

# Oracle Banking Digital Experience

JMS Configuration Multi Entity Guide

Part No. E92727-01

January 2018

**ORACLE®**

Oracle Financial Services Software Limited

Oracle Park

Off Western Express Highway

Goregaon (East)

Mumbai, Maharashtra 400 063

India

Worldwide Inquiries:

Phone: +91 22 6718 3000

Fax: +91 22 6718 3001

[www.oracle.com/financialservices/](http://www.oracle.com/financialservices/)

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are “commercial computer software” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate failsafe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

This software or hardware and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

# Table of Contents

<b>1. Preface.....</b>	<b>4</b>
1.1 Intended Audience .....	4
1.2 Documentation Accessibility .....	4
1.3 Access to Oracle Support .....	4
1.4 Structure.....	4
1.5 Related Information Sources.....	4
<b>2. Objective and Scope .....</b>	<b>5</b>
2.1 Background .....	5
2.2 Objective and Scope .....	5
<b>3. JMS Step 1: Create foreign server in a weblogic server .....</b>	<b>6</b>
3.1 Introduction and Definitions.....	6
3.1.1 Create a JMS Module .....	6
3.1.2 Create a foreign Server .....	6
3.1.3 To configure additional properties for the new foreign server.....	7
3.1.4 Create foreign connection factories .....	7
3.1.5 Create foreign destinations .....	8
<b>4. JMS Step 2 - How to Create a Simple JMS Queue in Weblogic Server.....</b>	<b>10</b>
4.1 Introduction and Definitions.....	10
4.1.1 Create a JMS Server- .....	12
4.1.2 Create a JMS Module .....	14
4.1.3 Create a SubDeployment.....	17
4.1.4 Create a Connection Factory .....	19
4.1.5 Create a JMS Queue .....	20

# 1. Preface

## 1.1 Intended Audience

This document is intended for the following audience:

- Customers
- Partners

## 1.2 Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc..>

## 1.3 Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit

<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit

<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

## 1.4 Structure

This manual is organized into the following categories:

*Preface* gives information on the intended audience. It also describes the overall structure of the User Manual.

The subsequent chapters describes following details

- Objective and Scope
- Steps for JMS Configuration

## 1.5 Related Information Sources

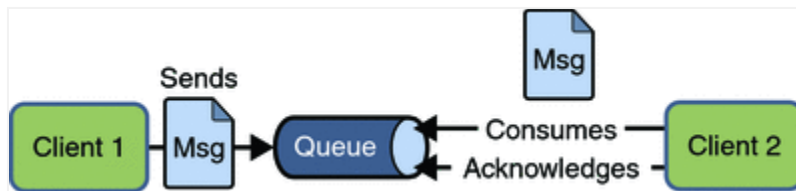
For more information on Oracle Banking Digital Experience Release 18.1.0.0.0, refer to the following documents:

- Oracle Banking Digital Experience Installation Manuals

## 2. Objective and Scope

### 2.1 Background

JMS (Java Message Service) is an API that provides the facility to create, send and read messages. It provides loosely coupled, reliable communication. Messaging enables distributed communication that is loosely coupled. A component sends a message to a destination, and the recipient can retrieve the message from the destination. However, the sender and the receiver do not have to be available at the same time in order to communicate. In fact, the sender does not need to know anything about the receiver; nor does the receiver need to know anything about the sender. The sender and the receiver need to know only which message format and which destination to use. JMS configuration is required to send message (request) to external system and receive processed message (response) from external system.



### 2.2 Objective and Scope

Define a common set of messaging concepts and facilities. The scope of this document is to provide steps to configure foreign server for connecting external system using JNDI provider and configure JMS queue to receive data from external system. Foreign server is used to send message to external system with help of JNDI Initial, JNDI connection url, JNDI connection factory and JNDI destination. To configure JMS receiver queue in web logic we have to create JMS server and JMS module. Where JMS module include creation of JMS connection factory, JMS queue and SubDeployment.

[Home](#)

### 3. JMS Step 1: Create foreign server in a weblogic server

#### 3.1 Introduction and Definitions

A Foreign Server represents a JNDI provider that is outside WebLogic server. It contains information that allows a local WebLogic Server instance to reach a remote JNDI provider, thereby allowing for a number of foreign connection factory and destination objects to be defined on one JNDI directory.

##### 3.1.1 Create a JMS Module

- Services > Messaging > JMS Modules
- Select New
- Name: HostSystemModule
- Leave the other options empty
- Targets: obdx\_server
- Press Next
- Leave “Would you like to add resources to this JMS system module” unchecked and press Finish .

**JMS Modules**

New Delete Showing 1 to 7 of 7 Previous | Next

Name	Type	Scope	Domain Partitions
AsyncFailureLogJMS	JMSSystemResource	Global	
AuditJMS	JMSSystemResource	Global	
EndPointJMSModule	JMSSystemResource	Global	
extXfaceJMSModule	JMSSystemResource	Global	
FileUploadJMS	JMSSystemResource	Global	
HostSystemModule	JMSSystemResource	Global	
UBSSystemModule	JMSSystemResource	Global	

New Delete Showing 1 to 7 of 7 Previous | Next

##### 3.1.2 Create a foreign Server

- Services > Messaging > JMS Modules
- Select HostSystemModule and press New
- Select Foreign Server and Next
- Name: ForeignServer (Once you create a foreign server, you cannot rename it. Instead, you must delete it and create another one that uses the new name) and Click Next to proceed to the targeting page or click **Finish** to create the foreign server.

**Summary of Resources**

New Delete Showing 1 to 1 of 1 Previous | Next

Name	Type	JNDI Name	Subdeployment	Targets
ForeignServer	Foreign Server	N/A	Default Targeting	obdx_server

New Delete Showing 1 to 1 of 1 Previous | Next

### 3.1.3 To configure additional properties for the new foreign server

- Services > Messaging > JMS Modules
- Select HostSystemModule
- Click on ForeignServer
- On the Configuration> General tab
- Enter Following details.
  - JNDI Initial: enter the name of the class that must be instantiated to access the JNDI provider. For example (weblogic.jndi.WLInitialContextFactory)
  - JNDI Connection URL: enter the URL that WebLogic Server uses to contact the JNDI provider. (http://IP:port)
- Click **Save**.

