



Red Hat AMQ 2021.Q3

AMQ Broker の使用

AMQ Broker 7.9 での使用について

Red Hat AMQ 2021.Q3 AMQ Broker の使用

AMQ Broker 7.9 での使用について

Enter your first name here. Enter your surname here.

Enter your organisation's name here. Enter your organisational division here.

Enter your email address here.

法律上の通知

Copyright © 2022 | You need to change the HOLDER entity in the en-US/Getting_Started_with_AMQ_Broker.ent file |.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

概要

このガイドでは、AMQ Broker を使い始める方法を説明しています。

目次

多様性を受け入れるオープンソースの強化	3
第1章 概要	4
1.1. 主な特長	4
1.2. サポート対象の標準およびプロトコル	4
1.3. サポートされる構成	4
1.4. 本書の表記慣例	4
sudo コマンド	5
本書におけるファイルパスの使用	5
交換可能な値	5
第2章 AMQ BROKER について	6
2.1. ブローカーインスタンス	6
2.2. メッセージの永続性	6
ジャーナルベースの永続性	6
データベースの永続性	6
2.3. リソースの消費	7
2.4. 監視および管理	7
第3章 AMQ BROKER のインストール	9
3.1. AMQ BROKER アーカイブのダウンロード	9
3.2. LINUX での AMQ BROKER アーカイブの展開	9
3.3. WINDOWS システムでの AMQ BROKER アーカイブの抽出	10
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	10
第4章 スタンドアロンブローカーの作成	12
4.1. ブローカーインスタンスの作成	12
4.2. ブローカーインスタンスの起動	14
4.3. テストメッセージの生成および使用	15
4.4. ブローカーインスタンスの停止	16
第5章 AMQ BROKER サンプルの実行	18
5.1. AMQ BROKER の例を実行するためのマシンの設定	18
5.1.1. Maven のダウンロードおよびインストール	18
5.1.2. AMQ Maven リポジトリのダウンロードおよびインストール	18
5.1.3. Maven 設定ファイルの設定	18
5.2. AMQ BROKER のサンプルプログラム	20
5.3. AMQ BROKER サンプルプログラムの実行	20
第6章 次のステップ	23
付録A サブスクリプションの使用	24
A.1. アカウントへのアクセス	24
A.2. サブスクリプションのアクティベート	24
A.3. リリースファイルのダウンロード	24
A.4. パッケージ用システムの登録	24
付録B APACHE MAVEN の概要	26
B.1. MAVEN POM ファイル	26
B.2. MAVEN リポジトリ	27
B.3. MAVEN 設定ファイル	27

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、[弊社](#) の CTO、Chris Wright の [メッセージ](#) を参照してください。

第1章 概要

AMQ Broker は、ActiveMQ Artemis をベースにした高性能なメッセージング実装です。非同期ジャーナルを使用してメッセージを高速に保存し、複数の言語、プロトコル、プラットフォームをサポートしています。

1.1. 主な特長

AMQ Broker には、以下の機能があります。

- クラスタリングおよび高可用性オプション
- 迅速でネイティブの IO 永続性
- ローカルランザクションをサポート
- 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サポート用に Java で記述されています。
- 複数の管理インターフェース: AMQ 管理コンソール、管理 API、および JMX

1.2. サポート対象の標準およびプロトコル

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:
 - コアプロトコル
 - 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 バージョンには含まれません。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 のサポートされる構成に関する最新情報は、Red Hat Customer Portal の記事 [Red Hat AMQ 7 Supported Configurations](#) を参照してください。

1.4. 本書の表記慣例

本書では、**sudo** コマンド、ファイルパス、および置換可能な値について、以下の規則を使用しています。

sudo コマンド

本書では、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 コマンド](#)」を参照してください。

本書におけるファイルパスの使用

本書では、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。置換可能な値は小文字で、角括弧 (<>) で囲まれ、イタリック体と **monospace** フォントを使用してスタイリングされています。複数の単語はアンダースコア(_)で区切られています。

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 を使用すると、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間に異種システムを結合できます。AMQ Broker を使用する前に、提供できる機能を理解しておく必要があります。

