

## Configuring and running simple JMS P2P and Pub/Sub applications in MQ 7.0, 7.1, 7.5 and 8.0

IBM Techdoc: 7023212

<http://www.ibm.com/support/docview.wss?rs=171&uid=swg27023212>

Date last updated: 13-Aug-2015

Angel Rivera - [rivera@us.ibm.com](mailto:rivera@us.ibm.com)  
IBM WebSphere MQ Support

### +++ Objective

The objective of this document is to provide step-by-step instructions for configuring and running simple JMS Point-to-Point (P2P) and Pub/Sub applications in WebSphere MQ V7 for UNIX and Windows.

MQ 7.0.1 was used to run the examples in this techdoc. When using MQ 7.1, 7.5, 8.0 or later, the new command "setmqenv" (introduced with MQ 7.1) needs to be used during the initial setup. The following technote has the complete list of steps customized for using MQ 7.1 or later:

<http://www-01.ibm.com/support/docview.wss?uid=swg21614256>

Detailed example of running JMSAdmin on MQ 7.1, 7.5 and 8.0

In UNIX, the JMSAdmin tool is used to populate the Java Naming and Directory Interface (JNDI) with the JMS Administrative objects that serve as a link between the JMS application and the physical queues and topics in the queue manager. This JNDI is in the form of a file named ".bindings".

In Windows and in Linux x86, in addition to the JMSAdmin tool, the GUI MQ Explorer tool can be used to populate the JNDI into a file named ".bindings".

This techdoc has the following chapters:

Chapter 1:

- The necessary customization for the environment in order to run JMS programs in UNIX
- The customization of JMSAdmin.config and how to run the JMSAdmin tool in UNIX.
- How to create a Connection Factory, a Queue and a Topic object into the JNDI via the JMSAdmin tool.

Chapter 2:

- The necessary customization for the environment in order to run JMS programs in Windows.
- The customization of JMSAdmin.config and how to run the JMSAdmin tool in Windows.
- How to create a Connection Factory, a Queue and a Topic object into the JNDI via the JMSAdmin tool

Chapter 3:

- How to create a Connection Factory, a Queue and a Topic object into the JNDI via the MQ Explorer in Windows.

Chapter 4:

- How to run in UNIX, 2 excellent samples provided with MQ V7 that show how to use the JNDI to do Point-to-Point (queue based) messaging, and how to do Pub/Sub (topic based).

Complete runs for the scenarios are included.

- These 2 samples are located at:

UNIX: /opt/mqm/samp/jms/samples

Windows: C:\Program Files\IBM\WebSphere MQ\tools\jms\samples

JmsJndiConsumer.java

JmsJndiProducer.java

These samples are provided in source Java code and in compiled format. The source files have good explanation of the code. The samples use: IBM JMS API (v1.1, unified domain).

MQ also ships 2 other samples that do not use the JNDI. They are not discussed in this techdoc, but are mentioned in case that you want to review and use them:

JmsConsumer.java

JmsProducer.java

Chapter 5:

- Similar to Chapter 4, but for Windows.

Chapter 6:

- How to troubleshoot common problems during setup and runtime.

+ Software used:

WebSphere MQ V7.0.1.3 in Solaris 10

- Java 1.5

WebSphere MQ V7.0.1.6 in Linux x86-32bit

- Java 1.6

WebSphere MQ V7.0.1.6 in Windows XP

- Java 1.6

```

+++++
+++ Chapter 1: Configuration in UNIX via JMSAdmin
+++++

```

It is necessary to use the JMSAdmin tool to create, view or alter the JNDI.

The JMSAdmin tool is located in /opt/mqm/java/bin (non-AIX) or /usr/mqm/java/bin (AIX)

a) Modify your profile to facilitate the running MQ tools and samples:

a.1) Add the following MQ directories into the PATH in your .profile (or .bashrc in Linux):

```

## Non-AIX:
export MQ_ROOT=/opt/mqm
export PATH=$PATH:$MQ_ROOT/bin:$MQ_ROOT/java/bin:$MQ_ROOT/samp/bin:
$MQ_ROOT/samp/jms/samples

```

```

## AIX:
export MQ_ROOT=/usr/mqm
export PATH=$PATH:$MQ_ROOT/bin:$MQ_ROOT/java/bin:$MQ_ROOT/samp/bin:
$MQ_ROOT/samp/jms/samples

```

a.2) Run the MQ script that sets the proper JMS and JAVA environment variables and CLASSPATH for MQ.

This can be accomplished by adding the following into the .profile (or .bashrc in Linux):

+ begin quote

```

## Customization for WMQ
if [ -f $MQ_ROOT/java/bin/setjmsenv64 ]
then
. setjmsenv64
else
. setjmsenv
fi
+ end quote

```

For running 32-bit applications (for example, under Linux X86 32-bit), run:

```

. setjmsenv

```

For running 64-bit applications (for other platforms, such as AIX, Solaris, HP-UX, Linux 64-bit), run the following script that is only provided if your platform supports 64-bit:

```
. setjmsenv64
```

You must type the dot, then space, then "setjmsenv" or "setjmsenv64".

An example of running it is shown below for Linux x86 32-bit:

```
rivera@veracruz: /home/rivera
$ . setjmsenv
MQ_JAVA_INSTALL_PATH is /opt/mqm/java
MQ_JAVA_DATA_PATH is /var/mqm
MQ_JAVA_LIB_PATH is /opt/mqm/java/lib
CLASSPATH is
:/opt/mqm/java/lib/com.ibm.mq.jar:/opt/mqm/java/lib/com.ibm.mqjms.jar:/opt/
mqm/samp/jms/samples:/opt/mqm/samp/wmqjava/samples
```

a.3) Expand the CLASSPATH in your .profile or .bashrc:

```
## Need to manually add the following to CLASSPATH to compile JMS programs
export CLASSPATH=$CLASSPATH:/opt/mqm/java/lib/jms.jar
```

b) Creation of physical objects (Queue and Topic) in the queue manager

The following were used in Linux:

```
Queue Manager name: QM_VER
Host: veracruz.x.com
Port: 1414
```

```
$ runmqsc QM_VER
DEFINE QL(Q2)
DEFINE TOPIC(T2) TOPICSTR('TOPIC2')
END
```

c) Create a directory where the JMS configuration objects will be located in a file name ".bindings".