The screenshot shows the 'Configuration' page for a 'ForeignServer' in the 'General' tab. The 'Save' button is at the top left. Below it is a descriptive paragraph: 'A foreign server represents a JNDI provider that resides outside a WebLogic Server. It contains information that allows WebLogic Server to reach the remote JNDI provider. This way, a number of connection factory and destination objects (queues or topics) can be defined on one JNDI directory. Use this page to configure a foreign server.'

The configuration fields are as follows:

- Name:** ForeignServer. Description: The name of this foreign server. [More Info...](#)
- JNDI Initial Context Factory:** weblogic.jndi.WLInitialCont. Description: The name of the class that must be instantiated to access the JNDI provider. This class name depends on the JNDI provider and the vendor that are being used. [More Info...](#)
- JNDI Connection URL:** http://mum00aoz.in.oracle.com:6003. Description: The URL that WebLogic Server will use to contact the JNDI provider. The syntax of this URL depends on which JNDI provider is being used. For WebLogic JMS, leave this field blank if you are referencing WebLogic JMS objects within the same cluster. [More Info...](#)
- JNDI Properties Credential:** (empty field). Description: Any Credentials that must be set for the JNDI provider. These Credentials will be part of the properties will be passed directly to the constructor for the JNDI provider's InitialContext class. Note: For secure credential management, use the Credential field. Using the Properties field results in the credential being stored and displayed as originally entered. [More](#)

### 3.1.4 Create foreign connection factories

- Services > Messaging > JMS Modules
- Select HostSystemModule
- Click on ForeignServer
- On the Configuration> **Connection** Factories tab press **New**
- Enter Following details
  - Name: enter a name for the foreign connection factory.
  - Local JNDI Name: specify the name that the remote object will be bound to in the local server's JNDI tree and is used to look up the object on the local server.
  - Remote JNDI Name: specify the name of the remote object that will be looked up in the remote JNDI directory.
- Click **Ok**.

Settings for ForeignConnectionFactory

Configuration Notes

Save

A foreign connection factory is a connection factory that resides on another server instance and is accessible via JNDI. A remote connection factory can be used to refer to another instance of WebLogic Server running in a different cluster or server, or a foreign provider, as long as that provider supports JNDI.

Use this page to create a foreign connection factory.

Name: ForeignConnectionFactory The name of this foreign connection factory. More Info...

Local JNDI Name: HostQCF The name that the remote object will be bound to in the local server's JNDI tree. This is the name that should be used to look up the object on the local server. More Info...

Remote JNDI Name: HostQCF The name of the remote object that will be looked up in the remote JNDI directory. More Info...

Settings for ForeignServer

Configuration Subdeployment Notes

General Destinations Connection Factories

A foreign connection factory represents a connection factory that resides on another server, and which is accessible via JNDI. A remote connection factory can be used to refer to another instance of WebLogic Server running in a different cluster or server, or a foreign provider, as long as that provider supports JNDI.

This page summarizes the foreign connection factories that have been created for this domain.

Customize this table

Foreign Connection Factories (Filtered - More Columns Exist)

Name	Local JNDI Name	Remote JNDI Name
ForeignConnectionFactory	HostQCF	HostQCF

### 3.1.5 Create foreign destinations

- Services > Messaging > JMS Modules
- Select HostSystemModule
- Click on ForeignServer
- On the Configuration>Destination tab press New
- Enter Following details
  - Name: enter a name for the foreign destination.
  - Local JNDI Name: specify the name that the remote object will be bound to in the local server's JNDI tree and is used to look up the object on the local server.
  - Remote JNDI Name: specify the name of the remote object that will be looked up in the remote JNDI directory.
- Click Ok.



Settings for ForeignDestination

Configuration Notes

Save

A foreign destination (topic or queue) is a destination on a remote server. When this destination is looked up on the local server, a look-up will be performed automatically on the remote JNDI directory, and the object will be returned from that directory.

Use this page to configure a foreign destination.

**Name:** ForeignDestination The name of this foreign destination. [More Info...](#)

**Local JNDI Name:**  The name that the remote object will be bound to in the local server's JNDI tree. This is the name that should be used to look up the object on the local server. [More Info...](#)

**Remote JNDI Name:**  The name of the remote object that will be looked up in the remote JNDI directory. [More Info...](#)

Configuration Subdeployment Notes

General Destinations Connection Factories

A foreign destination (topic or queue) can be found on a remote server. When this destination is looked up on the local server, a look-up will be performed automatically on the remote JNDI directory, and the object will be returned from that directory.

This page summarizes the foreign destinations that have been created for this domain.

[Customize this table](#)

**Foreign Destinations**

Showing 1 to 1 of 1 Previous | Next

<input type="checkbox"/>	Name ↕	Local JNDI Name	Remote JNDI Name
<input type="checkbox"/>	ForeignDestination	HostProcess	HostProcess

Showing 1 to 1 of 1 Previous | Next

[Home](#)

## 4. JMS Step 2 - How to Create a Simple JMS Queue in Weblogic Server

### 4.1 Introduction and Definitions

A JMS queue in Weblogic Server is associated with a number of additional resources:

#### JMS Server

A JMS server acts as a management container for resources within JMS modules. Some of its responsibilities include the maintenance of persistence and state of messages and subscribers. A JMS server is required in order to create a JMS module.

#### JMS Module

A JMS module is a definition which contains JMS resources such as queues and topics. A JMS module is required in order to create a JMS queue.

#### Subdeployment

JMS modules are targeted to one or more WLS instances or a cluster. Resources within a JMS module, such as queues and topics are also targeted to a JMS server or WLS server instances. A subdeployment is a grouping of targets. It is also known as advanced targeting.

