

FUSE™ Message Broker

FUSE™ Message Broker Administration Guide

Version 5.2
December 2008

FUSETM Message Broker Administration Guide

Progress Software

Version 5.2

Published 02 Dec 2008

Copyright © 2008 IONA Technologies PLC , a wholly-owned subsidiary of Progress Software Corporation.

Legal Notices

Progress Software Corporation and/or its subsidiaries may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this publication. Except as expressly provided in any written license agreement from Progress Software Corporation, the furnishing of this publication does not give you any license to these patents, trademarks, copyrights, or other intellectual property. Any rights not expressly granted herein are reserved.

Progress, IONA, IONA Technologies, the IONA logo, Orbix, High Performance Integration, Artix, FUSE, and Making Software Work Together are trademarks or registered trademarks of Progress Software Corporation and/or its subsidiaries in the US and other countries.

Java and J2EE are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. CORBA is a trademark or registered trademark of the Object Management Group, Inc. in the US and other countries. All other trademarks that appear herein are the property of their respective owners.

While the information in this publication is believed to be accurate Progress Software Corporation makes no warranty of any kind to this material including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Progress Software Corporation shall not be liable for errors contained herein, or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

All products or services mentioned in this manual are covered by the trademarks, service marks, or product names as designated by the companies who market those products.

No part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, photocopying, recording or otherwise, without prior written consent of IONA Technologies PLC. No third-party intellectual property right liability is assumed with respect to the use of the information contained herein. IONA Technologies PLC assumes no responsibility for errors or omissions contained in this publication. This publication and features described herein are subject to change without notice. Portions of this document may include Apache Foundation documentation, all rights reserved.

Table of Contents

Preface	9
The FUSE Message Broker Library	10
Open Source Project Resources	11
Document Conventions	12
Using the Admin Tool	15
Introducing the Admin Tool	16
Starting a Broker	18
Creating a Broker	19
Shutting Down a Broker	21
Listing Available Brokers	22
Querying a Broker	23
Browsing Messages on a Destination	24
Auditing the Persistence Journal	26
Deploying the Broker	29
Section Title	30
Subsection Title	31
Managing the Broker	33
Section Title	34
Subsection Title	35
Using Visualization	37
Section Title	38
Subsection Title	39
Securing the Broker	41
Section Title	42
Subsection Title	43
Index	45

DRAFT

List of Figures

1. Directory Structure Generated by activemq-admin create 19

DRAFT

DRAFT

List of Tables

1. JMS Headers 24

DRAFT

DRAFT

Preface

The FUSE Message Broker Library	10
Open Source Project Resources	11
Document Conventions	12

DRAFT

The FUSE Message Broker Library

The FUSE Message Broker documentation library consists of the following books:

- [Installing FUSE™ Message Broker](#) discusses the requirements and procedures for installing FUSE Message Broker
- [Getting Started with FUSE™ Message Broker](#) provides an overview of the central concepts behind FUSE Message Broker and walks you through a simple example.
- [Connectivity Guide](#) explains the different wire protocols and transports that FUSE Message Broker supports.
- [Using FUSE™ Message Broker's Persistence Features](#) describes how to enable message persistence using the AMQ Message Store or a relational database in FUSE Message Broker.

Open Source Project Resources

Apache CXF

Web site: <http://cxf.apache.org/>

User's list: <user@cxf.apache.org>

Apache Tomcat

Web site: <http://tomcat.apache.org/>

User's list: <users@tomcat.apache.org>

Apache ActiveMQ

Web site: <http://activemq.apache.org/>

User's list: <users@activemq.apache.org>

Apache Camel

Web site:
<http://activemq.apache.org/camel/enterprise-integration-patterns.html>

User's list: <camel-user@activemq.apache.org>

Apache ServiceMix

Web site: <http://servicemix.apache.org>

User's list: <users@servicemix.apache.org>

Document Conventions

Typographical conventions

This book uses the following typographical conventions:

<code>fixed width</code>	<p>Fixed width (Courier font) in normal text represents portions of code and literal names of items such as classes, functions, variables, and data structures. For example, text might refer to the <code>javax.xml.ws.Endpoint</code> class.</p> <p>Constant width paragraphs represent code examples or information a system displays on the screen. For example:</p> <pre>import java.util.logging.Logger;</pre>
<i>Fixed width italic</i>	<p>Fixed width italic words or characters in code and commands represent variable values you must supply, such as arguments to commands or path names for your particular system. For example:</p> <pre>% cd /users/YourUserName</pre>
<i>Italic</i>	Italic words in normal text represent <i>emphasis</i> and introduce <i>new terms</i> .
Bold	Bold words in normal text represent graphical user interface components such as menu commands and dialog boxes. For example: the User Preferences dialog.

