Red Hat Fuse 7.10

Managing Fuse on Karaf Standalone

Manage Fuse applications with the Fuse Console

Last Updated: 2023-07-24

Red Hat Fuse 7.10 Managing Fuse on Karaf Standalone

Manage Fuse applications with the Fuse Console

Legal Notice

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

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 the Prometheus documentation.

The audience for this guide is Red Hat Fuse on Apache Karaf 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. ACCESSING THE FUSE CONSOLE

To access the Fuse Console for Apache Karaf standalone, follow these steps.

Prerequisite

Install Fuse on the Karaf container. For step-by-step instructions, see Installing on Apache Karaf.

Procedure

1. In the command line, navigate to the directory in which you installed Red Hat Fuse and run the following command to start Fuse standalone:

   ```
   ./bin/fuse
   ```

   The Karaf console starts and shows version information, the default Fuse Console URL, and a list of common commands.

2. In a browser, type the URL to connect to the Fuse Console. For example:
   http://localhost:8181/hawtio

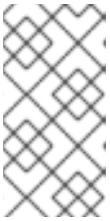
3. In the login page, type your user name and password and then click **Log In**.

By default, the Fuse Console shows the Home page. The left navigation tabs indicate the running plugins.

CHAPTER 3. SECURING THE FUSE CONSOLE

To secure the Fuse Console on Apache Karaf:

- **Disable the Fuse Console's proxy servlet when deploying to AWS**
  If you want to deploy a standalone Fuse application to Amazon Web Services (AWS), you should disable the Fuse Console's proxy servlet by setting the **hawtio.disableProxy** system property to **true**.



NOTE

When you disable the Fuse Console proxy servlet, the Fuse Console's **Connect** tab is disabled and you cannot connect to other JVMs from the Fuse Console. If you want to deploy more than one Fuse application on AWS, you must deploy the Fuse Console for each application.

- **Set HTTPS as the required protocol**
  You can use the **hawtio.http.strictTransportSecurity** property to require web browsers to use the secure HTTPS protocol to access the Fuse Console. This property specifies that web browsers that try to use HTTP to access the Fuse Console must automatically convert the request to use HTTPS.

- **Use public keys to secure responses**
  You can use the **hawtio.http.publicKeyPins** property to secure the HTTPS protocol by telling the web browser to associate a specific cryptographic public key with the Fuse Console to decrease the risk of "man-in-the-middle" attacks with forged certificates.

- **Enable SSL/TLS security**
  SSL/TLS security is not enabled by default for the Fuse Console. It is recommended that you enable SSL/TLS security on the Fuse Console to protect username/password credentials from snooping.

- **Implement Red Hat Single Sign On**

- **Control user access**
  The operations that an authenticated user is allowed to perform depend on the role (or roles) assigned to that user, as listed in Table 4.1, "Role-based access on Karaf standalone".

Procedure

1. To set HTTPS as the required protocol, set the **hawtio.http.strictTransportSecurity** property in the **$KARAF_HOME/etc/system.properties** file as shown in the following example:

```
hawtio.http.strictTransportSecurity = max-age=31536000; includeSubDomains; preload
```

2. To use public keys to secure responses, set the **hawtio.http.publicKeyPins** property in the **$KARAF_HOME/etc/system.properties** file as shown in the following example:

```
hawtio.http.publicKeyPins = pin-
sha256="cUPcTAZWKaASuYWhhneDttWpY3oBAkE3h2+soZS7sWs"; max-age=5184000;
includeSubDomains
```

3. (For deploying on AWS only) To disable the Fuse Console's proxy servlet, set the **hawtio.disableProxy** property to **true** in the **$KARAF_HOME/etc/system.properties** file as shown in the following example:

```
hawtio.disableProxy = true;
```

4. For detailed instructions on how to enable SSL/TLS security, see the "Enabling SSL/TLS for Undertow in an Apache Karaf container" section in the Apache Karaf Security Guide.

5. For information on how to secure the Fuse Console with Red Hat Single Sign-On, see the section on securing the Hawtio administration console in the *Red Hat Single Sign-on Securing Applications and Services Guide*.

