使用：

在代码中使用session需要听取项目经理的意见,项目经理需要在设计文档中登记项目中所有使用到的session的名字和作用。

3. 限制外部包的使用

开发员如果需要使用一个外部包需要听取项目经理的意见。在项目经理批准以前，严禁擅自使用一个外部的包。