



## DirXML Driver for IBM WebsphereMQ

---

# Driver Overview and Configuration

This document contains trade secrets and proprietary information belonging to Novell. No use or disclosure of the information contained herein is permitted without the prior written consent.

All trade names, trademarks, or registered trademarks are trade names, trademarks, or registered trademarks of their respective companies.



# Table of Contents

TABLE OF CONTENTS .....	4
PREFACE .....	6
DOCUMENTATION CONVENTIONS.....	6
INTRODUCING THE DIRXML™ DRIVER FOR IBM WEBSPHEREMQ* .....	7
OVERVIEW.....	7
UNDERSTANDING DRIVER CONCEPTS .....	7
<i>Publisher Channel</i> .....	7
<i>Integrating application data with eDirectory</i> .....	7
<i>Subscriber Channel</i> .....	8
BENEFITS.....	9
FEATURES .....	9
UNDERSTANDING CONSOLE ONE .....	10
<i>Required Skills</i> .....	10
<i>Expertise in IBM WebsphereMQ</i> .....	10
<i>Expertise in DirXML</i> .....	10
ADDING TO THE DIRXML BASICS.....	10
<i>Event Processing Support</i> .....	11
<i>Rules and Stylesheets</i> .....	11
<i>Associations</i> .....	12
<i>Setting the Global Association Attribute Value</i> .....	13
<i>Overriding the Global Association Attribute Value</i> .....	13
<i>The role of the Subscriber Channel in Setting Associations</i> .....	13
PRODUCT COMPONENTS .....	14
<i>Pre-configured Driver Import File</i> .....	14
<i>Driver Shim</i> .....	14
<i>Schema Import Utility</i> .....	14
PUBLISHING TO EDIRECTORY .....	14
SUBSCRIBING FROM EDIRECTORY .....	15
INSTALLING AND CONFIGURING THE DRIVER .....	15
PREREQUISITES .....	16
PLANNING FOR INSTALLATION.....	16
WEBSPHEREMQ CHECKLIST .....	16
PLANNING A LOCAL OR REMOTE INSTALLATION .....	17
<i>Local Installation</i> .....	17
<i>Remote Installation</i> .....	17
INSTALLING COMPONENTS .....	17
INSTALLING THE DRIVER.....	18
<i>Installing The Driver</i> .....	18
<i>Activating The Driver</i> .....	18
<i>Generating a Product Activation Request</i> .....	19
<i>Submitting a Product Activation Request</i> .....	20

<i>Activating the Driver By Applying the Product Activation Credential Received from Novell</i> .....	21
POST INSTALLATION TASKS .....	22
<i>Importing the Pre-configured Driver</i> .....	22
CONFIGURING THE REMOTE LOADER ON THE REMOTE SYSTEM .....	23
<i>Generating a Security Certificate on the eDirectory Container</i> .....	24
<i>Create an Organizational Trusted Root Certificate</i> .....	24
<i>Using the Wizard to Configure Remote Loader Properties on the Remote Loader Host Server</i> .....	25
<i>Starting the Remote Loader if Installed as a Service</i> .....	26
<i>Starting the Remote Loader Manually</i> .....	26
<i>Configuring Driver Object Properties for Remote Loader</i> .....	27
CONFIGURING DRIVER PARAMETERS .....	28
<i>Configuring Driver Parameters</i> .....	28
<b>TROUBLESHOOTING THE DRIVER</b> .....	<b>30</b>
USING THE DSTRACE FACILITY.....	30
COMMON ERRORS.....	30
<b>MANUALLY INSTALLING THE DRIVER</b> .....	<b>31</b>
INSTALLATION CHECKLIST .....	31
<i>MA88 SupportPac Installation Note</i> .....	32
INSTALLING THE DIRXML-MQ DRIVER .....	32
<i>Creating a driver set</i> .....	33
EXTENDING EDIRECTORY TO SUPPORT JAVA OBJECT STORAGE .....	34
CREATING THE WEBSPHEREMQ FACTORY OBJECTS .....	35
INTERACTING WITH THE DRIVER FOR WEBSPHEREMQ .....	36
<b>APPENDIX A DRIVER PARAMETERS XML</b> .....	<b>37</b>

## Preface

This document is for network administrators, consultants, and IBM WebsphereMQ administrators.

DirXML™ Driver for IBM WebsphereMQ\* (formerly known as IBM MQ Series) is designed to share data between eDirectory™ 8.6.1 and applications that are interfaced to IBM's messaging bus. This configurable solution gives organizations the ability to increase productivity and streamline business processes by integrating any business process and information system platform in the enterprise with Directory Services.

## Documentation Conventions

The term driver refers to all components of DirXML Driver for IBM WebsphereMQ and not to any one particular component.

In Novell documentation, a greater than symbol (>) is used to separate actions within a step and items in a cross-reference path.

In this documentation, a trademark symbol (®,™, etc.) denotes a Novell trademark. An asterisk (\*) denotes a third-party trademark.

# Introducing the DirXML™ Driver for IBM WebsphereMQ\*

## Overview

The DirXML™ Driver for IBM\* WebsphereMQ\*, subsequently referred to as “the driver,” allows the creation of an automated data synchronization link between the any application supported by IBM’s WebsphereMQ message bus and Novell’s eDirectory™.

This technology enables identity profile information managed within business processes across the enterprise to be integrated into a common directory services strategy. The driver can exchange and process information dynamically based on events, eliminating expensive and error-prone manual data entry into multiple databases. As new user profiles are added, modified, or deactivated, business processes associated with these events can be updated automatically.

