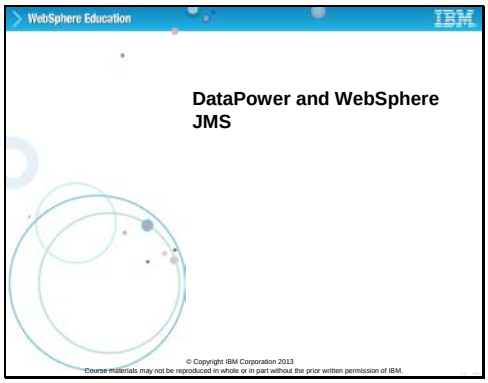
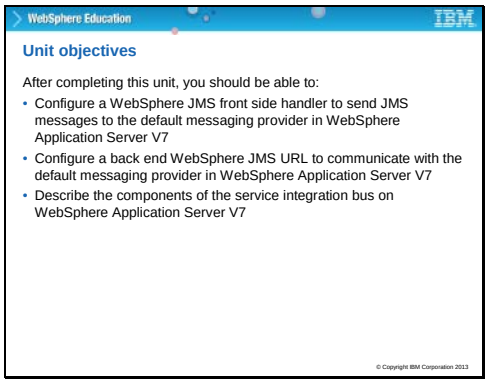
Slide 1

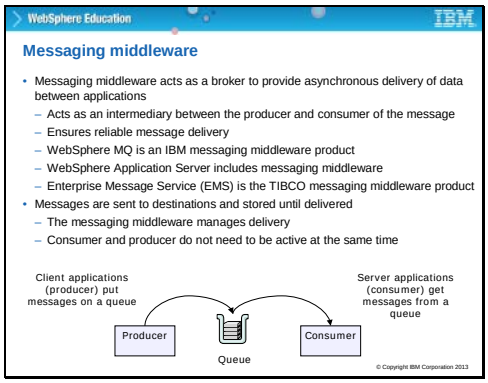

WebSphere Education
IBM
DataPower and WebSphere JMS
© Copyright IBM Corporation 2013
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Slide 2



This unit introduces the Java Message Service, also known as JMS, and DataPowers integration to JMS. DataPower can communicate with JMS on the front end and back-end. This presentation also covers how to configure a DataPower appliance for JMS communication.

Slide 3



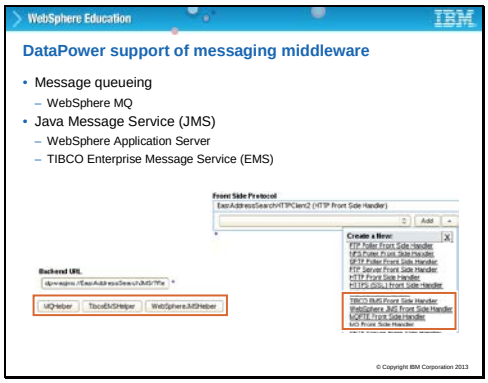
Messaging Middleware:

There are many different types of middleware: transaction processing monitors, remote procedure calls, and object request brokers can all be considered middleware. Messaging middleware stores, routes, and manages messages for delivery. Imagine a courier service that lets you drop off a package for delivery. That company ensures that your package is delivered by a certain time. If the other party is not available, it holds the package for later delivery or pickup. Messages are routed through the services of messaging middleware in this manner.

The application that produces the messages is known as the producer, and the application that receives the messages is known as the consumer.

Messages can remain on the queue for some length of time.

Slide 4



DataPower support of messaging middleware:

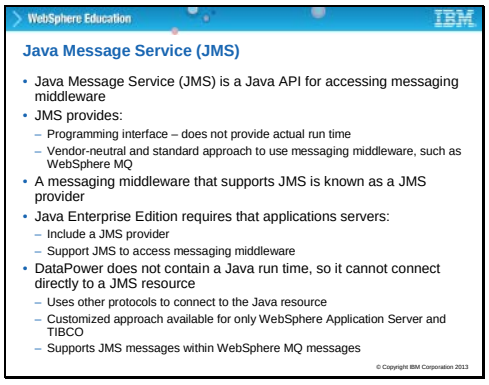
DataPower supports The WebSphere MQ model by using WebSphere MQ.

