Fuse MQ Enterprise

Glossary

Integration Everywhere

Glossary

Updated: 07 Jan 2014

Trademark Disclaimer

Third Party Acknowledgements

One or more products in the Fuse MQ Enterprise release includes third party components covered by licenses that require that the following documentation notices be provided:

- JLine (http://jline.sourceforge.net) jline:jline:jar:1.0

  License: BSD (LICENSE.txt) - Copyright (c) 2002-2006, Marc Prud'hommeaux <mwp1@cornell.edu>

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  - Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  - Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  - Neither the name of JLine nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Stax2 API (http://woodstox.codehaus.org/StAX2) org.codehaus.woodstox:stax2-api:jar:3.1.1

License: The BSD License (http://www.opensource.org/licenses/bsd-license.php)

Copyright (c) <YEAR>, <OWNER> All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- jibx-run - JiBX runtime (http://www.jibx.org/main-reactor/jibx-run) org.jibx:jibx-run:bundle:1.2.3

  License: BSD (http://jibx.sourceforge.net/jibx-license.html) Copyright (c) 2003-2010, Dennis M. Sosnoski.

  All rights reserved.

  Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

  - Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
  - Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
  - Neither the name of JiBX nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- JavaAssist (http://www.jboss.org/javassist) org.jboss.javassist:com.springsource.javassist:jar:3.9.0.GA:compile

  License: MPL (http://www.mozilla.org/MPL/MPL-1.1.html)

- HAPI-OSGI-Base Module (http://hl7api.sourceforge.net/hapi-osgi-base/) ca.uhn.hapi:hapi-osgi-base:bundle:1.2

Table of Contents

A

advisory	See advisory message.
advisory message	A special type of message that contains administrative information about the message broker. They are sent by the broker to special advisory topics. See Also advisory topic.
advisory topic	A group of special topics that are created by a message broker that are used for monitoring the state of the broker. The broker sends messages about a variety of internal broker events. Clients subscribing to these topics receive advisory messages about these objects.
agent	See Fabric Agent.
Apache ActiveMQ	An open source project that provides the messaging technology for Fuse MQ Enterprise.
Apache Karaf	An open source project that provides the OSGi runtime container used by Fuse MQ Enterprise.

C

child container	A container that is created by a container on the same host. Child containers are run on the same host as their parent container, but each child runs in a separate JVM. When created using the console's **admin:create-container**, a child container inherits the features, feature repositories, and configuration from its parent. When a child container is created using the **fabric:container-create** command, the **fabric:contaier-create-child**, command, or Fuse Management Console, it does not inherit any configuration from its parent. Regardless of how they are created, child containers can be started and shutdown from their parent container's console without using SSH.
client	An application that uses the message broker to communicate with other applications. These applications use one of the broker's client API to connect to and interact with the broker.
cluster	A group of brokers among which clients can failover.
composite destination	A virtual destination that serves as a proxy for multiple destinations. Producers can send messages to the composite destination an it will be automatically sent to all of the physical destinations that make up the composite destination. See Also virtual destination.
connection	A bridge between a client and a broker connector. See Also transport connector, network connector, network of brokers.
connection factory	An object that a client uses to create a connection to a broker. A factory supports attributes that configure the quality of service for the connections it creates.
connector	An object that connects clients to a broker.
consumer	An application that consumes messages from a messaging destination.

D

dead letter queue	A special destination used by the message broker to hold undeliverable messages.
dependency injection	A form of inversion of control, where an object's external dependencies are given to it, either programmatically or through a framework that is driven by configuration information. The result is to decouple dependent objects and allow the dependencies to be resolved at run time.
destination	A logical holding area for messages in a message broker. Clients publish messages to and consume messages from destinations. See Also queue, topic.
discovery agent	A mechanism that advertises the list of available message brokers to message clients and other message brokers. See Also dynamic discovery.
durable subscriber	A message consumer that receives all messages published on a topic, including those published while the subscriber is inactive.
dynamic discovery	A mechanism for clients to become aware of the existence of brokers through the use of a discovery agent. See Also discovery agent.

E

ensemble	See Fabric Ensemble.
exclusive consumer	A mechanism that ensures that only one consumer connected to a queue can consume messages.

F

fabric	A group of containers that are connected to a common Fabric Ensemble. The ensemble makes it possible for all of the containers to share runtime information about the services deployed in each container and allows them to share common configuration profiles.
Fabric Agent	The service running inside a Fabric Container that is responsible for configuring and provisioning the container according to the profiles assigned to the container . It is also responsible for updating the registry with runtime information about the services in container.
Fabric Container	A Apache Karaf-based container that is managed by a Fabric Agent. See Also Fabric Agent.
Fabric Ensemble	A group of one or more Fabric Servers that provide a number of services that bind a fabric. These services include the Fabric Registry, dynamic load balancing, and location transparency. See Also Fabric Registry, Fabric Server.
Fabric Registry	A ZooKeeper-based distributed registry that stores runtime and configuration information about the services in a fabric.
Fabric Server	A server that, as part of a Fabric Ensemble, provides a number of services that bind a fabric. These services include the Fabric Registry, dynamic load balancing, and location transparency. See Also Fabric Registry.
failover	A transport that automatically moves to a new connection in the event that its current connection fails. A cluster architecture where clients are able to migrate from a failed broker to a running broker.
feature	A unit of OSGi deployment that enables you to deploy multiple bundles in a single step.
feature repository	An XML file that defines one or more features.
feature URL	A URL that points to a feature repository file.
Fuse Application Bundle (FAB)	A bundle that uses a POM file to specify its dependencies.

