BEA WebLogic Java Adapter for Mainframe™

JCA Adapter Guide

Release 5.1
Document Date: August 2002

Copyright

Restricted Rights Legend

Trademarks or Service Marks

BEA WebLogic JAM JCA Adapter Guide

Document Edition	Part Number	Date	Software Version
5.1	N/A	August 2002	5.1

Contents

BEA WebLogic JAM JCA Adapter Guide

BEA WebLogic JAM JCA Adapter Guide

The WebLogic JAM JCA Adapter is a JCA 1.0 compliant adapter which installs into the Connector Container of WebLogic Server and provides a standard API interface to WebLogic JAM services.

The WebLogic JAM JCA Adapter implements all functionality documented in the J2EE Connector Architecture Specification Version 1.0 (JSR 016) including support for:

- Local and XA transactions
- J2EE Connector Common Client Interfaces (CCI)
- Resource Adapter and Connection Metadata
- Container-managed security

In addition, the following extensions are supported:

- DataView record type allowing DataViews generated using the eGen utility to be used as input or output of a mainframe service invocation.
- XML record type allowing a properly constructed XML document to be used as input or returned as the response of a mainframe service.

Pre-requisites

WebLogic Server 7.0 must be installed prior to installing the WebLogic JAM JCA Adapter.

Installation

The WebLogic JAM JCA Adapter is delivered as part of the WebLogic JAM product from the BEA Download site or the product CD. The following files are installed:

- `jamjca.jar`

  This file is installed into the `<JAM_INSTALL_DIR>/lib` directory and must be defined in the WebLogic `CLASSPATH`. This file contains the classes required to run the JCA Adapter functionality with WebLogic JAM.

- `jamjca.rar`

  This file is the Resource Adapter Archive file and is installed in `<JAM_INSTALL_DIR>/lib`. This is the file that is deployed via the WebLogic Administration Console.

Install and Define the jamjca.jar

The `jamjca.jar` file must be defined in the WebLogic system `CLASSPATH` so that WebLogic JAM JCA clients running under the control of the WebLogic Connector Container have access to the classes which are not defined by standard J2EE JCA interfaces.

In addition, many of the features of the WebLogic JAM JCA Adapter require access to DataViews generated using the eGen utility. These DataViews describe the contents of a mainframe data record. It is recommended that a new directory be created in the `<JAM_INSTALL_DIR>` to contain these classes and that this directory be added to the system `CLASSPATH`.

For Windows Systems

To accomplish these changes edit the `startWebLogic.cmd` and make the following changes:

1. Add the following lines to the script:

```
set JAM_HOME=<Install Directory for WebLogic JAM 5.1>
set DATAVIEW_DIR=%JAM_HOME%/DataView
set JAMJCA_JAR=%JAM_HOME%/lib/jamjca.jar
```

2. Locate the following line in the script:

```
set
CLASSPATH=.;.\lib\weblogic_sp.jar;.\lib\weblogic.jar;<JAM_INSTALL
_DIR>\lib\jamjca.jar
```

Append this line of code to it.

```
set CLASSPATH=%CLASSPATH%;%JAMJCA_JAR%;%DATAVIEW_DIR%
```

For Unix Systems

To accomplish these changes edit the `startWebLogic.sh` and make the following changes:

1. Add the following lines to the script:

```
export JAM_HOME=<JAM_INSTALL_DIR>
export DATAVIEW_DIR=$JAM_HOME/DataView
export JAMJCA_JAR=$JAM_HOME/lib/jamjca.jar
```

2. Locate the following line in the script:

```
CLASSPATH=.:./lib/weblogic_sp.jar:./lib/weblogic.jar;<JAM_INSTALL
_DIR>/lib/jamjca.jar
```