2.1. ブローカーインスタンス

AMQ Broker にインストールされている 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** に対して「home」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数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、その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1度更新する必要があります。

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構成され、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、ブローカーに予期せずシャットダウンしてもメッセージが失われないようにします。AMQ Broker には、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つのオプションを利用できます。

ジャーナルベースの永続性

デフォルトのメソッドは、ファイルシステムに保存されてい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、空のデータで入力されます。クライアントは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つがいっぱいになると、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 トランザクションなどの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 インターフェースが必要です。AMQ Broker は以下をサポートします。

Linux 非同期 IO (AIO)

通常、AIO はベスト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](#) でサポートされる構成でサポートされていることを確認します。

Java NIO

Java NIO は優れたパフォーマンスを実現し、Java 6 以降のランタイムを備えたプラットフォームで実行できます。

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あり、高パフォーマンスのデータベースプラットフォームを使用している場合や、会社ポリシーでデータベースを使用した

場合は、このオプションが推奨されます。

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すると、データベースに保存された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表示するには、「[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数多くのオプションがあります。

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、ユーザーは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します。

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 は大量のメッセージを含む大規模なキューをサポートでき、また大量のメモリーでも稼働します。ブローカーがメモリー容量を超えるメッセージを取得したら、ディスクにメッセージをページングします。このページングプロセスは透過的であり、必要に応じてブローカーページのメッセージがメモリー内に出入りします。

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のすべてのメッセージのサイズが最大サイズを超えると、アドレスの追加メッセージがアドレスのページファイルに表示されません。

大きなメッセージ

AMQ Broker では、制限されたメモリーリソースで実行される場合でも Huge Page メッセージを送受信できます。大きなメッセージをメモリーに格納するオーバーヘッドを回避するには、これらの大き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 を設定します。

2.4. 監視および管理

AMQ Broker には、ブローカーの監視および管理に使用できるツールが複数提供されます。

AMQ 管理コンソール

AMQ Management Console は、Web ブラウザーからアクセスできる Web インターフェースです。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 は、メッセージジャーナルを管理する複数のコマンドも提供します。

Management 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の設定を変更し、新規リソースの作成、これらのリソースを検査し、それらと対話できます。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。

- Java Management Extensions (JMX): 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 インターフェース経由で公開されます。
- JMS API: 管理操作は、特別な管理 JMS キューへの標準の JMS メッセージを使用して送信されます。

Logs

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をログに記録します。ログレベル、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、コンテンツを抽出する必要があります。また、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が AMQ Broker でサポートされる構成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。
製品のダウンロード 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 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で必要な AMQ Broker 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 リンクをクリックします。

3.2. LINUX での AMQ BROKER アーカイブの展開

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展開します。

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展開します。

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

amq-broker-7.9 と似たディレクトリーが作成されます。本書では、この場所は **<install_dir>** と呼ばれます。

3.3. WINDOWS システムでの AMQ BROKER アーカイブの抽出

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、そこでコンテンツを展開します。

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-7.9 と似たディレクトリーが作成されます。本書では、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

アーカイブを展開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になります。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。

このディレクトリー...	contains...
<install_dir>/web/api	API ドキュメント

このディレクトリー...	contains...
<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 の実行時にロードされる 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

ローカルマシンでスタンドアロンブローカーインスタンスを作成し、起動してテストメッセージを生成および消費することで、AMQ Broker ですぐに起動できます。

前提条件

- AMQ Broker がインストールされていること。
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、ローカルマシンで簡単なスタンドアロン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を作成します。

使用している場合は...	方法
Red Hat Enterprise Linux (RHEL)	<ol style="list-style-type: none"> 1. ブローカーインスタンスのロケーションとして使用する新しいディレクトリを作成します。 <pre>\$ sudo mkdir /var/opt/amq-broker</pre> 2. インストール時に作成したユーザーを割り当てます。 <pre>\$ sudo chown -R amq-broker:amq-broker /var/opt/amq-broker</pre>
Windows	Windows Explorer を使用して新しいフォルダーを作成し、ブローカーインスタンスの場所として機能します。