#### Connection Factory

A connection factory is a resource that enables JMS clients to create connections to JMS destinations.

#### JMS Queue

A JMS queue (as opposed to a JMS topic) is a point-to-point destination type. A message is written to a specific queue or received from a specific queue.

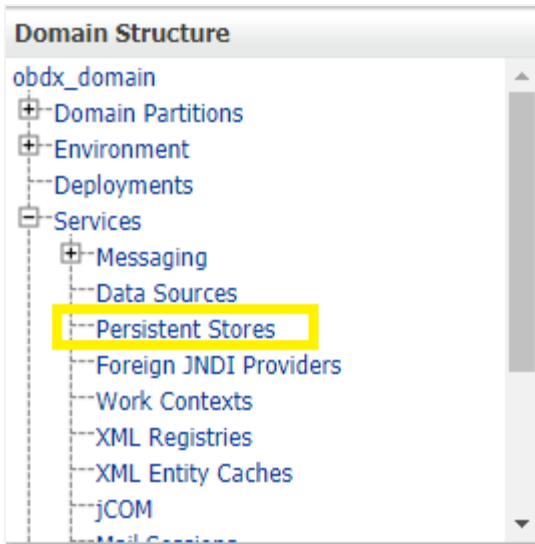
The objects used in this example are:

Object Name	Type	JNDI Name
ExtXfaceJMSServer	JMS Server	
extXfaceJMSModule	JMS Module	
extXfaceSubdeployment	Subdeployment	
ReceiverQCF	Connection Factory	
ReceiverQueue	JMS Queue	

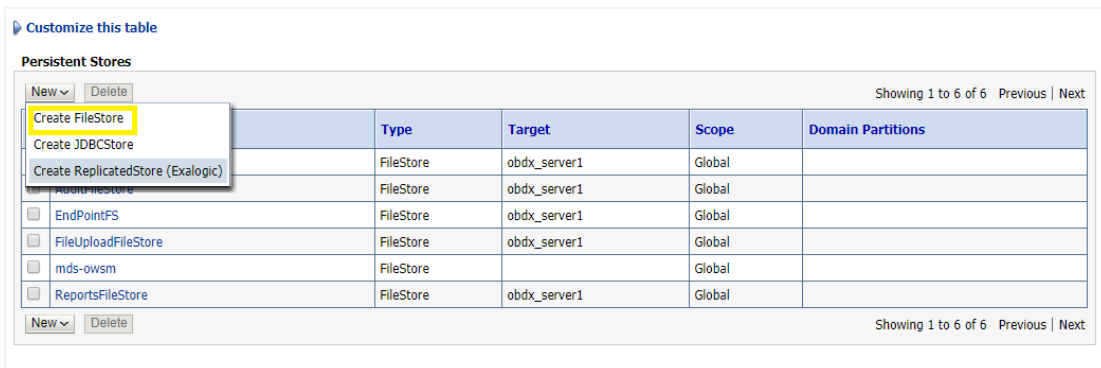
1. Configuration Steps-The following steps are done in the WebLogic Server Console, beginning with the left-hand navigation menu.

**Create Persistent store-**

- Here you have to Create a new persistent store (Once the persistent store is created that can be used for both sender and receiver server. Hence there is no need to create a different persistent store for two different servers.) Hence Before creating a JMS server you need to create the Persistent store if its not already created. Follow the steps shown below for creating a persistent store.
- Select **Services > Persistent Stores**.



- Select new and the select create FileStore from the list as shown below-



- Give the name of the filestore. Example- **EndPointFS** and the Directory location, example **/scratch/obdx/wls**.
- Click **Finish**.

**Create a New File Store**

Back Next Finish Cancel

**File Store Properties**

The following properties will be used to identify your new file store.  
\* Indicates required fields

What would you like to name your new file store?

\* Name:

What scope do you want to create your jms file store in ?

Scope:

The pathname to the directory on the file system where the file store is kept. This directory must exist on your system, so be sure to create it before completing this tab.

Directory:

Back Next Finish Cancel

### 4.1.1 Create a JMS Server-

Services > Messaging > JMS Servers



- Select **New**.

**JMS Servers (Filtered - More Columns Exist)**

New Delete Showing 1 to 6 of 6 Previous | Next

<input type="checkbox"/>	Name	Persistent Store	Target	Current Target	Health	Scope	Domain Partitions
<input type="checkbox"/>	AsyncFailureLogJMServer	AsyncFailureLogFileStore	obdx_server1	obdx_server1		Global	
<input type="checkbox"/>	AuditJMServer	AuditFileStore	obdx_server1	obdx_server1		Global	
<input type="checkbox"/>	ExtfaceReceiverServer	EndPointFS	obdx_server1	obdx_server1		Global	
<input type="checkbox"/>	ExtfaceSenderServer	EndPointFS	obdx_server1	obdx_server1		Global	
<input type="checkbox"/>	FileUploadJMServer	FileUploadFileStore	obdx_server1	obdx_server1		Global	
<input type="checkbox"/>	ReportsJMServer	ReportsFileStore	obdx_server1	obdx_server1		Global	

New Delete Showing 1 to 6 of 6 Previous | Next

- Name: Give name as for example-**ExtfaceReceiverServer** .

- After naming the server **Click next** as shown in following example screenshot.

**Create a New JMS Server**

Back Next Finish Cancel

**JMS Server Properties**

The following properties will be used to identify your new JMS Server.  
\* Indicates required fields

What would you like to name your new JMS server?

**Name:** ExtfaceReceiverServer

Would you like this new JMS server to be restricted to a specific resource group template or resource group ?

**Scope:** Global

Back Next Finish Cancel

- **Persistent Store:** Select the name Persistent store from the dropdown list which was created in the previous step. Example-**EndPointFS**.
- Click **Next**.

**Create a New JMS Server**

Back Next Finish Cancel

**Select Persistent Store**

Specify a persistent store for the new JMS server.

**Persistent Store:** EndPointFS Create a New Store

Back Next Finish Cancel

- **Target:** Target should Point to the **Weblogic server cluster** as in this case target is set to **obdx\_server1** cluster. (Or any other available cluster).
- Click **Finish**.

