



Red Hat Fuse 7.13

Managing Fuse on OpenShift

Manage Fuse applications with the Fuse Console

Red Hat Fuse 7.13 Managing Fuse on OpenShift

Manage Fuse applications with the Fuse Console

Legal Notice

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

When you deploy a Fuse application, you can use the Fuse Console to monitor and interact with Red Hat Fuse integrations.

Table of Contents

PREFACE	3
MAKING OPEN SOURCE MORE INCLUSIVE	4
CHAPTER 1. ABOUT THE FUSE CONSOLE	5
CHAPTER 2. SETTING UP THE FUSE CONSOLE ON OPENSIFT 4.X	6
2.1. INSTALLING AND DEPLOYING THE FUSE CONSOLE ON OPENSIFT 4.X BY USING THE OPERATORHUB	6
2.2. INSTALLING AND DEPLOYING THE FUSE CONSOLE ON OPENSIFT 4.X BY USING THE COMMAND LINE	8
2.2.1. Generating a certificate to secure the Fuse Console on OpenShift 4.x	10
2.3. ROLE-BASED ACCESS CONTROL FOR THE FUSE CONSOLE ON OPENSIFT 4.X	12
2.3.1. Determining access roles for the Fuse Console on OpenShift 4.x	13
2.3.2. Customizing role-based access to the Fuse Console on OpenShift 4.x	13
2.3.3. Disabling role-based access control for the Fuse Console on OpenShift 4.x	14
2.4. UPGRADING THE FUSE CONSOLE ON OPENSIFT 4.X	15
2.5. UPGRADING FUSE IMAGESTREAMS AND TEMPLATES ON THE OPENSIFT 4.X SERVER	16
2.6. TUNING THE PERFORMANCE OF THE FUSE CONSOLE ON OPENSIFT 4.X	18
2.6.1. Performance tuning for Fuse Console Operator installation	19
2.6.2. Performance tuning for Fuse Console template installation	20
2.6.3. Performance tuning for viewing applications on Fuse Console	22
CHAPTER 3. SETTING UP THE FUSE CONSOLE ON OPENSIFT 3.11	24
3.1. DEPLOYING THE FUSE CONSOLE ON OPENSIFT 3.11	24
3.2. MONITORING A SINGLE FUSE POD FROM THE FUSE CONSOLE ON OPENSIFT 3.11	26
CHAPTER 4. VIEWING CONTAINERS AND APPLICATIONS	28
CHAPTER 5. VIEWING AND MANAGING APACHE CAMEL APPLICATIONS	29
5.1. STARTING, SUSPENDING, OR DELETING A CONTEXT	29
5.2. VIEWING CAMEL APPLICATION DETAILS	29
5.3. VIEWING A LIST OF THE CAMEL ROUTES AND INTERACTING WITH THEM	30
5.4. DEBUGGING A ROUTE	31
CHAPTER 6. VIEWING AMQ BROKERS	33
CHAPTER 7. VIEWING AND MANAGING JMX DOMAINS AND MBEANS	34
CHAPTER 8. VIEWING AND MANAGING QUARTZ SCHEDULES	35
CHAPTER 9. VIEWING DIAGNOSTICS	36
CHAPTER 10. VIEWING THREADS	37
CHAPTER 11. ENSURING THAT DATA DISPLAYS CORRECTLY IN THE FUSE CONSOLE	38
APPENDIX A. FUSE CONSOLE CONFIGURATION PROPERTIES	39

PREFACE

Red Hat Fuse provides two enterprise monitoring tools for viewing and managing Fuse integrations:

- The Fuse Console is a web-based console that you access from a browser to monitor and manage a running Fuse container. The Fuse Console is based on Hawtio open source software (<https://hawt.io/>). This guide describes how to use the Fuse Console.
- Prometheus stores system and integration-level metrics for Fuse distributions. You can use a graphical analytics interface, such as Grafana, to view and analyze the stored historical data. To learn more about using Prometheus, see the following documentation:
 - [the Prometheus documentation](#)
 - [Fuse on OpenShift Guide](#)
 - [Installing and Operating Fuse Online on OpenShift Container Platform](#)

The audience for this guide is Red Hat Fuse on JBoss EAP administrators. This guide assumes that you are familiar with the Red Hat Fuse platform, Apache Camel, and the processing requirements for your organization.

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our [CTO Chris Wright's message](#).

CHAPTER 1. ABOUT THE FUSE CONSOLE

The Red Hat Fuse Console is a web console based on HawtIO open source software. For a list of supported browsers, go to [Supported Configurations](#).

The Fuse Console provides a central interface to examine and manage the details of one or more deployed Fuse containers. You can also monitor Red Hat Fuse and system resources, perform updates, and start or stop services.

The Fuse Console is available when you install Red Hat Fuse standalone or use Fuse on OpenShift. The integrations that you can view and manage in the Fuse Console depend on the plugins that are running. Possible plugins include:

- Camel
- JMX
- OSGI
- Runtime
- Logs

CHAPTER 2. SETTING UP THE FUSE CONSOLE ON OPENSIFT 4.X

On OpenShift 4.x, setting up the Fuse Console involves installing and deploying it. You have these options for installing and deploying the Fuse Console:

- [Section 2.1, “Installing and deploying the Fuse Console on OpenShift 4.x by using the OperatorHub”](#)
You can use the Fuse Console Operator to install and deploy the Fuse Console so that it has access to Fuse applications in a specific namespace. The Operator handles securing the Fuse Console for you.
- [Section 2.2, “Installing and deploying the Fuse Console on OpenShift 4.x by using the command line”](#)
You can use the command line and one of the Fuse Console templates to install and deploy the Fuse Console so that it has access to Fuse applications in multiple namespaces on the OpenShift cluster or in a specific namespace. You must secure the Fuse Console by generating a client certificate before you deploy it.

Optionally, you can customize role-based access control (RBAC) for the Fuse Console as described in [Section 2.3, “Role-based access control for the Fuse Console on OpenShift 4.x”](#).

2.1. INSTALLING AND DEPLOYING THE FUSE CONSOLE ON OPENSIFT 4.X BY USING THE OPERATORHUB

To install the Fuse Console on OpenShift 4.x, you can use the Fuse Console Operator provided in the OpenShift OperatorHub. To deploy the Fuse Console, you create an instance of the installed operator.

Prerequisites

- You have configured authentication with **registry.redhat.io** as described in [Authenticating with registry.redhat.io for container images](#).
- If you want to customize role-based access control (RBAC) for the Fuse Console, you must have a RBAC configuration map file in the same OpenShift namespace to which you install the Fuse Console Operator. If you want to use the default RBAC behavior, as described in [Role-based access control for the Fuse Console on OpenShift 4.x](#), you do not need to provide a configuration map file.

Procedure

To install and deploy the Fuse Console:


1. Log in to the OpenShift console in your web browser as a user with **cluster admin** access.
2. Click **Operators** and then click **OperatorHub**.
3. In the search field window, type **Fuse Console** to filter the list of operators.
4. Click **Fuse Console Operator**.
5. In the Fuse Console Operator install window, click **Install**.
The **Create Operator Subscription** form opens.
 - For **Update Channel**, select **7.13.x**.

