Red Hat JBoss BRMS 6.4

ユーザーガイド

Red Hat JBoss BRMS のユーザーガイド

Last Updated: 2020-05-19

Red Hat JBoss BRMS 6.4 ユーザーガイド

Red Hat JBoss BRMS のユーザーガイド

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

Emily Murphy

Gemma Sheldon

Michele Haglund

Mikhail Ramendik

Stetson Robinson

Vidya Iyengar

法律上の通知

概要

Red Hat JBoss BRMS でビジネスプロセスを定義し、管理する方法についてのガイド

目次

第1章 はじめに

1.1. RED HAT JBOSS BRMS について

Red Hat JBoss BRMS は、ビジネスルール管理と複合イベント処理を組み合わせるオープンソースの意思決定管理プラットフォームであり、ビジネス上の意思決定を自動化し、ビジネスロジックをビジネス全体で利用できるようにします。

Red Hat JBoss BRMS は、すべてのリソースが保存される集中リポジトリーを使用します。これにより、ビジネス全体で一貫性や透明性を維持し、監査を行えます。ビジネスユーザーは、IT 担当者のサポートを受けなくてもビジネスロジックを編集できます。

このリリースには、Business Resource Planner が含まれています。

Red Hat JBoss BRMS は Red Hat Enterprise Linux 7 (RHEL7) での使用がサポートされています。

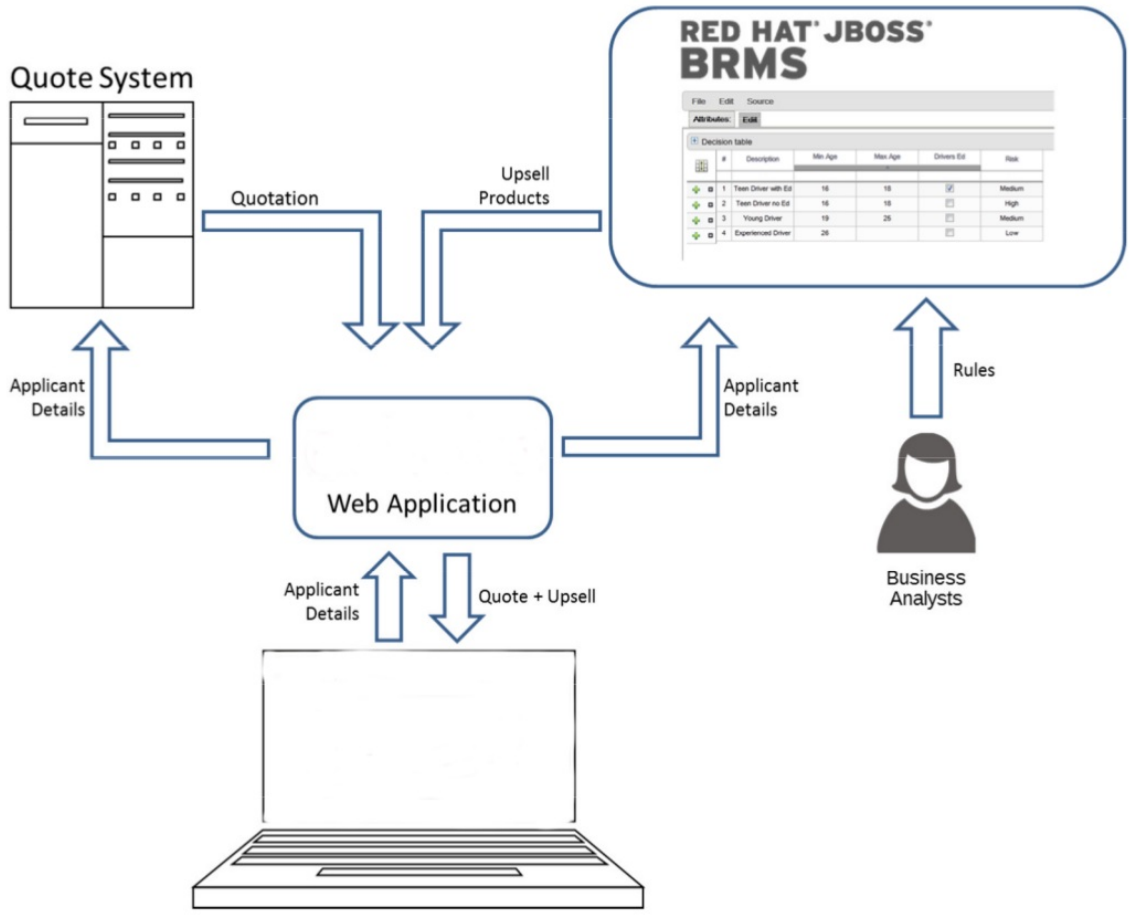
1.2. ユースケース: RED HAT JBOSS BRMS を使用した保険業界における意思決定管理

Red Hat JBoss BRMS は、高性能なルールエンジン、ルールリポジトリー、使いやすいルールオーサリングツール、および複合イベント処理エンジン拡張によって構成されます。以下のユースケースは、これらの JBoss BRMS 機能がどのように保険業界で実装されているかについて説明しています。

消費者向け保険市場は大変競争が激しいため、オンラインの保険見積もりでは効率が良く、競合に負けない包括的なサービスを提供する必要があります。ある保険会社は、オンラインの見積もりプロセスで関連性のある追加製品のアップセルを行って売上を伸ばしました。

以下の図は、JBoss BRMS が保険会社のインフラストラクチャーにどのように統合されているかを示しています。この統合は、保険のリクエストが処理されると JBoss BRMS へ確認が行われ、適切な追加製品が保険の見積もりと共に提示される点で効果的です。

図1.1 JBoss BRMS ユースケース: 保険業界における意思決定



JBoss BRMS は、ビジネスアナリストによって定義されたルールに基づいて申込者に提示する製品の自動決定機能などの意思決定管理機能を提供します。ルールはデシジョンテーブルとして実装されるため、IT 部門の追加サポートを必要とせずに簡単に理解し、変更することができます。

1.3. アセット

アーティファクトリポジトリーにバージョンとして格納できるものはすべてアセットです。たとえば、アセットにはルール、パッケージ、ビジネスプロセス、デシジョンテーブル、ファクトモデル、DSL などがあります。

ルール

ルールは、ルールエンジンがベースとして実行するロジックを提供します。ルールには、名前、属性、ルールの左側にある 'when' ステートメント、ルールの右側にある 'then' ステートメントなどが含まれます。

ビジネスルール

ビジネスルールはビジネスの特定の側面を定義するもので、ビジネス構造を確立したり、ビジネスの振る舞いに影響を及ぼすことを目的としています。ビジネスルールはアクセス制御の問題に重点を置いたもの、またビジネス上の計算や組織のポリシーに関連しているものが多いです。

ビジネスプロセス

ビジネスプロセスとは、ビジネス目標を達成するために必要なステップを説明したフローチャートです (詳細は、『Red Hat JBoss BRMS Business Process Management Guide』を参照してください)。

プロジェクト

プロジェクトは、ナレッジリポジトリーに置かれるアセット (ビジネスプロセス、ルール、作業定義、デシジョンテーブル、ファクトモデル、データモデルおよび DSL) のパッケージ用のコンテナーです。このコンテナーがコンテンツに適用される KIE ベースおよび KIE セッションのプロパティーを定義します。GUI では、Project Editor でこれらのエンティティーを編集することができます。

パッケージ化されたアセットのみをデプロイ可能

プロセスまたはルール定義などのアセットがプロジェクトのパッケージにない場合、それをデプロイすることはできません。そのため、アセットをパッケージとしてまとめる必要があります。また、パッケージの名前は KIE セッション名と同じなければならないことに注意してください。

プロジェクトは Maven プロジェクトであるため、出力アーティファクトをビルドする方法についての情報と共にプロジェクトオブジェクトモデルファイル (**pom.xml**) が含まれます。さらに、プロジェクト内のアセットの KIE ベースおよび KIE セッションの設定を記載するモジュール記述子ファイル **kmodule.xml** も含まれます。

パッケージ

パッケージは、デプロイ可能なアセットのコレクションです。ルールとその他のアセットはパッケージとしてまとめられてからでないとデプロイできません。パッケージの構築時に、パッケージ内のアセットが検証され、デプロイ可能なパッケージにコンパイルされます。

ドメイン固有言語

ドメイン固有言語または DSL は、問題ドメインに特化したルール言語です。

デシジョンテーブル

デシジョンテーブルは、ガイド付きデシジョンテーブルとしてスプレッドシートまたは JBoss BRMS ユーザーインターフェースに格納されたルールのコレクションです。

データモデル

データモデルは、ビジネスドメインに関するファクトのコレクションです。ルールとデータモデルの対話は、ルールベースのアプリケーションで展開されます。

第2章 BUSINESS CENTRAL

Business Central は、Red Hat JBoss BRMS 6 と Red Hat JBoss BPM Suite 6 の両方に使用される Web ベースのユーザーインターフェースです。

これは Business Rules Manager のユーザーインターフェースであり、コアとなる drools エンジンおよびその他のツールと組み合わされています。このインターフェースを使用するビジネスユーザーは、マルチユーザー環境でルールを管理し、制御された方法で変更を実装することができます。

Business Central が使用されるのは以下の場合です。

- ユーザーがルールのバージョン/デプロイメントを管理する必要がある。
- 様々なスキルレベルを持つ複数のユーザーがルールにアクセスし、編集する必要がある。
- ルールを管理するためのインフラストラクチャーが必要である。

Business Central は、ビジネスアナリスト、ルールの専門家、開発者および管理者 (ルール管理者) によって管理されます。

Business Central の主な機能には以下が含まれます。

- 以下を含むルールエディターの複数のタイプ (GUI、テキスト)
  - ガイド付きルールエディター
  - ルールのテンプレート
  - デシジョンテーブル
- 複数のルール「アセット」を 1 つのパッケージに格納
- ドメイン固有言語サポート
- 複合イベント処理サポート
- バージョン管理 (過去のアセット)
- ルールのテスト
- ルールの検証および確認
- カテゴリー化
- 以下を含むビルドおよびデプロイ
  - ChangeSet または KnowledgeBuilder で使用するための、アセットのバイナリーパッケージへのアセンブリー
- アセットの操作に使用する REST API

2.1. BUSINESS CENTRAL へのログイン

サーバーが正常に起動したら、Business Central にログインします。

1. Web ブラウザーで http://localhost:8080/business-central に移動します。ユーザーインターフェースがドメイン名で実行するよう設定されている場合、**localhost** をドメイン名に置き換えます。たとえば、http://www.example.com:8080/business-central のようになります。

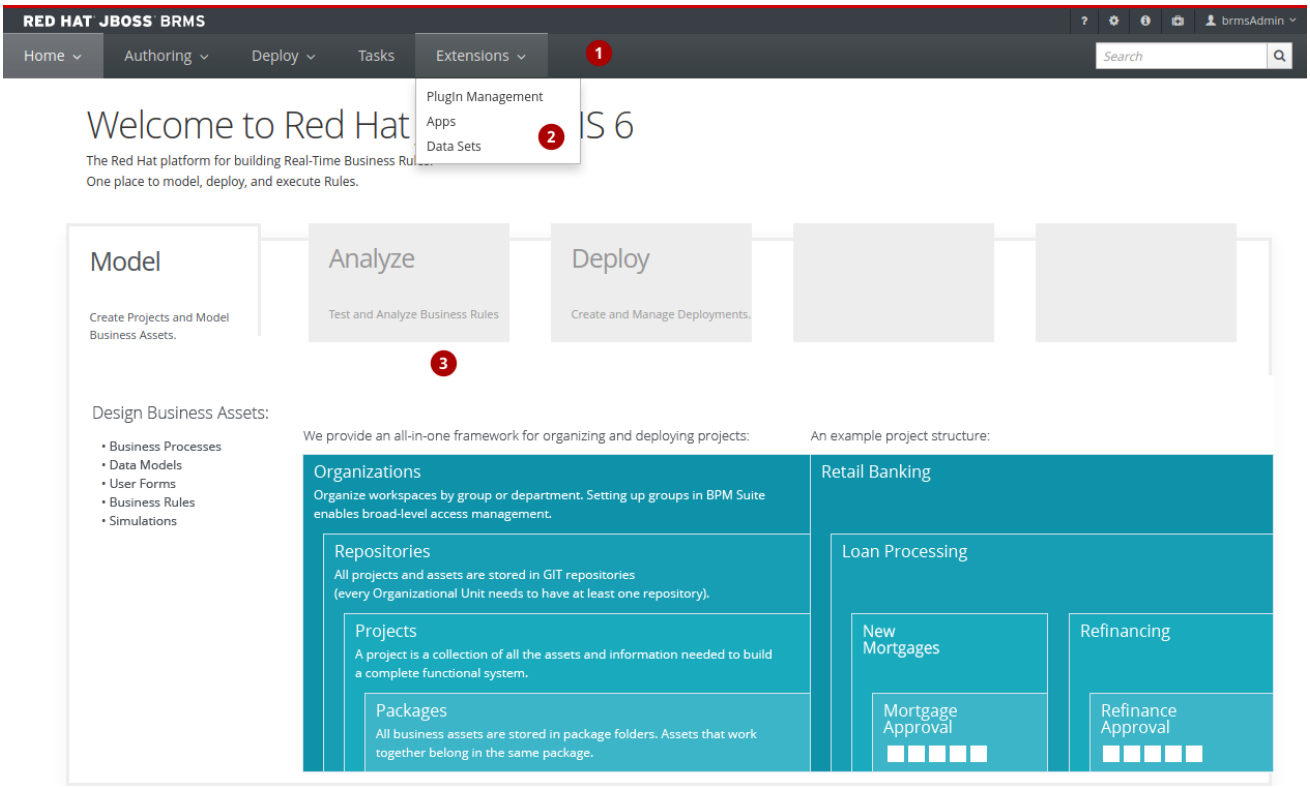
2. インストール時に作成されたユーザー認証情報を使用してログインします (例: ユーザー = **helloworlduser** およびパスワード = **Helloworld@123**)。

2.2. HOME 画面

Home ビューまたは「ランディングページ」はアプリケーションのデフォルトビューです。このビューでは、**Authoring** と **Deployment** という 2 つのメニュー項目が **Home** メニューオプションの横にあります。

以下の画面は Home ビューを示しています。

図2.1 Business Central の Home 画面



メインメニューには、**Home** ページへのリンクとその他すべての利用可能なパースペクティブが含まれます。

パースペクティブメニューには、選択したパースペクティブのメニューが含まれます。

パースペクティブエリアには、ビューやエディターなどのパースペクティブツールが含まれます (ホームページと個々のパースペクティブへのリンクおよびそれらのビューが表示されます)。

2.2.1. パースペクティブ

Business Central には、メインメニューからアクセスできる以下のパースペクティブのグループが表示されます。

- **Authoring** グループ:
  - **Project Authoring** パースペクティブには以下が含まれます。

- **Project Explorer** ビュー。利用可能なリポジトリー構成の概要、およびビジネスプロセス定義、フォームの定義などの利用可能なリソースについての情報が含まれます。
- エディターエリア。**Project Explorer** ビューの右側にあり、リソースが開かれるとそれぞれのエディターが表示されます。
- **Messages** ビュー。検証メッセージが含まれます。

- **Contributors** パースペクティブからは、組織単位、リポジトリー、作成者およびその他の条件でソートされるコミット数を表示できます。
- **Artifact Repository** パースペクティブには、依存関係として追加できる jar の一覧が含まれます。このパースペクティブでは、アーティファクトのアップロード/ダウンロード操作や **pom.xml** ファイルを開く (表示する) 操作を選択できます。
  このビューは **admin** ロールを持つユーザーのみに有効です。
- **Administration** パースペクティブには以下が含まれます。
  - **File Explorer** ビュー。利用可能なアセットリポジトリーが含まれます。
  - エディターエリア。**File Explorer** ビューの右側にあり、リソースが開かれるとそれぞれのエディターが表示されます。
    Administration パースペクティブ。管理者は、ここからナレッジストアをアセットを含むリポジトリーに接続し、新規リポジトリーを作成できます。詳細は、『**Red Hat JBoss BRMS Administration and Configuration Guide**』を参照してください。

    このビューは **admin** ロールを持つユーザーのみに有効です。

- **Deploy** グループ:
  - **Execution Servers** パースペクティブ。このパースペティブには、デプロイ済みの Realtime Decision Server テンプレートとテンプレートに関連付けられたコンテナーの一覧が含まれます。
- **Tasks** グループ:
  - **Task List** パースペクティブ。このパースペクティブには、Process インスタンスの Human Task で生成されたタスクか、手動で生成されたタスクの一覧が含まれます。ログインしたユーザーに割り当てられたタスクのみが表示されます。ここからメンバーとして属するグループに割り当てられたタスクを要求できます。
- **Extensions**
  - **PlugIn Management** パースペクティブ。このパースペクティブから新規の Business Central パースペクティブおよびプラグインをカスタマイズし、作成することができます。
  - **Apps** パースペクティブ。このパースペクティブからカスタムパースペクティブプラグインを参照でき、これらをカテゴリー化し、開くことができます。
  - **Data Sets** パースペクティブ。このパースペクティブから外部データセットを定義し、これらに接続することができます。

2.3. BUSINESS CENTRAL の埋め込み

Business Central は、各種の形式のアセットをオーサリングするために使用するエディターのセットを提供します。アセットの形式に応じて特殊なエディターが使用されます。

Business Central には、スタンドアロンモードでこれを独自の (Web) アプリケーションに埋め込む機能を提供します。これにより、Business Central に切り替えなくても独自のアプリケーションでルール、プロセス、デシジョンテーブルおよびその他のアセットを編集することができます。

Business Central をアプリケーションに埋め込むには、Business Central アプリケーションを web サーバーまたはアプリケーションサーバーで実行する必要があります。次に web アプリケーション内で、以下の表に示されるように iframe を HTTP クエリーパラメーターで設定します。

表2.1 スタンドアロンモードの HTTP クエリーパラメーター

パラメーター名	説明	複数の値を許可	例
standalone	このパラメーターは Business Central をスタンドアロンモードに切り替えます。	no	(なし)
path	編集するアセットへのパス。アセットはすでに存在している必要があることに注意してください。	no	git://master@uf-playground/todo.md
perspective	既存のパースペクティブ名への参照。	no	org.guvnor.m2repo.client.perspectives.GuvnorM2RepoPerspective
header	表示されるヘッダーの名前を定義します (コンテキストメニューのヘッダーの場合に役立ちます)。	yes	ComplementNavArea

以下の例は、Business Central の埋め込み Author パースペクティブを設定する方法について示しています。

```
===test.html===
 <html>
  <head>
    <title>Test</title>
  </head>
  <body>
    <iframe id="ifrm" width="1920" height="1080" src='http://localhost:8080/business-central?
standalone=&perspective=AuthoringPerspective&header=AppNavBar'></iframe>
  </body>
</html>
```

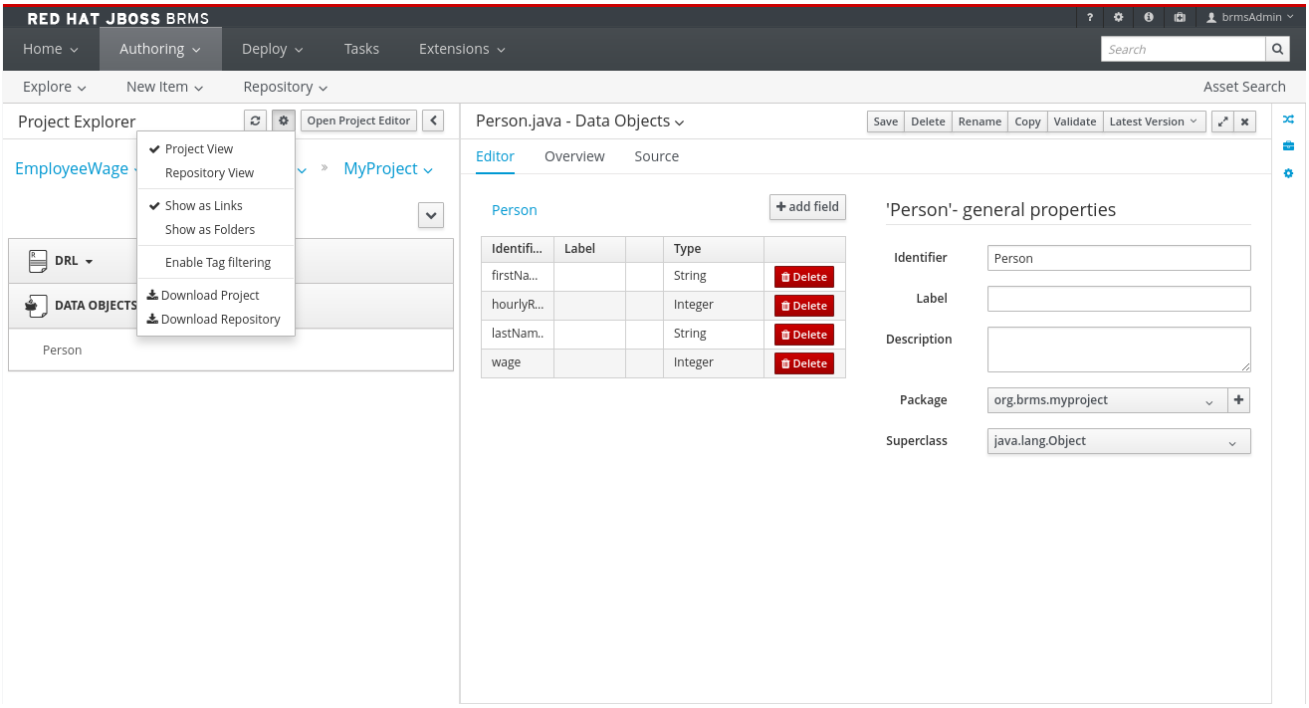
X-frame オプションは business-central の **web.xml** に設定できます。**x-frame-options** のデフォルト値は以下ののようになります。

```
<param-name>x-frame-options</param-name>
  <param-value>SAMEORIGIN</param-value>
```

2.4. プロジェクトのオーサリング

プロジェクトおよび関連付けられたアセットのオーサリングは Project Explorer から実行できます。Project Explorer には、**Authoring → Project Authoring** をクリックして Home 画面からアクセスできます。

図2.2 Project Explorer 画面



Project Authoring 画面は 3 つのセクションに分かれています。

- **Project Explorer**: Project Authoring 画面の左ペインには Project Explorer があり、ここからプロジェクトに移動し、必要なパッケージおよびアセットを作成できます。() ボタンをクリックすると、プロジェクトビューまたはリポジトリービューへのビューを設定できます。プロジェクトの内容は、**Show as Folders** をクリックしてツリービューで移動したり、**Show as Links** をクリックして単一行のパスで移動したりできます。

- **Content** エリア: Content エリアには、編集用に開かれているアセットが表示されます。**Save**、**Delete**、**Rename**、**Copy**、および **Validate** などのボタンを含むツールバーがあり、これを使用して作業対象のアセットに対して必要なアクションを実行できます。

- **Problems**: Problems エリアには、特定のアセットの保存または検証中に発生するプロジェクトの検証エラーが表示されます。

2.4.1. レイアウトの変更

ユーザーはすべてのパネルのレイアウトを変更することができます。それぞれのパネルのサイズを変更したり、位置を変更したりすることができます。ただし、Project Explorer パネルの場合は、サイズ変更は可能でも位置を変更することはできません。

レイアウトのサイズ変更
レイアウトのサイズ変更は以下の方法で行うことができます。

1. 画面の幅のサイズを変更するには、以下を実行します。

   a. 縦方向のパネルスプリッターにマウスポインターを移動します。ポインターは  に切り替わります。

b. スプリッターをドラッグして必要な位置にこれを設定し、画面の幅を調整します。

2. 画面の高さのサイズを変更するには、以下を実行します。

a. 横方向のパネルスプリッターにカーソルを移動します。ポインターは  に切り替わります。

b. スプリッターをドラッグして必要な位置にこれを設定し、画面の高さを調整します。

レイアウトの位置の変更
レイアウトの位置を変更するには、以下を実行します。

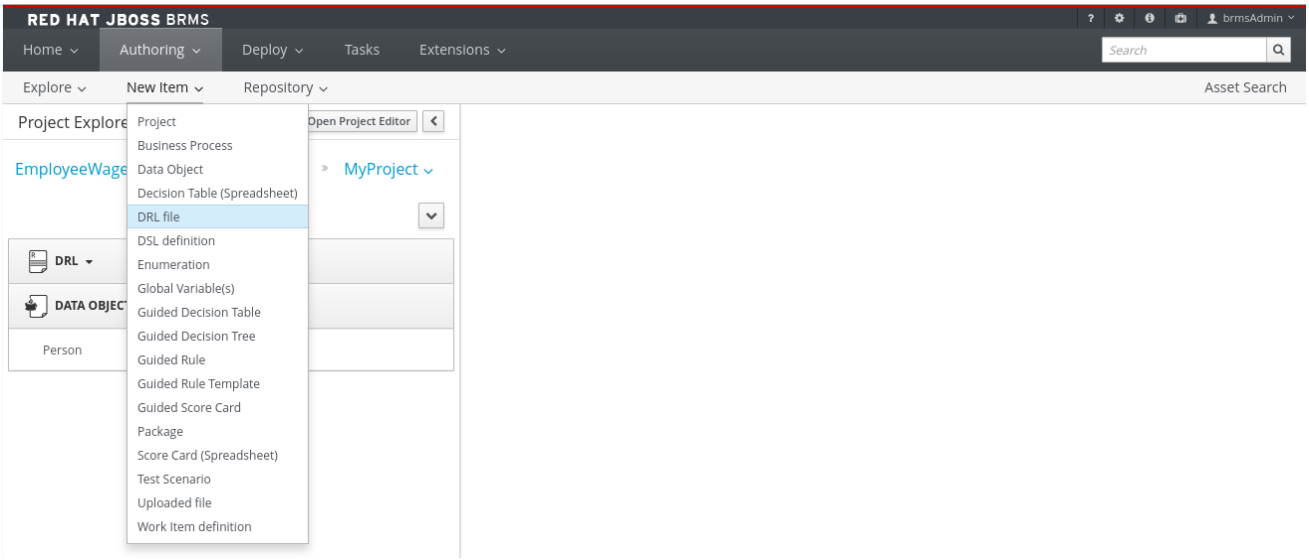
1. パネルのタイトルにマウスポインターを移動します。ポインターは  に切り替わります。

2. マウスの左クリックを押しながら画面を必要な位置にドラッグします。目的の位置を示す  記号が表示され、画面の位置を設定できます。

2.4.2. 新規アセットの作成

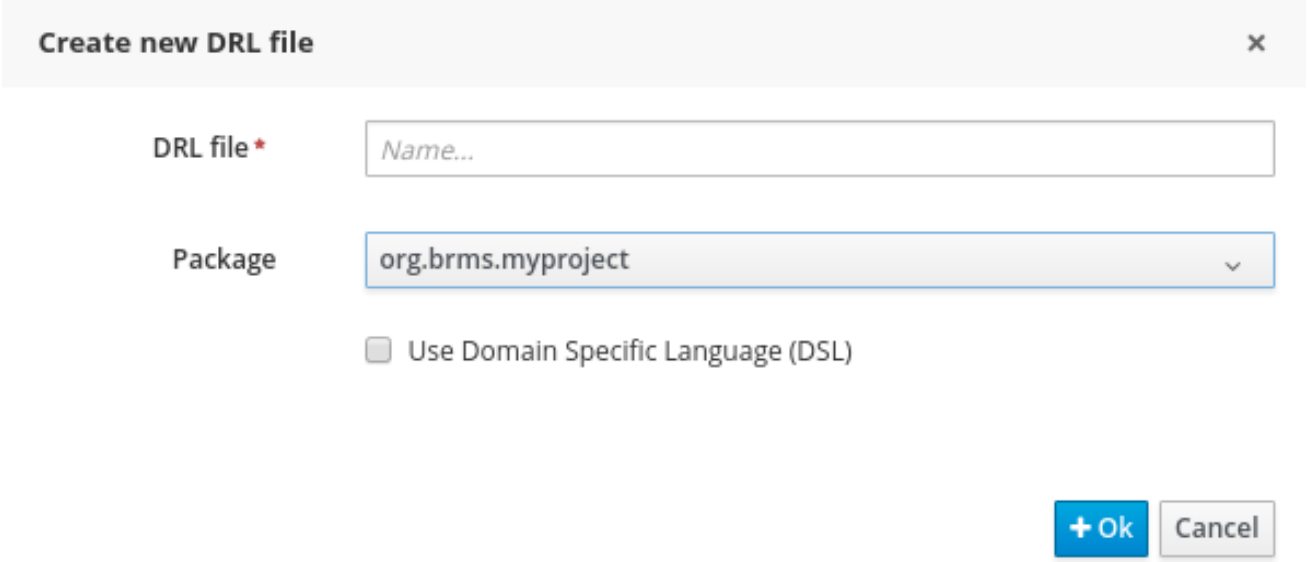
アセットは、**New Item** パースペクティブのメニューオプションを使用して作成できます。

図2.3 新規アセットの作成画面



New Item メニューからアセットをクリックすると、**Create new (Asset-type)** ポップアップダイアログが開かれます。ここでユーザーはアセットの名前を入力できます。

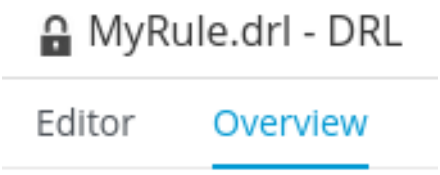
図2.4 新規作成のポップアップダイアログの作成



2.4.3. アセットのメタデータおよびバージョン管理

Business Central 内の大半のアセットには、それらに関連付けられたメタデータおよびバージョン管理情報があります。本セクションでは、このようなアセット (DRL アセット) のメタデータ画面およびバージョン管理について見ていきます。他のアセットのメタデータおよびバージョンの表示と編集についても同様のステップを実行できます。

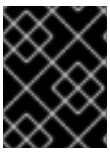
メタデータ管理
DRL アセットのメタデータ画面を開くには、**Overview** タブをクリックします。アセットに **Overview** タブがない場合は、そのアセットには関連付けられたメタデータがないことを意味します。



Overview セクションは **Version history** タブに表示されますが、**Metadata** タブをクリックして実際のメタデータに切り換えることができます。

メタデータセクションでは、アセットの **Tags**、**Subject**、**Type**、**External Link** および **Source metadata** を表示し、編集することができます。ただし、最も重要なメタデータは、説明フィールドで表示/編集できるアセットの説明と、このアセットにアクセスできるユーザーが入力し、表示できるコメントと言えるでしょう。

コメントは、コメントセクションにあるテキストボックスに入力できます。コメントを入力したら Enter を押し、それらをコメントセクションに表示させます。

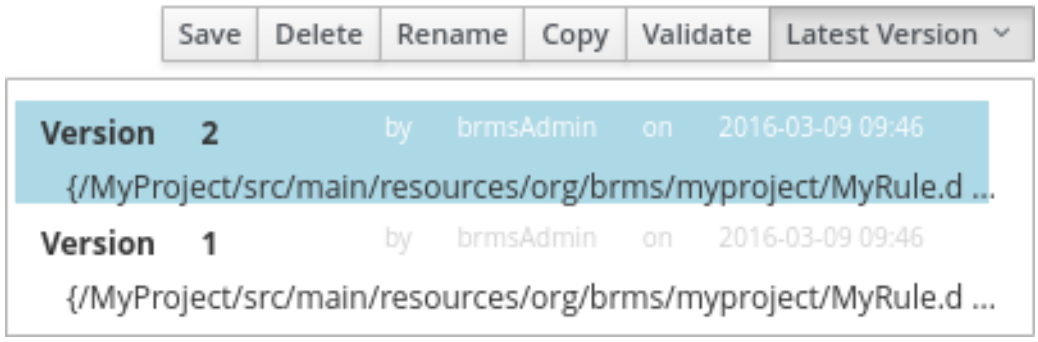


重要

コメントを含むすべてのメタデータを永続化するには、**Save** ボタンを押す必要があります。

バージョン管理
アセットに変更を加えて保存するたびに、アセットの新規バージョンが作成されます。以下の 2 つの方法のいずれかによってバージョン間の切り替えを行うことができます。

- アセットツールバーで **Latest Version** ボタンをクリックし、関連のバージョンを選択します。Business Central でこのバージョンのアセットが読み込まれます。



- または、**Overview** セクションを開きます。**Version history** セクションには利用可能なすべてのバージョンが表示されます。復元するバージョンを **選択** します。

いずれの場合も、**Save** ボタンが **Restore** に切り替わります。このボタンをクリックして変更を永続化します。

2.5. アセットのロックサポート

Business Central での BPM および BRMS アセットの更新時にこれらをロックするためのデフォルトのロックメカニズムは悲観的ロックです。Business Central でアセットを開き、変更する際は常に、複数ユーザーの設定での競合を避けられるようアセットが排他的な使用のために自動的にロックされます。悲観的ロックは、セッションの終了時やアセットを保存するか、または閉じる際に自動的に解放されます。

悲観ロック機能は、複数のユーザーがそれぞれの変更を上書きすることを防止するために提供されています。ただし、別のユーザーがロックしたファイルを編集する必要が生じる場合もあります。この場合、Business Central からロックされたアセットのロック解除を強制的に実行できます。以下を実行します。

手順: アセットのロック解除

1. アセットを開きます。
2. **Overview** タブをクリックし、**Metadata** 画面を開きます。
   アセットが別のユーザーによってすでに編集されている場合は、以下が **Lock status** フィールドに表示されます。

   Locked by <user_name>

3. 別のユーザーによってロックされたアセットを編集するには、**Force unclock asset** ボタンをクリックします。
   以下の確認ポップアップメッセージが表示されます。

   Are you sure you want to release the lock of this asset? This might cause <user_name> to lose unsaved changes!

