Red Hat JBoss BRMS 6.0

Administration And Configuration Guide

The Administration guide for Red Hat JBoss BRMS 6

Last Updated: 2017-11-17

Red Hat JBoss BRMS 6.0 Administration And Configuration Guide

The Administration guide for Red Hat JBoss BRMS 6

Red Hat Content Services

Legal Notice

Abstract

A guide for administrators and advanced users dealing with Red Hat JBoss BRMS setup, configuration, and advanced usage.

Table of Contents

CHAPTER 1. INTRODUCTION

1.1. ABOUT RED HAT JBOSS BRMS

Red Hat JBoss BRMS is an open source decision management platform that combines Business Rules Management and Complex Event Processing. It automates business decisions and makes that logic available to the entire business.

Red Hat JBoss BRMS use a centralized repository where all resources are stored. This ensures consistency, transparency, and the ability to audit across the business. Business users can modify business logic without requiring assistance from IT personnel.

Business Resource Planner is included as a technical preview with this release.

Report a bug

1.2. USE CASE: BUSINESS DECISION MANAGEMENT IN THE INSURANCE INDUSTRY WITH JBOSS BRMS

BRMS comprises a high-performance rule engine from the Drools project, a rule repository and easy to use rule authoring tools from the Drools Guvnor project, and Complex Event Processing rule engine extensions from the Drools Fusion project. It also includes Business Resource Planner, a solver for complex planning problems, as a technology preview.

The consumer insurance market is extremely competitive, and it is imperative that customers receive efficient, competitive, and comprehensive services when visiting an online insurance quotation solution. An insurance provider increased revenue from their online quotation solution by upselling to the visitors of the solution relevant, additional products during the quotation process.

JBoss BRMS was integrated with the insurance providers's infrastructure so that when a request for insurance was processed, BRMS was consulted and appropriate additional products were presented with the insurance quotation:

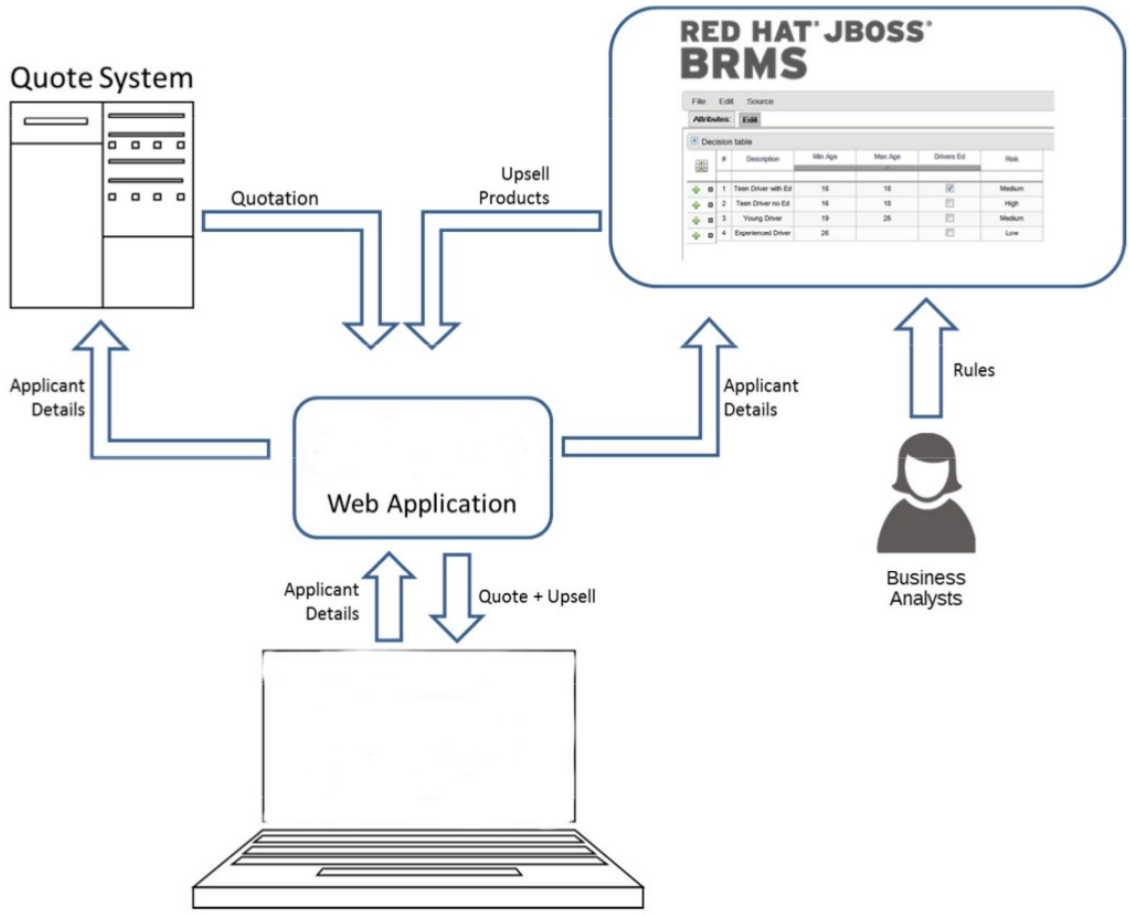


Figure 1.1. BRMS Use Case: Insurance Industry Decision Making

BRMS provided the decision management functionality, i.e. the automatic determination of the products to present to the applicant based on the rules defined by business analysts. The rules were implemented as decision tables, so they could be easily understood and modified without requiring additional support from IT.

