Using the JNBridge JMS Adapter for BizTalk Server with Oracle WebLogic
Version 4.0

www.jnbridge.com

Using the JMS Adapter with WebLogic

JNBridge, LLC
www.jnbridge.com

Contents

Quick Config for WebLogic example JMS server

Adapter Transport Handler Properties

■ Initial Context Factory: `weblogic.jndi.WLInitialContextFactory`

■ JMS Scheme: `t3`

■ Queue Connection Factory: `weblogic/examples/jms/QueueConnectionFactory`

■ Topic Connection Factory: `weblogic/examples/jms/TopicConnectionFactory`

■ Class Path

WebLogic 9.2

`wljmsclient.jar, wlclient.jar`

WebLogic 10.x

`wljmsclient.jar, wlclient.jar, wlsafclient.jar`

WebLogic 11g and 12c

`wlthint3client.jar`

! *WebLogic version 9.x jar files may fail on a send with a* java.lang. NoClassDefFoundError *as the anonymous inner class* weblogic.jms.common. MessageImpl$1 *cannot be found. Please use the jar files from WebLogic version 10.x: they're backward compatible with WebLogic version 9.x.*

■ JVM Path (examples)

`C:\Program Files\Java\jre7\bin\client\jvm.dll`

Adapter Send or Receive Port Properties

■ Port Number: `7001`

■ JMS Object Name

`weblogic/examples/jms/exampleQueue`

`weblogic/examples/jms/exampleTopic`

Using the JMS Adapter with WebLogic

This document uses the example JMS service that comes pre-configured in WebLogic. 9.2. WebLogic versions 10, 11g and 12c also contain an example JMS service: they will be similar to the configurations, user interface and properties discussed here. This document assumes some passing familiarity with enterprise Java such as JNDI contexts, factories and general connection requirements and parameters. This document also assumes that the BizTalk developer has access to particular information peculiar to the target JMS implementation. Finally, this document assumes that the reader is knowledgable and experienced with BizTalk Server.

Resources

- The user guide, *Using the JNBridge JMS Adapter for BizTalk Server.*

- Chances are, if the target WebLogic JMS implementation is mature, the values for the configuration of BizTalk transport handlers and send/receive ports can be supplied by the WebLogic administrator, gleaned from existing JMS client code or property files, e.g. *jndi.properties*.

- If the WebLogic JMS implementation targeted is not configured, then the default example JMS service installed with WebLogic can be used for proof-of-concept evaluations. This document uses the example JMS service in WebLogic 9.2.

BizTalk Machine Prerequisites

The following prerequisites are needed for the adapter.

- A Java Run-time Environment (JRE) must be installed on the target machine. The JNBridge JMS Adapter supports the Standard Edition JRE 7 or above. For WebLogic 8.1, the adapter can support JRE 1.4. A JRE may be downloaded from java.sun.com/javase/downloads.

- The JNBridge JMS Adapter for BizTalk uses the stand-alone or thin-client JMS environment supplied by Oracle.

Deploy JMS Header Schema to BizTalk Application

In order to properly handle JMS header properties within BizTalk, you must deploy the assembly, `JNBridgeBTS2006JMSProperties.dll`, to your BizTalk application. This assembly contains the XSD namespaces and schemas used by the JNBridge JMS Adapter to promote JMS header properties within messages stored in the BizTalk Message Box.

! *Deploying this assembly is mandatory.*

■ To deploy the schema assembly

1   Open up the BizTalk Administrator and open your BizTalk application in the left-side tree view.

2   Right click on the application's root node and choose Add ▶ Resources. This opens the Add Resources dialog.

3   In the dialog, click on the Add button and navigate to the schema DLL in the adapter install directory, e.g. `C:\Program Files\JNBridge\JMSAdapters\BTS2006\bin\ JNBridgeBTS2006JMSProperties.dll`.

4   Click on OK to close the Add Resources dialog.

5   Open the Schemas folder in your application. You should see the three schemas: `JMSSendProperty.SendPropertySchema`, `JMSRecvProperty. RecvPropertySchema` and `JMSConfProperty.ConfPropertySchema`.

6   Restart the host instance and application.

Configuring the Adapter Send and Receive Transport Handlers

The transport handler property grids for the Send and Receive sides contain properties global to all send or receive ports configured to use the JNBridge JMS Adapter and that reside in the same BTS host instance. In other words, all JMS Adapter send or receive ports in the BTS host instance will inherit these transport properties. You must configure send handler transport properties in order to produce messages to queues and topics. Likewise, you must configure receive handler transport properties in order to consume messages from queues and topics. In most cases, the values of the properties will be identical between the send and receive handlers; however, depending on the JMS server implementation, they may be different.

JMS Properties Category

The JMS Properties category are properties used to properly connect to a JMS server.

