Integrate IBM MQ with your IBM Business Process Manager processes 2015.10.15 11:52

freeman98 조회 수:481

http://www.ibm.com/developerworks/bpm/library/techarticles/1510_liu-trs/1510_liu.html

Integrate IBM MQ with your IBM Business Process Manager processes

Learn the various approaches to send and receive messages from a business process through IBM® MQ, communicating with back-end systems. Walk through examples with sample code to see the steps that are required to integrate IBM MQ V8.x with IBM Business Process Manager (BPM) V8.5.x. Gain an understanding of fundamental approaches to use IBM MQ to exchange data with other systems from IBM BPM processes.
PDF (1112 KB)

Introduction

IBM BPM V8.5 is a comprehensive business process management platform, providing full visibility and insight to managing business processes. IBM MQ is robust messaging middleware that simplifies and accelerates the integration of diverse applications and business data across multiple platforms. IBM MQ facilitates the assured, secure, and reliable exchange of information between applications and systems. If you need to send and receive data to other systems from within business processes, IBM MQ is usually the natural choice because of its track record of rapid and seamless connectivity.

Try the Workflow service

Create long-running, stateful workflows that orchestrate tasks and services with synchronous or asynchronous event-driven interactions with the Workflow service from Bluemix. Try it for free!
This tutorial describes approaches to send data from a business process to IBM MQ and approaches to receive data from IBM MQ to drive the business process to move forward. The examples step through a simple use case to describe different scenarios. This tutorial assumes that you have a basic knowledge of IBM BPM and IBM MQ. To follow and run the code sample that is provided in the tutorial, you need the following software:

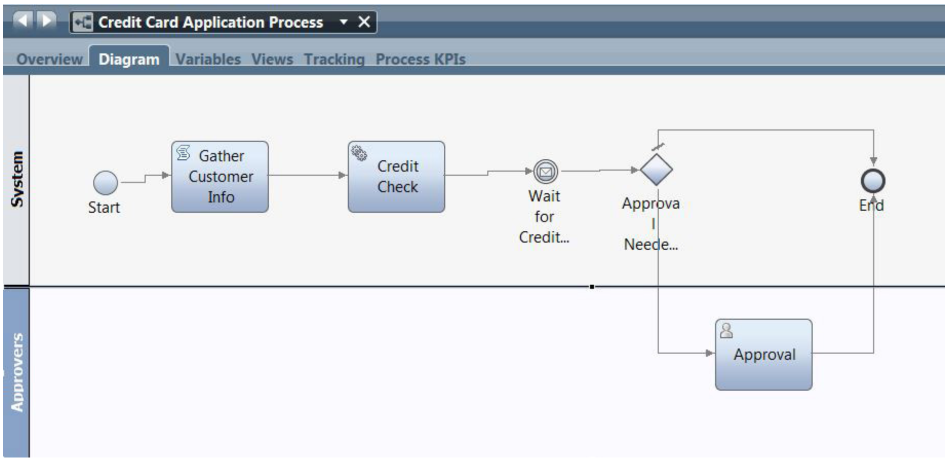
- IBM Business Process Manager Standard or Advanced V8.5.x (This tutorial uses IBM BPM V8.5.6, available at the time of publishing.)
- IBM MQ V8.x
- An eclipse-based J2EE development environment

Back to top

Business process use case overview

This tutorial uses a simple credit card application example business process to demonstrate various ways of integrating IBM BPM with IBM MQ. The first step in the process is to gather a customer's information, such as social security number (SSN) and customer name. Then, the customer SSN is sent through IBM MQ to a credit score service provider, and the provider sends back the credit score to the business process through IBM MQ. If the credit score is below 650, a manager approval is required. Figure 1 shows the example business process definition (BPD) in Process Designer.

Figure 1. Credit Card Application Process, an example process in IBM BPM



Click to see larger image

To simplify the process, a script activity is used to initialize the customer information. The **Credit Check** system service sends a message to IBM MQ with the customer information. Then, a message event is placed in the process to wait for the credit check result that is sent back through IBM MQ. The message event is attached to an undercover agent to wait for a message.

Back to top

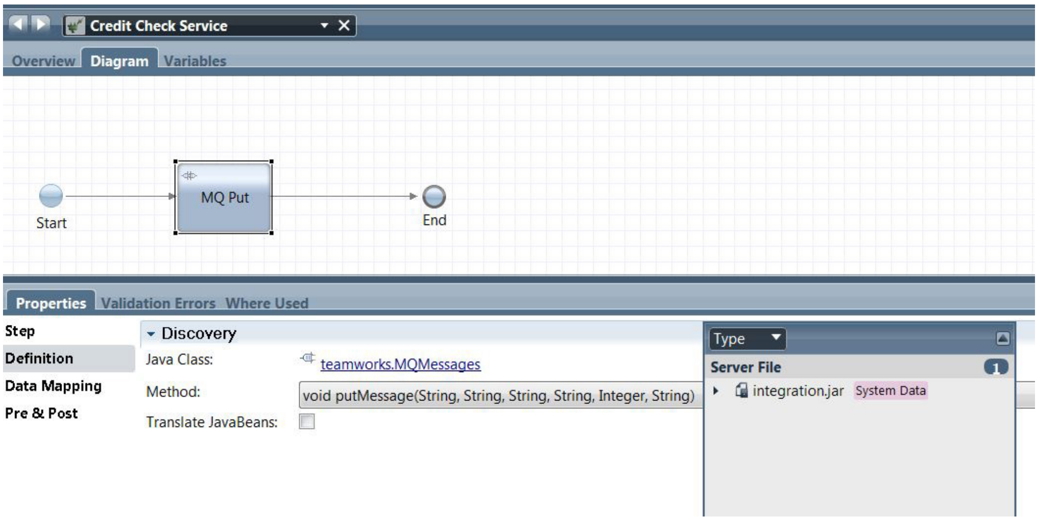
Sending messages to IBM MQ in IBM BPM processes

