Using an MDB that always rolls back a message to test the handling of poison messages (WebSphere MQ V7.x, V8, V9, WebSphere Application Server V7, V8.x, V9)

IBM Techdoc: 7016582
http://www.ibm.com/support/docview.wss?uid=swg27016582

Date last updated: 31-Aug-2016

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

+++ Objective +++

To demonstrate the use of a simple Message Driven Bean (MDB) in WebSphere Application Server V7, V8.x and V9, which interacts with WebSphere MQ as the Java™ Messaging Service (JMS) provider.

Shipped version of MQ RA with WAS:
- WAS 7.x and 8.0 ship MQ 7.0 Resource Adapter
- WAS 8.5 ships MQ 7.1 Resource Adapter
- The MQ V7.5 and V8 JMS clients are NOT shipped with any version of WAS.
- WAS 9.0 ships MQ 9.0 Resource Adapter.

This MDB always rolls back a message and does not complete successfully a transaction. This action indicates that the message is a “poison message”. This MDB can be used to better understand the handling of poison messages by the WebSphere Application Server and MQ.

Furthermore, with this MDB you can experiment with the different parameters of the relevant players:

WebSphere Application Server:
Listener Port: Maximum retries (default is 0)
Activation Specification: Number of sequential delivery failures before suspending endpoint (default is 0)

WebSphere MQ:
Queue: Backout Threshold => BOTHRESH (default is 0)
Backout Queue => BOQNAME (default is null)

This document has the following chapters:

- Using Listener Port:
  1. Setup for Scenario 1: using defaults for Listener Port and Queue
  2. Testing the Scenario 1: using defaults for Listener Port and Queue
  3. Setup for Scenario 2: using “Maximum retries” (2) for LP and backout queue and backout threshold (1) for Queue
  4. Testing the Scenario 2: using “Maximum retries” (2) for LP and backout queue and backout threshold (1) for Queue
- Using Activation Specification
  1. Setup for Scenario 3: using defaults for Activation Specification and Queue
  2. Testing the Scenario 3: using defaults for Activation Specification and Queue
  3. Setup for Scenario 4: using “delivery failures ” (1) for ActSpec and backout queue and backout threshold (1) for Queue
  4. Testing the Scenario 4: using “delivery failures” (1) for ActSpec and backout queue and backout threshold (1) for Queue

What is new in this update:

- On May-2013, the procedures in this techdoc were successfully tested with a queue manager running on MQ 7.5.0.1 and a WAS server 8.5.0.2 (using MQ RA 7.1.0.2).

- On Aug-2016, the procedures were tested with a queue manager running on MQ 9.0.0.0 and a WAS server 9.0.0.0 (using MQ RA 9.0.0.0)

+++ Related techdocs and articles +++

1) This techdoc is based on the configuration, deployment and test steps described in the following techdoc.

Using WebSphere MQ V7 as JMS Provider for WAS V7
http://www.ibm.com/support/docview.wss?rs=171&uid=swg27016505

2) The MDB was created with Rational Application Developer (RAD) 7.5 and the Enterprise Archive File (EAR) file which contains the MDB can be downloaded from this techdoc. For more details on how to create and test this MDB, see the following techdoc:

Developing and testing an MDB using RAD 7.5, WebSphere Application Server V7 and MQ V7 as JMS Provider
http://www.ibm.com/support/docview.wss?rs=171&uid=swg27016507

3) Excellent article on Poison Messages

How WebSphere Application Server V6 handles poison messages
http://www.ibm.com/developerworks/websphere/library/techarticles/0803_titheridge/0803_titheridge.html
This article describes how poison JMS messages can be handled by JMS provided with WebSphere Application Server, how the default behavior can be modified, and how the behavior changes if WebSphere MQ is used as the message service provider.

4) http://www.ibm.com/support/docview.wss?uid=swg21248089
Which version of WebSphere MQ Resource Adapter (RA) is shipped with WebSphere Application Server ?

+++ Requisite software +++

The following software was used:

SUSE Linux Enterprise Server (SLES) 9:
  WebSphere Application Server 7.0.0.5 or later
  WebSphere MQ 7.0.0.2 or later
  Firefox (also known as Mozilla)

+++ Downloadable files +++

The following files are included as attachments to this techdoc

EAR file with MDB:
        SamplePoisonMsgMdbEjbEAR.ear

Text file with code excerpt:
        onMessage-setRollbackOnly.txt

+++ About the MDB +++

The onMessage() method of this MDB has the following source code which displays the type of contents (payload) and an “eye catcher string” (+++ SAMPLE MDB) which can let you find quickly the output of the MDB in the SystemOut.log file.

The whole source for this method is available in the following text file associated with this techdoc:
        onMessage-setRollbackOnly.txt

```
public void onMessage(javax.jms.Message msg) {
  try {
    if (msg instanceof javax.jms.TextMessage) {
      System.out.println("+++ SAMPLE POISON MSG MDB: Text Message
=> " + ((javax.jms.TextMessage)msg).getText());
    }
…
    System.out.println("+++ SAMPLE POISON MSG MDB: Rolling back
the transaction to simulate a poison message.");
            getMessageDrivenContext().setRollbackOnly();
…
```

The relevant statement that indicates that the message should be considered a "poison message" is:

```
getMessageDrivenContext().setRollbackOnly();
```

The output from this MDB in the SystemOut.log looks like this:

```
SystemOut     O +++ SAMPLE POISON MSG MDB: Text Message =>
TEST POISON MESSAGE
SystemOut     O +++ SAMPLE POISON MSG MDB: Rolling back the
transaction to simulate a poison message.
```

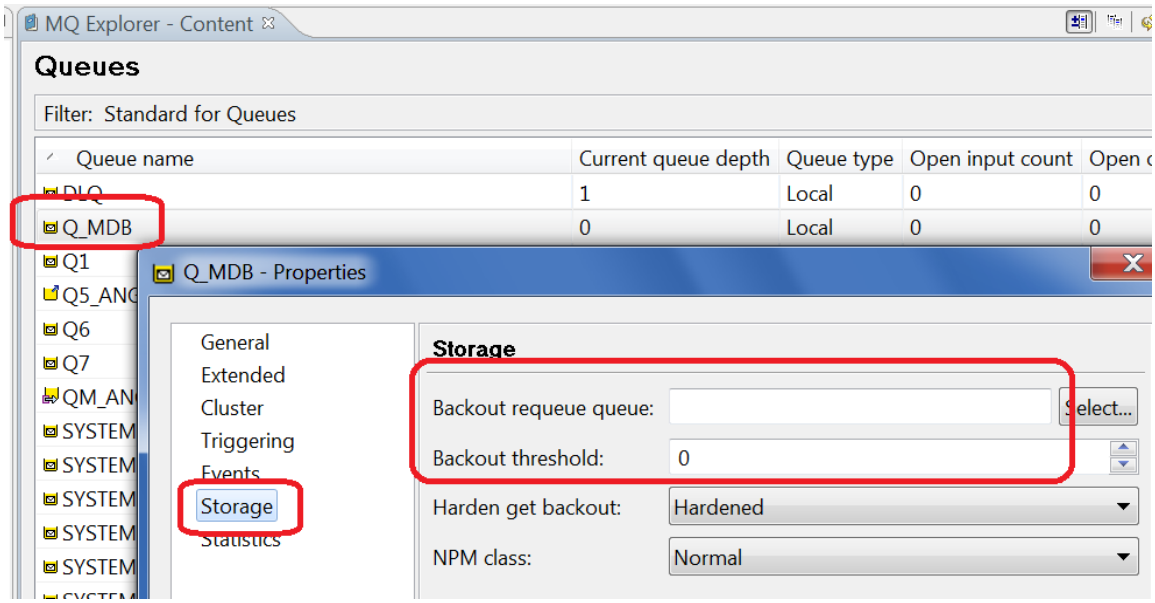
+++ Summary of objects and field values +++