For example, in UNIX, you can create the following subdirectory in the same directory where the other MQ objects are stored:

```
mkdir /var/mqm/JNDI-Directory
```

d) Copy the JMSAdmin.config file from its default location, into /var/mqm

The default file in Solaris, Linux and HP-UX is:

```
/opt/mqm/java/bin/JMSAdmin.config
```

The default file in AIX is:

```
/usr/mqm/java/bin/JMSAdmin.config
```

In this document we follow the best practice of not writing files under /opt/mqm (/usr/mqm). Thus, you need to copy the JMSAdmin.config file:

```
cp /opt/mqm/java/bin/JMSAdmin.config /var/mqm/JMSAdmin.config
```

Ensure that you can write into the file (the original file is read-only):

```
chmod 644 /var/mqm/JMSAdmin.config
```

e) Modify /var/mqm/JMSAdmin.config

These 2 variables need to be properly specified in the config file.

e.1) INITIAL\_CONTEXT\_FACTORY

The following is common for UNIX and Windows and it indicates that a file will be used:

```
INITIAL_CONTEXT_FACTORY=com.sun.jndi.fscontext.RefFSContextFactory
```

e.2) PROVIDER\_URL

For UNIX: notice the 3 forward slashes after "file:".

```
PROVIDER_URL=file:///var/mqm/JNDI-Directory
```

Note: If there are not 3 slashes, such as 2, 4, 5, etc. then the following error is displayed by JMSAdmin at runtime:

```
InitCtx> DEFINE TCF(testCF)
```

```
Unable to bind object
```

For Windows: notice only 1 forward slash after "file:" and notice the drive letter "C:/"

```
PROVIDER_URL=file:/C:/JNDI-Directory
```

Notice that ONLY one INITIAL\_CONTEXT\_FACTORY and PROVIDER\_URL must be uncommented. If there are more uncommented lines, then the last entry is the winner, and overwrites any earlier entries of the same type.

f) Create a shell script called "myJMSAdmin.sh" that has these lines:

```
<beginning of script>
#!/usr/bin/ksh
echo "running: JMSAdmin -cfg /var/mqm/JMSAdmin.config"

if [ -f /opt/mqm/java/bin/setjmsenv64 ]
then
. setjmsenv64
else
. setjmsenv
fi

JMSAdmin -cfg /var/mqm/JMSAdmin.config
<end of script>
```

This script will invoke the JMSAdmin tool using the JMSAdmin.config that was copied and modified under /var/mqm.

Ensure that the myJMSAdmin.sh script is in a directory under \$PATH.

Ensure that the script is executable:

```
chmod 755 myJMSAdmin.sh
```

g) Start JMSAdmin by running the script - notice the prompt: InitCtx>

```
rivera@veracruz: /home/rivera
$ myJMSAdmin.sh
```

```
+ begin quote to show a run of the tool
running: JMSAdmin -cfg /var/mqm/JMSAdmin.config
MQ_JAVA_INSTALL_PATH is /opt/mqm/java
```

```
MQ_JAVA_DATA_PATH is /var/mqm
MQ_JAVA_LIB_PATH is /opt/mqm/java/lib
CLASSPATH is
:/opt/mqm/java/lib/com.ibm.mq.jar:/opt/mqm/java/lib/com.ibm.mqjms.jar:/opt/
mqm/samp/jms/samples:/opt/mqm/samp/wmqjava/samples
```

Licensed Materials - Property of IBM

5724-H72, 5655-R36, 5724-L26, 5655-L82

(c) Copyright IBM Corp. 2008 All Rights Reserved.

US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Starting Websphere MQ classes for Java(tm) Message Service Administration

```
InitCtx>
```

+ end quote

Notice the prompt for the tool:

```
InitCtx>
```

It means: Initial Context

h) Creation of JMS Administrative objects

For a complete list of the properties for the JNDI objects, see:

<http://publib.boulder.ibm.com/infocenter/wmqv7/v7r0/index.jsp?topic=%2Fcom.ibm.mq.csqzaw.doc%2Fjm10910.htm>

WebSphere MQ > Using Java > WebSphere MQ classes for JMS

Properties of WebSphere MQ classes for JMS objects

```
# Define Connection Factory called CF1Reconnect
```

```
# Suggestion: Do not specify a value for QMGR to provide more flexibility. If you specify a name, then only when the queue manager is named the same (such as in a multi-instance queue manager) the reconnection will work.
```

```
InitCtx> DEF CF(CF1) QMGR(QM_VER) TRANSPORT(CLIENT) HOSTNAME(veracruz.x.com)
PORT(1414)
```

```
# Define Queue Q1
```

```
InitCtx> DEF Q(Q1) QMGR(QM_VER) QUEUE(Q1)
```



# Define Topic T1

InitCtx> DEF T(T1) TOPIC('TOPIC1')

# Display all items in the Context:

InitCtx> DIS CTX

JMSADM4089 InitCtx

.bindings	java.io.File
a T1	com.ibm.mq.jms.MQTopic
a CF1	com.ibm.mq.jms.MQConnectionFactory
a Q1	com.ibm.mq.jms.MQQueue

4 Object(s)

0 Context(s)

4 Binding(s), 3 Administered

# Display Connection Factory CF1

InitCtx> DIS CF(CF1)

ASYNCEXCEPTION(ALL)  
 BROKERCCSUBQ(SYSTEM.JMS.ND.CC.SUBSCRIBER.QUEUE)  
 BROKERCONQ(SYSTEM.BROKER.CONTROL.QUEUE)  
 BROKERPUBQ(SYSTEM.BROKER.DEFAULT.STREAM)  
 BROKERQMGR()  
 BROKERSUBQ(SYSTEM.JMS.ND.SUBSCRIBER.QUEUE)  
 BROKERVER(UNSPECIFIED)  
 CCSID(819)  
 CHANNEL(SYSTEM.DEF.SVRCONN)  
 CLEANUP(SAFE)  
 CLEANUPINT(3600000)  
 CLIENTRECONNECTOPTIONS(ASDEF)  
 CLIENTRECONNECTTIMEOUT(1800)  
 CLONESUPP(DISABLED)  
 COMPHDR(NONE )  
 COMPMSG(NONE )  
 CONNECTIONNAMELIST(veracruz.x.com(1414))  
 CONNOPT(STANDARD)  
 FAILIFQUIESCE(YES)  
 HOSTNAME(veracruz.x.com)

LOCALADDRESS()  
MAPNAMESTYLE(STANDARD)  
MSGBATCHSZ(10)  
MSGRETENTION(YES)  
MSGSELECTION(CLIENT)  
OPTIMISTICPUBLICATION(NO)  
OUTCOMENOTIFICATION(YES)  
POLLINGINT(5000)  
PORT(1414)  
PROCESSDURATION(UNKNOWN)  
PROVIDERVERSION(UNSPECIFIED)  
PUBACKINT(25)  
QMANAGER(QM\_VER)  
RECEIVEISOLATION(COMMITTED)  
RESCANINT(5000)  
SENDCHECKCOUNT(0)  
SHARECONVALLOWED(YES)  
SPARSESUBS(NO)  
SSLFIPSREQUIRED(NO)  
SSLRESETCOUNT(0)  
STATREFRESHINT(60000)  
SUBSTORE(BROKER)  
SYNCPOINTALLGETS(NO)  
TARGCLIENTMATCHING(YES)  
TEMPMODEL(SYSTEM.DEFAULT.MODEL.QUEUE)  
TEMPQPREFIX()  
TEMPTOPICPREFIX()  
TRANSPORT(CLIENT)  
USECONNPOOLING(YES)  
VERSION(7)  
WILDCARDFORMAT(TOPIC\_ONLY)

# Display Queue Q1

InitCtx> DIS Q(Q1)

CCSID(1208)  
ENCODING(NATIVE)  
EXPIRY(APP)  
FAILIFQUIESCE(YES)  
MDMSGCTX(DEFAULT)  
MDREAD(NO)  
MDWRITE(NO)

MSGBODY(UNSPECIFIED)  
PERSISTENCE(APP)  
PRIORITY(APP)  
PUTASYNCALLOWED(AS\_DEST)  
QMANAGER(QM\_VER)  
QUEUE(Q1)  
READAHEADALLOWED(AS\_DEST)  
READAHEADCLOSEPOLICY(DELIVER\_ALL)  
RECEIVECCSID(1208)  
RECEIVECONVERSION(CLIENT\_MSG)  
REPLYTOSTYLE(DEFAULT)  
TARGCLIENT(JMS)  
VERSION(7)

