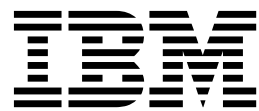


IBM Product Connectivity Scenarios and Patterns
Version 1 Release 0

*Connecting WebSphere Application
Server Liberty Profile V9 to IBM MQ V9*



Note

Before using this information and the product it supports, read the information in "Notices" on page 53.

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.

© **Copyright IBM Corporation 2017.**

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

Contents

| | | | |
|--|-----------|--|-----------|
| Figures | v | Installing Liberty | 31 |
| Chapter 1. Scenario: Connecting WebSphere Application Server Liberty to IBM MQ on Windows | 1 | Creating a Liberty profile server | 32 |
| Chapter 2. Planning the solution | 3 | Configuring Liberty | 32 |
| Assumptions | 3 | Deploying the sample application to Liberty | 34 |
| Business overview | 3 | Verifying the solution | 36 |
| User roles and interactions. | 4 | Option 2: Installing and configuring Liberty by using the graphical user interface | 38 |
| Technical solution. | 6 | Installing Liberty | 39 |
| Overview: Initial IT configuration | 7 | Installing WebSphere Application Server Developer Tools | 40 |
| Overview: The delivered logical topology. | 9 | Creating a Liberty profile server | 41 |
| Production physical topology and product mapping | 12 | Configuring Liberty | 42 |
| Chapter 3. Implementing the solution | 13 | Deploying the sample application to Liberty | 44 |
| Creating a sample initial IT configuration | 14 | Verifying the solution | 45 |
| Installing IBM MQ | 14 | Verifying IBM MQ and Liberty with the additional sample application | 47 |
| Configuring the JNDI namespace and administered objects | 17 | Preparing to run the additional sample application | 48 |
| Configuring security for the sample queue manager | 24 | Running the additional sample application | 50 |
| Running the sample JMS application | 27 | Notices | 53 |
| Obtaining the IBM MQ resource adapter | 29 | Programming interface information | 55 |
| Option 1: Installing and configuring Liberty by using the command line | 30 | Trademarks | 55 |
| | | Sending your comments to IBM | 57 |

Figures

| | | | | |
|--|----|-----|---|----|
| 1. User role interactions to deploy the solution | 6 | 8. | | 37 |
| 2. Initial IT configuration | 9 | 9. | A message traveling from the sample JMS client to IBM MQ, then onward to the MDB on Liberty | 44 |
| 3. Delivered logical topology | 11 | 10. | | 46 |
| 4. Production physical topology | 12 | | | |
| 5. Delivered logical topology, including Liberty | 13 | | | |
| 6. Objects created in IBM MQ | 18 | | | |
| 7. A message traveling from the sample JMS client to IBM MQ, then onward to the MDB on Liberty | 35 | | | |

Chapter 1. Scenario: Connecting WebSphere Application Server Liberty to IBM MQ on Windows

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

This scenario was developed using a sample IBM MQ installation, and uses sample applications to demonstrate the use of Liberty connected to IBM 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 Liberty with IBM MQ.

This scenario was developed and tested with WebSphere Application Server Liberty Version 9.0 and IBM MQ Version 9.0.

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:


This scenario in IBM Knowledge Center

This scenario in the Scenarios and Patterns product documentation

 Product web page
WebSphere Application Server product web page

 Library page
WebSphere Application Server library page

 Product web page
IBM MQ product web page

 Library page
IBM MQ library page

Chapter 2. Planning the solution

You can connect WebSphere Application Server Liberty to IBM MQ. 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 applies to the use of WebSphere Application Server Liberty Version 9.0 with IBM MQ Version 9.0. This scenario was developed by using IBM MQ Version 9.0.0.

This scenario assumes the following points:

- You are using the Windows operating system.
- You use the IBM MQ graphical user interface, rather than the command-line equivalents.

Business overview

A company wants to add a Java EE application on the WebSphere Application Server Liberty to consume and process messages from an existing messaging infrastructure that is provided by IBM MQ.

To date, company A has a business solution that uses IBM 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 IBM 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 current method of processing messages is inadequate. In addition, 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 updated methods of processing before the messages are returned to business users. The company decides that they want to use the Java Message Service (JMS), the Java Platform, Enterprise Edition (Java EE) standard for messaging, which provides a standard API for applications that implement enterprise messaging with application portability. The company also recognizes that the Java EE standard streamlines application development, and with these standards they can create reusable, platform-independent modules.

The new company model also focuses on reducing development and maintenance costs, shortening development schedules, and significantly improving time to

productivity. The development environment must be easy to use, collaborative, and dynamic so that the most frequent tasks can be completed more quickly and with less effort.

The company decides to add a Java EE application on Liberty to consume and process messages in the solution. Liberty is a dynamic WebSphere Application Server profile, available on WebSphere Application Server Version 8.5.5 and later that enables the server to provision only the features that are required by the applications that are deployed to the server. This dynamic profile minimizes the server startup time and memory footprint. Any new features can be added without needing to restart the server. This ability is important in a development environment, where application capabilities are built up iteratively, classes are modified, resources are added, and problems are fixed. Code and configuration changes are easy to make and reflected immediately in the development environment. The combination of Liberty and the WebSphere Application Server Developer Tools for Eclipse provides a rich experience for developers; focusing on Java EE web and mobile tools, and integration of Liberty for the debug and test server environment. Developers can now develop, assemble, test, and publish web applications by using lightweight tools to lightweight development run time environments. Continuing the development-centric focus of Liberty, configuration of the server is now through a simple XML file, which is easy to author, maintain in a version control system, and share.

With Liberty, the company appreciates that as their business demands increase, they can take advantage of the latest standards and programming models that Liberty supports, or even scale up their business capability by adopting higher-performance features or editions of WebSphere Application Server. The tools and capabilities of Liberty make it easy to deploy web applications to other WebSphere Application Server production environments without change.

Related information:

- [🔗 WebSphere Application Server features and benefits web page](#)
- [🔗 Liberty profile overview](#)
- [🔗 Installing the Liberty profile developer tools and \(optionally\) the Liberty profile](#)

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 Liberty to IBM 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 IBM MQ and Liberty. 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 IBM 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?

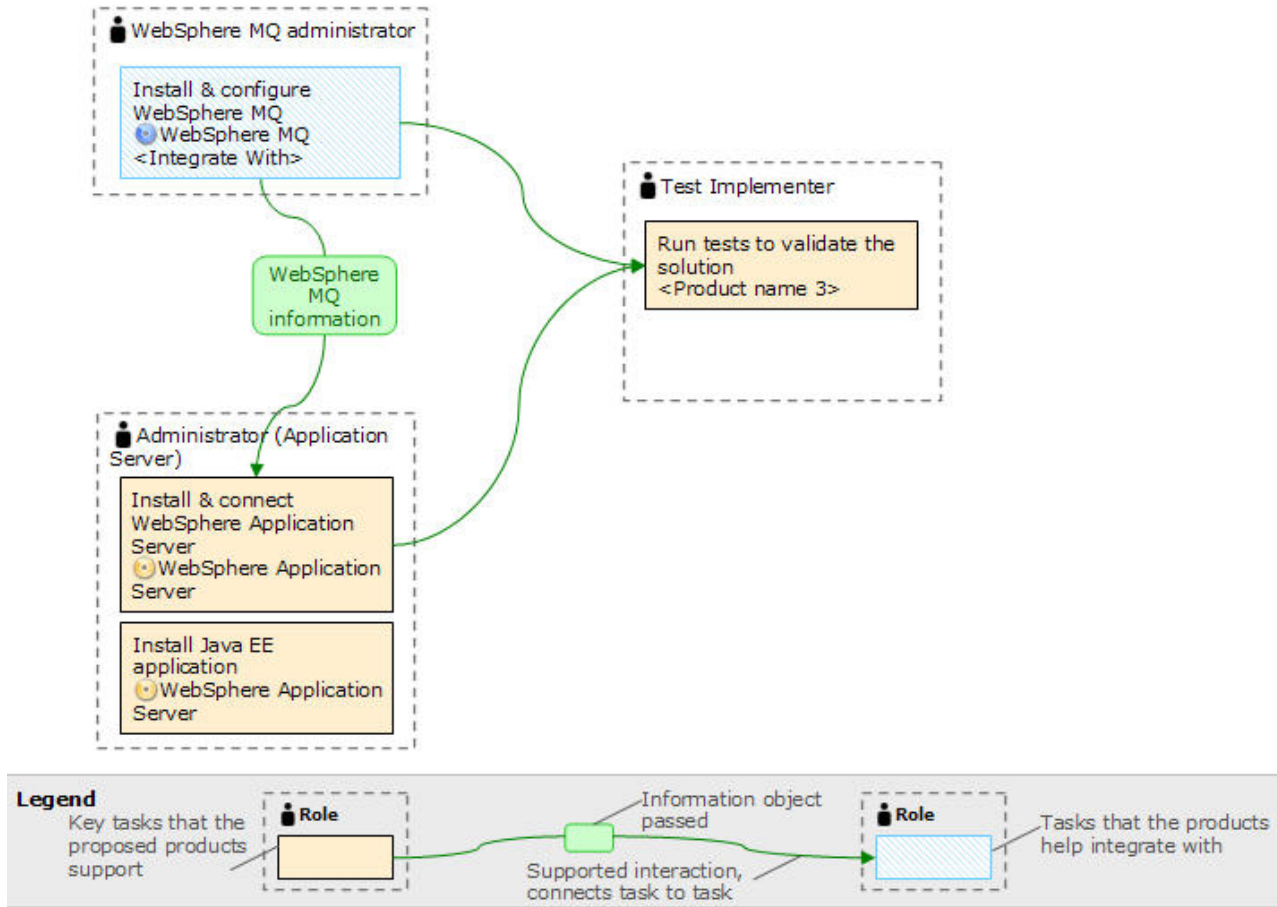


Figure 1. User role interactions to deploy the solution

Technical solution

This scenario describes one way of connecting WebSphere Application Server Liberty to an existing IBM MQ system, by using Java Message Service (JMS), Java EE Connector Architecture (JCA) and Java Naming Directory Interface (JNDI).

These concepts can be defined as follows:

Java Message Service

Java Message Oriented Middleware, which business users and clients use to interact with.

Java EE Connector Architecture(JCA)

Java-based solution for connecting application servers and enterprise information systems.

Java Naming Directory Interface(JNDI)

Java API for a directory service, that allows clients to search and look up data via a name.

Overview: Initial IT configuration

Company A has an existing IT configuration that uses IBM MQ for its messaging infrastructure. This scenario describes adding WebSphere Application Server Liberty 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 9:

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 Platform, Enterprise Edition (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 Client for C/C++ and IBM Message Service Client for .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 IBM MQ queue manager that provides the initial messaging infrastructure. It hosts the queue that the JMS application works with.

Q1 [Message queue]

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

JNDI namespace

A Java Naming Directory Interface (JNDI) namespace is used to hold JMS administered objects, which applications can use to connect to IBM 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 IBM 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 IBM MQ.

Destination, myQueue

A JMS destination can be a topic or a queue. In this scenario, the destination is a queue, and identifies the IBM 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 IBM MQ queue.

