Red Hat Integration 2022.Q1

Red Hat Integration 2022.Q1 リリースノート

Red Hat Integration の新機能

Last Updated: 2023-03-25

Red Hat Integration 2022.Q1 Red Hat Integration 2022.Q1 リリースノート

Red Hat Integration の新機能

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

概要

Red Hat Integration プラットフォームについて説明し、本リリースの新機能に関する最新情報を提供します。

目次

第1章 RED HAT INTEGRATION 4

第2章 QUARKUS リリースノートの CAMEL エクステンション 5
2.1. QUARKUS 機能の CAMEL エクステンション 5
2.2. サポートされるプラットフォーム、設定、データベース、およびエクステンション 5
2.3. テクノロジープレビューのエクステンション 5
2.4. 既知の問題 5
2.5. 重要事項 6
2.6. 解決した問題 7
2.7. 非推奨になった CAMEL EXTENSIONS FOR QUARKUS の機能 8
2.8. 関連情報 8

第3章 DEBEZIUM リリースノート 9
3.1. DEBEZIUM データベースコネクター 9
3.2. DEBEZIUM でサポートされる構成 10
3.3. DEBEZIUM のインストールオプション 10
3.4. DEBEZIUM の新機能 10
3.5. 非推奨の DEBEZIUM 機能 12

第4章 CAMEL K リリースノート 13
4.1. CAMEL K の新機能 13
4.2. SUPPORTED CONFIGURATIONS 13
4.2.1. Camel K Operator のメタデータ 13
4.3. 重要事項 14
4.4. サポートされる CAMEL QUARKUS エクステンション 14
4.4.1. サポートされる Camel Quarkus コネクターエクステンション 15
4.4.2. サポートされる Camel Quarkus データフォーマットエクステンション 16
4.4.3. サポートされる Camel Quarkus 言語エクステンション 16
4.4.4. サポートされる Camel K トレイト 16
4.5. サポート対象の KAMELETS 17
4.6. CAMEL K の既知の問題 21
4.7. CAMEL K の修正された問題 22
4.7.1. Camel K 1.6.4 で改良された機能 22
4.7.2. Camel K 1.6.4 で解決されたバグ 22

第5章 SERVICE REGISTRY リリースノート 25
5.1. SERVICE REGISTRY のインストールオプション 25
5.2. SERVICE REGISTRY プラットフォームコンポーネントのバージョン 25
5.3. SERVICE REGISTRY の新機能 25
Service Registry のセキュリティー 25
Service Registry コア 26
Service Registry データストレージ 26
Service Registry v2 REST API 26
Service Registry Operator 26
Service Registry ユーザーのドキュメントおよび例 27
5.4. 非推奨および削除された SERVICE REGISTRY の機能 27
非推奨となった Service Registry の機能 27
Service Registry で削除された機能 27
5.5. SERVICE REGISTRY デプロイメントの移行 27
5.6. SERVICE REGISTRY で解決された問題 27
Service Registry core で解決された問題 27
Service Registry Operator で解決された問題 29

5.7. SERVICE REGISTRY の既知の問題 29
Service Registry コアの既知の問題 29
Service Registry operator の既知の問題 29

第6章 RED HAT INTEGRATION の OPERATOR 30
6.1. OPERATOR とは 30
6.2. RED HAT INTEGRATION コンポーネント OPERATOR 30
6.2.1. 3scale Operators 30
6.2.2. AMQ Operator 30
6.2.3. Camel K Operator 31
6.2.4. Fuse Operator 31
6.2.5. Service Registry Operator 31
6.3. RED HAT INTEGRATION OPERATOR (非推奨) 31
6.3.1. サポートされるコンポーネント 32
6.3.2. サポートライフサイクル 33
6.3.3. 修正された問題 33

第1章 RED HAT INTEGRATION

Red Hat Integration は、ハイブリッド環境およびマルチクラウド環境全体でコンテナーベースの統合サービスを作成、拡張、デプロイするための包括的な統合およびイベント処理技術です。Red Hat Integration は、デジタル環境で必要となるアプリケーションとシステム間でデータを接続および共有するために組織が使用できる、アジャイルで API 中心の分散ソリューションを提供します。

Red Hat Integration には、以下の機能が含まれています。

- リアルタイムのメッセージング
- データセンター間のメッセージストリーミング
- API の接続
- アプリケーションコネクター
- エンタープライズ統合パターン
- API 管理
- データの変換
- サービスの構成とオーケストレーション

関連情報

- エンタープライズ統合について理解する

第2章 QUARKUS リリースノートの CAMEL エクステンション

2.1. QUARKUS 機能の CAMEL エクステンション

高速起動と低 RSS メモリー

Quarkus の最適化されたビルドタイムおよび事前 (AOT: Ahead-of-Time) コンパイラ機能を使用すると、ビルド時に Camel アプリケーションを事前に設定できるため、起動時間を短縮できます。

アプリケーションジェネレーター

Quarkus アプリケーションジェネレーター を使用して、アプリケーションをブートストラップし、エクステンションエコシステムを検出します。

高度な設定が可能

Quarkus アプリケーションの Camel エクステンションの重要な側面はすべて、CDI (Contexts and Dependency Injection) または設定プロパティーを使用してプログラム的に設定できます。デフォルトでは、CamelContext が設定され、自動的に開始されます。
アプリケーションのブートストラップおよび設定のさまざまな方法については、Configuring your Quarkus applications ガイドを参照してください。

既存の Quarkus エクステンションとの統合

Quarkus の Camel エクステンションは、ネイティブサポートおよび設定オプションを継承する Camel コンポーネントによって使用されるライブラリーおよびフレームワークのエクステンションを提供します。

2.2. サポートされるプラットフォーム、設定、データベース、およびエクステンション

- Quarkus バージョン 2.2 の Camel エクステンションにおけるサポート対象のプラットフォーム、設定、およびデータベースの詳細は、カスタマーポータルの Supported Configuration ページを参照してください（ログインが必要です）。

- Quarkus エクステンションの Red Hat Camel エクステンションおよび各エクステンションの Red Hat サポートレベルの一覧は、**Camel Extensions for Quarkus Reference**の Extensions Overview の章を参照してください（ログインが必要です）。

2.3. テクノロジープレビューのエクステンション

Red Hat は、Quarkus の Camel エクステンションの今回のリリースで提供されるテクノロジープレビューのコンポーネントに対するサポートを提供していません。**Camel Extensions for Quarkus Reference** の Extensions Overview の章でテクノロジープレビューに指定された項目については、「テクノロジプレビュー機能のサポート範囲」で定義されているように、サポート範囲が限定されています。

2.4. 既知の問題

CAMEL-17158 AWS2 SQS 遅延のあるキューにメッセージを送信するときに、遅延は考慮されません。

遅延のあるキューを作成する場合、**camel-aws2-sqs** コンポーネントをプロデューサーとして使用して送信されるメッセージは、キューに設定されている遅延を考慮しません。
この動作の理由は、キュー設定をオーバーライドするメッセージを送信するときに、Camel がデフォルトの遅延として '0秒' を設定するためです。

回避策として、Camel プロデューサーを使用する場合は同じ遅延を設定する必要があります。たとえば、5 秒 の遅延でキューを作成する場合は、**camel-aws2-sqs** プロデューサーを使用する際に 5 秒の遅延を設定する必要もあります。

ENTESB-17763 Missing productised transitive deps of**camel-quarkus-jira** extensions

camel-quarkus-jira エクステンションを使用するアプリケーションは、追加の Maven リポジトリー https://packages.atlassian.com/maven-external/ をMaven **settings.xml** ファイルやアプリケーションプロジェクトの **pom.xml** ファイルのいずれかに設定する必要があります。