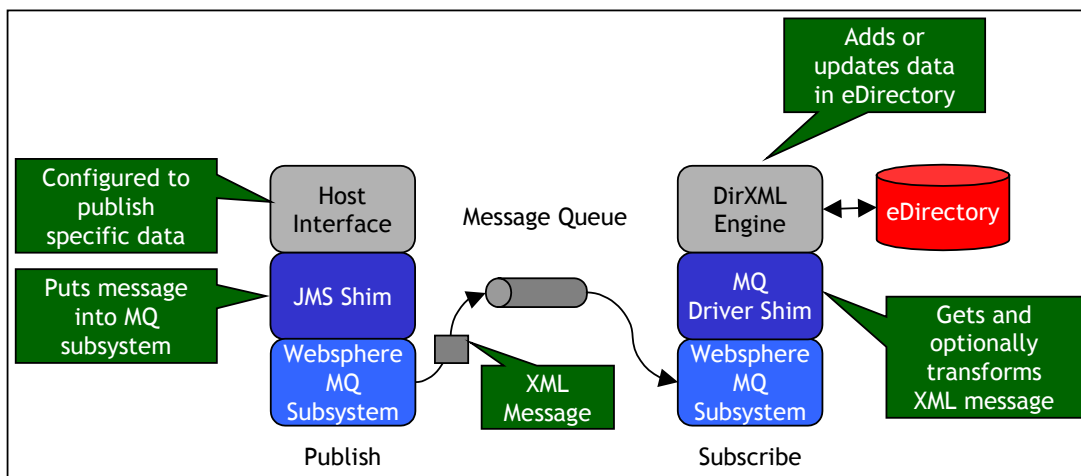
## Understanding Driver Concepts

The driver is a bi-directional synchronization product between IBM\* WebsphereMQ\*-supported systems and eDirectory. This framework uses XML to provide data and event transformation capabilities which convert eDirectory data and events into XML data and vice-versa.

IBM\* WebsphereMQ \*acts as a hub, with other applications and directories publishing changes to it. eDirectory acts as a specialized connector into this hub that sends changes to the applications and directories that have subscribed to them. This results in two main flows of data: the DirXML Publisher channel and the Subscriber channel.

## Publisher Channel

### Integrating application data with eDirectory



The DirXML Publisher channel supports a point-to-point messaging model.

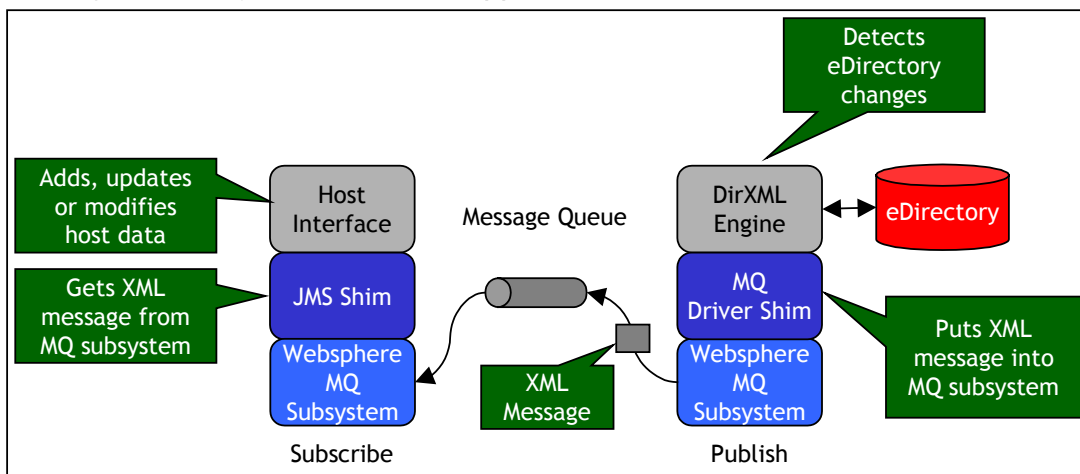
The DirXML-WebsphereMQ driver supports a point-to-point messaging model on the “publishing channel”. The “host application” publishes information in any available form. The JMS (Java Messaging Services) shim consumes the application output and converts the data into XML document format and “puts” the message into a specified WebsphereMQ message queue.

The DirXML Driver Shim for WebsphereMQ “gets” the message from the specified queue and submits XML-formatted changes to the DirXML engine for publication into eDirectory. The engine processes the document by sequentially applying all configured rules based on standard DirXML driver process flow.

The driver can then manipulate the information using various rules, filters, and style sheets defined by the system administrator. The driver then submits the data to eDirectory. Using eDirectory and other DirXML drivers, the data can be shared with other business applications and directories. Based on business rules, these other applications can add additional data that can in turn be submitted to other WebsphereMQ-connected applications.

### Subscriber Channel

Sharing eDirectory data with host applications



The DirXML-WebsphereMQ driver supports both Point-to-Point and Publish-Subscribe messaging models on the “subscriber channel.” The Subscriber channel receives XML-formatted eDirectory events from the DirXML engine.

The driver then converts these documents to an appropriate data format, and “puts” the message into a specified WebsphereMQ message queue. The JMS (Java Messaging Services) shim “gets” the document from the specified queue and hands it off to the Host Interface application. The Subscriber Channel supports both Point-to-Point and Publish-Subscribe messaging models.



All filtered events that are detected by eDirectory will be submitted to the WebsphereMQ-interfaced application via the Subscriber channel. It is up to the specific interface to handle these messages correctly, which can be used as follows:

- **Acknowledgement.** By filtering for a specific association attribute value, the application interface can accept messages that acknowledge receipt and processing of messages while discarding all others.
- **Update.** The application interface can update local tables with the value of various association changes. This allows an application to potentially work intelligently and bi-directional with multiple, authoritative sources of data.
- **Discard.** If the subscriber channel is open to support “global associations” to be set by the stylesheet (described in the section on “Associations, below), messages received on the Subscriber channel can be processed as in the above examples, or simply discarded.

## Benefits

As the following examples illustrate, the driver enables you to automate and maintain identity management business processes:

Automatically create, modify, or deactivate an eDirectory account based on an event in a host application.

Synchronize bi-directional data between the host application and eDirectory.

Maintain accurate and consistent eDirectory IDs.

Create other eDirectory objects associated with a host application object (for example, account codes or department records).

Synchronize attributes between enterprise applications and eDirectory.

You can configure enterprise applications and the Driver objects to enhance and secure your organization’s business processes. Before installing and configuring the driver, you evaluate and define those processes. During installation, you configure the driver’s rules and style sheets to automate these processes wherever possible.

For more information about DirXML, refer to the DirXML Administration Guide (<http://www.novell.com/documentation>). For more information about eDirectory, refer to the Novell® eDirectory 8.6.1 documentation (<http://www.novell.com/documentation/lg/ndsdir86/index.html>).

## Features

The driver provides the following features:

- Support for remote processing.
- Support for LDAP-based storage of JMS factory classes for enhanced security.
- Remote processing is possible between the eDirectory server and remote WebsphereMQ host systems. These machines are connected using any IBM-supported secure connection facility.

## Understanding ConsoleOne

ConsoleOne™ is a tool for managing eDirectory objects, schema, partitions, and replicas. Additional capabilities can be added to ConsoleOne through snap-ins. A Novell product, such as DirXML, adds snap-ins to ConsoleOne so it can manage the product's additional capabilities.

