Red Hat AMQ Broker 7.10

AMQ Broker スタートガイド

AMQ Broker 7.10 向け

Last Updated: 2024-06-11

Red Hat AMQ Broker 7.10 AMQ Broker スタートガイド

AMQ Broker 7.10 向け

法律上の通知

概要

このガイドでは、AMQ Broker の使用を開始する方法を説明します。

目次

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、Red Hat CTO である Chris Wright のメッセージ をご覧ください。

第1章 概要

AMQ Broker は、ActiveMQ Artemis をベースとした高パフォーマンスのメッセージング実装です。メッセージの永続性を迅速化するために非同期のジャーナルを使用し、複数の言語、プロトコル、およびプラットフォームをサポートします。

1.1. 主な特長

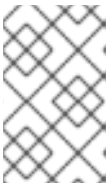
AMQ Broker には以下の機能があります。

- クラスター化および高可用性オプション
- 高速なネイティブ IO の永続性
- ローカルトランザクションのサポート
- AMQ Core Protocol JMS および AMQ OpenWire JMS クライアントを使用する場合の XA トランザクションのサポート
- 幅広いプラットフォームをサポートするための Java 言語の使用
- 複数の管理インターフェイス: AMQ 管理コンソール、管理 API、および JMX

1.2. SUPPORTED STANDARDS AND PROTOCOLS

AMQ Broker では、以下の標準およびプロトコルがサポートされます。

- ワイヤプロトコル:
  - Core Protocol
  - AMQP 1.0
  - MQTT
  - OpenWire(A-MQ 6 クライアントにより使用)
  - STOMP
- JMS 2.0



注記

AMQP 内の分散トランザクション (XA) の詳細は、仕様の 1.0 バージョンでは提供されません。お使いの環境で分散トランザクションのサポートが必要な場合は、AMQ Core Protocol JMS を使用することが推奨されます。

1.3. サポートされる構成

AMQ Broker のサポートされる設定に関する最新情報は、Red Hat カスタマーポータルの記事 Red Hat AMQ 7 でサポートされる構成 を参照してください。

1.4. このドキュメントの表記慣例

このドキュメントでは、**sudo** コマンド、ファイルパス、および置き換え可能な値について、以下の規則を使用します。

sudo コマンド

このドキュメントでは、root 権限を必要とするすべてのコマンドに対して **sudo** が使用されています。何らかの変更がシステム全体に影響を与える可能性があるため、**sudo** を使用する場合は、常に注意が必要です。

sudo の使用の詳細は、sudo アクセスの管理 を参照してください。

このドキュメントにおけるファイルパスの使用

このドキュメントでは、すべてのファイルパスは Linux、UNIX、および同様のオペレーティングシステムで有効です (例: **/home/...**)。Microsoft Windows を使用している場合は、同等の Microsoft Windows パスを使用する必要があります (例: **C:\Users\...**)。

交換可能な値

このドキュメントでは、お客様の環境に合わせた値に置き換える必要のある置換可能な値を使用している場合があります。置き換え可能な値は小文字で、角括弧 (**< >**) で囲まれ、イタリックおよび **monospace** フォントを使用してスタイルされます。単語が複数になる場合は、アンダースコア (_) で区切ります。

たとえば、**<install_dir>** を独自のディレクトリー名に置き換えます。

```
$ <install_dir>/bin/artemis create mybroker
```

第2章 AMQ BROKER について

AMQ Broker は、信頼性、トランザクション、およびその他の多くの機能を提供する一方で、異種システムを疎結合できます。AMQ Broker を使用する前に、提供される機能を理解する必要があります。

2.1. ブローカーインスタンス

AMQ Broker では、インストールされた AMQ Broker ソフトウェアは、1 つ以上の **ブローカーインスタンス** の基盤として機能します。このアーキテクチャーには、以下のような利点があります。

- 単一の AMQ Broker インストールから必要なだけブローカーインスタンスを作成できます。AMQ Broker インストールには、各ブローカーインスタンスの実行に必要なバイナリーおよびリソースが含まれます。これらのリソースはブローカーインスタンス間で共有されます。
- 新しいバージョンの AMQ Broker にアップグレードする場合は、ホストで複数のブローカーインスタンスを実行している場合でも、ソフトウェアを更新する必要があるのは 1 度だけです。

ブローカーインスタンスをメッセージブローカーとみなすことができます。各ブローカーインスタンスには、一意の設定およびランタイムデータが含まれる独自のディレクトリーがあります。このランタイムデータはログとデータファイルで設定され、ホスト上の一意のブローカープロセスに関連付けられます。

2.2. メッセージの永続性

AMQ Broker はメッセージデータを永続化し、予期せずブローカーに障害が発生したりシャットダウンしたりしても、メッセージが失われないようにします。メッセージの永続性に関して、AMQ Broker では、ジャーナルベースの永続性とデータベースの永続性の 2 つのオプションが提供されます。

ジャーナルベースの永続性
デフォルトの手法であるこのオプションでは、ファイルシステムに保存されるメッセージジャーナルファイルにメッセージデータを書き込みます。最初に、これらのジャーナルファイルは、固定サイズで自動的に作成され、空のデータが入力されます。クライアントがさまざまなブローカー操作を実行すると、レコードがジャーナルに追加されます。ジャーナルファイルの 1 つが満杯になると、ブローカーは次のジャーナルファイルに移動します。

ジャーナルベースの永続性は、ローカルトランザクションと XA トランザクションの両方を含むトランザクション操作をサポートします。

ジャーナルベースの永続化には、ファイルシステムへの IO インターフェイスが必要です。AMQ Broker は以下をサポートします。

Linux 非同期 IO (AIO)