ENTESB-18306 NSQ、HDFS、および Sparkで、製品化した**netty-transport-native-epoll:jar:linux-aarch_64**が不足する

この動作の理由は、ネイティブの epoll ライブラリーが **camel-quarkus-2.2.1-product** ビルドに含まれていないためです。ただし、これらのライブラリーは NSQ、HDFS、または Spark コンポーネントには必要ないため、回避策としては **netty-all** をアプリケーションから除外し、**quarkus-netty** を依存関係として組み込むことが推奨されます。
たとえば、以下のように **hdfs** コンポーネントのアプリケーションの **pom.xml** を更新できます。

```
<dependency>
  <groupId>org.apache.camel.quarkus</groupId>
  <artifactId>camel-quarkus-hdfs</artifactId>
  <exclusions>
    <exclusion>
      <groupId>io.netty</groupId>
      <artifactId>netty-all</artifactId>
    </exclusion>
  </exclusions>
</dependency>
<dependency>
  <groupId>io.quarkus</groupId>
  <artifactId>quarkus-netty</artifactId>
</dependency>
```

2.5. 重要事項

Camel のバージョン 3.11.1 から 3.11.5 へのアップグレード

Camel Extensions for Quarkus バージョン 2.2.1 の Camel のバージョンが 3.11.1 から 3.11.5にアップグレードされました。その間の各 Camel パッチリリースに関する詳細は、以下を参照してください。

- Apache Camel 3.11.2 リリースノート
- Apache Camel 3.11.3 リリースノート
- Apache Camel 3.11.4 リリースノート
- Apache Camel 3.11.5 リリースノート

CVE-2021-44228 log4j-core:Log4j 2.x でのリモートコード実行

Camel Extensions for Quarkus エクステンションは以下のアーティファクトに依存しないため、Camel Extensions for Quarkus バージョン 2.2.1ではこれらのアーティファクトは管理されなくなりました。

- org.apache.logging.log4j:log4j-1.2-api
- org.apache.logging.log4j:log4j-core

- org.apache.logging.log4j:log4j-jcl
- org.apache.logging.log4j:log4j-jul
- org.apache.logging.log4j:log4j-slf4j-impl
- org.apache.logging.log4j:log4j-web
  アプリケーションがこれらのいずれかのアーティファクトの依存関係を追加する場合は、Log4j 2.17.1 より前のバージョンに関連する既知の CVE を回避するために、Log4j 2.x の最新バージョンを使用するようにしてください。

注記

quarkus-bom は、引き続き **org.apache.logging.log4j:log4j-api** を管理します。

最低限必要な Apache Maven バージョンの 3.8.1 への変更

Camel Extensions for Quarkus バージョン 2.2.1 の本リリースでは、Red Hat ビルドの Quarkus プロジェクトのコンパイルに必要となる Apache Maven の最低バージョンが 3.8.1 に変更されました。Red Hat ビルドの Quarkus 2.2 をベースにプロジェクトをコンパイルできるようにするには、Apache Maven のインストールをバージョン 3.8.1 にアップグレードする必要があります。このアップグレードは、Apache Maven ビルドを中間者攻撃に対して脆弱にする可能性のあるセキュリティーの問題に対応するために必要です。このセキュリティー脆弱性についての詳細は、Red Hat カスタマーポータルの CVE-2021-26291 に関するエントリーを参照してください。

2.6. 解決した問題

ENTESB-18560 NativeモードのJSONPath の出力が JVM モードと異なる

JSONPath 言語で使用されるクラスのリフレクションの登録がありませんでした。以前のバージョンのCamel Extensions for Quarkusでは、Nativeモードの JSONPath 表現の出力は以下のとおりでした。

```
{name=Jan, age=28}
```

Camel Extensions for Quarkus バージョン 2.2.1では、この問題が解決され、Nativeモードの JSONPath 表現の出力は以下のようになりました。

```
{"name":"Jan","age":28}
```

ENTESB-18016 Quarkus Dev UI がRed Hatの製品ドキュメントではなくコミュニティーのドキュメントを参照する

mvn quarkus:dev を使用して Quarqus dev モードでプロジェクトを実行する場合、Quarkus はエンドポイント **/q/dev** 経由で **Dev UI** を提供します。このインターフェースには、関連するドキュメントページへのリンクが含まれるデプロイされたエクステンションが表示されます。以前のバージョンの Camel Extensions for Quarkusでは、これらのリンクはコミュニティーのエクステンションページを参照していました。本リリースでは、これらのリンクが更新され、Red Hat の製品ドキュメントページを参照するようになりました。

ENTESB-17855 geronimo-jms_*_spec* artifacts アーティファクトが jakarta.jms アーティファクトによって置き換えられる

本リリースでは、さまざまな JMS 関連のエクステンションで使用される **org.apache.geronimo.specs:geronimo-jms_1.1_spec** および **org.apache.geronimo.specs:geronimo-jms_2.0_spec** アーティファクトが、より新しいベンダーに依存しない等価な **jakarta.jms:jakarta.jms-api** に置き換えられています。

ENTESB-17939 jakarta.activationによる javax.activation の置き換え

本リリースでは、さまざまな Camel Extensions for Quarkusエクステンションで使用される **com.sun.activation:javax.activation** および **javax.activation:activation** アーティファクトが、より新しい等価な **com.sun.activation:jakarta.activation** に置き換えられました。

2.7. 非推奨になった CAMEL EXTENSIONS FOR QUARKUS の機能

Elasticsearch Rest エクステンション

Camel Extensions for Quarkusの **camel-quarkus-elasticsearch-rest** エクステンションは本リリースで非推奨となり、今後のリリースで削除される予定です。

2.8. 関連情報

- Supported Configurations
- Camel Extensions for Quarkus
- Getting Started with Camel Extensions for Quarkus
- Developing Applications with Camel Extensions for Quarkus

第3章 DEBEZIUM リリースノート

Debezium は、分散型変更データキャプチャープラットフォームで、データベーステーブルで発生する行レベルの変更をキャプチャーし、対応する変更イベントレコードを Apache Kafka トピックに渡します。アプリケーションはこれらの **変更イベントストリーム** を読み取りでき、変更イベントが発生した順にアクセスできます。Debezium は Apache Karaf に構築され、AMQ Streams とデプロイおよび統合されます。

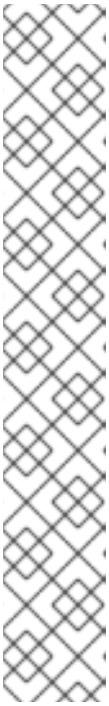
リリースの詳細は以下を参照してください。

- 「Debezium データベースコネクター」
- 「Debezium でサポートされる構成」
- 「Debezium のインストールオプション」
- 「Debezium の新機能」
- 「非推奨の Debezium 機能」

3.1. DEBEZIUM データベースコネクター

Debezium は、以下の共通データベースの Kafka Connect をベースとしたコネクターを提供します。

- Db2
- MongoDB
- MySQL
- Oracle (テクノロジープレビュー)
- PostgreSQL
- SQL Server



注記

- Db2 コネクターには、Linux 用の Db2 の標準部分として利用できる抽象構文表記 (ASN) ライブラリーを使用する必要があります。
  - ASN ライブラリーを使用するには、IBM InfoSphere Data Replication (IIDR) のライセンスが必要です。
  - ライブラリーを使用するために IIDR をインストールする必要はありません。
- 現在、MongoDB 4.2 で Debezium MongoDB コネクターのトランザクションメタデータ機能を使用することはできません。
- Debezium PostgreSQL コネクターでは、PostgreSQL バージョン 10 以降のデフォルトである **pgoutput** 論理デコーディング出力プラグインを使用する必要があります。
- Debezium Oracle コネクターを使用するには、Oracle から Oracle JDBC ドライバー (ojdbc8.jar) のコピーをダウンロードする 必要があります。