- For **Installation Mode**, accept the default (a specific namespace on the cluster).
Note that after you install the operator, when you deploy the Fuse Console, you can choose to monitor applications in all namespaces on the cluster or to monitor applications only in the namespace in which the Fuse Console operator is installed.
 - For **Installed Namespace**, select the namespace in which you want to install the Fuse Console Operator.
 - For the **Update Approval**, you can select **Automatic** or **Manual** to configure how OpenShift handles updates to the Fuse Console Operator.
 - If you select **Automatic** updates, when a new version of the Fuse Console Operator is available, the OpenShift Operator Lifecycle Manager (OLM) automatically upgrades the running instance of the Fuse Console without human intervention.
 - If you select **Manual** updates, when a newer version of an Operator is available, the OLM creates an update request. As a cluster administrator, you must then manually approve that update request to have the Fuse Console Operator updated to the new version.
6. Click **Install**.
OpenShift installs the Fuse Console Operator in the current namespace.
 7. To verify the installation, click **Operators** and then click **Installed Operators**. You can see the Fuse Console in the list of operators.
 8. To deploy the Fuse Console by using the OpenShift web console:
 - a. In the list of **Installed Operators**, under the **Name** column, click **Fuse Console**.
 - b. On the **Operator Details** page under **Provided APIs**, click **Create Instance**.
Accept the configuration default values or optionally edit them.

For **Replicas**, if you want to increase the Fuse Console performance (for example, in a high availability environment), you can increase the number of pods allocated to the Fuse Console.

For **Rbac** (role-based access control), only specify a value in the **config Map** field if you want to customize the default RBAC behavior and if the ConfigMap file already exists in the namespace in which you installed the Fuse Console Operator. For more information about RBAC, see [Role-based access control for the Fuse Console on OpenShift 4.x](#) .

For **Nginx**, see [Performance tuning for Fuse Console Operator installation](#).
 - c. Click **Create**.
The **Fuse Console Operator Details** page opens and shows the status of the deployment.
 9. To open the Fuse Console:
 - a. For a **namespace** deployment: In the OpenShift web console, open the project in which you installed the Fuse Console operator, and then select **Overview**. In the **Project Overview** page, scroll down to the **Launcher** section and click the Fuse Console link.

For a **cluster** deployment, in the OpenShift web console's title bar, click the grid icon (). In the popup menu, under **Red Hat applications**, click the Fuse Console URL link.
 - b. Log into the Fuse Console.
An **Authorize Access** page opens in the browser listing the required permissions.

- c. Click **Allow selected permissions**.
The Fuse Console opens in the browser and shows the Fuse application pods that you have authorization to access.
10. Click **Connect** for the application that you want to view.
A new browser window opens showing the application in the Fuse Console.

2.2. INSTALLING AND DEPLOYING THE FUSE CONSOLE ON OPENSIFT 4.X BY USING THE COMMAND LINE

On OpenShift 4.x, you can choose one of these deployment options to install and deploy the Fuse Console from the command line:

- **cluster** - The Fuse Console can discover and connect to Fuse applications deployed across multiple namespaces (projects) on the OpenShift cluster. To deploy this template, you must have the administrator role for the OpenShift cluster.
- **cluster with role-based access control**- The cluster template with configurable role-based access control (RBAC). For more information, see [Role-based access control for the Fuse Console on OpenShift 4.x](#).
- **namespace** - The Fuse Console has access to a specific OpenShift project (namespace). To deploy this template, you must have the administrator role for the OpenShift project.
- **namespace with role-based access control**- The namespace template with configurable RBAC. For more information, see [Role-based access control for the Fuse Console on OpenShift 4.x](#).

To view a list of the parameters for the Fuse Console templates, run the following OpenShift command:

```
oc process --parameters -f https://raw.githubusercontent.com/jboss-fuse/application-templates/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001/fuse-console-namespace-os4.json
```

Prerequisites

- Before you install and deploy the Fuse Console, you must generate a client certificate that is signed with the service signing certificate authority as described in [Generating a certificate to secure the Fuse Console on OpenShift 4.x](#).
- You have the **cluster admin** role for the OpenShift cluster.
- You have configured authentication with **registry.redhat.io** as described in [Authenticating with registry.redhat.io for container images](#).
- The Fuse Console image stream (along with the other Fuse image streams) are installed, as described in [Installing Fuse imagestreams and templates on the OpenShift 4.x server](#).

Procedure

1. Verify that the Fuse Console image stream is installed by using the following command to retrieve a list of all templates:

```
oc get template -n openshift
```

- Optionally, if you want to update the already installed image stream with new release tags, use the following command to import the Fuse Console image to the **openshift** namespace:

```
oc import-image fuse7/fuse-console-rhel8:1.10 --from=registry.redhat.io/fuse7/fuse-console-rhel8:1.10 --confirm -n openshift
```

- Obtain the Fuse Console **APP_NAME** value by running the following command:

```
oc process --parameters -f TEMPLATE-FILENAME
```

where **TEMPLATE-FILENAME** is one of the following templates:

- Cluster template:
https://github.com/jboss-fuse/application-templates/blob/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001/fuse-console-cluster-os4.json
- Cluster template with configurable RBAC:
https://github.com/jboss-fuse/application-templates/blob/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001/fuse-console-cluster-rbac.yml
- Namespace template:
https://github.com/jboss-fuse/application-templates/blob/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001/fuse-console-namespace-os4.json
- Namespace template with configurable RBAC:
https://github.com/jboss-fuse/application-templates/blob/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001/fuse-console-namespace-rbac.yml

For example, for the cluster template with configurable RBAC, run this command:

```
oc process --parameters -f https://raw.githubusercontent.com/jboss-fuse/application-templates/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001/fuse-console-cluster-rbac.yml
```

- From the certificate that you generated in [Securing the Fuse Console on OpenShift 4.x](#), create the secret and mount it in the Fuse Console by using the following command (where **APP_NAME** is the name of the Fuse Console application).

```
oc create secret tls APP_NAME-tls-proxying --cert server.crt --key server.key
```

- Create a new application based on your local copy of the Fuse Console template by running the following command (where **myproject** is the name of your OpenShift project, **mytemp** is the path to the local directory that contains the Fuse Console template, and **myhost** is the hostname to access the Fuse Console:

- For the cluster template:

```
oc new-app -n myproject -f https://raw.githubusercontent.com/jboss-fuse/application-templates/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001/fuse-console-cluster-os4.json -p ROUTE_HOSTNAME=myhost
```

- For the cluster with RBAC template:

```
oc new-app -n myproject -f https://raw.githubusercontent.com/jboss-fuse/application-templates/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001/fuse-console-cluster-rbac.yml -p ROUTE_HOSTNAME=myhost
```

- For the namespace template:

```
oc new-app -n myproject -f https://raw.githubusercontent.com/jboss-fuse/application-templates/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001/fuse-console-namespace-os4.json
```

- For the namespace with RBAC template:

```
oc new-app -n myproject -f https://raw.githubusercontent.com/jboss-fuse/application-templates/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001/fuse-console-namespace-rbac.yml
```

6. To configure the Fuse Console so that it can open the OpenShift Web console, set the **OPENSHIFT_WEB_CONSOLE_URL** environment variable by running the following command:

```
oc set env dc/${APP_NAME} OPENSHIFT_WEB_CONSOLE_URL=`oc get -n openshift-config-managed cm console-public -o jsonpath={.data.consoleURL}`
```

7. Obtain the status and the URL of your Fuse Console deployment by running this command:

```
oc status
```

8. To access the Fuse Console from a browser, use the URL that is returned in Step 7 (for example, <https://fuse-console.192.168.64.12.nip.io>).

2.2.1. Generating a certificate to secure the Fuse Console on OpenShift 4.x

On OpenShift 4.x, to keep the connection between the Fuse Console proxy and the Jolokia agent secure, a client certificate must be generated before the Fuse Console is deployed. The service signing certificate authority private key must be used to sign the client certificate.