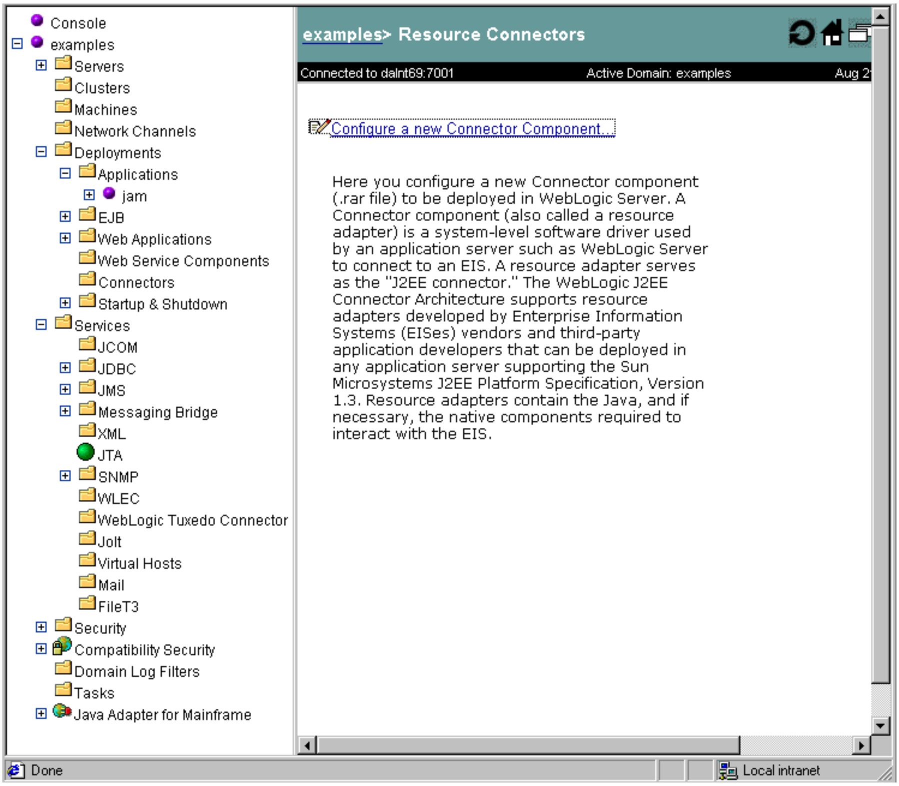
Append this line of code to it.

```
CLASSPATH=$CLASSPATH:$JAMJCA_JAR:$DATAVIEW_DIR
```

Deploying the WebLogic JAM JCA Adapter via the WebLogic Console

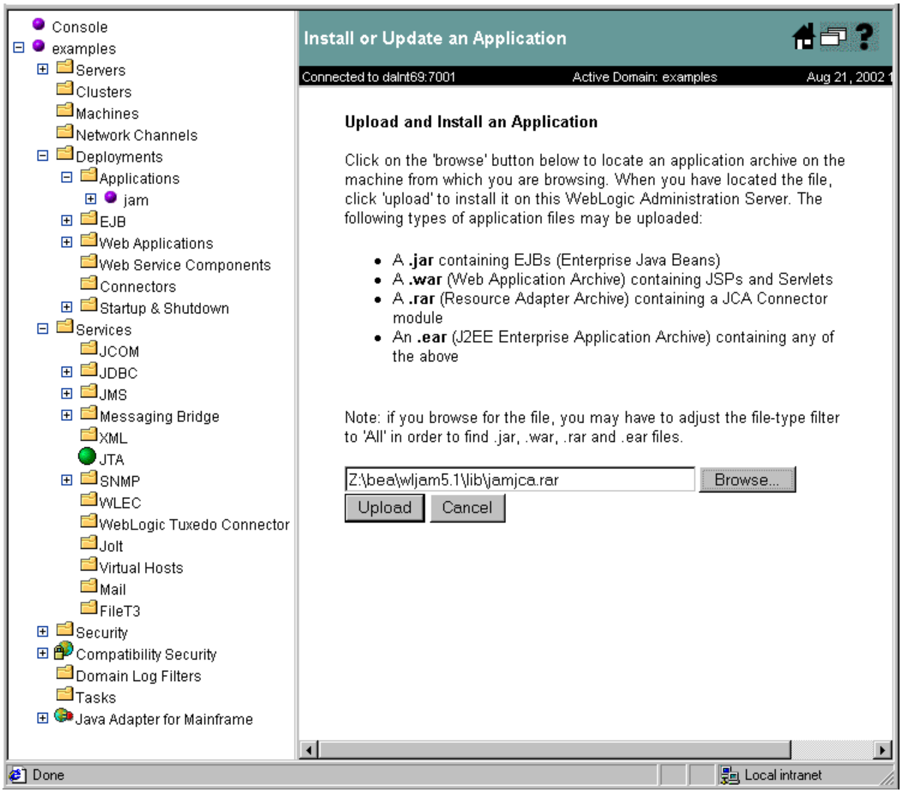
Deployment of the WebLogic JAM JCA Adapter via the WebLogic Administration Console is accomplished as follows:

1. Select the Connectors item in the console tree and then click on Configure a new Connector Component in the right hand pane of the WebLogic Administration Console:



Here you configure a new Connector component (.rar file) to be deployed in WebLogic Server. A Connector component (also called a resource adapter) is a system-level software driver used by an application server such as WebLogic Server to connect to an EIS. A resource adapter serves as the "J2EE connector." The WebLogic J2EE Connector Architecture supports resource adapters developed by Enterprise Information Systems (EISes) vendors and third-party application developers that can be deployed in any application server supporting the Sun Microsystems J2EE Platform Specification, Version 1.3. Resource adapters contain the Java, and if necessary, the native components required to interact with the EIS.

2. Enter the path and filename of the `jamjca.rar` file or use the Browse button to locate the file in the `<JAM_INSTALL_DIR>/lib` directory and then click on the Upload button:



Upload and Install an Application

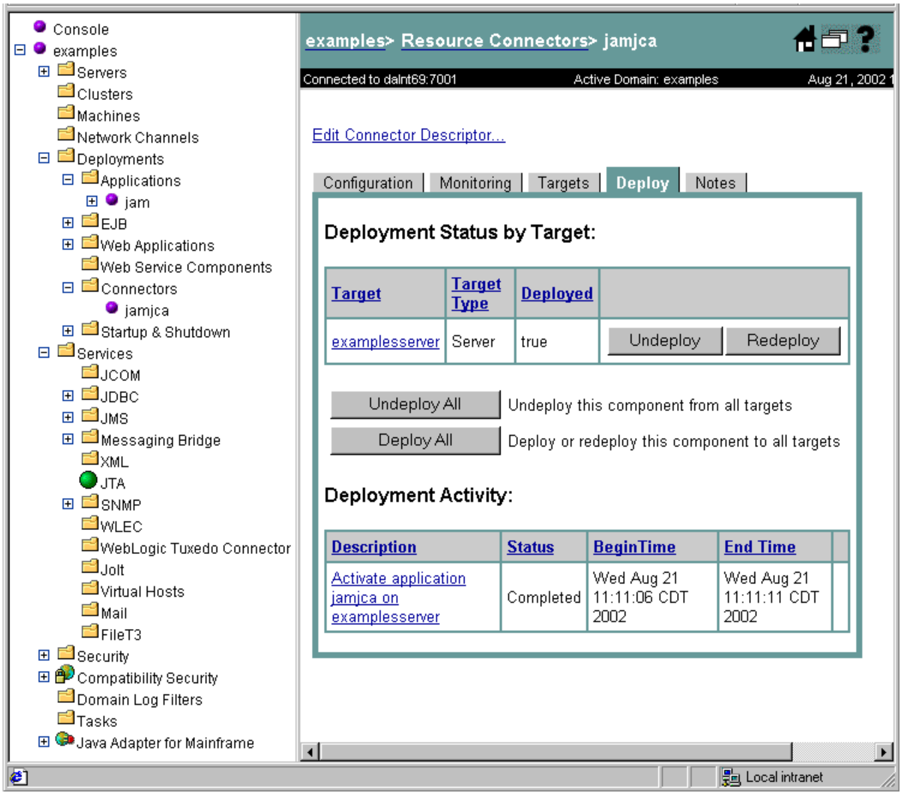
Click on the 'browse' button below to locate an application archive on the machine from which you are browsing. When you have located the file, click 'upload' to install it on this WebLogic Administration Server. The following types of application files may be uploaded:

- A **.jar** containing EJBs (Enterprise Java Beans)
- A **.war** (Web Application Archive) containing JSPs and Servlets
- A **.rar** (Resource Adapter Archive) containing a JCA Connector module
- An **.ear** (J2EE Enterprise Application Archive) containing any of the above

Note: if you browse for the file, you may have to adjust the file-type filter to 'All' in order to find .jar, .war, .rar and .ear files.

