IBM Product Connectivity Scenarios and Patterns
Version 1 Release 0

Connecting WebSphere Application Server to WebSphere MQ; getting started on Linux

Note

Before using this information and the product it supports, read the information in “Notices” on page 41.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Chapter 1. Scenario: Connecting WebSphere Application Server to WebSphere MQ; getting started on Linux

By connecting to WebSphere® MQ, Java™ EE applications on WebSphere Application Server can consume and work with messages from WebSphere MQ. Starting with an existing WebSphere MQ installation, this scenario leads you through the key tasks required to install and connect WebSphere Application Server on the same Linux computer.

This scenario was developed using a sample WebSphere MQ installation, and uses sample applications to demonstrate the use of WebSphere Application Server connected to WebSphere MQ. If you want to try the scenario, you can set up a copy of the sample installation as described in the scenario. You can use the sample applications provided with the scenario to verify your progress through each stage.

Notes®

This scenario applies to the use of WebSphere Application Server Version 8 with WebSphere MQ version 7.
This scenario was developed and tested with WebSphere Application Server Version 8 and WebSphere MQ version 7. If you use WebSphere Application Server Version 7, you will notice some differences like using the installation wizard instead of the IBM® Installation Manager described in this scenario.

Optional information to help you learn while implementing the solution
The scenario contains blocks of optional information marked by Why? or What else?. You do not need to read this information to complete the scenario, but might choose to learn more:

Why? Describes *why* you are instructed to do something. For example:

Why am I doing this?

You use the **Scope** property to set the level at which the activation specification is visible. The cell scope is the highest level, giving the activation specification the greatest visibility.

What else?
Describes *what else* you might do, or want to learn about, related to what you are reading in the main window. For example:

What else might I do or be interested in?

You can also create activation specifications at other levels. For example, if you have multiple servers you might create an activation specification for each server, using the server scope, so that you can specify different settings to be used for each server.

Tip: Some “Why?” and “What else?” information provides links that would take you to information resources outside the scenario. To complete a scenario, you do not need to follow such links; they are provided only as optional aids for your learning.

Related information:

Scenario: Connecting WebSphere Application Server to WebSphere MQ; getting started on Windows

Scenarios and Patterns
This scenario in the Scenarios and Patterns information center

Product web page
WebSphere Application Server product web page

Library page
WebSphere Application Server library page

Product web page
WebSphere MQ product web page

Library page
WebSphere MQ library page

Chapter 2. Planning the solution

You can connect WebSphere Application Server to WebSphere MQ in a number of different ways. Review the topics in this section to understand what is covered in this scenario, the reasons why a business might want to follow the scenario, the user roles involved, and an overview of the solution proposed by the scenario.

Assumptions

This scenario makes several assumptions about your system, such as the version of the products that you are using.

This scenario assumes the following points:

- You already have WebSphere MQ Version 7.1 running on your system. However, instructions are provided for installing a sample configuration of WebSphere MQ if you do not already have it.
- You already have WebSphere Application Server Version 8 with Fix Pack 3 running on your host. However, instructions are provided for installing a sample configuration of WebSphere Application Server if you do not already have it.
- You have root access to your host.
- You use both the graphical user interfaces and the command-line equivalents of WebSphere MQ and WebSphere Application Server.

Business overview

A company wants to add a Java EE application on WebSphere Application Server to consume and work with messages from an existing messaging infrastructure provided by WebSphere MQ.

To date, company A has a business solution that uses WebSphere MQ for its messaging infrastructure. Business users interact with a stand-alone client application, for example to register an order, which sends a message into the infrastructure. Such messages are transported by the infrastructure for processing by some separate WebSphere MQ application, possibly in a different business unit. The client application waits for a reply message, to provide the business user with an appropriate response.

Changes to the business model of the company mean that the processing performed on messages is inadequate, yet the processing is not implemented in a standards-based way that can best be developed for the future. The company recognizes that the dynamic business needs of today are driving IT departments to implement standards-based computing. The company wants to use standards-based programming to provide the updated processing that must be performed on messages before they are returned to business users. The company decides that they want to use the Java Message Service (JMS), the Java EE standard for messaging, that provides a standard API for applications to use when performing enterprise messaging with application portability. The company also recognize that the Java EE standard streamlines application development, and by using standards they can create reusable, platform-independent modules.

The company decides to add a Java EE application on WebSphere Application Server to consume and work with messages in the solution. WebSphere

Application Server is a proven, standards-based platform that offers enhanced support for standards, emerging technology and a choice of development frameworks that simplifies programming models.

With WebSphere Application Server, the company appreciates that as their business demands increase, they can take advantage of the latest standards and programming models that WebSphere Application Server supports or scale up their business capability by adopting higher-performance features or editions of WebSphere Application Server.

Related information:

➡ WebSphere Application Server features and benefits web page

User roles and interactions

Roles used throughout these scenarios, with interactions between roles for scenario tasks.

Although roles to develop the solution are listed, this scenario focuses on deploying a solution that connects WebSphere Application Server to WebSphere MQ.

Develop the solution

Develop the software aspects of the solution.

Software Architect

The Software Architect is responsible for dividing the required function between the components that make up a software solution. This person works with the specifications and standards used by existing IT systems, and determines where enhancements or new components must be written by the Developer.

Developer

The Developer is responsible for creating and testing the software components and linking them together. In some cases the Developer might need information from the Administrator, for example if the Developer must use an existing queue name in their software code. After the Developer finishes creating the software components, the Developer gives those components to the Administrator for deployment.

Deploy the solution

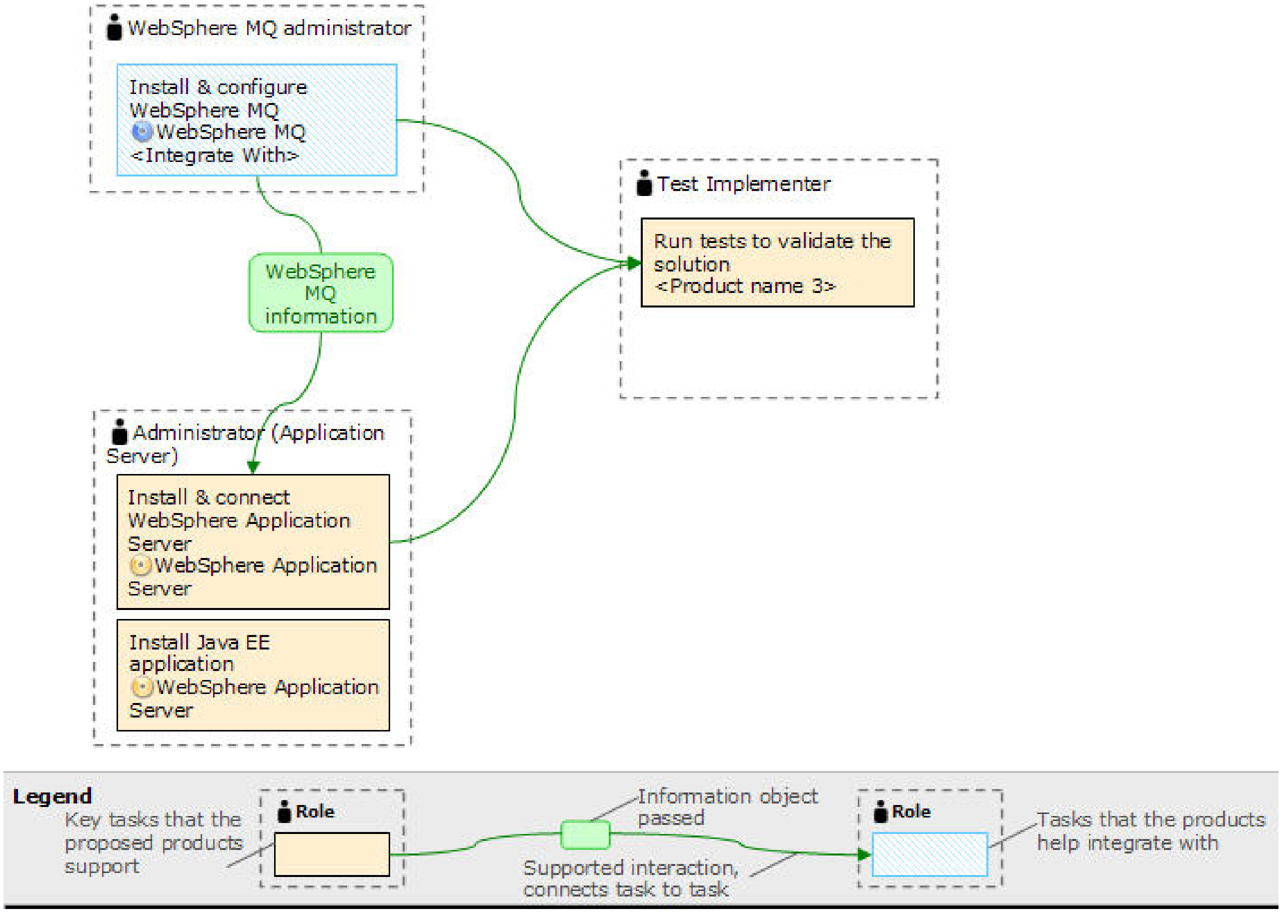
Deploy the solution for production use, by installing, configuring, and testing the components that provide the solution.

Administrator

The Administrator installs and configures the components that support the solution, in this case WebSphere MQ and WebSphere Application Server. The business might have a different Administrator for each product, or one Administrator might perform installation and configuration for all components. If the Administrators are different they might need to exchange information. For example, in this scenario the WebSphere MQ administrator must give the name of the queue and queue manager to the WebSphere Application Server administrator. The Administrator also deploys the software components given to them by the Developer.

Test Implementer

The Test Implementer runs the tests to validate the solution and that the solution is ready for production use. For example, can the solution be started, stopped, backed up, and recovered after a system failure as well as be maintained?



WebSphere MQ administrator
Install & configure WebSphere MQ
WebSphere MQ
<Integrate With>
WebSphere MQ information
Test Implementer
Run tests to validate the solution
<Product name 3>
Administrator (Application Server)
Install & connect WebSphere Application Server
WebSphere Application Server
Install Java EE application
WebSphere Application Server
Legend
Key tasks that the proposed products support
Role
Information object passed
Supported interaction, connects task to task
Role
Tasks that the products help integrate with

Figure 1. User role interactions to deploy the solution

Technical solution

This scenario describes one way of connecting WebSphere Application Server to an existing WebSphere MQ system, by using JMS, JCA and JNDI.

Overview: Initial IT configuration

Company A has an existing IT configuration that uses WebSphere MQ for its messaging infrastructure. This scenario describes adding WebSphere Application Server to that initial messaging infrastructure.

The initial IT configuration includes several components that an administrator configures or uses, as shown in Figure 2 on page 8:

JMS application

A stand-alone application that business users interact with, for example to register an order. The application uses the Java Message Service (JMS) for asynchronous messaging.

Why am I doing this?

- JMS is the Java EE messaging standard that is widely supported. JMS-based applications are therefore portable across many messaging products.
- JMS provides a level of abstraction from the details of the messaging layer, simplifying the application development process.
- JMS provides asynchronous communication, enabling applications to run without having to wait for a reply, unlike tightly coupled systems such as remote procedure call (RPC).
- Applications that use JMS do not directly specify details to access resources. Instead, they look up and use administered JMS objects such as a connection factory and a destination.