■ JMS Acknowledge Mode

The Acknowledge Mode is a drop-down list containing the JMS specification that determines

how a JMS client and server institute a reliable messaging protocol. The choices are AUTO_ ACKNOWLEDGE, CLIENT_ACKNOWLEDGE and DUPS_OK_ACKNOWLEDGE. Regardless of the choice, the JNBridge JMS Adapter will correctly implement the protocol. For the example WebLogic JMS service, AUTO_ACKNOWLEDGE is the default configuration.

■ Initial Context Factory

This is a text-editable field containing the name of the JNDI initial context factory. The initial context factory is a JNDI class used to locate and instance factories and JMS destinations. The default initial context factory for WebLogic is:

> **`weblogic.jndi.WLInitialContextFactory`**

! *Factory names are case sensitive—be sure the name (including the complete namespace, if necessary) is typed correctly.*

■ JMS Scheme

This is a text-editable field. The JMS Scheme or *Protocol* is particular to each vendor's RMI implementation. The protocol is part of the URI used to connect to the JMS service.

For WebLogic, the scheme is:

> **`t3`** or **`t3s`**.

When using the WebLogic thin-client, the t3 protocol is actually implemented as RMI-IIOP.

■ JMS Version

This property tells the adapter which JMS implementation to expect when it loads the vendor's client stack—the JAR files in the Class Path property. The drop-down list contains two choices, 'JMS 1.1' and 'JMS 2.0'.

■ JMS Security Mode

The JMS Security Mode is a drop-down list that specifies the type of security required by the JMS server implementation. The choices are none, simple and strong. If the choice is simple, then the server expects a user name and password. If the choice is strong, then the server expects a digital certificate security strategy, usually based on the X.509 standard.

! *If the JMS server implements simple security, it is not necessary to configure this property. Enter a user name and password—the JNBridge JMS Adapter will automatically switch to the simple security mode.*

- Queue Connection Factory

    This is a text-editable field. The default queue connection factory in WebLogic is:

    `weblogic/examples/jms/QueueConnectionFactory`

! *This value includes the JNDI directory path to the connection factory. The complete JNDI path must be provided.*

- Topic Connection Factory

    This is a text-editable field. The default queue connection factory in WebLogic is:

    `weblogic/examples/jms/TopicConnectionFactory`

! *This value includes the JNDI directory path to the connection factory. The complete JNDI path must be provided.*

JNBridge Properties Category

The JNBridge Properties Category correctly configure the .NET-to-Java interoperability core components.

- Class Path

    The Java class path is a set of semicolon-separated paths to the JAR or class files required for a JMS client installation. The class path is used by the JNBridge Java and .NET interoperability components to locate the JMS and JNDI client Java classes so they can be instanced in the Java Virtual Machine.

    To edit the class path, click in the field to enable the browse button. Click on the button to launch the Edit Class Path dialog. Note that only checked elements will be added to the class path when the dialog is dismissed.

    WebLogic 9.2

    ```
    [WebLogic_Directory]\server\lib\wljmsclient.jar
    [WebLogic_Directory]\server\lib\wlclient.jar
    ```

    WebLogic 10.x

    ```
    [WebLogic_Directory]\server\lib\wljmsclient.jar
    [WebLogic_Directory]\server\lib\wlclient.jar
    [WebLogic_Directory]\server\lib\wlsafclient.jar
    ```

    WebLogic 11g and 12c

    ```
    [WebLogic_Directory]\server\lib\wlthint3client.jar
    ```

! *WebLogic version 9.x jar files may fail on a send with a* java.lang. NoClassDefFoundError *as the anonymous inner class* weblogic.jms.common. MessageImpl$1 *cannot be found. Please use the jar files from WebLogic version 10.x: they're backward compatible with WebLogic version 9.x.*

■ JVM Path

The JVM Path property is the absolute path to the Java Virtual Machine implementation, **jvm.dll**. To edit the JVM Path property, click in the field to enable the browse button. Click on the button to launch the standard File Open dialog. Navigate to jvm.dll and click OK. In the example shown, the JSE used is:

> **C:\Program Files\Java\jre7\bin\client\jvm.dll**

Security Properties Category

This category provides security credentials necessary to connect to a JMS server, if the JMS implementation supports security mode simple.

■ Password

Click in this field to drop-down the password edit field. Type in the password.

■ User Name

This is a text editable field. Enter the user name necessary to connect to the JMS server.

Configuring Send Ports and Receive Locations

Configuring Send Ports

Connection Properties Category

These properties determine where the JMS server resides and the port number where it is listening for connections.

- Host Name

  This a text-editable field. Enter the name or IP address of the machine hosting the JMS server.

- Port Number

  This is a text-editable field. Enter the port number on which the JMS server is accepting connections. For WebLogic, this port is usually, by default, `7001`.