EJB Project:	SamplePoisonMsgMdbEjb
EAR Project Name:	SamplePoisonMsgMdbEjbEAR
Message-driven bean: "Bean Name":	SamplePoisonMsgMdb
Listener type:	Javax.jms.MessageListener
destinationType:	javax.jms.Queue
WebSphere Bindings:	Listener Port
Listener Port Name:	SampleMDBQueueLP
Maximum Retries: 0	
WebSphere Bindings:	Activation Specification
Act Spec JNDI Name:	jms/SampleMDBQueueActivationSpec
Maximum Retries: 0	
Queue Manager Name:	QM_MDB
Queue Name:	Q_MDB
Initial setup:	
Backout Queue Name:	(blank)
Backout Threshold:	0
After configuration:	
Backout Queue Name:	Q_MDB_BO => You need to create it
Backout Threshold:	1

Note:
It is recommended NOT to use the SYSTEM.DEAD.LETTER.QUEUE or another Dead Letter Queue as a backout queue, because the messages that are sent to the backout queue do NOT have the special header that identifies the message as a dead letter (MQ DLQ).

From the MQ Explorer, the relevant backout properties for the queue Q_MDB are shown below:

Backout requeue queue
Backout threshold



++
+++ Setup for Scenario 1: using defaults for Listener Port and Queue
++

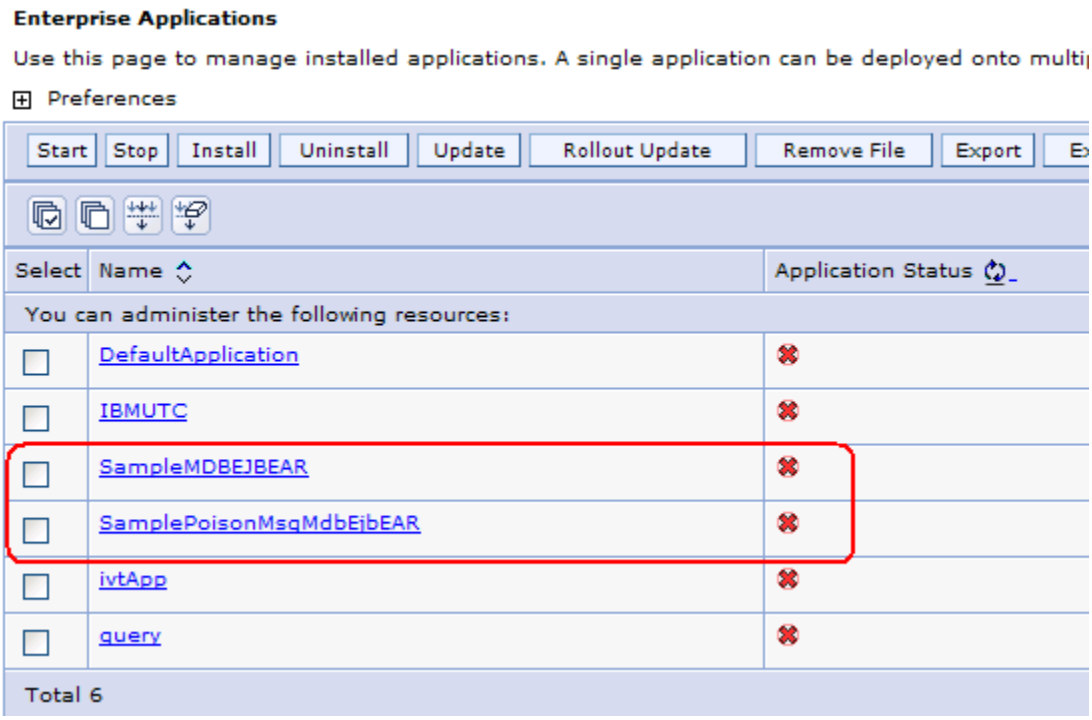
For further details on the tasks that are mentioned in this section, see the techdoc:
Using WebSphere MQ V7 as JMS Provider for WebSphere Application Server V7

It is assumed in this techdoc that all the objects needed by WebSphere Application Server and by MQ are properly configured.

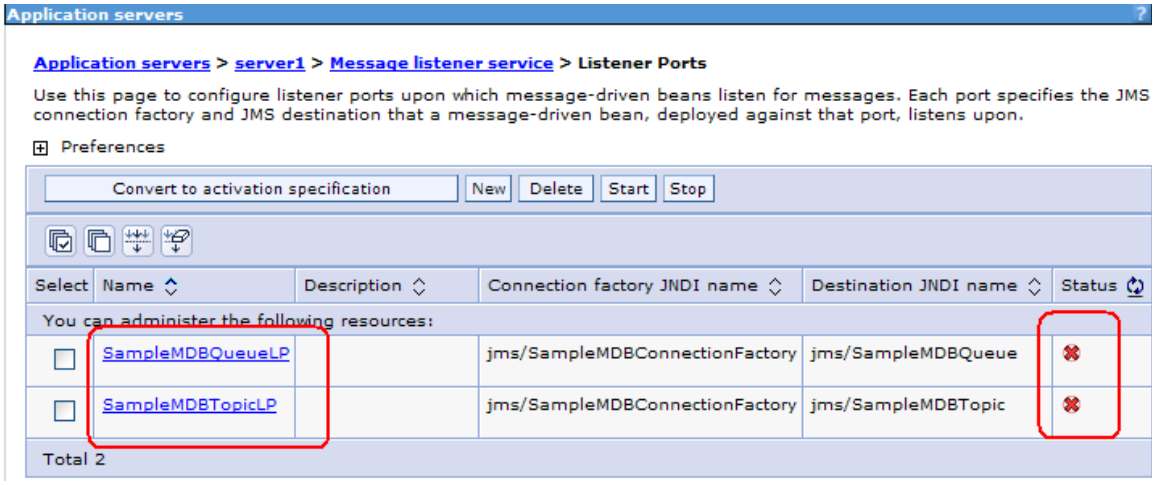
Download the sample MDB provided with this techdoc and deploy it.
Use the EAR file: SamplePoisonMsgMdbEjbEAR.ear
At “Step 6: Bind listeners for message-driven beans”, specify:
Listener Port: SampleMDBQueueLP

!!! Do NOT start yet the MDB !!!