# Display Topic T1

InitCtx> DIS T(T1)

BROKERCCDURSUBQ(SYSTEM.JMS.D.CC.SUBSCRIBER.QUEUE)  
BROKERDURSUBQ(SYSTEM.JMS.D.SUBSCRIBER.QUEUE)  
BROKERPUBQ()  
BROKERPUBQMGR()  
BROKERVER(V1)  
CCSID(1208)  
ENCODING(NATIVE)  
EXPIRY(APP)  
FAILIFQUIESCE(YES)  
MDMSGCTX(DEFAULT)  
MDREAD(NO)  
MDWRITE(NO)  
MSGBODY(UNSPECIFIED)  
MULTICAST(ASCF)  
PERSISTENCE(APP)  
PRIORITY(APP)  
PUTASYNCALLOWED(AS\_DEST)  
READAHEADALLOWED(AS\_DEST)  
READAHEADCLOSEPOLICY(DELIVER\_ALL)  
RECEIVECCSID(1208)  
RECEIVECONVERSION(CLIENT\_MSG)  
REPLYTOSTYLE(DEFAULT)  
TARGCLIENT(JMS)  
TOPIC('TOPIC1')  
VERSION(7)

WILDCARDFORMAT(TOPIC\_ONLY)

# Exit JMSAdmin

InitCtx> end

Stopping Websphere MQ classes for Java(tm) Message Service Administration

+++++  
+++ Chapter 2: Configuration in Windows via JMSAdmin and MQ Explorer  
+++++

++ Section: JMSAdmin

Only the specific details for Windows are covered here.  
For the common details, see the usage notes in Chapter 1.

a) Create a directory where the JMS configuration objects will be located in a file name ".bindings".

For example, in Windows, you can create the following subdirectory in the same directory where the other MQ objects are stored:

```
mkdir c:\var\mqm\JNDI-Directory
```

b) Copy the JMSAdmin.config file from its default location into c:\var\mqm

The default file in Windows is located at:

```
C:\Program Files\IBM\WebSphere MQ\java\bin\JMSAdmin.config
```

In this document we follow the best practice of not writing files under the directory structure that as the MQ runtime code. Thus, you need to copy the JMSAdmin.config file:

```
Copy "C:\Program Files\IBM\WebSphere MQ\java\bin\JMSAdmin.config"  
C:\var\mqm\JMSAdmin.config
```

c) Modify C:\var\mqm\JMSAdmin.config

These 2 variables need to be properly specified in the config file.

c.1) INITIAL\_CONTEXT\_FACTORY

The following is common for UNIX and Windows and it indicates that a file will be used:

```
INITIAL_CONTEXT_FACTORY=com.sun.jndi.fscontext.RefFSContextFactory
```

c.2) PROVIDER\_URL

For Windows: notice only 1 forward slash after file: and notice the drive letter C:/

```
PROVIDER_URL=file:/C:/var/mqm/JNDI-Directory
```

For UNIX (notice the 3 forward slashes after file:):

```
PROVIDER_URL=file:///var/mqm/JNDI-Directory
```

Note: If there are not 3 slashes, such as 2, 4, 5, etc. then the following error is displayed by JMSAdmin at runtime:

```
InitCtx> DEFINE TCF(testCF)
Unable to bind object
```

Notice that ONLY one INITIAL\_CONTEXT\_FACTORY and PROVIDER\_URL must be uncommented. If there are more uncommented lines, then the last entry is the winner, and overwrites any earlier entries of the same type.

d) Create a batch file called "myJMSAdmin.bat" that has this one line:

```
+ begin
```

```
"%MQ_JAVA_INSTALL_PATH%\bin\JMSAdmin" -cfg C:\var\mqm\JMSAdmin.config
```

```
+ end
```

This batch file will invoke the JMSAdmin tool using the JMSAdmin.config that was copied and modified under C:\var\mqm.

Ensure that the myJMSAdmin.bat is in a directory under PATH.

e) Start the myJMSAdmin batch file:

```
C:\> myJMSAdmin
```

You will see the following. Notice the prompt: InitCtx>

```
Licensed Materials - Property of IBM
```

```
5724-H72, 5655-R36, 5724-L26, 5655-L82
```

```
(c) Copyright IBM Corp. 2008 All Rights Reserved.
```

```
US Government Users Restricted Rights - Use, duplication or  
disclosure restricted by GSA ADP Schedule Contract with  
IBM Corp.
```

```
Starting Websphere MQ classes for Java(tm) Message Service Administration
```

InitCtx>

f) To create the JMS administrative objects, see the corresponding steps in Chapter 1.

g) Creation of physical objects (Queue and Topic) in the queue manager

The following were used in Windows:

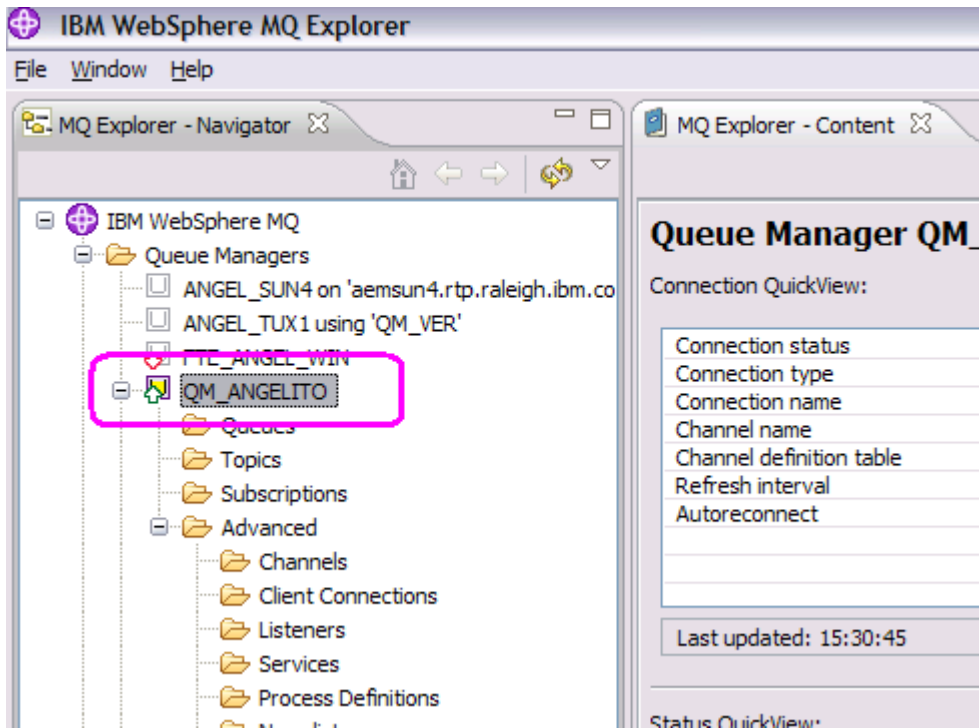
Queue Manager name: QM\_ANGELITO  
Host: angelito.x.com  
Port: 1414

```
$ runmqsc QM_ANGELITO  
define ql(Q1)  
define topic(T1) topicstr('TOPIC1')  
end
```