- Proprietary Connection String

  This is a text-editable field. This property is only used if the JMS implementation uses complex URLs containing query expressions, or some proprietary connection string. that can not be constructed from the Host Name and Port Number properties. If this property contains a value, then the Host Name and Port Number properties will be ignored.

JMS Operation Properties Category

These properties determine what operation the send port will enable.

- JMS Object Name

  This is a text-editable field. Enter the JNDI name bound to the JMS queue or topic. For WebLogic 9.2, the example queue is at:

  ```
  weblogic/examples/jms/exampleQueue
  weblogic/examples/jms/exampleTopic
  ```

- JMS Object Type

  This is a drop-down list containing two values: Queue or Topic.

- Message Type

  This is a drop-down list containing the values: Text, Text UTF, Text ISO-8859-15 or Bytes.

If Text is chosen, then the JNBridge JMS Adapter will send a JMS Text Message. Because a JMS Text Message is by definition UTF-16 BE, the types Text UTF and Text ISO-8859-15 ensure that the string which is the payload of the JMS message is correctly encoded from the binary representation in the BizTalk Message Box. If the type Text is chosen, then the binary representation is considered UTF-8. If Bytes is chosen, then the JNBridge JMS Adapter will send a JMS Bytes Message.

Configuring Receive Ports and Locations

Connection Properties Category

These properties determine where the JMS server resides and the port number where it is listening for connections.

■ Host Name

This a text-editable field. Enter the name or IP address of the machine hosting the JMS server.

■ Port Number

This is a text-editable field. Enter the port number on which the JMS server is accepting connections. For WebLogic, this port is usually, by default, **7001**, **7002** or **7003**.

■ Proprietary Connection String

This is a text-editable field. This property is only used if the JMS implementation uses complex URLs containing query expressions, or some proprietary connection string. that can not be constructed from the Host Name and Port Number properties. If this property contains a value, then the Host Name and Port Number properties will be ignored.

JMS Operation Properties Category

These properties determine what operation type of operation the send port will enable.

■ JMS Object Name

This is a text-editable field. Enter the JNDI name bound to the JMS queue or topic. For WebLogic 9.2, the example topic is at:

```
weblogic/examples/jms/exampleTopic
weblogic/examples/jms/exampleQueue
```

■ JMS Object Type

This is a drop-down list containing two values: Queue, Topic or SharedTopic.

■ Message Type

This is a drop-down list containing the values: Text, Text UTF-8, Text UTF-16, Text ISO-8859-15, Bytes or Map. If Text is chosen, then the JNBridge JMS Adapter expects to receive a JMS Text Message. If Bytes is chosen, then the JNBridge JMS Adapter expects to receive a JMS Bytes Message. If Map is chosen, then the JNBridge JMS Adapter expects to receive a JMS Map Message. Because a JMS Text Message by definition contains UTF-16 BE text, the types Text UTF-8, Text UTF-16 and Text ISO-8859-15 ensure that the text in the body of the message is encoded correctly to binary for submittal to the BizTalk Message Box. The type Text without a qualifier means UTF-8.

■ Client ID

This is a unique string that identifies the receive port connection to WebLogic. It is only used if durable subscriptions are enabled.

■ Durable Subscription Name

Durable subscriptions are particular to topics only. A durable subscription for a topic allows consumers to register a name with the JMS server such that whenever a receive port is active, all messages in the topic will be received. In this way, a receive port does not have to be continually connected to receive messages from a topic. A receive port that does not use durable subscriptions must be active and connected in order to subscribe to a topic—any messages published by the topic while a nondurable receive port is not active will not be available to that receive port when it becomes active. This is a text-editable field. Enter the durable subscription name.

■ Message Selector Filter

Message selectors are used by receive ports to filter or select messages from topics and queues based on JMS and custom message header properties.

This is a text-editable field. Enter in a selector expression. The expression is derived from a subset of the SQL92 standard.

