Oracle9*i*AS Containers for J2EE

Release Notes

Release 2 (9.0.2)

April 2002

Part No. A97300-01

This document summarizes the differences between Oracle9*i*AS Containers for J2EE (OC4J) and its documented functionality. It covers the following topics:

- Release Notes for J2EE Connector Architecture
- Release Notes for Oracle9iAS EJB Container
- Release Notes for Oracle SQLJ
- Release Notes for Oracle JDBC
- Release Notes for Oracle9iAS Servlet Container
- Release Notes for Oracle9iAS JSP Container
- Release Notes for JAZN
- Release Notes for OC4J Administration and Management

See Also: *Oracle9i Application Server Release Notes*

1 Release Notes for J2EE Connector Architecture

This section includes issues with J2EE Connector Architecture that are not reflected in the Oracle9*i*AS 9.0.2 documentation.

1.1 IllegalArgumentException

A bug causes an `IllegalArgumentException`; an example is:

```
java.lang.IllegalArgumentException: No such property: ConnectionURL,
existing writable properties are: [logWriter, connectionURL]
```

Such an exception may occur if the resource adapter deployment descriptor specifies any configuration property whose `<config-property>` element `name` attribute setting begins with a capital letter when the second character is not a capital. This includes the standard properties defined in section 10.4.3 in Sun Microsystems' J2EE Platform Connector Architecture 1.0 Specifications, such as `ConnectionURL`. This is in conflict with sections 8.3.1 and 8.8 of Sun's JavaBeans 1.01 Specifications, which states that a JavaBean with methods `setConnectionURL()` and `getConnectionURL()` should have a property name `connectionURL`, not `ConnectionURL`.

1.1.1 Workaround

The workaround here is to change the `<config-property>` element `name` attribute setting in the resource adapter's `ra.xml` file from `ConnectionURL` to `connectionURL`. The `ra.xml` file is found in the

`$J2EE_HOME/`*`connectors/connector-name/rar-file-name/`*
`META-INF`

directory if the resource adapter is deployed standalone, or

`$J2EE_HOME/applications/`*`app-name/rar-file-name/`*`META-INF`

directory if the resource adapter is packaged within an EAR file. (`$J2EE_HOME` is `j2ee/home`.) If the resource adapter has been deployed, also change the value of the `name` setting in the `<config-property>` element in the generated `oc4j-ra.xml` file under the `application-deployment` directory. For example, change the following `<config-property>` element from:

```
<config-property name="ConnectionURL"
value="jdbc:cloudscape:rmi:CloudscapeDB;create=true"/>
```

to:

```
<config-property name="connectionURL"
value="jdbc:cloudscape:rmi:CloudscapeDB;create=true"/>
```

and restart OC4J.

1.2 deployconnector Switch Not Available in this Release

The `-deployconnector` switch in the `admin` command line tool (`admin.jar`) documented in *Oracle9iAS Containers for J2EE Services Guide* is not available for deploying standalone resource adapters.

1.2.1 Workaround

Standalone resource adapters can be deployed manually. Follow these steps:

1. Create a new directory under the `$J2EE_HOME/connectors` directory.
2. Copy the resource adapter RAR file into the new directory.
3. Add the following to the `$J2EE_HOME/config/oc4j-connectors.xml` file:

   ```
   <connector name="your_resource_adapter_name"
   path="your_resource_adapter.rar"> </connector>
   ```

4. In the `$J2EE_HOME/config/server.xml` file, make sure that the `connector-directory` attribute is specified in the `<application-server>` element as follows:

   ```
   <application-server
      application-directory="../applications"
      deployment-directory="../application-deployments"
      connector-directory="../connectors">
   ```

In the `$J2EE_HOME/config/application.xml` file, if there is no `<connectors>` element under `<orion-application>` that looks like this, add it:

```
<connectors path="./oc4j-connectors.xml"/>
```

In these first four steps, you have deployed the standalone resource adapter to OC4J.

5. Start or restart OC4J process. OC4J will automatically unpack your RAR file in the

   `$J2EE_HOME/connectors/`*`your-directory-name/`*
   `your-resource-adapter-name/`

   directory.

   In step 5, OC4J created a directory called *`your_resource_adapter_name`* in `$J2EE_HOME/application-deployments/default/` when you started the OC4J process.

