Red Hat Process Automation Manager 7.8

Installing and configuring Red Hat Process Automation Manager on Red Hat JBoss Web Server

Last Updated: 2020-09-15

Red Hat Process Automation Manager 7.8 Installing and configuring Red Hat Process Automation Manager on Red Hat JBoss Web Server

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

Legal Notice

Copyright © 2020 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

Abstract

This document describes how to install Red Hat Process Automation Manager 7.8 on Red Hat JBoss Web Server 5.3.1.

Table of Contents

PREFACE 3

CHAPTER 1. ABOUT RED HAT PROCESS AUTOMATION MANAGER 4
1.1. RED HAT PROCESS AUTOMATION MANAGER COMPONENTS 4
1.2. RED HAT PROCESS AUTOMATION MANAGER ROLES AND USERS 5

CHAPTER 2. DOWNLOADING THE RED HAT PROCESS AUTOMATION MANAGER INSTALLATION FILES 7

CHAPTER 3. USING THE RED HAT PROCESS AUTOMATION MANAGER INSTALLER 8
3.1. USING THE INSTALLER IN INTERACTIVE MODE 8
3.2. USING THE INSTALLER IN CLI MODE 10

CHAPTER 4. KIE SERVER ZIP FILE INSTALLATION AND CONFIGURATION 13
4.1. INSTALLING KIE SERVER FROM ZIP FILES 13
4.2. CONFIGURING JDBC WEB SERVER DATA SOURCES 14

CHAPTER 5. SECURING PASSWORDS WITH A KEYSTORE 19

CHAPTER 6. VERIFYING THE KIE SERVER INSTALLATION 22

CHAPTER 7. DOWNLOADING AND INSTALLING THE HEADLESS PROCESS AUTOMATION MANAGER CONTROLLER 23

CHAPTER 8. RUNNING STANDALONE BUSINESS CENTRAL 26
8.1. BUSINESS CENTRAL SYSTEM PROPERTIES 26

CHAPTER 9. MAVEN SETTINGS AND REPOSITORIES FOR RED HAT PROCESS AUTOMATION MANAGER 33
9.1. CONFIGURING MAVEN USING THE PROJECT CONFIGURATION FILE (POM.XML) 33
9.2. MODIFYING THE MAVEN SETTINGS FILE 33
9.3. ADDING MAVEN DEPENDENCIES FOR RED HAT PROCESS AUTOMATION MANAGER 34
9.4. PREPARING A MAVEN MIRROR REPOSITORY FOR OFFLINE USE 37

CHAPTER 10. IMPORTING PROJECTS FROM GIT REPOSITORIES 40

CHAPTER 11. INTEGRATING LDAP AND SSL 41

APPENDIX A. VERSIONING INFORMATION 42

PREFACE

This document describes how to install Red Hat Process Automation Manager 7.8 on JBoss Web Server.

Prerequisites

- You have reviewed the information in *Planning a Red Hat Process Automation Manager installation*.
- You have installed Red Hat JBoss Web Server 5.3.1. For information about installing Red Hat JBoss Web Server, see the *Red Hat JBoss Web Server Installation Guide* .

NOTE

When you run the Red Hat Process Automation Manager installer, the installation path step references to Red Hat JBoss Web Server 5.2 instead of Red Hat JBoss Web Server 5.3. For more information, see the *Installer* known issue in the *Release notes for Red Hat Process Automation Manager 7.8*.

CHAPTER 1. ABOUT RED HAT PROCESS AUTOMATION MANAGER

Red Hat Process Automation Manager is the Red Hat middleware platform for creating business automation applications and microservices. It enables enterprise business and IT users to document, simulate, manage, automate, and monitor business processes and policies. It is designed to empower business and IT users to collaborate more effectively, so business applications can be changed easily and quickly.

Red Hat JBoss Web Server is an enterprise ready web server designed for medium and large applications, based on Tomcat. Red Hat JBoss Web Server provides organizations with a single deployment platform for Java Server Pages (JSP) and Java Servlet technologies, PHP, and CGI.

On a Red Hat JBoss Web Server installation, you can install KIE Server and the headless Process Automation Manager controller. Alternatively, you can run the standalone Business Central JAR file.

The instructions in this document explain how to install Red Hat Process Automation Manager in a Red Hat JBoss Web Server instance.

For instructions on how to install Red Hat Process Automation Manager in other environments, see the following documents:

- *Installing and configuring Red Hat Process Automation Manager on Red Hat JBoss EAP 7.3*
- *Installing and configuring KIE Server on IBM WebSphere Application Server*
- *Installing and configuring KIE Server on Oracle WebLogic Server*
- *Deploying a Red Hat Process Automation Manager immutable server environment on Red Hat OpenShift Container Platform*
- *Deploying a Red Hat Process Automation Manager authoring environment on Red Hat OpenShift Container Platform*
- *Deploying a Red Hat Process Automation Manager freeform managed server environment on Red Hat OpenShift Container Platform*
- *Deploying a Red Hat Process Automation Manager environment on Red Hat OpenShift Container Platform using Operators*

For information about supported components, see the following documents:

- What is the mapping between Red Hat Process Automation Manager and the Maven library version?
- Red Hat Process Automation Manager 7 Supported Configurations

1.1. RED HAT PROCESS AUTOMATION MANAGER COMPONENTS

Red Hat Process Automation Manager is made up of Business Central and KIE Server.

- Business Central is the graphical user interface where you create and manage business rules. You can install Business Central in a Red Hat JBoss EAP instance or on the Red Hat OpenShift Container Platform (OpenShift).

Business Central is also available as a standalone JAR file. You can use the Business Central standalone JAR file to run Business Central without needing to deploy it to an application server.

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

  On a Red Hat JBoss Web Server installation, you can install KIE Server and the headless Process Automation Manager controller. Alternatively, you can run the standalone Business Central JAR file.

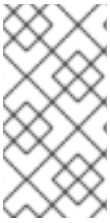
1.2. RED HAT PROCESS AUTOMATION MANAGER ROLES AND USERS

To access Business Central or KIE Server, you must create users and assign them appropriate roles before the servers are started.

The Business Central and KIE Server use Java Authentication and Authorization Service (JAAS) login module to authenticate the users. If both Business Central and KIE Server are running on a single instance, then they share the same JAAS subject and security domain. Therefore, a user, who is authenticated for Business Central can also access KIE Server.

However, if Business Central and KIE Server are running on different instances, then the JAAS login module is triggered for both individually. Therefore, a user, who is authenticated for Business Central, needs to be authenticated separately to access the KIE Server (for example, to view or manage process definitions in Business Central). In case, the user is not authenticated on the KIE Server, then 401 error is logged in the log file, displaying **Invalid credentials to load data from remote server. Contact your system administrator.** message in Business Central.

This section describes available Red Hat Process Automation Manager user roles.



NOTE

The **admin**, **analyst**, **developer**, **manager**, **process-admin**, **user**, and **rest-all** roles are reserved for Business Central. The **kie-server** role is reserved for KIE Server. For this reason, the available roles can differ depending on whether Business Central, KIE Server, or both are installed.

- **admin**: Users with the **admin** role are the Business Central administrators. They can manage users and create, clone, and manage the repositories. They have full access to make required changes in the application. Users with the **admin** role have access to all areas within Red Hat Process Automation Manager.

- **analyst**: Users with the **analyst** role have access to all high-level features. They can model and execute their projects. However, these users cannot add contributors to spaces or delete spaces in the **Design → Projects** view. Access to the **Deploy → Execution Servers** view, which is intended for administrators, is not available to users with the **analyst** role. However, the **Deploy** button is available to these users when they access the Library perspective.

- **developer**: Users with the **developer** role have access to almost all features and can manage rules, models, process flows, forms, and dashboards. They can manage the asset repository, they can create, build, and deploy projects, and they can use Red Hat CodeReady Studio to view processes. Only certain administrative functions such as creating and cloning a new repository are hidden from users with the **developer** role.

- **manager**: Users with the **manager** role can view reports. These users are usually interested in statistics about the business processes and their performance, business indicators, and other business-related reporting. A user with this role has access only to process and task reports.

- **process-admin**: Users with the **process-admin** role are business process administrators. They have full access to business processes, business tasks, and execution errors. These users can also view business reports and have access to the Task Inbox list.

