

# JOnAS OpenSource Java EE Application Server - CookBooks - Code Convention

## [Entry Point](#)

written on 04/13/2005 by Florent Benoit Entry Point

A good starting document is the [java code conventions](#) This convention should be used by contributors when submitting source code in JOnAS.

Also a tools used to check the compliance is the [checkstyle plugin](#) and the [eclipse checkstyle plugin](#) Settings used with eclipse are available on [JOnAS SVN](#)

Here is the detailed settings used by JOnAS File Organization

## **Header**

The header of each file should contains the LGPL and the date.

When modifying a file, change the year to the current year by appending it to the existing.

ie : if there was '1999' or '2004' put '1999-2006' or '2004-2006'

Also, the tag \$Id:\$ should be added.

Here is an example of a header used in JOnAS java files:

```
/** * JOnAS: Java(TM) Open Application Server * Copyright (C) 2006 Bull S.A.S. * Contact:
jonas-team@objectweb.org * * 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 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 * *
----- * $Id:$ *
*/
```

## **Imports**

No wildcard imports are authorized. They have to be avoided.

The imports should reference the correct classes.

ie : if you need to use List and ArrayList classes, don't do:

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

but :

```
import java.util.List; import java.util.ArrayList;
```

Unused imports need to be removed.

Note that [eclipse IDE](#) allow you to do the job. (Source/Organize imports or Shift+Ctrl+O)

# JOnAS OpenSource Java EE Application Server - CookBooks - Code Convention

## ***Class and Interface Declarations***

The Class and Interface name begin with an Uppercase and have a javadoc comment with the @author tag. ie :

```
/** * This is an example of a class * @author Florent Benoit */ public class MyClass implements MyInterface { }
```

Indentation / WhiteSpace

## ***Indentation***

**No tabs** characters are used. Always use space characters and the number of the characters for an indent is **4 spaces**.

There is eclipse plugin like [AnyEdit](#) for changing tabs into spaces.

For the wrapping of lines, follow the [Java code conventions on this part](#)

## ***WhiteSpace***

Also, the trailing spaces should be removed. [AnyEdit](#) plugin do this.

Use whitespaces in for() loop, while(), when concatenating strings.

One before the separator and one after. ie :

```
for (int i = 0; i < TEST.length; i++) { String str = "this is" + " " + "a test with value = " + i; }
```

and not :

```
for (int i= 0; i <TEST.length; i++){ String str = "this is"+" "+"a test with value = "+i; }
```

JavaDoc Comments

It's pretty simple, **all** attributes and methods need javadoc comment.(Even if they are private/protected).

If a method throw an exception, the javadoc contains the @throws tag for each exception thrown.

If a method return something, the javadoc contains the @return element.

Example :

```
/** * This is a sample attribute used for JOnAS code convention wiki */ private String simpleExample = null;
```

```
/** * This method is a sample example with a parameter and a return value and an exception * @param dummyArg an example of parameter * @return an object * @throws Exception to illustrate the example */ private Object myMethod(String dummyArg) throws Exception { return null; }
```

Statements

## ***if/else***

When using if else blocks, even if there is a single statement, the braces have to be used :

# JOnAS OpenSource Java EE Application Server - CookBooks - Code Convention

```
if (true) { doThis(); }
```

and not

```
if (true) doThis();
```

Note that the braces follow the previous example. Don't use:

```
if (true) { test1(); test2(); }
```

## ***try/catch***

All exception required a statement, no silent catching like :

```
try { doThis(); } catch (Exception e) { // should not occur }
```

You can use a logger :

```
try { doThis(); } catch (Exception e) { logger.logDebug("Exception while doing .....", e); }
```

## ***Inline Conditionals***

Don't use inline conditionals like

```
b = isOk() ? true : false;
```

but

```
if (isOk()) { b = true; } else { b = false; }
```

Naming conventions

## ***Attributes***

Static final attributes Following the JLS suggestions, the declaration is "static final" and not "final static". Constants should match the pattern

```
'^[A-Z][A-Z0-9]*(_[A-Z0-9]+)*$'
```

Also they need to be declared static and final. No magic number, use constants ie : Don't do :

```
private int myAttribute = 5;
```

but :

```
/** * Default value */ private static final int DEFAULT_VALUE = 5; /** * This attribute is initialized with the default value */ private int myAttribute = DEFAULT_VALUE;
```

Attributes name

no \_ (underscore) is authorized in the name of the attributes. \_ character is used only in constants (UPPERCASE).

so avoid m\_value or p\_value, use instead mValue or pValue.

The regexp is the following :

```
^[a-z][a-zA-Z0-9]*$.
```

Tools

By using eclipse IDE, there are other good checks like :

## JOnAS OpenSource Java EE Application Server - CookBooks - Code Convention

- unnecessary cast or instance of
- unused variable
- methods which should be used in a static way
- unused methods
- unnecessary throws
- etc.

[Entry Point](#) (en)

Creator: xwiki:XWiki.sauthieg Date: 2008/03/25 07:56

Last Author: xwiki:XWiki.sauthieg Date: 2008/03/25 08:07

Copyright (c) 2006, [ObjectWeb Consortium](#)