It supports the JMS model by using two of the implementations: WebSphere Application Server and TIBCO EMS.

No other implementations of JMS are supported currently.

The support is at both the front side, and the back-end. It is also supported within style sheets. The url-open extension element can call the WebSphere MQ and JMS implementations from within a style sheet.

Slide 5



The Java Message Service (JMS):

Before JMS, you needed to use a proprietary client API to access messaging middleware.

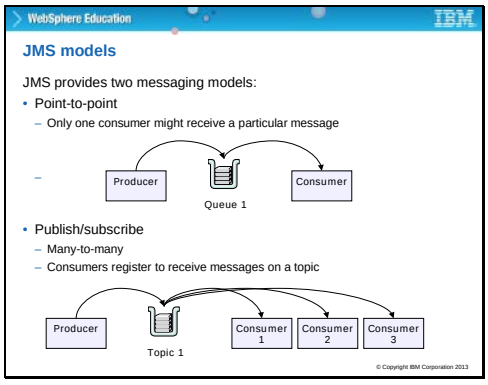
WebSphere Application Server V7 is a Java EE V5-compliant server that also provides a pure Java JMS V1.1 provider.

In a way that is similar too many Java APIs, the programming interfaces for JMS are abstract. Java product vendors such as IBM, BEA, and Sun Microsystems implement these APIs in their products. This scheme promotes the Java "write once, run anywhere" characteristic. Therefore, the slide states that JMS provides a "vendor-neutral" method to allow applications to use message-oriented middleware from any vendor.

WebSphere MQ is a stand-alone product that is provided by IBM. WebSphere MQ is a JMS V1.1-compliant messaging provider.

WebSphere Application Server V8 is a Java EE V6-compliant server that also provides a pure Java JMS V1.1 provider.

Slide 6



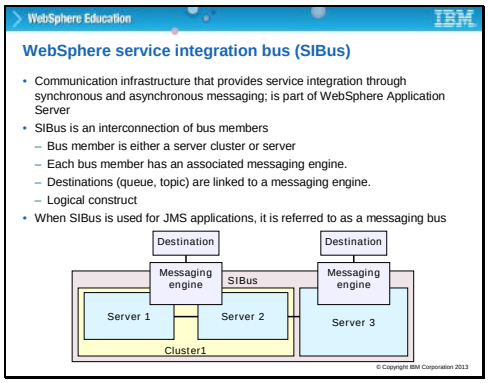
JMS models:

In the point-to-point messaging domain, the message consumer and producer do not must be available at the same time. If either one is unavailable, the JMS queue keeps the message.

The publish/subscribe model defines one or more message consumers that subscribe to a particular topic. The JMS sender transmits a message to a particular topic.

Use **durable subscriptions** to deliver messages even if the message consumer is temporarily unavailable. This option stores the message in the topic, ready to be retrieved when the consumer becomes available.

Slide 7



WebSphere service integration bus (the SIBus):

Any application can exchange messages with any other application by using a **destination**.

A message-producing application, that is, a **producer**, can produce messages for a destination regardless of which messaging engine the producer uses to connect to the bus.

A message-consuming application, that is, a **consumer**, can consume messages from a destination (whenever that destination is available) regardless of which messaging engine the consumer uses to connect to the bus.

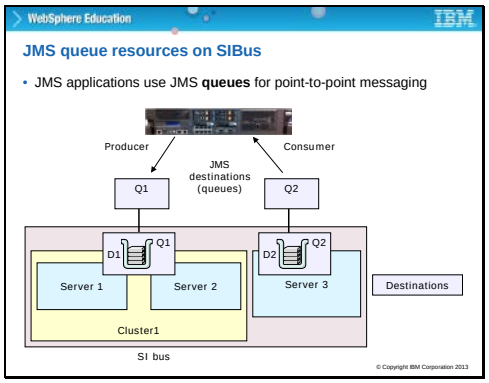
The bus supports the following types of messaging:

Sending messages synchronously. (This type of messaging requires the consuming application to be running and reachable.)