- **user**: Users with the **user** role can work on the Task Inbox list, which contains business tasks that are part of currently running processes. Users with this role can view process and task reports and manage processes.

- **rest-all**: Users with the **rest-all** role can access Business Central REST capabilities.

- **kie-server**: Users with the **kie-server** role can access KIE Server (KIE Server) REST capabilities. This role is mandatory for users to have access to **Manage** and **Track** views in Business Central.

CHAPTER 2. DOWNLOADING THE RED HAT PROCESS AUTOMATION MANAGER INSTALLATION FILES

Depending on your environment and installation requirements, download a Red Hat Process Automation Manager distribution.

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:** Process Automation Manager
   - **Version:** 7.8
2. Download one of the following product distributions, depending on your preferred installation method:

NOTE

You only need to download one of these distributions.

   - If you want to use the installer to install Red Hat Process Automation Manager on Red Hat JBoss Web Server, download **Red Hat Process Automation Manager 7.8.0 Installer** (**rhpam-installer-7.8.0.jar**). The installer graphical user interface guides you through the installation process.
   - To install KIE Server on Red Hat JBoss Web Server using the deployable ZIP files, download the following files:
     - **Red Hat Process Automation Manager 7.8.0 Add Ons**(**rhpam-7.8.0-add-ons.zip**)
     - **Red Hat Process Automation Manager 7.8.0 Maven Repository**(**rhpam-7.8.0-maven-repository.zip**)
       The ZIP files do not require a graphical user interface so you can install Red Hat Process Automation Manager using SSH.
   - To run Business Central without needing to deploy it to an application server, download **Red Hat Process Automation Manager 7.8.0 Business Central Standalone** (**rhpam-7.8.0-business-central-standalone.jar**).

CHAPTER 3. USING THE RED HAT PROCESS AUTOMATION MANAGER INSTALLER

This section describes how to install KIE Server and the headless Process Automation Manager controller using the installer JAR file. The JAR file is an executable file that installs Red Hat Process Automation Manager in an existing Red Hat JBoss Web Server 5.3.1 server installation. You can run the installer in interactive or command line interface (CLI) mode.

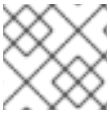
Next steps:

Follow the instructions in one of the following sections:

- Section 3.1, “Using the installer in interactive mode”
- Section 3.2, “Using the installer in CLI mode”

3.1. USING THE INSTALLER IN INTERACTIVE MODE

The installer for Red Hat Process Automation Manager is an executable JAR file. You can use it to install Red Hat Process Automation Manager in an existing Red Hat JBoss Web Server 5.3.1 server installation.



NOTE

For security reasons, you should run the installer as a non-root user.

Prerequisites

- A supported JDK is installed. For a list of supported JDKs, see Red Hat Process Automation Manager 7 Supported Configurations.
- A backed-up Red Hat JBoss Web Server 5.3.1 server installation is available.
- Sufficient user permissions to complete the installation are granted.

NOTE

Ensure that you are logged in with a user that has write permission for Tomcat.

- The JAR binary is included in **$PATH** environment variable. On Red Hat Enterprise Linux, it is included in the **java-$*JAVA_VERSION*-openjdk-devel** package.

NOTE

Red Hat Process Automation Manager is designed to work with UTF-8 encoding. If a different encoding system is used by the underlying JVM, unexpected errors might occur. To ensure UTF-8 is used by the JVM, use the **"-Dfile.encoding=UTF-8"** system property.

Procedure

1. In a terminal window, navigate to the directory where you downloaded the installer JAR file and enter the following command:

```
java -jar rhpam-installer-7.8.0.jar
```

NOTE

When running the installer on Windows, you may be prompted to provide administrator credentials during the installation. To prevent this requirement, add the **izpack.mode=privileged** option to the installation command:

```
java -Dizpack.mode=privileged -jar
rhpam-installer-7.8.0.jar
```

Furthermore, when running the installer on a 32-bit Java virtual machine, you might encounter memory limitations. To prevent this issue, run this command:

```
java -XX:MaxHeapSize=4g -jar
rhpam-installer-7.8.0.jar
```

The graphical installer displays a splash screen and a license agreement page.

2. Click **I accept the terms of this license agreement**and click **Next**.

3. Specify the Red Hat JBoss Web Server 5.3.1 server home where you want to install Red Hat Process Automation Manager and click **Next**.

4. Select the components that you want to install and click **Next**.
   You cannot install Business Central on Red Hat JBoss Web Server. You can only install it on Red Hat JBoss EAP. However, you can install KIE Server and the headless Process Automation Manager controller on Red Hat JBoss Web Server. The headless Process Automation Manager controller is used to manage KIE Server. Install the headless Process Automation Manager controller if you plan to manage multiple KIE Server instances.

5. Create a user and click **Next**. By default, if you install both Business Central and KIE Server in the same container the new user is given the **admin**, **kie-server**, and **rest-all** roles. If you install only KIE Server, the user is given the **kie-server** role. The **kie-server** role is required to access KIE Server REST capabilities.

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).

Make a note of the user name and password. You will need them to access Business Central and KIE Server.

6. On the **Installation Overview** page, click **Next** to start the installation. The Installation Overview page lists the components that you will install.

7. When the installation has completed, click **Next**.

8. If KIE Server is installed, the **Configure Runtime** step appears under **Component Installation**. On the **Configure Runtime Environment** page, choose to perform the default installation or perform an advanced configuration.
   If you choose **Perform advanced configuration**, you can choose to configure database settings or customize certain KIE Server options.

9. If you selected **Customize database settings**, on the **JDBC Drive Configuration** page specify a data source JDBC driver vendor, select one or more Driver JAR files, and click **Next**.
   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 KIE Server to ensure proper data exchange between the servers and the designated database.

10. If you selected **Customize KIE Server properties**, on the **KIE Server Properties Configuration**, change any of the following:

    - **KIE Server ID**: The name of the KIE Server property.
    - Clear any KIE Server function you want to disable.

11. Click **Next** to configure the runtime environment.

12. When **Processing finished** appears at the top of the screen, click **Next** to complete the installation.

13. Optionally, click **Generate Installation Script and Properties File** to save the installation data in XML files, and then click **Done**.
    The installer generates two files. The **auto.xml** file automates future installations and the **auto.xml.variables** file stores user passwords and other sensitive variables. Use the **auto.xml** file on multiple systems to easily repeat a Red Hat Process Automation Manager installation on the same type of server with the same configuration as the original installation. If necessary, update the **installpath** parameter in the **auto.xml** file. Enter the following command to perform an installation with the XML file:

    ```
    java -jar rhpam-installer-7.8.0.jar <path-to-auto.xml-file>
    ```

You have successfully installed Red Hat Process Automation Manager using the installer. If you installed only Business Central, repeat these steps to install KIE Server on a separate server.

NOTE

If you use Microsoft SQL Server, make sure you have configured proper transaction isolation for your database. If you do not, you may experience deadlocks. The recommended configuration is to turn on ALLOW_SNAPSHOT_ISOLATION and READ_COMMITTED_SNAPSHOT by entering the following statements:

```
ALTER DATABASE <DBNAME> SET ALLOW_SNAPSHOT_ISOLATION ON
ALTER DATABASE <DBNAME> SET READ_COMMITTED_SNAPSHOT ON
```

3.2. USING THE INSTALLER IN CLI MODE

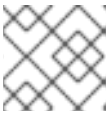
You can run the Red Hat Process Automation Manager installer through the command-line interface (CLI).

NOTE

For security reasons, you should run the installer as a non-root user.

Prerequisites

- A supported JDK is installed. For a list of supported JDKs, see Red Hat Process Automation Manager 7 Supported Configurations.
- A backed-up Red Hat JBoss Web Server 5.3.1 server installation is available.
- Sufficient user permissions to complete the installation are granted.



NOTE

Ensure that you are logged in with a user that has write permission for Tomcat.

- The JAR binary is included in the **$PATH** environment variable. On Red Hat Enterprise Linux, it is included in the **java-$*JAVA_VERSION*-openjdk-devel** package.

NOTE

Red Hat Process Automation Manager is designed to work with UTF-8 encoding. If a different encoding system is used by the underlying JVM, unexpected errors might occur. To ensure UTF-8 is used by the JVM, use the **"-Dfile.encoding=UTF-8"** system property.

