

Using a CCDT file to connect to WebSphere MQ multi-instance queue managers from WebSphere Application Server V7

IBM Techdoc: 7020701

<http://www.ibm.com/support/docview.wss?rs=171&uid=swg27020701>

Date last updated: 19-Jun-2012

Angel Rivera - rivera@us.ibm.com
IBM WebSphere MQ Support

+++ Objective

The objective of this technical document is to describe in detail how to exploit from the WebSphere Application Server the connection to Multi-Instance Queue Managers in WebSphere MQ 7.0.1, by using an MQ "Client Channel Definition Table" (CCDT).

There are 2 ways to configure the WebSphere Application Server to work with multi-instance queue managers.

a) By using a custom property called "connectionNameList", in which the host names and ports of the different instances are specified. This is the easiest way.

NOTE: This approach is covered in the following techdoc:

<http://www.ibm.com/support/docview.wss?rs=171&uid=swg27020700>

Using custom property connectionNameList to use WebSphere MQ multi-instance queue managers with WebSphere Application Server V7

b) By using an MQ "Client Channel Definition Table" (CCDT).

This approach is covered in this techdoc. It shows how to create a CCDT that has multiple queue managers (via a "connection list"). It also shows how a Connection Factory is created via the Administrative Console and specifying the CCDT.

Technically speaking, these 2 approaches can be used also for connecting to different queue managers, and not necessarily to multi-instance queue managers. That is, if queue-manager-1 is not available, then the connection name list in the CCDT can tell the MQ JMS client that the next queue manager to try to connect is queue-manager-2, and so on. However, the focus is on multi-instance queue managers.

This document concentrates on the WebSphere Application Server V7 using a Listener Port, although it is possible to use V6.1.

The following is the configuration used to test the material in this techdoc:

- Version of the MQ Queue Manager: 7.0.1.3.
- Host 1 and Port 1: veracruz.x.ibm.com(1421)
- Host 2 and Port 2: cbeech.x.ibm.com(1421)
- Queue Manager Name: QMMI1
- Local queue; Q_MDB
- Version of the WebSphere Application Server: 7.0.0.13

+++ Important technote

It is highly recommended that you review the details described in the following technote:

<http://www.ibm.com/support/docview.wss?uid=swg21508357>

Technote: 1508357

Using WebSphere MQ automatic client reconnection with the WebSphere MQ classes for JMS

+++ Restrictions

Note: XA is not supported during automatic client reconnect

The MQ 7.0.1 JMS client introduced also the feature "automatic client reconnection", which is used on JMS standalone applications, but it is not used when running a JMS application within an application server. Instead, the automatic retry mechanisms for Listener Ports and Activation Specifications are used.

The following section of the WAS V7 Information Center mentions that:

"Automatic client reconnection is not supported by WebSphere Application Server."

http://publib.boulder.ibm.com/infocenter/wmqv7/v7r0/index.jsp?topic=/com.ibm.mq.csqzao.doc/mi21350_.htm

Integration of WebSphere MQ classes for JMS with WebSphere Application Server

http://publib.boulder.ibm.com/infocenter/wmqv7/v7r0/index.jsp?topic=/com.ibm.mq.amqzag.doc/fa70190_.htm

WebSphere MQ V7 Information Center

Automatic client reconnection

"For a JMS client, set the reconnection options in the connection factory configuration. JMS clients running in a managed Java EE container are always

reconnectable. The reconnection is managed by the Java EE resource adapter, rather than by the WebSphere MQ client support."

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.nd.iseries.doc/info/iseriend/ae/tmj_wmq_miqm.html

WebSphere Application Server, Version 7.0

Configuring multi-instance queue manager connections and automatic client reconnect with WebSphere MQ messaging provider custom properties

There are some restrictions regarding automatic client reconnect:

+ Begin quote

Notes:

* Automatic client reconnect is not supported within application server processes as it can cause issues with XA transactions

* Automatic client reconnect cannot be configured on WebSphere MQ messaging provider activation specifications.

+ End quote

+++ Table of Contents

The chapters of this document are:

Chapter 1: Prerequisites (Sample MDB, userid with proper authority, CCDT file)

Chapter 2: Creating a Client Channel Definition Table (CCDT)

Chapter 3: Defining a Connection Factory that uses the CCDT

Chapter 4: Switchover testing of the Sample MDB - Listener Port

Test 1: Messages are passed to the MDB, from host-1 "veracruz"

Test 2: Doing a switchover of the Queue Manager from host-1 "veracruz" to host-2

Test 3: MDB gets message that is put into the other instance of the queue manager (host-2 "cbeech")

Chapter 5: Switchover testing of the Sample MDB - Activation Specification

Similar steps as in Chapter 4

Chapter 6: Troubleshooting

+++ Miscellaneous

Question 1: Is it possible to change the CCDT file and the WAS server is able to recognize it dynamically? Or does the WAS server reads the CCDT only during startup and ignores any dynamic changes to the CCDT?

Answer 1: The WAS Server reads the CCDT file during the startup and caches the contents and ignores any dynamic changes to the file.

It is not enough to just stop and restart the Listener Port.

You will need to restart the WAS Server after you update the CCDT file.

+++ Related articles +++

a) Sample MDB

The testing done in the techdoc is based on the Sample MDB and configuration steps that are described in the following techdoc:

Using WebSphere MQ V7 as JMS Provider for WebSphere Application Server V7

<http://www.ibm.com/support/docview.wss?rs=171&uid=swg27016505>

Chapter 1: MQ V7 configuration: queue and topic

Chapter 2: WebSphere Application Server V7 configuration: JNDI objects

The following files are included as attachments to the mentioned techdoc

EAR file with MDB: SampleMDBEJB.ear

b) Multi-Instance Queue Managers in Linux

The following references have more information on how to create and do basic switchover and failover tests for Multi-Instance Queue Managers in Linux.

http://www.ibm.com/developerworks/websphere/library/techarticles/1006_sampige/1006_sampige.html?ca=drs-

Creating a multi-instance queue manager for WebSphere MQ on Linux

<http://www.ibm.com/support/docview.wss?uid=swg27017883>

Creating and using a WebSphere MQ 7.0.1 multi-instance queue manager in Linux

<http://www.ibm.com/support/docview.wss?uid=swg27018726>

Using MQ Explorer 7.0.1 to create and to connect to a Multi-instance Queue Manager

c) For those users who are not too familiar with the Administrative Console from WAS, the following technote could be useful to find the items that are related to MQ.

<http://www.ibm.com/support/docview.wss?uid=swg27018556>

Quick Guide of the Administrative Console from WebSphere Application Server for WebSphere MQ users

+++ Section added on 19-Jun-2012: Using CCDTURL custom property in WAS 6.1

In WAS 6.1 the custom property CCDTURL needs to be defined on the connection factories that are going to use the CCDT.

The custom property should be configured under the custom properties section for the relevant connection factories under the JMS resources section of the WAS admin console interface.

Be aware, that the file specified by the CCDTURL must be accessible to all of the nodes that are attempting to utilize it.

The following techdoc is for WAS V7, but it includes the detailed steps for creating a custom property for a connection factory and those steps can be used with WAS V6 to create the CCDTURL custom property.

<http://www.ibm.com/support/docview.wss?rs=171&uid=swg27020700>

Using custom property connectionNameList to use WebSphere MQ multi-instance queue managers with WebSphere Application Server V7

It is highly recommended that you review the details described in the following technote:

<http://www.ibm.com/support/docview.wss?uid=swg21508357>

Using WebSphere MQ automatic client reconnection with the WebSphere MQ classes for JMS (Technote: 1508357)

See the section: **Automatic client reconnection in Java EE environments**

The WebSphere MQ Resource Adapter, which can be deployed into Java EE environments, and the WebSphere Application Server WebSphere MQ messaging provider use the WebSphere MQ classes for JMS to communicate with WebSphere MQ queue managers.

In this section, we will look at how both the WebSphere MQ Resource Adapter and the WebSphere Application Server WebSphere MQ messaging provider provide support for automatic client reconnection.