6. To ensure that a user has the necessary user role authorization to perform the Fuse Console operations that the user needs to perform, follow these steps to set a user role:

   a. Open the Red Hat Fuse **etc/users.properties** file in an editor.

   b. Add an entry for the user name, password, and role.
   For example, the following entry in the **etc/users.properties** file defines the admin user and grants the admin role.

```
admin = secretpass,admin
```

   c. Save the file.

Additional resources

- For a description of the **hawtio.http.strictTransportSecurity** property's syntax, see the description page for the HTTP Strict Transport Security (HSTS) response header.
- For a description of the **hawtio.http.publicKeyPins** property's syntax, including instructions on how to extract the Base64 encoded public key, see the description page for the HTTP Public Key Pinning response header.

CHAPTER 4. ROLE-BASED ACCESS REFERENCE

The operations that an authenticated user is allowed to perform depend on the role (or roles) assigned to that user, as listed in Table 4.1, “Role-based access on Karaf standalone”.

Table 4.1. Role-based access on Karaf standalone

Operation	admin	manager	viewer
Log in/Log out	Y	Y	Y
View Help topics	Y	Y	Y
Set user preferences	Y	Y	Y
Connect			
Discover and connect to remote integrations	Y	Y	Y
Discover and connect to local integrations	Y	Y	Y
Camel			
View all running Camel applications	Y	Y	Y
Start, suspend, resume, and delete Camel contexts	Y	Y	
Send messages	Y	Y	
Add endpoints	Y	Y	
View routes, route diagrams, and runtime statistics	Y	Y	Y
Start and stop routes	Y	Y	
Delete routes	Y	Y	
JMX			
Change attribute values	Y	Y	

Operation	admin	manager	viewer
Select and view attributes in a time-based chart	Y	Y	Y
View operations	Y	Y	Y
OSGI			
View bundles, features, packages, services, servers, framework, and configurations	Y	Y	Y
Add and delete bundles	Y	Y	
Add configurations	Y	Y	
Install and uninstall features	Y		
Runtime			
View system properties, metrics, and threads	Y	Y	Y
Logs			
View logs	Y	Y	Y

Additional resources

For more information on role-based access control, see Deploying into Apache Karaf.

CHAPTER 5. CUSTOMIZING THE FUSE CONSOLE BRANDING

You can customize the Fuse Console branding information, such as title, logo, and login page information, by using the Fuse Console branding plugin.

By default, the Fuse Console branding is defined in the **hawtconfig.json** that is located in the Fuse Console WAR file (**karaf-install-dir/system/io/hawt/hawtio-war/<version>/hawtio-war-<version>.war**). When you implement the Fuse Console branding plugin, you can override the default branding with your own custom branding.

Procedure

1. Download the branding plugin example from **https://github.com/hawtio/hawtio/tree/master/examples/branding-plugin** to a local directory of your choice.

2. In an editor of your choice, open the Fuse Console branding plugin's **src/main/webapp/plugin/brandingPlugin.js** file to customize the Fuse Console branding. You can change the values of the configuration properties listed in Table A.1, "Fuse Console Configuration Properties".

3. Save your changes.

4. In an editor of your choice, open the Fuse Console branding plugin's **pom.xml** file to its **<parent>** section:

```
<parent>
  <groupId>io.hawt</groupId>
  <artifactId>project</artifactId>
  <version>2.9-SNAPSHOT</version>
  <relativePath>../..</relativePath>
 </parent>
```

5. Edit the **<parent>** section as follows:

   a. Change the value of the **<version>** property to match the version of your Fuse on Karaf installation. For example, if your Fuse on Karaf installation directory name is **2.0.0.fuse-760015**, set the version to **2.0.0.fuse-760015**.

   b. Remove the **<relativePath>../..</relativePath>** line.
   For example:

```
<parent>
  <groupId>io.hawt</groupId>
  <artifactId>project</artifactId>
  <version> 2.0.0.fuse-760015</version>
   </parent>
```

6. In a Terminal window, build the branding-plugin project by running the following command:

```
mvn clean install
```

7. If Fuse is not already running, start it by running the following command:
Linux/Unix: bin/fuse

Windows: **bin\fuse.bat`**

8. At the Karaf CLI prompt, type the following command to install the Fuse Console branding plugin (where **<version>** is the version of your Fuse on Karaf installation):
   Linux/Unix: install -s mvn:io.hawt/branding-plugin/<version>/war

   Windows: install -s mvn:io.hawt\branding-plugin\<version>\war

9. In a web browser, open the Fuse Console by using the URL that the start command returned in Step 7 (the default URL is **http://localhost:8181/hawtio/**).