Procedure

1. In a terminal window, navigate to the directory where you downloaded the installer file and enter the following command:

```
java -jar rhpam-installer-7.8.0.jar -console
```

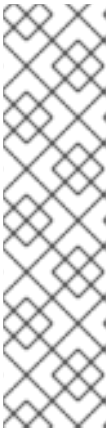
   The command-line interactive process will start and display the End-User License Agreement.

```
press 1 to continue, 2 to quit, 3 to redisplay.
```

2. Read the license agreement, enter **1**, and press Enter to continue:

```
Specify the home directory of one of the following servers:  Red Hat JBoss EAP 7.3 or Red Hat JBoss Web Server 5.3.1
```

3. Enter the parent directory of an existing Red Hat JBoss Web Server 5.3.1 installation. The installer will verify the location of the installation at the location provided. Enter **1** to confirm and continue.
4. Follow the instructions in the installer to complete the installation.



NOTE

When you create the user name and password, make sure that the specified user name does not conflict with any known title of a role or a group. For example, if there is a role called **admin**, you should 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 (*not* including the character **&**).

Make a note of the user name and password. You will need them to access Business Central and KIE Server.

5. When the installation has completed, you will see this message:

```
Would you like to generate an automatic installation script and properties file?
```

6. Enter **y** to create XML files that contain the installation data, or **n** to complete the installation. If you enter **y**, you are prompted to specify a path for the XML files.

7. Enter a path or press the Enter key to accept the suggested path.
The installer generates two files. The **auto.xml** file automates future installations and the **auto.xml.variables** file stores user passwords and other sensitive variables. Use the **auto.xml** file on multiple systems to easily repeat a Red Hat Process Automation Manager installation on the same type of server with the same configuration as the original installation. If necessary, update the **installpath** parameter in the **auto.xml** file. Enter the following command to perform an installation with the XML file:

```
java -jar rhpam-installer-7.8.0.jar <path-to-auto.xml-file>
```

8. If you installed only KIE Server, repeat these steps to install the headless Process Automation Manager controller on a separate server.

NOTE

If you use Microsoft SQL Server, make sure you have configured proper transaction isolation for your database. If you do not, you may experience deadlocks. The recommended configuration is to turn on ALLOW_SNAPSHOT_ISOLATION and READ_COMMITTED_SNAPSHOT by entering the following statements:

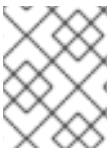
```
ALTER DATABASE <DBNAME> SET ALLOW_SNAPSHOT_ISOLATION ON
ALTER DATABASE <DBNAME> SET READ_COMMITTED_SNAPSHOT ON
```

CHAPTER 4. KIE SERVER ZIP FILE INSTALLATION AND CONFIGURATION

You can install KIE Server using the **rhpam-7.8.0-kie-server-jws.zip** file available from the **Red Hat Process Automation Manager 7.8.0 Add Ons** (**rhpam-7.8.0-add-ons.zip**) file on the Customer Portal and then configure the Java Database Connectivity (JDBC) web server data sources on Red Hat JBoss Web Server .

4.1. INSTALLING KIE SERVER FROM ZIP FILES

KIE Server provides the runtime environment for business assets and accesses the data stored in the assets repository (knowledge store). You can use ZIP files to install KIE Server on an existing Red Hat JBoss Web Server 5.3.1 server instance.



NOTE

To use the installer JAR file to install KIE Server, see Chapter 3, *Using the Red Hat Process Automation Manager installer*.

- The following files have been downloaded, as described in Chapter 2, *Downloading the Red Hat Process Automation Manager installation files*:
  - **Red Hat Process Automation Manager 7.8.0 Add Ons**(**rhpam-7.8.0-add-ons.zip**)
  - **Red Hat Process Automation Manager 7.8.0 Maven Repository**(**rhpam-7.8.0-maven-repository.zip**)
- A backed-up Red Hat JBoss Web Server 5.3.1 server installation is available. The base directory of the Red Hat JBoss Web Server installation is referred to as ***JWS_HOME***.
- Sufficient user permissions to complete the installation are granted.

Procedure

1. Unzip the **rhpam-7.8.0-add-ons.zip** file.
2. From the unzipped **rhpam-7.8.0-add-ons.zip** file, extract the following files:
   - **rhpam-7.8.0-kie-server-jws.zip**
   - **rhpam-7.8.0-process-engine.zip**

   In the following instructions, the directory that contains the extracted **rhpam-7.8.0-kie-server-jws.zip** file is called **JWS_TEMP_DIR** and the directory that contains the extracted **rhpam-7.8.0-process-engine.zip** file is called **ENGINE_TEMP_DIR**.
3. Copy the **JWS_TEMP_DIR/rhpam-7.8.0-kie-server-jws/kie-server.war** directory to the ***JWS_HOME*/tomcat/webapps** directory.

NOTE

Ensure the names of the Red Hat Process Automation Manager deployments you copy do not conflict with your existing deployments in the Red Hat JBoss Web Server instance.

4. Remove the **.war** extensions from the **kie-server.war** folder.

5. Move the **kie-tomcat-integration-7.39.0.Final-redhat-00005.jar** file from the **ENGINE_TEMP_DIR** directory to the ***JWS_HOME*/tomcat/lib** directory.

6. Move the **jboss-jacc-api-<VERSION>.jar**, **slf4j-api-<VERSION>.jar**, and **slf4j-jdk14-<VERSION>.jar** files from the **ENGINE_TEMP_DIR/lib** directory to the ***JWS_HOME*/tomcat/lib** directory, where **<VERSION>** is the version artifact file name, in the **lib** directory.

7. Add the following line to the **<host>** element in the ***JWS_HOME*/tomcat/conf/server.xml** file after the last Valve definition:

```
<Valve className="org.kie.integration.tomcat.JACCValve" />
```

8. Open the ***JWS_HOME*/tomcat/conf/tomcat-users.xml** file in a text editor.

9. Add users and roles to the ***JWS_HOME*/tomcat/conf/tomcat-users.xml** file. In the following example, **<ROLE_NAME>** is a role supported by Red Hat Process Automation Manager. **<USER_NAME>** and **<USER_PWD>** are the user name and password of your choice:

```
<role rolename="<ROLE_NAME>"/>
<user username="<USER_NAME>" password="<USER_PWD>" roles="<ROLE_NAME>"/>
```

   If a user has more than one role, as shown in the following example, separate the roles with a comma:

```
<role rolename="admin"/>
<role rolename="kie-server"/>
<user username="rhpamUser" password="user1234" roles="admin,kie-server"/>
```

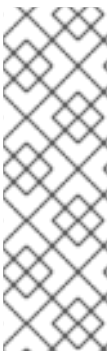
10. Complete one of the following steps in the ***JWS_HOME*/tomcat/bin** directory:

    - On Linux or UNIX, create the **setenv.sh** file with the following content:

```
CATALINA_OPTS="-Xmx1024m -Dorg.jboss.logging.provider=jdk"
```

    - On Windows, add the following content to the **setenv.bat** file:

```
set CATALINA_OPTS=-Xmx1024m -Dorg.jboss.logging.provider=jdk
```



NOTE

If you use Microsoft SQL Server, make sure you have configured proper transaction isolation for your database. If you do not, you may experience deadlocks. The recommended configuration is to turn on ALLOW_SNAPSHOT_ISOLATION and READ_COMMITTED_SNAPSHOT by entering the following statements:

```
ALTER DATABASE <DBNAME> SET ALLOW_SNAPSHOT_ISOLATION ON
ALTER DATABASE <DBNAME> SET READ_COMMITTED_SNAPSHOT ON
```

4.2. CONFIGURING JDBC WEB SERVER DATA SOURCES

Java Database Connectivity (JDBC) is an API specification that is used to connect programs written in Java to the data in popular databases. A data source is an object that enables a 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 KIE Server to ensure proper data exchange between the servers and the designated database.

