

CHAPTER 8: CONNECT TO A MESSAGING PROVIDER

Theory

WebSphere Application Server supports asynchronous messaging that allows to create, send, receive and read requests. Asynchronous messaging support is based on Java Message Service (JMS) and Java EE Connector Architecture (JCA).

Synchronous messaging requires both applications to be available at the same time as in the example of phone talk. On the other hand, asynchronous messaging doesn't require sending and receiving applications available at the same time. We can use postal service as an example.

Java Message Service (JMS) is a Java API to access to message providers. JMS provider or in other words messaging provider gives the basis for the exchange of messages between applications.

A JMS message consist of three parts;

- Header, must exist in every JMS message and it is assigned automatically mostly by JMS provider when the message put to a JMS destination.
- Properties, are optional and can be application related, provider related and standard properties. Properties provide an efficient mechanism to filter application defined messages.
- Body, contains the main information which needs to be exchanged.

There are 6 different types of JMS messages based on the data it contains:

- Message, it is the base class and used for event notification.
- BytesMessage, stores the data as sequence of bytes and used for exchanging data in a format that is native to the application.
- TextMessage, stores the data as a string and used for simple text message exchanging.
- StreamMessage, stores the data as a sequence of primitive Java types.
- MapMessage, stores the data as a set of key-value pairs and used for delivering keyed data that can change in different messages.
- ObjectMessage, stores a serialized Java object and used for Java object exchanging.

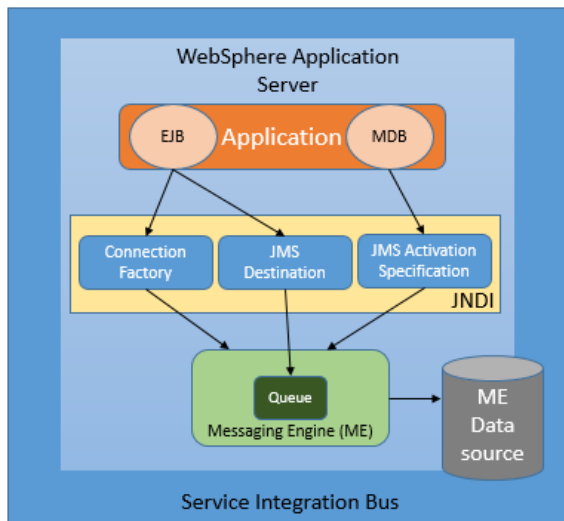
XML (Extensible Markup Language) is mostly used format in Java applications. It allows you to describe the data independent of the language that the application is written in or the platform which the application runs on.

A service integration bus is a group of servers or clusters that cooperate to provide asynchronous messaging service. An application server or a cluster can be a member of a bus. With service integration bus

- Any application can exchange messages with any application using destination.
- A message producing application can produce messages for a destination regardless of the messaging engine that the producer uses to connect to the bus.
- A message consuming application can consume messages from a destination regardless of which messaging engine the consumer uses to connect to the bus.

A connection factory is an object that is used by JMS client to establish the connection to a JMS provider. It is stored and managed object in a JNDI namespace. A connection factory can create connections to one messaging provider only. If you need to connect to a different provider, you need to create a new connection factory.

A JMS destination is an object that can be a JMS queue or a JMS topic. It represents the target of messages the client produces and the source of messages that the client consumes.



JMS activation specification is the standard way to manage the relation between Message-driven beans (MDB) which are a special class of Enterprise Java Beans (EJB) and a messaging engine.

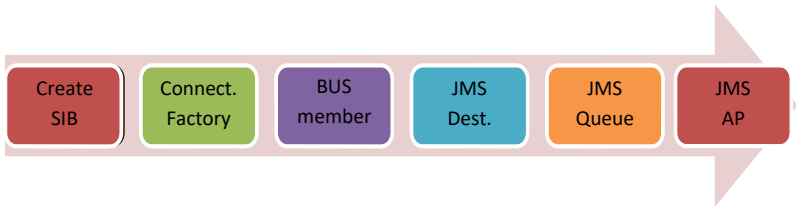
AIM

In this lab exercise, you will configure default messaging provider in WebSphere Application Server to enable the communication between applications that run on the application server and the other applications.

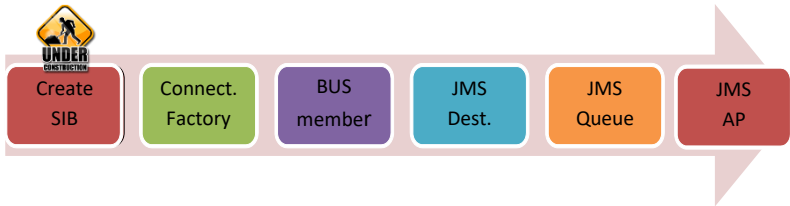
In order to achieve this, you need to complete following steps:

1. Create a Service Integration Bus
2. Configure JMS connection factory
3. Add a Bus member
4. Configure JMS destination
5. Configure JMS queue
6. Configure JMS Activation Specifications

Lab Exercise 8: CONNECT TO A MESSAGING PROVIDER

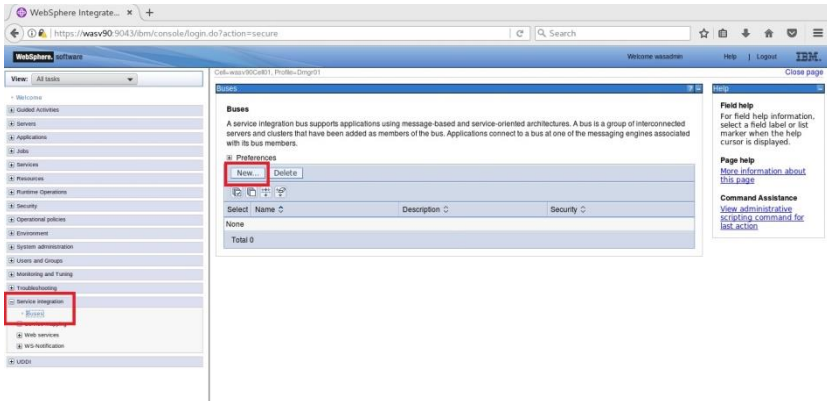


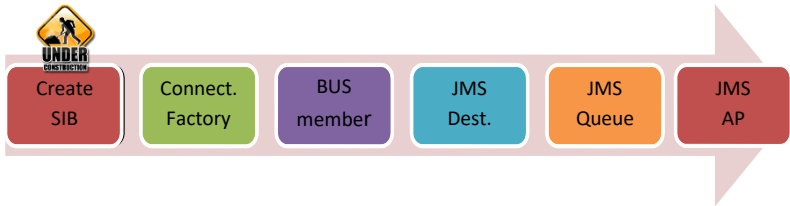
1. **Create a Service Integration Bus**
2. **Configure JMS connection factory**
3. **Add a Bus member**
4. **Configure JMS destination**
5. **Configure JMS queue**
6. **Configure JMS Activation Specifications**



Task 1: Create a Service Integration Bus

Step 1: Navigate to “Service integration>Buses” and then click on “New” to add a new bus.



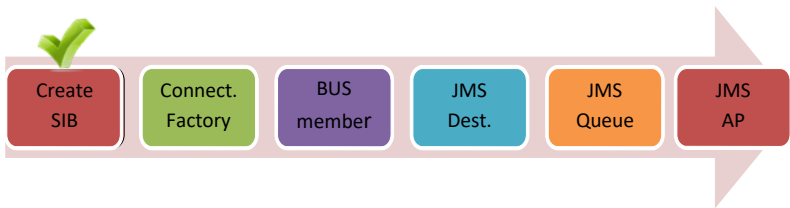


Step 2: You need to enter the name for the new bus, eg. “FC_SampleBus” and then click “Next”.

WebSphere Integration Bus console screenshot showing the 'Create a new Service Integration Bus' wizard. The wizard is at Step 1: Create a new bus. The 'Create a new bus' section shows the name 'WAS_BUS' entered in the 'Enter the name for your new bus' field. The 'Next' button is highlighted with a red box.

Step 3: Review the summary of actions and click “Finish”.

WebSphere Integration Bus console screenshot showing the 'Create a new Service Integration Bus' wizard. The wizard is at Step 2: Confirm create of new bus. The 'Confirm create of new bus' section shows a summary of actions: 'New bus "WAS_BUS" will be created with bus security setting "Disabled"'. The 'Finish' button is highlighted with a red box.



Step 4: Click on “Save” to write the changes directly to the master configuration.

