Using Rhapsody 4.01 with GlassFish v2.x-bundled Sun Java System Message Server JMS implementation

Michael.W.Czapski@gmail.com

December 2011

Table of Contents

Introduction

Recently I had an occasion to work on an integration project which required the Rhapsody 4.01-based integration solution to receive messages from a JMS Topic hosted by the Sun Java System Message Queue bundled with the Oracle/Sun GlassFish v2.x JMS . Product documentation and Internet searches did not offer assistance in terms of how the Rhapsody JMS Adapter needs to be configured to support this. While there are a number of articles which discuss the topic of configuring JMS Client to interact with GlassFish-hosted SJSMQ JMS Server, none of the solutions described in these articles worked for me. A degree of experimentation and creative adaptation resulted in a working configuration. This article discusses this solution for the benefit of these who will be faced with this problem and for my own benefit if I need to do this again in the future.

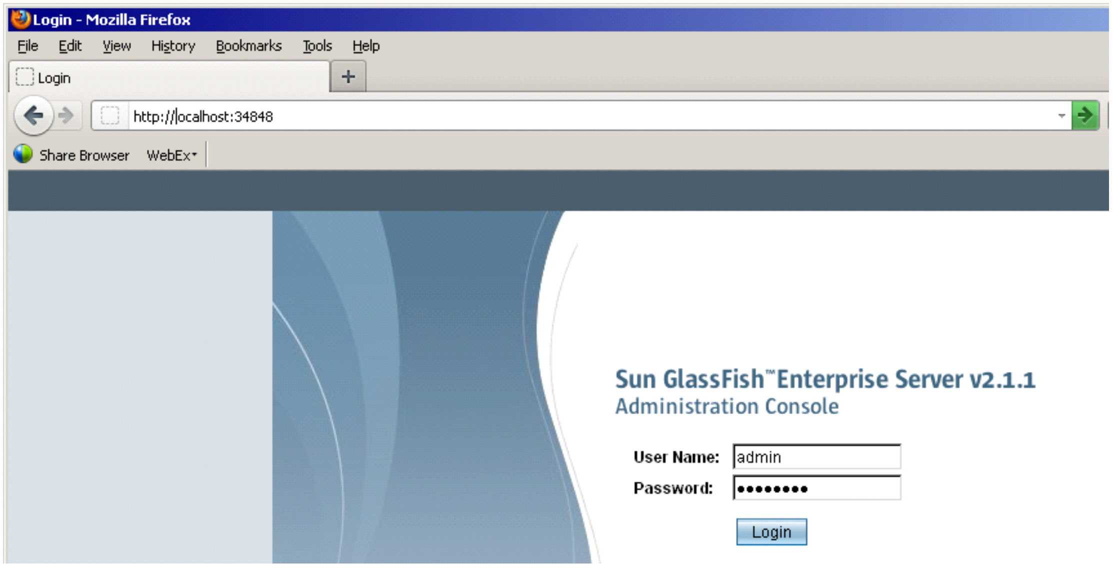
In this article I deal with JMS client access to the JMS destinations using the "com.sun.jndi.fscontext.RefFSContextFactory", to which references can be found on the Internet but which is not documented as well as I would have liked when I had a need to use this method.

Here I use Windows conventions for directory and file paths. For Unix, adjust as required. I assume that you have a GlassFish 2.x installation, perhaps as part of a Sun/Oracle Java CAPS SOA infrastructure or as part of the Oracle Healthcare Master Person Index infrastructure. I also assume that you need to create a Rhapsody integration solution using GlassFish environment-hosted JMS topics and/or queues. This last may or may not be your motivation. Except for the Rhapsody bits, the method should hold for any JMS client, but I have not tried this method in any other client deployment. Perhaps someday I will.

Creating JMS Objects on the GlassFish Side, if needed

If you don't have the JMS destination you need, and you need to create one, here are the steps describing the process.

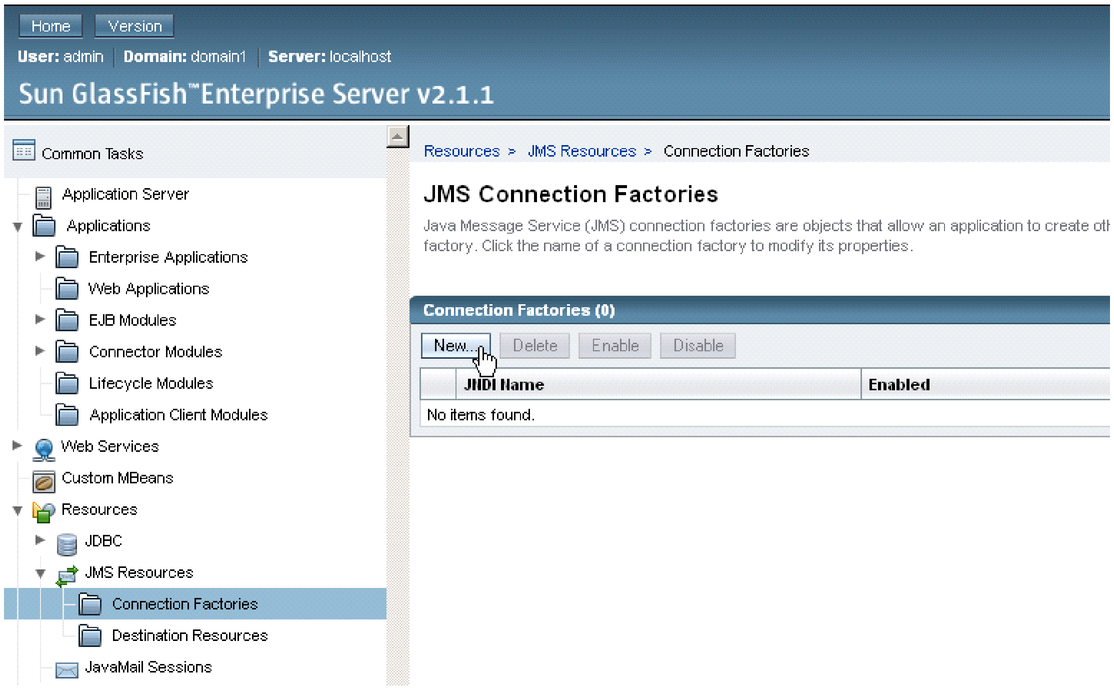
- ☐ Start the GlassFish Admin Console, typically on http://localhost:4848, and log in as user "admin" with password, by default "adminadmin".