What else might I do or be interested in?

For some situations, other messaging standards might be more suitable than JMS. For example IBM Message Service Clients for C, C++ and .NET, also known as XMS, are APIs that provide similar benefits to JMS for non-Java applications. XMS is therefore more suitable if you are using the .NET platform, or you want to integrate existing C++ applications with newer Java EE applications.

The application uses point-to-point messaging to send messages to a queue in the infrastructure and processes reply messages, to provide the business user with an appropriate response.

Why am I doing this?

In this messaging model, an application sends a message to a queue, and another application receives the message from the queue and acknowledges receipt of the message. This model is the simplest form of messaging because it involves only two endpoints. This model is also the most appropriate for the scenario sample application: a single client requests information from a single server.

What else might I do or be interested in?

In the alternative messaging model, publish/subscribe, a publisher publishes a message to a message topic. Subscribers subscribe to the topic to receive messages. The publisher and subscriber do not have any information about each other, and the message is received by zero or more recipients.

Queue manager sampleQM

The WebSphere MQ queue manager that provides the initial messaging infrastructure. It hosts the queue that the JMS application works with.

Q1 [Message queue]

The WebSphere MQ queue that the JMS application sends messages to.

JNDI namespace

A Java Naming and Directory Interface (JNDI) namespace is used to hold JMS administered objects, which applications can use to connect to WebSphere MQ and access destinations to send or receive messages.

Why am I doing this?

JNDI is part of Java EE, and provides a standard way for applications to access various types of naming and directory services, for the retrieval of application components. For example, you can use JNDI to access a naming service on a file system to retrieve the location of a printer object, or to access a directory service on an LDAP server to retrieve a user object which contains ID and password information. JNDI therefore enhances the portability of JMS-based applications, and makes it easier to integrate those applications with each other and into existing systems. For JMS messaging, you use JNDI to store objects that represent the target destination of a message, or the connection factory that creates the connection between your application and its messaging destination.

Any application or process with access to the JNDI namespace can use the same administered objects. The properties of the administered objects can be changed in JNDI, with all the applications or processes able to benefit from those same changes.

Initial context

An initial context defines the root of the JNDI namespace. To use WebSphere MQ Explorer to create and configure administered objects, you first add an initial context that defines the root of the JNDI namespace. Similarly, a JMS application first obtains an initial context, before it can retrieve administered objects from the JNDI namespace.

Connection factory, myCF

A JMS connection factory object defines a set of standard configuration properties for connections. An application uses a connection factory to create a connection to WebSphere MQ.

Destination, myQueue

A JMS destination can be a topic or a queue. In this scenario, the destination is a queue, and identifies the WebSphere MQ queue that applications send messages to, or from which an application receives messages, or both. An application looks up the destination in the JNDI namespace to create a connection to the WebSphere MQ queue.

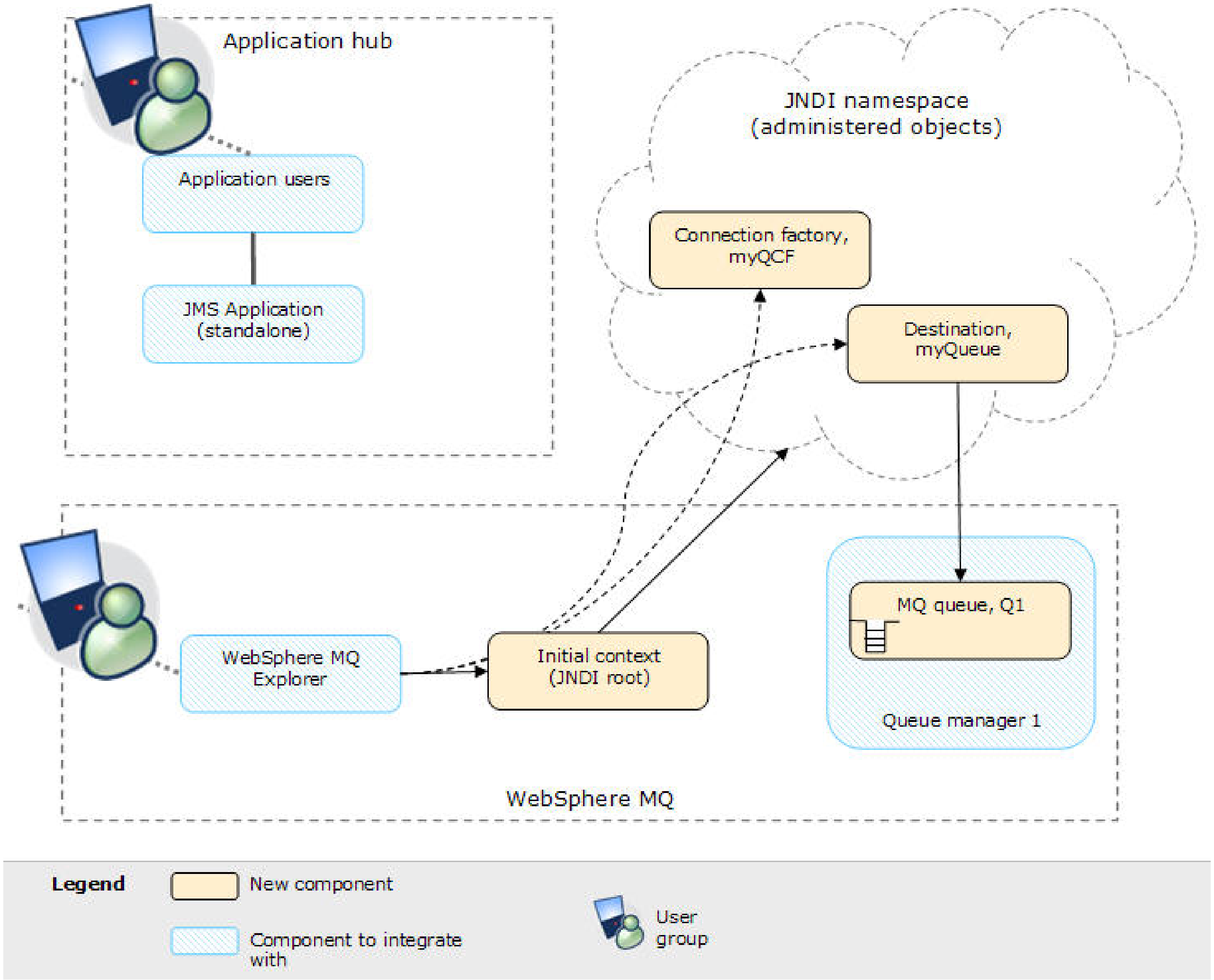

Application hub
Application users
JMS Application (standalone)
JNDI namespace (administered objects)
Connection factory, myQCF
Destination, myQueue
MQ queue, Q1
Queue manager 1
WebSphere MQ Explorer
Initial context (JNDI root)
WebSphere MQ
Legend
New component
Component to integrate with
User group

Figure 2. Initial IT configuration. The initial IT configuration includes an initial context, added for the WebSphere MQ Explorer to connect to the root of the JNDI namespace. The JNDI namespace includes a connection factory, added for the sample JMS application to use to connect to WebSphere MQ, and a destination, added for the sample JMS application to connect to the WebSphere MQ queue. That WebSphere MQ queue has also been added into the initial IT configuration for use by the sample JMS application.

Tip: This scenario was developed by using a sample version of the initial IT configuration described. If you want to try out the scenario, you can either follow the instructions to configure WebSphere Application Server to communicate with your own WebSphere MQ, or set up a copy of the sample IT configuration as described in “Creating a sample initial IT configuration” on page 14.

Overview: The delivered logical topology

The company adds a Java EE application on WebSphere Application Server to consume and work with messages from an existing messaging infrastructure provided by WebSphere MQ.

The delivered IT configuration includes several components that an administrator configures or uses, as shown in Figure 3 on page 11.

WebSphere MQ as a *messaging provider* for WebSphere Application Server
The WebSphere MQ messaging provider in WebSphere Application Server

makes JMS messaging available to WebSphere Application Server applications by using the existing capabilities in the WebSphere MQ environment.

Why am I doing this?

WebSphere Application Server applications can interact with WebSphere MQ destinations to send and receive messages in the same way as any JMS application in the WebSphere MQ environment.

What else might I do or be interested in?

- You can connect a WebSphere MQ network to a service integration bus within WebSphere Application Server, by using WebSphere MQ links. A WebSphere MQ link provides support for sender-receiver channels between the service integration bus and a WebSphere MQ queue manager or queue-sharing group. This option requires more complex configuration in WebSphere Application Server; you must configure a service integration bus and messaging engines.
- You can add a WebSphere MQ server (representing a queue manager or queue-sharing group) as a member of a service integration bus within WebSphere Application Server. A WebSphere MQ server provides a direct client or bindings connection between a service integration bus and queues on a WebSphere MQ queue manager. This option also requires you to configure a service integration bus and messaging engines in WebSphere Application Server.

Java EE application

The application consumes and works with messages on the WebSphere MQ queue, Q1. This application runs on an *application server* of the WebSphere Application Server product that has been installed and connected with WebSphere MQ.

The sample application in this scenario provides a *message-driven bean (MDB)* as an asynchronous message consumer. When a message arrives at the queue, the MDB automatically processes the message without the application having to explicitly poll the queue.

Why am I doing this?

MDBs are activated by the EJB container in WebSphere Application Server on receipt of a message. A typical MDB performs messaging functions, and calls one or more session beans to perform business logic. Because of this separation of function, you can easily change and reuse units of business logic without affecting the messaging function of the application.

Application server

A server program in WebSphere Application Server that provides the execution environment for Java EE application programs.

WebSphere Application Server JNDI namespace

WebSphere Application Server includes a name server which provides access to the following JMS administered objects through the Java Naming and Directory Interface (JNDI). The use of JNDI, the connection factory, and the destination, are the same as described for the initial IT configuration in "Overview: Initial IT configuration" on page 5.

Activation specification, myActSpec

A JMS activation specification can be associated with one or more MDBs and provides the configuration necessary for them to listen for messages arriving at a destination. Activation specifications process inbound messages to the MDB.

Why am I doing this?

Activation specifications are part of the Java EE Connector architecture (JCA) 1.5 standard. JCA 1.5 provides a standard way to integrate JMS providers, such as WebSphere MQ, with Java EE application servers such as WebSphere Application Server.

What else might I do or be interested in?

Use of listener ports is an older configuration method for MDBs to listen for messages arriving at a destination. The use of listener ports is stabilized in WebSphere Application Server Version 7 and later. If you are using WebSphere Application Server for the first time avoid using listener ports.

Connection factory, myCF

A JMS connection factory object defines a set of standard configuration properties for connections. An application uses a connection factory to create a connection to WebSphere MQ.

If your application uses an MDB, as the sample application does, the connection factory is used for outbound messages only; inbound messages are received by the activation specification.

Destination, myQueue

A JMS destination can be a topic or a queue. In this scenario, the destination is a queue, and identifies the WebSphere MQ queue that applications send messages to, or from which an application receives messages, or both. An application looks up the destination in the JNDI namespace to create a connection to the WebSphere MQ queue.