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

使用している場合は...	方法
--------------	----

使用している場合は...	方法
Red Hat Enterprise Linux (RHEL)	<ol style="list-style-type: none"> インストール時に作成したユーザーアカウントに切り替えます。 <pre>\$ su - amq-broker</pre> 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。 <pre>\$ cd /var/opt/amq-broker</pre> 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。 <pre>\$ <install_dir>/bin/artemis create mybroker</pre>
Windows	<ol style="list-style-type: none"> 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。 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。 <pre>> <install_dir>\bin\artemis.cmd create mybroker</pre>

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
```

```

  _ _ _ _ _
 / \ | V | / _ \ | _ \ | | | |
 / \ | \ / | | | | | | | | | | | |
 / \ | | M | | | | | | | | | | | |
 / \ | _ \ | | | | | | | | | | | |
 / \ | \ \ | | | | | | | | | | | |

```

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,largeMessagesDirectory=./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、以下の情報で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、サポートされる任意のプロトコル (CORE、MQTT、AMQP、STOMP、HORNETQ、および OPEN) からの接続を許可できます。各プロトコルには個別のポートも存在します。
- 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) は <http://localhost:8161/jolokia> から入手できます。

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、これ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。これには、いくつかのテストメッセージを作成し、それらをブローカーに送信し、それらを消費する必要があります。

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許可す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> に移動します。

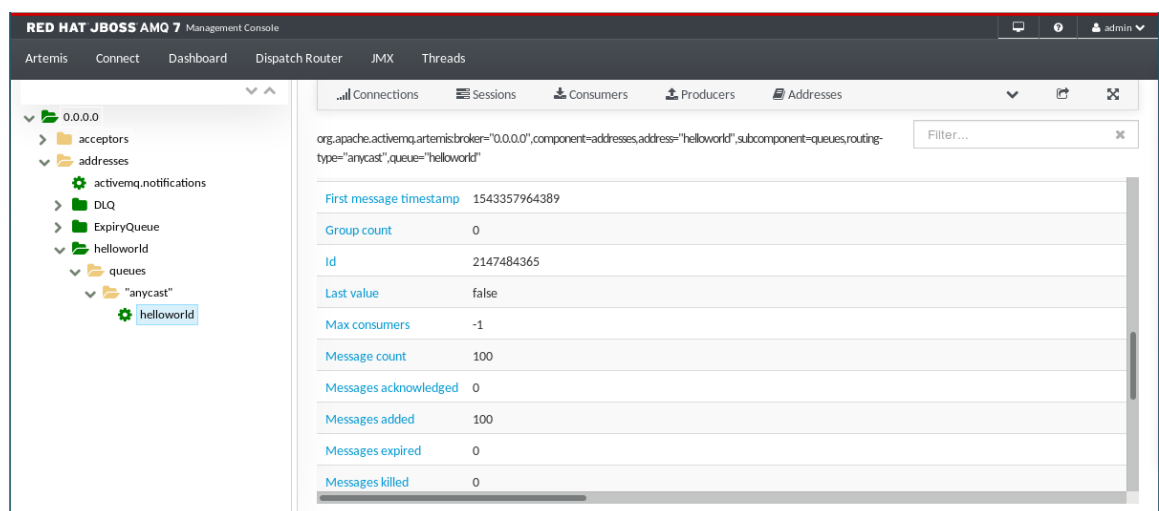
- b. ブローカーインスタンスの作成時に作成したデフォルトのユーザー名およびパスワードを使用して、コンソールにログインします。

Attributes タブが表示されます。

- c. **Attributes** タブでメニュー:[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されたメッセージの 50 を消費します。

```
$ /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、**メッセージ数** 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. ブローカーが稼働し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 メッセージを消費します。

```
$ /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、**メッセージ数** 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プログラムが多数含まれています。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、最初に Apache Maven および AMQ Maven リポジトリ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。次に、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 の例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、最初に Maven および AMQ Maven リポジトリ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

Maven は AMQ Broker サンプルを実行するために必要になります。

手順

1. [Apache Maven Download](#) ページに移動して、お使いのオペレーティングシステムの最新ディストリビューションをダウンロードします。
2. オペレーティングシステムの 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。

その他のリソース

- Maven の詳細は、「[Introduction to Apache Maven](#)」を参照してください。

5.1.2. AMQ Maven リポジトリ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。
製品のダウンロード ページが表示されます。
2. **Integration and Automation** セクションで、**Red Hat AMQ Broker** 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で必要な AMQ Broker バージョンを選択します。
4. **Releases** タブで、AMQ Broker Maven リポジトリの **Download** リンクをクリックします。
AMQ Maven リポジトリファイルは zip ファイルとしてダウンロードされます。
5. マシンで、AMQ リポジトリファイルを選択したディレクトリーに展開します。
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 リポジトリのダウンロードおよびインストール後に、リポジトリを 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。

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

