Red Hat build of Quarkus 3.8

将应用程序迁移到红帽构建的 Quarkus 3.8

Last Updated: 2024-05-20

法律通告

摘要

本指南论述了如何将应用程序从 Red Hat build of Quarkus 的早期版本迁移到当前版本。

目录

提供有关红帽构建的 QUARKUS 文档的反馈

要报告错误或改进文档，请登录到 Red Hat JIRA 帐户并提交问题。如果您没有 Red Hat Jira 帐户，则会提示您创建一个帐户。

流程

1. 单击以下链接 **以创建 ticket**。
2. 在 **Summary** 中输入问题的简短描述。
3. 在 **Description** 中提供问题或功能增强的详细描述。包括一个指向文档中问题的 URL。
4. 点 **Submit** 创建问题，并将问题路由到适当的文档团队。

使开源包含更多

红帽致力于替换我们的代码、文档和 Web 属性中有问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。由于此项工作十分艰巨，这些更改将在即将推出的几个发行版本中逐步实施。详情请查看 CTO Chris Wright 的信息。

第 1 章 将应用程序迁移到红帽构建的 QUARKUS 3.8

作为应用程序开发人员，您可以使用 Quarkus CLI 的 **update** 命令将 基于 Red Hat build of Quarkus 的早期版本的应用程序迁移到 3.8 版本。

重要

Quarkus CLI 仅用于 dev 模式。红帽不支持在生产环境中使用 Quarkus CLI。

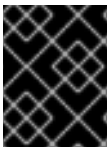
1.1. 将项目更新至最新的红帽构建的 QUARKUS 版本

您可以使用 update 命令更新或将红帽构建的 Quarkus 项目升级到最新版本。

update 命令主要使用 OpenRewrite recipes 自动更新大多数项目依赖项、源代码和文档。虽然这些方法执行许多迁移任务，但它们并不涵盖迁移指南中详述的所有任务。

更新后，如果缺少预期的更新，请考虑以下原因：

- **update** 命令应用的配方可能不包括项目所需的迁移任务。
- 您的项目可能会使用与最新的红帽构建的 Quarkus 版本不兼容的扩展。



重要

对于使用 Hibernate ORM 或 Hibernate Reactive 的项目，请查看 Hibernate ORM 5 到 6 的迁移 快速参考。以下 update 命令只涵盖本指南的子集。

1.1.1. 先决条件

- 大约 30 分钟
- IDE
- 正确配置了 **JAVA_HOME** 的 JDK 11+
- Apache Maven 3.8.6 或更高版本
- 另外，如果要使用它，红帽构建的 Quarkus CLI
- 如果要构建原生可执行文件（如果使用原生容器构建，则为 Docker）正确安装和配置 Mandrel 或 GraalVM
- 基于红帽构建的 Quarkus 版本 2.13 或更高版本的项目。

1.1.2. 流程

1. 使用您的版本控制系统为您的项目创建可正常工作的分支。
2. 要在下一步中使用 Red Hat build of Quarkus CLI，请安装 Red Hat build of Quarkus CLI 的最新版本。使用 **quarkus -v** 确认版本号。
3. 配置扩展 registry 客户端，如 Quarkus "Getting Started" 指南中的 Configuring Red Hat build of Quarkus extension registry client 部分所述。
4. 要使用红帽构建的 Quarkus CLI 更新，请进入项目目录并将项目更新至最新的流：

```
quarkus update
```

可选：默认情况下，这个命令会更新到最新的当前版本。要更新到特定流而不是最新的当前版本，请在这个命令中添加 **stream** 选项，后跟版本；例如 **：--stream=3.2**

5. 要使用 Maven 而不是红帽构建的 Quarkus CLI 更新，请进入项目目录并将项目更新至最新的流：
   a. 确保红帽构建的 Quarkus Maven 插件版本与最新支持的 Red Hat build of Quarkus 版本一致。
   b. 根据 Getting started with Quarkus 指南中提供的指南配置您的项目。

```
mvn com.redhat.quarkus.platform:quarkus-maven-plugin:3.8.4.redhat-00002:update
```