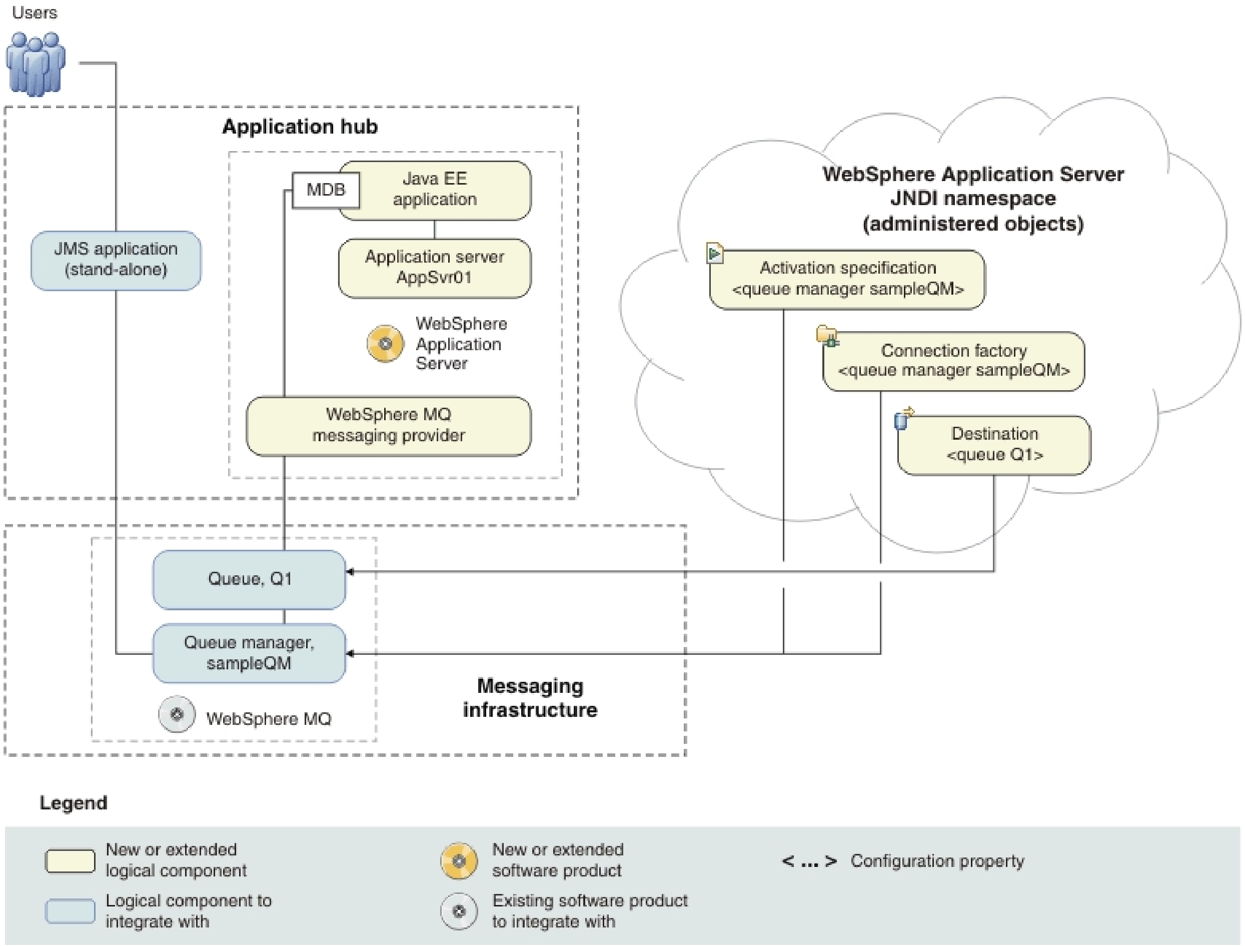
In this scenario, when creating JMS objects in WebSphere Application Server, you specify full details rather than use a *client channel definition table (CCDT)* that has been exported from WebSphere MQ.

Why am I doing this?

A CCDT makes configuring objects in WebSphere Application Server easier, because it contains information that is required to connect to WebSphere MQ. If you do not specify a CCDT you must enter this information yourself. This scenario does not use a CCDT, so that you can see more clearly what information is required to make a connection.

What else might I do or be interested in?

A CCDT is particularly useful if your client applications might need to connect to a number of alternative queue managers. This scenario does not cover that situation.



Users
Application hub
JMS application (stand-alone)
MDB
Java EE application
Application server AppSvr01
WebSphere Application Server
WebSphere MQ messaging provider
WebSphere Application Server JNDI namespace (administered objects)
Activation specification <queue manager sampleQM>
Connection factory <queue manager sampleQM>
Destination <queue Q1>
Queue, Q1
Queue manager, sampleQM
WebSphere MQ
Messaging infrastructure
Legend
New or extended logical component
Logical component to integrate with
New or extended software product
Existing software product to integrate with
< ... > Configuration property

Figure 3. Delivered logical topology. The high-level logical topology diagram for new software functions delivered by integrating WebSphere Application Server into the initial WebSphere MQ messaging infrastructure. This delivered logical topology shows the new functions as new or extended logical components, integrated with the software functions and products from the initial logical topology.

Production physical topology and product mapping

The production physical topology specifies the recommended operational environment (machines, operating systems, and software products) for running the applications and services of the solution.

Node	Operating system and hardware	Software
Host 1 (runtime node)	• Linux • 1 GB physical memory • 3 GB free disk space	• WebSphere MQ Version 7.1 • WebSphere Application Server Version 8 Prerequisite software: • A web browser to use the administrative console.

Chapter 3. Implementing the solution

Implementing the solution in this scenario involves connecting WebSphere Application Server to WebSphere MQ, which provided the initial messaging infrastructure for a company.

Before you begin

The starting point for this scenario is an existing, verified, WebSphere MQ installation as the initial messaging infrastructure.

If you want to try out the scenario, you can either follow the instructions to add WebSphere Application Server to your own WebSphere MQ, or set up a copy of the sample messaging infrastructure as described in "Creating a sample initial IT configuration" on page 14.

About this task

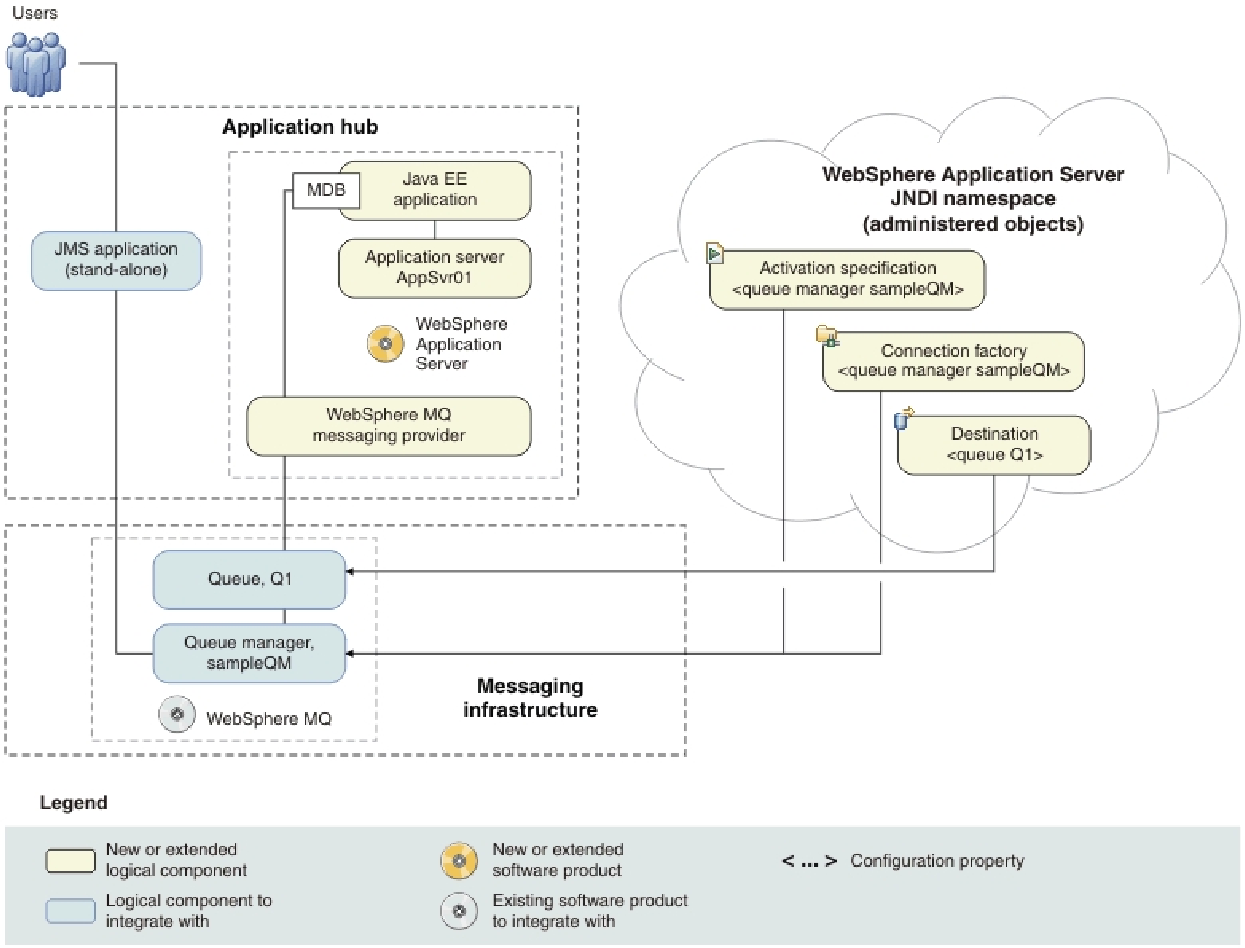

Users
Application hub
MDB
Java EE application
Application server AppSvr01
WebSphere Application Server
WebSphere MQ messaging provider
JMS application (stand-alone)
WebSphere Application Server JNDI namespace (administered objects)
Activation specification <queue manager sampleQM>
Connection factory <queue manager sampleQM>
Destination <queue Q1>
Queue, Q1
Queue manager, sampleQM
WebSphere MQ
Messaging infrastructure
Legend
New or extended logical component
Logical component to integrate with
New or extended software product
Existing software product to integrate with
< ... > Configuration property

Figure 4. Delivered logical topology, including WebSphere Application Server. The high-level logical topology diagram for new software functions delivered by integrating WebSphere Application Server into the initial WebSphere MQ messaging infrastructure. This delivered logical topology shows the new functions as new or extended logical components, integrated with the software functions and products from the initial logical topology.

After connecting WebSphere Application Server to WebSphere MQ, you can use the connection by deploying a Java EE application to run on WebSphere Application Server. This scenario describes use of a sample Java EE application that is deployed onto WebSphere Application Server and used to verify the solution.

Creating a sample initial IT configuration

This scenario was developed by using a sample initial IT configuration. Follow the instructions to set up the sample initial IT configuration to try out the scenario in the same way as it was originally developed.

Installing WebSphere MQ on Linux

Install WebSphere MQ on the Linux operating system by using the command line.

