WAY2LEARN

LAB- Using JMS Endpoint

In this lab, you will be working on **02-jmsendpoint-start** project under **04-file-jms** section

In this lab you will understand

a) how to use a JMS Outbound endpoint to send the messages to Queue

b) how to use JMS inbound endpoint to receive messages

c) What happens when JMS outbound endpoint is request-reply type

Before you start this lab, you need to start activemq locally.

Extract the active mq zip which you downloaded. cd to bin directory and execute the below command to start activemq

activemq start

STEP 1

We want to receive product details through HTTP POST request and it to JMS queue.

Another flow Should receive the JMS message and process it.

1) Drag a Http end point and configure it with URL http://localhost:8081/addproduct

2)We are expecting product details as XML through HTTP POST. After, Http endpoint the payload will be a input stream. We need a transformer to conver the stream to XML.

So, Drag a "DOM TO XML" transformer.

3) Now Drag a JMS end point. Configure its connector as ActiveMQ Connector.

Since we are connecting to ActiveMq locally, the default broker URL is sufficient.

4) Now select the radio button for Queue and give the queue name as productsq

Now your xml file should look like below :

```
<http:listener-config name="HTTP_Listener_Configuration" host="0.0.0.0" port="8081"
doc:name="HTTP Listener Configuration"/>
    <jms:activemq-connector name="Active_MQ" brokerURL="tcp://localhost:61616"
validateConnections="true" doc:name="Active MQ"/>
    <flow name="jmsoutboundflow">
        <http:listener config-ref="HTTP_Listener_Configuration" path="addproduct"
doc:name="HTTP"/>
        <mulexml:dom-to-xml-transformer doc:name="DOM to XML"/>
        <jms:outbound-endpoint queue="productsq" connector-ref="Active_MQ"
doc:name="JMS" />
    </flow>
```

5) Deploy the application and give a HTTP POST request to http://localhost:8081/addproduct .

Give the body for post as the contents of the file src/main/resources/product.xml

6) You should get the same content as response. Now go to ActiveMQ console at localhost:8161 and click on " Manage ActiveMqbroker" link . the default username/password is admin/admin.

Observe that there is one message in productsq

STEP 2

1) Now we want to receive message from productsq.

2)Drag another flow in same xml file. Drag JMS endpoint into message source part. Configure it with the same JMS connector which you configured in step 1.

Give the queue name as productsq. Let it be one-way only.

3) Drag a Logger after JMS end point to log payload. Now deploy the application and observe that the logs. You should observe the log which prints the product xml. So, you understood how to receive messages using JMS inbound endpoint.

4)Now make the JMS inbound endpoint as request-reply type.