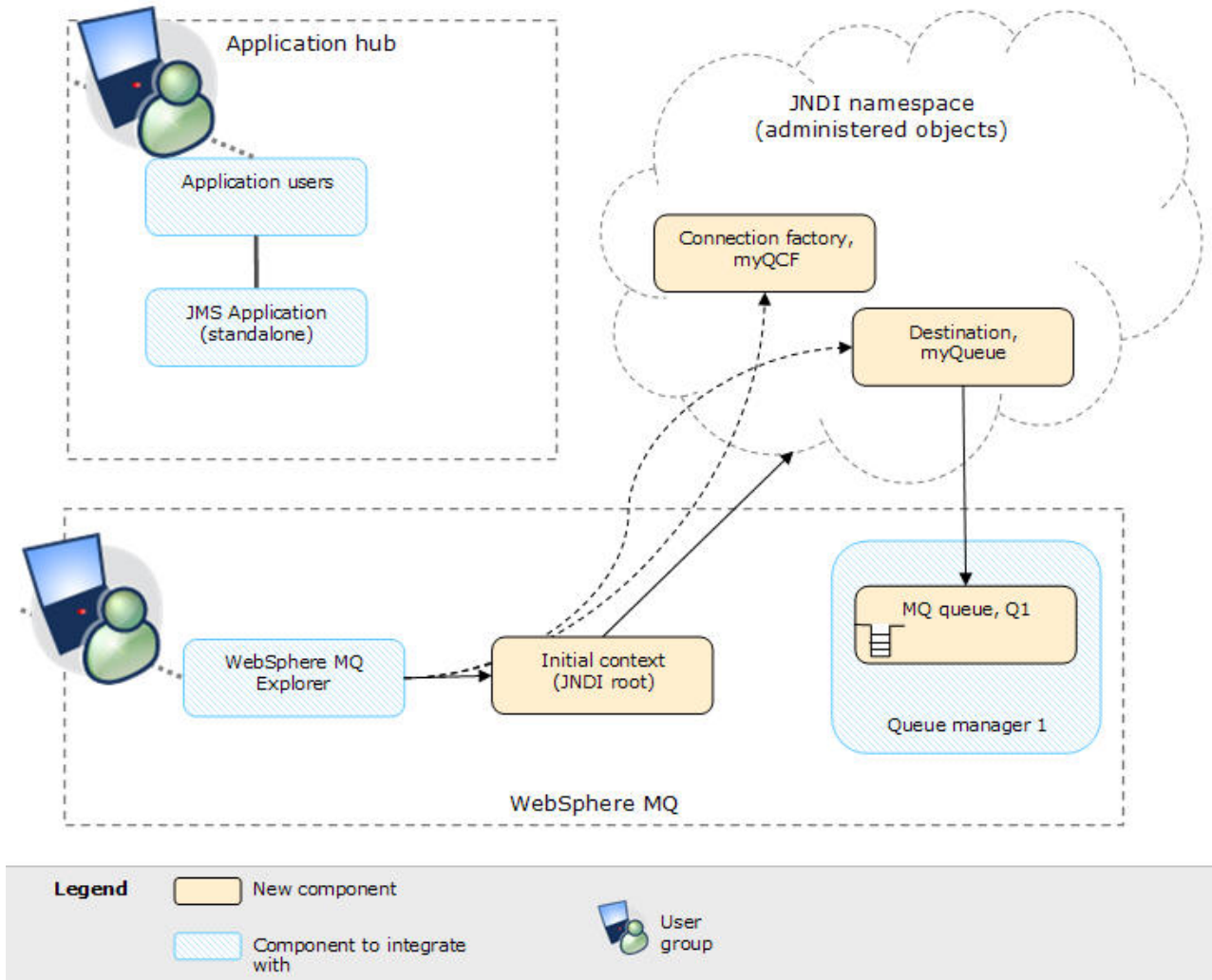


Figure 2. Initial IT configuration. The initial IT configuration includes an initial context, added for IBM MQ Explorer to connect to the root of the JNDI namespace. The JNDI namespace includes a connection factory, added for the sample JMS application to use to connect to IBM MQ, and a destination, added for the sample JMS application to connect to the IBM MQ queue. That IBM 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 in Figure 1. If you want to try out the scenario, 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 Liberty to consume and work with messages from an existing messaging infrastructure that is provided by IBM MQ.

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

IBM MQ as a *messaging provider* for Liberty

The IBM MQ messaging provider in Liberty makes JMS messaging available to Liberty by using the existing capabilities in the IBM MQ environment.

Why am I doing this?

Liberty can interact with IBM MQ destinations to send and receive messages in the same way as any JMS application in the IBM MQ environment.

Java EE application

The application consumes and works with messages on the Java EE queue, Q1. This application runs on a Liberty server that has been created and connected with IBM 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 Liberty 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.

Server A program in Liberty that provides the execution environment for Java EE application programs.

Liberty JNDI namespace

Liberty includes a name server that provides access to the following JMS administered objects through the Java Naming 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 7.

Activation specification

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 IBM MQ, with Java EE application servers such as Liberty.

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 IBM 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 IBM 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 IBM MQ queue.

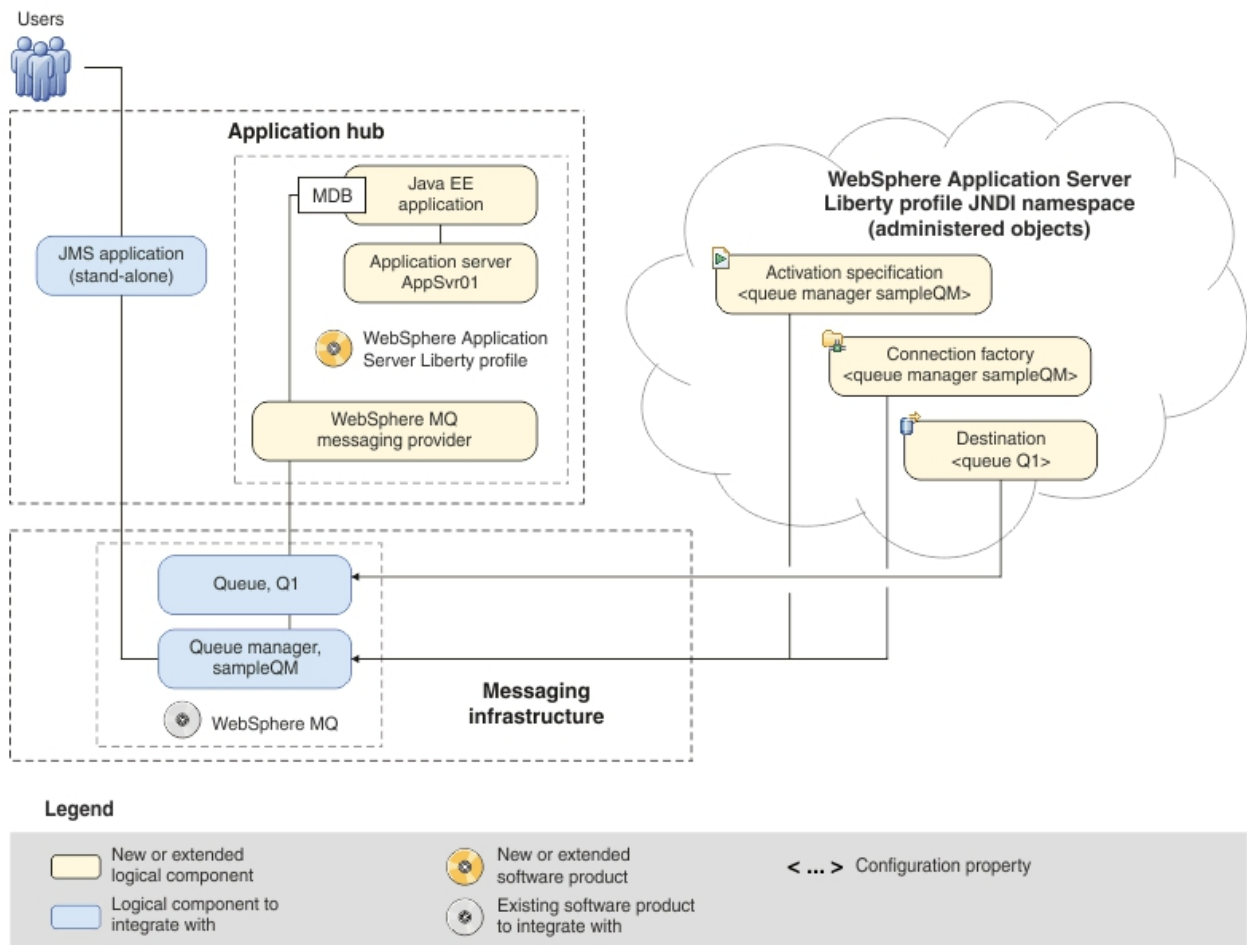


Figure 3. *Delivered logical topology.* The high-level logical topology diagram for new software functions that are delivered by integrating Liberty into the initial IBM MQ messaging infrastructure. This delivered logical topology shows the new functions as new or extended logical components, integrated 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.



Figure 4. Production physical topology

| Node | Operating system and hardware | Software |
|-----------------------|--|---|
| Host 1 (runtime node) | <ul style="list-style-type: none"> Windows Server, Standard or Enterprise Edition. For more information about supported versions, see the System Requirements web pages for IBM MQ and WebSphere Application Server For information about hardware requirements, see the System Requirements web pages for IBM MQ and WebSphere Application Server | <ul style="list-style-type: none"> IBM MQ WebSphere Application Server Liberty Prerequisite software: <ul style="list-style-type: none"> Java Runtime Environment supported by WebSphere Application Server Liberty. The minimum versions of Java is listed here |

Related information:

[↗ System requirements for IBM MQ Version 9.0](#)

[↗ System Requirements for WebSphere Application Server Version 9.0](#)

Chapter 3. Implementing the solution

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

Before you begin

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

If you want to try out the scenario, you can set up a copy of the sample messaging infrastructure as described in “Creating a sample initial IT configuration” on page 14.

About this task

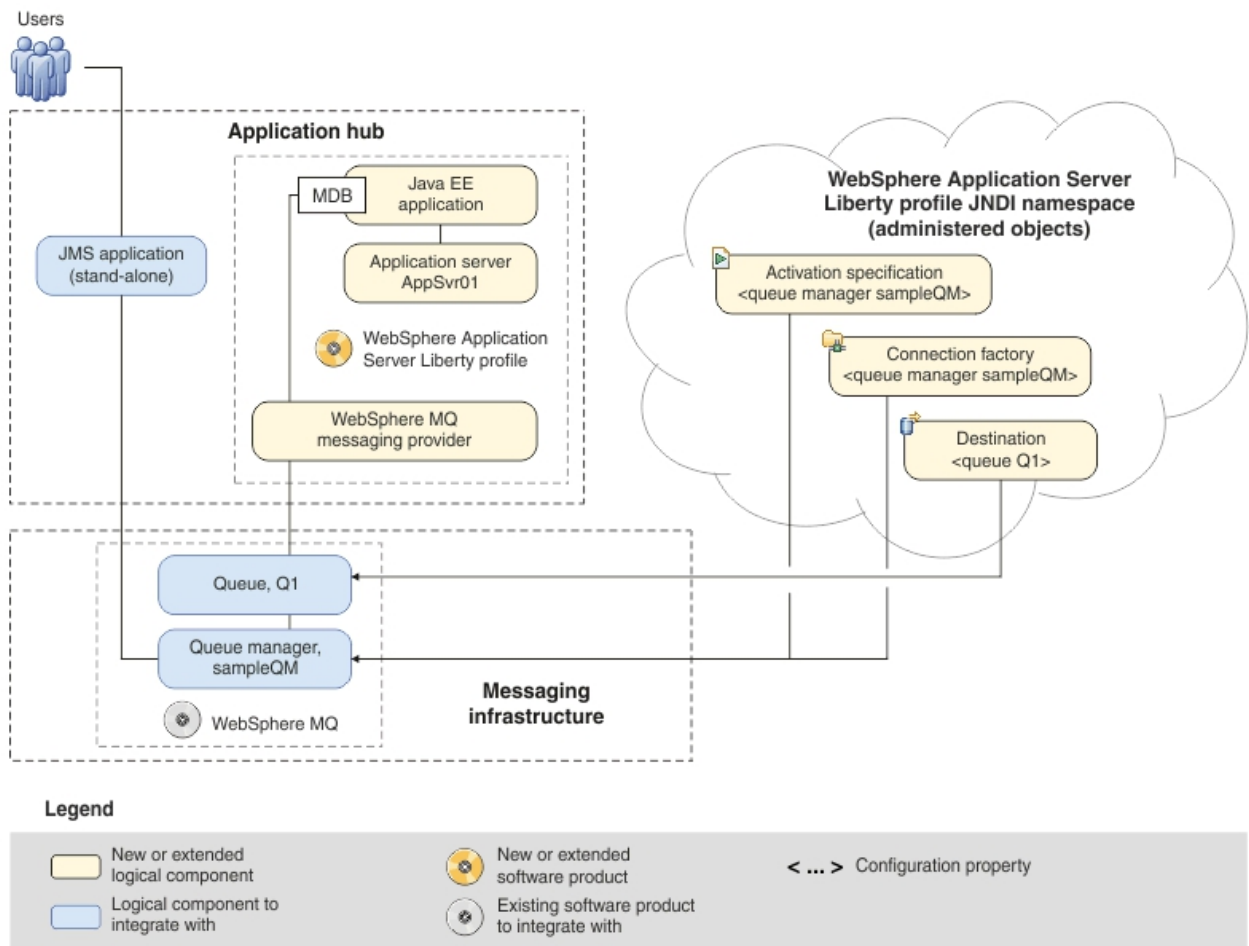


Figure 5. Delivered logical topology, including Liberty. The high-level logical topology diagram for new software functions delivered by integrating Liberty into the initial IBM MQ messaging infrastructure. This delivered logical topology shows the new functions as new or extended logical components, integrated with the software functions and products from the initial logical topology.