4. **Yes** をクリックして確定します。
   アセットがロック解除された状態に戻ります。

2.6. PROJECT EDITOR

2.6.1. Project Editor

Project Editor を使用してユーザーはプロジェクトをビルドし、デプロイすることができます。このビューは、Web インターフェースで編集できる Red Hat JBoss BRMS プロジェクトの各種プロパティーへのアクセスを提供します。グループアーティファクトバージョン (Group artifact version)、依存関係 (Dependencies)、メタデータ (Metadata)、ナレッジベース設定 (Knowledge Base Settings) およびインポート (Imports) などのプロパティーはこのビューで管理できます。このエディターには、コードリポジトリー内を移動する際にアクティブな現行プロジェクトの設定オプションやコンテンツの変更が表示されます。

Project Editor にアクセスするには、以下を実行します。

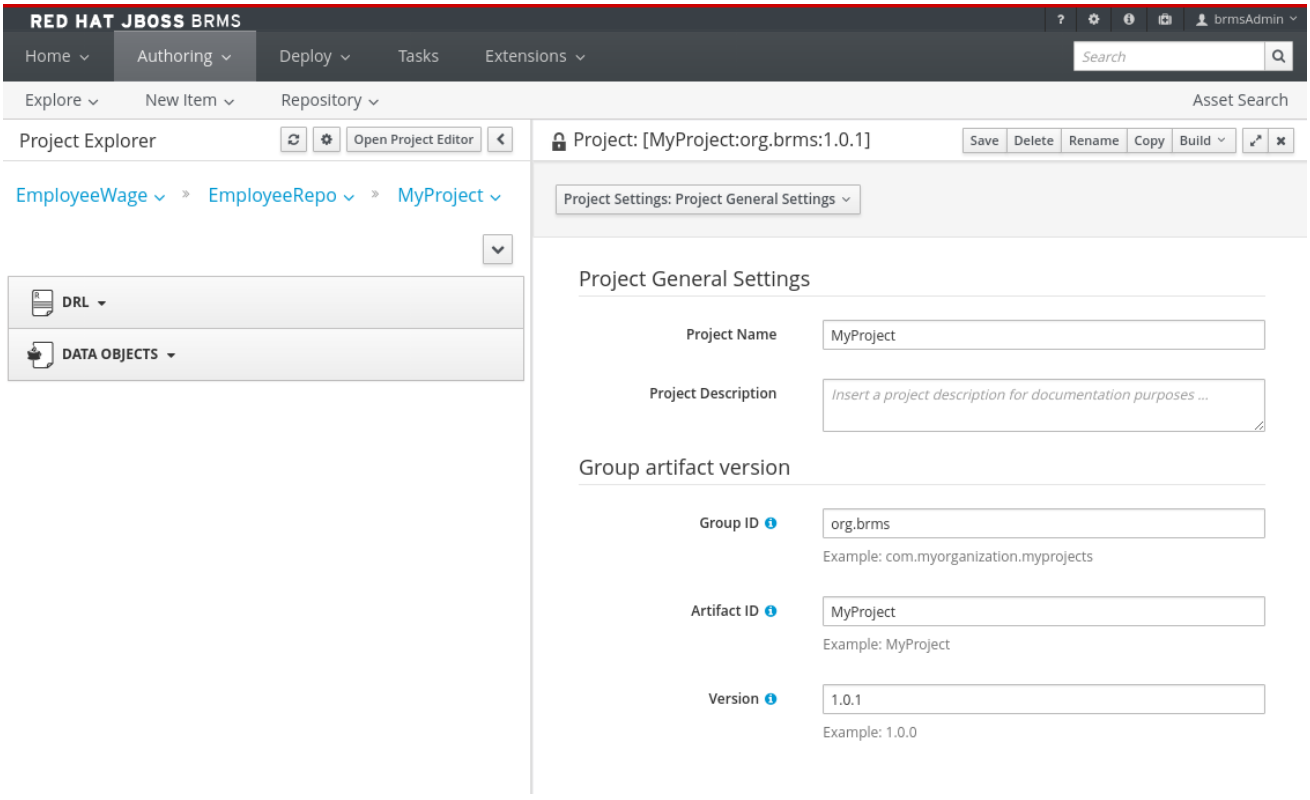
1. **Authoring → Project Authoring** をクリックします。
2. プロジェクトを選択します。
3. **Open Project Editor** をクリックします。

2.6.2. プロジェクトの設定

プロジェクトの全般設定

プロジェクトの設定画面では、ユーザーはプロジェクトのグループ (Group)、アーティファクト (Artifact)、およびバージョン ID (Version ID) を設定できます。Maven を使用してプロジェクトをビルドしているため、**pom.xml** 設定ファイルの編集が行われます。

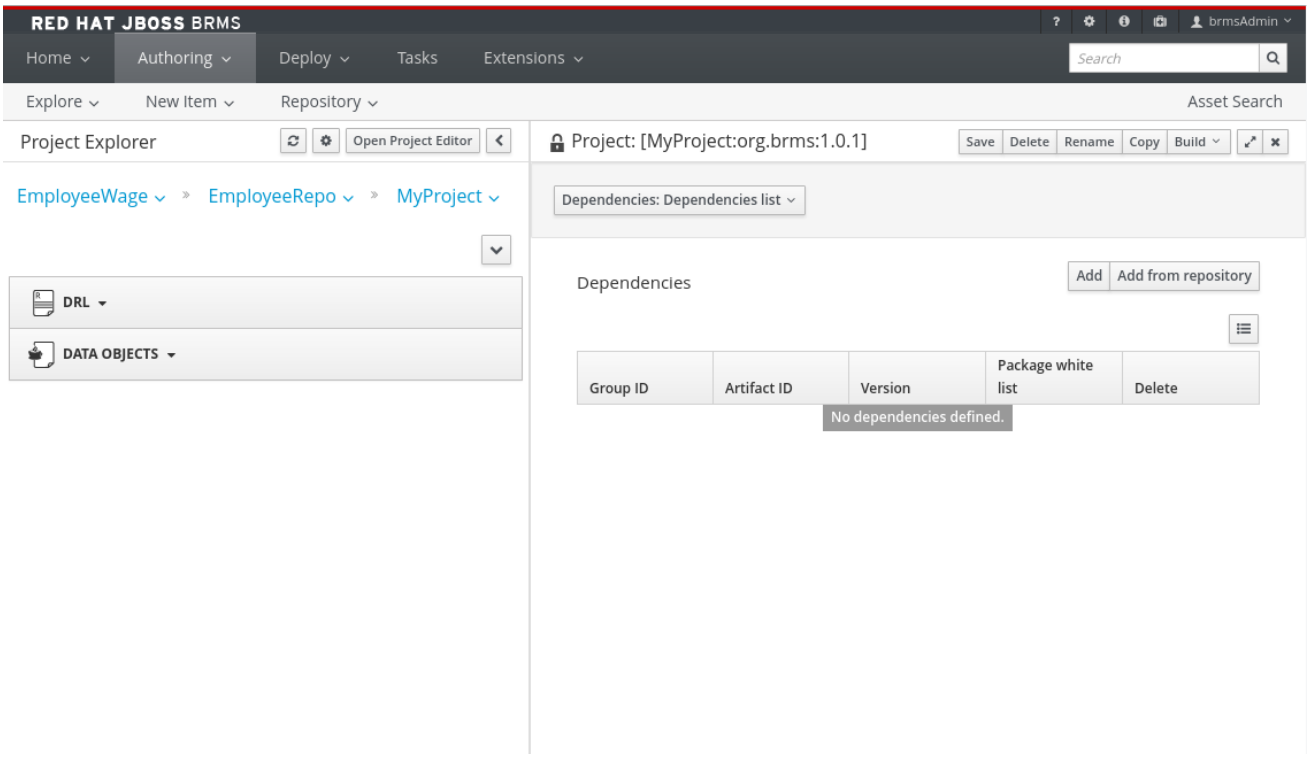
図2.5 Project Editor - Project Settings



依存関係

依存関係のオプションを選択すると、現在のプロジェクトの依存関係を設定することができます。**Project Settings → Dependencies** オプションを使用して依存関係にアクセスできます。**Add from repository** ボタンをクリックするか、または **Add** ボタンをクリックしてプロジェクトのグループ ID、アーティファクト ID およびバージョン ID を直接入力してアーティファクトリポジトリーから依存関係を追加することができます。

図2.6 Project Editor - Project Dependencies

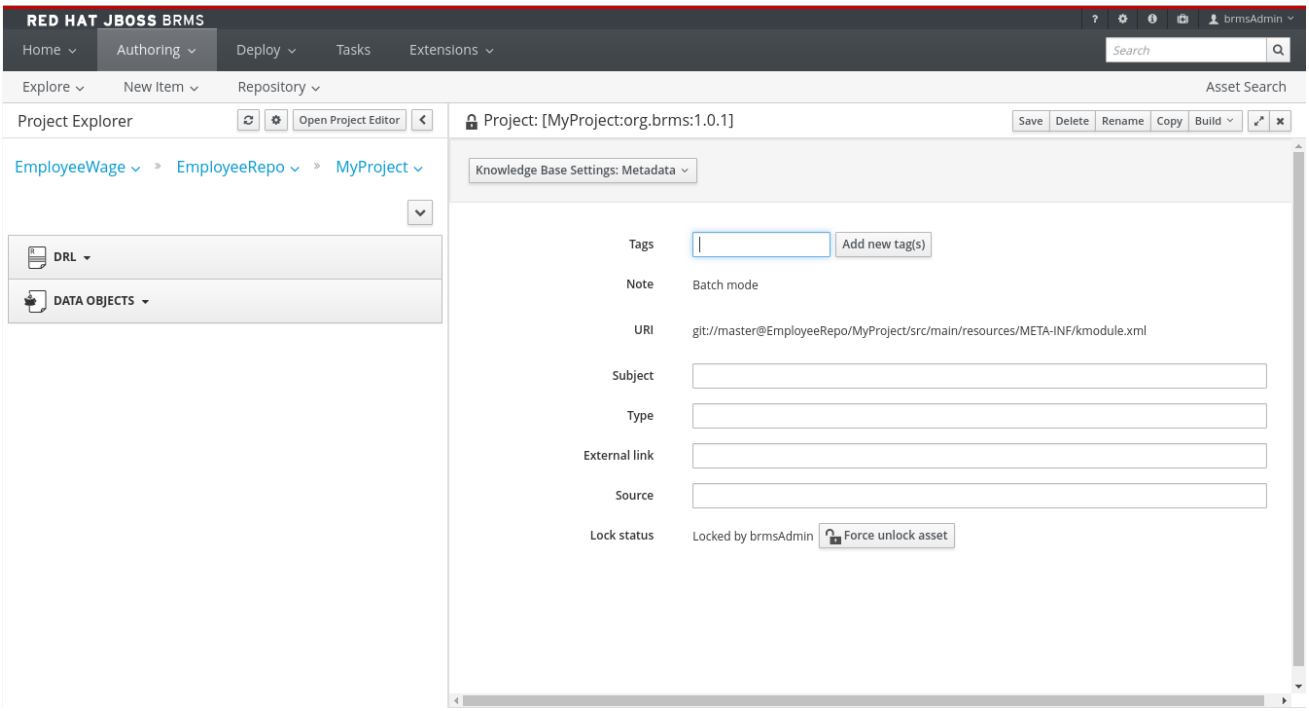


メタデータ

メタデータの画面には、プロジェクトの各種データとバージョン履歴が表示されます。この画面からメタデータの詳細を編集し、説明を追加し、選択したアセットに固有のディスカッションに参加することができます。メタデータはプロジェクト、ナレッジベース (kmodule) およびプロジェクトのインポートに追加できます。各メタデータタブには以下のフィールドがあります。

- **Tags**: アセットを分類するためのタグ付けシステム。
- **Note**: 最近のアセット更新時のコメント。
- **URI**: Git リポジトリー内でのアセットの固有の識別子。
- **Subject**、**Type**、**External link**、**Source**: 各種のアセットのメタデータ。
- **Lock status**: アセットのロック状態。

図2.7 Knowledge Base Settings - Metadata



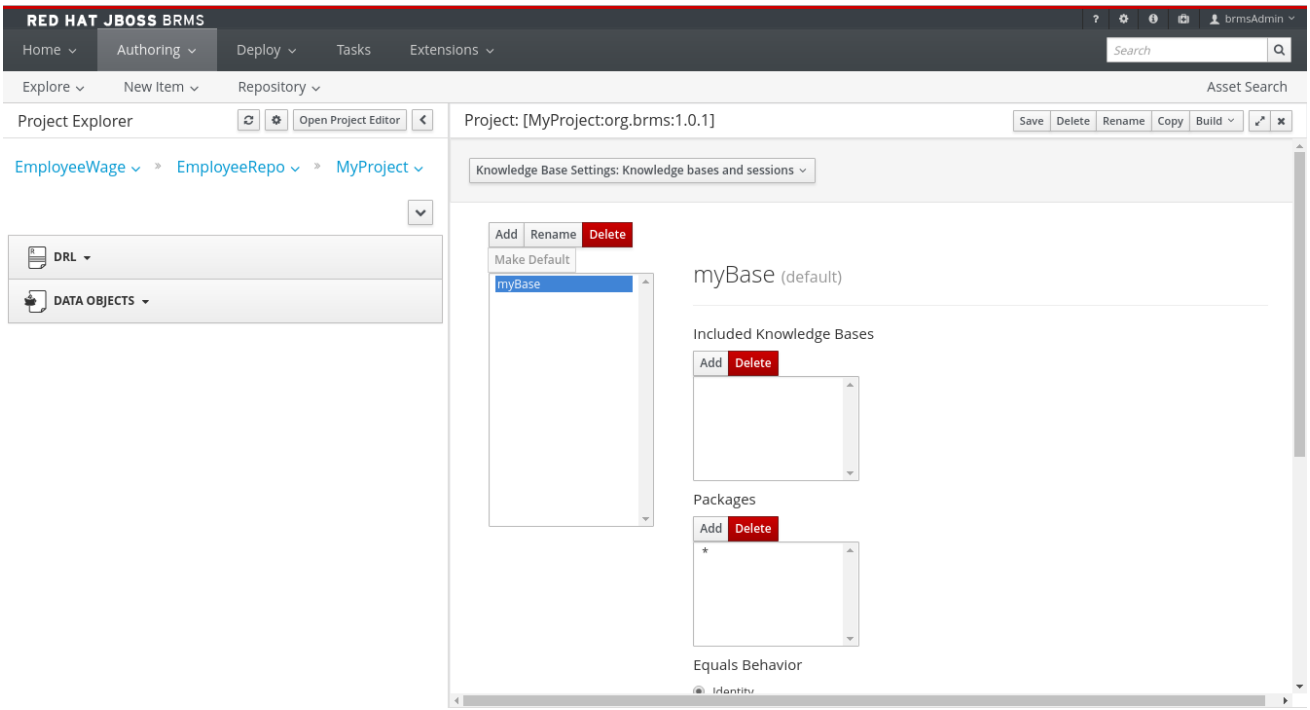
2.6.3. ナレッジベースの設定

ナレッジベースおよびセッション

ナレッジベースの設定では、ユーザーはプロジェクトの **kmodule.xml** プロジェクト記述子ファイルを使用して KIE ベースおよびセッションを作成できます。また、これに応じて **kmodule.xml** プロジェクト設定ファイルが編集されます。

ナレッジベースとセッションページには、名前別にすべてのナレッジベースが一覧表示されます。一覧の上には Add、Rename、Delete、および Make Default オプションがあります。1 度に 1 つのナレッジベースをデフォルトとして設定できます。

図2.8 Project Editor - Knowledge Base Settings



Included Knowledge Bases セクションには、組み込まれたナレッジベースのモデル、ルール、およびその他のコンテンツが表示されます。これは左側のナレッジベースの一覧からナレッジベースを選択した場合にのみ表示され、使用できます。この一覧を使用してコンテンツを追加したり、削除したりできます。

Packages セクションでは、ユーザーはナレッジベースに指定されているパッケージを追加したり、削除したりできます。

Equals Behavior セクションでは、ユーザーは **Identity** または **Equality** アサーションモデルのいずれかを選択できます。

- **Identity** の場合、**IdentityHashMap** を使用してアサートされたすべてのオブジェクトを格納します。
- **Equality** の場合は、**HashMap** を使用してアサートされたすべてのオブジェクトを格納します。

注記

Identity および **Equality** アサーションモデルについての詳細は、『Red Hat JBoss BRMS Development Guide』の kbase 属性についてのセクションを参照してください。

Event Processing Mode セクションでは、ユーザーは **Cloud** または **Stream** 処理モードのいずれかを選択できます。

- **Cloud** 処理モードはデフォルトの処理モードです。このモードは、純粋な前向き連鎖のルールエンジンと同様に動作します。
- **Stream** 処理モードは、アプリケーションがイベントのストリームを処理する必要がある場合に適切なモードになります。

注記

Cloud および **Stream** 処理モードについての詳細は、『Red Hat JBoss BRMS Development Guide』の kbase 属性についてのセクションを参照してください。

Knowledge Sessions 表には、選択されたナレッジベースのすべてのナレッジセッションが一覧表示されます。 

ボタンをクリックすると、新規ナレッジセッションを表に追加できます。

- **Name**フィールドには、セッションの名前が表示されます。
- **Default** オプションのみを各タイプのセッションのいずれかに割り当てることができます。
- **State** ドロップダウンでは、ステートレスまたはステートフルタイプのいずれかを選択できます。
- **Clock** ドロップダウンでは、Realtime または Pseudo オプションのいずれかを選択できます。
-  をクリックすると、ナレッジセッションの追加のプロパティーを表示するポップアップが開きます。

メタデータ

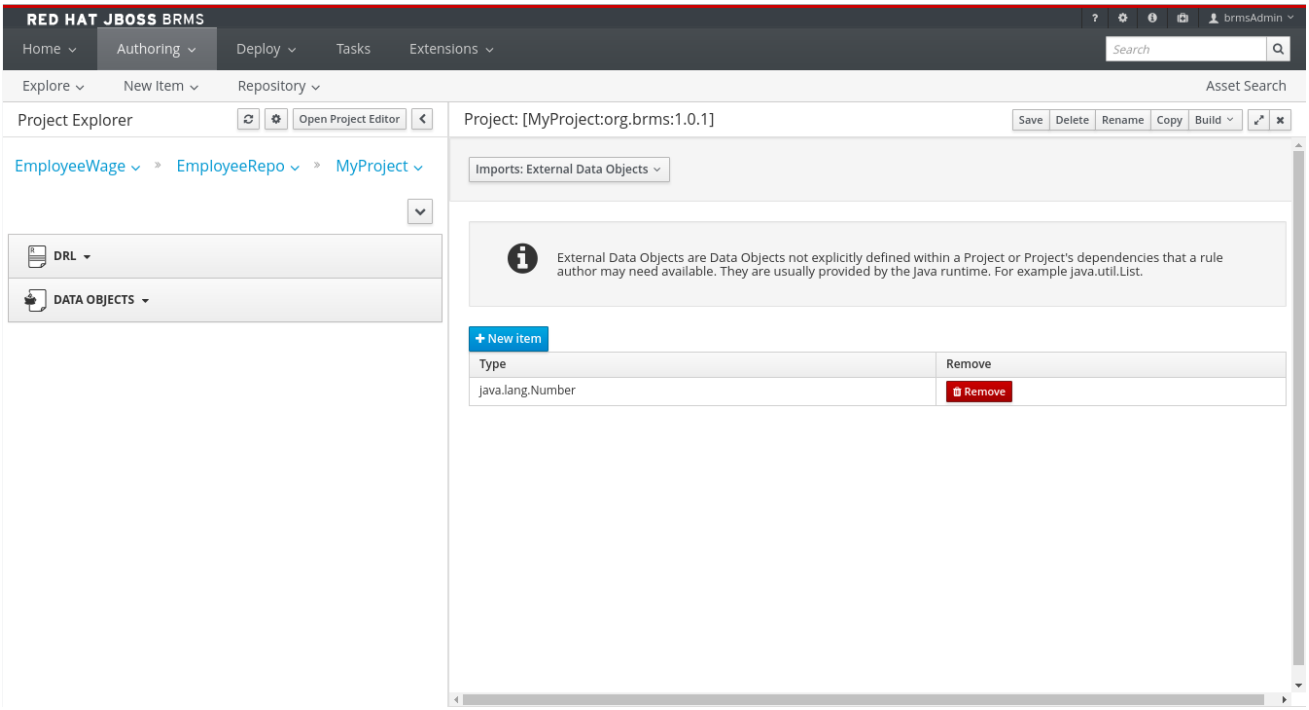
メタデータについての詳細は、「プロジェクトの設定」 を参照してください。

2.6.4. インポート

外部データオブジェクト

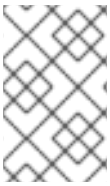
外部データオブジェクトは、プロジェクトで使用されるインポートのセット、または外部データオブジェクトのセットを指定します。プロジェクト内のそれぞれのアセットには独自のインポートがあります。インポートは、ワークベンチで提供されるガイド付きエディターを使用する際に提案として使用されます。これにより、インポートを使用するすべてのファイルにそれぞれのインポートを入力する必要がなくなるため、ワークベンチの使用が容易になります。インポートの設定を変更すると、**project.imports** 設定ファイルが編集されます。通常、データオブジェクトは Java runtime で提供されます (例: **java.util.List**)。

図2.9 Project Editor - Imports



ファクトモデルをインポートセクションに追加するには、**New Item** をクリックします。これにより、**Add Import** の情報に対してポップアップダイアログが表示されます。インポートタイプの入力後に **OK** をクリックします。

ファクトモデルをインポートセクションから削除するには、**Remove** をクリックします。



注記

インポートの提案に一覧表示されるインポートはナレッジベースまたはワークベンチのパッケージに自動的に追加される訳ではありません。それぞれのインポートはそれぞれのファイルに追加される必要があります。

メタデータ

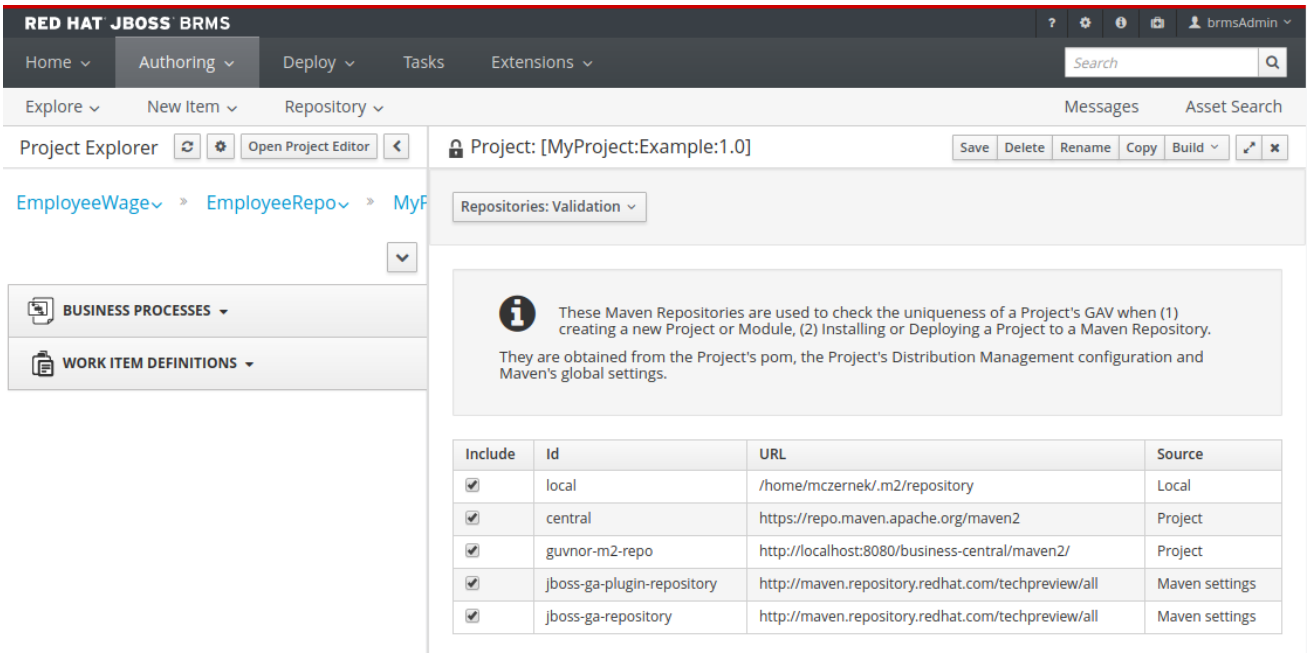
メタデータについての詳細は、「プロジェクトの設定」 を参照してください。

2.6.5. リポジトリー

検証

Validation セクションでは、プロジェクトの GAV (グループ ID、アーティファクト ID、およびバージョン) の一意性を確認するために使用される maven リポジトリーを選択できます。

図2.10 Project Editor - Validation



2.6.6. 永続性

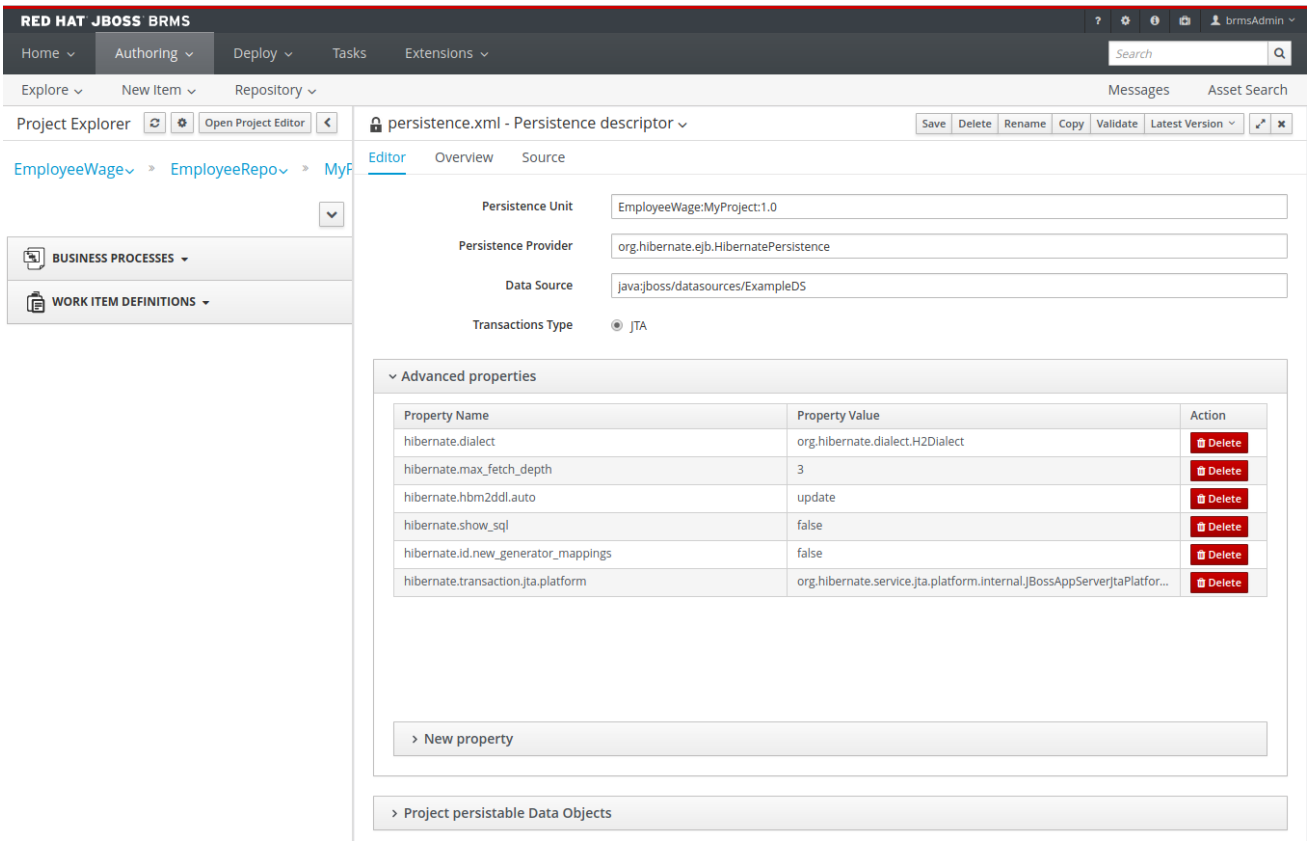
永続性記述子

Persistence descriptor セクションでは、GUI を使って **persistence.xml** を変更できます。以下を実行できます。

- 永続ユニットプロバイダーの定義
- データソースの定義
- 永続ユニットの事前定義プロパティーの変更
- 永続ユニットへの新規プロパティーの追加
- 永続型データオブジェクトの管理
  永続型データオブジェクトは JPA 仕様をベースとしており、すべての基礎的なメタデータが自動的に生成されます。

または、**Source** タブをクリックして **persistence.xml** を直接編集します。

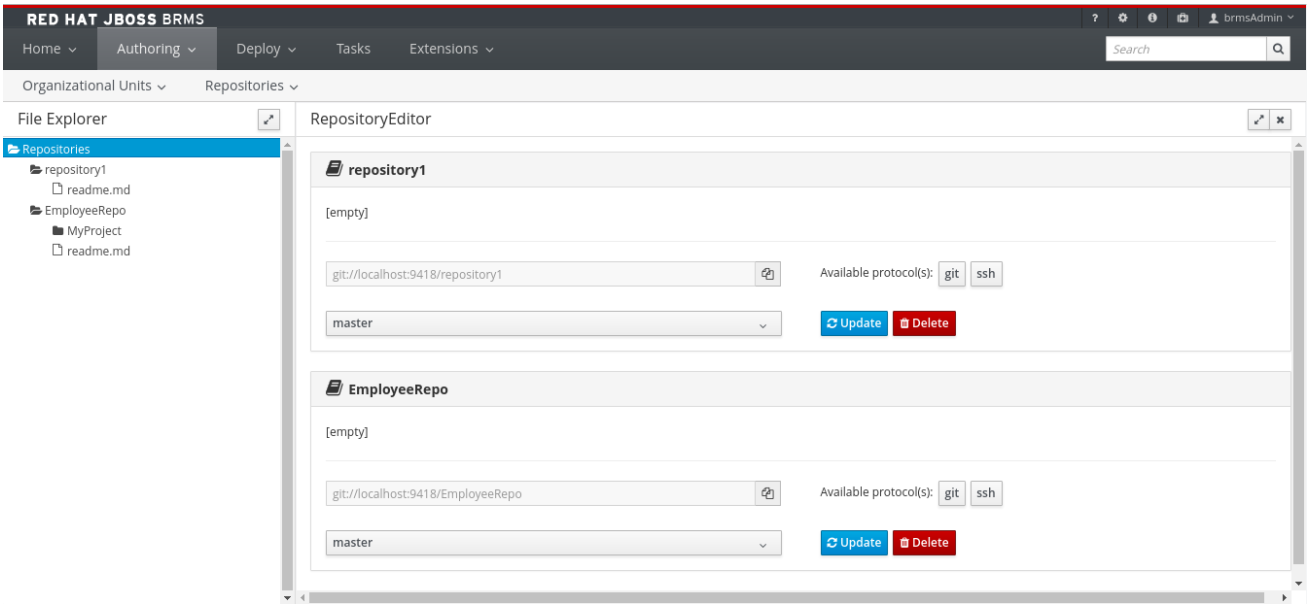
図2.11 永続性記述子



2.7. ADMINISTRATION メニュー

Administration ビューから組織単位とリポジトリーを管理することができます。**Authoring → Administration** をクリックしてこのビューに移動します。これらのアセットの作成および管理方法についての詳細は、*3章新規プロジェクトの設定* を参照してください。

図2.12 Administration 画面



2.8. アセットの名前変更、コピー、削除

2.8.1. ファイルまたはフォルダーの名前変更

ユーザーは Project Explorer でファイルまたはフォルダーの名前を直接変更できます。

1. ファイルまたはフォルダーの名前を変更するには、**Authoring → Project Authoring** を選択して **Project Explorer** を開きます。

2. **Project Explorer** ビューの右上にあるギアの  アイコンをクリックし、開かれるメニューで **Repository View** をクリックします。ギアのアイコンを再度クリックしてオプション **Show as Links** を選択します (すでに選択されていない場合)。

3. 名前を変更する必要のあるファイルまたはフォルダーの右側にある名前変更の  アイコンをクリックします。表示される **Rename this item** ダイアログボックスに新規の名前を入力し、**Rename item** ボタンをクリックします。

2.8.2. ファイルまたはフォルダーの削除

ユーザーは Project Explorer でファイルまたはフォルダーの名前を直接削除できます。

1. ファイルまたはフォルダーの名前を削除するには、**Authoring → Project Authoring** を選択して **Project Explorer** を開きます。

2. **Project Explorer** ビューの右上にあるギアの  アイコンをクリックし、開かれるメニューで **Repository View** をクリックします。ギアのアイコンを再度クリックしてオプション **Show as Links** を選択します (すでに選択されていない場合)。

3. 削除する必要のあるファイルまたはフォルダーの右側にある削除のアイコン () をクリックします。表示される **Delete this item** ダイアログボックスで **Delete item** ボタンをクリックします。

2.8.3. ファイルまたはフォルダーのコピー

ユーザーは Project Explorer でファイルまたはフォルダーの名前を直接変コピーできます。

1. ファイルまたはフォルダーをコピーするには、**Authoring → Project Authoring** を選択して **Project Explorer** を開きます。

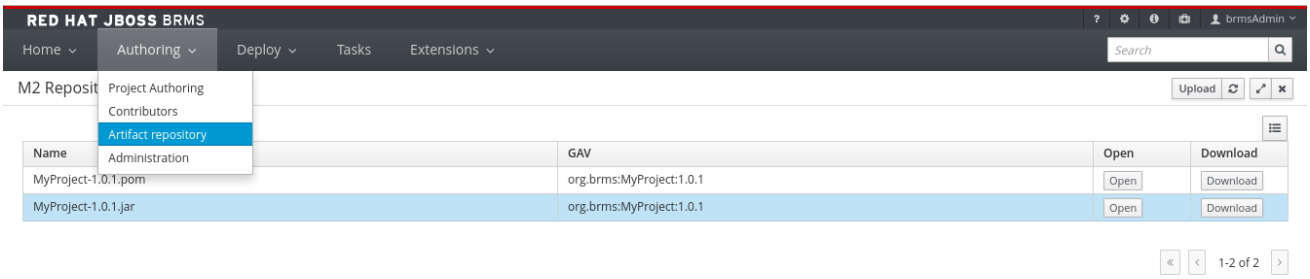
2. **Project Explorer** ビューの右上にあるギアの  アイコンをクリックし、開かれるメニューで **Repository View** をクリックします。ギアのアイコンを再度クリックしてオプション **Show as Links** を選択します (すでに選択されていない場合)。

3. コピーする必要のあるファイルまたはフォルダーの右側にあるコピー  アイコンをクリックします。表示される **Copy this item** ダイアログボックスに新規の名前を入力し、**Create copy** ボタンをクリックします。

2.9. デプロイメントメニュー: アーティファクトリポジトリー

Artifact Repository では Guvnor M2 リポジトリーを参照します。これは、既存プロジェクトで使用される利用可能な kjar ファイルの一覧を表示し、ユーザーによる kjar ファイルのアップロード、ダウンロードおよび管理を可能にします。これは、ツールバーの **Authoring → Artifact Repository** メニューをクリックしてアクセスできます。

図2.13 Artifact Repository 画面



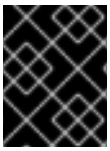
第3章 新規プロジェクトの設定

プロジェクトを作成するには、ビジネスユーザーは組織単位を作成する必要があります。組織単位は特定のビジネスセクターのドメインに基づくものです。これは、プロジェクトおよびパッケージを作成できるリポジトリーを保持します。パッケージは、ルール、ファクトモデル、デシジョンテーブルなどの、デプロイ用に検証し、コンパイルできるデプロイ可能なアセットです。

3.1. 組織単位の作成

kie-config-cli ツールまたは REST API 呼び出しを使用して、Business Central の **Administration** パースペクティブで組織単位を作成することができます。

Business Central での組織単位の作成



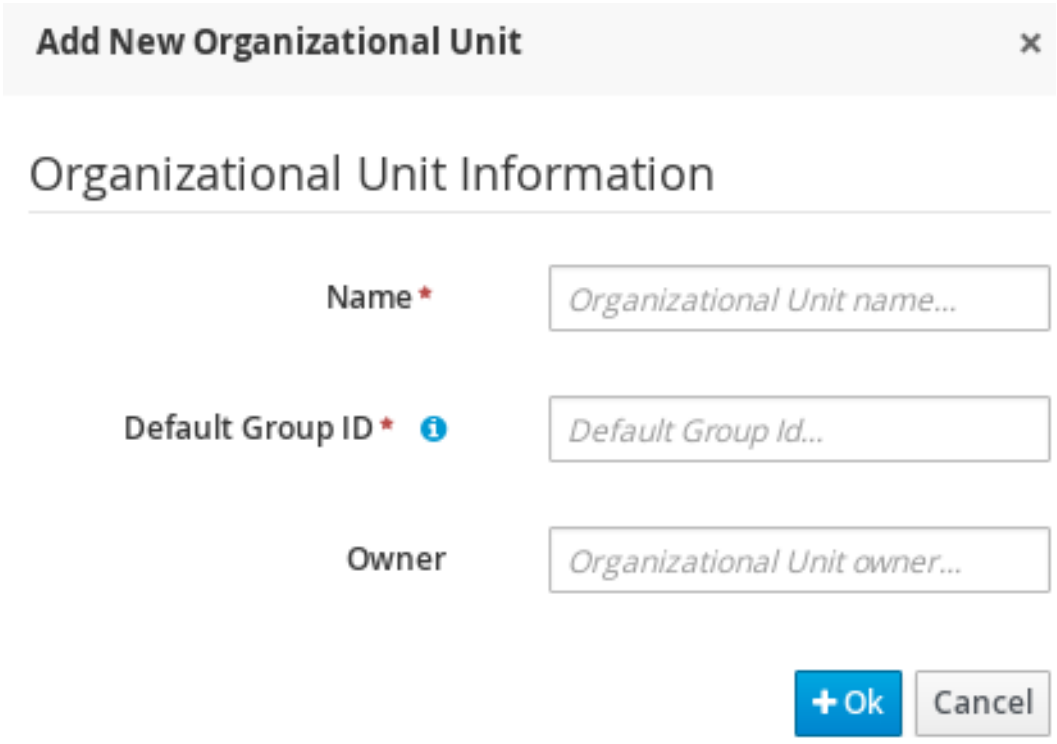
重要

Business Central では、組織単位を作成できるのは、**admin** ロールを持つユーザーのみであることに注意してください。

手順: Business Central を使用した組織単位の作成

1. Business Central で、**Authoring → Administration** に移動します。
2. パースペクティブメニューで、**Organizational Units → Manage Organizational Units** をクリックします。
3. **Organizational Unit Manager** ビューで、**Add** をクリックします。
   Add New Organizational Unit ダイアログウィンドウが開きます。