Messages

Changes have been made to your local configuration. You can:

- Save directly to the master configuration.
- Review changes before saving or discarding.

An option to synchronize the configuration across multiple nodes after saving can be enabled in [Preferences](#).

The server may need to be restarted for these changes to take effect.

Buses

A service integration bus supports applications using message-based and service-oriented architectures. A bus is a group of interconnected servers and clusters that have been added as members of the bus. Applications connect to a bus at one of the messaging engines associated with its bus members.

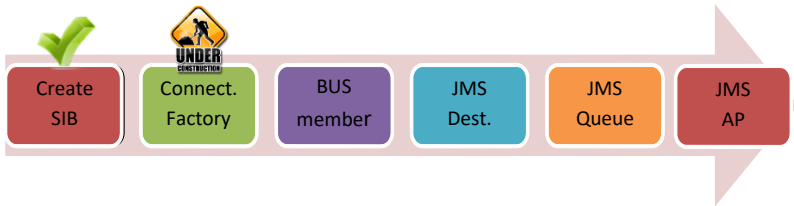
Preferences

New Delete

Select	Name	Description	Security
<input type="checkbox"/>	WAS_BUS		Disabled

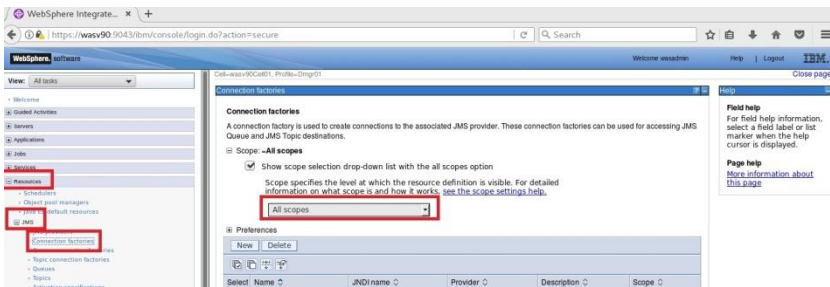
Total 1

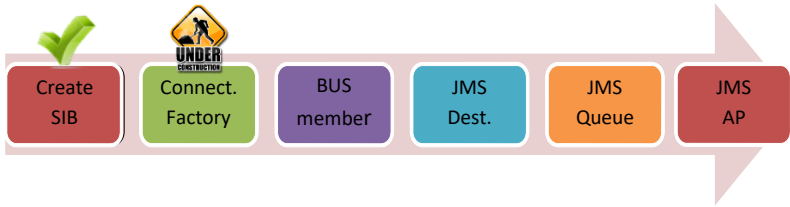
Task 1 is complete!



Task 2: Create JMS Factory

Step 1: Navigate to “Resources>JMS>Connection factories” and then select the scope you want to add a connection factory.

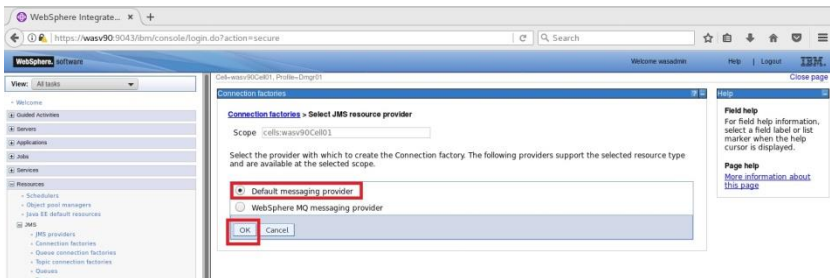


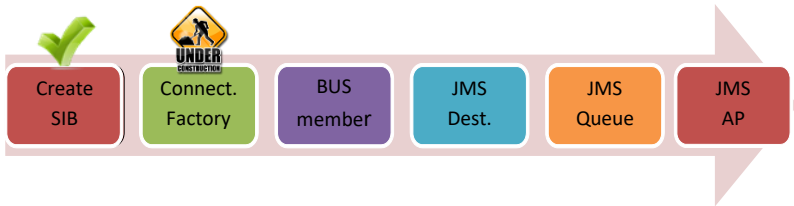


Step 2: In this example, we choose the cell as scope and then click “Next”.



Step 3: Select “Default messaging provider” and then click “OK”. You can choose “WebSphere MQ messaging provider” for WebSphere MQ configuration.





Step 4: You need to define name and JNDI name for the default messaging provider connection factory.

Name: WASV90JMSCF

JNDI name: jms/WASV90JMSCF

WebSphere Integrat... x +

https://wasv90.9043/bm/console/login.do?action=secure

WebSphere software

View: All tasks

Navigation pane:

- Home
- System
- Applications
- Jobs
- Services
- Resources
 - Schedulers
 - Object pool managers
 - Java EE default resources
 - JMS
 - JMS providers
 - Connection factories
 - Queue connection factories
 - Queues
 - Topics
 - Activation specifications
 - JDBC
 - Resource Adapters
 - Concurrency
 - Cache instances
 - Mail
 - URL
 - Resource Environment

Connection factories

Connection factories > Default messaging provider > New...

A JMS connection factory is used to create connections to the associated JMS provider of JMS destinations, for both point-to-point and publish/subscribe messaging. Use connection factory administrative objects to manage JMS connection factories for the default messaging provider.

Configuration

General Properties

Administration

Scope

Cell=wasv90Cell01

Provider

Default messaging provider

Name

WASV90JMSCF

JNDI name

.jms/WASV90JMSCF

Description

The additional properties will not be available until the general properties for this item are applied or saved.

Additional Properties

- Connection pool properties

Related Items

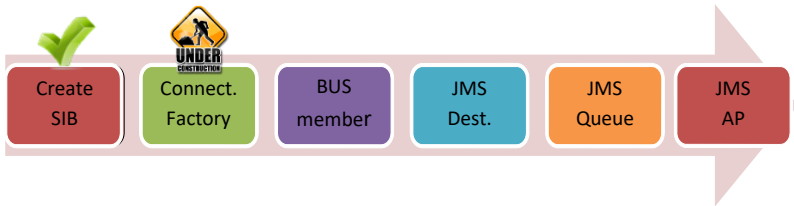
- JMS - J2C authentication data
- Buses

Field help

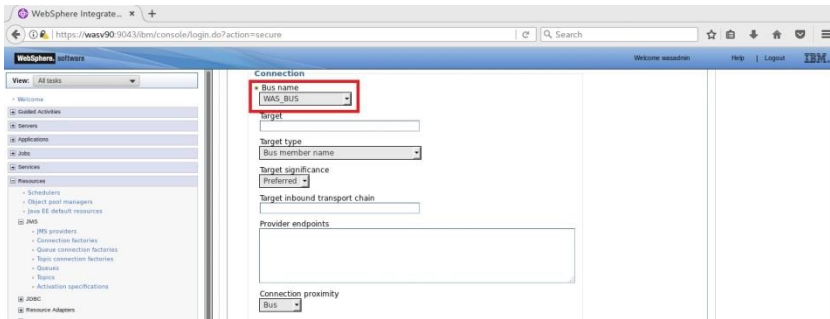
For field help information, select a field label or list marker when the help cursor is displayed.

Page help

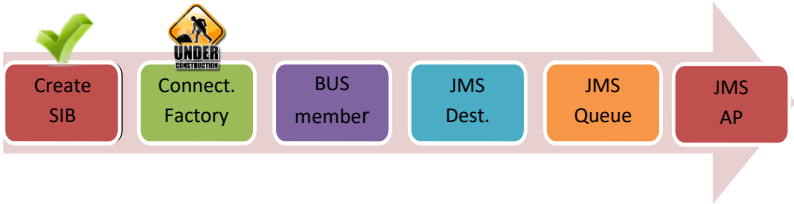
More information about this page



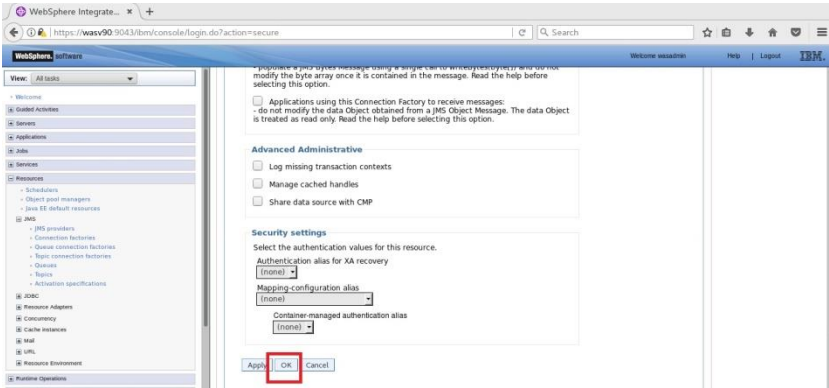
Scroll down to select “Bus name” that we added in the first task.