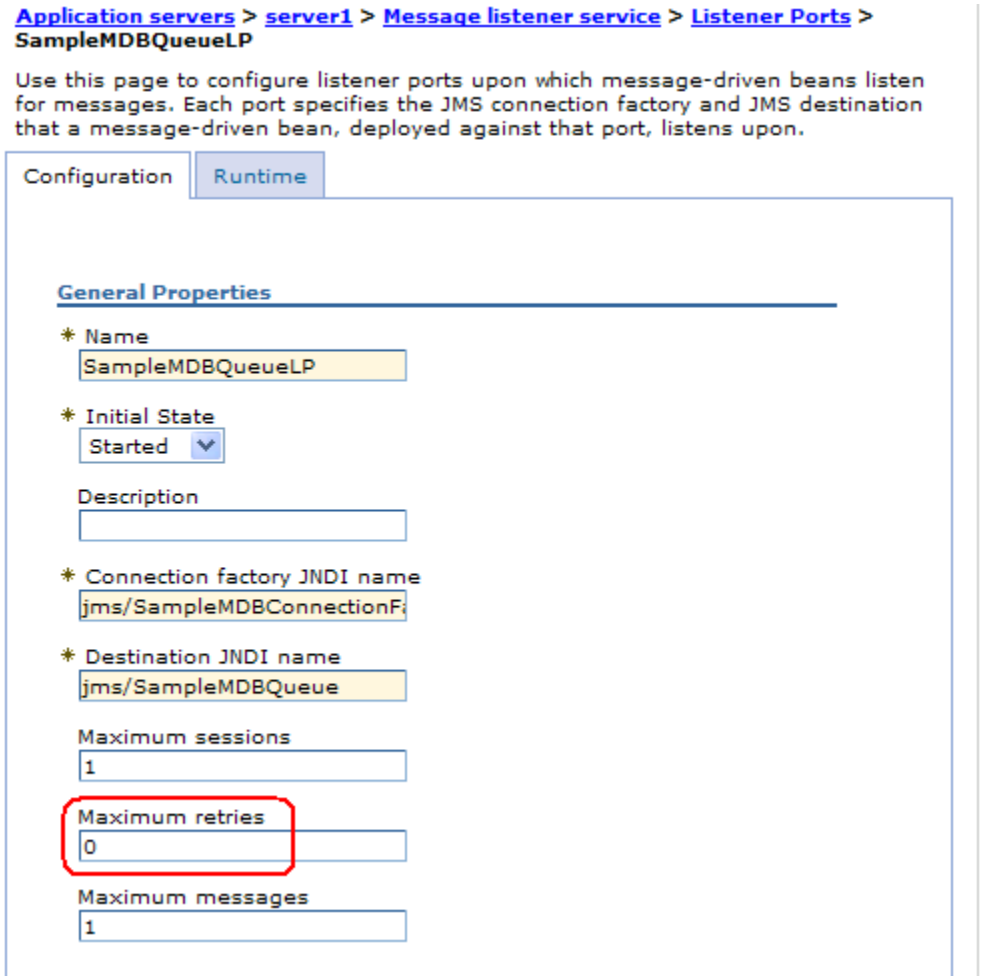
To avoid conflicts with related MDBs that consume messages from the same queue, ensure that they are stopped too:



Ensure that the Listener Port is stopped:
SampleMDBQueueLP



This techdoc assumes that the default value is used for the “Maximum retries” for the Listener Port: 0



Ensure that the queue “Q_MDB” from the queue manager “QM_MDB” is empty. You can accomplish it by issuing the following MQ sample that will read all the messages from the queue Q_MDB and the queue manager QM_MDB:

```
$ amqsget Q_MDB QM_MDB
```

Note: This sample will destructively get all the messages from the queue and waits for a new message. If there are no new messages within 30 seconds, then it terminates. It is OK not to wait 30 seconds and you can terminate it with Ctrl-C.

Issue the following command to verify that the queue is empty. Notice that CURDEPTH has a value of 0.

```
$ echo "display ql(Q_MDB) CURDEPTH" | runmqsc QM_MDB | grep
CURDEPTH
     1 : display ql(Q_MDB) CURDEPTH
   CURDEPTH(0)
```

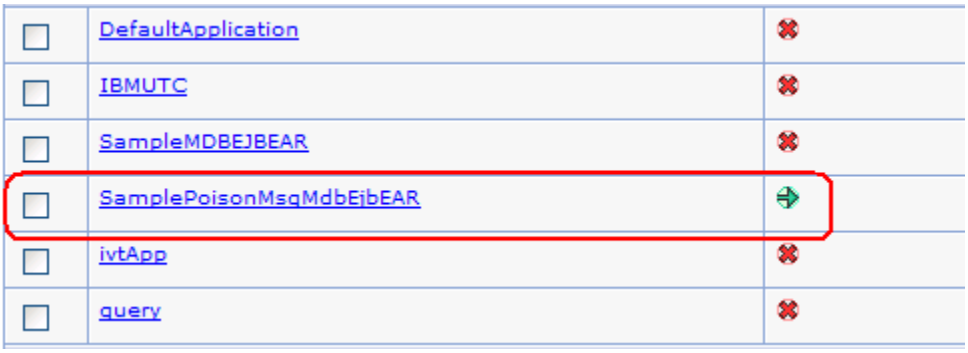
This techdoc assumes that the Queue has the default values for the following attributes:

Backout queue => BOQNAME( )
Backout threshold => BOTHRESH(0)

Issue the following to verify:

```
$ echo "display ql(Q_MDB)  BOQNAME BOTHRESH" | runmqsc
QM_MDB | grep BO
     1 : display ql(Q_MDB)  BOQNAME BOTHRESH
   BOQNAME( )                                BOTHRESH(0)
```

Start the EJB:



Start the Listener Port:

Select	Name	Description	Connection factory JNDI name	Destination JNDI name	Status
You can administer the following resources:					
☐	SampleMDBQueueLP		jms/SampleMDBConnectionFactory	jms/SampleMDBQueue	
☐	SampleMDBTopicLP		jms/SampleMDBConnectionFactory	jms/SampleMDBTopic	
Total 2					

Open a window and change the directory to the location of the App Server logs:

Window 1:

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

Monitor the end of the SystemOut.log in a continuous way:

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

Proceed to test the Poison Message MDB

++
+++ Testing the Scenario 1: using defaults for Listener Port and Queue
++

Let's do a quick review. At this point we have the following:
- An MDB that is deployed and running.
- The Listener Port associated with the MDB is running and monitoring the queue. It uses the default maximum retries of 1.
- The queue is empty. It uses the defaults of no backout queue and no backout threshold.

The expected behavior is the following: when a message is placed in the queue, the listener port begins an internal transaction (unit of work) and will pass the message to the MDB.

OK: If the message is successfully handled by the MDB, then the listener port commits the transaction, and the message is deleted from the queue.