関連情報

- Getting Started with Debezium
- Debezium User Guide

3.2. DEBEZIUM でサポートされる構成

サポートされるデータベースバージョンなどの、Debezium でサポートされる構成の詳細は「Debezium 1.7 Supported Configurations」のページを参照してください。

AMQ Streams の新しい API バージョン

Debezium は AMQ Streams 2.0 で実行されます。

AMQ Streams 1.7 は、AMQ Streams カスタムリソースのスキーマを更新する **v1beta2** API バージョンをサポートするようになりました。古い API バージョンは非推奨になりました。AMQ Streams 1.7 にアップグレードした後、AMQ Streams 1.8 以降にアップグレードする前に、API バージョン **v1beta2** を使用するようにカスタムリソースをアップグレードする必要があります。

詳細は、『Debezium User Guide』を参照してください。

3.3. DEBEZIUM のインストールオプション

AMQ Streams で Debezium を OpenShift または RHEL にインストールできます。

- Debezium の OpenShift へのインストール
- Debezium の RHEL へのインストール

3.4. DEBEZIUM の新機能

Debezium 1.7 には、以下の更新が含まれています。

新しいデプロイメントメカニズム

Maven アーティファクトに基づく新しい AMQStreams ビルドメカニズムを使用することで、AMQStreams を使用して Debezium コネクターをデプロイできるようになりました。詳細については、**Debezium のドキュメント** を参照してください。

Debezium のドキュメント

- Debezium シグナリングテーブルを有効にして使用し、アドホックインクリメンタルスナップショットをトリガーする方法に関する情報:Debezium コネクターへのシグナル送信
- Debezium ユーザーガイドのデプロイメント手順を改訂しました。
  - Debezium Db2 コネクターのデプロイ
  - Debezium MongoDB コネクターのデプロイ
  - Debezium MySQL コネクターのデプロイ
  - Debezium Oracle コネクターのデプロイ
  - Debezium PostgreSQL コネクターのデプロイ

- Debezium SQL Server コネクターのデプロイ

テクノロジープレビューの機能

重要

テクノロジープレビュー機能は、Red Hat の実稼働サービスレベルアグリーメント (SLA) でサポートされておらず、機能的に完全でない可能性があります。Red Hat は、本番環境でのテクノロジープレビュー機能の実装は推奨しません。テクノロジープレビュー機能は、近々発表予定の製品イノベーションをリリースに先駆けてご提供することにより、お客様は機能性をテストし、開発プロセス中にフィードバックをお寄せいただくことができます。サポート範囲の詳細は、「テクノロジプレビュー機能のサポート範囲」を参照してください。

Debezium コネクターへのシグナル送信

統合シグナルメカニズムは、コネクターの動作を変更したり、コネクターをトリガーして、テーブルのアドホック増分スナップショットを開始するなどの 1 回限りのアクションを実行する方法を提供します。

CloudEvents コンバーター

CloudEvents 仕様に準拠する変更イベントレコードが出力されます。Avro エンコーディングタイプが CloudEvents エンベロープ構造でサポートされるようになりました。

送信トレイ (Outbox) イベントルーター

複数の (マイクロ) サービス間でデータを安全かつ確実に交換するための送信トレイ (Outbox) パターンをサポートする SMT。

Debezium Oracle コネクター

Oracle Database のコネクターこのリリースの Debezium Oracle コネクターは、次の機能を提供します。

- Oracle の DDL 変更の追跡をサポートします 。
- ロックせずにスナップショットを実行する機能。
- 変更イベントのバッファリングの重複排除チェックが改善されました。
- ストリーミング中の SCN ギャップ検出の改善
- マイニングは、特定のプラグ可能なデータベース (PDB) にスコープを指定することができます。
- ユーザー名で redo エントリーをスキップまたは除外する機能。
- 安定性の向上。
- 改善された DML ステートメントパーサー。
- 同じデータベースまたはプラグ可能なデータベース内の複数のスキーマから変更をキャプチャーする機能。
- 新しいパフォーマンス関連の JMX メトリクス。
- **time.precision.mode** プロパティーにより、時間値の精度を設定する機能。
- 複数のアーカイブプロセス (ARC) プロセスを実行する環境との互換性。

- 複数のアーカイブログの宛先でメッセージを処理する機能 (Oracle Data Guard と連携)。

3.5. 非推奨の DEBEZIUM 機能

コネクタースナップショットおよびストリーミングメトリックの MonitoredTables オプション

Debezium コネクターメトリクスの **MonitoredTables** オプションは、このリリースでは非推奨であり、将来のリリースで削除される予定です。代わりに **CapturedTables** メトリックを使用します。

第4章 CAMEL K リリースノート

Camel K は、OpenShift のクラウドでネイティブで実行される Apache Camel K からビルドされる軽量のインテグレーションフレームワークです。Camel K は、サーバーレスおよびマイクロサービスアーキテクチャー向けに特別に設計されています。Camel K を使用すると、Camel Domain Specific Language (DSL) で書かれたインテグレーションコードを直接 OpenShift で即座に実行することができます。

Camel K を OpenShift Serverless および Knative とともに使用すると、コンテナーは必要な場合のみ自動的に作成され、負荷時の自動スケーリングやゼロへのスケーリングが行われます。これにより、サーバーのプロビジョニングとメンテナンスのオーバーヘッドがなくなり、アプリケーションの開発に集中することができます。

Camel K を OpenShift Serverless および Knative Eventing とともに使用すると、システムのコンポーネントがサーバーレスアプリケーションのイベント駆動型アーキテクチャーで通信する方法を管理できます。これにより、イベントプロデューサーとコンシューマー間の関係が切り離されたパブリッシュ/サブスクライブモデルまたはイベントストリーミングモデルを使用すると、柔軟性を提供し、効率化を実現できます。

4.1. CAMEL K の新機能

Camel K は、以下の主要機能でクラウドネイティブインテグレーションを提供します。

- 自動スケーリングおよびゼロへのスケーリングを行うための Knative Serving
- イベント駆動型アーキテクチャーのための Knative Eventing
- デフォルトで Quarkus ランタイムを使用するパフォーマンスの最適化
- Java または YAML DSL で書かれた Camel インテグレーション
- OpenShift で Prometheus を使用したインテグレーションのモニタリング
- クイックスタートチュートリアル
- AWS、Jira、Salesforce などの外部システムへのコネクター用の Kamelet Catalog
- Timer および Log Kamelets のサポート
- Camel K オペレーターと Pod の計測

4.2. SUPPORTED CONFIGURATIONS

Camel K でサポートされる設定、標準仕様、およびコンポーネントに関する詳細は、以下のカスタマーポータルの記事を参照してください。

- Camel K Supported Configurations
- Camel K Component Details

4.2.1. Camel K Operator のメタデータ

Camel K には、OpenShift OperatorHub から Camel K をインストールするために使用される更新された Operator メタデータが含まれています。この Operator メタデータには、OpenShift Container Platform 4.6 以降との使用を目的に設計されたリリースパッケージの Operator バンドル形式が含まれます。