   図3.1 *Add New Organizational Unit* ダイアログウィンドウ

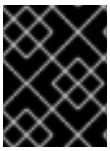


4. 必須パラメーターを 2 つ (**名前** と **デフォルトのグループ ID**) を入力して **Ok** をクリックします。

kie-config-cli ツールでの組織単位の作成
組織単位は、**kie-config-cli** ツールを使用して作成することもできます。これを実行するには、**create-org-unit** コマンドを実行します。このツールは、他に必要とされるパラメーターの入力を求めることで、組織単位作成の全プロセスを順に進めていきます。すべてのコマンドの一覧を参照するには **help** と入力してください。

kie-config-cli ツールについての詳細は、『Red Hat JBoss BRMS Administration and Configuration Guide』のコマンドラインの設定についての章を参照してください。

REST API を使用した組織単位の作成



重要

組織単位を作成できるのは、**rest-all** ロールを持つユーザーのみであることに注意してください。

ナレッジストアで組織単位を作成するには、**POST** REST API の呼び出しを実行します。組織単位の詳細は、JSON エンティティーで定義されます。

この呼び出しの入力パラメーターは **OrganizationalUnit** インスタンスで、呼び出しにより **CreateOrganizationalUnitRequest** インスタンスが返されます。

例3.1 Curl ユーティリティーを使用した組織単位の作成

作成する組織単位の詳細を記載した JSON エンティティーの例

```
{
 "name"        : "helloWorldUnit",
 "owner"       : "tester",
 "description" :  null,
 "repositories" :  []
}
```

以下のコマンドを実行します。

```
curl -X POST 'localhost:8080/business-central/rest/organizationalunits/' -u
USERNAME:PASSWORD -H 'Accept: application/json' -H 'Content-Type: application/json' -d
'{"name":"helloWorldUnit","owner":"tester","description":null,"repositories":[]}'
```

詳細は、『Red Hat JBoss BPM Suite Development Guide』の「Knowledge Store REST API」の章の「Organizational Unit Calls」のセクションを参照してください。

3.2. リポジトリーの作成

リポジトリーを作成する方法として、Business Central の **Administration** パースペクティブ、**kie-config-cli** ツール、または REST API 呼び出しを使用する 3 つの方法があります。

Business Central でのリポジトリーの作成

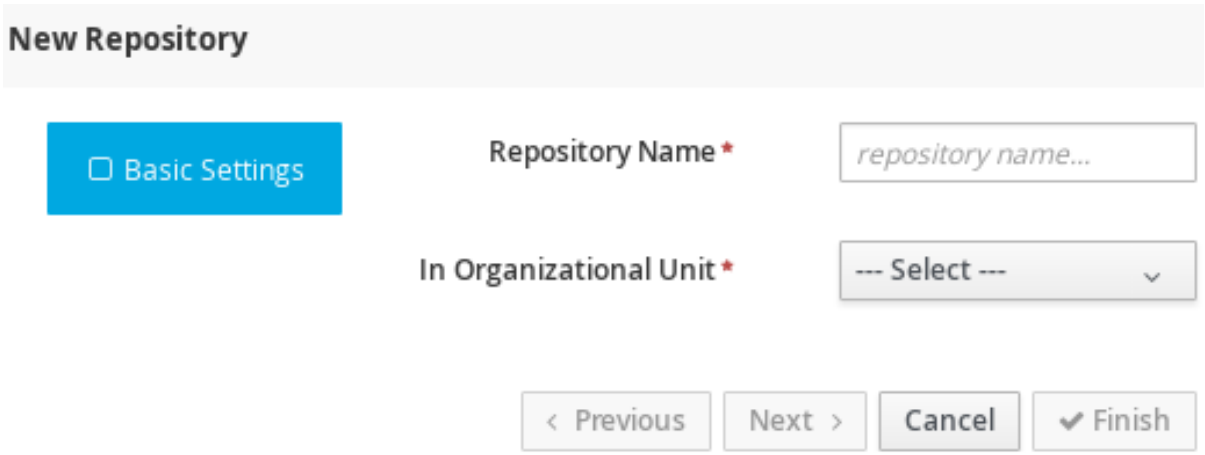
重要

Business Central では、リポジトリーを作成できるのは、**admin** ロールを持つユーザーのみであることに注意してください。

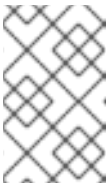
手順: Business Central を使用したリポジトリーの作成

1. Business Central で、**Authoring → Administration** に移動します。
2. パースペクティブメニューで **Repositories → New repository** をクリックします。
   New Repository のポップアップウィンドウが表示されます。

   図3.2 *New Repository* ダイアログウィンドウ



3. 必須パラメーター 2 つを指定します。
   - リポジトリー名



注記

リポジトリー名が有効なファイル名であることを確認してください。名前が無効になる可能性があるので、スペースまたは特殊文字の使用は避けるようにしてください。

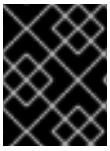
   - 組織単位: 新規に作成されたリポジトリーの場所を指定します。
4. **Finish** をクリックします。

新規に作成されたリポジトリーは、**File Explorer** または **Project Explorer** に表示できます。

kie-config-cli ツールを使用したリポジトリーの作成
kie-config-cli ツールを使用して新規 git リポジトリーを作成するには、**create-repo** コマンドを実行します。このツールは、他に必要とされるパラメーターの入力を求めることで、リポジトリー作成の全プロセスを順に進めていきます。すべてのコマンドの一覧を参照するには **help** と入力してください。

kie-config-cli ツールについての詳細は、『Red Hat JBoss BRMS Administration and Configuration Guide』を参照してください。

REST API を使用したリポジトリーの作成



重要

リポジトリーを作成できるのは、**rest-all** ロールを持つユーザーのみであることに注意してください。

ナレッジストアでリポジトリーを作成するには、**POST** REST API 呼び出しを実行します。リポジトリーの詳細は、JSON エンティティーで定義されます。この呼び出しを行う前に、HTTP セッションが確立され、認証済みであることを確認してください。

この呼び出しの入力パラメーターは **RepositoryRequest** インスタンスで、呼び出しにより **CreateOrCloneRepositoryRequest** インスタンスが返されます。

例3.2 Curl ユーティリティーを使用したリポジトリーの作成

作成するリポジトリーの詳細を記載した JSON エンティティーの例

```
{
 "name"                : "newRepository",
 "description"          :  null,
 "gitURL"               :  null,
 "requestType"          : "new",
 "organizationalUnitName" : "helloWorldUnit"
}
```

以下のコマンドを実行します。

```
curl -X POST 'localhost:8080/business-central/rest/repositories/' -u USERNAME:PASSWORD -H
'Accept: application/json' -H 'Content-Type: application/json' -d
'{"name":"newRepository","description":null,"requestType":"new","gitURL":null,"organizationalUnitNa
me":"helloWorldUnit"}'
```

詳細は、『**Red Hat JBoss BPM Suite Development Guide**』の「**Knowledge Store REST API**」の章の「**Repository Calls**」のセクションを参照してください。

3.3. リポジトリーのクローン作成

Business Central または REST API 呼び出しのいずれかでリポジトリーのクローンを作成することができます。任意のリポジトリーのクローン作成には、**kie-config-cli** ツールは使用できません。**git clone** を実行するか、以下のオプションの 1 つを使用してください。

Business Central でのリポジトリーのクローン作成

重要

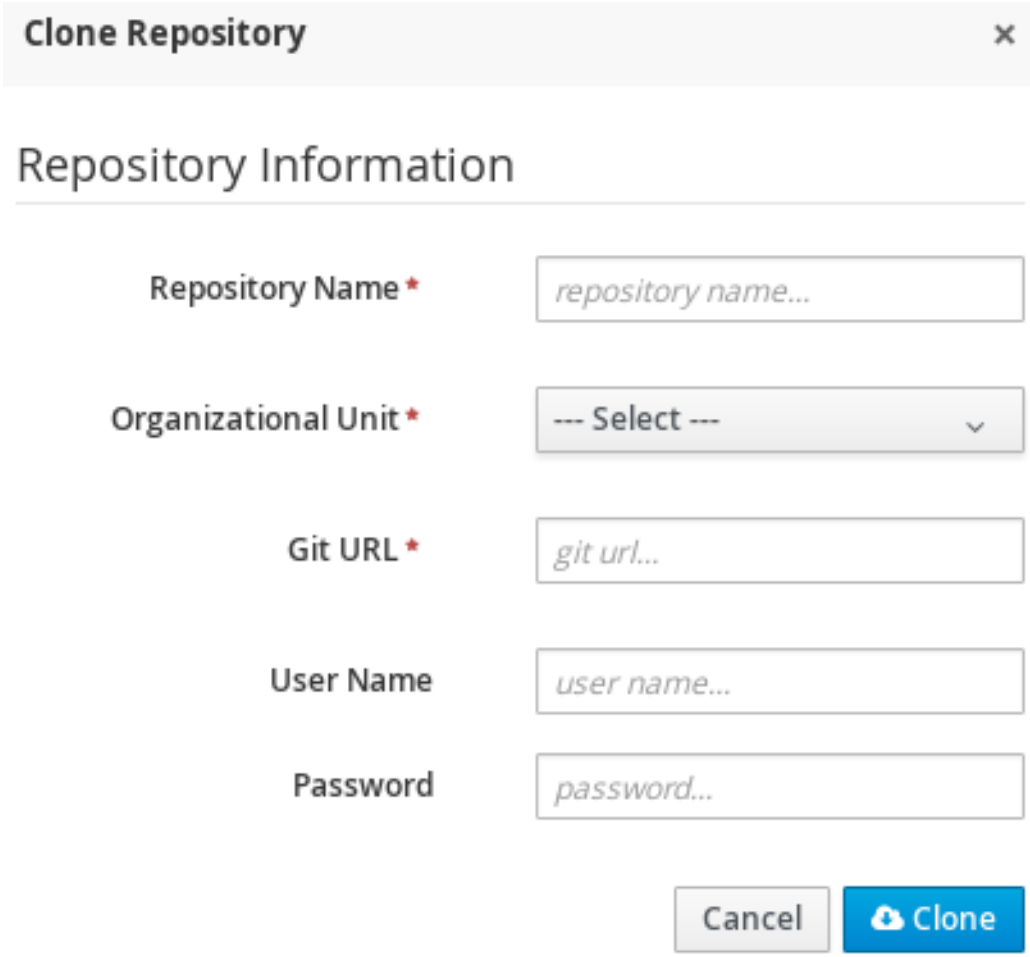
Business Central では、リポジトリーのクローンを作成できるのは、**admin** ロールを持つユーザーのみであることに注意してください。

手順: Business Central を使用したリポジトリーのクローン作成

1. Business Central で、**Authoring → Administration** に移動します。
2. パースペクティブメニューで **Repositories → Clone repository** を選択します。

Clone Repository のポップアップウィンドウが表示されます。

図3.3 *Clone Repository* ダイアログウィンドウ



3. Clone Repository ダイアログウィンドウで、リポジトリーの詳細を入力します。

   a. アセットリポジトリーでリポジトリーの識別子として使用する **Repository Name** を入力して、リポジトリーの追加先の **Organizational Unit** を選択します。

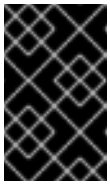
   b. git リポジトリーの URL を入力します。

   - ローカルのリポジトリーの場合は、**file:///PATH_TO_REPOSITORY/REPOSITORY_NAME** を使用します。

注記

ファイルプロトコルは、読み取り操作の場合にのみサポートされ、書き込み操作はサポート **されません**。

   - リモートまたは既存のリポジトリーの場合は、**https://github.com/USERNAME/REPOSITORY_NAME.git** または **git://HOST_NAME/REPOSITORY_NAME** を使用します。



重要

SCP スタイルの SSH URL ではなく、HTTPS または Git プロトコルを使用することが重要です。Business Central は、基本的な SSH URL をサポートせず、**Invalid URL format** で失敗してしまいます。

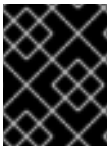
c. 適切な場合には、認証に使用する git アカウントの **User Name** と **Password** を入力します。

4. **Clone** をクリックします。
確認プロンプトとして、リポジトリーが正常に表示されたことを示す通知が表示されます。

5. **OK** をクリックします。
これでリポジトリーがインデックス化されます。インデックス化が完了するまで、ワークベンチの機能の一部が使用できない場合があります。

クローン作成したリポジトリーは、**File Explorer** または **Project Explorer** で確認できます。

REST API を使用したリポジトリーのクローン作成



重要

リポジトリーのクローンを作成できるのは、**rest-all** ロールを持つユーザーのみであることに注意してください。

リポジトリーのクローンを作成するには、**POST** REST API 呼び出しを実行します。この呼び出しにより、(**requestType** パラメーターの値に従い) JSON エンティティーで定義されたリポジトリーが作成されるか、またはそのクローンが作成されます。

この呼び出しの入力パラメーターは **RepositoryRequest** インスタンスで、呼び出しにより **CreateOrCloneRepositoryRequest** インスタンスが返されます。

例3.3 Curl ユーティリティーを使用したリポジトリーのクローン作成

クローン作成するリポジトリーの詳細を記載した JSON エンティティーの例

```
{
 "name"                 : "clonedRepository",
 "description"           :  null,
 "requestType"           : "clone",
 "gitURL"                : "git://localhost:9418/newRepository",
 "organizationalUnitName" : "helloWorldUnit"
}
```

以下のコマンドを実行します。

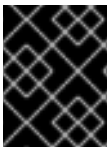
```
curl -X POST 'localhost:8080/business-central/rest/repositories/' -u USERNAME:PASSWORD -H
'Accept: application/json' -H 'Content-Type: application/json' -d
'{"name":"clonedRepository","description":null,"requestType":"clone","gitURL":"git://localhost:9418/ne
wRepository","organizationalUnitName":"helloWorldUnit"}'
```

詳細は、『**Red Hat JBoss BPM Suite Development Guide**』の「**Knowledge Store REST API**」の章の「**Repository Calls**」のセクションを参照してください。

3.4. プロジェクトの作成

Business Central の **Project Authoring** パースペクティブか、または REST API 呼び出しを使用してプロジェクトを作成することができます。

Business Central でのプロジェクトの作成

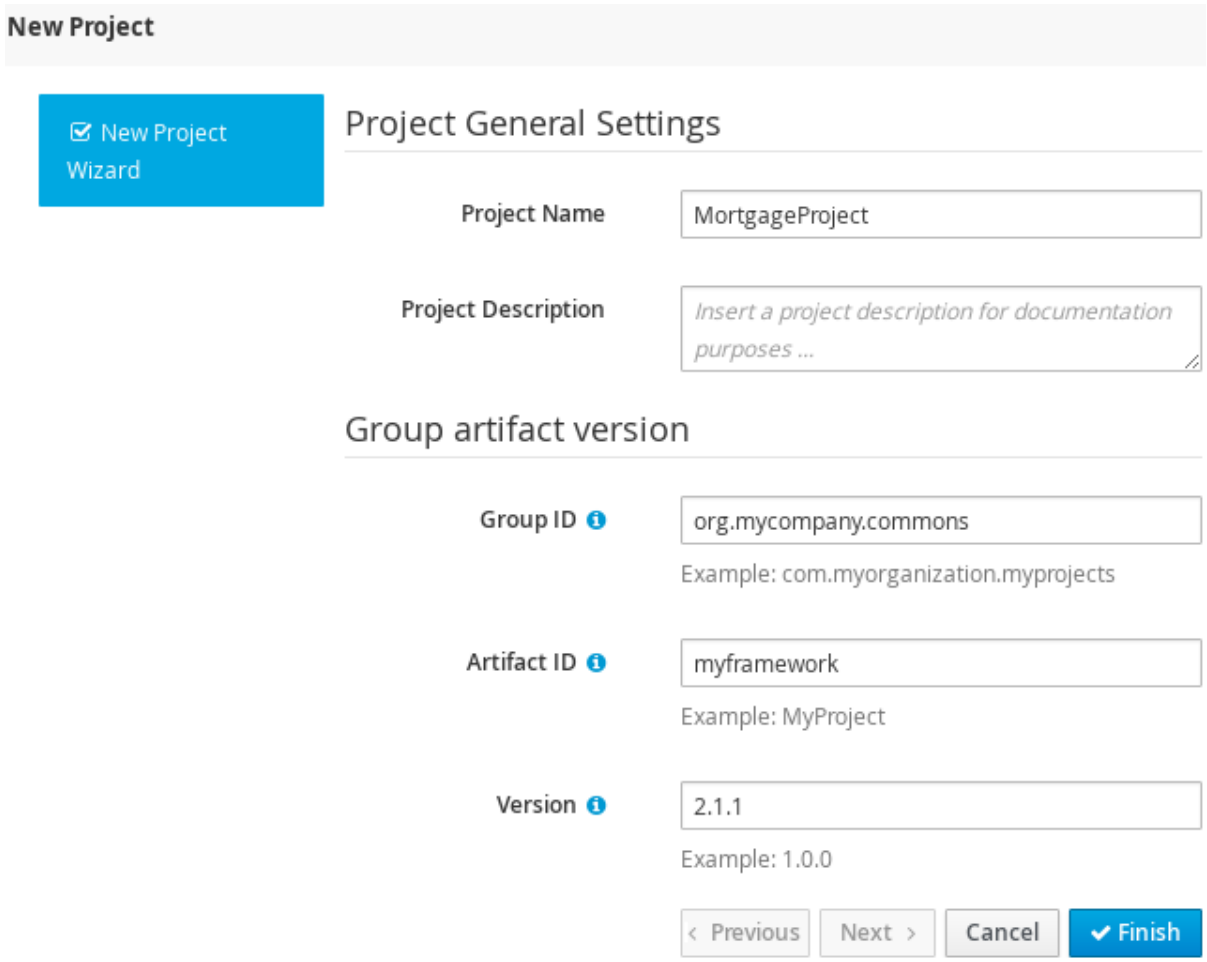


重要

Business Central では、プロジェクトを作成できるのは、**admin** ロールを持つユーザーのみであることに注意してください。

手順: Business Central を使用したプロジェクトの作成

1. Business Central で、**Authoring → Project Authoring** に移動します。
2. **Project Explorer** で、組織単位と、プロジェクトを作成する必要のあるリポジトリーを選択します。
3. パースペクティブメニューで **New Item → Project** をクリックします。
New Project のダイアログウィンドウが開きます。



4. 新規プロジェクトの **Project General Settings** および **Group artifact version** の詳細を定義します。これらのパラメーターは **pom.xml** Maven 設定ファイルに保存されます。
パラメーターの詳細な説明を参照してください。

- **Project Name**: プロジェクトの名前 (例: **MortgageProject**)。
- **Project Description**: プロジェクトの説明。プロジェクトの文書化に役立ちます。
- **Group ID**: プロジェクトのグループ ID (例: **org.mycompany.commons**)。

- **Artifact ID**: グループ内で一意のアーティファクト ID (例: **myframework**)。名前が無効になる可能性があるので、スペースまたは特殊文字の使用は避けるようにしてください。
- **Version**: プロジェクトのバージョン (例: **2.1.1**)。

5. **Finish** をクリックします。
プロジェクト画面のビューは、**pom.xml** ファイルで定義されるように新規プロジェクトの詳細で更新されます。プロジェクト画面のビューの先頭にある **Project Settings: Project General Settings** ボタンをクリックして、プロジェクト記述子ファイル間の切り替えや、コンテンツの編集を実行できます。

REST API を使用したプロジェクトの作成

重要

プロジェクトを作成できるのは、**rest-all** または **rest-project** ロールを持つユーザーのみであることに注意してください。

リポジトリーのプロジェクトを作成するには、**POST** REST API の呼び出しを実行します。プロジェクトの詳細は、対応する JSON エンティティーで定義されます。

この呼び出しの入力パラメーターは **Entity** インスタンスで、この呼び出しにより、**CreateProjectRequest** インスタンスが返されます。

例3.4 Curl ユーティリティーを使用したプロジェクトの作成

作成するプロジェクトの詳細を記載した JSON エンティティーの例

```
{
 "name"        : "MortgageProject",
 "description" :  null,
 "groupId"     : "org.mycompany.commons",
 "version"     : "2.1.1"
}
```

以下のコマンドを実行します。

```
curl -X POST 'localhost:8080/business-central/rest/repositories/REPOSITORY_NAME/projects/' -u USERNAME:PASSWORD -H 'Accept: application/json' -H 'Content-Type: application/json' -d '{"name":"MortgageProject","description":null,"groupId":"org.mycompany.commons","version":"2.1.1"}'
```

詳細は、『**Red Hat JBoss BPM Suite Development Guide**』の「**Knowledge Store REST API**」の章の「**Repository Calls**」のセクションを参照してください。

3.5. 新規パッケージの作成

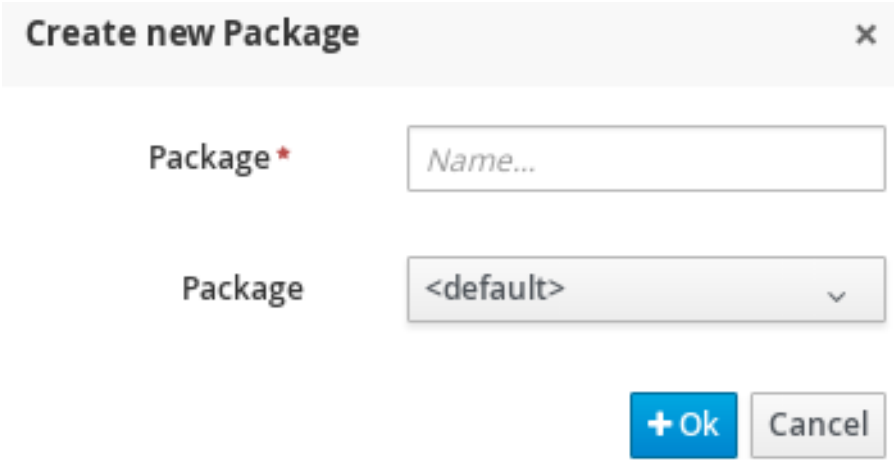
Business Central の **Project Authoring** パースペクティブで新規パッケージを作成することができます。

手順: Business Central での新規パッケージの作成

1. Business Central で、**Authoring → Project Authoring** に移動します。

2. **Project Explorer** ビューで、組織単位、リポジトリーおよびパッケージを作成するプロジェクトを選択します。

3. パースペクティブメニューで **New Item → Package** をクリックします。
Create new Package ダイアログウィンドウが開きます。



4. パッケージの詳細を定義します。パッケージ名を入力し、パッケージを指定します。

5. **OK** をクリックします。
これで新規パッケージが選択されたプロジェクトの下に作成されます。

3.6. 依存関係の追加

依存関係をプロジェクトに追加するには、以下を実行します。

1. 指定プロジェクトの Project Editor を開きます。

   a. **Project Authoring** パースペクティブの **Project Explorer** ビューで、プロジェクトディレクトリーを開きます。

   b. **Open Project Editor** をクリックしてプロジェクトビューを開きます。

2. **Project Screen** ビューで、**Project Settings** ドロップダウンボックスから **Dependencies** 項目を選択します。

3. 更新された **Project Screen** で、**Add** ボタンをクリックし、maven 依存関係を追加するか、または **Add from repository** ボタンをクリックしてナレッジストア (アーティファクトリポジトリー) から依存関係を追加します。

   a. maven 依存関係を追加する際に、ユーザーは **Dependency** ダイアログウィンドウで **Group ID**、**Artifact ID** および **Version ID** を定義する必要があります。

   b. ナレッジストアから依存関係を追加する際に、表示されるダイアログボックスで依存関係を選択します。依存関係は依存関係表に追加されます。

4. 各種の変更を適用するには、依存関係が保存されている必要があります。

また依存関係の使用時に **Package white list** を使用することもできます。リポジトリーの追加時にギアのアイコンをクリックし、**Add all** または **Add none** をクリックできます。これにより、追加した依存関係からすべてのパッケージが組み込まれるか、またはいずれのパッケージも組み込まれなくなります。

警告

変更されたアーティファクトを使用している場合、変更されたスナップショット以外のアーティファクトをリロードしないでください。Maven はこれらのアーティファクトが更新されていることを認識せず、また Maven はこの方法でデプロイされた場合は機能しなくなるためです。

3.7. KIE ベースおよびセッションの定義

KIE ベース は、アプリケーションのナレッジ定義を含むリポジトリーであり、ルール、プロセス、関数、およびタイプモデルが含まれます。KIE ベースにはランタイムデータは含まれません。代わりに、セッションが KIE ベースから作成されます。データを KIE ベースに挿入したり、プロセスインスタンスを KIE ベースから起動したりできます。

KIE セッション は、KIE ベースから作成されるランタイムデータを格納します。詳細は、『**Red Hat JBoss BPM Suite Development Guide**』の「KIE Sessions」の章を参照してください。

KIE ベースおよびセッションは、プロジェクトの **kmodule.xml** プロジェクト記述子ファイルを編集して作成できます。この作成は、Business Central からか、または **Repository** ビューに移動し、**src/main/resources/META-INF/** フォルダーの **kmodule.xml** を編集して実行できます。

Project Editor での KIE ベースおよびセッションの定義

Business Central で KIE ベースまたはセッションを定義するには、以下を実行します。

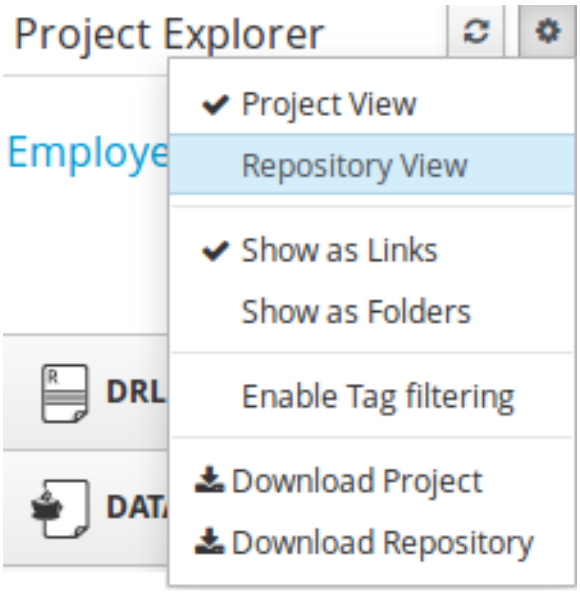
1. **Authoring → Project Authoring** をクリックしてプロジェクトに移動します。
2. **Project Explorer** ウィンドウで、**Open Project Editor** をクリックします。
3. **Project Settings: Project General Settings → Knowledge bases and sessions** をクリックします。このビューでは、**kmodule.xml** を変更するためのユーザーインターフェースが提供されます。
4. **Add** をクリックしてベースを定義し、これを追加します。
   a. ナレッジベースの名前を入力した後に、パッケージを追加します。すべてのパッケージを組み込むには、**Packages** の下にある **Add** をクリックしてからアスタリスク * を入力します。
5. **Knowledge Sessions** の下で、**Add** をクリックし、セッションの名前を入力します。
6. **Default** としてマークし、該当する状態を選択します。
   Red Hat JBoss BRMS の場合、**ステートフル** と **ステートレス** セッション間の切り替えを行えます。ファクトのインタラクティブな呼び出しが不要な場合には **ステートレス** を使用します。その他の場合には、**ステートフル** セッションを使用します。 |
7. いったん終了したら、右上にある **Save** をクリックします。

kmodule.xml での KIE ベースおよびセッションの定義

kmodule.xml を編集して KIE ベースまたはセッションを定義するには、以下を実行します。

1. プロジェクトのリポジトリービューを開きます。

図3.4 リポジトリービューへの切り替え



2. **/src/main/resources/META-INF** に移動します。**kmodule.xml** をクリックしてファイルを直接編集します。

3. **kbases** および **ksessions** を定義します。以下のようになります。

```
<kmodule xmlns="http://www.drools.org/xsd/kmodule"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <kbase name="myBase" default="true" eventProcessingMode="stream"
equalsBehavior="identity" packages="*">
    <ksession name="mySession" type="stateless" default="true" clockType="realtime"/>
  </kbase>
</kmodule>
```

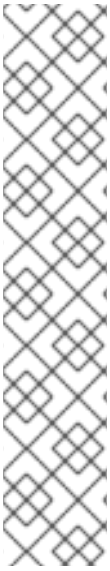
4. 右上にある **save** をクリックします。

Project Editor ビューおよびリポジトリービュー間での切り替えを行い、それぞれのビューで加えた変更を確認することができます。これを実行するには、変更が加えられるたびにビューを閉じ、再度開く必要があります。

3.8. リソースの作成

プロジェクトには、プロセス定義、作業アイテム定義、フォーム定義、ビジネスルール定義などのリソースを含むファイルからなる任意の数のパッケージが含まれる場合があります。

リソースを作成するには、**Project Explorer** でプロジェクトおよびパッケージを選択し、パースペクティブメニューで **New Item** をクリックしてから、作成する必要のあるリソースを選択します。



パッケージの作成

プロセス定義、作業アイテム定義、データモデルなどのリソースは、プロジェクトのパッケージ内に作成し、リソースのインポートやそれらのコンテンツの参照が可能になるようにすることが推奨されています。

パッケージを作成するには、以下を実行します。

- Project Explorer の **Repository** ビューで、**REPOSITORY/PROJECT/src/main/resources/** ディレクトリーに移動します。
- **New Item → Package** に移動します。
- **New resource** ダイアログで、パッケージ名を定義し、リポジトリー内のパッケージの位置を確認します。

第4章 データセット

Business Central でのデータセット機能は、データのアクセスおよび解析方法を定義します。データセットは、Dashbuilder ディスプレイヤーで表示できるデータのソースとして機能します。Dashbuilder ディスプレイヤーは、Plugin Management パースペクティブのカスタムパースペクティブに追加できます。データセットパースペクティブは管理者グループのユーザーのみに表示されることに注意してください。

4.1. データセットの管理

データセットの定義を追加するには、以下を実行します。

1. Business Central にログインして、**Extensions → Data Sets** をクリックします。
2. **New Data Set** をクリックします。
3. プロバイダーのタイプを選択し、**Next** をクリックします。現時点で、以下のプロバイダータイプがサポートされています。
   - Java クラス: Java クラスからデータセットを生成します。
   - SQL: ANSI-SQL 準拠データベースからデータセットを生成します。
   - CSV: リモートまたはローカル CSV ファイルからデータセットを生成します。
   - Elasticsearch – generate a data set from Elasticsearch nodes.

注記

Elasticsearch data set integration support is limited to commercially reasonable efforts. For details, see What is commercially reasonable support? .

1. **Data Set Creation Wizard** を完了し、**Test** をクリックします。
2. 選択するプロバイダーに応じて、設定手順は異なります。手順が完了したら、**Save** をクリックしてデータセットの定義を作成します。

データセットを編集するには、以下を実行します。

1. Business Central にログインして、**Extensions → Data Sets** をクリックします。
2. **Data Set Explorer** で、既存のデータセットをクリックしてから **Edit** をクリックします。
3. **Data Set Editor** が開きます。3 つのタブでデータセットを編集できます。タブの一部は選択するプロバイダーに応じて変わることに注意してください。以下は **CSV** データプロバイダーに適用されます。
   - **CSV Configuration**: データセット定義の名前、ソースファイル、区切り (separator) その他のプロパティーを変更することができます。
   - **Preview**: **CSV Configuration** タブで **Test** をクリックした後に、システムはデータセットのルックアップ呼び出しを実行し、データが利用可能な場合にはプレビューが表示されます。2 つのサブタブがあることに注意してください。
     - **Data columns**: データセット定義に含める列をカスタマイズできます。

- **Filter**: 新規フィルターを追加できます。

- **Advanced**: 以下を管理できます。
  - キャッシング: 詳細は、「キャッシング」 を参照してください。
  - キャッシュのライフサイクル: 詳細は、「データ更新」 を参照してください。

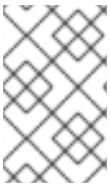
4.2. キャッシング

Red Hat JBoss BRMS データセット機能は 2 つのキャッシュレベルを提供します。

- クライアントレベル
- バックエンドレベル

Client Cache
これがオンになっていると、データセットはルックアップ操作時に web ブラウザーにキャッシュされます。そのため、以降のルックアップ操作の要求はバックエンドに対して行われません。

Backend Cache
これがオンになっていると、データセットは Red Hat JBoss BRMS エンジンによってキャッシュされます。これにより、リモートストレージシステムへの要求数が減少します。



注記

Java および CSV データプロバイダーはバックエンドのキャッシングに依存しています。そのため、バックエンドキャッシュの設定は **Data Set Explorer** の **Advanced** タブに常に表示される訳ではありません。

4.3. データ更新

更新機能により、特定の期間の後にキャッシュされたデータセットのデータを無効にすることができます。**Refresh on stale data**機能は、バックエンドデータが変更される際にキャッシュされたデータを無効にします。

第5章 ソーシャルイベント

Red Hat JBoss BRMS では、ユーザーは他のユーザーにフォローし、それらのユーザーが行っているアクティビティーへの洞察を得ることができます。ユーザーは通常のイベントについて通知されたり、それらのタイムラインをフォローすることもできます。この機能は、Social Activities フレームワークの実装によって利用可能になります。このフレームワークによって、イベント通知がシステム内の各種のアクティビティーごとに生成され、これらの通知が登録されているアクターが参照できるように配信されます。

複数のアクティビティーがイベントをトリガーします。これには、リポジトリーの新規作成、リソースの追加および更新、およびプロセスの追加および更新が含まれます。ユーザーは、Business Central にログインすると、適切な認証情報を使ってこれらの通知を確認できます。

ユーザーのフォロー

ユーザーをフォローするには、ユーザーの名前を **People** パースペクティブの検索ボックスに入力してユーザーを検索します。**Home → People** に移動してこのパースペクティブにアクセスできます。

まず、フォローするユーザーのログイン名を知っておく必要があります。検索ボックスに名前を入力すると、システムは名前の自動補完を試行し、部分的な入力に基づく一致を表示します。これらの一致項目からフォローするユーザーを選択すると、パースペクティブが更新され、このユーザーににについての詳細が表示されます。

Follow ボタンをクリックしてユーザーをフォローするよう選択できます。パースペクティブが更新され、ユーザーの詳細およびそれらユーザーの最近のアクティビティーが表示されます。

アクティビティーのタイムライン

Home → Timeline をクリックすると、変更された最近のアセットの一覧が表示され (左側のウィンドウ)、選択したリポジトリーでの変更の一覧が右側に表示されます。アセットをクリックし、アセットのエディターを直接開くことができます (適切なパーミッションがある場合)。

第6章 データモデル

データモデルはデータオブジェクトのモデルです。データオブジェクトは複雑なカスタムデータタイプです (例: Name、Address、および Date of Birth データフィールドのある Person オブジェクト).

