Using the JNBridge JMS Adapter for BizTalk Server with SonicMQ

Version 4.0

www.jnbridge.com

JNBridge, LLC
www.jnbridge.com

Contents

Quick Config for SonicMQ

Adapter Transport Handler Properties

- Initial Context Factory: **`com.sonicsw.jndi.mfcontext.MFContextFactory`**
- JMS Scheme: **`tcp`**
- Queue Connection Factory:

  No connection factories come preconfigured in the Sonic MQ JNDI repository. These must be configured

- Topic Connection Factory:

  No connection factories come preconfigured in the Sonic MQ JNDI repository. These must be configured

- Class Path:

  `sonic_Client.jar, mfcontext.jar`

  The client JAR file, **`sonic_Client.jar`**, contains a classpath in its manifest. As such, the directory must also contain: **`sonic_XA.jar`**. In addition, if SSL is being used, then **`rsa_ssl.jar`** must be included in the classpath. This jar file also contains a classpath in its manifest, so the directory must also contain: **`asn1.jar, certj.jar, sslj.jar, jsafe.jar`** and **`jsafeJCE.jar`**

- JVM Path (examples):

  `C:\Program Files\Java\jre7\bin\client`

Adapter Send or Receive Port Properties

- Port Number: **`2506`**
- JMS Object Name

  Sonic MQ does come with pre-configured queues and topics, however they must be mapped to a JNDI name in the Sonic MQ JNDI repository.

Using the JMS Adapter with SonicMQ

This document uses the default JMS broker pre-configured in SonicMQ. 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 knowledgeable and experienced with BizTalk Server.

This document assumes that SonicMQ is stand-alone rather than deployed to a JEE application server as the messaging provider using a JCA connector. If this is the case, then refer to the SonicMQ documentation and the JEE app server documentation.

This document only discusses those property values in the adapter transport handlers and location handlers that pertain to communicating with SonicMQ. Other properties that are not discussed here can be found in the companion *Using the JNBridge JMS Adapter for BizTalk Server* document.

Resources

- The user guide, *Using the JNBridge JMS Adapter for BizTalk Server.*
- Chances are, if the target SonicMQ implementation is mature, the values for the configuration of BizTalk transport handlers and send/receive ports can be supplied by the SonicMQ administrator, gleaned from existing JMS client code or property files, e.g. *jndi.properties*.
- If the SonicMQ implementation targeted is not configured, then the default example JMS broker can be used for proof-of-concept evaluations.

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.
- The JNBridge JMS Adapter for BizTalk uses the stand-alone JMS environment supplied by SonicMQ. This environment consists of several JAR files.

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 SonicMQ, **`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 the SonicMQ JNDI implementation:

  ```
  com.sonicsw.jndi.mfcontext.MFContextFactory
  ```

  If SonicMQ is used as a messaging provider with another J2EE app server, then the JNDI initial context class may be different.

  If a named SonicMQ domain is being used, or if a fail-over naming broker is being used, then see the section *Trouble Shooting*. This section will explain how to use the initial context factory installed along with the adapter to pass the domain name and fail-over naming broker name to SonicMQ. The initial context factory that passes the domain name and fail-over naming broker to the SonicMQ broker is:

  ```
  com.sonicsw.jndi.mfcontext.FixedMFContextFactory
  ```

! ***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 implementation. The protocol is part of the URI used to connect to the JMS service.

  For SonicMQ, the default scheme is:

  ```
  tcp
  ```

- 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 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. No connection factories come preconfigured in the Sonic MQ JNDI repository. These must be configured

- Topic Connection Factory

  This is a text-editable field. No connection factories come preconfigured in the Sonic MQ JNDI repository. These must be configured

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.

