Red Hat Decision Manager 7.1

Managing and monitoring Decision Server

Last Updated: 2018-10-12

Red Hat Decision Manager 7.1 Managing and monitoring Decision Server

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

Legal Notice

Abstract

This document explains how install, configure, and performance tune Red Hat Decision Manager 7.1

Table of Contents

PREFACE

As a systems administrator, you can install, configure, and upgrade Red Hat Decision Manager for production environments, quickly and easily troubleshoot system failures, and ensure that systems are running optimally.

Prerequisites

- Installed Red Hat JBoss Enterprise Application Platform 7.1.0. See *Red Hat JBoss EAP 7.1.0 Installation Guide*.
- Installed Red Hat Decision Manager. For more information, see the *Planning a Red Hat Decision Manager installation*.
- Red Hat Decision Manager is running and you can log in to Decision Central with the **`admin`** role. For more information, see the *Planning a Red Hat Decision Manager installation*.

CHAPTER 1. RED HAT DECISION MANAGER COMPONENTS

Red Hat Decision Manager is made up of Decision Central and Decision Server.

- Decision Central is the graphical user interface where you create and manage business rules. You can install Decision Central in a Red Hat JBoss EAP instance or on the Red Hat OpenShift Container Platform (OpenShift).
  Decision Central is also available as a standalone JAR file. You can use the Decision Central standalone JAR file to run Decision Central without needing to deploy it to an application server.

- Decision Server is the server where processes, rules, and other artifacts are executed. It is used to instantiate and execute processes and rules and solve planning problems. You can install Decision Server in a Red Hat JBoss EAP instance, on OpenShift, in an Oracle WebLogic server instance, or an IBM WebSphere Application Server instance.
  You can configure Decision Server to run in managed or unmanaged mode. If Decision Server is unmanaged, you must manually create and maintain KIE containers (deployment units). A KIE container is a specific version of a project. If Decision Server is managed, the Decision Manager controller manages the Decision Server configuration and you interact with the Decision Manager controller to create and maintain KIE containers.

  The Decision Manager controller is integrated with Decision Central. If you install Decision Central, use the Execution Server page to create and maintain KIE containers. However, if you do not install Decision Central, you can install the headless Decision Manager controller and use the REST API or the Decision Server Java Client API to interact with it.

Red Hat Business Optimizer is integrated in Decision Central and Decision Server. It is a lightweight, embeddable planning engine that optimizes planning problems. Red Hat Business Optimizer helps Java programmers solve planning problems efficiently, and it combines optimization heuristics and metaheuristics with efficient score calculations.

CHAPTER 2. SYSTEM INTEGRATION WITH MAVEN

Red Hat Decision Manager is designed to be used with Red Hat JBoss Middleware Maven Repository and Maven Central repository as dependency sources. Ensure that both the dependencies are available for projects builds.

Ensure that your project depends on specific versions of an artifact. **LATEST** or **RELEASE** are commonly used to specify and manage dependency versions in your application.

- **LATEST** refers to the latest deployed (snapshot) version of an artifact.
- **RELEASE** refers to the last non-snapshot version release in the repository.

By using **LATEST** or **RELEASE**, you do not have to update version numbers when a new release of a third-party library is released, however, you lose control over your build being affected by a software release.

2.1. PREEMPTIVE AUTHENTICATION FOR LOCAL PROJECTS

If your environment does not have access to the internet, set up an in-house nexus and use it instead of Maven Central or other public repositories. To import JARs from the remote Maven repository of Red Hat Decision Manager server to a local Maven project, turn on pre-emptive authentication for the repository server. You can do this by configuring authentication for **`guvnor-m2-repo`** in the **`pom.xml`** file as shown below:

```
 <server>
  <id>guvnor-m2-repo</id>
  <username>admin</username>
  <password>admin</password>
  <configuration>
    <wagonProvider>httpclient</wagonProvider>
    <httpConfiguration>
      <all>
        <usePreemptive>true</usePreemptive>
      </all>
    </httpConfiguration>
 </configuration>
</server>
```

Alternatively, you can set Authorization HTTP header with Base64 encoded credentials:

```
<server>
  <id>guvnor-m2-repo</id>
  <configuration>
    <httpHeaders>
      <property>
        <name>Authorization</name>
        <!-- Base64-encoded "admin:admin" -->
        <value>Basic YWRtaW46YWRtaW4=</value>
      </property>
    </httpHeaders>
  </configuration>
</server>
```

2.2. DUPLICATE GAV DETECTION IN DECISION CENTRAL

In Decision Central, all Maven repositories are checked for any duplicated **GroupId**, **ArtifactId**, and **Version** (GAV) values in a project. If a GAV duplicate exists, the performed operation is canceled.

Duplicate GAV detection is executed every time you perform the following operations:

- Save a project definition for the project.
- Save the **pom.xml** file.
- Install, build, or deploy a project.

The following Maven repositories are checked for duplicate GAVs:

- Repositories specified in the **<repositories>** and **<distributionManagement>** elements of the **pom.xml** file.
- Repositories specified in the Maven **settings.xml** configuration file.

2.3. MANAGING DUPLICATE GAV DETECTION SETTINGS IN DECISION CENTRAL

Decision Central users with the **admin** role can modify the list of repositories that are checked for duplicate **GroupId**, **ArtifactId**, and **Version** (GAV) values for a project.

Procedure

1. In Decision Central, go to **Menu** → **Design** → **Projects** and click the project name.
2. Click the project **Settings** tab and then click **Validation** to open the list of repositories.
3. Select or clear any of the listed repository options to enable or disable duplicate GAV detection. In the future, duplicate GAVs will be reported for only the repositories you have enabled for validation.

NOTE

To disable this feature, set the **org.guvnor.project.gav.check.disabled** system property to **true** for Decision Central at system startup:

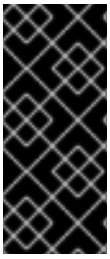
```
$ ~/EAP_HOME/bin/standalone.sh -c standalone-full.xml
-Dorg.guvnor.project.gav.check.disabled=true
```

CHAPTER 3. APPLYING PATCH UPDATES AND MINOR RELEASE UPGRADES TO RED HAT DECISION MANAGER

Automated update tools are often provided with both patch updates and new minor versions of Red Hat Decision Manager to facilitate updating certain components of Red Hat Decision Manager, such as Decision Central, Decision Server, and the headless Decision Manager controller. Other Red Hat Decision Manager artifacts, such as the decision engine and standalone Decision Central, are released as new artifacts with each minor release and you must re-install them to apply the update.

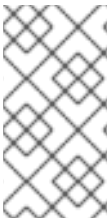
You can use the same automated update tool to apply both patch updates and minor release upgrades to Red Hat Decision Manager 7.1. Patch updates of Red Hat Decision Manager, such as an update from version 7.1 to 7.1.1, include the latest security updates and bug fixes. Minor release upgrades of Red Hat Decision Manager, such as an upgrade from version 7.1.x to 7.2, include enhancements, security updates, and bug fixes.

Before you upgrade to a new minor release, apply the latest patch update to your current version of Red Hat Decision Manager to ensure that the minor release upgrade is successful.



IMPORTANT

To upgrade from Red Hat Decision Manager 7.0.x to 7.1, you must use a Decision Central migration tool provided with the Red Hat Decision Manager 7.1 release to accommodate an improved project data structure in Red Hat Decision Manager 7.1. For migration instructions, see *Migrating from Red Hat Decision Manager 7.0 to Red Hat Decision Manager 7.1*.



NOTE

Only updates for Red Hat Decision Manager are included in Red Hat Decision Manager update tools. Updates to Red Hat JBoss EAP must be applied using Red Hat JBoss EAP patch distributions. For more information about Red Hat JBoss EAP patching, see the Red Hat JBoss EAP Patching and upgrading guide.

Prerequisite

Your Red Hat Decision Manager and Decision Server instances are not running. Do not apply updates while you are running an instance of Red Hat Decision Manager or Decision Server.

Procedure

1. Navigate to the Software Downloads page in the Red Hat Customer Portal (login required), and select the product and version from the drop-down options.
   Example:

   - **Product:** Decision Manager
   - **Version:** 7.1.1

   If you are upgrading to a new minor release of Red Hat Decision Manager, such as an upgrade from version 7.1.x to 7.2, first apply the latest patch update to your current version of Red Hat Decision Manager and then follow this procedure again to upgrade to the new minor release.

2. Click **Patches**, download **Red Hat Decision Manager [VERSION] Update**, and extract the downloaded **`rhdm-$VERSION-update.zip`** file to a temporary directory.

This update tool automates the update of certain components of Red Hat Decision Manager, such as Decision Central, Decision Server, and the headless Decision Manager controller. Use this update tool first to apply updates and then install any other updates or new release artifacts that are relevant to your Red Hat Decision Manager distribution.

3. If you want to preserve any files from being updated by the update tool, navigate to the extracted **rhdm-$VERSION-update** folder, open the **blacklist.txt** file, and add the relative paths to the files that you do not want to be updated.
When a file is listed in the **blacklist.txt** file, the update script does not replace the file with the new version but instead leaves the file in place and in the same location adds the new version with a **.new** suffix. If you blacklist files that are no longer being distributed, the update tool creates an empty marker file with a **.removed** suffix. You can then choose to retain, merge, or delete these new files manually.