Z:\bea\wljam5.1\lib\jamjca.rar

3. The WebLogic JAM JCA Adapter is now deployed:



Deployment Status by Target:

Target	Target Type	Deployed	
examplesserver	Server	true	Undeploy Redeploy

Undeploy All — Undeploy this component from all targets

Deploy All — Deploy or redeploy this component to all targets

Deployment Activity:

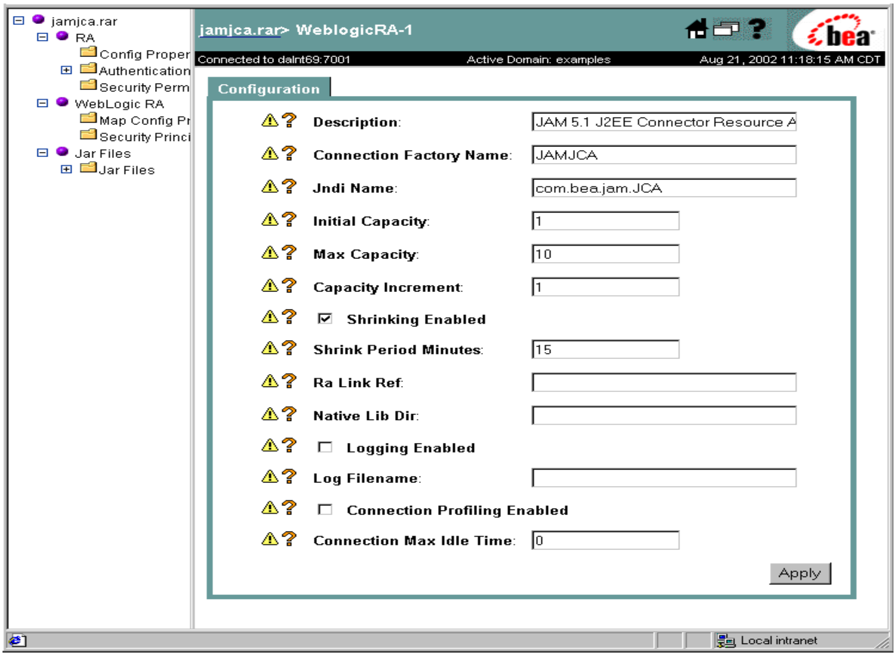
Description	Status	BeginTime	End Time
Activate application jamjca on examplesserver	Completed	Wed Aug 21 11:11:06 CDT 2002	Wed Aug 21 11:11:11 CDT 2002

Using the WebLogic JAM JCA Adapter

WebLogic JAM JCA Adapter provides a client interface using standard JCA system contracts to access WebLogic JAM services. The following sections provide guidelines for configuration and programming with the WebLogic JAM JCA Adapter.

Configuring the JCA Adapter

After installing the WebLogic JAM JCA Adapter, you may configure it using the Edit Connector Descriptor link of the WebLogic Administration Console. For more information on the Connector Container options in WebLogic Server, refer to the WebLogic Server documentation.



Description:	JAM 5.1 J2EE Connector Resource A
Connection Factory Name:	JAMJCA
Jndi Name:	com.bea.jam.JCA
Initial Capacity:	1
Max Capacity:	10
Capacity Increment:	1
☑ Shrinking Enabled	
Shrink Period Minutes:	15
Ra Link Ref:	
Native Lib Dir:	
☐ Logging Enabled	
Log Filename:	
☐ Connection Profiling Enabled	
Connection Max Idle Time:	0

Programming Client Applications

Creating client applications which make use of the WebLogic JAM JCA Adapter follows the standard practices for all Connector clients. The general steps are:

1. Obtain a JAM JCA Connection Factory. In the Managed case this factory is obtained from a Pool maintained by the WebLogic Connector Container. In the Non-managed (stand-alone) case the ConnectionFactory is directly instantiated by the client.

2. Use the ConnectionFactory to obtain a Connection object.
3. Using the Connection object, request an Interaction object.
4. Instantiate an InteractionSpec object used to identify the mainframe service to be invoked.
5. Create a Record object for the input data to the service and for the response.
6. Use the Interaction object to execute the mainframe service passing the InteractionSpec, Input Record, and Output Record.
7. Close the connection.