After connecting Liberty to IBM MQ, you can use the connection by deploying a Java EE application to run on Liberty. This scenario describes use of a sample Java EE application that is deployed onto Liberty 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 IBM MQ

To complete this scenario, you must have a functioning IBM MQ messaging network. This task provides basic instructions to install IBM MQ so you can set up the sample IT configuration that is used for this scenario.

Before you begin

- You must have local administrator authority when you are installing. Define this authority through the Windows facilities.
- Ensure that the machine name does not contain any spaces.
- Ensure that you have sufficient disk space, up to 1310 MB, to fully install IBM MQ for Windows.
- Determine whether any IBM MQ users are defined on Windows 2000 or later domain controllers.

Why am I doing this?

IBM MQ checks that only authorized users can access queue managers or queues. When a user attempts such access, IBM MQ uses its own local account to query information about the user. However, domain controllers that run on Windows 2000 or later can be configured such that IBM MQ cannot use local accounts to make these queries. In this situation, you must provide IBM MQ with a special account to use. This account is necessary when both of the following conditions apply:

- Any domain controller on your network is running on Windows 2000 or later.
- Local user accounts are not authorized to query the group membership of the domain user accounts.

If you require a special account, refer your domain administrator to the instructions in *Creating and setting up domain accounts for IBM MQ* in the IBM MQ product documentation.

If you want to investigate the requirements for installing IBM MQ in more detail, see *System Requirements for IBM MQ V9*.

About this task

This task describes the basic steps to install IBM MQ Version 9.0 on Windows operating system. This information relates to the Windows 7 Professional (SP1) operating system and later versions.

The installation programs contain links to further information if you require it during the installation process. The installation process has four parts:

1. Use the Launchpad to check and install software requirements, specify network information, and start the IBM MQ installation wizard.
2. Use the IBM MQ installation wizard to install the software, and start the Prepare IBM MQ wizard.
3. Use the Prepare IBM MQ Wizard to start the IBM MQ service and start the Default Configuration wizard.
4. Optionally, use the Default Configuration wizard to create objects for verifying the installation by using the Postcard application that is supplied with IBM MQ.

Procedure

1. Start the installation process appropriate for your media:
 - Using an installation image that is downloaded from Passport Advantage or Passport Advantage Express, double-click the `Setup.exe` file. Instructions on how to download the installation image by using Passport Advantage® can be found on the IBM Support website.
 - Alternatively, insert the IBM MQ DVD into the DVD drive where if AutoRun is enabled, the installation process starts. Otherwise, double-click the **Setup** icon in the root folder of the DVD.

The installation launchpad is started.

2. Use the buttons on the left of the Launchpad to review and, if necessary, modify the software requirements and network configuration.
3. On the IBM MQ Installation page of the Launchpad, select the installation language, and then click **Launch IBM MQ Installer** to start the IBM MQ installation wizard.
4. Click **I accept the terms in the license agreement**, and then click **next**.
5. Click **Typical**, and then click **Next**. Then, click **Install**. The following features are installed:
 - IBM MQ Server
 - IBM MQ Explorer: a graphical interface for administering and monitoring IBM MQ resources
 - IBM MQ Development Toolkit
 - Java and .NET Messaging and Web Services

At the end of the process, the IBM MQ Setup window displays the message **Installation Wizard Completed Successfully**. Click **Finish**; the Prepare IBM MQ Wizard starts automatically.

6. Follow the instructions in the Prepare IBM MQ Wizard.

Why am I doing this?

The Prepare IBM MQ Wizard helps you to configure IBM MQ files and a user account for your network. You must run the wizard to configure the IBM MQ Service before you can start any queue managers.

- a. On the first page of the wizard, click **Next**. The wizard performs some configuration and displays status messages, and then displays the IBM MQ Network Configuration page. The contents of the page varies according to your use of domain controllers.
- b. Follow the instructions for network configuration then click **Next** to continue through the wizard until the wizard displays the Completing the Prepare IBM MQ Wizard page.

- c. Select **Launch IBM MQ Explorer** and choose whether to start a web browser to view the release notes, and then click **Finish**. IBM MQ Explorer starts.
7. Optional: If you want to use the IBM MQ Postcard application to verify your installation, create the default configuration.

Why am I doing this?

The default configuration creates objects, such as a queue manager called *QM_machine_name*, that you can use to verify the success of your installation by using the Postcard application that is supplied as part of IBM MQ. If you do not create the default configuration, you can still use the Postcard application later on after you have created your own objects in IBM MQ.

Note: You cannot create the default configuration if you have already created other queue managers; you must first delete the other queue managers then run the Default Configuration wizard.

- a. If the Content page is not already displayed, click **Window > Show View > MQ Explorer - Content** to display it.
- b. Click **Create the Default Configuration**. The IBM MQ Default Configuration window is opened.
- c. Click **Set up Default Configuration**. The Default Configuration Wizard is opened.
- d. Click **Next** and **Next** again to move through the information pages.
- e. On the Select Options page, clear both **Allow remote administration of the queue manager** and **Join the queue manager to the default cluster**, and then click **Next**.

Why am I doing this?

This scenario assumes that you have one machine on which you are installing Liberty and IBM MQ. Therefore, you do not need to administer the queue from another machine. Clustering is not described in this scenario.

- f. On the summary page, click **Finish**. The Default Configuration Wizard is closed. In the IBM MQ Default Configuration window, the following message is displayed: "Default configuration is partially complete."
 - g. Click **Close**.
8. If you created a default configuration, verify your installation by using the Postcard application that is supplied with IBM MQ.
 - a. If the Content page is not already displayed, click **Window > Show View > MQ Explorer - Content** to display it.
 - b. Click **Launch Postcard** to open the Postcard - Sign On window.
 - c. Click **Help** in this and other Postcard application windows to view instructions about running the Postcard application.

Note: Running the Postcard application on a non-default configuration automatically creates a queue that is called postcard on your queue manager. You can delete this queue after you use the postcard application.

Results




IBM MQ is installed and you are ready to configure objects such as queues and queue managers.

If problems occurred during installation, use the .log files in the C:\Documents and Settings\userID\Local Settings\Temp directory for Windows XP or C:\Users\userID\AppData\Local\Temp for Windows Vista or later, and the amqmsccw.txt and amqmjpse.txt files in the installation directory, to investigate. The default installation directory is C:\Program Files\IBM\MQ for Windows 32-bit operating system or C:\Program Files (x86)\IBM\MQ for Windows 64-bit operating system.

What to do next

Follow the instructions in “Configuring the JNDI namespace and administered objects” to configure IBM MQ for use by the sample application.

Related information:

-  [Launchpad instructions](#)
-  [Configuring WebSphere MQ with the Prepare WebSphere MQ Wizard](#)
-  [Installing the WebSphere MQ Server](#)

Configuring the JNDI namespace and administered objects

Define to IBM MQ an initial context for the JNDI namespace, then define in the namespace the administered objects that the sample application can use.

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 IBM MQ JMS administration tool or IBM MQ Explorer to create and maintain administered objects in a JNDI namespace.

This scenario demonstrates the following aspects:

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

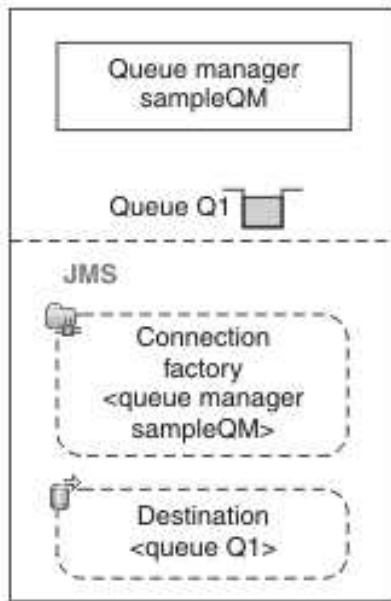
What else might I do or be interested in?

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, as required by the implementation.

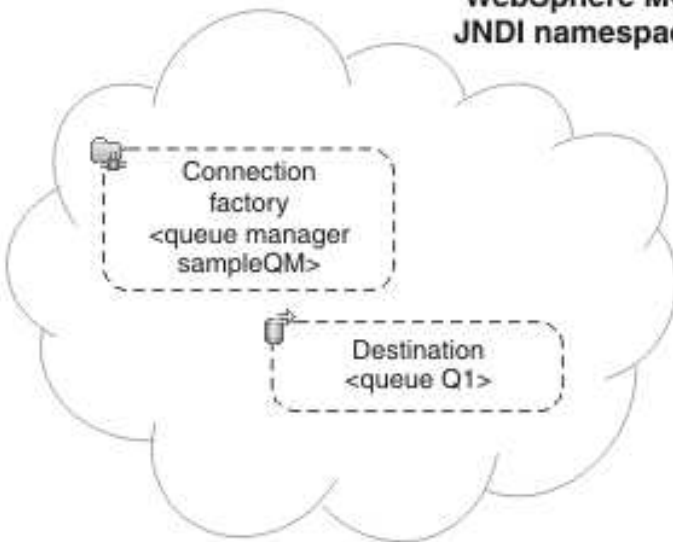
- Use of IBM MQ Explorer to create administered objects in the JNDI namespace. The JMS application can look up the administered objects to connect to IBM MQ and access MQ destinations with which to send or receive messages.

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

WebSphere MQ



WebSphere MQ JNDI namespace



Legend

< ... > Configuration property

Figure 6. Objects created in IBM MQ

Procedure

1. If not already started, Start IBM MQ Explorer by clicking **Start > All Programs > IBM MQ > MQ Explorer**.

2. Create a queue manager to use for the sample application.
 - a. Right-click **Queue Managers** and click **New > Queue Manager**. The Create Queue Manager wizard starts.
 - b. In the **Queue manager name** field type `sampleQM`.

What else might I do or be interested in?

You can choose a different name for the queue manager, but you must remember to use it in later configuration steps, in place of `sampleQM`.

Note: The name must have no more than 48 characters, from the following set:

- Uppercase or lowercase letters A-Z
- Numerics 0-9
- Period (.)
- Forward slash (/)
- Underscore (_)
- Percent sign (%)

Names are case-sensitive. Objects of the same type must have different names. For example, two queues cannot have the same name, but a queue manager and a queue can.

- c. In the **Dead-letter queue** field type `SYSTEM.DEAD.LETTER.QUEUE`. This field is the name of the dead-letter queue that is automatically created when you create the queue manager.

Why am I doing this?

A dead-letter queue stores messages that cannot be delivered to their correct destination, for example because the queue is full. All queue managers must have an associated dead-letter queue.

- d. Leave the other fields as default and click **Finish**, or if that is disabled, click **Next**. The **Finish** button is disabled if the port number conflicts with an existing queue manager, for example the queue manager that is created as part of the default configuration. You must continue through the wizard to change the default port number.
 - e. If you clicked **Next**, continue to accept the defaults and click **Next** on each page until you get to the final page of the wizard, when **Finish** becomes available. Change the specified port number, for example to 1414, and click **Finish**.

IBM MQ opens a window while the queue manager is created and started.

3. Add an initial context for the JNDI namespace then connect IBM MQ Explorer to that context.

Why am I doing this?

Before you can use IBM MQ Explorer to create and configure JMS administered objects, you must add an initial context to define the root of the JNDI namespace in which the administered objects are stored. Whenever you want to use IBM MQ Explorer to create or manage administered objects in the JNDI namespace, you must connect IBM MQ Explorer to the initial context of the JNDI namespace.

- a. In the MQ Explorer - Navigator pane, right-click **JMS Administered Objects**, then click **Add Initial Context**. This action displays the “Connection details” page.
- b. Under “Where is the JNDI namespace located?”, click **File system**.
- c. In the **Bindings directory** field, type C:\JNDI-Directory.

Why am I doing this?

This value matches the JNDI namespace location that is specified in the sample JMS application. If you must specify a different JNDI directory, you must modify the application to match.

If the directory does not exist on your system, the window displays the message Specified location does not exist or is not readable. Click **Browse** to open a file system window, go to Local Disk (C:), then click **Make New Folder** to create the JNDI-Directory folder. Click **OK**.

Click **Next**.

- d. On the User preferences page, leave the default settings.

Why am I doing this?

- **Context nickname:** The location of the JNDI namespace is used as the nickname to display the initial context in IBM MQ Explorer.
- **Connect immediately on finish:** This option connects IBM MQ Explorer to the JNDI namespace when you finish creating the initial context so that you can create administered objects immediately.
- **Automatically reconnect to context on startup:** This option is not selected because usually you do not need IBM MQ Explorer to automatically reconnect to the initial context every time that you close and reopen IBM MQ Explorer.

