



**Using the JNBridge JMS Adapter for .NET
with SonicMQ
version 3.0**

www.jnbridge.com

Using the JMS Adapter with SonicMQ

JNBridge, LLC
www.jnbridge.com

COPYRIGHT © 2008-2011 JNBridge, LLC. All rights reserved.

JNBridge is a registered trademark and JNBridgePro and the JNBridge logo are trademarks of JNBridge, LLC.

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

Microsoft, Windows, Windows Vista, Windows 7, Windows 2008, Windows 2008 R2, BizTalk and the Windows logo are trademarks, or registered trademarks of Microsoft Corporation in the United States and/or other countries.

All other marks are the property of their respective owners.

Contents

Quick Config for SonicMQ	4
Binding Properties Tab	4
URI Properties Tab	4
Using the JMS Adapter with SonicMQ.....	5
Resources.....	5
Machine Prerequisites	5
Configuring the Adapter Connection Properties	6
Binding Properties Tab	6
JNBridge Properties Category.....	8
URI Properties Tab	9
Security Tab.....	9
Touble Shooting.....	10
Using the Support Initial Context Factory	10

Using the JMS Adapter with SonicMQ

Quick Config for SonicMQ

Binding Properties Tab

- Initial Context Factory: `com.sonicsw.jndi.mfcontext.MFContextFactory`
- JMS Scheme: `tcp`
- Queue Factory:
No connection factories come preconfigured in the Sonic MQ JNDI repository. These must be configured
- Topic 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 (example)
`C:\Program Files\Java\jre1.5.0_10\bin\client\jvm.dll`

URI Properties Tab

- Port Number: `2506`

Using the JMS Adapter with SonicMQ

This document uses the example JMS service that comes 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 .NET developer has access to particular information peculiar to the target JMS implementation. Finally, this document assumes that the reader is knowledgeable and experienced with .NET.

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 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 .NET* document.

Resources

- The user guide, *JNBridge JMS Adapter for .NET Users' Guide*.
- Chances are, if the target JMS implementation is mature, the values for the configuration of can be supplied by the SonicMQ administrator, developers or gleaned from existing JMS client code.
- It is strongly suggested that the developer read the section *Tips and Tricks* in the *JNBridge JMS Adapter for .NET Users' Guide*.

Machine Prerequisites

The following prerequisites are needed for the adapter.

- A public Java Run-time Environment (JRE) must be installed on the target machine. The JNBridge JMS Adapter supports the Standard Edition JRE 5 or above.
- The JNBridge JMS Adapter for .NET uses a stand-alone client JMS environment supplied by SonicMQ. This stand-alone client consists of several files.

Using the JMS Adapter with SonicMQ

Configuring the Adapter Connection Properties

The **Add Adapter Service Reference** development tool in Visual Studio is used to generate the `app.config` file and the WCF client file. The WCF client contains the methods chosen to send and receive JMS messages. The `app.config` file contains the binding element whose attribute values are the parameters used to initialize and connect to the JMS server. Each of the binding attribute values can be entered in the **Binding Properties** tab of the **Configure Adapter** dialog box. While it is possible to enter these values and then connect to the JMS server from the **Add Adapter Service Reference** dialog box, it is more efficient to work off-line and generate the `app.config` and the WCF client using only the generic operations. Please see the section *Tips and Tricks* in the *JNBridge JMS Adapter for .NET Users' Guide* for more information about working off-line and using the generic operations.

What follows are the property values required to connect to the SonicMQ. The version of SonicMQ discussed in this document is 7.6.

Binding Properties Tab

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

■ Choose JMS Vendor

This is a drop-down control that comes pre-charged with default vendor connection properties. Click and select *SonicMQ* and the default configuration values will automatically appear.

■ 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 a default SonicMQ connection factory, **`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

Using the JMS Adapter with SonicMQ

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

For SonicMQ, the scheme is:

`tcp`

■ 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

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

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

■ Topic Connection Factory

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

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

Using the JMS Adapter with SonicMQ

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 to .NET interoperability components to locate the JMS and JNDI client Java classes so they can be instantiated 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 JRE used is:

```
C:\Program Files\Java\jre1.6.0_04\bin\client\jvm.dll
```

URI Properties Tab

This category provides the location of the host where SonicMQ is running and the port that SonicMQ listens to for connections.

- **Host**

The host name or IP address of the machine running SonicMQ.

- **Port**

This is a text editable field. Enter the port where SonicMQ is listening for client connections. By default this is port **2506**.

Security Tab

This tab need only be used if security is implemented in the SonicMQ server and is of type *simple*. Note that the password will appear in clear text in the app.config file. Please see the section *Deploying Solutions* in the *Users' Guide*.

- **Client credential type**

This is a drop-down control. Choose the credential type *Username*.

- **User name**

Enter the user name credential.

- **Password**

Enter the password credential

Using the JMS Adapter with SonicMQ

Touble 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 .NET 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\DotNet\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 Binding Properties tab 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 Binding Properties tab enter this class:
`com.sonicsw.jndi.mfcontext.FixedMFContextFactory`.
4. In the JVM Arguments property in the Binding Properties tab 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`

Using the JMS Adapter with SonicMQ

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
```