User Name: admin
Password: ••••••••

Create JMS Connection Factory

- ☐ Expand "Resources"☐"JMS Resources" and click "Connection Factories"
- ☐ In the right-hand panel click "New"



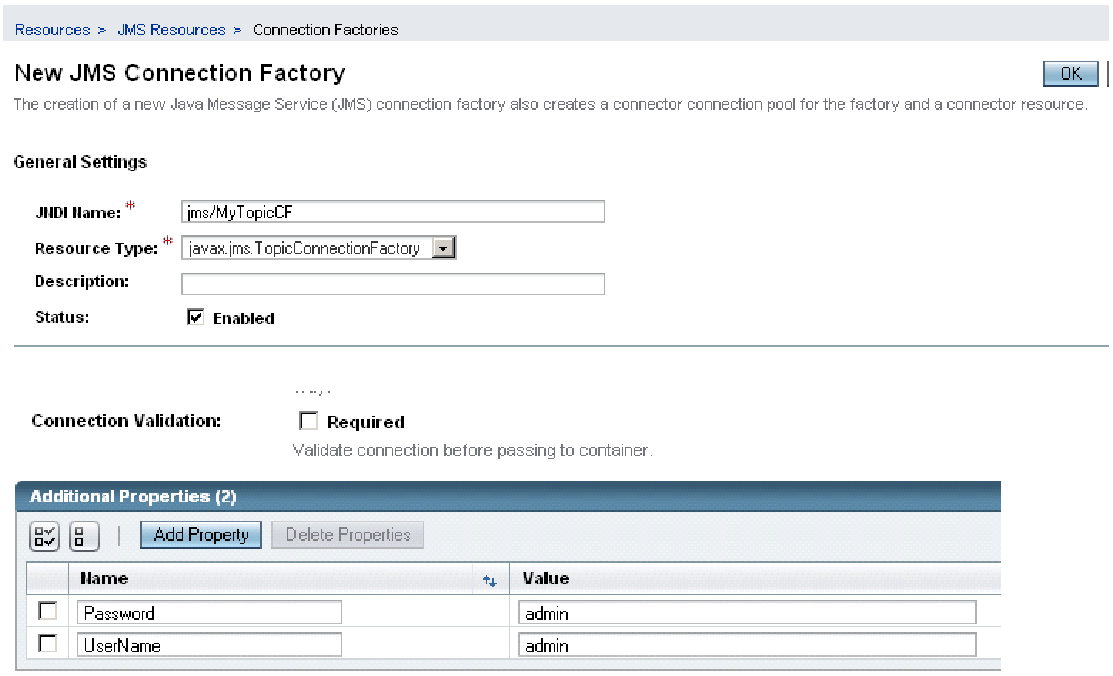
JMS Connection Factories

Java Message Service (JMS) connection factories are objects that allow an application to create ot factory. Click the name of a connection factory to modify its properties.

Connection Factories (0)

	JNDI Name	Enabled
No items found.		

- ☐ Enter "jms/MyTopicCF" as JNDI Name.
- ☐ Select "javax.jms.TopicConnectionFactory" as Resource Type
- ☐ Change the username and password in the "Additional Properties" to "admin" and "admin", assuming default configuration, then click "OK"


Resources > JMS Resources > Connection Factories

New JMS Connection Factory

The creation of a new Java Message Service (JMS) connection factory also creates a connector connection pool for the factory and a connector resource.

General Settings

JNDI Name: *	jms/MyTopicCF
Resource Type: *	javax.jms.TopicConnectionFactory
Description:	
Status:	☑ Enabled

Connection Validation: ☐ Required
Validate connection before passing to container.

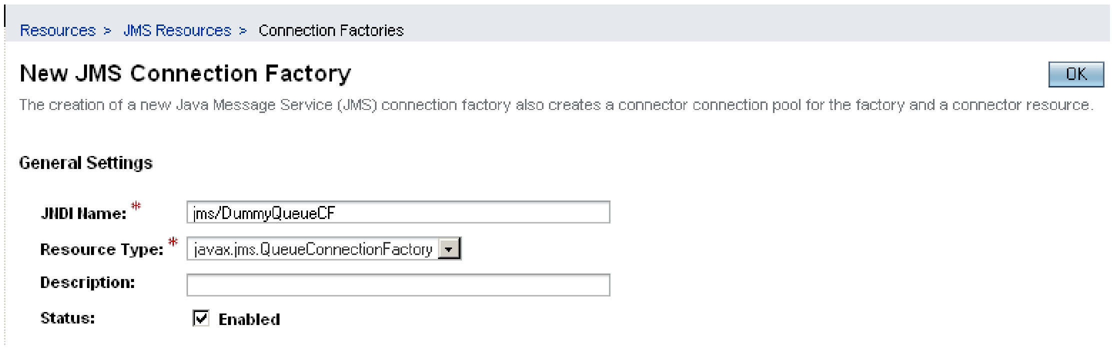
Additional Properties (2)

	Name	Value
☐	Password	admin
☐	UserName	admin

Create dummy Queue Connection Factory

The idiosyncratic behavior of the Rhapsody JMS Adapter configuration wizard requires that a Queue Connection Factory be provided even if the JMS destination is a Topic. This connection factory is not used at runtime but ...

- ☐ Click the "Connection Factory" node in the Resources"→"JMS Resources" tree and click "New"
- ☐ Name the factory "jms/DummyQueueCF", select "javax.jms.QueueConnectionFactory" as Resource Type, change the username and password in the "Additional Properties" to "admin" and "admin", assuming default configuration, then click "OK"


Resources > JMS Resources > Connection Factories

New JMS Connection Factory

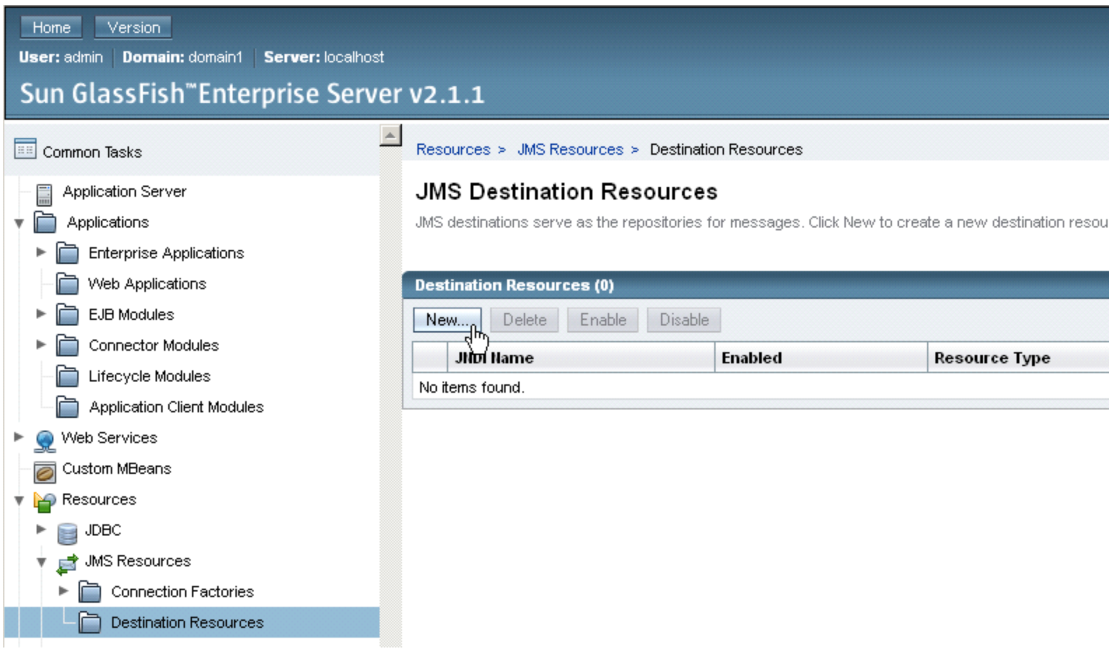
The creation of a new Java Message Service (JMS) connection factory also creates a connector connection pool for the factory and a connector resource.