Notes:

- In WAS V7, the WebSphere MQ Resource Adapter is used. This resource adapter is NOT supported in WAS V6.
- In WAS V6.1, you will need to install the MQ 7.0.1.x client and modify the WAS environment variable MQ_INSTALL_ROOT to point to the location of the MQ code, such as /usr/mqm in AIX or /opt/mqm in UNIX.

+++++ Chapter 1: Prerequisites (Sample MDB, userid with proper authority, CCDT file) +++++

a) The version of the MQ JMS Client must be 7.0.1.3 or later.

http://publib.boulder.ibm.com/infocenter/wmqv7/v7r0/topic/com.ibm.mq.amqzag.doc/fa70190_.htm

Automatic client reconnection

There is an important note in Table 1 of the above link:

Client interface: Messaging APIs

Client: JMS (JSE, and Java EE client container and managed containers)

Program access to reconnection options: 7.0.1.3

Reconnection support: 7.0.1.3

b) The requisite in (a) means that the minimum version of WAS 7 must be WAS 7.0.0.13. The MQ Resource Adapter (RA) shipped with this version of WAS has a version of: 7.0.1.3.

The following technote provides the mapping of which is the version of the MQ JMS client (or Resource Adapter) that is shipped with each version of WAS:

<http://www.ibm.com/support/docview.wss?uid=swg21248089>

Which version of MQ is shipped with WebSphere Application Server

c) The sample MDB called "SampleMDBEJB.ear" from the technical document mentioned in the "Related article" section will be used. Thus, you will need to consult that document to properly install the MDB.

d) In MQ JMS V7 when running from within WAS, the userid that is passed to the MQ queue manager is the userid that starts the WAS processes, such as "root" or "wsadmin".

It is CRITICAL that this userid must have the proper authorities defined in all the hosts where the MQ queue managers are running, otherwise, the security/authorization failure reason code 2035 (MQRC_NOT_AUTHORIZED) will be issued.

d) You can use WebSphere Application Server V6.1 and install the MQ V7.0.1.3 Client in the same machine and modify the MQ_INSTALL_ROOT environment variable in WAS to point to the location of the MQ client, such as /opt/mqm in Linux.

To download the MQ V7.0 Client code, which is a free download and includes already both the base code plus the latest fix pack, and the MQ classes for JMS, visit:

<http://www.ibm.com/support/docview.wss?uid=swg24019253>

MQC7: WebSphere MQ V7.0 Clients

To customize the MQ_INSTALL_ROOT environment variable in WAS 6.1 to use the MQ V7 Client, consult Chapter 4 "Using Bindings transport type and local native MQ libraries" from the following document:

<http://www.ibm.com/support/docview.wss?uid=swg27017881>

WebSphere MQ V5.3, V6 and V7 as JMS Provider for WebSphere Application Server V5, V6.0, V6.1 and V7

e) For simplicity, the example illustrated in this document has both the WebSphere Application Server and WebSphere MQ installed in the same machine. In case that they were installed in separate boxes, then after using MQ to create the CCDT file (explained later on in this document), it would be necessary to copy the CCDT file from the host that has the MQ queue manager into the host that has the Application Server.

f) Starting the active and standby instances of the queue manager

Window-1 (MQ): host-1 veracruz:

Login to host-1 ("veracruz" in this example) and start the queue manager to allow a standby instance and display the status:

```
$ strmqm -x QMMI1
WebSphere MQ queue manager 'QMMI1' starting.
5 log records accessed on queue manager 'QMMI1' during the log replay phase.
Log replay for queue manager 'QMMI1' complete.
Transaction manager state recovered for queue manager 'QMMI1'.
WebSphere MQ queue manager 'QMMI1' started.
```

```
$ dspmq -x -m QMMI1
QMNAME(QMMI1)                                STATUS(Running)
      INSTANCE(veracruz)  MODE(Active)
```

Notice that only the active instance is running.

Ensure that the local queue (Q_MDB) is defined. This is the queue which is going to be monitored by the Listener Port in WAS.

```
$ runmqsc QMMI1
define qlocal(Q_MDB) replace
```

```
1 : define qlocal(Q_MDB) replace
AMQ8006: WebSphere MQ queue created.
end
2 : end
One MQSC command read.
No commands have a syntax error.
All valid MQSC commands were processed.
```

Window-2 (MQ): host-2 cbeech:

Login to host-2 ("cbeech" in this example) and start the queue manager to allow a standby instance and display the status:

```
$ strmqm -x QMMI1
WebSphere MQ queue manager 'QMMI1' starting.
A standby instance of queue manager 'QMMI1' has been started. The active
instance is running elsewhere.

$ dspmq -x -m QMMI1
QMNAME(QMMI1)                                STATUS(Running as
standby)
  INSTANCE(veracruz) MODE(Active)
  INSTANCE(cbeech)  MODE(Standby)
```

At this point we have 2 instances of the queue manager:

- The active one is running in host-1 "veracruz"
- The standby is running in host-2 "cbeech".

+++++
+++ Chapter 2: Creating a Client Channel Definition Table (CCDT)
+++++

You can use runmqsc commands or the MQ Explorer to create client connection channels, which will be added into the Client Channel Definition Table (CCDT).

In this document, the use of runmqsc commands is illustrated.

In addition, because neither runmqsc nor the MQ Explorer can show the contents of a particular CCDT file, the use of the "mqsc" utility provided by the SupportPac MO72 is also illustrated.

++ Environment variables for the Client Channel Definition Table (CCDT)

The location of the CCDT file can be defined by using the following environment variable.

Unix (default):

```
export MQCHLLIB=/var/mqm/qmgrs/QMgrName/@ipcc/
```

Windows (default):

```
set MQCHLLIB=C:\var\mqm\Qmgrs\QMGrName\@ipcc
```

The name of the CCDT file can be defined by:

Unix (default):

```
export MQCHLTAB=AMQCLCHL.TAB
```

Windows (default):

```
set MQCHLTAB=AMQCLCHL.TAB
```

++ Queue manager groups in the CCDT

The reconnection mechanism works on the principle of "queue manager groups" which are specified in the CCDT via a "connection list". For more information on this concept, see the following section of the WebSphere MQ V7 Information Center.

<http://publib.boulder.ibm.com/infocenter/wmqv7/v7r0/index.jsp?topic=/com.ibm.mq.csqzaf.doc/cs12700.htm>

Queue manager groups in the CCDT

"A queue manager group is a set of connections defined in the client channel definition table (CCDT). The set is defined by its members having the same value of the QMNAME attribute in their channel definitions."

The example in this document uses the following:

Queue Manager Group (QMNAME): QMMI1

There are 2 hosts in the CCDT with QMNAME 'QMMI1' which have the active and the standby instance of a Multi-Instance Queue Manager:

```
host: veracruz.x.ibm.com    port: 1421
host: cbeech.x.ibm.com     port: 1421
```

The channel name is the same for both instances.

The CCDT will be created using the Queue Manager QMMI1.

Note that it is possible to use runmqsc with another queue manager to create the corresponding client-server connection channels.

It is necessary to create the server-connection channel that will be used by the client-connection channel. The following uses "runmqsc" but you could use the local MQ Explorer to do so.

- Login as an MQ administrator.
- Create the server connection channel. In this sample it is called "QMMI1".

```
$ runmqsc QMMI1
DEFINE CHANNEL(QMMI1) CHLTYPE(SVRCONN) TRPTYPE(TCP) REPLACE
```

- Create the client connection channel.