Sending messages asynchronously. (This type of messaging is possible whether the consuming application is running and whether the destination is reachable.) Both point-to-point and publish/subscribe messaging are supported.

Publishing events or other notifications. The bus itself can also generate notification messages.

Slide 8



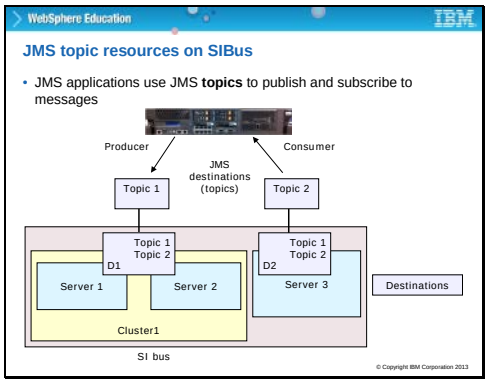
JMS queue resources on SIBus

JMS Destinations are configured on the application server. It encapsulates the name of the actual queue destination on the service integration bus.

D1 and D2 are destinations.

Q1 and Q2 are JMS queues.

Slide 9

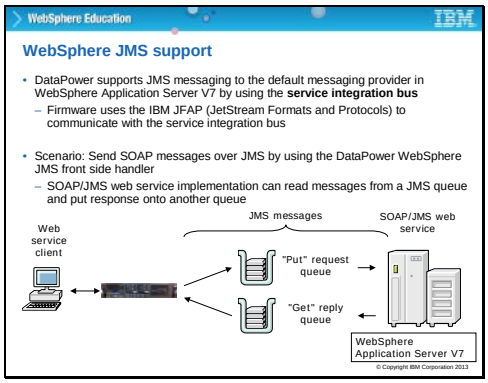


JMS topic resource on SIBus:

An application such as DataPower interacts with a JMS topic, which is a JMS resource that is configured on the default messaging provider.

JMS applications receive messages only from a topic when it is connected to a server.

Slide 10



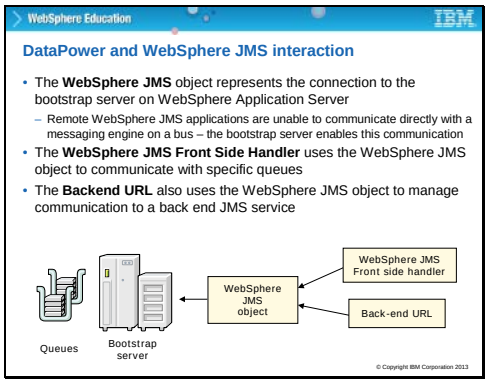
WebSphere JMS support:

You must create and configure both a service integration bus and the JMS resources for a JMS client to invoke a SOAP/JMS web service that is running on WebSphere Application Server V7.

Since the DataPower appliance does not contain a JVM, it cannot exchange messages with the JMS provider in WebSphere Application Server by using JMS. It uses the IBM JFAP (JetStream Formats and Protocols) to communicate with the service integration bus.

The default JMS provider in WebSphere Application Server Version 7 uses JFAP. With JFAP support, a multi-protocol gateway or web service proxy can provide default JMS capabilities both as a client-facing and server-facing messaging service.

