Red Hat Fuse 7.12

Managing Fuse on Springboot Standalone

Manage Fuse applications with the Fuse Console
Manage Fuse applications with the Fuse Console
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

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREFACE</td>
<td>3</td>
</tr>
<tr>
<td>MAKING OPEN SOURCE MORE INCLUSIVE</td>
<td>4</td>
</tr>
<tr>
<td>CHAPTER 1. ABOUT THE FUSE CONSOLE</td>
<td>5</td>
</tr>
<tr>
<td>CHAPTER 2. ACCESSING THE FUSE CONSOLE FOR SPRING BOOT 2.X</td>
<td>6</td>
</tr>
<tr>
<td>CHAPTER 3. CUSTOMIZING THE FUSE CONSOLE BRANDING</td>
<td>8</td>
</tr>
<tr>
<td>CHAPTER 4. SECURING THE FUSE CONSOLE</td>
<td>9</td>
</tr>
<tr>
<td>CHAPTER 5. ENSURING THAT DATA DISPLAYS CORRECTLY IN THE FUSE CONSOLE</td>
<td>11</td>
</tr>
<tr>
<td>CHAPTER 6. CONNECTING TO REMOTE FUSE APPLICATIONS</td>
<td>12</td>
</tr>
<tr>
<td>6.1. CONNECTING TO A REMOTE JOLOKIA AGENT</td>
<td>12</td>
</tr>
<tr>
<td>6.2. SETTING DATA MOVING PREFERENCES</td>
<td>12</td>
</tr>
<tr>
<td>6.3. VIEWING JVM RUNTIME INFORMATION</td>
<td>13</td>
</tr>
<tr>
<td>CHAPTER 7. VIEWING AND MANAGING APACHE CAMEL APPLICATIONS</td>
<td>14</td>
</tr>
<tr>
<td>7.1. STARTING, SUSPENDING, OR DELETING A CONTEXT</td>
<td>14</td>
</tr>
<tr>
<td>7.2. VIEWING CAMEL APPLICATION DETAILS</td>
<td>14</td>
</tr>
<tr>
<td>7.3. VIEWING A LIST OF THE CAMEL ROUTES AND INTERACTING WITH THEM</td>
<td>15</td>
</tr>
<tr>
<td>7.4. DEBUGGING A ROUTE</td>
<td>16</td>
</tr>
<tr>
<td>CHAPTER 8. VIEWING AND MANAGING JMX DOMAINS AND MBEANS</td>
<td>18</td>
</tr>
<tr>
<td>CHAPTER 9. VIEWING AND MANAGING QUARTZ SCHEDULES</td>
<td>19</td>
</tr>
<tr>
<td>CHAPTER 10. VIEWING DIAGNOSTICS</td>
<td>20</td>
</tr>
<tr>
<td>CHAPTER 11. VIEWING THREADS</td>
<td>21</td>
</tr>
<tr>
<td>APPENDIX A. FUSE CONSOLE CONFIGURATION PROPERTIES</td>
<td>22</td>
</tr>
</tbody>
</table>
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 Prometheus documentation.

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. ACCESSING THE FUSE CONSOLE FOR SPRING BOOT 2.X

You can access the Fuse Console for a standalone Fuse Spring Boot 2.x distribution.

Procedure

1. Add the following dependency to your Fuse application’s pom.xml file:

   <dependency>
   <groupId>io.hawt</groupId>
   <artifactId>hawtio-springboot</artifactId>
   </dependency>

   Note that you do not need to specify the exact version because it is provided by the Maven BOM.

2. Edit the src/main/resources/application.properties file:
   a. Set the following properties:
      - management.endpoints.web.exposure.include=hawtio,jolokia
      - hawtio.authenticationEnabled=false
      - management.endpoint.hawtio.enabled=true
      - management.endpoint.jolokia.enabled=true
   b. Optionally, set the management.endpoints.web.base-path property.
      By default for Spring Boot 2.x, the Fuse Console’s URL includes the context path (/actuator) of the management endpoints. For example:


      To change this default URL, for example to specify http://localhost:10001/hawtio, set the management.endpoints.web.base-path property as shown here:

      management.endpoints.web.base-path=/

      Your application.properties settings should look similar to the following example:

      # ports
      server.port=8080
      management.server.port=10001
      # enable management endpoints for healthchecks and hawtio
      management.endpoints.enabled-by-default = false
      management.endpoint.hawtio.enabled = true
      management.endpoint.jolokia.enabled = true
management.endpoints.health.enabled = true
management.health.defaults.enabled=false
camel.health.enabled=false
camel.health.indicator.enabled=true
management.endpoints.web.exposure.include=hawtio,jolokia
hawtio.authenticationEnabled=false

# change the URL so that it does not include the actuator folder
management.endpoints.web.base-path=/

NOTE

By default, authentication for the Fuse Console on Spring Boot is disabled. Optionally, you can enable authentication by writing code specific to your Fuse Console distribution. Here is an example that you can use for guidance:

https://github.com/hawtio/hawtio/tree/master/examples/springboot-authentication

3. Run the Fuse application:

   mvn spring-boot:run

4. To determine the port number for the Fuse Console URL, obtain the management.server.port value by looking at the value set in the src/main/resources/application.properties file. For example:

   management.server.port = 10001

5. To open the Fuse Console in a browser, use the following URL syntax where nnnnn is the value of the management.server.port property:

   http://localhost:nnnn/actuator/hawtio

   For example, if the management.server.port property value is 10001 and you have not set the management.endpoints.web.base-path property then the URL is:

CHAPTER 3. CUSTOMIZING THE FUSE CONSOLE BRANDING

You can customize the Fuse Console branding information, such as title, logo, and login page information, by adding a hawtconfig.json file into your Fuse on Spring Boot standalone application.

Procedure

1. Create a JSON file named hawtconfig.json in your local Fuse on Spring Boot standalone application’s src/main/webapp directory.

2. Open the src/main/webapp/hawtconfig.json in an editor of your choice, and then add the following content:

```json
{
   "branding": {
      "appName": "Red Hat Fuse Console",
      "appLogoUrl": "img/Logo-Red_Hat-Fuse-A-Reverse-RGB.png",
      "companyLogoUrl": "img/Logo-RedHat-A-Reverse-RGB.png"
   },
   "login": {
      "description": "",
      "links": []
   },
   "about": {
      "title": "Red Hat Fuse Console",
      "productInfo": [],
      "additionalInfo": "",
      "copyright": "",
      "imgSrc": "img/Logo-RedHat-A-Reverse-RGB.png"
   },
   "disabledRoutes": [
      "/camel/source",
      "/diagnostics",
      "/jvm/discover",
      "/jvm/local"
   ]
}
```

3. Change the values of the configuration properties listed in Table A.1, “Fuse Console Configuration Properties”.

4. Save your changes.

5. Run Fuse on Spring Boot by using the following command:

```
mvn spring-boot:run
```

6. In a web browser, open the Fuse Console by using this URL:

www://localhost:10001/actuator/hawtio/index.html

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 4. SECURING THE FUSE CONSOLE

To secure the Fuse Console on Spring Boot:

- **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.

**Procedure**

1. Set the `hawtio.http.strictTransportSecurity` and `hawtio.http.publicKeyPins` properties as shown in the following example:

   ```java
   public static void main(String[] args) {
     System.setProperty("hawtio.http.strictTransportSecurity", "max-age=31536000; includeSubDomains; preload");
     System.setProperty("hawtio.http.publicKeyPins", "pin-sha256=cUPcTAZWKaASuYWhheDttWpY3oBAkE3h2+soZS7sWs"; max-age=5184000; includeSubDomains");
     SpringApplication.run(YourSpringBootApplication.class, args);
   }
   ```

2. (For deploying on AWS only) To disable the Fuse Console’s proxy servlet, set the `hawtio.disableProxy` property as shown in the following example:

   ```java
   public static void main(String[] args) {
     System.setProperty("hawtio.disableProxy", "true");
   }
   ```

**Additional resources**

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 5. 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 6. 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).

Procedure

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 hawtio.proxyWhitelist system property in the main() method of your Spring Boot application:

```java
System.setProperty("hawtio.proxyWhitelist", "localhost, 127.0.0.1, myhost1, myhost2, myhost3");
```

6.1. 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 URLs for the Jolokia agent on Spring Boot is http://<host>:8080/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.