You must follow this procedure **only** if you are installing and deploying the Fuse Console by using the command line. If you are using the Fuse Console Operator, it handles this task for you.



IMPORTANT

You must generate and sign a separate client certificate for each OpenShift cluster. Do not use the same certificate for more than one cluster.

Prerequisites

- You have **cluster admin** access to the OpenShift cluster.
- If you are generating certificates for more than one OpenShift cluster and you previously generated a certificate for a different cluster in the current directory, do one of the following to ensure that you generate a different certificate for the current cluster:
 - Delete the existing certificate files (for example, **ca.crt**, **ca.key**, and **ca.srl**) from the current directory.

- Change to a different working directory. For example, if your current working directory is named **cluster1**, create a new **cluster2** directory and change your working directory to it:

```
mkdir ../cluster2
```

```
cd ../cluster2
```

Procedure

1. Login to OpenShift as a user with cluster admin access:

```
oc login -u <user_with_cluster_admin_role>
```

2. Retrieve the service signing certificate authority keys, by executing the following commands:

- To retrieve the certificate:

```
oc get secrets/signing-key -n openshift-service-ca -o "jsonpath={.data['tls.crt']}" | base64 --decode > ca.crt
```

- To retrieve the private key:

```
oc get secrets/signing-key -n openshift-service-ca -o "jsonpath={.data['tls.key']}" | base64 --decode > ca.key
```

3. Generate the client certificate, as documented in [Kubernetes certificates administration](#), using either **easysrsa**, **openssl**, or **cfssl**.

Here are the example commands using openssl:

- a. Generate the private key:

```
openssl genrsa -out server.key 2048
```

- b. Write the CSR config file.

```
cat <<EOT >> csr.conf
[ req ]
default_bits = 2048
prompt = no
default_md = sha256
distinguished_name = dn

[ dn ]
CN = fuse-console.fuse.svc

[ v3_ext ]
authorityKeyIdentifier=keyid,issuer:always
keyUsage=keyEncipherment,dataEncipherment,digitalSignature
extendedKeyUsage=serverAuth,clientAuth
EOT
```

Here, the values in the **CN** parameter refers to the application name and the namespace that the application uses.

- c. Generate the CSR:

```
openssl req -new -key server.key -out server.csr -config csr.conf
```

d. Issue the signed certificate:

```
openssl x509 -req -in server.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out server.crt
-days 10000 -extensions v3_ext -extfile csr.conf
```

Next steps

You need this certificate to create the secret for the Fuse Console as described in [Installing and deploying the Fuse Console on OpenShift 4.x by using the command line](#).

2.3. ROLE-BASED ACCESS CONTROL FOR THE FUSE CONSOLE ON OPENSIFT 4.X

The Fuse Console offers role-based access control (RBAC) that infers access according to the user authorization provided by OpenShift. In the Fuse Console, RBAC determines a user's ability to perform MBean operations on a pod.

For information on OpenShift authorization see the [Using RBAC to define and apply permissions](#) section of the OpenShift documentation.

Role-based access is enabled by default when you use the Operator to install the Fuse Console on OpenShift.

If you want to implement role-based access for the Fuse Console by installing it with a template, you must use one of the templates that are configurable with RBAC (**`fuse-console-cluster-rbac.yml`** or **`fuse-console-namespace-rbac.yml`**) to install the Fuse Console as described in [Installing and deploying the Fuse Console on OpenShift 4.x by using the command line](#).

Fuse Console RBAC leverages the user's **verb** access on a pod resource in OpenShift to determine the user's access to a pod's MBean operations in the Fuse Console. By default, there are two user roles for the Fuse Console:

- **admin**
If a user can **update** a pod in OpenShift, then the user is conferred the **admin** role for the Fuse Console. The user can perform **write** MBean operations in the Fuse Console for the pod.
- **viewer**
If a user can **get** a pod in OpenShift, then the user is conferred the **viewer** role for the Fuse Console. The user can perform **read-only** MBean operations in the Fuse Console for the pod.



NOTE

If you used a non-RBAC template to install the Fuse Console, only OpenShift users that are granted the **update** verb on the pod resource are authorized to perform the Fuse Console MBeans operations. Users that are granted the **get** verb on the pod resource can **view** the pod but they cannot perform any Fuse Console operations.

Additional resources

- [Determining access roles for the Fuse Console on OpenShift 4.x](#)
- [Customizing role-based access to the Fuse Console on OpenShift 4.x](#)

- [Disabling role-based access control for the Fuse Console on OpenShift 4.x](#)

2.3.1. Determining access roles for the Fuse Console on OpenShift 4.x

The Fuse Console role-based access control is inferred from a user's OpenShift permissions for a pod. To determine the Fuse Console access role granted to a particular user, obtain the OpenShift permissions granted to the user for a pod.

Prerequisites

- You know the user's name.
- You know the pod's name.

Procedure

- To determine whether a user has the Fuse Console **admin** role for the pod, run the following command to see whether the user can update the pod on OpenShift:

```
oc auth can-i update pods/<pod> --as <user>
```

If the response is **yes**, the user has the Fuse Console **admin** role for the pod. The user can perform **write** MBean operations in the Fuse Console for the pod.

- To determine whether a user has the Fuse Console **viewer** role for the pod, run the following command to see whether the user can **get** a pod on OpenShift:

```
oc auth can-i get pods/<pod> --as <user>
```

If the response is **yes**, the user has the Fuse Console **viewer** role for the pod. The user can perform **read-only** MBean operations in the Fuse Console for the pod. Depending on the context, the Fuse Console prevents the user with the **viewer** role from performing a **write** MBean operation, by disabling an option or by displaying an "operation not allowed for this user" message when the user attempts a **write** MBean operation.

If the response is **no**, the user is not bound to any Fuse Console roles and the user cannot view the pod in the Fuse Console.

Additional resources

- [Role-based access control for the Fuse Console on OpenShift 4.x](#)
- [Customizing role-based access to the Fuse Console on OpenShift 4.x](#)
- [Disabling role-based access control for the Fuse Console on OpenShift 4.x](#)

2.3.2. Customizing role-based access to the Fuse Console on OpenShift 4.x

If you use the OperatorHub to install the Fuse Console, role-based access control (RBAC) is enabled by default as described in [Role-based access control for the Fuse Console on OpenShift 4.x](#). If you want to customize the Fuse Console RBAC behavior, before you deploy the Fuse Console, you must provide a ConfigMap file (that defines the custom RBAC behavior). You must place the custom ConfigMap file in the same namespace in which you installed the Fuse Console Operator.

If you use the command line templates to install the Fuse Console, the **deployment-cluster-rbac.yml** and **deployment-namespace-rbac.yml** templates create a ConfigMap that contains the configuration file (**ACL.yml**). The configuration file defines the roles allowed for MBean operations.

