# Issues

## Issues found by applying the checklist

## Naming conventions

## Meaningful names

getCompEnvBinding OK

create OK

Other :

- L714 variable ic ← should be more explicit, like ctx

## One character variable

getCompEnvBinding OK

create OK

## Class names

getCompEnvBinding OK

create OK

## Interface names

getCompEnvBinding OK

create OK

## Method names

getCompEnvBinding OK

create OK

Other :

- L736, L744 dependencyAppliesToScope ← not a verbal form

- L774 descriptorToLogicalJndiName ← not a verbal form

- each occurrence of JNDI should be written in the same way, or JNDI or Jndi

- L458 undepoyResource ← should be written undeployResource

## 

## Attributes names

getCompEnvBinding OK

create OK

Other :

- L940 wsRedMgr ← should be written wSRefMgr, wSrvRefMgr or webSrvRefMgr

- L143 refcnt ← should be written refCnt

## Constants names

getCompEnvBinding OK

create OK

## Indention

## Number of spaces for indentation and consistency

getCompEnvBinding OK

create OK

## No use of tab

getCompEnvBinding OK

create OK

## Braces

## Consistency braces style

getCompEnvBinding OK

create OK

→ Kernighan and Ritchie

## Curly braces even for one-statement bloc

getCompEnvBinding OK

create OK

## File Organization

## Blank lines and optional comments

getCompEnvBinding OK

create OK

## Line does not exceed 80 characters

getCompEnvBinding

- lines that should have been broken after a comma/operator : L696,700,702,709

- lines impossible to break : L686

create OK

## Lines do not in any case exceed 120 characters

getCompEnvBinding OK

create OK

## Wrapping Lines

## Line break occur after a comma or an operator

getCompEnvBinding OK

create

- line short enough to not have been broken : L874

## Higher level breaks are used

getCompEnvBinding OK

create OK

## Position of the new statements

getCompEnvBinding

- indentation not justified : L688

- miss an indentation : L692

create OK

## Comments

## Used to explain what the related code part is doing

getCompEnvBinding OK

create

-L883 → not really clear

## An obsolete date for the comment should appear and the commented out code has to contains a reason for being commented out

getCompEnvBinding OK but no date

create OK but no date

## Java Source Files

## Each Java source file contains a single public class or interface

OK → Several classes but only one is public, no interfaces

## The public class is the first class or interface in the file

OK

## The external program interfaces are implemented consistently with what it is described in the javadoc

Three different interfaces are used in this file. They are implemented consistently with the following classes :

- ComponentEnvMangager← ComponentEnvManagerImpl **OK**

- JNDIBinding ← CompEnvBinding **OK**

**-** NamingObjectProxy ← EjbReferenceProxy, WebServiceRefProxy, ValidatorFactoryProxy, ValidatorProxy, EjbContextProxy **Ok**

*NamingObjectProxy* is also implemented with the class *FactoryForEntityManagerWrapper* but the method *create* doesn't throw any exception in this class whereas it has to do it according to the javadoc of the interface.

## The javadoc is complete

The Javadoc is not complete since it miss three public methods related to the class *ComponentEnvManagerImpl :*

*addToComponentNamespace(JndiNameEnvironment origEnv, Collection<EnvironmentProperty> envProps, Collection<ResourceReferenceDescriptor> resRefs) : void*

*getComponentEnvId(JndiNameEnvironment env) : String*

*getCurrentApplicationEnvironment() : ApplicationEnvironment*

Moreover, it miss all the methods explanation.

## Package and Import Statements

## Package statements position (before import statements)

OK

## Class and Interface Declarations

## Order of class and interface declarations

L106, 109 : package instance variable should be put before the private ones

This class doesn't have any constructor

L137 : One secondary private class is described before the primary class methods. It should be put at the end of the file.

## Methods are grouped by functionality rather than by scope or accessibility

1) getResource and getResourceDeployer should be next to getCompEnvBinding

2) addEnvironmentProperties; addResourceReferences; addJNDIBindings should be put after addToComponentNamespace

3) getCurrentApplicationEnvironment should be put next to getJndiNameEnvironment; getCurrentJndiNameEnvironment

4) dependencyAppliesToScope should be put after descriptorToLogicalJndiName; getComponentEnvId; getCurrentApplicationEnvironment

We would obtain the following structure

1) getJndiNameEnvironment; getCurrentJndiNameEnvironment

2) bindToComponentNamespace; addToComponentNamespace; addEnvironmentProperties; addResourceReferences; addJNDIBindings; addAllDescriptorBindings

3) unbindFromComponentNamespace; undeployAllDescriptors; undepoyResource

4) getResourceId; getResourceDeployer; getCompEnvBinding; getCompEnvBinding; descriptorToLogicalJndiName; getComponentEnvId; getCurrentApplicationEnvironment

5) dependencyAppliesToScope; dependencyAppliesToScope

## Long methods, big classes, encapsulation, cohesion, coupling

Seems OK

longest method : 128 lines

biggest secondary class : 53 lines

## Initialization and Declarations

## Variables and class members types and their visibility

getCompEnvBinding OK

create OK

## Variables are declared in the proper scope

getCompEnvBinding OK

create OK

## Constructors are called when a new object is desired

getCompEnvBinding OK

create OK

NB : Factories are often used in place of basic constructors

## All object references are initialized before use

getCompEnvBinding OK

create OK

## Variables are initialized where they are declared, unless dependent upon a computation

getCompEnvBinding OK

create OK

## Declarations appear at the beginning of blocks

getCompEnvBinding OK

create OK

## Method Calls

## Parameters are presented in the correct order

getCompEnvBinding OK

create OK

## The correct methods are being called

getCompEnvBinding OK

create OK

## Methods returned values are used properly

getCompEnvBinding OK

create OK

## Arrays

getCompEnvBinding : No arrays are used in this method

create : No arrays are used in this method

## Object Comparison

getCompEnvBinding OK

create

- comparison with “==” in place of equals : L881, 884, 896

- comparison with “!=” in place of not equals : L890

## Output Format

## Spelling and grammatical errors

getCompEnvBinding OK (no displayed output)

create OK

## Error messages are comprehensive and helpful

getCompEnvBinding OK (no displayed output)

create OK

## Output format

getCompEnvBinding OK (no displayed output)

create OK

## Computation, Comparisons and Assignments

## Brutish programming

getCompEnvBinding

A switch statement should be used in place of several if/else statements

create

OK

## Order of computation/evaluation, operator precedence and parenthesizing

getCompEnvBinding OK

create OK

## Use of parenthesis in operations

getCompEnvBinding OK (no operations)

create OK (no operations)

## Denominators can't be equal to zero

getCompEnvBinding OK (no denominators)

create OK (no denominators)

## Proper use of integers to avoid truncation issues

getCompEnvBinding OK (no numbers used)

create OK (no numbers used)

## Comparison and Boolean operators are correct

getCompEnvBinding OK

create OK

## Throw-catch expression are proper used

getCompEnvBinding OK

create OK

## Code free of any implicit type conversion

getCompEnvBinding OK

create OK

## Exceptions

## Relevant exceptions are caught

getCompEnvBinding OK (no exception caught)

create OK (no exception caught)

## Appropriate action are taken for each catch block

getCompEnvBinding OK (no catch block)

create OK (no catch block)

## Flow of Control

GetCompEnvBinding : No switch neither loop are used in this method

create : No switch neither loop are used in this method

## Files

GetCompEnvBinding : No files are open or closed in this method

create : No files are open or closed in this method