* Notice that there is only one DEFINE, but the CONNAME is used as a Connection Name List (or "connection list):

```
DEFINE CHANNEL(QMMI1) CHLTYPE(CLNTCONN) TRPTYPE(TCP) +
  CONNAME('veracruz.x.ibm.com(1421),cbeech.x.ibm.com(1421)') +
  QMNAME(QMMI1) REPLACE
```

The Client-Connection channels are stored in the Client Channel Definition Table (CCDT) of the queue manager, under the file name:

AMQCLCHL.TAB

The default location of the file is:

Unix: /var/mqm/qmgrs/QMgrName/@ipcc/ AMQCLCHL.TAB

Windows: LOCATION\ qmgrs\QMGrName\@ipcc\ AMQCLCHL.TAB

Note:

Because we are using a Multi-Instance Queue Manager, the data directory is in a shared directory and not under /var/mqm. Thus, for this document, the full path for the CCDT file is:

/mqexport/701/data/QMMI1/@ipcc/AMQCLCHL.TAB

++ Using SupportPac M072 to review the CCDT

The previous tasks using runmqsc and the MQ Explorer allow you to display the created Client-Connection channels that are known to the queue manager that was used to define them.

You can copy the CCDT file into a desired directory in each of the machines where a client application is going to use the CCDT.

When using an MQ Client (non-JMS, non-Java) application in those machines, it is necessary to setup the proper environment variables to point to the CCDT file:

Unix:

```
export MQCHLLIB=/var/mqm/
```

```
export MQCHLTAB=AMQCLCHL.TAB
```

These variables are not used when working with the WebSphere Application Server.

Interesting Problem:

Let's suppose that you are an MQ Administrator and you see a CCDT file that was copied from another machine: how do you look at the contents of the CCDT file to verify its contents? (Your company may have several CCDT files with different names and for different purposes)

The answer is that you cannot use runmqsc to do so.

But you can use the following SupportPac:

Note: Technical support for this SupportPac:

It is Category 2 SupportPac, which is provided in good faith and AS-IS. There is no warranty or further service implied or committed and any supplied sample code is not supported via IBM product service channels.

<http://www.ibm.com/support/docview.wss?uid=swg24007769>

MO72: MQSC Client for WebSphere MQ

The rest of this chapter provides information on how you can use this "mqsc" utility and what pitfall to avoid.

1) Visit the above web page and download the zip file with the code and the PDF file for the documentation.

Let's assume that it is downloaded into:

Linux Intel 32-bit:

/downloads/mq/mo72-mqsc

Windows:

C:\MQ-SupportPac\MO72 MQSC Client

2) Change to the directory of the platform:

Linux Intel 32-bit:

cd /downloads/mq/mo72-mqsc/Linux Intel

Windows:

cd C:\MQ-SupportPac\MO72 MQSC Client\Windows

3) For Unix, you must give execution permission:

```
$ chmod u+x mqsc
```

4) Run the command which uses the default CCDT file name of AMQCLCHL.TAB.

You need to specify the following flags:

-n => run in client mode to access the CCDT

-t full-path-ending-with-slash => Full directory location

NOTICE that you MUST end the string with a directory separator, or slash!

INCORRECT usage:

For example, the last character of the path name is NOT a directory separator

C:\MQ-SupportPac\MO72 MQSC Client, reads CCDT\Windows>

```
mqsc -n -t C:\var\mqm\Qmgrs\QM_ANGELITO\@ipcc
Current channel table file not found
Can not find file 'C:\var\mqm\Qmgrs\QM_ANGELITO\@ipcc'
>
```

CORRECT usage:

You need to ensure that the end of the path name has a directory separator.
Notice that you get a prompt: >

Linux example:

```
$ mqsc -n -t /mqexport/701/data/QMMI1/@ipcc/
>
```

Windows example:

```
C:\MQ-SupportPac\MO72 MQSC Client, reads CCDT\Windows>
mqsc -n -t C:\var\mqm\Qmgrs\QM_ANGELITO\@ipcc\
>
```

5) Issue the following to show the contents.

Note: only showing selected entries to keep this document shorter.

```
> display channel(*)
```

```
AMQ8414: Display Channel details.
CHANNEL(QMMI1)          CHLTYPE(CLNTCONN)          DESCR( )
TRPTYPE(TCP)           CONNAME(veracruz.x.ibm.com(1421)
,cbeech.x.ibm.com(1421))          QMNAME(QMMI1)
LOCLADDR( )            USERID( )                    PASSWORD( )
MODENAME( )            TPNAME( )                     MAXMSGL(4194304)
HBINT(300)             SCYEXIT( )                    SCYDATA( )
SENDEXIT( )            SENDDATA( )                   RCVEXIT( )
RCVDATA( )             COMPHDR(NONE)                 COMPMSG(NONE)
SSLCIPH( )             SSLPEER( )                    SHARECNV(10)
CLNTWGHT(0)            AFFINITY(PREFERRED)          ALTDATE(2010-05-26)
ALTTIME(14.08.45)
```

```
> display channel(*) version
```

```
AMQ8414: Display Channel details.
CHANNEL(QMMI1)          CHLTYPE(CLNTCONN)          VERSION(7.0)
AMQ8414: Display Channel details.
```

6) To end the session (same as in normal "runmqsc") issue:

```
> end
```

+++++
+++ Chapter 3: Defining a Connection Factory that uses the CCDT
+++++

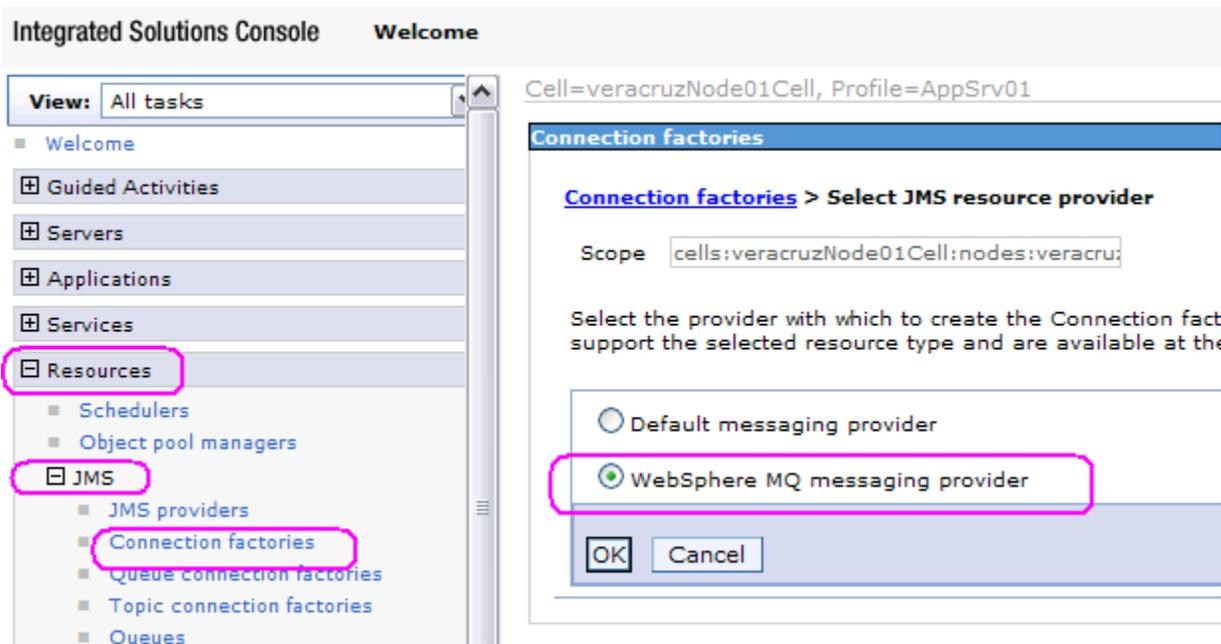
This chapter describes the steps to create a Connection Factory using the CCDT file using the Administrative Console of WAS.