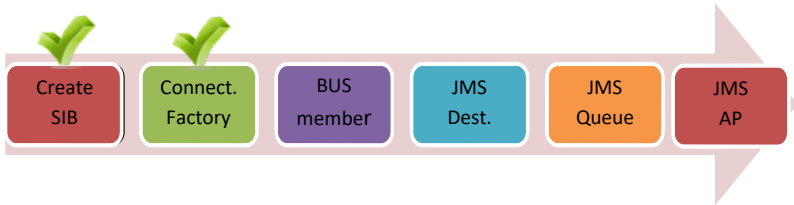


Scrolls

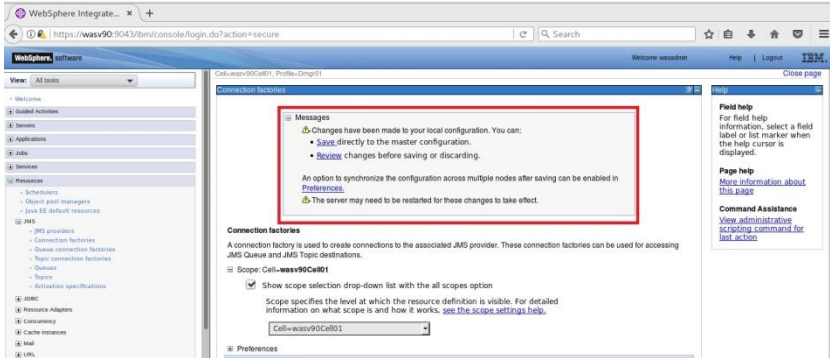


Scroll down and click “OK”.

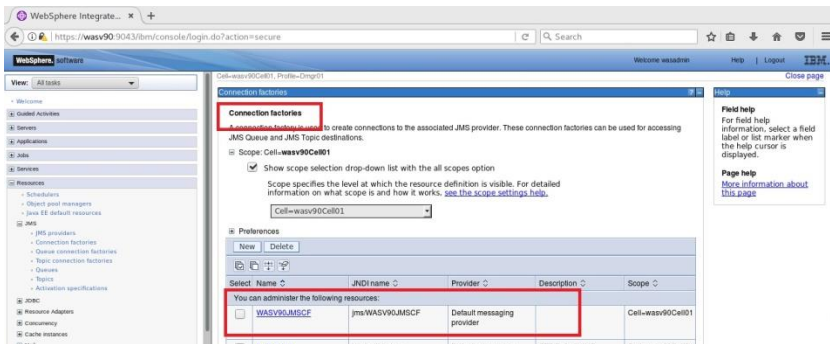




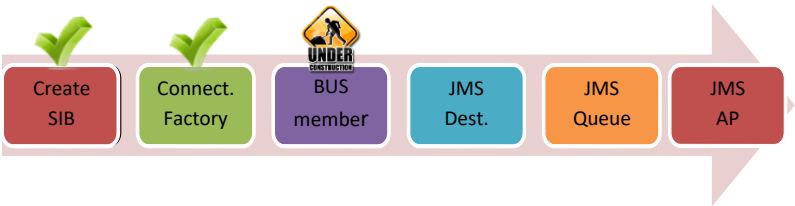
Step 5: Click on “Save” to write changes directly to master configuration.



You should see the newly added connection factory.

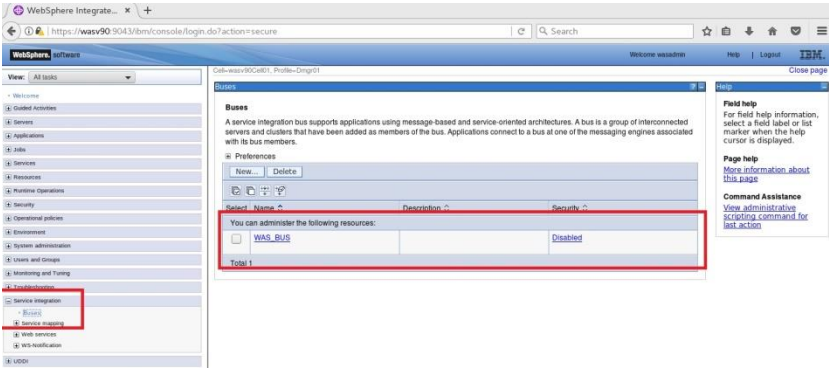


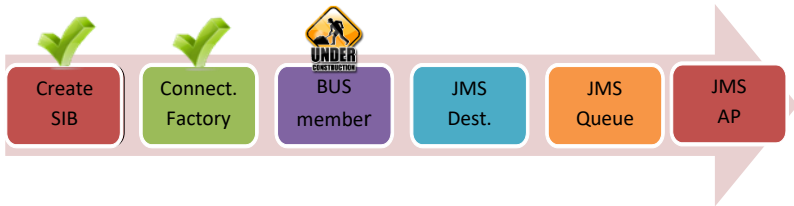
Task 2 is complete!



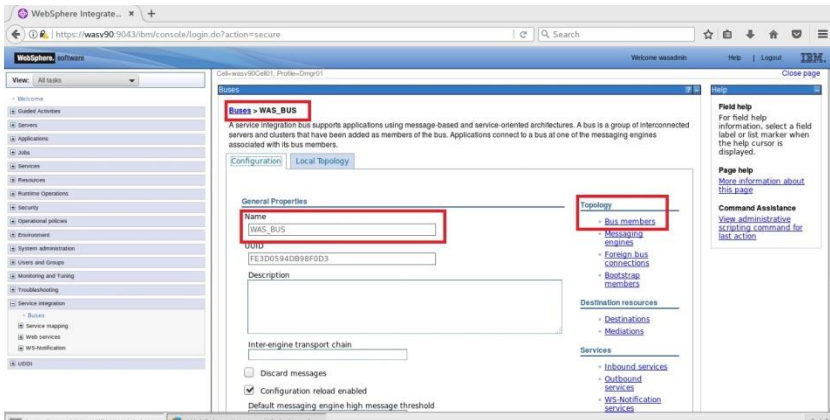
Task 3: Add a Bus member

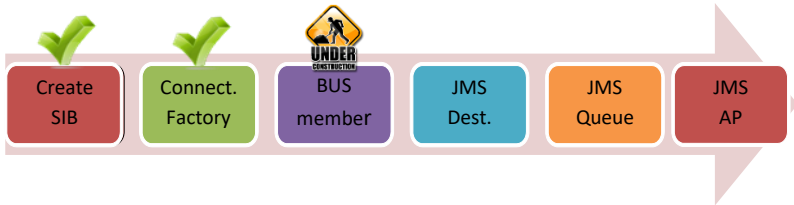
Step 1: Navigate to “Service integration>Buses” and click on the bus name (WAS_Bus) to change parameters.



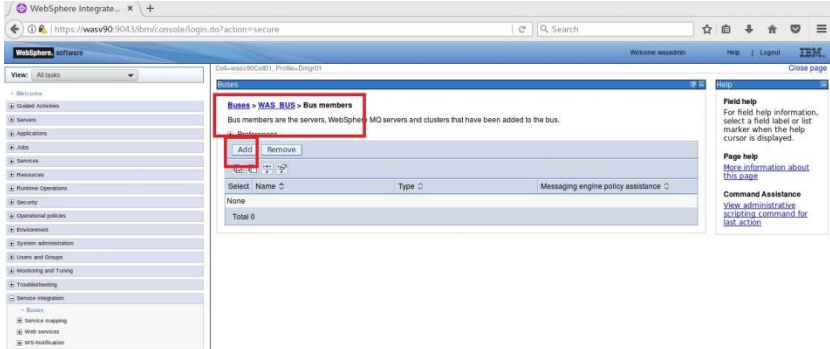


Step 2: Click on “Bus members” under “Topology” menu located on the right.