  The SonicMQ jar file required by the JMS adapter:

  `[SonicMQ_Directory]\MQ7.6\lib\sonic_Client.jar`

  `[SonicMQ_Directory]\MQ7.6\lib\mfcontext.jar`

  The client JAR file, **`sonic_Client.jar`**, contains a classpath in its manifest. As such, the directory must also contain: **`sonic_XA.jar`**. In addition, if SSL is being used, then **`rsa_ssl.jar`** must be included in the classpath. This jar file also contains a classpath in its manifest, so the directory must also contain: **`asn1.jar, certj.jar, sslj.jar, jsafe.jar`** and **`jsafeJCE.jar`**. These additional files are also found in the in the 'lib' directory.

  If the work-around initial context factory, installed with the adapter, that supplies the SonicMQ domain name and/or fail-over naming broker to the server is used, then the JAR file containing the fixed factory, **`jnb_sonic_icf_fix.jar`**, must be placed before **`mfcontext.jar`** in the Class Path property. Please see the section *Trouble Shooting*.

- 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 JVM 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 SonicMQ configured for the tcp transport, this port is usually, by default, `2506`.

- 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. Sonic MQ does come with pre-configured queues and topics, however they must be mapped to a JNDI name in the Sonic MQ JNDI repository.

- 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 SonicMQ configured for the tcp transport, this port is usually, by default, `2506`.

- 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. Sonic MQ does come with pre-configured queues and topics, however they must be mapped to a JNDI name in the Sonic MQ JNDI repository.

- 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 SonicMQ. 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.

Trouble Shooting

Using the Support Initial Context Factory

SonicMQ has the concept of broker domain names. As such, a JMS client, like the adapter, must specify the domain name in the connection properties. SonicMQ uses a JEE property, com.sonicsw.jndi.mfcontext.domain, to specify the domain name. If no domain name is used in the SonicMQ implementation, then there is no need to specify the name in the connection.

The problem is that the JEE property cannot be passed to the JVM using the '`-D`' argument. Instead, the property must be specified in the arguments used to construct the JNDI initial context. The JNBridge JMS Adapter for BizTalk does not allow this. Therefore, JNBridge support has supplied a special initial context factory that uses the '`-D`' argument to specify the domain name and, also, to specify a fail-over naming broker, if one is configured.

The special initial context factory can be found in the zip file, `SonicMQ_ICF.zip`. This archive is located in the adapter's installation directory, e.g. `C:\Program Files\JNBridge\JMSAdapters\BTS2006\support`.

The zip file contains the source for the initial context factory as well as the Eclipse project (Galileo JEE). To use the supplied work-around initial context factory unzip the archive to a convenient location. The JAR file containing the initial context factory is `jnb_sonic_icf_fix.jar`.

Follow these instructions to use the work-around initial context factory.

1. Copy the JAR file, `jnb_sonic_icf_fix.jar`, to the location where the other JAR files that constitute the SonicMQ JMS client, `sonic_Client.jar` and `mfcontext.jar`, are located.
2. Modify the Class Path property in the send and receive transport handlers by adding the new JAR file to the classpath. The JAR file, `jnb_sonic_icf_fix.jar`, must come before the JAR file, `mfcontext.jar`. Use the Edit ClassPath dialog to move the new JAR file ahead of `mfcontext.jar`.
3. In the Initial Context Factory property in the send and receive transport handlers enter this class: `com.sonicsw.jndi.mfcontext.FixedMFContextFactory`.
4. In the JVM Arguments property in the send and receive transport handlers, enter this value to specify the SonicMQ domain name:

   ```
   -Dcom.sonicsw.jndi.mfcontext.domain=[Domain_Name]
   ```

5. If a fail-over naming broker is being used, then enter this property:

   ```
   -Dcom.sonicsw.jndi.mfcontext.secondaryProviderURL=tcp://hostname:portnum
   ```

In addition to these two SonicMQ JEE properties, the following SonicMQ properties are also supported:

```
com.sonicsw.jndi.mfcontext.idleTimeout
com.sonicsw.jndi.mfcontext.requestTimeout
com.sonicsw.jndi.mfcontext.connectTimeout
com.sonicsw.jndi.mfcontext.node
com.sonicsw.jndi.mfcontext.secondaryNode
```