Red Hat Integration 2021.Q2

Release Notes for Red Hat Integration 2021.Q2

What's new in Red Hat Integration

Last Updated: 2021-04-30

What's new in Red Hat Integration
Abstract

Describes the Red Hat Integration platform and provides the latest details on what's new in this release.
Table of Contents

CHAPTER 1. RED HAT INTEGRATION ................................................................. 3

CHAPTER 2. NEW FEATURES IN THIS RELEASE ........................................... 4
  2.1. NEW INTEGRATION FEATURES .......................................................... 4

CHAPTER 3. CAMEL QUARKUS RELEASE NOTES ........................................ 5
  3.1. CAMEL QUARKUS FEATURES ......................................................... 5
    3.1.1. Platform and core component versions ....................................... 5
    3.1.2. Technology Preview features .................................................. 5
    3.1.3. Available Camel Quarkus extensions in Technology Preview .......... 6

CHAPTER 4. SERVICE REGISTRY RELEASE NOTES ..................................... 7
  4.1. SERVICE REGISTRY INSTALLATION OPTIONS ................................... 7
  4.2. SERVICE REGISTRY NEW FEATURES .............................................. 7
    Service Registry security ................................................................. 7
    Service Registry core ................................................................. 7
    Service Registry data storage ....................................................... 8
    Service Registry v2 REST API ....................................................... 8
    Service Registry Operator ............................................................ 8
    Service Registry platform component versions ................................. 8
    Service Registry user documentation and examples ......................... 9
  4.3. SERVICE REGISTRY REMOVED FEATURES ..................................... 9
  4.4. MIGRATING AND UPGRADEING SERVICE REGISTRY DATA .................. 9
  4.5. SERVICE REGISTRY RESOLVED ISSUES ....................................... 9
    Service Registry core resolved issues .......................................... 9
    Service Registry Operator resolved issues ................................... 10
  4.6. SERVICE REGISTRY KNOWN ISSUES .......................................... 10
CHAPTER 1. RED HAT INTEGRATION

Red Hat Integration is a comprehensive set of integration and event processing technologies for creating, extending, and deploying container-based integration services across hybrid and multicloud environments. Red Hat Integration provides an agile, distributed, and API-centric solution that organizations can use to connect and share data between applications and systems required in a digital world.

Red Hat Integration includes the following capabilities:

- API connectivity
- Data transformation
- Service composition and orchestration
- Real-time messaging
- Cross-datacenter message streaming
- API management

Additional resources

- Understanding enterprise integration
CHAPTER 2. NEW FEATURES IN THIS RELEASE

This section provides a summary of the key new features in Red Hat Integration 2021.Q2 and provides links to more details on new features available in different components.

NOTE

These release notes include details on components updated in Red Hat Integration 2021.Q2 only. For details on the latest versions of other components, such as Debezium and Camel K, see the Red Hat Integration Release Notes for 2021-Q1.

2.1. NEW INTEGRATION FEATURES

Camel Quarkus extensions

- Camel components available as Camel Quarkus extensions in Camel Quarkus Technology Preview.

Kafka schema registry

- Improved Apache Kafka schema and API registry with OpenShift Operator in Service Registry 2.0 Technology Preview.
CHAPTER 3. CAMEL QUARKUS RELEASE NOTES

CHAPTER 3. CAMEL QUARKUS RELEASE NOTES

Camel Quarkus is available as a Technology Preview component in Red Hat Integration 2021.Q2.

NOTE

In this Technology Preview, Camel Quarkus is supported in JVM mode only.

Camel Quarkus brings the integration capabilities of Apache Camel and its vast component library to the Quarkus runtime. By using Camel Quarkus, you can take advantage of the performance benefits, developer joy and the container first ethos which Quarkus provides.

Camel Quarkus provides Quarkus extensions which are units of the Quarkus distribution. Extensions configure, boot and integrate Camel components in your Quarkus application and are typically configured as dependencies in your project.

Camel Quarkus takes advantage of the many performance improvements made in Camel 3, which results in a lower memory footprint, less reliance on reflection and faster startup times.