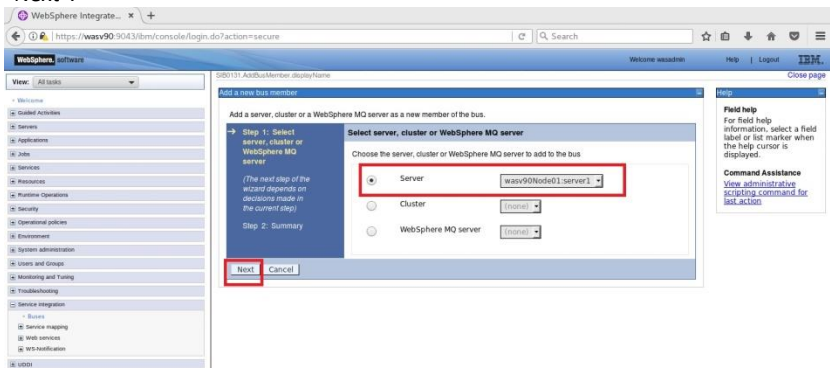


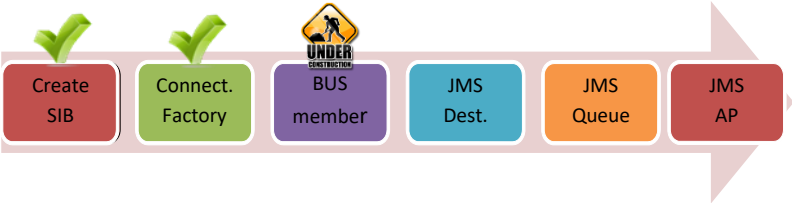


Step 3: Click on “Add” to define a member to the bus.

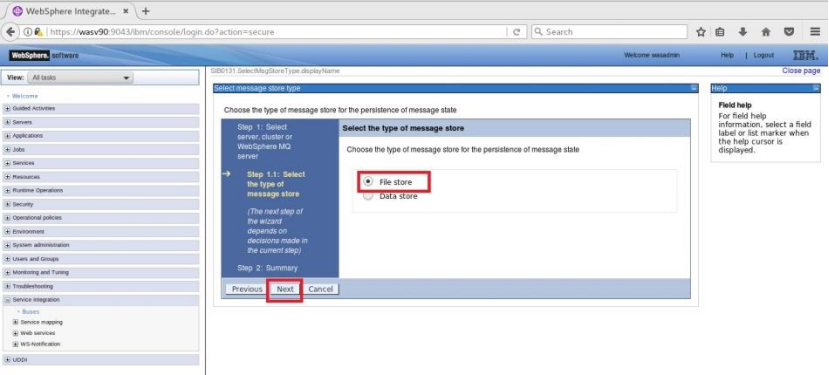


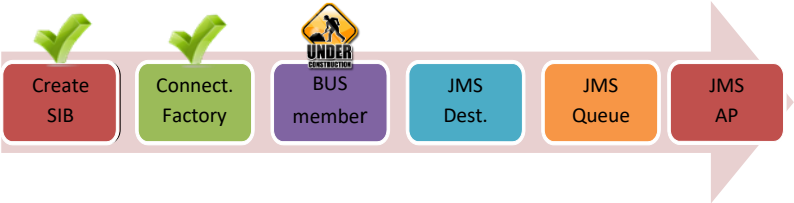
Step 4: Select “Server” or “Cluster” according to your needs and then click on “Next”.





Step 5: Select “File store” and then click “Next”.





Step 6: Configure the properties for the file store and then click “Next”.

WebSphere software console

View: All tasks

- Home
- Guided Activities
- Services
- Applications
- Jobs
- Resources
- Monitoring Operations
- Security
- Operational policies
- Environment
- System administration
- Users and Groups
- Monitoring and Tuning
- Toolchest/Tooling
- Service integration
 - Business
 - Service mapping
 - Web services
 - WS-Notification
- UDDI

Step 1: Select server, cluster or WebSphere MQ server

Step 1.1: Select the type of message store

Step 1.2: Configure file store

Step 2: Summary

Configure file store

Specify the properties for the file store

Log

- Log size: 100 MB
- ☒ Default log directory path
- ☐ Log directory path

Store

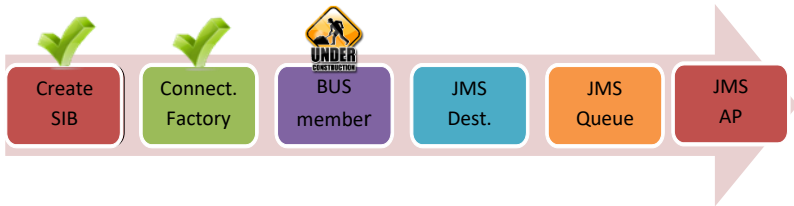
- ☒ Same settings for permanent and temporary stores

Permanent and temporary stores

- ☒ Minimum permanent store size: 500 MB
- ☐ Unlimited permanent store size
- ☒ Maximum permanent store size: 500 MB
- ☒ Default permanent store directory path
- ☐ Permanent store directory path

Previous Next Cancel

Field help: For field help information, select a field label or list marker when the help cursor is displayed.



Step 7: You can tune the application server for messaging performance selecting the option “Change heap sizes” and then click “Next”.

Tune application server for messaging performance.

Tune performance parameters

To improve performance of messaging within the application server, the proposed Java Virtual Machine settings are advised. By default the initial and maximum JVM settings will remain unchanged, select the “Change heap sizes” checkbox to modify the settings to the proposed values. On machines with low amounts of physical memory size or large numbers of application server instances, it may be necessary to reduce the proposed values accordingly.

☒ Change heap sizes

Current heap sizes Proposed heap sizes

Initial JVM heap size 0 MB 768 MB

Maximum JVM heap size 0 MB 768 MB

Previous **Next** Cancel

Step 8: Review the summary of actions and then click “Finish”.

Summary

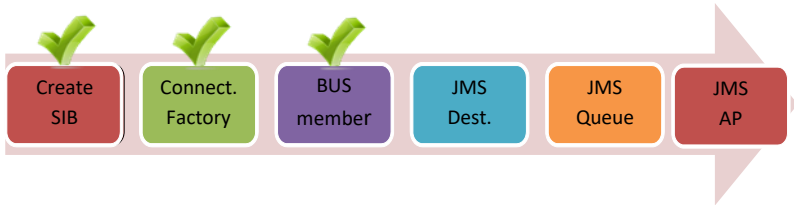
The actions that will be performed when selecting “Finish”:

Adding server “was90Node01.server1” as member of bus “WAS_BUS”.

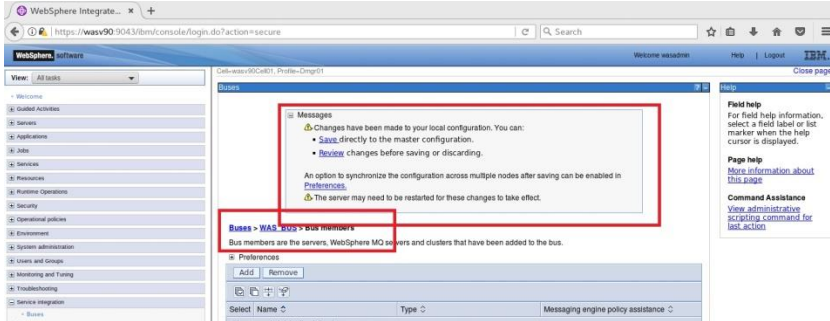
File store settings:

- Log size “100”
- Log directory path “Using default value”
- Minimum permanent store size “200”
- Maximum permanent store size “500”
- Permanent store directory path “Using default value”
- Minimum temporary store size “200”
- Maximum temporary store size “500”
- Temporary store directory path “Using default value”

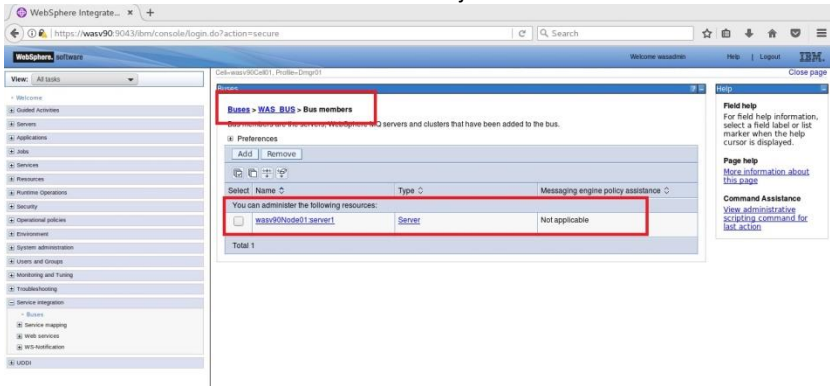
Previous **Finish** Cancel



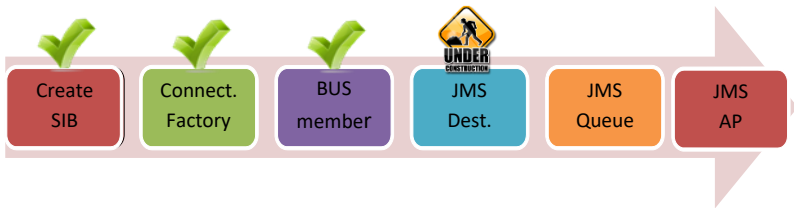
Step 9: Click on “Save” to write changes to the master configuration.