可选：默认情况下，这个命令会更新到最新的当前版本。要更新到特定流而不是最新的当前版本，请在这个命令中添加 **stream** 选项，后跟版本；例如： **-Dstream=3.2**

6. 分析 update 命令输出的潜在指令，并根据需要执行建议的任务。
7. 使用 diff 工具来检查所有更改。
8. 查看 update 命令没有更新的项目的迁移指南。如果您的项目有这样的项目，请实施这些主题中建议的额外步骤。
9. 在部署到生产之前，确保项目构建时没有错误，所有测试都通过应用功能。
10. 在将更新的 Red Hat build of Quarkus 应用程序部署到生产环境前，请确保以下内容：
    - 项目构建时无错误。
    - 所有测试都通过。
    - 应用程序可以正常工作。

1.2. 影响与早期版本兼容性的更改

这部分论述了红帽构建的 Quarkus 3.8 中的更改，它会影响与之前产品版本构建的应用程序的兼容性。

查看这些中断更改，并执行必要的步骤，以确保应用程序在将其更新至红帽构建的 Quarkus 3.8 后继续运行。

要自动执行许多这些更改，请使用 **quarkus update 命令将项目更新至最新的红帽构建的 Quarkus 版本**。

1.2.1. Core

1.2.1.1. Stork 负载平衡器配置的更改

您不再使用以前的配置名称 **stork."service-name".load-balancer** 和 **quarkus.stork."service-name".load-balancer** 来配置 Stork 负载均衡器。反之，使用 **quarkus.stork."service-name".load-balancer.type** 进行配置。

1.2.1.2. OkHttp 和 Okio 的依赖项管理更新

OkHttp 和 Okio 已从 Quarkus Platform BOM 中删除，它们的版本不再被强制执行，从而解决了与过时的依赖项相关的问题。这个更改会影响测试框架依赖项，并简化了运行时依赖项。使用这些依赖项的开发人员现在在构建文件中指定其版本。另外，**quarkus-test-infinispan-client** 工件已被删除，因为对 Infinispan 提供强大的 Dev Services 支持。

1.2.1.3. Java 版本要求更新

从此版本 Red Hat build of Quarkus 开始，删除了对 Java 11 的支持（在以前的版本中被弃用）。Java 21 现在是推荐的版本，但支持 Java 17。

1.2.1.4. RESTEasy Reactive 中的集合的 JAXB 限制

在红帽构建的 Quarkus 中，将 RESTEasy Reactive 与 XML Binding (JAXB)的 Java 架构搭配使用，不支持将集合、数组和映射用作 REST 方法中的参数或返回类型。为克服 JAXB 的这一限制，请在带有 **@XmlRootElement** 的类内封装这些类型。

1.2.1.5. 构建时 @StaticInitSafe 的强制规格

在静态初始化阶段，红帽构建的 Quarkus 会收集要注入 CDI Bean 的配置。然后，收集的值会与其运行时初始化对应的部分进行比较，如果检测到不匹配，应用程序启动会失败。使用红帽构建的 Quarkus 3.8，您现在可以使用 **@io.quarkus.runtime.annotations.StaticInitSafe** 注解来告知用户注入的配置：

- 在构建时设置
- 无法更改
- 在运行时安全地使用，指示红帽构建的 Quarkus 无法在配置不匹配时启动失败

1.2.1.6. Qute ：默认情况下标签模板的隔离执行

模板中的用户标签现在默认以隔离方式执行，从而限制对调用模板的上下文的访问。在这个版本中，可以在标签模板中更改数据处理，可能会影响其当前功能。要绕过此隔离并保持对父上下文的访问权限，请在标签调用中包含 **_isolated=false** 或 **_unisolated**，例如：NT **itemDetail item showImage=true _isolated=false**。这种方法允许标签从父上下文访问数据，如以前一样。此更改可最小化从父上下文到标签的意外数据暴露，从而增强模板数据完整性。但是，可能需要对现有模板的依赖共享上下文访问更新，代表一个显著的更改，可能会影响与这种隔离机制不熟悉的用户。