The full path of the CCDT file is:
/mqexport/701/data/QMMI1/@ipcc/

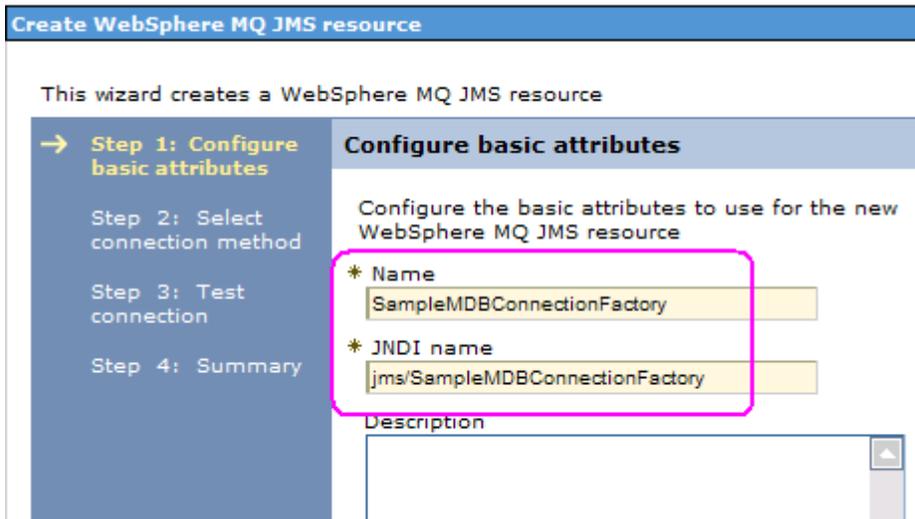
Notice that the CCDTURL MUST have 3 forward slashes. It is a very common error to specify only 2 slashes.

The data for the Connection Factory is:
Name: SampleMDBCConnectionFactory
JNDI Name: jms/SampleMDBCConnectionFactory
Transport: Client
Connection method: client channel definition table (CCDT)
CCDTURL: <file:///mqexport/701/data/QMMI1/@ipcc/AMQCLCHL.TAB>
Queue Manager name: QMMI1

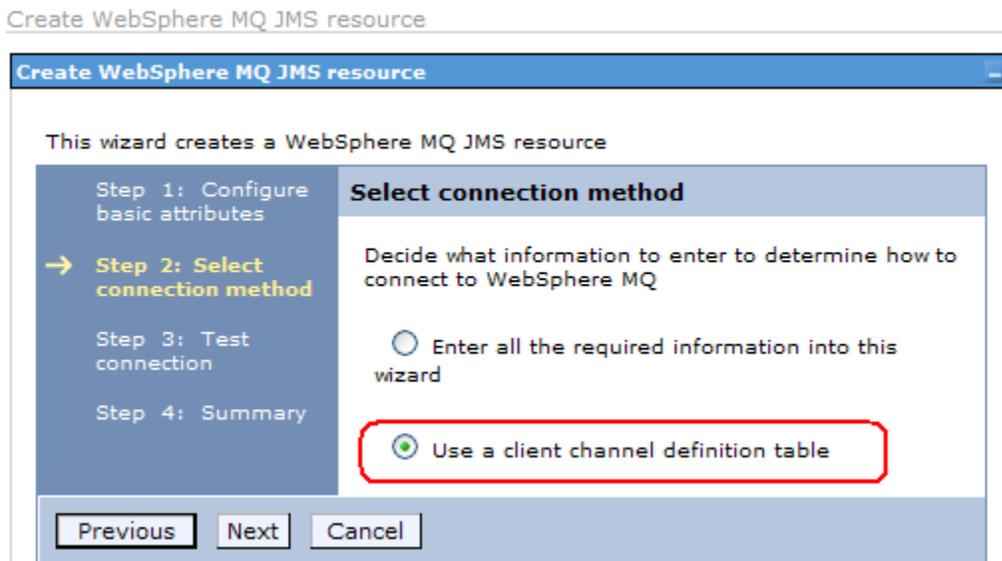
In the Administrative Console, select under "Resources":
JMS => Connection Factories
Then create a new one. Specify to use:
WebSphere MQ messaging provider



Specify the Name and JNDI name:



Then specify CCDT for the connection method:
Use a client channel definition table



You need to specify 3 slashes after "file:". Internally, this is the field "ccdturl"
file:///mqexport/701/data/QMMI1/@ipcc/AMQCLHL.TAB

+ CORRECT specification of CCDTURL and Queue Manager Name
The following shows when all the fields are properly specified:

Specify client channel definition table

Specify client channel definition table

Enter information about the client channel definition table that will be used to establish a connection to WebSphere MQ.

Step 1: Configure basic attributes

Step 2: Select connection method

→ **Step 2.1: Specify client channel definition table**

Step 3: Test connection

Step 3.1: Test connection result

Step 4: Summary

Specify client channel definition table

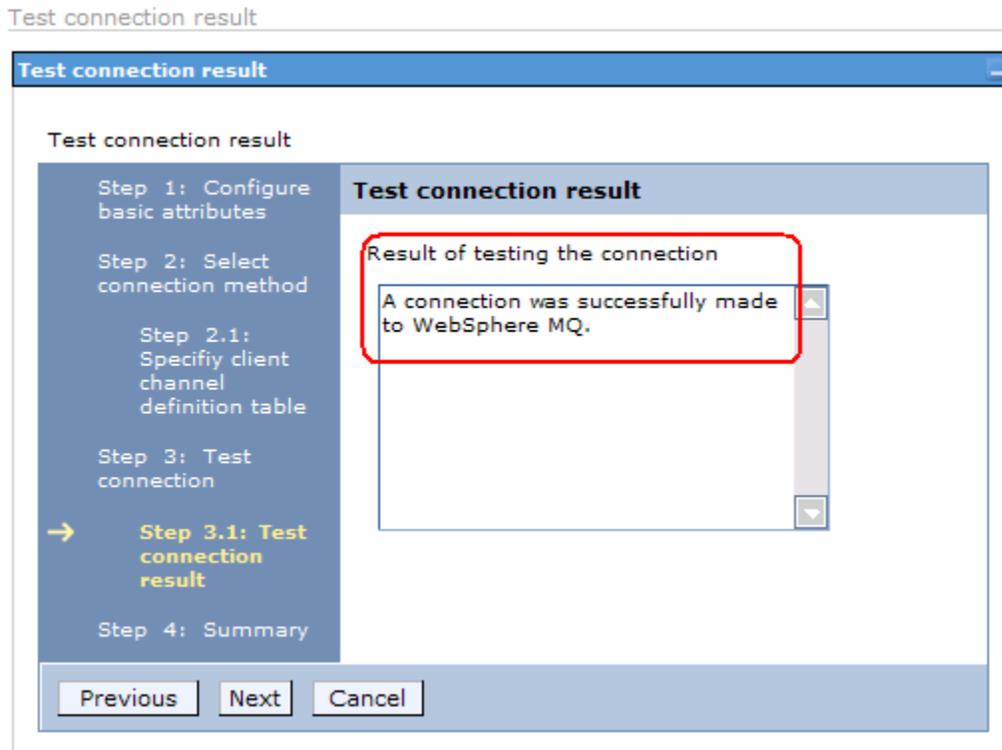
Enter information about the client channel definition table that will be used to establish a connection to WebSphere MQ.

* Client channel definition table URL
file:///mqexport/701/data/QMMI1/@ipcc/AMQCL

Queue manager
QMMI1

Previous Next Cancel

After clicking on "Test Connection" you should see a successful test:



Go ahead and continue with the prompts to create the Connection Factory.

++ Create an Activation Specification that uses a CCDT. Follow similar steps as the ones used when creating a Connection Factory. Specify it at server scope.

+ Activation Specification for a Queue:

Name: SampleMDBQueueActivationSpec
 JNDI Name: jms/SampleMDBQueueActivationSpec
 Destination JNDI name: jms/SampleMDBQueue
 Destination Type: Queue
 Queue manager: QMMI1
 Transport: Client
 Connection method: client channel definition table (CCDT)
 CCDTURL: <file:///mqexport/701/data/QMMI1/@ipcc/AMQCLCHL.TAB>
 Queue Manager name: QMMI1

+ Creating the rest of the JNDI objects

Follow the instructions to create the rest of the JNDI objects in WAS from the mentioned techdoc:

"Using WebSphere MQ V7 as JMS Provider for WebSphere Application Server V7"

Note: It is highly recommended that you click on the button "Test Connection" when creating a Connection Factory or an Activation Specification. However, in order for this test to be successful, the queue manager specified in the fields Hostname and in Port must be running. In the case of a multi-instance queue manager, the active instance must be the one specified in the field Hostname for the Connection Factory. For example, if Hostname is host-1 but the active instance is running in host-2, then the Test Connection will fail.