   Example files to be excluded in **blacklist.txt** file:

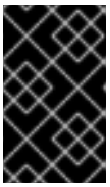
```
WEB-INF/web.xml  // Custom file
styles/base.css  // Obsolete custom file kept for record
```

   The contents of the blacklisted file directories after the update:

```
$ ls WEB-INF
web.xml web.xml.new
```

```
$ ls styles
base.css base.css.removed
```

4. In your command terminal, navigate to the temporary directory where you extracted the **rhdm-$VERSION-update.zip** file and run the **apply-updates** script in the following format:



IMPORTANT

Make sure that your Red Hat Decision Manager and Decision Server instances are not running before you apply updates. Do not apply updates while you are running an instance of Red Hat Decision Manager or Decision Server.

   On Linux or Unix-based systems:

```
$ ./apply-updates.sh $DISTRO_PATH $DISTRO_TYPE
```

   On Windows:

```
$ .\apply-updates.bat $DISTRO_PATH $DISTRO_TYPE
```

   The **$DISTRO_PATH** portion is the path to the relevant distribution directory and the **$DISTRO_TYPE** portion is the type of distribution that you are updating with this update.

   The following distribution types are supported in Red Hat Decision Manager update tool:

   - **rhdm-decision-central-eap7-deployable**: Updates Decision Central (**decision-central.war**)
   - **rhdm-kie-server-ee7**: Updates Decision Server (**kie-server.war**)

- **rhdm-kie-server-jws**: Updates Decision Server on Red Hat JBoss Web Server (**kie-server.war**)
- **rhdm-controller-ee7**: Updates the headless Decision Manager controller (**controller.war**)
- **rhdm-controller-jws**: Updates the headless Decision Manager controller on Red Hat JBoss Web Server (**controller.war**)

Example update to Decision Central and Decision Server for a full Red Hat Decision Manager distribution on Red Hat JBoss EAP:

```
./apply-updates.sh ~EAP_HOME/standalone/deployments/decision-central.war rhdm-decision-central-eap7-deployable

./apply-updates.sh ~EAP_HOME/standalone/deployments/kie-server.war rhdm-kie-server-ee7
```

Example update to headless Decision Manager controller, if used:

```
./apply-updates.sh ~EAP_HOME/standalone/deployments/controller.war rhdm-controller-ee7
```

The update script creates a **backup** folder in the extracted **rhdm-$VERSION-update** folder with a copy of the specified distribution, and then proceeds with the update.

5. After the update tool completes, return to the **Software Downloads** page of the Red Hat Customer Portal where you downloaded the update tool and install any other updates or new release artifacts that are relevant to your Red Hat Decision Manager distribution.
   For files that already exist in your Red Hat Decision Manager distribution, such as **.jar** files for the decision engine or other add-ons, replace the existing version of the file with the new version from the Red Hat Customer Portal.

6. If you use the standalone **Red Hat Decision Manager 7.1.0 Maven Repository** artifact (**rhdm-7.1.0-maven-repository.zip**), such as in air-gap environments, download **Red Hat Decision Manager [VERSION] Incremental Maven Repository** and extract the downloaded **rhdm-$VERSION-incremental-maven-repository.zip** file to your existing **~/maven-repository** directory to update the relevant contents.
   Example Maven repository update:

```
$ unzip -o rhdm-7.1.1-incremental-maven-repository.zip -d $REPO_PATH/rhdm-7.1.0-maven-repository/maven-repository/
```

7. After you finish applying all relevant updates, start Red Hat Decision Manager and Decision Server and log in to Decision Central.

8. Verify that all project data is present and accurate in Decision Central, and in the top-right corner of the Decision Central window, click your profile name and click **About** to verify the updated product version number.
   If you encounter errors or notice any missing data in Decision Central, you can restore the contents in the **backup** folder within the **rhdm-$VERSION-update** folder to revert the update tool changes. You can also re-install the relevant release artifacts from your previous version of Red Hat Decision Manager in the Red Hat Customer Portal. After restoring your previous distribution, you can try again to run the update.

CHAPTER 4. CONFIGURING AND STARTING DECISION SERVER

You can configure your Decision Server location, user name, password, and other related properties by defining the necessary configurations when you start Decision Server.

Procedure

Navigate to the Red Hat Decision Manager 7.1 **bin** directory and start the new Decision Server with the following properties. Adjust the specific properties according to your environment.

```
$ ~/EAP_HOME/bin/standalone.sh --server-config=standalone-full.xml (1)
-Dorg.kie.server.id=myserver (2)
-Dorg.kie.server.user=decision_server_username (3)
-Dorg.kie.server.pwd=decision_server_password (4)
-Dorg.kie.server.controller=http://localhost:8080/decision-
central/rest/controller (5)
-Dorg.kie.server.controller.user=controller_username (6)
-Dorg.kie.server.controller.pwd=controller_password (7)
-Dorg.kie.server.location=http://localhost:8080/kie-
server/services/rest/server (8)
-
Dorg.kie.server.persistence.dialect=org.hibernate.dialect.PostgreSQLDialec
t (9)
-Dorg.kie.server.persistence.ds=java:jboss/datasources/psjbpmDS (10)
```

1. Start command with **standalone-full.xml** server profile
2. Server ID that must match the server configuration name defined in Decision Central
3. User name to connect with Decision Server from the Decision Manager controller
4. Password to connect with Decision Server from the Decision Manager controller
5. Decision Manager controller location, Decision Central URL with **/rest/controller** suffix
6. User name to connect to the Decision Manager controller REST API
7. Password to connect to the Decision Manager controller REST API
8. Decision Server location (on the same instance as Decision Central in this example)
9. Hibernate dialect to be used
10. JNDI name of the data source used for your previous Red Hat JBoss BRMS database

NOTE

If Decision Central and Decision Server are installed on separate application server instances (Red Hat JBoss EAP or other), use a separate port for the Decision Server location to avoid port conflicts with Decision Central. If a separate Decision Server port has not already been configured, you can add a port offset and adjust the Decision Server port value accordingly in the Decision Server properties.

Example:

```
-Djboss.socket.binding.port-offset=150
-Dorg.kie.server.location=http://localhost:8230/kie-
server/services/rest/server
```

If the Decision Central port is 8080, as in this example, then the Decision Server port, with a defined offset of 150, is 8230.

Decision Server connects to the new Decision Central and collects the list of deployment units (KIE containers) to be deployed.

CHAPTER 5. CONFIGURING JDBC DATA SOURCES FOR DECISION SERVER

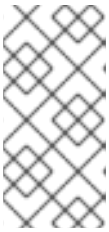
A data source is an object that enables a Java Database Connectivity (JDBC) client, such as an application server, to establish a connection with a database. Applications look up the data source on the Java Naming and Directory Interface (JNDI) tree or in the local application context and request a database connection to retrieve data. You must configure data sources for Decision Server to ensure proper data exchange between the servers and the designated database.

Prerequisite

The JDBC providers that you want to use to create database connections are configured on all servers on which you want to deploy Decision Server.

Procedure

1. Open ***EAP_HOME*/standalone/configuration/standalone-full.xml** in a text editor and locate the **<system-properties>** tag.
2. Add the following properties to the **<system-properties>** tag where **<DATASOURCE>** is the name of your data source and **<HIBERNATE_DIALECT>** is the hibernate dialect for your database.



NOTE

The default value of the **org.kie.server.persistence.ds** property is **java:jboss/datasources/ExampleDS**. The default value of the **org.kie.server.persistence.dialect** property is **org.hibernate.dialect.H2Dialect**.

```
<property name="org.kie.server.persistence.ds" value="
<DATASOURCE>"/>
<property name="org.kie.server.persistence.dialect" value="
<HIBERNATE_DIALECT>"/>
```

For example:

```
<system-properties>
    <property name="org.kie.server.repo"
value="${jboss.server.data.dir}"/>
    <property name="org.kie.example" value="true"/>
    <property name="org.jbpm.designer.perspective" value="full"/>
    <property name="designerdataobjects" value="false"/>
    <property name="org.kie.server.user" value="rhdmUser"/>
    <property name="org.kie.server.pwd" value="rhdm123!"/>
    <property name="org.kie.server.location"
value="http://localhost:8080/kie-server/services/rest/server"/>
    <property name="org.kie.server.controller"
value="http://localhost:8080/decision-central/rest/controller"/>
    <property name="org.kie.server.controller.user"
value="kieserver"/>
    <property name="org.kie.server.controller.pwd"
value="kieserver1!"/>
    <property name="org.kie.server.id" value="local-server-123"/>
```

```
    <!-- Data source properties. -->
    <property name="org.kie.server.persistence.ds"
value="java:jboss/datasources/KieServerDS"/>
    <property name="org.kie.server.persistence.dialect"
value="org.hibernate.dialect.PostgreSQLDialect"/>
</system-properties>
```

The following dialects are supported:

- DB2: **`org.hibernate.dialect.DB2Dialect`**
- MSSQL: **`org.hibernate.dialect.SQLServer2012Dialect`**
- MySQL: **`org.hibernate.dialect.MySQL5InnoDBDialect`**
- MariaDB: **`org.hibernate.dialect.MySQL5InnoDBDialect`**
- Oracle: **`org.hibernate.dialect.Oracle10gDialect`**
- PostgreSQL: **`org.hibernate.dialect.PostgreSQL82Dialect`**
- PostgreSQL plus: **`org.hibernate.dialect.PostgresPlusDialect`**
- Sybase: **`org.hibernate.dialect.SybaseASE157Dialect`**

CHAPTER 6. CONFIGURING DECISION SERVER WITH THE INTEGRATED DECISION MANAGER CONTROLLER

NOTE

Only make the changes described in this section if Decision Server will be managed by Decision Central and you installed Red Hat Decision Manager from the ZIP files. If you did not install Decision Central, you can use the headless Decision Manager controller to manage Decision Server, as described in Chapter 7, *Installing and running the headless Decision Manager controller*.