NOTE

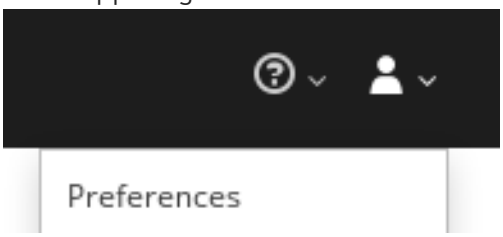
If you have already run the Fuse Console in a web browser, the branding is stored in the browser's local storage. To use new branding settings, you must clear the browser's local storage.

CHAPTER 6. 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**.

CHAPTER 7. DISABLING THE FUSE CONSOLE

You can disable the Fuse Console on Karaf so that it becomes inaccessible to all users without affecting any other component.

Procedure

1. To determine the hawtio-web bundle ID, use the following command to list the Fuse bundles that the Fuse Console uses:
   osgi:list | grep hawtio

2. To stop the bundle, use the **osgi:stop** command. For example, if the **hawtio :: Web console** bundle has an ID of 246, type this command:
   osgi:stop 246

The bundle goes into the resolved state and you can no longer access the Fuse Console.

Additional resources

For more information about managing bundles, see the "Lifecycle Management" chapter of Deploying into Apache Karaf.

CHAPTER 8. CONNECTING TO REMOTE FUSE APPLICATIONS

The Fuse Console uses Jolokia, an agent-based approach to Java Management Extensions (JMX) that requires extra software (an agent) installed on the client. By default, Red Hat Fuse includes a jolokia agent.

With standalone Fuse Console distributions, you can connect to remote integrations that already have a jolokia agent (https://jolokia.org/) running inside them. If the process that you want to connect to does not have a jolokia agent inside, refer to the jolokia documentation (http://jolokia.org/agent.html).

8.1. UNLOCKING THE FUSE CONSOLE

By default, Jolokia for Fuse 7 standalone on Apache Karaf is locked and the Fuse Console is not accessible remotely.

To unlock the Fuse Console for a hostname or IP address other than **locahost** or **127.0.0.1**, follow these steps:

1. Open the **$KARAF_HOME/etc/jolokia-access.xml** file in an editor.
2. Register the hostnames or IP addresses for the Fuse integrations that you want to access with the Fuse console by adding them to the **<cors>** section.
   For example, to access hostname **0.0.0.3** from the Fuse Console, add the

   ```
   *<allow-origin>http://0.0.0.3:*</allow-origin>*
   ```

   line as shown:

   ```
   <!--

     Cross-Origin Resource Sharing (CORS) restrictions

     By default, only CORS access within localhost is allowed for maximum security.

     You can add trusted hostnames in the <cors> section to unlock CORS access from them.

     -->

     <cors>

       <!-- Allow cross origin access only within localhost -->

       <allow-origin>http*://localhost:*</allow-origin>

       <allow-origin>http*://127.0.0.1:*</allow-origin>

       <allow-origin>http://0.0.0.3:*</allow-origin>

       <!-- Whitelist the hostname patterns as <allow-origin> -->

       <!--

       <allow-origin>http*://*.example.com</allow-origin>

       <allow-origin>http*://*.example.com:*</allow-origin>
   ```

```
    -->

    <!-- Check for the proper origin on the server side to protect against CSRF -->

    <strict-checking />

  </cors>
```

3. Save the file.

8.2. RESTRICTING REMOTE ACCESS

Optionally, you can restrict remote access to the Fuse Console for specific hosts and IP addresses.

You can grant overall access based on the IP address of an HTTP client. To specify these restrictions:

In the **jolokia-access.xml** file, add or edit a **<remote>** section that contains one or more **<host>** elements. For the **<host>** element, you can specify an IP address, a host name, or a netmask given in CIDR format (for example, **10.0.0.0/16** for all clients coming from the 10.0 network).