+++++  
+++ Chapter 3: Configuration in Windows via MQ Explorer  
+++++

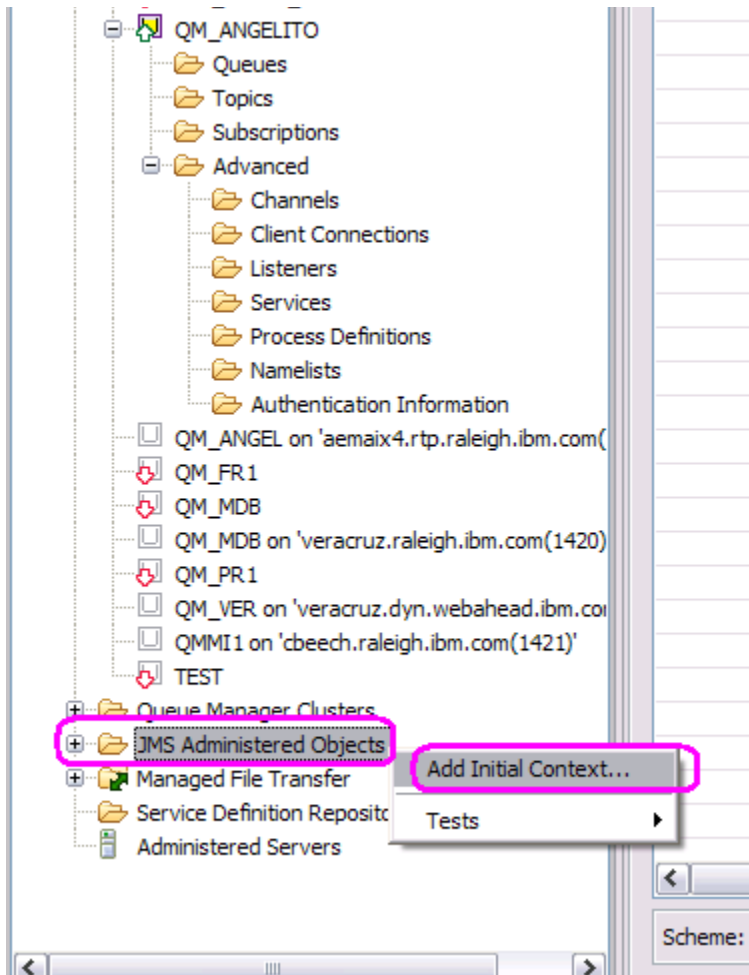
This chapter describes how to create a Connection Factory, a Queue and a Topic object into the JNDI via the MQ Explorer in Windows.

Start the MQ Explorer and the desired queue manager.  
In this example, the queue manager is called: QM\_ANGELITO





Scroll down to the bottom and select: JMS Administered Objects  
Do right Click and select "Add Initial Context ..."



In the dialog window for "Add Initial Context", specify:

Where is the JNDI namespace located?

(x) File System

JNDI Namespace location

Bindings directory: C:\var\mqm\JNDI-Directory

**Add Initial Context**

**Connection details**  
Enter the location details of the JNDI namespace.

JMS administered objects are stored in Java Naming and Directory Interface (JNDI) namespaces. An Initial context defines the root of a JNDI namespace and is used to access the JMS objects that are stored in the namespace.

Where is the JNDI namespace located?

LDAP server  
 File system  
 Other

JNDI Service Provider  
Factory class:

JNDI Namespace Location  
Bindings directory:    
Provider URL:

Click Next

Accept the defaults and click Finish.

**Add Initial Context**

**User preferences**

Configure user preferences for accessing the Initial Context.

Context nickname:

Connect immediately on finish

Automatically reconnect to context on startup

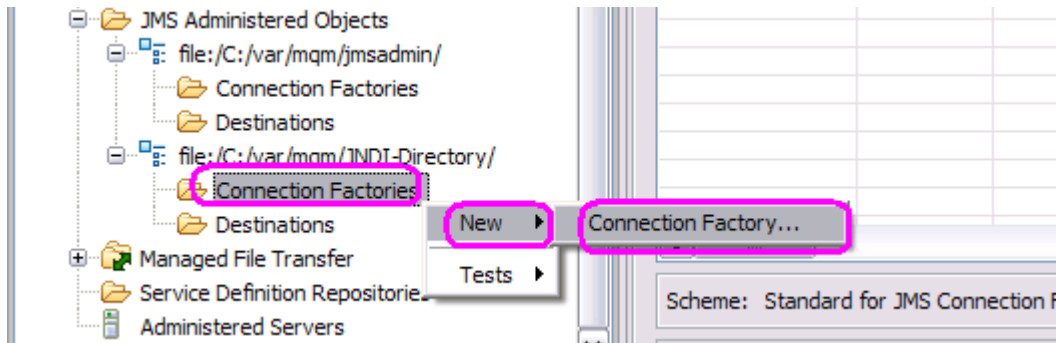
Notice that this initial context location will be shown in the right panel:

The screenshot shows the IBM WebSphere MQ Explorer interface. The left pane displays the Queue Manager hierarchy, including Queue Managers (e.g., ANGEL\_SUN4, ANGEL\_TUX1, QM\_ANGELITO) and Queue Manager Clusters. The right pane displays the JMS Administered Objects table, which is highlighted in pink. The table has the following columns: Name, Status, Provider URL, and Initial Context Factory.

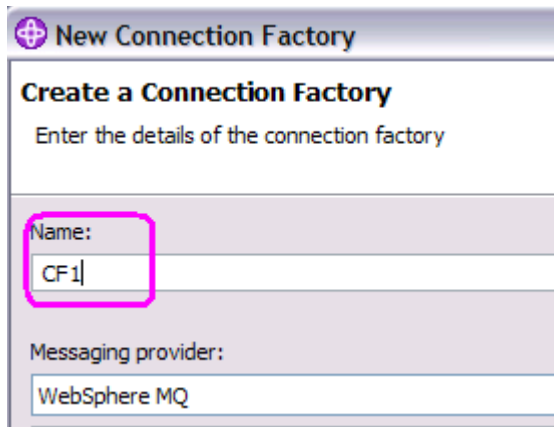
Name	Status	Provider URL	Initial Context Factory
file:/C:/var/mqm/imsadmin/	Connected	file:/C:/var/mqm/imsadmin/	com.sun.jndi.fscontext.RefFSContext
file:/C:/var/mqm/JNDI-Directory/	Connected	file:/C:/var/mqm/JNDI-Directory/	com.sun.jndi.fscontext.RefFSContext



Let's create a Connection Factory for the new initial context.  
Click on Connection Factories -> New -> Connection Factory ...



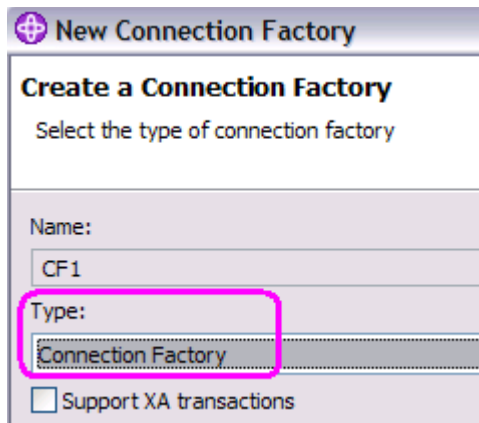
Specify the name, such as: CF1



Click Next.