FAILURE: If the message is NOT successfully handled by the MDB then the listener port rolls back the transaction and the message is kept in the queue (the message is now considered to be a "poison message"). This is accomplished by the following statement inside the MDB:

```
getMessageDrivenContext().setRollbackOnly();
```

Clarification:
Only by invoking the "setRollbackOnly", the message becomes a poison message.
If this method is NOT invoked, then even if the MDB throws an Exception, then the message is NOT considered a "poison message". The transaction is committed and the message is destroyed from the queue.

With a poison message, the queue manager does NOT move the message to another queue. It simply increments by 1 the value of the attribute "BackoutCount" for the message.
Originally, when the message is first placed in the queue, it has a value of 0 for this field, which means, that the message has not been involved in a delivery.
BackoutCount : 0

You can use the MQ utility "amqsbcg" to browse the messages in a queue and see the message descriptor. Let's see how the BackoutCount has been incremented from 0 to 1, to indicate that an unsuccessful attempt to deliver the message has happened. Only the top portion of the message descriptor is shown here.

```
$ amqsbcg Q_MDB QM_MDB
AMQSBCG0 - starts here
**********************
 MQOPEN - 'Q_MDB'
 MQGET of message number 1, CompCode:0 Reason:0
****Message descriptor****
  StrucId  : 'MD  '  Version : 2
  Report   : 0  MsgType : 8
  Expiry   : -1  Feedback : 0
  Encoding : 546  CodedCharSetId : 1208
  Format : 'MQSTR   '
  Priority : 0  Persistence : 0
  MsgId : X'414D5120514D5F5645523735202020201C314C5204AF1520'
  CorrelId : X'00000000000000000000000000000000000000000000000000'
  BackoutCount : 1
```

When using the default for the listener port (maximum retries of 0), then the App Server stops the Listener Port after the rollback of the transaction.
The reason is to prevent an infinite loop: the LP detects that there is a message, it delivers it to the MDB, the MDB rejects it, the LP detects that there is a message, it delivers it to the MDB, the MDB rejects it, etc.

Let's verify this behavior.

In "Window 1" we are monitoring the recent messages added to the end of the SystemOut.log. We want to see any text written by the MDB.

Let's open "Window 2" where we will issue MQ commands:

```
$ amqsput Q_MDB QM_MDB
Sample AMQSPUT0 start
target queue is Q_MDB
TEST POISON MESSAGE
Sample AMQSPUT0 end
```

In "Window 1" we see 4 new messages:

The first one is the indicator from the MDB that the message was identified as a Text Message and it shows the actual text. At this point, the transaction was not been committed nor rolled back.

```
[8/23/09 17:47:48:573 EDT] 0000002f SystemOut     O +++
SAMPLE POISON MSG MDB: Text Message => TEST POISON MESSAGE
```

The second entry indicates that that the transaction will be rolled back.

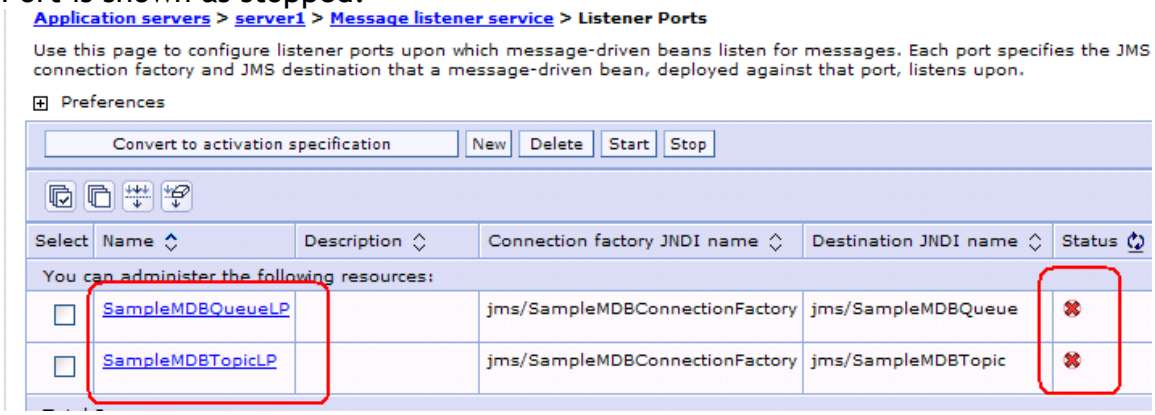
[8/23/09 17:47:48:579 EDT] 0000002f **SystemOut O +++ SAMPLE POISON MSG MDB: Rolling back the transaction to simulate a poison message.**

The last entries indicate that that Listener Port is being stopped (to avoid an infinite loop):

[8/23/09 17:47:48:648 EDT] 0000002f ServerSession E com.ibm.ejs.jms.listener.ServerSession run WMSG0036E: **Maximum message delivery retry count of 0 reached for MDB SamplePoisonMsgMdb, JMSDestination jms/SampleMDBQueue, MDBListener stopped**

[8/23/09 17:47:48:765 EDT] 000000fa MDBListenerIm I **WMSG0043I: MDB Listener SampleMDBQueueLP stopped for JMSDestination jms/SampleMDBQueue**

From the App Server Administrative Console, we can verify that the Listener Port is shown as stopped:



Let's go back to "Window 2" (MQ commands). Let's issue the following command to verify the number of messages in the queue is 1 (the message was not destroyed):

```
$ echo "display ql(Q_MDB) CURDEPTH" | runmqsc QM_MDB | grep
CURDEPTH
     1 : display ql(Q_MDB) CURDEPTH
   CURDEPTH(1)
```

Now, let's browse the message. We do not want to destroy it yet. Thus, we will use the following sample to browse messages from the queue:

```
$ amqsbcg Q_MDB QM_MDB
```

< begin output of amqsbcg >

```
AMQSBCG0 - starts here
**********************
 MQOPEN - 'Q_MDB'

 MQGET of message number 1
****Message descriptor****

  StrucId  : 'MD  '  Version : 2
  Report   : 0  MsgType : 8
  Expiry   : -1  Feedback : 0
  Encoding : 546  CodedCharSetId : 1208
  Format : 'MQSTR   '
  Priority : 0  Persistence : 0
  MsgId : X'414D5120514D5F4D44422020202020200D7C7C4A020D0520'
  CorrelId : X'000000000000000000000000000000000000000000000000'
  BackoutCount : 1
  ReplyToQ       : '                                                '
  ReplyToQMgr    : 'QM_MDB                                        '
  ** Identity Context
```

```
  UserIdentifier : 'rivera      '
  AccountingToken :
   X'0431303030000000000000000000000000000000000000000000000000000006'
  ApplIdentityData : '                                '
  ** Origin Context
  PutApplType    : '6'
  PutApplName    : 'amqsput                     '
  PutDate  : '20090823'    PutTime  : '21474830'
  ApplOriginData : '    '

  GroupId : X'000000000000000000000000000000000000000000000000'
  MsgSeqNumber   : '1'
  Offset         : '0'
  MsgFlags       : '0'
  OriginalLength : '-1'

****   Message      ****

 length - 19 bytes

00000000:  5445 5354 2050 4F49 534F 4E20 4D45 5353  'TEST POISON MESS'
00000010:  4147 45                                  'AGE             '

 No more messages
 MQCLOSE
```

< end >

Notice that the following field was incremented from 0 to 1. It indicates that the message was delivered once, but that delivery failed and the message was backed out.

```
  BackoutCount : 1
```

A quick way to find the value for this attribute is:

```
$ amqsbcg Q_MDB QM_MDB | grep "Backout"
  BackoutCount : 1
```

++
+++ Setup for Scenario 2: using “Maximum retries” (2) for LP and backout queue and backout threshold (1) for Queue
++

Now we want to make few configuration changes in order to NOT stop the Listener Port when a poison message is identified (maximum retries changed from 0 to 2), but instead, we want the poison message to be moved from the original queue to a backout queue at the first failed attempt to deliver the message (backout threshold of 1).