通常、AIO は最高のパフォーマンスを提供します。ネットワークファイルシステムが Red Hat AMQ 7 のサポートされる設定 にサポート対象としてリストされていることを確認します。

Java NIO

Java NIO により優れたパフォーマンスを実現できます。また、Java 6 以降のランタイムの任意のプラットフォーム上で稼働します。

データベースの永続性
このオプションは、Java Database Connectivity (JDBC) を使用して、メッセージとバインディングデータをデータベースに保存します。このオプションは、お使いの環境に信頼性が高く、高パフォーマンスのデータベースプラットフォームがある場合や、会社ポリシーでデータベースを使用することが義務付けられている場合に適しています。

ブローカー JDBC 永続ストアは標準の JDBC ドライバーを使用して、メッセージおよびバインディングデータをデータベーステーブルに保存する JDBC コネクションを作成します。データベーステーブルのデータは、ジャーナルベースの永続化と同じエンコーディングを使用してエンコードされます。つまり、SQL を使用して直接アクセスしても、データベースに保存されているメッセージは人間が判読できません。

データベースの永続性を使用するには、サポートされるデータベースプラットフォームを使用する必要があります。現在サポートされているデータベースプラットフォームを確認するには、Red Hat AMQ 7 でサポートされる構成 を参照してください。

2.3. リソースの消費

AMQ Broker には、ブローカーのメモリーおよびリソース消費を制限するオプションが多数用意されています。

リソース制限

各ユーザーに接続およびキュー制限を設定できます。これにより、ユーザーがブローカーのリソースを過剰に消費し、他のユーザーのパフォーマンスが低下するのを防ぐことができます。

メッセージのページング

メッセージのページングにより、AMQ Broker は、限られたメモリー使用量で動作中でも、大量のメッセージが含まれる大規模なキューをサポートすることができます。ブローカーが受信するメッセージが急増しメモリー容量を超えると、ディスクへのメッセージのページングが開始します。このページングプロセスは透過的で、必要に応じてブローカーはメモリーとの間でメッセージをページングします。

メッセージのページングはアドレスベースです。アドレスのメモリー内でメッセージのサイズ合計が最大サイズを超えると、そのアドレスに追加されたメッセージはアドレスのページファイルにページングされます。

大規模なメッセージ

AMQ Broker を使用すると、限られたメモリーリソースで動作中でも、大規模なメッセージを送受信できます。大規模なメッセージをメモリーに格納する際のオーバーヘッドを回避するには、これらの大規模なメッセージをファイルシステムまたはデータベーステーブルに保存するように AMQ Broker を設定できます。

2.4. 監視および管理

AMQ Broker は、ブローカーの監視および管理に使用できるツールを複数提供します。

AMQ 管理コンソール

AMQ 管理コンソールは、Web ブラウザーからアクセスできる Web インターフェイスです。ネットワークの正常性を監視し、ブローカートポロジーを表示し、ブローカーリソースの作成および削除に使用できます。

CLI

AMQ Broker では、ブローカーの管理に使用できる **artemis** CLI が提供されます。CLI を使用すると、ブローカーインスタンスを作成、起動、および停止できます。CLI には、メッセージジャーナルを管理するための複数のコマンドもあります。

管理 API

AMQ Broker では、豊富な管理 API が提供されます。これを使用して、ブローカー設定の変更、新規リソースの作成、これらのリソースの検査、およびそれらとの連携を行うことができます。クライアントは管理 API を使用してブローカーを管理し、管理通知をサブスクライブすることもできます。

AMQ Broker は、管理 API を使用するために以下の方法を提供します。

- JMX (Java Management Extensions) - JMX は Java アプリケーションを管理するための標準技術です。ブローカーの管理操作は AMQ MBean インターフェイスを介して公開されます。
- JMS API - 管理操作は、標準の JMS メッセージを使用して特別な管理 JMS キューに送信されます。

ログ

各ブローカーインスタンスは、エラーメッセージ、警告、およびその他のブローカー関連の情報およびアクティビティーをログに記録します。ロギングレベル、ログファイルの場所、およびログ形式を設定できます。その後、作成されるログファイルを使用してブローカーを監視し、エラー状態を診断できます。

第3章 AMQ BROKER のインストール

AMQ Broker はプラットフォームに依存しないアーカイブファイルとして配布されます。システムに AMQ Broker をインストールするには、アーカイブをダウンロードし、コンテンツをデプロイメントする必要があります。アーカイブに含まれるディレクトリーも理解する必要があります。

前提条件

- AMQ Broker をインストールするホストは、AMQ Broker でサポートされる設定を満たしている必要があります。
  詳細は、Red Hat AMQ 7 でサポートされる構成 を参照してください。

3.1. AMQ BROKER アーカイブのダウンロード

AMQ Broker はプラットフォームに依存しないアーカイブファイルとして配布されます。Red Hat カスタマーポータルからダウンロードできます。

前提条件

- Red Hat サブスクリプションが必要です。
  詳細は、サブスクリプションの使用を参照してください。

手順

1. Web ブラウザーで https://access.redhat.com/downloads/ に移動し、ログインします。
   Product Downloads ページが表示されます。
2. **JBoss Integration and Automation** セクションで、**Red Hat AMQ Broker**のリンクをクリックします。
   Software Downloads ページが表示されます。
3. **Version** ドロップダウンメニューから必要な AMQ Broker のバージョンを選択します。
4. **Releases** タブで、ダウンロードする特定の AMQ Broker ファイルの **Download** のリンクをクリックします。