In a business process, you can send a message to IBM MQ in the following three ways:

1. Use the `teamworks.MQMessages` Java™ service `putMessage` method that is available for immediate use in the `integration.jar` file. This option is the simplest approach, but it provides very limited customization. To allow access, you must add both the user ID that the IBM BPM Java process runs under and the IBM BPM host to the server-connection channel authorization in IBM MQ.
2. If you are using IBM BPM Advanced, the Advanced Integration service (AIS) with the IBM MQ binding is a better option. The IBM MQ Binding offers a higher degree of customization of IBM MQ connectivity. To create an AIS service with the IBM MQ binding, see the WebSphere MQ binding topic in the IBM BPM documentation on IBM Knowledge Center.
3. If you are using IBM BPM Standard, if the `MQMessages` class that is available for immediate use cannot meet your needs, consider a Java service that uses a custom Java Message Service (JMS) client to send messages to IBM MQ. Treat the custom JMS client like any JMS client. The JMS client class needs to be packaged in a .jar file and included in the business process application. IBM MQ resources must be defined in the IBM BPM service that is used by the JMS client.

The sample code that is provided with this tutorial illustrates how to use option 1, using the `MQMessages` class. See Figure 2.

Figure 2. MQMessages class example in the Credit Check Service diagram



Java Class: teamworks.MQMessages

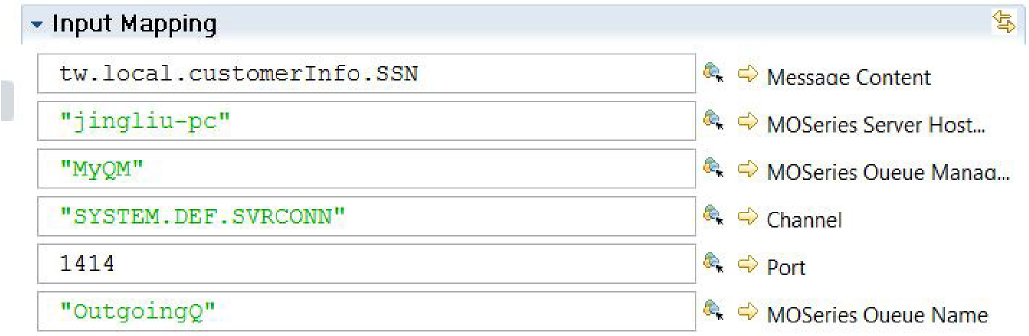
Method: void putMessage(String, String, String, String, Integer, String)

Translate JavaBeans: □

Click to see larger image

The `MQMessages` class is packaged in the `integration.jar` file that is available for immediate use in the System Data toolkit. The `putMessagemethod` is used to PUT a message to IBM MQ. On the Properties tab, click **Data Mapping** to configure the customization that you need to connect to IBM MQ, as shown in Figure 3.

Figure 3. Example of putMessage data mapping



Input Mapping

tw.local.customerInfo.SSN	Message Content
"jingliu-pc"	MQSeries Server Host...
"MyQM"	MQSeries Queue Manag...
"SYSTEM.DEF.SVRCONN"	Channel
1414	Port
"OutgoingQ"	MQSeries Queue Name

Back to top

Receiving messages from IBM MQ in IBM BPM processes

When a business process needs to wait for an IBM MQ message to continue to the next step in the process flow, usually it is implemented with a message event in the BPD that is designed to wait for a message to arrive. The message event is attached to an undercover agent that needs to be triggered when the message is placed in the IBM MQ queue. The IBM BPM runtime environment uses an internal Event Manager JMS queue to receive the message event.

In a business process, you can receive messages from IBM MQ and trigger the undercover agent in the following three ways:

1. You can use an integration service that uses the getMessage method that is available for immediate use in the `MQMessages` class. This method has the same limitation as the putMessage method that is available with limited customization. After the message is retrieved from IBM MQ, you can use an **Invoke UCA** activity to trigger the undercover agent that is attached to the waiting message event.
2. Similarly, if you have IBM BPM Advanced, you can use an AIS that uses the IBM MQ binding with to receive IBM MQ messages. After the message is retrieved from IBM MQ, you can use an **Invoke UCA** activity to trigger the undercover agent that is attached to the waiting message event.
3. You can use a message-driven bean (MDB) that listens to the IBM MQ queue and retrieves the IBM MQ message. Then, use the JMS application programming interface (API) to directly send messages to the internal JMS event manager queue to trigger the undercover agent.

The following examples in this tutorial illustrate options 1 and 3. Option 2 requires IBM BPM Advanced, which is not shown in the examples with this tutorial.