データモデルはプロジェクトに格納されるデータモデル定義に保存されます。Red Hat JBoss BRMS は、データオブジェクトを定義するためのデータモデラー、つまりカスタムグラフィカルエディターを提供します。

6.1. データモデラー

データモデラーは、Business Central からプロジェクトデータモデルの一部としてファクトまたはデータオブジェクトを作成するためのビルトインエディターです。データオブジェクトは、POJO として実装されるカスタムデータタイプです。これらのカスタムデータタイプは、インポート後にすべてのリソース (ガイド付きデシジョンテーブルなど) で使用できます。

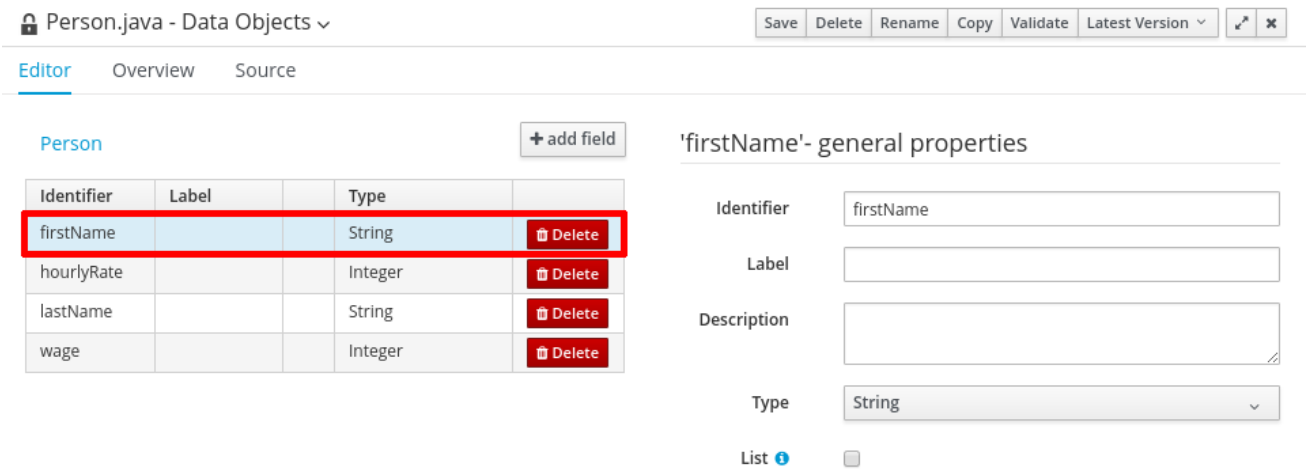
エディターを開くには、Project Authoring パースペクティブを開き、パースペクティブメニューで **New Item → Data Object** をクリックします。既存モデルを編集する必要がある場合、これらのファイルは **Project Explorer** の **Data Objects** の下にあります。

新規モデルの作成時に、このモデルオブジェクトの名前の入力を求めるプロンプトが表示され、(パッケージのコンテキストでの) 場所を選択するよう求められます。正常に実行されると、モデルオブジェクトのフィールドを作成できるエディターが表示されます。

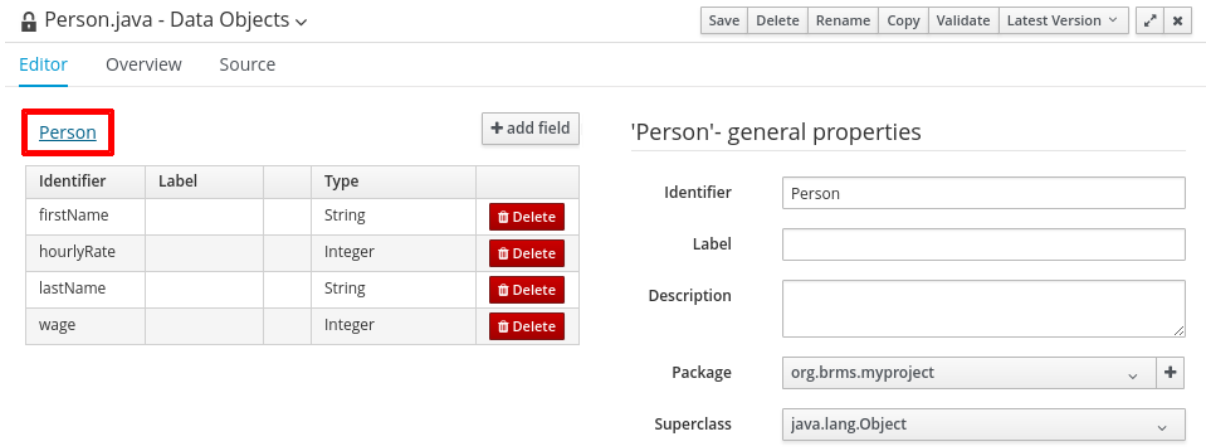
データモデラーは、ソースコードを保持した状態で **Editor** と **Source** タブ間の切り替えをサポートします。これにより、JBDS などの外部ツールでモデルに変更を加え、データモデラーは必要なコードブロックを自動的に更新することができます。

メインエディターウィンドウで、ユーザーは以下を実行できます。

- フィールドの追加または削除
- 指定のフィールドを選択します。フィールドが選択されると、フィールドの情報はすべてのドメインエディターに読み込まれます。



- データオブジェクトクラスを選択します。たとえば、フィールドプロパティーを読み込む代わりに (メインウィンドウで) データオブジェクト名をクリックすると、ドメインエディターがクラスプロパティーを読み込みます。



6.2. 利用可能なフィールドタイプ

データオブジェクトフィールドは、以下のタイプのいずれかに割り当てることができます。

- **Java オブジェクトプリミティブタイプ:**
  BigDecimal、**BigInteger**、**Boolean**、**Byte**、**Character**、**Date**、**Double**、**Float**、**Integer**、**Long**、**Short**、および **String**。
- **Java プリミティブタイプ:**
  boolean、**byte**、**char**、**double**、**float**、**int**、**long**、および **short**。
- **Java 列挙 (Enum) タイプ:**
  現在のプロジェクトで定義されているか、または依存関係としてインポートされる Java 列挙 (enum) タイプ。「依存関係の追加」を参照してください。
- **現在のプロジェクトデータオブジェクト:**
  ユーザー定義のデータオブジェクトは、フィールドタイプとして割り当てられるよう自動的に利用可能になります。
- **プロジェクト依存関係:**
  現在のプロジェクトで Java 依存関係としてインポートされるその他の Java クラス。「依存関係の追加」を参照してください。

6.3. データモデラーのアノテーション

Red Hat JBoss BRMS は、デフォルトですべての Drools アノテーションをサポートし、**Drools & jBPM** ドメイン画面を使用してカスタマイズできます。利用可能なドメイン画面についての詳細は、「データオブジェクトドメイン画面」 を参照してください。

カスタムまたは事前定義のアノテーションを追加または編集するには、**Source** タブに切り替え、ソースコードを直接変更します。ソースコードは、Red Hat JBoss Developer Studio および Business Central の両方で直接編集できます。**Advanced** 画面を使用して任意のアノテーションを管理します。

フィールドを作成するか、永続可能なデータオブジェクトに追加する際に、デフォルトで追加される JPA アノテーションはランタイム時に Red Hat JBoss BRMS が使用できるモデルを生成します。通常、モデルがプロセスで使用されるデフォルトの設定を変更することは推奨されていません。

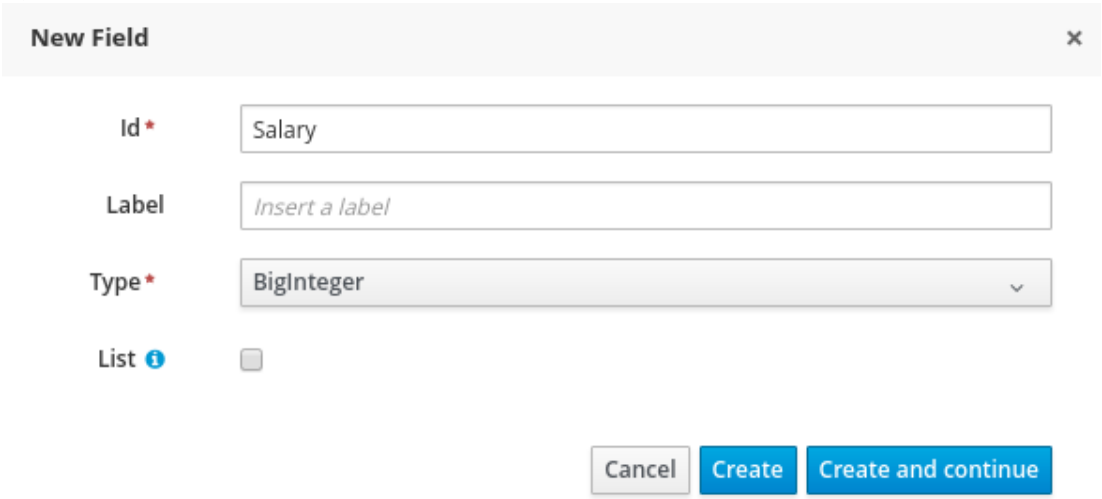
Red Hat JBoss BRMS 6.2 以降は、Hibernate がデフォルトの JPA 実装として利用可能な状態で JPA 固有のアノテーションをサポートします。JPA プロバイダーがクラスパスに読み込まれる場合は、他の JPA アノテーションもサポートされます。

注記

データモデラーでアノテーションを追加する際に、アノテーションクラスをワークベンチクラスパスに置く必要があるか、またはプロジェクト依存関係をアノテーションを含む **.jar** ファイルに追加できます。データモデラーは検証チェックを実行し、アノテーションがクラスパス上にあることを確認します。アノテーションが存在しない場合はプロジェクトのビルドは実行されません。

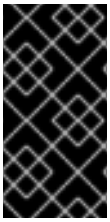
6.4. データオブジェクトの作成

1. Project Authoring パースペクティブで、パースペクティブメニューにある **New Item → Data Object** をクリックします。
2. 名前を入力し、パッケージを選択します。名前はパッケージ内で一意である必要がありますが、2 つの異なるパッケージにまたがる場合には同じ名前のデータオブジェクトが 2 つ存在しても問題ありません。
3. データオブジェクトを永続化するには、**Persistable** チェックボックスにチェックを付けます。
4. **OK** をクリックします。
5. データオブジェクトのフィールドを作成します。
   a. メインエディターウィンドウで **add field** をクリックし、属性 **Id**、**Label** および **Type** を持つオブジェクトにフィールドを追加します。必須属性には * のマークが付けられます。
      - **Id**: データオブジェクト内で一意のフィールド ID。
      - **Label**: **Fields** パネルで使用されるラベル。このフィールドはオプションになります。
      - **Type**: フィールドのデータタイプ。



   b. **Create** をクリックして新規フィールドを作成し、**New field** ウィンドウを閉じます。または、**Create and continue** をクリックして **New field** ウィンドウを開いたままにします。

属性を編集するには、属性を選択し、通常のプロパティー画面を使用します。



データオブジェクトの使用

データオブジェクトを使用するには、データモデルをリソースにインポートしてください。データモデルとリソースの (ガイド付きリソースエディターなど) がいずれも同じパッケージにない場合、それらが同じプロジェクト内にあったとしてもこのタスクは必要になります。

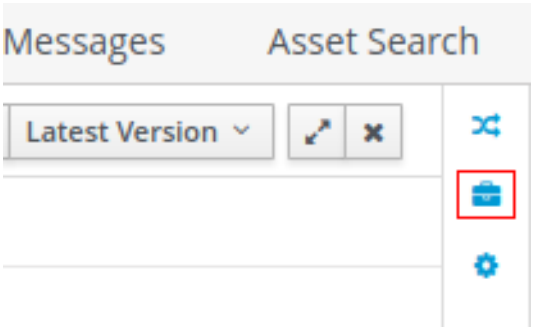
6.5. 永続型データオブジェクト

Red Hat JBoss BRMS 6.2 以降、データモデラーは永続型データモデルの生成をサポートしています。永続型データオブジェクトは JPA 仕様をベースとしています。**Persistable** チェックボックスにチェックを付けると、プラットフォームはデフォルトの永続性設定を使用します。データオブジェクトは以下の 2 つの方法で永続化できます。

- 新規データオブジェクトの作成時。
  新規オブジェクトの作成時には、「データオブジェクトの作成」 の手順に従ってください。
- データオブジェクトがすでに作成されている場合。

すでに作成されているデータオブジェクトを永続化するには、以下を実行します。

1. Business Central でデータオブジェクトを開きます。
2. **Editor** タブをクリックします。
3. 右側のメニューから **Persistence** アイコンを選択します。



4. **Persistable** にチェックを付けます。
5. **Save** をクリックして変更を保存します。

6.6. データオブジェクトドメイン画面

以下のドメイン画面タブは、データオブジェクトエディター画面の右側から選択できます。

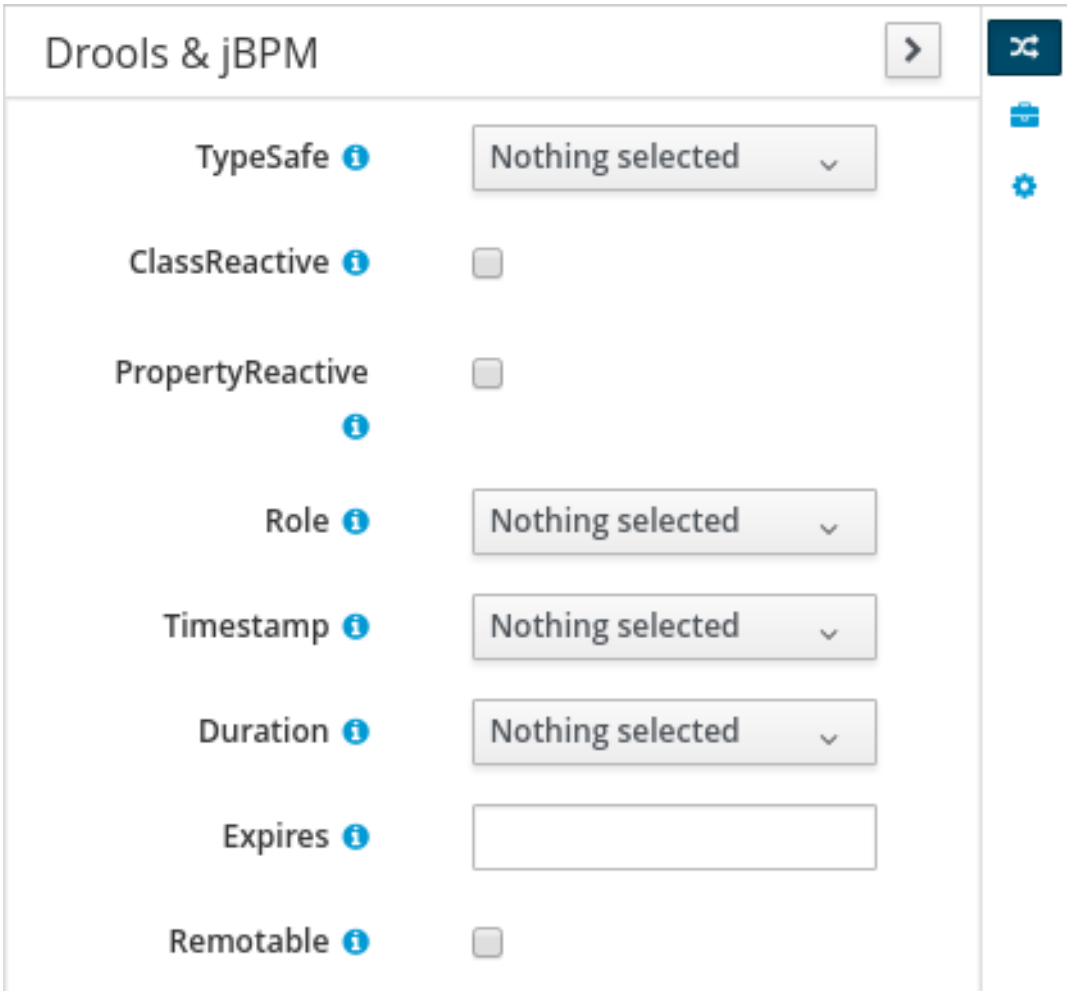
Drools & jBPM
Drools & jBPM 画面では、Drools 固有の属性の設定が可能です。

Business Central のデータモデラーは、ファクトモデルクラスおよび属性の事前定義のアノテーションの編集をサポートしています。以下の Drools アノテーションはサポートされており、**Drools & jBPM** インターフェースを使ってカスタマイズできます。

- **TypeSafe**
- **ClassReactive**
- **PropertyReactive**

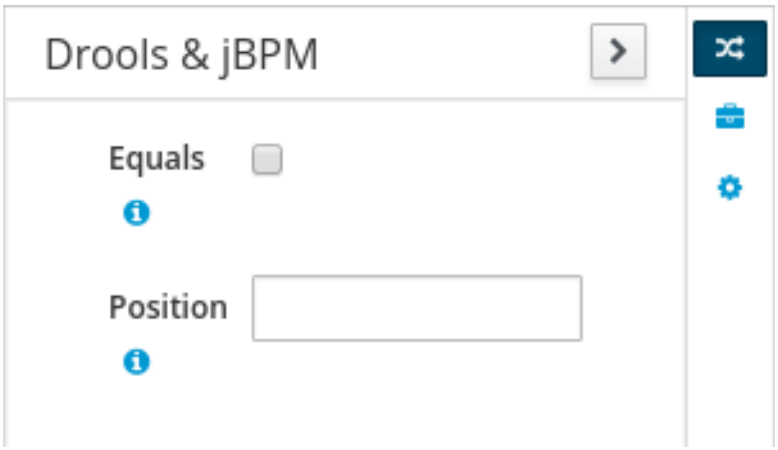
- **Role**
- **Timestamp**
- **Duration**
- **Expires**
- **Remotable**

図6.1 Drools & jBPM クラスビュー



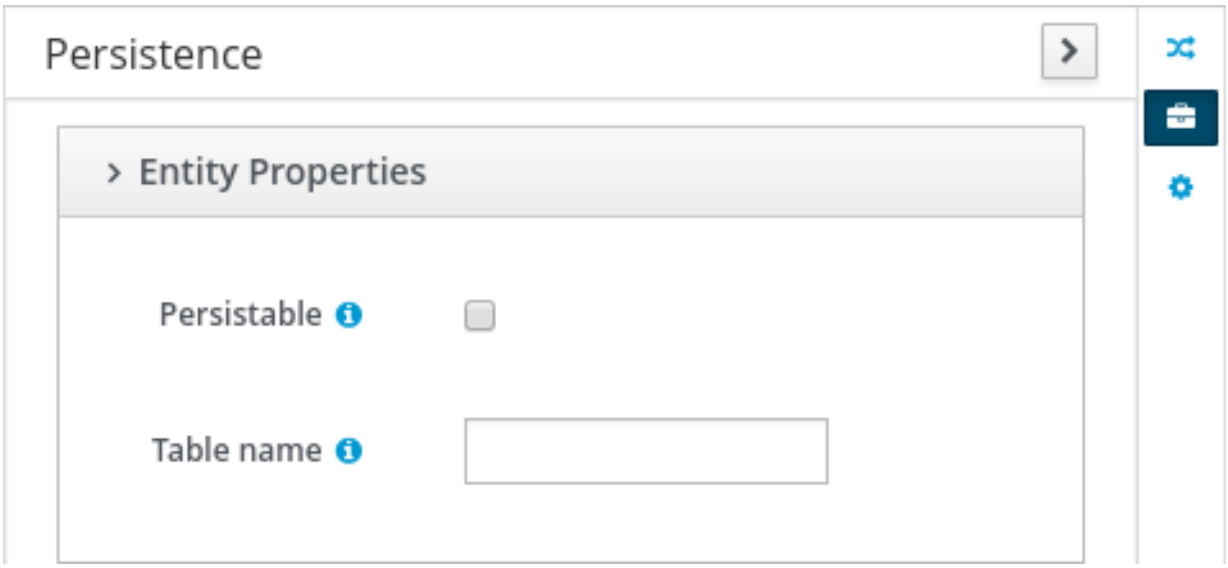
ファクトモデル内のフィールドについて、**position** および **Equals** アノテーションがサポートされています。特定のフィールドが選択されている場合の **Drools & jBPM** 画面は以下のようになります。

図6.2 Drools & jBPM フィールドビュー



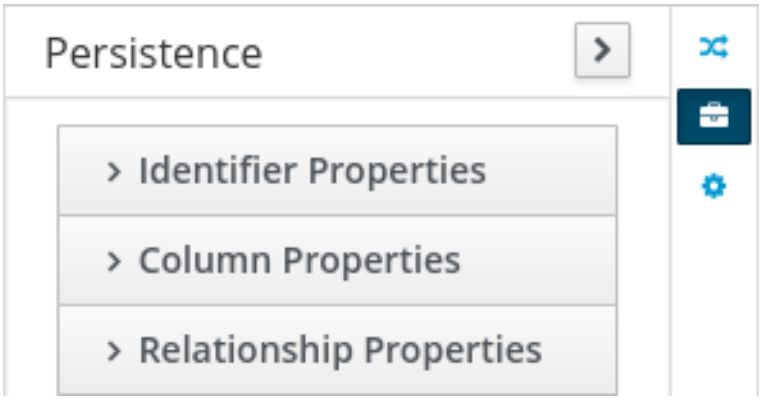
永続性
Persistence 画面を使用して、永続性について基本的な JPA アノテーションの属性を設定できます。アノテーションの微調整または特定のアノテーションを追加を実行するには、Advanced 画面を使用します。

図6.3 Class Persistence ビュー



特定のフィールドが選択されている場合の Persistence 画面は以下のようになります。

図6.4 Field Persistence ビュー



以下のアノテーションは Persistence 画面で管理できます。

表6.1 タイプアノテーション

アノテーション	データオブジェクトが永続型の場合に自動生成される
javax.persistence.Entity	Yes
javax.persistence.Table	No

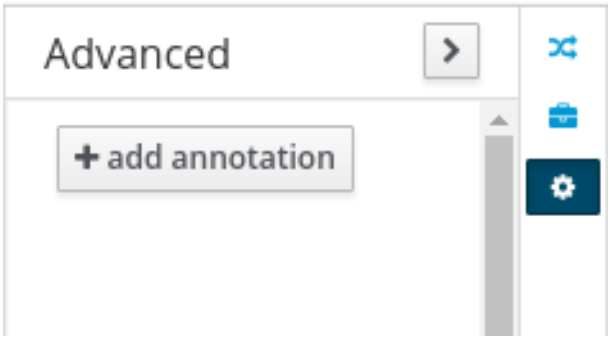
表6.2 フィールドアノテーション

アノテーション	データオブジェクトが永続型の場合に自動生成される	関連する UI 要素
javax.persistence.Id	Yes	Is 識別子
javax.persistence.GeneratedValue	Yes	生成戦略
javax.persistence.SequenceGenerator	Yes	シーケンスジェネレーター
javax.persistence.Column	No	列プロパティー属性
javax.persistence.OneToOne	No	リレーションシップタイプ
javax.persistence.OneToMany	Yes: フィールドに 1 つまたは複数の値がある場合	リレーションシップタイプ
javax.persistence.ManyToOne	Yes: フィールドに複数の値がある場合	リレーションシップタイプ
javax.persistence.ManyToMany	No	リレーションシップタイプ
javax.persistence.ElementCollection	Yes: 新規フィールドに、Integer、Boolean、String などの基本 Java タイプが 1 つまたは複数含まれる場合に UI で生成される。このアノテーションは **Persistence** 画面ツールでは編集できません (代わりに **Advanced** 画面ツールを使用します)。	**list** とマークされるフィールドで作成される。

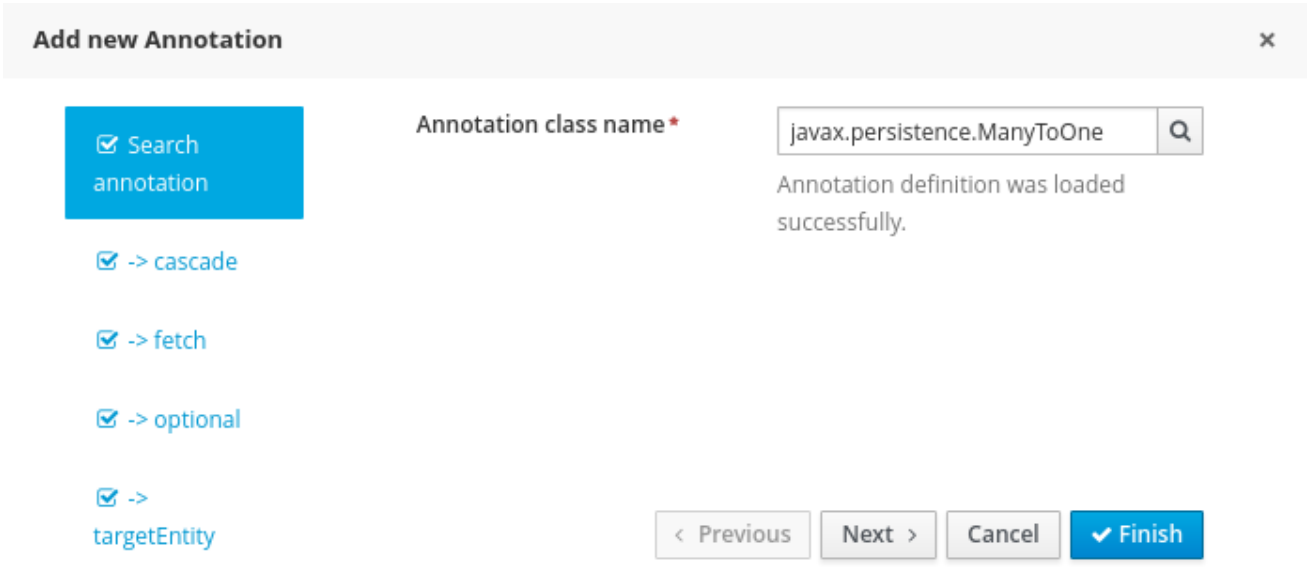
その他すべての JPA アノテーションは、**Advanced** 画面を使用して追加できます。

Advanced

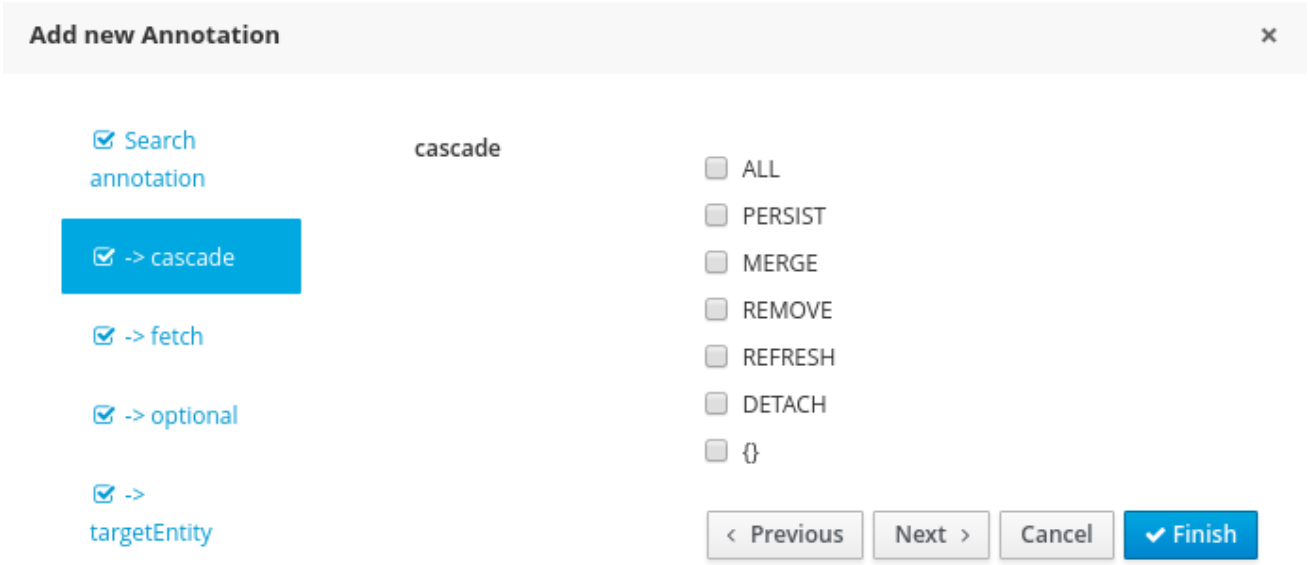
Advanced 画面はアノテーションの微調整に使用されます。アノテーションは、Advanced Domain 画面を使用して設定し、追加し、削除することができます。クラスパスにあるすべてのアノテーションについてこれらを実行できます。



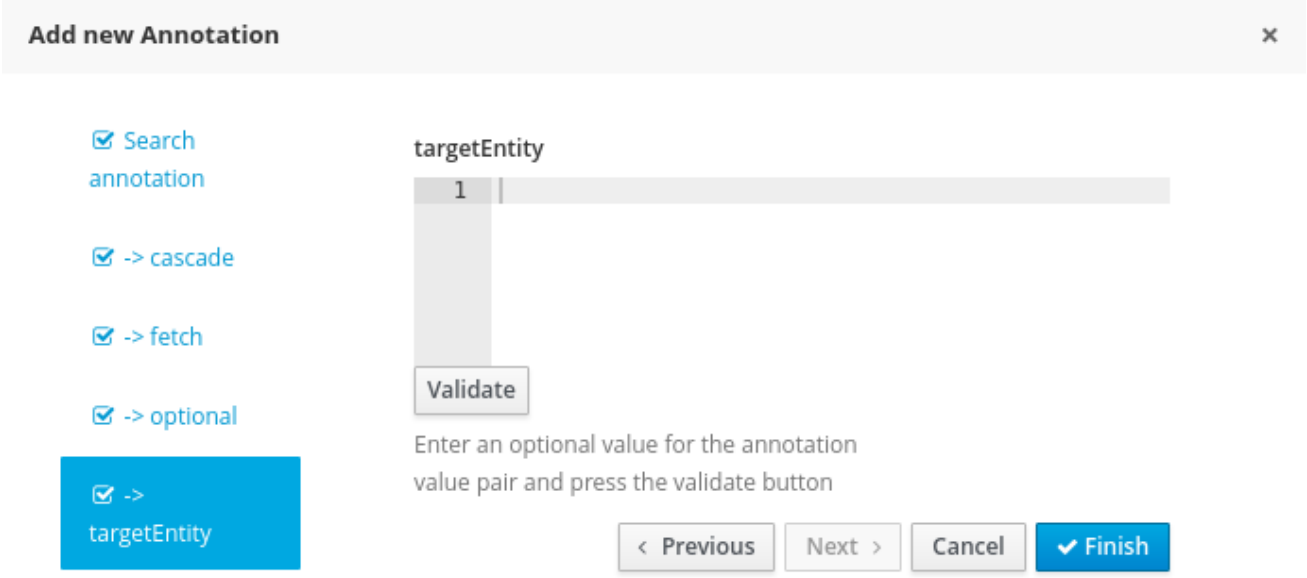
add annotation オプションをクリックした後に Add new Annotation ウィンドウが表示されます。アノテーションの完全修飾名を入力する必要があり、search アイコンを押すと、アノテーションの定義がウィザードに読み込まれます。その後に各種のアノテーションパラメーターを設定できます (必須パラメーターには * のマークが付けられます)。



ウィザードで指定パラメーターに適したエディターが提供される場合もあります。



カスタマイズされたエディターを提供できない場合、ウィザードは汎用パラメーターエディターを提供します。



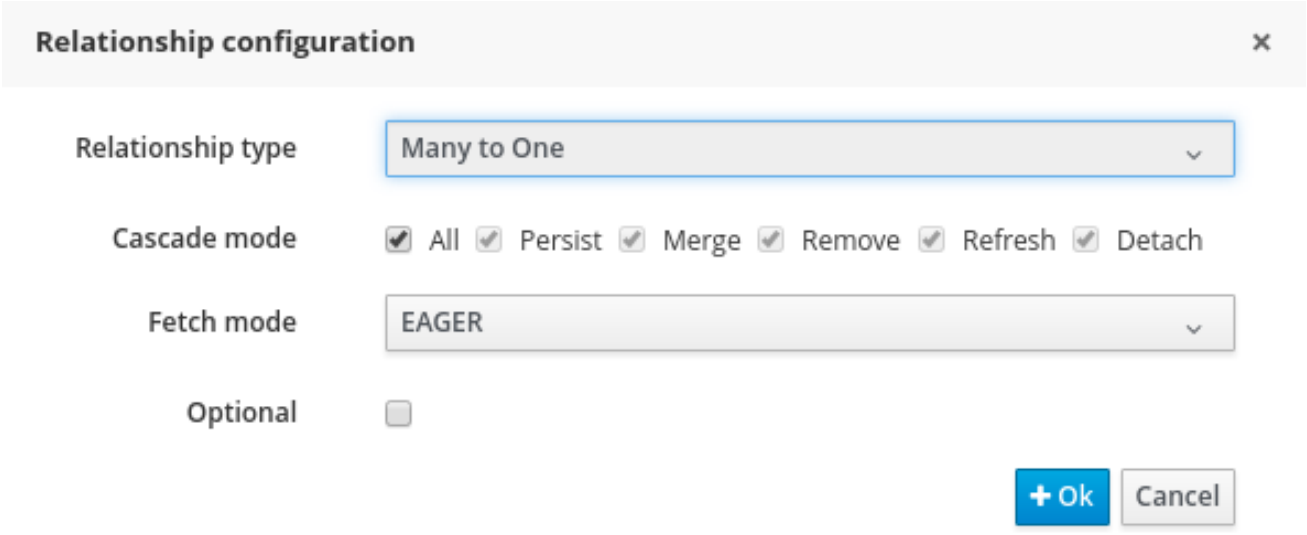
すべての必須パラメーターを入力した後に、**Finish** ボタンが有効にされ、アノテーションを指定のフィールドまたはデータオブジェクトに追加できます。

6.7. データオブジェクト間の関係の設定

属性タイプが別のデータオブジェクトとして定義される場合、関係は、オブジェクト属性一覧の 記号で特定され、定義されます。このアイコンをクリックして、データオブジェクトの定義にジャンプし、表示および編集を行います。

関係のカスタマイズが必要になるのはデータオブジェクトが永続型である場合のみです。

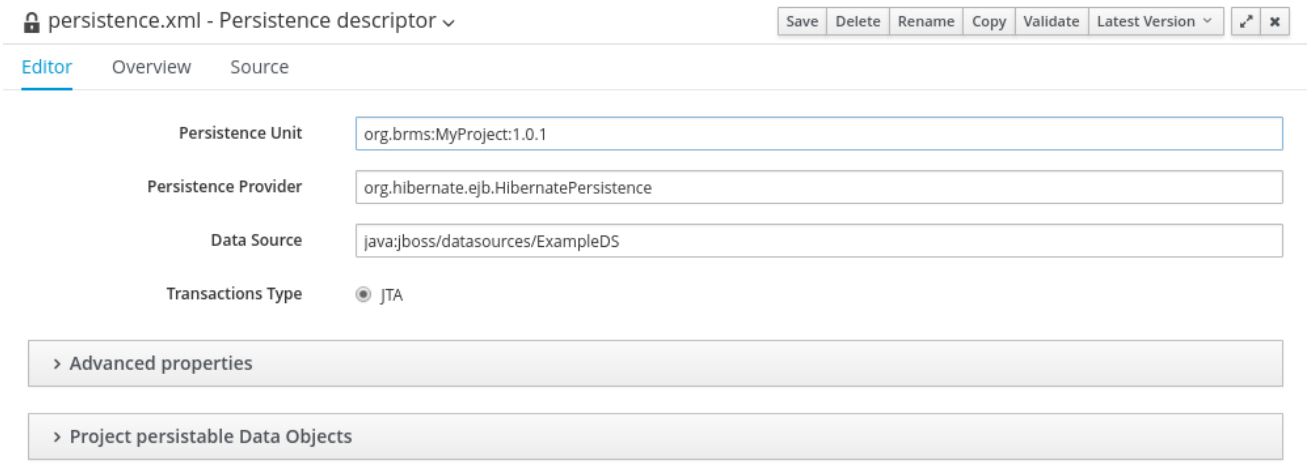
関係は、関係が設定された属性を選択し、右側にある **Persistence** ボタンを選択して設定できます。**Relationship Properties** の下で、**Relationship Type** プロパティーの編集オプションをクリックします。



