



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

www.jnbridge.com

Using the JMS Adapter with JBoss

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, 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 JBoss JMS.....	4
Binding Properties Tab	4
URI Properties Tab	4
Using the JMS Adapter with JBoss JMS	5
Resources.....	5
Machine Prerequisites	5
Configuring the Adapter Connection Properties	6
Binding Properties Tab	6
JNBridge Properties Category.....	7
URI Properties Tab	9
Security Tab.....	9
JNDI Names	10
Touble Shooting.....	11
Unable to connect to the JBoss server.....	11
Unable to cast object	11

Using the JMS Adapter with JBoss

Quick Config for JBoss JMS

Binding Properties Tab

- Initial Context Factory: `org.jnp.interfaces.NamingContextFactory`
- JMS Scheme: `jnp`
- Queue Factory: `ConnectionFactory`
- Topic Factory: `ConnectionFactory`
- Class Path

For JBoss version 6.x and 7.x, HornetQ is the messaging provider, so use these JAR files:

```
...\common\lib\hornetq-bootstrap.jar  
...\common\lib\hornetq-core.jar  
...\common\lib\hornetq-jboss-as-integration.jar  
...\common\lib\hornetq-jms.jar  
...\common\lib\hornetq-logging.jar
```

For JBoss versions 5.x, use this jar file. Note that this jar file contains an embedded class path, so it must remain in situ with all other JBoss jar files:

```
jboss-messaging-client.jar
```

For JBoss versions 4.3.x, use these jar files:

```
jboss-messaging-client.jar, jbossall-client.jar, javassist.jar,  
trove.jar, log4j.jar, jboss-aop-jdk50.jar
```

For JBoss versions 4.2.x and greater, please use these jar files.

```
jbossall-client.jar, javassist.jar, trove.jar, log4j.jar,  
jboss-aop-jdk50.jar
```

For JBoss versions 4.0.x, use this jar file:

```
jbossall-client.jar
```

- JVM Path (example)

```
C:\Program Files\Java\jre1.5.0_10\bin\client\jvm.dll
```

URI Properties Tab

- Port Number: `1099`

Using the JMS Adapter with JBoss JMS

This document uses the example JMS service that comes pre-configured in JBoss. 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.

The configuration in this document is intended for both the open source version of JBoss as well as the subscription version from Red Hat. However, there may be some differences between the two. Please check documentation at either Red Hat or at www.jboss.org.

This document only discusses those property values that pertain to communicating with JBoss. 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 JBoss administrator, developers or gleaned from existing JMS client code.
- If the JBoss JMS implementation targeted is not configured, then the default example JMS service installed with JBoss can be used for proof-of-concept evaluations.
- 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 JBoss. This stand-alone client consists of one to six jar files depending on the JBoss version. Please see the section, *Class Path*, below

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 default native JMS implementation in JBoss. The versions of JBoss discussed in this document are 4.0, 4.2, 4.3, 5.0 and 5.1.

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 *JBoss* 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 JBoss connection factory, `AUTO_ACKNOWLEDGE` is the default configuration.

■ Initial Context Factory

This is a text-editable field containing the name of the initial context factory. The initial context factory is a class used to create a JNDI initial context used to look-up connection factories and destinations. The default initial context factory for JBoss is:

`org.jnp.interfaces.NamingContextFactory`

! *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 JBoss, the scheme is:

`jnp`

■ 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. The default queue connection factory in JBoss JMS is:

`ConnectionFactory`

! *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 JBoss JMS is:

`ConnectionFactory`

! *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 to .NET interoperability components to locate the JMS and JNDI client Java classes so they can be instantiated in the Java Virtual Machine.

Using the JMS Adapter with JBoss

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.

For JBoss version 6.x and 7.x, HornetQ is the messaging provider:

```
[JBOSS_HOME]\common\lib\hornetq-bootstrap.jar
[JBOSS_HOME]\common\lib\hornetq-core.jar
[JBOSS_HOME]\common\lib\hornetq-jboss-as-integration.jar
[JBOSS_HOME]\common\lib\hornetq-jms.jar
[JBOSS_HOME]\common\lib\hornetq-logging.jar
```

JBoss version 5.x

```
[JBOSS_HOME]\client\jbossall-client.jar
```

This JAR file contains an embedded classpath in its manifest that points to many of the JAR files in the `client` directory. As such, it is recommended that the entire `client` directory be copied to the development or deployment machine.

JBoss versions 4.3.x and above

```
[JBOSS_HOME]\client\jboss-messaging-client.jar
[JBOSS_HOME]\client\jbossall-client.jar
[JBOSS_HOME]\client\log4j.jar
[JBOSS_HOME]\server\default\deploy\jboss-aop-jdk50.deployer\jboss-aop-jdk50.jar
[JBOSS_HOME]\client\javassist.jar
[JBOSS_HOME]\client\trove.jar
```

These six JAR files can be copied from the above location in the JBoss directories to a convenient location on the development or deployment machine.

JBoss versions 4.2.x and above

```
[JBOSS_HOME]\client\jbossall-client.jar
[JBOSS_HOME]\client\log4j.jar
[JBOSS_HOME]\server\<SERVER_NAME>\deploy\jboss-aop.deployer\jboss-aop.jar
[JBOSS_HOME]\server\<SERVER_NAME>\lib\javassist.jar
[JBOSS_HOME]\server\<SERVER_NAME>\lib\trove.jar
```

These five JAR files can be copied from the above location in the JBoss directories to a convenient location on the development or deployment machine.