3.2. LINUX での AMQ BROKER アーカイブのデプロイメント

Red Hat Enterprise Linux に AMQ Broker をインストールする場合は、AMQ Broker の新しいユーザーアカウントを作成し、インストールアーカイブからコンテンツをデプロイメントします。

手順

1. **amq-broker** という名前の新規ユーザーを作成して、パスワードを指定します。

```
$ sudo useradd amq-broker
$ sudo passwd amq-broker
```

2. **/opt/redhat/amq-broker** ディレクトリーを作成して、新しい **amq-broker** ユーザーとグループの所有者として指定します。

```
$ sudo mkdir /opt/redhat
$ sudo mkdir /opt/redhat/amq-broker
$ sudo chown -R amq-broker:amq-broker /opt/redhat/amq-broker
```

3. アーカイブの所有者を新規ユーザーに変更します。

```
$ sudo chown amq-broker:amq-broker amq-broker-7.x.x-bin.zip
```

4. インストールアーカイブを、作成したディレクトリーに移動します。

```
$ sudo mv amq-broker-7.x.x-bin.zip /opt/redhat/amq-broker
```

5. 新しい **amq-broker** ユーザーとして、**unzip** コマンドを使用してコンテンツをデプロイメントします。

```
$ su - amq-broker
$ cd /opt/redhat/amq-broker
$ unzip <archive_name>.zip
$ exit
```

amq-broker-7.10 と似たディレクトリーが作成されます。このドキュメントでは、この場所は **<install_dir>** と呼ばれます。

3.3. WINDOWS システムでの AMQ BROKER アーカイブのデプロイメント

Windows システムに AMQ Broker をインストールする場合は、AMQ Broker の新しいディレクトリーフォルダーを作成してから、そこにコンテンツをデプロイメントします。

手順

1. Windows Explorer を使用して、任意のドライブ文字にディレクトリー **\redhat\amq-broker** を作成します。
例: **C:\redhat\amq-broker**

2. Windows Explorer を使用して、作成したディレクトリーにインストールアーカイブを移動します。

3. **\redhat\amq-broker** ディレクトリーで、インストールアーカイブ zip ファイルを右クリックし、**Extract All** を選択します。
amq-broker-7.10 と似たディレクトリーが作成されます。このドキュメントでは、この場所は **<install_dir>** と呼ばれます。

3.4. AMQ BROKER インストールアーカイブのコンテンツについて

アーカイブをデプロイメントして作成したディレクトリーは、AMQ Broker インストールの最上位ディレクトリーとなります。このディレクトリーは **<install_dir>** と呼ばれ、以下の内容が含まれます。

ディレクトリー	内容
<install_dir>/web/api	API ドキュメント。

ディレクトリー	内容
<install_dir>/bin	AMQ Broker の実行に必要なバイナリーおよびスクリプト。
<install_dir>/etc	設定ファイル。
<install_dir>/examples	JMS および Java EE の例。
<install_dir>/lib	AMQ Broker の実行に必要な JAR およびライブラリー。
<install_dir>/schema	AMQ Broker 設定の検証に使用される XML スキーマ。
<install_dir>/web	AMQ Broker の実行時に読み込まれる Web コンテキスト。

第4章 スタンドアロンブローカーの作成

ローカルマシンにスタンドアロンのブローカーインスタンスを作成し、それを起動してテストメッセージを生成および使用することで、AMQ Broker ですぐに使い始めることができます。

前提条件

- AMQ Broker がインストールされている。
  詳細は、3章*AMQ Broker のインストール* を参照してください。

4.1. ブローカーインスタンスの作成

ブローカーインスタンスは、ブローカーの設定およびランタイムデータが含まれるディレクトリーです。新規ブローカーインスタンスを作成するには、まずブローカーインスタンスのディレクトリーを作成し、**artemis create** コマンドを使用してブローカーインスタンスを作成します。

この手順では、ローカルマシンに簡単なスタンドアロンのブローカーを作成する方法を説明します。ブローカーは基本的なデフォルト設定を使用し、サポートされるメッセージングプロトコルのいずれかを使用してクライアントからの接続を受け入れます。

手順

1. ブローカーインスタンスのディレクトリーを作成します。

使用しているプラットフォーム	実行内容

Red Hat Enterprise Linux	1. ブローカーインスタンスの場所として機能する新しいディレクトリーを作成します。
	`$ sudo mkdir /var/opt/amq-broker`
	2. インストール時に作成したユーザーを割り当てます。
	`$ sudo chown -R amq-broker:amq-broker /var/opt/amq-broker`

Windows	Windows Explorer を使用して、ブローカーインスタンスの場所として機能する新規フォルダーを作成します。

2. **artemis create** コマンドを使用してブローカーを作成します。

使用しているプラットフォーム	実行内容

使用しているプラットフォーム	実行内容

Red Hat Enterprise Linux	1. インストール時に作成したユーザーのアカウントに切り替えます。
	`$ su - amq-broker`
	2. ブローカーインスタンス用に作成したディレクトリーに移動します。
	`$ cd /var/opt/amq-broker`
	3. ブローカーインスタンスのディレクトリーから、ブローカーインスタンスを作成します。
	`$ <install_dir>/bin/artemis create mybroker`

Windows	1. ブローカーインスタンス用に作成したディレクトリーからコマンドプロンプトを開きます。 2. ブローカーインスタンスのディレクトリーから、ブローカーインスタンスを作成します。 `> <install_dir>\bin\artemis.cmd create mybroker`