IMPORTANT

Technology Preview features are not supported with Red Hat production service-level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend implementing any Technology Preview features in production environments.

This Technology Preview feature provides early access to upcoming product innovations, enabling you to test functionality and provide feedback during the development process. For more information about support scope, see Technology Preview Features Support Scope.

3.1. CAMEL QUARKUS FEATURES

The Camel Quarkus Technology Preview provides the following main features:

3.1.1. Platform and core component versions

- OpenShift Container Platform 4.6
- Quarkus 1.11 Java runtime
- Apache Camel 3.7
- Apache Camel Quarkus 1.6
- OpenJDK 11

3.1.2. Technology Preview features

Fast startup and low RSS memory

Using the optimized build-time and ahead-of-time (AOT) compilation features of Quarkus, your Camel application can be pre-configured at build time resulting in fast startup times.

Highly configurable
All of the important aspects of a Camel Quarkus application can be set up programatically with CDI (Contexts and Dependency Injection) or via configuration properties. By default, a CamelContext is configured and automatically started for you. Check out the Configuring your Quarkus applications guide for more information on the different ways to bootstrap and configure an application.

Integrates with existing Quarkus extensions

Quarkus provides extensions for libraries and frameworks that are used by some Camel components which inherit native support and configuration options.

3.1.3. Available Camel Quarkus extensions in Technology Preview

- camel-quarkus-main
- camel-quarkus-timer
- camel-quarkus-log
- camel-quarkus-file
- camel-quarkus-ftp
- camel-quarkus-direct
- camel-quarkus-bean
- camel-quarkus-mock
- camel-quarkus-bindy
- camel-quarkus-seda
- camel-quarkus-core
- camel-quarkus-microprofile-health

Additional resources

- Getting Started with Camel Quarkus
CHAPTER 4. SERVICE REGISTRY RELEASE NOTES

Red Hat Integration - Service Registry 2.0 is available as a Technology Preview component in Red Hat Integration 2021.Q2. Service Registry is a datastore for standard event schemas and API designs, which is based on the Apicurio Registry open source community project.

You can use Service Registry to manage and share the structure of your data using a web console, REST API, Maven plug-in, or Java client. For example, client applications can dynamically push or pull the latest schema updates to or from Service Registry without needing to redeploy. You can also use Service Registry to create optional rules to govern how registry content evolves over time. For example, this includes rules for content validation or backwards and forwards compatibility of schema or API versions.

**IMPORTANT**

Technology Preview features are not supported with Red Hat production service-level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend implementing any Technology Preview features in production environments.

This Technology Preview feature provides early access to upcoming product innovations, enabling you to test functionality and provide feedback during the development process. For more information about support scope, see Technology Preview Features Support Scope.

4.1. SERVICE REGISTRY INSTALLATION OPTIONS

You can install Service Registry on OpenShift with the following data storage options:

**Table 4.1. Service Registry storage options**

<table>
<thead>
<tr>
<th>Storage option</th>
<th>Release category</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMQ Streams 1.7</td>
<td>Technology Preview</td>
</tr>
<tr>
<td>PostgreSQL 12 database</td>
<td>Technology Preview</td>
</tr>
</tbody>
</table>

4.2. SERVICE REGISTRY NEW FEATURES

Service Registry 2.0 includes the following updates:

**Service Registry security**

- **Authentication based on Red Hat Single Sign-On** - optionally protect the registry so that its REST API requires users to authenticate (Basic and OAuth are supported)
- **Role-based authorization** - when authentication is enabled, users must have at least one role of **sr-admin**, **sr-developer**, or **sr-readonly**
- **Creator-only authorization** - option to prevent changes to artifacts unless the authenticated user originally created the artifact

**Service Registry core**
- **Registry artifact groups** - optionally organize schema and API artifacts into custom named logical groupings

- **Refactored Kafka serializer/deserializer (SerDe) classes** - significant updates to the Java SerDe layer to address ease of use, consistency, and functionality

- **Event sourcing** - option to configure the registry to fire events whenever a change is made, based on the CloudEvents specification