6. Configure the `oc4j-ra.xml` file under the

   `$J2EE_HOME/application-deployments/default/`
   `your_resource_adapter_name`

directory with the desired connector property settings, each with its distinct JNDI name for look-up from application components, and, optionally, with different configuration property values. Here is an example of an `oc4j-ra.xml` file:

```
<oc4j-connector-factories>
 <connector-factory location="eis/eisJNDIforCloudscape"
   connector-name="BlackBoxNoTx">
     <config-property name="connectionURL"
         value="jdbc:cloudscape:rmi:CloudscapeDB;create=true"/>
 </connector-factory>
 <connector-factory location="eis/eisJNDIforOracle"
   connector-name="BlackBoxNoTx">
     <config-property name="connectionURL"
       value="jdbc:oracle:thin:@localhost:1521:orcl"/>
 </connector-factory>
</oc4j-connector-factories>
```

Restart the OC4J process again for the configuration to take effect.

1.3 native-library Element Problem

The `<native-library>` element under `<connector>` in `oc4j-connectors.xml` does not work.

1.3.1 Workaround

Copy the native libraries, such as `.so` or `.dll` files, if any, to the top-level directory in which the resource adapter files are expanded when your application is deployed into OC4J. For example, suppose a standalone resource adapter named `myRAfile.rar` is deployed into the

```
$J2EE_HOME/connectors/myRAname
```

directory. The top-level directory would be

```
$J2EE_HOME/connectors/myRAname/myRAfile
```

Copy any native libraries that are packaged within that RAR file to this directory.

2 Release Notes for Oracle9*i*AS EJB Container

This section includes issues with EJB that are not reflected in the Oracle9*i*AS 9.0.2 documentation.

- For read-only entity beans, the default for `exclusive-write-access` is set to `true`.

- OC4J does not support using both emulated and non-emulated `OrionCMTDataSource` data sources for database operations in one transaction. Having multiple non-emulated `OrionCMTDataSource` data sources result in a two-phase commit operation.
- The property `cacheScheme` for non-emulated data sources takes integer values, not `String`, as indicated in *Oracle9iAS Containers for J2EE Enterprise JavaBeans Developer's Guide and Reference*.

  The correspondence is as follows:

  ```
  DYNAMIC_SCHEME = 1
  FIXED_WAIT_SCHEME = 2
  FIXED_RETURN_NULL_SCHEME = 3
  ```

  The following example sets the cache scheme to `FIXED_WAIT_SCHEME`:

  ```
  <data-source
      class="com.evermind.sql.OrionCMTDataSource"
      name="OracleDS"
      location="jdbc/OracleCMTDS1"
      connection-driver="oracle.jdbc.driver.OracleDriver"
      username="scott"
      password="tiger"
      url="jdbc:oracle:thin:@localhost:5521:derdbms"
      inactivity-timeout="30"
      max-connections="2">
    <property   name="cacheScheme" value="2"/>
  </data-source>
  ```

- `wsdl2ejb` demos:

  The shipped `build.xml ant` script generates and deploys EJBs from sample WSDL files. The deployment is performed using OC4J's `admin.jar` file. This technique works for the OC4J standalone package, but fails under a full Oracle9*i*AS installation. To execute the demo in an Oracle9*i*AS environment, run the `ant` EJB generation target (`rpc_dog_gen` or `interop_gen`), deploy the generated EAR file using recommended Oracle9*i*AS tools, and then run the EJB client target.
- OC4J exposes two permissions:
  - the RMI Login permission (`com.evermind.server.rmi.RMIPermission`)
  - the Administration permission (`com.evermind.server.AdministrationPermission`)

Both of these permissions are automatically granted to a group. EJB clients must have the RMI permission assigned to themselves before accessing an EJB.

See the *Oracle9iAS Containers for J2EE Services Guide* for information on how to assign permissions using the JAZN-XML or JAZN-LDAP providers.