When using ConsoleOne you need to be aware of the snap-ins that are installed. The list of snap-ins registered in ConsoleOne can be viewed at [Help | About Snap-ins](#). Prior to installing and configuring the driver, you should install the DirXML snap-ins found on the DirXML CD. This snap-in contains the tools needed for managing DirXML objects.

## Required Skills

Implementing the driver requires expertise in IBM Websphere MQ and DirXML.

## Expertise in IBM WebsphereMQ

This document assumes that your expertise in IBM WebsphereMQ is equivalent to one of the following:

- An MQ Series or Java Messaging Services developer
- A security administrator
- An application designer
- An upgrade administrator
- An WebsphereMQ administrator

## Expertise in DirXML

This document assumes that your expertise in DirXML is equivalent to one of the following:

- An eDirectory administrator
- An XML developer
- An XSLT developer
- A DirXML administrator

## Adding to the DirXML Basics

The following areas of DirXML functionality are important to the driver:

- Event Processing Support
- Rules and Style Sheets
- Associations

## Event Processing Support

The driver supports the following events on the Publisher and Subscriber Channels:

Functionality	Event
Publisher	<ul style="list-style-type: none"><li>• Add</li><li>• Modify</li><li>• Move</li><li>• Delete</li><li>• Rename</li></ul>
Subscriber	<ul style="list-style-type: none"><li>• Add</li><li>• Modify</li><li>• Move</li><li>• Delete</li><li>• Rename</li></ul>

While the Subscriber channel supports the same NDS event as the Publisher channel, in reality the events are supported using style sheet-based event transformation. This is due to the fact that the Subscriber channel events will always contain polymorphic, application-specific meaning that differ from application to application.

The driver also supports a query capability so that DirXML can query the synchronized application or directory.

- Querying on the Publisher channel is performed against the defined eDirectory schema.
- Querying on the Subscriber channel requires that the application interface is able to accept and respond to such query messages. This is often the case with off-the-shelf WebsphereMQ connectors from 3rd parties and can be implemented within custom interfaces as well. However, the driver is general-purpose and does not include any application-specific interfaces.

## Rules and Stylesheets

Rules are used to control the synchronization of the driver with eDirectory and the WebsphereMQ-connected application, database, or directory. Rules help DirXML transform an event on a channel input into a set of commands on the channel output.

Style sheets are XSLT documents that define transformations or modifications of XML documents. You can configure and create rules and style sheets using the DirXML ConsoleOne snap-ins. The driver includes the following set of pre-configured rules and style sheets.

Rule or Style Sheet	Description
Placement Rule	Configured only on the Publisher channel
Matching Rule	Operates on both the Publisher and Subscriber channels
Mapping Rule	Configured on the Driver object
InputTransform Style Sheet	Configured on the Driver object
OutputTransform Style Sheet	Configured on the Driver object
Create Style Sheet	Found on the Publisher channel
Event Transform Style Sheet	Found on the Publisher channel

For more information about creating your own rules and style sheets, refer to the DirXML Administration Guide (<http://www.novell.com/documentation>).

## Associations

Associations are made between WebsphereMQ messages and eDirectory objects through a style sheet. A unique ID can be created for records relating to each system connected via WebsphereMQ.

The association attribute received from WebsphereMQ is unique to the interfaced application, based on each driver instance for WebsphereMQ that you install and enable. If other drivers are installed, they should use an association specific to that application.

The association attribute is multi-valued. Therefore, if DirXML is being used to connect multiple applications, all of their associations can be stored on this attribute. The Association attribute can be found in ConsoleOne on the eDirectory object's DirXML property page.

The unique ID association links objects in the WebsphereMQ-connected application to their objects in eDirectory. When an ADD event occurs, the association is made and referenced in the style sheets. This association allows the driver to perform subsequent tasks on the appropriate object.

If a User object is added in eDirectory, the unique user's ID might be populated so that the ID can be linked to the individual's record in the WebsphereMQ-connected application. The matching rule is executed and, if the ID is found within the application, the association is then created. The Matching rule matches the eDirectory user ID and the appropriate unique ID on the host system.

## Setting the Global Association Attribute Value

A “global” association attribute value can be set by modifying a value in the command transformation stylesheet, within the following section:

```
<stylesheet name="Command Transformation Rule">
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="add">
<xsl:copy>
<xsl:apply-templates select="@*" />
<association state="associated">MQAssociation</association>
<xsl:apply-templates select="node()" />
</xsl:copy>
</xsl:template>
<xsl:template match="node()|@">
<xsl:copy>
<xsl:apply-templates select="node()|@" />
</xsl:copy>
</xsl:template>
</xsl:stylesheet>
</stylesheet>
```

The association value can be statically set by modifying the “association state” element, or dynamically computing this value based on additional logic that you can add to the stylesheet to meet an application’s requirements.

## Overriding the Global Association Attribute Value

The global association value that you establish in the stylesheet can be dynamically overridden within a Publisher channel message. This is a powerful feature that is made possible by the open-ended nature of the driver’s design.

To override the global value, you must include an association attribute value inside an XDS message submitted to the Publisher channel as follows:

```
<association> value <association/>
```

## The role of the Subscriber Channel in Setting Associations

In the WebsphereMQ driver, the Subscriber channel plays an important role in setting the association values. At first, this may seem counter-intuitive, but it takes advantage of DirXML engine features and is very efficient.

Due to nature of marrying DirXML to the flexible and asynchronous WebsphereMQ interface, the driver automatically creates a “dummy” association for each “add” event. This association value is then updated according to application needs, either by setting a global, static value in the spreadsheet or dynamically within a Publisher channel message (as described above).

When setting the association attribute value using the stylesheet, a filter for the Subscriber channel must be enabled. This allows the relevant XSL script to “fire” in order to correctly update the association values.

When setting the association attribute value using an XDS message for the Publisher channel, a filter for the Subscriber channel may be either enabled or disabled.

## **Product Components**

The driver contains the following components:

- Pre-configured driver import files
- Driver shim
- Schema extension installer utility (Windows NT/2000 only) that generally complies with RFC 2713 (“Schema for Java Objects, October 1999”)

### **Pre-configured Driver Import File**

The Driver Import File imports pre-configured rules, style sheets, and driver parameters. One pre-configured driver file is provided with the driver for installation on a system that includes DirXML, WebsphereMQ and the driver. Other architectures are also supported via manual configuration.