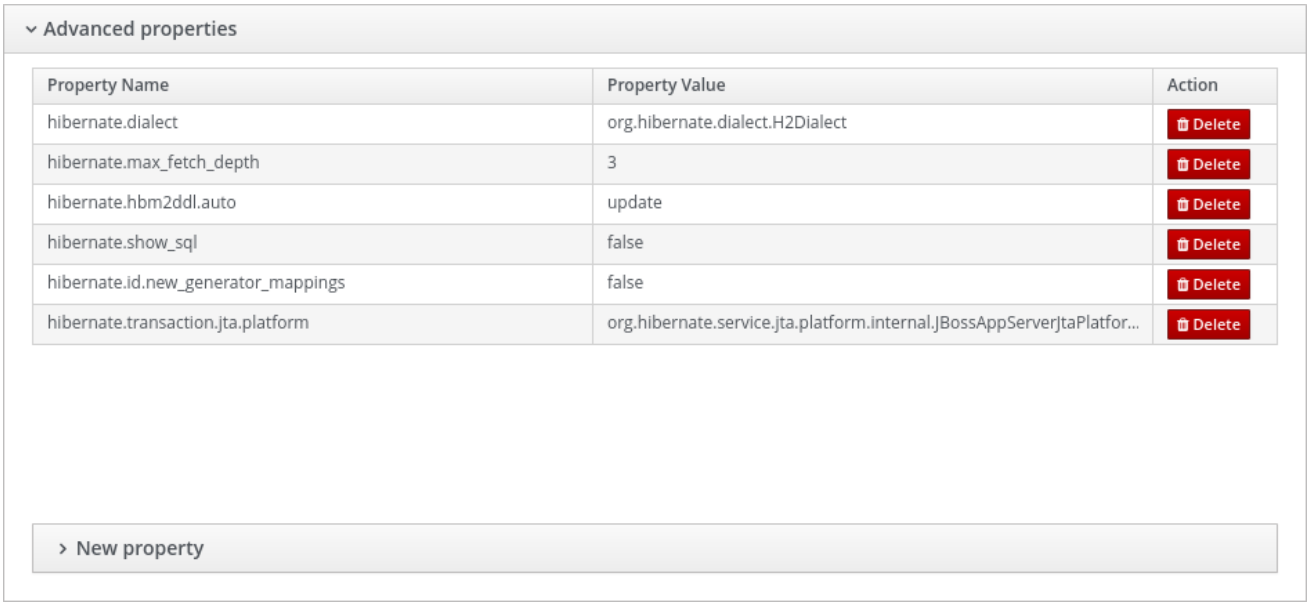
異なるデータオブジェクトで使用されるデータオブジェクトの削除を試行すると、**Usage Detected** 画面が表示されます。ここからオブジェクトを削除することもできますが、この場合、結果として生じるエラーが解決されるまでプロジェクトのビルドは停止します。

6.8. 永続性記述子

Business Central には、デフォルトの永続性設定のある **persistence.xml** ファイルが含まれます。永続性設定を行うには、**Project Settings: Project General Settings → Persistence descriptor** をクリックします。

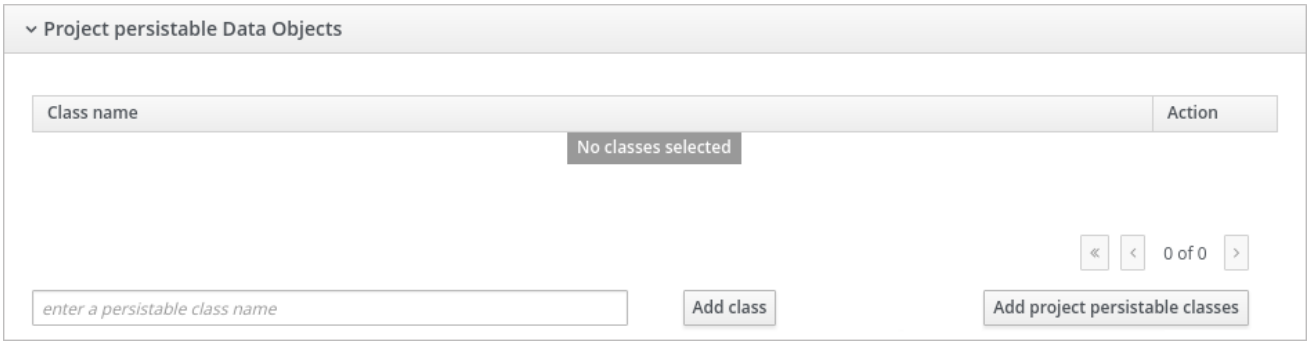


Advanced properties セクションを使用してプロパティーを変更したり、削除したり、追加したりします。



永続性記述子で **Project persistable Data Objects** セクションを開く場合、2 つのボタンが表示されます。

- **Add class**: ユーザーはエンティティーとして宣言されるよう任意のクラスを **persistence.xml** ファイルに追加できます。
- **Add project persistable classes**: 現行プロジェクトにすべての永続型データオブジェクトを自動的に読み込みます。


Project persistable Data Objects
Class name
Action
No classes selected
0 of 0
enter a persistable class name
Add class
Add project persistable classes

第7章 ルールの記述

7.1. ルールの作成

手順: 新規ルールの作成

1. **Project Explorer** ビューで、以下を実行します。
   a. Project Explorer の Project ビューにいる場合、組織単位、リポジトリーおよびルールを作成する必要のあるプロジェクトを選択します。
   b. Project Explorer の Repository ビューにいる場合、**src/main/resources/** とルールテンプレートのプロジェクトフォルダーを作成する必要のある **SUBFOLDER/PACKAGE** に移動します。
2. パースペクティブメニューで、**New Item → Guided Rule** に移動します。
3. **Create new Guided Rule** ダイアログウィンドウで、パッケージの詳細を定義します。
   - **Guided Rule** テキストボックスで、ルールの名前を入力し、**OK** をクリックします。DSL を使用するには **Use Domain Specific Language (DSL)** にチェックを付けます。詳細は、「ドメイン固有言語エディター」 を参照してください。
4. 新規のガイド付きルールが選択されたプロジェクトの下に作成されます。

7.2. ルールの編集

7.2.1. アセットエディターを使用したルールの編集

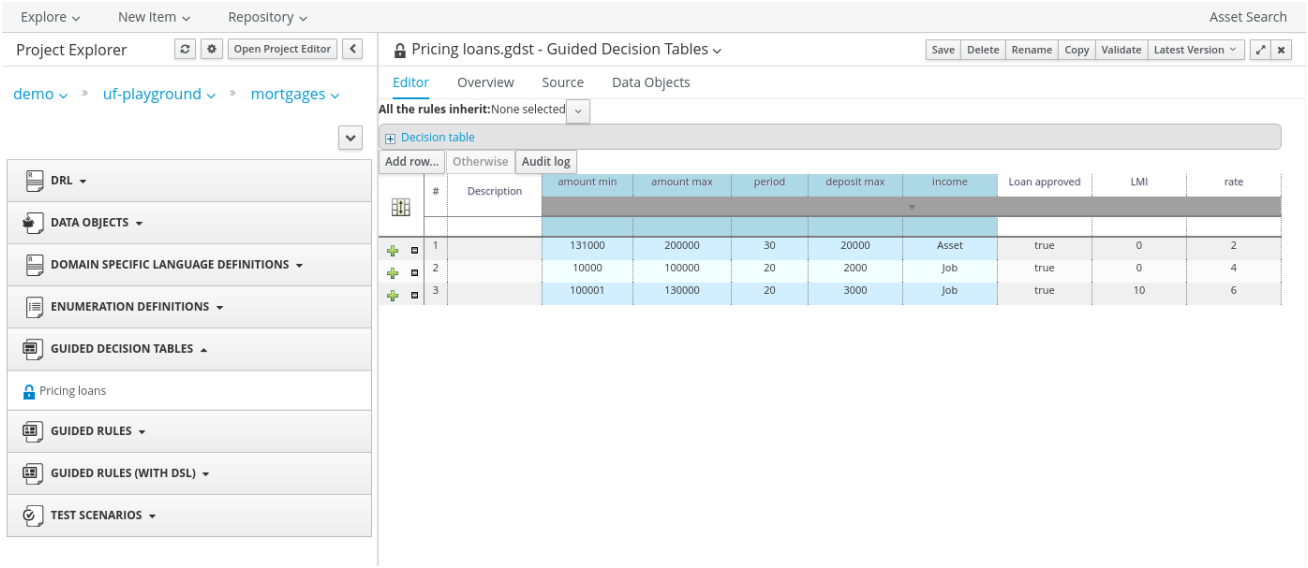
アセットエディターはアセットに関する情報へのアクセスを提供し、ユーザーはこのエディターを使ってアセットを編集することができます。

エディターには、**Editor**、**Overview**、**Source**、および **Data Objects** タブが含まれます。

Editor タブ

Editor タブを使ってアセットを編集することができます。Edit タブで利用可能なオプションは、編集中のアセットによって異なります。

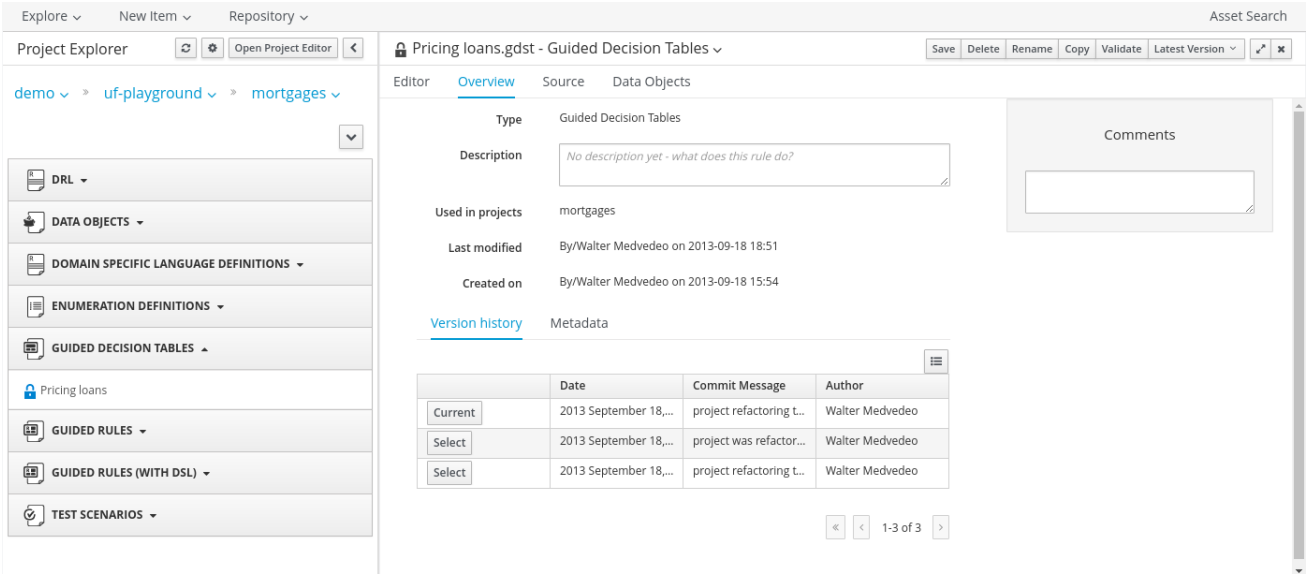
図7.1 ガイド付きデシジョンテーブル - Editor タブ



Overview タブ

Overview 画面には、アセットの汎用データおよびバージョン履歴が表示されます。この画面から、ユーザーは他のメタデータの詳細を編集したり、選択したアセットに固有の説明およびディスカッションを追加したりできます。

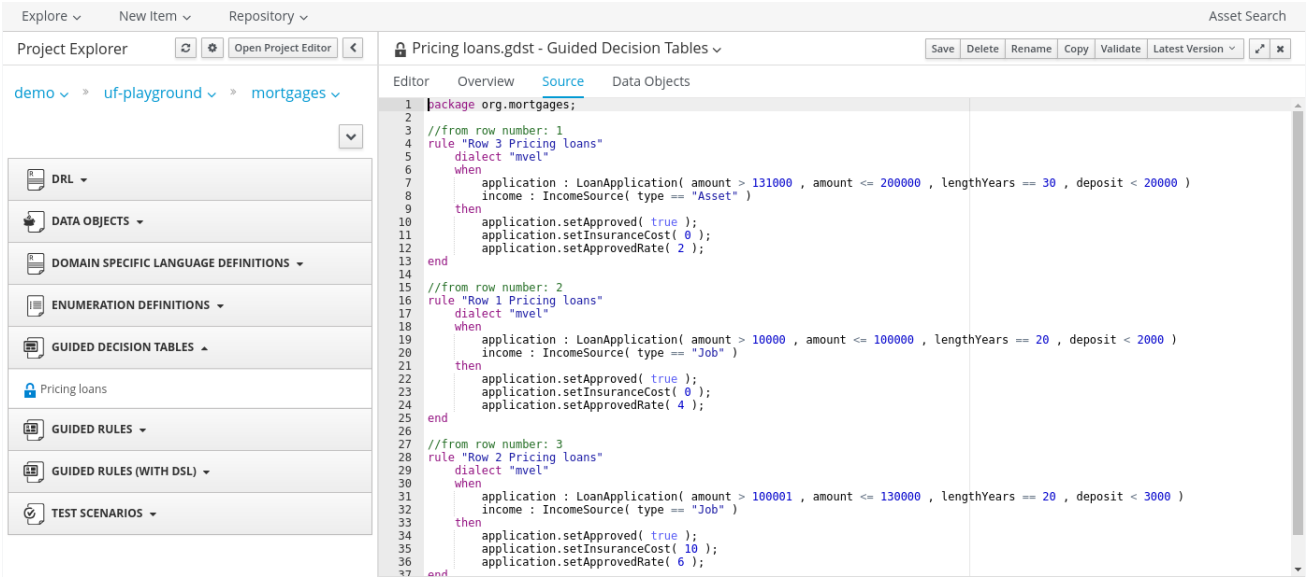
図7.2 ガイド付きデシジョンテーブル - Overview タブ



Source タブ

Source タブには、選択したアセットの DRL ソースが表示されます。

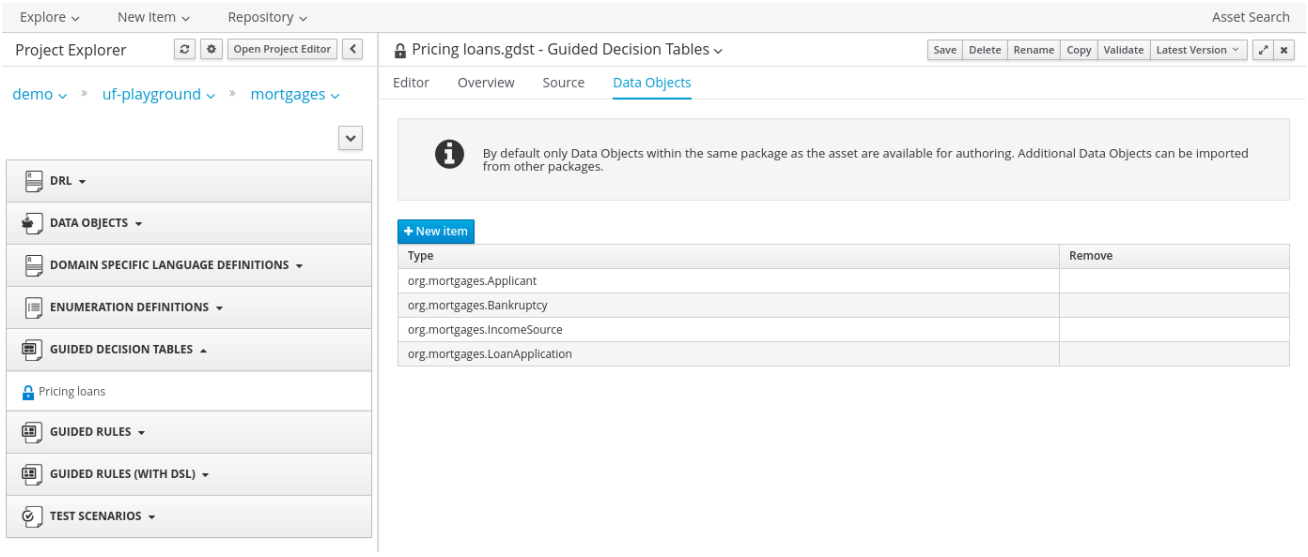
図7.3 ガイド付きデシジョンテーブル - Source タブ



Data Objects タブ

Data Objects タブでは、プロジェクトで使用されるインポートのセットが提案されます。それぞれのアセットには独自のインポートと、ユーザーが使用する必要がある可能性のある提案されるファクトタイプ (つまりデータオブジェクト) が含まれます。データオブジェクトについての詳細は、「データオブジェクトの作成」 を参照してください。

図7.4 ガイド付きデシジョンテーブル - Data Objects タブ

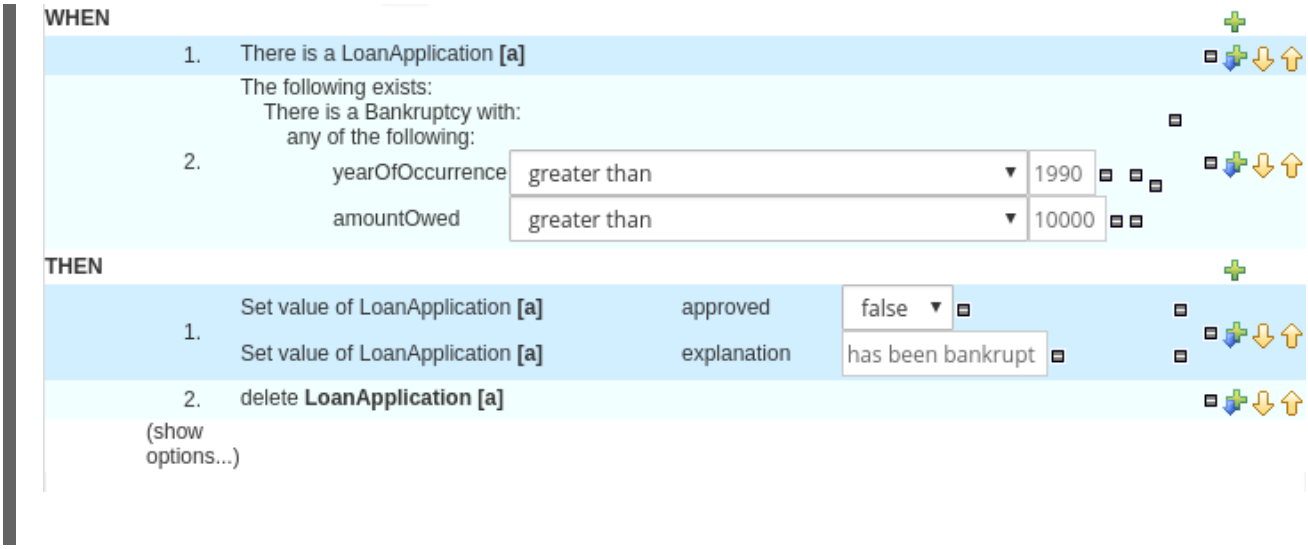


7.2.2. ガイド付きルールエディターでのビジネスルール

ビジネスルールはガイド付きルールエディターで編集します。ガイド付きルールエディターは、編集中のルールのオブジェクトモデルに基づいてユーザーに入力を求めるプロンプトを出します。

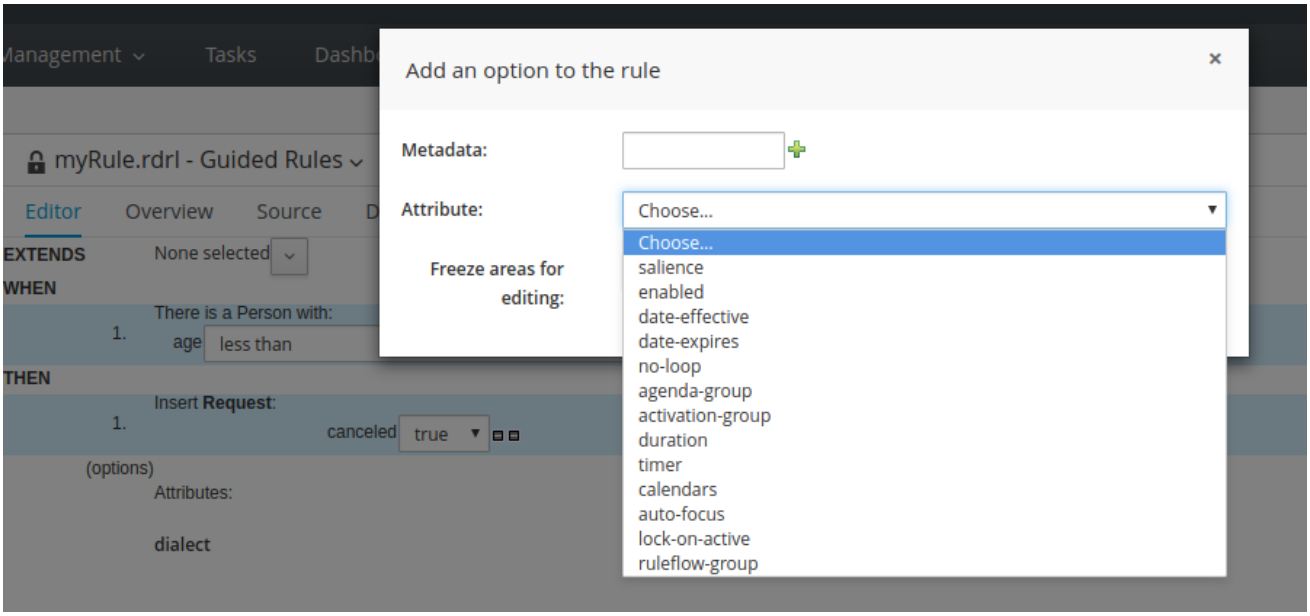
別のパッケージに属するルールで使用されるアセットはルールにインポートする必要があります。同じパッケージのアセットはデフォルトでインポートされます。**Data Objects** タブでインポートを定義します。

> **例7.1 ガイド付きルールエディター**



ガイド付きルールエディターは、**When** と **Then** の部分で構成されます。**Then** の部分には、**(show options...)** リンクがあり、ここから ruleflow-group、salience などの追加のルール属性を設定できます。ガイド付きルールエディターについての詳細は、「ガイド付きルールテンプレート」 を参照してください。

図7.5 ルール属性



7.2.3. Package White List (パッケージホワイトリスト) の使用によるファクトの絞り込み

ルールの作成および変更時に、**package-names-white-list** というファイルを使用して利用可能なファクトを絞り込むことができます。このファイルを使用すると、読み込み済みで表示されるファクトのグループを絞り込むことができます。これは、新規ルールの作成中にこれらのファクトの読み込みを加速させるのに役立ちます。

ルートディレクトリーに新規プロジェクトを作成すると、空の **package-names-white-list** ファイルが **pom.xml** および **project.imports** プロジェクトファイルと共に自動作成されます。既存プロジェクトについては、**package-names-white-list** ファイルを手動で作成します。Business Central では、**Project Explorer** のプロジェクトのリポジトリービューでファイルを表示できます。

デフォルト (ファイルが空の状態) で、プロジェクトの依存関係からのファクトは制限される一方で、プロジェクト自体にあるすべてのファクトは表示されます。

パッケージを定義するためのルール
package-names-white-list ファイルは、各行に単一のパッケージ名を受け入れるテキストファイルです。パッケージには、ワイルドカードなどが含まれる場合があります。

- com.redhat.finance: com.redhat.finance パッケージから**のみ** ファクトを受け入れます。そのため、**com.redhat.finance.Person** と **com.redhat.finance.Salary** は許可されますが、**com.redhat.finance.senior.Management** は許可されません。

- com.redhat.finance.*: com.redhat.finance パッケージ **のみ** のサブパッケージからのファクトを受け入れます。そのため、**com.redhat.finance.senior.Management** と **com.redhat.finance.junior.Management** は許可されますが、**com.redhat.finance.Person** は許可 **されません**。

- com.redhat.finance.**: これは上記の 2 つのルールの組み合わせになります。**com.redhat.finance.Person** と **com.redhat.finance.senior.Management**、さらに **com.redhat.finance.really.senior.Management** クラスを受け入れます。

適切なエントリーを **package-names-white-list** ファイルに追加することで、依存関係からの特定のパッケージを組み込むことができます。詳細は、「依存関係の追加」を参照してください。

7.2.4. ルールの構成

ルールは複数の部分で構成されます。

When

ルールの when 部分には、満たす必要のある条件を入れます。たとえば、ローンとして銀行が貸付をする場合、顧客が 21 歳以上であるよう指定する場合があります。これは、顧客が 21 歳以上であることを判断するために when を使用して表現します。

Then

ルールの then 部分は、ルールの条件部分が満たされた場合に実行すべきアクションが入ります。たとえば、when 部分には「顧客が 21 歳以下の場合」、then 部分には「申請者が年齢制限に達していないためローンは却下」などが入ります。

オプション

ルールでは、Salience (優先順位) などのオプション属性を定義することができます。

ガイド付きルールエディターを使うと、ルールの When (条件) 部分に条件を追加し、Then (アクション) 部分にアクションを追加することができます。たとえば、ローン申請の際に 21歳未満の申請者に保証人がいる場合、銀行がそのローン申請を承認するか決定することができます。

7.2.5. Salience (優先順位)

ルールごとに Salience の値をもたせることができます。Salience には整数の値が入り、デフォルトは 0 となっています。Salience の値は、ルールの優先順位を表しており、値が高いと優先順位が高くなります。Salience 値は、正負どちらでも利用できます。

7.2.6. ルールへの条件やアクションの追加

手順: 条件またはアクションのルールへの追加

1. ガイド付きルールエディターの When 部分の + アイコンをクリックして条件を追加するか、ガイド付きルールエディターの Then 部分の + アイコンをクリックしてアクションを追加してください。

2. メニューから条件やアクションを選択して、**Ok** をクリックします。ルールが属するパッケージに DSL (ドメイン固有言語) の文を含めるように設定した場合、DSL 文をメニューから選択することができます。

3. 条件またはアクションに入力 (日付、True または False、整数、その他の入力タイプ) が必要な場合、必要となる値を入力します。

7.2.7. ファクトタイプへのフィールドの追加

ガイド付きルールエディターを使うと、ルールの When (条件) 部分に条件を追加し、Then (アクション) 部分にアクションを追加することができます。例えば、ローン申請の際に 21歳未満の申請者に保証人がいる場合、銀行がそのローン申請を承認するか決定することができます。

保証人を条件に追加するには、まず **guarantor** フィールドを、担保モデルの申請ファクトタイプ (Business Central のデータオブジェクト) に追加します。フィールドをファクトタイプに追加する方法については、「データモデラー」 を参照してください。

保証人フィールドに申請者のファクトタイプが追加されたため、ルールを変更し、保証人を条件として含めることができます。

7.2.8. テクニカルルール (DRL)

テクニカルルール (DRL) はテキストとして保存され、これらのルールは JBoss Enterprise BRMS ユーザーインターフェースで管理されます。DRL ファイルには、1 つ以上のルールが含めることができます。ルールの条件とアクション部分は、それぞれ「when」と「then」になります。

Red Hat JBoss Developer Studio には、DRL ファイルの作成、編集、デバッグのツールが含まれており、これらはそれぞれの目的に合わせて使用できます。ただし、DRL ルールは Red Hat JBoss Enterprise BRMS ユーザーインターフェースで管理することができます。DRL エディターは Java、DRL および XML の構文強調表示を行います。

テクニカルルール (DRL) の例

```
rule "approval"
  salience 100 // This can short-circuit any processing.
  when
    a : Approve()
    p : Policy()
  then
    p.setApproved(true);
    System.out.println("APPROVED:" + a.getReason());
end
```

7.3. デシジョンテーブル

7.3.1. スプレッドシートのデシジョンテーブル

ルールは、スプレッドシートのデシジョンテーブルに保存できます。スプレッドシートの各行はルールで、各列は条件、アクション、オプションのいずれかになります。『**Red Hat JBoss BPM Suite Development Guide**』では、デシジョンテーブルの使用方法の詳細が説明されています。

7.3.2. スプレッドシートのデシジョンテーブルのアップロード

手順: スプレッドシートのデシジョンテーブルのアップロード

1. 既存のスプレッドシートをアップロードするには、**New Item → Decision Table (Spreadsheet)** を選択します。

2. スプレッドシートの名前を入力し、**Choose file...** をクリックし、スプレッドシートを選択します。**.xls** または **.xlsx** ファイルを選択できます。終了したら、**Ok** をクリックします。

アップロードされたスプレッドシートをガイド付きデシジョンテーブルに変換するには、以下を実行します。

1. アップロードされたスプレッドシートは、プロジェクト画面のメニューバーにある **Validate** ボタンをクリックして検証します。

2. **Convert** をクリックします。

7.3.3. スプレッドシートのデシジョンテーブルサンプル

ここでは、注文されたアイテムの配送料金を記載するオンラインショッピングサイトの簡単な例について見てみましょう。このサイトでは、以下の条件に基づいて配送料を無料とすることに同意しています。

- 注文されたアイテムの数が 4 点以上であり、合計金額が $300 以上である。
- 購入日から標準配送日とされている 4 ないしは 5 営業日後に配送される。

記載される配送料金は以下のようになります。

表7.1 注文金額が $300 未満の場合

アイテム数	配送日	配送料金、N = アイテム数
3 点以下	翌日	$35
	2 日後	$15
	標準	$10
4 点以上	翌日	N*7.50
	2 日後	N*3.50
	標準	N*2.50

表7.2 注文金額が $300 を超える場合

アイテム数	配送日	配送料金、N = アイテム数
3 点以下	翌日	$25
	2 日後	$10
	標準	N*1.50

アイテム数	配送日	配送料金、N = アイテム数
4 点以上	翌日	N*5
	2 日後	N*2
	標準	無料

上記の条件はスプレッドシートでは以下のように表されます。

Calculating Shipping Charges												
Purchase Amount	**Over $300**						**Less than $300**					
Number of Items	**3 or less**			**4 or more**			**3 or less**			**4 or more**		
Delivery Day	Next	2nd day	Standard	Next	2nd day	Standard	Next	2nd day	Standard	Next	2nd day	Standard
Shipping Charges ($)	25	10	N * 1.50	N * 5	N * 2	*FREE*	35	15	10	N * 7.50	N * 3.50	N * 2.50

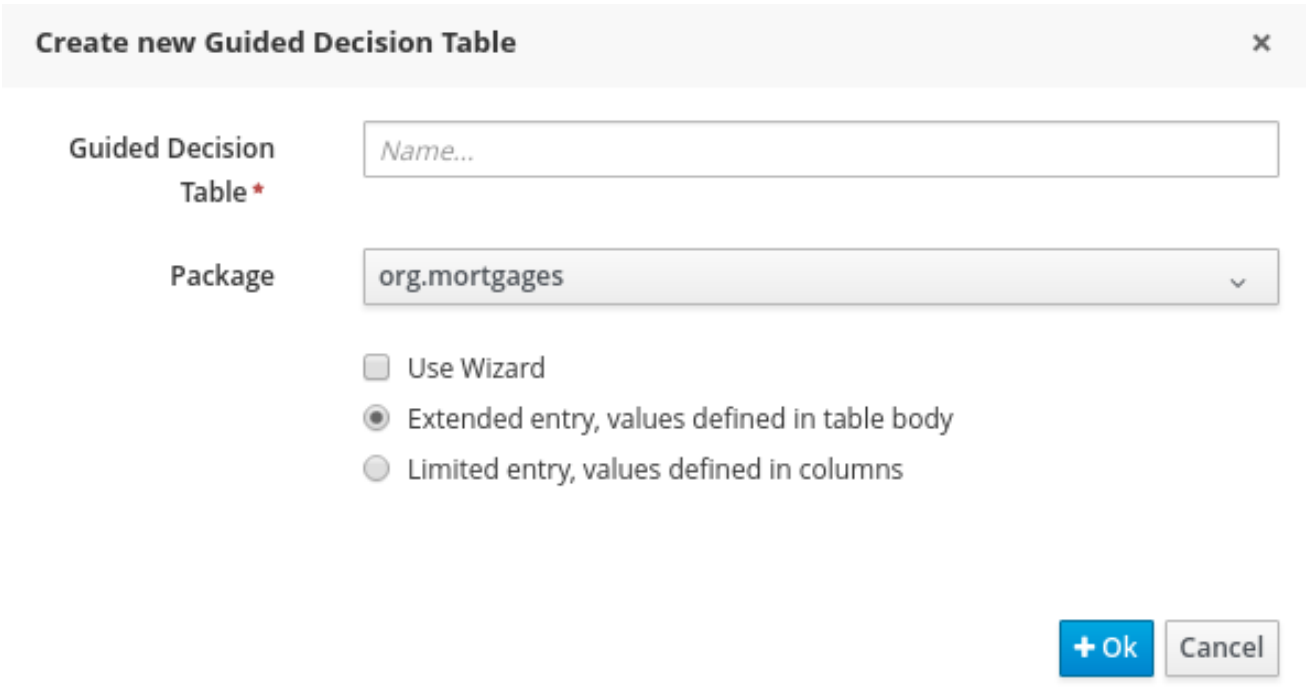
7.4. WEB ベースのガイド付きデシジョンテーブル

7.4.1. Web ベースのガイド付きデシジョンテーブル

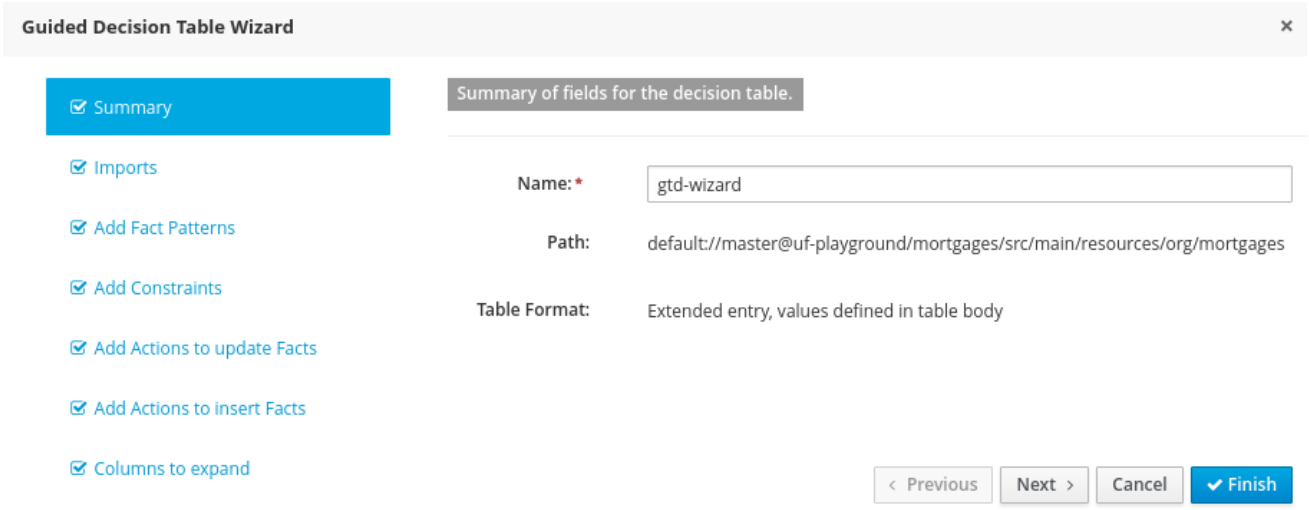
(web ベースの) ガイド付きデシジョンテーブル機能は、どのファクトやフィールドを利用すればデシジョンテーブル作成に役立つかを確認する点で、ガイド付きエディターのように機能します。

ルール属性、メタデータ、条件、アクションは、表形式で定義可能で、関連ルールの多くをすばやく入力できるようにします。Web ベースのデシジョンテーブルのルールは他のルールアセットと同様に、DRL にコンパイルされます。