1.2.1.7. Qute ：解决类型轮询问题

ResultNode 类已更新为抽象类，而不是接口，尽管在公共 API 中不应该被用户实现。**Qute** API 现在将 **CompletionStage** 实现限制为 **java.util.concurrent.CompletableFuture** 和 **io.quarkus.qute.CompletedStage**，这是可通过 **-Dquarkus.qute.unrestricted-completion-stage-support=true** 的一个限制。

1.2.1.8. Quarkus-rest-client 扩展重命名为 quarkus-resteasy-client

在 Red Hat build of Quarkus 3.8 中，以下 **quarkus-rest-client** 扩展被重命名：

旧名称	新名称
quarkus-rest-client	**quarkus-resteasy-client**
quarkus-rest-client-mutiny	**quarkus-resteasy-client-mutiny**

旧名称	新名称
quarkus-rest-client-jackson	**quarkus-resteasy-client-jackson**
quarkus-rest-client-jaxb	**quarkus-resteasy-client-jaxb**
quarkus-rest-client-jsonb	**quarkus-resteasy-client-jsonb**

1.2.1.9. 在注入 @TestHTTPResource 时删除 URI 验证

@TestHTTPResource 注释现在支持路径参数。由于 URI 格式的不合规，不再应用作为 URI 字符串的验证。

1.2.1.10. 使用依赖项调整对 GraalVM SDK 23.1.2 的更新

GraalVM SDK 版本在 Red Hat build of Quarkus 3.8 中已更新至 23.1.2。使用需要 GraalVM 替换扩展的开发人员应该从 **org.graalvm.sdk:graal-sdk** 切换到 **org.graalvm.sdk:nativeimage** 以访问必要的类。对于使用 **org.graalvm.js:js** 的用户，请将这个依赖项替换为 **org.graalvm.polyglot:js-community** 用于社区版本。对于企业版本，请将此依赖项替换为 **org.graalvm.polyglot:js**。使用 **quarkus update** 自动调整 **graal-sdk**。但是，必须手动更改 **js** 依赖项。虽然这个变化不太可能，但这个更改可能会影响依赖的用户：

- **org.graalvm.sdk:collections**
- **org.graalvm.sdk:word**

1.2.1.11. 对 QuarkusComponentTest的各种调整

在本发行版本中，**QuarkusComponentTest** 有几个调整。它仍然是实验性的，不受 Red Hat build of Quarkus 支持。此实验状态表示 API 可能会随时更改，以反映收到的反馈。

QuarkusComponentTestExtension 现在不可变，需要通过简化的 constructor **QuarkusComponentTestExtension (Class...)** 或 **QuarkusComponentTestExtension.builder** （） 方法进行编程注册。测试实例生命周期（可以是 **Lifecycle#PER_METHOD** （默认）或 **Lifecycle#PER_CLASS** 规定，决定 CDI 容器何时启动和停止； **PER_METHOD** 会在每个测试后启动容器并停止它，而 **PER_CLASS** 在所有测试之前都会启动它，并在所有测试后停止它。这代表了之前版本的变化，容器总是在所有测试之前启动并停止。

1.2.2. data

1.2.2.1. Hibernate ORM 升级到 6.4

在红帽构建的 Quarkus 3.8 中，Hibernate Object-Relational Mapping (ORM)已升级到版本 6.4，并引进了以下破坏更改：

- 与一些旧的数据库版本兼容将被丢弃。有关支持的版本的更多信息，请参阅支持的结束。
- 现在，数字文字被解释为 Jakarta Persistence 3.2 中定义的。