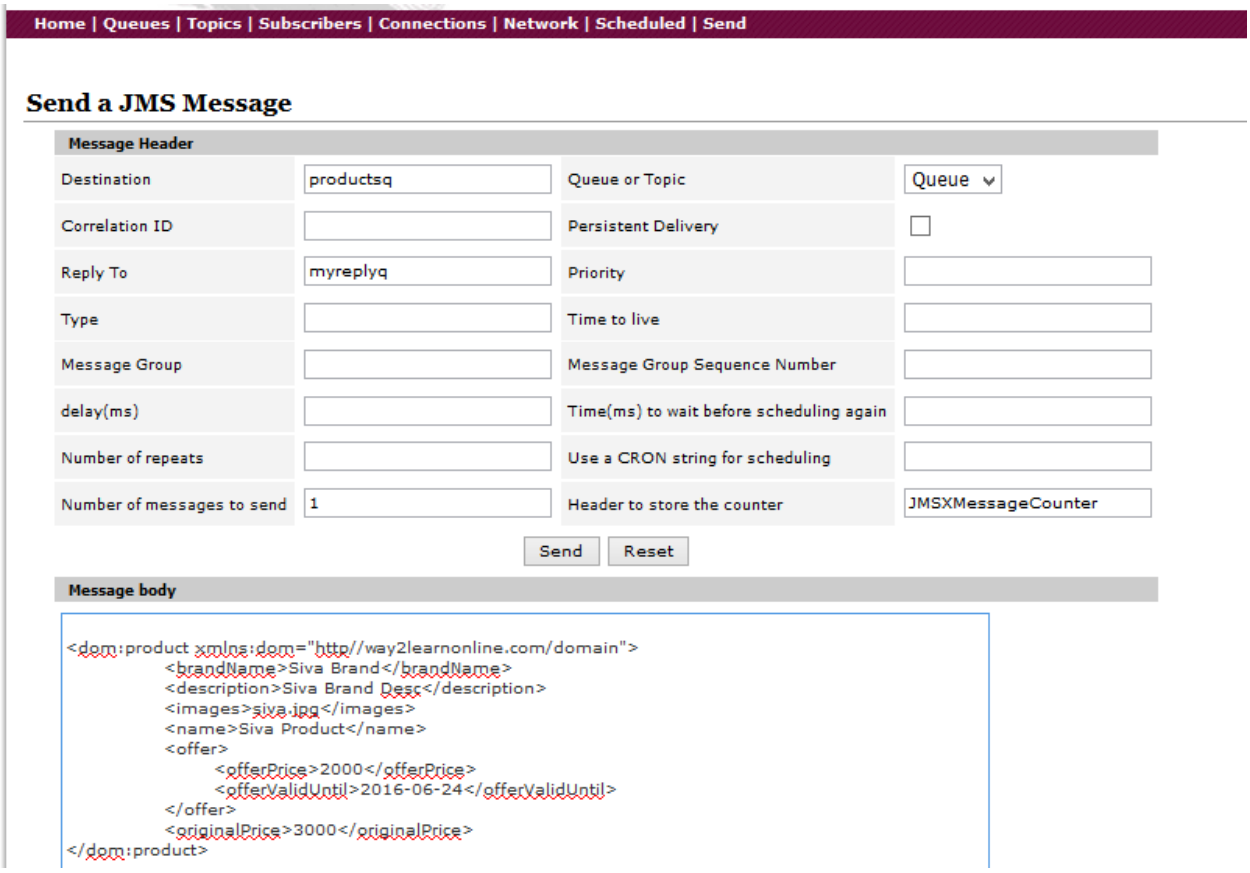
After Logger Drag a "Set Payload" transformer and set the payload as "Product Added Successfully! "

Where will the reply be sent?

If JMS inbound endpoint is request-reply type, it expects a inbound header "JMSReplyTo" . Then it send sends the response to the queue mentioned by this header.

Go to ActiveMq console and click on "send" link.

Then fill the form as shown and click in Send . Observe that ReplyTo header



You should observe that the response comes to myreplyq.

5) Now Make the outbound JMS endpoint also as "request-reply ". Where do you think that this will wait for response?

Two way outbound JMS endpoint create a temporary queue on the broker and add its name as JMSReplyTO header. Then sends the message out. It will wait for response message in that temporary queue.

The two way Inbound endpoint which received this message from broker keeps the response in the same temporary queue where the outbound endpoint is waiting for response

Now make the same HTTP POST request as you have done in step 1.

Now you should get a response as " Product Added Successfully! "

Temporary queue will be deleted after the response comes back.

6) What if you dont want a temporary response queue?

If we set an outbound property "JMSReplyTo" in the mule message before the JMS outbound endpoint, it uses this header value as Reply queue.

Try it out by setting outbound property.

Your xml should finally look like below :

```
  <http:listener-config name="HTTP_Listener_Configuration" host="0.0.0.0" port="8081"
doc:name="HTTP Listener Configuration"/>
    <jms:activemq-connector name="Active_MQ" brokerURL="tcp://localhost:61616"
validateConnections="true" doc:name="Active MQ"/>


    <flow name="jmsoutboundflow">
        <http:listener config-ref="HTTP_Listener_Configuration" path="addproduct"
doc:name="HTTP"/>
        <mulexml:dom-to-xml-transformer doc:name="DOM to XML"/>
        <set-property propertyName="JMSReplyTo" value="myreplyq" doc:name="Property"/>
        <jms:outbound-endpoint queue="productsq" connector-ref="Active_MQ"
doc:name="JMS" />
    </flow>



    <flow name="jmsinboundflow" initialState="started">
        <jms:inbound-endpoint queue="productsq" connector-ref="Active_MQ" doc:name="JMS"
exchange-pattern="request-response"/>
        <logger message="Got Order #[payload]" level="INFO" doc:name="Logger"/>
        <set-payload value="Product is Added Succesfully!!" doc:name="Set Payload"/>
    </flow>
```

Deploy the applocation and make a HTTP POST request like earlier.

You should observe that the response comes back in what ever queue name you gave as a value to "JMSReplyTo"

This is the end of the Exercise