Report a bug

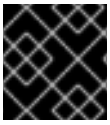
CHAPTER 2. REPOSITORY

Business Rules and other assets and resources created in Business Central are stored in Asset repository, which is otherwise known as the Knowledge Store.

Knowledge store is a centralized repository for your business knowledge. It connects with the GIT repository that allows you to store different kinds of knowledge assets and artifacts at a single location. Business Central provides a web front-end that allows users to view and update the stored content. You can access it using the **Project Explorer** from the unified environment of Red Hat JBoss BRMS.

Report a bug

2.1. CREATING A REPOSITORY



IMPORTANT

Note that only user with the **ADMIN** role can create a repository.

Procedure 2.1. Creating a New Repository

1. Open the **Administration** perspective: on the main menu, click **Authoring** → **Administration**.
2. On the perspective menu, click **Repositories** → **New Repository**.
3. The **Create Repository** pop-up window is displayed.

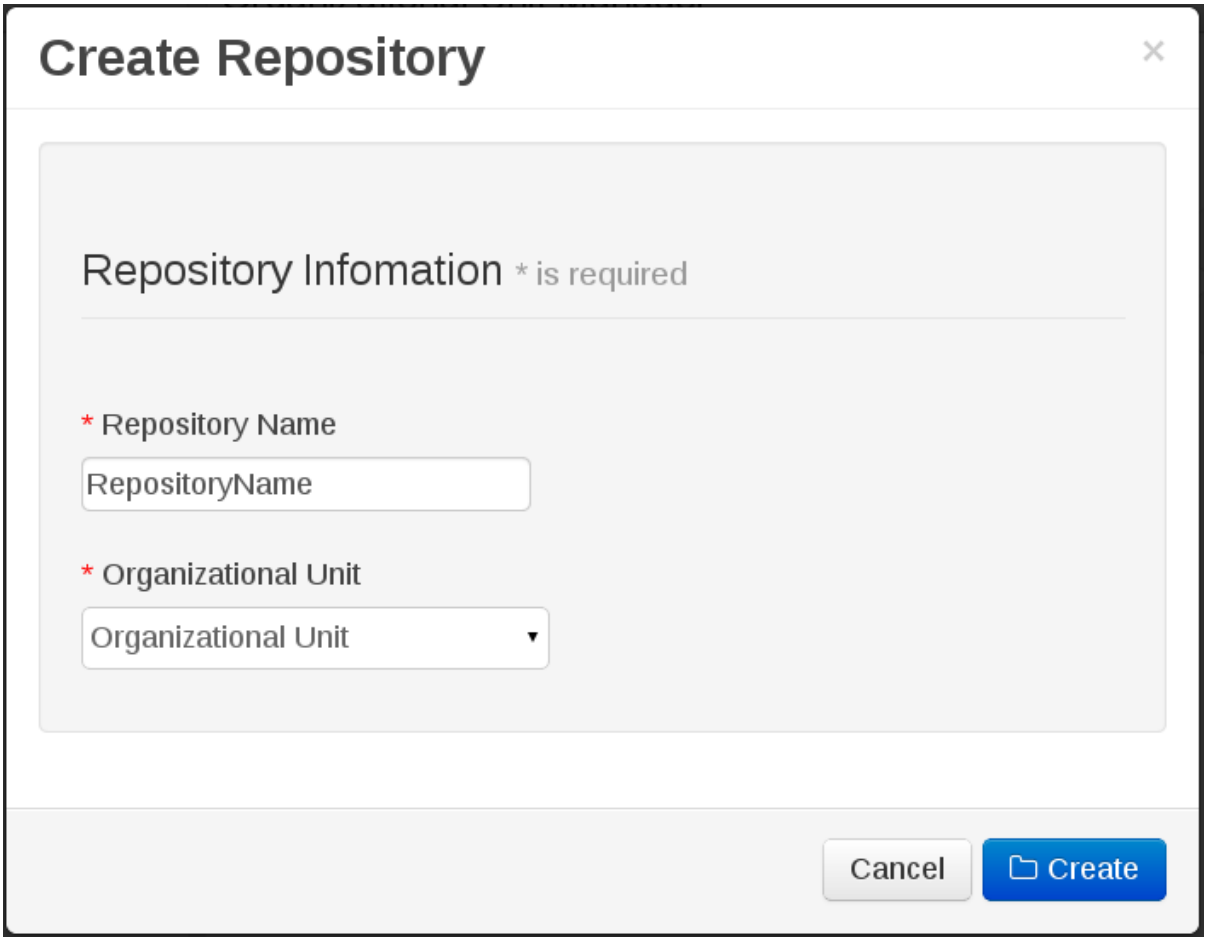


Figure 2.1. Create Repository Pop-up

4. Enter the mandatory details:
   - Repository name.

     NOTE

     Note that the repository name should be a valid filename. Avoid using a space or any special character that might lead to an invalid folder name.