Keying conventions






This book uses the following keying conventions:

No prompt	When a command's format is the same for multiple platforms, the command prompt is not shown.
%	A percent sign represents the UNIX command shell prompt for a command that does not require root privileges.
#	A number sign represents the UNIX command shell prompt for a command that requires root privileges.
>	The notation > represents the MS-DOS or Windows command prompt.
...	Horizontal or vertical ellipses in format and syntax descriptions indicate that material has been eliminated to simplify a discussion.
[]	Brackets enclose optional items in format and syntax descriptions.
{ }	Braces enclose a list from which you must choose an item in format and syntax descriptions.

	In format and syntax descriptions, a vertical bar separates items in a list of choices enclosed in {} (braces).
--	---

Admonition conventions

This book uses the following conventions for admonitions:

	Notes display information that may be useful, but not critical.
	Tips provide hints about completing a task or using a tool. They may also provide information about workarounds to possible problems.
	Important notes display information that is critical to the task at hand.
	Cautions display information about likely errors that can be encountered. These errors are unlikely to cause damage to your data or your systems.
	Warnings display information about errors that may cause damage to your systems. Possible damage from these errors include system failures and loss of data.

DRAFT

Using the Admin Tool

FUSE Message Broker provides a command-line tool that allows you to perform a number of common administration tasks.

Introducing the Admin Tool	16
Starting a Broker	18
Creating a Broker	19
Shutting Down a Broker	21
Listing Available Brokers	22
Querying a Broker	23
Browsing Messages on a Destination	24
Auditing the Persistence Journal	26

Introducing the Admin Tool

Overview

The FUSE Message Broker Admin tool allows you to perform the following tasks from the command line:

- Start and stop a broker
 - Create a runnable instance of a broker
 - List all available brokers in the specified JMX context
 - Display a selected broker's components and attributes
 - Display selected messages on a specified destination
 - View records stored in the performance journal
-

Running the Admin tool

To run the Admin tool:

1. In a command prompt or terminal window, change directory to the FUSE Message Broker installation directory.
2. Change directory to the `bin` directory.
3. Type the following:

- Windows:

```
activemq-admin
```

- UNIX:

```
./activemq-admin
```

A list of tasks that you can call using the Admin tool is returned. These tasks are:

- **start**
- **create**
- **stop**

- **list**
- **query**
- **bstat**
- **browse**
- **journal-audit**

Each task is described in the following sections. For additional information, see the [Apache ActiveMQ Command Line Tools Reference](http://activemq.apache.org/activemq-command-line-tools-reference.html) [http://activemq.apache.org/activemq-command-line-tools-reference.html].

Getting help

To display help on the Admin tool type the **activemq-admin** command followed by any of the following:

- **-h**
- **-?**
- **--help**

To display help on a particular task, type **activemq-admin *taskname*** followed by **-h**, **-?**, or **--help**.

Displaying version information

To display version information for FUSE Message Broker, type either of the following:

activemq-admin --version

activemq --version

Starting a Broker

Overview

You can use the Admin tool to start an instance of FUSE Message Broker.

For example, to start a broker under the current FUSE Message Broker installation directory, type:

activemq-admin start



Tip

Instead of running **activemq-admin start**, you can simply run the shortened version, **activemq**.

Specifying configuration

When you run either **activemq** or **activemq-admin start**, there are two main ways of specifying configuration for the broker:

- Using XBean XML
- Using a URI syntax

Both methods are discussed in ????.