- If you specify `max-connection-attempts` in `data-sources.xml`, then you must also specify `connection-retry-interval` in `data-sources.xml`, or else there will be a null pointer exception (bug 2282743).
- In `data-sources.xml`, you can specify a minimum number of connections. However, emulated data sources do *not* support a setting for minimum number of connections.
- If you use a message-driven bean (MDB) with Oracle Java Messaging Service (JMS), the MDB must be configured as a "durable subscriber." Oracle JMS supports only durable subscription in release 9.0.2 (bug 2237811).
- There is a functional difference when using the JNDI property `dedicated.rmicontext` instead of `dedicated.connection`. In either case, whenever you create a new `InitialContext` instance, a new RMI context is created. With `dedicated.rmicontext`, these RMI contexts all share the same RMI connection. With `dedicated.connection`, the RMI contexts do not share the same RMI connection.
- To avoid wrapper cache problems, verify that the `disable-wrapper-cache` attribute is set to `true` (the default) in the `orion-ejb-jar.xml` configuration file.

3 Release Notes for Oracle SQLJ

This section includes issues with SQLJ that are not reflected in the Oracle9*i*AS 9.0.2 documentation.

- The demo for SQLJ-specific connection support is called `bmp`.
- To set up SQLJ-specific data sources, follow the demo instructions.
- The SQLJ-specific OJSP connection beans are not distributed as part of `runtime12ee.jar`:

```
oracle.sqlj.runtime.SqljConnBean
oracle.sqlj.runtime.SqljConnCacheBean
```

Instead, they are provided with `ojsputil.jar`, which also contains the other OJSP connection bean classes. This library is located at:

```
$ORACLE_HOME/jsp/lib/ojsputil.jar
```

To use the SQLJ-specific OJSP connection beans in Oracle9*i*AS, ensure that `ojsputil.jar` is either directly included in `server.xml` or in a path specified in `server.xml`.

For example, the following entry in `server.xml` makes SQLJ-specific connection beans available to Oracle9*i*AS, assuming that `$ORACLE_HOME` has been set to `/iasv2`:

```
<library path = "/iasv2/jsp/lib">
```

4 Release Notes for Oracle JDBC

This section includes issues with JDBC that are not reflected in the Oracle9*i*AS 9.0.2 documentation.

The JDBC drivers shipped with this Oracle9*i*AS version have known problems. The supplemental CD contains a JDBC patch (with patch instruction). Follow the instructions in the *Oracle9iAS Installation Guide* to install the necessary patches before you run Oracle9*i*AS. Also, read the accompanying release note, which includes the list of known problems.

5 Release Notes for Oracle9*i*AS Servlet Container

This section includes issues with servlets that are not reflected in the Oracle9*i*AS 9.0.2 documentation.

For security reasons, OC4J uses the class `java.security.SecureRandom` for secure seed generation. Session-based requests use this facility. Unfortunately, the amount of time required for the first instantiation to complete can be unacceptable, depending upon your application needs. Since OC4J makes this call lazily, it can cause an unexpected delay when it is first called during the course of application execution. If this occurs, one solution is for an application to enable the `load-on-startup` attribute in the `<web-site>` element of the `web-site.xml` configuration file and to create an instance of `SecureRandom` during the class initialization of the application. The result will be a longer startup time in place of a delay during the course of servicing clients.

6 Release Notes for Oracle9*i*AS JSP Container

This section includes issues with JSP that are not reflected in the Oracle9*i*AS 9.0.2 documentation.

Note: Starting with Oracle9*i*AS 9.0.2, components that ship with Oracle9*i*AS use the same version numbering. The major change in the Oracle9*i*AS JSP (OJSP) container in release 9.0.2.0 is better integration with the other Oracle9*i*AS containers for J2EE.

Note: OJSP demos are located in `ojspdemos.ear` in the J2EE demo instance of a regular Oracle9*i*AS 9.0 installation. They are not available with the `oc4j/j2ee` basic OTN download.

6.1 General Notes

- Starting with the 9.0.2.0 release, the default JSP engine is the Oracle9*i*AS release 2 version. The JSP engine is configured in `global-web-application.xml`. However, some JSP-related attributes in the Orion configuration files, such as `development` in `global-web-application.xml`, are not applicable.
- For `page` scope, a new `check_page_scope` parameter has been introduced. Users can set this parameter to `true` to enable `page` scope checking by the `JspScopeListener` utility for OC4J environments. It would be `false` by default, for performance reasons, but is set to `true` in your predefined `global-web-application.xml` file.
- You can use the `location` or `ejb-location` element (but not the deprecated element `pooled-location` as mentioned in *Oracle9iAS Containers for J2EE Support for JavaServer Pages Reference*).