You should be able to see the bus member we just added as follows.



Task 3 is complete!



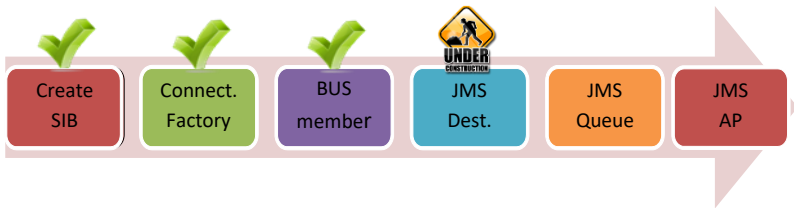
Task 4: Configure JMS Destination

Step 1: Navigate to “Service integration>Buses” and click on the bus name.

WebSphere Integration Developer console screenshot showing the 'Buses' page. The left sidebar shows the navigation tree with 'Buses' selected under 'Service integration'. The main content area shows a table of buses with one entry, 'WAS_BUS', which is highlighted with a red box. The table has columns for 'Name', 'Description', and 'Security'. The 'WAS_BUS' entry has a description of 'A service integration bus supports applications using message-based and service-oriented architectures...' and a security status of 'Disabled'.

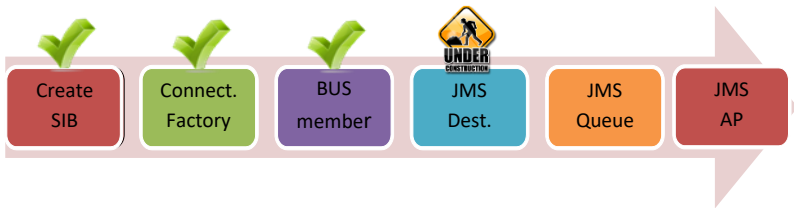
Name	Description	Security
WAS_BUS	A service integration bus supports applications using message-based and service-oriented architectures. A bus is a group of interconnected servers and clusters that have been added as members of the bus. Applications connect to a bus at one of the messaging engines associated with its bus members.	Disabled

Total 1

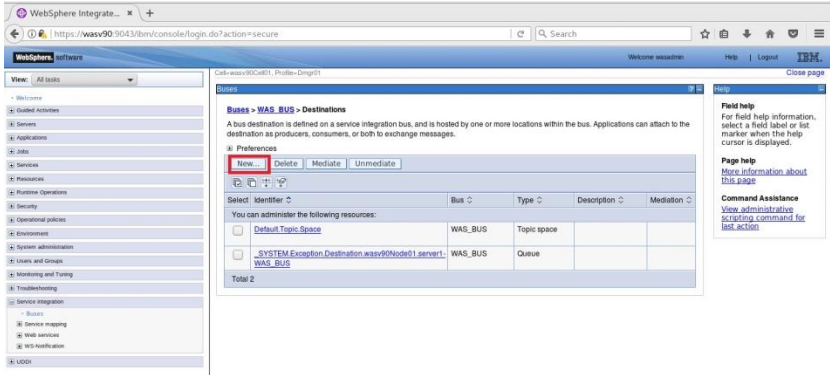


Step 2: Click on “Destinations” under “Destination resources” menu located on the right.

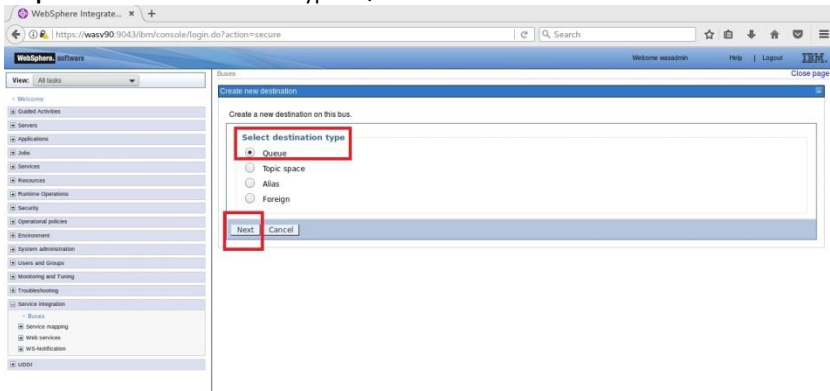
WebSphere Integration Developer console screenshot showing the configuration page for 'WAS_BUS'. The 'Configuration' tab is active, and the 'Local Topology' sub-tab is selected. The 'General Properties' section shows the bus name 'WAS_BUS' and UUID 'FE3D0594DB96F0D3'. The 'Topology' section lists various resources, including 'Destination resources' which is highlighted with a red box, indicating the next step in the process.

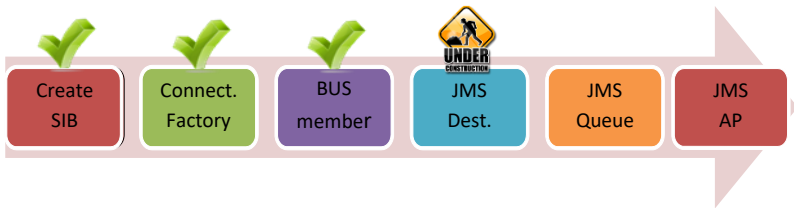


Step 3: Click on “New” to add a new destination.



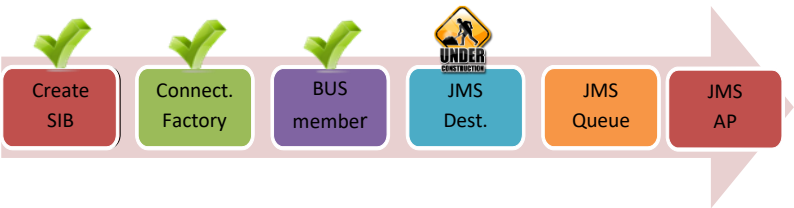
Step 4: Select the destination type “Queue” and click “Next”.



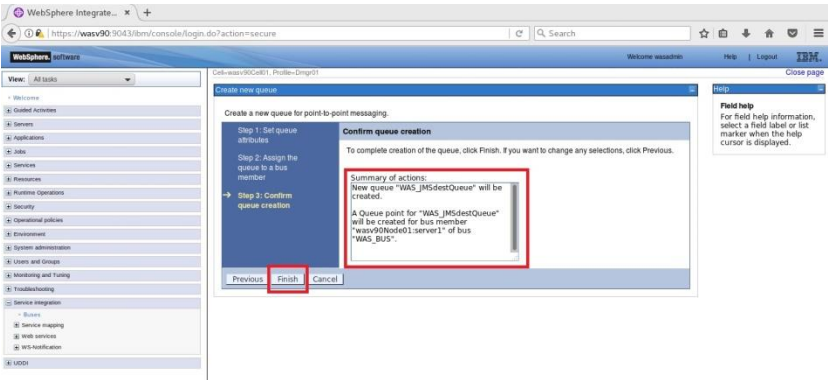


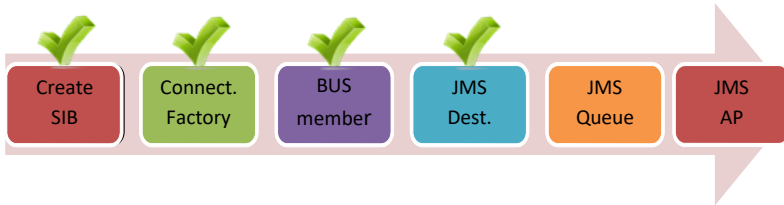
Step 5: Enter an identifier for the new queue and click “Next”.
(WAS_JMSdestQueue)

Step 6: Click on “Next”.



Step 7: Review the summary of actions and click “Finish”.





Step 8: Click on “Save” to write changes directly to the master configuration.