General Settings

JNDI Name: *	jms/DummyQueueCF
Resource Type: *	javax.jms.QueueConnectionFactory
Description:	
Status:	☑ Enabled

Create JMS Topic

- ☐ Navigate "Resources"→"JMS Resources", click "Destination Resources" node and click "New"



User: admin | Domain: domain1 | Server: localhost

Sun GlassFish™ Enterprise Server v2.1.1

Resources > JMS Resources > Destination Resources

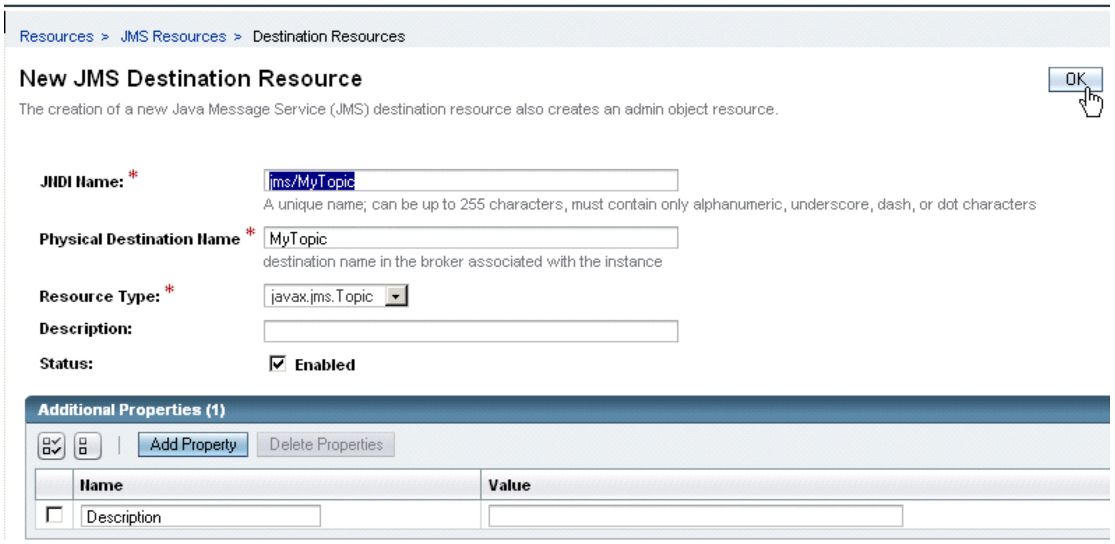
JMS Destination Resources

JMS destinations serve as the repositories for messages. Click New to create a new destination resou

Destination Resources (0)

JNDI Name	Enabled	Resource Type
No items found.		

☐ Set "JNDI Name" to "jms/MyTopic", "Physical Destination Name" to "MyTopic", "Resource Type" to "javax.jms.Topic" and click "OK"



Resources > JMS Resources > Destination Resources

New JMS Destination Resource

The creation of a new Java Message Service (JMS) destination resource also creates an admin object resource.

JNDI Name: * jms/MyTopic
A unique name; can be up to 255 characters, must contain only alphanumeric, underscore, dash, or dot characters

Physical Destination Name * MyTopic
destination name in the broker associated with the instance

Resource Type: * javax.jms.Topic

Description:

Status: ☑ Enabled

Additional Properties (1)

	Name	Value
☐	Description	

☐ Log out of the GlassFish Application Server Administration Console and close the browser window.

We have a JMS topic and a JMS Connection factory created. We can use them in subsequent discussion if we don't have others which to use.

The topic may not be immediately visible to the tooling. You may need to subscribe to it and unsubscribe from it, or submit a message to it using some tooling like QBrowser, in order for it to be visible to the imqadmin tool, discussed below.

On GlassFish-hosted SJSMQ Side

To configure Rhapsody, or another JMS Client, to use the "com.sun.jndi.fscontext.RefFSContextFactory" as the initial JNDI context, one must have a ".bindings" file which contains the necessary magic incantations. These incantations are created by the Sun Java System Message Queue imqamdin tool. The necessary steps are detailed below.

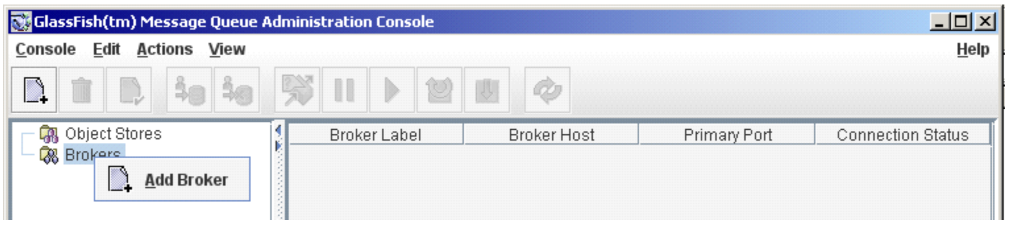
- ☐ Create a directory in which to place the Sun Java System Message Queue (SJSMQ) JMS Object Store - here {FullHomeDirectoryPath} is the full path to the user's home directory - you can specify some other directory where to keep the object store - change paths accordingly - mine is "c:¥jndi_store¥myJMSObjectStore"

```
mkdir {FullDirectoryPath}¥myJMSObjectStore
```