3. **artemis create** プロンプトに従ってブローカーインスタンスを設定します。

例4.1 artemis create を使用したブローカーインスタンスの設定

```
$ /opt/redhat/amq-broker/bin/artemis create mybroker

Creating ActiveMQ Artemis instance at: /var/opt/amq-broker/mybroker

--user: is mandatory with this configuration:
Please provide the default username:
admin

--password: is mandatory with this configuration:
Please provide the default password:

--role: is mandatory with this configuration:
Please provide the default role:
amq

--allow-anonymous | --require-login: is mandatory with this configuration:
Allow anonymous access? (Y/N):
Y

Auto tuning journal ...
done! Your system can make 19.23 writes per millisecond, your journal-buffer-timeout will be 52000
```

```
You can now start the broker by executing:

"/var/opt/amq-broker/mybroker/bin/artemis" run

Or you can run the broker in the background using:

"/var/opt/amq-broker/mybroker/bin/artemis-service" start
```

4.2. ブローカーインスタンスの起動

ブローカーインスタンスの作成後に、**artemis run** コマンドを使用して起動します。

手順

1. インストール時に作成したユーザーのアカウントに切り替えます。

```
$ su - amq-broker
```

2. **artemis run** コマンドを使用してブローカーインスタンスを起動します。

```
$ /var/opt/amq-broker/mybroker/bin/artemis run


        __  __  ____    ____          _
      /\   |  \/  |/ __ \  |  _ \        | |
     /  \  | \  / | |  | | | |_) |_ __ ___ | | _____ _ __
    / /\ \ | |\/| | |  | | |  _ <| '__/ _ \| |/ / _ \ '__|
   / ____ \| |  | | |__| | | |_) | | | (_) |   <  __/ |
  /_/    \_\_|  |_|\____\_\ |____/|_|  \___/|_|\_\___|_|

Red Hat JBoss AMQ 7.2.1.GA

10:53:43,959 INFO  [org.apache.activemq.artemis.integration.bootstrap] AMQ101000:
Starting ActiveMQ Artemis Server
10:53:44,076 INFO  [org.apache.activemq.artemis.core.server] AMQ221000: live Message
Broker is starting with configuration Broker Configuration
(clustered=false,journalDirectory=./data/journal,bindingsDirectory=./data/bindings,largeMessage
sDirectory=./data/large-messages,pagingDirectory=./data/paging)
10:53:44,099 INFO  [org.apache.activemq.artemis.core.server] AMQ221012: Using AIO
Journal
...
```

ブローカーが起動し、以下の情報が含まれるログ出力が表示されます。

- トランザクションログとクラスター設定の場所。
- メッセージの永続性に使用されるジャーナルのタイプ (この場合は AIO)。
- クライアント接続を許可できる URI。
  デフォルトでは、ポート 61616 はサポートされるプロトコル (CORE、MQTT、AMQP、STOMP、HORNETQ、および OPENWIRE) からの接続を受け入れることができます。各プロトコルには個別のポートもあります。

- Web コンソールは、http://localhost:8161 から入手できます。
- Jolokia サービス (JMX over REST) には、http://localhost:8161/jolokia からアクセスできます。

4.3. テストメッセージの生成および使用

ブローカーの起動後に、ブローカーが適切に実行されていることを確認する必要があります。そのためには、いくつかのテストメッセージを作成してブローカーに送信し、使用します。

手順

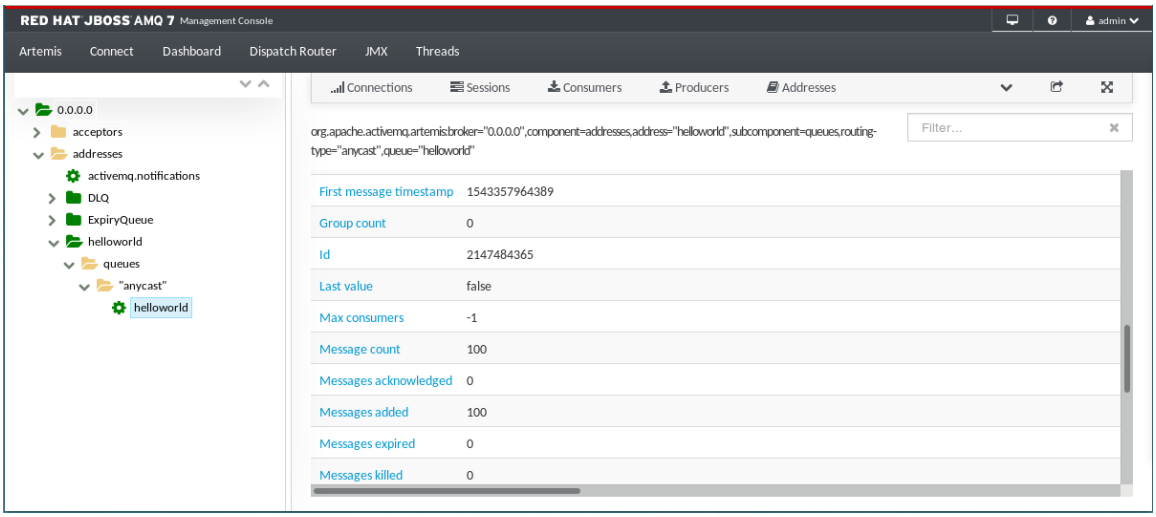
1. **artemis producer** コマンドを使用していくつかのテストメッセージを生成し、ブローカーに送信します。
   このコマンドは、ブローカーに自動的に作成される **helloworld** アドレスにメッセージを 100 個送信します。プロデューサーは、サポートされるすべてのメッセージングプロトコルを受け入れるデフォルトのポート 61616 を使用してブローカーに接続します。

```
$ /opt/redhat/amq-broker/amq-broker-7.2.0/bin/artemis producer --destination helloworld --message-count 100 --url tcp://localhost:61616
Producer ActiveMQQueue[helloworld], thread=0 Started to calculate elapsed time ...

Producer ActiveMQQueue[helloworld], thread=0 Produced: 100 messages
Producer ActiveMQQueue[helloworld], thread=0 Elapsed time in second : 1 s
Producer ActiveMQQueue[helloworld], thread=0 Elapsed time in milli second : 1289 milli seconds
```

