Getting Started with Camel Quarkus Extensions
Abstract

This guide introduces Camel Extensions for Quarkus, explains how to install Camel Quarkus Extensions on OpenShift, and how to get started with a sample Camel Quarkus Extensions deployment.
# Table of Contents

CHAPTER 1. TECHNOLOGY PREVIEW - GETTING STARTED WITH CAMEL QUARKUS EXTENSIONS  ..... 3  
1.1. CAMEL QUARKUS OVERVIEW 3  
1.2. DEPLOYING QUARKUS APPLICATIONS ON OPENS SHIFT 3  
1.3. GETTING STARTED WITH THE FILE CONSUMER QUICKSTART EXAMPLE 4
CHAPTER 1. TECHNOLOGY PREVIEW - GETTING STARTED WITH CAMEL QUARKUS EXTENSIONS

IMPORTANT

Camel Quarkus is a Technology Preview feature only. 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 using them in 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 about the support scope of Red Hat Technology Preview features, see https://access.redhat.com/support/offerings/techpreview.

This guide introduces Red Hat Integration - Camel Quarkus Extensions, how to get started building a quickstart Camel Quarkus application and how to deploy the application on OpenShift.

NOTE

Red Hat provides Maven repositories that host the content we ship with our products. These repositories are available to download from the software downloads page.

For Camel Quarkus the following repositories are required:

- rhi-common
- rhi-camel-quarkus

Installation of Camel Quarkus in offline mode is not supported in this release.

1.1. CAMEL QUARKUS OVERVIEW

The Camel Quarkus project brings the integration capabilities of Apache Camel and its vast component library to the Quarkus runtime.

The benefits of using Camel Quarkus include the following:

- Enables users to take advantage of the performance benefits, developer joy and the container first ethos which Quarkus provides.
- Provides Quarkus extensions for many of the Apache Camel components.
- 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.

You can define Camel routes using the Java DSL.

1.2. DEPLOYING QUARKUS APPLICATIONS ON OPENSHIFT

To learn how to deploy your Camel Quarkus applications on OpenShift, please refer to the Deploying your Quarkus applications on Red Hat OpenShift Container Platform guide.
1.3. GETTING STARTED WITH THE FILE CONSUMER QUICKSTART EXAMPLE

You can download or clone the quickstarts from the Camel Quarkus Examples Git repository. The example is in the file-bindy-ftp directory.

Extract the contents of the zip file or clone the repository to a local folder, for example a new folder named quickstarts.

You can run this Camel Quarkus example in development mode on your local machine from the command line. Using development mode, you can iterate quickly on integrations in development and get fast feedback on your code. Please refer to the Development mode section of the Camel Quarkus User guide for more details.

**NOTE**

If you need to configure container resource limits or enable the Quarkus Kubernetes client to trust self signed certificates, you can find these configuration options in the src/main/resources/application.properties file.

**Prerequisites**

- You have cluster admin access to the OpenShift cluster.
- You have access to an SFTP server and you have set the server properties (which are prefixed by ftp) in the application properties configuration file: src/main/resources/application.properties.

**Procedure**

1. Use Maven to build the example application in development mode:

   ```sh
   cd quickstarts/file-bindy-ftp
   mvn clean compile quarkus:dev
   ```

   The application triggers the timer component every 10 seconds, generates some random "books" data and creates a CSV file in a temporary directory with 100 entries. The following message is displayed in the console:

   ```
   [route1] (Camel (camel-1) thread #3 - timer://generateBooks) Generating randomized books CSV data
   ```

   Next, the CSV file is read by a file consumer and Bindy is used to marshal the individual data rows into Book objects:

   ```
   [route2] (Camel (camel-1) thread #1 - file:///tmp/books) Reading books CSV data from 89A0EE24CB03A69-0000000000000000
   ```

   Next the collection of Book objects is split into individual items and is aggregated based on the genre property:

   ```
   [route3] (Camel (camel-1) thread #0 - AggregateTimeoutChecker) Processed 34 books for genre 'Action'
   ```
   ```
   [route3] (Camel (camel-1) thread #0 - AggregateTimeoutChecker) Processed 31 books for
Finally, the aggregated book collections are unmarshalled back to CSV format and uploaded to the test FTP server.

2. To run the application in JVM mode, enter the following commands:

   $ mvn clean package -DskipTests
   $ java -jar target/*-runner.jar

3. You can build and deploy the example application to OpenShift, by entering the following command:

   $ mvn clean package -DskipTests -Dquarkus.kubernetes.deploy=true

4. Check that the pods are running:

   $ oc get pods

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>camel-quarkus-examples-file-bindy-ftp-1-d72mb</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>5m15s</td>
</tr>
<tr>
<td>ssh-server-deployment-5f6f685658-jtr9n</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>5m28s</td>
</tr>
</tbody>
</table>

5. Optional: Enter the following command to monitor the application log:

   $ oc logs -f camel-quarkus-examples-file-bindy-ftp-5d48f4d85c-sjl8k

Additional resources

- Developing applications with Camel Quarkus
- Camel Quarkus User guide