Before you begin

- You must have local administrator authority when you are installing. Define this authority through the Linux facilities.

- Ensure that the host name does not contain any spaces.
- Ensure that you have sufficient disk space, up to 600 MB, to fully install WebSphere MQ for Linux.

If you want to investigate the requirements for installing WebSphere MQ in more detail, see Checking hardware and software requirements in the WebSphere MQ information center.

About this task

Install the WebSphere MQ packages on the WebSphere MQ host to the default location by using the command line.

Procedure

1. Log in to the WebSphere MQ host as root and create a user and group called mqm.
   a. Create the mqm user, in the command prompt enter useradd mqm.
   b. Add the mqm user to the mqm group, enter usermod -g mqm mqm.
2. Insert the WebSphere MQ for Linux Server DVD into the DVD drive.
3. Accept the WebSphere MQ license.
   a. In a terminal, navigate to the / directory. Enter ./mqlicense.sh -accept. The following message is displayed:

```
Licensed Materials - Property of IBM

 5724-H72

 (C) Copyright IBM Corporation 1994, 2019 All rights reserved.

US Government Users Restricted Rights - Use, duplication or disclosure
restricted by GSA ADP Schedule Contract with IBM Corp.



Agreement accepted:  Proceed with install.
```

4. Install the WebSphere MQ packages.
   a. Enter rpm -ivh MQSeries*.rpm to install WebSphere MQ to the /opt/mqm directory. The following is displayed throughout the installation process:

```
Preparing...                ########################################### [100%]
   1:MQSeriesRuntime        ########################################### [  4%]
   2:MQSeriesClient         ########################################### [  9%]
   3:MQSeriesJava           ########################################### [ 13%]
   4:MQSeriesJRE            ########################################### [ 17%]
   5:MQSeriesGSKit          ########################################### [ 22%]
   6:MQSeriesMan            ########################################### [ 26%]
   7:MQSeriesMsg_cs         ########################################### [ 30%]
   8:MQSeriesMsg_de         ########################################### [ 35%]
   9:MQSeriesMsg_es         ########################################### [ 39%]
  10:MQSeriesMsg_fr         ########################################### [ 43%]
  11:MQSeriesMsg_hu         ########################################### [ 48%]
  12:MQSeriesMsg_it         ########################################### [ 52%]
  13:MQSeriesMsg_ja         ########################################### [ 57%]
  14:MQSeriesMsg_ko         ########################################### [ 61%]
  15:MQSeriesMsg_pl         ########################################### [ 65%]
  16:MQSeriesMsg_pt         ########################################### [ 70%]
  17:MQSeriesMsg_ru         ########################################### [ 74%]
  18:MQSeriesMsg_Zh_CN      ########################################### [ 78%]
  19:MQSeriesMsg_Zh_TW      ########################################### [ 83%]
  20:MQSeriesSamples        ########################################### [ 87%]
  21:MQSeriesSDK          ########################################### [ 91%]
  22:MQSeriesServer       ########################################### [ 96%]
  23:MQSeriesTXClient     ########################################### [100%]
```

The minimum components that you must install are:

- MQSeriesRuntime
- MQSeriesServer
- MQSeriesSamples

To install individual components to the default location, such as runtime and server components, use the following command:

```
rpm -ivh MQSeriesRuntime-*.rpm MQSeriesServer-*.rpm MQSeriesSamples-*.rpm
```

Why am I doing this?

You must install the Samples to verify the installation and use the sample applications provided.

What else might I do or be interested in?

If you want to install to a non-default location, use the **rpm --prefix** command. The installation path must be to an empty directory, the root of an unused file system, or a path that does not exist. The length of the path is limited to 256 bytes and must not contain spaces. For example, to install the runtime and server components to /opt/customLocation, use the following command:

```
rpm --prefix /opt/customLocation -ivh MQSeriesRuntime-*.rpm MQSeriesSamples-*.rpm
MQSeriesServer-*.rpm
```

Results

WebSphere MQ is now installed and you are now ready to verify the installation to ensure that the queue manager and queue communicate.

What to do next

To configure the JNDI administered objects, complete the instructions in "Configuring the JNDI namespace and administered objects on Linux."

Related information:

Installing a WebSphere MQ Server

Configuring the JNDI namespace and administered objects on Linux

Configure the JNDI namespace to administer objects that the sample application can use; a queue, two channels and a listener.

About this task

Before an application can retrieve administered objects from a JNDI namespace, an administrator must first create the administered objects. The administrator can use the WebSphere MQ JMS administration tool or WebSphere MQ Explorer to create and maintain administered objects in a JNDI namespace.

This scenario demonstrates the following aspects:

- Use of a JNDI namespace in a local file system. A file system is used because it is the simplest JNDI mechanism for a sample scenario.

What else might I do or be interested in?

The JNDI namespace can be on a file system, Lightweight Directory Access Protocol (LDAP) server, or on another JNDI implementation. If you want to use a JNDI namespace on an LDAP server or another JNDI implementation, you must configure the JNDI namespace and modify the sample application to reference the JNDI namespace.

- Use of the WebSphere MQ *MQSC* commands to create administered objects in the JNDI namespace. The JMS application can look up the administered objects to connect to WebSphere MQ and to access WebSphere MQ destinations with which to send or receive messages.

In this task, you create the following objects in WebSphere MQ:

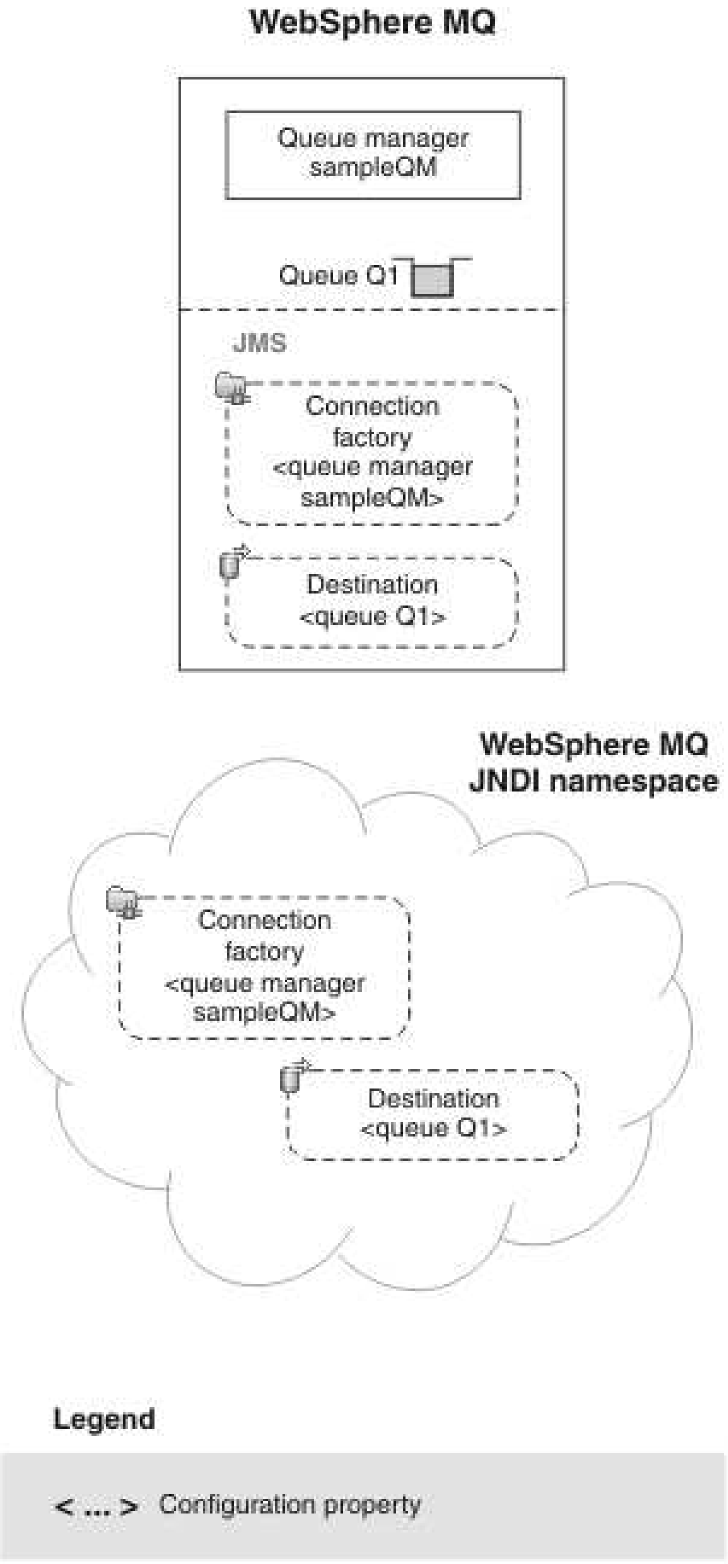

WebSphere MQ
Queue manager sampleQM
Queue Q1
JMS
Connection factory <queue manager sampleQM>
Destination <queue Q1>
WebSphere MQ JNDI namespace
Connection factory <queue manager sampleQM>
Destination <queue Q1>
Legend
< ... > Configuration property

Figure 5. Objects created in WebSphere MQ

Procedure

1. Log in as the root user.

2. Create a JNDI-Directory. Navigate to the / directory, enter mkdir JNDI-Directory.
3. Log in as the mqm user.
4. Create a queue manager.
   a. Navigate to the /opt/mqm/bin directory.
   b. Enter ./crtmqm -ld /var/mnt/mqm/logs -md /var/mnt/mqm/qmgrs sampleQM.

Why am I doing this?

The **-ld** and **-md** parameters are set to use the directories that are mounted on the NFS, providing a highly available set up. The paths must be set up because they are not the default paths.

The following message is displayed:

```
WebSphere MQ queue manager created.
Directory '/var/mnt/mqm/qmgrs/sampleQM' created.
The queue manager is associated with installation 'Installation1'.
Creating or replacing default objects for queue manager 'sampleQM'.
Default objects statistics : 71 created. 0 replaced. 0 failed.
Completing setup.
Setup completed.
```

5. Start the queue manager, enter ./strmqm -x sampleQM.

Why am I doing this?

If you omit the -x, the queue manager starts, but does not permit a standby instance; it is not a multi-instance queue manager.

The following message is displayed:

```
WebSphere MQ queue manager 'sampleQM' starting.
The queue manager is associated with installation 'Installation1'.
5 log records accessed on queue manager 'sampleQM' during the log replay phase.
Log replay for queue manager 'sampleQM' complete.
Transaction manager state recovered for queue manager 'sampleQM'.
WebSphere MQ queue manager 'sampleQM' started using V7.1.0.0.
```

6. Create a configuration file to connect WebSphere MQ to the initial context of the *JNDI namespace*
   a. Create a file called MyAdmin.config. Navigate to the /opt/mqm/java/bin directory. Enter vi MyAdmin.config.

Why am I doing this?