Prerequisite

- You installed the Fuse Console by using the OperatorHub or by using one of the Fuse Console RBAC templates (**deployment-cluster-rbac.yml** or **deployment-namespace-rbac.yml**)

Procedure

To customize the Fuse Console RBAC roles:

1. If you installed the Fuse Console by using the command line, the installation templates include a default ConfigMap file and so you can skip to the next step.
If you installed the Fuse Console by using the OperatorHub, before you deploy the Fuse Console create a RBAC ConfigMap:

- a. Make sure the current OpenShift project is the project to which you want to install the Fuse Console. For example, if you want to install the Fuse Console in the **fusetest** project, run this command:

```
oc project fusetest
```

- b. To create a Fuse Console RBAC ConfigMap file from a template, run this command:

```
oc process -f {sb2-templates-base-url}/fuse-console-operator-rbac.yml -p  
APP_NAME=fuse-console | oc create -f -
```

2. Open the ConfigMap in an editor by running the following command:

```
oc edit cm $APP_NAME-rbac
```

For example:

```
oc edit cm fuse-console-rbac
```

3. Edit the file.
4. Save the file to apply the changes. OpenShift automatically restarts the Fuse Console pod.

Additional resources

- [Role-based access control for the Fuse Console on OpenShift 4.x](#)
- [Determining access roles for the Fuse Console on OpenShift 4.x](#)
- [Disabling role-based access control for the Fuse Console on OpenShift 4.x](#)

2.3.3. Disabling role-based access control for the Fuse Console on OpenShift 4.x

If you installed the Fuse Console by using the command line and you specified one of the Fuse Console RBAC templates, the Fuse Console's **HAWTIO_ONLINE_RBAC_ACL** environment variable passes the role-based access control (RBAC) ConfigMap configuration file path to the OpenShift server. If the

HAWTIO_ONLINE_RBAC_ACL environment variable is not specified, RBAC support is disabled and only users that are granted the **update** verb on the pod resource (in OpenShift) are authorized to call MBeans operations on the pod in the Fuse Console.

Note that when you use the OperatorHub to install the Fuse Console, role-based access is enabled by default and the **HAWTIO_ONLINE_RBAC_ACL** environment variable does not apply.

Prerequisite

You installed the Fuse Console by using the command line and you specified one of the Fuse Console RBAC templates (**deployment-cluster-rbac.yml** or **deployment-namespace-rbac.yml**).

Procedure

To disable role-based access for the Fuse Console:

1. In OpenShift, edit the **Deployment Config** resource for the Fuse Console.
2. Delete the entire **HAWTIO_ONLINE_RBAC_ACL** environment variable definition. (Note that only clearing its value is not sufficient).
3. Save the file to apply the changes. OpenShift automatically restarts the Fuse Console pod.

Additional resources

- [Role-based access control for the Fuse Console on OpenShift 4.x](#)
- [Determining access roles for the Fuse Console on OpenShift 4.x](#)
- [Customizing role-based access to the Fuse Console on OpenShift 4.x](#)

2.4. UPGRADING THE FUSE CONSOLE ON OPENSIFT 4.X

Red Hat OpenShift 4.x handles updates to operators, including the Red Hat Fuse operators. For more information see the [Operators OpenShift documentation](#).

In turn, operator updates can trigger application upgrades, depending on how the application is configured.

For Fuse Console applications, you can also trigger an upgrade to an application by editing the **.spec.version** field of the application custom resource definition.

Prerequisite

- You have OpenShift cluster admin permissions.

Procedure

To upgrade a Fuse Console application:

1. In a terminal window, use the following command to change the **.spec.version** field of the application custom resource definition:

```
oc patch -n <project-name> <custom-resource-name> --type='merge' -p '{"spec": {"version": "1.7.1"}}'
```

For example:

```
oc patch -n myproject hawtio/example-fuseconsole --type='merge' -p '{"spec": {"version": "1.7.1"}}'
```

2. Check that the application's status has updated:

```
oc get -n myproject hawtio/example-fuseconsole
```

The response shows information about the application, including the version number:

```
NAME          AGE  URL
example-fuseconsole 1m   https://fuseconsole.192.168.64.38.nip.io
docker.io/fuseconsole/online:1.7.1
```

When you change the value of the **.spec.version** field, OpenShift automatically redeploys the application.

3. To check the status of the redeployment that is triggered by the version change:

```
oc rollout status deployment.v1.apps/example-fuseconsole
```

A successful deployment shows this response:

```
deployment "example-fuseconsole" successfully rolled out
```

2.5. UPGRADING FUSE IMAGESTREAMS AND TEMPLATES ON THE OPENSIFT 4.X SERVER

OpenShift Container Platform 4.x uses the Samples Operator, which operates in the OpenShift namespace, upgrades and updates the Red Hat Enterprise Linux (RHEL)-based OpenShift Container Platform imagestreams and templates.

To upgrade the Fuse on OpenShift imagestreams and templates:

- Reconfigure the Samples Operator
- Add Fuse imagestreams and templates to **Skipped Imagestreams and Skipped Templates** fields.
 - Skipped Imagestreams: Imagestreams that are in the Samples Operator's inventory, but that the cluster administrator wants the Operator to ignore or not manage.
 - Skipped Templates: Templates that are in the Samples Operator's inventory, but that the cluster administrator wants the Operator to ignore or not manage.

Prerequisites

- You have access to OpenShift Server.
- You have configured authentication to **registry.redhat.io**.

Procedure

1. Start the OpenShift 4 Server.

2. Log in to the OpenShift Server as an administrator.

```
oc login --user system:admin --token=my-token --server=https://my-cluster.example.com:6443
```

3. Verify that you are using the project for which you created a docker-registry secret.

```
oc project openshift
```

4. View the current configuration of Samples operator.

```
oc get configs.samples.operator.openshift.io -n openshift-cluster-samples-operator -o yaml
```

5. Configure Samples operator to ignore the fuse templates and image streams that are added.

```
oc edit configs.samples.operator.openshift.io -n openshift-cluster-samples-operator
```

6. Add the Fuse imagestreams Skipped Imagestreams section and add Fuse and Spring Boot 2 templates to Skipped Templates section.

```
[...]
spec:
  architectures:
  - x86_64
  managementState: Managed
  skippedImagestreams:
  - fuse-console-rhel8
  - fuse-eap-openshift-jdk8-rhel7
  - fuse-eap-openshift-jdk11-rhel8
  - fuse-java-openshift-rhel8
  - fuse-java-openshift-jdk11-rhel8
  - fuse-karaf-openshift-rhel8
  - fuse-karaf-openshift-jdk11-rhel8
  - fuse-apicurito-generator-rhel8
  - fuse-apicurito-rhel8
  skippedTemplates:
  - s2i-fuse713-eap-camel-amq
  - s2i-fuse713-eap-camel-cdi
  - s2i-fuse713-eap-camel-cxf-jaxrs
  - s2i-fuse713-eap-camel-cxf-jaxws
  - s2i-fuse713-karaf-camel-amq
  - s2i-fuse713-karaf-camel-log
  - s2i-fuse713-karaf-camel-rest-sql
  - s2i-fuse713-karaf-cxf-rest
  - s2i-fuse713-spring-boot-2-camel-amq
  - s2i-fuse713-spring-boot-2-camel-config
  - s2i-fuse713-spring-boot-2-camel-drools
  - s2i-fuse713-spring-boot-2-camel-infinispan
  - s2i-fuse713-spring-boot-2-camel-rest-3scale
  - s2i-fuse713-spring-boot-2-camel-rest-sql
  - s2i-fuse713-spring-boot-2-camel
  - s2i-fuse713-spring-boot-2-camel-xa
  - s2i-fuse713-spring-boot-2-camel-xml
  - s2i-fuse713-spring-boot-2-cxf-jaxrs
```

```
- s2i-fuse713-spring-boot-2-cxf-jaxws
- s2i-fuse713-spring-boot-2-cxf-jaxrs-xml
- s2i-fuse713-spring-boot-2-cxf-jaxws-xml
```

- Upgrade Fuse on OpenShift image streams.

```
BASEURL=https://raw.githubusercontent.com/jboss-fuse/application-templates/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001
```

```
oc replace -n openshift -f ${BASEURL}/fis-image-streams.json
```