```

<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 機能およびサポートされるメッセージングプロトコルの使用方法を示す 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。

- 特長
ブローカー固有の機能 (以下を含む)。
 - クラスタ化: 負荷分散および分散機能を示す例
 - 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対象とするサンプル
 - サブモジュール: 統合外部モジュールの例
- プロトコル
サポートされるメッセージングプロトコルの例:
 - AMQP
 - MQTT
 - OpenWire
 - STOMP

その他のリソース

- 各サンプルプログラムの説明は、[Apache Artemis](#) ドキュメントの「サンプル」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プログラムが多数含まれています。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

- マシンが AMQ Broker サンプルを実行するよう設定されている必要があります。詳細は、「[AMQ Broker の例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。以下は例になります。

```
$ 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 はブローカーを開始し、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。

この場合、**queue** のサンプルプログラムが実行され、プロデューサーを作成し、テストメッセージを送信してから、メッセージを受信するコンシューマーを作成します。

```
$ 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

一部のプログラムでは UDP クラスタリングを使用しますが、デフォルトではお使用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 にダイレクトされたトラフィックをループバックインターフェー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。これらのゴールを完了するために、追加のリソースを使用できます。

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、大きいメッセージサポートを含む)
- 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

[AMQ Broker の設定](#) を使用して、以下のメソッドのいずれかを実装し、ブローカーを保護します。

- ゲスト/匿名のアクセス制御
- 基本的なユーザーとパスワードのアクセス制御
- 証明書ベースのアクセス制御
- LDAP 統合
- Kerberos の統合

クラスタリングと高可用性の設定

「[Configuring AMQ Broker](#)」を使用して、追加のブローカーを追加してブローカークラスターを作成し、メッセージングスループットを増やします。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 Clients Overview](#) を使用して、AMQ Clients を確認し、ブローカーに接続し、メッセージを生成および消費するメッセージングクライアントアプリケーションを作成できます。

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 (またはブローカー) の実行後に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. サブスクリプション 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 リポジトリを使用している場合は、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。ダウンロードしたりリリースファイルを使用している場合は、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. システムの端末に一覧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 は分散型ビルド自動化ツールで、ソフトウェアプロジェクトの作成、管理、ビルドを行う Java アプリケーション開発で使用されます。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するのに使用します。

AMQ Broker のサンプルプログラムを実行するには、複数の Maven コンポーネントと対話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

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。こうすることで、プロジェクトが適切にビルドされ、統一されていることを確認します。POM ファイルは XML ファイル (**pom.xml**) です。

Maven は「設定に注意をする」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デフォルトの他のすべての値が必要です。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
- 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、プロジェクトに関する補足情報。

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

Maven リポジトリには、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は [Maven 2 Central Repository](#) ですが、共通のアーティファクトが開発チーム間で共有されるように、社内のプライベートおよび内部リポジトリ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も利用できます。AMQ には、テスト済みのサポートされたバージョンの AMQ 7.9 Java パッケージおよび依存関係が含まれる Maven リポジトリが含まれています。

その他のリソース

- Maven リポジトリの詳細は、「[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、プロキシ情報、ローカルリポジトリの場所、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 インストールで以下を行います。
settings.xml ファイルは `<maven-install-dir>/conf/` ディレクトリにあります。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、ユーザー設定ファイルにコピーおよび使用できるテンプレートです。
- ユーザーのインストールで以下を行います。
settings.xml ファイルは `${user.home}/.m2/` ディレクトリにあります。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>
        </repository>
      </repositories>
    </profile>
  </profiles>
</settings>
```

```
<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： 2022-04-03 09:56:39 +1000