This file creates a separate configuration specifically for JMS administration of the administered objects in the JNDI Namespace.

   b. Type the following into the file:

```
INITIAL_CONTEXT_FACTORY=com.sun.jndi.fscontext.RefFSContextFactory
PROVIDER_URL=file:///JNDI-Directory
SECURITY_AUTHENTICATION=none
export classpath="/opt/mqm/java/lib/com.ibm.mq.jar:/opt/mqm/java/lib/com.ibm.mq.jms.Nojndi.jar:
/opt/mqm/java/lib/com.ibm.mq.soap.jar:/opt/mqm/java/lib/com.ibm.mqjms.jar:/opt/mqm/java/lib/connector.jar:
/opt/mqm/java/lib/fscontext.jar":$CLASSPATH
```

Note: Type the last 3 lines shown for "export classpath" on one line in the file. The lines are split here only for page width restrictions.

c. Press Escape then type `:wq`.

d. On the command line, enter `chmod u+x MyAdmin.config`. The file now has execute permissions.

7. Create the queue manager administered objects.

   a. Start the *MQSC* commands on `sampleQM`. Enter `runmqsc sampleQM`.

   b. Create a local queue, enter `DEFINE QLOCAL(Q1)`.

   c. Create a server-connection channel, enter `DEFINE CHANNEL(JMS.CHANNEL) CHLTYPE(SVRCONN) TRPTYPE(TCP) MCAUSER(mqm)`.

   d. Create a client-connection channel, enter `DEFINE CHANNEL(JMS.CHANNEL) CHLTYPE(CLNTCONN) TRPTYPE(TCP) CONNAME(hostNameOfMQ) QMNAME(sampleQM)`.

   e. Create a listener object, enter `DEFINE LISTENER(listener1) TRPTYPE(TCP) CONTROL(QMGR)`.

   f. Enter `end`. The following message is displayed:

```
 2 : end
Four MQSC command read.
No commands have a syntax error.
All valid MQSC commands were processed.
```

8. Set the JMS environment.

   a. Log out of the `mqm` user and the login as root.

   b. Set the Java path. Enter `export PATH=$PATH:/opt/mqm/java/jre64/jre/bin`.

   c. Navigate to the `/opt/mqm/java/bin` directory. Enter `. ./setjmsenv64` to set the file paths that are associated with Java. The following message is displayed:

```
MQ_JAVA_INSTALL_PATH is /opt/mqm/java
MQ_JAVA_DATA_PATH is /var/mqm
MQ_JAVA_LIB_PATH is /opt/mqm/java/lib64
CLASSPATH is /opt/mqm/java/lib/com.ibm.mq.jar:/opt/mqm/java/lib/com.ibm.mqjms.jar:/opt/mqm/samp/wmqjava/samples:
  /opt/mqm/samp/jms/samples
```

Note: The system displays the CLASSPATH value on one line. The lines are split here only for page width restrictions.

Note: If there is a Java error, repeat steps b and c again.

9. Start the `JMSAdmin`. Enter `./JMSAdmin -cfg MyAdmin.config`. The **`InitCtx>`** prompt is displayed.

10. Define a `JMS` context and change to it.

    a. Type `define ctx(JMS)`.

    b. Type `change ctx(JMS)`. The prompt changes to **`InitCtx/JMS>`**.

11. Create a connection factory that is called `myCF`, enter `define cf(myCF) tran(CLIENT) host(hostNameOfMQ) chan(JMS.CHANNEL) qmgr(sampleQM)`.

12. Create a queue called `Q1`. Enter `define q(myQueue) queue(Q1) qmgr(sampleQM)`

13. Verify the initial context. Enter `display ctx`. The output provides the following information:

```
  JMSADM4089 InitCtx/JMS

      .bindings                 java.io.File
   a  myCF                      com.ibm.mq.jms.MQConnectionFactory
   a  Q1                   com.ibm.mq.jms.MQQueue
```

```
3 Object(s)
  0 Context(s)
  3 Binding(s), 2 Administered
```

Results

The WebSphere MQ objects that are required to use the sample JMS application are configured.

What to do next

Complete the instructions in "Running the sample JMS application" to verify WebSphere MQ communication.

Related information:

JMS connection factories

Creating a destination

Running the sample JMS application

Run the sample stand-alone JMS application to send and receive messages through WebSphere MQ, and verify that you configured WebSphere MQ correctly for use with the sample application.

Before you begin

Download the sample application package. Click the following link and save the file: SampleJMSApp.zip, then extract the contents. The package contains a sample JMS application `.jar` file and shell script files for running the application.

- The sample `sampleJMSApp.jar` file and the script files must be in the same directory.
- The script files use environment variables to set the class path for running the JMS application. When the JMS application is run, if you see a Java `java.lang.NoClassDefFoundError`, you might be required to adjust the class path line in the command file.

About this task

The JMS application comprises a requester client, which sends the initial message, and a responder client, which receives the message and sends a reply.

- Run `runres.sh` in a terminal window to run the responder client.
- Run `runreq.sh` in a terminal window to send a requestor message and receive a reply.

With two windows, you can see the actions of the requester and responder separately and more clearly.

Procedure

1. Log in as the `mqm` user.
2. Run the responder client.
   a. On the command line, navigate to the directory that contains the `sampleJMSApp.jar` and the script files.
   b. Enter `./runres.sh` . The responder client waits for a message.

```
> Connection factory located in JNDI.
> Destination located in JNDI.
> Creating connection to QueueManager.
> Created connection.

> Waiting for message.
```

3. Run the requestor client.
   a. On the command line, navigate to the directory that contains the `sampleJMSApp.jar` and the script files.
   b. Enter `./runreq.sh`. In the Requester window, observe the requester messages. In the Responder window, observe the updated responder messages; the message it received (from the requester client) and the reply message that it sent.

Results

The requestor client window shows the connection status, the message it sent, and the reply message that it received from the responder client:

```
> Connection factory located in JNDI.
> Destination located in JNDI.
> Creating connection to QueueManager.
> Connection created.

> Sending stock request for 'BakedBeans'
> Sent Message ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002b02

> Received Message ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002902 for 'B
akedBeans - 15 tins in stock'

> Closing connection to QueueManager.
> Closed Connection.
--------------------------------------------------------
In this window, observe the messages sent through WebSphere MQ:
- The request message sent
- The reply message received
-----
When ready, press any key to close this window
Press any key to continue . . .
```

The responder client window, shows the updated responder messages; the message it received (from the requester client) and the reply message that it sent:

```
> Connection factory located in JNDI.
> Destination located in JNDI.
> Creating connection to QueueManager.
> Created connection.

> Waiting for message.

> Received Message ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002b02 for 'B
akedBeans'

> Sending Reply Message 'BakedBeans - 15 tins in stock'
> Sent Message ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002902

> Closing connection to QueueManager.
> Closed connection.
--------------------------------------------------------
In this window, observe the updated responder messages
- The request message received (from the requester)
- The reply message sent
-----
When ready, press any key to close this window
Press any key to continue . . .
```

The messages that are shown in the two terminals verify that the requester and responder clients of the sample application can communicate with each other through WebSphere MQ.

What to do next

Complete the instructions in "Installing WebSphere Application Server" to begin the WebSphere Application Server section of the scenario.

Installing WebSphere Application Server

Install WebSphere Application Server by using the graphical user interfaces. At the end of this task, you are ready to configure WebSphere Application Server to communicate with WebSphere MQ.

About this task

The installation process varies depending on which version of WebSphere Application Server you want to install. Starting from Version 8, you install WebSphere Application Server by using IBM Installation Manager. IBM Installation Manager is a product that you can use to install, uninstall, update, and modify IBM software packages. It helps you to manage IBM software more easily.

Preparing your Linux system for installation

Prepare your Linux system for the installation of WebSphere Application Server. Check the hardware and software requirements and set permissions for the files and directories that are created by the installation of WebSphere Application Server.

Before you begin

- Ensure that the default shell for your Linux operating system is `/bin/bash`. Use the following command `readlink /bin/sh` to ensure that your default shell is `bash` not `dash`.
- Check for any post release technical notes available for your operating system at the product support site Support home - WebSphere Application Server.
- Make sure that your Linux system has the following items:
  - Kernel and C runtime library
  - Current® and all compatibility versions of the C++ runtime library
  - X Window System libraries
  - GTK runtime libraries
- Ensure that there is adequate disk space. Installing all features and products requires approximately 2 GB of disk space. This estimate includes the following products, components, and features:
  - Main application server product installation
  - Profiles
  - Sample applications
  - IBM HTTP Server
  - Web Server plug-ins

- Application Client for WebSphere Application Server

- Verify that prerequisites and corequisites are at the required release levels, which can be reviewed on the Support hardware and software website.
- Complete any distribution-specific setup requirements as described in Preparing Linux system for installation.

About this task

This task checks that the default settings of your Linux operating system meet the requirements for the installation of WebSphere Application Server Version 8.0.

Procedure

1. Log on to the WebSphere Application Server host as root and set the suitable authorities for the installation.
   a. Verify the **umask** setting. In the command line, type `umask`.
   b. Set the **umask** setting to `022`, enter `umask 022`. Setting the **umask** to `022` allows only the user to write data, but other users can read the data.

   > **Why am I doing this?**
   >
   > The umask sets the permission modes for files and directories that are created by the installation. Preventing WebSphere Application Server from creating its own permissions for files or directories that you might not want to be written by other users of the group.

2. Increase the **ulimit** setting in the bash command shell profile to prevent **addNode** and **importWasprofile** problems. Set the **ulimit** on your Linux command shells by the `.bashrc` file. The shell profile script is found under your home directory. Open `.bashrc` in a text editor and enter `ulimit -n 8192`.

Results

Your Linux operating system is now ready for WebSphere Application Server to be installed.

What to do next

Follow the instructions in "Installing WebSphere Application Server Version 8 or later" to install IBM Installation Manager, then install WebSphere Application Server.

Installing WebSphere Application Server Version 8 or later

Install WebSphere Application Server 8 or later by using Installation Manager. At the end of this task, you are ready to configure WebSphere Application Server to communicate with WebSphere MQ.

Before you begin

- Follow the instructions in "Preparing your Linux system for installation" on page 23 to prepare for installing WebSphere Application Server on a Linux operating system.
- Make sure that you either have available the installation disk that is labeled "WebSphere Application Server Network Deployment", or know where to access the root directory of the downloaded Passport Advantage® image on your hard disk.

- Set your rollback preference in the installation manager by following the instructions on the Rollback preference page.
- Make sure that you have X Window System installed on your Linux operating system.

About this task

Install WebSphere Application Server by using Installation Manager.

Procedure

1. If you haven't installed IBM Installation Manager before, install it.
   a. Log in as root user.
   b. Extract the Installation Manager archive file.
   c. In a terminal, navigate to Installation Manager archive directory enter `./installc -c`. The following message is displayed:

```
Preprocessing the input.
Loading repositories...
Preparing and resolving the selected packages...


=====> IBM Installation Manager> Install

Select packages to install:
     1. [X] IBM® Installation Manager 1.5.2

     O. Check for Other Versions, Fixes, and Extensions

     N. Next,      C. Cancel
-----> [N]
```

   d. Enter `N` then `A` to accept the license agreement.
   e. Enter `N` until the installation summary like following is displayed:

```
=====> IBM Installation Manager> Install> Licenses> Location> Summary

Target Location:
  Package Group Name         :  IBM Installation Manager
  Installation Directory     :  /opt/IBM/InstallationManager/eclipse

Packages to be installed:
        IBM® Installation Manager 1.5.2

Options:
     G. Generate an Installation Response File

     B. Back,      I. Install,      C. Cancel
-----> [I]
```

   Press Enter. The installation process begins. When the installation is complete, the following message is displayed:

```
=====> IBM Installation Manager> Install> Licenses> Location> Summary>
  Completion

The install completed successfully.

Options:
     R. Restart Installation Manager
-----> [R]
```

   f. Press Enter, and then enter `X` to exit the Installation Manager.
2. Navigate to the `/opt/IBM/InstallationManager/eclipse` subdirectory in the directory where you installed IBM Installation Manager.