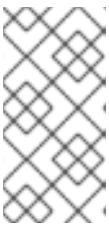
Decision Server can be managed or it can be unmanaged. If Decision Server is unmanaged, you must manually create and maintain KIE containers (deployment units). If Decision Server is managed, the Decision Manager controller manages the Decision Server configuration and you interact with the Decision Manager controller to create and maintain KIE containers.

The Decision Manager controller is integrated with Decision Central. If you install Decision Central, you can use the **Execution Server** page in Decision Central to interact with the Decision Manager controller.

If you installed Red Hat Decision Manager from the ZIP files, you must edit the **`standalone-full.xml`** file in both the Decision Server and Decision Central installations to configure Decision Server with the integrated Decision Manager controller.

Prerequisites

- Decision Central and Decision Server are installed in the base directory of the Red Hat JBoss EAP installation (***`EAP_HOME`***).



NOTE

You should install Decision Central and Decision Server on different servers in production environments. However, if you install Decision Server and Decision Central on the same server, for example in a development environment, make the changes described in this section in the shared **`standalone-full.xml`** file.

- On Decision Central server nodes, a user with the **`rest-all`** role exists.

Procedure

1. In the Decision Central ***`EAP_HOME`***`/standalone/configuration/standalone-full.xml` file, uncomment the following properties in the **`<system-properties>`** section and replace **`<USERNAME>`** and **`<USER_PWD>`** with the credentials of a user with the **`kie-server`** role:

```
    <property name="org.kie.server.user" value="<USERNAME>"/>
    <property name="org.kie.server.pwd" value="<USER_PWD>"/>
```

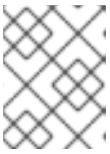
2. In the Decision Server ***`EAP_HOME`***`/standalone/configuration/standalone-full.xml` file, uncomment the following properties in the **`<system-properties>`** section.

```
  <property name="org.kie.server.controller.user" value="
<CONTROLLER_USER>"/>
```

```
  <property name="org.kie.server.controller.pwd" value="
<CONTROLLER_PWD>"/>
  <property name="org.kie.server.id" value="<KIE_SERVER_ID>"/>
  <property name="org.kie.server.location" value="http://<HOST>:
<PORT>/kie-server/services/rest/server"/>
  <property name="org.kie.server.controller" value="
<CONTROLLER_URL>"/>
```

3. Replace the following values:

- Replace **<CONTROLLER_USER>** and **<CONTROLLER_PWD>** with the credentials of a user with the **rest-all** role.
- Replace **<KIE_SERVER_ID>** with the ID or name of the Decision Server installation, for example, **rhdm-7.1.0-decision_server-1**.
- Replace **<HOST>** with the ID or name of the Decision Server host, for example, **localhost** or **192.7.8.9**.
- Replace **<PORT>** with the port of the Decision Server host, for example, **8080**.



NOTE

The **org.kie.server.location** property specifies the location of Decision Server.

- Replace **<CONTROLLER_URL>** with the URL of Decision Central. Decision Server connects to this URL during startup.
  - If you installed Decision Central using the installer or Red Hat JBoss EAP zip installations, **<CONTROLLER_URL>** has this format:
    http://<HOST>:<PORT>/decision-central/rest/controller
  - If you are running Decision Central using the **standalone.jar** file, **<CONTROLLER_URL>** has this format:
    http://<HOST>:<PORT>/rest/controller

CHAPTER 7. INSTALLING AND RUNNING THE HEADLESS DECISION MANAGER CONTROLLER

You can configure Decision Server to run in managed or unmanaged mode. If Decision Server is unmanaged, you must manually create and maintain KIE containers (deployment units). If Decision Server is managed, the Decision Manager controller manages the Decision Server configuration and you interact with the Decision Manager controller to create and maintain KIE containers.

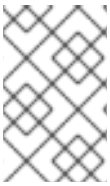
Decision Central has an embedded Decision Manager controller. If you install Decision Central, use the **Execution Server** page to create and maintain KIE containers. If you want to automate Decision Server management without Decision Central, you can use the headless Decision Manager controller.

7.1. USING THE INSTALLER TO CONFIGURE DECISION SERVER WITH THE DECISION MANAGER CONTROLLER

Decision Server can be managed by the Decision Manager controller or it can be unmanaged. If Decision Server is unmanaged, you must manually create and maintain KIE containers (deployment units). If Decision Server is managed, the Decision Manager controller manages the Decision Server configuration and you interact with the Decision Manager controller to create and maintain KIE containers.

The Decision Manager controller is integrated with Decision Central. If you install Decision Central, you can use the **Execution Server** page in Decision Central to interact with the Decision Manager controller.

You can use the installer in interactive or CLI mode to install Decision Central and Decision Server, and then configure Decision Server with the Decision Manager controller.



NOTE

If you do not install Decision Central, see Chapter 7, *Installing and running the headless Decision Manager controller* for information about using the headless Decision Manager controller.

Prerequisites

- Two computers with backed-up Red Hat JBoss EAP 7.1 or higher server installations are available.
- Sufficient user permissions to complete the installation are granted.

Procedure

1. On the first computer, run the installer in interactive mode or CLI mode. See *Installing and configuring Red Hat Decision Manager on Red Hat JBoss EAP 7.1* for more information.
2. On the **Component Selection** page, clear the **Decision Server** box.
3. Complete the Decision Central installation.
4. On the second computer, run the installer in interactive mode or CLI mode.
5. On the **Component Selection** page, clear the **Decision Central** box.
6. On the **Configure Runtime Environment** page, select **Perform Advanced Configuration**.

7. Select **Customize Decision Server properties** and click **Next**.

8. On the **Process Server Properties Configuration** page, click **New Server Configuration** to add a Decision Server and specify a unique name for that Decision Server. This name will appear in Decision Central and enable you to distinguish between different Decision Servers.

7.2. INSTALLING THE HEADLESS DECISION MANAGER CONTROLLER

You can install the headless Decision Manager controller and use the REST API or the Decision Server Java Client API to interact with it.

Prerequisites

- A backed-up Red Hat JBoss EAP installation version 7.1 or higher is available. The base directory of the Red Hat JBoss EAP installation is referred to as ***EAP_HOME***.
- Sufficient user permissions to complete the installation are granted.

Procedure

1. Navigate to the Software Downloads page in the Red Hat Customer Portal (login required), and select the product and version from the drop-down options:
   - **Product:** Decision Manager
   - **Version:** 7.1
2. Download **Red Hat Decision Manager 7.1.0 Add Ons** (the **`rhdm-7.1.0-add-ons.zip`** file).
3. Unzip the **`rhdm-7.1.0-add-ons.zip`** file. The **`rhdm-7.1-controller-ee7.zip`** file is in the unzipped directory.
4. Extract the **`rhdm-7.1-controller-ee7`** archive to a temporary directory. In the following examples this directory is called ***`TEMP_DIR`***.
5. Copy the ***`TEMP_DIR`*`/rhdm-7.1-controller-ee7/controller.war`** directory to ***`EAP_HOME`*`/standalone/deployments/`**.

WARNING

Ensure that the names of the headless Decision Manager controller deployments you are copying do not conflict with your existing deployments in the Red Hat JBoss EAP instance.

6. Copy the contents of the ***`TEMP_DIR`*`/rhdm-7.1-controller-ee7/SecurityPolicy/`** directory to ***`EAP_HOME`*`/bin`**. When asked to overwrite files, select **Yes**.
7. In the ***`EAP_HOME`*`/standalone/deployments/`** directory, create an empty file named **`controller.war.dodeploy`**. This file ensures that the headless Decision Manager controller is automatically deployed when the server starts.

7.2.1. Creating a headless Decision Manager controller user

Before you can use the headless Decision Manager controller, you must create a user that has the **kie-server** role.

Prerequisite

The headless Decision Manager controller is installed in the base directory of the Red Hat JBoss EAP installation (***EAP_HOME***).

Procedure

1. In a terminal application, navigate to the ***EAP_HOME*/bin** directory.
2. Enter the following command and replace **<USER_NAME>** and **<PASSWORD>** with the user name and password of your choice.

```
$ ./add-user.sh -a --user <username> --password <password> --role
kie-server
```

NOTE

Make sure that the specified user name is not the same as an existing user, role, or group. For example, do not create a user with the user name **admin**.

The password must have at least eight characters and must contain at least one number and one non-alphanumeric character, but not & (ampersand).

