16L71

IBM Integration Bus

MQ Flexible Topologies: Configuring MQ nodes using MQ Endpoint Policies

Featuring:

- Running MQ applications on IIB Nodes without an associated queue manager
- Generating and attaching an MQ Endpoint Policy in Toolkit
- Configuring an MQ Endpoint Policy in the Web Administration Console
- IIB Flow Exerciser testing
- Connecting to queues on an MQ Appliance

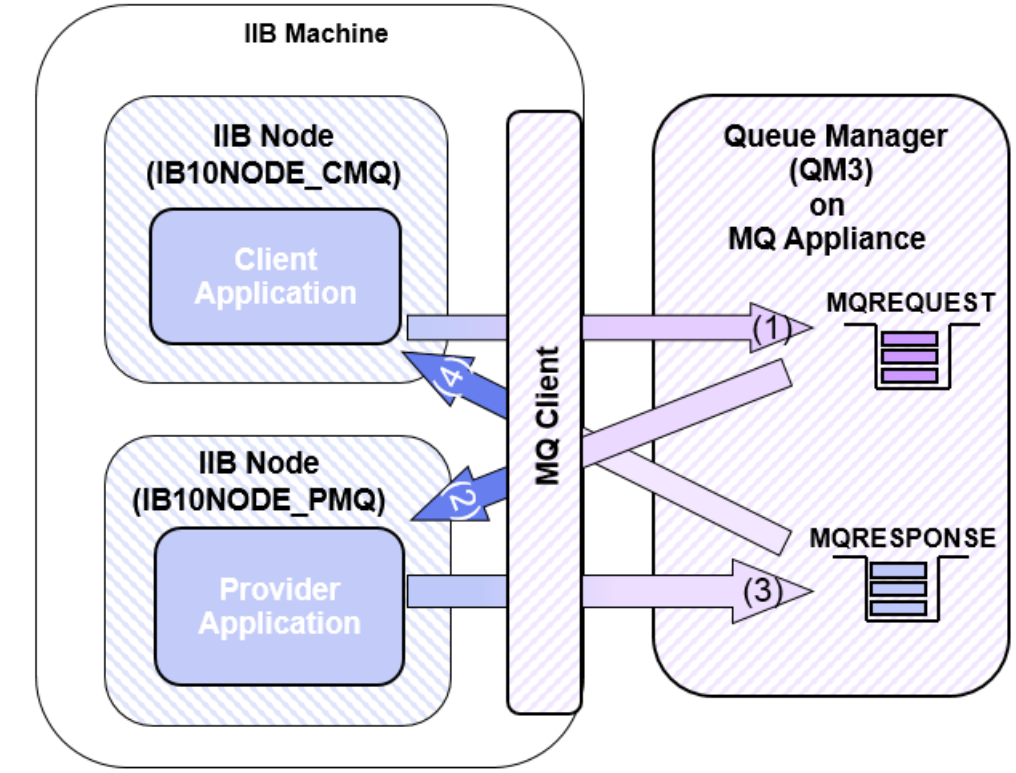
January 2016

Hands-on lab built at product Version 10.0.0.3

1. Introduction

This lab guide covers how to override MQ Connection properties using MQEndpoint policies.

In this lab guide you will configure two MQ Applications to show how MQ queues can be accessed on a remote queue manager, using an MQ Client connection. This lab will use an MQAppliance to host the queue manager, but any remote queue manager can be accessed in this way.



MQEndpoint policies in IIB V10 control operational behavior at run time. If an MQEndpoint policy is specified by an MQ node in a message flow, the properties of the policy override the properties that are set on the MQ Connection tab in the Integration Toolkit.

The Integration Toolkit generates MQEndpoint policies based on the MQ Connection properties defined on an MQ node and provides a wizard to store the MQEndpoint policy in the Integration Registry of the IIB node where the message flow will run.

Attaching the policy enables the MQ node to use the policy to override properties defined in the MQ Connection tab. To attach a policy, the MQEndpoint policy URL is specified on the Policy URL field (on the MQnode property) using the Integration Toolkit. Once the Policy URL field is set using Toolkit, the application does require deployment to the IIB node. Configuration changes to the content of the policy can either be done using the command line or web browser interfaces, and these changes happen dynamically, as soon as the modifications to the Policy are saved.

Important note

This lab, version 10.0.0.3, has been updated significantly from earlier versions. The following changes have been made:

You should use the Windows user "iibuser". This user is a member of mqbrkrs and mqm, but is not a member of Administrators. The user "iibuser" can create new IIB nodes and do all required IIB development work. However, installation of the IIB product requires Administrator privileges (not required in this lab).

The database has been changed from the DB2 SAMPLE database to the DB2 HRDB database. HRDB contains two tables, EMPLOYEE and DEPARTMENT. These tables have been populated with data required for this lab. (The DDL for the HRDB is available in the student10 folder; we intend to provide corresponding DDL for Microsoft SQL/Server and Oracle over time).

The map node now retrieves multiple rows from the database, using an SQL "LIKE" function . Additionally, the map has been refactored to use a main map and a submap. Both the main map and submap are located in a shared library.

Input to the integration service is now a simple schema containing just one element, the required employee number.

As a consequence, this version of the lab, and the associated solution, can only be used with the corresponding changes in other labs. Use version 10.0.0.3 of all labs in this series of lab guides.

1.1 The MQ applications

(*Note: these are the same application used in the Lab Guide "MQ Flexible Topology using MQ Connection Properties", the explanation of how these applications work is duplicated here for consistency*).

Client and Provider applications are supplied. The Provider application obtains requests and provides responses using queues. The Client application is driven by sending a JSON message to an HTTP Input node; responses are returned in JSON format using an HTTP Reply node.

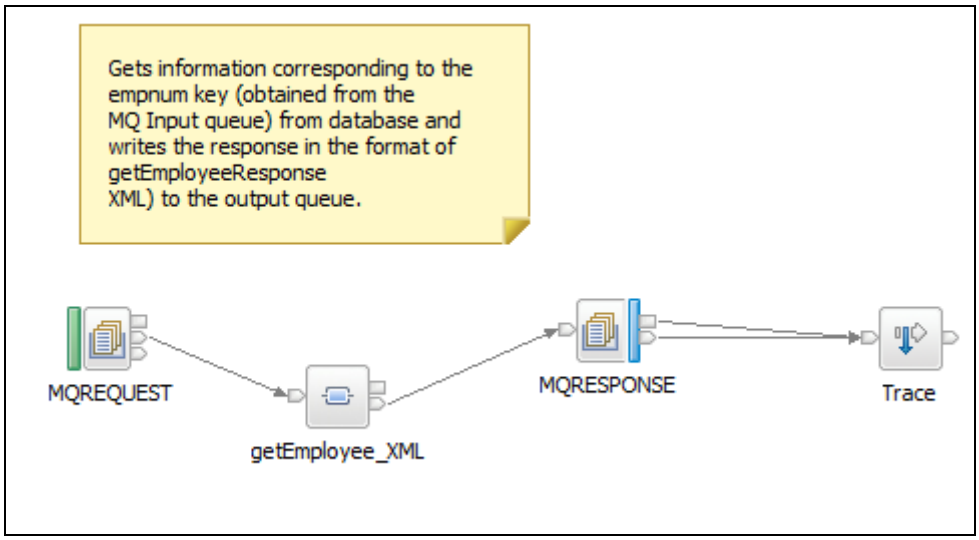
The following sections describe these applications in more detail.

NOTE: *The applications are provided purely to show (within the context of this workshop).*

1) *The flexibility of MQ in IIB V10 and*
2) *The reuse of Maps stored in a Shared Library*

The applications only work as expected when one user is submitting requests in a controlled way. A more complex Request/Response message correlation Pattern is available in the Patterns gallery, however is out of the scope of this lab guide.

1.1.1 The Provider application

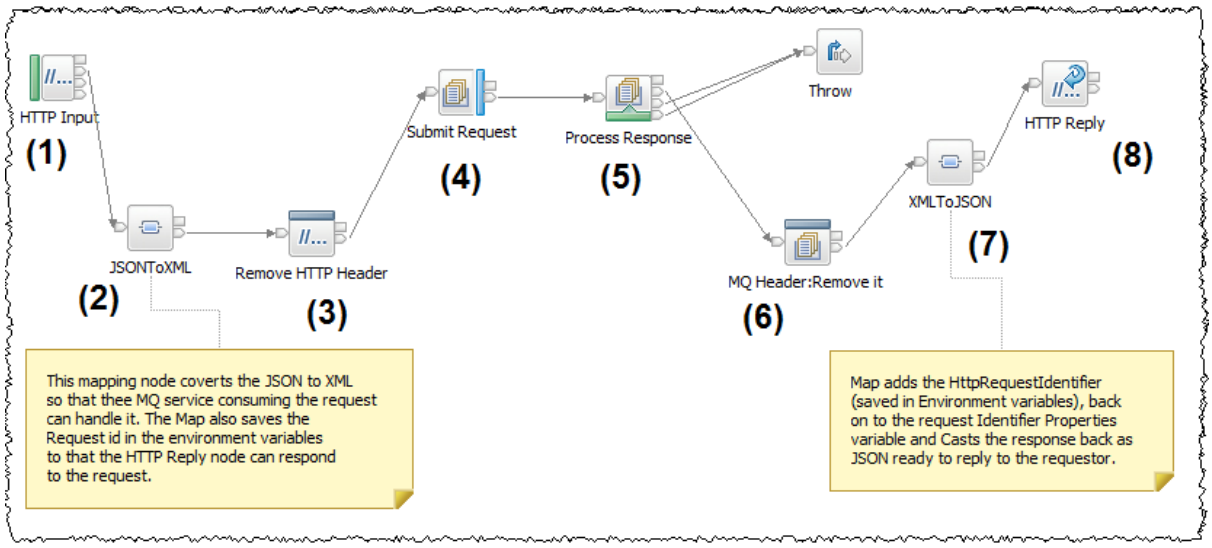


The function of the EmployeeMQProvider application is to retrieve Employee details from a DB2 database.

It will use two queues (MQREQUEST and MQRESPONSE) to handle requests and provide the responses. Request data from MQREQUEST queue is passed to a mapping node. The Mapping node uses XML data passed in the request to obtain details of the employee from the EMPLOYEE table in the database and provides XML Response data. The Response data is then written to a queue called MQRESPONSE.

The Mapping node is supplied in a Shared Library. The lab guide will demonstrate how to reuse assets previously created and stored in a Shared Library in V10.

1.1.2 The (MQ) Client application



The EmployeeService_JSON_MQClient contains a message flow that:

1) Accepts a JSON request from an HTTP Input node
2) Converts the JSON to the required XML format for the EmployeeMQProvider to process the request using a mapping node. Note this node also demonstrates a new Graphical Data mapping feature in IIB V10 where it is possible to address the IIB Environment tree in the mapping node. The map saves the HTTP Request ID in Environment variables so that the HTTP Reply node works correctly after removing the HTTP headers from the Message tree in the scenario.
3) Removes the HTTP Headers
4) Writes the XML version of the request to the MQREQUEST queue
5) Waits for XML response data to appear on the MQRESPONSE queue
6) Removes the MQ headers from the Message Tree
7) Uses a second mapping node:
    a. Transforms the XML provided through the MQRESPONSE queue back to JSON
    b. Reinstates the data saved HTTP Request ID from the Environment Variables into the message tree so that the HTTP reply node can work correctly.
8) Provides the Response data back to the requestor as JSON data.

2. Reset the IIB nodes

Login to Windows as the user "iibuser", password = "passw0rd".

Start the IIB Toolkit from the Start menu.

1. Stop TESTNODE_iibuser (right click on the node in the Integration Toolkit and click Stop).

2. If the IB10NODE_CMQ and IB10NODE_PMQ nodes are not defined on your system (***note there is no need to do this if you have the nodes already defined***), open an Integration Bus Console and Navigate to:

   ```
   C:\student10\MQ_Topology\commands
   ```

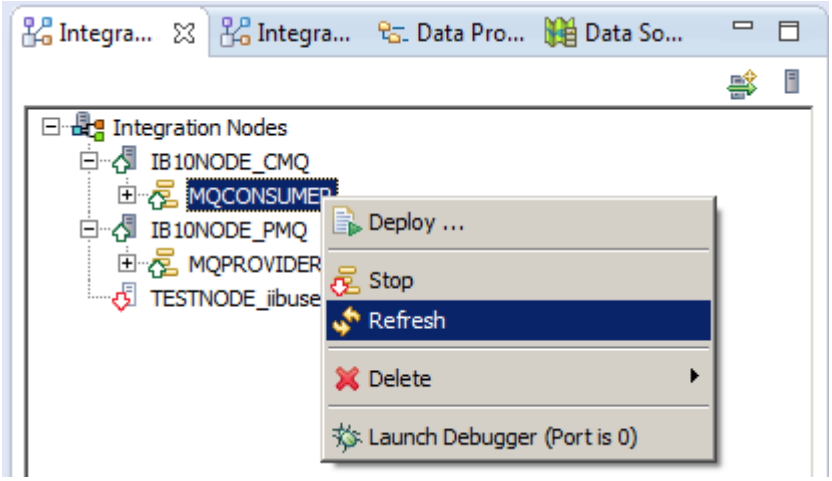
   Run the command:

   ```
   00CreateMQIIBNodes.cmd
   ```

   Accept the defaults in the script and respond to any prompts.