6.2. 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**.

6.3. VIEWING JVM RUNTIME INFORMATION

To view JVM runtime information, such as system properties, metrics, and threads, click the **Runtime** tab.
CHAPTER 7. 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.

7.1. 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.

7.2. 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**.

### 7.3. 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 Table]

<table>
<thead>
<tr>
<th>Name</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>_route1</td>
<td>Started</td>
</tr>
<tr>
<td>_route2</td>
<td>Started</td>
</tr>
</tbody>
</table>

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.

7.4. 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:

![Breakpoint Added](image)

The node is added to the list of breakpoints:

<table>
<thead>
<tr>
<th>Breakpoints</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>setBody1</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>log1</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

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 8. 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 9. 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 10. VIEWING DIAGNOSTICS

Use the Diagnostics tab to view diagnostic information about the JVM via the JVM DiagnosticCommand and HotspotDiagnostics 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://openjdk.java.net/groups/hotspot/docs/RuntimeOverview.html
CHAPTER 11. 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.
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>
<thead>
<tr>
<th>Section</th>
<th>Property Name</th>
<th>Default Value</th>
<th>Description</th>
<th>Required?</th>
</tr>
</thead>
<tbody>
<tr>
<td>About</td>
<td>Title</td>
<td>Red Hat Fuse Management Console</td>
<td>The title that shows on the About page of the Fuse Console.</td>
<td>Required</td>
</tr>
<tr>
<td></td>
<td>productInfo</td>
<td>Empty value</td>
<td>Product information that shows on the About page of the Fuse Console.</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>additionalInfo</td>
<td>Empty value</td>
<td>Any additional information that shows on the About page of the Fuse Console.</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>copyright</td>
<td>Empty value</td>
<td>Copyright information that shows on the About page of the Fuse Console.</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>imgSrc</td>
<td>img/Logo-RedHat-A-Reverse-RGB.png</td>
<td>The image that appears on the About page of the Fuse Console.</td>
<td>Required</td>
</tr>
<tr>
<td>branding</td>
<td>appName</td>
<td>Red Hat Fuse Management Console</td>
<td>The name for your application. This name displays in the title bar of the Fuse Console.</td>
<td>Required</td>
</tr>
<tr>
<td>Section</td>
<td>Property Name</td>
<td>Default Value</td>
<td>Description</td>
<td>Required?</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------</td>
<td>-----------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td>appLogoUrl</td>
<td><img src="img/Logo-Red_Hat-Fuse-A-Reverse-RGB.png" alt="img/Logo-Red_Hat-Fuse-A-Reverse-RGB.png" /></td>
<td>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 Hawkio status URL or an absolute URL.</td>
<td>Required</td>
</tr>
<tr>
<td></td>
<td>Css</td>
<td><img src="img/Logo-RedHat-A-Reverse-RGB.png" alt="img/Logo-RedHat-A-Reverse-RGB.png" /></td>
<td>The URL of an external CSS stylesheet, that can be used to style the application. It can be a path, relative to the Hawkio status URL, or it can be an absolute URL.</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>Favicon</td>
<td><img src="img/Logo-Red_Hat-Fuse-A-Reverse-RGB.png" alt="img/Logo-Red_Hat-Fuse-A-Reverse-RGB.png" /></td>
<td>The URL of the favicon, that usually displays in the Web browser tab. It can be a path, relative to the Hawkio status URL, or it can be an absolute URL.</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>login</td>
<td><strong>Empty value</strong></td>
<td>Descriptive text that displays on the Fuse Console Login page (for example, <a href="http://localhost:8181/hawtio">http://localhost:8181/hawtio</a>).</td>
<td>Optional</td>
</tr>
<tr>
<td>Section</td>
<td>Property Name</td>
<td>Default Value</td>
<td>Description</td>
<td>Required?</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td>links</td>
<td>[ ]</td>
<td>Specify an array of &quot;url&quot; and &quot;text&quot; pairs to provide additional links to pages where the user can get more information or help.</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>disabledRoutes</td>
<td>none</td>
<td>[ ]</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Disables specific paths (i.e., plugins) on the console. Do not change this section. Any change is not supported for distributions other than OpenShift.</td>
<td></td>
</tr>
</tbody>
</table>