Slide 11



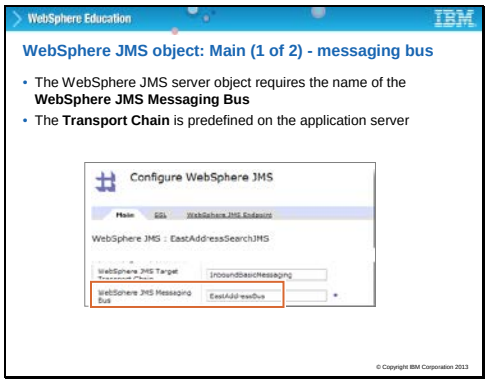
DataPower and WebSphere JMS integration

This slide describes the DataPower objects that are needed to access WebSphere JMS

The default WebSphere JMS port is 7276.

The WebSphere JMS FSH and the back-end URL refer to the WebSphere JMS object, as well as define which queues on that object are being accessed.

Slide 12



WebSphere JMS object: Main 1 of 2

The WebSphere JMS messaging bus name is specified in WebSphere Application Server.

The options for the **WebSphere JMS Target Transport Chain** are:

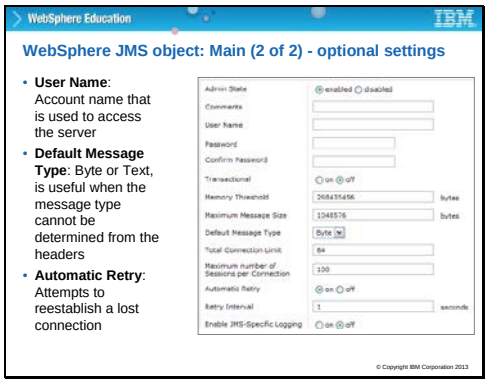
InboundBasicMessaging (the default): specifies the predefined InboundBasicMessaging transport chain (JFAP-TCP/IP)

InboundHTTPMessaging: specifies the predefined InboundHTTPMessaging transport chain (tunnels JFAP that uses HTTP wrappers)

InboundHTTPSMessaging: specifies the predefined InboundHTTPSMessaging transport chain (tunnels JFAP that uses HTTPS wrappers)

InboundSecureMessaging: specifies the predefined InboundSecureMessaging transport chain (JFAP-SSL-TCP/IP)

Slide 13



WebSphere JMS object: Main 2 of 2

Transactional: enables (**on**) or disables (**off**) transaction-based processing, in which messages are acknowledged only after the transaction succeeds. Transaction-based processing is disabled by default.

Memory Threshold: specifies the maximum memory allocation for pending messages. Enter an integer (within the range 1048576 through 1073741824) that specifies the maximum memory (in bytes) that is allocated for pending messages. By default, the maximum memory allocation is set at 268,435,456 bytes.

Maximum Message Size: specifies the maximum message size that is supported by the WebSphere Application Server JMS object. Enter an integer (with the range 0 through 1073741824) that specifies the maximum message size in bytes. By default, the maximum message size is set at 1048576 bytes. You can use the special value **0** to disable the enforcement of maximum message sizes.

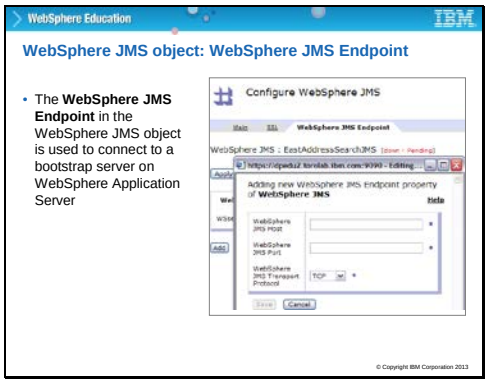
Default Message Type:

Byte: The message payload is accessed as a Java byte array.

Text: The message payload is accessed as a Java string value. This message type is suitable for SOAP and XML input.

Enable JMS-Specific Logging: enables or disables an expanded JMS logging facility. Disabled is the default state.