Prerequisites

- Red Hat Process Automation Manager is installed on Red Hat JBoss Web Server.
- The Red Hat Process Automation Manager 7.8.x Maven Repository (**rhpam-7.8.0-maven-repository.zip**) and the Red Hat Process Automation Manager 7.8.x Add-Ons (**rhpam-7.8.0-add-ons.zip**) files have been downloaded, as described in Chapter 2, *Downloading the Red Hat Process Automation Manager installation files*.
- You want to configure one of the following supported databases and Hibernate dialects:
  - 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**

Procedure

1. Complete the following steps to prepare your database:
   a. Extract **rhpam-7.8.0-add-ons.zip** in a temporary directory, for example **TEMP_DIR**.
   b. Extract **TEMP_DIR/rhpam-7.8.0-migration-tool.zip**.
   c. Change your current directory to the **TEMP_DIR/rhpam-7.8.0-migration-tool/ddl-scripts** directory. This directory contains DDL scripts for several database types.
   d. Import the DDL script for your database type into the database that you want to use, for example:

```
psql jbpm < /ddl-scripts/postgresql/postgresql-jbpm-schema.sql
```

NOTE

If you are using PostgreSQL or Oracle in conjunction with Spring Boot, you must import the respective Spring Boot DDL script, for example **/ddl-scripts/oracle/oracle-springboot-jbpm-schema.sql** or **/ddl-scripts/postgresql/postgresql-springboot-jbpm-schema.sql**.

2. Extract the **rhpam-7.8.0-maven-repository.zip** offline Maven repository file.
3. Copy the following libraries from the extracted offline Maven repository to the **JWS_HOME/tomcat/lib** folder where **VERSION** is the version of that library:

```
org/jboss/spec/javax/transaction/jboss-transaction-api_1.2_spec/{VERSION}/jboss-transaction-api_1.2_spec-{VERSION}.jar
org/jboss/integration/narayana-tomcat/{VERSION}/narayana-tomcat-{VERSION}.jar
org/jboss/narayana/jta/narayana-jta/{VERSION}/narayana-jta-{VERSION}.jar
org/jboss/jboss-transaction-spi/{VERSION}/jboss-transaction-spi-{VERSION}.jar
```

4. Copy your database JDBC driver to the **JWS_HOME/tomcat/lib** folder.
5. Configure the pooling XA data source in the **JWS_HOME/tomcat/conf/context.xml** file:

NOTE

Some of the properties in the following examples might not apply to your database server. Check the documentation for your JDBC driver to determine which properties to set.

a. Configure an XA data source without pooling capabilities. This XA data source is used to create new connections to the target database. In the following example, the XA datasource is **xads** and the variables are defined in Table 4.1, "XA data source variables":

```
<?xml version="1.0" encoding="UTF-8"?>
<Context>
<Resource
auth="Container"
databaseName="${datasource.dbName}"
description="XA Data Source"
factory="org.apache.tomcat.jdbc.naming.GenericNamingResourcesFactory"
loginTimeout="0"
name="xads"
uniqueName="xads"
portNumber="${datasource.port}"
serverName="${datasource.hostname}"
testOnBorrow="false"
type="${datasource.class}"
url="${datasource.url}"
URL="${datasource.url}"
user="${datasource.username}"
password="${datasource.password}"
driverType="4"
schema="${datasource.schema}"
/>
</Context>
```

Table 4.1. XA data source variables

Variable	Description
<datasource.dbName>	The name of the database.

Variable	Description
<datasource.port>	The port number of the database.
<datasource.hostname>	The name of the database host.
<datasource.class>	**XADataSource** class of JDBC driver.
<datasource.url>	The JDBC database connection URL. With some databases, the URL property is **url** and with other databases (for example H2 databases) this property is **URL**.
<datasource.username>	User name for the database connection.
<datasource.password>	Password for the database connection.
<datasource.schema>	The database schema.

b. Configure a pooling data source that relies on the XA data source for creating new connections. In this example, the data source is **poolingXaDs**, **<datasource.username>** is the user name for the database connection, and **<datasource.password>** is the password for the database connection:

```
<?xml version="1.0" encoding="UTF-8"?>
<Context>
<Resource
name="poolingXaDs"
uniqueName="poolingXaDs"
auth="Container"
description="Pooling XA Data Source"
factory="org.jboss.narayana.tomcat.jta.TransactionalDataSourceFactory"
testOnBorrow="true"
transactionManager="TransactionManager"
transactionSynchronizationRegistry="TransactionSynchronizationRegistry"
type="javax.sql.XADataSource"
username="${datasource.username}"
password="${datasource.password}"
xaDataSource="xads"
/>
</Context>
```

The data source is now available under the **java:comp/env/poolingXaDs** JNDI name and passes it to the KIE Server through the **org.kie.server.persistence.ds** system property as described in the next steps.

NOTE

The pooling data source configuration relies on additional resources that have been previously configured in **context.xml** file in **kie-server** application, specifically **TransactionManager** and **TransactionSynchronizationRegistry**.

6. Configure KIE Server to use the data source:

   a. Open one of the following scripts in a text editor:

NOTE

The **setenv.sh** or **setenv.bat** script should already exist. However, if it does not, create it.

   - For Linux or Unix:

```
JWS_HOME/tomcat/bin/setenv.sh
```

   - For Windows:

```
JWS_HOME/tomcat/bin/setenv.bat
```

   b. Add the following properties to **CATALINA_OPS** where **<hibernate.dialect>** is the Hibernate dialect for your database:

```
CATALINA_OPTS="-Xmx1024m
-Dorg.jboss.logging.provider=jdk
-Dorg.kie.server.persistence.ds=java:comp/env/poolingXaDs
-Dorg.kie.server.persistence.tm=JBossTS
-Dorg.kie.server.persistence.dialect=${<hibernate.dialect>}"
```

CHAPTER 5. SECURING PASSWORDS WITH A KEYSTORE

You can use a keystore to encrypt passwords that are used for communication between Business Central and KIE Server. You should encrypt both controller and KIE Server passwords. If Business Central and KIE Server are deployed to different application servers, then both application servers should use the keystore.

Use Java Cryptography Extension KeyStore (JCEKS) for your keystore because it supports symmetric keys. Use KeyTool, which is part of the JDK installation, to create a new JCEKS.

NOTE

If KIE Server is not configured with JCEKS, KIE Server passwords are stored in system properties in plain text form.

Prerequisites

- KIE Server is installed in Red Hat JBoss Web Server.
- Java 8 or higher is installed.

Procedure

1. Open the ***JWS_HOME*/tomcat/conf/tomcat-users.xml** file in a text editor.
2. Add a KIE Server user with the **kie-server** role to the ***JWS_HOME*/tomcat/conf/tomcat-users.xml** file. In the following example, replace **<USER_NAME>** and **<PASSWORD>** with the user name and password of your choice.

```
<role rolename="kie-server"/>
<user username="<USER_NAME>" password="<PASSWORD>" roles="kie-server"/>
```

3. To use KeyTool to create a JCEKS, enter the following command in the Java 8 home directory:

```
$<JAVA_HOME>/bin/keytool -importpassword -keystore <KEYSTORE_PATH> -keypass
<ALIAS_KEY_PASSWORD> -alias <PASSWORD_ALIAS> -storepass
<KEYSTORE_PASSWORD> -storetype JCEKS
```

   In this example, replace the following variables:

   - **<KEYSTORE_PATH>**: The path where the keystore will be stored
   - **<KEYSTORE_PASSWORD>**: The keystore password
   - **<ALIAS_KEY_PASSWORD>**: The password used to access values stored with the alias
   - **<PASSWORD_ALIAS>**: The alias of the entry to the process

4. When prompted, enter the password for the KIE Server user that you created.
5. To set the system properties, complete one of these steps in the ***JWS_HOME*/tomcat/bin** directory and replace the variables as described in the following table:

NOTE

If Business Central or the standalone controller are installed in separate instances from Red Hat JBoss Web Server, do not add the **kie.keystore.key.server.alias** and **kie.keystore.key.server.pwd** properties to **CATALINA_OPTS**.