Creating a Broker

Overview

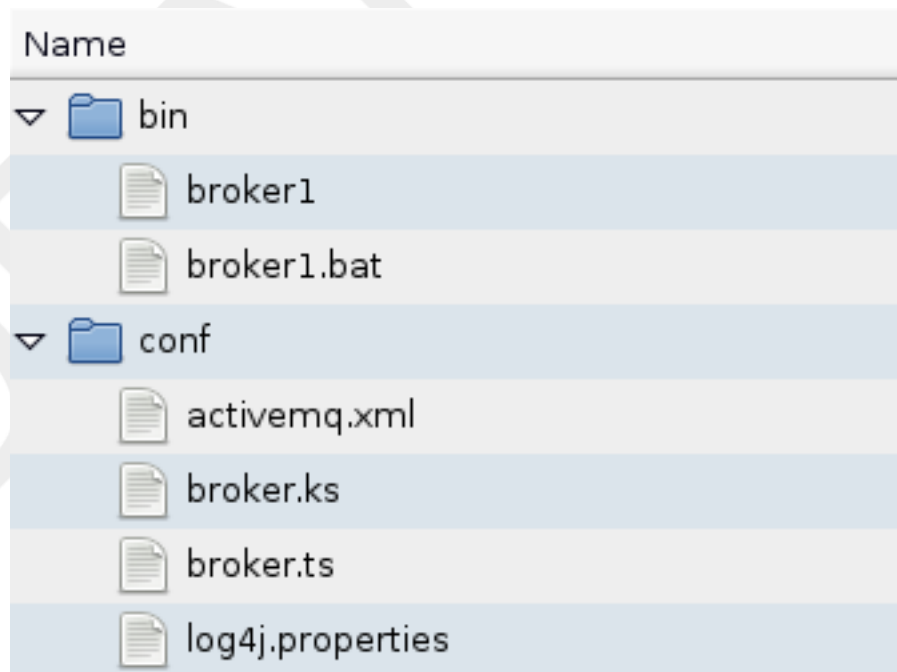
The Admin tool's **create** task allows you to create a directory structure for an instance of a broker. This is useful if you want to avoid FUSE Message Broker's default directory structure whereby data and binaries are stored under the same location.

For example, the following command creates a `broker1` directory under your home directory:

```
./activemq-admin create ~/broker1
```

The `broker1` directory contains the minimal start scripts and configuration files needed to run an instance of FUSE Message Broker, as shown in [Figure 1 on page 19](#).

Figure 1. Directory Structure Generated by `activemq-admin create`



You run the **broker1** script to start an instance of FUSE Message Broker, named `broker1`. This adds a `data` directory to below the `broker1` directory, where the messages handled by `broker1` are stored.

DRAFT

Shutting Down a Broker

Stopping a broker in the default JMX context

You can shut down a broker running in the default JMX context using the **stop** task, as follows:

```
activemq-admin stop broker1
```



Note

The **stop** task requires JMX to be enabled, which it is by default.

Shutting down all running brokers

To shut down all the brokers running in the default JMX context (service:jmx:rmi:///jndi/rmi://localhost:1099/jmxrmi), type the following:

```
activemq-admin stop --all
```

Shutting down a broker in a non-default JMX context

If the broker is running in a JMX context other than the default, you need to pass the `--jmxurl` option to the **stop** task, as follows:

```
activemq-admin stop --jmxurl  
service:jmx:rmi:///jndi/rmi://localhost:8050/jmxrmi
```

Listing Available Brokers

Listing brokers in the default JMX context

To list all the brokers running in the default JMX context, type:

activemq-admin list

Listing brokers in a non-default JMX context

If the broker is running in a JMX context other than the default, you need to pass the `--jmxurl` option to the **list** task, as follows:

**activemq-admin list --jmxurl
service:jmx:rmi:///jndi/rmi://localhost:8050/jmxrmi**

Querying a Broker

Overview

The **query** task allows you to query the current JMX context for broker statistics and information.

To display all attributes and object name information of all registered MBeans in the default JMX context, type:

activemq-admin query

Filtering specific MBeans by type

You can query a specific MBean object based on its type and an identifier using the `-Q` option as follows:

activemq-admin query -Q`MBeanType=Name`

For example, to display all the attributes and object name information of the destination topic 'TEST.FOO', type:

activemq-admin query -QTopic=TEST.FOO

To display all the brokers in a context whose name ends in "host", type:

activemq-admin query -QBroker=*host

To display the attributes and object name information of all registered queues, type:

activemq-admin query -QQueue=*

Excluding MBeans from a query by type

To exclude a specific MBean object based on its type and an identifier, use the `-xQ` option as follows:

activemq-admin query -xQ`MBeanType=Name`

For example, to display all the attributes and object name information of all topics ending with '.FOO' except those that also begin with 'ActiveMQ.Advisory.', type:

activemq-admin query -QTopic=*.FOO -xQTopic=ActiveMQ.Advisory.*

Browsing Messages on a Destination

Overview

You can browse the messages on a selected destination using the **browse** task and a broker URL, as follows:

activemq-admin browse --amqurl *URL DestinationName*

For example, to print the JMS message header, custom message header, and message body of all the messages in the queue 'TEST.FOO' on the localhost broker, type:

activemq-admin browse --amqurl tcp://localhost:61616 TEST.FOO

Specifying message attributes

You can specify a particular group of message attributes to view using the **-v** flag, as follows:

-Vheader

Shows all the standard JMS message headers.