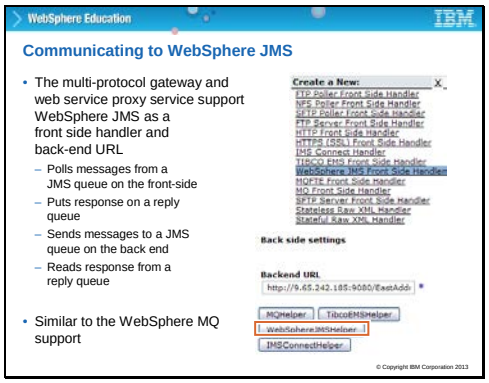
Slide 14



WebSphere JMS object – the JMS endpoint.

WebSphere JMS Transport Protocol: selects the predefined transport chain that the WebSphere bootstrap server supports and is used for information exchange between the WebSphere JMS object and the bootstrap server. The choices are: TCP, SSL, HTTP, and HTTPS.

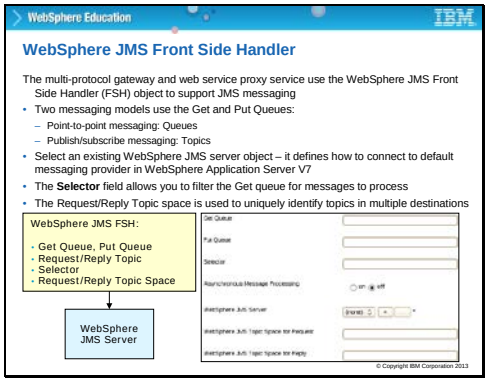
Slide 15



Communication to JMS

A response queue is optional. The WebSphere JMS front side handler also supports one-way messaging. Another way to communicate with a JMS queue is by using an **url-open** in a style sheet.

Slide 16

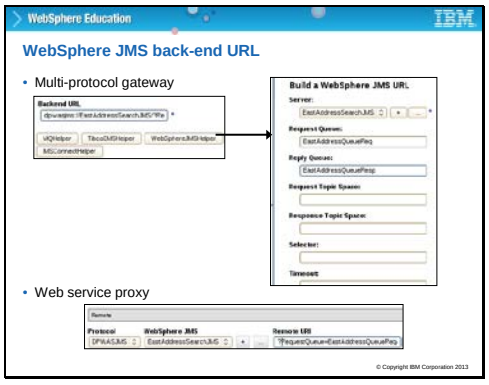


WebSphere JMS Front Side Handler

The actual queues and topics that are defined in the WebSphere JMS FSH are mapped to destinations on the SIBus messaging engine.

The **Selector** field uses an SQL-like syntax for specifying the filter conditions.

Slide 17



WebSphere JMS back-end URL

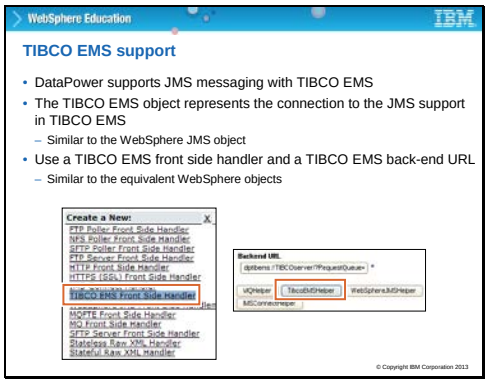
The generated JMS URL in this example is **dpwasjms://EastAddressSearchJMS/?RequestQueue=EastAddressQueueReq&ReplyQueue=EastAddressQueueResp**.

The generated URL is DataPower specific.

The actual queues and topics that are defined in the WebSphere JMS FSH are mapped to destinations on the SIBus messaging engine.

The **Selector** field uses an SQL-like syntax for specifying the filter conditions.

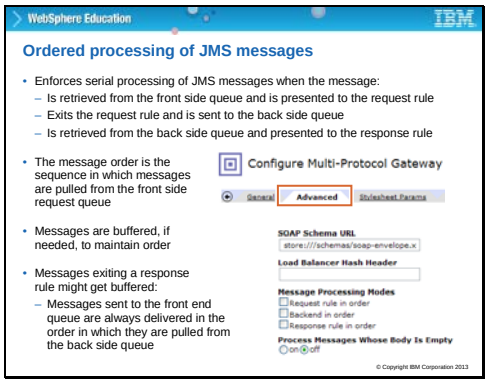
Slide 18



TIBCO EMS Support

JMS support with TIBCO EMS is similar to the support with WebSphere Application Server.