如需更多信息，请参阅 Hibernate ORM 6.4 迁移指南。

1.2.2.2. 在启动时 Hibernate ORM 数据库版本验证

当在红帽构建的 Quarkus 3.8 上使用 Hibernate ORM 时，您可以在应用程序启动时验证指定的数据库版本。

要让 Hibernate ORM 生成效率更高的 SQL 并利用更多数据库功能，您可以在 **applications.properties** 文件中为 Hibernate 数据库设置特定的数据库版本。

例如： **quarkus.datasource.db-version = 14.0**

在这个 3.8 发行版本中，红帽构建的 Quarkus 会根据应用程序要连接到的实际数据库版本验证指定的数据库版本，并在出现不匹配时在启动时抛出异常。

如需更多信息，请参阅 Quarkus "使用 Hibernate ORM 和 Jakarta persistence" 指南中的 支持的数据库 部分。

1.2.2.3. Hibernate Search 升级到 7.0

在 Red Hat build of Quarkus 3.8 中，Hibernate Search 被升级到 7.0 版本，并引入了以下破坏更改：

- **quarkus.hibernate-search-orm.coordination.entity-mapping.outbox-event.uuid-type** 和 **quarkus.hibernate-search-orm.coordination.entity-mapping.agent.uuid-type** 配置属性已更改：
  - **uuid-binary** 已被弃用，而是 **二进制**
  - **uuid-char** 已弃用，而是使用 **char**
- **quarkus.hibernate-search-orm.elasticsearch.query.shard-failure.ignore** 属性的默认值从 **true** 改为 **false**，这意味着如果搜索操作中至少有一个分片失败，则 Hibernate Search 现在会抛出异常。要获得前面的行为，请将此配置属性设为 **true**。

注意

如果定义多个后端，您必须为每个 Elasticsearch 后端设置这个配置属性。

- 正则表达式 predicate 中的补充运算符(~)被移除，没有替换它的替代方法。
- Hibernate 搜索依赖项在其工件 ID 中不再有 **-orm6** 后缀；例如，应用程序现在依赖于 **hibernate-search-mapper-orm** 模块，而不是 **hibernate-search-mapper-orm-orm6**。

如需更多信息，请参阅以下资源：

- Hibernate 搜索文档
- Hibernate Search 7.0.0.Final: Migration guide from 6.2

1.2.2.4. SQL Server Dev Services 升级到 2022-latest

SQL Server 的 dev Services 将其默认镜像从 **mcr.microsoft.com/mssql/server:2019-latest** 更新至 **mcr.microsoft.com/mssql/server:2022-latest**。

用户更喜欢使用 Red Hat build of Quarkus "Configure data sources" 指南中的 References 部分中详述的配置属性来指定一个替代版本。

1.2.2.5. 升级到 Flyway 会为 Oracle 用户添加额外的依赖项

在 Red Hat build of Quarkus 3.8 中，Flyway 扩展被升级到 Flyway 9.20.0，它为 Oracle 用户提供额外的依赖项 **flyway-database-oracle**。

Oracle 用户必须更新 **pom.xml** 文件，使其包含 **flyway-database-oracle** 依赖项。要做到这一点，请执行以下操作：

```
<dependency>
  <groupId>org.flywaydb</groupId>
  <artifactId>flyway-database-oracle</artifactId>
</dependency>
```

如需更多信息，请参阅 使用 Flyway 的 Quarkus 指南。

1.2.3. 原生

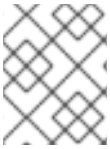
1.2.3.1. Kafka 扩展中的 strimzi OAuth 支持问题

quarkus-bom 中的 Kafka 扩展的 Strimzi OAuth 支持现在使用 **io.strimzi:strimzi-kafka-oauth** 版本 0.14.0，引入一个已知问题，导致原生构建失败。**'io.smallrye.reactive.kafka.graal.Target_com_jayway_jsonpath_internal_DefaultsImpl 的错误不会被加载**，方法是将 **io.strimzi:kafka-oauth-common** 添加到项目的 classpath 中。