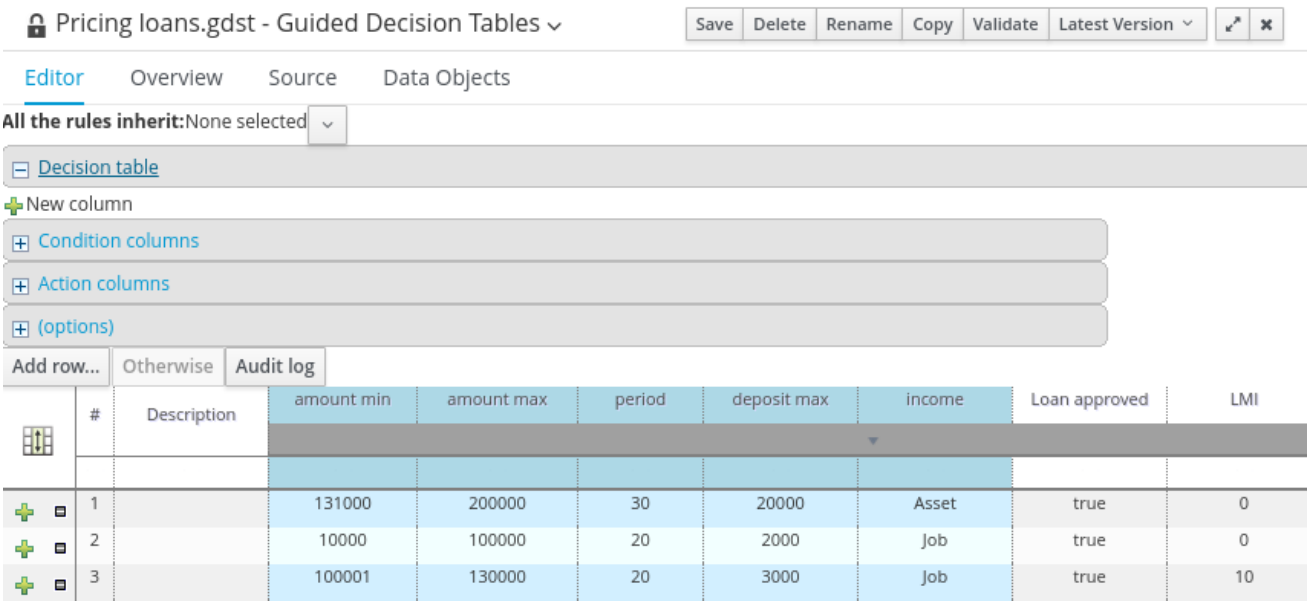
新規デシジョンテーブルを作成するには、**New Item → Guided Decision Table** をクリックします。表の名前を入力し、拡張エントリーまたは制限エントリー表のどちらかを選択します (「デシジョンテーブルのタイプ」)。オプションで、ガイド付きデシジョンテーブルウィザードを使用するよう選択します。



終了したら、**OK** をクリックします。ウィザードを選択しない場合は、ガイド付きデシジョンテーブルのエディターが表示されます。ウィザードを選択した場合、ウィザードの最初の画面が表示されます。



ウィザードでは、インポート、ファクト、パターンおよび列を定義することができますが、行は定義できません。行はガイド付きデシジョンテーブルエディターに追加され、これはウィザードの終わり (またはウィザードを使用しない場合は直接) 表示されます。



ガイド付きデシジョンテーブルで構成される独自のアプリケーションをビルドする場合、クラスパスに必要な依存関係が追加されていることを確認してください。ガイド付きデシジョンテーブルの依存関係についての詳細は、『**Red Hat JBoss BPM Suite Development Guide**』の「**Dependency Management for Guided Decision Tables, Scorecards, and Rule Templates**」セクションを参照してください。

7.4.2. デシジョンテーブルのタイプ

大別すると、以下の 2 つのタイプのデシジョンテーブルがサポートされています。

- 拡張エントリー
- 制限エントリー

拡張エントリー

拡張エントリーのデシジョンテーブルは、列定義で値以外のパターン、フィールド、演算子を指定するタイプです。値またはステート (state) はデシジョンテーブルの本体に保持されます。エントリーをリストの値に制限して使用される値の範囲を制限することが通例となっていますが、必須ではありませ

ん。Business Central は Java 列挙またはデシジョンテーブルの「optional value lists (オプションの値リスト」を使用して値エントリーを制限する方法をサポートしています。

制限エントリー

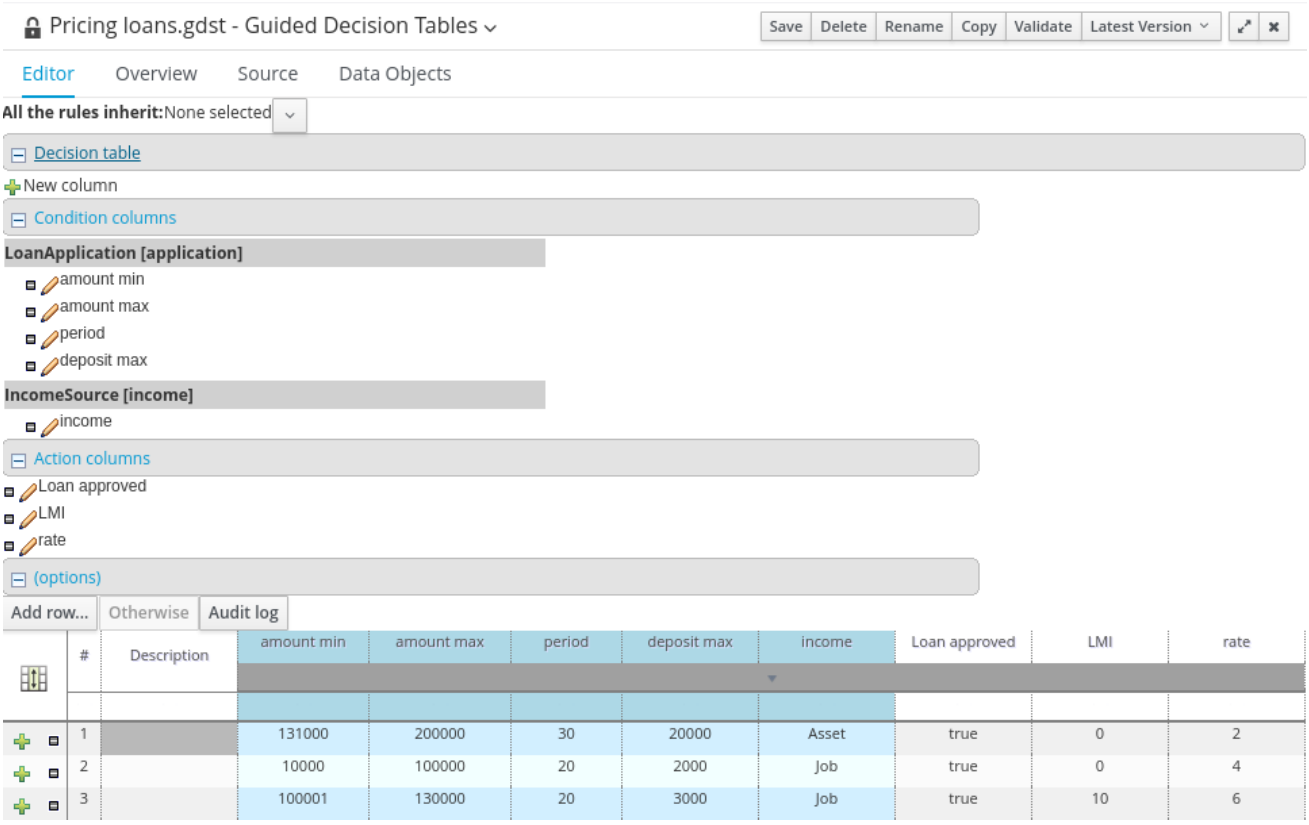
制限エントリーのデシジョンテーブルは、列定義でパターン、フィールドおよび演算子に加えて値を指定します。デシジョンテーブルの本体に保持されるデシジョンテーブルのステート (state) は、正の値 (チェックボックスにチェックが付けられる) の場合は列が適用されることを意味するブール値です。負の値 (チェックボックスにチェックが付けられない) は列が適用されないことを意味します。

7.4.3. 列の設定

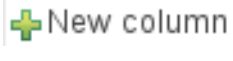
列の制限についての説明は、「条件列」 を参照してください。

デフォルト値を設定することはできますが、通常セルに値が入っていない場合は、その制約は適用されません。

図7.6 列の設定



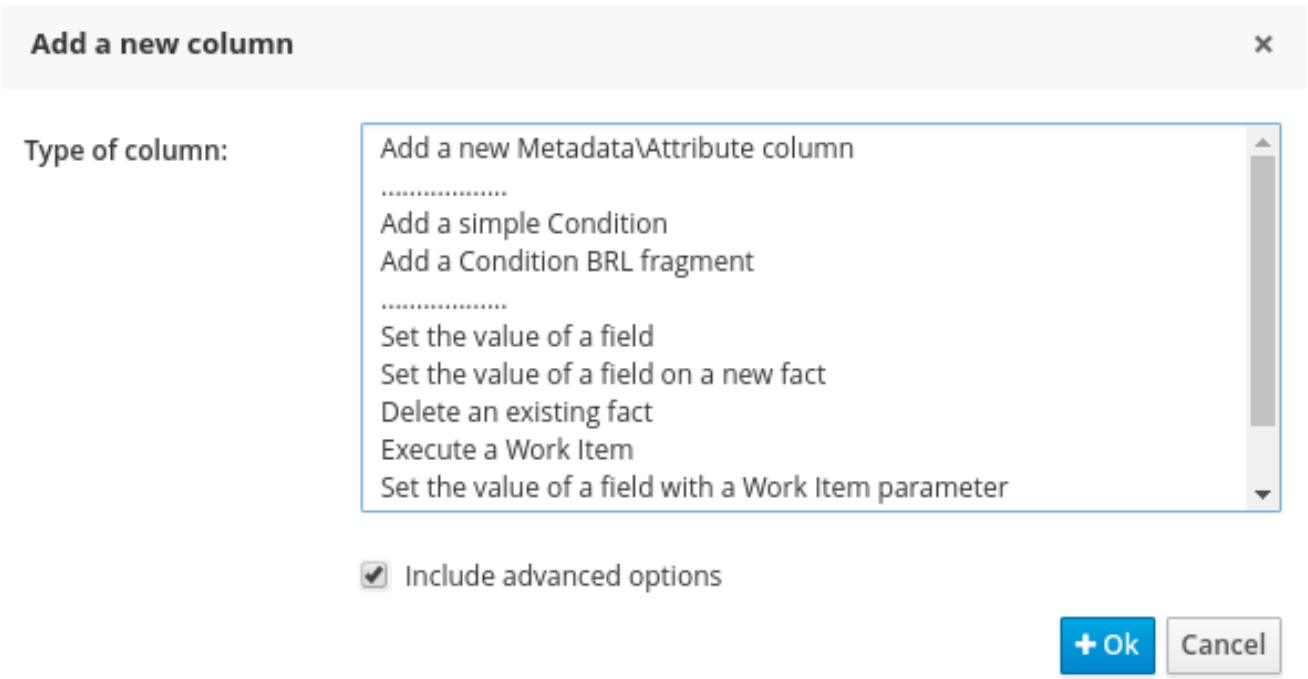
7.4.4. 列の追加

ガイド付きデシジョンテーブルエディター内に列を追加するには、 

アイコンをクリックします。

以下の列タイプの選択ダイアログが表示されます。

図7.7 詳細の列オプション



デフォルトで、列タイプのダイアログに以下のタイプが表示されます。

- Add a new Metadata\Attribute column (新規メタデータ/属性列の追加)
- Add a simple Condition (単純な条件の追加)
- Set the value of a field (列の値の設定)
- Set the value of a field on a new fact (新規ファクトでのフィールドの値の設定)
- Delete an existing fact (既存ファクトの削除)

「Include advanced options (詳細オプションの組み込み)」をクリックすると、以下のオプションが追加されます。

- Add a Condition BRL fragment (条件 BRLフラグメントの追加)
- Execute a Work Item (作業アイテムの実行)
- Set the value of a field with a Work Item parameter (作業アイテムパラメーターでのフィールドの値の設定)
- Set the value of a field on a new Fact with a Work Item parameter (作業アイテムパラメーターでの新規ファクトのフィールド値の設定)
- Add an action BRL fragment (アクション BRL フラグメントの追加)

7.4.5. 列タイプ

7.4.5.1. 属性列

複数の DRL ルール属性を表現する属性列を追加することができます。また、この列はなくても構いません。以下は例になります。

```
rule "Rule1"
salience 100 // This rule has the highest priority
when
  $c : Cheese( name == "Cheddar" )
then
  ...
end
```

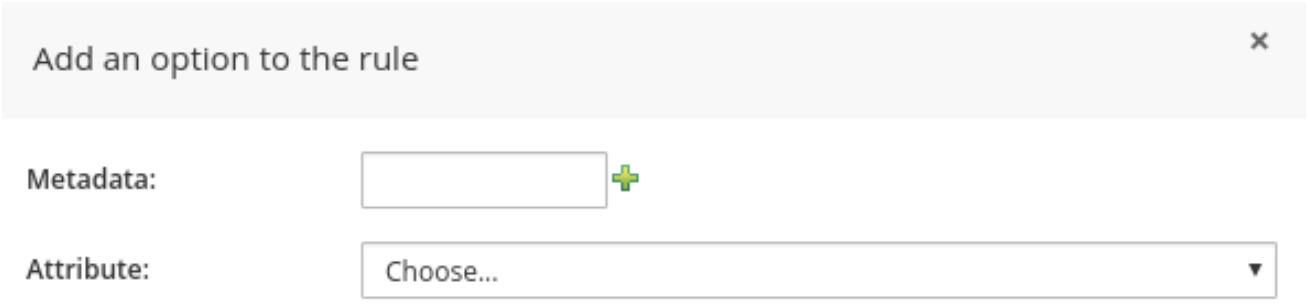
属性の一覧については、『**Red Hat JBoss BPM Suite Development Guide**』の「Rule Set Entries」の章を参照してください。

7.4.5.2. メタデータ列

メタデータ列を定義でき (定義しなくてもよい)、それぞれは DRL ルールに付く通常のメタデータアノテーションを指します。メタデータを追加するには、以下を実行します。

1. **New column** をクリックしてから、**Add a new Metadata\Attribute column**を選択します。
2. **Metadata** フィールドに入力してから、＋ボタン ( ) をクリックしてメタデータを追加します。

図7.8 属性およびメタデータオプション



7.4.5.3. 条件列

条件は、右側で定義したファクトのパターン、またはルールの「When」の部分を示します。条件列を定義するには、モデルクラスへのバインドを定義するか、定義済みのものを選択する必要があります。パターンを無効にするように選択することもできます。

```
when
  $c : Cheese( name == "Cheddar" ) //Binds the Cheese object to the $c variable
then
  ...
end
```

```
when
  not Cheese( name == "Cheddar" ) //Negates matching pattern
then
  ...
end
```

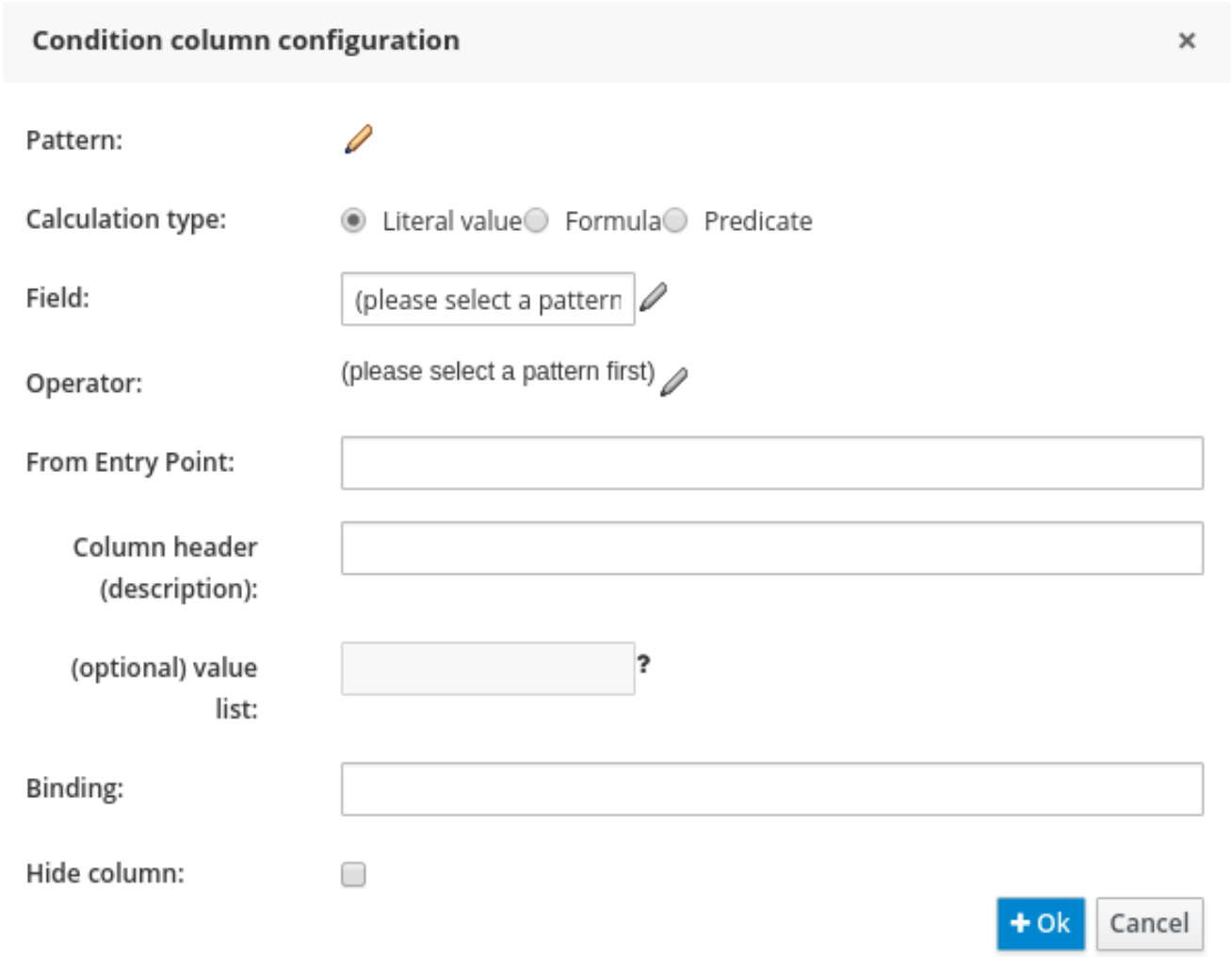
この作業を完了すると、フィールドの制約を定義することができます。同じファクトパターンのバインドを使って 2 つ以上の列を定義した場合、フィールド制約は、同じパターンに対する複合フィールド制約となります。 1 つのモデルクラスに対して複数のバインドを定義した場合、各バインドはルールの左

側で別のモデルクラスになります。

新規列の編集または作成時に、制限タイプのオプションが表示されます。

- Literal: 演算子を使ってセルの値がフィールドと比較されます。
- Formula: セルの式が評価され、フィールドと比較されます。
- Predicate: フィールドは不要であり、式が true または false に評価されます。

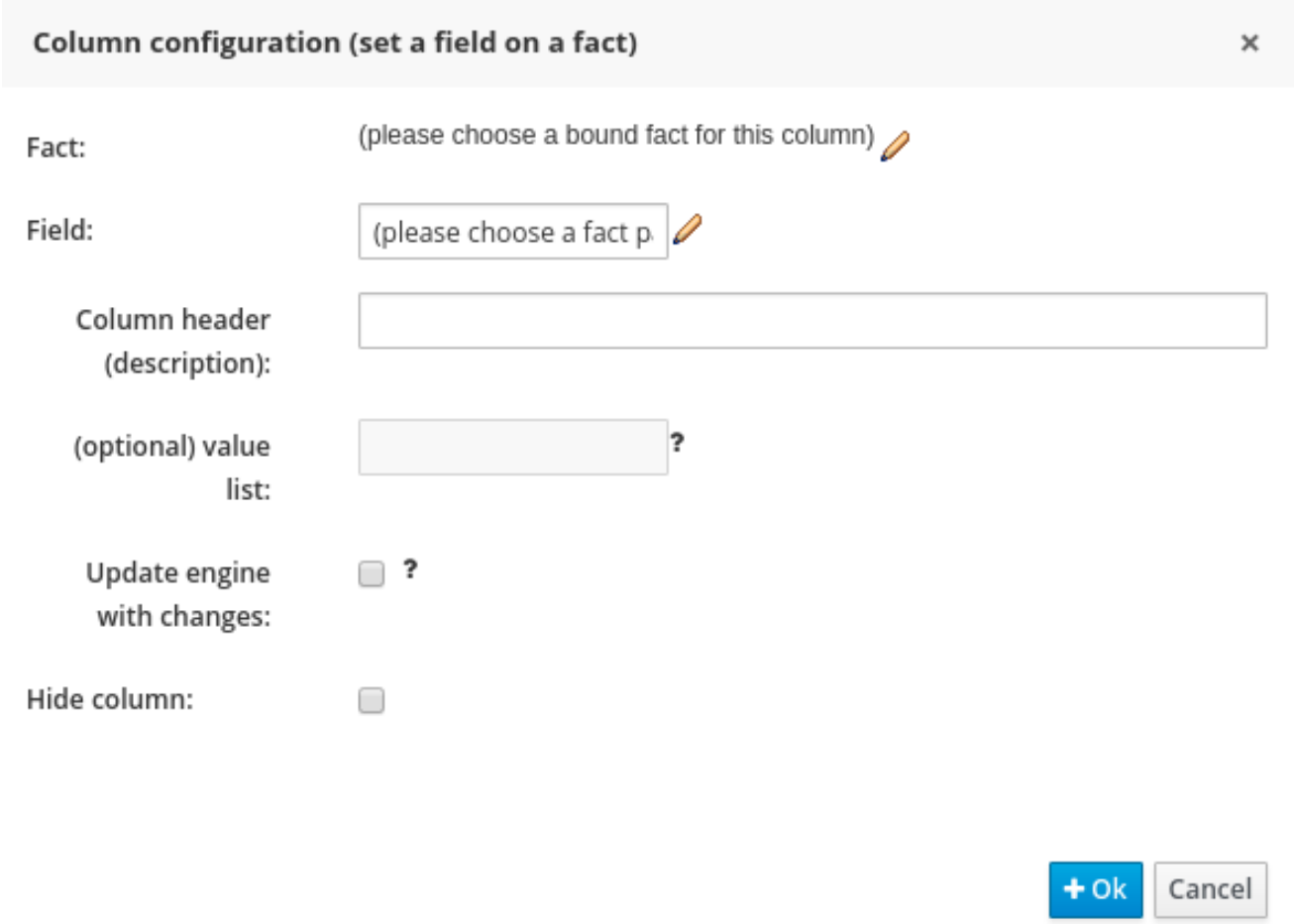
図7.9 単純な条件列



7.4.5.4. フィールド値の列

この列では、事前にバインドされたファクトのフィールドの値を設定するためのアクションを作成します。ルールエンジンに対して、他のルールを再度アクティブにする可能性のある変更された値について通知するオプションがあります。

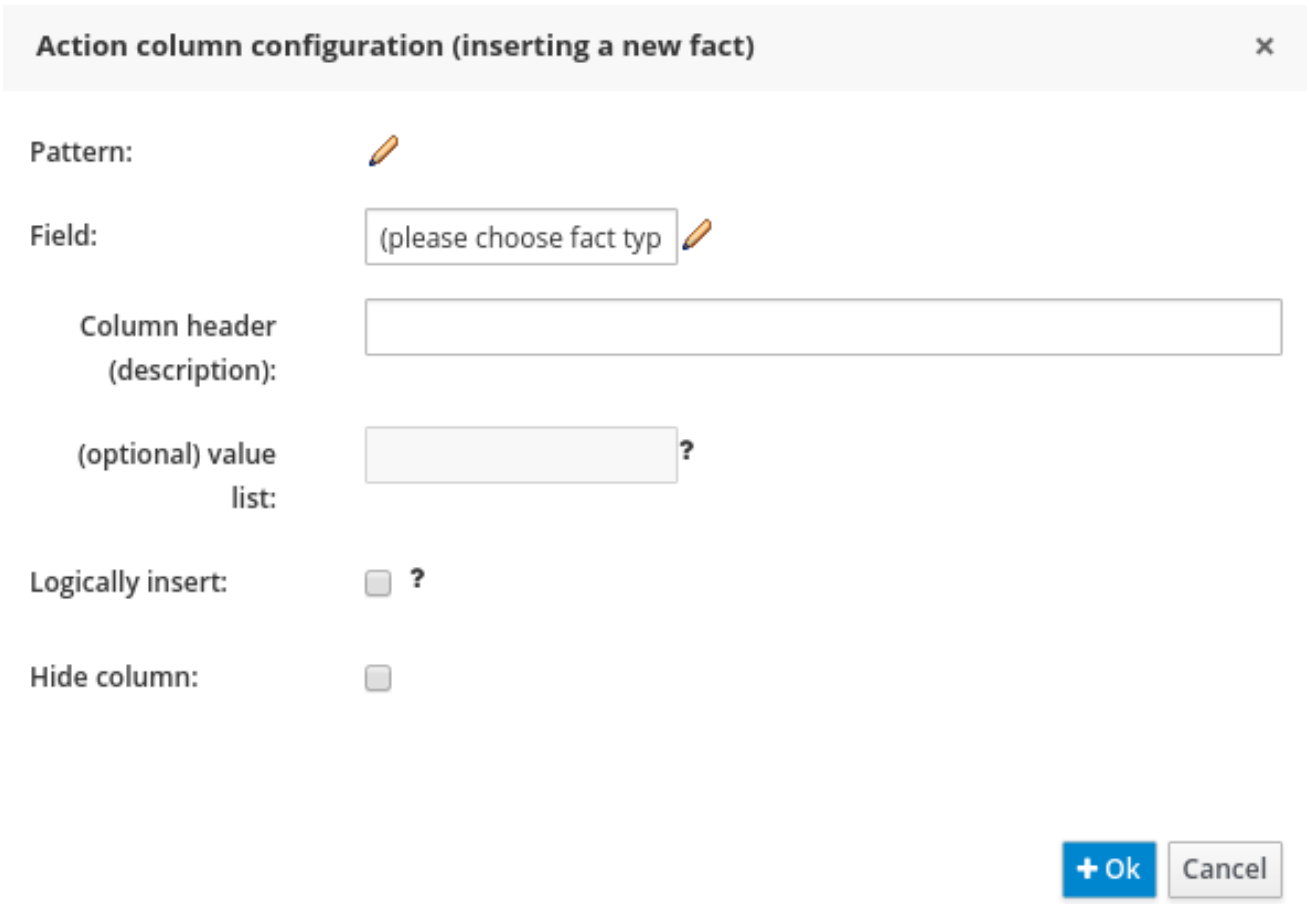
図7.10 Set the value of a field (列の値の設定)



7.4.5.5. 新規のファクトフィールド値の列

この列では、新規ファクト (オブジェクト) をルールエンジンの作業メモリーに挿入するアクションを有効にします。新規ファクトは **論理的に挿入** することが可能です。ファクトを論理的に挿入すると、挿入されたファクトは、そのファクトを挿入したルールの条件が true ではなくなるとすぐに取り消されます。true 状態の維持および論理的な挿入についての詳細は、『**Red Hat JBoss Development Guide**』を参照してください。

図7.11 新規ファクトのフィールド値の設定



7.4.5.6. 既存ファクトの削除

バインドされたファクトを削除するアクションを実行します。

図7.12 Delete an existing fact (既存ファクトの削除)

7.4.6. 詳細の列タイプ

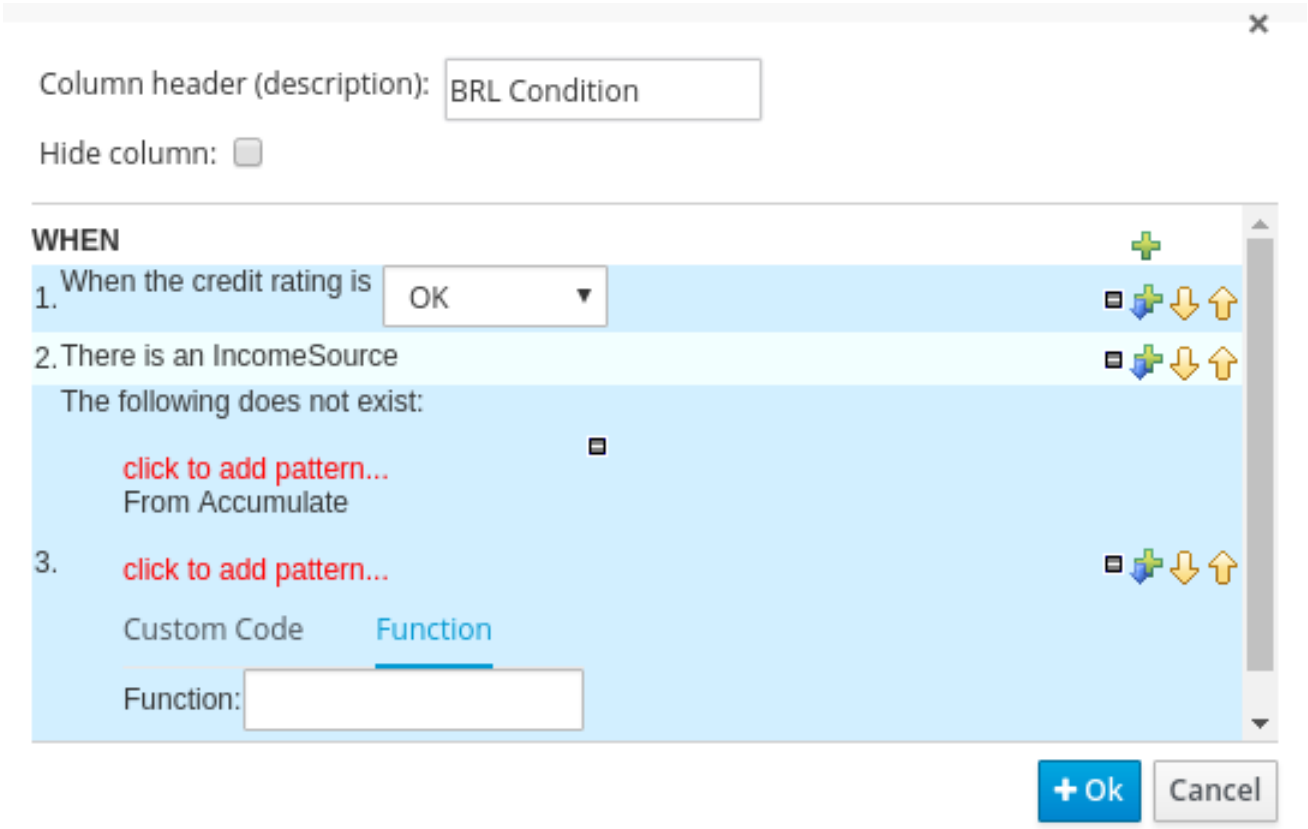
7.4.6.1. 条件付き BRL フラグメント列

BRL フラグメントがルールに左側で使用されることを許可するコンストラクトです。BRL フラグメン

トは、ガイド付きルールエディターを使用して作成されるため、このエディターで選択可能なすべての機能を使用して、「from」、「collect」、および「accumulate」などのデシジョンテーブル列を定義することができます。埋め込みガイド付きルールエディターを使用する場合、「Template Keys」として定義されるフィールド値はデシジョンテーブルの列を構成します。BRL フラグメントでバインドされたファクトおよびファクトのフィールドはより単純な列タイプで参照でき、これらのファクトおよびファクトのフィールドから列タイプの参照も可能です。

以下の例は、ショッピング層の BRL 条件を表示しています。

図7.13 Add a Condition BRL fragment (条件 BRLフラグメントの追加)



7.4.6.2. 作業アイテムの実行列

ここでは、Red Hat JBoss Business Process Management Suite の作業アイテムハンドラーの起動を実行します。入力パラメーターをバインドされたファクト/ファクトのフィールドに値に設定します。これは、すべての作業アイテムの定義について同様に機能します。

図7.14 Execute a Work Item (作業アイテムの実行)