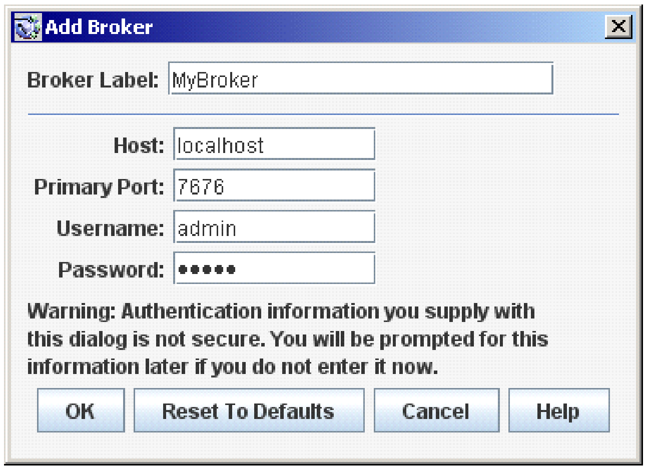
- ☐ Locate Sun Java System Message Queue (SJSMQ) binaries and run imqadmin - this needs a graphical environment if you are working on Unix

```
cd <GlassFishInstallRoot>¥imq¥bin
.\imqadmin
```

- ☐ If you don't see an existing broker then right-click on the "Brokers" node and select "Add Broker"



- ☐ Name the broker "MyBroker" (or whatever feels good to you), enter the password (admin), and click "OK" (assuming port number is the default 7676 - change as required in your environment)


Add Broker

Broker Label: MyBroker

Host: localhost

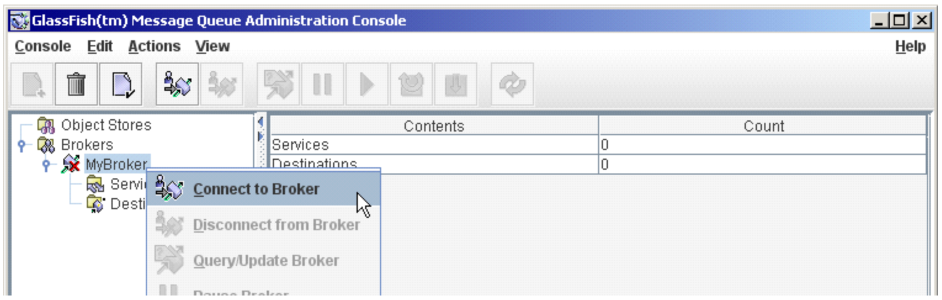
Primary Port: 7676

Username: admin

Password: •••••

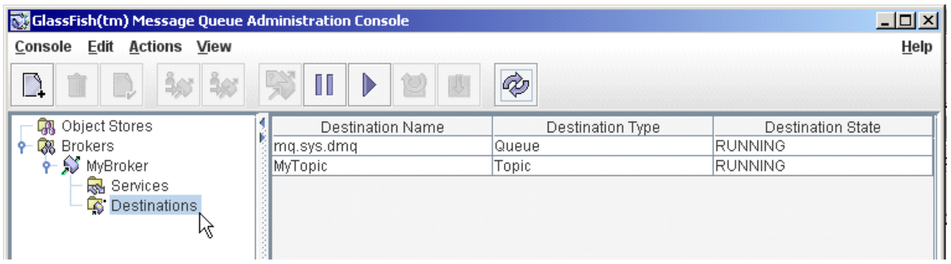
Warning: Authentication information you supply with this dialog is not secure. You will be prompted for this information later if you do not enter it now.

- ☐ Connect to the broker to discover what JMS destinations there are - you can create destinations using different tools - in this I assume that you need to provide a client access to an existing destination hence "connect and discover ..."



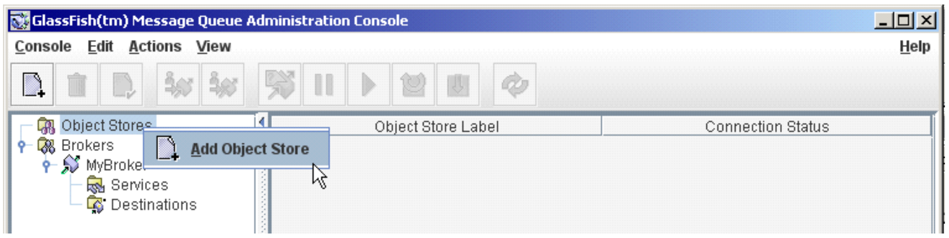
Contents	Count
Services	0
Destinations	0

- ☐ Click on the "Destinations" node to see what destinations there are and what type they are

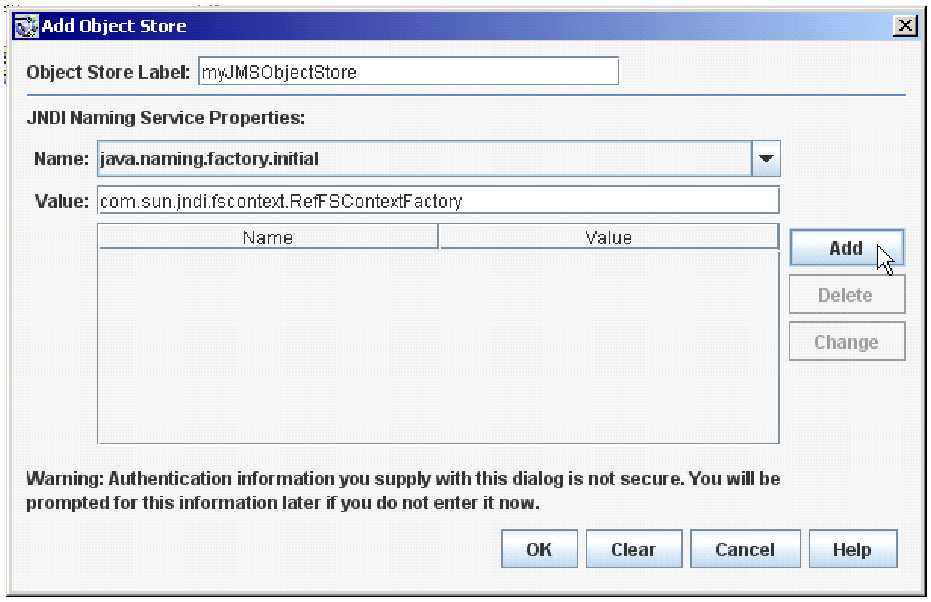

GlassFish(tm) Message Queue Administration Console
Console Edit Actions View
Help
Object Stores
Brokers
MyBroker
Services
Destinations

Destination Name	Destination Type	Destination State
mq.sys.dmq	Queue	RUNNING
MyTopic	Topic	RUNNING

- ☐ Take note of the names and types of destination(s) to which your client needs to connect
- ☐ Right-click the node called "ObjectStore" and choose "Add Object Store"