+ MQ Changes

Let’s cleanup the messages from the queue:

```
$ amqsget Q_MDB QM_MDB
Sample AMQSGET0 start
message <TEST POISON MESSAGE>
```

For the sake of completeness, let’s cleanup too the queue that we are going to use as the backout queue:

```
$ amqsget Q_MDB_BO QM_MDB
```

Verify the initial values of the backout queue and threshold:

```
$ echo "display ql(Q_MDB)  BOQNAME BOTHRESH" | runmqsc
QM_MDB | grep BO
     1 : display ql(Q_MDB)  BOQNAME BOTHRESH
   BOQNAME(  )                                BOTHRESH(0)
```

Now, alter the definition for the queue to add a backout queue (the Q_MDB_BO is commonly used for this) and a backout threshold of 1.

```
$ runmqsc QM_MDB
alter ql(Q_MDB) BOQNAME(Q_MDB_BO) BOTHRESH(1)
     1 : alter ql(Q_MDB) BOQNAME(Q_MDB_BO) BOTHRESH(1)
AMQ8008: WebSphere MQ queue changed.
end
```

+ App Server Changes

Using the App Server console modify the Listener Port as follows:
Maximum retries: 2 (from 0)
It should look like this:

Question: Why the value of 2?

Answer: This is recommended in the online manual for “WebSphere Application Server V6”
http://www.ibm.com/support/knowledgecenter/SSEQTP_6.1.0/com.ibm.websphere.base.iseries.doc/info/iseries/ae/umb_prolp.html
Listener Port settings
“When using WebSphere MQ as the JMS provider, this property should be greater than the value of the WebSphere MQ queue property Backout threshold (BOTHRESH). This allows WebSphere MQ to remove the message from the queue without the need to shut down the listener.”

Attention:
When changing this property for the Listener Port, it is necessary to stop and restart the application server.

After restarting the server, ensure that the Listener Port has started.

Also, to avoid conflict with the Queue Q_MDB, stop other applications that are using listener ports that monitor the same queue.

Proceed to the next chapter to test the scenario.

++
+++ Testing the Scenario 2: using “Maximum retries” (2) for LP and backout queue and backout threshold (1) for Queue
++

Let’s review the setup. We have the following:

- An MDB that is running.

- A Listener Port that has a “maximum retries” value of 2. Which means that if a poison message is encountered it will not stop immediately. Rather, it will try to deliver it again and if that 2nd attempt fails, then stop. However, we are relying on the MQ JMS client to move the poison message from the original queue to the backout queue and thus, the listener port will not have the chance to re-deliver the poison message.

- A queue that has a backout threshold of 1, which means that when the MQ JMS client detects that the value of the message attribute “Backout Count” is 1, then the MQ JMS client (not the queue manager), will move the message to the backout queue which is in this example: Q_MDB_BO

In “Window 1” (App Server) let’s continue the monitoring of the recently added lines to SystemOut.log.

In “Window 2” (MQ) let’s enter one message:

```
$ amqsput Q_MDB QM_MDB
Sample AMQSPUT0 start
target queue is Q_MDB
TEST POISON MSG 2
```

In “Window 1” the following lines are shown in the SystemOut.log:

The first 2 lines show that the MDB handled the poison message and market it for roll back. There is a delay of around 1 minute between lines 1-2 and line 3, which is the retry time for the Listener Port.

```
SystemOut     O +++ SAMPLE POISON MSG MDB: Text Message =>
TEST POISON MSG 2
SystemOut     O +++ SAMPLE POISON MSG MDB: Rolling back the
transaction to simulate a poison message.
```

There is another line indicating that an FFDC was generated.

```
FfdcProvider  I com.ibm.ws.ffdc.impl.FfdcProvider logInci-
dent FFDC1003I: FFDC Incident emitted on
/
opt/IBM/WebSphere/AppServer/profiles/AppSrv01/logs/ffdc/ser
ver1_2e352e35_09.08.23_18.51.40.3391171878247551094313.txt
com.ibm.ejs.container.EJSContainer.postInvoke 2326
```

This FFDC file indicates that there was a roll back:

```
FFDC Exception:com.ibm.websphere.csi.CSITransactionRolled-
backException SourceId:com.ibm.ejs.container.EJSContainer.-
postInvoke ProbeId:2326
com.ibm.websphere.csi.CSITransactionRolledbackException:
Transaction marked rollbackonly
```

Notice that the Listener Port is still running!

Now let's check the queues:

The message is no longer in the original queue:

```
$ echo "display ql(Q_MDB) CURDEPTH" | runmqsc QM_MDB | grep
CURDEPTH
     1 : display ql(Q_MDB) CURDEPTH
   CURDEPTH(0)
```

The message was moved to the backout queue:

```
$ echo "display ql(Q_MDB_BO) CURDEPTH" | runmqsc QM_MDB |
grep CURDEPTH
     1 : display ql(Q_MDB_BO) CURDEPTH
   CURDEPTH(1)
```

Let's see the details for the message:

< begin >

```
$ amqsbcg Q_MDB_BO QM_MDB

AMQSBCG0 - starts here
**********************

 MQOPEN - 'Q_MDB_BO'

 MQGET of message number 1
****Message descriptor****

  StrucId  : 'MD  '  Version : 2
  Report   : 0  MsgType : 8
  Expiry   : -1  Feedback : 0
  Encoding : 546  CodedCharSetId : 1208
  Format : 'MQSTR   '
  Priority : 0  Persistence : 0
  MsgId : X'414D5120514D5F4D44422020202020200D7C7C4A081F0520'
  CorrelId : X'00000000000000000000000000000000000000000000000000'
  BackoutCount : 0
  ReplyToQ       : '
'
  ReplyToQMgr    : 'QM_MDB
'
  ** Identity Context
  UserIdentifier : 'rivera      '
  AccountingToken :

X'043130303000000000000000000000000000000000000000000000000000000
6'
```

```
  ApplIdentityData : '                                '
  ** Origin Context
  PutApplType    : '6'
  PutApplName    : 'amqsput                      '
  PutDate  : '20090823'    PutTime  : '22513969'
  ApplOriginData : '    '

  GroupId : X'000000000000000000000000000000000000000000000000'
  MsgSeqNumber   : '1'
  Offset         : '0'
  MsgFlags       : '0'
  OriginalLength : '-1'

****   Message      ****

 length - 17 bytes

00000000:  5445 5354 2050 4F49 534F 4E20 4D53 4720
'TEST POISON MSG '
00000010:  32                                              '2
'

 No more messages
 MQCLOSE
```
< end >

Notice that the value is 0 for:

`BackoutCount : 0`

When the message was in the original queue Q_MDB, the value for Backout-Count was incremented from 0 to 1 when the MDB rejected the message and the message was placed back into the original Q_MDB.
The Listener Port was NOT stopped because it had a Maximum Retries of 2. The value of the retries is now 1.