What else might I do or be interested in?

If you routinely use IBM MQ Explorer to create or manage administered objects in the JNDI namespace, you can select the **Automatically reconnect to context on startup** check box to cause IBM MQ Explorer to automatically reconnect to the initial context whenever IBM MQ Explorer is started. This option saves you having to manually connect IBM MQ Explorer to the initial context.

Click **Finish** to create and display the initial context.

4. Create a connection factory administered object.

Why am I doing this?

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

- a. In the MQ Explorer - Navigator pane, expand **JMS Administered Objects**, and then expand the initial context, labeled **file:/C:/JNDI-Directory/**.
- b. Right-click **Connection Factories**, then select **New > Connection Factory**. This action opens the New Connection Factory wizard
- c. In the **Name** field, type **myCF**

Why am I doing this?

The sample JMS application contains code that looks up a connection factory with the name **myCF**. If you must use a different name, you must modify the application to match. IBM MQ is used for the messaging provider because the sample application uses point-to-point messaging.

Click **Next**.

- d. Leave the type of connection factory as **Connection Factory** because this option is the most flexible for general JMS use.

Why am I doing this?

A domain-independent connection factory enables JMS applications to use both point-to-point messaging and publish/subscribe messaging, especially if you want the JMS application to perform both types of messaging under the same transaction.

What else might I do or be interested in?

If a JMS application is intended to use only point-to-point messaging or only publish/subscribe messaging, you can select the specific messaging domain when you create the connection factory and a domain-specific (queue or topic) connection factory is created.

- e. Leave the support for XA transactions as cleared.

Why am I doing this?

The sample application does not use XA-compliant transactions.

What else might I do or be interested in?

IBM MQ JMS supports XA-compliant transactions in bindings mode. If you want the sample application to use XA-compliant transactions, you must modify the sample application.

Click **Next**.

- f. Leave the transport as **Bindings**.

Why am I doing this?

The sample JMS application that uses the connection factory runs on the same computer as the queue manager, so can use Bindings mode transport. This option means that the JMS application connects directly to the queue manager, and offers a performance advantage over the alternative client mode.

Click **Next**, then **Next** again.

- g. On the Change properties page, select **Connection** from the menu on the left, then in the Connection pane select sampleQM as the **Base queue manager**.

Why am I doing this?

The base queue manager is the queue manager that the application connects to. If you want the application to be able to connect to more than one queue manager, leave this value blank.

- h. Click **Finish**. IBM MQ opens a window to show that the object was created successfully. Click **OK** to close the window.
5. Create a destination administered object.

Why am I doing this?

A destination administered object identifies the IBM 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 IBM MQ queue.

What else might I do or be interested in?

In publish/subscribe messaging, the destination identifies a topic rather than a queue.

- a. In the MQ Explorer - Navigator pane, expand **JMS Administered Objects**, and then expand the initial context, labeled **file:/C:/JNDI-Directory/**.
- b. Right-click **Destinations**, and then click **New > Destination**. The New Destination wizard is opened.
- c. In the **Name** field, type myQueue. Leave the **Type** as **Queue**.

Why am I doing this?

The sample JMS application contains code that looks up a destination with the name myQueue. The sample JMS application uses point-to-point messaging, so requires a destination of type queue. Destinations of type topic are used for publish/subscribe messaging.

- d. Select **Start wizard to create a matching MQ Queue**.

Why am I doing this?

The destination object needs a matching IBM MQ queue, and it is convenient to use IBM MQ Explorer to create both together. When you complete the New Destination wizard, the Create an MQ Queue wizard opens, with many of the destination details mapped to the IBM MQ queue.

Click **Next**.

Click **Next** again.

- e. On the "Change properties" page, click **Select** next to **Queue manager**. Select the **sampleQM** queue manager that you created earlier, and then click **OK**.
- f. Specify Q1 as the name of the IBM MQ queue.

What else might I do or be interested in?

You can choose a different name for the queue, but you must remember to use it in later configuration steps, in place of Q1.

Note: The name must have no more than 48 characters, from the following set:

- Uppercase or lowercase letters A-Z
- Numerics 0-9
- Period (.)
- Forward slash (/)
- Underscore (_)
- Percent sign (%)

Names are case-sensitive. Objects of the same type must have different names. For example, two queues cannot have the same name, but a queue manager and a queue can.

- g. Click **Finish**. A window is opened, which indicates the object is created successfully. Click **OK**. The Create an MQ Queue wizard starts.
If the wizard does not start, you might not have selected the **Start wizard to create a matching MQ Queue** check box in an earlier step. In the MQ Explorer - Navigator pane, expand the **sampleQM** queue manager, right-click **Queues**, then select **New > Local Queue**.
6. Create the IBM MQ queue Q1.

Why am I doing this?

The destination administered object that is created earlier represents an IBM MQ queue. This queue is where the JMS messages are stored.

- a. Click **Next** to accept the **sampleQM** queue manager that you specified earlier.
- b. Click **Next**.
- c. Click **Finish** to create the IBM MQ queue by using the information from the destination administered object that you created earlier. IBM MQ Explorer opens a window with the message that the object was created successfully.
- d. Click **OK**.

The new queue **Q1** is now visible in the **Queues** section under the queue manager.

Results


You created the IBM MQ objects that are required to use the sample JMS application.

What to do next

Follow the steps in “Configuring security for the sample queue manager” to configure or disable IBM MQ Security ready to run the sample JMS application.

Related information:

 [JMS connection factories](#)

 [Creating a destination](#)

Configuring security for the sample queue manager

For this scenario, you can choose either to configure IBM MQ security to a basic configuration level, or alternatively turn off IBM MQ security by using IBM MQ Explorer.

About this task

For an application to successfully use IBM MQ for message transfer, IBM MQ security must be handled. In this scenario, the sample application works successfully when either:

- IBM MQ security is turned off.
- IBM MQ security is configured to some basic level.

IBM MQ security can be disabled quickly and easily by using IBM MQ Explorer.

Important: While the sample applications in this scenario work successfully with security disabled, it is often not useful to disable security for other applications where message transfer is of higher importance. Likewise, the security configuration in this scenario is unlikely to be suitable for use in other systems. For more information on setting up IBM MQ security, see [Getting going without turning off MQ Security](#).

Disabling IBM MQ security

Disable IBM MQ security for this scenario by editing the properties of the sample queue manager.

About this task

You disable IBM MQ security for this scenario by editing the properties of the sample queue manager by using IBM MQ Explorer.

Procedure

1. If not already started, start IBM MQ Explorer by clicking **Start > All Programs > IBM MQ > MQ Explorer**.
2. In the **MQ Explorer - Navigator** pane, expand **IBM MQ > Queue Managers**, then click **sampleQM**.

3. Right-click the created queue manager **sampleQM** and then click **Properties...**. The Properties window opens.
4. Disable channel authentication records.
 - a. Click **Communication** on the left of the Properties window.
 - b. In the **Communication** settings, select **Disabled** from the **Channel Authentication Records** list.
 - c. Click **Apply**.
5. Disable connection authentication.
 - a. Click **Extended** on the left of the Properties window.
 - b. In the **Extended** settings, clear the **Connection Authentication** field so that it is empty.
 - c. Click **Apply**.
6. Click **OK** to save the changes and close the Properties window.
7. Right-click the **sampleQM** queue manager, then click **Security** and refresh each of the IBM MQ security options.

Results

IBM MQ security is now disabled ready to run the sample JMS application.

What to do next

Follow the instructions in “Running the sample JMS application” on page 27 to verify your configuration.

Configuring IBM MQ security

Configure IBM MQ security for this scenario by creating channel authentication records.

About this task

To configure IBM MQ security for this scenario, you create channel authentication records by using IBM MQ Explorer. Channel authentication records act as rules to define which connections are allowed to the specified IBM MQ queue manager.

Procedure

1. If not already started, start IBM MQ Explorer by clicking **Start > All Programs > IBM MQ > MQ Explorer**.
2. Ensure that Channel and Connection Authentication are turned on for **sampleQM**.
 - a. Right-click the created queue manager **sampleQM** and then click **Properties**. The Properties window opens.
 - b. In the **Communication** settings on the left of the window, select **Disabled** from the **Channel Authentication Records** list, if not already selected, and click **Apply**.
 - c. In the **Extended** settings on the left of the window, type `SYSTEM.DEFAULT.AUTHINFO.IDPWOS` into the **Connection Authentication** field, if not already there, and click **Apply**.
 - d. Click **OK** to save the changes and close the Properties window.
3. Remove the default IBM MQ channel authentication records.

- a. In the **MQ Explorer - Navigator** pane, expand **IBM MQ > Queue Managers > sampleQM > Channels**, then click **Channel Authentication Records**.
 - b. Ensure that are showing system objects by selecting the first option in the upper right of the window.
 - c. Right-click each of the default channel authentication records and click **Delete...** and then **Delete**. Delete each of the three channel authentication records.
4. Create a channel authentication record that allows clients on the local host to connect to the SYSTEM channels.
- a. In the **MQ Explorer - Navigator** pane, expand **IBM MQ > Queue Managers > sampleQM > Channels**, then click **Channel Authentication Records**.
 - b. Right-click **Channel Authentication Records**, then select **New > Channel Authentication Record**.
 - c. On the "Create a Channel Authentication Record" page, click **Next**, leaving the default option, **Allow access**, selected.
 - d. On the "Match part of the identity" page, select **Address**, then click **Next**.
 - e. On the "Matching the channels" page, type SYSTEM* into the **Channel profile** field, then click **Next**.
 - f. On the "Matching an address" page, type the local host address 127.0.0.1 in the **IP address or host name pattern** field, and then click **Next**.
 - g. On the "Authorization user ID" page, select the **Channel's user ID** option, then click **Next**.
 - h. On the "Authentication" page, click **Next**, leaving the default option, **As queue manager**, selected.
 - i. On the "Optional attributes" page, leave both of the fields blank, and then click **Next**.
 - j. On the "Summary" page, check through the settings and if the **Command preview** command matches the following, click **Finish**.

```
SET CHLAUTH('SYSTEM*') TYPE(ADDRESSMAP) ADDRESS('127.0.0.1') USERSRC(CHANNEL) ACTION(ADD)
```

5. Create a channel authentication record that blocks all clients from connecting to any channel.
- a. In the **MQ Explorer - Navigator** pane, expand **IBM MQ > Queue Managers > sampleQM > Channels**, then click **Channel Authentication Records**.
 - b. Right-click **Channel Authentication Records**, then select **New > Channel Authentication Record**.
 - c. On the "Create a Channel Authentication Record" page, select the **Block access** option and leave the **Warning mode** option cleared, and then click **Next**.
 - d. On the "Match part of the identity" page, select **Address**, then click **Next**.
 - e. On the "Address matching" page, select **At the channel**, and then click **Next**.
 - f. On the "Matching the channels" page, type * into the **Channel profile** field, then click **Next**.
 - g. On the "Matching an address" page, type * in the **IP address or host name pattern** field, and then click **Next**.
 - h. On the "Optional attributes" page, leave both of the fields blank, and then click **Next**.

- i. On the “Summary” page, check through the settings and if the **Command preview** command matches the following, click **Finish**.

```
SET CHLAUTH('*') TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(NOACCESS) WARN(NO) ACTION(ADD)
```

Note: Channel authentication records work together through allowing the more specific rule to override the others. So, in the example configuration for this scenario, this record is overridden by the previous that allows a client on the local host to connect to all of the SYSTEM channels. This basic IBM MQ security configuration blocks access to anything other than the local host, which is allowed to connect to the SYSTEM channels.

6. Right-click the **sampleQM** queue manager, then click **Security** and refresh each of the IBM MQ security options.

Results

Some basic IBM MQ Security is now configured ready to run the sample JMS application.

What to do next

Follow the instructions in “Running the sample JMS application” to verify your configuration.

Running the sample JMS application

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

Before you begin

Download the sample application package. Click the following link and save the file to your computer: [sampleJMSApp.zip](#) (in online information center), then extract the contents. The package contains a sample JMS application .jar file and batch files for running the application.

- The sample `sampleJMSApp.jar` file and the .cmd files must be in the same directory.
- The .cmd files use environment variables to set the class path for running the JMS application. When running the JMS application, if you see a Java `java.lang.NoClassDefFoundError`, you might need 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. The supplied batch files perform the following actions:

- `runresponder.cmd` opens a command prompt window in which the responder client starts then waits for a message.
- `runrequester.cmd` opens a separate command prompt window in which the requester client starts then sends a request message and receives a reply.