3. Make a note of your user name and password.

7.2.2. Configuring Decision Server and the headless Decision Manager controller

If Decision Server will be managed by the headless Decision Manager controller, you must edit the **standalone-full.xml** file in both the Decision Server and headless Decision Manager controller installations, as described in this section.

Prerequisites

- Decision Server is installed in the base directory of the Red Hat JBoss EAP installation (***EAP_HOME***).
- The headless Decision Manager controller is installed in an ***EAP_HOME***.

NOTE

You should install Decision Server and the headless Decision Manager controller on different servers in production environments. However, if you install Decision Server and the headless Decision Manager controller on the same server, for example in a development environment, make these changes in the shared **standalone-full.xml** file.

- On Decision Server nodes, a user with the **kie-server** role exists.

- On the server nodes, a user with the **kie-server** role exists.

Procedure

1. In the ***EAP_HOME*/standalone/configuration/standalone-full.xml** file, add the following properties to the **<system-properties>** section and replace **<USERNAME>** and **<USER_PWD>** with the credentials of a user with the **kie-server** role:

```
    <property name="org.kie.server.user" value="<USERNAME>"/>
    <property name="org.kie.server.pwd" value="<USER_PWD>"/>
```

2. In the Decision Server ***EAP_HOME*/standalone/configuration/standalone-full.xml** file, add the following properties to the **<system-properties>** section:

```
  <property name="org.kie.server.controller.user" value="
<CONTROLLER_USER>"/>
  <property name="org.kie.server.controller.pwd" value="
<CONTROLLER_PWD>"/>
  <property name="org.kie.server.id" value="<KIE_SERVER_ID>"/>
  <property name="org.kie.server.location" value="http://<HOST>:
<PORT>/kie-server/services/rest/server"/>
  <property name="org.kie.server.controller" value="
<CONTROLLER_URL>"/>
```

3. In this file, replace the following values:

- Replace **<CONTROLLER_USER>** and **<CONTROLLER_PWD>** with the credentials of a user with the **kie-server** role.
- Replace **<KIE_SERVER_ID>** with the ID or name of the Decision Server installation, for example, **rhdm-7.1.0-decision_server-1**.
- Replace **<HOST>** with the ID or name of the Decision Server host, for example, **localhost** or **192.7.8.9**.
- Replace **<PORT>** with the port of the Decision Server host, for example, **8080**.

NOTE

The **org.kie.server.location** property specifies the location of Decision Server.

- Replace **<CONTROLLER_URL>** with the URL of the headless Decision Manager controller.

1. Decision Server connects to this URL during startup.

7.3. RUNNING THE HEADLESS DECISION MANAGER CONTROLLER

After you have installed the headless Decision Manager controller on Red Hat JBoss EAP, use this procedure to run the headless Decision Manager controller.

Prerequisite

The headless Decision Manager controller is installed and configured in the base directory of the Red Hat JBoss EAP installation (***EAP_HOME***).

Procedure

1. In a terminal application, navigate to ***EAP_HOME*/bin**.
2. Enter the following command:
   - On Linux or UNIX-based systems:

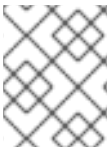
     ```
     $ ./standalone.sh
     ```

   - On Windows:

     ```
     standalone.bat
     ```

3. To verify that the headless Decision Manager controller is working on Red Hat JBoss EAP, enter the following command where **<CONTROLLER>** and **<CONTROLLER_PWD>** is the user name and password. The output of this command provides information about the Decision Server instance.

   ```
   curl -X GET "http://<HOST>:
   <PORT>/controller/rest/controller/management/servers" -H  "accept:
   application/xml" -u '<CONTROLLER>:<CONTROLLER_PWD>'
   ```



NOTE

Alternatively, you can use the Decision Server Java API Client to access the headless Decision Manager controller.

7.4. CLUSTERING WITH THE HEADLESS DECISION MANAGER CONTROLLER

The Decision Manager controller is integrated with Decision Central. However, if you do not install Decision Central, you can install the headless Decision Manager controller and use the REST API or the Decision Server Java Client API to interact with it.

Prerequisites

- A backed-up Red Hat JBoss EAP installation version 7.1 or later is available. The base directory of the Red Hat JBoss EAP installation is referred to as ***EAP_HOME***.
- Sufficient user permissions to complete the installation are granted.
- An NFS server with a mounted partition is available.

Procedure

1. Navigate to the Software Downloads page in the Red Hat Customer Portal (login required), and select the product and version from the drop-down options:
   - **Product: Decision Manager**

- **Version: 7.1**

2. Download **Red Hat Decision Manager 7.1.0 Add Ons** (the **`rhdm-7.1.0-add-ons.zip`** file).
3. Unzip the **`rhdm-7.1.0-add-ons.zip`** file. The **`rhdm-7.1-controller-ee7.zip`** file is in the unzipped directory.
4. Extract the **`rhdm-7.1-controller-ee7.zip`** file to a temporary directory. In the following examples this directory is called ***`TEMP_DIR`***.
5. Repackage the **`controller.war`** directory:
   a. Navigate to the ***`TEMP_DIR`*****`/rhdm-7.1.0-add-ons/rhdm-7.1-controller-ee7/controller.war`** directory.
   b. Select the contents of the ***`TEMP_DIR`*****`/rhdm-7.1.0-add-ons/rhdm-7.1-controller-ee7/controller.war`** directory and create the **`controller.zip`** file.
   c. Rename **`controller.zip`** to **`controller.war`**. This is the file that you will use to deploy the headless Decision Manager controller on the cluster nodes.
   d. If desired, copy the new **`controller.war`** file to a location that is more convenient to deploy from.
6. If you want to use a security manager with the headless Decision Manager controller, copy the contents of the ***`TEMP_DIR`*****`/rhdm-7.1.0-add-ons/rhdm-7.1-controller-ee7/SecurityPolicy`** directory to the ***`EAP_HOME`*****`/bin`** directory on each node of the cluster.
7. Add Red Hat JBoss EAP management users to the master node (where you configured the **`domain.xml`** file) as described in the *Red Hat JBoss EAP 7.1 Configuration Guide*.
8. On each node of the cluster, add users for the headless Decision Manager controller as described in the "Installing the headless Decision Manager controller" section of the *Installing and configuring Red Hat Decision Manager on Red Hat JBoss EAP 7.1*.
9. Complete the following steps in the **`host.xml`** file on the master node and in the **`host-slave.xml`** file on each slave node:
   a. Open the ***`EAP_HOME`*****`/domain/configuration/host.xml`** or ***`EAP_HOME`*****`/domain/configuration/host-slave.xml`** file in a text editor.
   b. In the **`main-server-group`** **`<servers>`** element, add the servers that will be part of the cluster.
   c. Add the following properties to the **`<system-properties>`** element and replace **`<NFS_STORAGE>`** with the absolute path to the NFS storage where the template configuration is stored.

```
<system-properties>
  <property
name="org.kie.server.controller.templatefile.watcher.enabled"
value="true"/>
  <property name="org.kie.server.controller.templatefile" value="
<NFS_STORAGE>"/>
</system-properties>
```

If the value of the **org.kie.server.controller.templatefile.watcher.enabled** property is set to **true**, a separate thread is started to watch for modifications of the template file. The default interval for these checks is 30000 milliseconds and can be further controlled by the **org.kie.server.controller.templatefile.watcher.interval** system property. If the value of this property is set to **false**, changes to the template file are detected only when the server restarts.

10. To deploy the **controller.war** file that you created previously into the server group, complete the following steps on the master node:

    a. Log in to the Red Hat JBoss EAP **Administration** console of your domain as a **management** user.

    b. Click **Deployments** → **Server Groups**→ **main-server-group** and click **Add**.

    c. In the dialog box, click **Upload a new deployment** and click **Next**.

    d. In the **Upload Deployments** dialog box, click **Browse**, select the **controller.war** file, and click **Next**.

    e. Click **Enable** and click **Next**.

NOTE

Make sure to check deployment unit readiness with every cluster member.

When a deployment unit is created on a cluster node, it takes some time before it is distributed among all cluster members. You can check the deployment status using the server Administration console or REST API. However, if the query is sent to the node where the deployment was originally issued, the query will return a value of **deployed**. If the query is sent to a node where the deployment has not yet been distributed, the query returns **DeploymentNotFoundException**.

For more information about installing Decision Central, see *Installing and configuring Red Hat Decision Manager on Red Hat JBoss EAP 7.1*.

7.5. CONFIGURING THE HEADLESS DECISION MANAGER CONTROLLER

Prerequisites

- Decision Server is installed on each node of a Red Hat JBoss EAP 7.1 cluster.
- An NFS server with a mounted partition accessible to a Red Hat JBoss EAP user is available.

Procedure

On the java process running Smart Router, do the following:

1. Enable the Decision Server watcher service system property:

```
org.kie.server.controller.templatefile.watcher.enabled=true
```

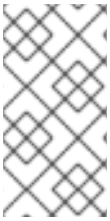
2. In the **`org.kie.server.controller.templatefile`** property, specify the absolute path to the NFS storage where the memory configuration is stored:

```
org.kie.server.controller.templatefile=
<absolute_path_to_NFS_storage>
```

For more information about running Red Hat Decision Manager in a Red Hat JBoss Enterprise Application Platform clustered environment, see *Installing and configuring Red Hat Decision Manager in a Red Hat JBoss EAP 7.1 clustered environment*.