The following example allows access from localhost and all clients whose IP addresses start with **10.0**. For all other IP addresses, access is denied.

```
<remote>
  <host>localhost</host>
  <host>10.0.0.0/16</host>
</remote>
```

For more details, see the Jolokia security documentation (https://jolokia.org/reference/html/security.html).

8.3. ALLOWING CONNECTIONS TO REMOTE FUSE INSTANCES

The Fuse Console's proxy servlet uses whitelist host protection, with which by default the Fuse Console can only connect to localhost. If you want to connect the Fuse Console to other remote Fuse instances, you need to configure the whitelist as follows:

For Apache Karaf, make the following configuration changes in **etc/system.properties** file:

```
hawtio.proxyWhitelist = localhost, 127.0.0.1, myhost1, myhost2, myhost3
```

8.4. CONNECTING TO A REMOTE JOLOKIA AGENT

Before you begin, you need to know the connection details (host name, port, and path) of the remote Jolokia agent.

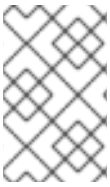
The default connection URL for the Jolokia agent for Fuse on Apache Karaf is **http://<host>:8181/hawtio/jolokia**.

As a system administrator, you can change this default.

Typically, the URL to remotely connect to a Jolokia agent is the URL to open the Fuse Console plus **/jolokia**. For example, if the URL to open the Fuse Console is **http://<host>:1234/hawtio**, then the URL to remotely connect to it would probably be **http://<host>:1234/hawtio/jolokia**.

To connect to a remote Jolokia instance so that you can examine its JVM:

1. Click the **Connect** tab.
2. Click the **Remote** tab, and then **Add connection**.
3. Type the **Name**, **Scheme** (HTTP or HTTPS), and the **hostname**.
4. Click **Test Connection**.
5. Click **Add**.



NOTE

The Fuse Console automatically probes the local network interfaces other than localhost and 127.0.0.1 and adds them to the whitelist. Hence, you do not need to manually register the local machine's addresses to the whitelist.

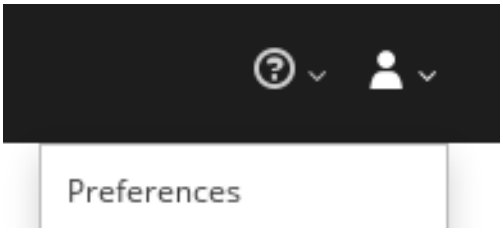
8.5. SETTING DATA MOVING PREFERENCES

You can change the following Jolokia preferences, for example, if you want to more frequently refresh data that displays in the Fuse Console. Note that increasing the frequency of data updates impacts networking traffic and increases the number of requests made to the server.

- **Update rate** - The period between polls to Jolokia to fetch JMX data (the default is 5 seconds).
- **Maximum depth** - The number of levels that Jolokia will marshal an object to JSON on the server side before returning (the default is 7).
- **Maximum collection size** - The maximum number of elements in an array that Jolokia marshals in a response (the default is 50,000).

To change the values of these settings:

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



2. Edit the options and then click **Close**.

8.6. VIEWING JVM RUNTIME INFORMATION

To view JVM runtime information, such as system properties, metrics, and threads, click the **Runtime** tab.

CHAPTER 9. 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.

9.1. 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.

9.2. 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**.