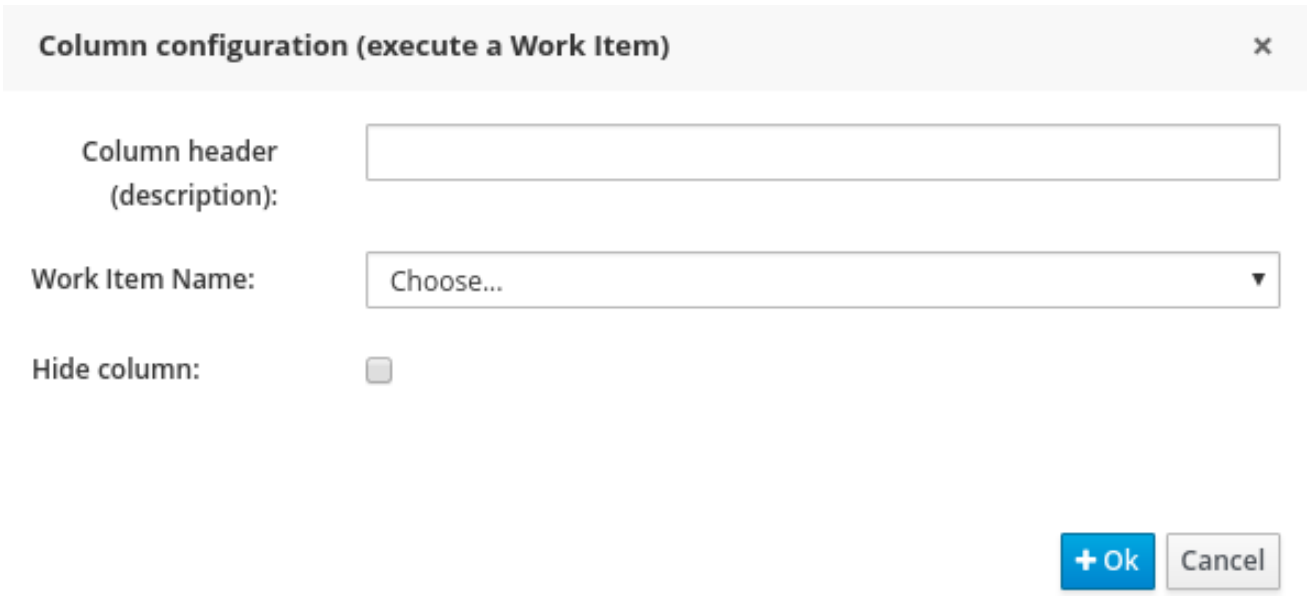


図7.15 WS 作業アイテム

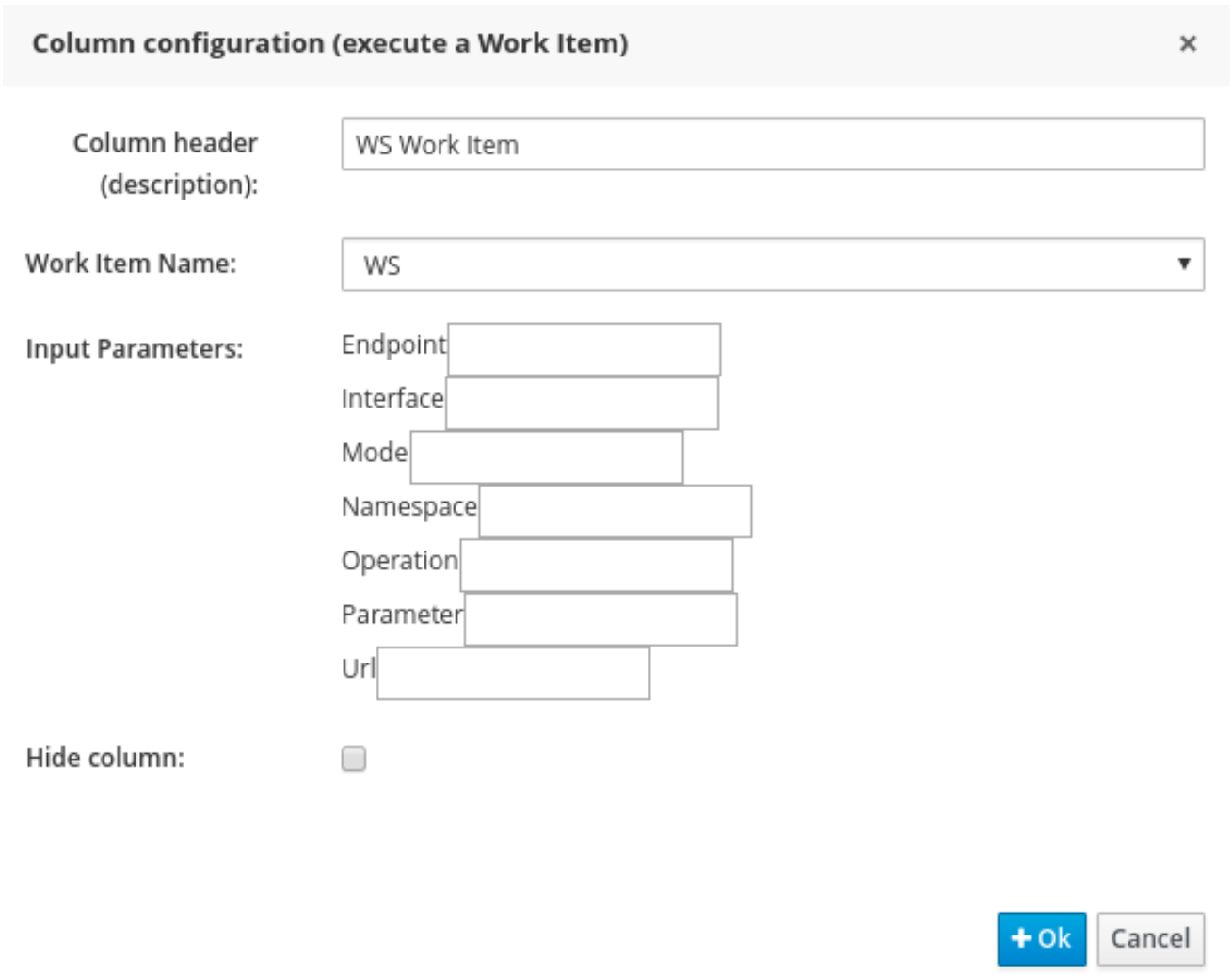


図7.16 REST 作業アイテム

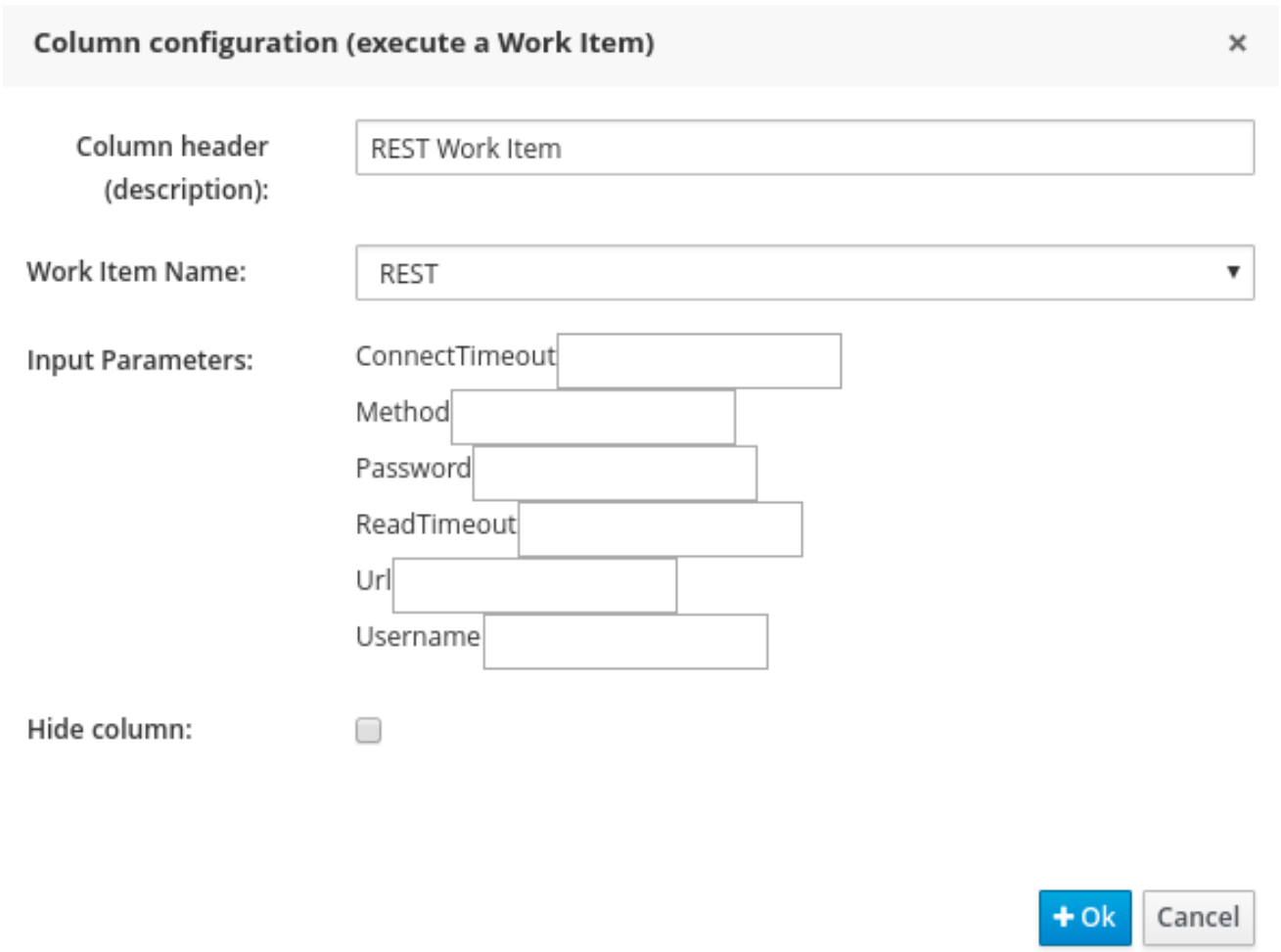


図7.17 ログ作業アイテム

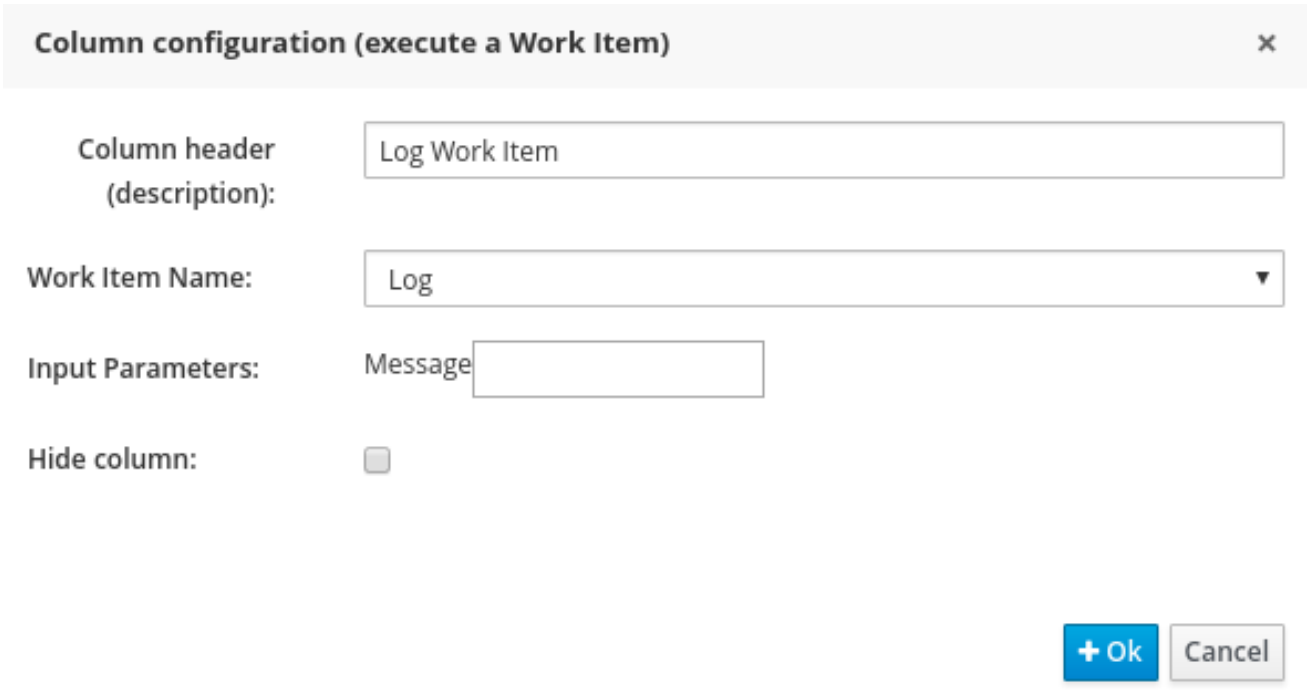
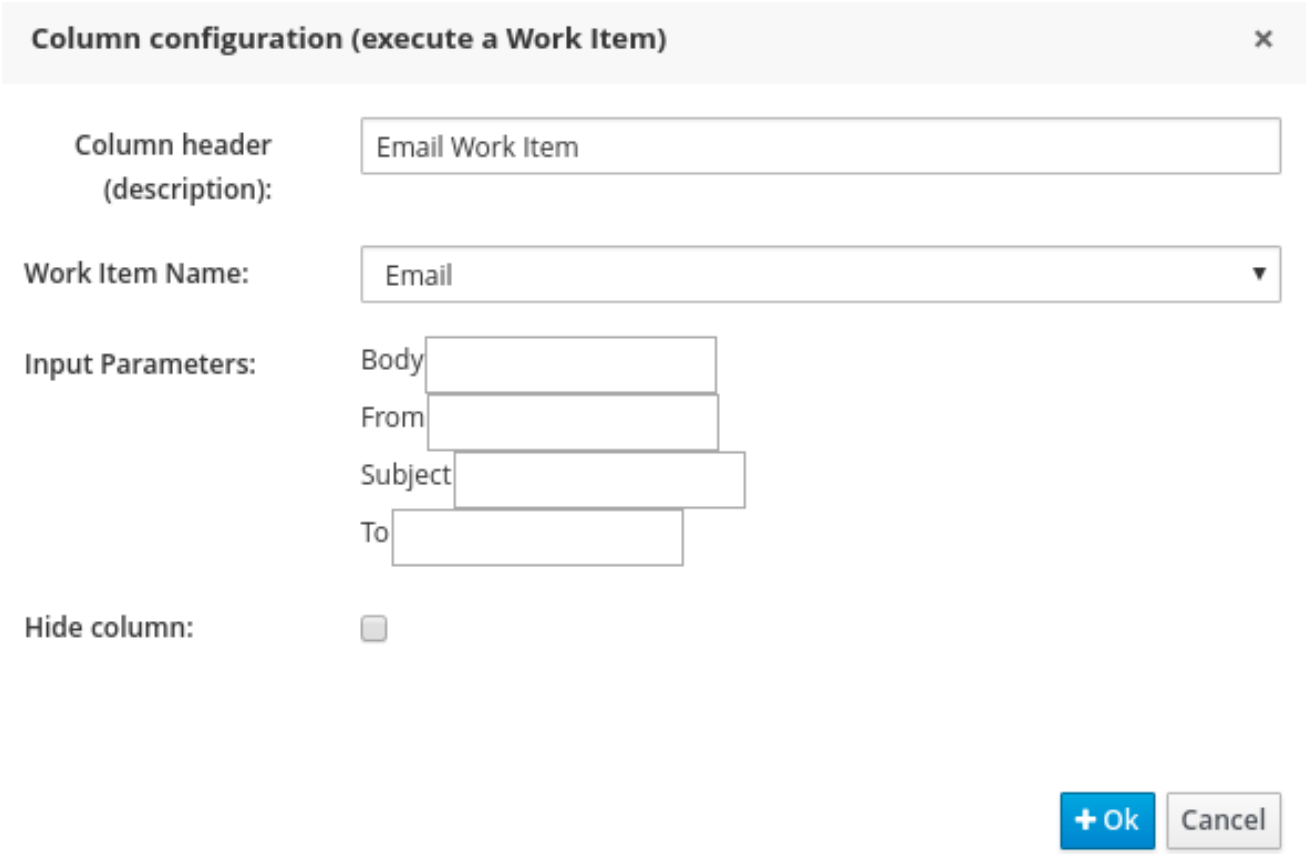


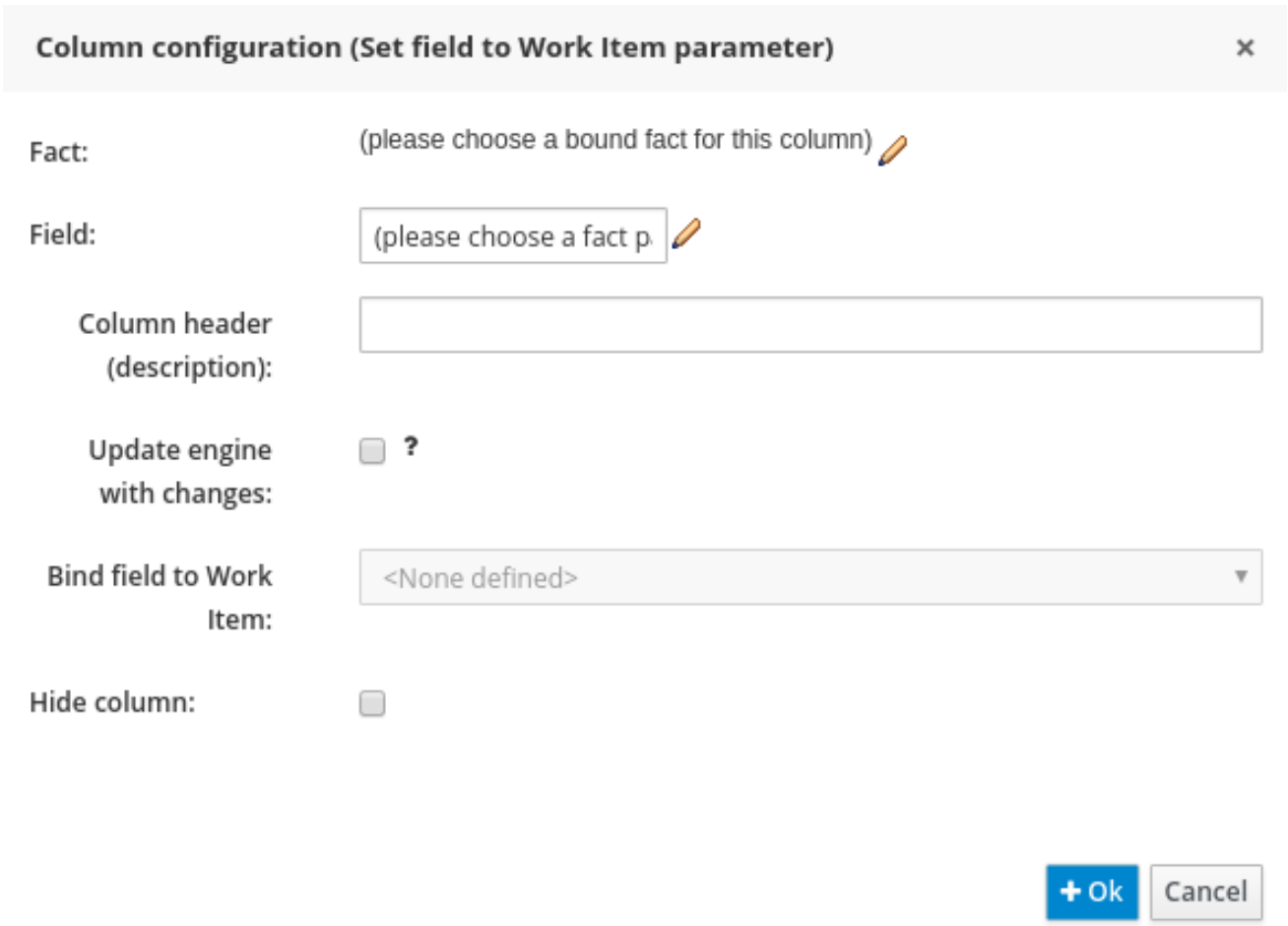
図7.18 メール作業アイテム



7.4.6.3. フィールド値と作業アイテムパラメーター列

ここでは、ファクトフィールドの値を Red Hat JBoss BPM Suite の作業アイテムハンドラーの result パラメーターの値に設定するアクションを実行します。作業アイテムでは、バインドされたファクトのフィールドと同じデータタイプの result パラメーターを定義する必要があります。これにより、フィールドを return パラメーターに設定できます。

図7.19 作業アイテムパラメーターでのフィールド値の設定



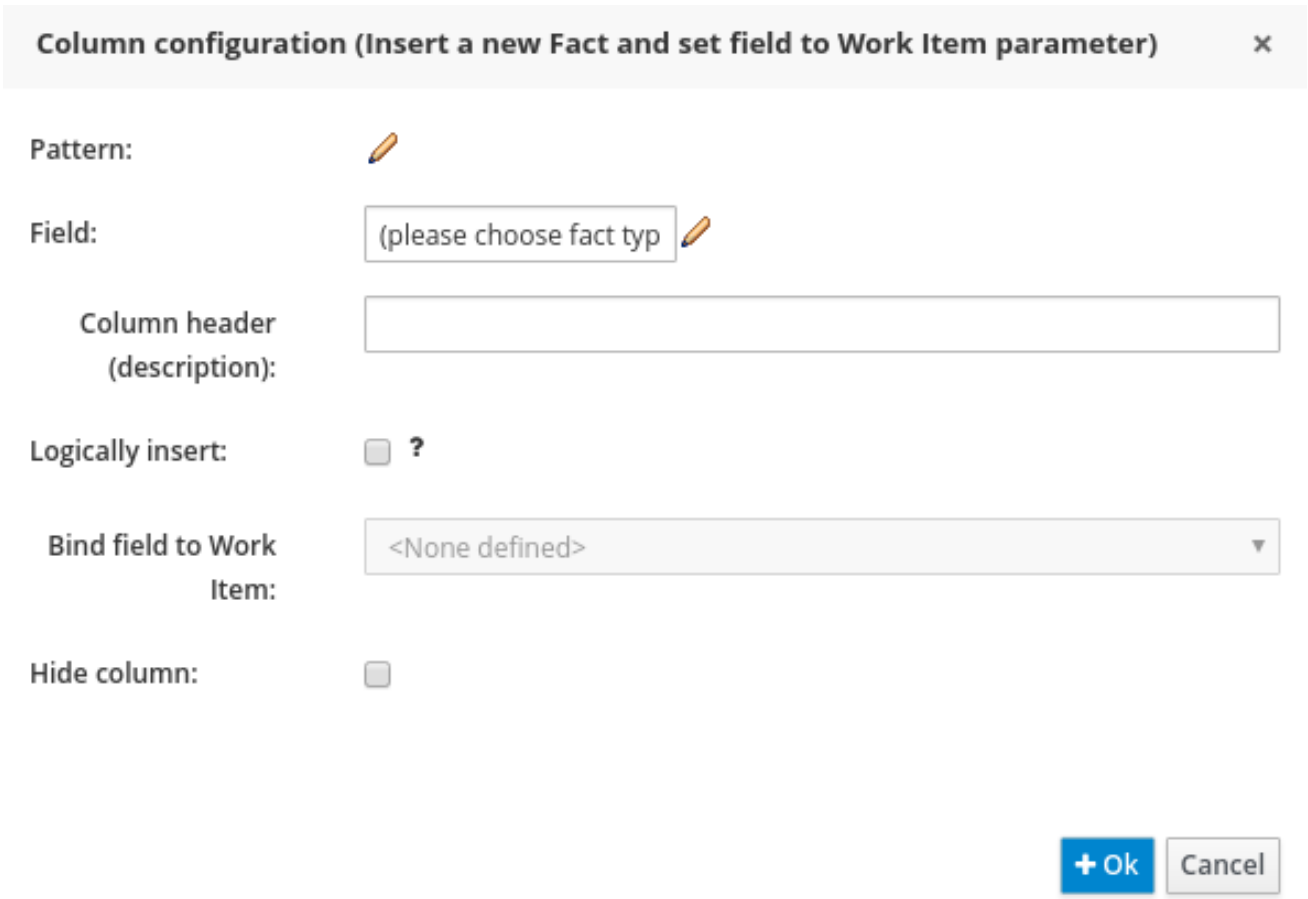
注記

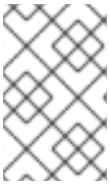
Bind field to Work Item オプションを設定するには、まず作業アイテムを実行するアクションを設定する必要があります。これにより、作業アイテムの結果に基づいて既存ファクトのフィールドを設定できます。

7.4.6.4. 新規フィールド値と作業アイテムパラメーター列

ここでは、ファクトフィールドの値を Red Hat JBoss BPM Suite 作業アイテムハンドラーの result パラメーターの値に設定するアクションを実行します。作業アイテムでは、バインドされたファクトのフィールドと同じデータタイプの result パラメーターを定義する必要があります。この場合、フィールドを return パラメーターに設定できます。

図7.20 作業アイテムパラメーターの新規ファクトのフィールド値の設定





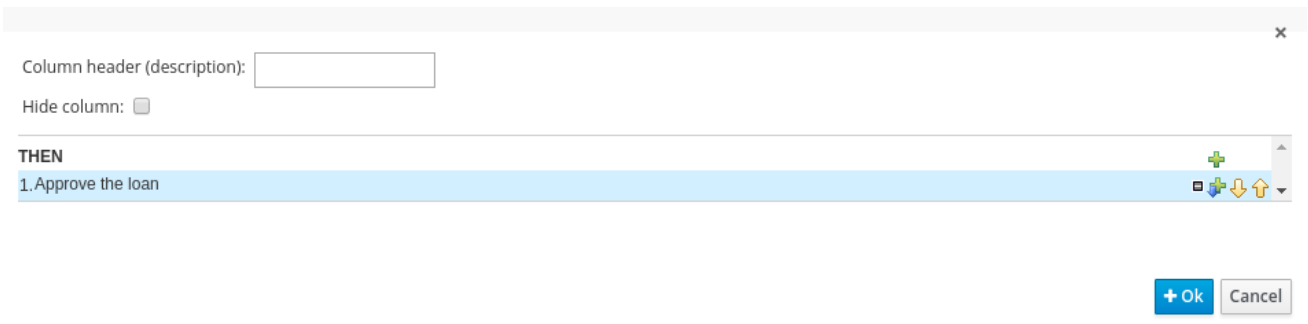
注記

Bind field to Work Item オプションを設定するには、まず作業アイテムを実行するアクションを設定する必要があります。これにより、作業アイテムの結果のフィールド値を持つ新規ファクトを挿入できます。

7.4.6.5. アクション BRL フラグメント列

BRL フラグメントがルールの右側で使用されることを許可するコンストラクトです。BRL フラグメントは、ガイド付きルールエディターを使用して作成されるため、このエディターで選択可能なすべての機能を使用して、デシジョンテーブル列を定義することができます。埋め込みガイド付きルールエディターを使用する場合、「Template Keys」として定義されるフィールドの値はデシジョンテーブルの列を構成します。BRL フラグメントでバインドされたファクトはより単純な列タイプで参照でき、これらのファクトから列タイプの参照も可能です。

図7.21 アクション BRL フラグメントを追加するための単純なレイアウト

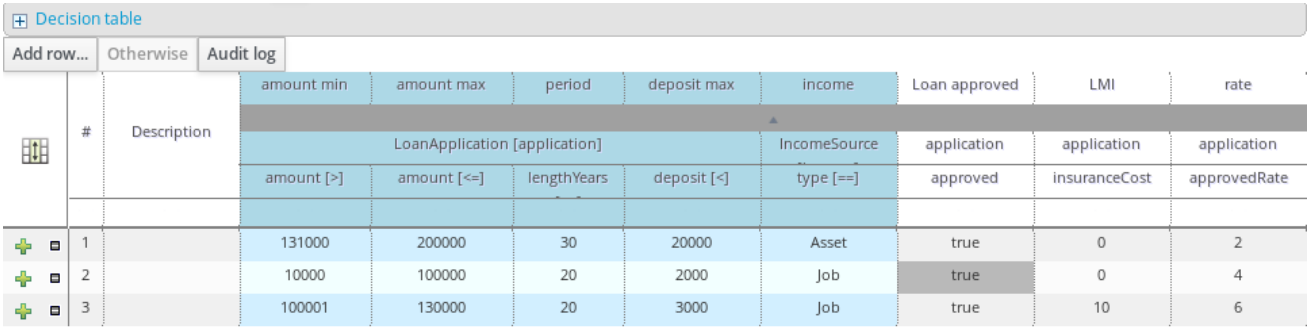


7.4.7. ルール定義

ルールは、定義済みの列を使用してデシジョンテーブルの本体に作成されます。

ルールの行は + のマークをクリックして追加するか、または - のマークをクリックして削除できます。

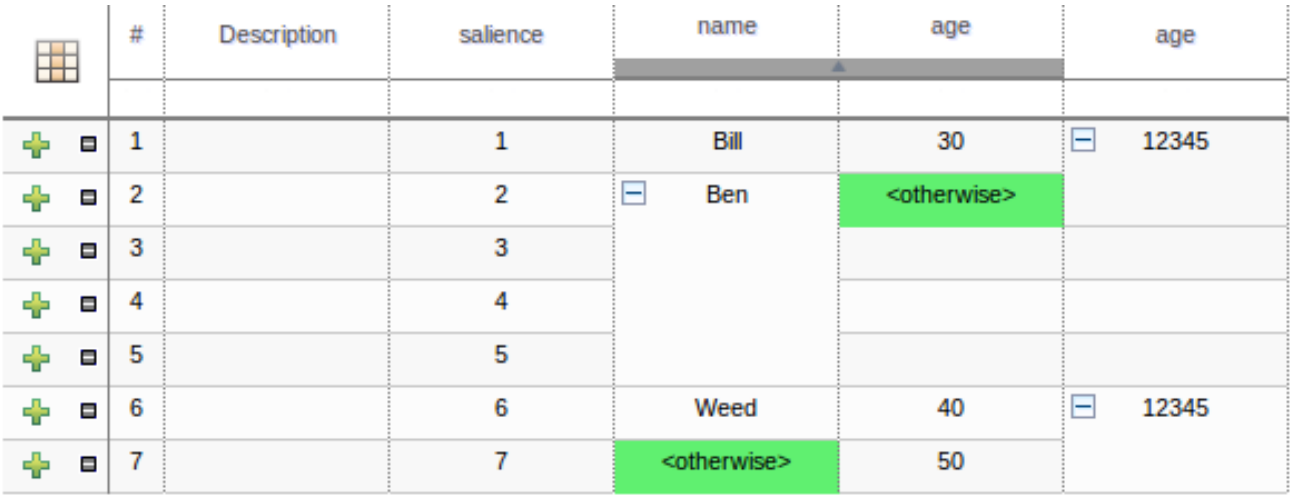
図7.22 ルール定義



7.4.8. セルの統合

デシジョンテーブルの左上にあるアイコンでセルの統合をオンまたはオフにすることができます。セルの統合時に、同じ列に同じ値を持つセルがある場合は 1 つのセルに統合されます。これにより、元の値が同じであった複数のセルの値を簡単に変更できるようになります。セルを統合するとセルの左上にアイコンが現れ、複数のセルにわたる列をグループ化することができます。

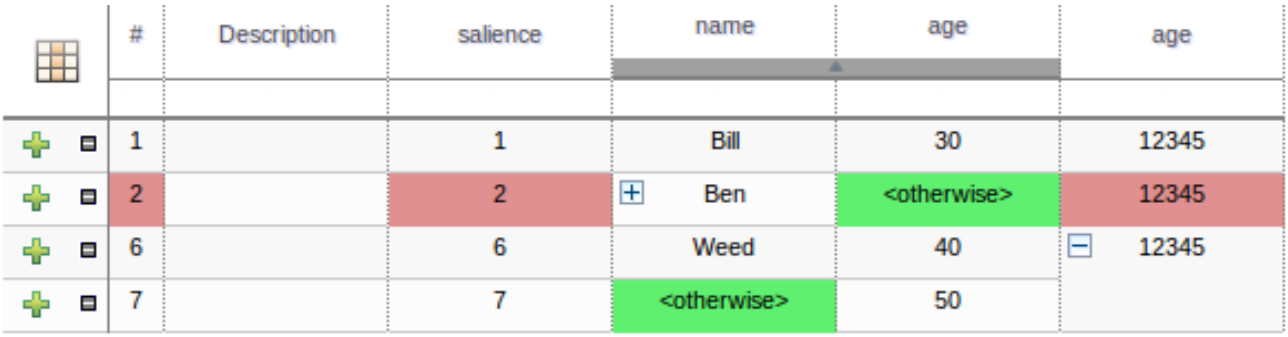
図7.23 セルの統合



7.4.9. セルのグループ化

統合したセルをさらに 1 つの行に折りたたむことが可能です。統合したセルの左上にある [+\-] アイコンをクリックすることで、該当の列を 1 つのエントリーに折りたたみます。他の列にあるセルで、折りたたまれた列にまたがり、同じ値を持つ場合は表示の変更はありません。しかし、他の列にあるセルで、折りたたまれた行にまたがり、別の値を持つ場合は、ハイライトされ、最初の値が表示されます。

図7.24 セルのグループ化



グループ化されたセルの値が変更された場合、折りたたまれたセルの値もすべて更新されます。

7.4.10. Otherwise の操作

等価演算子 **==** あるいは、非等価演算子 **!=** を使うリテラル値で定義した条件列は、デシジョンテーブルの特別なセル値である **otherwise** を活用できます。この特別な値により、表で定義されている他のルールには明示的に定義されていない値すべてにマッチするようにルールを定義することができます。以下の例が最もわかりやすいでしょう。

```
when
  Cheese( name not in ("Cheddar", "Edam", "Brie") )
  ...
then
  ...
end
```

```
when
  Cheese( name in ("Cheddar", "Edam", "Brie") )
  ...
then
  ...
end
```

otherwise キーワードを使用するには、以下を実行します。

1. **==** or **!=** 演算子を使用する条件列のセルを選択します。
2. **Otherwise** をクリックします。

7.5. ルールのテンプレート

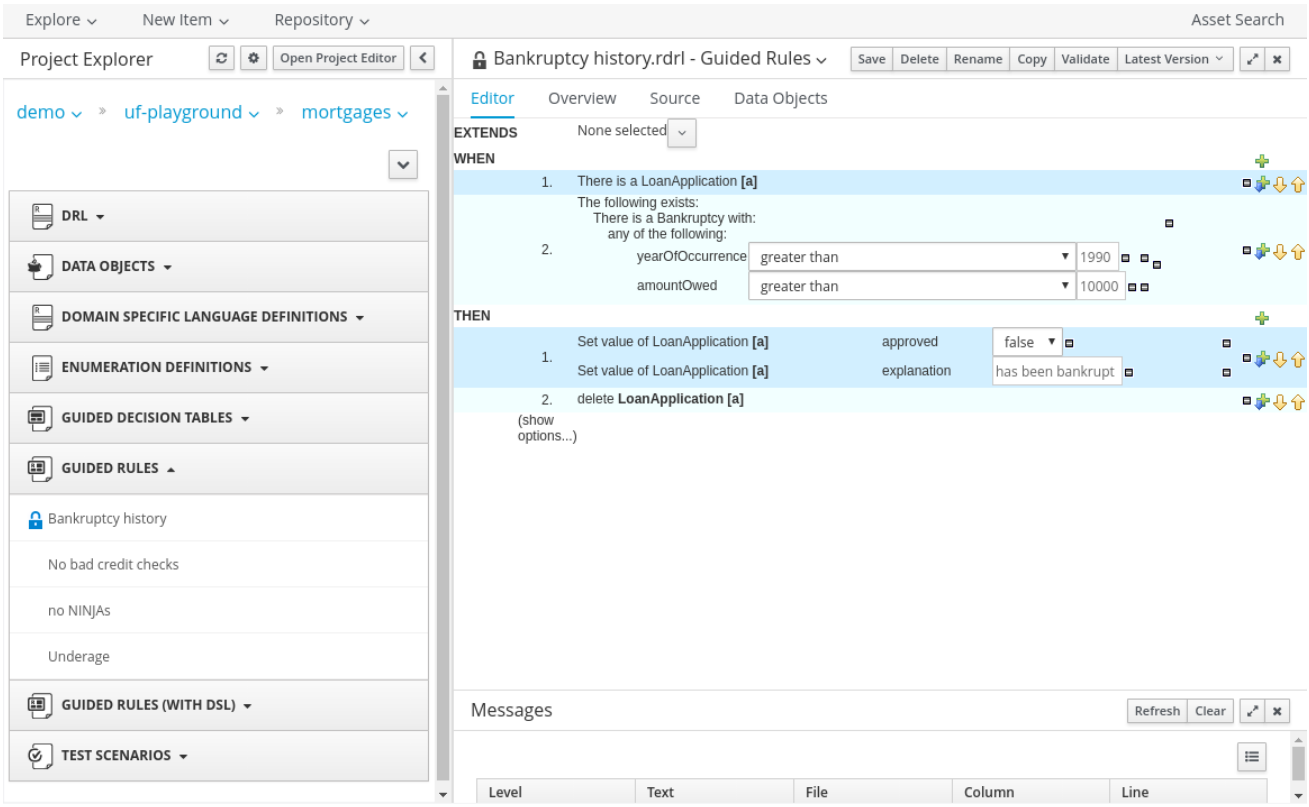
7.5.1. ガイド付きルールテンプレート

ルールテンプレートを使用すると、ルール構造を定義することができます。値やデータのプレースホルダーがあり、テンプレートにデータを設定して数多くのルールが生成されます。ユーザーから見ると、ガイド付きルールテンプレートは、パラメーター値を提供するデータ表を含む、パラメーター化されたガイド付きルールです。テンプレートにより、より柔軟性のあるデシジョンテーブルを作成でき、既存データベースのルールに柔軟性を増し加えることができます。ルールテンプレートの依存関係の管理についての詳細は、『**Red Hat JBoss BPM Suite Development Guide**』の「**Dependency Management for Guided Decision Tables, Scorecards, and Rule Templates**」セクションを参照してください。

手順: ガイド付きルールテンプレートの新規作成

1. **Project Explorer** ビューで、以下のいずれかを実行します。
   a. Project ビューを使用している場合は、組織単位、リポジトリー、およびテンプレートを作成するプロジェクトを選択します。
   b. Repository ビューを使用している場合は、**src/main/resources/** およびルールテンプレートのプロジェクトフォルダーを作成する **SUBFOLDER/PACKAGE** に移動します。
2. パースペクティブメニューで、**New Item → Guided Rule Template** に移動します。
3. **Create new Guided Rule Template** ダイアログウィンドウで、ルールのテンプレート名を指定します。
   a. **Guided Rule Template** テキストボックスで、アセットの名前を入力し、**Package** セレクターからパッケージを選択してから **OK** をクリックします。
4. 新規のガイド付きルールテンプレートが作成され、選択されたプロジェクトから表示されます。

図7.25 ガイド付きテンプレートエディター



注記

単純なルールテンプレートを使用し、ルールとスプレッドシートを Business Central から直接操作することはサポートされていません。Red Hat は Business Central を使用してガイド付きルールテンプレートを作成し、使用することを推奨しています。

