Code Inspection Document for GlassFish 4.1.1 Server

Petar Korda

Ranjithkumar Krishnan

Politecnico di Milano, 23.12.2015

Contents

[Classes Assigned 3](#_Toc438475278)

[Functional Role of the Classes Assigned 3](#_Toc438475279)

[List of Issues 4](#_Toc438475280)

# Classes Assigned

Class assigned for inspection: *Class* ***ActiveJmsResourceAdapter***

Namespace: ***com.sun.enterprise.connectors.jms.system***

Details:

[*java.lang.Object*](http://docs.oracle.com/javase/7/docs/api/java/lang/Object.html?is-external=true)

[*com.sun.enterprise.connectors.ActiveResourceAdapterImpl*](http://glassfish.pompel.me/com/sun/enterprise/connectors/ActiveResourceAdapterImpl.html)

[*com.sun.enterprise.connectors.ActiveOutboundResourceAdapter*](http://glassfish.pompel.me/com/sun/enterprise/connectors/ActiveOutboundResourceAdapter.html)

[*com.sun.enterprise.connectors.inbound.ActiveInboundResourceAdapterImpl*](http://glassfish.pompel.me/com/sun/enterprise/connectors/inbound/ActiveInboundResourceAdapterImpl.html)

*com.sun.enterprise.connectors.jms.system.ActiveJmsResourceAdapter*

# Functional Role of the Classes Assigned

*ActiveJmsResourceAdapter* represents an active JMS resource adapter, which does additional configurations to *ManagedConnectionFactory* and *ResourceAdapter* java beans. It is part of the JMS module of GlassFish server.

The GlassFish Server support for JMS messaging, in general, and for message-driven beans, in particular, requires messaging middleware that implements the JMS specification: a JMS provider. The GlassFish Server uses the GlassFish Message Queue software as its native JMS provider. The Message Queue software is tightly integrated into theGlassFish Server, providing transparent JMS messaging support. This support is known within GlassFish Server as the JMS Service (Oracle GlassFish guide: https://docs.oracle.com/cd/E19798-01/821-1752/beaob/index.html).

GlassFish server must use a resource adapter furnished by a given JMS provider to use the functionality of that provider. Message Queue provides such a resource adapter. Using the support of a plugged in JMS provider, Java EE components, including MDBs, deployed and running in the application server environment can exchange JMS messages among themselves and with external JMS components. This provides a powerful integration capability for distributed components.

Message Queue JMS Resource Adapter (JMS RA),enables you to integrate GlassFish Message Queue with any J2EE 1.4 application server by means of the standard J2EE connector architecture (JCA). When plugged into an application server, the Resource Adapter allows applications deployed in that application server to use Message Queue to send and receive JMS messages.

The Message Queue JMS Resource Adapter exposes its configuration properties through three JavaBean components:

The ResourceAdapter JavaBean ([ResourceAdapter JavaBean](https://docs.oracle.com/cd/E19798-01/821-1794/aeooo/index.html)) affects the behavior of the Resource Adapter as a whole.

The ManagedConnectionFactory JavaBean ([ManagedConnectionFactory JavaBean](https://docs.oracle.com/cd/E19798-01/821-1794/aeoop/index.html)) affects connections created by the Resource Adapter for use by message-driven beans (MDBs).

The ActivationSpec JavaBean ([ActivationSpec JavaBean](https://docs.oracle.com/cd/E19798-01/821-1794/aeooq/index.html)) affects message endpoints that represent MDBs in their interactions with the messaging system.

The ***ResourceAdapter*** configuration configures the default JMS Resource Adapter behavior.

A ***ManagedConnectionFactory*** defines the connections that the ResourceAdapter provides to a message-driven bean.

***ActiveJmsResourceAdapter***does additional configurations to *ManagedConnectionFactory* and *ResourceAdapter* java beans.

# List of Issues

In this section the list of issues found with the code by applying the checklist is given in order of the checklists modules. Only the points from the checklist that were found as issues are given (if a point of the checklist is skipped that mean no issue in the code for that point was found).