   This command file will create the two IIB nodes and corresponding integration servers.

3. In the Toolkit, the nodes should look like this (if you have restarted the nodes, you may need to refresh their status, as shown).



3. Use the MQ Appliance Console to add queues

The MQ applications you will be using in this environment will be configured to communicate with a queue manager QM3. QM3 has been predefined on an MQ Appliance in your environment.

In this section you will configure the MQ Console and add additional queues to the queue manager using the MQ appliance widgets.

3.1 Find the IP address of the MQ Appliance

4. Open the MQ Appliance console and sign in using
   user=**admin** and password=**passw0rd**

   (*If the screen is blank and there appears to be no prompt, click the centre of the black screen and then press the enter key*)

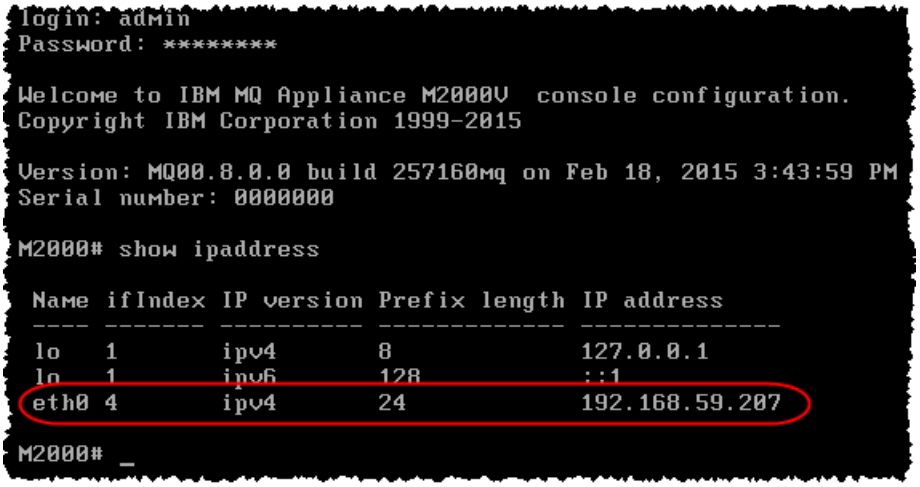
5. At the "**M2000#**" prompt type in:

   show ipaddress

6. The MQ Appliance you are using is configured with one Ethernet adapter (***eth0***).

   Write down the IP address associated with this entry in the table:

   MQ Appliance IP address _____



(*in the above example the ip address is 192.168.59.207*)

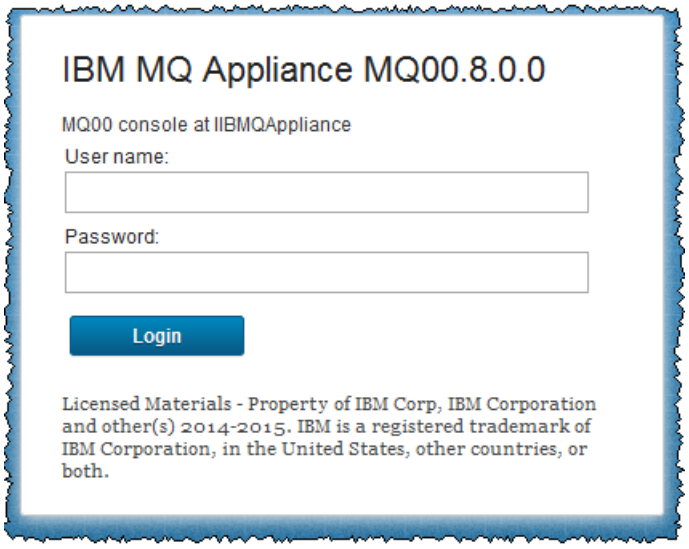
3.2 Configure the MQ Appliance Console

1. In the IIB VM, open a Firefox browser tab and enter the following (*replace the IP address with the specific address you found in the previous section*):

   `https://<yourIPAddress>:9090`

   If the browser request responds with an untrusted connection warning, follow the prompts to accept the risks.

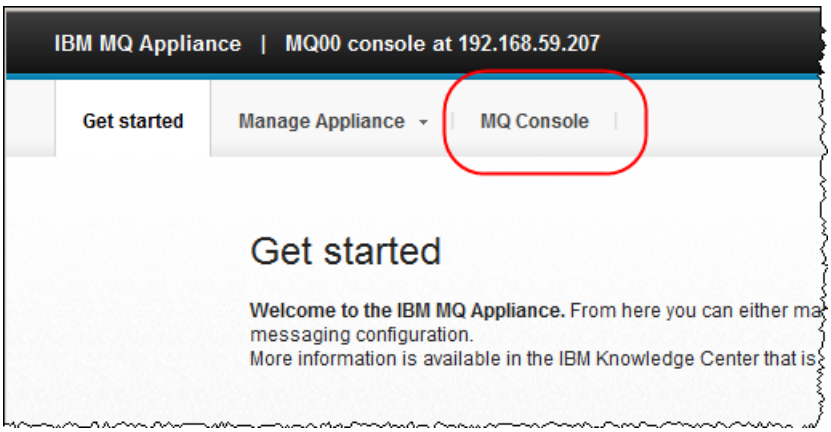
2. The MQ Appliance login console will appear:



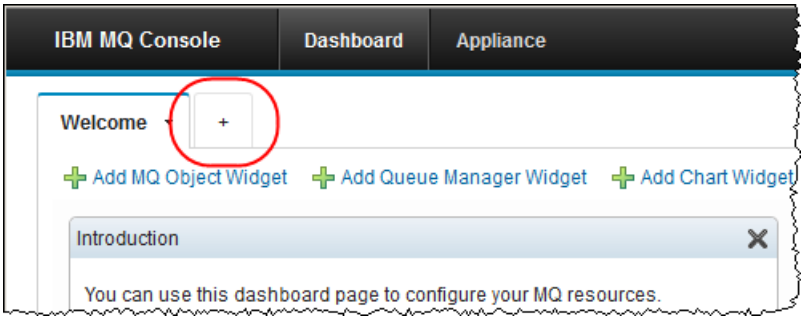
3. Login using:

   user= **`admin`**
   password= **`passw0rd`** (with a zero)

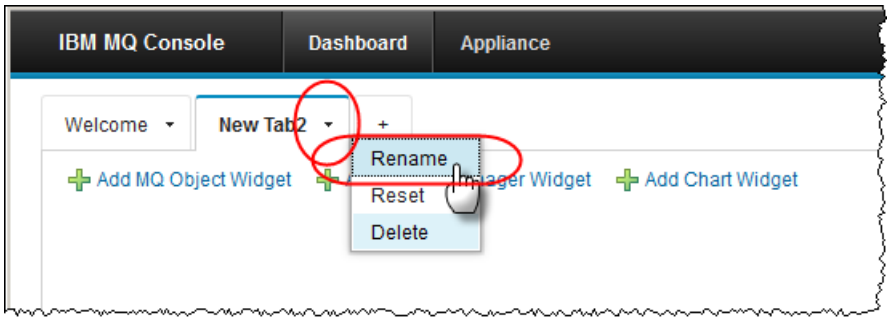
4. On the main page, click the MQ Console tab:



5. This will show the MQ console Dashboard. Click on the tab with a “+” sign to add a new tab:



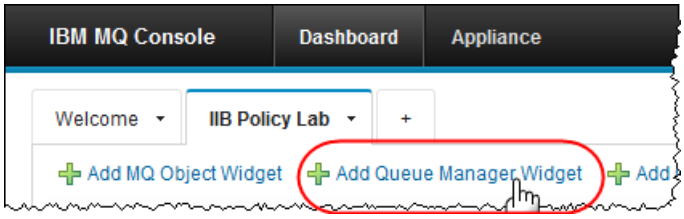
6. Rename the tab “**`IIB Policy Lab`**” (Click on the small triangle to list the tab options and chose “Rename”).



3.3 Add Widgets to your console

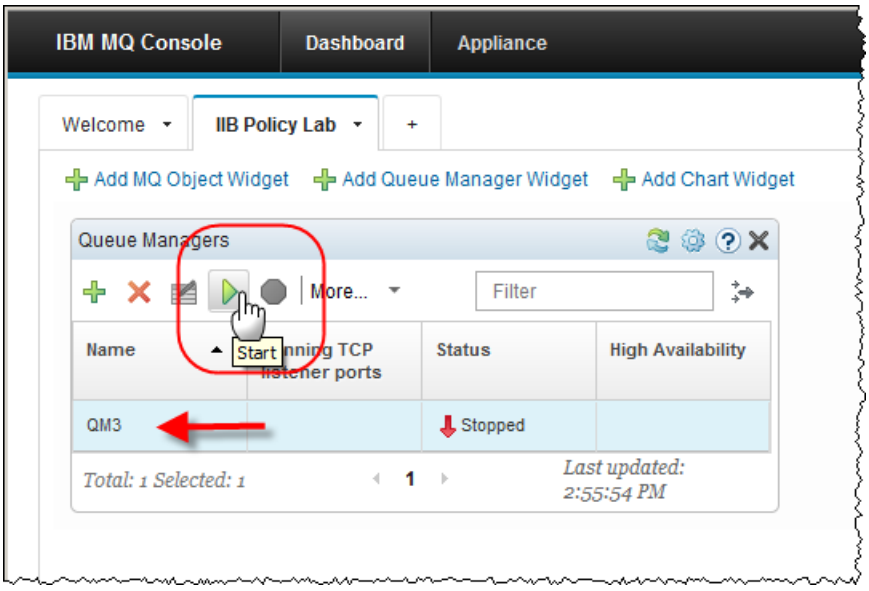
3.3.1 Queue Manager Widget

1. Click “Add Queue Manager Widget” to add a widget that will display queue managers to the tab:

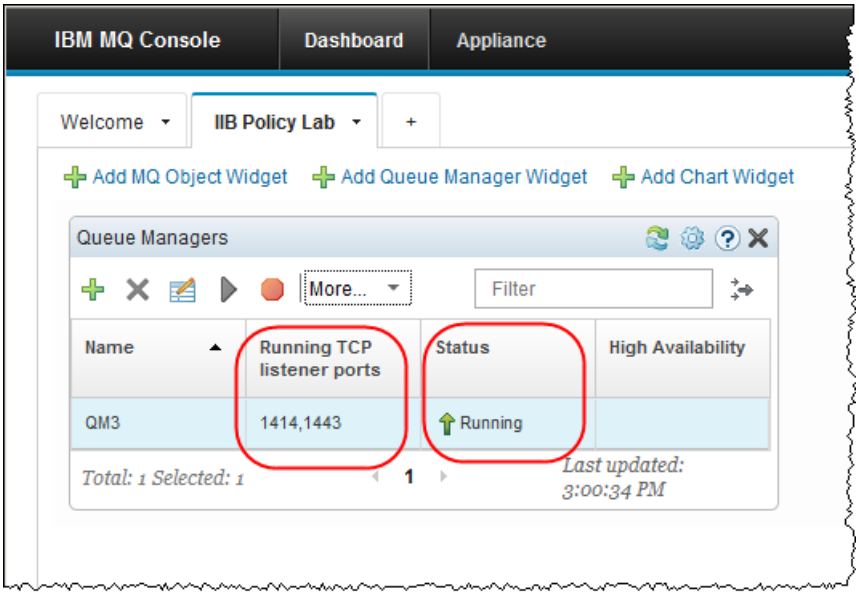


2. The widget will display the predefined queue manager “QM3” and its status.

   Click QM3 and the start icon to start the queue manager:

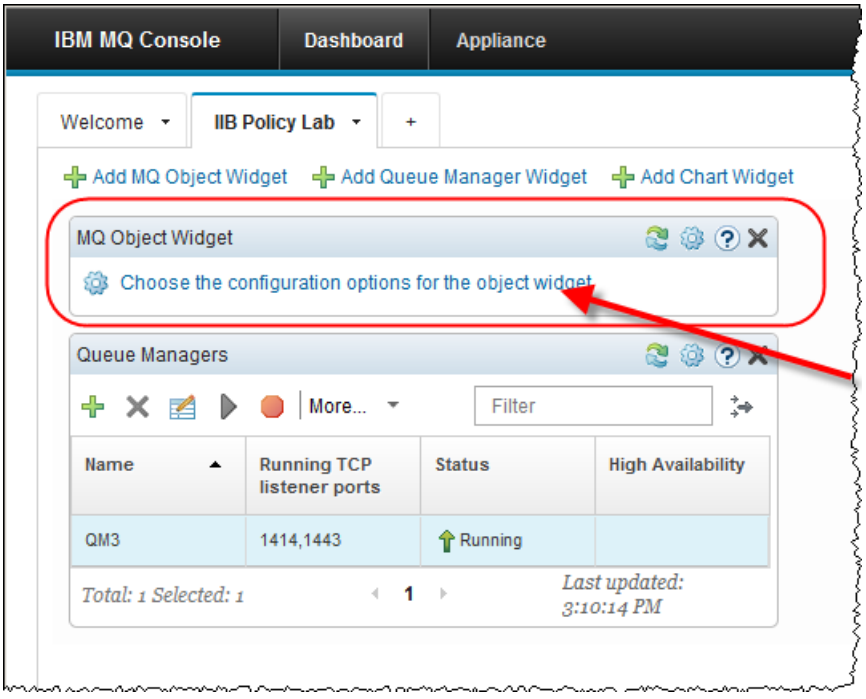


3. Once started the queue manager widget will display the change in status and the running TCP listener ports:

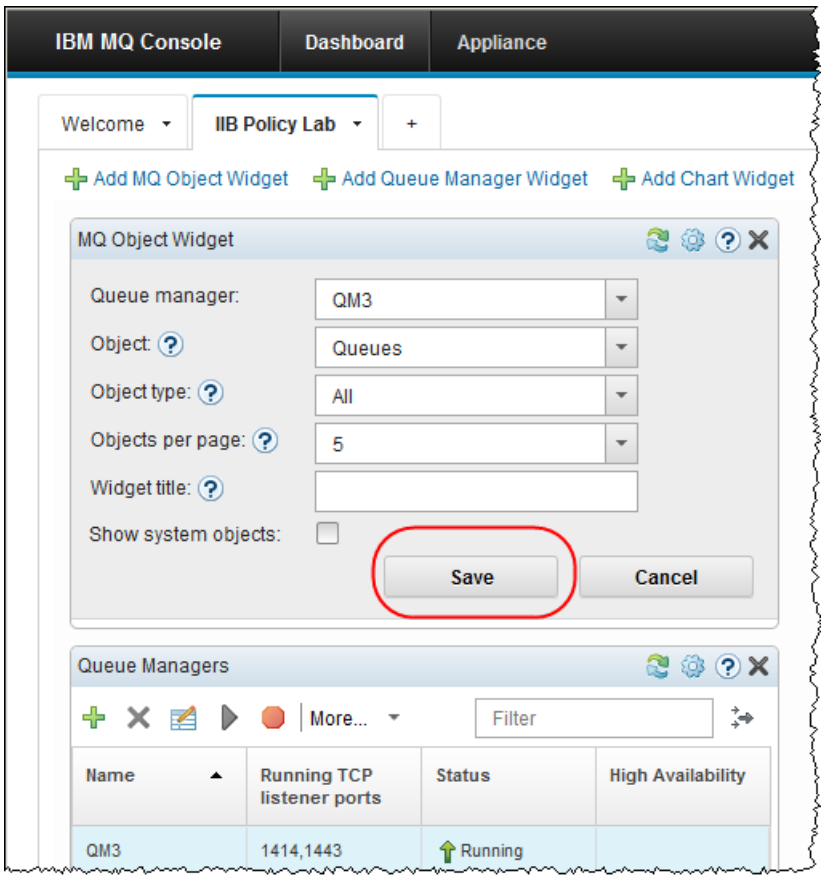


3.3.2 Add a Widget to show queues

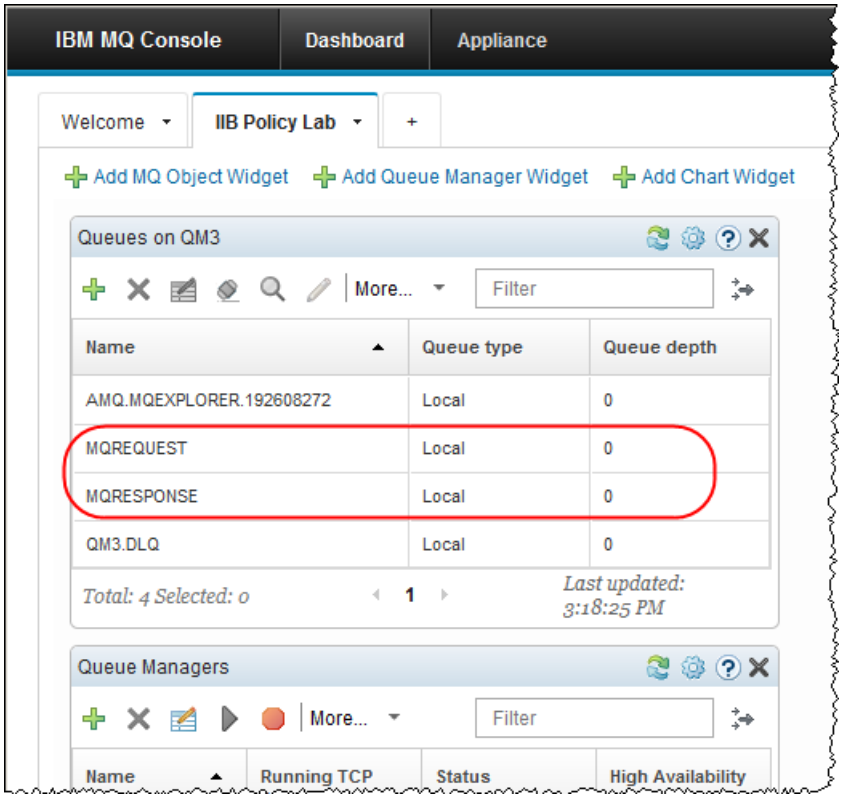
1. Click on the “Add MQ Object Widget”, This will add an MQ Object Widget:



2. Click "**Choose the configuration options for the object widget**" and configure it to show all queues on the queue manager, then click save:



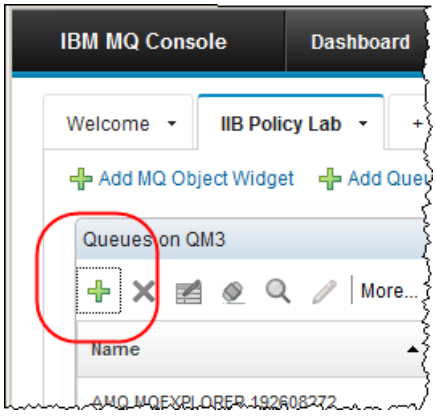
3. The queue widget will update and display the queues on the queue manager:



The queues you will use in this lab guide are MQREQUEST and MQRESPONSE.

3.3.3 Add two queues using the Queue Widget

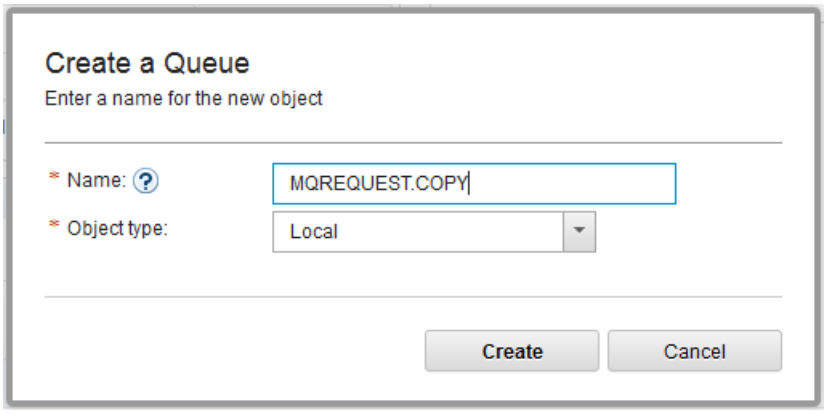
1. Click on the green plus sign in the “Queues on QM3” widget:



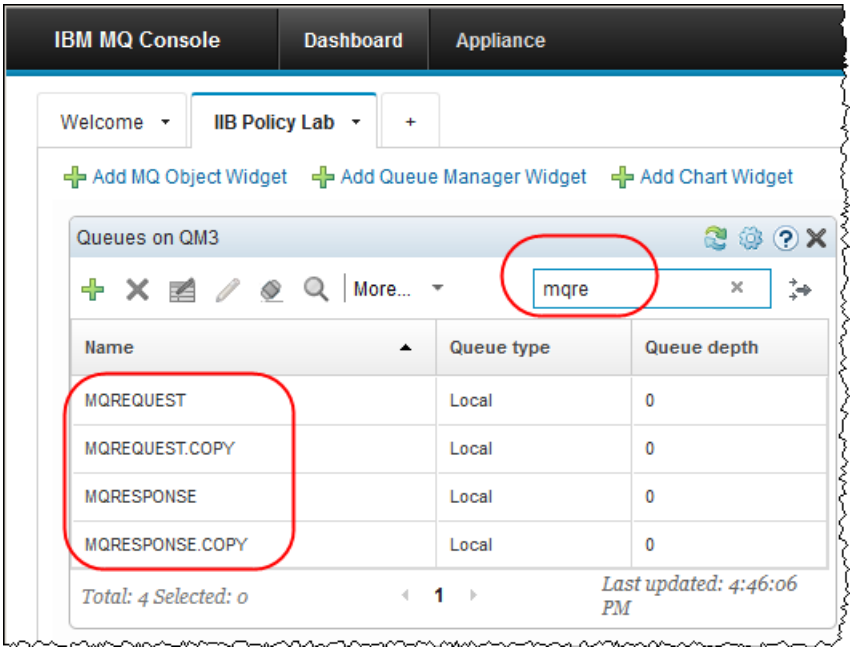
2. Add **two** further “Local” queues to the queue manager called:

   `MQREQUEST.COPY` and
   `MQRESPONSE.COPY`:

   For example:



3. In the "**Queues on QM3**" widget type "**`mqre`**" to show the four queues you will be using in this lab guide:

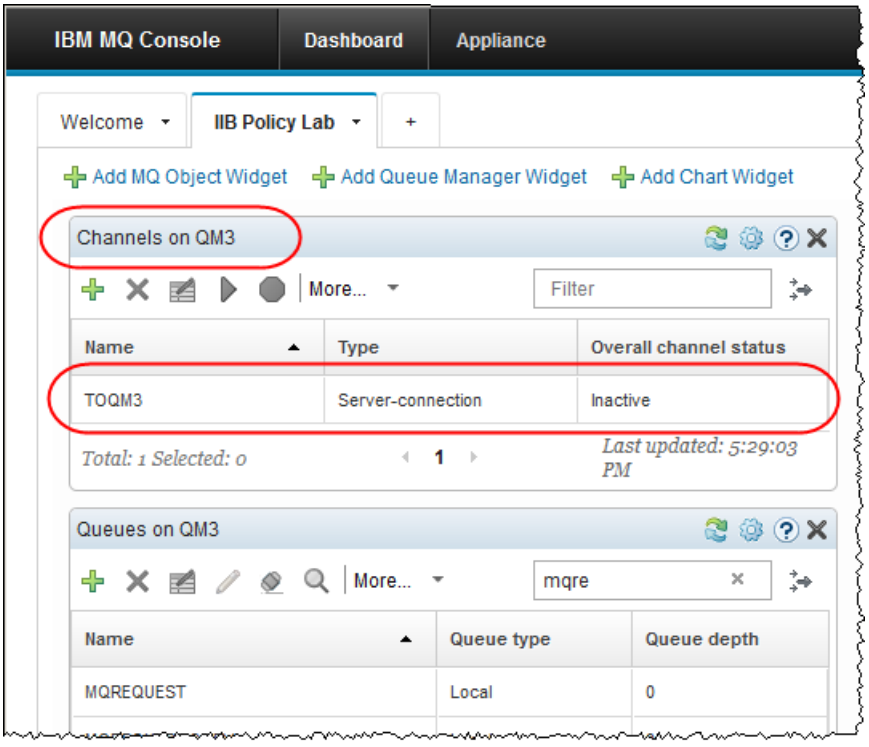


3.3.4 Add a widget to show Channels

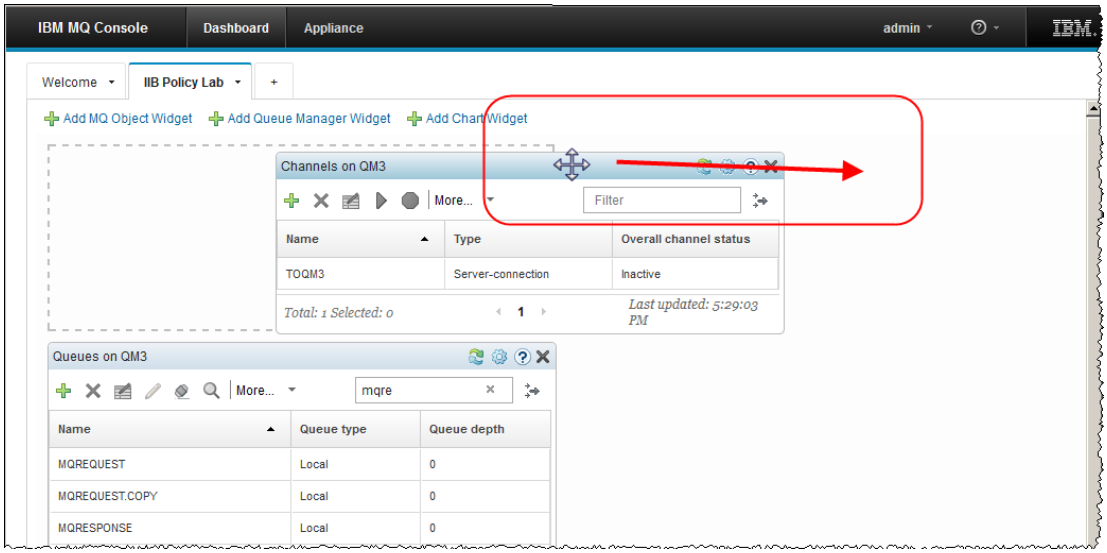
1. Add another MQ Object Widget to show the channel definitions on QM3.