1.2.4. Observability（可观察性）

1.2.4.1. 添加了 @AddingSpanAttributes 注释

当在 Quarkus 3.8 中使用 Opentelemetry (oTel)检测时，您现在可以使用 **io.opentelemetry.instrumentation.annotations.AddingSpanAttributes** 注解来在任何上下文依赖注入 (CDI)-aware bean 中添加注解的方法。



注意

如果您错误地给方法标上 **@AddingSpanAttributes** 和 **@WithSpan** 注释，则 **@WithSpan** 注释将具有优先权。

如需更多信息，请参阅 Quarkus "Using OpenTelemetry" 指南中的 CDI 部分。

1.2.4.2. quarkus-smallrye-metrics 扩展不再被支持

使用红帽构建的 Quarkus 3.8 时，**quarkus-smallrye-metrics** 扩展不再被支持。现在，它只作为社区扩展提供。不建议在生产环境中使用它。

在 Red Hat build of Quarkus 3.8 中，**quarkus-smallrye-metrics** 被完全支持的 **quarkus-micrometer** 扩展替代。

1.2.4.3. quarkus-smallrye-opentracing 扩展不再被支持

使用红帽构建的 Quarkus 3.8 时，不再支持 SmallRye OpenTracing。要继续使用分布式追踪，请将您的应用程序迁移到 SmallRye OpenTelemetry，现在在这个发行版本中被完全支持，不再是一个技术预览功能。如果您仍然需要使用 **quarkus-smallrye-opentracing**，请通过更新 **groupId** 并手动指定版本，调整应用程序以使用 Quarkiverse 中的扩展。

1.2.4.4. 重构调度程序和 OpenTelemetry Tracing 扩展

在 Red Hat build of Quarkus 3.8 中，已重构 OpenTelemetry Tracing 和 **quarkus-scheduler** 扩展的集成。

在此次更新之前，只有 **@Scheduled** 方法具有一个新的 **io.opentelemetry.api.trace.Span** 类，该类在启用追踪时自动关联。也就是说，当 **quarkus.scheduler.tracing.enabled** 配置属性设为 **true** 时，**quarkus-opentelemetry** 扩展可用。

在这个 3.8 发行版本中，所有调度的作业（包括以编程方式调度的作业）在启用追踪时自动关联 Span。每个调度方法的唯一作业标识符都是生成的，通过设置 **io.quarkus.scheduler.Scheduled#identity** 属性或使用 **JobDefinition** 方法来指定。在此次更新之前，范围名称后是 < **simpleclassname>.<methodName>** 格式。

如需更多信息，请参阅以下 Quarkus 资源：

- 调度程序参考
- 使用 OpenTelemetry

1.2.5. 安全性

1.2.5.1. 使用 mTLS 和 HTTP Restrictions 增强安全性

当将 mTLS 客户端身份验证(**quarkus.http.ssl.client-auth**)设置为 **required** 时，Red Hat build of Quarkus 会自动禁用普通的 HTTP 端口，以确保只接受安全 HTTPS 请求。要启用普通的 HTTP，请将 **quarkus.http.ssl.client-auth** 配置为 **请求** 或设置 **quarkus.http.ssl.client-auth=required** 和 **quarkus.http.insecure-requests=enabled**。

1.2.5.2. JWT 扩展会删除不必要的 Reactive 路由依赖项

JWT 扩展不再依赖于主动路由扩展。如果您的应用程序同时使用 JWT 和 Reactive Routes 功能，但没有声明对 Reactive Routes 的明确依赖项，您必须添加此依赖项。

1.2.5.3. Keycloak 授权丢弃了 keycloak-adapter-core 依赖项

因为更新到 Keycloak 22.0.0 及其对扩展功能的更新，**quarkus-keycloak-authorization** 扩展不再包含 **org.keycloak:keycloak-adapter-core** 依赖项。在以后的 Keycloak 版本中，计划删除 Keycloak Java 适配器代码。如果您的应用程序需要这个依赖项，请手动将它添加到项目的 **pom.xml** 中。