+ Destination: Queue

Name:	SampleMDBQueue
JNDI Name:	jms/SampleMDBQueue
Queue Name in MQ:	Q_MDB
Queue Manager:	QMMI1

+ Listener Port for Queue:

Name:	SampleMDBQueueLP
Initial State:	Started
Connection factory JNDI name:	jms/SampleMDBConnectionFactory
Destination JNDI name:	jms/SampleMDBQueue

++ NOTICE: As mentioned in the "Objective" section, this technote uses the MDB that was defined in the related article mentioned earlier. It is assumed that the Listener Port and the Activation Specification are properly defined, etc.

++ Ensure that the MDB is properly connected to a Listener Port (used in this example) or an Activation Specification (this is ONLY supported with WAS V7).

Enterprise Applications

[Enterprise Applications](#) > **SampleMDBEJB**

Use this page to configure an enterprise application. Click the links to access pages for further configuring of the application or its modules.

Configuration **Runtime**

General Properties

* Name

Application reference validation

Detail Properties

- [Target specific application status](#)
- [Startup behavior](#)
- [Application binaries](#)
- [Class loading and update detection](#)

Modules

- [Manage Modules](#)

Enterprise Java Bean Properties

- [Default messaging provider references](#)
- [Application profiles](#)
- [Message Driven Bean listener bindings](#)

Database Profiles

- [SQLJ profiles and pureQuery bind files](#)

Enterprise Applications

[Enterprise Applications](#) > [SampleMDBEJB](#) > **Message Driven Bean listener bindings**

Message Driven Bean listener bindings

Each message-driven enterprise bean in your application or module must be bound to a listener port name or to an activation specification JNDI name. When a message-driven enterprise bean is bound to an activation specification JNDI name you can also specify the destination JNDI name and authentication alias.

Apply Multiple Mappings

Select	EJB module	EJB	URI	Messaging type	Listener Bindings
<input type="checkbox"/>	SampleMDBEJB	SampleMDB	SampleMDBEJB.jar,META-INF/ejb-jar.xml	javax.jms.MessageListener	<div style="border: 2px solid magenta; padding: 5px;"> <input checked="" type="radio"/> Listener port Name <input type="text" value="SampleMDBQueueLP"/> </div> <div style="margin-top: 5px;"> <input type="radio"/> Activation Specification Target Resource JNDI Name <input type="text"/> Destination JNDI name <input type="text"/> </div>

+ Ensure that the Listener Port is defined:

Integrated Solutions Console Welcome Help

Cell=veracruzNode01Cell, Profile=AppSrv01

Application servers

Use this page to view a list of the application servers in your environment of each of these servers. You can also use this page to change the status application server.

Preferences

Name	Node	Host Name	Version
server1	veracruzNode01	veracruz.dyn.webahead.ibm.com	B...
Total 1			

Apply
OK
Reset
Cancel

- [Messaging engines](#)
- [Messaging engine inbound transports](#)
- [WebSphere MQ link inbound transports](#)
- [SIB service](#)

Server Infrastructure

- ⊕ Java and Process Management
- ⊕ Administration

Communications

- ⊕ [Ports](#)
- ⊖ Messaging
 - [Message listener service](#)

Performance

Application servers

Application servers > server1 > Message listener service > Listener Ports

Use this page to configure listener ports upon which message-driven beans listen for messages. Each port specifies a connection factory and JMS destination that a message-driven bean, deployed against that port, listens upon.

☒ Preferences

Convert to activation specification New Delete Start Stop






Select	Name ↕	Description ↕	Connection factory JNDI name ↕	Destination JNDI name
You can administer the following resources:				
<input type="checkbox"/>	SampleMDBQueueLP		jms/SampleMDBConnectionFactory	jms/SampleMDBQueue
<input type="checkbox"/>	SampleMDBTopicLP		jms/SampleMDBConnectionFactory	jms/SampleMDBTopic
Total 2				

++ Because of changes to the JMS Resources, you need to stop and restart the WAS server in order to actually use the Connection Factory.

Note:

After you create or edit JNDI items, you must stop and restart the application server, in order to make the changes available to the applications.

+++++
 +++ Chapter 4: Switchover testing of the Sample MDB - Listener Port
 +++++

At this point, it is assumed that you have defined the proper JNDI objects for JMS Resources, installed the Sample MDB with its default mapping to a Listener and restarted the application server.

This chapter shows the steps needed to test the switchover of the multi-instance queue managers and the corresponding reconnection retry from the Listener Port associated with the MDB, using the CCDT file from the connection factory.

++ Test 1: Messages are passed to the MDB, from host-1 "veracruz"

1.1) In WAS, verify that the application is running:

Cell=veracruzNode01Cell, Profile=AppSrv01

Enterprise Applications

Use this page to manage installed applications. A single application can be deployed onto m...

⊕ Preferences

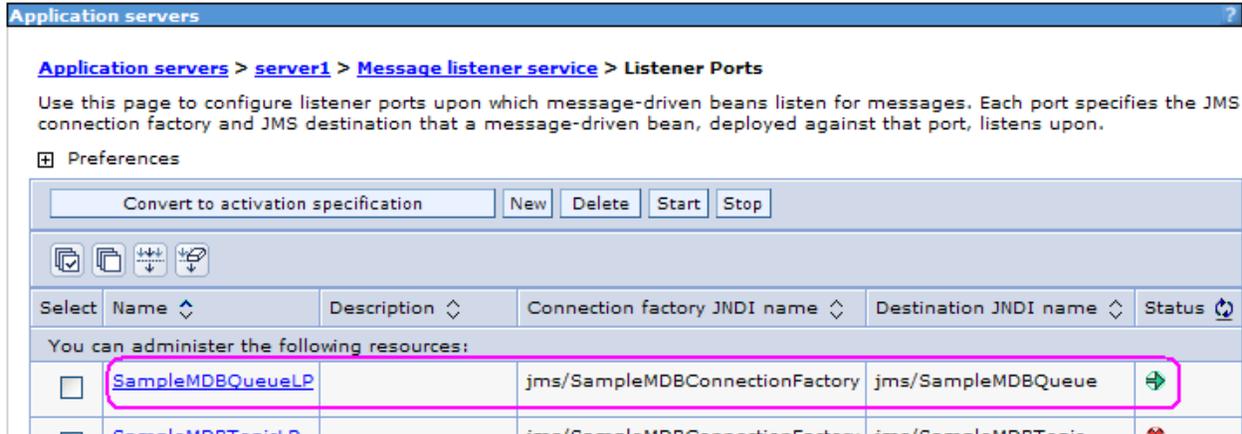
Start Stop Install Uninstall Update Rollout Update Remove File Export

Select Name Application Status

You can administer the following resources:

Select	Name	Application Status
<input type="checkbox"/>	DefaultApplication	✖
<input type="checkbox"/>	IBMUTC	✖
<input type="checkbox"/>	SampleJMSSelectorMdbEjbEAR	✖
<input type="checkbox"/>	SampleMDBEJB EAR	➡
<input type="checkbox"/>	SamplePoisonMsgMdbEjbEAR	✖

1.2) In WAS, verify that the Listener Port is running:



Window-3 (WAS): host-1 veracruz:

Login to the host where WAS is running and open a command prompt.

Change to the directory where SystemOut.log is stored, such as:

```
$ cd /opt/IBM/WebSphere/AppServer/profiles/AppSrv01/logs/server1
```

Issue the following command to continuously show the most recent entries appended to the end of the SystemOut.log:

```
$ tail -f SystemOut.log
```

Window-1 (MQ): host-1 veracruz:

Login to the host where the Queue Manager is running and open a command prompt.