関連情報

- Operator バンドルフォーマットに関する OpenShift ドキュメント

4.3. 重要事項

Red Hat Integration - Camel K リリースにおける重要事項

CVE-2022-22965 spring-framework:JDK 9+ の Data Binding 経由の RCE

Camel K のパッチが適用されたバージョン（バージョン 1.6.5）が、spring-framwork セキュリティー問題 CVE-2022-22965 に対応するためにリリースされました。このパッチが適用されたバージョンを選択するために Camel K デプロイメントおよびアプリケーションプロジェクトを更新するには、第 4 章のアップグレード手順に従います。 **Getting Started with Camel K**ガイドから Camel K をアップグレードします。

CVE-2021-44228 log4j-core:Log4j 2.x でのリモートコード実行

Log4j 2.x のセキュリティー問題である CVE-2021-44228 (通称 Log4Shell) に対処するために、パッチが適用されたバージョンの Camel K (version 1.6.0-1) がリリースされました。このパッチが適用されたバージョンを選択するために Camel K デプロイメントおよびアプリケーションプロジェクトを更新するには、第 4 章のアップグレード手順に従います。 **Getting Started with Camel K**ガイドから Camel K をアップグレードします。パッチが適用されたバージョンの Camel K は **latest** Operator チャネルを介して配信されます。

Camel K でサポートされるエンタープライズ統合パターン (EIP)

以下を除くすべての Camel 3 EIP パターン は、Camel K で完全にサポートされます。

- サーキットブレーカー
- Saga
- Change Data Capture の変更

YAML DSL の制限

YAML DSL インテグレーションは Camel K 1.6.5 でサポートされていますが、誤った YAML DSL コードのエラーメッセージは引き続き開発中です。

JAVA DSL の制限

Camel K 1.6.5 の Java DSL は単一のクラス/設定メソッドに限定され、すべてのユーティリティーはサードパーティーの JARS で提供する必要があります。エンドポイント URI は、Camel K の自動依存関係サポートのエンドポイント文字列に直接定義する必要があります。定義しないと、依存関係をモード行に指定する必要があります。

XML DSL はサポートされません。

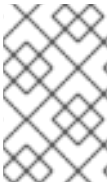
XML DSL は Camel K 1.6.5 ではサポートされません。

Camel K 1.6.5 runtime can access only access that support HTTPS（Camel K 1.6.5 ランタイムが HTTPS をサポートする Maven リポジトリーのみにアクセスできる）

HTTPS によってセキュア化された Maven リポジトリーのみを使用できます。安全ではない HTTP プロトコルに対応しなくなりました。

4.4. サポートされる CAMEL QUARKUS エクステンション

ここでは、本リリースの Camel K でサポートされる Camel Quarkus エクステンションを一覧表示します (Camel K アプリケーション内で使用する場合のみ)。



注記

これらの Camel Quarkus エクステンションは、Camel K アプリケーション内で使用される場合にのみサポートされます。これらの Camel Quarkus エクステンションは、スタンドアロンモード（Camel K なし）での使用はサポートされていません。

4.4.1. サポートされる Camel Quarkus コネクターエクステンション

以下の表は、本リリースの Camel K でサポートされる Camel Quarkus コネクターエクステンションを示しています（Camel K アプリケーション内で使用する場合のみ）。

名前	パッケージ
AWS 2 Kinesis	**camel-quarkus-aws2-kinesis**
AWS 2 Lambda	**camel-quarkus-aws2-lambda**
AWS 2 S3 Storage Service	**camel-quarkus-aws2-s3**
AWS 2 Simple Notification System (SNS)	**camel-quarkus-aws2-sns**
AWS 2 Simple Queue Service (SQS)	**camel-quarkus-aws2-sqs**
ファイル	**camel-quarkus-file**
FTP	**camel-quarkus-ftp**
FTPS	**camel-quarkus-ftp**
SFTP	**camel-quarkus-ftp**
HTTP	**camel-quarkus-http**
JMS	**camel-quarkus-jms**
Kafka	**camel-quarkus-kafka**
Kamelets	**camel-quarkus-kamelet**
メトリクス	**camel-quarkus-microprofile-metrics**
MongoDB	**camel-quarkus-mongodb**
Salesforce	**camel-quarkus-salesforce**
SQL	**camel-quarkus-sql**
Timer	**camel-quarkus-timer**

4.4.2. サポートされる Camel Quarkus データフォーマットエクステンション

以下の表は、本リリースの Camel K でサポートされる Camel Quarkus データフォーマットエクステンションを示しています（Camel K アプリケーション内で使用する場合のみ）。

名前	パッケージ
Avro	**camel-quarkus-avro**
Bindy (CSV 用)	**camel-qaurkus-bindy**
JSON Jackson	**camel-quarkus-jackson**
Jackson Avro	**camel-quarkus-jackson-avro**

4.4.3. サポートされる Camel Quarkus 言語エクステンション

本リリースでは、Camel K は以下の Camel Quarkus 言語エクステンションをサポートします（Camel 式および述語での使用）。

- Constant
- ExchangeProperty
- File
- Header
- Ref
- Simple
- Tokenize
- JsonPath

4.4.4. サポートされる Camel K トレイト

本リリースでは、Camel K は以下の Camel K トレイトをサポートします。

- Builder トレイト
- Camel トレイト
- Container トレイト
- Dependencies トレイト
- Deployer トレイト
- Deployment トレイト
- Environment トレイト

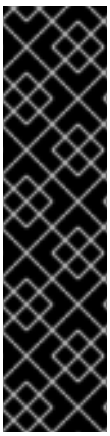
- JVM トレイト
- Kamelets トレイト
- Owner トレイト
- Platform トレイト
- Pull Secret トレイト
- Prometheus トレイト
- Quarkus トレイト
- Route トレイト
- Service トレイト
- Error Handler トレイト

4.5. サポート対象の KAMELETS

以下の表には、Camel K Operator をインストールする際に OpenShift リソースとして提供される kamelets が記載されています。

これらの kamelets の詳細は、https://github.com/openshift-integration/kamelet-catalog/tree/kamelet-catalog-1.6 を参照してください。

kamelets を使用してアプリケーションやサービスを接続する方法の詳細は https://access.redhat.com/documentation/ja-jp/red_hat_integration/2021.q4/html-single/integrating_applications_with_kamelets を参照してください。



重要

アスタリスク (*) のマークが付いた Kamelets はテクノロジープレビュー機能のみです。テクノロジープレビュー機能は、Red Hat の本番環境のサービスレベルアグリーメント (SLA) ではサポートされず、機能的に完全ではないことがあるため、Red Hat は本番環境での使用は推奨しません。Red Hat は実稼働環境でこれらを使用することを推奨していません。

これらの機能は、近々発表予定の製品機能をリリースに先駆けてご提供することにより、お客様は機能性をテストし、開発プロセス中にフィードバックをお寄せいただくことができます。Red Hat のテクノロジープレビュー機能のサポート範囲に関する詳細は、「テクノロジプレビュー機能のサポート範囲」を参照してください。

表4.1 Camel K Operatorで提供される Kamelets

Kamelet	ファイル名	タイプ (シンク、ソース、アクション)
Avro デシリアライズアクション	**avro-deserialize-action.kamelet.yaml**	アクション（データ変換）