Messages

Changes have been made to your local configuration. You can:

- Save directly to the master configuration.
- Revert changes before saving or discarding.

An option to synchronize the configuration across multiple nodes after saving can be enabled in Preferences.

The server may need to be restarted for these changes to take effect.

Rules > WAS_BUS > Destinations

A bus destination is defined on a service integration bus, and is hosted by one or more locations within the bus. Applications can attach to the destination as producers, consumers, or both to exchange messages.

Preferences

Select	Identifier	Bus	Type	Description	Mediation
<input type="checkbox"/>	Default Topic Space	WAS_BUS	Topic space		
<input type="checkbox"/>	WAS_MQSeriesQueue	WAS_BUS	Queue		
<input type="checkbox"/>	SYSTEM Exception Destination wasv0Node01 server1 WAS_BUS	WAS_BUS	Queue		

Total 3

You should see the newly added destination queue listed.

Rules > WAS_BUS > Destinations

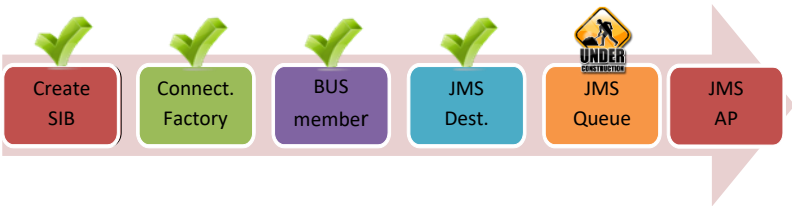
A bus destination is defined on a service integration bus, and is hosted by one or more locations within the bus. Applications can attach to the destination as producers, consumers, or both to exchange messages.

Preferences

Select	Identifier	Bus	Type	Description	Mediation
<input type="checkbox"/>	Default Topic Space	WAS_BUS	Topic space		
<input type="checkbox"/>	WAS_MQSeriesQueue	WAS_BUS	Queue		
<input type="checkbox"/>	SYSTEM Exception Destination wasv0Node01 server1 WAS_BUS	WAS_BUS	Queue		

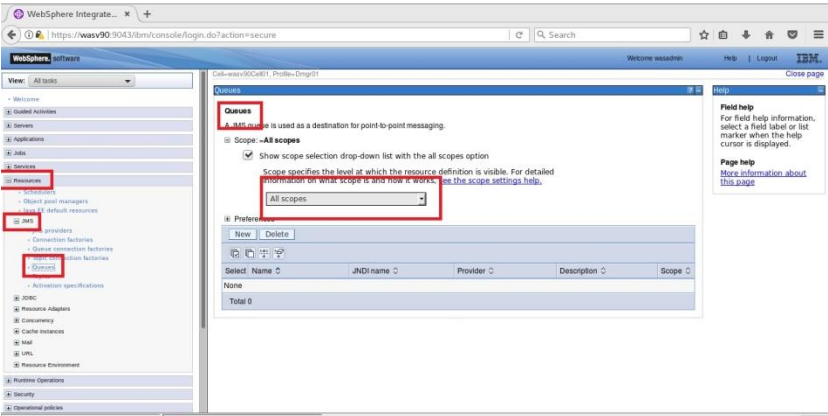
Total 3

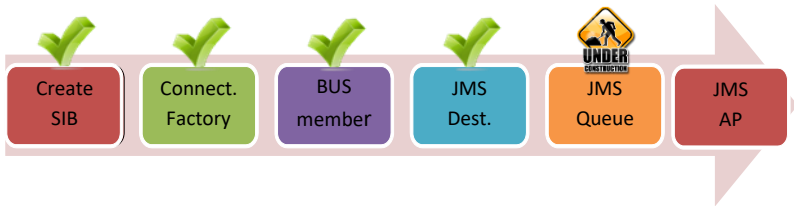
Task 4 is complete!



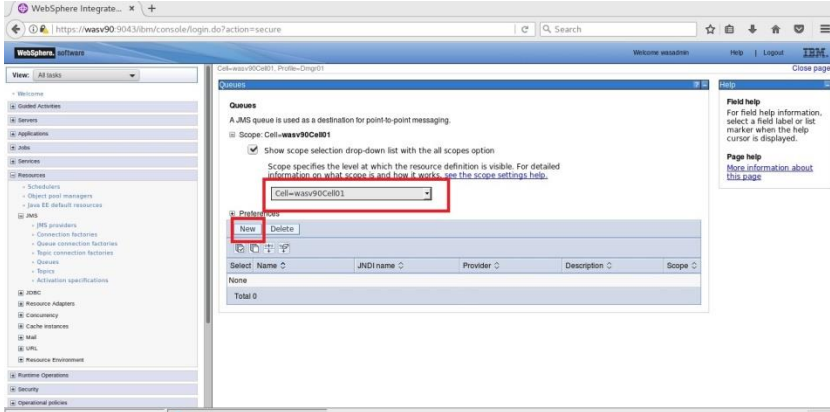
Task 5: Configure JMS queue

Step 1: Navigate to “Resources>JMS>Queues” and select the scope of the definition.

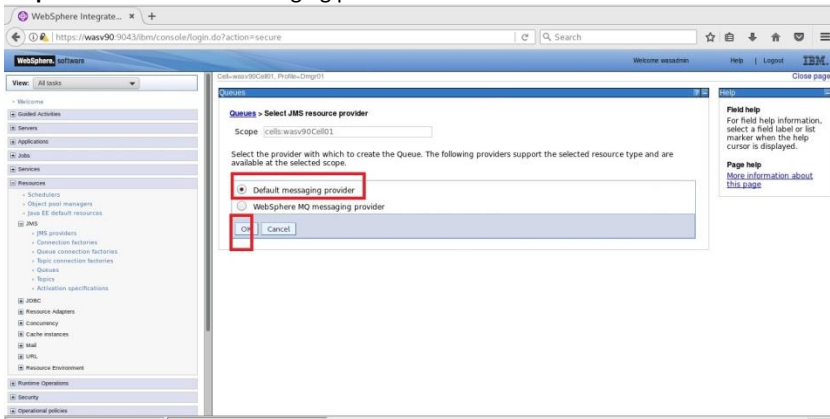


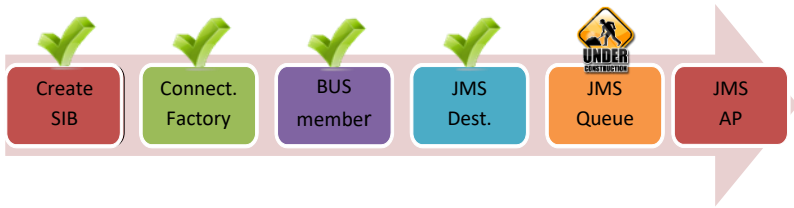


Step 2: Click “New” to start.



Step 3: Select “Default messaging provider” and then click “OK”.

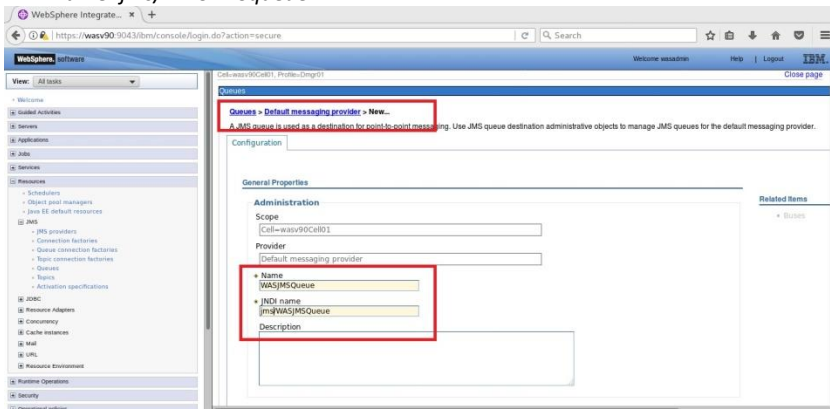


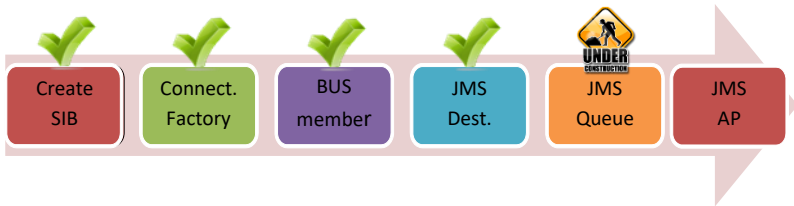


Step 4: Enter “Name” and “JNDI name”.