### **Driver Shim**

The driver shim handles communication between the WebsphereMQ message bus and the DirXML engine.

### **Schema Extension**

The driver installation utility comes packaged with an LDAP schema extension that generally complies with RFC 2713 (“Schema for Java Objects, October 1999”) and is used to store the driver’s Java Messaging Services (JMS) factory classes within eDirectory. This technique enhances the security of the interface.

The driver installer is designed to update this schema when installed on Windows\* NT/2000 server platforms where eDirectory has already been installed and is operational.

Instructions for performing manual installation and schema extension suitable for other platforms can be found in the section “Manually Installing the Driver.”

## **Publishing to eDirectory**

When a WebsphereMQ-connected application is determined to be an authoritative source of user profile data, that system can automatically propagate all Add, Delete, and Modify object event data to eDirectory. The Publisher channel is the component used for propagation.

For data to flow from the WebsphereMQ-connected application, the driver utilizes the Java Messaging Services (JMS) technology to publish XML documents from an application interface. Interfaces can be developed on a custom bases or purchased commercially as “off-the-shelf” connectors from 3rd parties. Both character string and binary data can be transported and WebsphereMQ ensures that the XML document is securely and reliably transported from the host system to eDirectory. More information on using Java Messaging Services with WebsphereMQ can be found at:

<http://www3.ibm.com/software/ts/mqseries/library/manualsa/csqzaw/csqzaw03.htm>

### **Subscribing from eDirectory**

The Subscriber channel of the driver is the component responsible for synchronizing data from eDirectory, which may have been obtained from other authoritative data sources, into a specific WebsphereMQ message queue. This data can then be used to query, update and delete data managed by the host application.

Any combination of attributes, including those containing character string and binary data, can be subscribed from eDirectory by a WebsphereMQ message queue. WebsphereMQ ensures that the XML document is securely and reliably transported from eDirectory to the host application.

## **Installing and Configuring the Driver**

This section helps you do the following:

- Understand prerequisites for the driver
- Planning for installation
- Install driver components
- Configure driver parameters

## Prerequisites

- eDirectory™ 8.6.1.
- Novell® DirXML™ 1.1 or higher.
- It is recommended that you create the Driver Set object before you install the driver. For more information about the Driver Set object, refer to Creating Driver Sets and Objects in the DirXML Administration Guide at:
  - <http://www.novell.com/documentation>
- IBM WebsphereMQ or IBM MQSeries revision level 5.x or higher.
- IBM MA88 SupportPac. This client is freely available through IBM and is provided for most popular server operating systems. You can download the SupportPac from the IBM MQSeries SupportPac site at <http://www-3.ibm.com/software/ts/mqseries/txppacs/ma88.html>.
- ConsoleOne™ 1.3.3.

## Planning for Installation

Before you install and use the driver, you must first plan for the installation. You should choose between a local or remote installation.

### WebSphereMQ Checklist

Establish or identify the queues that will be used either to hold event messages published to eDirectory and/or event messages subscribed from eDirectory

Ensure that SupportPac MA88 is installed on the server hosting the driver. These include the IBM WebsphereMQ classes for java and java messaging services.

Advanced features: If event messages from eDirectory need to be subscribed to by more than one WebsphereMQ application (“MQSeries topic”), ensure that the MA0C SupportPac has been installed and configured properly on the WebsphereMQ server. Also, ensure that the message broker (strmqbrk) been started.



## Planning a Local or Remote Installation

This section explains the difference between a local and remote installation of the driver.

### Local Installation

A local installation installs the driver on a server machine where you have WebsphereMQ , DirXML and eDirectory installed.

### Remote Installation

A remote installation installs the driver on a different computer than the one where DirXML and eDirectory are installed. Remote installation WebsphereMQ security features to ensure data privacy. You will want to use this configuration when WebsphereMQ is installed on a platform other than Windows\* NT/2000, Solaris\* or Linux\*.

## Installing Components

The driver installation program installs the following components on the server:

Component	Description
Driver Shim	A java driver shim that communicates between the WebsphereMQ message bus and the DirXML engine.
Pre-configured DirXML Driver	All eDirectory objects, including the appropriate rules and style sheets for adding, modifying, and deleting or disabling objects. Controls the information being sent from WebsphereMQ to eDirectory and from eDirectory to WebsphereMQ.
Schema Extension Installer	NOTE: Import the driver into eDirectory using the Driver Wizard in ConsoleOne. RFC 2713 Schema extension for managing Java objects within eDirectory.

Although the installation program installs the components, setup is not complete until you properly configure the Driver object and the WebsphereMQ system and MA88 SupportPac.

## Installing The Driver

This section can help you install and configure the driver. Use the Application Driver Creation Wizard to install and configure the driver.

### Installing The Driver

1. Run INSTALL.EXE found on the CD.
2. Accept the license agreement > click Next.
3. Select the components you want to install and the directory paths that each component will be installed to > click Next.
4. Install the pre-configured driver files on the system where ConsoleOne resides.

Component	Filename
Driver Shim	MQDirXML.jar
Pre-configured DirXML Driver	WebsphereMQ.xml
Schema Extension Installer	Install.exe

5. Click Finish.

### Activating The Driver

After you install the driver, you must activate it. Unless activated, the driver will expire after 90 days. When the driver expires, the driver will immediately shut down.

In addition, because DirXML 1.1 is a requirement for the driver, you must make sure DirXML 1.1 is activated as well. Unless activated, DirXML 1.1 will expire after 90 days. If you are using DirXML 1.1, the driver (activated or not) will shut down at the expiration date of DirXML 1.1. To make sure the driver and DirXML 1.1 are both activated, you must purchase a license for both, and apply a Product Activation Credential for both, in order to have the driver work after 90 days.

Activating the driver involves these tasks:

- Generating a Product Activation Request
- Submitting the Product Activation Request
- Activating the driver by applying the Product Activation Credential received from Novell

## Generating a Product Activation Request

You use your Customer ID to generate a Product Activation Request in ConsoleOne. You must have a driver set already set up to generate a Product Activation Request to activate the driver.

**NOTE:** You will need to generate only one Product Activation Request for each tree where the driver is installed for each driver license that you have acquired.

1. Open ConsoleOne. You must use the version of ConsoleOne that you installed with DirXML
2. Choose Wizards > Create a DirXML Activation Request.
3. Select the driver set or browse to a driver set you want to be activated > click Next.
4. If the driver set is not associated with a server or is associated with multiple servers, select a server to associate with a driver set > click Next.