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

Procedure

1. Double-click the `runresponder.cmd` file. In the command prompt window, labeled Responder window, the responder client starts then waits for a message.

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

```
> Waiting for message.
```

2. Double-click the `runrequester.cmd` file. 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

In the command prompt window, labeled Requester window, the requester client shows the connection status, the message it sent, then 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 'BakedBeans - 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 . . .
```

In the Responder window, observe 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 'BakedBeans'
```

```
> 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 command windows verify that the requester and responder clients of the sample application can communicate with each other through IBM MQ.

What to do next

Follow the instructions in “Obtaining the IBM MQ resource adapter” to download the IBM MQ resource adapter.

Obtaining the IBM MQ resource adapter

Obtain and install the IBM MQ resource adapter in order to connect to IBM MQ from WebSphere Application Server Liberty.

About this task

To connect to IBM MQ from Liberty, the IBM MQ resource adapter must be used. Liberty does not contain the IBM MQ resource adapter, so you must obtain the resource adapter separately by downloading it from Fix Central and then install it in the selected location.

Before IBM MQ Version 9.0, the name of the file to be downloaded was in the format of *V.R.M.F-WS-MQ-Java-InstallRA.jar*, for example *8.0.0.6-WS-MQ-Java-InstallRA.jar*. From IBM MQ Version 9.0, the format of the file name is *V.R.M.F-IBM-MQ-Java-InstallRA.jar*, for example *9.0.0.0-IBM-MQ-Java-InstallRA.jar*. According to the version naming scheme for IBM MQ, the *V.R.M.F* acronym stands for *Version. Release. Modification. Fix.*, and therefore corresponds to the number that precedes the name of the file.

Procedure

1. Download the IBM MQ resource adapter from Fix Central:
 - a. Click Find Product then add the information for your IBM MQ installation to the following fields:
 - In the **Product Selector** field, type MQ then select **WebSphere MQ** from the displayed list.
 - In the **Installed Version** field, click the arrow then select the version number from the displayed list, for example **9.0.0.0**.
 - In the **Platform** field, click the arrow and select your platform, for example, **Windows 64-bit, x86**.Click **Continue**.
 - b. Make sure that **Browse for Fixes** is selected and, under **Additional Query options**, clear **Show fixes that apply to this version**, and select **Show fixes that get me to this version**, then click **Continue**. Fix Central searches for the available fixes for your selected product, version and platform, for example WebSphere, WebSphere MQ (9.0.0.0, Windows 64-bit, x86).
 - c. Find the resource adapter in the displayed list of available fixes. For example:

```
release level: 9.0.0.0-IBM-MQ-Install-Java-All
9.0.0.0 MQ Resource Adapter for use with Application Servers
```

- Then click the resource adapter file name and follow the download process.
2. After downloading the IBM MQ resource adapter, install it in your chosen location.
 - a. In a command prompt, go to the folder where the IBM MQ resource adapter is located, and then enter the command: `java -jar V.R.M.F-IBM-MQ-Java-InstallRA.jar` The following message is displayed:

Before you can use, extract, or install IBM MQ V9.0, you must accept the terms of 1. IBM International License Agreement for Evaluation of Programs 2. IBM International Program License Agreement and additional license information. Please read the following license agreements carefully.

The license agreement is separately viewable using the `--viewLicenseAgreement` option.

Press Enter to display the license terms now, or "x" to skip.

- b. Press Enter and Enter again to display the full license terms. At the end of the license this message is displayed:

By choosing the "I Agree" option below, you agree to the terms of the license agreement and non-IBM terms, if applicable. If you do not agree, select "I do not Agree".

Select [1] I Agree, or [2] I do not Agree:

- c. Press 1 to agree to the license agreement. The following message is displayed:

Enter directory for product files or leave blank to accept the default value.
The default target directory is H:\Liberty\WMQ

Target directory for product files?

- d. Enter the directory to install the files to, or press Enter without entering any value to install the default location as displayed. After the files are installed to the requested location, a confirmation message is displayed:

Extracting files to H:\Liberty\WMQ\wmq
Successfully extracted all product files.

Results

Within the selected directory, a new directory that is called `wmq` is created. Within this directory, the following files are installed:

```
wmq.jmsra.ivt.ear
wmq.jmsra.rar
```

What to do next

Follow the instructions in either "Option 1: Installing and configuring Liberty by using the command line" or "Option 2: Installing and configuring Liberty by using the graphical user interface" on page 38 to install and configure Liberty.

Option 1: Installing and configuring Liberty by using the command line

In Option 1, you install WebSphere Application Server Liberty by extracting the distribution image. You then configure the server by editing the server configuration file in a text editor.

Before you begin

This option assumes that you have finished your configuration of IBM MQ and you have got the IBM MQ resource adapter. Make sure that you have finished the following tasks:

- Create an initial IT configuration by “Creating a sample initial IT configuration” on page 14
- Obtain the IBM MQ resource adapter by “Obtaining the IBM MQ resource adapter” on page 29

Installing Liberty

Install WebSphere Application Server Liberty by extracting the distribution image. At the end of this task, you are ready to configure Liberty to communicate with IBM MQ.

Before you begin

Ensure that your system meets the operating system and Liberty requirements for using Liberty. See System Requirements for WebSphere Application Server - Liberty.

About this task

Install Liberty by extracting the archive file. The distribution image is packaged as a self-extracting archive file. Running the self-extracting archive file installs Liberty so that you are then ready to create a server.

Procedure

1. Obtain a copy of the archive file:
 - a. For the no-charge developer edition (with no IBM support), you can download the archive file from the WASdev community download page. Download both Liberty and the extended content for Liberty.
 - b. For all other editions of Liberty, you can use the archive file that is included with each edition. For detailed installation instructions, see Installing Liberty in the Liberty product documentation.
 - c. You can also download an edition-specific Liberty archive file, including the developer edition with IBM support, from Passport Advantage Online.
2. On the WASdev community download page, underneath Assets, click WAS Liberty with Java EE 7 Full Platform. Click **Download** on the following page.
3. Extract the downloaded contents into a directory of your choice.
4. Optional: Set the **JAVA_HOME** property for your environment. You can use the following command to set the **JAVA_HOME** property, and to add the Java /bin directory to the path:

```
set JAVA_HOME=C:\Program~1\Java\JDK16
set PATH=%JAVA_HOME%\bin;%PATH%
```

The Liberty profile runtime environment searches for the **java** command in this order: **JAVA_HOME** property, **JRE_HOME** property, and system **PATH** property.

What to do next

Follow the instructions in “Creating a Liberty profile server” to create a Liberty server to communicate with IBM MQ.

Creating a Liberty profile server

Create a WebSphere Application Server Liberty profile server in the command prompt.

Procedure

1. Open a command prompt and navigate to the `wlp\bin` in the directory where the distribution image was extracted.
2. Create a server that is called `SCENARIO`. In the command prompt, type `server create SCENARIO`.

Results

The server is created.

What to do next

Configure your server to connect to the IBM MQ messaging provider. See “Configuring Liberty”

Configuring Liberty

Configure the WebSphere Application Server Liberty server to communicate with IBM MQ.

Before you begin

- You must have the IBM MQ resource adapter. If not, follow the instruction in “Obtaining the IBM MQ resource adapter” on page 29.
- You must already have a Liberty profile server on which to deploy a messaging application that uses JMS. If not, follow the instructions in “Creating a Liberty profile server.”

About this task

To configure the server, edit the `wlp\usr\servers\SCENARIO\server.xml` file in a text editor. Alternatively, you can download the sample `server.xml` to the `wlp\usr\servers\SCENARIO` directory. Click the following link and save the file: `server.xml` `server.xml` (in the online product documentation). You must edit the location of the IBM MQ resource adapter and the IBM MQ native libraries.

Procedure

1. Add the features to the feature manager in the `server.xml` file. In the `server.xml` file, add the following features into the feature manager section.

```
<featureManager>
  <feature>jsp-2.3</feature>
  <feature>jndi-1.0</feature>
  <feature>mdb-3.2</feature>
  <feature>wmqJmsClient-2.0</feature>
  <feature>servlet-3.1</feature>
  <feature>osgiconsole-1.0</feature>
</featureManager>
```

Why am I doing this?

Adding the `wmqJmsClient-2.0` feature enables the Liberty server to load the necessary IBM MQ bundles that can be used to define the IBM MQ JMS resources. For example, the connection factory and activation specification properties provide client libraries to connect to the IBM MQ network.

2. If you decided to configure some IBM MQ security in “Configuring IBM MQ security” on page 25, add the following line and replace the username and password to match a valid IBM MQ Administrator account.

```
<authData id="auth1" user="username" password="password"/>
```

3. Specify the location of the IBM MQ resource adapter. To do so, replace the path to the `wmq.jmsra.rar` file with the appropriate one for your system. Add the following entry to the `server.xml` file:

```
<variable name="wmqJmsClient.rar.location" value="path_to\wmq\wmq.jmsra.rar"/>
```

Where the value attribute specifies the absolute path to the IBM MQ resource adapter file that was downloaded, `wmq.jmsra.rar`.

4. Add the connection factory definitions to the `server.xml` file. In the `server.xml` file, add the following entry:

```
<jmsQueueConnectionFactory jndiName="jms/replyCF" connectionManagerRef="ConMgr6" containerAuthDataRef="auth1">  
  <properties.wmqJms channel="SYSTEM.DEF.SVRCONN" port="1414" hostName="localhost"/>  
</jmsQueueConnectionFactory>
```

```
<connectionManager id="ConMgr6" maxPoolSize="2"/>
```

If you didn't configure IBM MQ security in “Configuring IBM MQ security” on page 25, remove the following part from the previous code block before you add it into your `server.xml` file.

```
containerAuthDataRef="auth1"
```

Note: If you are using another port rather than 1414 for your queue manager `sampleQM`, use the port number that is used. This port number is displayed in IBM MQ Explorer under **Queue managers** > `sampleQM` > **Listeners**, in the Port column in the LISTENER.TCP row.

Why am I doing this?

- The value of **jndiName** is 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.
- The **channel** is the name of the IBM MQ channel to use for communication. If you do not have a specific channel that you want to use, specify the default channel that is created when you created the queue manager, `SYSTEM.DEF.SVRCONN`.

5. Create a queue.

```
<jmsQueue id="MDBQ2">  
  <properties.wmqJms baseQueueName="Q1"/>  
</jmsQueue>
```

Why am I doing this?

The **baseQueueName** is the name of queue that is used by the destination *myQueue*.

6. Create an activation specification.

```
<jmsActivationSpec id="sampleMDB/pishopExampleReplyMDB/PiShopReplyMDB" authDataRef="auth1">  
  <properties.wmqJms destinationRef="MDBQ2" destinationType="javax.jms.Queue" port="1414"/>  
</jmsActivationSpec>
```

If you didn't configure IBM MQ security in “Configuring IBM MQ security” on page 25, remove the following part from the above code block before adding it into your `server.xml` file.

```
authDataRef="auth1"
```

Note: If you are using another port rather than 1414 for your queue manager `sampleQM`, use the port number that you use. This port number is displayed in IBM MQ Explorer under **Queue managers > sampleQM > Listeners**, in the Port column in the LISTENER.TCP row.

Why am I doing this?

- The **id** is the path of executable file in `sampleMDB`.
- The value of **destinationRef** must be the same as the value of **id** of **jmsQueue** you created.
- The JMS objects that are created in the Liberty server are separate from the JMS objects that are created in IBM MQ.

7. Configure JMS applications to connect in the BINDING mode.

To allow the JMS applications to connect by using the shared memories or in BINDING mode to IBM MQ, you must have both Liberty and IBM MQ deployed on the same server. To allow JMS applications to connect in BINDING mode, use the `<nativeLibraryPath>` element in the `server.xml` file to specify the location of the IBM MQ native libraries.

```
<wmqJmsClient nativeLibraryPath="C:\Program Files\IBM\MQ\java\lib"/>
```

Now that you've configured the necessary features for the Liberty server, save the `server.xml` file.

8. Install the utility features within the `server.xml`. Open a command prompt and navigate to the `wlp\bin` subdirectory where you extracted the contents of the download and run the following, pressing 1 to accept the license agreement. The server shall be stopped before performing this step.