**Create a New JMS Server**

Back Next Finish Cancel

**Select targets**

Select the server instance or migratable target on which you would like to deploy this JMS server.

**Target:** obdx\_server1

Back Next Finish Cancel

The JMS server should now be visible in the list with Health OK.

Customize this table

**JMS Servers (Filtered - More Columns Exist)**

Click the *Lock & Edit* button in the Change Center to activate all the buttons on this page.

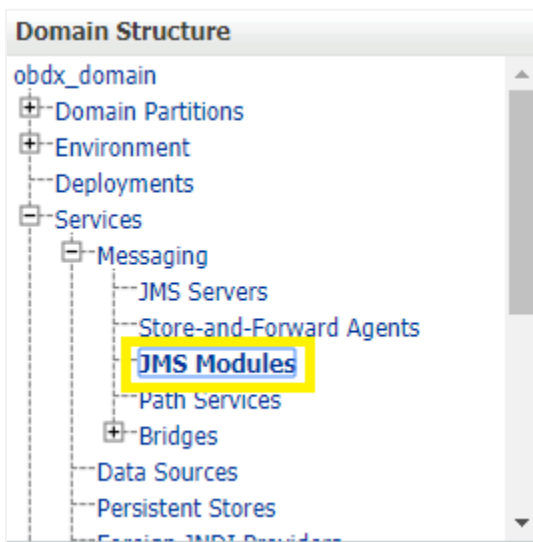
New Delete Showing 1 to 6 of 6 Previous | Next

<input type="checkbox"/>	Name	Persistent Store	Target	Current Target	Health	Scope	Domain Partitions
<input type="checkbox"/>	AsyncFailureLogJMServer	AsyncFailureLogFileStore	obdx_server1	obdx_server1		Global	
<input type="checkbox"/>	AuditJMServer	AuditFileStore	obdx_server1	obdx_server1		Global	
<input type="checkbox"/>	ExtxfceReceiverServer	EndPointFS	obdx_server1	obdx_server1		Global	
<input type="checkbox"/>	ExtxfceSenderServer	EndPointFS	obdx_server1	obdx_server1		Global	
<input type="checkbox"/>	FileUploadJMServer	FileUploadFileStore	obdx_server1	obdx_server1		Global	
<input type="checkbox"/>	ReportsJMServer	ReportsFileStore	obdx_server1	obdx_server1		Global	

New Delete Showing 1 to 6 of 6 Previous | Next

#### 4.1.2 Create a JMS Module

- Services > Messaging > JMS Modules.



- Select **New**.

Customize this table

**JMS Modules**

New Delete Showing 1 to 9 of 9 Previous | Next

Name	Type	Scope	Domain Partitions
AsyncFailureLogJMS	JMSSystemResource	Global	
AuditJMS	JMSSystemResource	Global	
ExtfaceReceiverModule	JMSSystemResource	Global	
ExtfaceReceiverModule2	JMSSystemResource	Global	
ExtfaceSenderModule	JMSSystemResource	Global	
ExtfaceSenderModule2	JMSSystemResource	Global	
FileUploadJMS	JMSSystemResource	Global	
ReportsJMSModule	JMSSystemResource	Global	
UBSSystemModule	JMSSystemResource	Global	

New Delete Showing 1 to 9 of 9 Previous | Next

- Name: Provide name for JMS Module. Example- **ExtfaceReceiverModule**.
- Leave the other options empty.
- Click **Next**.

What would you like to name your System Module?

\* Name:

Would you like this new JMS System Module to be restricted to a specific resource group template or resource group ?

Scope:

What would you like to name the descriptor file name? If you do not provide a name, a default will be assigned.

Descriptor File Name:

Where would like to place the descriptor for this System Module, relative to the jms configuration sub-directory of your domain?

Location In Domain:

Back Next Finish Cancel

- Targets: **Obdx\_Cluster** (or choose any other clusters available).
- Press **Next**.

**Targets :**

**Servers**

AdminServer

**Clusters**

obdx\_cluster

- All servers in the cluster
- Part of the cluster
  - obdx\_server1

Back Next Finish Cancel

Leave **“Would you like to add resources to this JMS system module”** unchecked and press **Finish** .

**Create JMS System Module**

---

**Add resources to this JMS system module**

Use this page to indicate whether you want to immediately add resources to this JMS system module after it is created. JMS resources include queues, topics, connection factories, and such.

**Would you like to add resources to this JMS system module?**

▶ [Customize this table](#)

**JMS Modules**

Showing 1 to 9 of 9 Previous | Next

Name ↕	Type	Scope	Domain Partitions
<input type="checkbox"/> AsyncFailureLogJMS	JMSSystemResource	Global	
<input type="checkbox"/> AuditJMS	JMSSystemResource	Global	
<input type="checkbox"/> ExtxfaceReceiverModule	JMSSystemResource	Global	
<input type="checkbox"/> ExtxfaceReceiverModule2	JMSSystemResource	Global	
<input type="checkbox"/> ExtxfaceSenderModule	JMSSystemResource	Global	
<input type="checkbox"/> ExtxfaceSenderModule2	JMSSystemResource	Global	
<input type="checkbox"/> FileUploadJMS	JMSSystemResource	Global	
<input type="checkbox"/> ReportsJMSModule	JMSSystemResource	Global	
<input type="checkbox"/> UBSSystemModule	JMSSystemResource	Global	

Showing 1 to 9 of 9 Previous | Next



### 4.1.3 Create a SubDeployment

A subdeployment is not necessary for the JMS queue to work, but it allows you to easily target subcomponents of the JMS module to a single target or group of targets. We will use the subdeployment in this example to target the following connection factory and JMS queue to the JMS server we created earlier.

- Services > Messaging > JMS Modules.
- Select **ExtxfaceReceiverModule**.