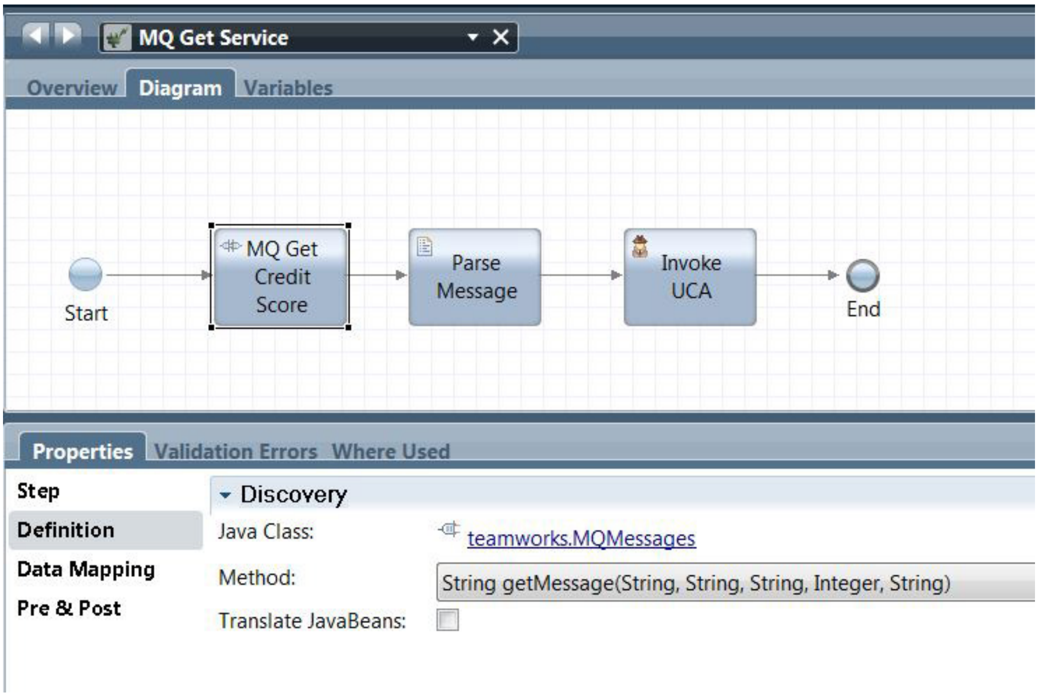
The benefit of option 1 is that you don't need to understand the internal Event Manager JMS queue details and message-format details. Disadvantages are you have limited customization options, and you must manually start or schedule the service (the IBM MQ Get Service show in the examples).

The benefit of using option 3 is the MDB can constantly monitor the IBM MQ queue. The MDB is run whenever the message arrives at the queue. Also, you can use the complete set of the API for MDB and JMS. The disadvantage is that additional effort is needed to send messages to the internal JMS event Manager queue to trigger the undercover agent.

Using the >MQMessages class and the Invoke UCA activity

Figure 4 and Figure 5 show how you can use an integration service that uses the `getMessage` method in the `MQMessages` class. After the message is retrieved from IBM MQ, you use an **Invoke UCA** activity to trigger the undercover agent that is attached to the waiting message event.

Figure 4. The getMessage method in the MQMessage class



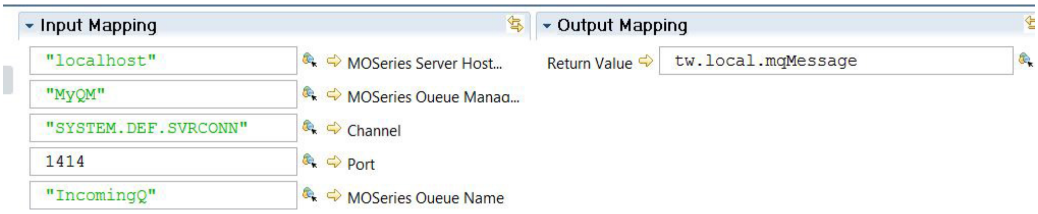
Discovery

Java Class: teamworks.MQMessages

Method: String getMessage(String, String, String, Integer, String)

Translate JavaBeans: □

Figure 5. The getMessage method data mapping



Input Mapping

Value	Input
"localhost"	MQSeries Server Host...
"MyQM"	MQSeries Queue Manag...
"SYSTEM.DEF.SVRCONN"	Channel
1414	Port
"IncomingQ"	MQSeries Queue Name

Output Mapping

Return Value → tw.local.mqMessage

Click to see larger image

As you can see from Figure 4 and Figure 5, after the `MQMessages.getMessage` method is called, the message is saved in the `tw.local.mqMessage` return value and is parsed to retrieve the credit score information. Then, the **Invoke UCA** activity is called to trigger the undercover agent.

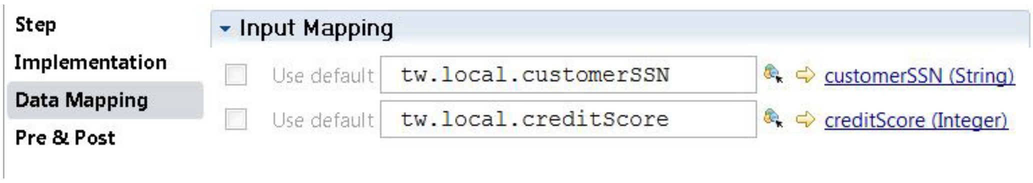
The code sample that is available to download with this tutorial uses a credit card application scenario that has a credit score check provider. With the code sample, you can mimic the credit score service provider by directly putting a message to the reply queue with the comma-separated message format of `customerSSN,creditScore`. Parse the message script as shown in Listing 1.