   - Select an organizational unit in which the repository is to be created from the **`Organizational Unit`** drop-down option.
5. Click **`Create`**
6. A confirmation prompt with an **OK** button is displayed which notifies the user that the repository is created successfully. Click **OK**.

The new repository can be viewed either in the **`File Explorer`** or **`Project Explorer`** views.

Report a bug

2.2. CLONING A REPOSITORY

IMPORTANT

Note that only user with the **ADMIN** role can clone a repository.

Procedure 2.2. Cloning a repository

1. Open the **Administration** perspective.
2. On the **Repositories** menu, select **Clone repository**.
3. The **Clone Repository** pop-up window is displayed.

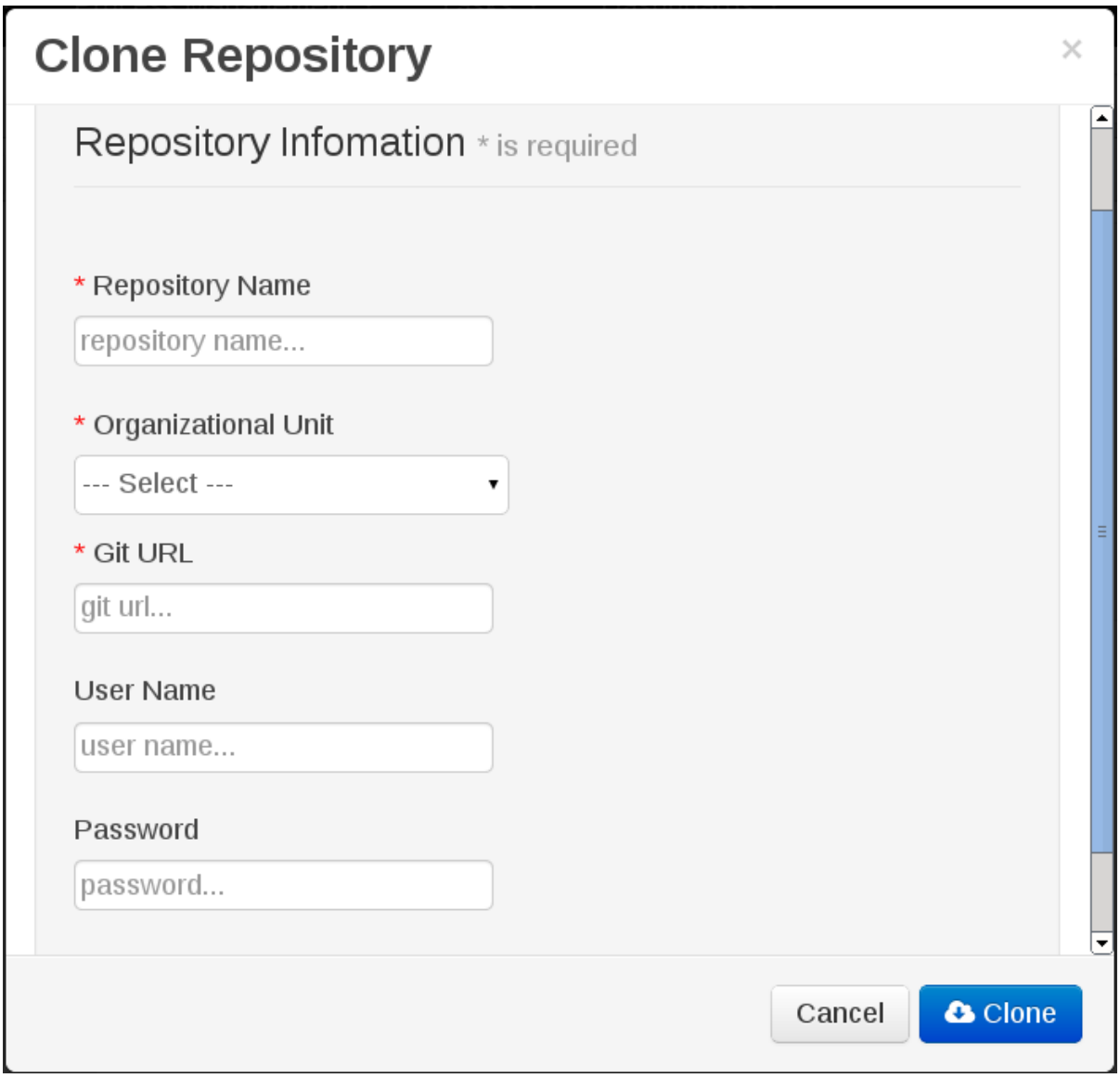


Figure 2.2. Clone Repository Pop-up

4. In the **Clone Repository** dialog window, enter the repository details:
   a. Enter the **Repository Name** to be used as the repository identifier in the Asset repository and select the **Organizational Unit** it should be added to.
   b. Enter the URL of the GIT repository:
      - For a Local Repository: **file:///*path-to-repository/reponame***

- For a Remote or preexisting Repository: **`git://hostname/reponame`**

NOTE

The file protocol is only supported for 'READ' operations. 'WRITE' operations are *not* supported.

c. If applicable, enter the **`User Name`** and **`Password`** to be used for authentication when cloning the repository.

5. Click **`Clone`**.

6. A confirmation prompt with an **OK** button is displayed which notifies the user that the repository is created successfully. Click **OK**.