Name: WASJMSQueue

JNDI name: jms/WASJMSQueue





Select the “Bus name” and “Queue name” values we added in previous tasks.

WebSphere Integrated Solutions Console - Mozilla Firefox

View: All tasks

Connection

Bus name: FC_SampleBus

Queue name: FC_JMSdestQueue

Delivery mode: Application

Time to live: milliseconds

Priority:

Advanced

Read ahead: Inherit from connection factory

Message control across multiple queue points (supported from WebSphere Application Server 7.0)

☐ Restrict messages to the local queue point if a queue point is configured on the connected messaging endpoint

Control across multiple queue points per MessageProducer

Local queue point preference

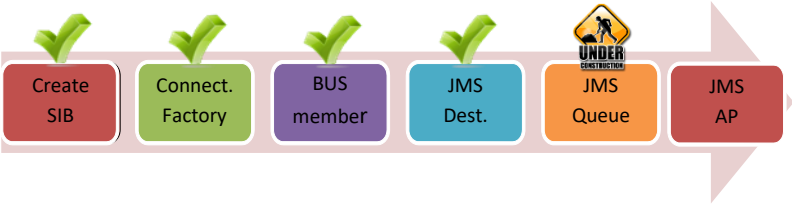
☒ Prefer to send messages to a local queue point

☐ Do not prefer a local queue point over other queue points

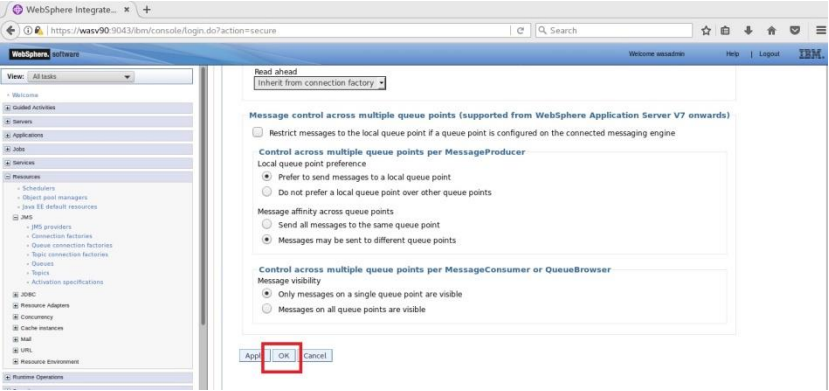
Message affinity across queue points

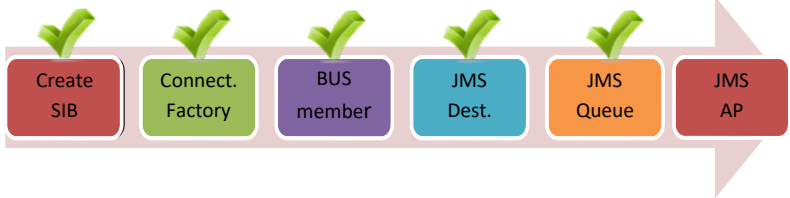
☐ Send all messages to the same queue point

☒ Messages may be sent to different queue points

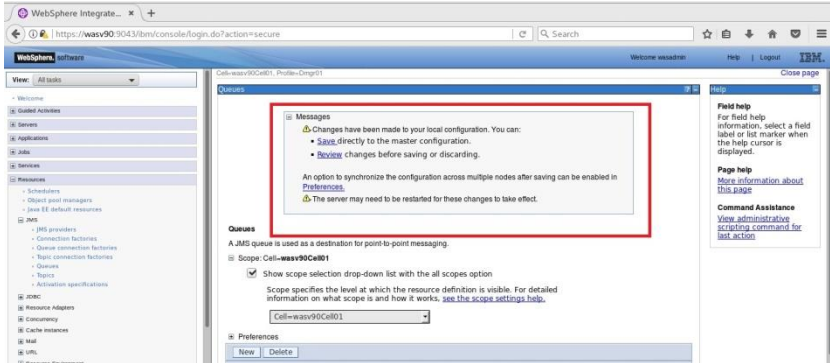


Click on “OK” to continue.

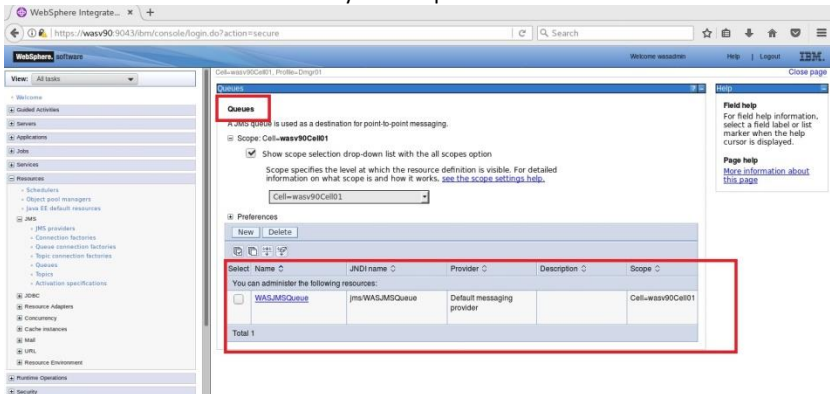




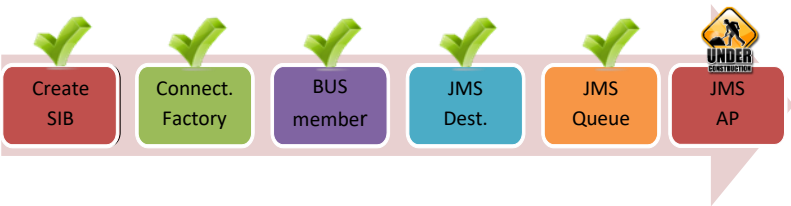
Step 5: Click “Save” to write changes to master file.



You should be able to see the newly added queue.

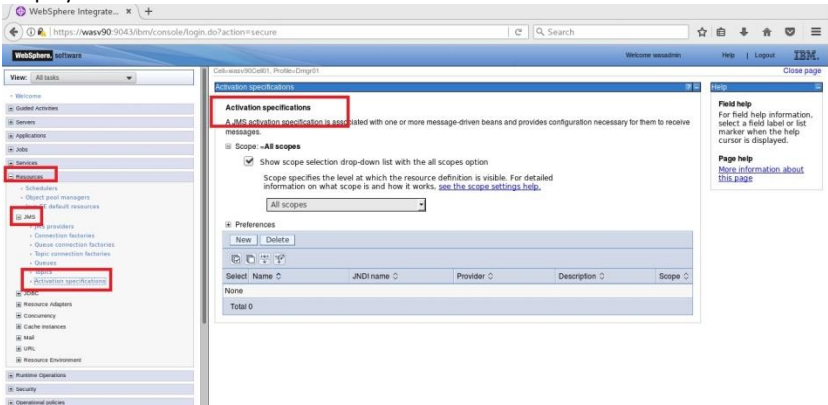


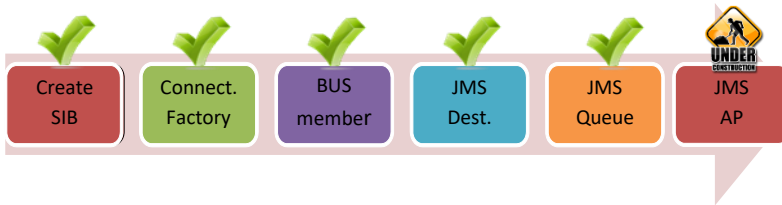
Task 5 is complete!



Task 6: Configure JMS Activation Specifications

Step 1: Navigate to “Resources>JMS>Activation specifications” and select your scope you want to define.





Step 2: Click on “New” to start definition.

Activation specifications

A JMS activation specification is associated with one or more message-driven beans and provides configuration necessary for them to receive messages.

Scope: **Cell=wasv90Cell01**

Show scope selection drop-down list with the all scopes option

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, see the [scope settings help](#).

Name **Delete**

Select	Name	JNDI name	Provider	Description	Scope
None					
Total: 0					

Step 3: Select “Default messaging provider” and click “OK”.