CHAPTER 8. CONFIGURING A DECISION SERVER TO CONNECT TO DECISION CENTRAL

If a Decision Server is not already configured in your Red Hat Decision Manager environment, or if you require additional Decision Servers in your Red Hat Decision Manager environment, you must configure a Decision Server to connect to Decision Central.



NOTE

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

Prerequisite

Decision Server is installed. For installation options, see *Planning a Red Hat Decision Manager installation*.

Procedure

1. In your Red Hat Decision Manager installation directory, navigate to the **`standalone-full.xml`** file. For example, if you use a Red Hat JBoss EAP installation for Red Hat Decision Manager, go to **`$EAP_HOME/standalone/configuration/standalone-full.xml`**.

2. Open **`standalone-full.xml`** and under the **`<system-properties>`** tag, set the following properties:

   - **org.kie.server.controller.user:** The user name of a user who can log in to the Decision Central.
   - **org.kie.server.controller.pwd:** The password of the user who can log in to the Decision Central.
   - **org.kie.server.controller:** The URL for connecting to the API of Decision Central. Normally, the URL is **`http://<centralhost>:<centralport>/decision-central/rest/controller`**, where **`<centralhost>`** and **`<centralport>`** are the host name and port for Decision Central. If Decision Central is deployed on OpenShift, remove **`decision-central/`** from the URL.
   - **org.kie.server.location:** The URL for connecting to the API of Decision Server. Normally, the URL is **`http://<serverhost>:<serverport>/kie-server/services/rest/server`**, where **`<serverhost>`** and **`<serverport>`** are the host name and port for Decision Server.
   - **org.kie.server.id:** The name of a server configuration. If this server configuration does not exist in Decision Central, it is created automatically when Decision Server connects to Decision Central.

   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"/>
```

3. Start or restart the Decision Server.

CHAPTER 9. CONFIGURING DECISION SERVER MANAGED BY DECISION CENTRAL

WARNING

This section provides a sample setup that you can use for testing purposes. Some of the values are unsuitable for a production environment, and are marked as such.

Use this procedure to configure Decision Central to manage a Decision Server instance.

Prerequisite

Users with the following roles exist:

- In Decision Central, a user with the role **`rest-all`**.
- On the Decision Server, a user with the role **`kie-server`**.

NOTE

In production environments, use two distinct users, each with one role. In this sample situation, we use only one user named **`controllerUser`** that has both the **`rest-all`** and the **`kie-server`** roles.

Procedure

1. Set the following JVM properties.
   The location of Decision Central and the Decision Server may be different. In such case, ensure you set the properties on the correct server instances.
   - On Red Hat JBoss EAP, modify the **`<system-properties>`** section in:
     - ***`EAP_HOME`*`/standalone/configuration/standalone*.xml`** for standalone mode.
     - ***`EAP_HOME`*`/domain/configuration/domain.xml`** for domain mode.

Table 9.1. JVM Properties for Managed Decision Server Instance

Property	Value	Note
`org.kie.server.id`	`default-kie-server`	The Decision Server ID.
`org.kie.server.controller`	`http://localhost:8080/decision-central/rest/controller`	The location of Decision Central.

Property	Value	Note
org.kie.server.controller.user	**controllerUser**	The user name with the role **rest-all** as mentioned in the previous step.
org.kie.server.controller.pwd	**controllerUser1234;**	The password of the user mentioned in the previous step.
org.kie.server.location	**http://localhost:8080/kie-server/services/rest/server**	The location of the Decision Server.

Table 9.2. JVM Properties for Decision Central Instance

Property	Value	Note
org.kie.server.user	**controllerUser**	The user name with the role **kie-server** as mentioned in the previous step.
org.kie.server.pwd	**controllerUser1234;**	The password of the user mentioned in the previous step.

2. Verify the successful start of the Decision Server by sending a GET request to **http://*SERVER:PORT*/kie-server/services/rest/server/**. Once authenticated, you get an XML response similar to this:

```
<response type="SUCCESS" msg="Kie Server info">
    <kie-server-info>
        <capabilities>KieServer</capabilities>
        <capabilities>BRM</capabilities>
        <capabilities>BPM</capabilities>
        <capabilities>CaseMgmt</capabilities>
        <capabilities>BPM-UI</capabilities>
        <capabilities>BRP</capabilities>
        <capabilities>DMN</capabilities>
        <capabilities>Swagger</capabilities>
        <location>http://localhost:8230/kie-
server/services/rest/server</location>
        <messages>
            <content>Server KieServerInfo{serverId='first-kie-
server', version='7.5.1.Final-redhat-1',
location='http://localhost:8230/kie-server/services/rest/server',
capabilities=[KieServer, BRM, BPM, CaseMgmt, BPM-UI, BRP, DMN,
Swagger]}started successfully at Mon Feb 05 15:44:35 AEST
2018</content>
            <severity>INFO</severity>
```