Fuse Fabric	An open source project that provides a distributed runtime registry that provides configuration, deployment, and discovery services to a collection of distributed containers. See Also fabric.

I

i18n

An abbreviation for internationalization, used in the context of preparing products, especially software and documentation, for use in more than one national locale and language.

J

Java Architecture for XML Binding (JAXB)	An API that provides a way to bind an XML Schema to a representation in Java code.
Java Authentication and Authorization Service (JAAS)	A Java security framework for user-centric security to augment the Java code-based security.
Java Database Connectivity (JDBC)	An API specified in Java technology that provides Java applications with access to databases and other data sources.
Java Management eXtensions (JMX)	A Java technology that supplies tools for managing and monitoring applications, system objects, devices, and service-oriented networks.
Java Message Service (JMS)	A Java API implementing a messaging standard that allows application components based on J2EE to create, send, receive, and read messages. It enables distributed communication that is loosely coupled, reliable, and asynchronous.
Java Naming and Directory Interface (JNDI)	A set of APIs specified in Java technology that assists Java applications with interfacing to multiple naming and directory services.

L

l10n

An abbreviation for localization, used in the context of preparing products, especially software and documentation, for use in more than one national locale and language. Localization is the process of translating the elements of a product for a particular locale and language.

M

managed container

See Fabric Container.

marshalling

The process of taking in-memory objects and converting them to a binary or textual format for transmission over a transport.
See Also unmarshalling.

master/slave

A topology in which a single instance, the master, is active and one or more instances, the slaves, are ready to resume when the active instance stops.

message

An atomic unit of data that is passed between two or more clients. A message consists of three components:

- headers—contain a predefined set of metadata that is used to communicate information about a message between the different parties that handle the message
- properties—contain application defined metadata about a message to the different parties that handle the message
- body—contains the messages payload

message group

A collection of JMS messages that are assigned the same `JMSXGroupID`.

When used in conjunction with the `JMSXGroupSeq` message groups can be used to ensure that messages are processed in the proper sequence.

message selector

A string containing a boolean SQL statement using SQL 92 syntax that is used to select messages based on JMS message header properties.

N

network bridge	A runtime directional link between brokers that is used to forward messages. Network bridges are created by network connectors. See Also network connector.
network connector	A configuration entity used to link brokers together to form a network of brokers. See Also network of brokers.
network of brokers	A group of brokers that are linked together to operate as a single logical unit.
non-managed container	A Apache Karaf-based container that is registered with a fabric, but is *not* managed by a Fabric Agent.

P

persistent identifier (PID)	A registration property used by the OSGi Configuration Admin Service to identify a group of configuration attributes.
point-to-point messaging	A messaging style where messages are sent between two known endpoints. This messaging style is typically implemented using queues.
producer	An application that creates messages and posts them to a messaging destination.
profile	A set of data that defines runtime artifacts and configuration settings for provisioning a Fabric Container.
publish and subscribe messaging (pub/sub)	A messaging style where message producers send(publish) messages to a destination and interested consumers can register(subscribe) to receive messages from the destination. This style of messaging is implemented using topics.

Q

queue

A destination that uses first in/first out semantics.
See Also destination.

R

registry	See Fabric Registry.
request-reply pattern	A messaging pattern in which a message producer receives a message and returns a correlated message.
retroactive consumer	A consumer that indicates to the topic that every attempt is to be made to send messages that the consumer may have missed.

S

Session	A JMS object that provides a single-threaded context for producing and consuming messages. JMS clients use the `Session` object to create producers, consumers, messages, and other artifacts used to work with messages.
Spring framework	A comprehensive programming and configuration model for modern Java-based enterprise applications the uses dependency injection. See Also dependency injection.
standalone broker	See standalone container.
standalone container	A container that is not part of a fabric and does not have a Fabric Agent installed.
store and forward	A paradigm in which brokers receive messages, store them locally, and forwards the message to a recipient when it is able to do so. The message is only deleted once it has been successfully delivered.
Streaming Text Orientated Messaging Protocol (STOMP)	A language agnostic, simple text-based protocol that allows clients to talk with any message broker supporting the protocol.

T

topic	A destination that uses publish and subscribe semantics. See Also destination.
transport	A standards-based network protocol, such as HTTP or STOMP, that defines how objects communicate over a network.
transport connector	An address at which a message broker accepts client connections.

U

Uniform Resource Identifier (URI)	A string of characters used to identify or name a resource on the Internet.
unmarshalling	The process of taking a binary or textual format payload and converting that into objects. See Also marshalling.

V

version	A collection of configuration profiles in a Fabric Registry. See Also profile.
virtual destination	A logical destination that represents one or more physical destinations. See Also composite destination, virtual topic.
virtual topic	A logical topic that allows consumers to use a physical queue to consume messages from the destination. See Also virtual destination, topic, queue.