```
installUtility install SCENARIO
```

What to do next

Follow the instructions in “Deploying the sample application to Liberty” to deploy the sample MDB application.

Deploying the sample application to Liberty

Deploy the sample message driven bean (MDB) application to verify that WebSphere Application Server Liberty is communicating with IBM MQ.

Before you begin

Download the sample application. Click the following link and save the file to the computer that hosts Liberty: `sampleMDB.ear` (in the online product documentation).

About this task

The sample MDB application, `sampleMDB.ear`, is designed to use the objects that you created earlier in Liberty. The MDB application uses these objects to send a message to IBM MQ, for receipt by the sample JMS application requester client that you used in “Running the sample JMS application” on page 27.

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

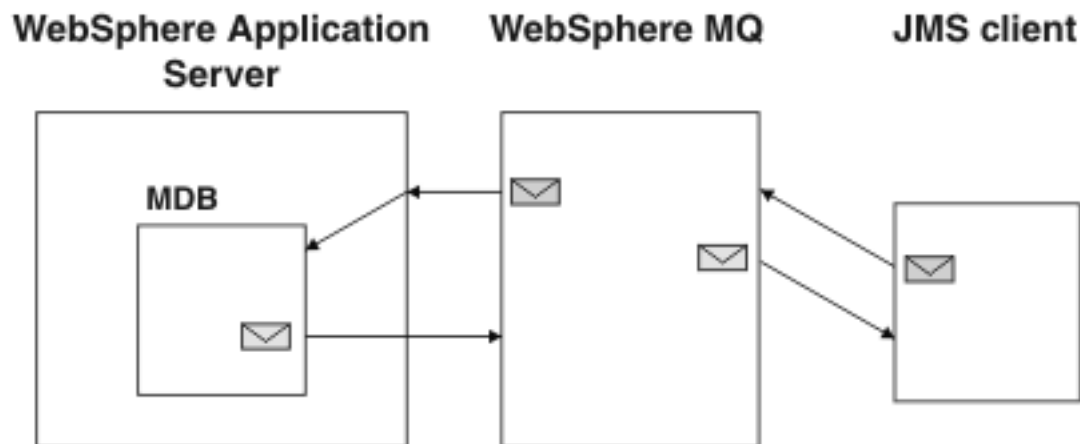


Figure 7. A message traveling from the sample JMS client to IBM MQ, then onward to the MDB on Liberty

Procedure

1. Add the downloaded `sampleMDB.ear` file to `${server.config.dir}\apps` folder. The default location for `${server.config.dir}` is `wlp\usr\servers\SCENARIO`.
2. Add the configuration of the `sampleMDB.ear` file to the `server.xml` by adding the following statement to the `server.xml` file.

```
<application type="ear" id="PiShopReplyMDB" name="PiShopReplyMDB" location="sampleMDB.ear" />
```

3. Save the `server.xml` file.

Results

The MDB application is deployed to Liberty.

What to do next

Follow the instructions in “Verifying the solution” on page 36 to run the sample applications and verify that the message is passed between the two products.

Verifying the solution

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

Before you begin

You must have configured WebSphere Application Server Liberty and IBM MQ as described in “Configuring Liberty” on page 32 and “Configuring the JNDI namespace and administered objects” on page 17.

About this task

In “Running the sample JMS application” on page 27, you ran the supplied JMS sample application to verify that the requester and responder clients of the application could communicate through IBM MQ. In “Deploying the sample application to Liberty” 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 previous responder application, verifying that messages are being passed between IBM MQ and Liberty.

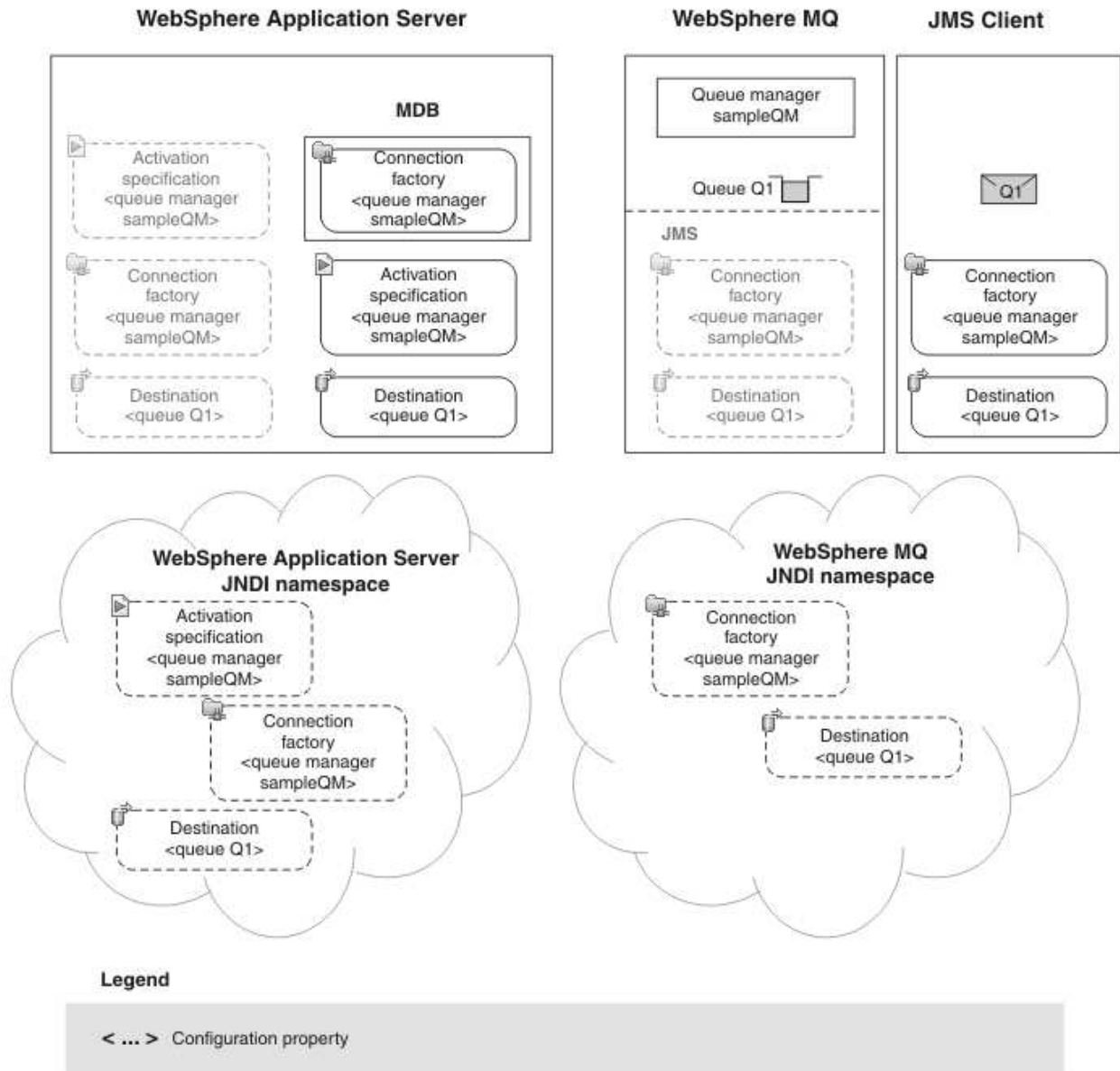


Figure 8.

Procedure

1. If Liberty is not already started, open a command prompt window, and navigate to `wlp\bin`, and then input the command `server start servername`.
2. Check the log file in `wlp\usr\servers\SCENARIO\logs\messages.log` for details. If the following error message is displayed, open IBM MQ Explorer and expand **IBM MQ > Queue Managers**, then right click **sampleQM**, and then click **Properties > Communication**. Select **Disabled** in **Channel authentication records** list.

JMSCMQ0001: IBM MQ call failed with compcode '2' ('MQCC_FAILED') reason '2035' ('MQRC_NOT_AUTHORIZED').

If exceptions are shown or the following error message is displayed in the log file:

The message endpoint for the PiShopReplyMDB message-driven bean cannot be activated because the sampleMDB/pishopExampleReplyMDB/PiShopReplyMDB activation specification is not available.

In this case, try the following things:

- Check the parameter names in your server.xml with the sample configuration file the sample configuration file (in online information center).
 - Check the attribute values in your server.xml to make sure they meet the requirements described in “Configuring Liberty” on page 32.
3. Double-click the runrequester.cmd batch file in the sample package. This command file opens a Command Prompt window then runs the sample JMS requester client, which sends a message to IBM MQ. The runrequester.cmd file uses environment variables to set the class path for running the JMS application. If you see a Java java.lang.NoClassDefFoundError, you might need to adjust the class path line in the command file.

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:414d512073616d706c65514d202020206beeb6512001ad02

> Received Message ID=ID:414d512073616d706c65514d202020206beeb6512001bb02 for 'From MDB: BakedBeans -
  15 tins in stock. Expected delivery time 1 day'

> 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 . . .
```

You have finished implementing the solution, and verified that Liberty can connect to IBM MQ for the successful transmission of messages.

Option 2: Installing and configuring Liberty by using the graphical user interface

In Option 2, install WebSphere Application Server Liberty by using the Installation Manager. Then configure the server by using the WebSphere Application Server developer tools.

Before you begin

You must have completed the configuration of IBM MQ. For instruction on how to do this, see “Creating a sample initial IT configuration” on page 14. You must have downloaded the IBM MQ resource adapter. For instructions on how to do this, see

“Obtaining the IBM MQ resource adapter” on page 29.

Installing Liberty

Install WebSphere Application Server Liberty by using the IBM Installation Manager.

Before you begin

Your system must meet the operating system and Java requirements for using Liberty. See System Requirements for WebSphere Application Server - Liberty.

About this task

Install the IBM Installation Manager. Then, install Liberty by using the IBM Installation Manager.

Procedure

1. If IBM Installation Manager is not already installed, install it.
 - a. Extract the Installation Manager archive file.
 - b. Switch to Installation Manager archive directory and double-click the `install.exe` file.
 - c. In the Install Packages window, select the version number, for example **Version 1.5.2**, and then click **Next**.
 - d. Click **I accept the terms in the license agreement**, and then click **Next**.
 - e. In **Installation Manager Directory** field, type the file path. The file path is the location where you want to install IBM Installation Manager.
 - f. Click **Next**, and then click **Install**. A progress bar is displayed, which shows that the installation is in progress.

When the installation is complete, a window is displayed, indicating that IBM Installation Manager is installed.

2. Click **Restart Installation Manager** in the lower right of the window. Alternatively, you can click **Start > All Programs > IBM Installation Manager > IBM Installation Manager** to start Installation Manager.
3. In the “IBM Installation Manager” window, click **File > Preference**, and then click **Add Repository**. In the “Add Repository” window, browse to the `repository.config` file of your Liberty repository folder. Then, click **OK** to confirm adding the repository.
4. In the Preference window, click **Test Connections**. If successful, the following message is displayed:

All the selected repositories are connected.

5. Click **OK** to close the Preference window.
6. In the “IBM Installation Manager” window, click **Install**.
7. Select the Liberty version number. To complete this scenario, you must use **Version 9.0.0** or later. Click **Next**.
8. Click **I accept the terms in the license agreement**, and then click **Next**.
9. Specify the installation root directory for the product binary files. If you use a Windows 64-bit operating system, choose whether you use the 32-bit or 64-bit architecture. Then, click **Next**.
10. Select **Embeddable EJB Container and JPA Client** and then click **Next**.

11. Choose the custom installation option and then open the Addons section of the menu. Click **Install** on the **Extended Programming Models**, and then click **Next**.
12. Review the summary information, and click **Install**.
 - a. If the installation is successful, the program displays a message, which indicates that installation is successful.

Note: The program might also display important post-installation instructions.
 - b. If the installation is not successful, click **View Log File** to troubleshoot the problem.
13. Click **Finish**.
14. Click **File > Exit** to close Installation Manager.

Results

Liberty is installed.

What to do next

Follow the instructions in “Installing WebSphere Application Server Developer Tools” to install the developer tools.

Installing WebSphere Application Server Developer Tools

Install WebSphere Application Server Developer Tools for Eclipse into an existing Eclipse workbench.

Before you begin

- Install Eclipse 4.2.2 for Java EE Developers.
- If you have an earlier version of WebSphere Application Server Developer Tools for Eclipse that is installed on your computer, you must uninstall it before you start this task.
- You must be connected to the internet.

Procedure

1. Start your Eclipse IDE for Java EE Developers workbench.
2. Locate the installation files from your Eclipse workbench.
 - a. Click **Help > Eclipse Marketplace**.
 - b. In the **Find** field, type Liberty Profile.
 - c. In the list of results, locate the **IBM WebSphere Application Server Liberty Profile Developer Tools Beta**, and then click **Install**. If the **Uninstall** button is displayed rather than the **Install** button, you have other version of WebSphere Application Server Developer Tools installed. In that case, click **Uninstall** to uninstall the other version, and then repeat Step 2.