2. Web コンソールを使用して、ブローカーに保存されているメッセージを表示します。
   a. Web ブラウザーで、http://localhost:8161 にアクセスします。
   b. ブローカーインスタンスの作成時に作成したデフォルトのユーザー名およびパスワードを使用して、コンソールにログインします。
      Attributes タブが表示されます。
   c. **Attributes** タブで、**addresses → helloworld → queues → "anycast" → helloworld** に移動します。
      前の手順で、**helloworld** アドレスにメッセージを送信しています。これにより、キュー (**helloworld** という名前) を持つ新しい anycast **helloworld** アドレスが作成されました。**Message count** 属性は、現在 **helloworld** に送信された 100 個のメッセージがすべてこのキューに保存されていることを示しています。

図4.1 メッセージ数



3. **artemis consumer** コマンドを使用して、ブローカーに保存されているメッセージ 50 個を消費します。
このコマンドは、以前にブローカーに送信した 50 のメッセージを使用します。

```
$ /opt/redhat/amq-broker/amq-broker-7.2.0/bin/artemis consumer --destination helloworld --message-count 50 --url tcp://localhost:61616

Consumer:: filter = null
Consumer ActiveMQQueue[helloworld], thread=0 wait until 50 messages are consumed
Consumer ActiveMQQueue[helloworld], thread=0 Consumed: 50 messages
Consumer ActiveMQQueue[helloworld], thread=0 Consumer thread finished
```

4. Web コンソールで、**Message count** が 50 であることを確認します。
メッセージが 50 個消費され、**helloworld** キューに格納されているメッセージ数の残りは 50 個になります。

5. ブローカーを停止し、残りのメッセージ 50 個が **helloworld** キューに保存されていることを確認します。

   a. ブローカーが実行されているターミナルで、**Ctrl**+**C**を押してブローカーを停止します。

   b. ブローカーを再起動します。

   ```
   $ /var/opt/amq-broker/mybroker/bin/artemis run
   ```

   c. Web コンソールで、**helloworld** キューに戻り、キューに保存されているメッセージが 50 個であることを確認します。

6. 残りの 50 個メッセージを使用します。

```
$ /opt/redhat/amq-broker/amq-broker-7.2.0/bin/artemis consumer --destination helloworld --message-count 50 --url tcp://localhost:61616

Consumer:: filter = null
Consumer ActiveMQQueue[helloworld], thread=0 wait until 50 messages are consumed
Consumer ActiveMQQueue[helloworld], thread=0 Consumed: 50 messages
Consumer ActiveMQQueue[helloworld], thread=0 Consumer thread finished
```

7. Web コンソールで、**Message count** が 0 であることを確認します。
helloworld キューに保存されているメッセージはすべて消費され、キューが空になりました。

4.4. ブローカーインスタンスの停止

スタンドアロンブローカーを作成し、テストメッセージを生成および消費した後、ブローカーインスタンスを停止できます。

この手順では、ブローカーを手動で停止し、クライアント接続をすべて強制的に閉じます。実稼働環境では、クライアント接続を適切に閉じるようにブローカーを正常に停止するようにブローカーを設定する必要があります。

手順

- **artemis stop** コマンドを使用してブローカーインスタンスを停止します。

```
$ /var/opt/amq-broker/mybroker/bin/artemis stop
2018-12-03 14:37:30,630 INFO  [org.apache.activemq.artemis.core.server] AMQ221002: Apache ActiveMQ Artemis Message Broker version 2.6.1.amq-720004-redhat-1 [b6c244ef-f1cb-11e8-a2d7-0800271b03bd] stopped, uptime 35 minutes
Server stopped!
```

第5章 AMQ BROKER サンプルの実行

AMQ Broker には、製品の基本的な機能と高度な機能を示す多くのサンプルプログラムが同梱されています。これらのサンプルを実行して、AMQ Broker の機能を理解することができます。

AMQ Broker のサンプルを実行するには、まず Apache Maven および AMQ Maven リポジトリーをインストールおよび設定してマシンを設定する必要があります。続いて、Maven を使用して AMQ Broker サンプルプログラムを実行します。

5.1. AMQ BROKER サンプルを実行するためのマシンの設定

含まれる AMQ Broker サンプルプログラムを実行する前に、まず Maven および AMQ Maven リポジトリーをダウンロードしてインストールし、Maven 設定ファイルを設定する必要があります。

5.1.1. Maven のダウンロードおよびインストール

AMQ Broker サンプルを実行するには、Maven が必要です。

手順

1. Apache Maven Download ページにアクセスし、ご使用のオペレーティングシステムに対応する最新のディストリビューションをダウンロードします。
2. ご使用のオペレーティングシステム用の Maven をインストールします。
   詳細は、Installing Apache Maven を参照してください。

関連情報

- Maven の詳細は、Apache Maven の概要を参照してください。

5.1.2. AMQ Maven リポジトリーのダウンロードおよびインストール

Maven をマシンにインストールしたら、AMQ Maven リポジトリーをダウンロードしてインストールします。このリポジトリーは Red Hat カスタマーポータルから入手できます。