Kamelet	ファイル名	タイプ (シンク、ソース、アクション)
Avro シリアライズアクション	**avro-serialize-action.kamelet.yaml**	アクション（データ変換）
AWS 2 S3 シンク	**aws-s3-sink.kamelet.yaml**	シンク
AWS 2 S3 ソース	**aws-s3-source.kamelet.yaml**	Source
AWS 2 S3 ストリーミングアップロードシンク	**aws-s3-streaming-upload-sink.kamelet.yaml**	シンク
AWS 2 Kinesis シンク	**aws-kinesis-sink.kamelet.yaml**	シンク
AWS 2 Kinesis ソース	**aws-kinesis-source.kamelet.yaml**	Source
AWS 2 Lambda シンク	**aws-lambda-sink.kamelet.yaml**	シンク
AWS 2 Simple Notification System シンク	**aws-sns-sink.kamelet.yaml**	シンク
AWS 2 Simple Queue Service シンク	**aws-sqs-sink.kamelet.yaml**	シンク
AWS 2 Simple Queue Service ソース	**aws-sqs-source.kamelet.yaml**	Source
AWS SQS FIFO シンク	**aws-sqs-fifo-sink.kamelet.yaml**	シンク
Cassandra sink*	**cassandra-sink.kamelet.yaml**	シンク
Cassandra source*	**cassandra-source.kamelet.yaml**	Source
Elasticsearch Index sink*	**elasticsearch-index-sink.kamelet.yaml**	シンク
フィールドアクションの抽出	**extract-field-action.kamelet.yaml**	アクション
FTP シンク	**ftp-sink.kamelet.yaml**	シンク
FTP ソース	**ftp-source.kamelet.yaml**	Source
ヘッダーフィルターアクションあり	**has-header-filter-action.kamelet.yaml**	Action（データ変換）

Kamelet	ファイル名	タイプ (シンク、ソース、アクション)
Hoist フィールドアクション	**hoist-field-action.kamelet.yaml**	アクション
HTTP シンク	**http-sink.kamelet.yaml**	シンク
フィールドアクションの挿入	**insert-field-action.kamelet.yaml**	Action（データ変換）
Header アクションの挿入	**insert-header-action.kamelet.yaml**	Action（データ変換）
Tombstone Filter アクション	**is-tombstone-filter-action.kamelet.yaml**	Action（データ変換）
Jira ソース*	**jira-source.kamelet.yaml**	Source
JMS シンク	**jms-amqp-10-sink.kamelet.yaml**	シンク
JMS ソース	**jms-amqp-10-source.kamelet.yaml**	Source
JSON デシリアライズアクション	**json-deserialize-action.kamelet.yaml**	アクション（データ変換）
JSON シリアライズアクション	**json-serialize-action.kamelet.yaml**	アクション（データ変換）
Kafka シンクコ	**kafka-sink.kamelet.yaml**	シンク
Kafka ソース	**kafka-source.kamelet.yaml**	Source
Kafka トピック名フィルターアクション	**topic-name-matches-filter-action.kamelet.yaml**	Action（データ変換）
ログシンク	**log-sink.kamelet.yaml**	シンク（開発およびテスト目的）
mask フィールドアクション	**mask-field-action.kamelet.yaml**	Action（データ変換）
メッセージタイムスタンプルーターアクション	**message-timestamp-router-action.kamelet.yaml**	アクション（ルーター）
MongoDB シンク	**mongodb-sink.kamelet.yaml**	シンク

Kamelet	ファイル名	タイプ (シンク、ソース、アクション)
MongoDB ソース	**mongodb-source.kamelet.yaml**	Source
MySQL シンクコネクター	**mysql-sink.kamelet.yaml**	シンク
PostgreSQL シンクコネクター	**postgresql-sink.kamelet.yaml**	シンク
述語フィルターアクション	**predicate-filter-action.kamelet.yaml**	アクション（ルーター/フィルター）
Protobuf デシリアライズアクション	**protobuf-deserialize-action.kamelet.yaml**	アクション（データ変換）
Protobuf シリアライズアクション	**protobuf-serialize-action.kamelet.yaml**	アクション（データ変換）
regex ルーターのアクション	**regex-router-action.kamelet.yaml**	アクション（ルーター）
フィールドアクションの置き換え	**replace-field-action.kamelet.yaml**	アクション
Salesforce ソース	**salesforce-source.kamelet.yaml**	Source
SFTP シンク	**sftp-sink.kamelet.yaml**	シンク
SFTP ソース	**sftp-source.kamelet.yaml**	Source
Slack ソース	**slack-source.kamelet.yaml**	Source
SQL サーバーデータベースシンク	**sqlserver-sink.kamelet.yaml**	シンク
Telegram ソース*	**telegram-source.kamelet.yaml**	Source
タイマーソース	**timer-source.kamelet.yaml**	ソース（開発およびテスト目的）
ルーターアクションのタイムスタンプ	**timestamp-router-action.kamelet.yaml**	アクション（ルーター）

Kamelet	ファイル名	タイプ (シンク、ソース、アクション)
キーアクションに対する値	**value-to-key-action.kamelet.yaml**	Action（データ変換）

4.6. CAMEL K の既知の問題

Camel K 1.6.5 には、以下の既知の問題が適用されます。

ENTESB-15306 - CRD conflicts between Camel K and Fuse Online

古いバージョンの Camel K が同じ OpenShift クラスターにインストールされたことがある場合、カスタムリソース定義との競合により、OperatorHub から Camel K のインストールに失敗します。たとえば、これには Fuse Online で以前使用できた古いバージョンの Camel K が含まれます。

この問題を回避するには、Camel K を別の OpenShift クラスターにインストールするか、Camel K をインストールする前に以下のコマンドを入力します。

```
$ oc get crds -l app=camel-k -o json | oc delete -f -
```

ENTESB-15858 - Added ability to package and run Camel integrations locally or as container images

ローカルでの Camel インテグレーションのパッケージ化および実行、またはコンテナーイメージとしての Camel インテグレーションのパッケージ化および実行は、現在 Camel K には含まれておらず、コミュニティーのみによるサポートの対象となります。

詳細は「Apache Camel K community」を参照してください。

ENTESB-16477 - Unable to download jira client dependency with productized build

Camel K Operator を使用する場合、インテグレーションは jira クライアントの依存関係を見つけることができません。回避策として、atlassian リポジトリーを手動で追加します。

```
apiVersion: camel.apache.org/v1
kind: IntegrationPlatform
metadata:
  labels:
    app: camel-k
  name: camel-k
spec:
  configuration:
  - type: repository
    value: <atlassian repo here>
```

ENTESB-17033 - Camel-K ElasticsearchComponent options ignored

Elasticsearch コンポーネントを設定する場合、Camel K ElasticsearchComponent オプションは無視されます。回避策として、Elasticsearch コンポーネントの使用時に **getContext().setAutowiredEnabled(false)** を追加します。

ENTESB-17061 - Can't run mongo-db-source kamelet route with non-admin user - Failed to start route mongodb-source-1 because of null

管理者でないユーザー認証情報で **mongo-db-source kamelet** ルートを実行することはできません。コンポーネントの一部には管理者の認証情報が必要なため、管理者以外のユーザーとしてルートを実行することはできません。

4.7. CAMEL K の修正された問題

以下のセクションには、Camel K 1.6.4 で修正された問題が記載されています。

- 「Camel K 1.6.4 で改良された機能」
- 「Camel K 1.6.4 で解決されたバグ」

4.7.1. Camel K 1.6.4 で改良された機能

以下の表に Camel K 1.6.4 で改良された機能を示します。

表4.2 Camel K 1.6.4 で改良された機能

問題	詳細
ENTESB-14406	Camel K とデプロイされた Pod に計測ラベルを提供する
ENTESB-18389	Kamelets カタログの BOM
ENTESB-18389	**.spec.template** と **.spec.flow** を使用する Kamelet カタログ

