


Grouper Messaging with ActiveMQ

[Wiki Home](#)[Download Grouper](#)[Grouper Guides](#)[Community Contributions](#)[Developer Resources](#)[Deployment Guide](#)

 The info on this page applies to Grouper 2.5 and above.

The Grouper Messaging system interface has an ActiveMQ implementation.

Install ActiveMQ

On Mac, Download ActiveMQ from <http://activemq.apache.org/activemq-5150-release.html>. Unzip the file and go to bin directory. Run `./activemq console`. It will launch ActiveMQ server and the web client.

Login to <http://localhost:8161> to access the web UI.

Notes:

- Grouper ActiveMQ messaging client uses `qpido-jms-client 0.41.0` to connect to the broker which supports `amqp 1.0`.
- Grouper ActiveMQ messaging doesn't support acknowledging the individual messages. `acknowledge` method is a No-Op.
- Clients always have to set the `autoCreateObjects` property to `true` while calling `send` and `receive` messages on `GrouperMessagingActiveMQSystem` object.

Set the following properties in `grouper.client.properties` file

```
# class that implements edu.internet2.middleware.grouperClient.messaging.GrouperMessagingSystem
grouper.messaging.system.activeMqSystem.class = edu.internet2.middleware.grouperMessagingActiveMQ.
GrouperMessagingActiveMQSystem

grouper.messaging.system.activeMqSystem.name=activeMqSystem

# host address of activemq queue
grouper.messaging.system.activeMqSystem.host = localhost

# port of activemq queue
grouper.messaging.system.activeMqSystem.port = 5672

# username of activemq queue
grouper.messaging.system.activeMqSystem.username =

# password of activemq queue
grouper.messaging.system.activeMqSystem.password =

# number of seconds to sleep while waiting
grouper.messaging.system.activeMqSystem.polling.sleep.seconds = 5

grouper.messaging.system.activeMqSystem.defaultPageSize = 5

grouper.messaging.system.activeMqSystem.maxPageSize = 10
```

Here is an esb example of sending messages

```

#####
## Messaging integration with ESB, send change log entries to a messaging system
#####

# note, change "messagingEsb" in key to be the name of the consumer. e.g. changeLog.consumer.myAzureConsumer.
class
# note, routingKey property is valid only for rabbitmq. For other messaging systems, it is ignored.
# {valueType: "class", readOnly: true, mustExtendClass: "edu.internet2.middleware.grouper.changeLog.
ChangeLogConsumerBase"}
changeLog.consumer.cmuActiveMQ.class = edu.internet2.middleware.grouper.changeLog.esb.consumer.EsbConsumer

# quartz cron
# {valueType: "cron"}
changeLog.consumer.cmuActiveMQ.quartzCron = 0 * * * * ?

# el filter
# {valueType: "string", regex: "^changeLog\\.consumer\\.([^.]+)\\.elfilter$"}
changeLog.consumer.cmuActiveMQ.elfilter = event.eventType eq 'GROUP_DELETE' || event.eventType eq 'GROUP_ADD'
|| event.eventType eq 'MEMBERSHIP_DELETE' || event.eventType eq 'MEMBERSHIP_ADD'

# publishing class
# {valueType: "class", readOnly: true, mustExtendClass: "edu.internet2.middleware.grouper.changeLog.esb.
consumer.EsbMessagingPublisher"}
changeLog.consumer.cmuActiveMQ.publisher.class = edu.internet2.middleware.grouper.changeLog.esb.consumer.
EsbMessagingPublisher

# messaging system name
# {valueType: "string", regex: "^changeLog\\.consumer\\.([^.]+)\\.messagingSystemName$"}
changeLog.consumer.cmuActiveMQ.publisher.messagingSystemName = cmuActiveMQ

# queue or topic
# {valueType: "string", regex: "^changeLog\\.consumer\\.([^.]+)\\.publisher\\.messageQueueType$"}
changeLog.consumer.cmuActiveMQ.publisher.messageQueueType = queue

# queue or topic name
# {valueType: "string", regex: "^changeLog\\.consumer\\.([^.]+)\\.publisher\\.queueOrTopicName$"}
changeLog.consumer.cmuActiveMQ.publisher.queueOrTopicName = test.grouper.ws

```