1. Web ブラウザーで https://access.redhat.com/downloads/ に移動し、ログインします。
   Product Downloads ページが表示されます。
2. **Integration and Automation** セクションで、**Red Hat AMQ Broker**のリンクをクリックします。
   Software Downloads ページが表示されます。
3. **Version** ドロップダウンメニューから必要な AMQ Broker のバージョンを選択します。
4. **Releases** タブで、AMQ Broker Maven Repository の **Download** のリンクをクリックします。
   AMQ Maven リポジトリーファイルが zip ファイルとしてダウンロードされます。
5. ご自分のマシンで、AMQ リポジトリーファイルを選択したディレクトリーにデプロイメントします。
   新しいディレクトリーがマシンに作成されます。このファイルには、**maven-repository/** という名前のサブディレクトリーに Maven リポジトリーが含まれます。

5.1.3. Maven 設定ファイルの設定

AMQ Maven リポジトリーのダウンロードおよびインストール後に、リポジトリーを Maven 設定ファイルに追加する必要があります。

手順

1. Maven の **settings.xml** ファイルを開きます。
   settings.xml ファイルは、通常 **${user.home}/.m2/** ディレクトリーにあります。

   - Linux の場合は **~/.m2/** です。
   - Windows では、これは **\Documents and Settings\.m2** または **\Users\.m2** になります。

   ${user.home}/.m2/** に **settings.xml** ファイルがない場合は、Maven インストールの **conf/** ディレクトリーにデフォルトのバージョンがあります。デフォルトの **settings.xml** ファイルを **${user.home}/.m2/ ディレクトリーにコピーします。

2. **<profiles>** 要素に、AMQ Maven リポジトリーのプロファイルを追加します。

```
<!-- Configure the JBoss AMQ Maven repository -->
<profile>
 <id>jboss-amq-maven-repository</id>
 <repositories>
  <repository>
   <id>jboss-amq-maven-repository</id>
   <url>file://<JBoss-AMQ-repository-path></url> (1)
   <releases>
    <enabled>true</enabled>
   </releases>
   <snapshots>
    <enabled>false</enabled>
   </snapshots>
  </repository>
 </repositories>
 <pluginRepositories>
  <pluginRepository>
   <id>jboss-amq-maven-repository</id>
   <url>file://<JBoss-AMQ-repository-path></url> (2)
   <releases>
    <enabled>true</enabled>
   </releases>
   <snapshots>
    <enabled>false</enabled>
   </snapshots>
  </pluginRepository>
 </pluginRepositories>
</profile>
```

(1) (2) **<jboss-AMQ-repository-path>** は、インストールした Maven リポジトリーの場所に置き換えます。通常、この場所は **/maven-repository** で終わります。以下に例を示します。

```
<url>file:///path/to/repo/amq-broker-7.2.0-maven-repository/maven-repository</url>
```

3. **<activeProfiles>** 要素で、AMQ Maven リポジトリーをアクティブにします。

```
<activeProfiles>
 <activeProfile>jboss-amq-maven-repository</activeProfile>
...
</activeProfiles>
```

4. Maven インストールからデフォルトの **settings.xml** をコピーし、デフォルトで **<active-profiles>** セクションがコメントアウトされている場合はコメントを解除します。

5. **settings.xml** を保存して閉じます。

6. キャッシュした **${user.home}/.m2/repository/** ディレクトリーを削除します。
Maven リポジトリーに古いアーティファクトが含まれる場合は、プロジェクトをビルドまたはデプロイしたときに以下のいずれかの Maven エラーメッセージが表示されることがあります。

   - **Missing artifact <artifact-name>**
   - **[ERROR] Failed to execute goal on project <project-name>; Could not resolve dependencies for <project-name>**

5.2. AMQ BROKER のサンプルプログラム

AMQ Broker には、AMQ Broker 機能の使用方法およびサポートされるメッセージングプロトコルを示す 90 を超えるサンプルプログラムが同梱されています。

このプログラムの例は **<install_dir>/examples** にあり、以下が含まれます。

- 機能
ブローカー固有の機能 (以下を含む)
  - Clustered: ロードバランシングおよび負荷分散機能を示す例
  - HA: フェイルオーバーおよび再接続機能を示す例
  - Perf: サーバーでいくつかのパフォーマンステストを実行できる例
  - Standard: さまざまなブローカー機能を示すサンプル
  - Sub-modules: 統合外部モジュールの例
- プロトコル
サポートされる各メッセージングプロトコルの例:
  - AMQP
  - MQTT
  - OpenWire
  - STOMP

関連情報

- 各サンプルプログラムの説明は、Apache Artemis ドキュメントのExamplesを参照してください。

5.3. AMQ BROKER サンプルプログラムの実行

AMQ Broker には、製品の基本的な機能と高度な機能を示す多くのサンプルプログラムが同梱されています。Maven を使用してこれらのプログラムを実行します。

前提条件

- AMQ Broker サンプルを実行するようにマシンを設定する必要があります。
  詳細は、「AMQ Broker サンプルを実行するためのマシンの設定」 を参照してください。

手順

1. 実行する例のディレクトリーに移動します。
   このプログラムの例は **<install_dir>/examples** にあります。以下に例を示します。

   ```
   $ cd <install_dir>/examples/features/standard/queue
   ```

2. **mvn clean verify** コマンドを使用して、サンプルプログラムを実行します。
   Maven はブローカーを起動し、サンプルプログラムを実行します。サンプルプログラムの初回実行時に、Maven は不足している依存関係をダウンロードします。この処理には時間がかかる場合があります。