   a) Click "Add MQ Object Widget"
   b) Click "Choose the configuration options for the object widget"
   c) Change Object to Channels
   d) Click Save

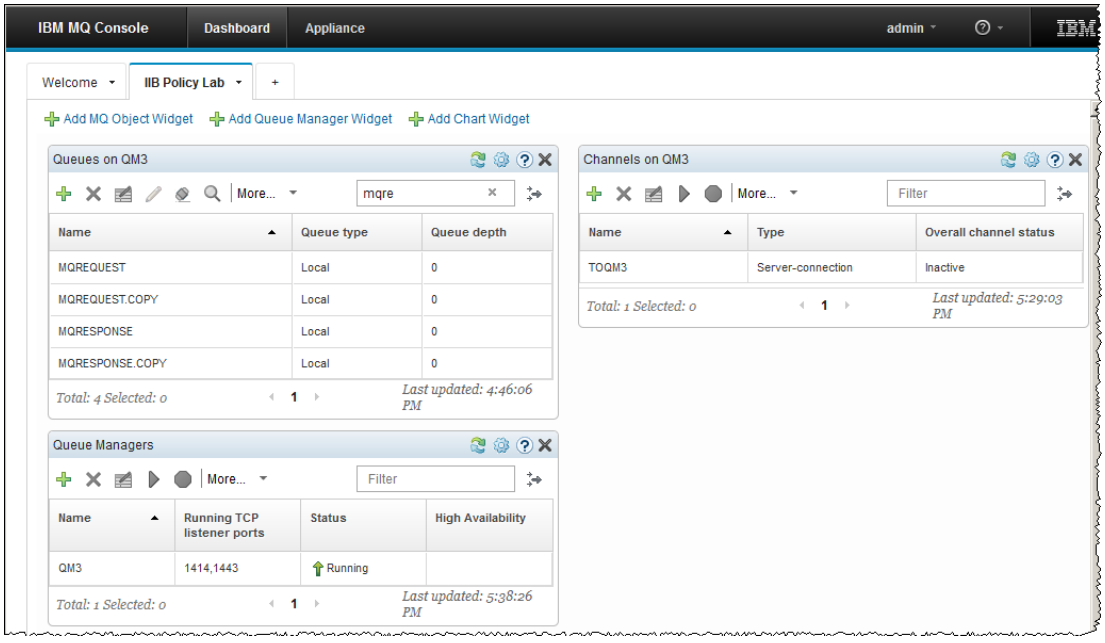
   The widget will show the Channel definition called "**`TOQM3`**", its status will be "Inactive":



2. Click the blue header of the "**Channels on QM3**" widget and move it to the right hand side of the "IIB Policy Lab":



3. The widgets "Queues on QM3" and "Queue Managers" will resposition themselves. The "IIB Policy Lab" tab will now look like this:



4. Customise the JSON_MQCLIENT application

4.1 Add two MQOutput nodes to the JSON_MQClient

In this section you will add two MQOutput nodes to the JSON_MQClient message flow. The queues will be used to copy the Requests and Responses made to the MQ provider application.

1. To avoid naming clashes with earlier labs, this lab will be developed using a new workspace.

   If you already have a workspace open, click File, Switch Workspace. Give the new workspace the name

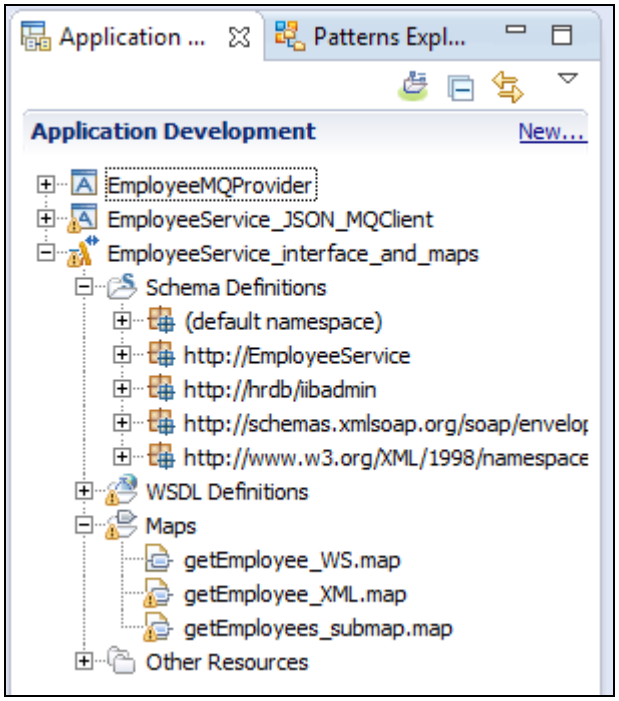
   `c:\users\iibuser\IBM\IIB 10\workspace_MQPolicy`

2. Import the PI file

   `c:\student10\MQTopology\solution\EmployeeService_JSON_MQClient(QM2-ServerConn).10.0.0.3.zip`

   Import all projects in the PI file.

   Your workspace will be updated and look like this:



3. Open the JSON_MQClient.msgflow in the message flow.

   Add two MQOutput nodes, as shown highlighted below, with the following properties:

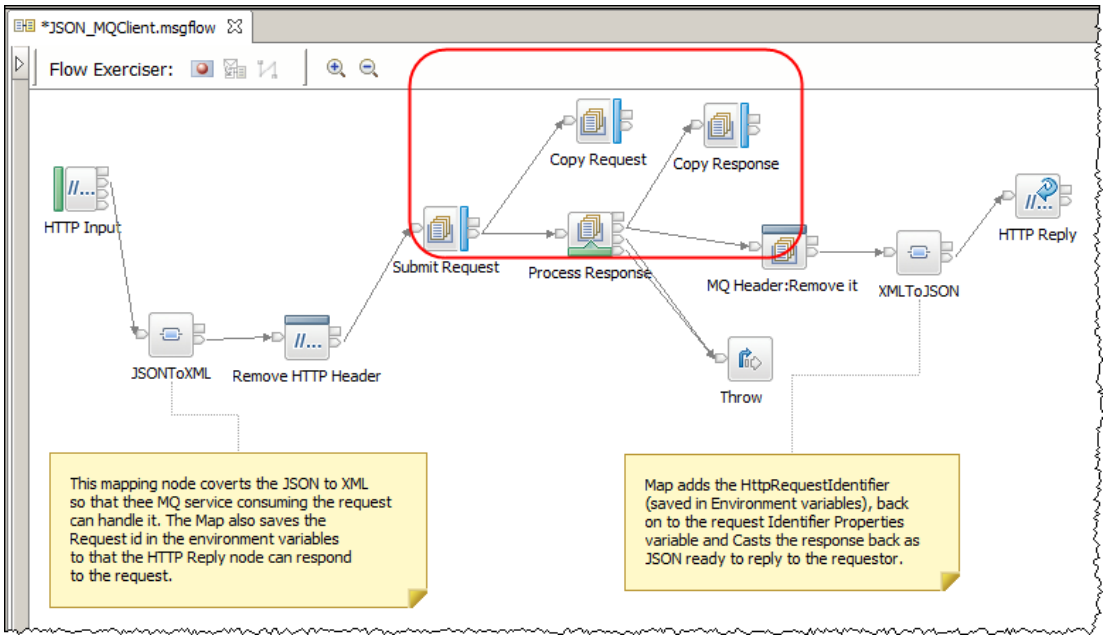
   MQ Output node: Copy Request

   Node Name (*Description tab*): **Copy Request**
   Queue name (*Basic tab*): **MQREQUEST.COPY**
   Connection (*MQ Connection tab*) **: Local queue manager**
   Destination queue manager (*MQ Connection tab*): **QM2**
   Terminal Connection: **Connect Out terminal on "Submit Request" node to this node**

   MQ Output node: Copy Response

   Node Name (*Description tab*): **Copy Response**
   Queue name (*Basic tab*): **MQRESPONSE.COPY**
   Connection (*MQ Connection tab*) **: Local queue manager**
   Destination queue manager (*MQ Connection tab*): **QM2**
   Terminal Connection: **Connect Out terminal on "Process Response" node to this node**

   The message flow will look like this when you have completed this step:



4. Save the message flow (ctrl , s)

4.2 Create and attach an MQ Endpoint Policy

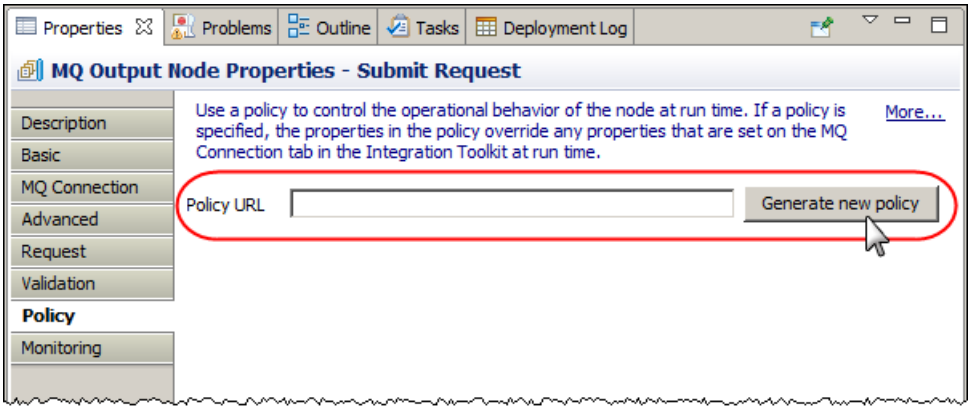
In this section you will configure the MQ nodes used in the JSON MQClient application to use an MQEndpoint policy.

1. Click on the MQOutput node “Submit Request”.

   In the Properties view, click the “Policy” tab.

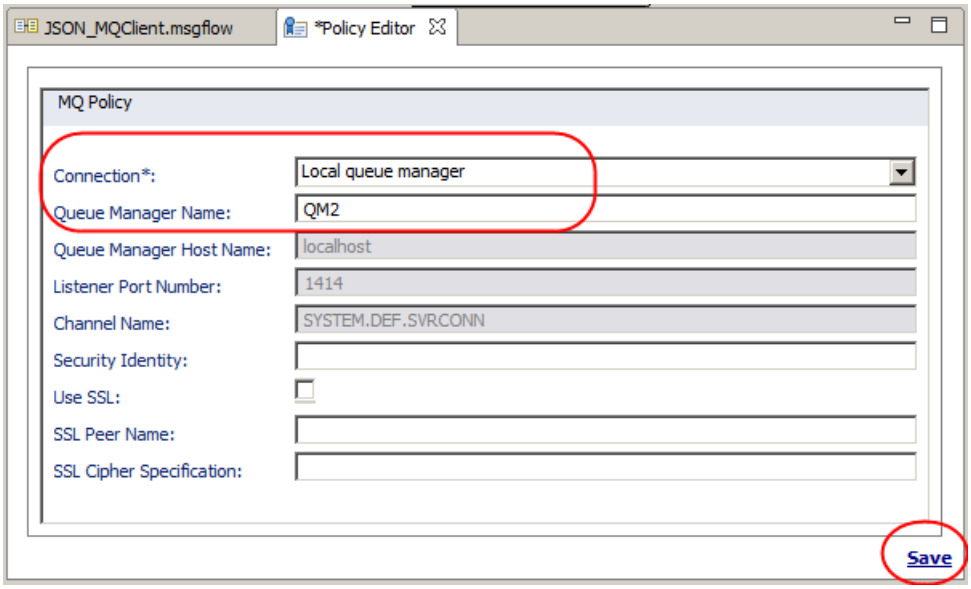
   Note the Policy URL field is blank.

   Click the “Generate new policy” button:



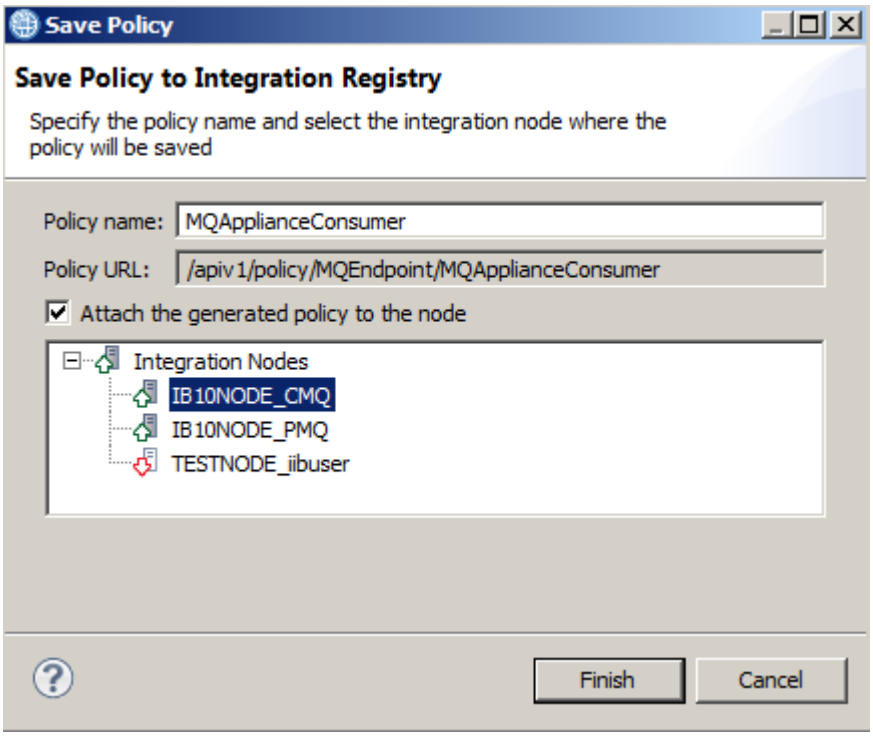
2. The Policy Editor will open. Note the definitions are copied from the current MQ Connections tab.

