

# **jSyncManager.org Coding Standards and Guidelines**

**Initial Draft -- June 25th, 2003**

Written by Brad BARCLAY <bbarclay@jsyncmanager.org>

## **TABLE OF CONTENTS**

Section 0 -- Introduction

Section 1 -- Java Source File Standards

Section 2 -- Property File Standards

## **Section 0 -- Introduction**

### **0.1: The Need for Coding Standards:**

When the jSyncManager was first developed, coding standards weren't an issue -- I coded everything, and I formatted it however I liked, and if someone else didn't like how I formatted my code, it was their problem. The project was closed-source, and it was originally intended for few eyes to see it. It had to be clean to make it maintainable, but otherwise how it was formatted was up to me. And as an individual, I tended to do things my way -- but in a consistent manner.

Things have changed since making the jSyncManager Open Source. There are now more than a dozen developers associated directly with the project, and an untold number of other people who are looking at and working with our code. Many of these developers also have their own way of formatting their code in ways that they're used to, which is often quite different from the way I tend to format my code. Left unchecked, this can quickly lead to code that is difficult to follow and harder to maintain.

Besides which, well-formatted, consistent code lets people know that we care about our project, and how we present it to the world.

As such, I've drawn up a pretty comprehensive set of coding guidelines. These guidelines are neither better nor worse than any other system of formatting code -- it's simply the standard upon which the jSyncManager developers should strive to comply with. Some of these guidelines are the result of agreement by the current jSyncManager developers, some are arbitrary, and some are executive decisions made by myself.

This document is digital, stored on magnetic and optical media. It is not carved in stone, and thus is subject to change as the need arises.

### **0.2: A Note on Indentation**

For the purposes of the specification, indentation is defined as being three spaces per level of indentation. Please do not use tab characters for indentation, nor more or less than three spaces. Three should be the number thou shalt count, and the number of the counting shall be three. Four shalt thou not count, nor either count thou two, excepting that thou then proceed to three. Five is right out. If you're using a text editor, please ensure that it is set to insert three spaces whenever 'tab' is pressed. If you're using an Integrated Development Environment, please set it now to three spaces per tab key press.

## **Section 1 -- Java Source File Standards**

### **1.0: License Headers**

All Java source files must start with one of our two standard source headers. The header used depends on the license for the package the source file is contained within -- either GNU General Public License (GPL) or GNU Lesser General Public License (LGPL).

The GPL header is as follows (and may be found in the jSyncManager CVS Repository as `gpl_java_source_header.txt`):

```
// -----  
// The jSyncManager Project -- Source File.  
// Copyright (c) 1998 - 2003 Brad BARCLAY <bbarclay@jsyncmanager.org>  
// -----  
// OSI Certified Open Source Software  
// -----  
//  
// This program is free software; you can redistribute it and/or modify it  
// under the terms of the GNU General Public License as published by the Free  
// Software Foundation; either version 2 of the License, or (at your option)  
// any later version.  
//  
// This program is distributed in the hope that it will be useful, but WITHOUT  
// ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or  
// FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for  
// more details.  
//  
// You should have received a copy of the GNU General Public License along  
// with this program; if not, write to the Free Software Foundation, Inc.,  
// 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA  
//  
// -----  
// $Id$  
// -----
```

The LGPL header is as follows (and may be found in the jSyncManager CVS Repository as lgpl\_java\_source\_header.txt):

```
// -----  
// The jSyncManager Project -- Source File.  
// Copyright (c) 1998 - 2003 Brad BARCLAY <bbarclay@jsyncmanager.org>  
// -----  
// OSI Certified Open Source Software  
// -----  
//  
// This library is free software; you can redistribute it and/or modify it  
// under the terms of the GNU Lesser General Public License as published  
// by the Free Software Foundation; either version 2.1 of the License, or  
// (at your option) any later version.  
//  
// This library is distributed in the hope that it will be useful, but  
// WITHOUT ANY WARRANTY; without even the implied warranty of  
// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU  
// Lesser General Public License for more details.  
//  
// You should have received a copy of the GNU Lesser General Public  
// License along with this library; if not, write to the Free Software  
// Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA  
// -----  
// $Id$  
// -----
```

The correct header *must* be the first element of any Java source file.