1.2.5.4. 使用 CDI 拦截器解析 RESTEasy Classic 中的 OIDC 租户不再被支持

您不再使用 Context 和 Dependency Injection (CDI)注解和拦截器来解析 RESTEasy Classic 应用的租户 OIDC 配置。

由于在 CDI 拦截器和需要身份验证的检查之前强制进行安全检查，使用 CDI 拦截器解析多个 OIDC 供应商配置标识符不再可以正常工作。

使用 **@Tenant** 注释或自定义 **io.quarkus.oidc.TenantResolver**。

如需更多信息，请参阅 Quarkus" 使用 OIDC 多租户指南"中的解决注解 部分。

1.2.5.5. 使用 OIDC @Tenant 注解将 OIDC 功能绑定到租户不再可能

在 Red Hat build of Quarkus 3.8 中，现在使用 **quarkus.oidc.TenantFeature** 注解而不是 **quarkus.oidc.Tenant** 将 OpenID Connect (OIDC)功能绑定到 OIDC 租户。

quarkus.oidc.Tenant 注解现在用于解析租户配置。

1.2.5.6. 安全配置集灵活性增强

Red Hat build of Quarkus 3.8 允许运行时配置 HTTP 权限和角色，跨配置集启用灵活的安全设置。这解决了原生可执行文件锁定到构建时安全配置的问题。现在，针对每个配置文件动态调整安全性，适用于 JVM 和原生模式。

1.2.6. Standards

1.2.6.1. 修正 GraphQL 指令应用程序

已修正了基于注解的 GraphQL 指令的应用程序，以确保它们只应用于声明它们的 schema 元素类型。

例如，如果声明了一个指令应用到 GraphQL 元素类型 FIELD，但错误地应用到不同的元素类型，它仍然在不应该适用的元素的 schema 中可见，从而导致一个无效的模式。现在，这个问题已被修正，指令会根据其适用性声明进行了检查。

如果您以这种方式应用了指令，它们将不再出现在架构中，红帽构建的 Quarkus 3.8 会在构建期间记录警告。

1.2.7. OpenAPI 标准化 POJO 和原语的内容类型默认值

当未提供 **@ContentType** 注释时，此更改已标准化了用于生成 OpenAPI 文档的默认内容类型。在以前的版本中，默认内容类型在不同的扩展间有所变化，如 RestEasy Reactive、RestEasy Classic、Spring Web 和 OpenAPI。例如，OpenAPI 始终使用 JSON 作为默认值，而 RestEasy 将 JSON 用于对象类型，以及用于 primitive 类型的文本。现在，所有扩展都已使用统一的默认设置，确保一致性：

- **原语类型** 现在统一设置为 **text/plain**。
- **复杂的 POJO (Plain Old Java Object)类型**默认为 **application/json**。

虽然跨扩展的行为是一致的，但它根据数据类型来适当地区分，并使用 **application** /plain 和 POJO 使用 **text/plain** 和 POJO。这种方法并不意味着相同的内容类型适用于所有 Java 类型，而所有扩展现在都以同样的方式处理内容类型，并针对数据的性质量身定制。

1.2.8. Web

1.2.8.1. 改进了 REST 客户端中的 SSE 处理

红帽构建的 Quarkus 3.8 增强了其 REST 客户端的 Server-Sent Events (SSE)功能，从而实现了完整的事件返回和过滤。REST 客户端中的这些更新和新描述为开发人员提供了管理实时数据流的控制和灵活性。

1.2.8.2. 手动添加 Reactive Routes 依赖项

在版本 3.8 之前，红帽构建的 Quarkus SmallRye JWT 会自动纳入 **quarkus-reactive-routes**，此功能从版本 3.8 开始停用。为确保继续功能，请手动添加 **quarkus-reactive-routes** 作为构建配置中的依赖项。

1.3. 其他资源

- Red Hat build of Quarkus 版本 3.2 发行注记