- Upgrade Fuse on OpenShift quickstart templates:

```
for template in eap-camel-amq-template.json \
eap-camel-cdi-template.json \
eap-camel-cxf-jaxrs-template.json \
eap-camel-cxf-jaxws-template.json \
karaf-camel-amq-template.json \
karaf-camel-log-template.json \
karaf-camel-rest-sql-template.json \
karaf-cxf-rest-template.json ;
do
oc replace -n openshift \
${BASEURL}/quickstarts/${template}
done
```

- Upgrade Spring Boot 2 quickstart templates:

```
for template in spring-boot-2-camel-amq-template.json \
spring-boot-2-camel-config-template.json \
spring-boot-2-camel-drools-template.json \
spring-boot-2-camel-infinispan-template.json \
spring-boot-2-camel-rest-3scale-template.json \
spring-boot-2-camel-rest-sql-template.json \
spring-boot-2-camel-template.json \
spring-boot-2-camel-xa-template.json \
spring-boot-2-camel-xml-template.json \
spring-boot-2-cxf-jaxrs-template.json \
spring-boot-2-cxf-jaxws-template.json \
spring-boot-2-cxf-jaxrs-xml-template.json \
spring-boot-2-cxf-jaxws-xml-template.json ;
do oc replace -n openshift \
${BASEURL}/quickstarts/${template}
done
```

- (Optional)* View the upgraded Fuse on OpenShift templates:

```
oc get template -n openshift
```

2.6. TUNING THE PERFORMANCE OF THE FUSE CONSOLE ON OPENSIFT 4.X

By default, the Fuse Console uses the following Nginx settings:

- **clientBodyBufferSize: 256k**
- **proxyBuffers: 16 128k**
- **subrequestOutputBufferSize: 10m**

Note: For descriptions of these settings, see the Nginx documentation:
<http://nginx.org/en/docs/dirindex.html>

To tune performance of the Fuse Console, you can set any of the **clientBodyBufferSize**, **proxyBuffers**, and **subrequestOutputBufferSize** environment variables. For example, if you are using the Fuse Console to monitor numerous pods and routes (for instance, 100 routes in total), you can resolve a loading timeout issue by setting the Fuse Console's **subrequestOutputBufferSize** environment variable to between **60m** to **100m**.

How you set these environment variables depends on how you installed the Fuse Console on OpenShift 4.x:

- By using the Fuse Console Operator
- By using a Fuse Console template

2.6.1. Performance tuning for Fuse Console Operator installation

On OpenShift 4.x, you can set the Nginx performance tuning environment variables before or after you deploy the Fuse Console. If you do so afterwards, OpenShift redeploys the Fuse Console.

Prerequisites

- You have **cluster admin** access to the OpenShift cluster.
- You have installed the Fuse Console Operator as described in [Installing and deploying the Fuse Console on OpenShift 4.x by using the OperatorHub](#).

Procedure

You can set the environment variables before or after you deploy the Fuse Console.

- **To set the environment variables before you deploy the Fuse Console:**
 1. In the OpenShift web console, in a project that has the Fuse Console Operator installed, select **Operators** > **Installed Operators** > **Red Hat Integration - Fuse Console**
 2. Click the **Hawtio** tab, and then click **Create Hawtio**.
 3. On the **Create Hawtio** page, in the **Form view**, scroll down to the **Config** > **Nginx** section.
 4. Expand the **Nginx** section and then set the environment variables. For example:
 - **clientBodyBufferSize: 256k**
 - **proxyBuffers: 16 128k**
 - **subrequestOutputBufferSize: 100m**
 5. Save the configuration.
 6. Click **Create** to deploy the Fuse Console.

7. After the deployment completes, open the **Deployments** > **fuse-console** page, and then click **Environment** to verify that the environment variables are in the list.
- **To set the environment variables after you deploy the Fuse Console:**
 1. In the OpenShift web console, open the project in which the Fuse Console is deployed.
 2. Select **Operators** > **Installed Operators** > **Red Hat Integration - Fuse Console**
 3. Click the **Hawtio** tab, and then click **fuse-console**.
 4. Select **Actions** > **Edit Hawtio**.
 5. In the Editor window, scroll down to the **spec** section.
 6. Under the **spec** section, add a new **nginx** section and specify one or more environment variables, for example:

```

apiVersion: hawt.io/v1alpha1
kind: Hawtio
metadata:
  name: fuse-console
spec:
  type: Namespace
  nginx:
    clientBodyBufferSize: 256k
    proxyBuffers: 16 128k
    subrequestOutputBufferSize: 100m
  .
  .
  .

```

7. Click **Save**.
OpenShift redeploys the Fuse Console.
8. After the redeployment completes, open the **Workloads** > **Deployments** > **fuse-console** page, and then click **Environment** to see the environment variables in the list.

2.6.2. Performance tuning for Fuse Console template installation

On Openshift 4.x, you can set the Nginx performance tuning environment variables before or after you deploy the Fuse Console. If you do so afterwards, OpenShift redeploys the Fuse Console.

Prerequisites

- You have **cluster admin** access to the OpenShift cluster.
- You have installed the Fuse Console templates on your OpenShift as described in [Installing Fuse imagestreams and templates on the OpenShift 4.x server](#).

Procedure

You can set the environment variables before or after you deploy the Fuse Console.

- **To set the environment variables before you deploy the Fuse Console:**
 1. Determine which Fuse Console template that you want to use:

- Cluster template (**fuse-console-cluster-os4.json**)
 - Cluster template with configurable RBAC (**fuse-console-cluster-rbac.yml**)
 - Namespace template (**fuse-console-namespace-os4.json**)
 - Namespace template with configurable RBAC (**fuse-console-namespace-rbac.yml**)
2. Edit the local copy of the Fuse Console template that you want to use for the Fuse Console to include the **NGINX_CLIENT_BODY_BUFFER_SIZE**, **NGINX_PROXY_BUFFERS**, and/or **NGINX_SUBREQUEST_OUTPUT_BUFFER_SIZE** environment variables as shown in the following example:

```
apiVersion: apps.openshift.io/v1
kind: DeploymentConfig
metadata:
  name: fuse-console
spec:
  template:
    spec:
      containers:
      - env:
        - name: NGINX_CLIENT_BODY_BUFFER_SIZE
          value: 256k
        - name: NGINX_PROXY_BUFFERS
          value: 16 128k
        - name: NGINX_SUBREQUEST_OUTPUT_BUFFER_SIZE
          value: 100m
```

3. Save your changes.
 4. Follow the steps for installing and deploying the Fuse Console as as described in [Setting up the Fuse Console on OpenShift 4.x](#).
- **To set the environment variables after you deploy the Fuse Console:**

1. In a Terminal window, login to the OpenShift cluster.
2. Open the project in which the Fuse Console is deployed. For example, if the Fuse Console is deployed in the **myfuse** project, use the following command:
oc project myfuse

3. Obtain the name for the Fuse Console deployment:
oc get deployments

This command returns a list of the deployments running in the current project. For example:

```
NAME                READY  UP-TO-DATE  AVAILABLE  AGE
fuse-console        1/1    1            1          114m
```

4. Run one or more of the following commands to set the environment variables for the Fuse Console deployment:

```
oc set env dc/fuse-console NGINX_CLIENT_BODY_BUFFER_SIZE="256k"
oc set env dc/fuse-console NGINX_PROXY_BUFFERS="16 128k"
```

```
oc set env dc/fuse-console NGINX_SUBREQUEST_OUTPUT_BUFFER_SIZE="10m"
```

OpenShift redeploys the Fuse Console.