   Click “Save”:

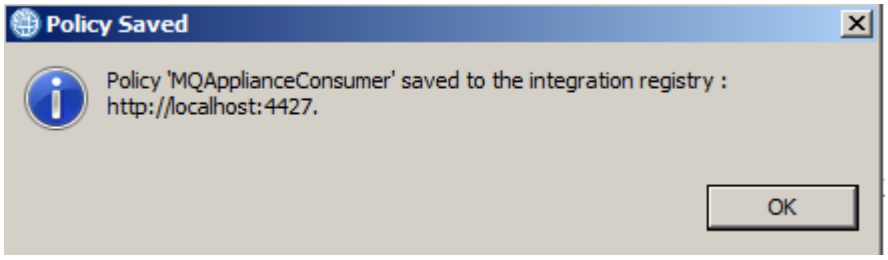


3. In the save window,

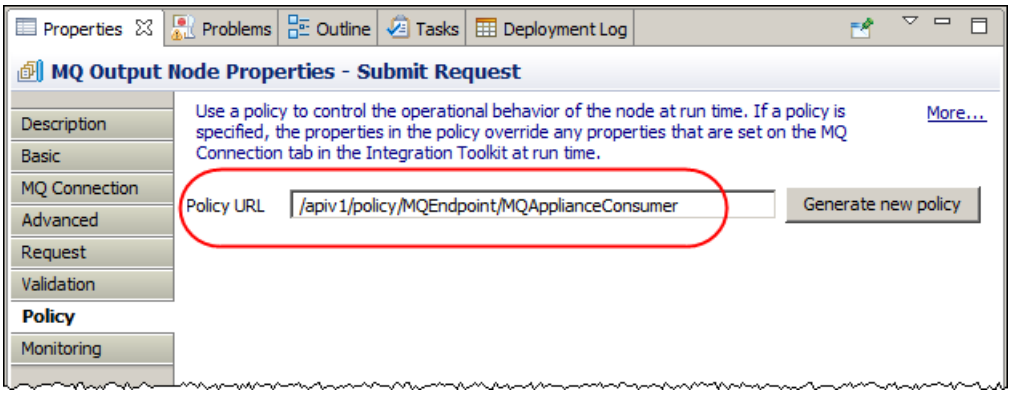
   a) Call the Policy “MQApplianceConsumer”.
   b) Click the link “**Configure hostname and port by selecting the integration node**”
   c) In the “**Select Integration Registry**” window, Select **IB10NODE_CMQ** and click Finish:



4. Click OK on the “Policy Saved” window:



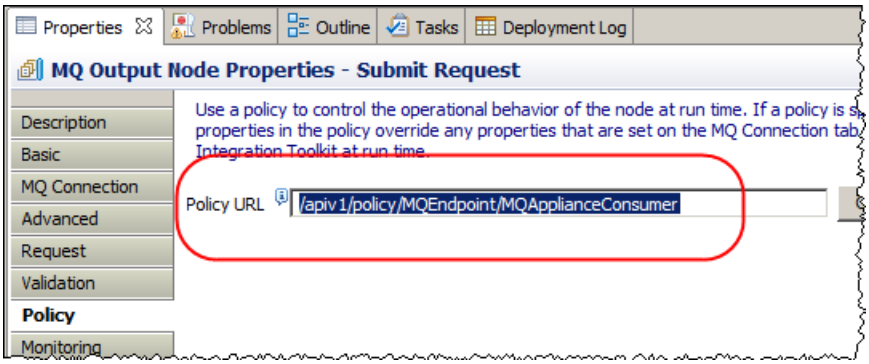
5. Note the “Submit Request” MQOutput node properties (Policy tab) now contains a Policy URL:



4.3 Attach the MQEndpoint Policy to all MQ nodes

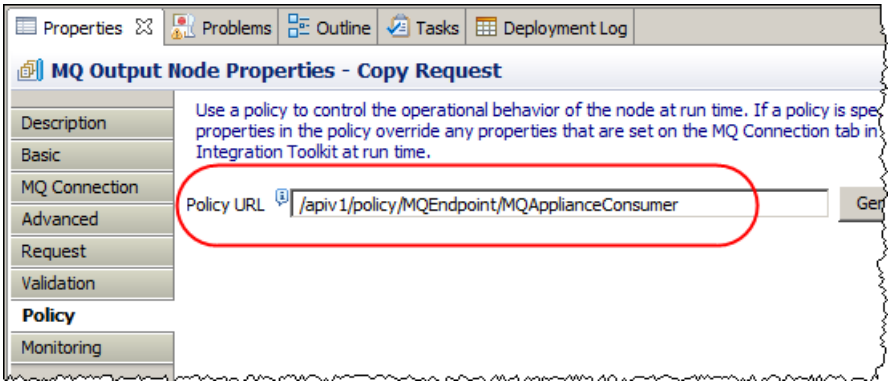
Once generated, an MQEndpoint policy can be “attached” to other MQ nodes which run in the same IIB node. You will now attach the MQEndpoint policy to all the MQ nodes in the JSON_MQClient message flow.

1. Copy the Policy URL into the click board (highlight the policy URL field with <Ctrl-A> and then <Ctrl-C>).



2. In the message flow, click on the MQ Output node called “**Copy Request**”.

   In the Properties tab, click Policy URL and paste (right click, choose “paste”) the contents of the clip board into the Policy URL field:



3. Repeat the above step on:

   a) the MQGet node “**Process Response**”
   b) the MQOutput node “**Copy Response**”.

4. Save the message flow (ctrl s).

All MQ nodes in the message flow now have the **MQApplianceConsumer** MQEndpoint policy attached.

5. Customise the Provider Application

5.1 Copy an existing MQEndpoint Policy

The MQ Provider application will now be configured to use an MQEndpoint policy. A command line interface exists that enables:

1) Exporting an existing MQEndpoint policy and
2) Creating an MQEndpoint policy on a different IIB node.

In the following section you will export the MQEndpoint policy “MQApplianceConsumer” that you just created for the MQ Consumer application. The exported policy will then be used to create a new MQEndpoint Policy “MQApplianceProvider”. This will then be attached to the MQ nodes in the MQ Provider application.

1. In an IIB Command Console navigate to **c:\student10\MQ_Policy** and type the following command:

```
mqsireportpolicy IB10NODE_CMQ
                 -t MQEndpoint
                 –l MQApplianceConsumer
                 -f MQApplianceConsumer.xml
```

   The response of the command will be to display the current values of the MQEndpoint policy:

```
BIP1895I: Policy type 'MQEndpoint' Policy name 'MQApplianceConsumer' Policy URI
'/apiv1/policy/MQEndpoint/MQApplianceConsumer' Policy content '<?xml version="1.
0" encoding="UTF-8" standalone="yes"?><policy type="MQEndpoint">    <policyPrope
rties>        <mqConnectionDetailsPolicy>           <connection>SERVER</connect
ion>            <destinationQueueManagerName>QM2</destinationQueueManagerName>
          <useSSL>false</useSSL>        </mqConnectionDetailsPolicy>    </policy
Properties></policy>'

BIP8071I: Successful command completion.
```

   The policy XML will have been written to the file “MQApplianceConsumer.xml”

2. To use the exported MQEndpoint policy definition to create an MQEndpoint Policy on **IB10NODE_PMQ** enter the following command:

```
mqsicreatepolicy IB10NODE_PMQ
                 -t MQEndpoint
                 –l MQApplianceProvider
                 -f MQApplianceConsumer.xml
```

   The response from the command should be :

```
BIP8071I: Successful command completion.
```

3. To see the new MQEndpoint policy type in the following command:

```
mqsireportpolicy IB10NODE_PMQ
                 -t MQEndpoint
                 -r
                 > MQApplianceProvider.out
```

The MQEndpoint policy XML will be written to the specified file.

The output will have the contents similar to this:

```
BIP1895I: Policy type 'MQEndpoint' Policy name 'MQApplianceProvider' Policy URI
'/apiv1/policy/MQEndpoint/MQApplianceProvider' Policy content '<?xml version="1.
0" encoding="UTF-8" standalone="yes"?><policy type="MQEndpoint">    <policyPrope
rties>        <mqConnectionDetailsPolicy>           <connection>SERVER</connect
ion>            <destinationQueueManagerName>QM2</destinationQueueManagerName>
         <useSSL>false</useSSL>        </mqConnectionDetailsPolicy>    </policy
Properties></policy>'


BIP8071I: Successful command completion.
```

4. Open the file MQApplianceProvider.out, and copy the Policy URI field into the Windows clipboard.

```
/apiv1/policy/MQEndpoint/MQApplianceProvider
```

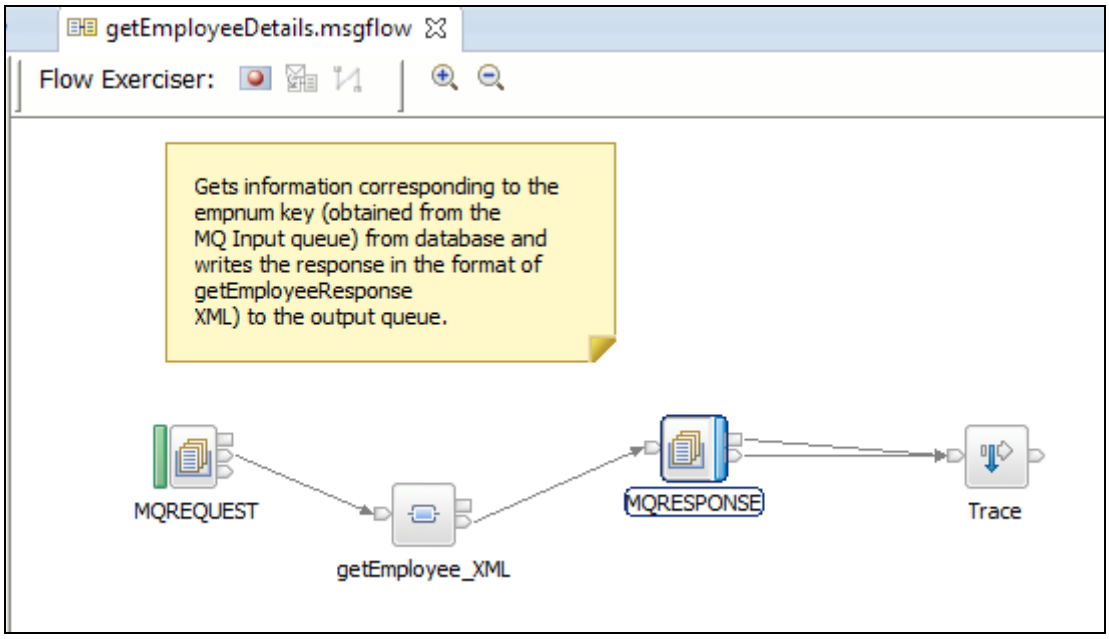
(don't copy the single quotes)

You will paste this value into the "**Policy**" field in Integration Toolkit in order to attach the policy to the MQnodes in the Provider message flow.

5.2 Attach the MQEndpoint Policy

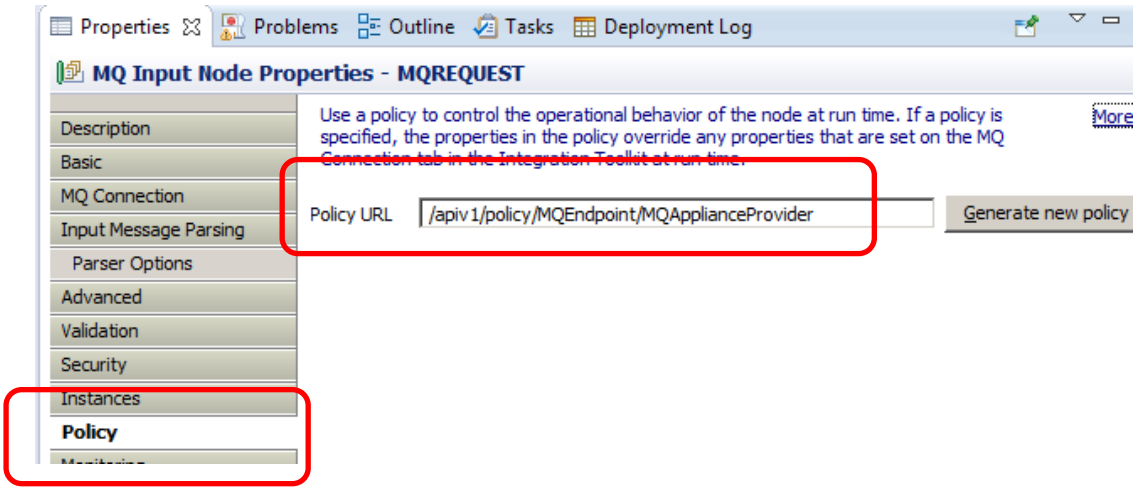
In the following section you will attach the MQEndpoint policy "MQApplianceProvider" to the MQ nodes in the Provider application using the Integration Toolkit.