When you select the server, your customer ID and identifying information about the server's tree are stored in the Product Activation Request.

5. Enter your Novell Customer ID > click Next.
6. Do one of these steps
7. Specify the name of the Activation Request file and where you want the file written to > click Next

Or

- Copy the Activation Request file in the text area to the clipboard. You paste the contents of this file in a text area at the Novell Product Activator website.

**NOTE:** Do not edit the content of the Product Activation Request.

8. Click Launch to launch the Novell Product Activator website.

**NOTE:** You need to submit this Product Activation Request to Novell at the Novell Product Activator website (<http://www.novell.com/activator>). At this site, you can either paste the text of the Product Activation Request in a text area or specify the path to the file. If you are working on a machine without a browser, save the file to a diskette to upload at a machine that has a browser.

After you purchase a driver license, Novell will send you a Customer ID via email. If you do not remember or do not receive your Customer ID, please call the Novell Activation Center at 1-800-418-8373 in the U.S. In all other locations, call 1-801-861-8373. (You will be charged for calls made using the 801 area code.)

You will use this Customer ID to create a Product Activation Request. After you create a Product Activation Request, you submit this request to Novell at the Novell Product Activator site (<http://www.novell.com/activator>). After you submit the Product Activation Request, Novell will then send an e-mail containing a Product Activation Credential that you will use to activate the driver.

NOTE: For DirXML 1.1, each driver set needs to be activated, but you can use the same Product Activation Credential to activate all driver sets in the same tree corresponding to each license that you have acquired.

## **Submitting a Product Activation Request**

You submit the Product Activation Request that you generated in ConsoleOne to Novell. You submit this request on the Product Activator website (<http://www.novell.com/activator>).

1. Log in at the Product Activator website (<http://www.novell.com/activator>). You must have an eLogin account to access the Product Activator website. If you don't already have an eLogin account, you must create one when you visit the Product Activator site.
2. Click Browse to specify the path to the Product Activation Request file or paste the text of the Product Activation Request into the text area. If you copied the Product Activation Request to a diskette, make sure you put the request on the computer you are working on.

**IMPORTANT:** Do not edit the content of the Product Activation Request.

3. Click Submit.
4. Mark the product you are activating. You need to activate each line item.
5. Click Submit.

Novell generates a Product Activation Credential based on the Product Activation Request you submitted and sends that credential to you via email.

You activate the driver by installing the Product Activation Credential you received from Novell. You install this file via ConsoleOne. If a tree has multiple driver sets, the Product Activation Credential must be installed for each driver set.

NOTE: Make sure you install the Product Activation Credential on the same tree where you generated the Product Activation Request.

1. Open the email from Novell that contains the Product Activation Credential.

2. Do one of these steps:

- Save the Product Activation Credential file

Or

- Open the Product Activation Credential file > copy the contents of the Product Activation Credential file to your clipboard.

NOTE: Do not edit the contents of the Product Activation Credential file.

3. Open ConsoleOne.

4. Choose Wizards > Install a DirXML Activation.

5. Select the driver set or browse to a driver set > click Next.

6. If the driver set is not associated with a server or is associated with multiple servers, select a server to associate with a driver set > click Next.

7. Do one of these steps:

- Specify where you saved the Activation Credential > click Next.

Or,

- Paste the contents of the Activation Credential in the text area > click Next.

8. Click Finish.

### **Activating the Driver By Applying the Product Activation Credential Received from Novell**

For each of your purchases of a DirXML license, you can see the Product Activations you have installed for those licenses for DirXML and DirXML drivers in ConsoleOne.

1. Open ConsoleOne.

2. Right-click the Driver Set object with activations you want to view > click Properties.
3. Click DirXML > Activation.

DirXML and DirXML driver activations display on this page. You can view the activation text or remove the activation from DirXML or any drivers. (You might want to remove an activation if it contains an error.)

## Post Installation Tasks

Now that you have installed the driver, you will need to do the following:

- Import the pre-configured driver file.
- Configure the Remote Loader on the remote system.

### Importing the Pre-configured Driver

The Application Driver Creation Wizard will help you import a pre-configured driver file. This file will create and configure objects needed in eDirectory to make the driver work properly.

1. Right-click Network Neighborhood > click NetWare Connections.

Ensure the following:

- You are authenticated to the eDirectory server you are installing to.
- The server where the driver will be running is set as the primary server.

2. In ConsoleOne, click Wizards > Create a New Application Driver.
3. Select the driver set where you want the driver installed > click Next.
4. Enter the following driver set properties as prompted:

- Name
- Context
- Whether to create a new partition on the driver set

5. Select WebsphereMQ.xml pre-configured driver file > click Next.

Typically, this file is in the C:\NOVELL\CONSOLEONE\1.2\SNAPINS directory.

6. Follow on screen prompts to enter information about eDirectory structure and driver connectivity.
7. Click Next and enter the following information as prompted:

## DirXML Driver for IBM WebsphereMQ

- The name you want to use for the driver
- The driver password

The shim authentication password is used for logging into WebsphereMQ. The driver password is used to authenticate to the DirXML server.

8. When the driver import is finished, click Yes to define security equivalences on the driver.

- Click Add > select an object with Admin rights (or any other rights you want the driver to have)

- Click Apply > Close.

9. Exclude Administrative Roles from replication > click Apply > click Close.

10. Click Finish.

## Configuring the Remote Loader on the Remote System

The Remote Loader can be configured to use a SSL connection for secure data transfer between eDirectory and the WebsphereMQ system. This section explains what needs to be completed to establish an SSL connection.

- Generate a security certificate on the eDirectory container.
- Create an organizational trusted root certificate.
- Validate the certificate file.
- Run Remote Loader wizard and configure options.
- Configure Driver Object properties in ConsoleOne.

## **Generating a Security Certificate on the eDirectory Container**

In ConsoleOne, select the Container in which the eDirectory server resides.

1. Right-click the container > select New Object > create the NDSPKI: Key Material object.
2. Enter a name for the certificate > select the creation method > click Next.  
  
IMPORTANT: Write down the name of the certificate. You will need this information when configuring the driver.
3. Choose the Standard Creation Method > click Next to view the Confirmation Screen.
4. Click Finish to generate a certificate for the eDirectory container.