Listing 1. Example of how to parse an IBM MQ message

```
var splittedString = tw.local.mqMessage.split(",");
tw.local.customerSSN = splittedString[0];
tw.local.creditScore = Number(splittedString[1]);
```

After the message is parsed, the undercover agent is started with the mapping shown in Figure 6.

Figure 6. Undercover agent input

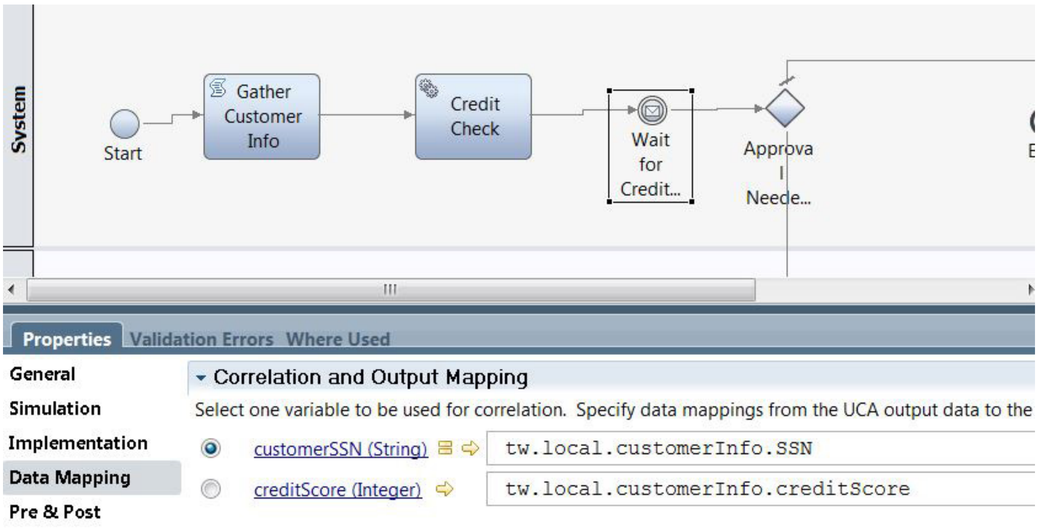


Input Mapping

	Value	Input
□ Use default	tw.local.customerSSN	customerSSN (String)
□ Use default	tw.local.creditScore	creditScore (Integer)

The same undercover agent implementation is attached to the message event in the BPD, and the `customerSSN` string is used as the correlation key, as shown in Figure 7.

Figure 7. Undercover agent data mapping for the message event



Correlation and Output Mapping

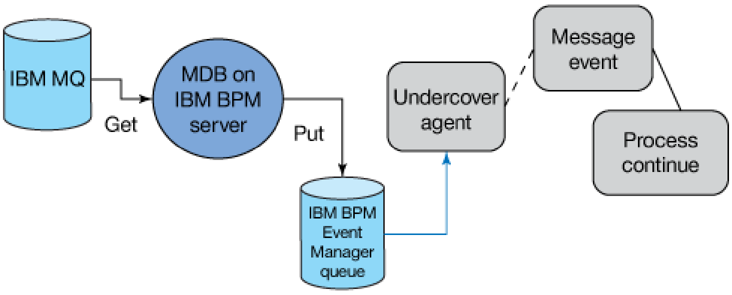
Select one variable to be used for correlation. Specify data mappings from the UCA output data to the

●	customerSSN (String)	tw.local.customerInfo.SSN
○	creditScore (Integer)	tw.local.customerInfo.creditScore

Using an MDB to trigger an undercover agent

Another option to receive messages from IBM MQ is to use an MDB. Figure 8 shows the event flow of using an MDB to trigger an undercover agent.

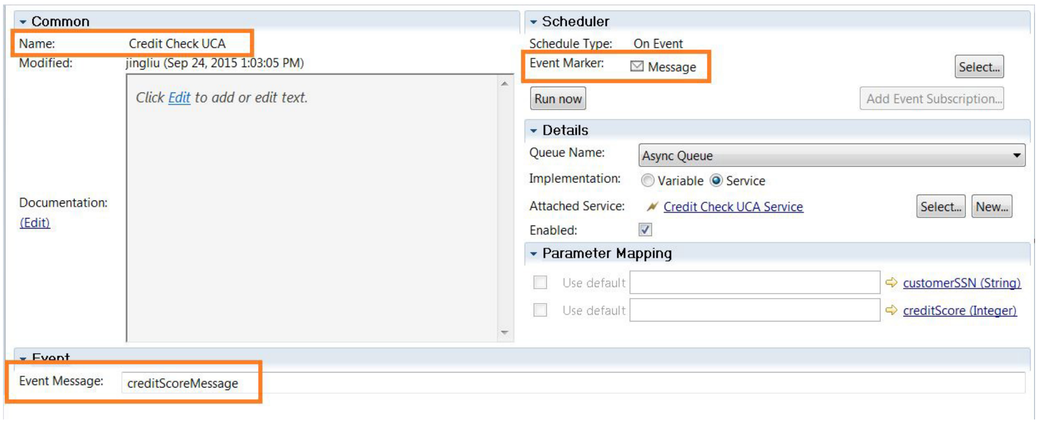
Figure 8. Triggering an undercover agent with an MDB



The MDB runs in a separate J2EE application, which runs on the IBM BPM server and acts as a bridge between IBM MQ and the business process. The bean receives the message from IBM MQ, processes the message, and sends another message through JMS to the internal IBM BPM Event Manager queue. The message triggers the undercover agent for the message event, and moves the process along.

