Red Hat AMQ Clients 2.11

AMQ クライアントの概要

AMQ Clients 2.11 向け

Last Updated: 2023-06-28

Red Hat AMQ Clients 2.11 AMQ クライアントの概要

AMQ Clients 2.11 向け

法律上の通知

概要

このドキュメントでは、AMQ Clients 2.11 の機能およびコンポーネントを説明します。また、本リリースでサポートされる一般的なユースケースと設計パターンも示しています。

目次

多様性を受け入れるオープンソースの強化

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。

AMQ Clients は AMQP 1.0 および JMS クライアント、アダプター、およびライブラリーのスイートです。これには、JMS 2.0 のサポートおよび既存のアプリケーションへのインテグレーションを可能にする新しいイベント駆動型 API が含まれます。

AMQ クライアントは Red Hat AMQ の一部です。詳細は、Red Hat AMQ 7 の紹介 を参照してください。

第1章 主な特長

- オープンスタンダードプロトコル - AMQP 1.0
- 業界標準の API - JMS 1.1 および 2.0
- 高速で効率的なメッセージングのための新しいイベント駆動型 API
- 他のプラットフォームやコンポーネントと統合するためのアダプター
- 幅広い言語のサポート - C++、Java、JavaScript、Python、Ruby、および .NET
- 幅広い可用性 - Linux、Windows、および JVM ベースの環境

第2章 コンポーネント

2.1. AMQP クライアント

AMQ クライアントには、AMQP 1.0 メッセージング API のスイートが含まれています。AMQP は、豊富なメッセージング機能を備えた ISO 標準の汎用メッセージングプロトコルです。AMQ Broker および AMQ Interconnect は AMQP 1.0 をサポートしているため、AMQP 1.0 クライアントと相互運用できます。

- AMQ C++
- AMQ JavaScript
- AMQ JMS (Java)
- AMQ .NET
- AMQ Python
- AMQ Ruby

2.2. JMS クライアント

AMQ クライアントは、広く使用されている Java Message Service (JMS) API の複数の実装を提供します。

- AMQ JMS - AMQ JMS は、完全な AMQP 1.0 サポートを提供し、任意の AMQ AMQP 1.0 サーバーまたはサービスで動作します。
- AMQ Core Protocol JMS - ActiveMQ Artemis Core プロトコルに基づく既存のアプリケーションをサポートするために、AMQ Core Protocol JMS クライアントが含まれています。

2.3. アダプターおよびライブラリー

AMQ Clients には、他のプラットフォームやコンポーネントと統合するためのコンポーネントが含まれています。

- AMQ JMS Pool - JMS リソースの効率的な使用をサポートするために、AMQ には AMQ JMS Pool ライブラリーが含まれています。これにより、JMS API で定義された標準のライフサイクルを超えて接続リソースを再利用できます。
- AMQ Spring Boot Starter - AMQP 1.0 メッセージングを使用する標準の Spring アプリケーションを構築できる Spring Boot スターター。
- Qpid JMS 用の Quarkus 拡張機能 (AMQ JMS) - AMQP 1.0 メッセージングを使用する Quarkus アプリケーションを構築できるようにする Quarkus 拡張機能。

2.4. コンポーネントの互換性

次の表に、AMQ Clients コンポーネントでサポートされている言語、プラットフォーム、プロトコル、およびサーバーを示します。

コンポーネント	言語	プラットフォーム	プロトコル	Servers
AMQ C++	C++	Linux、Windows	AMQP 1.0	AMQ Broker、AMQ Interconnect、および A-MQ 6
AMQ JavaScript	JavaScript	Linux、Windows、ブラウザー	AMQP 1.0	AMQ Broker、AMQ Interconnect、および A-MQ 6
AMQ JMS	Java	JVM	AMQP 1.0	AMQ Broker、AMQ Interconnect、および A-MQ 6
AMQ JMS Pool	Java	JVM	-	-
AMQ Core Protocol JMS	Java	JVM	Core Protocol	AMQ Broker および A-MQ6
AMQ .NET	C#	Linux、Windows	AMQP 1.0	AMQ Broker、AMQ Interconnect、および A-MQ 6
AMQ Python	Python	Linux、Windows	AMQP 1.0	AMQ Broker、AMQ Interconnect、および A-MQ 6
AMQ Ruby	Ruby	Linux	AMQP 1.0	AMQ Broker、AMQ Interconnect、および A-MQ 6
AMQ Spring Boot Starter	Java	JVM	AMQP 1.0	AMQ Broker、AMQ Interconnect、および A-MQ 6
Qpid JMS の Quarkus 拡張機能 (AMQ JMS)	Java	JVM	AMQP 1.0	AMQ Broker、AMQ Interconnect、および A-MQ 6

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。

第3章 イベント駆動型 API

AMQ クライアントで提供される API の多くは、非同期のイベント駆動型 API です。これらには、C++、JavaScript、Python、および Ruby API が含まれます。

これらの API は、ネットワークアクティビティーに応答してアプリケーションイベント処理関数を実行することによって機能します。ライブラリーはネットワーク I/O を監視し、イベントを発生させます。イベントハンドラーは、メインライブラリースレッドで順番に実行されます。