1. In the Integration Toolkit open the message flow **getEmployeeDetails** (in the EmployeeMQProvider application):

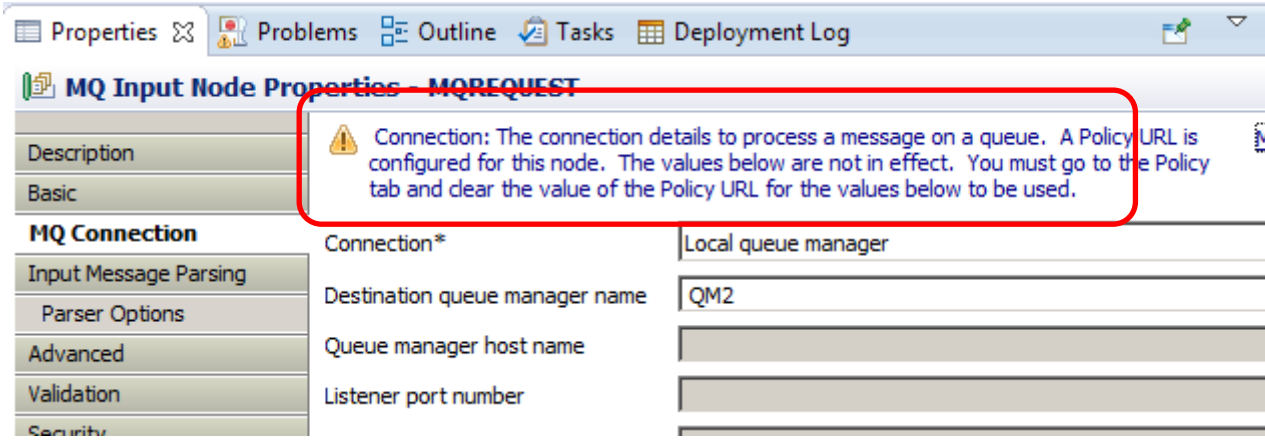


2. Click the MQInput node "**MQREQUEST**".

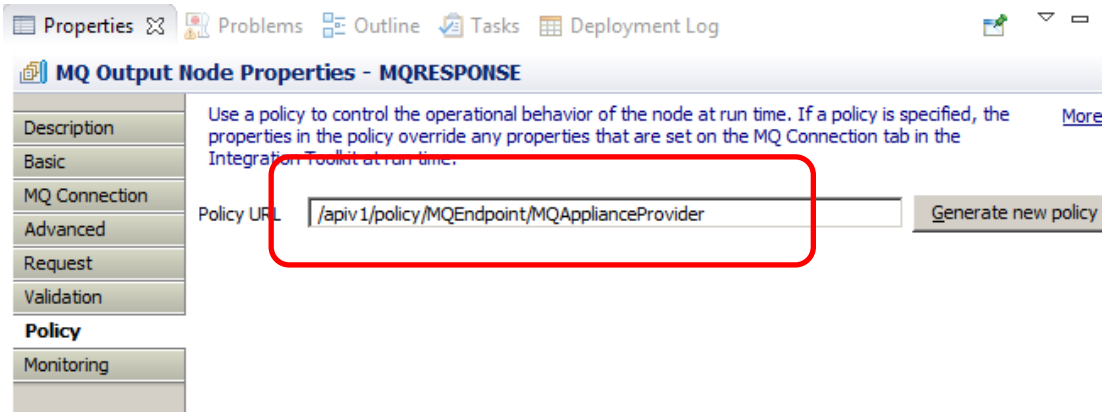
   In the node Properties Policy tab, paste the URI you just copied into the Policy URL field:



3. On the MQ Connection tab, note the message that now appears warning that a Policy URL is configured for the node and that the values in the MQ Connection Tab are not in effect:



4. Repeat the previous step for the MQOutput node "**MQRESPONSE**"



5. Save the message flow (Ctrl-S).

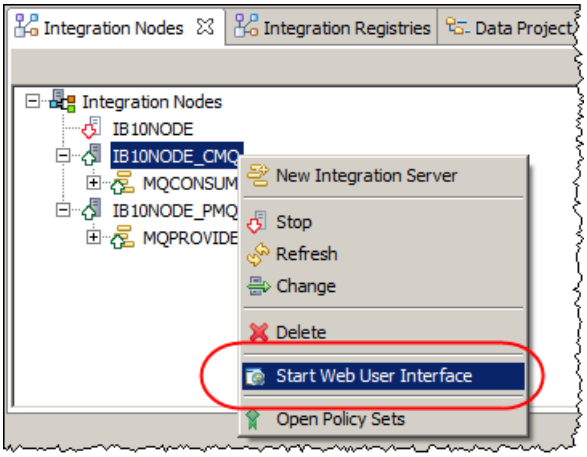
6. Modify the MQEndpoint Policies

The MQ Consumer and Provider applications are now configured to use the MQApplianceConsumer and MQApplianceProvider policies respectively.

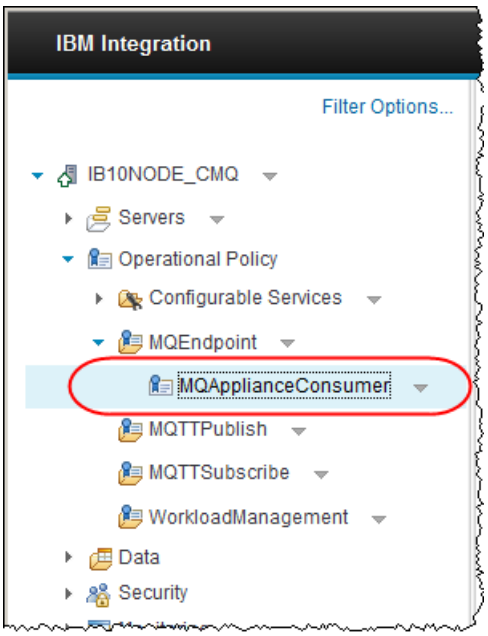
You will now modify the two MQEndpoint policies to override the initial configuration taken from the MQ nodes they were based on.

6.1 Configure MQApplianceConsumer

1. In the “Integration Nodes” view in Toolkit, right click on the IB10NODE_CMQ node and Start the Web User Interface (if you see a security exception, choose the option to “accept the risks”):



2. In the Web UI, expand **IB10NODE_CMQ** > **Operational Policy** > **MQEndpoint**

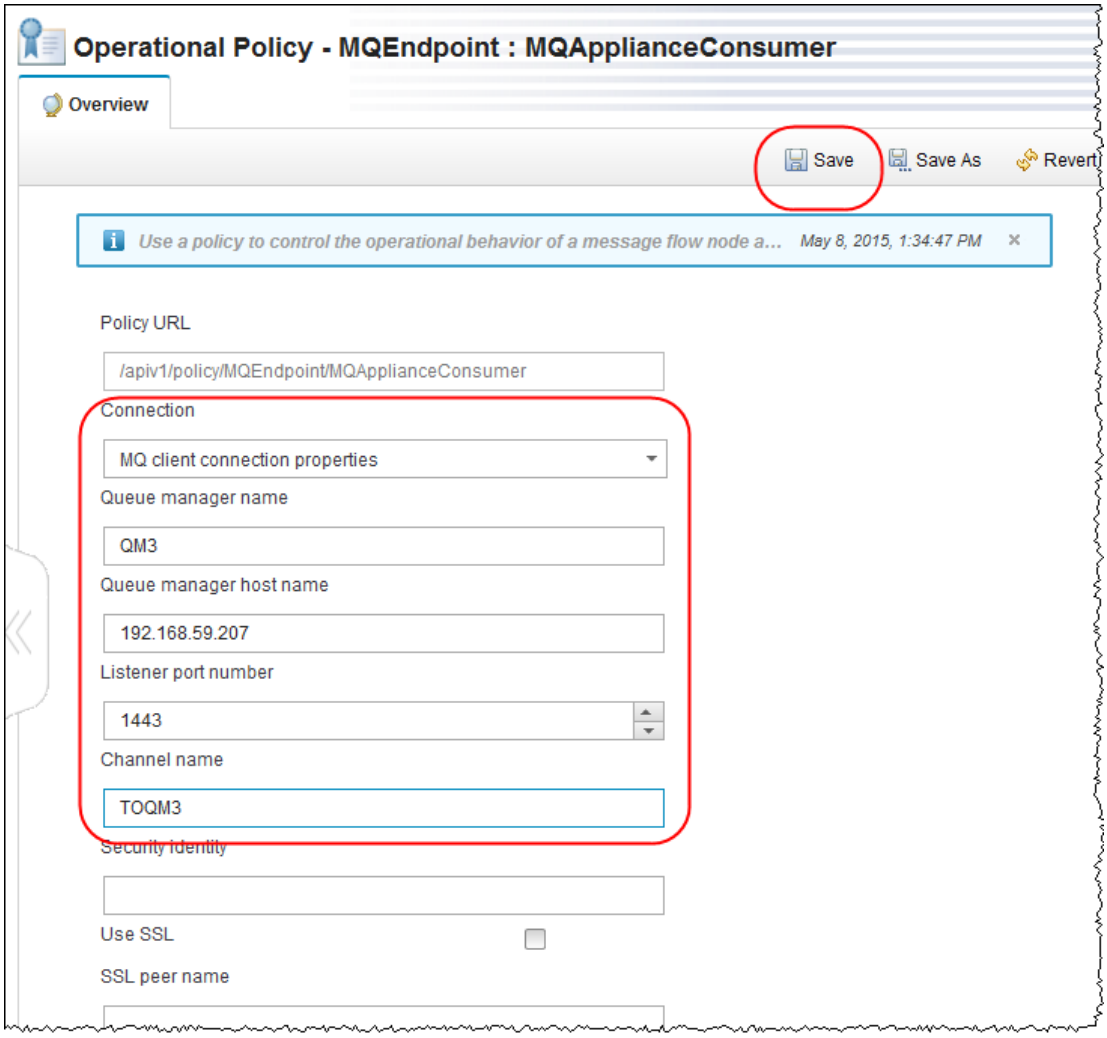


Click “**MQApplianceConsumer**” to show the MQEndpoint policy details.

3. Configure the policy with the following details:

   Connection: “MQ client connection properties”
   Queue manager name: QM3
   Queue manager host name: *<your MQ Appliance IP address>*
   Listener port number: 1443
   Channel name: TOQM3

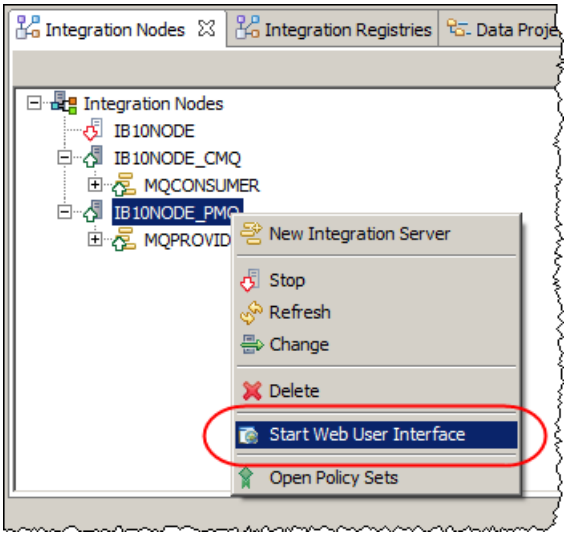
   Click Save when complete.



6.2 Configure MQApplianceProvider

In this section you will repeat the above steps to modify the policy defined in the Integration Registry associated with IB10NODE_PMQ.