-Vcustom

Shows all the custom fields added to each JMS message.

-Vbody

Shows the body of the message

For example, to show on the message body of messages in the queue TEST.FOO on localhost, type:

activemq-admin browse --amqurl tcp://localhost:61616 -Vbody TEST.FOO

Filtering messages

You can filter specific messages using the **--msgsel** option combined with one of the following JMS message headers.

Table 1. JMS Headers

Header Name	Header Type
JMSCorrelation ID	string
JMSDelivery Mode	int (1-Non-Persistent, 2-Persistent)
JMSDestination	javax.jms.Destination
JMSExpiration	long

Header Name	Header Type
JMSMessageID	String
JMSPriority	int
JMSRedelivered	boolean
JMSReplyTo	javax.jms.Destination
JMSTimestamp	long
JMSType	String

For example, to view a message with an ID ending in "10", type:

```
activemq-admin browse --amqurl tcp://localhost:61616 --msgsel  
JMSMessageID='*:10' TEST.FOO
```

To view messages with a priority of 3, type:

```
activemq-admin browse --amqurl tcp://localhost:61616 --msgsel  
JMSPriority=3 TEST.FOO
```

Auditing the Persistence Journal

Overview

To view the persistence journal data log files, use the **journal-audit** task. Velocity Template Language (VTL) expressions allow you to control how the records are displayed. See [Using the AMQ Message Store](#) in the *Using FUSE™ Message Broker's Persistence Features* for details.

You can run the command against the journal archive directory and the active journal directory, as follows:

```
activemq-admin journal-audit Options JournalDirectory
```

Displaying message records

Message records are created every time a producer sends a persistent message to the broker. The message is recorded in the journal even if its transaction is rolled back.

Use the following syntax to display message records:

```
activemq-admin journal-audit --message-format=VTLExpression  
JournalDirectory
```

The default VTL expression is as follows:

```
${brandId}|${brandId}|${type}|${recordId}|${recordId}|${recordId}|${body}
```



Note

Some operating systems require you to place the entire VTL expression inside single quotes.

Displaying topic ack records

A topic ack records that a durable subscription for a topic has acknowledged a set of messages.

Use the **--topic-ack-format=VTLExpression** option to display topic ack records:

The default VTL expression is as follows:

```
${brandId}|${brandId}|${type}|${recordId}|${recordId}|${recordId}|${body}
```

Displaying queue ack records

A queue ack records that a consumer for a queue has acknowledged a message.

Use the **--queue-ack-format=VTLExpression** option to display queue ack records:

The default VTL expression is as follows:

```
${location.dataFileId},${location.offset}|${type}|${record.destination}|${record.messageAckLastMessageId}
```

Displaying transaction records

Transaction records are used to record transaction related actions like commit and rollback.

You can use the **--transaction-format=VTLExpression** to display transaction records:

The default VTL expression is:

```
${location.dataFileId},${location.offset}|${type}|${record.transactionId}
```

Displaying trace records

Trace records are informational messages stored in the journal that assist in auditing. For example, a trace message is recorded whenever the broker is restarted or when the long term store is checkpointed.

You can display trace records using the **--trace-format=VTLExpression** option.

The default VTL Expression is:

```
${location.dataFileId},${location.offset}|${type}|${record.message}
```

Filtering records

The journal-audit task allows you to filter out records using a SQL like WHERE syntax, as follows:

```
activemq-admin journal-audit --where=VALUE
```

For example:

```
activemq-admin journal-audit --where="type='ActiveMQTextMessage' and location.dataFileId > 2"
```

DRAFT

Deploying the Broker

This chapter is about some stuff.

Section Title	30
Subsection Title	31

DRAFT

Section Title

Subsection Title 31

DRAFT

Subsection Title

Block Title

DRAFT

DRAFT

Managing the Broker

This chapter is about some stuff.

Section Title	34
Subsection Title	35

DRAFT

Section Title

Subsection Title 35

DRAFT

Subsection Title

Block Title

DRAFT

DRAFT

Using Visualization

This chapter is about some stuff.

Section Title	38
Subsection Title	39

DRAFT

Section Title

Subsection Title 39

DRAFT

Subsection Title

Block Title

DRAFT

DRAFT

Securing the Broker

This chapter is about some stuff.

Section Title	42
Subsection Title	43

DRAFT

Section Title

Subsection Title 43

DRAFT

Subsection Title

Block Title

DRAFT

DRAFT

Index

DRAFT

DRAFT