JNDI Names

Figure1 shows the WebLogic 9.2 Admin Console. The console is displaying the JNDI directory structure for a server. It is important to use complete JNDI paths, either forward slash ('/') or dot ('.') delineated, for connection factories and JMS destinations.
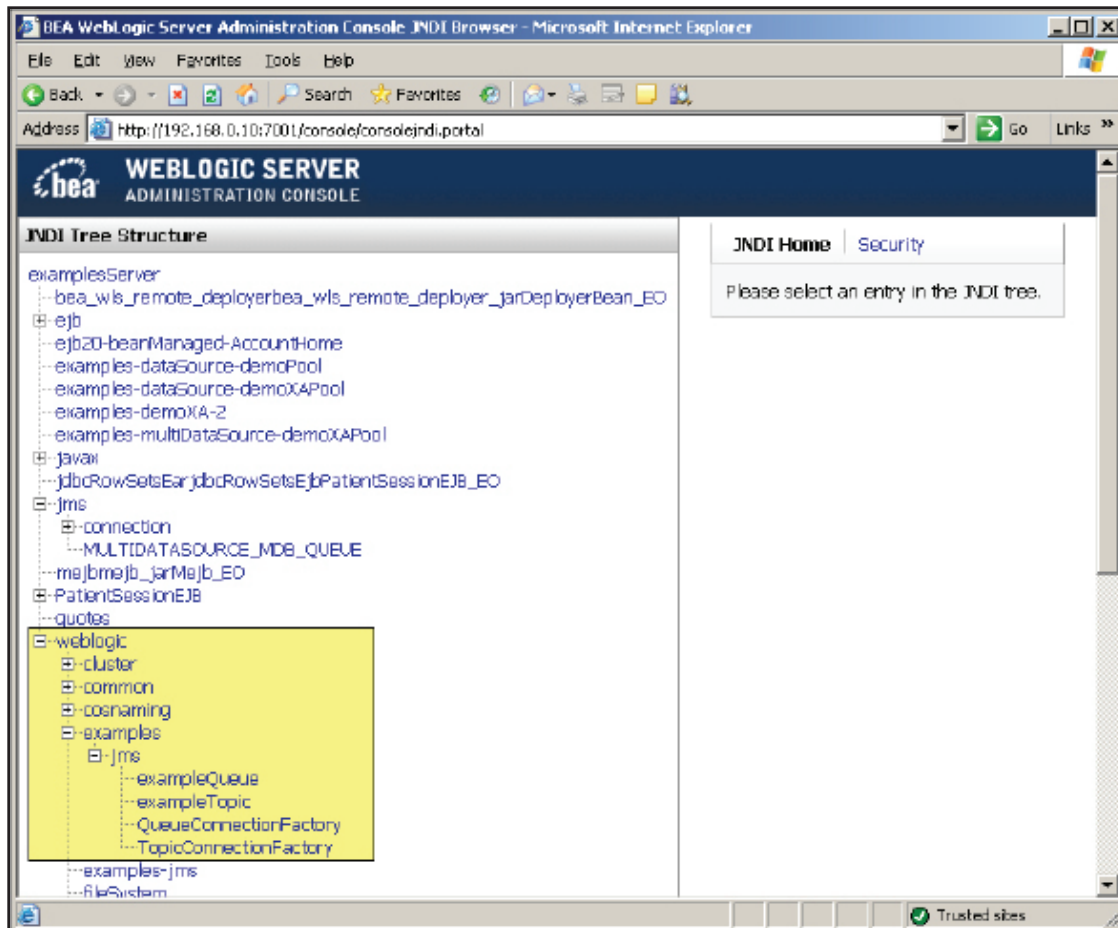

Figure 1. The JNDI tree

Touble Shooting

Could not initialize LookupInitializer class

On occasion, the WebLogic JMS client may throw an exception complaining about the inability to initialize an i18n class. This exception seems to be related to closing, then reopening the connection to the server. This can happen if the send or receive location is disabled or enabled. If the fault-tolerant connection properties are used and the adapter drops and then re-connects to WebLogic, this exception can also occur. This is the exception:

```
An unexpected failure occurred while processing a message. The
text associated with the exception is "Could not initialize class
weblogic.i18ntools.L10nLookup$L10nLookupInitializer".
```

The associated Java-side stack trace looks like this:

```
java.lang.ExceptionInInitializerError
at weblogic.i18ntools.L10nLookup.getLocalizer(L10nLookup.java:362)
at weblogic.i18n.logging.Loggable.getMessage(Loggable.java:181)
at weblogic.i18n.logging.Loggable.getMessage(Loggable.java:207)
at weblogic.jms.common.JMSException.<init>(JMSException.java:66)
        .
        .
        .
Caused by: java.lang.NullPointerException
at weblogic.i18ntools.L10nLookup.loadProps(L10nLookup.java:125)
at weblogic.i18ntools.L10nLookup.<init>(L10nLookup.java:187)
at weblogic.i18ntools.L10nLookup.<init>(L10nLookup.java:27)
at weblogic.i18ntools.L10nLookup$L10nLookupInitializer.<clinit>(L10nLookup.
java:79)
```

This exception is caused by the Java class loader not being able to locate an inner class. This can be solved by appending the WebLogic thin-client JAR files to the JVM boot class path using the JVM Arguments property in the send and receive transport handlers. Enter this into the JVM Arguments property:

-Xbootclasspath/a:[CLASSPATH]

where [CLASSPATH] is the same class path as in the Class Path property.

Using the WebLogic Cluster Initial Context Factory

Sometimes it is necessary for a JMS client to connect to a WebLogic cluster. The usual—and preferred—configuration is to use DNS for the list of hostname:port pairs for each machine in the cluster. If it is necessary to configure each client with the list, them the supplied ICF,