Example Code

The following listing is simple example code which outlines the previous programming steps for a non-managed (standalone) invocation:

Listing 1 Example Programming Code

```
import javax.resource.cci.*;
import com.bea.jam.jca.*;
String url = "t3://localhost:7001";
ConnectionFactory fact = new ConnectionfactoryImpl(url);
Connection conn = fact.getConnection();
Interaction action = conn.createInteraction();
InteractionSpecImpl actionSpec = new InteractionSpecImpl();
actionSpec.setFunctionName("sampleRead");
actionSpec.setInteractionVerb(InteractionSpec.SYNC_SEND_RECEIVE);
RecordFactory rfact = fact.getRecordFactory();
XmlRecord input = new XmlRecord("EmployeeRecord");
MappedRecord output = rfact.createMappedRecord("EmployeeRecord");
String xml = "<?xml version='1.0'?><empRecord><empName><empLastName>Smith
</empLastName></empName></empRecord>";
input.setData(xml);
action.execute(actionSpec, input, output);
conn.close();
```

Line Number	Description
1 and 2	Import the packages for the J2EE Connector CCI and the WebLogic JAM implementation of these interfaces.
3 and 4	Obtain a connection factory using the URL of the WebLogic Server which is hosting the WebLogic JAM Gateway.
5	Obtain a connection from the connection factory.
6	Use the connection to obtain an Interaction object.
7-9	Create an InteractionSpec object and the following properties: ♦ The FunctionName is the name of the mainframe service as defined in the WebLogic JAM configuration. ♦ The InteractionVerb for a synchronous send and receive of data. This is the only interaction verb supported by the WebLogic JAM JCA Adapter.
10	Obtain a record factory which can be used for the creation of Indexed and Mapped Records.
11	Create an instance of an XmlRecord. This is a WebLogic JAM JCA extension which provides support for conforming XML data. This Record object will contain data translation code to convert the XML data to mainframe data.
12	Create an instance of a MappedRecord which will be used to receive the response from the mainframe service.
13 and 14	Set the value of the input XmlRecord to the requested employee last name.
15	Invoke the mainframe service. On successful return the MappedRecord we passed in as output will contain the service response data.
16	Close the connection.

Record Types

The WebLogic JAM JCA Adapter supports four types of records for input and/or output when executing mainframe services:

Record Type	Description	Record Name Definition
IndexedRecord	This record always contains a single entry which is a byte array containing the binary data record.	An arbitrary name.
MappedRecord	This record contains a Java Map of name/value pairs. ♦ The name is the field name from the data record. ♦ The value is the data associated with that field. This Map is created using the WebLogic JAM HashtableLoader and HashtableUnloader classes. **Note:** Refer to the *WebLogic Java Adpater for Mainframe Programming Guide* for more information about these classes.	Must match the name of the DataView class for the record.
DataViewRecord	This record is a wrapper around the WebLogic JAM DataView class.	Must match the name of the DataView class for the record.
XmlRecord	This record converts an XML document to a binary data record using a WebLogic JAM generated DataView class. **Note:** Refer to the "Understanding How WebLogic JAM Uses XML" section in the *WebLogic Java Adpater for Mainframe Programming Guide* for more information about these classes.	Must match the name of the DataView class for the record.

Samples

The WebLogic JAM JCA Adapter provides two samples: a container-managed connection sample and a non-managed connection sample. These samples are in the following locations:

- Container-Managed Connection Sample

  `<JAM_INSTALL_DIR>/samples/jca/managed`

- Non-Managed Connection Sample

  `<JAM_INSTALL_DIR>/samples/jca/nonmanaged`

Container Managed Connections

The Container Managed sample is a simple JSP page that accesses the `readSample` mainframe service using the `EmployeeRecord` which ships as a WebLogic JAM 5.1 sample.