### Service Registry data storage

- **SQL-based storage** - new SQL storage implementation with support for PostgreSQL database

- **Kafka-based storage** - new hybrid storage using AMQ Streams to store artifact data and an embedded SQL database to represent it in memory

### Service Registry v2 REST API

- **Custom versioning** - option to provide a custom version number when using the REST API to create or update artifacts

- **Improved artifact searching** - updates to the REST API to allow improved searching of artifacts

- **Import/export API** - updates to the REST API with operations to export and import registry data in .zip format

- **CNCF Schema Registry API support** - implementation of the Cloud Native Computing Foundation Schema Registry REST API

**NOTE**

The Service Registry v2 REST API is compatible with the Confluent schema registry REST API, which does not include the new artifact groups. Backwards compatibility is also maintained with the existing Service Registry v1 REST API.

### Service Registry Operator

- **Improved performance and streamlining** - Operator uses Deployment on OpenShift (instead of DeploymentConfig), predictable resource naming (no random suffixes), and resources created in parallel.

- **Registry data storage** - support for the new SQL and Kafka-based storage options.


- **ApicurioRegistry CRD v1** - uses standardized conditions field in the status block to better indicate issues or errors in the Operator or the application.

### Service Registry platform component versions

- OpenShift Container Platform 4.6 or 4.7

- OpenJDK 11

- AMQ Streams 1.7
CHAPTER 4. SERVICE REGISTRY RELEASE NOTES

- PostgreSQL 12
-Debezium 1.4
- Camel Kafka Connector Technology Preview

Service Registry user documentation and examples
- New documentation library updated with new features in version 2.0:
  - Installing and deploying Service Registry on OpenShift
  - Service Registry User Guide
  - Registry v2 core REST API documentation
- New open source demonstration applications:
  - https://github.com/Apicurio/apicurio-registry-examples

4.3. SERVICE REGISTRY REMOVED FEATURES
- The cache-based Infinispan storage option has been removed
- The Java Persistence API (JPA) storage option has been replaced by the new PostgreSQL database storage option
- The Kafka-based storage option in AMQ Streams has been replaced by the new hybrid storage option in AMQ Streams with in-memory H2 database
- The Service Registry Java client no longer supports OpenJDK 8 and supports OpenJDK 11 instead.

4.4. MIGRATING AND UPGRADING SERVICE REGISTRY DATA
For details on migrating registry data between Service Registry version 2.x instances, see Exporting and importing registry content using the Registry REST API.

For details on open source tools for migrating and upgrading from Service Registry version 1.x to 2.x, see Apicurio Registry export utility for 1.x versions.

4.5. SERVICE REGISTRY RESOLVED ISSUES
The following issues were resolved in Service Registry 2.0:

Service Registry core resolved issues

Registry-1289 - Registry does not work on IPv6
When trying to deploy Service Registry using the Operator on a Kubernetes server with Internet Protocol v6, the registry server fails to start.

Registry-1151 - Error fetching JavaScript libraries when running in a closed network
When running in a closed network, the Redoc JavaScript libraries do not load properly because they reference a CDN rather than get included or bundled in the application.
**Registry-1007** - Registry REST API returns 406 error

The Registry REST API returns a 406 error when the `Accept: application/json` header is included in the request.

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

When the Jersey and RESTEasy JAX-RS providers are both in the classpath, RESTEasy takes precedence and breaks other HTTP client functionality relying on Jersey client support for the `application/octet-stream` transport, which RESTEasy does not seem to support.

Service Registry Operator resolved issues

**Operator-41** - Example CRD should not be empty

The provided example `ApicurioRegistry` custom resource definition should not be empty.

### 4.6. SERVICE REGISTRY KNOWN ISSUES

The following known issues apply to Service Registry 2.0:

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

The Service Registry Operator should support Salted Challenge Response Authentication Mechanism (SCRAM) authorization without Transport Layer Security (TLS), not only SCRAM+TLS.

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

The auto-generation of the Service Registry OpenShift route may use a wrong base host value if there are multiple `routerCanonicalHostname` values.