The cloned repository can be checked either in the **`File Explorer`** or **`Project Explorer`** views.

Report a bug

2.3. MAVEN REPOSITORY

Maven is a software project management tool which uses a project object model (POM) file to manage:

- Builds
- Documentation
- Reporting
- Dependencies
- Releases
- SCMs
- Distribution

A Maven repository is used to hold or store the build artifacts and project dependencies and is generally of two types:

- Local: refers to a local repository where all the project dependencies are stored and is located with the current installation in the default folder as "m2". It is a cache of the remote downloads, and also contains the temporary build artifacts which have not yet been released.
- Remote: refers to any other type of repository that can be accessed by a variety of protocols such as file:// or http://. These repositories can be at a remote location set up by a third-party for downloading of artifacts or an internal repository set up on a file or HTTP server, used to share private artifacts between the development teams for managing internal releases.

Report a bug

2.4. PROJECT

A project is a container for asset packages (rules, decision tables, fact models, data models, and DSLs) that lives in the Knowledge Repository. It is this container that defines the properties of the KIE Base

and KIE Session that are applied to its content. In the GUI, you can edit these entities in the Project Editor.

As a project is a Maven project, it contains the Project Object Model file (`pom.xml`) with information on how to build the output artifact. It also contains the Module Descriptor file, `kmodule.xml`, that contains the KIE Base and KIE Session configuration for the assets in the project.

Report a bug

2.5. CREATING A PROJECT

To create a project, do the following:

1. Open the `Project Authoring` perspective: on the main menu, click **Authoring** → **Project Authoring**.
2. In the `Project Explorer`, select the organizational unit and the repository where you want to create the project.
3. In the perspective menu, go to **New Item** → **Project**.
4. In the `Create new Project` dialog window, define the project details:
   a. In the `Resource Name` text box, enter the project name.

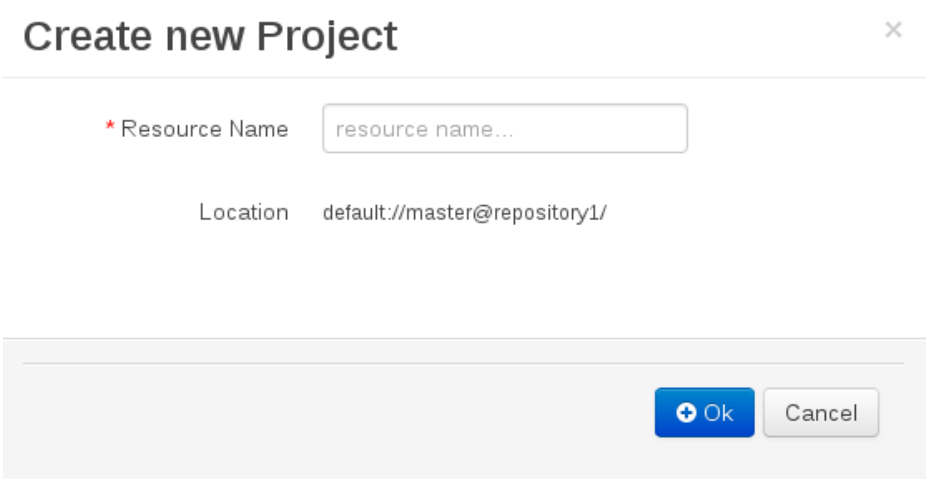


Figure 2.3. New Project Screen

NOTE

Note that the project name should be a valid filename. Avoid using a space or any special character that might lead to an invalid folder name.