### 1.1: Package Declaration

The next element in any Java source file should be a single line declaring what package the source file belongs to. This statement should be aligned with column 0 of the source file.

### 1.2: Import Statements

The third element in any Java source file should be a list of import statements, if necessary. It is strongly recommended that import statements are grouped together based on their API set (i.e.: jSyncManager API, core Java API, Java Communications API, etc.). These statements, one per line, should be aligned with column 0.

### 1.3: Major Section Separator

The next element in any Java source file should be a *major section separator*. The *major section separator* is a single line of text, defined as follows:

1. A pair of forward-slash characters, aligned at column 0, followed by
2. A single space, followed by
3. 75 '=' (equals) characters

## 1.4: Class or Interface JavaDoc Comment

The next element in every Java source file must be a JavaDoc comment describing the purpose of the class, along with some information on the source. The first line should be aligned at column 0, with subsequent lines aligned with their beginning '\*' (asterix) character at column 2.

Source file JavaDoc comments should take the following form:

```
/** {single sentence briefly describing the class or interface}.
 * {Paragraph with more detail on what the class does}.
 * @author {author NAME} &lt;{author e-mail}>
 * <br>Last modified by: $Author$ on $Date$.
 * @version $Revision$
 */
```

Elements in curly braces should be filled in by the developer creating the class. The developers name should be formatted such that their inherited (family) name is capitalized. The authors e-mail address should be a valid e-mail address, with preference given to the developers @jsyncmanager.org address, if possible.

## 1.5: Class or Interface Declaration

The next element in the source file should be the class or interface declaration, as described in the *Java Language Specification* sections 8.1 and 9.1.

This statement should be aligned at column 0, and should comprise a single line, with data presented in the following order (as relevant):

```
{qualifier} class {Class name} extends {Parent Class} implements {Implemented Interfaces} {
```

Note that the block-open character at the end of the statement ('{') should be on the same line as the class declaration.

## 1.6: Class and Instance Variable Declarations

Following the class or interface declaration, any and all class and interface variable declarations should follow. Additionally, class-wide named inner classes should also be defined in this section, with an order defined by the developer of the class, as should any static initializers. All statements should be indented by three spaces.

## 1.7: Major Section Separator

Following all field and class-wide named inner class declarations, a major section separator, as defined in s1.3 should be inserted, aligned at column 0.

## 1.8 Instance Constructors

Following the major separator, zero or more constructors may be specified. Each constructor declaration should be indented by three spaces, with the contents indented using three spaces as necessary.

If multiple constructors are present in a source file, they should be separated by a *minor section separator*. A *minor section separator* is defined as follows:

1. A pair of '/' (forward-slash) characters, followed by
2. A single space character, followed by
3. 75 '-' (minus) characters.

All constructor declarations should be one a single line. They should end with the following line, indented by three spaces:

```
    } // end-constructor
```

## 1.9: Major Section Separator

Following all constructors, a major section separator, as defined in s1.3 should be inserted, aligned at column 0.

## 1.10: Method Declarations

Next, all static and non-static methods should be declared (and where necessary, implemented). Declarations should be indented by three spaces, and should be preceded with a full JavaDoc comment header, with references to all parameters, return information, and throwables. As with constructors, the block opening character ('{') should be on the same line as the method declaration.

If multiple methods are present, each should be separated by a *minor section separator*, as defined in s1.8.

All static methods that contain an implementation should end with the following line, indented by three spaces:

```
    } // end-static-method
```

All instance methods that contain an implementation should end with the following line, indented by three spaces:

```
} // end-method
```

### 1.11: Major Section Separator

Following all methods, a major section separator, as defined in s1.3 should be inserted, aligned at column 0.

### 1.12: End of Class Block

All source files should end with the following line, aligned at column 0:

```
} // end-class
```

### 1.13: Sub-minor Section Separator

At the discretion of the developer, any exceedingly long section that would benefit from visible separation that wouldn't otherwise be separated by a *major* or *minor* separator may elect to use a *sub-minor section separator*, defined as follows:

1. A pair of forward-slash '/' characters, followed by
2. A single space, followed by
3. '.' (period) characters to column 78.