Slide 19



Order processing of JMS messages:

Request rule in order: enforces first-in first-out serial processing of messages for actions in the request rule. The appliance initiates and completes request rule processing for messages in the order in which they were pulled from the front-end request queue. The appliance starts the request rule for the second message in the request queue only after it completes processing the first message. The back-end request queue accepts whatever message arrives first, except in the case where you enforce back-end in order serial processing. In that case, the appliance buffers messages so that it sends messages to the back-end request queue in the same order in which they were pulled from the front-end request queue.

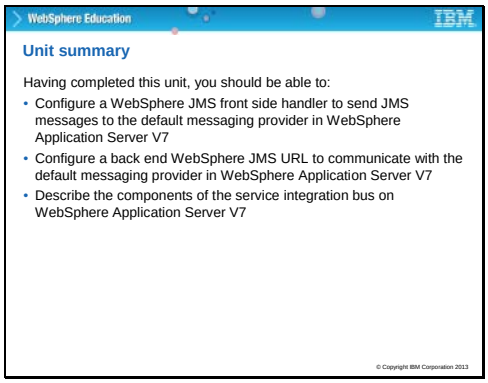
Backend in order: enforces the serial processing of messages that are delivered to the back end request queue. If needed, the appliance buffers messages that complete request rule processing out of order and delivers only messages to the back end request queue in the same order in which they were pulled from the front-end request queue.

Response rule in order: enforces serial processing of messages for actions in the response rule. The appliance initiates and completes response rule processing for messages in the order in which they were pulled from the back end reply queue. The appliance starts the response rule for the second message in the reply queue only after it completes processing the first message. The appliance always buffers messages so that it sends messages to the front-end reply queue in the same order in which they were pulled from the back end reply queue.

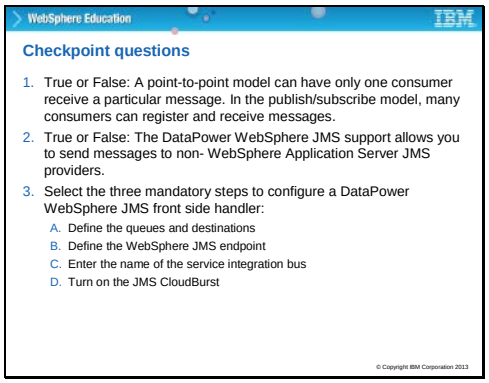
Ordered messaging applies to WebSphere JMS and TIBCO EMS.

The **web service proxy** has the message processing modes under the **Advanced Proxy Settings** tab.

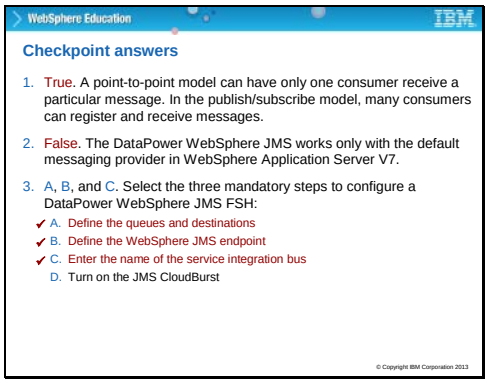
Slide 20


WebSphere Education
IBM
Unit summary
Having completed this unit, you should be able to:
• Configure a WebSphere JMS front side handler to send JMS messages to the default messaging provider in WebSphere Application Server V7
• Configure a back end WebSphere JMS URL to communicate with the default messaging provider in WebSphere Application Server V7
• Describe the components of the service integration bus on WebSphere Application Server V7
© Copyright IBM Corporation 2013