Building the Sample

To build the Container Managed sample using `ant`:

1. Make sure that the WebLogic `bin` directory is in your `PATH`.
2. Enter the following command to start `ant` passing the directory where WebLogic is installed. For example:

   ```
   ant -DWLSDIR=d:/bea/wlserver7.0
   ```

3. Copy the generated EmployeeRecord DataView to your DataView directory. For example:

   ```
   copy EmployeeRecord*.class d:\bea\wljam5.1\DataView\.
   ```

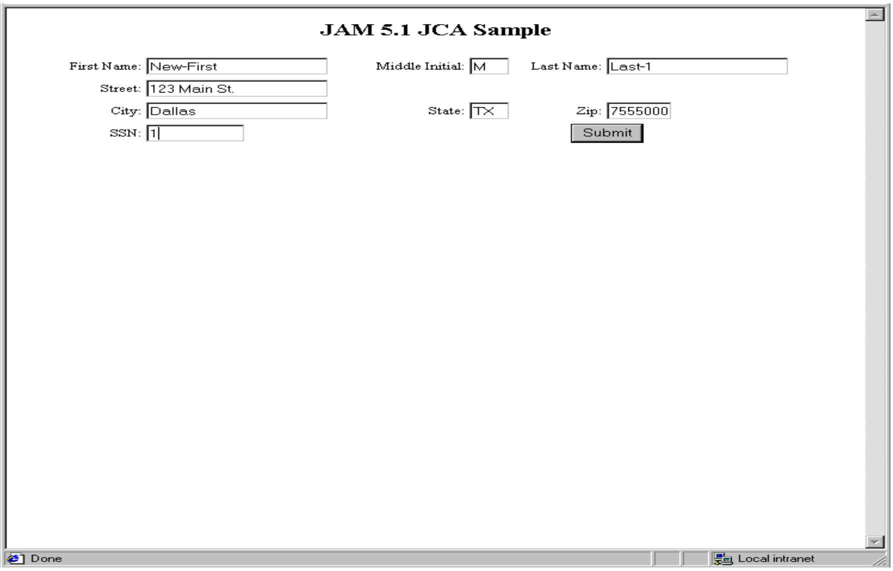
4. Install the `managedjca.war` file.

Running the Sample

After installing the `managedjca.war` file, use the following HTTP URL to launch the JSP in a browser:

```
http://localhost:7001/managedjca/DisplayEmployee.jsp
```

The following page displays.



JAM 5.1 JCA Sample

First Name: New-First
Middle Initial: M
Last Name: Last-1
Street: 123 Main St.
City: Dallas
State: TX
Zip: 7555000
SSN: 1

Enter a name in the Last Name field and click Submit to access the mainframe and display the returned data.

Non-Managed Connections

The non-managed JCA sample is a command line application, which runs the `readSample` service and displays results in various formats.

Building the Sample

To build this sample using `ant`:

1. Make sure the WebLogic `bin` directory is in your `PATH`.
2. Enter the command to start `ant` passing the directory where WebLogic is installed. For example:

   ```
   ant -DWLSDIR=d:/bea/wlserver7.0
   ```

3. Copy the generated EmployeeRecord DataView to your DataView directory. For example:

   ```
   copy EmployeeRecord*.class d:\bea\wljam5.1\DataView\.
   ```

4. Make sure the `weblogic.jar`, `jam.jar`, and `jamjca.jar` files are in your `CLASSPATH`. Also include the current directory in the `CLASSPATH`.

Running the Sample

Run the sample with the following command line format:

```
java DisplayEmployee <URL of WLS> <Last Name of Employee>
```

Running this sample does the following:

1. Obtains the JAM JCA metadata classes and displays product name, version, etc.

2. Executes the `readSample` service returning a MappedRecord. The first and last names are displayed from this response.

3. Executes the `readSample` service returning an XmlRecord. The resulting XML document displays.

Listing 2 Running Non-managed Connection Sample

```
java DisplayEmployee t3://localhost:7001 Last-1

Product Name: WebLogic Java Adapter for Mainframe
Product Version: 5.1
Support for J2EE Connector Version 1.0

First Name:   New-First
Last Name:    Last-1

<?xml version="1.0"?>
<empRecord>
<empSsn>1</empSsn>
```

```
<empName>
<empNameLast>Last-1</empNameLast>
<empNameFirst>New-First</empNameFirst>
<empNameMi>M</empNameMi>
</empName>
<empAddr>
<empAddrStreet>123 Main St.</empAddrStreet>
<empAddrSt>TX</empAddrSt>
<empAddrZip>775550000</empAddrZip>
</empAddr>
</empRecord>
```