JBoss versions 4.0.x

```
jbossall-client.jar
```

Using the JMS Adapter with JBoss

■ 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\jre1.5.0_10\bin\client\jvm.dll
```

JBoss versions 4.x support Java 5. JBoss version 5.x supports Java 6. Java 4 is not supported.

URI Properties Tab

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

■ Host

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

If you are using a default, out-of-the-box instance of JBoss that is running on the same machine as the development or deployment machine where the adapter is installed, then you can use the loopback interface, localhost (127.0.0.1). If the default, out-of-the-box instance of JBoss is running on a different machine, then you must start JBoss with this argument: `'-b0.0.0.0'`. This allows any IP address to connect to JBoss. If you are trying to connect to a mature instance of JBoss with fully implemented security, make sure that the development machine's IP address is accepted (contact the JBoss admin).

■ Port

This is a text editable field. Enter the port where JBoss is listening for client connections. By default this is port [1099](#).

Security Tab

This tab need only be used if security is implemented in the JBoss 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 JBoss

JNDI Names

Figure 1 shows the JBoss Global JNDI Namespace. This listing can be obtained by invoking the `list()` method of the JMX MBean JNDIView service from the JBoss `jmx-console`. The list is displaying the JNDI directory structure for the initial JBoss installation. It is important to use complete JNDI paths, either forward slash (`/`) or dot (`.`) delineated, for connection factories and JMS destinations.

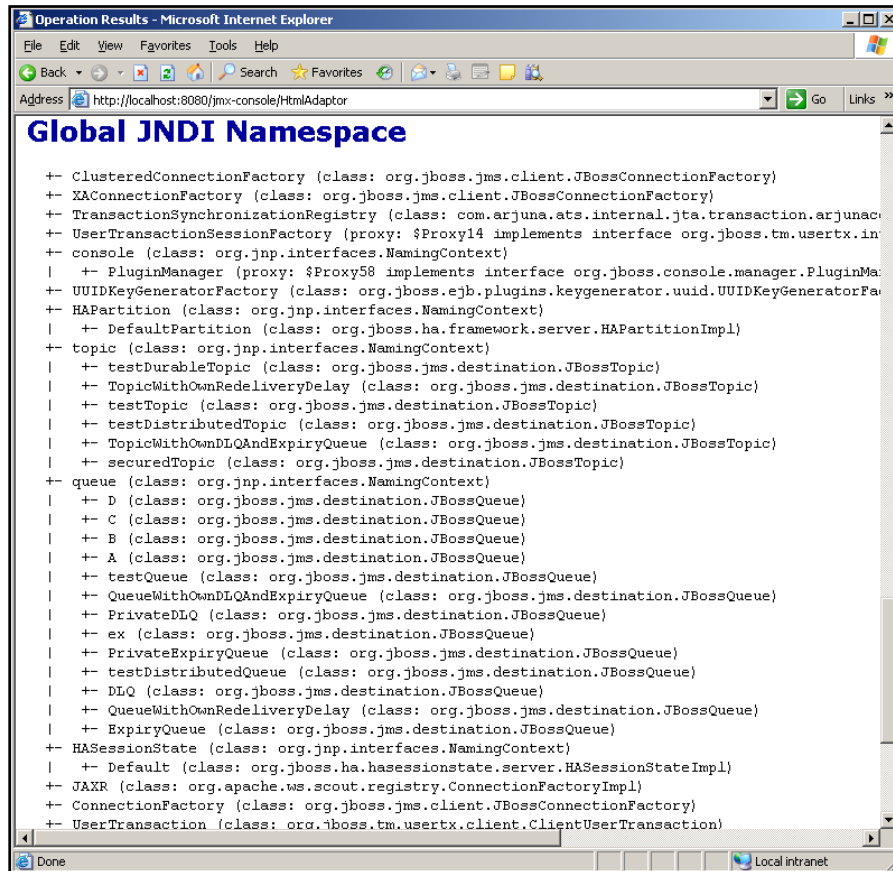


Figure 1. The JNDI tree

Touble Shooting

This section discusses issues and exceptions and their work arounds.

Unable to connect to the JBoss server

This problem can manifest as several exceptions all which indicate that a connection could not be made. Make sure that the machine name and port are correct. Use `telnet.exe` to try and connect to the machine and port as a test.

JBoss connection security is based on allowable client IP addresses. JBoss out-of-the-box only allows connections from the loopback interface, i.e. localhost (127.0.0.1). Starting JBoss with the argument `-b0.0.0.0` will allow connections from any IP address.

Unable to cast object

Depending on the version of JBoss, the following exception, or one like it, may be thrown at start-up or when first accessing a queue or topic:

```
Unable to cast object of type 'javax.naming.Reference' to type 'javax.jms.Queue'. Exception : Method 'getReference' in type 'org.jboss.mq.SpyQueue'
```

This is caused by the JBoss class loader isolation mechanism in JNDI naming environments. It can be solved by placing the JAR files in the JVM boot classpath. This can be done by using the **JVM Arguments** property in the **Binding Properties** tab of the **Configure Adapter** dialog box or by placing this element in the `app.config` file:

```
JVMArgs="-Xbootclasspath/a:[Class Path]"
```

Where [Class Path] is the same class path as the value for the **Class Path** property.