4.7.2. Camel K 1.6.4 で解決されたバグ

以下の表に Camel K 1.6.4 で解決されたバグを示します。

表4.3 Camel K 1.6.4 で解決されたバグ

問題	詳細
ENTESB-14343	Camel K のクラスター全体のプロキシー設定を考慮する
ENTESB-15385	CVE-2020-27218 jetty: Gzip リクエストのインフレーションでバッファーが正しくリサイクルされていません [rhint- camel -k-1]
ENTESB-15448	CVE-2020-8908 guava: 安全でない権限で作成された一時ディレクトリーを介したローカル情報の開示 [rhint- camel -k-1]
ENTESB-16081	CVE-2021-20293 resteasy-core:RESTEasy:RESTEasy では、反映される XSS 攻撃につながる可能性がある [rhint-camel-k-1]

問題	詳細
ENTESB-16165	CVE-2021-21349 xstream:SSRF は、XStream とのアンマーシャリングでアクティブ化して、イントラネットまたはローカルホストのリソースを参照する任意の URL からデータストリームにアクセスできます [rhint- camel -k-1]
ENTESB-16366	CVE-2021-28168 jersey-common: jersey:システムの一時ディレクトリーを介したローカル情報の公開 [rhint-camel-k-1]
ENTESB-16500	CVE-2021-26291 maven-core: maven:http をデフォルトで使用するブロックリポジトリー [rhint-camel-k-1]
ENTESB-16521	CVE-2020-15522 bouncycastle:EC math ライブラリー内のタイミングの問題 [rhint-camel-k-1]
ENTESB-16629	CVE-2021-28170 jakarta.el: jakarta-el:ELParserTokenManager enables invalid EL expressions to be evaluate [rhint-camel-k-1]
ENTESB-17118	CVE-2021-33813 jdom:XXE により、攻撃者は作成済みの HTTP リクエストを介して DoS が発生する [rhint-camel-k-1]
ENTESB-17119	CVE-2021-33813 jdom2: jdom:XXE により、攻撃者は作成済みの HTTP リクエストを介して DoS が発生する [rhint-camel-k-1]
ENTESB-17149	CVE-2021-3690 undertow: 受信 WebSocket PONG メッセージでのバッファーリークにより、DoS が発生する可能性がある [rhint- camel -k-1]
ENTESB-17407	[Camel K] HTTP プロキシー設定を指定できるようにする
ENTESB-17594	KafkaKamelet ソースのカスタムヘッダーをデシリアライズする
ENTESB-17851	カスタムブローカーに knative イベントを生成できない
ENTESB-17957	クイックスタートの Camel K のフロントエンド「FailedMount」：イベントストリームの例
ENTESB-17959	クイックスタートの Jitpack 依存関係は、1.6.x ブランチの main-SNAPSHOT を参照する
ENTESB-18003	Camel K の mrcc リポジトリーから欠陥のあるライブラリーを削除する
ENTESB-18296	Camel-K MRRC に存在する kamel バイナリー
ENTESB-18372	Missing camel-kamelets-utils-1.0.0.fuse-800050-redhat-00002.jar
ENTESB-18510	Camel K ランタイムテストがバージョン 1.9.0.fuse-800037-redhat-00001 で失敗する

問題	詳細
ENTESB-18511	一部の kamelets に camel-k-kamelet-reify 1.9.0.fuse-800037-redhat-00001 依存関係がない
ENTESB-18605	CVE respin: cyrus-sasl セキュリティー更新 RHSA:88692 重要な期限:2022 年 3 月 25 日
ENTESB-18621	Camel K operator を 1.6.3 から 1.6.4 へアップグレードできない
ENTESB-18624	Salesforce-source kamelet が CK3 で機能しない

第5章 SERVICE REGISTRY リリースノート

Red Hat Integration - Service Registry 2.0 は、Red Hat Integration 2022.Q1 の一般公開 (GA) コンポーネントとして使用できます。Service Registry は、Apicurio Registry オープンソースコミュニティープロジェクトをベースとする標準イベントスキーマおよび API デザインのデータストアです。

Service Registry を使用して、Web コンソール、REST API、Maven プラグイン、または Java クライアントを使用してデータの構造を管理および共有できます。たとえば、クライアントアプリケーションは、再デプロイせずに最新のスキーマ更新を Service Registry に動的にプッシュまたはプルできます。また、Service Registry を使用して任意のルールを作成し、レジストリーコンテンツが時間の経過と共にどのように進化するかを制御することもできます。たとえば、これには、コンテンツ検証のルールやスキーマまたは API バージョンの後方互換性と前方互換性に関するルールが含まれます。

5.1. SERVICE REGISTRY のインストールオプション

以下のデータストレージオプションのいずれかを使用して、Service Registry を OpenShift にインストールできます。

- AMQ Streams
- PostgreSQL データベース

詳細は、Installing and deploying Service Registry on OpenShift を参照してください。

5.2. SERVICE REGISTRY プラットフォームコンポーネントのバージョン

Service Registry 2.0.3 は以下のバージョンをサポートします。

- OpenShift Container Platform 4.9 または 4.6
- OpenJDK 11
- AMQ Streams 1.8
- PostgreSQL 12
- Debezium 1.4
- Camel Kafka Connector - テクノロジープレビュー

5.3. SERVICE REGISTRY の新機能

Service Registry のセキュリティー

- **Red Hat Single Sign-On に基づく認証** オプションでレジストリーを保護するため、REST API でユーザーを認証する必要があります (OAuth と HTTP Basic はサポートされます)。
- **ロールベースの承認**: 認証が有効になっている場合、ユーザーには少なくとも **sr-admin**、**sr-developer**、または **sr-readonly** ロールの 1 つが必要です。
- **作成者のみの承認** - 認証されたユーザーがアーティファクトを元々作成しない限り、アーティファクトへの変更が阻止されるオプション。
- **Kafka OAuth 認証**: AMQ Streams のストレージでは、OAuth 認証を必要とする Kafka クラスターへのアクセスを設定できます。

Service Registry コア

- **レジストリーアーティファクトグループ** - オプションでスキーマと API アーティファクトをカスタムの名前付きの論理グループへ整理
- **リファクタリングされた Kafka シリアライザー/デシリアライザー (SerDes) クラス**- 使いやすく、一貫性、機能に対処するための Java SerDes レイヤーへの大幅な更新
- **イベントソーシング** - CloudEvents 仕様に基づいて変更が行われるたびにイベントをトリガーするようにレジストリーを設定するオプション。

Service Registry データストレージ

- **SQL ベースのストレージ** - PostgreSQL データベースをサポートする新しい SQL ストレージ実装
- **Kafka ベースのストレージ** - AMQ Streams を使用してアーティファクトデータと組み込み SQL データベースを保存し、メモリーに表す組み込み SQL データベースを使用した新しいハイブリッドストレージ

Service Registry v2 REST API

- **カスタムバージョン管理** - REST API を使用してアーティファクトを作成または更新する際に、カスタムバージョン番号を提供するオプション
- **アーティファクト検索の改善** - REST API への更新により、アーティファクトの検索が改善
- **API のインポート/エクスポート** - レジストリーデータを **.zip** 形式でエクスポートおよびインポートする操作が含まれる REST API への更新
- **CNCF Schema Registry API のサポート** - Cloud Native Computing Foundation Schema Registry REST API の実装

注記

Service Registry v2 REST API は、新しいアーティファクトグループを含まない Confluent Schema Registry REST API と互換性があります。後方互換性は、既存の Service Registry v1 REST API で維持されます。