## **Create an Organizational Trusted Root Certificate**

In ConsoleOne, browse to the Security container and locate the Organizational CA for the tree.

1. Right-click the certificate object > click Properties.
2. Click the Certificate tab and go to the Self-Signed Certificate option.
3. Select Export > click Next.
4. Select the Base64 option > click Next.
5. Click Finish.
6. Click Validate to verify the Certificate's validity.
7. Copy the certificate file to the Remote Loader directory on the Remote Loader host system.



## Using the Wizard to Configure Remote Loader Properties on the Remote Loader Host Server

1. From a DOS command prompt on the Remote Loader host system, change directory to the Remote Loader directory and enter: `dirxml_remote`
2. Click Next to begin the wizard.
3. Following the wizard prompts, you should configure the following items:

- Command Port. Used to differentiate between instances of Remote Loader.
- Increments. The first instance of Remote Loader defaults to port 8000 with each additional instance incrementing in number by one.
- Configuration File. The configuration file is a filename that is automatically generated. Verify the filename > click Next.

DirXML Driver. Select the Java option. Enter the case-sensitive java class name: `com.novell.nds.dirxml.driver.mq.MqDirXMLDriverShim`

- Connection to DirXML. Enter the port number. The default is 8090, but you can enter a different, unique port number. Select the IP address to which you want to apply changes. You should select the "All IP addresses" option.
- Certificate. Select Use SSL. Browse to the certificate generated in the previous section. The browser automatically looks for Base64 files > click Next.
- Tracing. Choose a level of tracing. In addition to trace files, you can log trace events to a log file. Configure this log file if you want. Trace level 0 provides no trace application window.
- Install as a service. When you install Remote Loader as a service, it will automatically launch when the machine boots.
- Passwords. Enter a password for the remote loader and driver object.

**IMPORTANT:** Write down these passwords. You will need them for the next step in driver configuration.

- Summary. Click Next to continue with the DirXML Remote Loader configuration.

4. Click Finish to complete the configuration. When prompted to start the DirXML Remote Loader, click Yes.

## Starting the Remote Loader if Installed as a Service

If you configured Remote Loader to run as a service, you can start it by completing the following steps:

1. Open the Windows Service Control Manager.
2. Select the DirXML Loader (com.novell.nds.dirxmldriver) > click the Startup Option > click Start.

## Starting the Remote Loader Manually

1. From a command prompt (wherever Remote Loader is installed), enter:

```
-dir *.txt
```

This will display the name of the configuration file (for example, config8001.txt).

2. Enter the following:

```
dirxml_remote -config <configfilename.txt>
```

For example:

```
dirxml_remote -config config8001.txt
```

You can open the Windows Task Manager to verify that Remote Loader is running.

Note: The driver retrieves Java Messaging Services classes from the directory prior to initialization. If the remote driver exists in a DMZ or separated from eDirectory by a firewall, the default TCP ports 8090 for the remote loader, 389 for LDAP and optionally 636 for LDAP/S must be open to the server that hosts the driver.

## Configuring Driver Object Properties for Remote Loader

1. In ConsoleOne, right-click the WebSphereMQ Driver object > click Properties.
2. Click the Authentication tab.
3. Enter the following Remote Loader connection parameters with a space between each parameter:

Hostname. Specifies the address or name of the machine on which the Remote Loader will run. For example, hostname=192.168.0.1

Port. Specifies the port on which the Remote Loader will accept connections from the remote interface shim. For example, port=8090

KMO. Specifies the key name of the Key Material Object containing the keys and certificate used for SSL. For example: kmo=remotedrivercert.

NOTE: If SSL is enabled, you need to enter the hostname, port, and KMO information.

4. If SSL is not enabled, you only need to enter the hostname and port information.
5. Enter passwords for the application and the Remote Loader.
6. Click the Driver Parameters tab > scroll to the Publisher Settings.
7. Verify that the Publisher queue is configured. This queue contains XML documents that will be processed against eDirectory. Refer to Configuring Driver Object Properties for more information.

## Configuring Driver Parameters

This section helps you do the following:

Configure Driver object properties

In addition to the following driver-specific configuration parameters, you should configure basic DirXML Driver object fields. For more information regarding these basic configuration fields, see the DirXML Administration Guide.

### Configuring Driver Parameters

In ConsoleOne, right-click the Driver object > click Properties > Properties. Configure the following:

Tabs/ Parameters	Description
<b>Driver Module</b>	
Java	This is the type and name of the DirXML driver: <code>com.novell.nds.dirxml.driver.mq.MqDirXMLDriverShim</code>
Byte Messaging	These are the transmission options that can be supported, bytestream messaging or text messaging. Input “True” for sending a bytestream, or leave blank for the default text option.
<b>Authentication</b>	
Authentication	The driver authenticates with an WebsphereMQ server via a server connection channel. If the server-connection channel’s MCAUSER attribute is set to “” (default), then the user id that started the MqDirXMLDriver is used to authenticate the connection. This must be a valid user id on the system that is running the WebsphereMQ server and the user id must be included in the mqm group.  To authenticate under a different user id and password (which also must be a valid object in the mqm group), use ConsoleOne to browse to the DirXML driver set object, select the ‘MqDirXMLDriver’, open the properties, select the Authentication tab, enter in an Authentication ID and Application Password.
Authentication Context	Optional directory context of the authentication object.
<b>Driver Settings</b>	
Queue Manager	This element specifies the name of a queue manager to connect to or the DN of the MQQueueConnectionFactory java object store on a NDS eDirectory. If a dn is specified, the <i>Provider URL</i> , <i>Security Principal</i> and <i>Security Credentials</i> elements must contain connection values.
Message Broker	This element specifies the name of message broker to connect to or the DN of the MQTopicConnectionFactory java object store on a NDS eDirectory. If a dn is specified, <i>Provider URL</i> , <i>Security Principal</i> and <i>Security Credentials</i> elements must contain connection