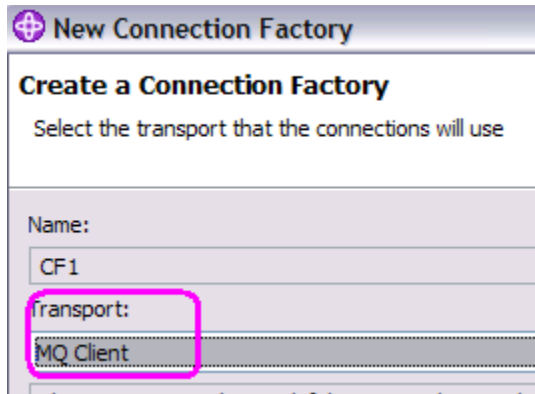
Accept the default Type of "Connection Factory"

In case that you want "Queue Connection Factory", this is the panel where you can specify it.



Click Next

The default value for the Transport is Bindings.  
Because this document is trying to illustrate Client mode, then change to:  
MQ Client



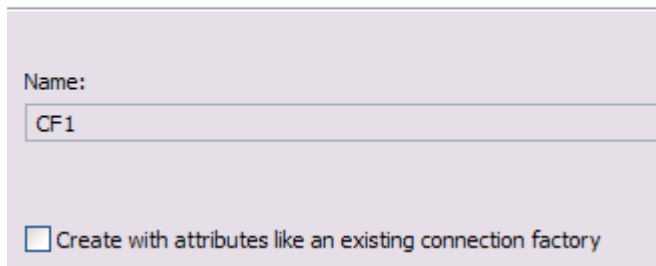
The screenshot shows a dialog box titled "New Connection Factory". Below the title bar, it says "Create a Connection Factory" and "Select the transport that the connections will use". There are two input fields: "Name:" with the value "CF1" and "Transport:" with the value "MQ Client". The "Transport:" field and its dropdown menu are highlighted with a pink rectangular box.

Click Next

If you have an existing connection factory that you want to use as a model for the rest of the fields, the following panel is the place to do it.  
In this simple example, there is no such existing CF. Thus, just Click Next.

### Create a Connection Factory

Enter the details of the object you wish to create



The screenshot shows a dialog box titled "Create a Connection Factory". It contains the text "Enter the details of the object you wish to create". There is a "Name:" label above a text box containing "CF1". At the bottom, there is a checkbox labeled "Create with attributes like an existing connection factory" which is currently unchecked.

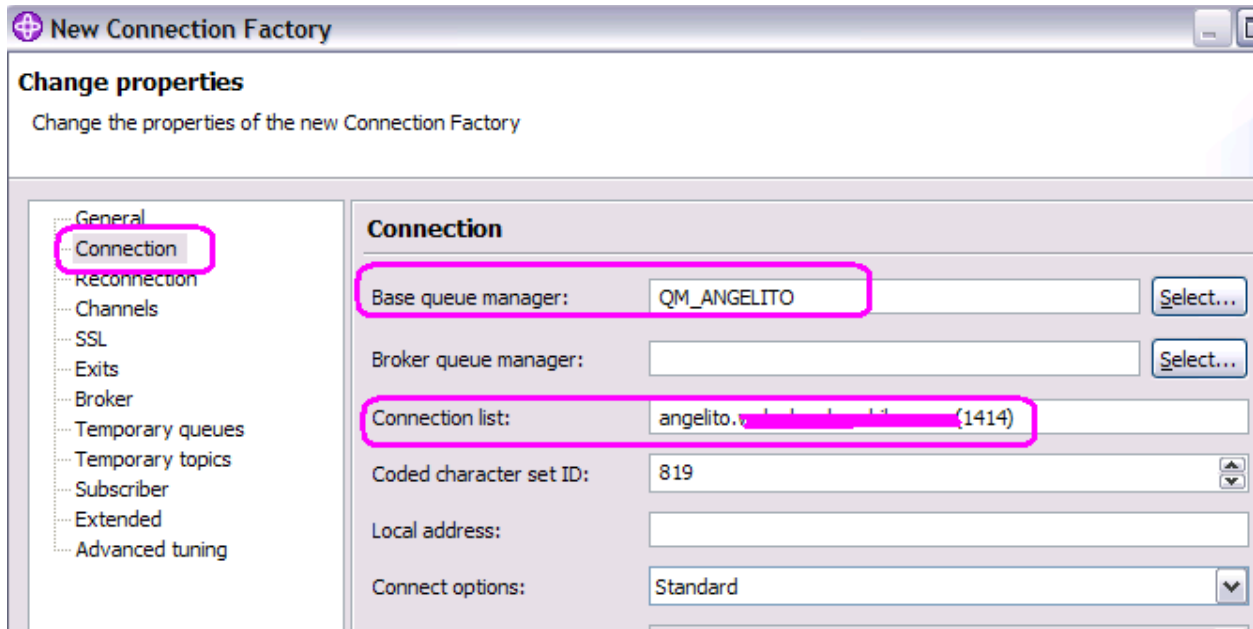
You will see the following dialog with several tabs.  
For this example, there are no changes in the General tab:

The image shows a screenshot of a software dialog box titled "New Connection Factory". The dialog has a header bar with a plus icon and the title. Below the header, the text "Change properties" is displayed, followed by a subtitle "Change the properties of the new Connection Factory". The main area is divided into two panes. The left pane contains a tree view of tabs: "General" (selected), "Connection", "Reconnection", "Channels", "SSL", "Exits", "Broker", "Temporary queues", "Temporary topics", "Subscriber", "Extended", and "Advanced tuning". The right pane is titled "General" and contains several input fields: "Name:" with the value "CF1", "Description:" (empty), "Class name:" with the value "MQConnectionFactory", "Messaging provider:" with the value "WebSphere MQ", "Transport:" with the value "Client", "Provider version:" with a radio button selected next to "unspecified" and another radio button (unselected) next to an empty field, and "Client identifier:" (empty).

In the Connection tab, enter:

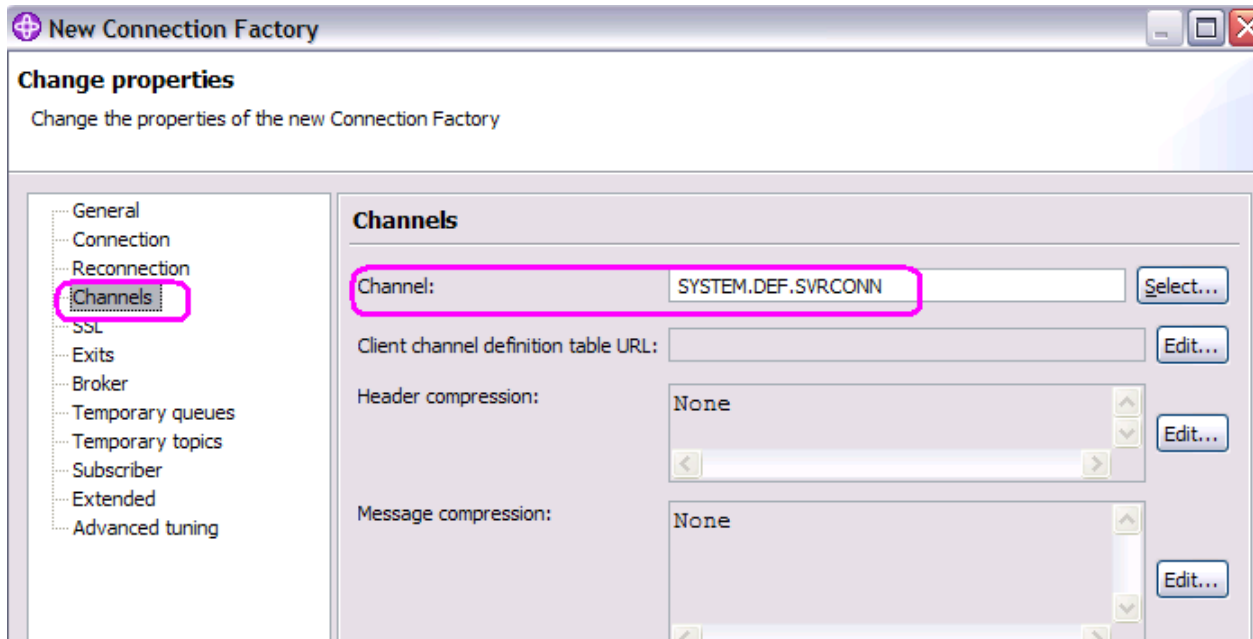
Base queue manager: QM\_ANGELITO

Connection list: the hostname address and the port number



Notice that the channel to be used is by default:

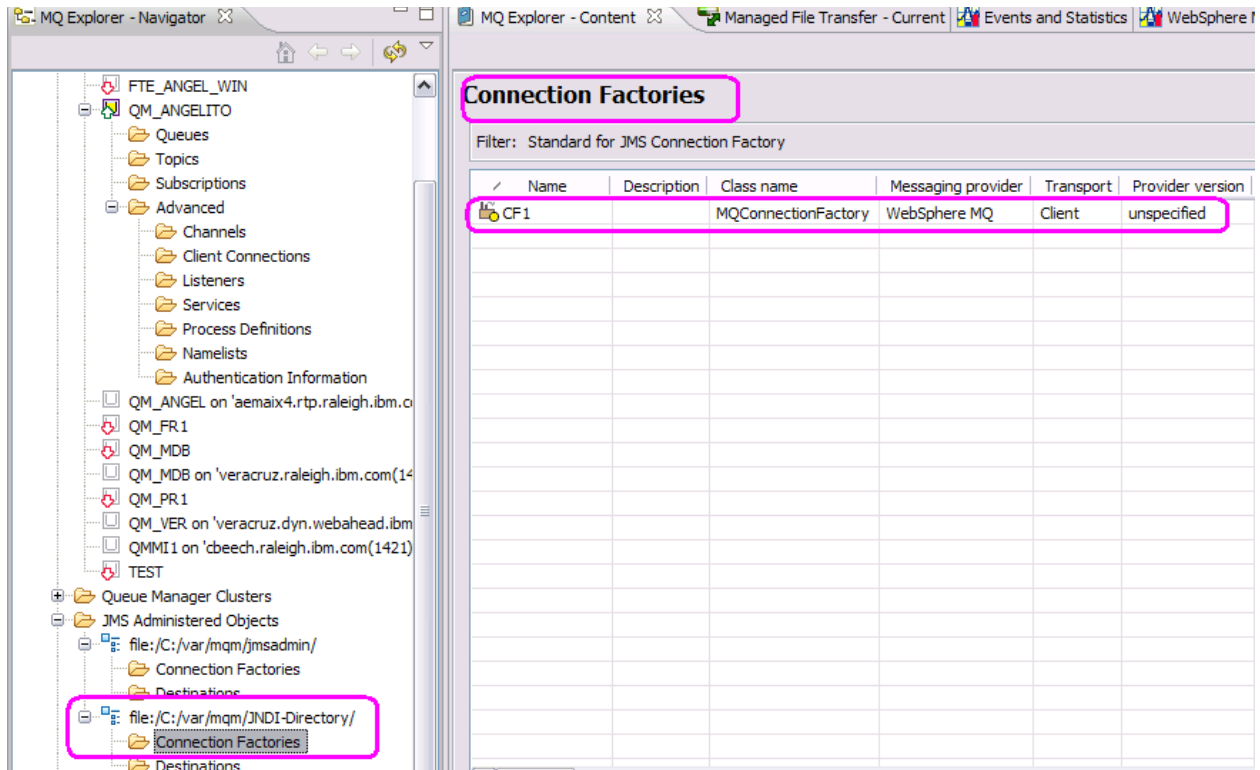
SYSTEM.DEF.SVRCONN



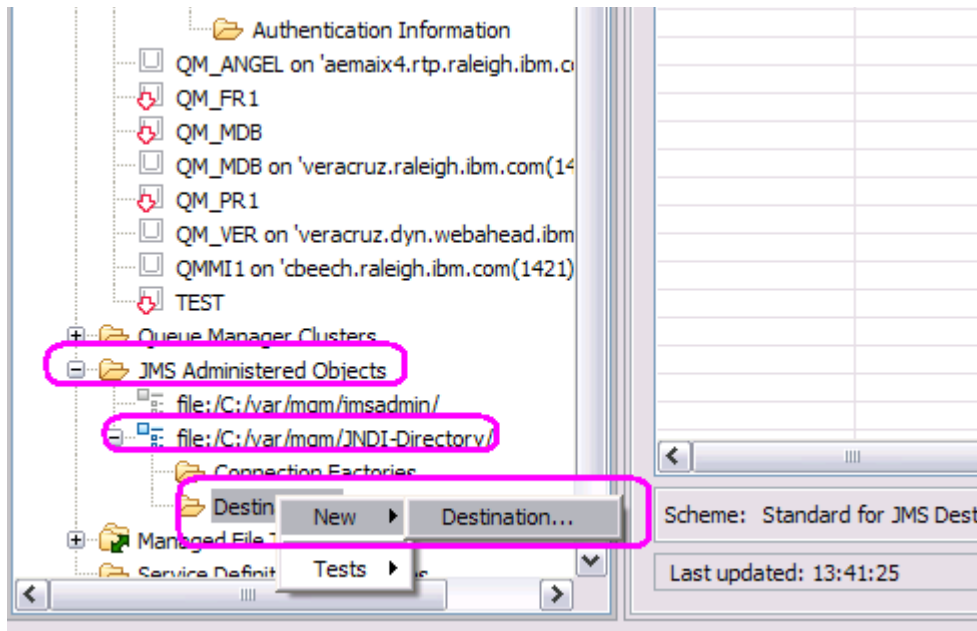
Click Finish



Notice that this new Connection Factory will be listed in the right panel.



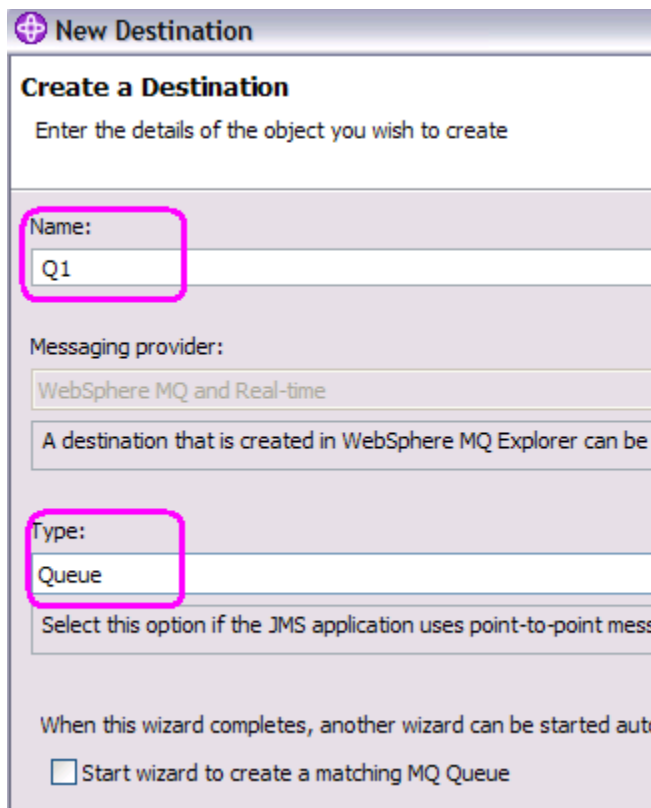
Let's create the Destination Queue Q1:



In the Create Destination, specify:

Name: Q1

Type: Queue



Click Next.