The Listener Port asks again the MQ JMS client for a message, the MQ JMS client gets the message from Q_MDB. This is the same message, but now, the MQ JMS compares the BackoutCount from the message (which is now 1) and the Backout Threshold from the queue (which is 1) and determines that the back-out action needs to be performed. The MQ JMS client does NOT give the mes-sage to the Listener Port, instead it moves the message to the backout queue which is 'Q_MDB_BO, and resets the BackoutCount of the message back to 0.

++
+++ Setup for Scenario 3: using defaults for Activation Specification and Queue
++

The previous chapters showed the handling of poison messages when using Listener Ports. Now we will proceed to use Activation Specifications instead.

+ MQ Changes

From the MQ side, let's reset the backout queue and threshold for the queue to the default values.

Let's cleanup the messages from the queue:

```
$ amqsget Q_MDB QM_MDB
```

For the sake of completeness, let's cleanup too the queue that we are going to use as the backout queue:

```
$ amqsget Q_MDB_BO QM_MDB
```

Now, alter the definition for the queue to reset the backout queue to null and the backout threshold to 0.

```
$ runmqsc QM_MDB
alter ql(Q_MDB) BOQNAME('') BOTHRESH(0)
```

+ App Server changes

Stop the application.

Change the listener bindings for the MDB:
Go to the screen:
Enterprise Applications > SamplePoisonMsgMdbEjbEAR > Message Driven Bean listener bindings

Uncheck "Listener Port".
Check "Activation Specification" and specify the Activation Spec for Queues.
Target Resource JNDI Name: jms/SampleMDBQueueActivationSpec

Click on OK then click on Save.

Let's review some default properties for this Activation Spec:
Go to the screen: Resources > JMS >
Activation specifications > SampleMDBQueueActivationSpec

On the right side, in the section “Additional Properties” click on:
Advanced properties

Notice the default values:

(X) Stop endpoint if message delivery fails
Number of sequential delivery failures before suspending endpoint: 0

This means that if the Act Spec fails to deliver the message at the first try, then it will stop. This is good default, to avoid getting into an infinite loop.

Restart the application.

++
+++ Testing the Scenario 3: using defaults for Activation Specification and Queue
++

In “Window 2” (MQ), issue the following command:

```
$ amqsput Q_MDB QM_MDB
Sample AMQSPUT0 start
target queue is Q_MDB
TEST POISON MSG 3
```

In “Window 1”, where we are monitoring SystemOut.log we see 5 sets of the messages written by the MDB.

```
[8/23/09 19:27:41:180 EDT] 0000004a SystemOut     O +++
SAMPLE POISON MSG MDB: Text Message => TEST POISON MSG 3
[8/23/09 19:27:41:180 EDT] 0000004a SystemOut     O +++
SAMPLE POISON MSG MDB: Rolling back the transaction to sim-
ulate a poison message.
```

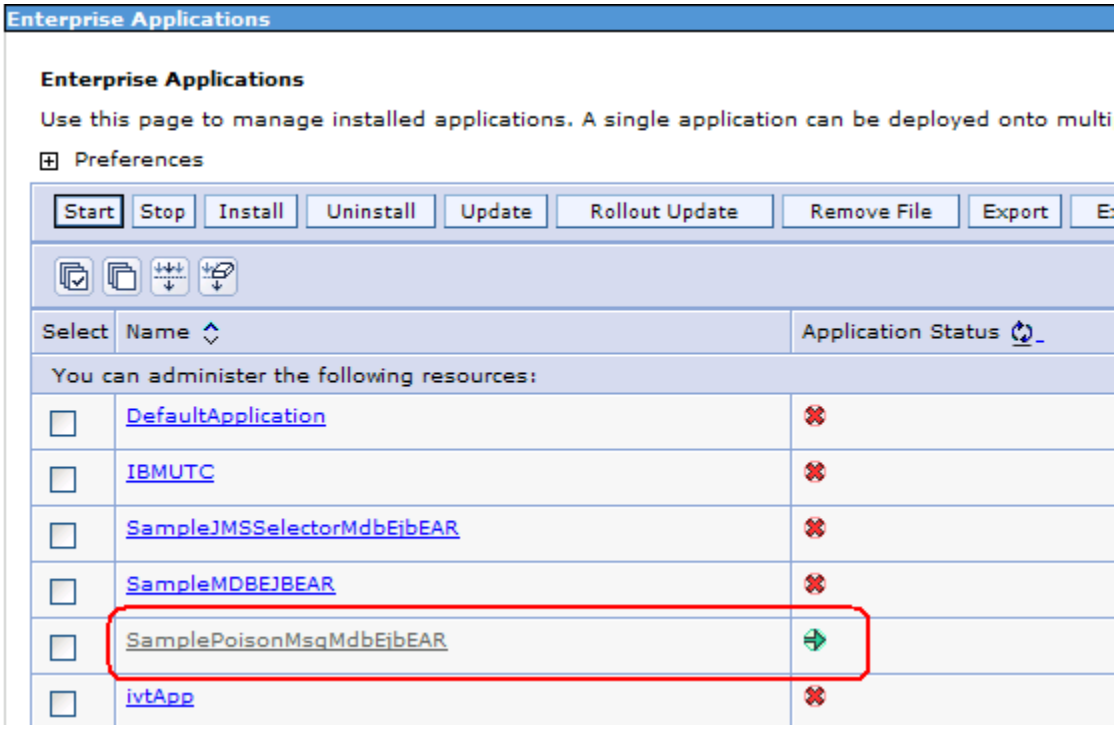
Then after trying to deliver the message 5 times, the Act Spec is paused:

```
[8/23/09 19:27:41:449 EDT] 0000004a ActivationSpe I   J2-
CA0524I: The Message Endpoint for ActivationSpec jms/Sam-
pleMDBQueueActivationSpec (com.ibm.mq.connector.inbound.Ac-
tivationSpecImpl) and MDB Application SamplePoisonMsgMdbE-
jbEAR#SamplePoisonMsgMdbEjb.jar#SamplePoisonMsgMdb is deac-
tivated.

[8/23/09 19:27:41:468 EDT] 0000004b MessageEndpoi W   J2-
CA0140W: The Message Endpoint pause operation failed for
ActivationSpec jms/SampleMDBQueueActivationSpec (com.ib-
m.mq.connector.inbound.ActivationSpecImpl) and MDB Applica-
tion
SamplePoisonMsgMdbEjbEAR#SamplePoisonMsgMdbEjb.jar#Sample-
PoisonMsgMdb because the endpoint is currently deactivated
or stopped, .or an unexpected exception occurred deactivat-
ing the endpoint.

 [8/23/09 19:27:41:492 EDT] 0000004a SibMessage    W   [:]
CWWMQ0007W: The message endpoint
SamplePoisonMsgMdbEjbEAR#SamplePoisonMsgMdbEjb.jar#Sample-
PoisonMsgMdb has been paused by the system.  Message deliv-
ery failed to the endpoint more than 0 times.  The last at-
```