Customize this table

**JMS Modules**

New Delete Showing 1 to 9 of 9 Previous Next

Name	Type	Scope	Domain Partitions
AsyncFailureLogJMS	JMSSystemResource	Global	
AuditJMS	JMSSystemResource	Global	
ExtxfaceReceiverModule	JMSSystemResource	Global	
ExtxfaceReceiverModule2	JMSSystemResource	Global	
ExtxfaceSenderModule	JMSSystemResource	Global	
ExtxfaceSenderModule2	JMSSystemResource	Global	
FileUploadJMS	JMSSystemResource	Global	
ReportsJMSModule	JMSSystemResource	Global	
UBSSystemModule	JMSSystemResource	Global	

New Delete Showing 1 to 9 of 9 Previous Next

- Select the **Subdeployments** tab and click **New**.

Settings for ExtxfaceReceiverModule

Configuration **Subdeployments** Targets Security Notes

This page displays subdeployments created for a JMS system module. A subdeployment is a mechanism by which JMS module resources (such as queues, topics, and connection factories) are grouped and targeted to a server resource (such as JMS servers, server instances, or cluster).

Customize this table

**Subdeployments**

New Delete Showing 1 to 1 of 1 Previous Next

Name	Resources	Targets
ExtxfaceReceiverSubDep	ExtxfaceReceiverQueue	ExtxfaceReceiverServer

New Delete Showing 1 to 1 of 1 Previous Next

- Subdeployment Name: give subdeployment name. example- **ExtxfaceReceiverSubDep**
- Press **Next**.

Create a New Subdeployment

Back Next Finish Cancel

**Subdeployment Properties**

The following properties will be used to identify your new subdeployment.

\* Indicates required fields

\* Subdeployment Name:

Back Next Finish Cancel

- Here you can select the target(s) for the subdeployment. You can choose either Servers (i.e. WebLogic managed servers, such as the **obdx\_server**) or JMS Servers such as the JMS Server created earlier. As the purpose of our subdeployment in this example is to target a specific JMS server, we will choose the JMS Server option. Select the **ExtxfaceReceiverServer** created earlier.
- Press **Finish**.

**Targets**

Please select targets for the Subdeployment

**Clusters**

obdx\_cluster

All servers in the cluster

Part of the cluster

obdx\_server1

**JMS Servers**

AsyncFailureLogJMSServer

AuditJMSServer

ExtxfaceReceiverServer

ExtxfaceSenderServer

FileUploadJMSServer

ReportsJMSServer

Back Next Finish Cancel

#### 4.1.4 Create a Connection Factory

- Services > Messaging > JMS Modules
- Select **ExtxfaceReceiverModule** and press **New**.

Customize this table

**JMS Modules**

New Delete Showing 1 to 9 of 9 Previous | Next

<input type="checkbox"/>	Name ↕	Type	Scope	Domain Partitions
<input type="checkbox"/>	AsyncFailureLogJMS	JMSSystemResource	Global	
<input type="checkbox"/>	AuditJMS	JMSSystemResource	Global	
<input type="checkbox"/>	ExtxfaceReceiverModule	JMSSystemResource	Global	
<input type="checkbox"/>	ExtxfaceReceiverModule2	JMSSystemResource	Global	
<input type="checkbox"/>	ExtxfaceSenderModule	JMSSystemResource	Global	
<input type="checkbox"/>	ExtxfaceSenderModule2	JMSSystemResource	Global	
<input type="checkbox"/>	FileUploadJMS	JMSSystemResource	Global	
<input type="checkbox"/>	ReportsJMSModule	JMSSystemResource	Global	
<input type="checkbox"/>	UBSSystemModule	JMSSystemResource	Global	

New Delete Showing 1 to 9 of 9 Previous | Next

Customize this table

**Summary of Resources**

New Delete Showing 1 to 2 of 2 Previous | Next

<input type="checkbox"/>	Name ↕	Type	JNDI Name	Subdeployment	Targets
<input type="checkbox"/>	ExtxfaceReceiverQCF	Connection Factory	ExtSystemReceiverQCF	Default Targeting	obdx_server1
<input type="checkbox"/>	ExtxfaceReceiverQueue	Queue	ExtSystemReceiverQueue	ExtxfaceReceiverSubDep	ExtxfaceReceiverServer

New Delete Showing 1 to 2 of 2 Previous | Next

- Select **Connection Factory** and click **Next**.

Create a New JMS System Module Resource

Back **Next** Finish Cancel

Choose the type of resource you want to create.

Use these pages to create resources in a JMS system module, such as queues, topics, templates, and connection factories.

Depending on the type of resource you select, you are prompted to enter basic information for creating the resource. For targetable resources, like stand-alone queues and topics, connection factories, distributed queues and topics, foreign servers, and JMS SAF destinations, you can also proceed to targeting pages for selecting appropriate server targets. You can also associate targetable resources with subdeployments, which is an advanced mechanism for grouping JMS module resources and the members to server resources.

- Connection Factory** Defines a set of connection configuration parameters that are used to create connections for JMS clients. [More Info...](#)
- Queue** Defines a point-to-point destination type, which are used for asynchronous peer communications. A message delivered to a queue is distributed to only one consumer. [More Info...](#)
- Topic** Defines a publish/subscribe destination type, which are used for asynchronous peer communications. A message delivered to a topic is distributed to all topic consumers. [More Info...](#)

- Name: Give name of the connection factory example- **ExtxfaceReceiverQCF**.  
JNDI Name: **ExtSystemReceiverQCF**.
- Click **Next**.

**Create a New JMS System Module Resource**

Back Next Finish Cancel

**Connection Factory Properties**

The following properties will be used to identify your new connection factory. The current module is ExtxfaceReceiverModule.  
\* Indicates required fields

What would you like to name your new connection factory?

\* Name:

What JNDI Name would you like to use to look up your new connection factory?

JNDI Name:

The Connection Factory Subscription Sharing Policy Subscribers can be used to control which subscribers can access new subscriptions. Should subscriptions created using this factory be sharable?