5. The explorer refreshes to show a `New Project Wizard` pop-up window.

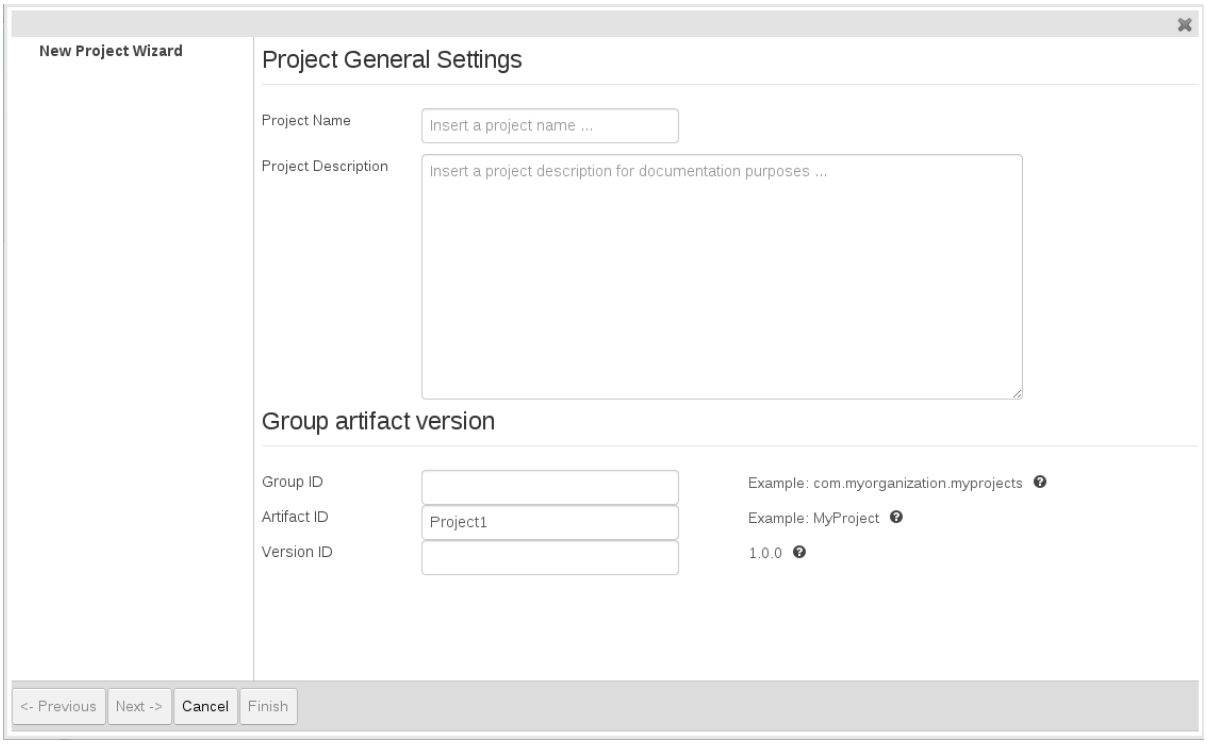


Figure 2.4. New Project Wizard Pop-up

6. Define the **Project General Settings** and **Group artifact version** details for this new project. These parameters are stored inside the **pom.xml** maven configuration file.
   - **Project Name**: The name for the project; for example **Mortgages**
   - **Project Description**: The description of the project which may be useful for the project documentation purpose.
   - **Group ID**: group ID of the project; for example **org.mycompany.commons**
   - **Artifact ID**: artifact ID unique in the group; for example **myframework**
   - **Version ID**: version of the project; for example **2.1.1**

   The **Project Screen** view is updated with the new project details as defined in the **pom.xml** file. Note, that you can switch between project descriptor files in the drop down-box with **Project Settings** and **Knowledge Base Setting**, and edit their contents.

Report a bug

2.6. ADDING DEPENDENCIES

To add dependencies to your project, do the following:

1. Open the Project Editor for the given project:
   a. In the **Project Explorer** view of the **Project Authoring** perspective, open the project directory.
   b. In the perspective menu, go to **Tools** → **Project Editor**.

2. In the **Project Screen** view, select in the **Project Settings** drop-down box the **Dependencies** item.

3. On the updated **Project Screen**, click the **Add** button to add a maven dependency or click the **Add from repository** button to add a dependency from the Knowledge Store (Artifact repository):

    - When adding a maven dependency, a user has to define the **Group ID**, **Artifact ID** and the **Version ID** in the new row which is created in the dependency table.

    - When adding a dependency from the Knowledge Store, select the dependency in the displayed dialog box: the dependency will be added to the dependency table.

4. To apply the various changes, the dependencies must be saved.

Report a bug

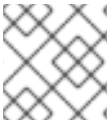
CHAPTER 3. LOGGING

3.1. LOGGING

JBoss Enterprise BRMS Platform provides **logback** functionality for logging configuration.

Accordingly, everything configured is logged to the *Simple Logging Facade for Java*SLF4J, which delegates any log to Logback, Apache Commons Logging, Log4j or java.util.logging. Add a dependency to the logging adaptor for your logging framework of choice. If you're not using any logging framework yet, you can use Logback by adding this Maven dependency:

```
<dependency>
   <groupId>ch.qos.logback</groupId>
   <artifactId>logback-classic</artifactId>
   <version>1.x</version>
</dependency>
```



NOTE

slf4j-nop and **slf4j-simple** are ideal for a light environment.

Report a bug

3.2. CONFIGURING LOGGING

To configure the logging level of the packages, edit the **logback.xml** file located at **business-central.war/WEB-INF/classes/logback.xml**. To set the logging level of the **org.drools** package to "debug" for verbose logging, you would need to add the following line to the file:

```
<configuration>

    <logger name="org.drools" level="debug"/>

    ...

<configuration>
```

Similarly, you can configure logging for packages such as the following:

- org.guvnor
- org.jbpm
- org.kie
- org.slf4j
- org.dashbuilder
- org.uberfire
- org.errai

- etc...

If configuring with **log4j**, the **log4j.xml** can be located at **business-central.war/WEB-INF/log4j.xml** and can be configured in the following way:

```
<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">

    <category name="org.drools">
      <priority value="debug" />
    </category>

    ...

</log4j:configuration>
```

NOTE

Additional logging can be configured in the individual container. To configure logging for Enterprise Application Platform, please refer to the Red Hat JBoss Enterprise Application Platform Administration and Configuration Guide.

Report a bug

CHAPTER 4. COMMAND LINE CONFIGURATION

The **kie-config-cli** tool is a command line configuration tool that provides capabilities to manage the system repository from the command line and can be used in an online or offline mode.

1. **Online mode** (default and recommended) - on startup, the tool connects to a Git repository using a Git server provided by **kie-wb**. All changes are made locally and published to upstream only after explicitly executing the push-changes command. Use the exit command to publish local changes. To discard local changes on exit, use the discard command.

