Abstract

This document contains release notes for Red Hat build of Quarkus 1.3.
Table of Contents

PREFACE ......................................................................................................................... 3

CHAPTER 1. RED HAT BUILD OF QUARKUS ................................................................. 4

CHAPTER 2. RED HAT BUILD OF QUARKUS SUPPORTED PLATFORMS, CONFIGURATIONS, EXTENSIONS, AND DEPENDENCIES ................................................................. 5
  2.1. TESTED AND VERIFIED ENVIRONMENTS ......................................................... 5
  2.2. SUPPORTED CONFIGURATIONS ......................................................................... 5
  2.3. SUPPORTED EXTENSIONS AND DEPENDENCIES ........................................... 5
  2.4. DEVELOPMENT SUPPORT ............................................................................... 7

CHAPTER 3. DEPRECATED COMPONENTS AND FEATURES ........................................... 8

CHAPTER 4. TECHNOLOGY PREVIEW ........................................................................ 9

CHAPTER 5. KNOWN ISSUES ....................................................................................... 10
These release notes list new features, features in technology preview, known issues, and issues fixed in Red Hat build of Quarkus 1.3.
CHAPTER 1. RED HAT BUILD OF QUARKUS

Red Hat build of Quarkus is a Kubernetes-native Java stack that is optimized for use with containers and Red Hat OpenShift Container Platform. Quarkus is designed to work with popular Java standards, frameworks, and libraries such as Eclipse MicroProfile, Apache Kafka, RESTEasy (JAX-RS), Hibernate ORM (JPA), Spring, Infinispan, and Apache Camel.

The Quarkus dependency injection solution is based on CDI (contexts and dependency injection) and includes an extension framework to expand functionality and to configure, boot, and integrate a framework into your application.

Quarkus provides a container-first approach to building Java applications. This approach makes it much easier to build microservices-based applications written in Java as well as enabling those applications to invoke functions running on serverless computing frameworks. For this reason, Quarkus applications have small memory footprints and fast start-up times.
CHAPTER 2. RED HAT BUILD OF QUARKUS SUPPORTED PLATFORMS, CONFIGURATIONS, EXTENSIONS, AND DEPENDENCIES

This section lists supported environments, configurations, extensions, and dependencies in Red Hat build of Quarkus.

2.1. TESTED AND VERIFIED ENVIRONMENTS

Red Hat build of Quarkus is supported on the following platforms:

- Red Hat Enterprise Linux 8
- Red Hat OpenShift Container Platform 4.3

2.2. SUPPORTED CONFIGURATIONS

For a list of supported configurations, see the Red Hat build of Quarkus Supported Configurations page (login required).

2.3. SUPPORTED EXTENSIONS AND DEPENDENCIES

Red Hat provides production support for the following Red Hat build of Quarkus extensions and dependencies:

- quarkus-agroal
- quarkus-config-yaml
- quarkus-core
- quarkus-hibernate-orm
- quarkus-hibernate-orm-panache
- quarkus-hibernate-validator
- quarkus-jackson
- quarkus-jaxb
- quarkus-jdbc-mariadb
- quarkus-jdbc-mssql
- quarkus-jdbc-mysql
- quarkus-jdbc-postgresql
- quarkus-jsonb
- quarkus-jsonp
- quarkus-kafka-client
- quarkus-logging-json
- quarkus-narayana-jta
- quarkus-oidc
- quarkus-quartz
- quarkus-reactive-pg-client
- quarkus-rest-client
- quarkus-resteasy
- quarkus-resteasy-jackson
- quarkus-resteasy-jaxb
- quarkus-resteasy-jsonb
- quarkus-scheduler
- quarkus-smallrye-context-propagation
- quarkus-smallrye-fault-tolerance
- quarkus-smallrye-health
- quarkus-smallrye-jwt
- quarkus-smallrye-metrics
- quarkus-smallrye-openapi
- quarkus-smallrye-opentracing
- quarkus-smallrye-reactive-messaging
- quarkus-smallrye-reactive-messaging-amqp
- quarkus-smallrye-reactive-messaging-kafka
- quarkus-smallrye-reactive-streams-operators
- quarkus-spring-boot-properties
- quarkus-spring-data-jpa
- quarkus-spring-di
- quarkus-spring-security
- quarkus-spring-web
- quarkus-undertow
- quarkus-undertow-websockets
• quarkus-vertx
• quarkus-vertx-web

2.4. DEVELOPMENT SUPPORT

Red Hat provides development support for the following Red Hat build of Quarkus features, plug-ins, extensions, and dependencies:

Features
• Live development mode
• Debugging

Plug-ins
• Quarkus Maven plug-in (quarkus-maven-plugin)
• Maven Surefire plug-in (maven-surefire-plugin)

Extensions and dependencies
• quarkus-jdbc-derby
• quarkus-jdbc-h2
• quarkus-openshift
• quarkus-junit5
• rest-assured
CHAPTER 3. DEPRECATED COMPONENTS AND FEATURES

The components and features listed in this section are deprecated with Red Hat build of Quarkus 1.3. They are included and supported in this release, however no enhancements will be made to these components and features and they might be removed in the future.

- The `quarkus-smallrye-opentracing` extension
- The use of ReactiveX APIs
- The `reactive-streams-operators` API
CHAPTER 4. TECHNOLOGY PREVIEW

This section lists features and extensions that are in Technology Preview in Red Hat build of Quarkus 1.3.

IMPORTANT

These features are for Technology Preview only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs), might not be functionally complete, and Red Hat does not recommend to use them for production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information on Red Hat Technology Preview features, see Technology Preview Features Scope.

Features

- Native compilation using GraalVM
- Remote development

Extensions and dependencies

- quarkus-mutiny
- quarkus-resteasy-mutiny
- quarkus-keycloak-authorization
CHAPTER 5. KNOWN ISSUES

This section lists known issues with Red Hat build of Quarkus 1.3.

- If you add the `smallrye-reactive-messaging` extension, a warning message about ReactiveX appears when you close Quarkus even if you are not using Reactive APIs in your code. For more information, see Red Hat article 4954651.

- The platform Maven Resolver does not work when a remote repository is set in the `pom.xml` file. For more information, see Quarkus GitHub issue 8381.

- The platform Maven Resolver does not use the correct platform group Id when it is set by a Maven property from a profile in the `pom.xml` file. For more information, see Quarkus GitHub issue 8382.

_Revised on 2020-04-16 13:24:50 UTC_