Subscription Sharing Policy:

The Client ID Policy indicates whether more than one JMS connection can use the same Client ID. Oracle recommends setting the Client ID policy to Unrestricted if sharing durable subscribers. Subscriptions created with different Client ID policies are always treated as independent subscriptions. What Client ID Policy would you like to use?

Client ID Policy:

A connection factory can limit the number of messages that can be queued for an asynchronous session. Should this connection factory impose a limit?

Maximum Messages per Session:

- Select Default Targeting Enabled and Press **Finish**
- The connection factory should be listed on the following page with **Default Targeting** as Subdeployment and WebLogic cluster as the target.

#### 4.1.5 Create a JMS Queue

- Services > Messaging > JMS Modules
- Select **ExtxfaceReceiverModule** and Click **New**.

Customize this table

**JMS Modules**

New Delete Showing 1 to 9 of 9 Previous | Next

Name	Type	Scope	Domain Partitions
AsynFailureLogJMS	JMSSystemResource	Global	
AuditJMS	JMSSystemResource	Global	
ExtxfaceReceiverModule	JMSSystemResource	Global	
ExtxfaceReceiverModule2	JMSSystemResource	Global	
ExtxfaceSenderModule	JMSSystemResource	Global	
ExtxfaceSenderModule2	JMSSystemResource	Global	
FileUploadJMS	JMSSystemResource	Global	
ReportsJMSSModule	JMSSystemResource	Global	
UBSSystemModule	JMSSystemResource	Global	

New Delete Showing 1 to 9 of 9 Previous | Next

Customize this table

Summary of Resources

New Delete Showing 1 to 2 of 2 Previous | Next

<input type="checkbox"/>	Name ^	Type	JNDI Name	Subdeployment	Targets
<input type="checkbox"/>	ExtfaceReceiverQCF	Connection Factory	ExtSystemReceiverQCF	Default Targeting	obdx_server1
<input type="checkbox"/>	ExtfaceReceiverQueue	Queue	ExtSystemReceiverQueue	ExtfaceReceiverSubDep	ExtfaceReceiverServer

New Delete Showing 1 to 2 of 2 Previous | Next

- Select **Queue** and Click **Next**.

Back Next Finish Cancel

Choose the type of resource you want to create.

Use these pages to create resources in a JMS system module, such as queues, topics, templates, and connection factories.

Depending on the type of resource you select, you are prompted to enter basic information for creating the resource. For targetable resources, like stand-alone queues and topics, connection factories, distributed queues and topics, foreign servers, and JMS SAF destinations, you can also proceed to targeting pages for selecting appropriate server targets. You can also associate targetable resources with subdeployments, which is an advanced mechanism for grouping JMS module resources and the members to server resources.

Connection Factory Defines a set of connection configuration parameters that are used to create connections for JMS clients. [More Info...](#)

Queue Defines a point-to-point destination type, which are used for asynchronous peer communications. A message delivered to a queue is distributed to only one consumer. [More Info...](#)

Topic Defines a publish/subscribe destination type, which are used for asynchronous peer communications. A message delivered to a topic is distributed to all topic consumers. [More Info...](#)

Distributed Queue Defines a set of queues that are distributed on multiple JMS servers, but which are accessible as a single, logical queue to JMS clients. [More Info...](#)

- **Name:** Provide name of the message queue. example- **ExtfaceReceiverQueue**.
- **JNDI Name:** Provide JNDI name. example- **ExtSystemReceiverQueue**.
- **Template:** **None**.
- Press **Next**.

Create a New JMS System Module Resource

Back Next Finish Cancel

JMS Destination Properties

The following properties will be used to identify your new Queue. The current module is ExtfaceReceiverModule.

\* Indicates required fields

\* Name: ExtfaceReceiverQueue

JNDI Name: ExtSystemReceiverQueue

Template: None ▾

Back Next Finish Cancel

- **Subdeployments:** Give the name of the sub-deployment name in which Queue is supposed to be added. **Example-** ExtxfaceReceiverSubDep.
- Select the Target as **ExtxfaceReceiverServer**  
Click **Finish**.

The following properties will be used to target your new JMS system module resource

Use this page to select a subdeployment to assign this system module resource. A subdeployment is a mechanism by which JMS resources are grouped and targeted to a server instance, cluster, or SAF agent. If necessary, you can create a new subdeployment by clicking the **Create a New Subdeployment** button. You can also reconfigure subdeployment targets later by using the parent module's subdeployment management page.

Select the subdeployment you want to use. If you select (none), no targeting will occur.

Subdeployments: **ExtxfaceReceiverSubDep** [Create a New Subdeployment](#)

What targets do you want to assign to this subdeployment?

Targets :

JMS Servers
<input type="radio"/> AsyncFailureLogJMSServer
<input type="radio"/> AuditJMSServer
<input checked="" type="radio"/> ExtxfaceReceiverServer
<input type="radio"/> ExtxfaceSenderServer
<input type="radio"/> FileUploadJMSServer
<input type="radio"/> ReportsJMSServer

The **ReceiverQueue** should be listed on the following page with Sub-deployment as **ExtxfaceReceiverSubDep** and target as **ExtxfaceReceiverServer**.