Issue the MQ command to put a message into the queue Q_MDB. Enter the text:

```
TEST-MDB <ENTER>
```

Then press <ENTER> again to terminate the amspu sample.

```
$ amqspu Q_MDB QMMI1
Sample AMQSPUT0 start
target queue is Q_MDB
TEST-MDB

Sample AMQSPUT0 end
```

Window-3 (WAS): host-1 veracruz:

You should notice a new line at the bottom of the SystemOut.log:

```
[1/12/11 13:46:17:437 EST] 00000022 SystemOut      O +++ SAMPLE MDB: Text  
Message => TEST-MDB
```

This line is generated by the Sample MDB, which got a message from the WAS Listener Port. Then read the message, extracted the text and printed an entry into the SystemOut.log with the text.

At this point, the scenario has shown that the Sample MDB is working fine to get a message from the active instance running in host-1 "veracruz" of the MQ queue manager.

++ Test 2: Doing a switchover of the Queue Manager from host-1 "veracruz" to host-2 "cbeech"

Window-1 (MQ): host-1 veracruz:

Issue the following command to switchover the running instance of the queue manager from host-1 to host-2.

It is critical to use the following flags

-s = to switch to another instance

-i = to stop immediately, forcing an interruption with existing client connections.

This is critical, because the Listener Port which uses the MQ JMS client in WAS is connected to the queue manager, waiting for new messages into the queue Q_MDB. If this flag is not specified, then the endmqm will end only when the Listener Port in WAS is stopped.

```
$ endmqm -i -s QMMI1
WebSphere MQ queue manager 'QMMI1' ending.
```

Note: the switchover may take more than 30 seconds, depending on the NFS server. Thus, it is important to allow for sufficient time for the Listener Port to retry the connection. The default is 60 seconds.

Eventually you should see the following message:

```
WebSphere MQ queue manager 'QMMI1' ended, permitting switchover to a standby instance.
```

Show the status of the queue manager. It shows that the old standby is now the active (and only instance).

```
$ dspmq -x -m QMMI1
QMNAME(QMMI1)                                STATUS(Running)
elsewhere)
    INSTANCE(cbeech) MODE(Active)
```

Window-2 (MQ): host-2 cbeech:

From host-2, verify again that the queue manager is running.

```
$ dspmq -x -m QMMI1
QMNAME(QMMI1)                                STATUS(Running)
    INSTANCE(cbeech) MODE(Active)
```

Window-3 (WAS): host-1 veracruz:

The switchover of the queue manager from host-1 to host-2 will cause a break with the connection of the Listener Port which uses the MQ JMS client.

This will cause a generation of several informational, warning and error messages in the SystemOut.log. This is normal behavior and these messages can be ignored.

The following is a sample of such messages:

```
[1/17/11 15:01:06:338 EST] 00000026 SibMessage      W      [:] CWSJY0003W:
JMSSC3036: An exception has been delivered to the connections exception
listener: '
                Message : com.ibm.msg.client.jms.DetailedJMSEException:
JMSWMQ1107: A problem with this connection has occurred. An error has occurred
with the WebSphere MQ JMS connection. Use the linked exception to determine
the cause of this error.
                Class : class
com.ibm.msg.client.jms.DetailedJMSEException
.
        Caused by [1] --> Message : com.ibm.mq.MQException: JMSCMQ0001: WebSphere
MQ call failed with compcode '2' ('MQCC_FAILED') reason '2009'
('MQRC_CONNECTION_BROKEN').
                Class : class com.ibm.mq.MQException

[1/17/11 15:01:06:360 EST] 00000029 JMSEExceptionL E      WMSG0018E: Error on
JMSConnection for MDB SampleMDB , JMSDestination jms/SampleMDBQueue :
com.ibm.msg.client.jms.DetailedJMSEException: JMSWMQ1107: A problem with this
connection has occurred. An error has occurred with the WebSphere MQ JMS
connection. Use the linked exception to determine the cause of this error.
                at
com.ibm.msg.client.wmq.common.internal.Reason.reasonToException(Reason.java:60
8)
.
Caused by: com.ibm.mq.MQException: JMSCMQ0001: WebSphere MQ call failed with
compcode '2' ('MQCC_FAILED') reason '2009' ('MQRC_CONNECTION_BROKEN').

[1/17/11 15:01:06:385 EST] 00000029 JMSEExceptionL E      WMSG0057E: Error on
JMSConnection for MDB SampleMDB , JMSDestination jms/SampleMDBQueue , JMS
Linked Exception : com.ibm.mq.MQException: JMSCMQ0001: WebSphere MQ call
failed with compcode '2' ('MQCC_FAILED') reason '2009'
('MQRC_CONNECTION_BROKEN').

[1/17/11 15:01:06:756 EST] 00000029 MCWrapper      E      J2CA0081E: Method
cleanup failed while trying to execute method cleanup on ManagedConnection
...
Caused by: com.ibm.msg.client.jms.DetailedJMSEException: JMSCMQ0002: The method
'MQCTL' failed. A WebSphere MQ call failed. Please see the linked exception
for more information.

[1/17/11 15:01:06:851 EST] 00000029 SibMessage      W      [:] CWSJY0003W:
JMSSC3034: The exception is ignored as no exception listener is registered: '
                Message : com.ibm.msg.client.jms.DetailedJMSEException:
JMSWMQ1107: A problem with this connection has occurred. An error has occurred
with the WebSphere MQ JMS connection. Use the linked exception to determine
the cause of this error.

[1/17/11 15:01:07:009 EST] 00000029 MCWrapper      E      J2CA0081E: Method
destroy failed while trying to execute method destroy on ManagedConnection
com.ibm.ejs.jms
```

[1/17/11 15:01:07:590 EST] 00000029 MDBListenerIm I WMSG0043I: MDB Listener SampleMDBQueueLP stopped for JMSDestination jms/SampleMDBQueue

NOTE: The following message indicates that the Listener Port will try the reconnection

[1/17/11 15:01:07:593 EST] 00000029 MDBListenerIm I WMSG0058I: Listener Port SampleMDBQueueLP will attempt to restart in 60 seconds

[1/17/11 15:02:07:651 EST] 00000006 ConnectionEve W J2CA0206W: A connection error occurred. To help determine the problem, enable the Diagnose Connection Usage option on the Connection Factory or Data Source.

[1/17/11 15:02:07:655 EST] 00000006 ConnectionEve A J2CA0056I: The Connection Manager received a fatal connection error from the Resource Adapter for resource jms/SampleMDBConnectionFactory. The exception is: com.ibm.msg.client.jms.DetailedJMSEException: JMSWMQ1107: A problem with this connection has occurred. An error has occurred with the WebSphere MQ JMS connection. Use the linked exception to determine the cause of this error.:com.ibm.mq.MQException: JMSCMQ0001: WebSphere MQ call failed with compcode '2' ('MQCC_FAILED') reason '2009' ('MQRC_CONNECTION_BROKEN').

NOTE: The following message indicates that the Listener Port has reconnected

[1/17/11 15:02:09:598 EST] 00000006 MDBListenerIm I WMSG0042I: MDB Listener SampleMDBQueueLP started successfully for JMSDestination jms/SampleMDBQueue

++ Test 3: MDB gets message that is put into the other instance of the queue manager (host-2 "cbeech")

Window-2 (MQ): host-2 cbeech:

Put another message into the queue:

```
$ amqsput Q_MDB QMMI1
Sample AMQSPUT0 start
target queue is Q_MDB
TEST-CBEECH
```

```
Sample AMQSPUT0 end
```

Window-3 (WAS): host-1 veracruz:

You should notice a new line at the bottom of the SystemOut.log:

```
[1/12/11 14:13:03:683 EST] 00000022 SystemOut      O+++ SAMPLE MDB: Text
Message => TEST-CBEECH
```

This proves that the Listener Port that uses the MQ JMS client is able to reconnect to another instance of the queue manager.

+++++ Chapter 5: Switchover testing of the Sample MDB - Activation Specification +++++

This chapter is practically the same as Chapter 3, but instead of using a Listener Port, an Activation Specification is used.

We need to modify the mapping of the MDB to use an Activation Specification instead of a Listener Port.

You need to stop the application and modify it.