5. After the redeployment completes, verify the environment variables settings:
 - a. Obtain the Fuse Console pod name:

```
oc get pods
```

- b. Run the following command to view the environment settings

```
oc exec <fuse-console-podname> -- cat /opt/app-root/etc/nginx.d/nginx-gateway.conf | grep "Performance tuning" -A 3
```

For example, if the pod name is **fuse-console-6646cbbd4c-9rplg**, run this command:

```
oc exec fuse-console-6646cbbd4c-9rplg -- cat /opt/app-root/etc/nginx.d/nginx-gateway.conf | grep "Performance tuning" -A 3
```

2.6.3. Performance tuning for viewing applications on Fuse Console

Enhanced performance tuning capability of Fuse console allows you to view the applications with a large number of MBeans. To use this capability perform following steps.

Prerequisites

- You have **cluster admin** access to the OpenShift cluster.
- You have installed the Fuse Console Operator as described in [Installing and deploying the Fuse Console on OpenShift 4.x by using the OperatorHub](#).

Procedure

1. Increase the memory limit for the applications.

It is necessary to increase the memory limit, for example from 256Mi to 512 Mi, so that applications do not crash with OOM error before reaching the Fuse console. For Fuse quickstart, edit your application's **src/main/jkube/deployment.yml** file.

```
spec:
  template:
    spec:
      containers:
      -
        resources:
          [...]
          limits:
            cpu: "1.0"
            memory: 512Mi
```

2. Ensure that the **Fuse Console Deployment** or **DeploymentConfig** has an enough memory limit. If it's not enough, increase the limit, for example, from 200Mi to 512Mi.

3. If you see the "too big subrequest response while sending to client" error in nginx log, apply the solution mentioned in the [Section 2.6.1, "Performance tuning for Fuse Console Operator installation"](#) section.

CHAPTER 3. SETTING UP THE FUSE CONSOLE ON OPENSIFT 3.11

On OpenShift 3.11, you can access the Fuse Console:

- By adding the Fuse Console to an OpenShift project so that you can monitor all the running Fuse containers in the project.
- By adding the Fuse Console to an OpenShift cluster so that you can monitor all the running Fuse containers in all projects on the cluster.
- By opening it from a specific Fuse pod so that you can monitor that single running Fuse container.

You deploy the Fuse Console templates from the command line.



NOTE

To install Fuse Console on Minishift or CDK based environments, follow the steps explained in the KCS article below.

- To install Fuse Console on Minishift or CDK based environments, see [KCS 4998441](#).
- If it is necessary to disable Jolokia authentication see the workaround described in [KCS 3988671](#).

Prerequisite

- Install the Fuse on OpenShift image streams and the templates for the Fuse Console as described in [Fuse on OpenShift Guide](#).



NOTE

- User management for the Fuse Console is handled by OpenShift.
- Role-based access control (for users accessing the Fuse Console after it is deployed) is not yet available for Fuse on OpenShift 3.11.

[Section 3.1, “Deploying the Fuse Console on OpenShift 3.11”](#)

[Section 3.2, “Monitoring a single Fuse pod from the Fuse Console on OpenShift 3.11”](#)

3.1. DEPLOYING THE FUSE CONSOLE ON OPENSIFT 3.11

[Table 3.1, “Fuse Console templates”](#) describes the OpenShift 3.11 templates that you can use to deploy the Fuse Console from the command line, depending on the type of Fuse application deployment.

Table 3.1. Fuse Console templates

Type	Description
------	-------------

Type	Description
fis-console-cluster-template.json	The Fuse Console can discover and connect to Fuse applications deployed across multiple namespaces or projects. To deploy this template, you must have the OpenShift cluster-admin role.
fis-console-namespace-template.json	This template restricts the Fuse Console access to the current OpenShift project (namespace), and as such acts as a single tenant deployment. To deploy this template, you must have the admin role for the current OpenShift project.

Optionally, you can view a list of the parameters for all of the templates by running this command:

```
oc process --parameters -f https://raw.githubusercontent.com/jboss-fuse/application-templates/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001/fis-console-namespace-template.json
```



NOTE

The Fuse Console templates configure end-to-end encryption by default so that your Fuse Console requests are secured end-to-end, from the browser to the in-cluster services.

Prerequisite

- For cluster mode on OpenShift 3.11, you need the cluster admin role and the cluster mode template. Run the following command:

```
oc adm policy add-cluster-role-to-user cluster-admin system:serviceaccount:openshift-infra:template-instance-controller
```

Procedure

To deploy the Fuse Console from the command line:

1. Create a new application based on a Fuse Console template by running one of the following commands (where **myproject** is the name of your project):
 - For the Fuse Console **cluster** template, where **myhost** is the hostname to access the Fuse Console:

```
oc new-app -n myproject -f https://raw.githubusercontent.com/jboss-fuse/application-templates/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001/fis-console-cluster-template.json -p ROUTE_HOSTNAME=myhost
```

- For the Fuse Console **namespace** template:

```
oc new-app -n myproject -f https://raw.githubusercontent.com/jboss-fuse/application-templates/application-templates-2.1.0.fuse-7_13_0-00014-redhat-00001/fis-console-namespace-template.json
```

**NOTE**

You can omit the `route_hostname` parameter for the **namespace** template because OpenShift automatically generates one.

2. Obtain the status and the URL of your Fuse Console deployment by running this command:

```
oc status
```

3. To access the Fuse Console from a browser, use the provided URL.

Example:

+<https://fuse-console.192.168.64.12.nip.io>.

3.2. MONITORING A SINGLE FUSE POD FROM THE FUSE CONSOLE ON OPENSIFT 3.11

You can open the Fuse Console for a Fuse pod running on OpenShift 3.11.

Prerequisite

- In order to configure OpenShift to display a link to Fuse Console in the pod view, the pod running a Fuse on OpenShift image must declare a TCP port within a name attribute set to **jolokia**:

```
{
  "kind": "Pod",
  [...]
  "spec": {
    "containers": [
      {
        [...]
        "ports": [
          {
            "name": "jolokia",
            "containerPort": 8778,
            "protocol": "TCP"
          }
        ]
      }
    ]
  }
}
```





Procedure

1. From the **Applications → Pods** view in your OpenShift project, click on the pod name to view the details of the running Fuse pod. On the right-hand side of this page, you see a summary of the container template:

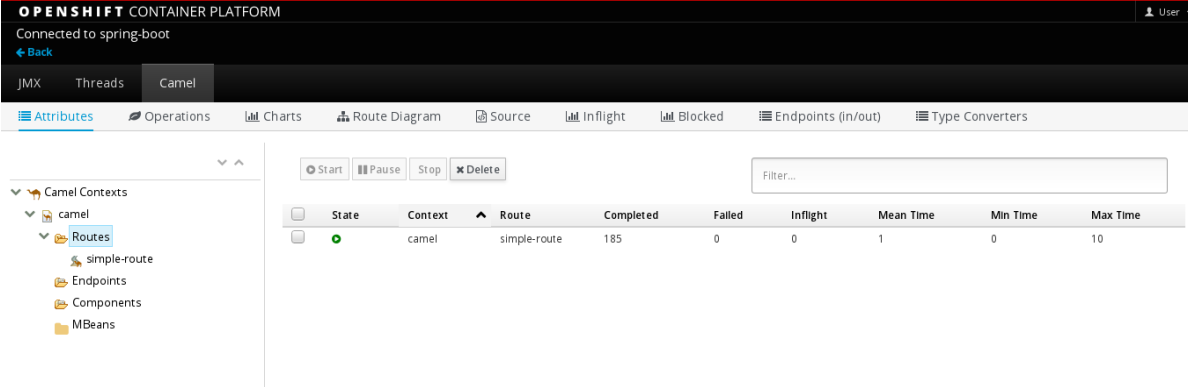
Template

Containers

CONTAINER: SPRING-BOOT

-  **Image:** [test/fuse70-spring-boot](#) eda527f 193.1 MiB
 -  **Build:** [fuse70-spring-boot-s2i, #2](#)
 -  **Source:** Binary
 -  **Ports:** 8080/TCP (http), 8778/TCP (jolokia), 9779/TCP (prometheus)
 -  **Mount:** default-token-p4zsn → /var/run/secrets/kubernetes.io/serviceaccount
read-only
 -  **CPU:** 200 millicores to 1 core
 -  **Readiness Probe:** GET /health on port 8081 (HTTP) 10s delay, 1s timeout
 -  **Liveness Probe:** GET /health on port 8081 (HTTP) 180s delay, 1s timeout
-  [Open Java Console](#)