	values. The message broker feature is only available on the subscriber channel.
Security Principal	This element specifies the DN of a user that has the rights to access the NDS server to retrieve either the MQTopicConnectionFactory or MQQueueConnectionFactory object.
Security Credentials	This element specifies the password of the user DN contained in the sp element.
Security Protocol	This element enables a ssl connection to be created while connecting to the NDS server. The only valid value is 'ssl', otherwise leave empty.
Verbose	This element enables verbose logging to the DirXML trace to enhance debugging. The only valid value is 'true', otherwise leave empty.
<b>Subscriber Settings</b>	
Queue Sender	This element specifies the queue to send NDS event information to.
Queue Reply	This element specifies the queue to receiver replys from the WebsphereMQ application regarding NDS event information. The returned messages must be in XDS format. The only valid values are blank, noreply messages will be returned from the WebsphereMQ application, "" (a literal double quote), a temporary queue will be created to receive reply messages from a WebsphereMQ application, or an existing WebsphereMQ queue.
Reply Queue Timeout Period	This element specifies that number of seconds the subscriber channel will wait for a reply message from the WebsphereMQ application. Not enabled with pub/sub messaging. The only valid value is #, otherwise leave empty, the default of 5 seconds will be used.
Topic Sender	This element specifies the topic to publish NDS event information to.
Message Persistence	This element specifies whether messages will be non-persistent. Default is persistent. The only valid value is 'true', otherwise leave empty.
<b>Publisher Settings</b>	
Queue Receiver	This element specifies an existing WebsphereMQ queue to receive XML formatted DirXML commands from.
JMS Client Type	This element specifies that the message format will be JMS enabled. Default is WebsphereMQ formatted messages. The only valid value is 'true', otherwise leave empty.

## Troubleshooting the Driver

This section contains potential problems and error codes you may encounter while configuring or using the driver.

### Using the DSTrace Utility

You can troubleshoot the driver using the DSTrace utility. You will want to configure the utility's options by selecting Edit > Properties > "DirXML Drivers."

For each event or operation received, the driver returns an XML document containing a status report. If the operation or event is not successful, the status report also contains a reason, a text message describing the error condition. If the result is "fatal," the driver shuts down.

### Common Errors

Driver Load Errors	Solution
java.lang.ClassNotFoundException:com.novell.nds.dirxml.driver.mq.MqDirXMLDriverShim	This is a fatal error that occurs when MqDirXMLDriverShim.jar is not installed properly. You should ensure that the file is in the proper location for either a local or remote loader configuration.
java.lang.ClassNotFoundException:com.novell.nds.dirxml.driver.mq.MqDirXMLDriver	This is a fatal error that occurs when the class name for the MqDirXMLDriverShim.jar is incorrect. You should ensure that the "Java" class name is set on the Driver Module tab in a local installation and that the "-class" parameter is set in a remote loader configuration.  The proper class name is: novell.nds.dirxml.driver.mq.MqDirXMLDriverShim
Driver Initialization Errors	Solution
'no mqjbnd02 in java.library.path'	Ensure that the <MQSeries_Install_Path>\java\lib is in your System path.
java.lang.ArrayIndexOutOfBoundsException # > #	There is a problem w/the JDK version when trying to retrieve MQQueueConnectionFactory object via JNDI. This does not apply to a local WebSphereMQ connection. Replace the <Novell_Install_Path>\NDS\jre\lib\rt.jar with a more JDK version 1.3.1 and above rt.jar
mqjms1046: The character set 437 is not supported.	Set the sending client ccsid to 819 (the default).
MQJMS2008: failed to open MQ queue.	Ensure the specified queue in the 'Driver Parameters Xml' exists on the WebsphereMQ server.
MQJMS2005: failed to create MQQueueManager for <>	Ensure the WebsphereMQ server is running.
MQJMS2006: MQ problem: com.ibm.mq.MQException: Completion Code 2, Reason 2085	Regarding MQ JMS Pub/Sub calls, ensure to create the mqjms system queues. Example: C:\<MQSeries_Install_Path>\java\bin\runmqsc <queue_manager> < MQJMS_PSQ.mqsc

## Manually Installing the Driver

This section contains potential problems and error codes you may encounter while configuring or using the driver.

### Installation Checklist

WebsphereMQ setup	<p>Establish or identify the queues that will be used either to hold event messages destined for eDirectory and/or event messages from the NDS eDirectory</p> <p>If event messages from eDirectory need to be published to a topic, ensure that the MAOC SupportPac has been installed and configured properly on the WebsphereMQ server. Also, ensure that the message broker been started (strmqbrk).</p>
Java Messaging Services:	Ensure that SupportPac MA88 is installed on the WebsphereMQ server. These include the IBM WebsphereMQ classes for java and java messaging services.
eDirectory Setup:	Ensure that eDirectory version 8.6.1 or later is installed and running.
DirXML:	Verify that the DirXML driver-set object exists in eDirectory.
DSTrace:	<p>Ensure that the DirXML-DriverTraceLevel attribute is set and added to the driver-set. Ensure that the DirXML-MQ Driver is installed in the driver-set object.</p> <p>Ensure that the MqDirXMLDriver driver parameters have been set</p> <p>Set the DSTrace screen to track DirXML operationsRemote Operations</p>
Remote Operations:	Ensure that the WebsphereMQ server and NDS eDirectory are visible to each other over an IP network.
Schema:	<p>Ensure that the eDirectory schema has been extended to support storing Java objects.</p> <p>Ensure that the MQQueueConnectionFactory and/or MQTopicConnectionFactory objects have been created and stored within eDirectory.</p>

## MA88 SupportPac Installation Note

After installing the MA88 support Pac, the 'java' directory should reside within your <MQSeries\_Install\_Path>. Example:

c:\program files\ibm\mqseries\java

If the WebsphereMQ server resides on the same machine as eDirectory, be sure to include the <MQSeries\_Install\_Path>\java\lib in your system path.

This is required to allow the DirXML-MQ driver to find all WebsphereMQ libraries.

If the WebsphereMQ server(s) will be remote to the eDirectory, be sure to copy the following files (located in the on the WebsphereMQ server's <MQSeries\_Install\_Path>\java\bin directory ) to the <Novell\_Install\_Path>\NDS\lib directory on the eDirectory server:

jms.jar  
com.ibm.mq.jar  
com.ibm.mqjms.jar  
connection.jar

This is required to the DirXML-MQ driver to communicate remotely with the WebsphereMQ server.

## Installing the DirXML-MQ Driver

- Place the MQDIRXML.JAR file into the <Novell\_Install\_Path>\NDS\lib directory. Open the WebsphereMQ.XML file in edit mode.
- The WebsphereMQ.XML file holds skeleton configuration information to create the MqDirXMLDriver driver object within the driver-set object included in the NDS eDirectory.
- The WebsphereMQ.XML file will need to be modified to reflect the proper DN= of the driver-set object within eDirectory.