1. In the “Integration Nodes” view in Toolkit, right click on the IB10NODE_PMQ node and Start the Web User Interface:



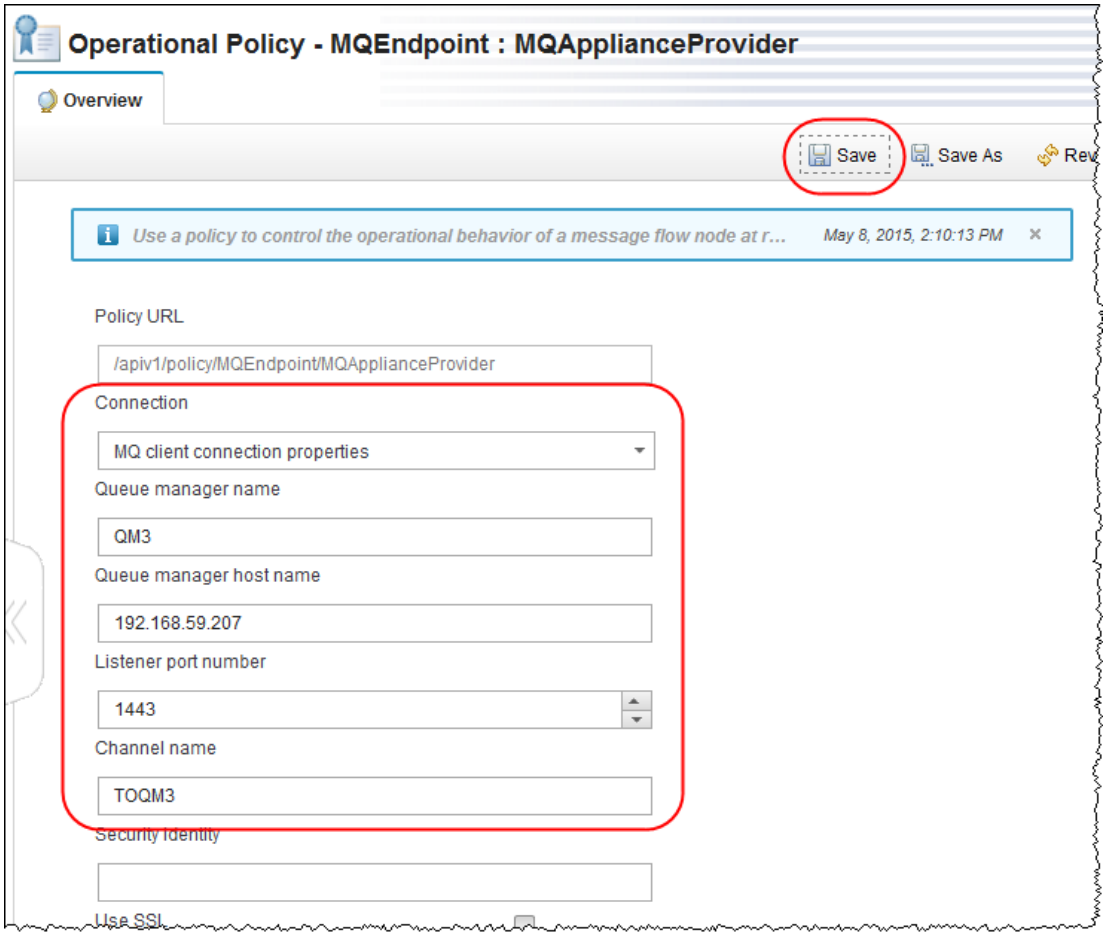
2. In the Web UI, expand **IB10NODE_PMQ** > **Operational Policy** > **MQEndpoint**

   Click “**MQApplianceProvider**” to show the MQEndpoint policy details.

3. Configure the policy with the following details:

Connection: “MQ client connection properties”
Queue manager name: QM3
Queue manager host name: *<your MQ Appliance IP address>*
Listener port number: 1443
Channel name: TOQM3

Click Save when complete.



7. Test the applications

The MQ Consumer and Provider applications have the two queues MQREQUEST and MQRESPONSE configured as being on Local queue manager QM2 (on the MQ node Connection properties in the Integration Toolkit).

In the previous section two MQEndpoint policies were configured and attached to override the MQ Connection properties on the MQ nodes defined in the two application message flows. The MQEndpoint policies are now configured to use the queues on QM3 on the MQ Appliance.

Two additional queues were defined MQREQUEST.COPY and MQRESPONSE.COPY. These are used to “safe store” the request and response messages on QM3.

In this section you will test the effect of the MQEndpoint policy configuration using the IIB Flow Exerciser.

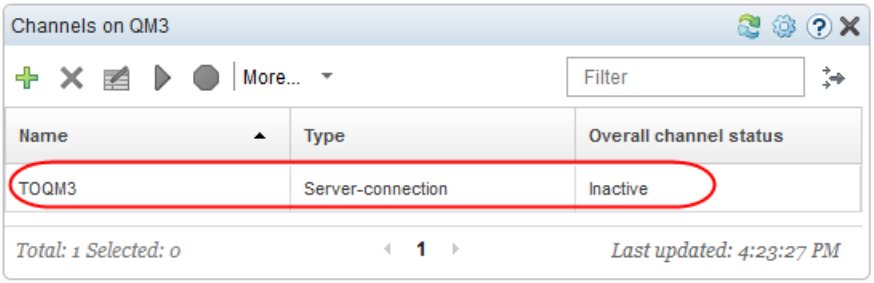
7.1 Start the Flow Exerciser (MQ Provider)

Starting the Flow Exerciser for the provider will deploy the application which is now configured to use an MQEndpoint policy.

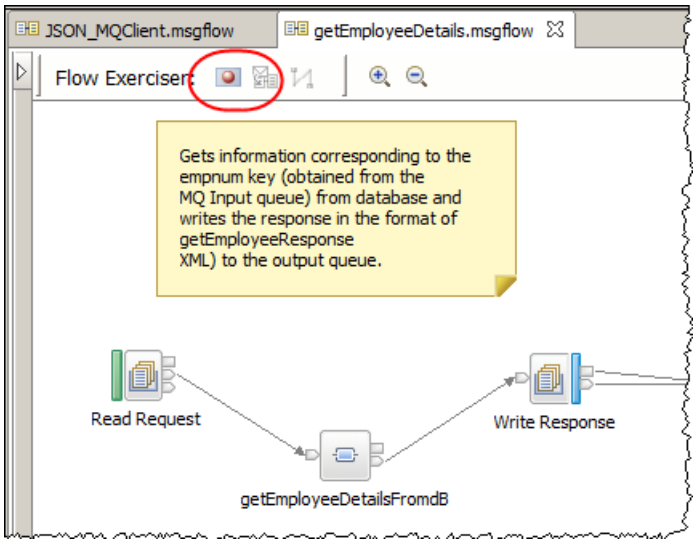
1. Earlier you configured the MQ Appliance Console to have a MQ Object widget that displays the status of the MQ channels on QM3.

   In a browser displaying the IBM MQ Console for the MQ Appliance, check the status of the TOQM3 Channel on QM3. (you may need to log in again if your sessions has timed out: user=**`admin`**; password=**`passw0rd`**)

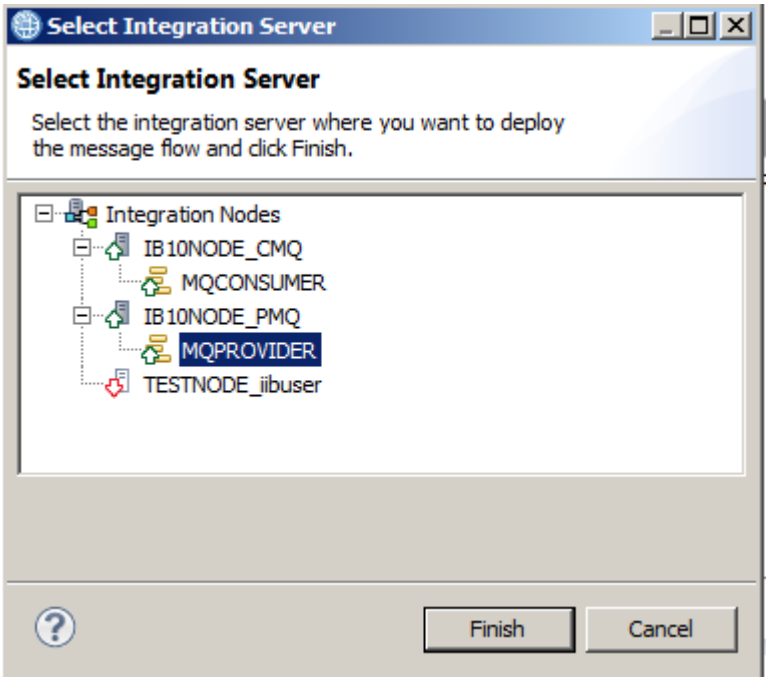
   The status of the channel should be “Inactive”:



2. In the Integration Toolkit, click the record button on the getEmployeeDetails.msgflow:



3. When prompted to select the Integration Server, choose **IB10NODE_PMQ/MQPROVIDER** and click Finish:



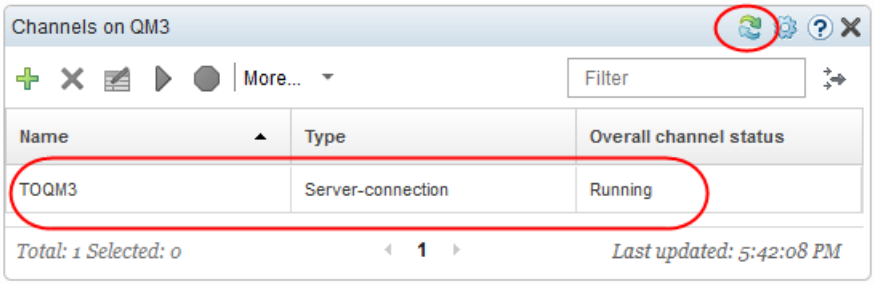
(***Note*** *if you chose a different Integration Server where the MQEndpoint Policy attached to the MQ nodes in a message flow is not defined in the IIB node's Integration Registry, the deployment of the application will fail.*

If this happens you will see a message "BIP7989E: The policy of type 'MQEndpoint' with name 'MQApplianceProvider" can not be found in the Integration Registry.").

Ensure the application deploys successfully.

7.1.1 Verify the Provider application can connect to the MQ Appliance

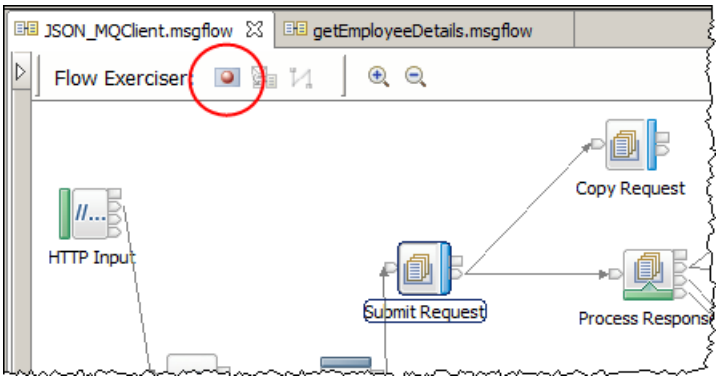
1. Switch to the MQ Appliance Console in the browser and refresh the channels widget to see that the status of the "TOQM3" channel is now displaying as "Running":



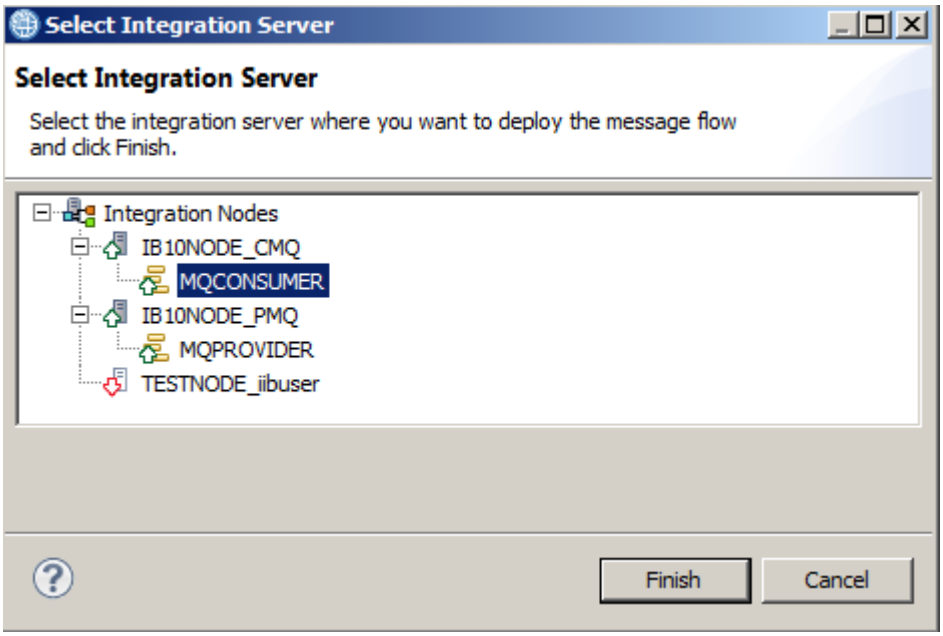
Deploying the Provider MQ application has started an MQ Client connection to the MQ Appliance – the "MQApplianceProvider" MQEndpoint Policy has overridden the MQ Connection properties defined on the MQ nodes so that the application connects to QM3 on the MQ Appliance.