- ☐ Give the new object store the label of "myJMSObjectStore", like the directory name you created earlier - any name will do I am just trying to be consistent and help myself remember what is related to what.
- ☐ Enter "com.sun.jndi.fscontext.RefFSContextFactory" as the value of the "java.naming.factory.initial" property and click "Add"


Add Object Store

Object Store Label: myJMSObjectStore

JNDI Naming Service Properties:

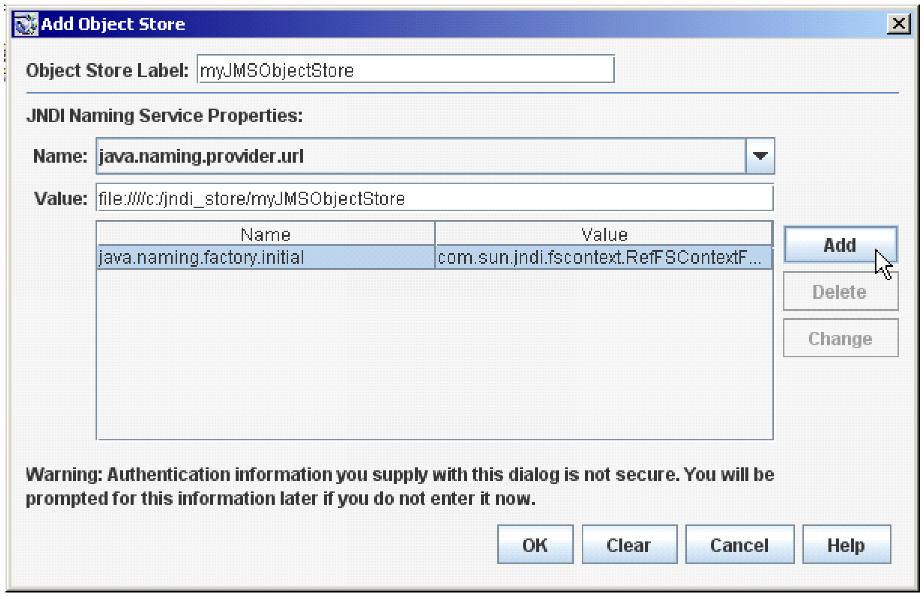
Name: java.naming.factory.initial

Value: com.sun.jndi.fscontext.RefFSContextFactory

Name	Value

Warning: Authentication information you supply with this dialog is not secure. You will be prompted for this information later if you do not enter it now.

- ☐ Pull down the "Name:" drop down and choose the "java.naming.provider.url" property. Enter "file:///c:/jndi_store/myJMSObjectStore" as the value of the "java.naming.provider.url" property, where and click "Add" - mind that you replace {FullHomeDirectoryPath}, including {}, with the full directory path - mind also that you have the correct number of forward slashes


Add Object Store

Object Store Label: myJMSObjectStore

JNDI Naming Service Properties:

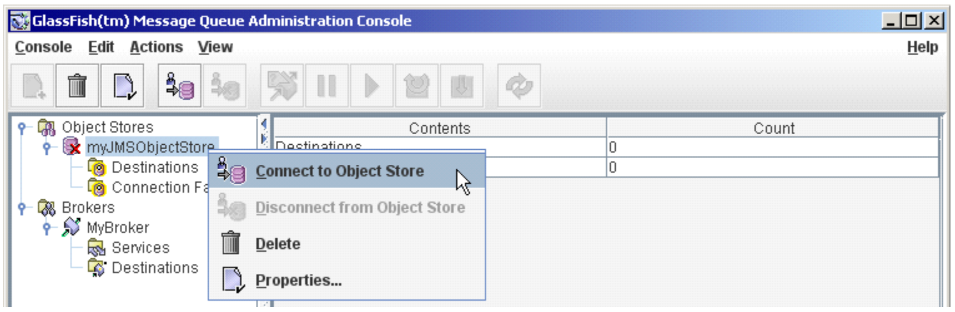
Name: java.naming.provider.url

Value: file:///c:/jndi_store/myJMSObjectStore

Name	Value
java.naming.factory.initial	com.sun.jndi.fscontext.RefFSContextF...

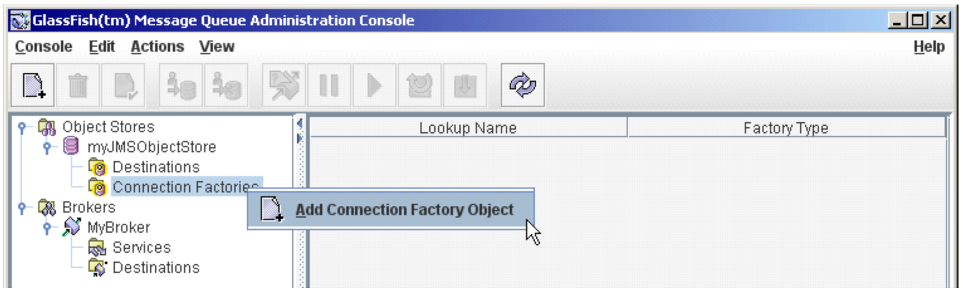
Warning: Authentication information you supply with this dialog is not secure. You will be prompted for this information later if you do not enter it now.

- ☐ Click "OK"
- ☐ Right-click on the name of the new object store and select "Connect to Object Store"

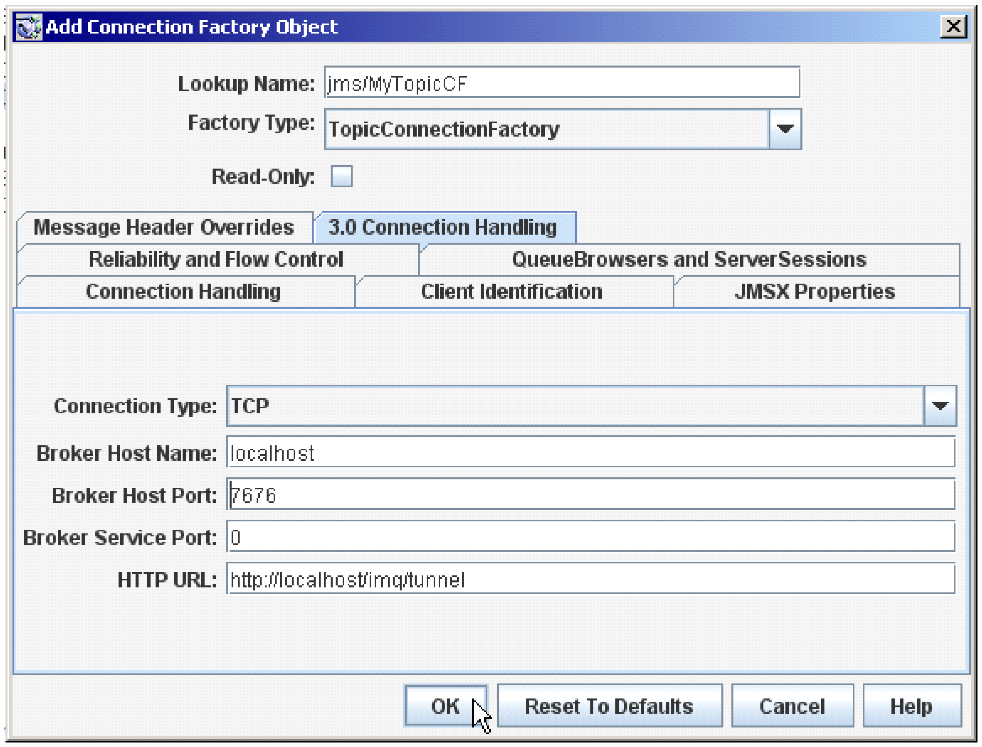


Note the change in icon - the red cross is gone.