Service Registry Operator

- **パフォーマンスおよび合理化の向上** - Operator は OpenShift での **Deployment** (**DeploymentConfig** ではなく)、予測可能なリソース命名 (ランダムなサフィックスなし)、および並行して作成されるリソースを使用します。
- **レジストリーデータストレージ** - 新しい SQL および Kafka ベースのストレージオプションのサポート。
- **レジストリーセキュリティー** - Red Hat Single Sign-On を使用した認証および承認設定のサポート。
- ApicurioRegistry CRD v1 - **status** ブロックの標準化された **conditions** フィールドを使用して、Operator またはアプリケーションの問題またはエラーをより適切に示します。
- **マルチ namespace デプロイメント**: Operator が namespace にインストールされている場合、すべての namespace (または選択したサブセット) を監視することができ、アプリケーションはすべてまたは複数の namespace にデプロイすることができます。

- **非接続インストール** - ネットワークが制限された環境での OpenShift へのインストールのサポートが、バージョン 2.0.1 および 1.1.2 で追加されました。詳細は、「Mirroring images for a disconnected installation」を参照してください。

Service Registry ユーザーのドキュメントおよび例

- バージョン 2.0 の新機能で更新されたドキュメントライブラリー:
  - OpenShift での Service Registry のインストールおよびデプロイ
  - Service Registry デプロイメントの移行
  - Service Registry ユーザーガイド
  - Registry v2 core REST API ドキュメント
- 更新されたオープンソースデモアプリケーション:
  - https://github.com/Apicurio/apicurio-registry-examples

5.4. 非推奨および削除された SERVICE REGISTRY の機能

非推奨となった Service Registry の機能

- Service Registry バージョン 1.x はバージョン 2.0 で非推奨となり、近日中にフルサポートの対象外となる予定です。詳細は、「Red Hat ミドルウェア製品のアップデートとサポートポリシー」を参照してください。

Service Registry で削除された機能

- Infinispan のキャッシュベースのストレージオプションが削除されました。
- Java Persistence API (JPA) ストレージオプションが、新しい PostgreSQL データベースストレージオプションに置き換えられました。
- AMQ Streams の Kafka ベースのストレージオプションは、インメモリー H2 データベースを使用した AMQ Streams の新しいハイブリッドストレージ オプションに置き換えられました。
- Service Registry Java クライアントは OpenJDK 8 に対応しなくなり、代わりに OpenJDK 11 をサポートするようになりました。

5.5. SERVICE REGISTRY デプロイメントの移行

Service Registry バージョン 1.1 から 2.x への移行に関する詳細は、『Migrating Service Registry deployments』を参照してください。

Service Registry バージョン 2.x インスタンス間でのレジストリーデータの移行に関する詳細は、「Exporting and importing registry content using the Registry REST API 」を参照してください。

5.6. SERVICE REGISTRY で解決された問題

Service Registry core で解決された問題

IPT-651 - Service Registry からスキーマを取得する際の Protobuf メッセージのデシリアライズエラー

Kafka プロデューサーアプリケーションはスキーマを設定できますが、コンシューマーアプリケーションはレジストリーからのスキーマ取得に失敗し、以下のようなエラーメッセージと共に **org.apache.kafka.common.errors.SerializationException** を発生させます。

Error deserializing Protobuf message for id 3\nCaused by: java.io.IOException:Invalid schema syntax = \"proto3\";\npackage ...commons;\n\nimport \"head.proto\";\n\noption java_package = \"packaget\";\noption java_multiple_files = true;\n\nmessage AuditMessage {\n commons.Head head = 1;\n int64 id = 5;\n string user = 6;\n bytes extraData = 7;\n string signature = 8;\n}\n with refs [] of type AVRO\n\tat

IPT-625 - KafkaSQL ストレージオプションでの Service Registry へのアーティファクトのアップロードエラー

KafkaSQL ストレージオプションで Service Registry をインストールすると、新しいアーティファクトのアップロード時に **io.apicurio.registry.storage.RegistryStorageException** が発生します。考えられるエラーメッセージには、**SQL error:Expected one element, but found none**が含まれます。

このエラーは、データベースシーケンスを制御するメッセージを削除するKafka ログコンパクションにより発生しました。この問題は、データベースシーケンスを制御するメッセージが圧縮されないようにすることで修正されています。詳細は、Apicurio コミュニティーのブログポスト Resolving a bug in KafkaSQL storage for Apicurio Registryを参照してください。

IPT-159 - レジストリー v1 API および Confluent 互換性 API の不一致

Service Registry v2.x に移行する既存ユーザーは、Service Registry v1 シリアライザー/デシリアライザー(SerDes)を使用するすべての Kafka クライアントアプリケーションを、代わりにService Registry v2 SerDes を使用するようにアップグレードする必要がありました。

Service Registry は、**ENABLE_CCOMPAT_LEGACY_ID_MODE** という名前の新しい環境変数を提供し、これを使用してv1互換性APIの従来の動作に戻すことができます。この変数が **true** に設定されている場合、Service Registry は、互換性 API を使用してアップロードされたスキーマの一意の整数識別子として **contentId** の代わりに **globalId** を使用します。

Registry-1619 - Service Registry server cannot be properly configured to require authentication without role-based authorization

Service Registry サーバーでロールベースの承認が無効になっている場合、認証も事実上無効になります。Quarkus で OpenID Connect が有効になっている場合でも、ユーザーはクレデンシャルを提供する必要はありません。ユーザーが無効なクレデンシャルを提供すると、要求は失敗します。ただし、ユーザーがクレデンシャルを提供しない場合は、匿名ユーザーの代わりに要求は成功します。また、ロールが無効になっているため、追加のチェックは行われません。

Registry-1289 - Registry does not work on IPv6

Internet Protocol v6 を使用する Kubernetes サーバーに Service Registry をデプロイしようとすると、レジストリーサーバーが起動に失敗します。

Registry-1151 - Error fetching JavaScript libraries when running in a closed network

閉じられたネットワークで実行する場合、Redoc JavaScript ライブラリは、アプリケーションに組み込まれたりバンドルされたりするのではなく、CDN を参照するため、正しく読み込まれません。

Registry-1007 - Registry REST API returns 406 error

リクエストに **Accept: application/json** ヘッダーが含まれる場合、Registry REST API は **406** エラーを返します。

Registry-711 - Service Registry client does not work with Jersey HTTP client

Jersey プロバイダーと RESTEasy JAX-RS プロバイダーの両方がクラスパスにある場合、RESTEasy が優先され、RESTEasy がサポートしないように見受けられる **application/octet-stream** トランスポートのJersey クライアントのサポートに依存する他の HTTP クライアント機能が中断されます。

Service Registry Operator で解決された問題

Operator-41: CRD の例が空白。

提供された例の **ApicurioRegistry** カスタムリソース定義を空にしない必要があります。

5.7. SERVICE REGISTRY の既知の問題

Service Registry コアの既知の問題

Registry-2394 - コア v1 互換性のための Service Registry API エンドポイントが認証によって適切に保護されていない

従来の **MY-REGISTRY-URL/api/** エンドポイントは、ServiceRegistryv2.x の **MY-REGISTRY-URL/apis/registry/v1** に移動した v1 コアレジストリー API のエイリアスです。このレガシーエンドポイントは、認証が設定されている場合、現在保護されていません。この問題は、認証層が **application.properties** で設定された Quarkus ポリシーを使用しなくなった ApicurioRegistryv2.1.x で修正されています。

Service Registry v2.0.x でこの問題を回避するには、 **REGISTRY_DISABLE_APIS** 環境変数を **/apis/ibmcompat/ の値に設定して、コア v1 レガシーエンドポイントを無効にします。 、/api/。** 。

IPT-701 - CVE-2022-23221 H2:JNDI を介したリモートサーバーからのカスタムクラスの読み込み

Service Registry データが AMQ Streams に保存されている場合、H2 データベースコンソールにより、リモート攻撃者は JDBC URL を使用して任意コードを実行できます。Service Registry はデフォルトで脆弱ではなく、悪意のある設定変更が必要になります。