- On Linux or UNIX, create the **setenv.sh** file with the following content:

```
set CATALINA_OPTS="
-Dkie.keystore.keyStoreURL=<KEYSTORE_URL>
-Dkie.keystore.keyStorePwd=<KEYSTORE_PWD>
-Dkie.keystore.key.server.alias=<KEY_SERVER_ALIAS>
-Dkie.keystore.key.server.pwd=<KEY_SERVER_PWD>
-Dkie.keystore.key.ctrl.alias=<KEY_CONTROL_ALIAS>
-Dkie.keystore.key.ctrl.key.ctrl.pwd=<KEY_CONTROL_PWD>
```

- On Windows, add the following content to the **setenv.bat** file:

```
set CATALINA_OPTS="
-Dkie.keystore.keyStoreURL=<KEYSTORE_URL>
-Dkie.keystore.keyStorePwd=<KEYSTORE_PWD>
-Dkie.keystore.key.server.alias=<KEY_SERVER_ALIAS>
-Dkie.keystore.key.server.pwd=<KEY_SERVER_PWD>
-Dkie.keystore.key.ctrl.alias=<KEY_CONTROL_ALIAS>
-Dkie.keystore.key.ctrl.pwd=<KEY_CONTROL_PWD>
```

Table 5.1. System properties used to load a KIE Server JCEKS

System property	Placeholder	Description
kie.keystore.keyStoreURL	**<KEYSTORE_URL>**	URL for the JCEKS that you want to use, for example **file:///home/kie/keystores/keystore.jceks**
kie.keystore.keyStorePwd	**<KEYSTORE_PWD>**	Password for the JCEKS
kie.keystore.key.server.alias	**<KEY_SERVER_ALIAS>**	Alias of the key for REST services where the password is stored
kie.keystore.key.server.pwd	**<KEY_SERVER_PWD>**	Password of the alias for REST services with the stored password
kie.keystore.key.ctrl.alias	**<KEY_CONTROL_ALIAS>**	Alias of the key for default REST Process Automation Controller where the password is stored
kie.keystore.key.ctrl.pwd	**<KEY_CONTROL_PWD>**	Password of the alias for default REST Process Automation Controller with the stored password

6. Start KIE Server to verify the configuration.

CHAPTER 6. VERIFYING THE KIE SERVER INSTALLATION

Verify that KIE Server is installed correctly.

Prerequisites

- KIE Server is installed and configured.

Procedure

1. To start KIE Server, enter one of the following commands in the ***JWS_HOME*/tomcat/bin** directory:
   - On Linux or UNIX-based systems:

     ```
     $ ./startup.sh
     ```

   - On Windows:

     ```
     startup.bat
     ```

2. After a few minutes, review the files in the ***JWS_HOME*/tomcat/logs** directory and correct any errors.
3. To verify that KIE Server is working correctly, enter **http://localhost:8080/kie-server/services/rest/server** in a web browser.
4. Enter the user name and password stored in the **tomcat-users.xml** file.

CHAPTER 7. DOWNLOADING AND INSTALLING THE HEADLESS PROCESS AUTOMATION MANAGER CONTROLLER

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

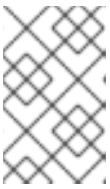
The Process Automation Manager controller is integrated with Business Central. If you install Business Central, use the **Execution Server** page to create and maintain KIE containers. However, if you do not install Business Central, you can install the headless Process Automation Manager controller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

Prerequisites

- The **Red Hat Process Automation Manager 7.8.0 Add Ons**(**rhpam-7.8.0-add-ons.zip**) file has been downloaded, as described in Chapter 2, *Downloading the Red Hat Process Automation Manager installation files*.
- A Red Hat JBoss Web Server 5.3.1 server installation is available. The base directory of the Red Hat JBoss Web Server installation is referred to as ***JWS_HOME***.
- Sufficient user permissions to complete the installation are granted.

Procedure

1. Unzip the **rhpam-7.8.0-add-ons.zip** file. The **rhpam-7.8.0-controller-jws.zip** file is in the unzipped directory.
2. Extract the **rhpam-7.8.0-controller-jws.zip** archive to a temporary directory. In the following examples this directory is called ***TEMP_DIR***.
3. Copy the ***TEMP_DIR*/rhpam-7.8.0-controller-jws.zip/controller.war** directory to the ***JWS_HOME*/tomcat/webapps** directory.



NOTE

Ensure the names of the Red Hat Process Automation Manager deployments you copy do not conflict with your existing deployments in the Red Hat JBoss Web Server instance.

4. Remove the **.war** extensions from the **controller.war** folder.
5. Copy the contents of the ***TEMP_DIR*/rhpam-7.8.0-controller-jws/SecurityPolicy/** directory to ***JWS_HOME*/bin**
6. When asked to overwrite files, select **Yes**.
7. Add the **kie-server** role and user to the ***JWS_HOME*/tomcat/conf/tomcat-users.xml** file. In the following example, **<USER_NAME>** and **<PASSWORD>** are the user name and password of your choice:

```
<role rolename="kie-server"/>
<user username="<USER_NAME>" password="<PASSWORD>" roles="kie-server"/>
```

8. Complete one of the following tasks in the ***JWS_HOME*/tomcat/bin** directory of the instance running KIE Server:

   - On Linux or UNIX, create the **setenv.sh** file with the following content:

     ```
     CATALINA_OPTS="-Xmx1024m
      -Dorg.jboss.logging.provider=jdk
      -Dorg.kie.server.controller.user=<CONTROLLER_USER>
      -Dorg.kie.server.controller.pwd=<CONTROLLER_PWD>
      -Dorg.kie.server.id=<KIE_SERVER_ID>
      -Dorg.kie.server.location=http://<HOST>:<PORT>/kie-server/services/rest/server
      -Dorg.kie.server.controller=http://<HOST>:<PORT>/controller/rest/controller"
     ```

   - On Windows, add the following content to the **setenv.bat** file:

     ```
     set CATALINA_OPTS=-Xmx1024m -Dorg.jboss.logging.provider=jdk
      -Dorg.kie.server.controller.user=<CONTROLLER_USER>
      -Dorg.kie.server.controller.pwd=<CONTROLLER_PWD>
      -Dorg.kie.server.id=<KIE_SERVER_ID>
      -Dorg.kie.server.location=http://<HOST>:<PORT>/kie-server/services/rest/server
      -Dorg.kie.server.controller=http://<HOST>:<PORT>/controller/rest/controller
     ```

     In the preceding examples, replace the following variables:

   - Replace **<CONTROLLER_USER>** and **<CONTROLLER_PWD>** with the user name and password for the **kie-server** role that you defined earlier in this procedure.
   - Replace **<KIE_SERVER_ID>** with a unique identifier.
   - Replace **<CONTROLLER_HOST>:<CONTROLLER_PORT>** with the IP address (host and port) of the controller. If you use the same server for KIE Server and the controller, **<CONTROLLER_HOST>:<CONTROLLER_PORT>** is **localhost:8080**.

9. In the ***JWS_HOME*/tomcat/bin** directory of the instance running the headless Process Automation Manager controller, create a readable **setenv.sh** file with the following content, where **<USERNAME>** is the KIE Server user and **<USER_PWD>** is the password for that user: **CATALINA_OPTS="-Dorg.kie.server.user=<USERNAME> -Dorg.kie.server.pwd=<USER_PWD>"**

10. To start the headless Process Automation Manager controller, enter one of the following commands in the ***JWS_HOME*/tomcat/bin** directory:

    - On Linux or UNIX-based systems:

      ```
      $ ./startup.sh
      ```

    - On Windows:

      ```
      startup.bat
      ```

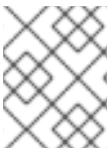
11. After a few minutes, review the ***JWS_HOME*/tomcat/logs** directory and correct any errors.

12. To verify that the headless Process Automation Manager controller is working correctly, enter **http://<CONTROLLER_HOST>:<CONTROLLER_PORT>/controller/rest/controller/management/servers** in a web browser. If you use the same server for KIE Server and the controller, **<CONTROLLER_HOST>:<CONTROLLER_PORT>** is **localhost:8080**.

13. Enter the user name and password stored in the **tomcat-users.xml** file.

CHAPTER 8. RUNNING STANDALONE BUSINESS CENTRAL

You can use the Business Central standalone JAR file to run Business Central without needing to deploy it to an application server such as Red Hat JBoss EAP.



NOTE

Red Hat supports this installation type only when it is installed on Red Hat Enterprise Linux.

NOTE

Example configuration files are available in **rhpam-7.8.0-standalone-sample-configuration.zip** of **rhpam-7.8.0-addons.zip**.