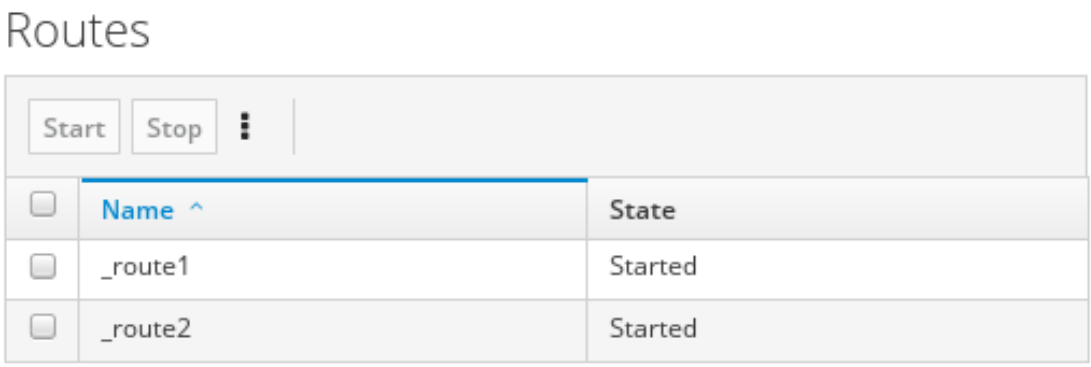
7. To view and execute JMX operations, such as adding or updating routes from XML or finding all Camel components available in the classpath, click **Operations**.

9.3. 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:

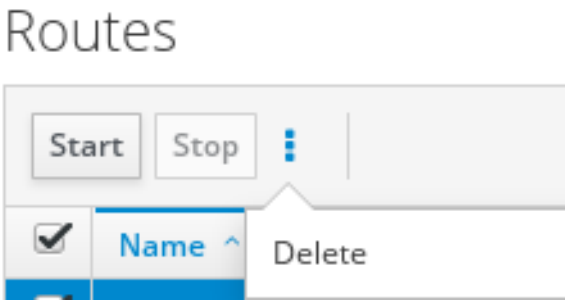


2. To start, stop, or delete one or more routes:

   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.



NOTE

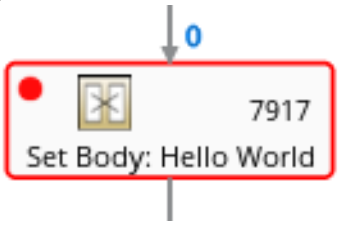
- When you delete a route, you remove it from the deployed application.
- You can also select a specific route in the tree view and then click the upper-right menu to start, stop, or delete it.

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.

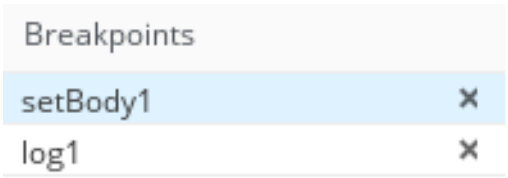
9.4. DEBUGGING A ROUTE

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

3. 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:



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

CHAPTER 10. 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 11. 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 12. VIEWING AND MANAGING YOUR OSGI ENVIRONMENT

For Apache Karaf standalone distributions, you can view and manage the Red Hat Fuse OSGi environment. You can view and manage container bundles, features, and configurations, as well as Java packages and OSGi services.

The **OSGi** tab contains a series of subtabs with options for each container component:

Bundles

List of installed bundles. You can install and uninstall bundles, start and stop bundles, and edit bundle properties. You can also filter the list and toggle between list and grid view.

Features

List of available features. You can install and uninstall features or feature repositories, and drill down to view feature details.

Packages

List of installed Java packages. You can view package versions and associated bundles.

Services

List of running services. You can view service IDs, associated bundles and object classes.

Declarative Services

List of declarative OSGi services. You can view the service state and drill down to view service details. You can also activate and deactivate services.

Server

Detailed information about the local or remote host in read-only mode.

Framework

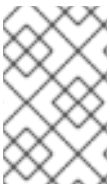
Configuration options for the container OSGi framework. You can set the framework start level and the initial bundle start level.

Configuration

List of configuration objects. You can view the state of each object and drill down to view or edit object details. You can also create a new configuration object.

CHAPTER 13. VIEWING DIAGNOSTICS

Use the **Diagnostics** tab to view diagnostic information about the JVM via the JVM DiagnosticCommand and HotspotDiangostic 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 14. 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 15. VIEWING LOG ENTRIES

You can view log entries for Red Hat Fuse in the **Logs** tab.

Prerequisite

The **Logs** tab is available when the Java application includes the Log MBean.

