Red Hat Decision Manager 7.0

Packaging and deploying a decision service

Last Updated: 2018-11-20

Red Hat Decision Manager 7.0 Packaging and deploying a decision service

Red Hat Customer Content Services
brms-docs@redhat.com

Legal Notice

Abstract

This document describes packaging and deploying a decision service for Red Hat Decision Manager 7.0

Table of Contents

PREFACE

After developing a decision service using Decision Central, you need to deploy it in order to use it. You can use several ways to package a decision service and deploy it onto Decision Servers.

CHAPTER 1. OVERVIEW

You can develop a decision service in Red Hat Decision Manager using Decision Central. To run a decision service, you need to deploy it onto a Decision Server.

1.1. PACKAGING DECISION SERVICES

A decision service is always packaged into a *KJAR file*. This file is a version of a Java JAR file. A Decision Server runs the service from its KJAR file. However, you never need to work with KJAR files directly.

Instead, Red Hat Decision Manager stores KJAR files in *Maven repositories*. Maven is a comprehensive system for automating Java package lifecycle and dependency management. Among other features, Maven supports a repository system where JAR files (including KJAR files) can be uploaded and downloaded.

In Maven, a KJAR file is not identified by its name, but instead using the following three values: *group ID*, *artifact ID*, *version*. Normally, the group ID is the identifier for a project and the artifact identified a particular artifact (for example, a decision service) within the project. The version value must be unique for every new version that might need to be deployed.

To identify an artifact (including a KJAR file) in Maven, you need all three of these variables. This set of information is often named GAV (for Group, Artifact, Version).

When developing a decision service in Decision Central, you can set the GAV in the project settings screen.

1.2. DEPLOYING DECISION SERVICES

The deployment process can vary based on the needs of your infrastructure.

In a simple deployment of Red Hat Decision Manager, there is one Decision Central and one Decision Server. You can use the Decision Central to develop your decision services and also to manage the Decision Server. You can simply build your project in Decision Central and it is automatically deployed onto the Decision Server.

To enable automatic deployment, Decision Central includes a built-in Maven repository. You can use Decision Central to manage the Decision Server, deploying, removing, starting, and stopping any of the decision services and their versions that you have built.

You can also connect several Decision Servers to the same Decision Central. You can group them into different *server templates*. Servers belonging to the same server template run the same decision services. But you can deploy different decision services or different versions of decision services on different templates.

For example, you could have test servers in the **`Test`** template and production servers in a **`Production`** template. When developing a decision service, deploy it on the **`Test`** template. Then, when a version of the service is sufficiently tested, you can deploy it on the **`Production`** template.

In this case, to keep developing the service, change the version in the project settings. Then the new version and the old version are seen as different artifacts in the built-in Maven repository. You can deploy the new version on the **`Test`** template and keep running the old version on the **`Production`** template.

This deployment process is simple but has significant limitations. Notably, there is not enough access control; a developer can deploy a project directly into production.

If you require a proper integration process, you can use an external Maven repository (for example, Nexus). You can configure Decision Central to push decision service files into the external repository instead of the built-in repository.

You can still use Decision Central to deploy decision services on a test Decision Server. However, in this process, other Decision Servers (for example, staging and production) are not connected to Decision Central. Instead, they retrieve the decision service KJAR files and any necessary dependencies from your Maven repository. You can progress the KJAR versions through your repository as necessary, in line with the integration process that you want to implement.

If you deploy a Decision Server on OpenShift, you can configure it to load and start a decision service from a Maven repository automatically. If you deploy a Decision Server on premises, you can configure access to a Maven repository when setting up the server; then you can use the REST API of the server to load and start a decision service from the repository. A Maven project and a Java client library for automating the API calls are available.

CHAPTER 2. SETTING THE GROUP ID, ARTIFACT ID, AND VERSION (GAV) VALUES IN A DECISION SERVICE

When developing a decision service in Decision Central, you can set the GAV in the project settings screen.

These values are the identifier for the decision service in a Maven repository. In any deployment method the decision service is identitied by the GAV. This means that if you build a new version without changing the GAV, it automatically displaces the old version in a Maven repository and on any Decision Servers. However, if you change any value in the GAV (usually the version), the result is seen as a different artifact and can exist alongside the old version.