2. From this view, click on the **Open Java Console** link to open the Fuse Console.



OPENSIFT CONTAINER PLATFORM

Connected to spring-boot

← Back

JMX Threads Camel

Attributes Operations Charts Route Diagram Source Inflight Blocked Endpoints (in/out) Type Converters

Start Pause Stop Delete

Filter...

State	Context	Route	Completed	Failed	Inflight	Mean Time	Min Time	Max Time
●	camel	simple-route	185	0	0	1	0	10

Camel Contexts

- camel
 - Routes
 - simple-route
 - Endpoints
 - Components
 - MBeans

CHAPTER 4. VIEWING CONTAINERS AND APPLICATIONS

When you login to the Fuse Console for OpenShift, the Fuse Console home page shows the available containers.

Procedure

- To manage (create, edit, or delete) containers, use the OpenShift console.
- To view Fuse applications and AMQ Brokers (if applicable) on the OpenShift cluster, click the **Online** tab.

CHAPTER 5. VIEWING AND MANAGING APACHE CAMEL APPLICATIONS

In the Fuse Console's **Camel** tab, you view and manage Apache Camel contexts, routes, and dependencies.

You can view the following details:

- A list of all running Camel contexts
- Detailed information of each Camel context such as Camel version number and runtime statics
- Lists of all routes in each Camel application and their runtime statistics
- Graphical representation of the running routes along with real time metrics

You can also interact with a Camel application by:

- Starting and suspending contexts
- Managing the lifecycle of all Camel applications and their routes, so you can restart, stop, pause, resume, etc.
- Live tracing and debugging of running routes
- Browsing and sending messages to Camel endpoints

Prerequisite

The **Camel** tab is only available when you connect to a container that uses one or more Camel routes.

5.1. STARTING, SUSPENDING, OR DELETING A CONTEXT

1. In the **Camel** tab's tree view, click **Camel Contexts**.
2. Check the box next to one or more contexts in the list.
3. Click **Start** or **Suspend**.
4. To delete a context:
 - a. Stop the context.
 - b. Click the ellipse icon and then select **Delete** from the dropdown menu.



NOTE

When you delete a context, you remove it from the deployed application.

5.2. VIEWING CAMEL APPLICATION DETAILS

1. In the **Camel** tab's tree view, click a Camel application.
2. To view a list of application attributes and values, click **Attributes**.

3. To view a graphical representation of the application attributes, click **Chart** and then click **Edit** to select the attributes that you want to see in the chart.
4. To view inflight and blocked exchanges, click **Exchanges**.
5. To view application endpoints, click **Endpoints**. You can filter the list by **URL**, **Route ID**, and **direction**.
6. To view, enable, and disable statistics related to the Camel built-in type conversion mechanism that is used to convert message bodies and message headers to different types, click **Type Converters**.
7. To view and execute JMX operations, such as adding or updating routes from XML or finding all Camel components available in the classpath, click **Operations**.

5.3. VIEWING A LIST OF THE CAMEL ROUTES AND INTERACTING WITH THEM

1. To view a list of routes:
 - a. Click the **Camel** tab.
 - b. In the tree view, click the application's routes folder:

Routes

<input type="checkbox"/>	Name ^	State
<input type="checkbox"/>	_route1	Started
<input type="checkbox"/>	_route2	Started

2. To start, stop, or delete one or more routes:
 - a. Check the box next to one or more routes in the list.
 - b. Click **Start** or **Stop**.
 - c. To delete a route, you must first stop it. Then click the ellipse icon and select **Delete** from the dropdown menu.

Routes

<input checked="" type="checkbox"/>	Name ^	State
<input checked="" type="checkbox"/>	_route1	Started
<input checked="" type="checkbox"/>	_route2	Started

**NOTE**

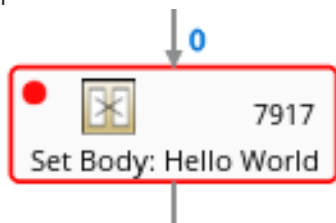
- When you delete a route, you remove it from the deployed application.
- You can also select a specific route in the tree view and then click the upper-right menu to start, stop, or delete it.

3. To view a graphical diagram of the routes, click **Route Diagram**.
4. To view inflight and blocked exchanges, click **Exchanges**.
5. To view endpoints, click **Endpoints**. You can filter the list by URL, Route ID, and direction.
6. Click **Type Converters** to view, enable, and disable statistics related to the Camel built-in type conversion mechanism, which is used to convert message bodies and message headers to different types.
7. To interact with a specific route:
 - a. In the **Camel** tab's tree view, select a route.
 - b. To view a list of route attributes and values, click **Attributes**.
 - c. To view a graphical representation of the route attributes, click **Chart**. You can click **Edit** to select the attributes that you want to see in the chart.
 - d. To view inflight and blocked exchanges, click **Exchanges**.
 - e. Click **Operations** to view and execute JMX operations on the route, such as dumping the route as XML or getting the route's Camel ID value.
8. To trace messages through a route:
 - a. In the **Camel** tab's tree view, select a route.
 - b. Select **Trace**, and then click **Start tracing**.
9. To send messages to a route:
 - a. In the **Camel** tab's tree view, open the context's endpoints folder and then select an endpoint.
 - b. Click the **Send** subtab.
 - c. Configure the message in JSON or XML format.
 - d. Click **Send**.
 - e. Return to the route's **Trace** tab to view the flow of messages through the route.

5.4. DEBUGGING A ROUTE

1. In the **Camel** tab's tree view, select a route.
2. Select **Debug**, and then click **Start debugging**.

- To add a breakpoint, select a node in the diagram and then click **Add breakpoint**. A red dot appears in the node:



The node is added to the list of breakpoints:

Breakpoints	
setBody1	×
log1	×

- Click the down arrow to step to the next node or the **Play** button to resume running the route.
- Click the **Pause** button to suspend all threads for the route.
- Click **Stop debugging** when you are done. All breakpoints are cleared.

CHAPTER 6. VIEWING AMQ BROKERS

You can configure the Fuse Console to view all AMQ brokers that are deployed on the OpenShift cluster.

Prerequisites

Each AMQ broker image (that you want to view in the Fuse Console) must be:

- Installed on the same OpenShift cluster that the Fuse Console is installed on.
- Configured so that the Fuse Console can recognize and connect to it, as described in the [section on enabling the Artemis plugin in the Fuse Console](#) in the AMQ Broker documentation.