   この場合は、**queue** のサンプルプログラムが実行します。これにより、プロデューサーが作成され、テストメッセージが送信されてから、メッセージを受信するコンシューマーが作成されます。

   ```
   $ mvn clean verify
   [INFO] Scanning for projects...
   [INFO]
   [INFO] ------------< org.apache.activemq.examples.broker:queue >--------------
   [INFO] Building ActiveMQ Artemis JMS Queue Example 2.6.1.amq-720004-redhat-1
   [INFO] --------------------------------[ jar ]---------------------------------
   ...
   server-out:2018-12-05 16:37:57,023 INFO  [org.apache.activemq.artemis.core.server]
   AMQ221001: Apache ActiveMQ Artemis Message Broker version 2.6.1.amq-720004-redhat-
   1 [0.0.0.0, nodeID=06f529d3-f8d6-11e8-9bea-0800271b03bd]
   [INFO] Server started
   [INFO]
   [INFO] --- artemis-maven-plugin:2.6.1.amq-720004-redhat-1:runClient (runClient) @ queue --
   -
   Sent message: This is a text message
   Received message: This is a text message
   [INFO]
   [INFO] --- artemis-maven-plugin:2.6.1.amq-720004-redhat-1:cli (stop) @ queue ---
   server-out:2018-12-05 16:37:59,519 INFO  [org.apache.activemq.artemis.core.server]
   AMQ221002: Apache ActiveMQ Artemis Message Broker version 2.6.1.amq-720004-redhat-
   1 [06f529d3-f8d6-11e8-9bea-0800271b03bd] stopped, uptime 3.734 seconds
   server-out:Server stopped!
   [INFO] ------------------------------------------------------------------------
   [INFO] BUILD SUCCESS
   [INFO] ------------------------------------------------------------------------
   [INFO] Total time:  48.681 s
   [INFO] Finished at: 2018-12-05T16:37:59-05:00
   [INFO] ------------------------------------------------------------------------
   ```

注記

一部のサンプルプログラムは UDP クラスタリングを使用しますが、デフォルトではお使いの環境で機能しない場合があります。これらの例を正常に実行するには、224.0.0.0 に送付されたトラフィックをループバックインターフェイスにリダイレクトします。

```
$ sudo route add -net 224.0.0.0 netmask 240.0.0.0 dev lo
```

第6章 次のステップ

AMQ Broker をインストールし、デフォルト設定でスタンドアロンのブローカーを作成したら、メッセージング要件を満たすように設定し、メッセージングクライアントアプリケーションを接続して監視および管理できます。これらの目標を達成するのに役立つ追加のリソースがあります。

ブローカーの設定

Configuring AMQ Broker を使用して、要件を満たすようにブローカーを設定します。以下を設定できます。

- クライアント接続を受け入れるブローカー
- アドレス空間 (ポイントツーポイントおよびパブリッシュサブスクライブメッセージングを含む)
- メッセージの永続性
- ブローカーのリソース消費 (リソース制限、メッセージのページング、大規模なメッセージのサポートを含む)
- 重複メッセージの検出
- ロギング

ブローカーのセキュリティー保護

Configuring AMQ Broker を使用して、以下のメソッドのいずれかを実装してブローカーを保護します。

- ゲスト/匿名アクセスの制御
- 基本的なユーザーとパスワードによるアクセス制御
- 証明書ベースのアクセス制御
- LDAP インテグレーション
- Kerberos インテグレーション

クラスター化および高可用性の設定

Configuring AMQ Broker を使用して、さらにブローカーを追加してブローカークラスターを形成し、メッセージングのスループットを向上させます。また、メッセージングの信頼性を向上させるために、高可用性を設定することもできます。

メッセージングクライアントアプリケーションの作成

AMQ クライアントの概要 を参照して、AMQ クライアントについて理解すると共に、ブローカーに接続してメッセージを生成および使用するメッセージングクライアントアプリケーションの作成にどのように役立つのかを理解します。

ブローカーの監視および管理

Managing AMQ Broker を使用して、ブローカーの実行後にこれを監視および管理します。

付録A サブスクリプションの使用

AMQ は、ソフトウェアサブスクリプションから提供されます。サブスクリプションを管理するには、Red Hat カスタマーポータルでアカウントにアクセスします。

A.1. アカウントへのアクセス

手順

1. access.redhat.com に移動します。
2. アカウントがない場合は作成します。
3. アカウントにログインします。

A.2. サブスクリプションのアクティベート

手順

1. access.redhat.com に移動します。
2. **My Subscriptions** に移動します。
3. **Activate a subscription** に移動し、16 桁のアクティベーション番号を入力します。

A.3. リリースファイルのダウンロード

.zip、.tar.gz およびその他のリリースファイルにアクセスするには、カスタマーポータルを使用してダウンロードする関連ファイルを検索します。RPM パッケージまたは Red Hat Maven リポジトリーを使用している場合、この手順は必要ありません。

手順

1. ブラウザーを開き、access.redhat.com/downloads で Red Hat カスタマーポータルの **Product Downloads** ページにログインします。
2. **JBOSS INTEGRATION AND AUTOMATION**カテゴリーの **Red Hat AMQ** エントリーを見つけます。
3. 必要な AMQ 製品を選択します。**Software Downloads** ページが開きます。
4. コンポーネントの **Download** リンクをクリックします。

A.4. パッケージ用システムの登録

この製品の RPM パッケージを Red Hat Enterprise Linux にインストールするには、システムが登録されている必要があります。ダウンロードしたリリースファイルを使用している場合、この手順は必要ありません。