3. Enter `./IBMIM` and press Enter. The IBM Installation Manager window is displayed.
4. Click **Install**.
5. If a message is displayed that states `Installation Manager cannot find any packages to install`, add WebSphere Application Server to the repository:
   a. Click the **Repositories** link in the **Repository connections** field. The Preferences window is displayed.
   b. Click **Add Repository** in the upper right of the Preferences window. The Add Repository window is displayed.
   c. Click **Browse**, then find the `repository.config` file for WebSphere Application Server, and then click **OK**
   d. Click **OK**.
   e. Click **Test Connections** on the right of the Preferences window. If a message is displayed that states `All the selected repositories are connected`, WebSphere Application Server is added to the repository.
   f. Click **OK** to close the Test Connection window, then click **OK** again to confirm adding the repository.
6. Select the version number of WebSphere Application Server, for example **Version 8.0.0.0**, and then click **Next**.
7. Click **I accept the terms in the license agreement**, and then click **Next**.
8. Click **Next**, then click **Next** again.
9. In the Features window, select the languages that you want and click **Next**.
10. Select the components that you want and click **Next**.
11. Click **Install**. A progress bar is displayed, which shows that the installation is in progress.
12. When the installation is complete, a window is displayed, which indicates that the packages are installed. Click **Profile Management Tool to create a profile**, and then click **Finish**. WebSphere Application Server is successfully installed and "WebSphere Customization Toolbox" is displayed.

    If "WebSphere Customization Toolbox" window is not displayed, open a terminal and navigate to `/opt/IBM/WebSphere/AppServer1/bin/ProfileManagement`, and then enter `./pmt.sh` to start it.
13. Create an application server profile
   a. In the "WebSphere Customization Toolbox" window, click **Profile Management Tool** among the three buttons on the top.
   b. In the Profile Management Tool, click **Create**.
   c. In Environment Selection window, expand **WebSphere Application Server**, and click **Application server**, and then click **Next**.
   d. In Profile Creation Options window, leave the option as **Typical profile creation**, and then click **Next**.
   e. Choose whether to enable administrative security and click **Next**. If you choose to enable it, every time you use the Administrative Console, you must prove your identity.
   f. Confirm your settings and click **Create**.
   g. When the profile creation is complete, the Profile Creation Complete window is displayed. Clear **Launch the First steps console** and click **Finish**. An application server profile is successfully created.

Results

WebSphere Application Server is installed. If you encountered a problem when installing, see the IBM Installation Manager Information Center. For further information about where to find this information center, see About IBM Installation Manager Information Center.

What to do next

Complete the instructions in "Configuring WebSphere Application Server in the administrative console" to create the required WebSphere Application Server objects.

Related information:

Installing the Update Installer for WebSphere Software

Installing WebSphere Application Server

Configuring WebSphere Application Server in the administrative console

Create an activation specification, connection factory, and queue for WebSphere Application Server applications to use to communicate with WebSphere MQ.

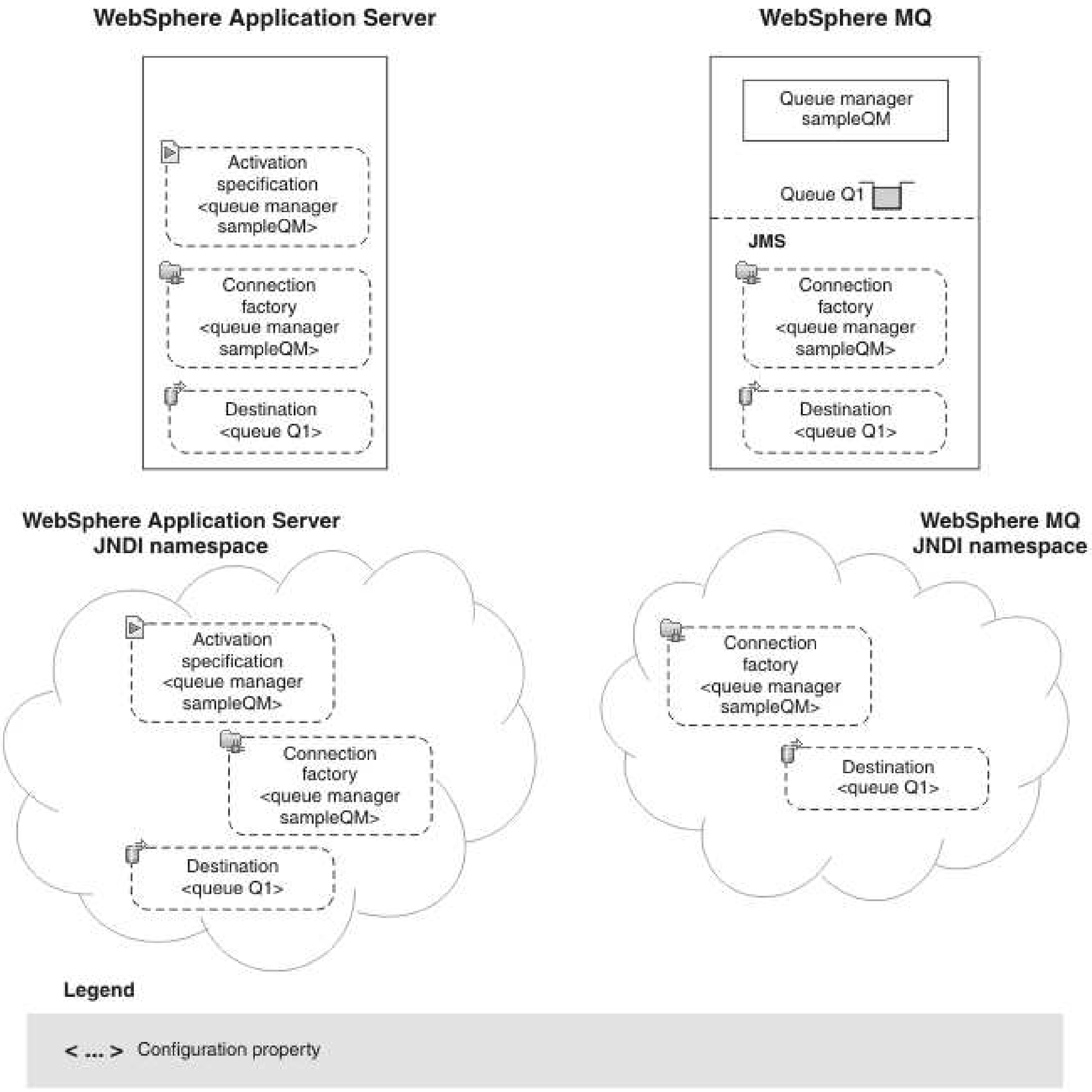
About this task

To use JMS messaging between WebSphere Application Server and WebSphere MQ, you create the following objects in WebSphere Application Server.

- Activation specification
- Connection factory
- Queue (a type of JMS destination)

These objects are described in "Overview: Initial IT configuration" on page 5.

The following image shows these objects in WebSphere Application Server, and the corresponding set of objects in WebSphere MQ that you created when you configured WebSphere MQ for use by the sample application. When you create one of these objects in WebSphere Application Server, it is stored in the WebSphere Application Server JNDI namespace.



WebSphere Application Server
Activation specification <queue manager sampleQM>
Connection factory <queue manager sampleQM>
Destination <queue Q1>
WebSphere MQ
Queue manager sampleQM
Queue Q1
JMS
Connection factory <queue manager sampleQM>
Destination <queue Q1>
WebSphere Application Server JNDI namespace
Activation specification <queue manager sampleQM>
Connection factory <queue manager sampleQM>
Destination <queue Q1>
WebSphere MQ JNDI namespace
Connection factory <queue manager sampleQM>
Destination <queue Q1>
Legend
<...> Configuration property

Figure 6. Objects created in WebSphere Application Server, and the corresponding objects in WebSphere MQ

Procedure

1. If WebSphere Application Server is not already started, navigate to `/opt/IBM/WebSphere/AppServer/bin` and enter `./startServer.sh server1`. An example of the message that is displayed is:

```
ADMU0116I: Tool information is being logged in file
           /opt/IBM/WebSphere/AppServer/profiles/AppSrv01/logs/server1/startServer.log
ADMU0128I: Starting tool with the AppSrv01 profile
ADMU3100I: Reading configuration for server: server1
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server server1 open for e-business; process ID is 26590
```

2. Start a web browser, in the address field enter `host_name:9060/ibm/console`. The `host_name` is the name of your WebSphere Application Server host.

Note: You must log on even if you did not enable administrative security when you installed the product. If you did not enable administrative security, the user ID is used only to distinguish changes that you make in the administrative console from changes that are made by other users.

3. Create an activation specification.

Why am I doing this?

If you want to use message-driven beans with a messaging provider that does not have a JCA 1.5 resource adapter (for example the WebSphere MQ messaging provider), you cannot use activation specifications and therefore you must configure your beans against a listener port.

What else might I do or be interested in?

You can use listener ports instead of activation specifications, however this option is an earlier method which is based on an optional, and now deprecated, part of the JMS specification. Listener ports are stabilized in this version of WebSphere Application Server. For more information, see Message-driven beans, activation specifications, and listener ports and Stabilized features.

a. In the administrative console, click **Resources** > **JMS** > **Activation specifications**.

b. Ensure that the scope is set to **Cell=*cell_name***, then click **New**.

Why am I doing this?

You use the **Scope** property to set the level at which the activation specification is visible. The cell scope is the highest level, giving the activation specification the greatest visibility.

What else might I do or be interested in?

You can also create activation specifications at other levels. For example, if you have multiple servers you might create an activation specification for each server, using the server scope, so that you can specify different settings to be used for each server.

c. Select **WebSphere MQ messaging provider**, then click **OK**. The Configure basic attributes page is displayed.

d. On the Configure basic attributes page, complete the following fields as follows:

Name A name of your choice; for example, `SampleActivationSpec`

JNDI name
Enter `jms/PiShopReplyMDB`

e. Click **Next**.

f. On the Specify MDB destination data page, complete the following field:

Destination JNDI name
The JNDI name for the destination to be used by the activation specification: `jms/wasQueue`

Tip: This name must match what you specify later when you configure the queue.

g. Click **Next**. On the Select connection method page, leave as default, Enter all the required information into this wizard and then click **Next**.

h. On the Supply queue manager details page complete, the following field:

Queue manager or queue-sharing group name
Specify `sampleQM`.

Tip: The queue manager name must be the same as the queue manager you created in the WebSphere MQ configuration.

i. Click **Next** and then on the Enter connection details page, change the following fields:

Transport
Leave as the default setting, **Bindings, then client**.