Procedure

1. In the main menu, click **Projects**.
2. Click the project name.
   Alternatively, click the **Add Project** button to create a new project.
3. Click the **Settings** button.
4. Modify the **Group ID**, **Artifact ID**, or **Version** fields as necessary. If you have deployed the decision service and are developing a new version, usually you need to increase the version number.
5. Click the **Save** button.

CHAPTER 3. USING DECISION CENTRAL TO DEPLOY AND MANAGE DECISION SERVICES

You can use Decision Central to build and deploy decision services. If you connect multiple Decision Servers to the Decision Central, you can use the web UI to deploy and manage decision services on all the servers.

You can group Decision Servers into different *server templates*. The same decision services are automatically run servers belonging to the same server template. But you can deploy different decision services or different versions of decision services on different templates.

You can configure the template name for each Decision Server in the configuration file for that server.

IMPORTANT

You can not move a Decision Server into a different template using Decision Central. You must change the configuration file of the server to change the server template name for it.

3.1. BUILDING AND DEPLOYING A DECISION SERVICE IN DECISION CENTRAL

You can build a decision service in Decision Central and automatically deploy it on a Decision Server.

Procedure

1. In the main menu, click **Projects**.
2. Click the project name.
3. Click the **Build & Deploy** button.
4. If only one Decision Server is connected to the Decision Central, or if all connected Decision Servers are in the same server template, the project is automatically built and deployed. Otherwise, Decision Central displays a dialog. In this dialog, set the following values:
   - **Container ID** and **Container Alias** identify the container running the service within the Decision Server. You normally do not need to change these settings.
   - **Server template** : select the server template for deploying this project. You can later deploy it to other templates without rebuilding it.
   - **Start container?** : if you clear this box, the service is deployed onto the server but not started. Normally, leave this box checked.

3.2. MANAGING DEPLOYMENT OF DECISION SERVICES IN DECISION CENTRAL

You can use Decision Central to manage deployment of decision services on one or more Decision Servers. Each Decision Server must be configured to connect to the Decision Central.

A decision service always runs in a *container* on a Decision Server. You can create containers and start them on Decision Servers; when you build and deploy a project, the container is created automatically. A container is always created in a particular server template.

You can also start, stop, and delete containers.

3.2.1. Creating a container

You can create a container from a decision service that was previously built in Decision Central.

Procedure

1. In the main menu, click **Execution servers**.
2. Under **Server templates**, select a template.
3. Under **KIE Containers**, click **Add Container**.
4. Decision Central displays a dialog. In the table in the dialog, select a GAV and click the **Select** button next to the GAV.
5. If you want to start the service immediately, select the **Start container?** box.
6. Click the **Finish** button.

Result

A container is created for the service. This container is placed on the Decision Servers that are configured for this server template. If you have selected **Start container?**, the service is started.

3.2.2. Managing containers

When a container is started, the service in the container is available for use. You can start, stop, or remove containers with decision services as necessary.

Procedure

1. In the main menu, click **Execution servers**.
2. Under **Server templates**, select a template.
3. Under **KIE Containers**, select a container.
4. Click the **Start**, **Stop**, or **Remove** button.

IMPORTANT

You cannot remove a container that is running. To remove a running container, stop it and then remove it.

3.3. CONFIGURING A DECISION SERVER TO CONNECT TO DECISION CENTRAL

You can configure a Decision Server to connect to an existing Decision Central. You can also configure the name of the server template. The server becomes a part of that server template; any decision services deployed on the template are deployed on this server.

IMPORTANT

If you are deploying the Decision Server on OpenShift, see *Deploying Red Hat Decision Manager on Red Hat OpenShift Container Platform* for instructions about configuring it to connect to a Decision Central.