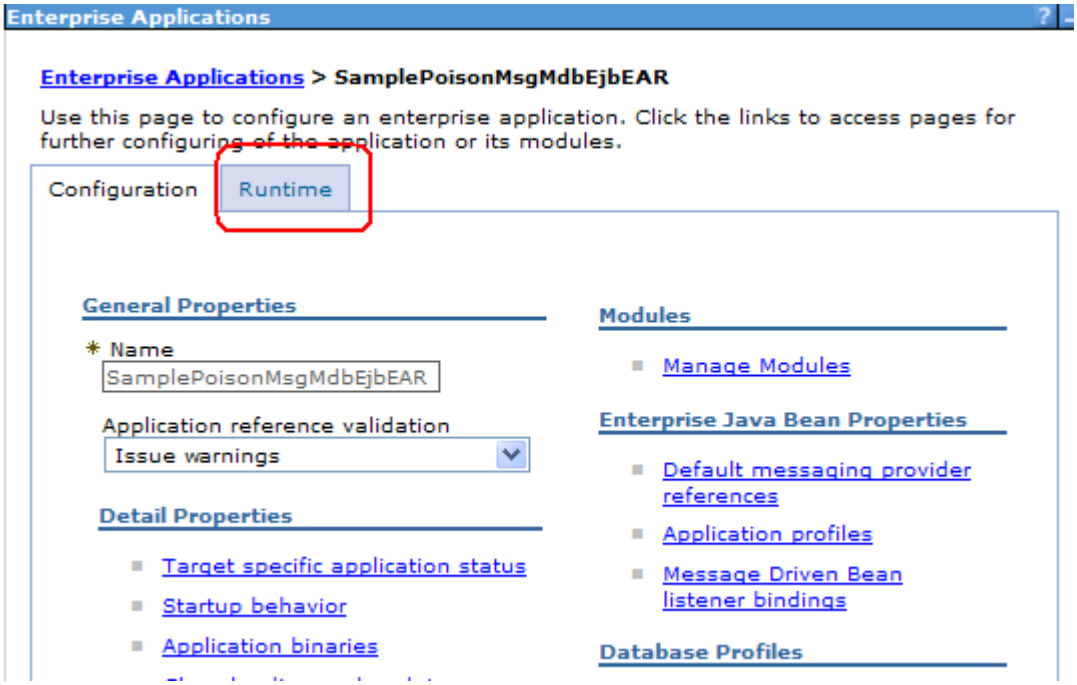
```
tempted delivery failed with the following error: javax.jm-
s.TransactionRolledBackException:
```

From the Admin Console, let's verify that the "message endpoint" is paused:

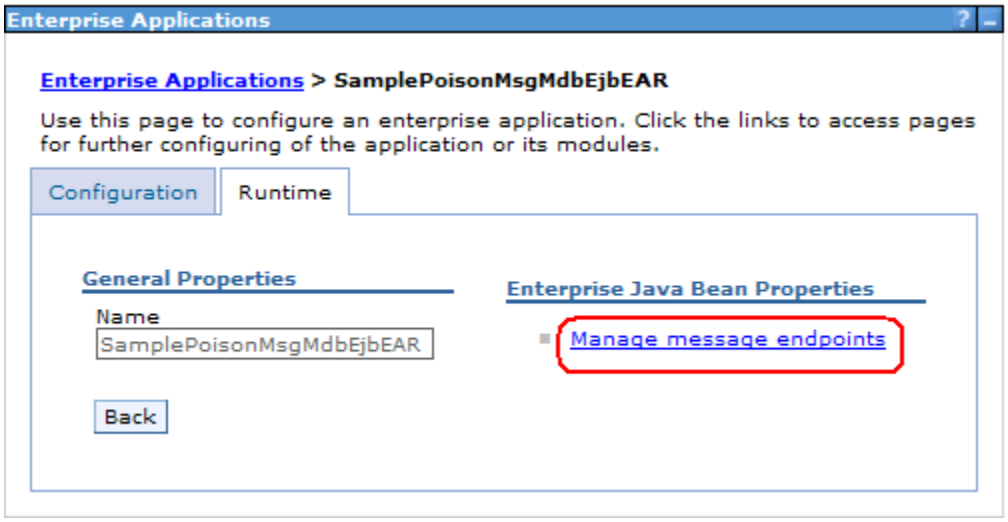
Select the Applications > Application Types > WebSphere enterprise applications > application_name.



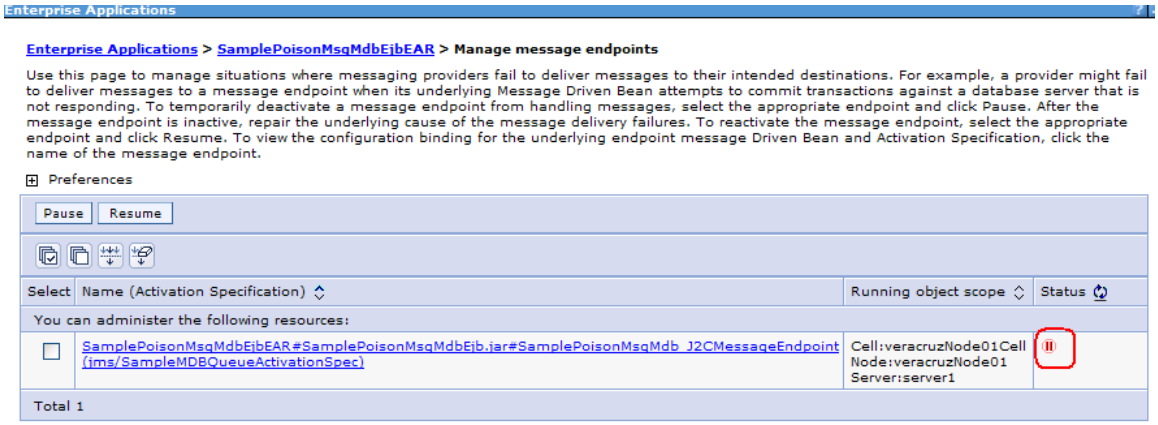
Select the Runtime panel. You will only see the Runtime panel if you have an application installed that is hosting message-driven beans.



Select Message Endpoints.