The alignment of the beginning of a *sub-minor* separator may either be at column 0, or at the level of indentation of the code surrounding it, at the developers discretion.

### 1.14: Block Opening and Closing

As should be noted, typically block opening is aligned to the end of the statement line that defines the block, while block closing characters are typically (excepting comments, or line terminators (';')) as may be required by the *JLS* the last character on their line.

There are, however, some exceptions to be noted.

If statements that define a multi-line "else" condition should imbed the "else" statement between the preceding block close character, and the proceeding block opening character, i.e.:

```
if (statement) {  
    ...code...
```



```
    } else {  
        ...code...  
    }
```

Likewise, try...catch...finally statements should likewise be compacted, using:

```
    try {  
        ...code...  
    } catch (type declaration) {  
        ...code...  
    } finally {  
        ...code...  
    }
```

Do...while blocks should likewise have the "while" statement on the same line as, and immediately following the block closure character, i.e.:

```
    do {  
        ...code...  
    } while (statement);
```

## **Section 12-- Property File Standards**

### **2.0: License Headers**

All property files must start with one of our two standard source headers. The header used depends on the license for the package the source file is contained within -- either GNU General Public License (GPL) or GNU Lesser General Public License (LGPL).

The GPL header is as follows (and may be found in the jSyncManager CVS Repository as `gpl_property_file_header.txt`):

```
# -----
# The jSyncManager Project -- Source File.
# Copyright (c) 1998 - 2003 Brad BARCLAY <bbarclay@jsyncmanager.org>
# -----
# OSI Certified Open Source Software
# -----
#
# This program is free software; you can redistribute it and/or modify it
# under the terms of the GNU General Public License as published by the Free
# Software Foundation; either version 2 of the License, or (at your option)
# any later version.
#
# This program is distributed in the hope that it will be useful, but WITHOUT
# ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
# FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for
# more details.
#
# You should have received a copy of the GNU General Public License along
# with this program; if not, write to the Free Software Foundation, Inc.,
# 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
#
# -----
# $Id$
# -----
```

The LGPL header is as follows (and may be found in the jSyncManager CVS Repository as lgpl\_property\_file\_header.txt):

```
# -----
# The jSyncManager Project -- Source File.
# Copyright (c) 1998 - 2003 Brad BARCLAY <bbarclay@jsyncmanager.org>
# -----
# OSI Certified Open Source Software
# -----
#
# This library is free software; you can redistribute it and/or modify it
# under the terms of the GNU Lesser General Public License as published
# by the Free Software Foundation; either version 2.1 of the License, or
# (at your option) any later version.
#
# This library is distributed in the hope that it will be useful, but
# WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
# Lesser General Public License for more details.
#
# You should have received a copy of the GNU Lesser General Public
# License along with this library; if not, write to the Free Software
# Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
#
# -----
# $Id$
# -----
```

The correct header *must* be the first element of any property file.

## 2.1: Required Property File Comments

The next element of the property file must be the following, aligned at column 0:

```
# {Description of the property file}.
# Author: {author NAME} &lt;{authors e-mail address}&gt;;
# Last modified by: $Author$ on $Date$.
# Version: $Revision$
```

Information enclosed above in curly braces should be filled in by the initial creator of the file. The inherited (family) name of the author should be capitalized. The authors e-mail address should be a valid e-mail address, preferably using their [@jsyncmanager.org](mailto:@jsyncmanager.org) address, if available.

## 2.2: Required Properties

Following the property file comment, the following properties *must* be defined:

```
Language={Name of language}
Translator={name of AUTHOR}
Translator E-Mail={authors e-mail address}
```

The name of the language should be as it is *in* that language (i.e.: *français* for French), and it should match the language used by the resources in the file. As with the preceding section, the inherited (family) name of the author should be capitalized. A valid e-mail address should be used (with preference given to an *@jsyncmanager.org* address for the author, if available).