|  |  |  |  |
| --- | --- | --- | --- |
| **NAMING CONVENTIONS** |  |  |  |
| **Checklist point** | **Issue** | **Line** | **Details** |
| 1 - All class names, interface names, method names, class variables, method variables, and constants used should have meaningful names and do what the name suggests | Class Variable: sm | 250 | This is a StringManager but it is not really easy to conclude this from the name of the variable |
| 1 | Class Variable: nm | 306 | This is aNameManager but it is not really easy to conclude this from the name of the variable |
| 1 | Methode: handles | 1161 | Ambiguous, maybe a better name would be doesHandle |
| 1 | Methode: setProperty | 1833 | Not indicative enough, this sets the property of ManagedConnectionFactory, but can be missunderstood as setting the property of ResourceAdapter |
| 7 - Constants are declared using all uppercase with words separated by an underscore | RECONNECTENABLED, RECONNECTINTERVAL, RECONNECTATTEMPTS, GROUPNAME, CLUSTERCONTAINER | 166 - 170 | Not following the convention |
| 7 | BROKERTYPE, BROKERINSTANCENAME, BROKERBINDADDRESS, BROKERPORT, BROKERARGS, BROKERHOMEDIR, BROKERLIBDIR, BROKERVARDIR, BROKERJAVADIR, BROKERSTARTTIMEOUT, ADMINUSERNAME, ADMINPASSWORD, ADMINPASSFILE | 173 - 185 | Not following the convention |
| 7 | JMXSERVICEURL, JMXSERVICEURLLIST, JMXCONNECTORENV, USEJNDIRMISERVICEURL, RMIREGISTRYPORT, USEEXTERNALRMIREGISTRY, DEFAULTRMIREGISTRYPORT, BROKERRMIPORTOFFSET, SSLJMXCONNECTOR | 192-202 | Not following the convention |
| 7 | PINGINTERVAL, DBTYPE, DBTYPE\_HADB, BROKERENABLEHA | 210-213 | Not following the convention |
| 7 | MAXPOOLSIZE, MINPOOLSIZE, RESIZECOUNT, RESIZETIMEOUT, REDELIVERYCOUNT, LOWERCASE\_REDELIVERYCOUNT | 233 - 238 | Not following the convention |

|  |  |  |  |
| --- | --- | --- | --- |
| **INDENTATION** |  |  |  |
| **Checklist point** | **Issue** | **Line** | **Details** |
| 8 - Three or four spaces are used for indentation and done so consistently | No indentation | 454 | Line not indented |

|  |  |  |  |
| --- | --- | --- | --- |
| **BRACES** |  |  |  |
| **Checklist point** | **Issue** | **Line** | **Details** |
| 9 - Consistent bracing style is used, either the preferred “Allman” style or the “Kernighan and Ritchie” style | Kernighan and Ritchie style is used |  | Allman style is prefered |
| 9 | Not consistent with K&R style | 548 | The brace should be on the same line as the if statement |

|  |  |  |  |
| --- | --- | --- | --- |
| **FILE ORGANIZATION** |  |  |  |
| **Checklist point** | **Issue** | **Line** | **Details** |
| 12 - Blank lines and optional comments are used to separate sections | Missing blank line | 524 | No blank line to divide comment |

|  |  |  |  |
| --- | --- | --- | --- |
| **WRAPPING LINES** |  |  |  |
| **Checklist point** | **Issue** | **Line** | **Details** |
| 12 - Blank lines and optional comments are used to separate sections | New line after brace | 545 | Line break does not occur after a comma or an operator but a brace |
| 12 | New line after brace | 546 | Line break does not occur after a comma or an operator but a brace |

|  |  |  |  |
| --- | --- | --- | --- |
| **COMMENTS** |  |  |  |
| **Checklist point** | **Issue** | **Line** | **Details** |
| 19 - Commented out code contains a reason for being commented out and a date it can be removed from the source file if determined it is no longer needed | Unexplained code comment | 524 | A reason for commenting a code is not given |

|  |  |  |  |
| --- | --- | --- | --- |
| **JAVA SOURCE FILE** |  |  |  |
| **Checklist point** | **Issue** | **Line** | **Details** |
| 23 - Check that the javadoc is complete | Method: isEjbInWar | 2315 | No method description in javadoc |

|  |  |  |  |
| --- | --- | --- | --- |
| **CLASS AND INTERFACE DECLARATIONS** |  |  |  |
| **Checklist point** | **Issue** | **Line** | **Details** |
| 25 | static variables declarations | ~156 | Not defined in the correct order, there are some public variables declared before private, also there are some instance variable defined before the static ones. They are however arranged in the block of logical modules (ex. RA Javabean properties line 164 - 170. However first the public constant was defined, then a private ones which is wrong) |

|  |  |  |  |
| --- | --- | --- | --- |
| **INITIALIZATION AND DECLARATION** |  |  |  |
| **Checklist point** | **Issue** | **Line** | **Details** |
| 33 - Declarations appear at the beginning of blocks | Local Var: dummy | 463 | Declaration of dummy local variable not appearing at the beginning of the block |
| 33 | Local Var: name | 469 | Declaration of name local variable not appearing at the beginging of the block |
| 33 | Local Var: mergedProps | 528 | Declaration of mergedProps local variable not appearing at the beginning of the block |
| 33 | Local Var: brokerType | 529 | Declaration of brokerType local variable not appearing at the beginning of the block |
| 33 | Local Var: cluster | 537 | Declaration of cluster local variable not appearing at the beginning of the block |

|  |  |  |  |
| --- | --- | --- | --- |
| **OUTPUT FORMAT** |  |  |  |
| **Checklist point** | **Issue** | **Line** | **Details** |
| 42 - Check that error messages are comprehensive and provide guidance as to how to correct the problem | Error message not comprehensive | 541 | No error message is printed but only the stack trace |