The undercover agent definition needs to specify a unique Event Message name (for example, `creditScoreMessage`, shown in Figure 9).

Figure 9. Undercover agent definition


Common
Name: Credit Check UCA
Modified: jingliu (Sep 24, 2015 1:03:05 PM)
Documentation: (Edit)
Click Edit to add or edit text.

Scheduler
Schedule Type: On Event
Event Marker: Message
Select...
Run now
Add Event Subscription...

Details
Queue Name: Async Queue
Implementation: ○ Variable ● Service
Attached Service: Credit Check UCA Service
Select... New...
Enabled: ☑

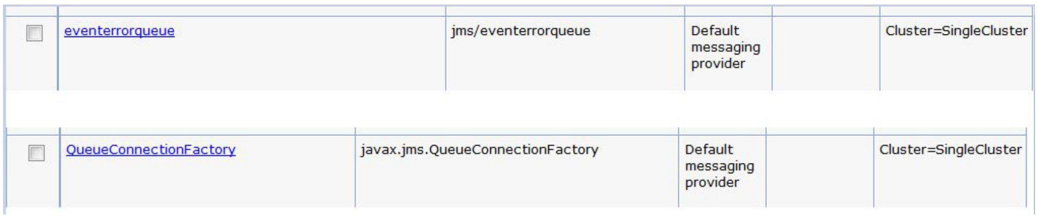
Parameter Mapping
□ Use default customerSSN (String)
□ Use default creditScore (Integer)

Event
Event Message: creditScoreMessage

Click to see larger image

An example of the JMS resources definition for the internal IBM BPM Event Manager queue is shown in the administrative console example in Figure 10. The Event Manager queue is an internal IBM BPM queue.

Figure 10. The JMS resources definition for the IBM BPM Event Manager queue



□	eventerrorqueue	jms/eventerrorqueue	Default messaging provider		Cluster=SingleCluster
□	QueueConnectionFactory	javax.jms.QueueConnectionFactory	Default messaging provider		Cluster=SingleCluster

Click to see larger image

To send a message to the Event Manager, you send a message through an undercover agent to the Event Manager queue. Encapsulate the message data in a predefined XML structure with eventmsg as the root, as shown Listing 2:

Listing 2. Event message format

```
<eventmsg>
  <event processApp="MQAPP" ucaname="Credit Check UCA">creditScoreMessage</event>
  <parameters>
        <parameter>
            <key>customerSSN</key>
            <value>123456789</value>
     </parameter>
     <parameter>
            <key>creditScore</key>
            <value>620</value>
    </parameter>
 </parameters>
</eventmsg>
```

For more information about the message structure, see Posting a message to IBM Business Process Manager Event Manager in the IBM BPM documentation on IBM Knowledge Center.

Listing 3 shows the example MDB code that reads from IBM MQ, processes the message, and then sends it to the Event Manager queue.

Listing 3. Example MDB code

Click to see code listing

Back to top

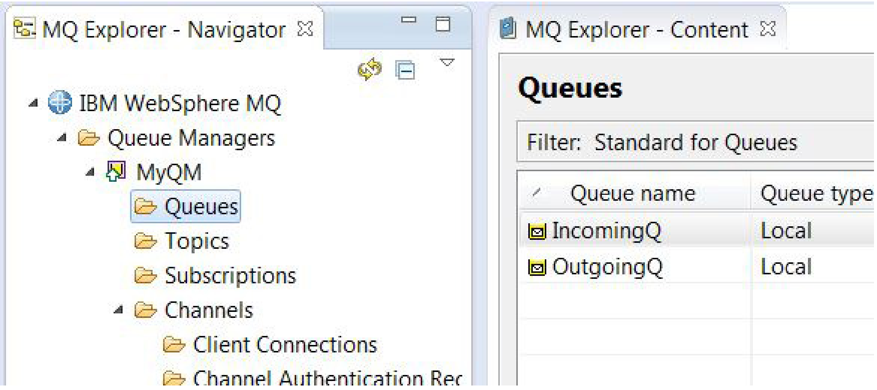
Running the sample code

The sample code contains a business process .twx file that you can import into Process Designer and an .ear file with MDB source code that you can install on the application server for IBM BPM. From the Download section of this tutorial, download the code_sample.zip file and extract the files.

Configuring IBM MQ

The sample code uses the MyQM as the IBM MQ queue manager name, OutgoingQ as the queue name to send the message to IBM MQ, and IncomingQ as the queue to receive the message from IBM MQ. These IBM MQ resources must be created before you run the test. If you named anything differently, update the configuration in the business process and in the MDB settings. Use Figure 11 as an example.