7.2 Test the JSON_MQClient Application

1. Click the record button on the JSON_MQClient.msgflow:



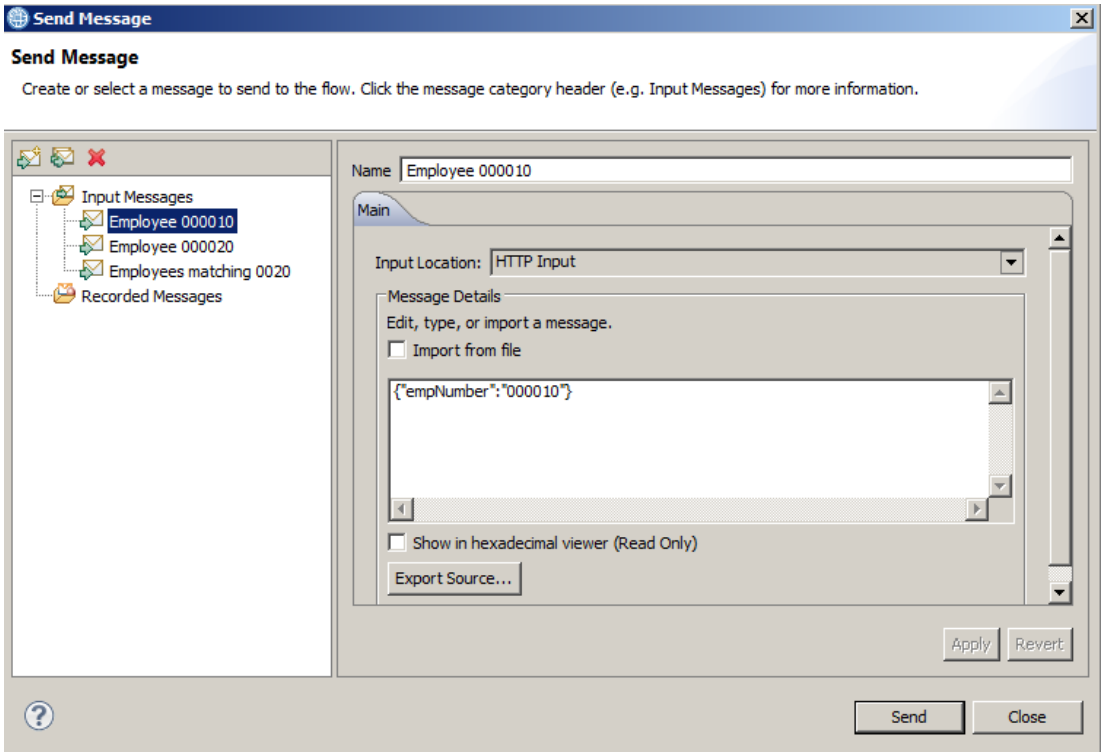
2. When prompted to select the Integration Server, choose IB10NODE_CMQ.MQCONSUMER and click Finish:



3. Dismiss the “Ready to record message” by clicking OK.

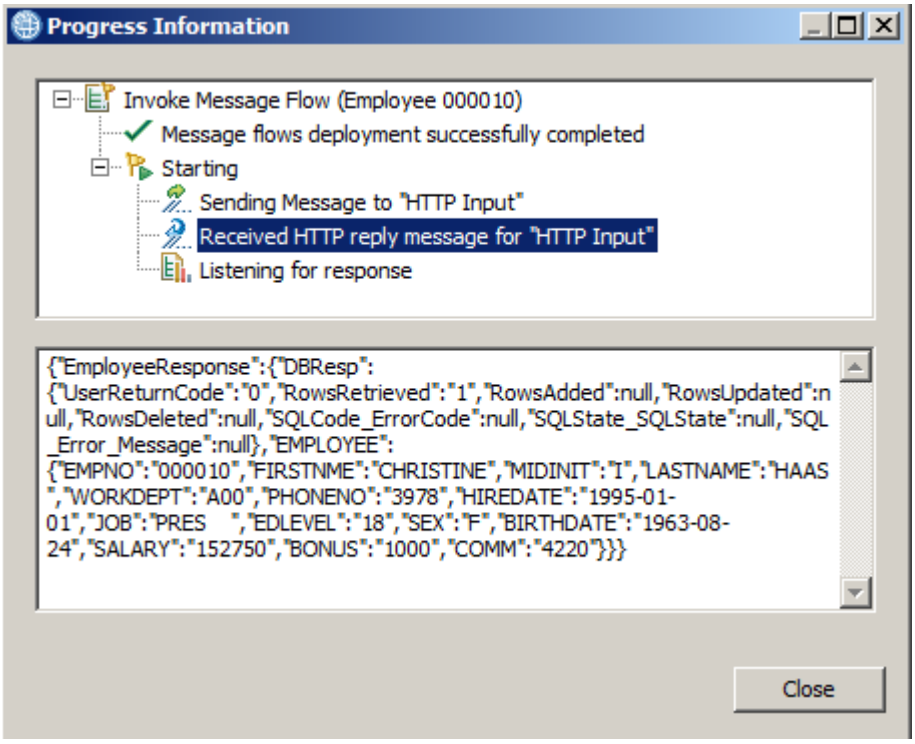
4. Click the Send Message icon to open the dialogue to send a message to the flow.

5. Highlight User 000010 and click Send:



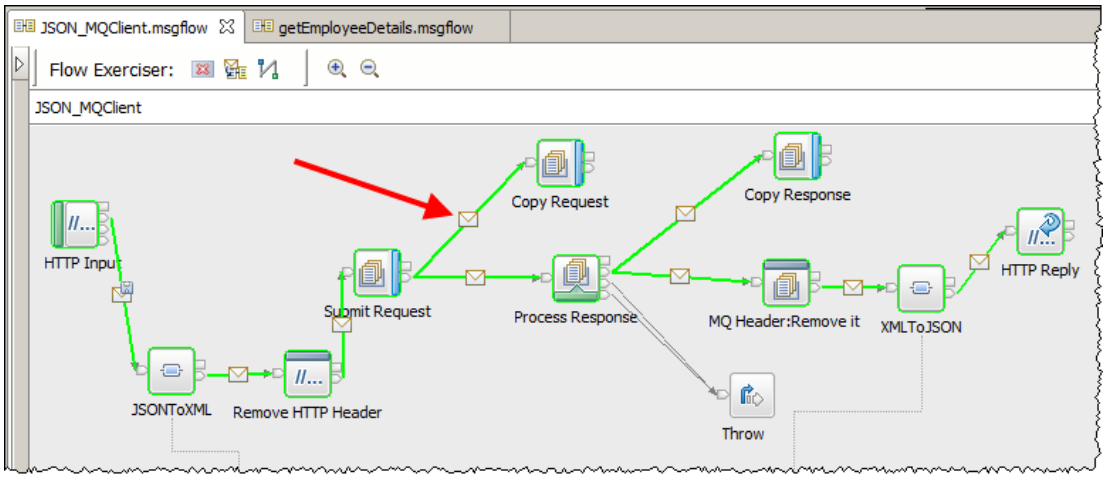
7.3 Verify the two applications have worked correctly

1. In the progress Information window you will see data from an HTTP reply node containing a data from the EMPLOYEE table formatted as a JSON message:

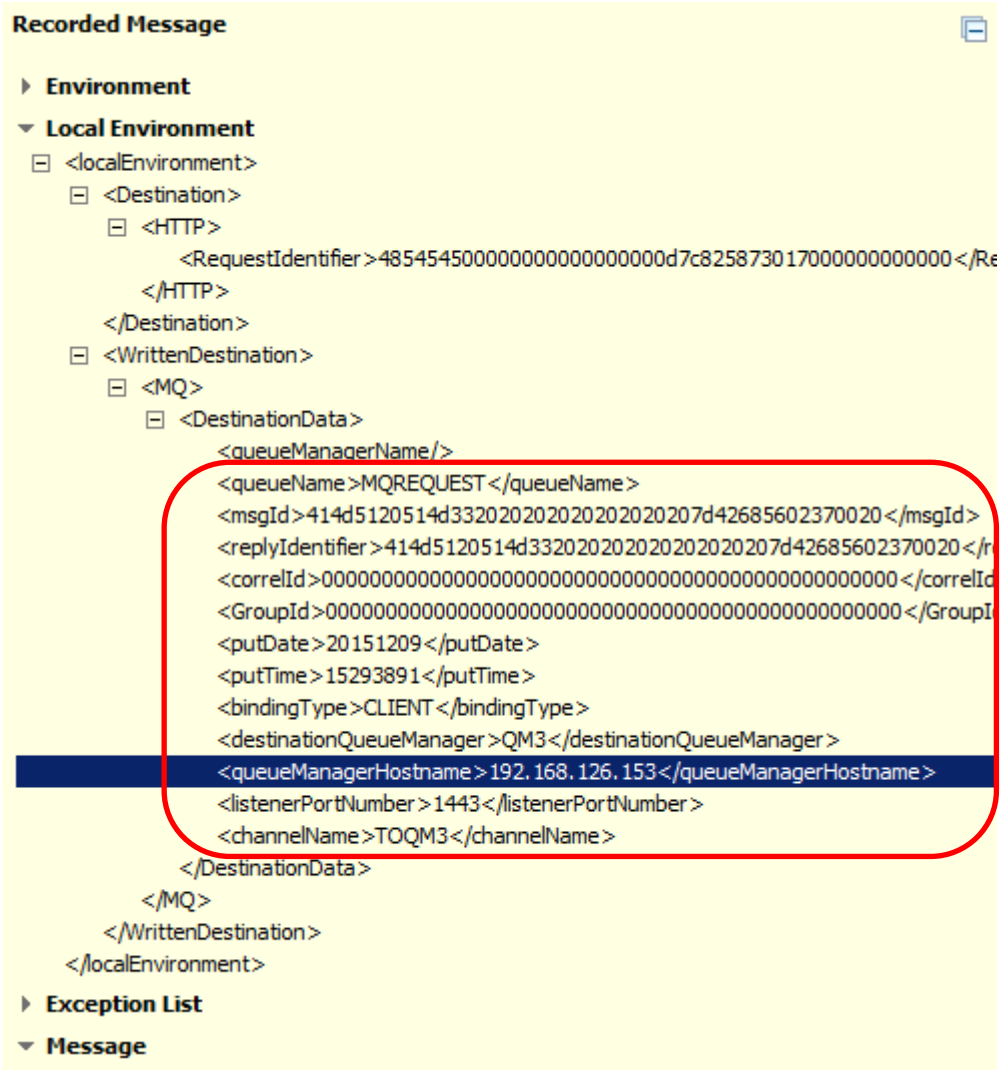


2. Close the Progress Information window to display the path to the message took through the message flow.

3. Click on the envelope icon displayed on the connector between “Submit Request” and “Copy Request” to display the recorded message:



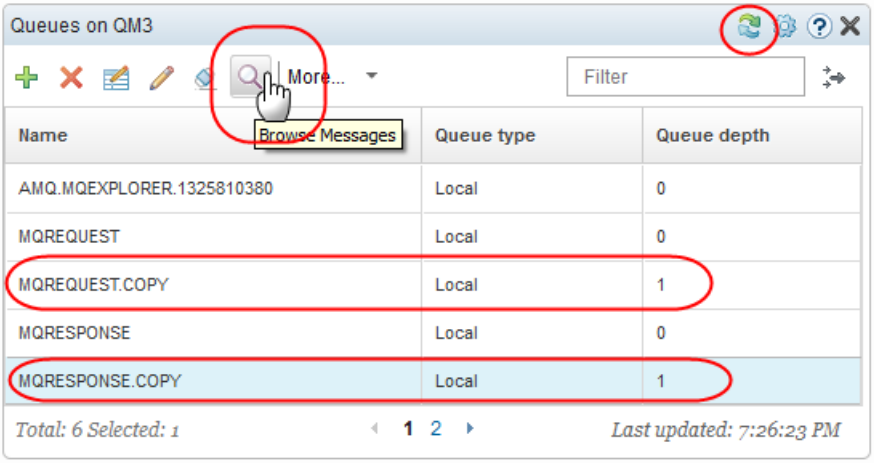
4. In the recorded message display window, expand Local Environment > Written Destination > MQ > Destination Data to display the properties used by the message flow to write the message to the MQREQUEST queue:



Note the properties used are those defined on the MQApplianceConsumer MQEndpoint Policy.

7.4 Verify the messages written to the MQ Appliance

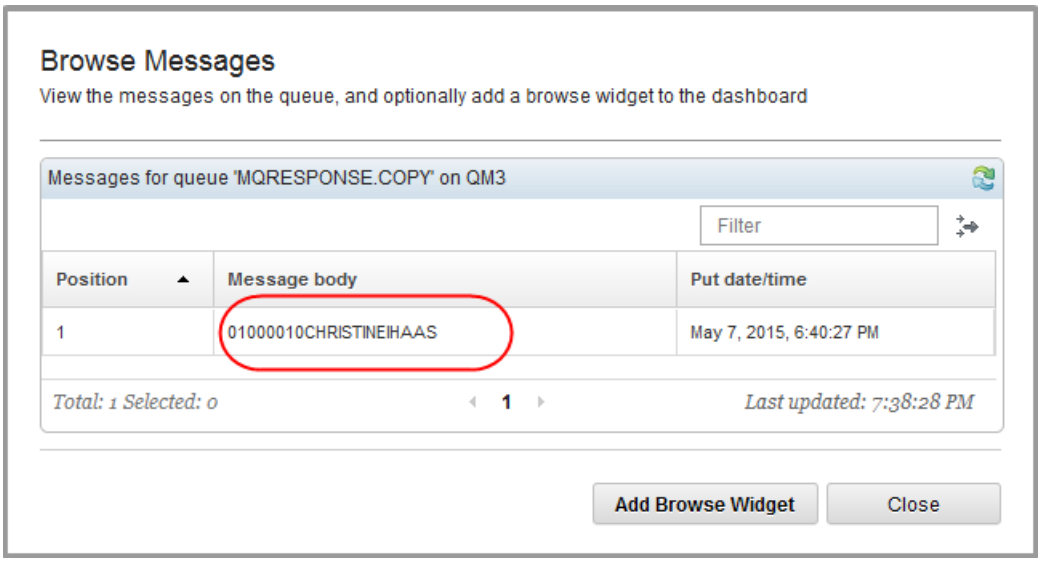
1. Switch to the MQ Appliance Console in the browser and refresh the “Queues on QM3” widget:



Note that the MQREQUEST.COPY and MQRESPONSE.COPY queues now have a queue depth of 1.

Highlight the MQRESPONSE.COPY queue and select “Browse Messages” from the widget header to see the contents

2. The message is displayed in a pop out window:



3. Close the Browse Messages Window.

END OF LAB GUIDE