```
            <timestamp>2018-02-05T15:44:35.355+10:00</timestamp>
        </messages>
        <name>first-kie-server</name>
        <id>first-kie-server</id>
        <version>7.5.1.Final-redhat-1</version>
    </kie-server-info>
</response>
```

3. Verify successful registration:

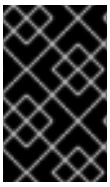
    a. Log in to Decision Central.

    b. Click **Menu** → **Deploy** → **Execution Servers**.
    If registration is successful, you can see the registered server ID.

CHAPTER 10. MANAGED DECISION SERVER

A managed instance requires an available Decision Manager controller to start the Decision Server.

A Decision Manager controller manages the Decision Server configuration in a centralized way. Each Decision Manager controller can manage multiple configurations at once, and there can be multiple more than one Decision Manager controller in the environment. Managed Decision Server can be configured with a list of Decision Manager controllers, but will only connect to one at a time.



IMPORTANT

All Decision Manager controllers should be synchronized to ensure that the same set of configuration is provided to the server, regardless of the Decision Manager controller to which it connects.

When the Decision Server is configured with a list of Decision Manager controllers, it will attempt to connect to each of them at startup until a connection is successfully established with one of them. If a connection cannot be established, the server will not start, even if there is a local storage available with configuration. This ensures consistence and prevents the server from running with redundant configuration.

NOTE

To run the Decision Server in standalone mode without connecting to Decision Manager controllers, see Chapter 11, *Unmanaged Decision Server*.

CHAPTER 11. UNMANAGED DECISION SERVER

An unmanaged Decision Server is a standalone instance, and therefore must be configured individually using REST/JMS API from the Decision Server itself, or using a headless Decision Manager controller. The configuration is automatically persisted by the server into a file and that is used as the internal server state, in case of restarts.

The configuration is updated during the following operations:

- Deploy KIE container
- Undeploy KIE container
- Start KIE container
- Stop KIE container

NOTE

If the Decision Server is restarted, it will attempt to re-establish the same state that was persisted before shutdown. Therefore, KIE containers (deployment units) that were running will be started, but the ones that were stopped will not.

CHAPTER 12. EXECUTION ERROR MANAGEMENT

When an execution error occurs the process stops and rolls back to the most recent stable state (the closest safe point) and continues its execution. If an error of any kind is not handled by the process the entire transaction rolls back, leaving the process instance in the previous wait state. Any trace of this is only visible in the logs, and usually displayed to the caller who sent the request to the decision engine.

Users with process administrator (**`process-admin`**) or administrator (**`admin`**) roles are able to access error messages in Decision Central, which has the following features:

- Better traceability
- Visibility in case of critical processes
- Reporting and analytics based on error situations
- External system error handling and compensation

Configurable error handling is responsible for receiving any technical errors thrown throughout the decision engine execution (including task service). The following technical exceptions apply:

- Anything that extends **`java.lang.Throwable`**.
- Process level error handling and any other exceptions not previously handled.

There are several components that make up the error handling mechanism and allow a pluggable approach to extend its capabilities.

The decision engine entry point for error handling is the **`ExecutionErrorManager`**. This is integrated with RuntimeManager, which is then responsible for providing it to underlying components - **`KieSession`** and **`TaskService`**. From the API point of view, **`ExecutionErrorManager`** gives access to:

- **`ExecutionErrorHandler`** - the primary mechanism for error handling.
- **`ExecutionErrorStorage`** - pluggable storage for execution error information.

12.1. MANAGE EXECUTION ERRORS

By definition, every error that is caught and stored is unacknowledged, meaning that it is to be handled by someone or something (in case of automatic error recovery). Errors are filtered on the basis of whether or not they have been acknowledged. Acknowledging an error saves the user information and time stamp for traceability.

You can access the **Error Management** view at any time.

1. In Decision Central, click **Menu** → **Manage** → **Execution Errors**.
2. Select an error from the list to open the **Details** tab. This displays information about the error or errors.
3. Click the **Acknowledge** button to acknowledge and clear the error. The error can still be viewed later by selecting **Yes** on the **Acknowledged** filter in the **Manage Execution Errors** page.
4. If the error was related to a task, a **Go to Task** button is displayed.
   Click the **Go to Task** button to view the associated job information in the **Manage Tasks** page.

The **Manage Tasks** page allows you to restart, reschedule, or retry the corresponding task.

12.2. THE EXECUTIONERRORHANDLER

The **`ExecutionErrorHandler`** is the primary mechanism for all process error handling. It is bound to the life cycle of RuntimeEngine; meaning it is created when a new runtime engine is created, and is destroyed when RuntimeEngine is disposed. A single instance of the **`ExecutionErrorHandler`** is used within a given execution context or transaction. Both **`KieSession`** and **`TaskService`** use that instance to inform the error handling about processed nodes/tasks. **`ExecutionErrorHandler`** is informed about:

- Starting of processing of a given node instance.
- Completion of processing of a given node instance.
- Starting of processing of a given task instance.
- Completion of processing of a given task instance.

This information is mainly used for errors that are of unknown type; that is, errors that do not provide information about the process context. For example, upon commit time, database exceptions do not carry any process information.

12.3. EXECUTION ERROR STORAGE

`ExecutionErrorStorage` is a pluggable strategy that permits various ways of persisting information about execution errors. Storage is used directly by the handler that gets an instance of the store when it is created (when RuntimeEngine is created). Default storage implementation is based on the database table, which stores every error and includes all of the available information. Some errors may not contain details, as this depends on the type of error and whether or not it is possible to extract specific information.

12.4. ERROR TYPES AND FILTERS

Error handling attempts to catch and handle any kind of error, therefore it needs a way to categorize errors. By doing this, it is able to properly extract information from the error and make it pluggable, as some users may require specific types of errors to be thrown and handled in different ways than what is provided by default.

Error categorization and filtering is based on **`ExecutionErrorFilters`**. This interface is solely responsible for building instances of **`ExecutionError`**, which are later stored by way of the **`ExecutionErrorStorage`** strategy. It has following methods:

- **`accept`**: indicates if given error can be handled by the filter.
- **`filter`**: where the actual filtering, handling, and so on happens.
- **`getPriority`**: indicates the priority that is used when calling filters.

As only one filter can process given error, filters use a priority system to avoid having multiple filters returning alternative “views” of the same error. Priority allows more specialized filters to see if the error can be accepted, or otherwise allow another filter to handle it.

ExecutionErrorFilter can be provided using the **ServiceLoader** mechanism, which allows the capability of error handling to be easily extended.

Red Hat Decision Manager ships with the following **ExecutionErrorFilters** :

Table 12.1. ExecutionErrorFilters

Class name	Type	Priority
org.jbpm.runtime.manager.impl.error.filters.ProcessExecutionErrorFilter	Process	100
org.jbpm.runtime.manager.impl.error.filters.TaskExecutionErrorFilter	Task	80
org.jbpm.runtime.manager.impl.error.filters.DBExecutionErrorFilter	DB	200
org.jbpm.executor.impl.error.JobExecutionErrorFilter	Job	100

Filters are given a higher execution order based on the lowest value of the priority. In above table, filters are invoked in following order:

1. Task
2. Process
3. Job
4. DB

12.5. AUTO ACKNOWLEDGING EXECUTION ERRORS

When executions errors occur they are unacknowledged by default, and require a manual acknowledgment to be performed otherwise they are always seen as information that requires attention. In case of larger volumes, manual actions can be time consuming and not suitable in some situations.

Auto acknowledgment resolves this issue. It is based on scheduled jobs by way of the **jbpm-executor**, with the following three types of jobs available:

org.jbpm.executor.commands.error.JobAutoAckErrorCommand

Responsible for finding jobs that previously failed but now are either canceled, completed, or rescheduled for another execution. This job only acknowledges execution errors of type **Job**.

org.jbpm.executor.commands.error.TaskAutoAckErrorCommand

Responsible for auto acknowledgment of user task execution errors for tasks that previously failed but now are in one of the exit states (completed, failed, exited, obsolete). This job only acknowledges execution errors of type **Task**.

org.jbpm.executor.commands.error.ProcessAutoAckErrorCommand

Responsible for auto acknowledgment of process instances that have errors attached. It acknowledges errors where the process instance is already finished (completed or aborted), or the task that the error originated from is already finished. This is based on **init_activity_id** value.

This job acknowledges any type of execution error that matches the above criteria.

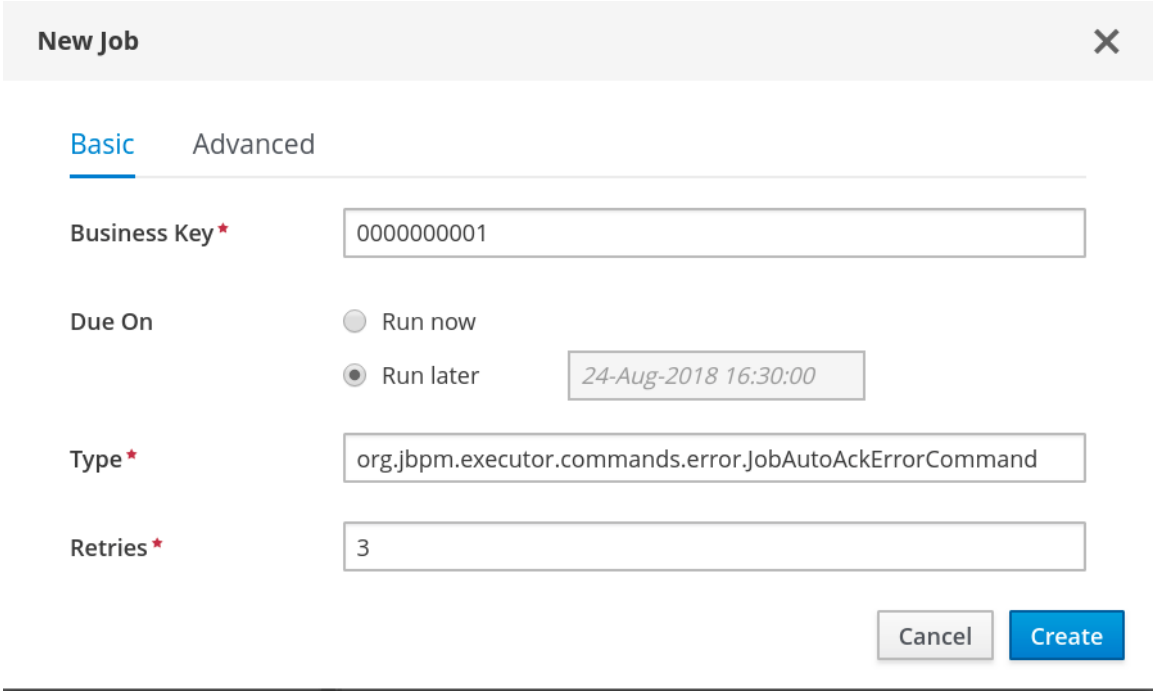
Jobs can be registered on the Decision Server. In Decision Central you can configure auto acknowledge jobs for errors:

Prerequisite

1. Execution errors of one or more type have accumulated during processes execution but require no further attention.

Procedure

1. In Decision Central, click **Menu** → **Manage** → **Jobs**.
2. In the top right of the screen, click **New Job**.
3. Type the process correlation key into the **Business Key** field.
4. In the **Type** field, add type of the auto acknowledge job type from the list above.
5. Select a **Due On** time for the job to be completed:
   a. To run the job immediately, select the **`Run now`** option.
   b. To run the job at a specific time, select **`Run later`**. A date and time field appears next to the **`Run later`** option. Click the field to open the calendar and schedule a specific time and date for the job.

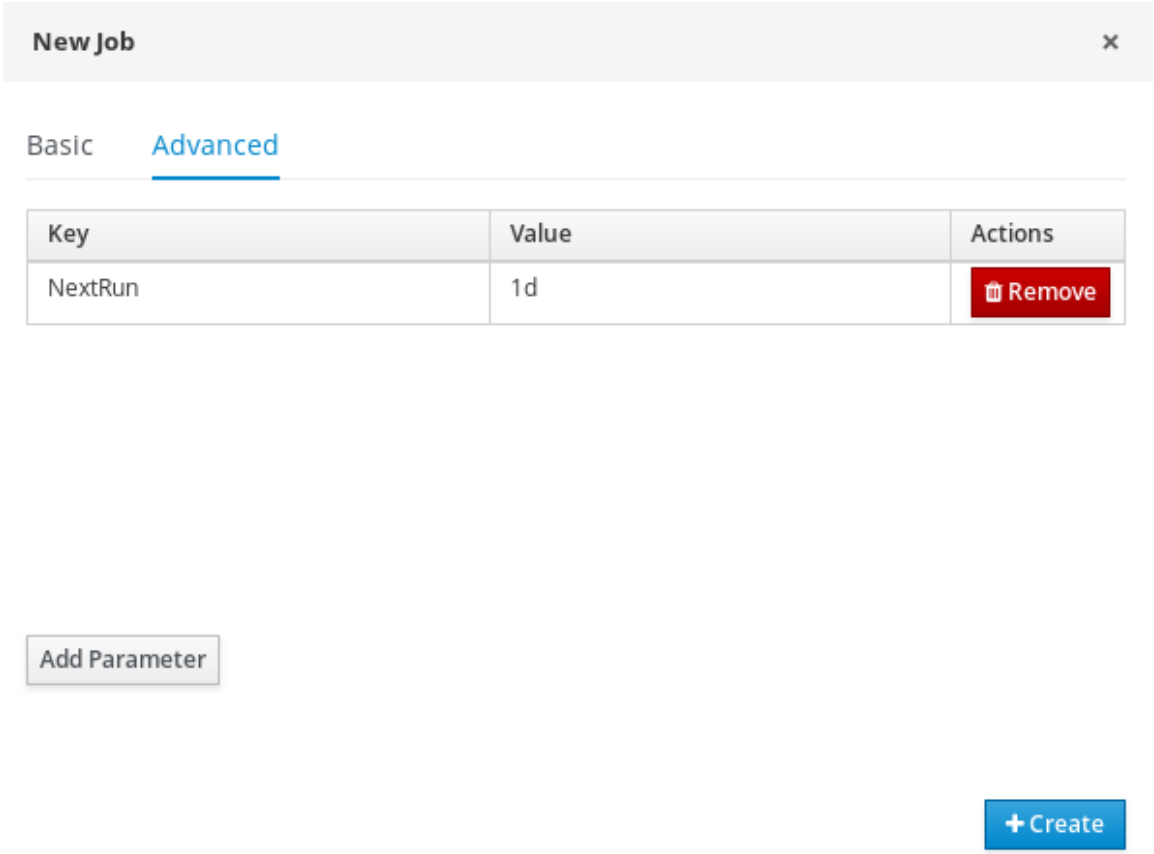


6. Click **Create** to create the job and return to the **Manage Jobs** page.

The following steps are optional, and allow you to configure auto acknowledge jobs to run either once (**`SingleRun`**), on specific time intervals (**`NextRun`**), or using the custom name of an entity manager factory to search for jobs to acknowledge (**`EmfName`**).

1. Click on the **Advanced** tab.
2. Click the **Add Parameter** button.

3. Enter the configuration parameter you want to apply to the job:
   a. **`SingleRun`**: **`true`** or **`false`**
   b. **`NextRun`**: time expression, such as 2h, 5d, 1m, and so on.
   c. **`EmfName`**: custom entity manager factory name.



12.6. CLEANING UP THE ERROR LIST

The **`ExecutionErrorInfo`** error list table can be cleaned up to remove redundant information. Depending on the life cycle of the process, errors may remain in the list for some time, and there is no direct API with which to clean up the list. Instead, **`jbpm-executor`** commands can be scheduled to periodically clean up errors.

The following options can be used as clean up commands, and are restricted to deleting execution errors of already completed or aborted process instances:

- **`DateFormat`**
  - Date format for further date related parameters - if not given **`yyyy-MM-dd`** is used (pattern of **`SimpleDateFormat`** class).
- **`EmfName`**
  - Name of the entity manager factory to be used for queries (valid persistence unit name).
- **`SingleRun`**
  - Indicates if execution should be single run only (**`true`**|**`false`**).
- **`NextRun`**

- Provides next execution time (valid time expression, for example: 1d, 5h, and so on)

- **`OlderThan`**
  - Indicates what errors should be deleted - older than given date.
- **`OlderThanPeriod`**
  - Indicated what errors should be deleted older than given time expression (valid time expression e.g. 1d, 5h, and so on)
- **`ForProcess`**
  - Indicates errors to be deleted only for given process definition.
- **`ForProcessInstance`**
  - Indicates errors to be deleted only for given process instance.
- **`ForDeployment`**
  - Indicates errors to be deleted that are from given deployment ID.

CHAPTER 13. CONFIGURING OPENSHIFT CONNECTION TIMEOUT

By default, the OpenShift route is configured to time out HTTP requests that are longer than 30 seconds. This may cause session timeout issues in Decision Central resulting in the following behaviors:

- "Unable to complete your request. The following exception occurred: (TypeError) : Cannot read property 'indexOf' of null."
- "Unable to complete your request. The following exception occurred: (TypeError) : b is null."
- A blank page is displayed when clicking the **Project** or **Server** links in Decision Central.

All Decision Central templates already include extended timeout configuration.

To configure longer timeout on Decision Central OpenShift routes, add the **`haproxy.router.openshift.io/timeout: 60s`** annotation on the target route:

```
- kind: Route
  apiVersion: v1
  id: "$APPLICATION_NAME-rhdmcentr-http"
  metadata:
    name: "$APPLICATION_NAME-rhdmcentr"
    labels:
      application: "$APPLICATION_NAME"
    annotations:
      description: Route for Decision Central's http service.
      haproxy.router.openshift.io/timeout: 60s
  spec:
    host: "$DECISION_CENTRAL_HOSTNAME_HTTP"
    to:
      name: "$APPLICATION_NAME-rhdmcentr"
```

For a full list of global route-specific timeout annotations, see the OpenShift Documentation.

CHAPTER 14. PERSISTENCE

Binary persistence, or marshaling, converts the state of the process instance into a binary data set. Binary persistence is a mechanism used to store and retrieve information persistently. The same mechanism is also applied to the session state and work item states.

When you enable persistence of a process instance:

- Red Hat Decision Manager transforms the process instance information into binary data. Custom serialization is used instead of Java serialization for performance reasons.
- The binary data is stored together with other process instance metadata, such as process instance ID, process ID, and the process start date.

The session can also store other forms of state, such as the state of timer jobs, or data required for business rules evaluation. Session state is stored separately as a binary data set along with the ID of the session and metadata. You can restore the session state by reloading a session with given ID. Use **`ksession.getId()`** to get the session ID.

Red Hat Decision Manager will persist the following when persistence is configured:

- *Session state*: This includes the session ID, date of last modification, the session data that business rules would need for evaluation, state of timer jobs.
- *Process instance state*: This includes the process instance ID, process ID, date of last modification, date of last read access, process instance start date, runtime data (the execution status including the node being executed, variable values, and other process instance data) and the event types.
- *Work item runtime state*: This includes the work item ID, creation date, name, process instance ID, and the work item state itself.

Based on the persisted data, you can restore the state of execution of all running process instances in case of failure or to temporarily remove running instances from memory and restore them later.

14.1. CONFIGURING SAFE POINTS

To allow persistence, add the **`jbpm-persistence`** JAR files to the classpath of your application and configure the decision engine to use persistence. The decision engine automatically stores the runtime state in the storage when the decision engine reaches a safe point.

Safe points are points where the process instance has paused. When a process instance invocation reaches a safe point in the decision engine, the decision engine stores any changes to the process instance as a snapshot of the process runtime data. However, when a process instance is completed, the persisted snapshot of process instance runtime data is automatically deleted.

If a failure occurs and you need to restore the decision engine runtime from the storage, the process instances are automatically restored and their execution resumes so there is no need to reload and trigger the process instances manually.

The runtime persistence data is to be considered internal to the decision engine. You should not access persisted runtime data or modify them directly as this might have unexpected side effects.

For more information about the current execution state, refer to the history log. Query the database for runtime data only if absolutely necessary.

14.2. SESSION PERSISTENCE ENTITIES

Sessions are persisted as **`SessionInfo`** entities. These persist the state of the runtime KIE session, and store the following data:

Table 14.1. SessionInfo

Field	Description	Nullable
`id`	The primary key.	NOT NULL
`lastModificationDate`	The last time that entity was saved to a database.	
`rulesByteArray`	The state of a session.	NOT NULL
`startDate`	The session start time.	
`OPTLOCK`	A version field containing a lock value.	

14.3. PROCESS INSTANCE PERSISTENCE ENTITIES

Process instances are persisted as **`ProcessInstanceInfo`** entities, which persist the state of a process instance on runtime and store the following data:

Table 14.2. ProcessInstanceInfo

Field	Description	Nullable
`instanceId`	The primary key.	NOT NULL
`lastModificationDate`	The last time that the entity was saved to a database.	
`lastReadDate`	The last time that the entity was retrieved from the database.	
`processId`	The ID of the process.	
`processInstanceByteArray`	The state of a process instance in form of a binary data set.	NOT NULL
`startDate`	The start time of the process.	
`state`	An integer representing the state of a process instance.	NOT NULL

Field	Description	Nullable
OPTLOCK	A version field containing a lock value.	

ProcessInstanceInfo has a 1:N relationship to the **EventTypes** entity.

The **EventTypes** entity contains the following data:

Table 14.3. EventTypes

Field	Description	Nullable
instanceId	A reference to the **ProcessInstanceInfo** primary key and foreign key constraint on this column.	NOT NULL
element	A finished event in the process.	

14.4. WORK ITEM PERSISTENCE ENTITIES

Work items are persisted as **workiteminfo** entities, which persist the state of the particular work item instance on runtime and store the following data:

Table 14.4. WorkItemInfo

Field	Description	Nullable
workItemId	The primary key.	NOT NULL
name	The name of the work item.	
processInstanceId	The (primary key) ID of the process. There is no foreign key constraint on this field.	NOT NULL
state	The state of a work item.	NOT NULL
OPTLOCK	A version field containing a lock value.	
workitembytearay	The work item state in as a binary data set.	NOT NULL

14.5. CORRELATION KEY ENTITIES

The **CorrelationKeyInfo** entity contains information about the correlation key assigned to the given process instance. This table is optional. Use it only when you require correlation capabilities.

Table 14.5. CorrelationKeyInfo

Field	Description	Nullable
keyId	The primary key.	NOT NULL
name	The assigned name of the correlation key.	
processInstanceId	The ID of the process instance which is assigned to the correlation key.	NOT NULL
OPTLOCK	A version field containing a lock value.	

The **CorrelationPropertyInfo** entity contains information about correlation properties for a correlation key assigned the process instance.

Table 14.6. CorrelationPropertyInfo

Field	Description	Nullable
propertyId	The primary key.	NOT NULL
name	The name of the property.	
value	The value of the property.	NOT NULL
OPTLOCK	A version field containing a lock value.	
correlationKey_keyId	A foreign key mapped to the correlation key.	NOT NULL

14.6. CONTEXT MAPPING ENTITY

The **ContextMappingInfo** entity contains information about the contextual information mapped to a **KieSession**. This is an internal part of **RuntimeManager** and can be considered optional when **RuntimeManager** is not used.

Table 14.7. ContextMappingInfo

Field	Description	Nullable
mappingId	The primary key.	NOT NULL

Field	Description	Nullable
CONTEXT_ID	The context identifier.	NOT NULL
KSESSION_ID	The **KieSession** identifier.	NOT NULL
OPTLOCK	A version field containing a lock value.	
OWNER_ID	Holds the identifier of the deployment unit that the given mapping is associated with	

14.7. PESSIMISTIC LOCKING SUPPORT

The default locking mechanism for persistence of processes is *optimistic*. With multi-thread high concurrency to the same process instance, this locking strategy can result in bad performance.

This can be changed at runtime to allow the user to set locking on a per process basis and to allow it to be *pessimistic* (the change can be made at a per KIE Session level or Runtime Manager level as well and not just at the process level).

To set a process to use pessimistic locking, use the following configuration in the runtime environment:

```
import org.kie.api.runtime.Environment;
import org.kie.api.runtime.EnvironmentName;
import org.kie.api.runtime.manager.RuntimeManager;
import org.kie.api.runtime.manager.RuntimeManagerFactory;

...

env.set(EnvironmentName.USE_PESSIMISTIC_LOCKING, true); (1)

RuntimeManager manager =
RuntimeManagerFactory.Factory.get().newPerRequestRuntimeManager(environmen
t); (2)
```

1. **env** is an instance of **org.kie.api.runtime.Environment**.
2. Create your Runtime Manager by using this environment.

CHAPTER 15. DEFINE THE LDAP LOGIN DOMAIN

When you are setting up Red Hat Decision Manager to use LDAP for authentication and authorization, define the LDAP login domain. This is because the Git SSH authentication may be using another security domain, in which case you may face authentication failure.

To define the LDAP login domain, use the **`org.uberfire.domain`** system property. For example, on Red Hat JBoss Enterprise Application Platform, add this property in the **`standalone.xml`** file as shown below:

```
<system-properties>
  <!-- other system properties -->
  <property name="org.uberfire.domain" value="LDAPAuth"/>
</system-properties>
```

Ensure that the authenticated user has appropriate roles (**`admin`**,**`analyst`**,**`reviewer`**) associated with it in LDAP.

CHAPTER 16. AUTHENTICATING THIRD-PARTY CLIENTS THROUGH RH-SSO

To use the different remote services provided by Decision Central or by Decision Server, your client, such as curl, wget, web browser, or a custom REST client, must authenticate through the RH-SSO server and have a valid token to perform the requests. To use the remote services, the authenticated user must have the following roles:

- **`rest-all`** for using Decision Central remote services.
- **`kie-server`** for using the Decision Server remote services.

Use the RH-SSO Admin Console to create these roles and assign them to the users that will consume the remote services.

Your client can authenticate through RH-SSO using one of these options:

- Basic authentication, if it is supported by the client
- Token-based authentication

16.1. BASIC AUTHENTICATION

If you enabled basic authentication in the RH-SSO client adapter configuration for both Decision Central and Decision Server, you can avoid the token grant and refresh calls and call the services as shown in the following examples:

- For web based remote repositories endpoint:

```
curl http://admin:password@localhost:8080/decision-
central/rest/repositories
```

- For Decision Server:

```
curl http://admin:password@localhost:8080/kie-execution-
server/services/rest/server/
```

CHAPTER 17. SUPPORTED PROPERTIES

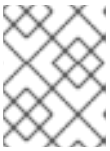
When you install standalone Decision Central, you can use the properties listed in this section in the following command:

```
java -jar rhdm-7.1.0-decision-central-standalone.jar -s application-
config.yaml -D<property>=<value> -D<property>=<value>
```

In this command, **<property>** is a property from the following list and **<value>** is a value that you assign to that property:

- **org.uberfire.nio.git.dir**: Location of the Decision Server Git directory.
- **org.uberfire.nio.git.dirname**: Name of the Decision Server Git directory. Default value: **.niogit**.
- **org.uberfire.nio.git.proxy.ssh.over.http**: Specifies whether SSH should use an HTTP proxy. Default: **false**
- **http.proxyHost**: Defines the host name of the HTTP proxy. Default: **null**
- **http.proxyPort**: Defines the host port (integer value) of the HTTP proxy. Default: **null**
- **org.uberfire.nio.git.proxy.ssh.over.https**: Specifies whether SSH should use an HTTPS proxy. Default: **false**
- **https.proxyHost**: Defines the host name of the HTTPS proxy. Default: **null**
- **https.proxyPort**: Defines the host port (integer value) of the HTTPS proxy. Default: **null**
- **org.uberfire.nio.git.daemon.enabled**: Enables or disables the Git daemon. Default value: **true**.
- **org.uberfire.nio.git.daemon.host**: If the Git daemon is enabled, it uses this property as the local host identifier. Default value: **localhost**.
- **org.uberfire.nio.git.daemon.port**: If the Git daemon is enabled, it uses this property as the port number. Default value: **9418**.
- **org.uberfire.nio.git.ssh.enabled**: Enables or disables the SSH daemon. Default value: **true**.
- **org.uberfire.nio.git.ssh.host**: If the SSH daemon enabled, it uses this property as the local host identifier. Default value: **localhost**.
- **org.uberfire.nio.git.ssh.port**: If the SSH daemon is enabled, it uses this property as the port number. Default value: **8001**.
- **org.uberfire.nio.git.ssh.cert.dir**: Location of the **.security** directory where local certificates are stored. Default: the working directory.
- **org.uberfire.nio.git.ssh.passphrase**: Pass phrase used to access the public key store of your operating system when cloning git repositories with SCP style URLs. Example: **git@github.com:user/repository.git**.

- **`org.uberfire.nio.git.ssh.algorithm`**: Algorithm used by SSH. Default value: **`RSA`**.



NOTE

If you plan to use RSA or any algorithm other than DSA, make sure you set up your application server to use the Bouncy Castle JCE library.

- **`org.uberfire.metadata.index.dir`**: Place where the Lucene **`.index`** directory is stored. Default: the working directory
- **`org.uberfire.ldap.regex.role_mapper`**: Regex pattern used to map LDAP principal names to the application role name. Note that the variable role must be part of the pattern because it is substituted by the application role name when matching a principal value to a role name. Default: Not used.
- **`org.uberfire.sys.repo.monitor.disabled`**: Disables the configuration monitor. Do not disable unless you are sure. Default value: **`false`**
- **`org.uberfire.secure.key`**: Password used by password encryption. Default value: **`org.uberfire.admin`**
- **`org.uberfire.secure.alg`**: Crypto algorithm used by password encryption. Default value: **`PBEWithMD5AndDES`**
- **`org.uberfire.domain`**: Security-domain name used by uberfire. Default value: **`ApplicationRealm`**
- **`org.guvnor.m2repo.dir`**: Place where the Maven repository folder is stored. Default value: **`<working-directory>/repositories/kie`**
- **`org.guvnor.project.gav.check.disabled`**: Disables group ID, artifact ID, and version (GAV) checks. Default value: **`false`**
- **`org.kie.build.disable-project-explorer`**: Disables automatic build of a selected project in Project Explorer. Default value: **`false`**
- **`org.kie.verification.disable-dtable-realtime-verification`**: Disables the real-time validation and verification of decision tables. Default value: **`false`**
- **`org.kie.server.controller`**: URL for connecting with a Decision Manager controller, for example: **`ws://localhost:8080/decision-central/websocket/controller`**
- **`org.kie.server.user`**: User name used to connect with the Decision Server nodes from the Decision Manager controller. This property is only required when using this Decision Central installation as a Decision Manager controller.
- **`org.kie.server.pwd`**: Password used to connect with the Decision Server nodes from the Decision Manager controller. This property is only required when using this Decision Central installation as a Decision Manager controller.

CHAPTER 18. ADDITIONAL RESOURCES

- *Installing and configuring Red Hat Decision Manager on Red Hat JBoss EAP 7.1*
- *Planning a Red Hat Decision Manager installation*
- *Installing and configuring Red Hat Decision Manager on Red Hat JBoss EAP 7.1*
- *Deploying a Red Hat Decision Manager immutable server environment on Red Hat OpenShift Container Platform*
- *Deploying a Red Hat Decision Manager managed server environment on Red Hat OpenShift Container Platform*

APPENDIX A. VERSIONING INFORMATION

Documentation last updated on Friday, October 12, 2018.