Activation specifications > Select JMS resource provider

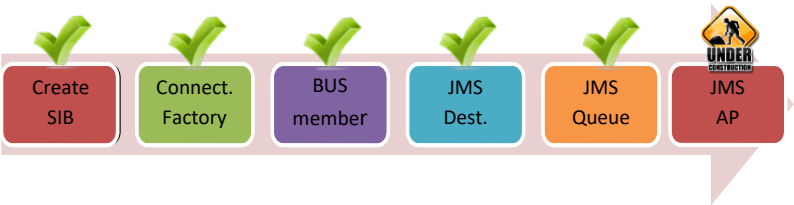
Scope: **Cell=wasv90Cell01**

Select the provider with which to create the Activation specification. The following providers support the selected resource type and are available at the selected scope.

☒ **Default messaging provider**

☐ WebSphere MQ messaging provider

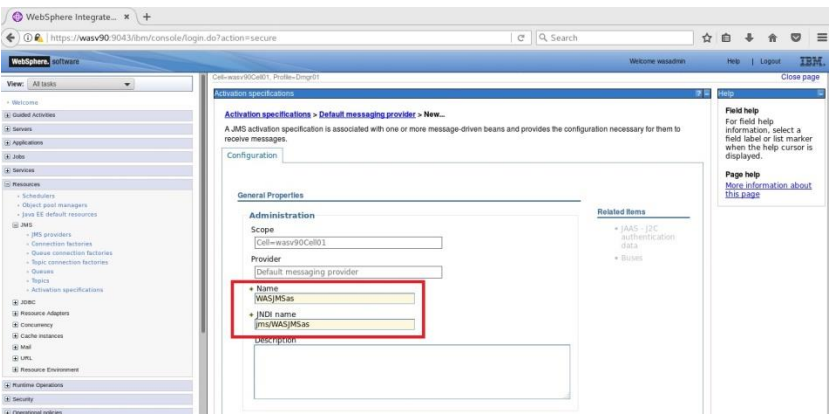
OK **Cancel**

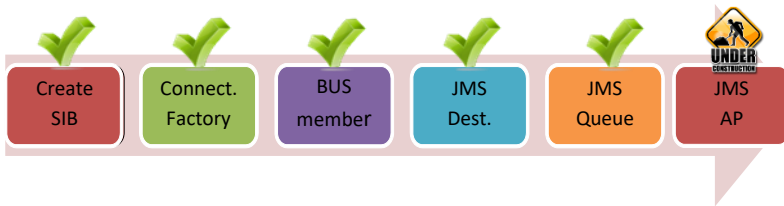


Step 4: Define “Name” and “JNDI name”.

Name: WASJMSas

JNDI name: jms/WASJMSas

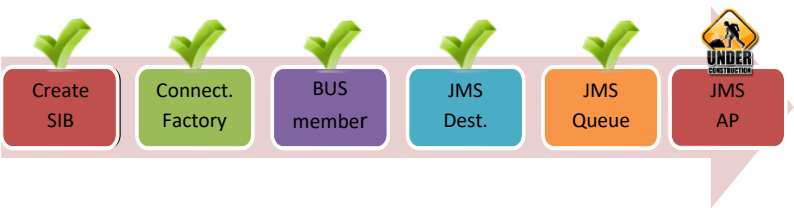




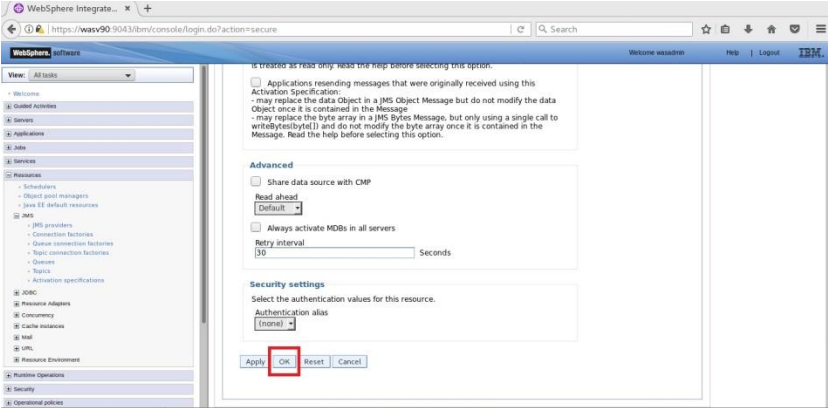
Select "Destination type" as "Queue", for "Destination JNDI name" enter "jms/WASJMSQueue" and select "Bus name" as "WAS_Bus".

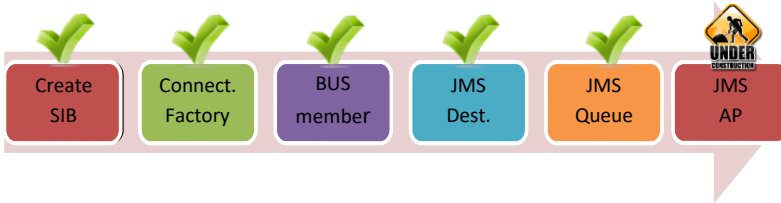
WebSphere Integration console configuration for a JMS destination:

- Destination type:** Queue
- Destination lookup:** jms/WASJMSQueue
- Connection factory lookup:**
- Message selector:**
- Bus name:** WAS_BUS
- Acknowledgement mode:** Auto-acknowledge
- Target:**
- Target type:** Bus member name
- Target significance:**

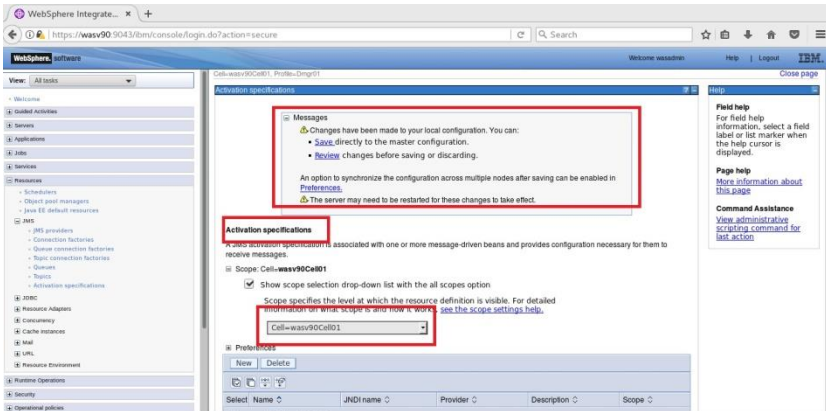


Scroll down and click “OK”.

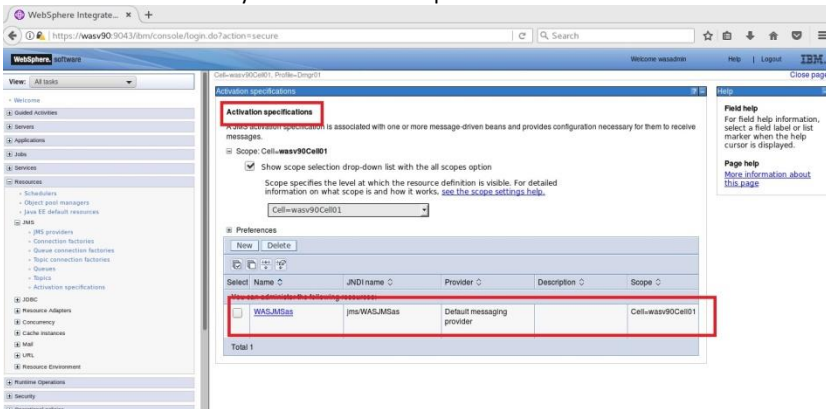




Step 5: Click on “Save” to write changes to the master configuration.



You should see the newly added activation specification listed.



Task 6 is complete!

SUMMARY

WebSphere Application Server supports asynchronous messaging that allows to create, send, receive and read requests between applications. For this communication, you need to configure different components of JMS depending on your application. WebSphere can work with default messaging provider, or with WebSphere MQ or with many other messaging software.

REFERENCES

- http://pic.dhe.ibm.com/infocenter/tivihelp/v50r1/index.jsp?topic=%2Fcom.ibm.mbs.doc%2Ffm_sag%2Fconfigsys%2Fc_jms_config_websphere.html
- http://pic.dhe.ibm.com/infocenter/prodconn/v1r0m0/index.jsp?topic=%2Fcom.ibm.scenarios.wmqwas101.doc%2Ftopics%2Fins_wascfg.htm

INDEX

Asynchronous messaging.....	235
connection factory.....	236
EJB.....	237
Java Message Service (JMS)	235
JMS activation specification.....	237
JMS destination	236
MDB	237
service integration bus	236
Synchronous messaging.....	235
XML.....	236