What else might I do or be interested in?

Alternatively, you can install from WASdev website:

- a. Open your web browser to <https://ibmdw.net/wasdev/downloads/>, and click **Download** under **WebSphere Application Server Developer Tools for Eclipse**.
- b. Drag the **Install** icon onto the Eclipse toolbar.

3. On the **Confirm Selected Features** page, expand the parent nodes and select the features that you want to install. When you are finished, click **Next**.
4. On the “Review Licenses” page, click **I accept the terms of the license agreement**, and then click **Finish**. The installation process starts.

Note: During the installation, a Security Warning dialog box might open and display the following message:

Warning: You are installing software that contains unsigned content. The authenticity or validity of this software cannot be established. Do you want to continue with the installation?

You can safely ignore the message and click **OK** to continue.

5. When the installation process is completed, restart the workbench.

Results

The WebSphere Application Server Developer Tools are installed. You are ready to use the tools to create and configure a server on the liberty profile.

What to do next

Follow the instructions in “Creating a Liberty profile server” to create a Liberty profile server to communicate with IBM MQ.

Creating a Liberty profile server

Create a WebSphere Application Server Liberty profile by using the WebSphere Application Server Developer tools.

Procedure

1. Start your Eclipse IDE for Java EE Developers workbench.
2. In the Java EE view, click the **Servers** tab.
3. Right-click in the **Servers** field, and then click **New > Server**. The Define a New Server window is displayed.
4. Select **WebSphere Application Server V8.5 Liberty Profile** from the server list, and then click **Next**.
5. In the “Liberty Profile Runtime Environment” window, browse to the root of your Liberty profile installation folder. By default, the installation folder is `C:\Program Files\IBM\WebSphere\Liberty`.
6. If the JRE you want to use is not listed in the window, click **Configure JREs** and follow the instructions to configure it. Then, click **Next**.
7. In the “New Liberty Profile Server” window, enter a server name in the **Server name** field. For example, SCENARIO. Then, click **Finish**. The name of the server that you just created is displayed in the **Servers** field.

Results

A server is created in Liberty.

What to do next

Configure your server to connect to the IBM MQ messaging provider. See “Configuring Liberty”

Configuring Liberty

Through the IBM MQ messaging provider in WebSphere Application Server Liberty, Java Message Service (JMS) messaging applications can use your IBM MQ system as an external provider of JMS messaging resources.

Before you begin

- You must have the IBM MQ resource adapter. If not, follow “Obtaining the IBM MQ resource adapter” on page 29 to get it.
- You must already have a Liberty profile server on which to deploy a messaging application that uses JMS. If not, follow “Creating a Liberty profile server” on page 41 to create one.

About this task

You configure the server by using the developer tools for Eclipse.

Procedure

1. Start your Eclipse IDE for Java Developers workbench.
2. Click the **Servers** tab, and then expand the server that you created. Then, double-click **Server Configuration**. The server.xml file is displayed in the main window.
3. Click **Design** tab to switch to the design view.
4. Add features to the server configuration.
 - a. In the main window, expand **Server Configuration**, and then click **Feature Manager**.
 - b. Click **Add** in the right of the main window.
 - c. Add the `wmqJmsClient-2.0` feature. Enter `wmqJmsClient-2.0` in the “Select the features to enable” window, and then click **OK**.
 - d. Using the same method to add `jndi-1.0`, `osgiconsole-1.0`, `servlet-3.1`, `localConnector-1.0`, and `Mdb-3.2`.
5. Specify the location of the IBM MQ resource adapter.
 - a. Click **Server Configuration** in the main window, and then click **Add**.
 - b. Enter `Variable Declaration`, and then click **OK**.
 - c. In the **Name** field, enter `wmqJmsClient.rar.location`; in the **Value** field, enter the full path of your `wmq.jmsra.rar` file that is obtained in “Obtaining the IBM MQ resource adapter” on page 29.
6. Add the connection factory definitions.
 - a. Click **Server Configuration** in the main window, and then click **Add**.
 - b. Enter `JMS Queue Connection Factory`, and then click **OK**.
 - c. In the **JNDI name** field, enter `jms/replyCF`.
 - d. Click **New** next to **Connection manager reference**.

- e. Click **JMS Queue Connection Factory** again, and click **Add**.
- f. Enter IBM MQ JMS Queue Connection Factory, and then click **OK**.
- g. Enter the following information:

Channel

The name of the IBM MQ channel to use for communication.

If you do not have a specific channel that you want to use, specify the default channel that is created when you created the queue manager, SYSTEM.DEF.SVRCONN.

Tip: To display the default channel in IBM MQ Explorer, click the Show System Objects icon at the upper right of the MQ Explorer - Content pane.

Host name

The name of your machine.

Port The port number of the IBM MQ queue manager sampleQM.

This port number is displayed in IBM MQ Explorer under **Queue managers > sampleQM > Advanced > Listeners**, in the Port column in the LISTENER.TCP row.

7. Create a queue.
 - a. Click **Server Configuration** in the main window, and then click **Add**.
 - b. Enter **JMS Queue**, and then click **OK**.
 - c. In the **Id** field, enter MDBQ2; in the **JNDI name** field, enter MDBQ2.
 - d. Click **JMS Queue**, and then click **Add**.
 - e. Enter IBM MQ JMS Queue, and then click **OK**.
 - f. In the **Base queue name** field, enter Q1.
8. Create an activation specification.
 - a. Click **Server Configuration** in the main window, and then click **Add**.
 - b. Enter JMS Activation Specification, and then click **OK**.
 - c. In the **Id** field, enter sampleMDB/pishopExampleReplyMDB/PiShopReplyMDB.
 - d. Click **JMS Activation Specification**, and then click **Add**.
 - e. Enter IBM MQ JMS Activation Specification, and then click **OK**.
 - f. Enter the following information:

Destination

MDBQ2

Port The port number of the IBM MQ queue manager sampleQM.

This port number is displayed in IBM MQ Explorer under **Queue managers > sampleQM > Advanced > Listeners**, in the Port column in the LISTENER.TCP row.

9. Configure JMS applications to connect in the BINDING mode.
 - a. Click **Server Configuration** in the main window, and then click **Add**.
 - b. Enter IBM MQ Messaging, and then click **OK**.
 - c. In the **Native library path** field, enter the location of the IBM MQ native libraries. If you are using 32-bit operating system, the default path is C:\Program Files\IBM\MQ\java\lib; if you are using 64-bit operating system, the default path is C:\Program Files (x86)\IBM\MQ\java\lib.
10. Click **File > Save** to save the configurations.

What to do next

Follow the instructions in “Deploying the sample application to Liberty” to deploy the sample MDB application.

Deploying the sample application to Liberty

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

Before you begin

Download the sample application. Click the following link and save the file to the machine that hosts Liberty: `sampleMDB.earsampleMDB.ear` (in online information center).

About this task

The sample MDB application, `sampleMDB.ear`, is designed to use the objects that you created earlier in Liberty. The MDB application uses these objects to send a message to IBM MQ, for receipt by the sample JMS application requester client that you used in “Running the sample JMS application” on page 27.

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

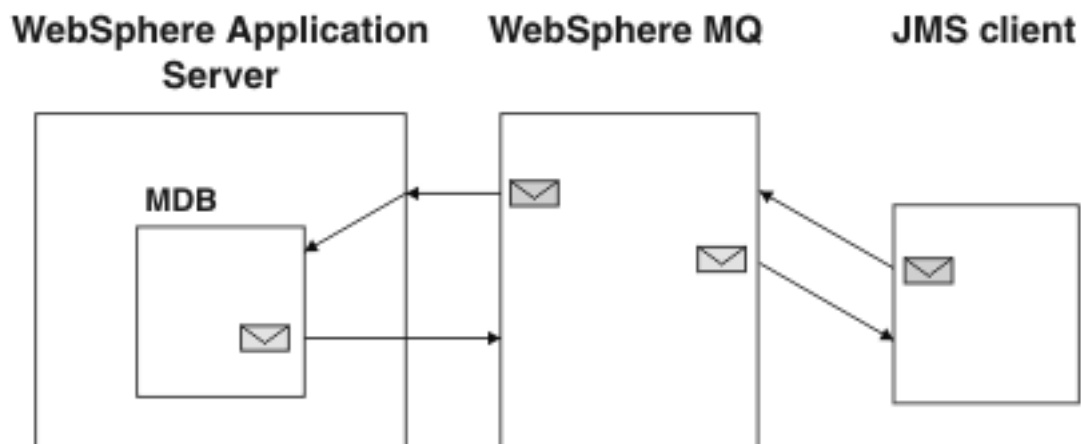


Figure 9. A message traveling from the sample JMS client to IBM MQ, then onward to the MDB on Liberty

Procedure

1. Copy the `sampleMDB.ear` file to `${server.config.dir}\apps` folder. The default location for `${server.config.dir}` is `C:\Program Files\IBM\WebSphere\Liberty\usr\servers\SCENARIO`.
2. Add the configuration of the `sampleMDB.ear` file to the `server.xml`.
 - a. Click **Server Configuration** in the main window, and then click **Add**.
 - b. Enter `Application`, and then click **OK**.
 - c. In the **Id** field, enter `PiShopReplyMDB`.

- d. Click **Browse** next to **Location** field.
 - e. Click **Relative Path** in the left panel and **sampleMDB.ear** on the right panel. Then, click **OK**.
 - f. Select **Automatically start**.
3. Click **File** > **Save** to save the configurations.

Results

The MDB application is deployed to Liberty.

What to do next

Follow the instructions in “Verifying the solution” to run the sample applications and verify that messages can be passed between the two products.

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 configured WebSphere Application Server Liberty profile and IBM MQ as described in “Configuring Liberty” on page 42 and “Configuring the JNDI namespace and administered objects” on page 17

About this task

In “Running the sample JMS application” on page 27, you ran the supplied JMS sample application to verify that the requester and responder clients of the application could communicate through IBM MQ. In “Deploying the sample application to Liberty” on page 44, 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 previous responder application, verifying that messages are being passed between IBM MQ and Liberty profile.

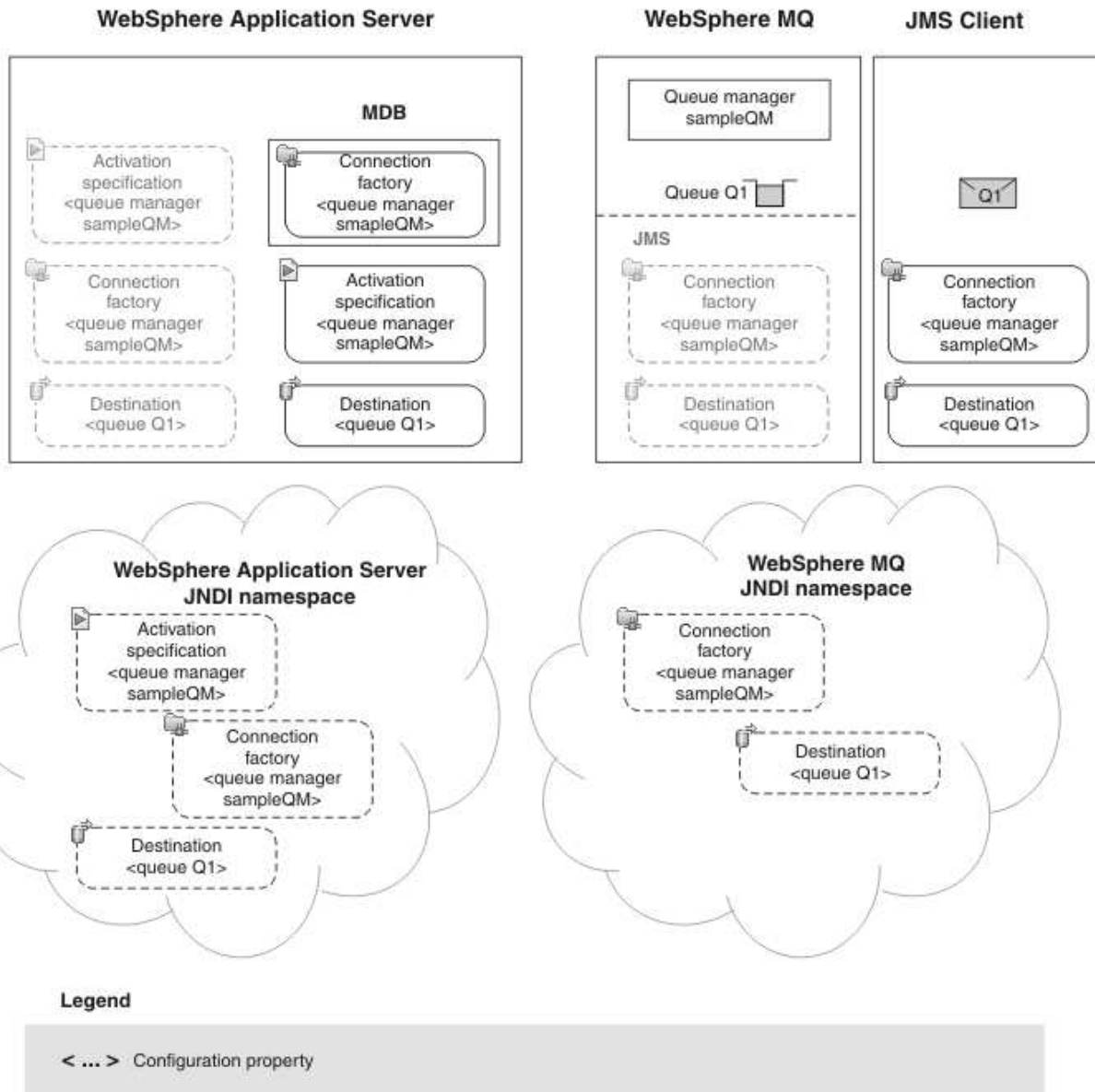


Figure 10.