6.2 Security Considerations

Follow these security practices:

- On Oracle9*i*AS running JServ, we highly recommend that Web access to the generated `_pages` directory be denied. On Oracle9*i*AS 9.0.2, access is denied in the default `_pages` directory. However, if you are using aliases, be sure to deny access to any `_pages` directory generated under each alias.

- On Oracle9*i*AS running JServ, we highly recommend that Web access to `globals.jsa` be denied. On Oracle9*i*AS 9.0.2, such access is denied by default.
- For applications using SQL tags, consider using the `dbSetParam` tag to supply only parameter values rather than textual completion of the SQL statement itself. This avoids “SQL poisoning,” which is the possibility of users entering additional SQL along with the expected value.
- You can suppress the display of the physical file path when nonexistent JSP files are requested, by setting the `debug_mode` parameter to `false`.

6.3 Known Issues and Restrictions

- In this release, you cannot use the JESI template/fragment model and explicit ESI markup of the form `<esi:inline>` within the same HTTP response. For example, there will be Web Cache errors if you use a JSP page with `<jesi:template>` and `<jesi:fragment>` tags, and the page includes a servlet that generates HTML with `<esi:inline>` tags in it.
- Desupport of the pre-1.1 JSP tag mechanism (bug 2125027). Prior to the JSP 1.1 support of tag libraries, OJSP supported its own compile-time mechanism for using custom code. This entailed using `uri="oracle.jsp.parser.OpenJspRegisterLib"` in the `taglib` directive. Now that 1.1 fully supports custom tag libraries, we intend to desupport this mechanism in favor of the standard tag library mechanism.
- Aliases and JSP (bug 2189308). When using JServ alias directives in combination with JSPs, there are issues when two aliases begin with the same partial directory path. Consider the following two aliases as an example:

  ```
  Alias /foo/bar1 "/path/to/my/dir/x/bar1"
  Alias /foo/bar2 "/path/to/my/dir/y/bar2"
  ```

  An initial request for `/foo/bar1/bar1.jsp` will work, but a subsequent request for `/foo/bar2/bar2.jsp` will incorrectly look in `/path/to/my/dir/x for bar2.jsp`, and will fail with a `FileNotFound` exception. This is due to further limitations with the JServ `getRealPath()` implementation, which returns incorrect information. There are two workarounds for this situation:

  - Have only one alias, with real directories underneath:

    ```
    Alias /foo  "/path/to/my/dir"
    ```

Here the bar1 and bar2 directories would physically exist as /path/to/my/dir/bar1 and /path/to/my/dir/bar2, and there would not be a problem.

or:

- Have more than one alias, but arrange it so that the physical directories do not have the same names as the alias directories:

```
Alias /foo/bar1  "/path/to/my/dir/x_bar1"
Alias /foo/bar2  "/path/to/my/dir/y_bar2"
```

  Note the use of x_bar1 instead of bar1 and y_bar2 instead of bar2. In the problematic example earlier, the first alias used bar1, which is the same as the directory name, and the second alias used bar2, which is the same as the directory name.

- On Windows NT, the ojspc translator tool does not support wildcards in file lists. Wildcards will work on UNIX shells, as the shell expands them.
- The database access beans do not support any classes from the oracle.jdbc2 package. This is to be consistent with different JDK versions.
- Not specifying the included page in a JSP include statement results in StringIndexOutOfBoundsException (bug 1234581). For example, the following directive:

```
<jsp:include page="" flush="true" />
```

  would result in the following error:

```
java.lang.StringIndexOutOfBoundsException: String index out of
range: Provide a non-empty string for the page attribute.
```

- Display of null values in JSP.

  In Oracle9*i*AS, a null value printed from a JSP page displays, by default, as the string “null.” To display nothing instead, set the attribute jsp-print-null to false in the <web-app> element of global-web-app.xml or orion-web.xml.

7 Release Notes for JAZN

This section includes issues with JAZN that are not reflected in the Oracle9*i*AS 9.0.2 documentation.

Note: JAZN is the Oracle9*i*AS implementation of the JAAS (Java Authentication and Authorization Service) standard.