2. **Offline mode** (a kind of installer style) - creates and manipulates the system repository directly on the server (there is no discard option).

The tool is available on the Red Hat Customer Portal . To download the kie-config-cli tool, do the following:

1. Go to the Red Hat Customer Portal and log in.

2. Click **Downloads** → **Products Downloads**.

3. In the **Product Downloads** page that opens, click **Red Hat JBoss BRMS**.

4. From the **Version** drop-down menu, select 6.0.3.

5. In the displayed table, navigate to the **Supplementary Tools** row and then click **Download**.

Extract the zip package for supplementary tools you downloaded from the Red Hat Customer Portal . It contains the directory **kie-config-cli-6.*MINOR_VERSION*-redhat-*x*-dist** with file **kie-config-cli.sh**.

Report a bug

4.1. STARTING THE KIE-CONFIG-CLI TOOL IN ONLINE MODE

1. To start the kie-config-cli tool in online mode, navigate to the **kie-config-cli-6.*MINOR_VERSION*-redhat-*x*-dist** directory where you installed the tool and then execute the following command.

2. In a Unix environment run:

```
./kie-config-cli.sh
```

   In a Windows environment run:

```
./kie-config-cli.bat
```

By default, the tool starts in online mode and asks for a Git URL to connect to (the default value is git://localhost/system). To connect to a remote server, replace the host and port with appropriate values. Example: **git://kie-wb-host:9148/system**

Report a bug

4.2. STARTING THE KIE-CONFIG-CLI TOOL IN OFFLINE MODE

To operate in offline mode, append the offline parameter to the command as below.

1. Navigate to the **kie-config-cli-6.*MINOR_VERSION*-redhat-*x*-dist** directory where you installed the tool.
2. In a Unix environment, run:

```
./kie-config-cli.sh offline
```

In a Windows environment, run:

```
./kie-config-cli.bat offline
```

Executing this command changes the tool's behaviour and displays a request to specify the folder where the system repository (**.niogit**) is located. If .niogit does not yet exist, the folder value can be left empty and a brand new setup is created.

Report a bug

4.3. COMMANDS AVAILABLE FOR THE KIE-CONFIG-CLI TOOL

The following commands are available for managing the GIT repository using the kie-config-cli tool:

- **add-deployment** - adds a new deployment unit
- **add-repo-org-unit** - adds a repository to the organizational unit
- **add-role-org-unit** - adds role(s) to an organizational unit
- **add-role-project** - adds role(s) to a project
- **add-role-repo** - adds role(s) to a repository
- **create-org-unit** - creates new organizational unit
- **create-repo** - creates a new git repository
- **discard** - does not publish local changes, cleans up temporary directories and closes the tool
- **exit** - publishes work, cleans up temporary directories and closes the tool
- **help** - prints available commands with descriptions
- **list-deployment** - lists available deployments
- **list-org-units** - lists available organizational units
- **list-repo** - lists available repositories
- **push-changes** - pushes changes to upstream repository (in online mode only)
- **remove-deployment** - removes existing deployment
- **remove-org-unit** - removes existing organizational unit

- **`remove-repo`** - removes an existing repository from config only
- **`remove-repo-org-unit`** - removes a repository from the organizational unit
- **`remove-role-org-unit`** - removes role(s) from an organizational unit
- **`remove-role-project`** - removes role(s) from a project
- **`remove-role-repo`** - removes role(s) from a repository

Report a bug

CHAPTER 5. MIGRATION

Migrating your projects from BRMS 5 to BPMS 6 requires careful planning and step by step evaluation of the various issues.

Because BPMS 6 uses GIT for storing assets, artifacts and code repositories including processes and rules, you should start by creating an empty project in BPMS 6 as the basis for your migration with dummy files as placeholders for the various assets and artifacts. Running a GIT clone of this empty project into your favorite IDE will initiate the migration process.

Based on the placeholder files in your cloned project, you can start adding assets at the correct locations. The BPMS 6 system is smart enough to pick these changes and apply them correctly. Make sure that when you are importing old rule files that they are imported with the right package name structure.

Since Maven is used for building projects, the projects assets like the rules, processes and models are accessible as a simple jar file.

Let's get started with the simple data migration step, which migrates your BRMS 5 repository from the JCR Jackrabbit to a GIT backend in the new BPMS 6.

Report a bug

5.1. DATA MIGRATION

To migrate data from JBoss BRMS 5, do the following:

1. Download the migration tool by logging in at the Red Hat Customer Portal and then navigating to Red Hat JBoss BPM Suite Software Downloads section. Click on **`Red Hat JBoss BPM Suite Migration Tool`** to download the zip archive.

2. Unzip the downloaded zip archive in a directory of your choice, navigate to this directory in a command prompt and then run the following command:

```
./bin/runMigration.sh -i <source-path> -o <destination-path> -r
<repository-name>
```

Where:

- **`<source-path>`** is a path to a source JCR repository.
- **`<desintation-path>`** is a path to a destination GIT VFS. This folder must not exist already.
- **`<repository-name>`** an arbitrary name for the new repository.

The repository is migrated at the specified destination.

Importing the repository in Business Central

The repository can be imported in business central by cloning it. In the Adminstration perspective, click on the **Repositories** menu and then click on **Clone Repository** menu to start the process.