[Customize this table](#)

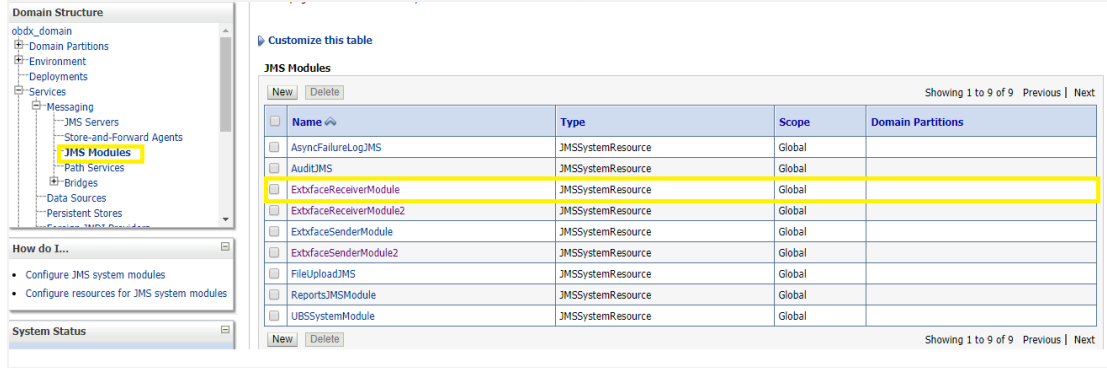
Summary of Resources

[New](#) [Delete](#) Showing 1 to 2 of 2 Previous | Next

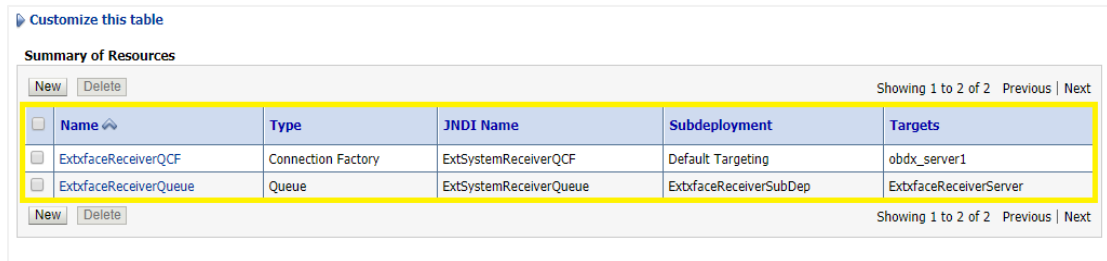
<input type="checkbox"/>	Name	Type	JNDI Name	Subdeployment	Targets
<input type="checkbox"/>	ExtxfaceReceiverQCF	Connection Factory	ExtSystemReceiverQCF	Default Targeting	obdx_server1
<input type="checkbox"/>	ExtxfaceReceiverQueue	Queue	ExtSystemReceiverQueue	ExtxfaceReceiverSubDep	ExtxfaceReceiverServer

[New](#) [Delete](#) Showing 1 to 2 of 2 Previous | Next

Confirm the resources for the **ExtxfaceReceiverModule**. Using the Domain Structure tree, navigate to Services > Messaging > JMS Modules then select **ExtxfaceReceiverModule**



You should see the following resources-



The JMS queue is now complete and can be accessed using the JNDI names **ExtSystemReceiverQCF** And **ExtSystemReceiverQueue**.

---

**Note:** Repeat the above process from the step [Create JMS Server](#) to create the JMS Configuration for Sender. Separate JMS Server , Module and Queues would get created for Sender.

---

**After Creating the JMS configuration for both the Receiver and sender you have to manually deploy the ExtxfaceSimulatorMDB.ear on weblogic server.**

---

**Note:** Whenever a new Entity is created within a setup the following mentioned steps should be followed in order to enable support for MultiEntity.

---

In order to enable the support for newly created Entity, Sender/Receiver Connection Factories and Queues are needed to be created within a new Sender/Receiver JMS Modules. These modules can be hosted on the same Sender/Receiver JMS server created as per the steps defined in section [4.1.1\(Create JMS Server\)](#) for the earlier Entity.

- Create a new JMS Module by repeating steps given in section [4.1.2\(Creating JMS Module\)](#) , on the same JMS server with new names as follows-

	JMS Module Name
Sender JMS Module	ExtxfaceSenderModule2
Receiver JMS Module	ExtxfaceReceiverModule2

- Create a new SubDeployment within both Sender/Receive module created with above step by repeating the procedure given in section [4.1.3](#) (Create JMS Subdeployment) with the new name as follows-

	<b>JMS Module Name</b>	<b>SubDeployment Name</b>
Sender JMS Module	ExtxfaceSenderModule2	ExtxfaceReceiverSubDep2
Receiver JMS Module	ExtxfaceReceiverModule2	ExtxfaceSenderSubDep2

- Create Sender/Receiver connection factories within newly created module by following the steps defined in the section [4.1.4](#)(Creating Connection Factories), with different names as follows-

	<b>Connection Factory Name</b>	<b>Connection Factory JNDI Name</b>
Sender Connection Factory	ExtxfaceSenderQCF2	ExtSystemSenderQCF2
Receiver Connection Factory	ExtxfaceReceiverQCF2	ExtSystemReceiverQCF2

- Create Sender/Receiver JMS queues within newly created JMS module by repeating the steps given in section [4.1.5](#) (Creating JMS Queues), with the new names to the sender/receiver queues as follows-

	<b>JMS Queue Name</b>	<b>JMS Queue JNDI Name</b>
Sender JMS Queue	ExtxfaceSenderQueue2	ExtSystemSenderQueue2
Receiver JMS Queue	ExtxfaceReceiverQueue2	ExtSystemReceiverQueue2



After creating the new JMS sender/receiver modules, connection factories and queues by following the above defined steps. Further Redeploy the ExtxfaceSimulatorMDB.ear with the following changes -

- Add new <message-driven> tag in **ejb-jar.xml** (Path- ExtxfaceSimulatorMDB.ear\com.ofss.digx.extxface.mdb.jar\META-INF\ **ejb-jar.xml**) as shown below –

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (c) 2012, Oracle and/or its affiliates. All rights reserved. -->
- <ejb-jar version="3.0" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/ejb-jar_3_0.xsd"
xmlns:ejb="http://java.sun.com/xml/ns/javaee/ejb-jar_3_0.xsd" xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <display-name>com.ofss.digx.extxface.mdb</display-name>
  - <enterprise-beans>
    - <message-driven>
      <display-name>ExtxfaceSimulatorMDB</display-name>
      <ejb-name>ExtxfaceSimulatorMDB</ejb-name>
      <ejb-class>com.ofss.digx.extxface.mdb.ExtxfaceSimulatorMDB</ejb-class>
      <transaction-type>Bean</transaction-type>
      <message-destination-type>javax.jms.Queue</message-destination-type>
    </message-driven>
    - <message-driven>
      <display-name>ExtxfaceSimulatorMDB2</display-name>
      <ejb-name>ExtxfaceSimulatorMDB2</ejb-name>
      <ejb-class>com.ofss.digx.extxface.mdb.ExtxfaceSimulatorMDB</ejb-class>
      <transaction-type>Bean</transaction-type>
      <message-destination-type>javax.jms.Queue</message-destination-type>
    </message-driven>
  </enterprise-beans>
</ejb-jar>

```

Fig.1 ExtxfaceSimulatorMDB.ear\com.ofss.digx.extxface.mdb.jar\META-INF\ejb-jar.xml