Prerequisites

- The **Red Hat Process Automation Manager 7.8.0 Business Central Standalone**(**rhpam-7.8.0-business-central-standalone.jar**) file has been downloaded from the Software Downloads page for Red Hat Process Automation Manager 7.8, as described in Chapter 2, *Downloading the Red Hat Process Automation Manager installation files*.

Procedure

1. Download the **rhpam-7.8.0-addons.zip** file from the Software Downloads page for Red Hat Process Automation Manager 7.8.
2. Unzip the downloaded **rhpam-7.8.0-addons.zip** to a temporary directory.
3. Navigate to the **rhpam-7.8.0-addons** folder and unzip the **rhpam-7.8.0-standalone-sample-configuration.zip**.
4. Navigate to the **rhpam-7.8.0-standalone-sample-configuration** folder and copy the contents to the directory containing your **rhpam-7.8.0-business-central-standalone.jar** file.
5. To run the standalone sample configuration, enter the following command:

```
java -jar rhpam-7.8.0-business-central-standalone.jar -s sample-standalone-config.yml
```

See Section 8.1, "Business Central system properties" for more information.

8.1. BUSINESS CENTRAL SYSTEM PROPERTIES

The Business Central system properties listed in this section are passed to **standalone*.xml** files. To install standalone Business Central, you can use the listed properties in the following command:

```
java -jar rhpam-7.8.0-business-central-standalone.jar -s application-config.yaml -D<property>=
<value> -D<property>=<value>
```

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

Git directory

Use the following properties to set the location and name for the Business Central Git directory:

- **org.uberfire.nio.git.dir**: Location of the Business Central Git directory.
- **org.uberfire.nio.git.dirname**: Name of the Business Central Git directory. Default value: **.niogit**.
- **org.uberfire.nio.git.ketch**: Enables or disables Git ketch.
- **org.uberfire.nio.git.hooks**: Location of the Git hooks directory.

Git over HTTP

Use the following properties to configure access to the Git repository over HTTP:

- **org.uberfire.nio.git.proxy.ssh.over.http**: Specifies whether SSH should use an HTTP proxy. Default value: **false**.
- **http.proxyHost**: Defines the host name of the HTTP proxy. Default value: **null**.
- **http.proxyPort**: Defines the host port (integer value) of the HTTP proxy. Default value: **null**.
- **http.proxyUser**: Defines the user name of the HTTP proxy.
- **http.proxyPassword**: Defines the user password of the HTTP proxy.
- **org.uberfire.nio.git.http.enabled**: Enables or disables the HTTP daemon. Default value: **true**.
- **org.uberfire.nio.git.http.host**: If the HTTP daemon is enabled, it uses this property as the host identifier. This is an informative property that is used to display how to access the Git repository over HTTP. The HTTP still relies on the servlet container. Default value: **localhost**.
- **org.uberfire.nio.git.http.hostname**: If the HTTP daemon is enabled, it uses this property as the host name identifier. This is an informative property that is used to display how to access the Git repository over HTTP. The HTTP still relies on the servlet container. Default value: **localhost**.
- **org.uberfire.nio.git.http.port**: If the HTTP daemon is enabled, it uses this property as the port number. This is an informative property that is used to display how to access the Git repository over HTTP. The HTTP still relies on the servlet container. Default value: **8080**.

Git over HTTPS

Use the following properties to configure access to the Git repository over HTTPS:

- **org.uberfire.nio.git.proxy.ssh.over.https**: Specifies whether SSH uses an HTTPS proxy. Default value: **false**.
- **https.proxyHost**: Defines the host name of the HTTPS proxy. Default value: **null**.
- **https.proxyPort**: Defines the host port (integer value) of the HTTPS proxy. Default value: **null**.
- **https.proxyUser**: Defines the user name of the HTTPS proxy.
- **https.proxyPassword**: Defines the user password of the HTTPS proxy.
- **user.dir**: Location of the user directory.

- **org.uberfire.nio.git.https.enabled**: Enables or disables the HTTPS daemon. Default value: **false**
- **org.uberfire.nio.git.https.host**: If the HTTPS daemon is enabled, it uses this property as the host identifier. This is an informative property that is used to display how to access the Git repository over HTTPS. The HTTPS still relies on the servlet container. Default value: **localhost**.
- **org.uberfire.nio.git.https.hostname**: If the HTTPS daemon is enabled, it uses this property as the host name identifier. This is an informative property that is used to display how to access the Git repository over HTTPS. The HTTPS still relies on the servlet container. Default value: **localhost**.
- **org.uberfire.nio.git.https.port**: If the HTTPS daemon is enabled, it uses this property as the port number. This is an informative property that is used to display how to access the Git repository over HTTPS. The HTTPS still relies on the servlet container. Default value: **8080**.

JGit

- **org.uberfire.nio.jgit.cache.instances**: Defines the JGit cache size.
- **org.uberfire.nio.jgit.cache.overflow.cleanup.size**: Defines the JGit cache overflow cleanup size.
- **org.uberfire.nio.jgit.remove.eldest.iterations**: Enables or disables whether to remove eldest JGit iterations.
- **org.uberfire.nio.jgit.cache.evict.threshold.duration**: Defines the JGit evict threshold duration.
- **org.uberfire.nio.jgit.cache.evict.threshold.time.unit**: Defines the JGit evict threshold time unit.

Git daemon

Use the following properties to enable and configure the Git daemon:

- **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.hostname**: If the Git daemon is enabled, it uses this property as the local host name 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.http.sslVerify**: Enables or disables SSL certificate checking for Git repositories. Default value: **true**.

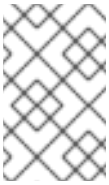
NOTE

If the default or assigned port is already in use, a new port is automatically selected. Ensure that the ports are available and check the log for more information.

Git SSH

Use the following properties to enable and configure the Git SSH daemon:

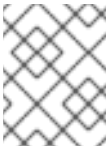
- **org.uberfire.nio.git.ssh.enabled**: Enables or disables the SSH daemon. Default value: **true**.
- **org.uberfire.nio.git.ssh.host**: If the SSH daemon enabled, it uses this property as the local host identifier. Default value: **localhost**.
- **org.uberfire.nio.git.ssh.hostname**: If the SSH daemon is enabled, it uses this property as local host name 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**.



NOTE

If the default or assigned port is already in use, a new port is automatically selected. Ensure that the ports are available and check the log for more information.

- **org.uberfire.nio.git.ssh.cert.dir**: Location of the **.security** directory where local certificates are stored. Default value: Working directory.
- **org.uberfire.nio.git.ssh.idle.timeout**: Sets the SSH idle timeout.
- **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**.
- **org.uberfire.nio.git.gc.limit**: Sets the GC limit.
- **org.uberfire.nio.git.ssh.ciphers**: A comma-separated string of ciphers. The available ciphers are **aes128-ctr**, **aes192-ctr**, **aes256-ctr**, **arcfour128**, **arcfour256**, **aes192-cbc**, **aes256-cbc**. If the property is not used, all available ciphers are loaded.
- **org.uberfire.nio.git.ssh.macs**: A comma-separated string of message authentication codes (MACs). The available MACs are **hmac-md5**, **hmac-md5-96**, **hmac-sha1**, **hmac-sha1-96**, **hmac-sha2-256**, **hmac-sha2-512**. If the property is not used, all available MACs are loaded.



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.

KIE Server nodes and Process Automation Manager controller

Use the following properties to configure the connections with the KIE Server nodes from the Process Automation Manager controller:

- **org.kie.server.controller**: The URL is used to connect to the Process Automation Manager controller. For example, **ws://localhost:8080/business-central/websocket/controller**.

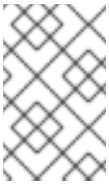
- **org.kie.server.user**: User name used to connect to the KIE Server nodes from the Process Automation Manager controller. This property is only required when using this Business Central installation as a Process Automation Manager controller.

- **org.kie.server.pwd**: Password used to connect to the KIE Server nodes from the Process Automation Manager controller. This property is only required when using this Business Central installation as a Process Automation Manager controller.

Maven and miscellaneous

Use the following properties to configure Maven and other miscellaneous functions:

- **kie.maven.offline.force**: Forces Maven to behave as if offline. If true, disables online dependency resolution. Default value: **false**.



NOTE

Use this property for Business Central only. If you share a runtime environment with any other component, isolate the configuration and apply it only to Business Central.