NOTE

Assets can also be migrated manually. After all, they are all just text files. The BPMN2 specification and the DRL syntax did not change between the different versions.

Importing the repository in JBDS

To import the repository in JBoss Developer Studio, do the following

1. Start JBoss Developer Studio.
2. Start the Red Hat JBoss BPM Suite server (if not already running) by selecting the server from the server tab and click the start icon.
3. Select **File** → **Import...** and navigate to the Git folder. Open the Git folder to select **Projects from Git** and click next.
4. Select the repository source as **Existing local repository** and click next.
5. Select the repository that is to be configured from the list of available repositories.
6. Import the project as a general project in the next window and click next. Name this project and click Finish.

Report a bug

5.2. API AND BACKWARDS COMPATIBILITY

The BPMS 6 system provides backward compatibility with the rule, event and process interactions from BRMS 5. You should eventually migrate (rewrite) these interactions to the all new revamped core API because this backward compatibility is likely to be deprecated.

If you can't migrate your code to use the new API, then you can use the API provided by the purpose built **`knowledge-api`** jar for backwards compatible code. This API is the public interface for working with BPMS and BRMS and is backwards compatible.

If you are instead using the REST API in BRMS 5, note that this has changed as well and there is no mechanism in it for backwards computability.

Report a bug

5.3. MIGRATING TASK SERVICE

BPMS 6 provides support for a locally running task server only. This means that you don't need to setup any messaging service in your project. This differs from BRMS 5 because it provided a task server that was bridged from the core engine by using, most commonly, the messaging system provided by HornetQ.

To help you bridge the gap until you can migrate this in your current architecture, there is a helper or utility method, **`LocalHTWorkItemHandler`**.

Since the TaskService API is part of the public API you will now need to refactor your imports because of package changes and refactor your methods due to API changes themselves.

Report a bug

CHAPTER 6. INTEGRATION

Red Hat JBoss BRMS provides integration modules for integrating BRMS with other frameworks, namely Spring and Apache Camel. These modules are provided in the generic deployment package. See *Installation Guide* for more details.

This section provides information about these modules and how to configure them.

Refer to the Red Hat JBoss BRMS Installation Guide to download the Apache Spring module. You will need to download the generic deployable version of BRMS.

The Spring module is present in the **`jboss-brms-engine.zip`** file and is called **`kie-spring-VERSION-redhat-MINORVERSION.jar`**.

How you intend to use the Spring modules in your application affects how you configure them.

AS A SELF MANAGED PROCESS ENGINE

This is the standard way to start using BRMS in your Spring application. You only configure it once and run as part of the application. Using **`RuntimeManager`** API, perfect synchronization between process engine and task service is managed internally and the end user does not have to deal with the internal code to make these two work together.

AS A SHARED TASK SERVICE

When you use a single instace of a **`TaskService`**, you have more flexibility in configuring the task service instance as it is independent of the **`RuntimeManager`**. Once configured it is then used by the RuntimeManager when requested.

To create a RuntimeEnvironment from your Spring application, you can use the **`org.ie.spring.factorybeans.RuntimeEnvironmentFactoryBean`** class. This factory class is responsible for producing instances of **`RuntimeEnvironment`** that are consumed by RuntimeManager upon creation.

The following RuntimeEnvironment can be created or configured:

- DEFAULT - default (most common) configuration for RuntimeManager
- EMPTY - completely empty environment to be manually populated
- DEFAULT_IN_MEMORY - same as DEFAULT but without persistence of the runtime engine
- DEFAULT_KJAR - same as DEFAULT but knowledge asset are taken from KJAR identified by releaseid or GAV
- DEFAULT_KJAR_CL - build directly from classpath that consists kmodule.xml descriptor

Depending upon the selected type, there are different mandatory properties that are required. However, at least one of the following knowledge properties must be provided:

- knowledgeBase
- assets
- releaseId
- groupId, artifactId, version

Finally, for DEFAULT, DEFAULT_KJAR, DEFAULT_KJAR_CL types, persistence needs to be configured in the form of values for **`entity manager factory`** and **`transaction manager`**.

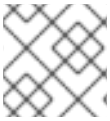
Report a bug

CHAPTER 7. LOCALIZATION AND CUSTOMIZATION

7.1. AVAILABLE LANGUAGES

The Red Hat JBoss BRMS web user interface can be viewed in multiple languages:

- United States English (**en_US**)
- Spanish (**es_ES**)
- Japanese (**ja_JP**)
- Chinese (**zh_CN**)
- Portuguese (**pt_BR**)
- French (**fr_CA**)
- German (**de_DE**)



NOTE

If a language is not specified, US English is used by default.

Report a bug

7.2. CHANGING LANGUAGE SETTINGS

Changing the User Interface Language in Business Central

By default, Business Central uses the system locale. If you need to change it, then append the required locale code at the end of the Business Central URL. For example, the following URL will set the locale to Portuguese (pt_BR).

```
http://localhost:8080/business-central/?locale=pt_BR
```

Report a bug

7.3. RUNNING THE JVM WITH UTF-8 ENCODING

Red Hat JBoss BRMS is designed to work with UTF-8 encoding. If a different encoding system is being used by the JVM, unexpected errors might occur.

To ensure UTF-8 is used by the JVM, use the JVM option "-Dfile.encoding=UTF-8".