Procedure

- Click **Artemis** to view the AMQ management console and monitor the status of AMQ Broker. (The AMQ Broker is based on [Apache ActiveMQ Artemis](#).)

For information on using the AMQ management console, see [Using AMQ Management Console](#) in the *Managing AMQ Broker* guide.

CHAPTER 7. VIEWING AND MANAGING JMX DOMAINS AND MBEANS

Java Management Extensions (JMX) is a Java technology that allows you to manage resources (services, devices, and applications) dynamically at runtime. The resources are represented by objects called MBeans (for Managed Bean). You can manage and monitor resources as soon as they are created, implemented, or installed.

With the JMX plugin on the Fuse Console, you can view and manage JMX domains and MBeans. You can view MBean attributes, run commands, and create charts that show statistics for the MBeans.

The **JMX** tab provides a tree view of the active JMX domains and MBeans organized in folders. You can view details and execute commands on the MBeans.

Procedure

1. To view and edit MBean attributes:
 - a. In the tree view, select an MBean.
 - b. Click the **Attributes** tab.
 - c. Click an attribute to see its details.
2. To perform operations:
 - a. In the tree view, select an MBean.
 - b. Click the **Operations** tab, expand one of the listed operations.
 - c. Click **Execute** to run the operation.
3. To view charts:
 - a. In the tree view, select an item.
 - b. Click the **Chart** tab.

CHAPTER 8. VIEWING AND MANAGING QUARTZ SCHEDULES

Quartz (<http://www.quartz-scheduler.org/>) is a richly featured, open source job scheduling library that you can integrate within most Java applications. You can use Quartz to create simple or complex schedules for executing jobs. A job is defined as a standard Java component that can execute virtually anything that you program it to do.

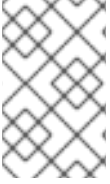
The Fuse Console shows the **Quartz** tab if your Camel route deploys the **camel-quartz2** component. Note that you can alternately access Quartz mbeans through the JMX tree view.

Procedure

1. In the Fuse Console, click the **Quartz** tab.
The **Quartz** page includes a treeview of the Quartz Schedulers and **Scheduler**, **Triggers**, and **Jobs** tabs.
2. To pause or start a scheduler, click the buttons on the **Scheduler** tab.
3. Click the **Triggers** tab to view the triggers that determine when jobs will run. For example, a trigger can specify to start a job at a certain time of day (to the millisecond), on specified days, or repeated a specified number of times or at specific times.
 - To filter the list of triggers select **State**, **Group**, **Name**, or **Type** from the drop-down list. You can then further filter the list by selecting or typing in the fill-on field.
 - To pause, resume, update, or manually fire a trigger, click the options in the **Action** column.
4. Click the **Jobs** tab to view the list of running jobs. You can sort the list by the columns in the table: **Group**, **Name**, **Durable**, **Recover**, **Job ClassName**, and **Description**.

CHAPTER 9. VIEWING DIAGNOSTICS

Use the **Diagnostics** tab to view diagnostic information about the JVM via the JVM DiagnosticCommand and HotspotDiagnostic interfaces.



NOTE

The functionality is similar to the **Diagnostic Commands** view in Java Mission Control (jmc) or the command line tool jcmd. The plugin will provide corresponding jcmd commands in some scenarios.

Procedure

1. To retrieve the number of instances of loaded classes and the amount of bytes they take up, click **Class Histogram**. If the operation is repeated, the tab shows the difference since last run.
2. To view the JVM diagnostic flag setting, click the **JVM flags**.
3. For a running JVM, you can also modify the flag settings.

Additional resources

The supported JVM depends on the platform, for more information go to one of the following sources:

- <http://www.oracle.com/technetwork/java/vmoptions-jsp-140102.html>
- <http://openjdk.java.net/groups/hotspot/docs/RuntimeOverview.html>

CHAPTER 10. VIEWING THREADS

You can view and monitor the state of threads.

Procedure

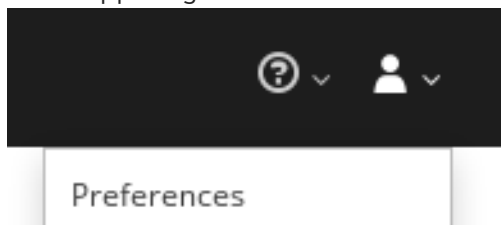
1. Click the **Runtime** tab and then the **Threads** subtab. The **Threads** page lists active threads and stack trace details for each thread. By default, the thread list shows all threads in descending ID order.
2. To sort the list by increasing ID, click the **ID** column label.
3. Optionally, filter the list by thread state (for example, **Blocked**) or by thread name.
4. To drill down to detailed information for a specific thread, such as the lock class name and full stack trace for that thread, in the **Actions** column, click **More**.

CHAPTER 11. ENSURING THAT DATA DISPLAYS CORRECTLY IN THE FUSE CONSOLE

If the display of the queues and connections in the Fuse Console is missing queues, missing connections, or displaying inconsistent icons, adjust the Jolokia collection size parameter that specifies the maximum number of elements in an array that Jolokia marshals in a response.

Procedure

1. In the upper right corner of the Fuse Console, click the user icon and then click **Preferences**.



2. Increase the value of the **Maximum collection size** option (the default is 50,000).
3. Click **Close**.

APPENDIX A. FUSE CONSOLE CONFIGURATION PROPERTIES

By default, the Fuse Console configuration is defined in the **hawtconfig.json** file. You can customize the Fuse Console configuration information, such as title, logo, and login page information.

Table A.1, “Fuse Console Configuration Properties” provides a description of the properties and lists whether or not each property requires a value.

Table A.1. Fuse Console Configuration Properties

Section	Property Name	Default Value	Description	Required?
About	Title	Red Hat Fuse Management Console	The title that shows on the About page of the Fuse Console.	Required
	productInfo	<i>Empty value</i>	Product information that shows on the About page of the Fuse Console.	Optional
	additionalInfo	<i>Empty value</i>	Any additional information that shows on the About page of the Fuse Console.	Optional
	copyright	<i>Empty value</i>	Copyright information that shows on the About page of the Fuse Console.	Optional
	imgSrc	img/Logo-RedHat-A-Reverse-RGB.png	The image that appears on the About page of the Fuse Console.	Required
branding	appName	Red Hat Fuse Management Console	The name for your application. This name displays in the title bar of the Fuse Console.	Required

Section	Property Name	Default Value	Description	Required?
	appLogoUrl	img/Logo-Red_Hat-Fuse-A-Reverse- RGB.png	The path to your application logo image file that displays in the Fuse Console navigation bar. The value can be a path relative to the Hawtio status URL or an absolute URL.	Required
	Css		The URL of an external CSS stylesheet, that can be used to style the application. It can be a path, relative to the Hawtio status URL, or it can be an absolute URL.	Optional
	companyLogoUrl	img/Logo-RedHat-A-Reverse- RGB.png	The path to your company logo image file.	Required
	Favicon		The URL of the favicon, that usually displays in the Web browser tab. It can be a path, relative to the Hawtio status URL, or it can be an absolute URL.	Optional
login	description	<i>Empty value</i>	Descriptive text that displays on the Fuse Console Login page (for example, http://localhost:8181/hawtio).	Optional

Section	Property Name	Default Value	Description	Required?
	links	[]	Specify an array of "url" and "text" pairs to provide additional links to pages where the user can get more information or help.	Optional
disabledRoutes	<i>none</i>	[]	Disables specific paths (i.e., plugins) on the console. Do not change this section. Any change is not supported for distributions other than OpenShift.	Optional