The screenshot shows the 'Enterprise Applications' console. At the top, there is a message box with a green 'i' icon and the text: 'Application SampleMDBEJB EAR on server server1 and nod successfully.' Below this, the 'Enterprise Applications' section is titled, followed by the instruction: 'Use this page to manage installed applications. A single application can be depl'. Underneath, there is a 'Preferences' section with a set of buttons: 'Start', 'Stop', 'Install', 'Uninstall', 'Update', 'Rollout Update', and 'Remove File'. Below the buttons are several icons for application management. A table lists the installed applications with columns for 'Select', 'Name', and 'Application'. The table contains the following entries:

Select	Name	Application
<input type="checkbox"/>	DefaultApplication	✘
<input type="checkbox"/>	IBMUTC	✘
<input type="checkbox"/>	SampleJMSSelectorMdbEjbEAR	✘
<input type="checkbox"/>	SampleMDBEJB EAR	✘
<input type="checkbox"/>	SamplePoisonMsgMdbEjbEAR	✘

The row for 'SampleMDBEJB EAR' is highlighted with a red rectangle.

Click on the Sample MDB: SampleMDBEJBEAR

Enterprise Applications > **SampleMDBEJBEAR**

Use this page to configure an enterprise application. Click the links to access pages for further configuring of the application or its modules.

Configuration

General Properties

* Name
SampleMDBEJBEAR

Application reference validation
Issue warnings

Detail Properties

- [Target specific application status](#)
- [Startup behavior](#)
- [Application binaries](#)
- [Class loading and update detection](#)
- [Request dispatcher properties](#)
- [View Deployment Descriptor](#)

Modules

- [Manage Modules](#)

Enterprise Java Bean Properties

- [Default messaging provider references](#)
- [Application profiles](#)
- [Message Driven Bean listener bindings](#)

Database Profiles

- [SQLJ profiles and pureQuery bind files](#)

Click on "Message Driven Bean listener bindings"

Unselect "Listener Port"

Check "Activation Specification" and specify the Activation Spec for Queues.
 Targer Resource JNDI Name: jms/SampleMDBQueueActivationSpec

Enterprise Applications

[Enterprise Applications](#) > [SampleMDBEJB](#) > [Message Driven Bean listener bindings](#)

Message Driven Bean listener bindings

Each message-driven enterprise bean in your application or module must be bound to a listener port name or to an activation specification JNDI name. When a message-driven enterprise bean is bound to an activation specification JNDI name you can also specify the destination JNDI name and authentication alias.

Apply Multiple Mappings

Select	EJB module	EJB	URI	Messaging type	Listener Bindings
<input type="checkbox"/>	SampleMDBEJB	SampleMDB	SampleMDBEJB.jar,META-INF/ejb-jar.xml	javax.jms.MessageListener	<input type="radio"/> Listener port Name <input type="text"/> <input checked="" type="radio"/> Activation Specification Target Resource JNDI Name <input type="text" value="jms/SampleMDBQueueA"/> Destination JNDI name <input type="text"/> ActivationSpec authentication alias <input type="text"/>

Click OK and then Save the configuration.

Restart the application

++ Test 1: Messages are passed to the MDB, from host-1 "veracruz"

Follow the same steps as the corresponding Test 1 in Chapter 4.

++ Test 2: Doing a switchover of the Queue Manager from host-1 "veracruz" to host-2 "cbeech"

Follow the same steps as the corresponding Test 2 in Chapter 4.

However, the set of warning and error messages that are displayed in the SystemOut.log when the Activation Specification detects a disconnection with the MQ queue manager from the instance running in host-1 and then the reconnection with the instance at host-2 after the switchover, is different:

[1/17/11 15:19:20:354 EST] 00000030 SibMessage W [:] CWSJY0003W: JMSSC3036:
An exception has been delivered to the connections exception listener: '

Message : com.ibm.msg.client.jms.DetailedJMSEException:
JMSWMQ1107: A problem with this connection has occurred. An error has occurred
with the WebSphere MQ JMS connection. Use the linked exception to determine the
cause of this error.

Caused by [1] --> Message : com.ibm.mq.MQException: JMSCMQ0001: WebSphere
MQ call failed with compcode '2' ('MQCC_FAILED') reason '2009'
(MQRC_CONNECTION_BROKEN').

Class : class com.ibm.mq.MQException

[1/17/11 15:19:20:367 EST] 00000035 SibMessage W [:] CWSJY0003W: MQJCA4013:
A connection to a queue manager failed for activationSpec
'javax.jms.Queue:jms/SampleMDBQueue@QMMI1 <2000844610>'. Check the queue
manager error logs for details.

[1/17/11 15:19:25:372 EST] 00000035 SibMessage W [:] CWSJY0003W: MQJCA4003:
A recoverable exception occurred in the JMS layer: 'JMSCMQ0002: The method
'MQCTL' failed.'

[1/17/11 15:19:25:384 EST] 00000035 SibMessage W [:] CWSJY0003W: MQJCA4003:
A recoverable exception occurred in the JMS layer: 'JMSWMQ0019: Failed to
disconnect from queue manager 'QMMI1' using connection mode '1' and host name
'localhost(1414)'.'

[1/13/11 10:24:43:697 EST] 00000032 SibMessage W [:] CWSJY0003W: MQJCA4003:
A recoverable exception occurred in the JMS layer: 'JMSWMQ0019: Failed to
disconnect from queue manager 'QMMI1' using connection mode '1' and host name
'veracruz.x.ibm.com(1421),cbeech.x.ibm.com(1421)'.'

**NOTE: The following message indicates that the Activation Specification has
reconnected**

[1/17/11 15:24:26:278 EST] 00000036 SibMessage W [:] CWSJY0003W: MQJCA4019:
ActivationSpec 'javax.jms.Queue:jms/SampleMDBQueue@QMMI1 <2000844610>'
reconnected successfully.

**++ Test 3: MDB gets message that is put into the other instance of the queue
manager (host-2 "cbeech")**

Window-2 (MQ): host-2 cbeech:

Put another message into the queue:

```
$ amqsput Q_MDB QMMI1
Sample AMQSPUT0 start
target queue is Q_MDB
TEST-ACT-SPEC-CBEECH
```

Sample AMQSPUT0 end

Window-3 (WAS): host-1 veracruz:

You should notice a new line at the bottom of the SystemOut.log:

```
[1/13/11 10:31:01:544 EST] 0000002f SystemOut    O +++ SAMPLE MDB: Text Message
=> TEST-ACT-SPEC-CBEECH
```

This proves that the Activation Specification that uses the MQ JMS client is able to reconnect to another instance of the queue manager.

+++++
+++ Chapter 6: Troubleshooting - problems when not using CCDT, incomplete data
+++++

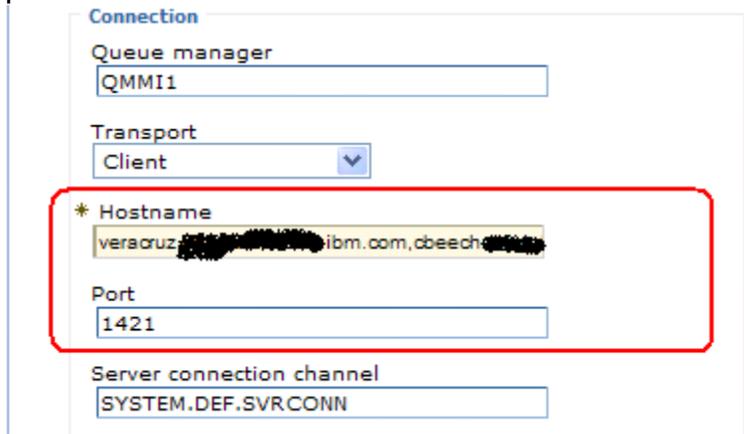
This chapter shows the error messages for several wrong scenarios.

+ Error that you may get when you do not use a CCDT for the Connection Factory in WAS, but rather, you try to use the Administrative Console to enter multiple hostnames and multiple ports in the fields for "hostname" and "port"

When specifying "Client" mode, this window, does NOT recognize multiple host names and port numbers.