> **Why am I doing this?**
>
> In this mode, WebSphere Application Server first attempts to connect in bindings mode, where the WebSphere MQ messaging provider within WebSphere Application Server connects directly to the WebSphere MQ queue manager. If this connection attempt fails, the WebSphere MQ messaging provider then attempts to connect by using client mode, which uses a TCP/IP-based network. Bindings mode usually has better performance, but can be used only when WebSphere Application Server and WebSphere MQ are installed on the same machine. The host name, port, and server connection channel information is required for client mode. Bindings mode requires the configuration of native library path information, which you will add in a later step in this task.

Hostname
The name of your machine.

Port Default value is **1414**, if you specified a port number during the WebSphere MQ configuration, enter this value here.

Server connection channel
The name of the WebSphere MQ channel to use for communication.

Specify `JMS.CHANNEL`

> **Why am I doing this?**
>
> Defines the name of the channel a client can use to connect to the queue manager, `sampleQM`.

j. On the “Test connection” page, click **Test connection**.

Why am I doing this?

The **Test connection** button verifies that the activation specification contains the necessary details to create a connection to WebSphere MQ. To use this connection, you must create further administrative objects, as described in the rest of the steps.

If successful, the wizard displays the message `A connection was successfully made to WebSphere MQ`. If not successful, check that the queue manager name, port, and server channel that you specified are correct, and match the corresponding values in WebSphere MQ. For example, the following error indicates that the channel name was not found on the queue manager that you specified.

```
A connection could not be made to WebSphere MQ for the following reason: JMSCMQ0001: WebSphere MQ call failed
 with compcode '2' ('MQCC_FAILED') reason '2540' ('MQRC_UNKNOWN_CHANNEL_NAME').
```

k. Click **Next**, then on the Summary page click **Finish**.

4. Create a connection factory.

Why am I doing this?

A JMS connection factory contains information about how to create a connection. When an application needs a JMS connection, the factory creates a connection instance. The connection factory is used for outbound messages from the MDB, whereas the listener port is used for inbound messages.

What else might I do or be interested in?

Before the JMS 1.1 specification, you had to create either a queue connection factory or a topic connection factory, depending on your required messaging pattern (point-to-point or publish/subscribe). With the *common interfaces* of the JMS 1.1 specification, you can now create a generic connection factory, which creates connections to both types of messaging destination. You can therefore use both messaging models in the same session.

a. In the administrative console, click **Resources** > **JMS** > **Connection factories**.

b. Ensure that the scope is set to **Cell=*cell_name***, then click **New**.

c. Select **WebSphere MQ messaging provider**, then click **OK**. The Connection Factory Settings page is displayed.

d. On the Configure basic attributes page, complete the required fields as follows:

Why am I doing this?

The name that you choose for the connection factory is used to display the connection factory in the administrative console. The JNDI name is used to reference the object in the JNDI namespace. The JNDI name is also used in the sample MDB application code; if you want to use a different JNDI name you must change the MDB to match, then redeploy the application.

Name Specify a name of your choice; for example: `myCF`

JNDI name
Specify `jms/replyCF`

e. Click **Next** and leave the default selection of **Enter all the required infomration into this wizard** and then click **Next**.

f. On the Supply the queue manager details page, complete the required field as follows:

Queue manager or queue-sharing group name
Specify `sampleQM`

Tip: The queue manager name must be the same as the queue manager you created in the WebSphere MQ configuration.

g. Click **Next** and on the Enter connections details page, complete the required fields as follows:

Transport
Leave as the default setting, **Bindings, then client**.

Why am I doing this?

In this mode, WebSphere Application Server first attempts to connect in bindings mode, where the WebSphere MQ messaging provider within WebSphere Application Server connects directly to the WebSphere MQ queue manager. If this connection attempt fails, the WebSphere MQ messaging provider then attempts to connect by using client mode, which uses a TCP/IP-based network. Bindings mode usually has better performance, but can be used only when WebSphere Application Server and WebSphere MQ are installed on the same machine. The host name, port, and server connection channel information is required for client mode. Bindings mode requires the configuration of native library path information, which you will add in a later step in this task.

Hostname
The name of your machine.

Port Specify the port number that is used by the first WebSphere MQ host.

The port number of the WebSphere MQ queue manager sampleQM.

Server connection channel
Specify `JMS.CHANNEL`

Why am I doing this?

Defines the name of the channel a client can use to connect to the queue manager, `sampleQM`.

h. On the "Test connection" page, click **Test connection**.

Why am I doing this?

The **Test connection** button verifies that the connection factory contains the necessary details to create a connection to WebSphere MQ. To use this connection, you must create a queue, as described in the rest of the steps.

If successful, the wizard displays the message `A connection was successfully made to WebSphere MQ`. If not successful, check that the queue manager name, port, and server channel that you specified are correct, then match the values in WebSphere MQ.

i. Click **Next**, then on the Summary page click **Finish**.

5. Create a queue.

Why am I doing this?

A queue is a type of JMS destination, which is used for point-to-point messaging. A destination identifies the messaging target or source.

What else might I do or be interested in?

In a publish/subscribe scenario, you create a topic instead of a queue.

a. In the administrative console, click **Resources** > **JMS** > **Queues**.

b. Ensure that the scope is set to **Cell=*cell_name***, then click **New**.

c. Select **WebSphere MQ messaging provider** then click **OK**.

d. Under General Properties, enter the following values for the required fields.

Name A name of your choice; for example: `myQueue`

This name is used to display the queue in the administrative console.

JNDI name
`jms/wasQueue`

The JNDI name is used to reference this queue in the WebSphere Application Server JNDI namespace.

Tip: This name must match what you specified earlier when you configured the listener port.

Queue name
`Q1`

This value is the administrative name of the queue in WebSphere MQ that you want to communicate with. This name is different from the JNDI name of the queue.

   e. Click **OK**.
6. Click the **Save** link in the Messages section at the top of the pane to save your changes to the master configuration.
7. Stop the application server.
   a. Log in to the WebSphere Application Server host as the root use.
   b. Navigate to /opt/IBM/WebSphere/AppServer/profiles/AppSrv01/bin.
   c. In the command line, enter ./stopServer.sh server1. The following message is displayed:

```
ADMU0116I: Tool information is being logged in file
           /opt/IBM/WebSphere/AppServer/profiles/AppSrv01/logs/server1/stopServer.log
ADMU0128I: Starting tool with the AppSrv01 profile
ADMU3100I: Reading configuration for server: server1
ADMU3201I: Server stop request issued. Waiting for stop status.
ADMU4000I: Server server1 stop completed.
```

8. Start the application server. In the command line, enter ./startServer.sh server1. An example of the message that is displayed is:

```
ADMU0116I: Tool information is being logged in file
           /opt/IBM/WebSphere/AppServer/profiles/AppSrv01/logs/server1/startServer.log
ADMU0128I: Starting tool with the AppSrv01 profile
ADMU3100I: Reading configuration for server: server1
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server server1 open for e-business; process ID is 26590
```

Results

WebSphere Application Server is now configured with the necessary objects to enable communication with WebSphere MQ.

What to do next

Complete the instructions in "Deploying the sample application" to deploy the sample MDB application.

Deploying the sample application

After you deploy the sample message driven bean (MDB) application, you can use it to verify that WebSphere Application Server is communicating with WebSphere MQ.

Before you begin

- Download the sample application. Click the following link and save to the host of WebSphere Application Server: sampleMDB.ear.

About this task

The sample MDB application, sampleMDB.ear, is designed to use the objects that you created earlier in WebSphere Application Server. The MDB application uses these objects to send a message to WebSphere MQ, for receipt by the sample JMS application requester client that you used in "Running the sample JMS application" on page 21.

The following diagram shows a message that travels from the sample JMS client to WebSphere MQ, and then on to WebSphere Application Server, where it is passed to the MDB running within WebSphere Application Server. A response message travels from the MDB to WebSphere MQ, and then on to the JMS client.

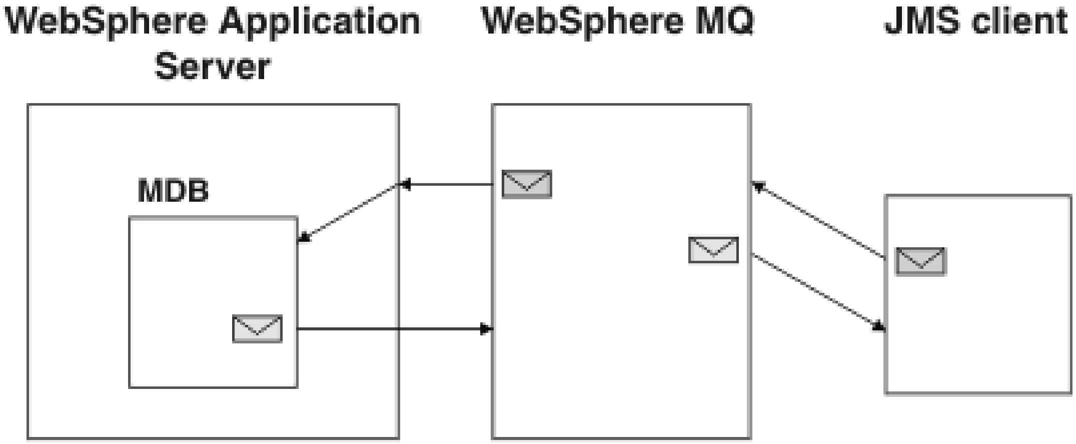


Figure 7. A message that travels from the sample JMS client to WebSphere MQ, then onward to the MDB on WebSphere Application Server

Procedure

1. If the WebSphere Application Server is not already started, navigate to `/opt/IBM/WebSphere/AppServer/profiles/Appsrv01/bin` and enter `./startServer.sh server1`.
2. Start a web browser, in the address field enter `host_name:9060/ibm/console`. The `host_name` is the name of your host.
3. Click **Applications** > **Application Types** > **WebSphere enterprise applications**.
4. In the Enterprise Applications pane, click **Install**.
5. On the first Preparing for the application installation page, select **Local file system**, then browse to the folder where you downloaded the sample MDB application, `sampleMDB.ear`.
   a. Select the application then click **Open**.
   b. Leave all other fields unchanged and click **Next**.
6. For How do you want to install the application?, leave the default option, **Fast path** selected.

   Why am I doing this?
   By choosing the fast path option, you can skip many of the steps on the Install New Application page; you are prompted only when more information is required. For more information, see the topic Preparing for application installation binding settings in the WebSphere Application Server information center.

   Click **Next**.
7. On the Install New Application page, consider whether you want to give a new administrative name to the application.

Why am I doing this?

Unless you specify otherwise, the name of the installed application is derived from the name of the MDB EAR file. For example, in this scenario the default application name *sampleMDB* comes from the EAR file name `sampleMDB.ear`. If you prefer, you can give the application a different name (which in this scenario you must use wherever action on the *sampleMDB* application is described). To do change the name, edit the **Application name** field to specify your chosen name. Select **Deploy enterprise beans** and click **Next**.

8. Click the **Step 3** link to go to the Bind listeners for message-driven beans page. In this step, you specify the information that is required to bind the MDB to an activation specification in the application server.

   Why am I doing this?

   Binding information is used to enable an application to find the enterprise beans or resources that it must work with in the application server. In this scenario, the MDB must be bound to the activation specification and the JMS queue that you configured in "Configuring WebSphere Application Server in the administrative console" on page 27.