Figure 11. IBM MQ configuration

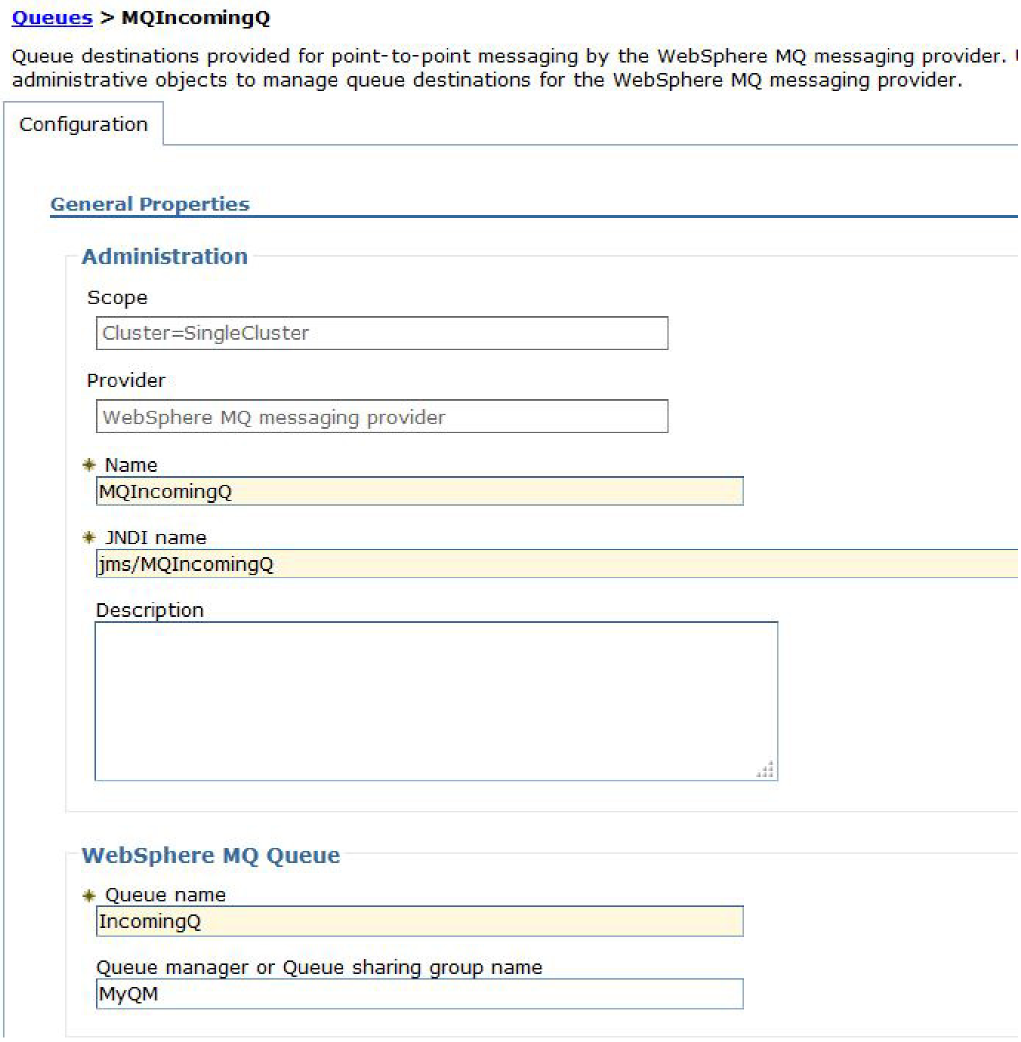


Queue name	Queue type
IncomingQ	Local
OutgoingQ	Local

Installing the MDB application

The J2EE application contains the MDB in the .ear file in the code_sample.zip file that available in the Download section. Before you install the file, you must create the IBM MQ JMS queue resource, pointing to the IncomingQ queue that was defined in IBM MQ and the MDB activation specification for the application server, as shown in Figure 12.

Figure 12. IBM BPM server configuration for the IBM MQ JMS queue resource



Queues > MQIncomingQ

Queue destinations provided for point-to-point messaging by the WebSphere MQ messaging provider. administrative objects to manage queue destinations for the WebSphere MQ messaging provider.

Scope: Cluster=SingleCluster

Provider: WebSphere MQ messaging provider

* Name: MQIncomingQ

* JNDI name: jms/MQIncomingQ

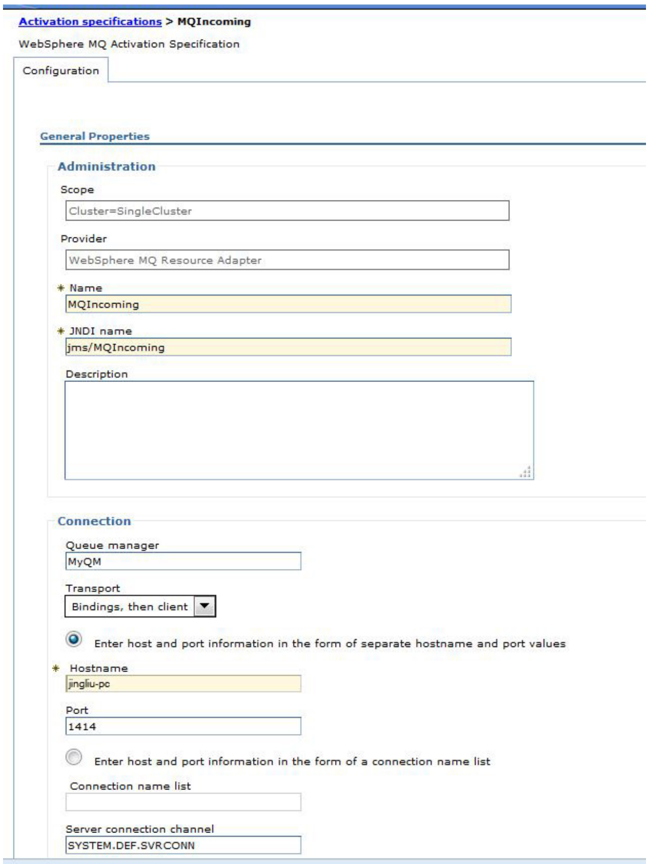
Description:

* Queue name: IncomingQ

Queue manager or Queue sharing group name: MyQM

Figure 13 shows the MDB activation specification configuration for installing the sample code with MDB.

Figure 13. MDB activation specification



Activation specifications > MQIncoming

WebSphere MQ Activation Specification

Configuration

General Properties

Administration

Scope: Cluster=SingleCluster

Provider: WebSphere MQ Resource Adapter

* Name: MQIncoming

* JNDI name: jms/MQIncoming

Description:

Connection

Queue manager: MyQM

Transport: Bindings, then client

● Enter host and port information in the form of separate hostname and port values

* Hostname: jingliu-pc

Port: 1414

○ Enter host and port information in the form of a connection name list

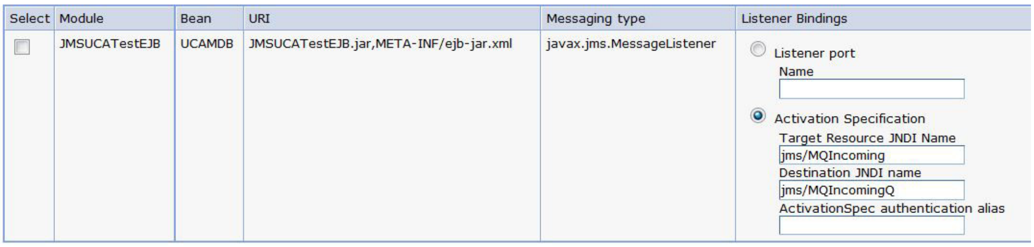
Connection name list:

Server connection channel: SYSTEM.DEF.SVRCONN

Click to see larger image

Then, install the `.ear` file by using the setting for the MDB that is shown in Figure 14.

Figure 14. MDB listener bindings



Select	Module	Bean	URI	Messaging type	Listener Bindings
□	JMSUCATestEJB	UCAMDB	JMSUCATestEJB.jar,META-INF/ejb-jar.xml	javax.jms.MessageListener	○ Listener port Name ● Activation Specification Target Resource JNDI Name: jms/MQIncoming Destination JNDI name: jms/MQIncomingQ ActivationSpec authentication alias:

Click to see larger image

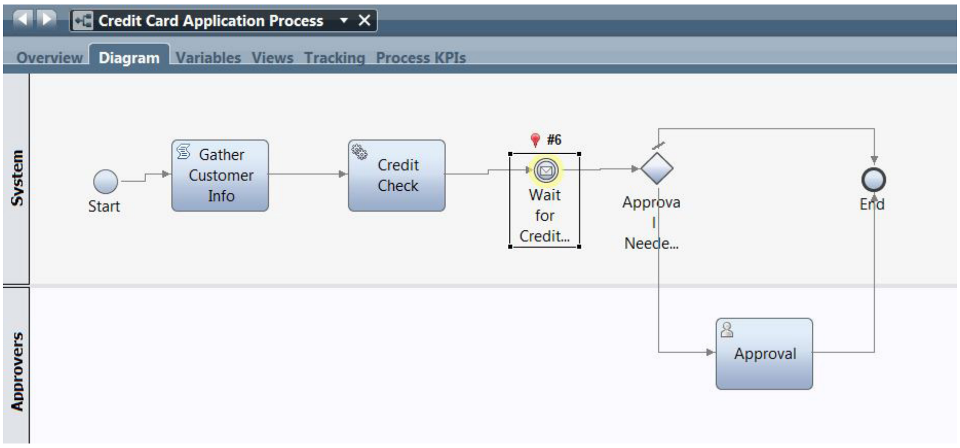
The enterprise archive (EAR) application is installed, but do not start it yet.

Testing how a message is sent to IBM MQ

In Process Designer, import the `.twx` file and click **Start** to start an instance of the **Credit Card Application Process** process. In the inspector, you can see that the message is sent to IBM MQ, and the token is waiting at the message event for the incoming IBM MQ message.

Figure 15 shows the point in the business process that waits for the IBM MQ message to arrive.

Figure 15. Process waiting at the message event



Click to see larger image

In the IBM MQ Explorer, check the `OutgoingQ` queue to view the sent message.

Testing how a message is received from IBM MQ

Now you can test how IBM MQ messages are received, for both options (if you use the `MQMessages` class and if you use the MDB).