- ☐ Double-click the name of the new object store to expand its node hierarchy, right-click the node "Connection Factories" in the new object store node tree and choose "Add Connection Factory Object"



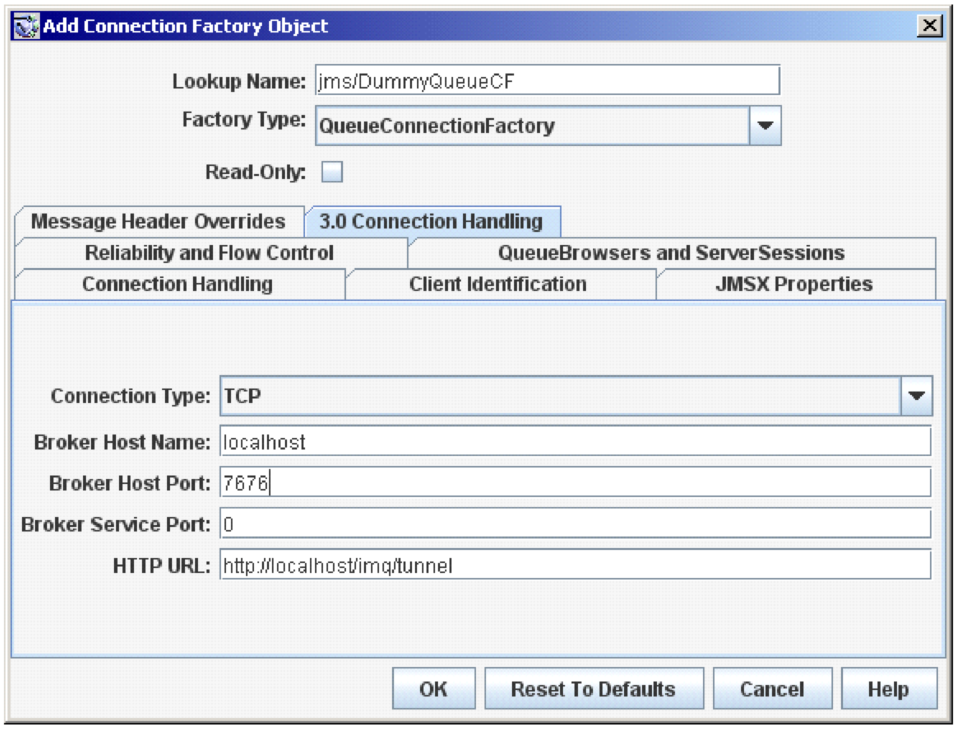
- ☐ Set the "Lookup Name" to "jms/MyTopicCF" and choose a "Factory Type" as "TopicConnectionFactory"
- ☐ Click the "3.0 Connection Handling" Tab
- ☐ Set the following properties to the following values:
    - Broker Host Name: localhost, or a host name appropriate for your environment, which the JMS client can resolve
    - Broker Host Port: "7676", a default,, or an actual port number if different from the default
- ☐ Click "OK" when done



Lookup Name: jms/MyTopicCF
Factory Type: TopicConnectionFactory
Read-Only: ☐

Connection Type: TCP
Broker Host Name: localhost
Broker Host Port: 7676
Broker Service Port: 0
HTTP URL: http://localhost/imq/tunnel

- ☐ Right-click the node "Connection Factories" in the object store node tree and choose "Add Connection Factory Object" to add another connection factory
- ☐ Set the "Lookup Name" to "jms/DummyQueueCF" and choose a "Factory Type" as "QueueConnectionFactory"
- ☐ Click the "3.0 Connection Handling" Tab
- ☐ Set the following properties to the following values:
    - Broker Host Name: localhost, or a host name appropriate for your environment, which the JMS client can resolve
    - Broker Host Port: "7676", a default,, or an actual port number if different from the default
- ☐ Click "OK" when done


Add Connection Factory Object

Lookup Name: jms/DummyQueueCF

Factory Type: QueueConnectionFactory

Read-Only: ☐

Connection Type: TCP

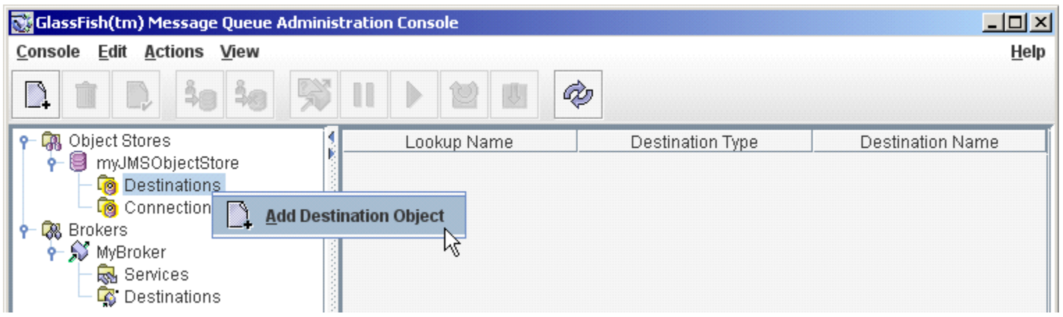
Broker Host Name: localhost

Broker Host Port: 7676

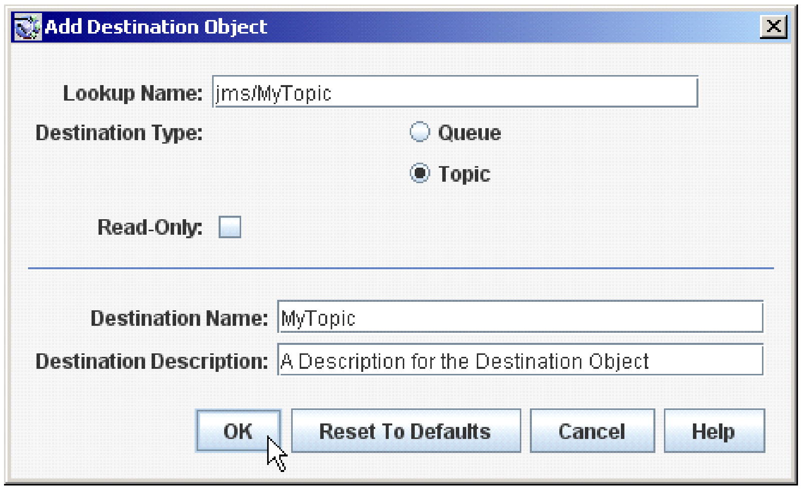
Broker Service Port: 0

HTTP URL: http://localhost/imq/tunnel

- ☐ Click "Destinations" node under the "MyJMSObjectStore" node and choose "Add Destination Object"