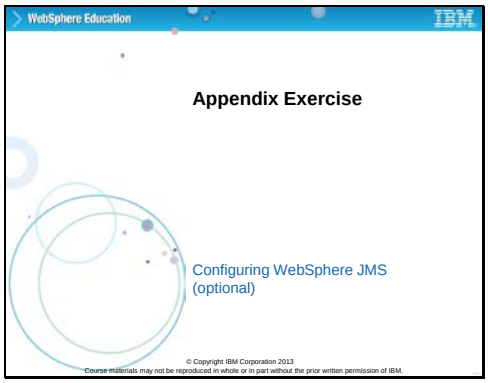
Slide 21


WebSphere Education
IBM
Checkpoint questions
1. True or False: A point-to-point model can have only one consumer receive a particular message. In the publish/subscribe model, many consumers can register and receive messages.
2. True or False: The DataPower WebSphere JMS support allows you to send messages to non- WebSphere Application Server JMS providers.
3. Select the three mandatory steps to configure a DataPower WebSphere JMS front side handler:
A. Define the queues and destinations
B. Define the WebSphere JMS endpoint
C. Enter the name of the service integration bus
D. Turn on the JMS CloudBurst
© Copyright IBM Corporation 2013

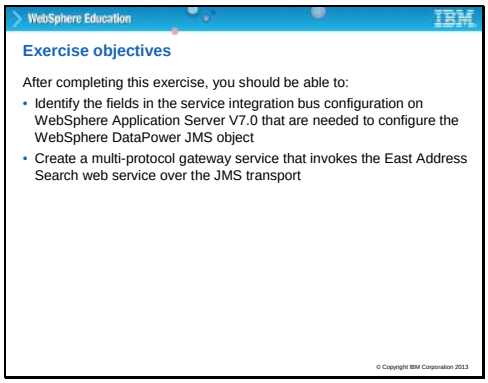
Slide 22


WebSphere Education
IBM
Checkpoint answers
1. True. A point-to-point model can have only one consumer receive a particular message. In the publish/subscribe model, many consumers can register and receive messages.
2. False. The DataPower WebSphere JMS works only with the default messaging provider in WebSphere Application Server V7.
3. A, B, and C. Select the three mandatory steps to configure a DataPower WebSphere JMS FSH:
✓ A. Define the queues and destinations
✓ B. Define the WebSphere JMS endpoint
✓ C. Enter the name of the service integration bus
D. Turn on the JMS CloudBurst
© Copyright IBM Corporation 2013

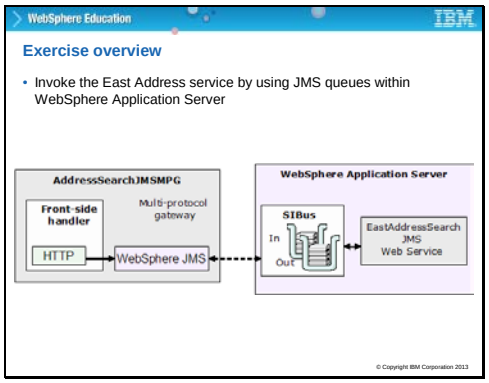
Slide 23


WebSphere Education
IBM
Appendix Exercise
Configuring WebSphere JMS (optional)
© Copyright IBM Corporation 2013
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Slide 24


WebSphere Education
IBM
Exercise objectives
After completing this exercise, you should be able to:
• Identify the fields in the service integration bus configuration on WebSphere Application Server V7.0 that are needed to configure the WebSphere DataPower JMS object
• Create a multi-protocol gateway service that invokes the East Address Search web service over the JMS transport
© Copyright IBM Corporation 2013

Slide 25


WebSphere Education
IBM
Exercise overview
• Invoke the East Address service by using JMS queues within WebSphere Application Server
AddressSearchJMSMPG
Front-side handler
Multi-protocol gateway
HTTP
WebSphere JMS
WebSphere Application Server
SIBus
In
Out
EastAddressSearch JMS Web Service
© Copyright IBM Corporation 2013