Testing the MQMessages Java service option

First, put a message in the `IncomingQ` queue with IBM MQ Explorer to mimic a Credit Score service provider that places a message in the queue. Remember, a comma-separated message format is used, so put a `123456789,620` message in the queue. In this example, `123456789` is the customer SSN and `620` is the credit score. The customer SSN is set in the `Gather Customer Info` script in the beginning of the process. You can update the script to provide a different SSN, but then, make sure to use a different message to IBM MQ when testing.
Next, run the **IBM MQ Get Service** service from Process Designer. You can see the token for the process instance moved to the Approval human service because the business rule determines that if the credit score is less than 650, a manager approval is needed.

Testing the MDB option

You need to start a new BPD instance for the testing. First, update the `Gather Customer Info` script to provide a different `customerSSN`, for example, `987654321`. Then, start the new BPD instance. The process instance waits at the **Wait for Credit Score** message event.
Next, put a message in the `IncomingQ` queue as `98765432,700`, and the message waits to get picked up the MDB. In the administrative console, start the **JMSUCATest** application that you installed earlier. Back in Process Designer, you see that the process instance moved through the flow to the end node. Because the credit score is 700, the manager approval step is not needed.
Back to top

Conclusion

You learned advantages and disadvantages of ways to send and receive messages from a business process to other systems through IBM MQ. After an explanation of three ways to send messages to IBM MQ, the next section stepped through an example of sending messages to IBM MQ with the `MQMessages` class. And after an explanation of three ways to receive messages from IBM MQ, the tutorial demonstrated two examples: using the `MQMessages` class and the **Invoke UCA** activity, and using an MDB to trigger an undercover agent.
The sample code that you can download with this tutorial gives you more opportunities to test sending and receiving messages through IBM MQ from a business process.
You now have the knowledge to determine which option is best for your applications as you plan how to integrate IBM BPM with IBM MQ.

Acknowledgements

The author would like to thank Madhu Channapatna for providing a valuable technical review of this tutorial.
Back to top

Download

Description	Name	Size
Code sample	code_sample.zip	1459KB

Resources

- developerWorks Business process management zone
- Using WebSphere ESB with WebSphere MQ to support publish/subscribe messaging
- IBM Business Process Manager documentation on IBM Knowledge Center
- IBM Business Process Manager Developer Center

이 게시물을

목록

엮인글 0 http://www.websphere.pe.kr/xe/new_tech/51550/c01/trackback

댓글 0

형식 ▼ 글꼴 ▼ 크기 ▼ 줄간격 ▼
확장 컴포넌트 HTML 편집기

□ 입력창 크기 자동조절

선택 삭제 본문 삽입

문서 첨부 제한 : 0Byte/ 10.00MB
파일 크기 제한 : 10.00MB (허용 확장자 : *.*)

글쓴이 비밀번호 홈페이지

댓글 등록

목록

번호	제목	글쓴이	날짜	조회 수
27	JSON 데이터를 가지고 편리하게 JSON 스키마를 만들고 테스트해볼 수 있는 사이트	freeman98	2016.06.20	932
26	Introducing the API Connect Getting Started Video Series	freeman98	2016.06.07	131
25	[APIC]Announcing IBM API Connect: A complete API lifecycle offering	freeman98	2016.04.28	496
24	[BPM]IBM Business Process Manager operation overview, Part 1: Topology, security, basic administration, and monitoring	admin	2016.03.16	291
23	[BPM]Manage and recover IBM Business Process Manager processes efficiently with the REST API	freeman98	2016.02.16	276
22	API 경제 with IBM API Management	freeman98	2016.01.25	253
21	IBM API Management 를 1년 무상으로 사용할 수 있는 no-charge trial offering 발표	freeman98	2015.11.20	406
20	API Use Cases for Every (?) Industry	freeman98	2015.11.04	408
19	클라우드 환경에서의 API 경제 활성화 방안	freeman98	2015.10.27	197
»	Integrate IBM MQ with your IBM Business Process Manager processes	freeman98	2015.10.15	481
17	IBM BPM 에서 API 를 이용하여 내부의 token 을 활용 task 간에 앞이나 뒤로 jump 할 수 있는 기능	freeman98	2015.10.08	289
16	APIM 4020 ‘First Look’ – API Editor & Swagger 2.0	freeman98	2015.08.25	303
15	APIM 4020 ‘First Look’ – Gateway Enforcement and Security	freeman98	2015.08.25	346
14	Configurations should be checked for connection between IBM BPM Process Center and Process Server (For IBM BPM V8.5+)	freeman98	2015.08.07	277
13	SOA Summit에서 SOA vs. API 관리의 차별점에 대해 설명한 기술 웹캐스트	freeman98	2015.08.05	281
12	Developing a high performance API for IBM Cognos integration using WebSphere DynaCache	freeman98	2015.07.14	468
11	Knowledge Collection: Performance documents for the IBM Business Process Manager products	freeman98	2015.06.23	233
10	BPMN2.0 WS BPEL관련 교육자료 [1]	euimok	2015.04.28	482
9	Event Manager in IBM BPM Standard - the missing manual	freeman98	2015.04.14	230
8	IBM Business Process Manager: Connecting to Process Center from Process Designer1 person likes this [1]	freeman98	2015.04.13	355

« 첫 페이지 | 1 | 2 | 3 | 4 | 끝 페이지 »

쓰기

Powered by XE.