- ☐ Enter "jms/MyTopic" as "Lookup Name" property value, click the "Topic" radio button to designate this object as a JMS Topic, set the value of the "Destination Name" to "MyTopic" and click "OK"



Lookup Name: jms/MyTopic

Destination Type: Queue [✓ Topic]

Read-Only: ☐

Destination Name: MyTopic

Destination Description: A Description for the Destination Object

- ☐ Click "Console"-->"Exit" to exit the JMS Admin console.

Have a look at the content of the object store directory, `{FullDirectoryPath}\myJMSObjectStore`. The file ".bindings" in that directory is what we need on the client side to facilitate access to JMS respurces using the "com.sun.jndi.fscontext.RefFSContextFactory" initial JNDI context. It is a text file - feel free to open it with a text editor and look at the content. Don't change it at this point.

We are done configuring the Object Store. We will copy this object store to the Rhapsody side, where we will use it to configure the Rhpasody JMS Adapter.

On the Rhapsody Side

If the JMS Client Application which needs to communicate with the GlassFish-bundled JMS is the Rhapsody 4.01 then its JMS Adapter must be correctly configured to support this communication, including making the appropriate JARs and a ".bindings" file available to the JMS Adapter. The following discuss the process.

On the Rhapsody side one must have the .bindings file which defines the magic incantations necessary for the "com.sun.jndi.fscontext.RefFSContextFactory" initial JNDI context to locate the JMS destinaiton and connect to it, and the required SJSMQ libraries. Once these are available one can configure the Rhapsody JMS Adapter and connect to the JMS destination and exchange messages with it.

Copy SJSMQ objects to Rhapsody environment

The ".bindings" file, created in the GlassFish environment, must be accessible to the Rhapsody host where Rhapsody IDE runs. It is necessary to configure the JMS Adapter. Copy this file to a directory of your choosing, say "C:\jndi_store".

From the SJSMQ host's GlassFish imq/lib directory copy the following files to a directory of you choosing on the Rhapsody host, say "c:\galssfish_jms_libs"

- fscontext.jar
- imq.jar

- o jms.jar

From the GlassFish host's lib directory copy the following file to a directory of you choosing on the Rhapsody host, say "c:\galssfish_jms_libs"

- o j2ee.jar

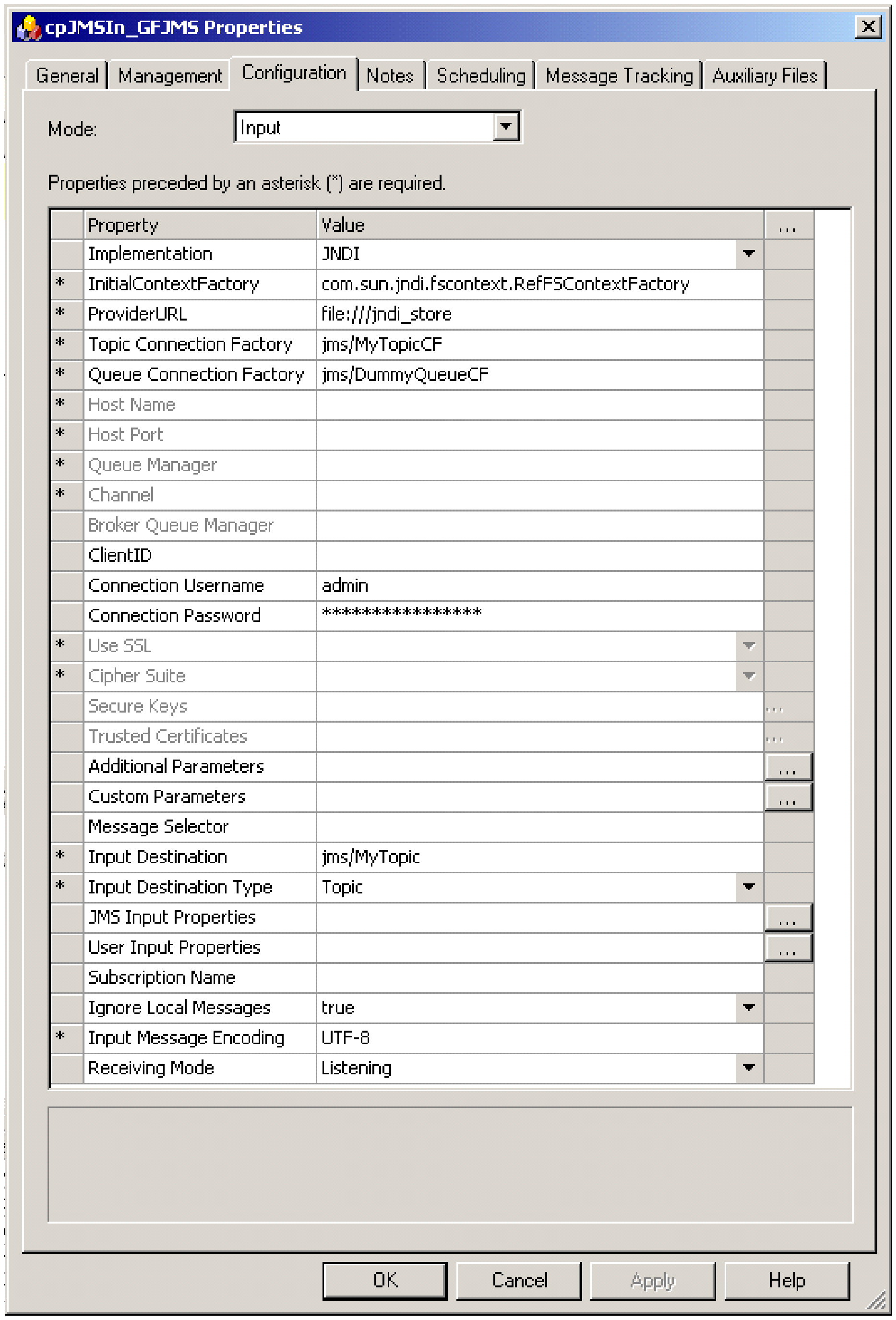
Configure Rhapsody JMS Adapter

Here are the key configuration options for Rhapsody 4.01 and their settings.