手順

1. access.redhat.com に移動します。

2. **Registration Assistant** に移動します。
3. ご使用の OS バージョンを選択し、次のページに進みます。
4. システムの端末にリスト表示されたコマンドを使用して、登録を完了します。

システムを登録する方法は、以下のリソースを参照してください。

- Red Hat Enterprise Linux 7 - システム登録およびサブスクリプション管理
- Red Hat Enterprise Linux 8 - システム登録およびサブスクリプション管理

付録B APACHE MAVEN の概要

Apache Maven は分散型構築自動化ツールで、ソフトウェアプロジェクトの作成、ビルド、および管理を行うために Java アプリケーション開発で使用されます。これを使用して、AMQ Broker インストールに含まれる AMQ Broker サンプルプログラムを実行できます。

AMQ Broker のサンプルプログラムを実行するには、複数の Maven コンポーネントと連携する必要があります。

プロジェクトオブジェクトモデル (POM) ファイル

プロジェクトのビルド方法に関する情報を保存します。

リポジトリー

ビルドのアーティファクトおよび依存関係を保持します。

Maven 設定ファイル

ユーザー固有の設定情報を保存します。

B.1. MAVEN POM ファイル

Maven は Project Object Model (POM) ファイルと呼ばれる標準の設定ファイルを使用して、プロジェクトの定義や構築プロセスの管理を行います。これにより、プロジェクトが適切かつ統一された状態でビルドされるようになります。POM ファイルは XML ファイル (**pom.xml**) です。

Maven は設定より規約 (Convention over Configuration) を優先します。そのため、POM ファイルには最小限の設定およびその他のデフォルト値が必要です。POM ファイルは Maven プロジェクトの以下の情報を定義できます。

- ソース、テスト、およびターゲットディレクトリーの場所
- プロジェクトの依存関係
- プラグインリポジトリー
- プロジェクトを実行できるゴール
- バージョン、説明、開発者、メーリングリスト、ライセンスなどのプロジェクトに関する追加情報。

例B.1 サンプル pom.xml ファイル

この基本的な **pom.xml** ファイルは、POM ファイルの最小要件を示しています。

```
<project>
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.jboss.app</groupId>
  <artifactId>my-app</artifactId>
  <version>1</version>
</project>
```

関連情報

- **pom.xml** ファイルのスキーマは http://maven.apache.org/maven-v4_0_0.xsd にあります。

- POM ファイルの詳細は Apache Maven Project POM Reference を参照してください。

B.2. MAVEN リポジトリー

Maven リポジトリーには、Java ライブラリー、プラグイン、およびその他のビルドアーティファクトが格納されます。リポジトリーはローカルまたはリモートのいずれかになります。

デフォルトのパブリックリポジトリーは Maven 2 Central Repository ですが、開発チームの間で共通のアーティファクトを共有する目的で、社内のプライベートリポジトリーとすることが可能です。

また、サードパーティーのリポジトリーも利用できます。AMQ には、テスト済みのサポート対象バージョンの AMQ 7.10 Java パッケージおよび依存関係が含まれる Maven リポジトリーが含まれています。

関連情報

- Maven リポジトリーの詳細は、Introduction to Repositoriesを参照してください。

B.3. MAVEN 設定ファイル

Maven の **settings.xml** ファイルには Maven のユーザー固有の設定情報が含まれています。開発者の ID、プロキシー情報、ローカルリポジトリーの場所、ユーザー固有のその他の設定など、**pom.xml** ファイルで配布されてはならない情報が含まれます。

settings.xml ファイルは以下の 2 つの場所にあります。

- Maven インストール内:
  settings.xml ファイルは **<maven-install-dir>/conf/** ディレクトリーにあります。これらの設定は **global** 設定と呼ばれます。デフォルトの Maven 設定ファイルはコピー可能なテンプレートであり、これを基にユーザー設定ファイルを設定することが可能です。

- ユーザーのインストール内:
  settings.xml ファイルは **${user.home}/.m2/** ディレクトリーにあります。Maven とユーザーの **settings.xml** ファイルが両方存在する場合、内容はマージされます。重複する内容がある場合は、ユーザーの **settings.xml** ファイルが優先されます。

例B.2 Maven 設定ファイル

```
<?xml version="1.0" encoding="UTF-8"?>
<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0
http://maven.apache.org/xsd/settings-1.0.0.xsd">
  <profiles>
    <!-- Configure the JBoss AMQ Maven repository -->
    <profile>
      <id>jboss-amq-maven-repository</id>
      <repositories>
        <repository>
          <id>jboss-amq</id>
          <url>file:///path/to/repo/</url>
          <releases>
            <enabled>true</enabled>
          </releases>
          <snapshots>
```

```
            <enabled>false</enabled>
          </snapshots>
        </repository>
      </repositories>
      <pluginRepositories>
        <pluginRepository>
          <id>jboss-amq-maven-plugin-repository</id>
          <url>file://path/to/repo</url>
          <releases>
            <enabled>true</enabled>
          </releases>
          <snapshots>
            <enabled>false</enabled>
          </snapshots>
        </pluginRepository>
      </pluginRepositories>
    </profile>
  </profiles>
  <activeProfiles>
    <!-- Optionally, make the repository active by default -->
    <activeProfile>jboss-amq-maven-repository</activeProfile>
  </activeProfiles>
</settings>
```

関連情報

- **settings.xml** ファイルのスキーマは http://maven.apache.org/xsd/settings-1.0.0.xsd にあります。
- Maven 設定ファイルの詳細は、Settings Referenceを参照してください。

改訂日時: 2024-06-11