Report a bug

7.4. ACCESS CONTROL

Workbench Configuration

Within the Red Hat JBoss BPM Suite, users may set up roles using LDAP to modify existing roles. Users may modify the roles in the workbench configuration to ensure the unique LDAP based roles conform to enterprise standards by editing the deployments directory located at ***$JBOSS_HOME***

business-central.war/WEB-INF/classes/workbench-policy.propeties

Report a bug

CHAPTER 8. MONITORING

8.1. JBOSS OPERATIONS NETWORK

A JBoss Operations Network plug-in can be used to monitor rules sessions for JBoss **BRMS**. The plug-in uses Java Management Extensions (JMX) to monitor rules sessions.

Please refer to the **JBoss Operations Network** *Installation Guide* for installation instructions for the JBoss ON server.

Report a bug

8.2. DOWNLOADING RED HAT JBOSS ON FOR BRMS

1. Go to the Red Hat Customer Portal and log in.
2. Click **Downloads** → **Products Downloads**.
3. In the **Product Downloads** page that opens, click **Red Hat JBoss Operations Network.**.
4. From the **Version** drop-down menu, select version 3.2.0.
5. Select **Red Hat JBoss Operations Network 3.2.0 Base Distribution** and then click **Download**.

Report a bug

8.3. INSTALLING THE BRMS PLUG-IN INTO JBOSS ON

JBoss BRMS plug-in for JBoss Operations Network can be installed by either copying the plug-in JAR files to the JBoss Operations Network plug-in directory or through the JBoss Operations Network GUI.

The following procedure guides a user to copy the plug-in JAR files to the JBoss Operations Network plug-in directory

Procedure 8.1. Copying the JBoss BRMS plug-in JAR files

1. Extract the JBoss BRMS plug-in pack archive to a temporary location. This creates a subdirectory with the name jon-plugin-pack-brms-bpms-3.2.0.GA. For example:

```
[root@server rhq-agent]# unzip jon-plugin-pack-brms-bpms-
3.2.0.GA.zip -d /tmp
```

2. Copy the extracted BRMS plug-in JAR files from the jon-plugin-pack-brms-bpms-3.2.0.GA/ directory to the JBoss ON server plug-in directory. For example:

```
[root@server rhq-agent]# cp /tmp/jon-plugin-pack-brms-bpms-
3.2.0.GA/*.jar /opt/jon/jon-server-3.2.0.GA1/plugins
```

3. Start the JBoss Operations Network server to update the BRMS plug-in.

To upload the BRMS plug-in through the JBoss Operations Network GUI, following is the procedure

Procedure 8.2. Uploading the BRMS plug-in through GUI

1. Start the JBoss Operations Network Server and Log in to access the GUI.
2. In the top navigation of the GUI, open the **Administration** menu.
3. In the **Configuration** area on the left, select the **Server Plugins** link.
4. At the bottom of the list of loaded server plug-ins, click the **Upload a plugin** button and choose the BRMS plugin.
5. The BRMS plug-in for JBoss Operations Network is now uploaded.

Report a bug

8.4. MONITORING KIE BASES AND KIE SESSIONS

In order for JBoss Operations Network to monitor KieBases and KieSessions, MBeans must be enabled.

MBeans can be enabled either by passing the parameter:

```
-kie.mbeans = enabled
```

Or via the API:

```
KieBaseConfiguration kbconf =
KieServices.Factory.get().newKieBaseConfiguration();
    kbconf.setOption(MBeansOption.ENABLED);
```

NOTE

Kie Services have been implemented for BRMS 6; for BRMS 5, **Drools Services** was the naming convention used and it had different measurements on sessions. For example, **activation** → **match** renaming occured in the updated version.

Please refer to the *JBoss Operations Network Resource Monitoring and Operations Reference*guide for information on importing Kie Sessions into the Inventory View for monitoring purposes.

Report a bug

8.5. THE JBOSS RULES KIE BASE MONITORING SERVICE

The JBoss Rules Kie Base Monitoring Service can configure the KieBase by providing the KieBase ID as a connection property.

The JBoss Rules knowledge base monitoring service provides the following operations:

- Start all internal MBeans
- Stop all internal MBeans

Report a bug

8.6. THE JBOSS RULES KIE SESSION MONITORING SERVICE

The JBoss Rules Kie Sessions Monitoring Service can configure the kie base and kie session by providing the kie base ID and kie session ID as connection properties.

The JBoss Rules Kie Session Monitoring Service provides the following operations:

- Reset all metrics counters.
- Return statistics for a specific rules.
- Get statistics for a specific process.
- Get statistics for a specific process instance.

The JBoss Rules Kie Session Monitoring Service provides the following metrics:

- The total number of facts in working memory.
- The total number of matches created since the last reset.
- The total number of matches fired since the last reset.
- The total number of matches canceled since the last reset.
- The total time spent firing rules since the last reset.
- The total number of process instances started since the last reset.
- The total number of process instances completed since the last reset.
- The timestamp of the last reset operation.

Report a bug

APPENDIX A. REVISION HISTORY

Revision 1.0.0-20 **Mon Feb 09 2015** **Vikram Goyal**

Built from Content Specification: 22696, Revision: 742037 by vigoyal