- **org.uberfire.gzip.enable**: Enables or disables Gzip compression on the **GzipFilter** compression filter. Default value: **true**.

- **org.kie.workbench.profile**: Selects the Business Central profile. Possible values are **FULL** or **PLANNER_AND_RULES**. A prefix **FULL_** sets the profile and hides the profile preferences from the administrator preferences. Default value: **FULL**

- **org.appformer.m2repo.url**: Business Central uses the default location of the Maven repository when looking for dependencies. It directs to the Maven repository inside Business Central, for example, **http://localhost:8080/business-central/maven2**. Set this property before starting Business Central. Default value: File path to the inner **m2** repository.

- **appformer.ssh.keystore**: Defines the custom SSH keystore to be used with Business Central by specifying a class name. If the property is not available, the default SSH keystore is used.

- **appformer.ssh.keys.storage.folder**: When using the default SSH keystore, this property defines the storage folder for the user's SSH public keys. If the property is not available, the keys are stored in the Business Central **.security** folder.

- **appformer.experimental.features**: Enables the experimental features framework. Default value: **false**.

- **org.kie.demo**: Enables an external clone of a demo application from GitHub.

- **org.uberfire.metadata.index.dir**: Place where the Lucene **.index** directory is stored. Default value: 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 a part of the pattern as the application role name substitutes the variable role when matching a principle value and role name.

- **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**.

Process Automation Manager controller

Use the following properties to configure how to connect to the Process Automation Manager controller:

- **org.kie.workbench.controller**: The URL used to connect to the Process Automation Manager controller, for example, **ws://localhost:8080/kie-server-controller/websocket/controller**.
- **org.kie.workbench.controller.user**: The Process Automation Manager controller user. Default value: **kieserver**.
- **org.kie.workbench.controller.pwd**: The Process Automation Manager controller password. Default value: **kieserver1!**.
- **org.kie.workbench.controller.token**: The token string used to connect to the Process Automation Manager controller.

Java Cryptography Extension KeyStore (JCEKS)

Use the following properties to configure JCEKS:

- **kie.keystore.keyStoreURL**: The URL used to load a Java Cryptography Extension KeyStore (JCEKS). For example, **file:///home/kie/keystores/keystore.jceks.**
- **kie.keystore.keyStorePwd**: The password used for the JCEKS.
- **kie.keystore.key.ctrl.alias**: The alias of the key for the default REST Process Automation Manager controller.
- **kie.keystore.key.ctrl.pwd**: The password of the alias for the default REST Process Automation Manager controller.

Rendering

Use the following properties to switch between Business Central and KIE Server rendered forms:

- **org.jbpm.wb.forms.renderer.ext**: Switches the form rendering between Business Central and KIE Server. By default, the form rendering is performed by Business Central. Default value: **false**.

- **org.jbpm.wb.forms.renderer.name**: Enables you to switch between Business Central and KIE Server rendered forms. Default value: **workbench**.

CHAPTER 9. MAVEN SETTINGS AND REPOSITORIES FOR RED HAT PROCESS AUTOMATION MANAGER

When you create a Red Hat Process Automation Manager project, Business Central uses the Maven repositories that are configured for Business Central. You can use the Maven global or user settings to direct all Red Hat Process Automation Manager projects to retrieve dependencies from the public Red Hat Process Automation Manager repository by modifying the Maven project object model (POM) file (**pom.xml**). You can also configure Business Central and KIE Server to use an external Maven repository or prepare a Maven mirror for offline use.

For more information about Red Hat Process Automation Manager packaging and deployment options, see *Packaging and deploying a Red Hat Process Automation Manager project* .

9.1. CONFIGURING MAVEN USING THE PROJECT CONFIGURATION FILE (POM.XML)

To use Maven for building and managing your Red Hat Process Automation Manager projects, you must create and configure the POM file (**pom.xml**). This file holds configuration information for your project. For more information, see Apache Maven Project.

Procedure

1. Generate a Maven project. A **pom.xml** file is automatically generated when you create a Maven project.

2. Edit the **pom.xml** file to add more dependencies and new repositories.
   Maven downloads all of the JAR files and the dependent JAR files from the Maven repository when you compile and package your project.

Find the schema for the **pom.xml** file at http://maven.apache.org/maven-v4_0_0.xsd. For more information about POM files, see Apache Maven Project POM .

9.2. MODIFYING THE MAVEN SETTINGS FILE

Red Hat Process Automation Manager uses Maven **settings.xml** file to configure it's Maven execution. You must create and activate a profile in the **settings.xml** file and declare the Maven repositories used by your Red Hat Process Automation Manager projects.

For information about the Maven **settings.xml** file, see the Apache Maven Project Setting Reference.

Procedure

1. In the **settings.xml** file, declare the repositories that your Red Hat Process Automation Manager projects use. Usually, this is either the online Red Hat Process Automation Manager Maven repository or the Red Hat Process Automation Manager Maven repository that you download from the Red Hat Customer Portal and any repositories for custom artifacts that you want to use.

2. Ensure that Business Central or KIE Server is configured to use the **settings.xml** file. For example, specify the **kie.maven.settings.custom=<SETTINGS_FILE_PATH>** property where **<SETTINGS_FILE_PATH>** is the path to the **settings.xml** file.

On Red Hat JBoss Web Server, for KIE Server add **-Dkie.maven.settings.custom=<SETTINGS_FILE_PATH>** to the **CATALINA_OPTS** section of the **setenv.sh** (Linux) or **setenv.bat** (Windows) file. For standalone Business Central, enter the following command:

```
java -jar rhpam-7.8.0-business-central-standalone.jar -s application-config.yaml -Dkie.maven.settings.custom=<SETTINGS_FILE_PATH>
```

9.3. ADDING MAVEN DEPENDENCIES FOR RED HAT PROCESS AUTOMATION MANAGER

To use the correct Maven dependencies in your Red Hat Process Automation Manager project, add the Red Hat Business Automation bill of materials (BOM) files to the project's **pom.xml** file. The Red Hat Business Automation BOM applies to both Red Hat Decision Manager and Red Hat Process Automation Manager. When you add the BOM files, the correct versions of transitive dependencies from the provided Maven repositories are included in the project.

For more information about the Red Hat Business Automation BOM, see What is the mapping between Red Hat Process Automation Manager and the Maven library version?.

Procedure

1. Declare the Red Hat Business Automation BOM in the **pom.xml** file:

```
<dependencyManagement>
 <dependencies>
  <dependency>
   <groupId>com.redhat.ba</groupId>
   <artifactId>ba-platform-bom</artifactId>
   <version>7.8.0.redhat-00005</version>
   <type>pom</type>
   <scope>import</scope>
  </dependency>
 </dependencies>
</dependencyManagement>
<dependencies>
<!-- Your dependencies -->
</dependencies>
```

2. Declare dependencies required for your project in the **<dependencies>** tag. After you import the product BOM into your project, the versions of the user-facing product dependencies are defined so you do not need to specify the **<version>** sub-element of these **<dependency>** elements. However, you must use the **<dependency>** element to declare dependencies which you want to use in your project.

3. For standalone projects that are not authored in Business Central, specify all dependencies required for your projects. In projects that you author in Business Central, the basic decision engine and process engine dependencies are provided automatically by Business Central.