You will see:

The screenshot shows a dialog box titled "New Destination" with a sub-header "Create a Destination". Below the sub-header is the instruction "Enter the details of the object you wish to create". There is a text input field labeled "Name:" containing the text "Q1". Below this is a checkbox labeled "Create with attributes like an existing destination" which is currently unchecked. Underneath the checkbox is a smaller text input field with the placeholder text "Select an existing object from which to copy the attributes". At the bottom of the dialog, there is a message: "No system default object available, please select one".

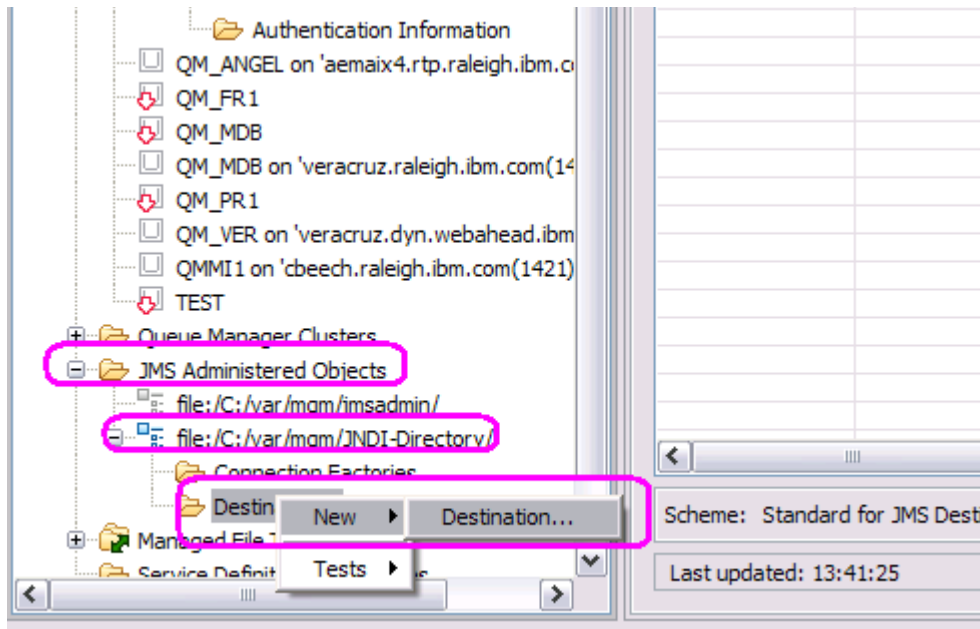
Click Next.

Then enter the name of the physical queue in the queue manager, in this case it is the same name: Q1

The screenshot shows the same "New Destination" dialog box, but now on the "Change properties" tab. The sub-header is "Change properties of the new Destination". On the left side, there is a tree view with "General" selected, and other options like "Message handling", "Producers", "Consumers", and "Extended". The main area is titled "General" and contains several fields: "Name:" with "Q1", "Description:" (empty), "Class name:" with "MQQueue", "Messaging provider:" with "WebSphere MQ and Real-time", "Queue manager:" (empty), and "Queue:" with "Q1". The "Queue:" field is highlighted with a red rectangular box.

Click Finish.

Let's create the Topic T1



In the Create Destination, specify:

Name: T1

Type: Topic

**New Destination**

**Create a Destination**

Enter the details of the object you wish to create

Name:  
T1

Messaging provider:  
WebSphere MQ and Real-time

A destination that is created in WebSphere MQ Explorer can be used

Type:  
Topic

Select this option if the JMS application uses publish/subscribe message

When this wizard completes, another wizard can be started automatically

Start wizard to create a matching MQ Topic

Click Next

**New Destination**

**Create a Destination**

Enter the details of the object you wish to create

Name:  
T1

Create with attributes like an existing destination

Select an existing object from which to copy the attributes

No system default object available, please select one

Click Next

Enter the Topic String: TOPIC1

The screenshot shows a software interface titled "New Destination" with a sub-header "Change properties" and the instruction "Change the properties of the new Destination". On the left is a tree view with "General" selected. The main area is titled "General" and contains the following fields:

Name:	T1
Description:	
Class name:	MQTopic
Messaging provider:	WebSphere MQ and Real-time
<b>Topic:</b>	TOPIC1

Click Finish

Now you can proceed to the next chapter.

```
+++++
+++ Chapter 4: Running simple JMS samples provided with MQ V7 - UNIX
+++++
```

MQ V7 ships 2 very useful JMS samples that use the JNDI:

```
JmsJndiConsumer.java
JmsJndiProducer.java
```

In UNIX, they are located at:  
/opt/mqm/samp/jms/samples

MQ also ships 2 other samples that do not use the JNDI. They are not discussed in this techdoc, but are mentioned in case that you want to review and use them:

```
JmsConsumer.java
JmsProducer.java
```

++ There are 2 scenarios:

Scenario 1: Point to Point: producing and consuming messages from a queue

Scenario 2: Pub/Sub: publishing and subscribing messages from a topic

++ Change to the directory of the samples

```
$ cd /opt/mqm/samp/jms/samples
```

++ Scenario 1: Point to Point: producing and consuming messages from a queue

+ Invoking the producer, using Connection Factory CF1 and Destination Queue Q1

```
$ java JmsJndiProducer -i file:///var/mqm/JNDI-Directory -c CF1 -d Q1
```

Initial context found!

Sent message:

```
JMSMessage class: jms_text
JMSType:      null
JMSDeliveryMode: 2
JMSExpiration: 0
JMSPriority:  4
JMSMessageID: ID:414d5120514d5f56455220202020202068538b4e02200020
JMSTimestamp: 1317778894855
JMSCorrelationID: null
```

JMSDestination: queue://QM\_VER/Q1  
JMSReplyTo: null  
JMSRedelivered: false  
JMSXAppID: WebSphere MQ Client for Java  
JMSXDeliveryCount: 0  
JMSXUserID: rivera  
JMS\_IBM\_PutApplType: 28  
JMS\_IBM\_PutDate: 20111005  
JMS\_IBM\_PutTime: 01413489  
JmsJndiProducer: Your lucky number today is 803  
SUCCESS

+ Invoking the Consumer:

```
rivera@veracruz: /opt/mqm/samp/jms/samples
$ java JmsJndiConsumer -i file:///var/mqm/JNDI-Directory -c CF1 -d Q1
Initial context found!
Received message:
```

JMSMessage class: jms\_text

...

```
JMS_IBM_PutTime: 01413489
JmsJndiProducer: Your lucky number today is 803
SUCCESS
```



++ Scenario 2: Pub/Sub: publishing and subscribing messages from a topic

You need to open 2 windows

Window 1: Consumer - to start the subscriber before the publisher

Window 2: Producer

The Consumer will connect for 15 seconds and if there are no messages, then it will terminate.