Example: <driver-configuration  
dn="MqDirXMLDriver.Tedsdev.servers.spec.Teds" driver-set-  
dn="Tedsdev.servers.spec.Teds" name="MqDirXMLDriver">



## Creating a driver set

If a driver-set doesn't yet exist in eDirectory, create one prior to installing the driver for WebsphereMQ.

From ConsoleOne, select the Wizards from the menu bar.

Select the Create new Application Driver or existing driver set.

Enter a name for the driver-set object, select the context where the driver-set will be created, and select the server.

Select the Import Preconfigured Driver button.

Select the WebsphereMQ.XML file.

Optionally, elect to password enable the driver.

Select No Password for the shim authorization password.

Select Yes to define the security equivalences on the driver.

Select Security equal to <user account> to associate a user account with administrative rights to the DirXML-MQ driver.

Click on the DirXML tab to select a user account with appropriate access privileges. Click Apply.

Press the Next and Finish button.

Right click on the driver set object and select the Properties option.

Select the driver and click on the Start button.

Configure the driver parameters by right clicking the MqDirXMLDriver, select properties, then select the driver parameters XML tab.

To enable DSTrace, right click on the driver-set object, select the "Other" tab, add the DirXML-DriverTraceLevel attribute and set that attribute to 5. Apply changes.

## Extending eDirectory to support Java object storage

Launch ConsoleOne and login as an administrator

Create and place 'ou=JavaObjects' in the desired tree structure.

Open Tools - Schema Manager.4. Select the Attributes tab

Check whether the following attributes already exist. If needed, create them using the following information:

NAME 'javaClassName', ASN1 ID: 1.3.6.1.4.1.42.2.27.4.1.1  
SYNTAX Case Ignore String, SINGLE-VALUE

NAME 'javaCodebase', ASN1 ID: 1.3.6.1.4.1.42.2.27.4.1.6  
SYNTAX Case Ignore String

NAME 'javaFactory', ASN1 ID: 1.3.6.1.4.1.42.2.27.4.1.4  
SYNTAX Case Ignore String, SINGLE-VALUE

NAME 'javaReferenceAddress', ASN1 ID: 1.3.6.1.4.1.42.2.27.4.1.3  
SYNTAX Case Ignore String

NAME 'javaSerializedData', ASN1 ID: 1.3.6.1.4.1.42.2.27.4.1.7  
SYNTAX Stream, SINGLE-VALUE6.

Switch to the Classes tab.

Check whether the following attributes already exist. If needed, create them using the following information:

NAME 'javaContainer', ASN1 ID: 1.3.6.1.4.1.42.2.27.4.2.1  
Effective Class +Container, INHERIT top  
MUST ( CN ) CONTAINER CLASS Organizational Unit

NAME 'javaObject', ASN1 ID: 1.3.6.1.4.1.42.2.27.4.2.4  
Non-Effective Class, INHERIT top  
MUST ( javaClassName )  
MAY ( javaCodebase )

NAME 'javaNamingReference', ASN1 ID: 1.3.6.1.4.1.42.2.27.4.2.7  
AUXILIARY, INHERIT javaObject  
MAY ( javaReferenceAddress | javaFactory )

NAME 'javaSerializedObject', ASN1 ID: 1.3.6.1.4.1.42.2.27.4.2.5  
INHERIT javaObject, AUXILIARY  
MUST ( javaSerializedData )

## Creating the WebsphereMQ factory objects

The JMSAdmin utility binds to NDS e-Directory using information contained in the JMSAdmin.config file and creates the WebsphereMQ factory objects that allow remote applications to use WebsphereMQ services without WebsphereMQ being installed locally to that application.

The JMSAdmin utility is part of the MA88 MQSeries Supportpac and is found in the MQSeries\_Install\_Path >\java\bin directory. Additional information regarding using the JMSAdmin utility can be found in the IBM documentation resource called MQ Series Using Java at:

<http://www4.ibm.com/software/ts/mqseries/library/manualsa/csqzaw/csqzaw03.htm>

Ensure that the jms.jar and com.ibm.mqjms.jar files are included in your CLASS-PATH.

Edit the JMSAdmin.config file. This file is read by the JMSAdmin utility and contains JNDI information required to connect to the ldap source.

Example of creating a queue connection factory: (NOTE: When defining the QCF(name\_of\_object\_stored\_in\_nds), don't use periods.)

```
<MQSeries_Install_Path>\java\bin\JMSAdmin
User DN: <the_dn_of_a_user_that_can_create_objects>
User Password: <password>
```

```
InitCtx> DEFINE QCF(name_of_object_stored_in_nds) QMGR(queue.manager.name)
DESC(The default Q Manager)
TRAN(CLIENT) HOST(192.168.1.10) PORT(1414)
```

## **Interacting with the driver for WebsphereMQ**

WebsphereMQ-enabled applications that need to exchange data with the driver must use messages created in XML format. The driver will pass XML messages to the DirXML engine and return the results in XML.

The desired XML message format should comply with Novell's nds.dtd document type definition. If the WebsphereMQ application is unable to work with the XML format specified within the nds.dtd, it can use any style of XML, however, a DirXML transformation style sheets must be created to allow the driver to do the conversion of data from one format to another.

The driver for WebsphereMQ can work with either JMS style messages or traditional MQSeries style messages. However, each instance of the driver for WebSphereMQ can only work with one type of message. By default, MQSeries style messages are enabled.

## Appendix A Driver Parameters XML

```
<driver-options>
  <qmgr display-name="Queue Manager">MY.QMGR</qmgr>
  <msgbrk display-name="Message Broker"/>
  <sp display-name="Security Principal">cn=Admin,o=TWC</sp>
  <purl display-name="Provider Url">ldap://192.168.1.10:389</purl>
  <sc display-name="Security Credentials">password</sc>
  <ss display-name="Security Protocol"/>
  <vb display-name="Verbose">true</vb>
</driver-options>

<subscriber-options>
  qsend display-name="Queue
    Sender">SYSTEM.DEFAULT.LOCAL.QUEUE</qsend>
  <qreply display-name="Queue Reply">""</qreply>
  <qreplytimeout display-name="Reply Timeout Period in Sec."/>
  <tpub display-name="Topic Publisher"/>
  <nonper display-name="Non-Persistent Delivery Mode"/>
  <jmsclient display-name="JMS Client Type"/>
</subscriber-options>

<publisher-options>
  <qrec display-name="Queue Receiver"/>
  <jmsclient display-name="JMS Client Type"/>
</publisher-options>
```