   - For a basic Red Hat Process Automation Manager project, declare the following dependencies, depending on the features that you want to use:

     Embedded process engine dependencies

     ```
     <!-- Public KIE API -->
     ```

```
<dependency>
  <groupId>org.kie</groupId>
  <artifactId>kie-api</artifactId>
</dependency>

<!-- Core dependencies for process engine -->
<dependency>
  <groupId>org.jbpm</groupId>
  <artifactId>jbpm-flow</artifactId>
</dependency>

<dependency>
  <groupId>org.jbpm</groupId>
  <artifactId>jbpm-flow-builder</artifactId>
</dependency>

<dependency>
  <groupId>org.jbpm</groupId>
  <artifactId>jbpm-bpmn2</artifactId>
</dependency>

<dependency>
  <groupId>org.jbpm</groupId>
  <artifactId>jbpm-runtime-manager</artifactId>
</dependency>

<dependency>
  <groupId>org.jbpm</groupId>
  <artifactId>jbpm-persistence-jpa</artifactId>
</dependency>

<dependency>
  <groupId>org.jbpm</groupId>
  <artifactId>jbpm-query-jpa</artifactId>
</dependency>

<dependency>
  <groupId>org.jbpm</groupId>
  <artifactId>jbpm-audit</artifactId>
</dependency>

<dependency>
  <groupId>org.jbpm</groupId>
  <artifactId>jbpm-kie-services</artifactId>
</dependency>

<!-- Dependency needed for default WorkItemHandler implementations. -->
<dependency>
  <groupId>org.jbpm</groupId>
  <artifactId>jbpm-workitems-core</artifactId>
</dependency>

<!-- Logging dependency. You can use any logging framework compatible with slf4j. -->
<dependency>
  <groupId>ch.qos.logback</groupId>
```

```
  <artifactId>logback-classic</artifactId>
  <version>${logback.version}</version>
</dependency>
```

- For a Red Hat Process Automation Manager project that uses CDI, you typically declare the following dependencies:

  CDI-enabled process engine dependencies

```
<dependency>
  <groupId>org.kie</groupId>
  <artifactId>kie-api</artifactId>
</dependency>

<dependency>
  <groupId>org.jbpm</groupId>
  <artifactId>jbpm-kie-services</artifactId>
</dependency>

<dependency>
  <groupId>org.jbpm</groupId>
  <artifactId>jbpm-services-cdi</artifactId>
</dependency>
```

- For a basic Red Hat Process Automation Manager project, declare the following dependencies:

  Embedded decision engine dependencies

```
<dependency>
  <groupId>org.drools</groupId>
  <artifactId>drools-compiler</artifactId>
</dependency>

<!-- Dependency for persistence support. -->
<dependency>
  <groupId>org.drools</groupId>
  <artifactId>drools-persistence-jpa</artifactId>
</dependency>

<!-- Dependencies for decision tables, templates, and scorecards.
For other assets, declare org.drools:business-central-models-* dependencies. -->
<dependency>
  <groupId>org.drools</groupId>
  <artifactId>drools-decisiontables</artifactId>
</dependency>
<dependency>
  <groupId>org.drools</groupId>
  <artifactId>drools-templates</artifactId>
</dependency>
<dependency>
  <groupId>org.drools</groupId>
  <artifactId>drools-scorecards</artifactId>
</dependency>
```

```
<!-- Dependency for loading KJARs from a Maven repository using KieScanner. -->
<dependency>
  <groupId>org.kie</groupId>
  <artifactId>kie-ci</artifactId>
</dependency>
```

- To use the KIE Server, declare the following dependencies:

  Client application KIE Server dependencies

  ```
  <dependency>
    <groupId>org.kie.server</groupId>
    <artifactId>kie-server-client</artifactId>
  </dependency>
  ```

- To create a remote client for Red Hat Process Automation Manager, declare the following dependency:

  Client dependency

  ```
  <dependency>
    <groupId>org.uberfire</groupId>
    <artifactId>uberfire-rest-client</artifactId>
  </dependency>
  ```

- When creating a JAR file that includes assets, such as rules and process definitions, specify the packaging type for your Maven project as **kjar** and use **org.kie:kie-maven-plugin** to process the **kjar** packaging type located under the **<project>** element. In the following example, **${kie.version}** is the Maven library version listed in What is the mapping between Red Hat Process Automation Manager and the Maven library version?:

  ```
  <packaging>kjar</packaging>
  <build>
   <plugins>
    <plugin>
     <groupId>org.kie</groupId>
     <artifactId>kie-maven-plugin</artifactId>
     <version>${kie.version}</version>
     <extensions>true</extensions>
    </plugin>
   </plugins>
  </build>
  ```

9.4. PREPARING A MAVEN MIRROR REPOSITORY FOR OFFLINE USE

If your Red Hat Process Automation Manager deployment does not have outgoing access to the public Internet, you must prepare a Maven repository with a mirror of all the necessary artifacts and make this repository available to your environment.

NOTE

You do not need to complete this procedure if your Red Hat Process Automation Manager deployment is connected to the Internet.

Prerequisites

- A computer that has outgoing access to the public Internet is available.

Procedure

1. On the computer that has an outgoing connection to the public Internet, complete the following steps:
   a. Click **Red Hat Process Automation Manager 7.8.0 Offliner Content List** to download the **rhpam-7.8.0-offliner.zip** product deliverable file from the Software Downloads page of the Red Hat Customer Portal.
   b. Extract the contents of the **rhpam-7.8.0-offliner.zip** file into any directory.
   c. Change to the directory and enter the following command:

```
./offline-repo-builder.sh offliner.txt
```

   This command creates a **repository** subdirectory and downloads the necessary artifacts into this subdirectory.

   If a message reports that some downloads have failed, run the same command again. If downloads fail again, contact Red Hat support.

2. If you developed services outside Business Central and they have additional dependencies, add the dependencies to the mirror repository. If you developed the services as Maven projects, you can use the following steps to prepare these dependencies automatically. Complete the steps on the computer that has an outgoing connection to the public Internet.
   a. Create a backup of the local Maven cache directory (**~/.m2/repository**) and then clear the directory.
   b. Build the source of your projects using the **mvn clean install** command.
   c. For every project, enter the following command to ensure that Maven downloads all runtime dependencies for all the artifacts generated by the project:

```
mvn -e -DskipTests dependency:go-offline -f /path/to/project/pom.xml --batch-mode -
Djava.net.preferIPv4Stack=true
```

   Replace **/path/to/project/pom.xml** with the correct path to the **pom.xml** file of the project.
   d. Copy the contents of the local Maven cache directory (**~/.m2/repository**) to the **repository** subdirectory that was created.

3. Copy the contents of the **repository** subdirectory to a directory on the computer on which you deployed Red Hat Process Automation Manager. This directory becomes the offline Maven mirror repository.

4. Create and configure a **settings.xml** file for your Red Hat Process Automation Manager deployment, according to instructions in Section 9.2, "Modifying the Maven settings file".

5. Make the following changes in the **settings.xml** file:
   - Under the **<profile>** tag, if a **<repositories>** or **<pluginRepositores>** tag is absent, add the tags as necessary.

- Under **<repositories>** add the following sequence:

```
<repository>
  <id>offline-repository</id>
  <url>file:///path/to/repo</url>
  <releases>
    <enabled>true</enabled>
  </releases>
  <snapshots>
    <enabled>false</enabled>
  </snapshots>
</repository>
```

  Replace /**path**/**to**/**repo** with the full path to the local Maven mirror repository directory.

- Under **<pluginRepositories>** add the following sequence:

```
<repository>
  <id>offline-plugin-repository</id>
  <url>file:///path/to/repo</url>
  <releases>
    <enabled>true</enabled>
  </releases>
  <snapshots>
    <enabled>false</enabled>
  </snapshots>
</repository>
```

  Replace /**path**/**to**/**repo** with the full path to the local Maven mirror repository directory.

CHAPTER 10. IMPORTING PROJECTS FROM GIT REPOSITORIES

Git is a distributed version control system. It implements revisions as commit objects. When you save your changes to a repository, a new commit object in the Git repository is created.

Business Central uses Git to store project data, including assets such as rules and processes. When you create a project in Business Central, it is added to a Git repository that is embedded in Business Central. If you have projects in other Git repositories, you can import those projects into the Business Central Git repository through Business Central spaces.

Prerequisites

- Red Hat Process Automation Manager projects exist in an external Git repository.
- You have the credentials required for read access to that external Git repository.

Procedure

1. In Business Central, click **Menu → Design → Projects**.
2. Select or create the space into which you want to import the projects. The default space is **MySpace**.
3. To import a project, do one of the following:
   - Click **Import Project**.
   - Select **Import Project** from the drop-down list.
4. In the **Import Project** window, enter the URL and credentials for the Git repository that contains the projects that you want to import and click **Import**. The projects are added to the Business Central Git repository and are available from the current space.

CHAPTER 11. INTEGRATING LDAP AND SSL

With Red Hat Process Automation Manager you can integrate LDAP and SSL through RH-SSO. For more information, see the *Red Hat Single Sign-On Server Administration Guide* .

APPENDIX A. VERSIONING INFORMATION

Documentation last updated on Thursday, September 08, 2020.