Property Name	Property Value
InitialContextFactory	com.sun.jndi.fscontext.RefFSContextFactory
ProviderURL	file:///jndi_store
Topic Connection Factory	jms/{JMSTopicCopnnectionFactory}
Queue Connection Factory	jms/{JMSQueueConnectoinFactory}
Connection Username	{JMSAdminUsername}
Connection Password	{JMSAdminPassword}
Input Destination	jms/{MyTopicOrQueueName}
Input Destination Type	Topic (or Queue, depending on what you use)
Receiving Mode	Listening

In the table above values in {}, including {} themselves, must be replaced with the actual names from your configuration.

Rhapsody JMS Adapter configuration panel has an annoying idiosyncrasy where one must specify a Queue Connection Factory even if one uses a JMS topic. Without this the configuration cannot be saved.



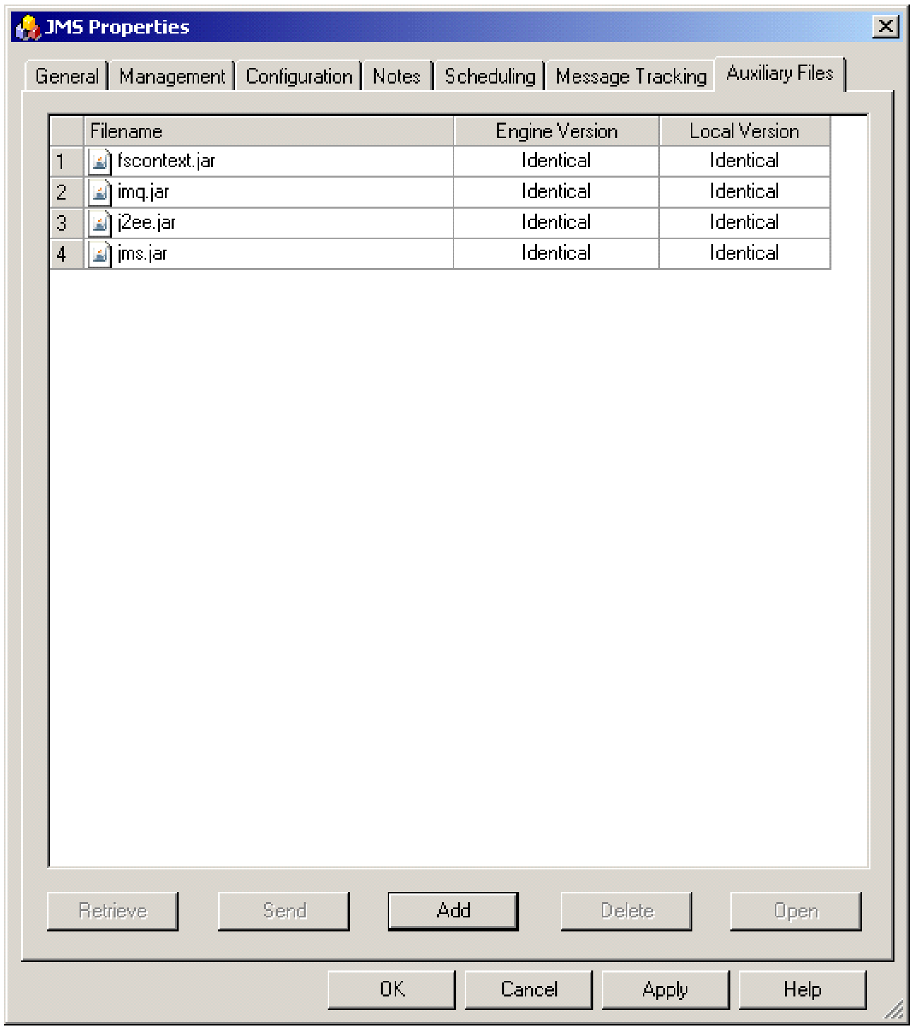
cpJMSIn_GFJMS Properties
General | Management | Configuration | Notes | Scheduling | Message Tracking | Auxiliary Files
OK | Cancel | Apply | Help

Mode: Input

Properties preceded by an asterisk (*) are required.

	Property	Value
	Implementation	JNDI
*	InitialContextFactory	com.sun.jndi.fscontext.RefFSContextFactory
*	ProviderURL	file:///jndi_store
*	Topic Connection Factory	jms/MyTopicCF
*	Queue Connection Factory	jms/DummyQueueCF
*	Host Name	
*	Host Port	
*	Queue Manager	
*	Channel	
	Broker Queue Manager	
	ClientID	
	Connection Username	admin
	Connection Password	******************
*	Use SSL	
*	Cipher Suite	
	Secure Keys	
	Trusted Certificates	
	Additional Parameters	
	Custom Parameters	
	Message Selector	
*	Input Destination	jms/MyTopic
*	Input Destination Type	Topic
	JMS Input Properties	
	User Input Properties	
	Subscription Name	
	Ignore Local Messages	true
*	Input Message Encoding	UTF-8
	Receiving Mode	Listening

Add all archives to Auxiliary Files for the JMS Communication Point.



	Filename	Engine Version	Local Version
1	fscontext.jar	Identical	Identical
2	imq.jar	Identical	Identical
3	j2ee.jar	Identical	Identical
4	jms.jar	Identical	Identical

Create a route with whatever additional communication points you might need and start the lot. Expect, if all is configured correctly, to receive messages from the topic hosted by the GlassFish JMS.

For the JMS objects, whose creation on the GlassFish Server side was discussed earlier, the settings would be:

Property Name	Property Value
InitialContextFactory	com.sun.jndi.fscontext.RefFSContextFactory
ProviderURL	file:///c:/jndi_store/myJMSObjectStore
Topic Connection Factory	jms/MyTopicCF
Queue Connection Factory	jms/DummyQueueCF
Connection Username	admin

Connection Password	admin
Input Destination	jms/MyTopic
Input Destination Type	Topic
Receiving Mode	Listening

Summary

In this article I walked through the process of setting up JMS Topic and its dependencies on the GlassFish v2.x platform and configuring Rhapsody 4.01 JMS Communication Point to receive messages from the JMS Topic hosted by the GlassFish v2-bundled Sun Java System Message Queue JMS Server. Perhaps this will save you the time I spent figuring out how to do this.