Procedure

1. Start your Eclipse IDE for Java EE Developers workbench.
2. Start the server.
 - a. Click the **Servers** tab.
 - b. Right click the server name. and then click **start**.

The following message is displayed in the console:

```

Launching SCENARIO (WebSphere Application Server 8.5.5.0/wlp-1.0.3.20130510-0831)
on IBM J9 VM, version pwa6460sr13ifx-20130303_02 (SR13) (en_GB)
[AUDIT ] CWWKE0001I: The server SCENARIO has been launched.
[AUDIT ] CWWKZ0058I: Monitoring dropins for applications.
[AUDIT ] CWWKZ0001I: Application sampleMDB started in 0.383 seconds.
[AUDIT ] CWWKF0011I: The server SCENARIO is ready to run a smarter planet.

```

If the following message is displayed, click **HTTP Endpoint** and change the value of **Port** and **Secure port**, and then start the server again until the message is no longer displayed.

Several ports (9080,9443) required by WebSphere Application Server V9.0 Liberty Profile at localhost are already in use. The server might be running in another process, or a system process might be using the port.
To start this server you will need to stop the other process or change the port numbers.

If the following error message is displayed in the console, open IBM MQ Explorer and expand **IBM MQ > Queue Managers**, then right click **sampleQM**, and then click **Properties > Communication**. Select **Disabled in Channel authentication records** list.

JMSCMQ0001: IBM MQ call failed with compcode '2' ('MQCC_FAILED') reason '2035' ('MQRC_NOT_AUTHORIZED').

3. Double-click the runrequester.cmd batch file in the sample package. This command file opens a Command Prompt window then runs the sample JMS requester client, which sends a message to IBM MQ. The runrequester.cmd file uses environment variables to set the class path for running the JMS application. If you see a Java java.lang.NoClassDefFoundError, you might need to adjust the class path line in the command file.

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:414d512073616d706c65514d202020206beeb6512001ad02

> Received Message ID=ID:414d512073616d706c65514d202020206beeb6512001bb02 for 'From MDB: BakedBeans -
  15 tins in stock. Expected delivery time 1 day'

> 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 . . .
```

In the console of your Eclipse, the following message is displayed:

Example MDB invoked

You have finished implementing the solution, and verified that Liberty can connect to IBM MQ for the successful transmission of messages.

Verifying IBM MQ and Liberty with the additional sample application

Further configure the existing, or configure a new Liberty server and some additional IBM MQ objects to run an additional sample application.

About this task

After you have successfully configured IBM MQ and set up a Liberty server as described in this scenario, the you can run an additional, more detailed, sample application. This additional application simulates a more real world application for the connectivity of the two technologies.

This task takes you through the steps necessary to configure IBM MQ and a Liberty server for use with the additional sample application.

Preparing to run the additional sample application

Before running the additional sample application, you must first configure your Liberty server and IBM MQ installation to correctly run the application.

Before you begin

Download the additional sample application package. Click the following link and save the file to your computer:
MQLibertyConversionApp.zipMQLibertyConversionApp.zip (in online product documentation), then extract the contents. The package contains a sample Java web application .war file for running the application and a server.xml file for the sample Liberty server.

About this task

This stage of the task uses IBM MQ Explorer to configure the additional objects that are needed by the application within your IBM MQ installation. It also describes the steps to use a text editor to configure your Liberty server for the additional sample application to run on.

Procedure

1. If not already started, start IBM MQ Explorer by clicking **Start > All Programs > IBM MQ > MQ Explorer**.
2. Create a new local queue in the existing queue manager sampleQM called "Q2".
 - a. In the MQ Explorer - Navigator pane, expand the **sampleQM** queue manager, and right-click **Queues** and then click **New > Local Queue** The New Local Queue wizard starts.
 - b. Type Q2 in the **Name** field.

What else might I do or be interested in?

You can choose a different name for the queue, but you must remember to use it in later steps, in place of Q2.

Note: The name must have no more than 48 characters, from the following set:

- Uppercase or lowercase letters A-Z
- Numerics 0-9
- Period (.)
- Forward slash (/)
- Underscore (_)
- Percent sign (%)

Names are case-sensitive. Objects of the same type must have different names. For example, two queues cannot have the same name, but a queue manager and a queue can.

- c. Click **Finish**. A new local queue that is named Q2 is created.
3. Configure the downloaded server.xml file for use on your system.
 - a. Open the server.xml file that is extracted from the MQLibertyConversionApp.zip folder in a text editor.
 - b. Edit the server.xml file to suit how you handled IBM MQ security in the “Configuring security for the sample queue manager” on page 24.

If you chose to follow the configuration instructions in “Configuring IBM MQ security” on page 25, edit the following line of code that is found within the server.xml file to use a valid IBM MQ Administrator account username and password.

```
<authData id="auth1" user="username" password="password"/>
```

However, if, you followed “Disabling IBM MQ security” on page 24 to disable IBM MQ security, remove the previous line entirely. Also, remove **containerAuthDataRef="auth1"** and **authDataRef="auth1"** from the following code blocks.

```
<jmsConnectionFactory jndiName="jms/replyCF" connectionManagerRef="jms/replyCF" containerAuthDataRef="auth1">  
  <properties.wmqJms channel="SYSTEM.DEF.SVRCONN" connectionNameList="localhost(1414)" queueManager="sampleQM" transportType="tcp">  
</jmsConnectionFactory>  
<jmsActivationSpec id="MQLibertyConversionApp/ProcessPayment" authDataRef="auth1">  
  <properties.wmqJms destinationRef="jms/INQ" destinationType="javax.jms.Queue" channel="SYSTEM.DEF.SVRCONN" connectionNameList="localhost(1414)" queueManager="sampleQM" transportType="tcp">  
</jmsActivationSpec>
```

- c. Edit the resource adapter file path in the file.

In the following lines of code that is found within the server.xml file, replace the path to the wmq.jmsra.rar file with the appropriate one for your system.

```
<variable name="wmqJmsClient.rar.location" value="path_to\wmq\wmq.jmsra.rar"/>
```

- d. If you opted to use a different queue manager name instead of sampleQM, edit the file to connect to your queue manager.

In the following lines of code that is found within the server.xml file, replace any mention of sampleQM with your system's queue manager name.

```
<properties.wmqJms channel="SYSTEM.DEF.SVRCONN" connectionNameList="localhost(1414)" queueManager="sampleQM" transportType="tcp">
```

```
<properties.wmqJms baseQueueManagerName="sampleQM" baseQueueName="Q1" priority="QDEF" putAsyncAllowed="DISABLED"/>
<properties.wmqJms baseQueueManagerName="sampleQM" baseQueueName="Q2" priority="QDEF" putAsyncAllowed="DISABLED"/>
```

- e. If you opted to create your queue manager on a different port to 1414, edit the file to use your appropriate port number.

In the following lines of code that is found within the `server.xml` file, replace the mentions of port 1414 with the port number that is used by your system.

```
<properties.wmqJms channel="SYSTEM.DEF.SVRCONN" connectionNameList="localhost(1414)" queueManager="sampleQM" transportType=
<properties.wmqJms destinationRef="jms/INQ" destinationType="javax.jms.Queue" channel="SYSTEM.DEF.SVRCONN" connectionNameLi
```

The `server.xml` file is now be correctly configured for use in your system, so save the changes and close the text editor.

4. Replace the existing `server.xml` file that is stored at `wlp\usr\servers\SCENARIO\server.xml` with the newly edited `server.xml` file.

Alternatively if you do not want to edit the existing SCENARIO Liberty server that you created, you can create a new Liberty server by running the following command:

```
server create server name
```

You can then replace the `server.xml` file at `wlp\usr\servers\server_name\server.xml` with the newly edited `server.xml` file.

5. Copy the `MQLibertyConversionApp.war` file that is extracted from the `MQLibertyConversionApp.zip` folder into your chosen server in the `/apps` folder, which can be found at `wlp\usr\servers\server_name\apps\`.

Results

You now have a Liberty server that is configured with all of the IBM MQ objects that are necessary to run the additional sample application.

What to do next

Run the `MQLibertyConversionApp.war` file against the new Liberty server setup.

Running the additional sample application

After preparing your IBM MQ installation and your Liberty server, run the additional sample application on your Liberty server and send messages through the application to IBM MQ.

About this task

You can run the additional sample application on your Liberty server by using the native commands in the Liberty installation directories. This task outlines how to use these commands to verify that your configuration is successfully set up.

Procedure

1. Start the Liberty server that you configured as described in “Preparing to run the additional sample application” on page 48.
 - a. Open a command prompt window at `\wlp\bin\` where `\wlp\` is your Liberty installation directory.
 - b. Start your Liberty server by running the following command.

```
server start server name
```

2. Ensure that the log files did not report any unexpected errors when the server started up by opening the following log files.

- wlp\usr\servers\SCENARIO\logs\console.log
- wlp\usr\servers\SCENARIO\logs\messages.log

If no errors were raised and the Liberty server started successfully, the console.log file at \wlp\usr\servers\SCENARIO\logs\console.log outputs these messages.

```

Launching SCENARIO (WebSphere Application Server 17.0.0.1/wlp-1.0.16.c1170120170227-0220) on Java HotSpot(TM) 64-Bit Ser
[AUDIT ] CWWKE0001I: The server SCENARIO has been launched.
[AUDIT ] CWWKE0100I: This product is licensed for development, and limited production use. The full license terms can
[AUDIT ] CWWKZ0058I: Monitoring dropins for applications.
[AUDIT ] CWWKT0016I: Web application available (default_host): http://localhost:9081/MQLibertyConversionApp/
[AUDIT ] CWWKZ0001I: Application MQLibertyConversionApp started in 0.879 seconds.
[AUDIT ] CWWKF0012I: The server installed the following features: [jsp-2.3, ejbLite-3.2, servlet-3.1, wmqJmsClient-2.0
[AUDIT ] CWWKF0011I: The server SCENARIO is ready to run a smarter planet.

```

3. Verify that the application successfully connects to your IBM MQ installation through your Liberty server.
 - a. Open a web browser to <http://localhost:9081/MQLibertyConversionApp/>, and then click **Next**.
 - b. On the new page, enter a number into the field that is labeled **How many dollars would you like to exchange?** for the application to simulate a conversion on, and then click **Convert**. Upon successful connection to IBM MQ, the simulated console at the bottom of the page outputs a number of messages appropriate to your input value as shown in the following example:

Console Window - This part of the application will simulate what the console will be outputting as you go through the screen

Right now \$12345 will buy you ...

```

(0) £8325.37 ...
(1) £8289.61 ...
(2) £8110.71 ...
(3) £8570.78 ...
(4) £8401.58 ...
(5) £8303.09 ...
(6) £8156.30 ...
(7) £8246.49 ...
(8) £8494.74 ...
(9) £8174.34 ...
(10) £8160.80 ...
(11) £8633.10 ...
(12) £8493.38 ...
(13) £8564.00 ...
(14) £8361.18 ...

```

>> End of Quotes <<

Results

You verified that the additional sample application runs on your Liberty server and successfully connects to your IBM MQ installation.

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