イベントハンドラーはメインライブラリースレッドで実行されるため、ハンドラーコードに長時間実行されるブロッキング操作を含めることはできません。イベントハンドラーでブロックすると、すべてのライブラリーの実行がブロックされます。長いブロッキング操作を実行する必要がある場合は、別のスレッドで呼び出す必要があります。イベント駆動型 API には、ライブラリースレッドとアプリケーションスレッド間の調整をサポートするクロススレッド通信機能が含まれています。

イベントハンドラーでのブロックを回避する

イベントハンドラーで長時間実行されるブロッキング呼び出しは、すべてのライブラリーの実行を停止し、ライブラリーが他のイベントを処理したり、定期的なタスクを実行したりできないようにします。長時間実行されるブロッキング手順は、常に別のアプリケーションスレッドで開始してください。

第4章 AMQP

AMQP は、メッセージを確実に送受信するためのオープンインターネットプロトコルです。複数のソフトウェアベンダーや主要な機関によってサポートされています。AMQP 1.0 は、2012 年に OASIS 標準になり、2014 年に ISO 標準になりました。

- セッション多重化を備えたフレーム化されたプロトコル
- ピアツーピアおよびクライアントサーバー接続をサポートする
- ロスレスデータ交換のための標準タイプのシステムを提供する
- 分散システムの信頼性を高めるためのフロー制御、ハートビート、およびリソース制限を提供する
- スペース効率の高いバイナリーエンコーディングおよびパイプラインを使用して、レイテンシーを削減する

4.1. AMQP 配信保証

決済用の AMQP モデルは、メッセージ配信のライフサイクルに基づいています。リンクの両端に、メッセージ転送を表すエンティティーが作成され、一定期間存在し、最後に解決されます。つまり、忘れることができます。配信のライフサイクルを組み合わせた場合は、関心のある 4 つのイベントがあります。

- 配信は送信者で作成されます。
- 配信は受信者で作成されます。
- 配達は送信者で決済されます。
- 配達は受信者で決済されます。

送信者と受信者は同時に動作しているため、これらのイベントはさまざまな順序で発生する可能性があり、これらのイベントの順序により、メッセージ配信の保証が異なります。

最大 1 回の配信

最大 1 回の配信は、事前決済またはファイアアンドフォーゲット配信とも呼ばれます。

1. 配信は送信者で作成されます。
2. 配達は送信者で決済されます。
3. 配信は受信者で作成されます。
4. 配達は受信者で決済されます。

この設定では、送信者は受信者に到着する前に配信を決済 (つまり、忘れる) し、実行中の配信に何かが起こった場合、送信者は再送の根拠がありません。

このモードは、定期的なセンサーデータなど、一時的なメッセージの損失が許容されるアプリケーション、またはアプリケーション自体が障害を検出して再送信できるアプリケーションに適しています。

少なくとも 1 回の配信

1. 配信は送信者で作成されます。

2. 配信は受信者で作成されます。

3. 配達は受信者で決済されます。

4. 配達は送信者で決済されます。

この設定では、受信者は配信を受信したときに決済し、送信者は受信者が決済したことを確認すると決済します。実行中の配達に問題が発生した場合、送信者は再送することができます。ただし、受信者はすでに配信を忘れているため、再送するとメッセージの配信が重複します。アプリケーションは、一意のメッセージ ID を使用して重複を除外できます。

第5章 重要事項

5.1. 優先クライアント

一般に、AMQP 1.0 標準をサポートする AMQ クライアントは、新しいアプリケーション開発に適しています。ただし、以下の例外が適用されます。

- 実装で分散トランザクションが必要な場合は、AMQ Core Protocol JMS を使用してください。
- ドメインで MQTT または STOMP が必要な場合 (たとえば、IoT アプリケーションの場合) は、コミュニティーでサポートされている MQTT または STOMP クライアントを使用します。

5.2. レガシークライアント

- **AMQ OpenWire JMS クライアントの廃止**
  AMQ OpenWire JMS クライアントは、AMQ7 で非推奨になりました。このクライアントのユーザーは、AMQ JMS または AMQ Core Protocol JMS に移行することが推奨されます。
- **CMS および NMS API の非推奨**
  ActiveMQ CMS および NMS メッセージング API は、AMQ7 で非推奨になりました。CMS API のユーザーは AMQC++ に移行し、NMS API のユーザーは AMQ.NET に移行することが推奨されます。CMS および NMS API は、AMQ7 の機能が低下している可能性があります。
- **従来の AMQC++ クライアントの廃止**
  従来の AMQC++ クライアント (以前は MRG Messaging で提供されていた C ++ クライアント) は、AMQ7 では非推奨になっています。この API のユーザーは AMQC++ に移行することが推奨されます。
- **Core API はサポート対象外**
  Artemis Core API クライアントはサポートされていません。このクライアントは、サポートされている AMQ Core Protocol JMS クライアントとは異なります。

第6章 重要なリンク

- Red Hat AMQ でサポートされる設定
- Red Hat AMQ コンポーネントの詳細

Revised on 2022-11-12 21:30:37 +1000