Procedure

1. Install the Decision Server. See *Installing Red Hat Decision Manager on premise* for instructions.

2. Create a **`settings.xml`** file for connecting to the Maven repository that the Decision Central uses. If you have not changed the configuration of Decision Central, it uses its own built-in Maven repository. Use the following contents for **`settings.xml`**:

```
<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0
http://maven.apache.org/xsd/settings-1.0.0.xsd">

  <servers>
    <server>
      <id>remote-repo</id>
      <username>user</username>
      <password>pwd</password>
    </server>
  </servers>
  <profiles>
    <profile>
      <id>additional-maven-repos</id>
      <repositories>
        <repository>
          <id>remote-repo</id>
          <url>http://centralhost:centralport/decision-
central/maven2/</url>
        </repository>
      </repositories>
    </profile>
  </profiles>
  <activeProfiles>
    <activeProfile>additional-maven-repos</activeProfile>
  </activeProfiles>
</settings>
```

Replace the following values:

- **user**: the user name for a user that can log into Decision Central
- **pwd**: the password for the user that can log into Decision Central
- **centralhost**: the host name for the Decision Central
- **centralport**: the port for the Decision Central
  If the Decision Central is deployed on OpenShift, remove **`decision-central/`** from the URL.

Save the file in a known location, for example, **`/opt/maven/settings.xml`** .

3. In the **`<eap_home>/standalone/configuration/standalone.xml`** file, under the **`<system-properties>`** tag, set the following properties:

   - **org.kie.server.controller.user**: user name of an account that can log on to the Decision Central.
   - **org.kie.server.controller.pwd**: password for the account.
   - **org.kie.server.controller**: the URL for connecting to the API of the Decision Central. Normally, it is **`http://<centralhost>:<centralport>/decision-central/rest/controller`** , where *<centralhost>* and *<centralport>* are the host name and port for the Decision Central. If the Decision Central is deployed on OpenShift, remove **`decision-central/`** from the URL.
   - **org.kie.server.location**: the URL for connecting to the API of the Decision Server. Normally, it is **`http://<serverhost>:<serverport>/kie-server/services/rest/server`** , where *<serverhost>* and *<serverport>* are the host name and port for the Decision Server.
   - **org.kie.server.id**: the name of a server template. If this server template does not exist on the Decision Central, it is created automatically when the Decision Server connects to the Decision Central.
   - **kie.maven.settings.custom** : the full path to the **`settings.xml`** file for connecting to the Maven repository.
     For example:

     ```
     <property name="org.kie.server.controller.user"
     value="central_user"/>
     <property name="org.kie.server.controller.password"
     value="central_password"/>
     <property name="org.kie.server.controller"
     value="http://central.example.com:8080/decision-
     central/rest/controller"/>
     <property name="org.kie.server.location"
     value="http://kieserver.example.com:8080/kie-
     server/services/rest/server"/>
     <property name="org.kie.server.id" value="production-servers"/>
     <property name="kie.maven.settings.custom"
     value="/opt/maven/settings.xml"/>
     ```

4. Start or restart the Decision Server.

CHAPTER 4. USING AN EXTERNAL MAVEN REPOSITORY TO DEPLOY A DECISION SERVICE

You can use an external Maven repository (for example, Nexus) to set up an integration process. You can configure Decision Central to push decision service files into the external repository instead of the built-in repository.

You can still use Decision Central to deploy decision services on a test Decision Server. However, in this process, other Decision Server (for example, staging and production) are not connected to Decision Central. Instead, they retrieve the decision service KJAR files and any necessary dependencies from your Maven repository. You can progress the KJAR versions through your repository as necessary, in line with the integration process that you want to implement.

If you deploy a Decision Server on OpenShift, you can configure it to load and start a decision service from a Maven repository automatically.

If you deploy a Decision Server on premises, you can configure access to a Maven repository when setting up the server. Then you can use the REST API of the server to load and start a decision service from the repository. A Maven project and a Java client library for automating the API calls are available.

4.1. CONFIGURING DECISION CENTRAL TO USE AN EXTERNAL MAVEN REPOSITORY

You can configure Decision Central to use an external Maven repository instead of its built-in repository. In this case, Decision Central pushes all the built KJAR files into this repository. You can progress these files through the repository as necessary to implement your integration process.

> **IMPORTANT**
>
> Using an external Maven repository in Decision Central deployed on OpenShift is not supported.

Procedure

1. Install the Decision Central. See *Installing Red Hat Decision Manager on premise*for instructions.

2. Create a Maven **`settings.xml`** file with connetion and access details for your external repository. For details about the **`settings.xml`** file, see Maven documentation: https://maven.apache.org/settings.html .