7.1 Admintool Changes

The JAZN Admintool now enforces authentication and authorization for most of the JAZN commands, including the JAZN shell. There are two ways to specify the user name and password for authentication purposes:

- You can specify the user name and password with the `-user` and `-password` switches.

  This option is considered insecure as the password is specified in clear text.

- You can enter the credentials information interactively when prompted by the Admintool.

  The Admintool obfuscates the password as you type it in. Unfortunately, due to limitations with the JDK I/O library, the mechanism sometimes does not fully obfuscate your password on the screen. Note that authentication is not required for the `-checkpasswd` and `-setpasswd` commands, and when JAZN-LDAP is the specified provider.

7.2 JAZNUserManager Delegation Support

`JAZNUserManager` now supports the OC4J “user manager delegation” model. If a user or group is not found at the application level `JAZNUserManager` instance, it delegates the request to the global user manager.

A known limitation is that delegation between `principals.xml`, which is the storage for `XMLUserManager`) and `JAZNUserManager` is not supported. For example, a configuration that sets `principals.xml` as the global user manager and `JAZNUserManager` as the application level user manager is not supported. (`JAZNUserManager` is the implementation class; it can be configured to use an XML file, `jazn-data.xml`, as storage or OID as storage.) This feature should be distinguished from the “identity delegation” feature discussed in the *Oracle9iAS Containers for J2EE Services Guide*. The “identity delegation” feature refers to the fact that when a servlet calls an enterprise bean on behalf of a client's request, the primary caller's identity is propagated to the enterprise bean for authorization purposes.

7.3 JAZN Clustering Support

JAZN-XML is integrated with DCM / SMI (System Management Interface, an API that EM uses to manage OC4J and OC4J applications) to provide cluster support. Any changes to `jazn-data.xml` via EM will be automatically propagated to all nodes participating in the same cluster.

However, be aware that any modification of `jazn-data.xml` will not be instantly picked up by the running OC4J instances. An OC4J instance needs to be restarted for the changes to take effect.

7.4 OC4J Services Guide, Chapter 5

The description regarding our demo application, `callerInfo`, is out of date. Refer to the file `README.txt` located at `$ORACLE_HOME/j2ee/home/jazn/demo/callerInfo` for a more up-to-date description of this JAZN demo.

7.5 JAAS and Java 2 Security

Oracle9*i*AS 9.0.2 does not support using the JAAS provider as the J2SE policy (the Java 2 security policy) provider. For code-based security, we recommend using the J2SE 1.3.1 reference implementation. We provide a J2SE policy file that works with the J2SE 1.3.1 reference implementation. This file is located at `$J2EE_HOME/config/java2.policy`.

7.5.1 How to Enable an Application with Java 2 Security

To enable an application with Java 2 security, do one of the following:

- You can start up any standard compliant JVM (Java Virtual Machine) with Java 2 security enabled by defining the system property `java.security.manager` (and, optionally, `java.security.policy`).

  For example, you can start up a JVM with Java 2 security enabled by the following command:

  ```
  > java -Djava.security.manager -Djava.security.policy=
  $OH/j2ee/home/config/java2.policy ...
  ```

- Alternatively, you can enable Java 2 security programmatically, enabling the security manager through the `System.setSecurityManager()` API.

7.5.2 How to Enable OC4J with Java 2 Security

OPMN (Oracle Process Management Notification) supports specification of Java options in `opmn.xml`. The following `opmn.xml` fragment illustrates how to enable OC4J for Java 2 security in an ADE view:

```
<oc4j instanceName="home" numProcs="1" maxRetry="3">
  <config-file path=
    "/ade/rkng_oc4j902/oracle/j2ee/home/config/server.xml" />
  <java-bin path="/usr/local/packages/jdk1.3.1/bin/java" />
  <java-option
    value="-Djava.security.manager
           -Djava.security.policy=/
             ade/rkng_oc4j902/j2ee/home/config/java2.policy"/>
  <port ajp="0"/>
  ...
</oc4j>
```

To start up OC4J in standalone mode, specify the relevant system properties before the `-jar` option. For example:

```
> java -Djava.security.manager
-Djava.security.policy=$ORACLE_HOME/j2ee/home/config/java2.policy
-Doracle.home=$ORACLE_HOME -jar oc4j.jar
```

At the minimum, the following system properties must be set:

Table 1 System Properties

Property Name	Description
java.security.manager	property to enable Java 2 security in this JVM
java.security.policy	location of your `java2.policy` (the default policy is located at `$ORACLE_HOME/j2ee/home/config/java2.policy`)
oracle.home	value of `$ORACLE_HOME`

7.6 JAAS Login Module Configuration Provider

The JAZN-XML provider type of JAZN is also a JAAS login module configuration provider.