   What else might I do or be interested in?

   To save having to specify the binding information whenever you deploy the MDB EAR file, you can preset the binding information for the EAR file in other ways, such as using an assembly tool. For more information, see the topic Preparing for application installation binding settings in the WebSphere Application Server information center.

   On the Bind listeners for message-driven beans page, specify the JNDI name of the activation specification to which the sample MDB application must be bound.

   a. Under **Listener Bindings**, select **Activation Specification**.

      Tip: You do not must select the EJB module, `pishopExampleReplyMDB`, because it is the only EJB module in the EAR file.

   b. In the **Target Resource JNDI Name** field, type the JNDI name, `jms/PiShopReplyMDB`.

      The value must match the JNDI name that you specified in "Configuring WebSphere Application Server in the administrative console" on page 27.

      The other properties are optional, they are provided so that you can override values that are configured on the activation specification. For example, in this scenario the MDB uses the JMS destination, `jms/wasQueue`, that you configured on the activation specification in. (The optional **Destination JNDI name** field overrides the destination that is configured on the activation specification.)

   c. Click **Next**.

9. On the Map resource references to resources page, map the connection factory resource reference that is defined in the sample MDB application to the JNDI name of the connection factory.

Why am I doing this?

Each resource reference that is defined in an application must be mapped to an actual resource that is configured for use by the application. A resource reference allows an application to access a resource by using a logical name, rather than the actual name that is defined in the runtime environment. Resource references eliminate you having to change the application code when you change resource runtime configurations.

a. Under **Target Resource JNDI Name**, click **Browse** to display a list of resources that are available to be used to bind the resource reference of the MDB.

   Tip: For this scenario, the resource that you must select is the connection factory `myCF` that you specified in "Configuring WebSphere Application Server in the administrative console" on page 27.

b. Select **replyCF** then click **Apply**.

c. Leave all other fields unchanged and click **Next**.

10. On the Summary page, click **Finish**. The program displays messages to indicate that the application is being installed.
11. Click **Finish**. The program displays messages to indicate that the application is being installed. If the application does not install successfully, examine the messages to identify why the installation failed. Correct the problems as needed and try installing the application again.
12. When the application installs successfully, click the **Save** link that is displayed at the end of the messages to save your changes to the master configuration.

Results

You have deployed the MDB application, which is now visible in the Enterprise Applications pane.

What to do next

To verify the solution, complete the instructions in "Verifying the solution."

Verifying the solution

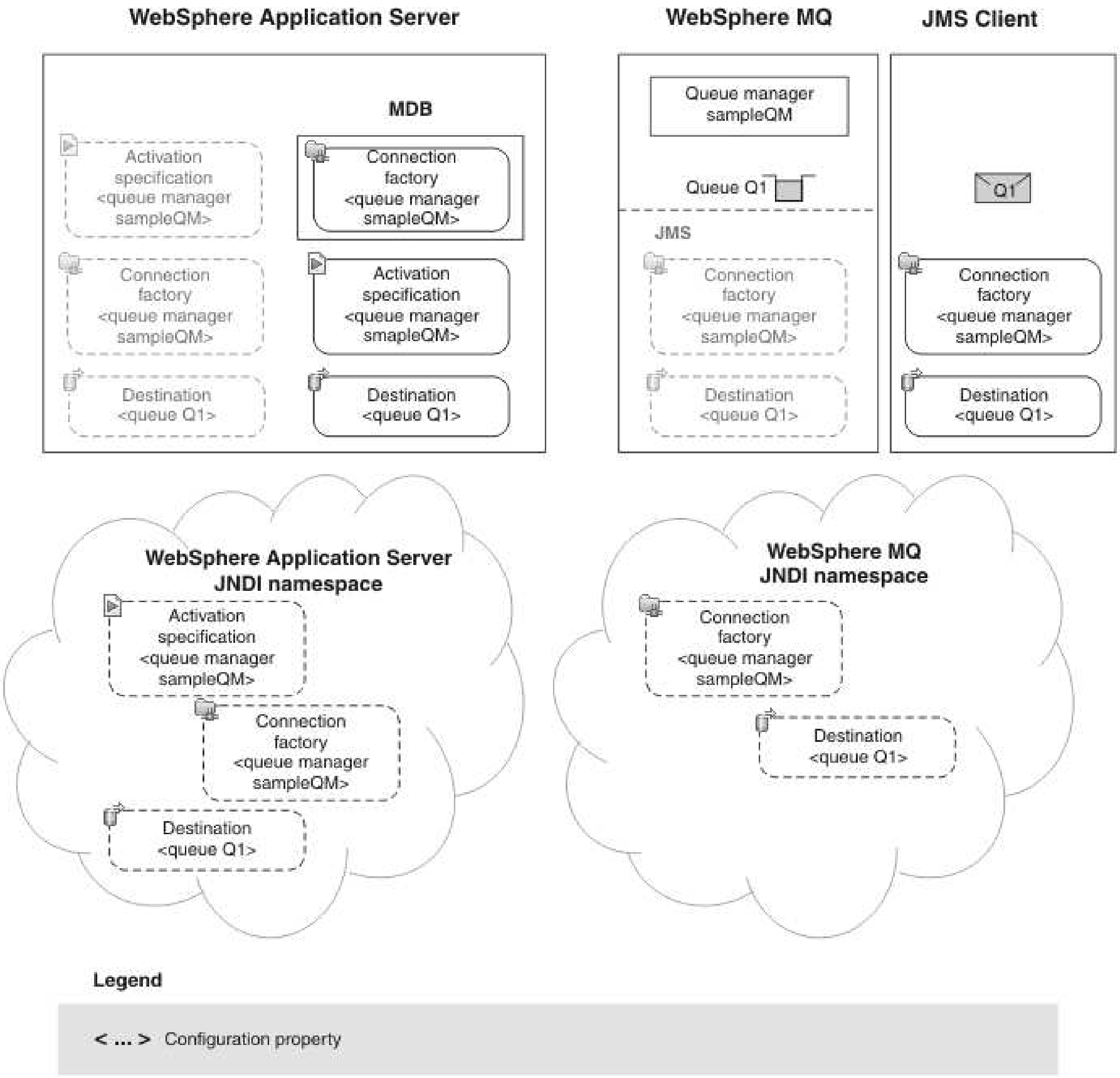
Run the sample JMS and message drive bean (MDB) applications to verify that they can communicate with each other.

Before you begin

You must have WebSphere Application Server and WebSphere MQ configured as described in "Configuring WebSphere Application Server in the administrative console" on page 27 and "Configuring the JNDI namespace and administered objects on Linux" on page 16.

About this task

In "Running the sample JMS application" on page 21, you ran the supplied JMS sample application to verify that the requester and responder clients of the application can communicate through WebSphere MQ. In "Deploying the sample application" on page 34, you installed an MDB application. In this task, you run the requester application as before but the reply comes from the MDB application, instead of the responder application. Verifying that messages are being passed between WebSphere MQ and WebSphere Application Server.


WebSphere Application Server
WebSphere MQ
JMS Client
MDB
Activation specification <queue manager sampleQM>
Connection factory <queue manager smapleQM>
Connection factory <queue manager sampleQM>
Activation specification <queue manager smapleQM>
Destination <queue Q1>
Destination <queue Q1>
Queue manager sampleQM
Queue Q1
JMS
Connection factory <queue manager sampleQM>
Destination <queue Q1>
Q1
Connection factory <queue manager sampleQM>
Destination <queue Q1>
WebSphere Application Server JNDI namespace
Activation specification <queue manager sampleQM>
Connection factory <queue manager sampleQM>
Destination <queue Q1>
WebSphere MQ JNDI namespace
Connection factory <queue manager sampleQM>
Destination <queue Q1>
Legend
< ... > Configuration property

Procedure

1. If WebSphere Application Server is not already started, log in as root and navigate to `/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/bin/` and enter `./startServer server1`.
2. Start a web browser, in the address field enter `host_name:9060/ibm/console`. The `host_name` is the name of your host.
3. Click **Applications** > **Application Types** > **WebSphere enterprise applications** to display a list of installed applications.

4. Select the sampleMDB application and click **Start**. The application status changes to a green arrow. The MDB application is now waiting to receive a message.

   If you see an error message similar to the following message, there might be a mismatch between your WebSphere Application Server configuration and the information that is contained in the sample application. Check in the WebSphere Application Server error logs `*install_directory*/profiles/AppSrv01/logs/server1` for more information.

```
ErrorsampleMDB failed to start. Check the logs for server server1 on node mymachineNode01 for details.
ErrorAn error occurred while starting sampleMDB. Check the logs for server server1 on node mymachineNode01
 for more information.
```

5. Log in to the WebSphere MQ host as the `mqm` user.
6. Run the requestor client.
   a. On a terminal or command prompt window, navigate to the directory that contains the `sampleJMSApp.jar` and the script files.
   b. Enter `./runreq.sh` This command runs the sample JMS requester client, which sends a message to WebSphere MQ.

Results

The sample requester application sends the same request message as before, but the response now comes from the MDB application. The following output, showing sent and received messages, is displayed in the requester application Command Prompt window:

```
> Connection factory located in JNDI.
> Destination located in JNDI.
> Creating connection to QueueManager.
> Connection created.

> Sending stock request for 'BakedBeans'
> Sent Message ID=ID:414d5120514d5f574153494e5354414c423fe14c20010602

> Received Message ID=ID:414d5120514d5f574153494e5354414c423fe14c2000f205 for
'BakedBeans - 15 tins in stock. MDB has connected to business session bean to find
 expected delivery time: 1 day'

> Closing connection to QueueManager.
> Closed Connection.
```

If you see errors similar to the following error in the WebSphere Application Server logs, check that the name of the queue in WebSphere Application Server matches the administrative name (not the JNDI name) of the queue in WebSphere MQ:

```
WebSphere MQ call failed with compcode '2' ('MQCC_FAILED') reason '2085' ('MQRC_UNKNOWN_OBJECT_NAME')

com.ibm.msg.client.jms.DetailedInvalidDestinationException: JMSWMQ2008: Failed to open MQ queue 'myQueue'.
 JMS attempted to perform an MQOPEN, but WebSphere MQ reported an error. Use the linked exception to
 determine the cause of this error. Check that the specified queue and queue manager are defined correctly.
```

Note: If you update the WebSphere Application Server configuration, you must restart the application server for the changes to take effect.

The sample It configuration is implemented, and verified. WebSphere Application Server can connect to WebSphere MQ for the successful transmission of messages.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing 2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Software Interoperability Coordinator, Department 49XA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

Programming interface information, if provided, is intended to help you create application software for use with this program.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Important: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at Copyright and trademark information (www.ibm.com/legal/copytrade.shtml).

Sending your comments to IBM

We appreciate your input on this publication. Feel free to comment on the clarity, accuracy, and completeness of the information or give us any other feedback that you might have.

Use one of the following methods to send us your comments:

- Send an email to ibmkc@us.ibm.com
- Use the form on the web here: www.ibm.com/software/data/rcf/

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

Include the following information:

- Your name and address
- Your email address
- Your telephone or fax number
- The publication title and order number
- The topic and page number related to your comment
- The text of your comment

IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you submit.

Thank you for your participation.

Printed in USA