Service Registry operator の既知の問題

Operator-42 - Auto-generation of OpenShift route may use wrong base host value

複数の **routerCanonicalHostname** 値がある場合、Service Registry OpenShift ルートの自動生成で間違ったベースホスト値が使用される可能性があります。

Operator-32 - Operator should support SCRAM authorization without TLS, not only SCRAM+TLS

Service Registry Operator は、SCRAM+TLS だけでなく、Transport Layer Security (TLS) を使用せずに Salted Challenge Response Authentication Mechanism (SCRAM) をサポートする必要があります。

第6章 RED HAT INTEGRATION の OPERATOR

Red Hat Integration 2022.Q1 では、Red Hat Integration Operator 1.3 が導入されました。

Red Hat Integration は、OpenShift で Red Hat Integration コンポーネントのデプロイメントを自動化するために、operator を提供します。Red Hat Integration Operator を使用して、これらの Operator を管理できます。

そのため、各コンポーネント operator を個別に管理できます。ここでは operator を紹介し、operator を使用して Red Hat Integration コンポーネントをデプロイする方法の詳細へのリンクを記載します。

6.1. OPERATOR とは

Operator は、Kubernetes アプリケーションのパッケージ化、デプロイメント、および管理を行う方法です。operator は運用上の人間の知識を仕様し、これをコンシューマーと簡単に共有できるソフトウェアにエンコードして、一般的なタスクや複雑なタスクを自動化します。

OpenShift Container Platform 4.x では、**Operator Lifecycle Manager (OLM)** を使用すると、ユーザーはすべての Operator とクラスター全体で実行される関連サービスをインストールおよび更新することができ、それらの Operator と関連サービスのライフサイクルを管理できます。これは、Kubernetes のネイティブアプリケーション (Operator) を効果的かつ自動化されたスケーラブルな方法で管理するために設計されたオープンソースツールキットである Operator Framework の一部です。

OLM は OpenShift Container Platform 4.x でデフォルトで実行されます。これは、クラスター管理者がクラスターで実行している Operator をインストールおよびアップグレードし、アクセスを付与するのに役立ちます。OpenShift Container Platform Web コンソールは、クラスター管理者が Operator をインストールし、クラスターで利用可能な Operator のカタログを使用するために特定のプロジェクトにアクセスを付与するための管理画面を提供します。

OperatorHub は、OpenShift クラスター管理者が Operator を検出、インストール、およびアップグレードするために使用するグラフィカルインターフェースです。1回クリックするだけで、これらの Operator を OpenHub からプルし、クラスターにインストールすることができ、OLM で Operator を管理することで、開発、テスト、および実稼働環境のソフトウェアをエンジニアリングチームが独自に管理することが可能です。

関連情報

- Operator についての詳細は、OpenShift のドキュメント を参照してください。

6.2. RED HAT INTEGRATION コンポーネント OPERATOR

たとえば、3scale Operator や Camel K Operator などを使用して、各 Red Hat Integration コンポーネント Operator を個別にインストールおよびアップグレードできます。

6.2.1. 3scale Operators

- 3scale Operator
- 3scale APIcast Operator

6.2.2. AMQ Operator

- AMQ Broker Operator

- AMQ Interconnect Operator
- AMQ Streams Cluster Operator
- AMQ Online Operator

6.2.3. Camel K Operator

- Camel K Operator - テクノロジープレビュー

6.2.4. Fuse Operator

- Fuse on OpenShift - Samples Operator
- Fuse on OpenShift - Fuse Console Operator
- Fuse on OpenShift - API Designer Operator
- Fuse Online Operator

6.2.5. Service Registry Operator

- Service Registry Operator

6.3. RED HAT INTEGRATION OPERATOR (非推奨)

Red Hat Integration Operator 1.3 を使用して、複数の Red Hat Integration コンポーネント Operator をインストールおよびアップグレードできます。

- 3scale
- 3scale APIcast
- AMQ Broker
- AMQ Interconnect
- AMQ Streams
- API Designer
- Camel K
- Fuse Console
- Fuse Online
- Service Registry

注記

Red Hat Integration Operator は非推奨となり、今後削除予定です。OpenShift4.6 から 4.10 の OperatorHub で利用できるようになります。個々の Red Hat Integration コンポーネント Operator は引き続きサポートされ、個別にインストールできます。

6.3.1. サポートされるコンポーネント

Red Hat Integration Operator 1.3 を使用して Operator をインストールする前に、コンポーネントのリリースノートで更新を確認します。サポートされるバージョンのリリースノートには、追加のアップグレード要件が記載されています。

- オンプレミス型 Red Hat 3scale API Management 2.10 向けリリースノート
- Red Hat AMQ Broker 7.8 のリリースノート
- Red Hat AMQ Interconnect 1.10 のリリースノート
- Red Hat AMQ Streams 2.0 on OpenShift リリースノート
- Red Hat Fuse 7.10 リリースノート (Fuse および API Designer)
- Release Notes for Red Hat Integration 2021.Q3 (Red Hat Integration - Service Registry 2.0 release notes)
- Red Hat Integration 2021.Q4 リリースノート (Camel K リリースノート)

AMQ Streams の新しい API バージョン

Red Hat Integration Operator 1.3 は、AMQ Streams 2.0 の Operator をインストールします。

AMQ Streams バージョン 1.8 以降にアップグレードする前に、API バージョン **v1beta2** を使用するようにカスタムリソースをアップグレードする必要があります。

AMQ Streams 1.7 では、AMQ Streams カスタムリソースのスキーマを更新する **v1beta2** API バージョンが導入されました。古い API バージョンは非推奨になりました。AMQ Streams 1.7 にアップグレードした後、AMQ Streams 2.0 にアップグレードする前に、API バージョン **v1beta2** を使用するようにカスタムリソースをアップグレードする必要があります。

バージョン 1.7 より前の AMQ Streams バージョンからアップグレードする場合は、以下を行います。

1. AMQ Streams 1.7 へのアップグレード
2. カスタムリソースを v1beta2 に変換します。
3. AMQ Streams 2.0 へのアップグレード

詳細は、以下のドキュメントを参照してください。

- アップグレードの要件
- v1beta2 API バージョンの導入

警告

カスタムリソースおよび CRD がバージョン **v1beta2** に変換されていない場合、AMQ Streams Operator をバージョン 2.0 にアップグレードすると、クラスターに失敗します。アップグレードは **Pending** で停止します。この場合は、以下を実行します。

1. Red Hat ソリューション (Forever pending cluster operator upgrade) で説明されている手順を実行します。
2. Integration Operator をゼロにスケーリングしてから 1 に戻し、AMQ Streams 2.0 Operator のインストールをトリガーします。

Service Registry 2.0 の移行

Red Hat Integration Operator は Red Hat Integration - Service Registry 2.0 をインストールします。

Service Registry 2.0 は、手動でアンインストールする必要がある Service Registry 1.x インストールを置き換えません。

Service Registry バージョン 1.x から 2.0 への移行に関する詳細は、Service Registry 2.0 release notes を参照してください。

6.3.2. サポートライフサイクル

サポート対象の設定を維持するには、最新の Red Hat Integration Operator バージョンをデプロイする必要があります。Red Hat Integration Operator の各リリースバージョンは 3 カ月間のみサポートされます。

6.3.3. 修正された問題

Red Hat Integration Operator 1.3 で修正された問題はありません。

関連情報

- 複数の Red Hat Integration コンポーネント Operator の管理に関する詳細は、「Installing the Red Hat Integration Operator on OpenShift」を参照してください。