This subsection documents JAAS login module support.

7.6.1 Configure JAAS

7.6.1.1 Configure JVM Add the following lines to your `java.security` configuration file, if not present already:

```
auth.policy.provider=oracle.security.jazn.spi.PolicyProvider
login.configuration.provider=oracle.security.jazn.spi.
LoginConfigProvider
```

This indicates that JAZN is to be used as the provider for JAAS login configuration as well as policy. This is the default configuration for the JDK shipped with Oracle9*i*AS release 2.

The `java.security` file is located at `$JAVA_HOME/jre/lib/security`.

7.6.1.2 Configure JAZN Configure your `jazn.xml` file to use JAZN-XML as the provider:

For example, consider this simple `jazn.xml` file:

```
<jazn provider="XML" location="jazn-data.xml" />
```

This informs JAZN that JAZN-XML is the provider of choice (as opposed to JAZN-LDAP, which does not yet support login module configuration).

You must also configure `jazn-data.xml` properly for login module configuration. You can accomplish that by invoking the `oracle.security.jazn.login.LoginModuleManager` API or editing `jazn-data.xml` manually.

Here is a sample fragment of `jazn-data.xml` that configures a login module for an application:

```
<!-- Login Module Data -->
<jazn-loginconfig>
      <application>
            <name>JAZNUserManager</name>
            <login-modules>
                  <login-module>
                    <class>oracle.security.
                       jazn.realm.RealmLoginModule</class>
                     <control-flag>required</control-flag>
                        <options>
                             <option>
                                 <name>addRoles</name>
                                 <value>true</value>
                             </option>
                        </options>
                  </login-module>
            </login-modules>
      </application>
</jazn-loginconfig>
```

The preceding fragment specifies that for the application `JAZNUserManager`, the login module `RealmLoginModule` is a required component in the authentication process, with the `addRoles` option set to `true`.

For more information about JAAS and JAAS login modules, refer to the JAAS Web site (`http://java.sun.com/products/jaas/`).

7.6.1.3 Start up JVM with JAAS Enabled

Since JAAS is based on Java 2 security, you must first enable the Java 2 security manager according to Section 7.5, "JAAS and Java 2 Security".

In addition to the system properties related to Java 2, the following property must be set:

Table 2 Security Properties

Property Name	Description
oracle.security.jazn.config	The location of your `jazn.xml` configuration file. The default location for this file: `$ORACLE_HOME/j2ee/home/config/jazn.xml`

For example, the following script starts up OC4J with Java 2 and JAAS enabled, in an ADE view:

```
${JAVA_HOME}/bin/java \
-Djava.security.manager \
-Djava.security.policy=${J2EE_HOME}/config/java2.policy \
-Doracle.home=${ADE_VIEW_ROOT} \
-Doracle.security.jazn.config=${J2EE_HOME}/config/jazn.xml \
-jar oc4j.jar
```

This feature is not supported by JAZN-LDAP in release 9.0.2.

8 Release Notes for OC4J Administration and Management

In an Oracle9*i*AS environment, the tools and steps used to manage OC4J processes and modify XML configuration files are *not* the same as for a standalone OC4J environment. This is a change from the Oracle9*i*AS 1.0.2.2 release.

In particular, in an Oracle9*i*AS environment you can no longer do the following:

- Use any Java `-jar` commands to start `oc4.jar`.
- Use `admin.jar` for any purpose.

- Make direct edits to the file system to change configuration and expect OC4J to process them automatically.

In Oracle9*i*AS, two Oracle tools—Oracle Enterprise Manager and the command-line `dcmctl` tool—are used to start, stop, and configure OC4J.

You must run the `dcmctl` tool appropriately after any manual modifications to XML configuration files.

Refer to the *Oracle9iAS Containers for J2EE User's Guide* for additional information. (There are separate versions of this document for Oracle9*i*AS and OC4J standalone. The standalone version is available through OTN.)