+ Window 1: Invoke the Consumer (Subscriber), using Connection Factory CF1 and Destination Topic T1. Notice that no messages are displayed yet.

```
$ java JmsJndiConsumer -i file:///var/mqm/JNDI-Directory -c CF1 -d T1  
Initial context found!
```

+ Window 2: Invoke the Producer (Publisher)

```
$ java JmsJndiProducer -i file:///var/mqm/JNDI-Directory -c CF1 -d T1  
Initial context found!  
Sent message:
```

```
    JMSMessage class: jms_text
```

```
...
```

```
    JMS_IBM_PutTime: 02035989  
JmsJndiProducer: Your lucky number today is 805  
SUCCESS
```

+ Window 1: Verify that the Consumer received a message:

Received message:

```
    JMSMessage class: jms_text
```

```
...
```

```
    JMS_IBM_PutTime: 02035989  
JmsJndiProducer: Your lucky number today is 805  
SUCCESS
```

```
+++++
+++ Chapter 5: Running simple JMS samples provided with MQ V7 - Windows
+++++
```

MQ V7 ships 2 very useful JMS samples that use the JNDI:

```
JmsJndiConsumer.java
JmsJndiProducer.java
```

In Windows, they are located at:

C:\Program Files\IBM\WebSphere MQ\tools\jms\samples

MQ also ships 2 other samples that do not use the JNDI. They are not discussed in this techdoc, but are mentioned in case that you want to review and use them:

```
JmsConsumer.java
JmsProducer.java
```

++ There are 2 scenarios:

Scenario 1: Point to Point: producing and consuming messages from a queue

Scenario 2: Pub/Sub: publishing and subscribing messages from a topic

++ Change to the directory of the samples

```
C:\> cd C:\Program Files\IBM\WebSphere MQ\tools\jms\samples
```

++ Scenario 1: Point to Point: producing and consuming messages from a queue

+ Invoking the producer, using Connection Factory CF1 and Destination Queue Q1

```
C:\> java JmsJndiProducer -i file:/var/mqm/JNDI-Directory -c CF1 -d Q1
```

Initial context found!

Sent message:

```
JMSMessage class: jms_text
JMSType:      null
JMSDeliveryMode: 2
JMSExpiration: 0
JMSPriority:  4
JMSMessageID: ID:414d5120514d5f414e47454c49544f20acf79e4e20002202
JMSTimestamp: 1319041121562
JMSCorrelationID: null
```

```
JMSDestination: queue:///Q1
JMSReplyTo: null
JMSRedelivered: false
  JMSXAppID: WebSphere MQ Client for Java
  JMSXDeliveryCount: 0
  JMSXUserID: rivera
  JMS_IBM_PutApplType: 28
  JMS_IBM_PutDate: 20111019
  JMS_IBM_PutTime: 16184157
JmsJndiProducer: Your lucky number today is 531
SUCCESS
```

+ Invoking the Consumer:

```
C:\> java JmsJndiConsumer -i file:/var/mqm/JNDI-Directory -c CF1 -d Q1
Initial context found!
Received message:

  JMSMessage class: jms_text
...
  JMS_IBM_PutTime: 16184157
JmsJndiProducer: Your lucky number today is 531
SUCCESS
```

++ Scenario 2: Pub/Sub: publishing and subscribing messages from a topic

You need to open 2 windows

Window 1: Consumer (Subscriber) - you need to start it before the publisher

Window 2: Producer (Publisher)

The Consumer (Subscriber) will connect for 15 seconds and if there are no messages, then it will terminate.

+ Window 1: Invoke the Consumer, using Connection Factory CF1 and Destination Topic T1. Notice that no messages are displayed yet.

```
C:\> cd C:\Program Files\IBM\WebSphere MQ\tools\jms\samples
```

```
C:\> java JmsJndiConsumer -i file:/var/mqm/JNDI-Directory -c CF1 -d T1
Initial context found!
```

+ Window 2: Invoke the Producer (Publisher)

```
C:\> cd C:\Program Files\IBM\WebSphere MQ\tools\jms\samples
```

```
C:\> java JmsJndiProducer -i file:/var/mqm/JNDI-Directory -c CF1 -d T1
Initial context found!
Sent message:
```

```
  JMSMessage class: jms_text
  JMSType:          null
```

...

```
  JMS_IBM_PutTime: 16220950
JmsJndiProducer: Your lucky number today is 468
SUCCESS
```

+ Window 1: Verify that the Consumer received a message:

Received message:

```
  JMSMessage class: jms_text
```

...

```
  JMS_IBM_PutTime: 16220950
JmsJndiProducer: Your lucky number today is 468
SUCCESS
```

```

+++++
+++ Chapter 6: Troubleshooting
+++++

```

```

++ Problem

```

```

$ cd /tmp
$ java JmsJndiProducer -i file:///var/mqm/JNDI-Directory -c CF1 -d T1
Exception in thread "main" java.lang.NoClassDefFoundError: JmsJndiProducer
Caused by: java.lang.ClassNotFoundException: JmsJndiProducer
    at java.net.URLClassLoader.findClass(URLClassLoader.java:434)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:660)
    at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:358)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:626)
Could not find the main class: JmsJndiProducer. Program will exit.

```

```

Resolution:

```

You do not have the following directory in your PATH or you are not under the directory when issuing the java command:

```

/opt/mqm/samp/jms/samples

```

You can add this directory into your \$PATH in your profile:

```

Export PATH=$PATH:/opt/mqm/samp/jms/samples

```

Or you can change to this directory and try again:

```

cd /opt/mqm/samp/jms/samples

```

```

++ Problem

```

```

$ java JmsJndiProducer -i file:///var/mqm/JNDI-Directory -c CF1 -d T1
Exception in thread "main" java.lang.NoClassDefFoundError: javax.jms.Destination
    at java.lang.J9VMInternals.verifyImpl(Native Method)
    at java.lang.J9VMInternals.verify(J9VMInternals.java:72)
    at java.lang.J9VMInternals.initialize(J9VMInternals.java:134)
Caused by: java.lang.ClassNotFoundException: javax.jms.Destination
    at java.net.URLClassLoader.findClass(URLClassLoader.java:434)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:660)
    at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:358)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:626)
    ... 3 more
Could not find the main class: JmsJndiProducer. Program will exit.

```

**Resolution:**

You need to execute the MQ script that sets the environment variables for using MQ Java and JMS:

For 32-bit:

```
/opt/mqm/java/bin/setjmsenv
```

For 64-bit:

```
/opt/mqm/java/bin/setjmsenv64
```

It is recommended that you modify your profile as follows, to automatically run the script when you log in:

```
+ begin
```

```
# For AIX:
```

```
export MQ_ROOT=/usr/mqm
```

```
# For non-AIX:
```

```
export MQ_ROOT=/opt/mqm
```

```
export PATH=$PATH:$MQ_ROOT/bin:$MQ_ROOT/java/bin:$MQ_ROOT/samp/bin:  
$MQ_ROOT/samp/jms/samples
```

```
if [ -f $MQ_ROOT/java/bin/setjmsenv64 ]
```

```
then
```

```
. setjmsenv64
```

```
else
```

```
. setjmsenv
```

```
fi
```

```
+ end
```

```
+++ end +++
```