3. In the **`<eap_home>/standalone/configuration/standalone.xml`** file, under the **`<system-properties>`** tag, set the **`kie.maven.settings.custom`** property to the full pathname of the **`settings.xml`** file, for example:

   ```
   <property name="kie.maven.settings.custom" value="/opt/custom-
   config/settings.xml"/>
   ```

4. Start or restart the Decision Central.

4.2. CONFIGURING A DECISION SERVER TO RUN A DECISION SERVICE

You can configure a Decision Server to load and run a decision service automatically without using a Decision Central. The service must be present in a Maven repository. You need access information for the repository and the GAV values for the service. You can also stop and delete the service from this server at a later time. You can start several services on one Decision Server at the same time.

IMPORTANT

Do not use this procedure for a Decision Server deployed on OpenShift. See *Deploying Red Hat Decision Manager on Red Hat OpenShift Container Platform* for instructions about configuring a a Decision Server to load and run a decision service from a Maven repository.

4.2.1. Configuring a Decision Server to use a Maven repository

Before you can load and start a decision service on a Decision Server, you need to configure the server to use your Maven repository.

Procedure

1. Install the Decision Server. See *Installing Red Hat Decision Manager on premise*for instructions.
2. Create a Maven **settings.xml** file with connetion and access details for your external repository. For details about the **settings.xml** file, see Maven documentation: https://maven.apache.org/settings.html .
3. In the **<eap_home>/standalone/configuration/standalone.xml** file, under the **<system-properties>** tag, set the **kie.maven.settings.custom** property to the full pathname of the **settings.xml** file, for example:

```
<property name="kie.maven.settings.custom" value="/opt/custom-
config/settings.xml"/>
```

4. Start or restart the Decision Server.

4.2.2. Loading and starting a service on the Decision Server

If you have configured a Decision Server to use a Maven repository, you can use an API call to load a service. The service is started automatically.

Procedure

1. To load a service into a container in the Decision Server and start it, run the following command to send an API request:

```
$ curl --user "<username>:<password>" -H "Content-Type:
application/json" -X PUT -d '{"container-id" : "
<containerID>","release-id" : {"group-id" : "<groupID>","artifact-
id" : "<artifactID>","version" : "<version>"}}' http://<serverhost>:
<serverport>/kie-
server/services/rest/server/containers/<containerID>
```

where:

- **<username>** is the user name for an account that can log onto the Decision Server and administer it.

- **<password>** is the password for the account.

- **<containerID>** is the identifier for the container. You can use any random identifier but it must be the same in both places in the command (the URL and the data).

- **<groupID>**, **<artifactID>**, **version** are GAV values.

- **<serverhost>** is the host name for the KIE server, or **`localhost`** if you are running the command on the same host as the KIE server.

- **<serverport>** is the port number for the KIE server
  For example:

  ```
  curl --user "rhdmAdmin:password@1" -H "Content-Type:
  application/json" -X PUT -d '{"container-id" : "kie1","release-
  id" : {"group-id" : "org.kie.server.testing","artifact-id" :
  "container-crud-tests1","version" : "2.1.0.GA"}}'
  http://localhost:39043/kie-
  server/services/rest/server/containers/kie1
  ```

4.2.3. Stopping and removing a service from the Decision Server

You can use an API call to stop and remove a service from the Decision Server. You need the container identifier that was configured when loading the service.

Procedure

1. To stop and remove a container with a service on the Decision Server, run the following command to send an API request:

   ```
   $ curl --user "<username>:<password>" -X DELETE http://<serverhost>:
   <serverport>/kie-
   server/services/rest/server/containers/<containerID>
   ```

   where:

   - **<username>** is the user name for an account that can log onto the Decision Server and administer it.

   - **<password>** is the password for the account.

   - **<containerID>** is the identifier for the container.

   - **<serverhost>** is the host name for the KIE server, or **`localhost`** if you are running the command on the same host as the KIE server.

   - **<serverport>** is the port number for the KIE server
     For example:

     ```
     curl --user "rhdmAdmin:password@1" -X DELETE
     http://localhost:39043/kie-
     server/services/rest/server/containers/kie1
     ```

APPENDIX A. VERSIONING INFORMATION

Documentation last updated on: Monday, October 1, 2018.