Procedure

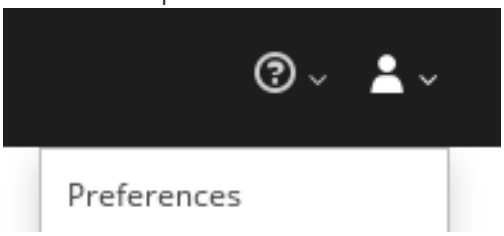
1. To view a list of the log entries, click the **Log Entries** tab.
   By default, the list shows log entries in ascending order.

   You can drill down to each log entry to view detailed information about the log entry.

2. To filter the list of logs to show specific log types, click the **Action Bar**. You can filter the log entries section according to a text string or the logging level.

3. To change the Fuse Console default settings:

   a. In the upper right corner of the Fuse Console, click the user icon and then click **Preferences** from the drop-down menu.

   b. To change the default sorting order, select **Server Logs** and then click the log entry link to drill down to details about the log entry, such as the bundle name, thread, and the full message text.

   c. Optionally, you can customize these settings for storing log messages:

      - The number of log statements to keep in the Fuse Console (the default is 100).
      - The global log level: **INFO** (the default), OFF, ERROR, WARN, and DEBUG.
      - The child-level messages to include, such as **hawtio-oauth** and **hawtio-core-utils**.

   d. To reset the Fuse Console Logs settings to the default values, click **Reset → Reset settings**.

CHAPTER 16. ENABLING PROMETHEUS METRICS

Prometheus is an open-source systems and service monitoring and alerting toolkit that you can use to monitor services deployed in a standalone Apache Karaf container. Prometheus collects and stores metrics from configured services at given intervals, evaluates rule expressions, displays the results, and can trigger alerts if a specified condition becomes true.

IMPORTANT

Red Hat support for Prometheus is limited to the setup and configuration recommendations provided in Red Hat product documentation.

Prometheus uses “exporters” that are installed and configured on the clients to expose an endpoint to Prometheus format. This endpoint is an HTTP interface that provides a list of metrics and the current values of the metrics. Prometheus periodically scrapes each target-defined endpoint and writes the collected data to its database. Prometheus gathers data over an extended time, rather than just for the currently running session. Prometheus stores the data so that you can graphically visualize and run queries on the data.

16.1. ENABLING THE EXPORT OF METRICS FROM A STANDALONE APACHE KARAF CONTAINER

Prometheus uses a configuration file (**https://raw.githubusercontent.com/jboss-fuse/application-templates/master/prometheus/prometheus-config.yml**) which includes metrics exposed by Camel.

NOTE

The metrics that you can identify are limited to those supplied in JMX.

You must deploy a Fuse application in order for Apache Camel metrics to be generated.

Procedure

To enable the export of Prometheus metrics from a standalone Apache Karaf container using the command line:

1. Open a command prompt and make sure you are in the **etc**/ directory of your Apache Karaf installation.
2. Type the following command to create the Prometheus configuration file from the example file in the **etc**/ directory:

```
cp prometheus-config.yml-example prometheus-config.yml
```

3. The exporter is only available when you use the **fuse** or **fuse.bat** command to start Fuse: run **bin/fuse** on Linux/Unix or **bin\fuse.bat** on Windows.
4. When Fuse has restarted, you can view the exposed metrics by opening a web browser at: **http://localhost:9779**

NOTE

Optionally you can change the default values of the **KARAF_PROMETHEUS_PORT** and **KARAF_PROMETHEUS_CONFIG** configuration variables from the command line.

16.2. CONFIGURING THE PROMETHEUS SERVER TO SCRAPE EXPOSED METRICS FROM THE APACHE KARAF CONTAINER

To enable the Prometheus server to scrape metrics from the Apache Karaf container, the endpoint which exposes the metrics must be added to the **targets** property of the Prometheus configuration file.

Procedure

1. Navigate to the /**prometheus.yml** configuration file in the Prometheus installation directory.
2. Add the Apache Karaf endpoint to scrape:

```
scrape_configs:
  - job_name: 'prometheus'

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.
    static_configs:
    - targets: ['localhost:9779']
```

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	*Empty value*	Product information that shows on the About page of the Fuse Console.	Optional
	additionalInfo	*Empty value*	Any additional information that shows on the About page of the Fuse Console.	Optional
	copyright	*Empty value*	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	*Empty value*	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	*none*	[ ]	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