```

<message-driven>
<display-name>ExtxfaceSimulatorMDB2</display-name>
<ejb-name>ExtxfaceSimulatorMDB2</ejb-name>
<ejb-class>com.ofss.digx.extxface.mdb.ExtxfaceSimulatorMDB</ejb-class>
<transaction-type>Bean</transaction-type>
<message-destination-type>javax.jms.Queue</message-destination-type>
</message-driven>

```

**Note:** As Shown in above example the value of the <ejb-class> sub-tag in <message-driven> tag should be same for all the Entities.

- Add new <weblogic-enterprise-bean> configuration tag in **weblogic-ejb-jar (Path- ExtxfaceSimulatorMDB.ear\com.ofss.digx.extxface.mdb.jar\META-INF\ weblogic-ejb-jar)** as shown below-

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (c) 2012, Oracle and/or its affiliates. All rights reserved. -->
<weblogic-ejb-jar xsi:schemaLocation="http://xmlns.oracle.com/weblogic/weblogic-ejb-jar/1.1/weblogic-ejb-jar.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://xmlns.oracle.com/weblogic/weblogic-ejb-jar">
- <weblogic-enterprise-bean>
  <ejb-name>ExtxfaceSimulatorMDB</ejb-name>
  <dispatch-policy>ExtxfaceWorkManager</dispatch-policy>
- <message-driven-descriptor>
  - <pool>
    <initial-beans-in-free-pool>10</initial-beans-in-free-pool>
    <max-beans-in-free-pool>100</max-beans-in-free-pool>
  </pool>
  <destination-jndi-name>ExtSystemSenderQueue</destination-jndi-name>
  <connection-factory-jndi-name>ExtSystemSenderQCF</connection-factory-jndi-name>
  <jms-polling-interval-seconds>1</jms-polling-interval-seconds>
</message-driven-descriptor>
  <jndi-name>ExtSystemSenderQueue</jndi-name>
- <transaction-descriptor>
  <trans-timeout-seconds>60</trans-timeout-seconds>
</transaction-descriptor>
</weblogic-enterprise-bean>
- <weblogic-enterprise-bean>
  <ejb-name>ExtxfaceSimulatorMDB2</ejb-name>
  <dispatch-policy>ExtxfaceWorkManager</dispatch-policy>
- <message-driven-descriptor>
  - <pool>
    <initial-beans-in-free-pool>10</initial-beans-in-free-pool>
    <max-beans-in-free-pool>100</max-beans-in-free-pool>
  </pool>
  <destination-jndi-name>ExtSystemSenderQueue2</destination-jndi-name>
  <connection-factory-jndi-name>ExtSystemSenderQCF2</connection-factory-jndi-name>
  <jms-polling-interval-seconds>1</jms-polling-interval-seconds>
</message-driven-descriptor>
  <jndi-name>ExtSystemSenderQueue2</jndi-name>
- <transaction-descriptor>
  <trans-timeout-seconds>60</trans-timeout-seconds>
</transaction-descriptor>
</weblogic-enterprise-bean>
- <run-as-role-assignment>
  <role-name>LookupRole</role-name>
  <run-as-principal-name>weblogic</run-as-principal-name>
</run-as-role-assignment>
- <work-manager>
  <name>ExtxfaceWorkManager</name>

```

Fig.2 ExtxfaceSimulatorMDB.ear\com.ofss.digx.extxface.mdb.jar\META-INF\weblogic-ejb-jar.xml

```

<weblogic-enterprise-bean>

    <ejb-name>ExtxfceSimulatorMDB2</ejb-name>

    <dispatch-policy>ExtxfceWorkManager</dispatch-policy>

    <message-driven-descriptor>

        <pool>

            <initial-beans-in-free-pool>10</initial-beans-in-free-pool>

            <max-beans-in-free-pool>100</max-beans-in-free-pool>

        </pool>

        <destination-jndi-name>ExtSystemSenderQueue2</destination-jndi-name>

        <connection-factory-jndi-name>ExtSystemSenderQCF2</connection-factory-jndi-
name>

        <jms-polling-interval-seconds>1</jms-polling-interval-seconds>

    </message-driven-descriptor>

    <jndi-name>ExtSystemSenderQueue2</jndi-name>

    <transaction-descriptor>

        <trans-timeout-seconds>60</trans-timeout-seconds>

    </transaction-descriptor>

</weblogic-enterprise-bean>

```

---

**Note:** <destination-jndi-name> i.e. JNDI name of the JMS sender queue should be same as given while creating the queue. In above example it is- ExtSystemSenderQueue2.

<connection-factory-jndi-name> i.e. connection factory JNDI name should be same as given while creating the new connection factory. In the above example it is- ExtSystemSenderQCF2.

<dispatch-policy> value should be same for all the Entities. i.e., ExtxfceWorkManager

---

After Redeploying the ExtxfaceSimulatorMDB and restarting the server, check the **state** of the application by going in **Deployments** wizard on the weblogic server console. If it is not in “Active” state, it needs to be started manually, to do so, follow the steps mentioned below by keeping server in ‘Running’ state-

- Go into the **Control** tab.
- From the List of applications select the checkbox before **ExtxfaceSimulatorMDB ear**.
- Select the ‘**Start**’ dropdown list and from that select option-“**Servicing all requests**”.

[Home](#)