At least for the Port field, if you attempt to enter 2 numbers, you will get an error message indicate that you need to enter a single integer.

For the Hostname however, there is no checking. Thus, if you enter 2 hostnames, the whole string is considered to be the single hostname to use at runtime, which causes problems:



The screenshot shows a configuration window titled "Connection". It contains several fields: "Queue manager" with the value "QMMI1", "Transport" set to "Client", "Hostname" with the value "veracruz.x.ibm.com,cbeech.x.ibm.com" (highlighted by a red box), "Port" with the value "1421", and "Server connection channel" with the value "SYSTEM.DEF.SVR.CONN".

The error message that you will

```
[2/8/10 11:02:25:855 EST] 00000007 MDBListenerIm W WMSG0019E: Unable to start  
MDB Listener SampleMDB, JMSDestination jms/SampleMDBQueue :  
com.ibm.msg.client.jms.DetailedIllegalStateException: JMSWMQ0018: Failed to  
connect to queue manager 'QMMI1' with connection mode 'Client' and host name  
'veracruz.x.ibm.com,cbeech.x.ibm.com'. Check the queue manager is started and if  
running in client mode, check there is a listener running. Please see the linked  
exception for more information.
```

+ Example of INCORRECT CCDTURL: If you specify only 2 slashes after "file:":
CCDTURL: file://mqexport/701/data/QMMI1/@ipcc/AMQCLCHL.TAB
... you will get the following error when doing "Test Connection" later on.

Specify client channel definition table

Specify client channel definition table

Enter information about the client channel definition table that will be used to establish a connection to WebSphere MQ.

Step 1: Configure basic attributes
Step 2: Select connection method
→ **Step 2.1: Specify client channel definition table**
Step 3: Test connection
Step 4: Summary

Specify client channel definition table

Enter information about the client channel definition table that will be used to establish a connection to WebSphere MQ.

* Client channel definition table URL
xport/701/data/QMMI1/@ipcc/AMQCLCHL.TAB

Queue manager
QMMI1

Previous Next Cancel

In the window for the test connection:

Test connection result

Test connection result

Test connection result

Step 1: Configure basic attributes
Step 2: Select connection method
Step 2.1: Specify client channel definition table
Step 3: Test connection
→ **Step 3.1: Test connection result**
Step 4: Summary

Test connection result

Result of testing the connection

A connection could not be made to WebSphere MQ for the following reason: JMSCMQ0001: WebSphere MQ call failed with compcode '2' ('MQCC_FAILED') reason '2278' ('MQRC_CLIENT_CONN_ERROR').

Previous Next Cancel

2278 (08E6) (RC2278): MQRC_CLIENT_CONN_ERROR

Explanation

An MQCONN call was issued to connect to a queue manager, but the MQCD channel definition structure is not specified correctly. One of the following applies:

- ClientConnOffset is not zero and ClientConnPtr is not zero and not the null pointer.
- ClientConnPtr is not a valid pointer.
- ClientConnPtr or ClientConnOffset points to storage that is not accessible.

This reason code occurs in the following environments: AIX®, HP-UX, i5/OS®, Solaris, Windows®, plus WebSphere® MQ clients connected to these systems.

It also occurs in Java™ applications when a client channel definition table is specified to determine the name of the channel, but the table itself cannot be found.

+ Incorrect specification (you do not specify a queue manager name):
If you correctly specify 3 slashes:
CCDTURL: <file:///mqexport/701/data/QMMI1/@ipcc/AMQCLHL.TAB>
... but if you forget to enter the queue manager name:

Specify client channel definition table

Specify client channel definition table

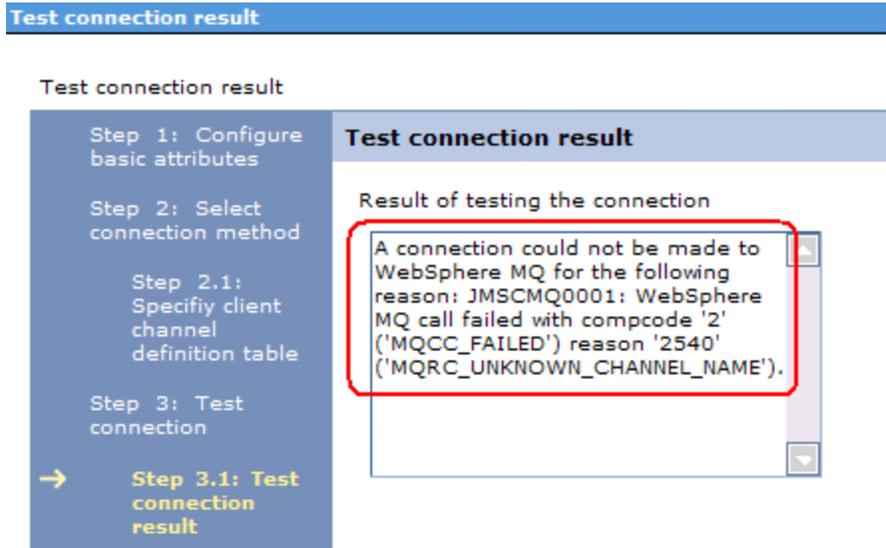
Enter information about the client channel definition table that will be used to establish a connection to WebSphere MQ.

Step 1: Configure basic attributes
Step 2: Select connection method
→ **Step 2.1: Specify client channel definition table**
Step 3: Test connection
Step 3.1: Test connection result
Step 4: Summary

*** Client channel definition table URL**

Queue manager

You will get errors during the "Test Connection".



Test connection result

Test connection result

Step 1: Configure basic attributes

Step 2: Select connection method

Step 2.1: Specify client channel definition table

Step 3: Test connection

→ **Step 3.1: Test connection result**

Test connection result

Result of testing the connection

A connection could not be made to WebSphere MQ for the following reason: JMSMQ0001: WebSphere MQ call failed with comcode '2' ('MQCC_FAILED') reason '2540' ('MQRC_UNKNOWN_CHANNEL_NAME').

Notice the error with reason 2540 MQRC_UNKNOWN_CHANNEL_NAME.

The channel name cannot be recognized because there are multiple channels in this particular CCDT and the key to find the proper entry is by using the Queue Manager name. Thus, if the queue manager name is not specified, then the proper channel name to use cannot be found.

+ Custom properties for the Activation Specification regarding reconnection retry

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/tmj_adm20.html

WAS V7 Information Center

Configuring an activation specification for the WebSphere MQ messaging provider

Note: There are four other WebSphere MQ connection properties that affect the activation specification:

- maxConnections
- connectionConcurrency
- reconnectionRetryCount
- reconnectionRetryInterval

Although these properties control the behavior of the WebSphere MQ activation specification, they are configured as custom properties of the WebSphere MQ messaging provider. For more information about these four connection properties, and how to configure them, see [Configuring custom properties for the WebSphere MQ messaging provider](#).

The `reconnectionRetryCount` and `reconnectionRetryInterval` properties are documented in the MQ V7 Info Center:

http://publib.boulder.ibm.com/infocenter/wmqv7/v7r0/index.jsp?topic=/com.ibm.mq.csqzaw.doc/jm40190_.htm

Configuration of the ResourceAdapter object

<code>reconnectionRetryCount</code>	String	5
-------------------------------------	--------	---

The maximum number of attempts made by the resource adapter to reconnect to a WebSphere MQ queue manager if a connection fails.

<code>reconnectionRetryInterval</code>	String	300 000
--	--------	---------

The time, in milliseconds, that the resource adapter waits before making another attempt to reconnect to a WebSphere MQ queue manager.

Notice that `reconnectionRetryInterval` is 5 minutes!

+ Custom properties for the Listener Port regarding reconnection retry

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/umb_prolscp.html

Message listener service custom properties

You can use the Custom properties page to define the following properties for use by the message listener service.

- * MAX.RECOVERY.RETRIES
- * NON.ASF.RECEIVE.TIMEOUT
- * RECOVERY.RETRY.INTERVAL

+++ end +++