The panel lists the set of message endpoints that are hosted by the application. Notice that the endpoint is paused:



The explanation in this panel is:

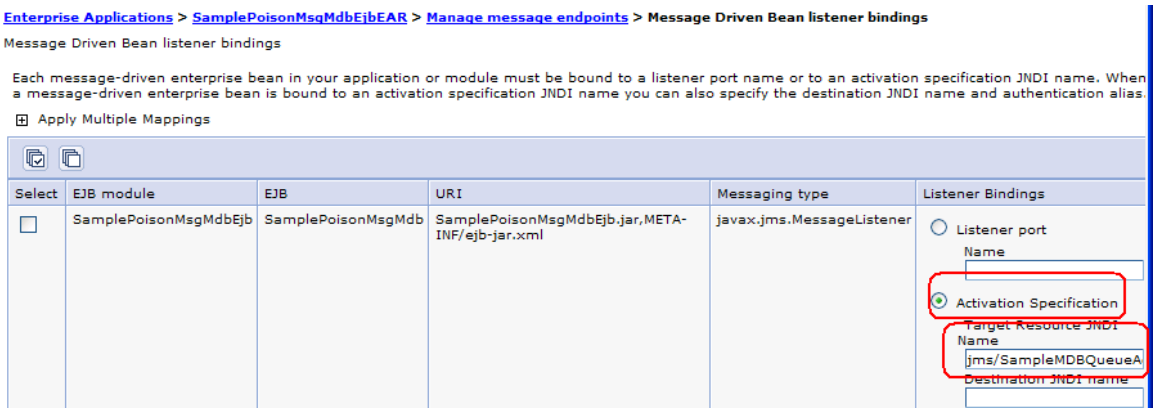
< begin >
Use this page to manage situations where messaging providers fail to deliver messages to their intended destinations. For example, a provider might fail to deliver messages to a message endpoint when its underlying Message Driven Bean attempts to commit transactions against a database server that is not responding.

To temporarily deactivate a message endpoint from handling messages, select the appropriate endpoint and click Pause.

After the message endpoint is inactive, repair the underlying cause of the message delivery failures. To reactivate the message endpoint, select the appropriate endpoint and click Resume.

To view the configuration binding for the underlying endpoint message Driven Bean and Activation Specification, click the name of the message endpoint.
< end >

The link for the endpoint points to the “MDB listener bindings” panel:



In “Window 2” we see that the message is still in the queue:

```
$ echo "display ql(Q_MDB) CURDEPTH" | runmqsc QM_MDB | grep
CURDEPTH             1 : display ql(Q_MDB) CURDEPTH
   CURDEPTH(1)
```

As shown in the SystemOut.log, the Act Spec tried to deliver the message 5 times, and thus, the backout count is 5:

```
$ amqsbcg Q_MDB QM_MDB | grep "Backout"
  BackoutCount : 5
```

++
+++ Setup for Scenario 4: using “delivery failures” (1) for ActSpec and backout queue and backout threshold (1) for Queue
++

+ MQ Changes

Let’s cleanup the messages from the queue:

```
$ amqsget Q_MDB QM_MDB
Sample AMQSGET0 start
```

For the sake of completeness, let’s cleanup too the queue that we are going to use as the backout queue:

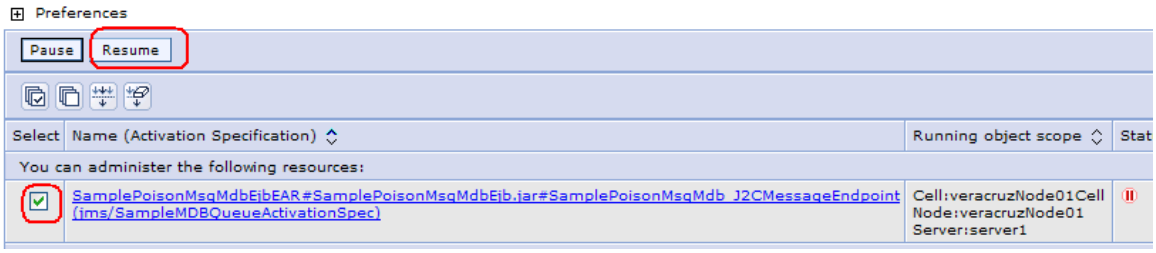
```
$ amqsget Q_MDB_BO QM_MDB
```

Alter the definition for the queue to add a backout queue (the Q_MDB_BO is commonly used for this) and a backout threshold of 1.

```
$ runmqsc QM_MDB
alter ql(Q_MDB) BOQNAME(Q_MDB_BO) BOTHRESH(1)
     1 : alter ql(Q_MDB) BOQNAME(Q_MDB_BO) BOTHRESH(1)
AMQ8008: WebSphere MQ queue changed.
end
```

+ App Server changes

Now that the queue is clean, resume the message endpoint.
Select the endpoint and click on “Resume”.



Notice that it is now in status of active:

This can also be verified by looking at the recent messages from the SystemOut.log:

```
[8/23/09 22:53:30:089 EDT] 0000001c ActivationSpe I   J2CA0523I: The Message Endpoint for ActivationSpec jms/SampleMDBQueueActivationSpec (com.ibm.mq.connector.inbound.ActivationSpecImpl) and MDB Application SamplePoisonMsgMdbEjbEAR#SamplePoisonMsgMdbEjb.jar#SamplePoisonMsgMdb is activated.
```

Change the value from 0 to 1 for the “Number of sequential delivery failures before suspending endpoint”

It is necessary to restart the server.

To avoid conflict with the queue Q_MDB ensure to stop:
- All Listener Ports associated with that queue.
- All other applications associated with that queue.

++
+++ Testing the Scenario 4: using “delivery failures” (1) for ActSpec and back-out queue and backout threshold (1) for Queue
++

In “Window 2” (MQ) issue the following:

```
$ amqsput Q_MDB QM_MDB
Sample AMQSPUT0 start
target queue is Q_MDB
TEST POISON MSG 4
```

Notice the following entries in the SystemOut.log:

```
[8/23/09 23:07:14:008 EDT] 0000002d SystemOut     O +++ SAMPLE
POISON MSG MDB: Text Message => TEST POISON MSG 4
[8/23/09 23:07:14:008 EDT] 0000002d SystemOut     O +++ SAMPLE
POISON MSG MDB: Rolling back the transaction to simulate a poison
message.
```

Notice that there is an FFDC for the rollback:

```
[8/23/09 23:07:17:172 EDT] 0000002d FfdcProvider  I com.ibm.ws.ffdc.im-
pl.FfdcProvider logIncident FFDC1003I: FFDC Incident emitted on
/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/logs/ffdc/server1_3fd53f
d5_09.08.23_23.07.14.027328912572815908843l.txt com.ibm.ejs.contain-
er.EJSContainer.postInvoke 2326
```

The message endpoint for the Act Spec is still running. It was not paused.

Notice that the original queue is now empty:

```
$ echo "display ql(Q_MDB) CURDEPTH" | runmqsc QM_MDB | grep CUR-
DEPTH
   CURDEPTH(0)
```

But there is a new message in the DLQ, which has its BackoutCount reset to 0:

```
$ echo "display ql(Q_MDB_BO) CURDEPTH" | runmqsc QM_MDB | grep
CURDEPTH
   CURDEPTH(1)

$ amqsbcg Q_MDB_BO QM_MDB  | grep "Backout"
  BackoutCount : 0
```

This is the end of the techdoc.

+++ end +++