7.5.2. ガイド付きルールテンプレートの WHEN 条件

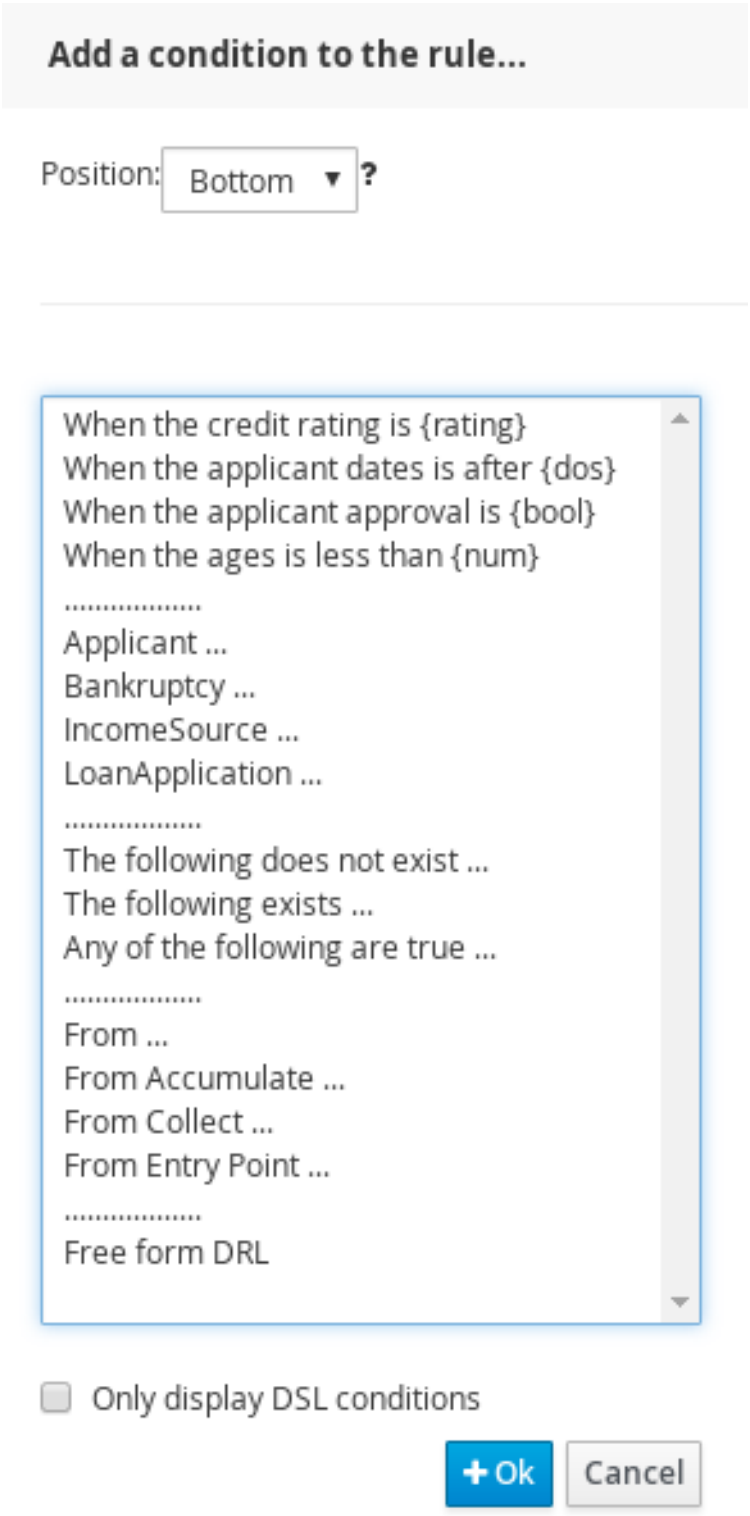
Business Central のガイド付きルールテンプレートでは、データとルールを切り離した状態でルールテンプレートを設定できます。

「データオブジェクトの作成」 で説明されているようにプロジェクトのデータモデルがすでに設定されていることを確認します。

手順: 制限および複数フィールドの制約のある単純な条件の作成

1. ガイド付きルールテンプレートエディターで、**WHEN** セクションの右側にある ＋ アイコン ( ) をクリックします。
利用可能な条件テンプレートを示す **Add a condition to the rule...**ダイアログウィンドウが開きます。

図7.26 条件のルールダイアログウィンドウへの追加

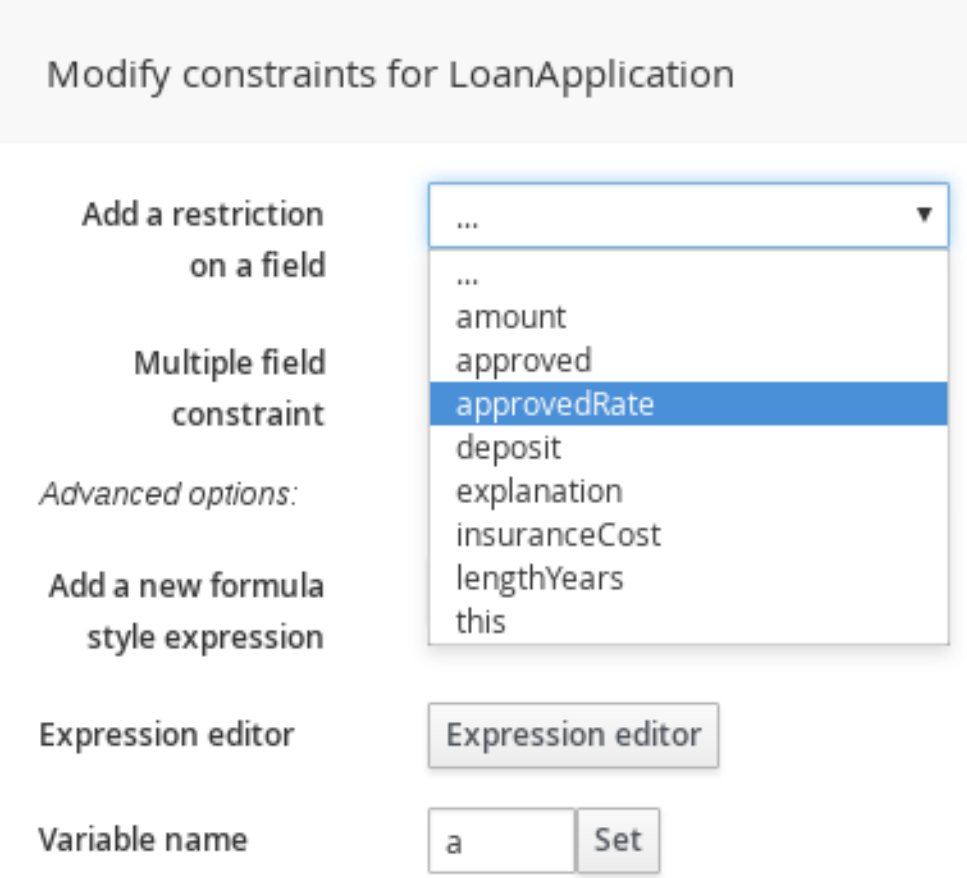


2. 条件 (例: **LoanApplication ...**) を選択して **Ok** をクリックします。
新規の条件がガイド付きルールテンプレートエディターに表示されます。

3. 条件をクリックします。
Modify constraints for LoanApplication ダイアログウィンドウが開きます。ここから、フィールドの制限を追加し、複数フィールドの制約を適用し、新規の数式表現を追加し、式エディターを適用するか、または変数名を設定することができます。

4. 必要に合わせて条件を変更します。たとえば、条件の定義、または制限の追加に役立つ変数名を設定します。
制限を選択すると、ダイアログウィンドウは自動的に閉じます。

図7.27 条件の変更

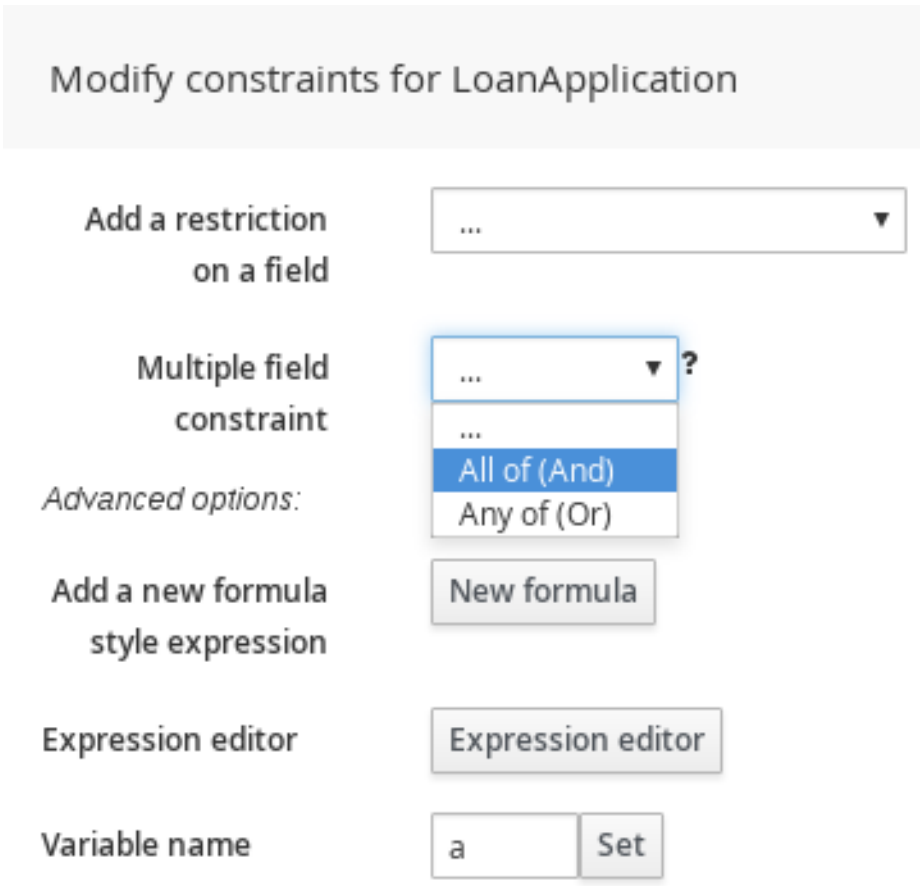


5. 追加した制限の横にあるドロップダウンメニューから制限の演算子 (例: **greater than**) を選択します。

6. **Edit** () をクリックしてフィールドの値を定義します。フィールドの値には、リテラル値、テンプレートキー、数式、または式エディターを使用できます。

7. 複数フィールドの制約を適用するには、条件をクリックし、**Modify constraints for LoanApplication** ダイアログウィンドウで、**Multiple field constraint** ドロップダウンメニューから **All of(And)** または **Any of(Or)** を選択します。

図7.28 複数フィールドの制約の追加

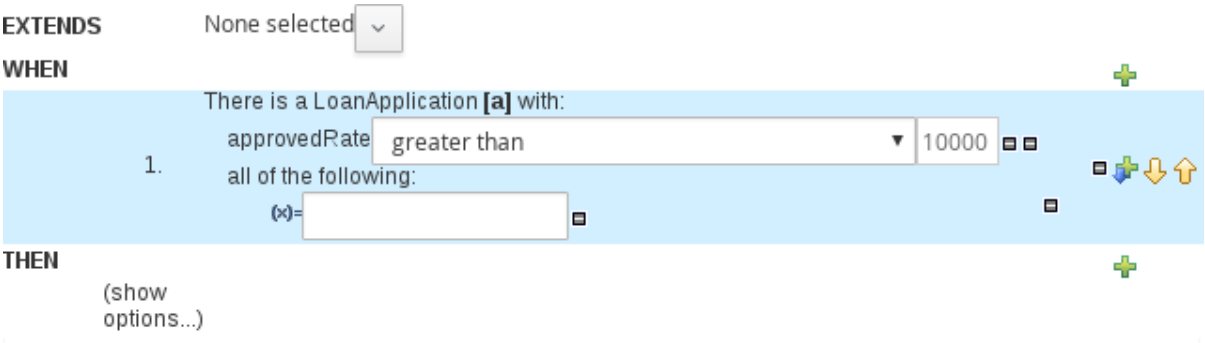


制約はガイド付きルールテンプレートエディターに表示されます。

8. 制約をクリックします。
Add fields to this constraintダイアログウィンドウが開きます。

9. フィールドとそれらの値を指定します。たとえば、**New Formula** をクリックすると、true または false に評価される数式表現を新規に追加できます。

図7.29 数式表現の複数フィールドの制約



7.5.3. ガイド付きルールテンプレートの THEN アクション

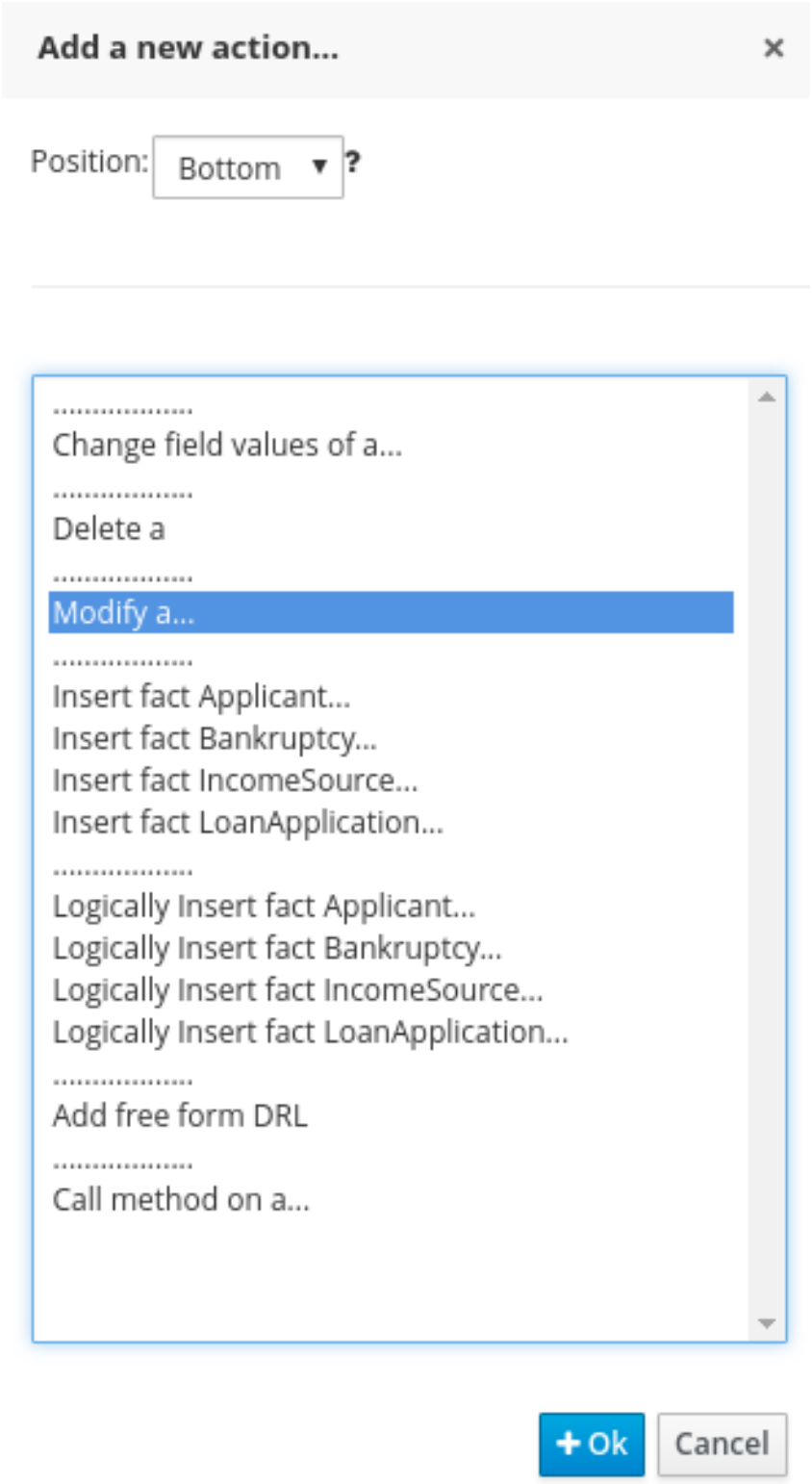
ルールの **THEN** セクションは、**WHEN** セクションに一致する際に実行されるアクションを保持します。

手順: THEN アクションについてのガイド付きテンプレートエディターの使用

1. ガイド付きテンプレートエディターの **THEN** セクションの右側にある ＋ アイコン を選択して **THEN** アクションを入力します。

2. ダイアログウィンドウが、選択可能なアクションテンプレートと共に開きます。以下の例では、一覧から **Modify a...** アクションを選択します。

図7.30 Nurse Roster THEN ダイアログウィンドウ

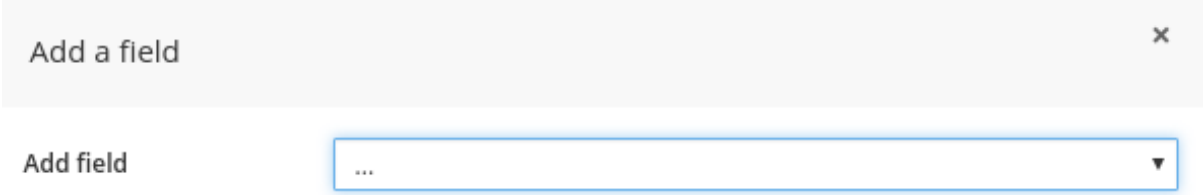


3. **OK** をクリックすると、ガイド付きテンプレートエディターに **THEN** アクションが表示されます。

4. 新規に追加された **THEN** アクションをクリックします。**Add a field** ダイアログが表示されま

4. 新規に追加された **THEN** アクションをクリックします。**Add a field** ダイアログが表示されます。以下は、「**Modify value of LoanApplication [a]**」 アクションの例になります。

図7.31 フィールドの追加ダイアログ

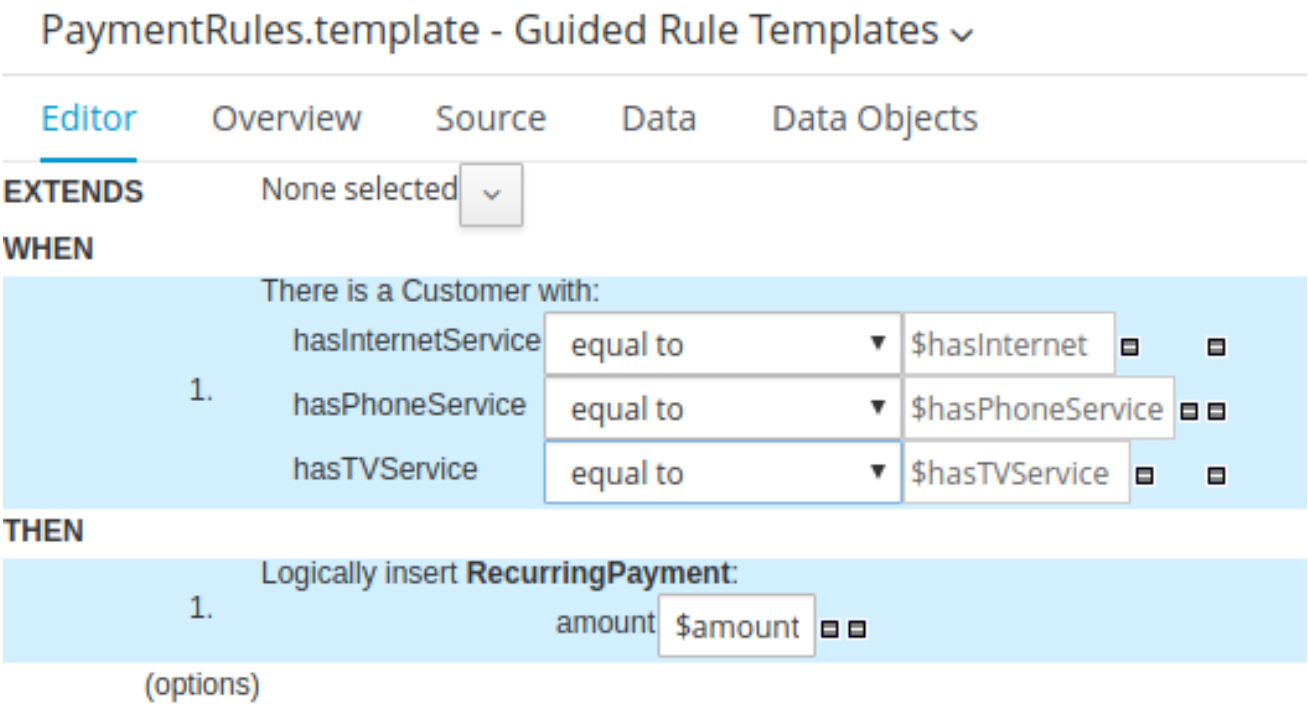


5. このダイアログ内で、**Add field** ドロップダウンメニューからフィールドを選択できます。
6. 選択が行われると、ダイアログウィンドウが自動的に閉じます。
7. このアイテムフィールド内で **Edit** アイコン ( ) を選択し、フィールドの値をリテラル値、テンプレートキー、または数式で定義します。

7.5.4. ガイド付きルールテンプレートのデータ表

データ表は、定義するガイド付きルールテンプレートの変数を提供します。設定に基づいて数多くのルールが生成されます。**Data** タブをクリックすることで、データ表はガイド付きテンプレートエディター内で変更できます。以下の例を考慮してみましょう。

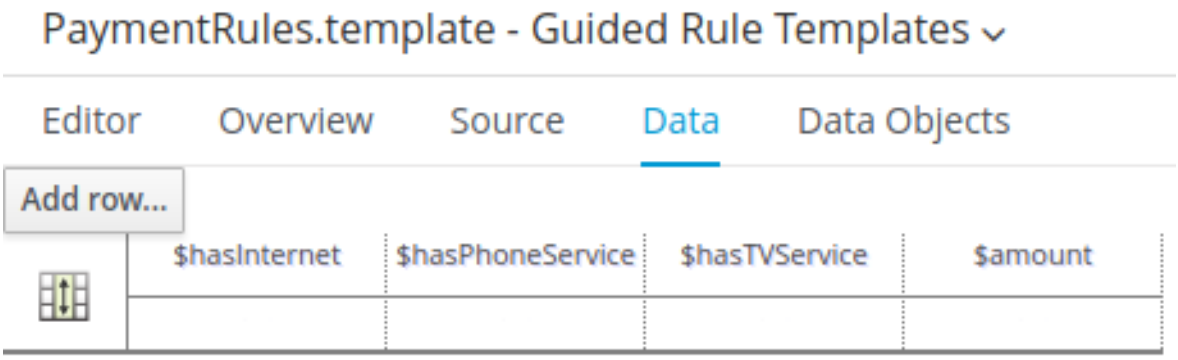
電話サービス、インターネットサービスおよび TV サービスを提供する会社があるとします。毎月の金額は顧客が使用するアクティブなサービスによって異なります。この場合、テンプレートは以下のようになります。



ルールを設定するには、以下を実行します。

手順: データ表についてのガイド付きテンプレートエディターの使用

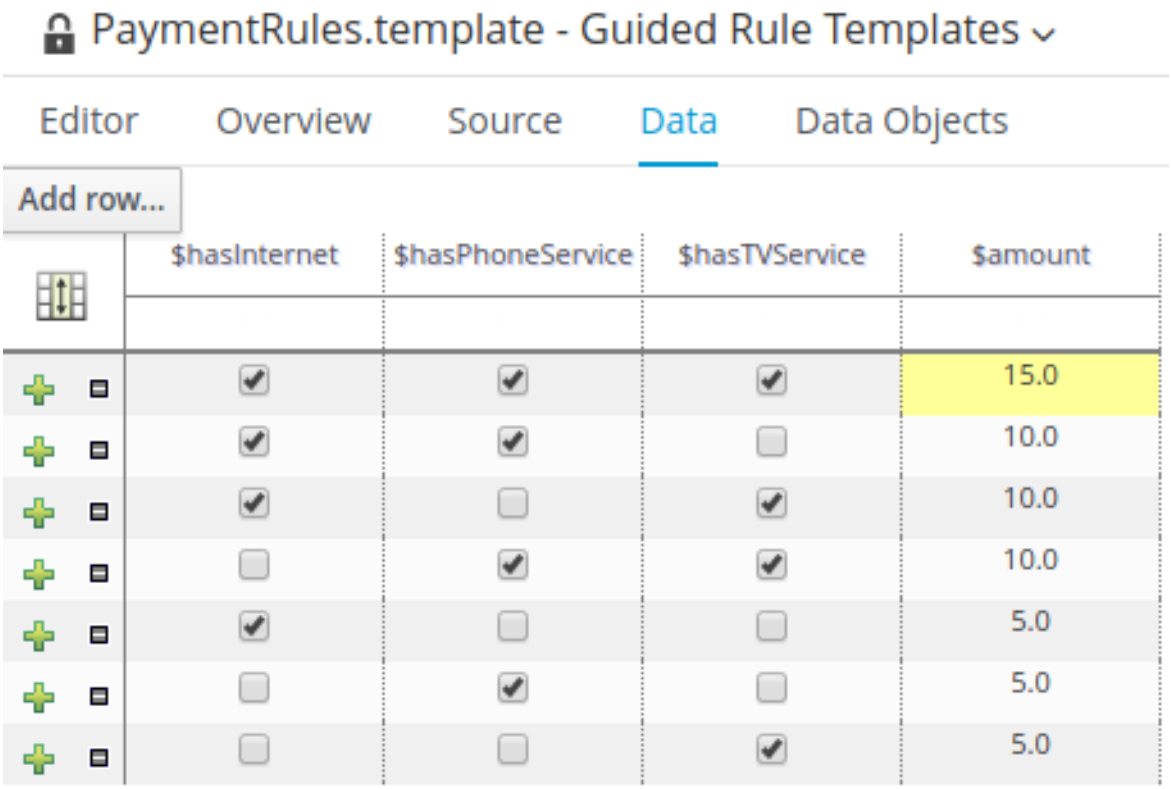
1. **Data** タブをクリックして新規に作成されたデータ表にアクセスします。表は、最初はテンプレートで定義された変数を含むヘッダーのみの空の状態です。



2. **Add row...** をクリックします。新規の列にはそれぞれ 1 つの新規ルールが設定されます。

3. 追加のデータを表示に入力します。以下は例になります。

図7.32 ガイド付きテンプレートエディターのデータ表



4. DRL コードを表示するには、**Source** タブをクリックします。

```
rule "PaymentRules_6"
 dialect "mvel"
 when
  Customer( hasInternetService == false , hasPhoneService == false , hasTVService == true
)
 then
  RecurringPayment fact0 = new RecurringPayment();
  fact0.setAmount( 5.0B );
  insertLogical( fact0 );
end
```

```
rule "PaymentRules_5"
 dialect "mvel"
 when
  Customer( hasInternetService == false , hasPhoneService == true , hasTVService == false
)
 then
  RecurringPayment fact0 = new RecurringPayment();
  fact0.setAmount( 5.0B );
  insertLogical( fact0 );
end
//Other rules omitted for brevity.
```

5. ガイド付きテンプレートエディターでの作業が終了したら、**Save** をクリックしてテンプレートを保存します。

7.6. ドメイン固有言語エディター

ドメイン固有言語機能により、問題ドメインに固有の言語を定義することができます。ユーザーまたはビジネスアナリストのユーザーは、その後に事前定義された言語のみを使用してルールを作成することができます。これにより、ルールやルール言語の深い知識がないユーザーのルール作成プロセスが単純化されます。ドメイン固有言語 (または DSL 文) から構築された文は、DSL Editor で編集可能です。ドメイン固有言語の詳細については、『**Red Hat JBoss BPM Suite Development Guide**』を参照してください。DSL 構文を拡張してヒントを提供し、DSL 変数がどのようにレンダリングされるかを制御します。以下のヒントに対応しています。

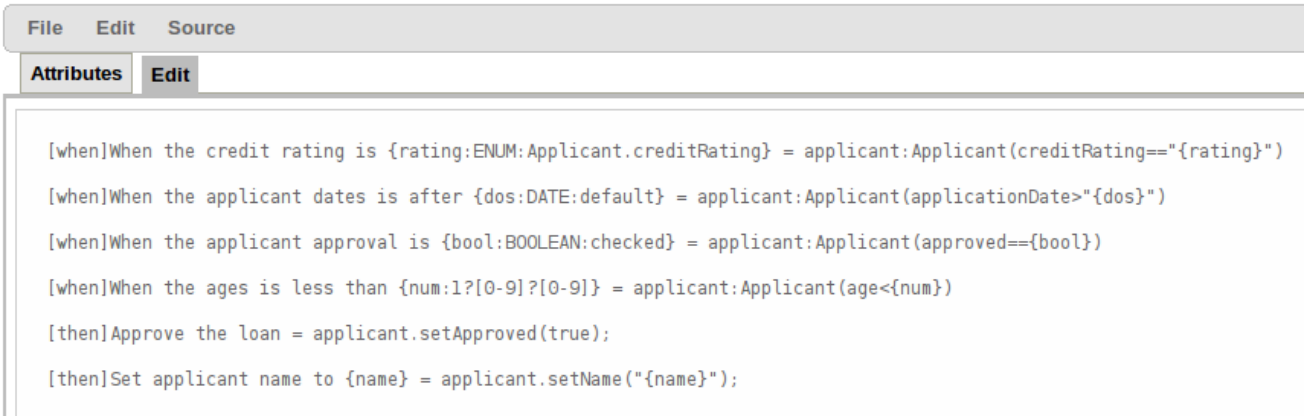
- {<varName>:<regular expression>}
  これは、DSL 文をガイド付きエディターで使用する場合に、DSL 変数の場所でテキストフィールドをレンダリングします。テキストフィールドの内容は、正規表現をベースに検証されます。

- {<varName>:ENUM:<factType.fieldName>}
  これは、DSL 文をガイド付きエディターで使用する場合に、DSL 変数の場所で列挙をレンダリングします。<factType.fieldName> は、列挙とモデルファクトおよびフィールド列挙定義をバインドします。これは、ナレッジベース列挙または Java 列挙 (モデル POJO JAR ファイルで定義) のいずれかです。

- {<varName>:DATE:<dateFormat>}
  これは、DSL 文をガイド付きエディターで使用する場合、DSL 変数の場所に Date selector をレンダリングします。

- {<varName>:BOOLEAN:<[checked | unchecked]>}
  これは、DSL 文をガイド付きエディターで使用する場合、DSL 変数の場所に Dropdown selector をレンダリングします。

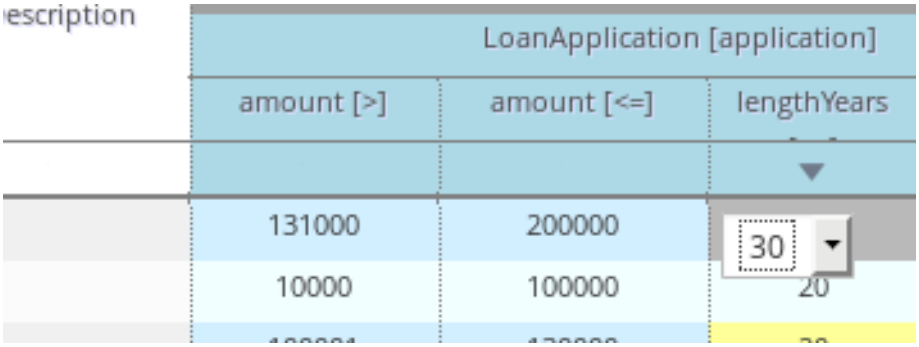
図7.33 DSL Editor



7.7. データ列挙

7.7.1. データ列挙のドロップダウンリスト設定

データ列挙はオプションのアセットタイプですが、ガイド付きデシジョンテーブルエディターにドロップダウンリストを付けるために設定できます。セルの条件が列挙と同じファクトおよびフィールドに基づく場合は、セルをダブルクリックして列挙のドロップダウンリストを表示します。



これらは、他のアセットと同様に保存、編集され、これらが作成されたパッケージに対してのみ適用されます。

列挙設定の内容は、**fact.field** を値の一覧にマッピングします。これらの値は、ドロップダウンメニューを設定するために使用されます。この一覧は、Literal あるいは Utility クラス (クラスパスに追加する必要あり) を用いて文字列を読み込みます。この文字列には、ドロップダウンメニューに表示する値か、コード値からのマッピング (ルールで利用されるもの) と表示値 (例: M=Mini) のいずれかが含まれています。

例7.2 列挙の設定例

```
'Board.type' : [ 'Short', 'Long', 'M=Mini', 'Boogie']
'Person.age' : [ '20', '25', '30', '35' ]
```

これは、以下のように Business Central で設定することもできます。

7.7.2. 列挙の詳細コンセプト

ドロップダウンのリストはフィールドの値により変化します。列挙があると、他のフィールドの値を基に複数のオプションを定義することができます。

たとえば、保険契約のファクトモデルには、**policyType** と **coverage** のフィールドで構成される保険と呼ばれるクラスが含まれます。**policyType** の選択肢として、**Home** または **Car** などが考えられます。また、保険契約の種類により、利用可能な補償タイプなどが決まります。住宅保険契約には **property** または **liability** が、車両保険契約には **collision** または **fullCoverage** など入る可能性があります。

policyType のフィールド値は、補償にはどのオプションを表示するかを決定しますが、以下のように表現されます。

```
'Insurance.policyType' : ['Home', 'Car']
'Insurance.coverage[policyType=Home]' : ['property', 'liability']
'Insurance.coverage[policyType=Car]' : ['collision', 'fullCoverage']
```

7.7.3. 外部ソースからのデーター覧の取得

外部ソースから文字列の一覧を取得して、列挙メニューで使用することができます。これは、(文字列の) **java.util.List** を返すクラスパスにコードを追加することで可能になります。ユーザーインターフェースで値の一覧を指定する代わりに、コードは文字列一覧を返します (通常、ルールの値と異なる表示値を使用したい場合は文字列の内側に "=" 記号を使用できます)。たとえば、以下を使用することができます。

```
'Person.age' : ['20','25', '30', '35']
```

上記の行を以下のように変更します。

```
'Person.age' : (new com.yourco.DataHelper()).getListOfAges()
```

これは、**DataHelper** と呼ばれるクラスを持ち、このクラスには、**getListOfAges()** というメソッドがあり、このメソッドで文字列一覧を返すことが想定されます。データ列挙は、セッションで初めてガイド付きエディターを使用する際に読み込まれます。列挙が読み込まれたかどうかを確認するには、パッケージ設定画面に移動します。パッケージの「保存と検証」を行うことができ、これによりパッケージのチェックを行い、エラーなどのフィードバックが提供されます。

7.8. スコアカード

7.8.1. スコアカード

スコアカードはリスク管理ツールであり、スコア全体を計算するために使用される数式のグラフィック

表示です。ほとんど場合、これは金融機関や銀行によって、商品を市場で販売することに関するリスクを計算するために使用されます。そのため、特定の結果が生じる可能性または確率を予想するために使用できます。Red Hat JBoss BRMS は、個別のルール条件に割り当てられる部分的なスコアを加算してスコア全体を計算する付加的なスコアカードをサポートしています。

さらに、Drools スコアカードでは理由コードの設定が可能であり、これによりスコア全体に影響を与えた特定ルール (バケット) を特定できます。Drools スコアカードは PMML 4.1 標準に基づいています。

通常、スコアカードは以下のように作成できます。

1. 統計分析は、通常は既存の顧客データベースから収集される履歴データで実行されます。
2. 予測特性 (属性または情報の一部) がこの分析に基づいて特定されます。
3. それぞれの特性は、使用される可能性のある値の範囲に分類され、それらにスコアが付されます。

この詳細の説明として、以下の例を見てみましょう。

図7.34 スコアカードサンプル

Characteristics	Expected Score	Range	Score
Family Income	50	1 - 30000	10
		30001 - 60000	25
		60001 - 90000	40
		90001 - 120000	65
		Over 120000	75

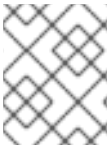
7.8.2. スコアカードの作成

手順: 新規スコアカードの作成 (スプレッドシート)

1. **Project Authoring** パースペクティブを開きます。メインメニューで、**Authoring → Project Authoring** をクリックします。
2. **Project Explorer** ビューで、以下を実行します。
   a. Project Explorer の Project ビューを使用している場合、組織単位、リポジトリー、およびスコアカードを作成するプロジェクトを選択します。
   b. Project Explorer の Repository ビューを使用している場合、スコアカードを作成するプロジェクトのルートに移動します。
3. パースペクティブメニューで、**New Item → Score Card (Spreadsheet)** に移動します。
4. **Create new Score Card (Spreadsheet)** ダイアログウィンドウで、パッケージの詳細を定義します。
   a. **Resource Name** テキストボックスで、スコアカード名を入力します。
   b. **Choose File** をクリックし、場所を参照して、スコアカードが最初に作成されるスプレッドシートを選択します。
5. **OK** をクリックします。

6. 新規スコアカードスプレッドシートが選択されたプロジェクトの下に作成されます。

ガイド付きスコアカードで構成される独自のアプリケーションをビルドする場合、クラスパスに必要な依存関係が追加されていることを確認してください。ガイド付きデシジョンテーブルの依存関係についての詳細は、『**Red Hat JBoss BPM Suite Development Guide**』の「**Dependency Management for Guided Decision Tables, Scorecards, and Rule Templates**」セクションを参照してください。



注記

スコアカード機能はテクノロジープレビューとして提供しているため、現時点で Red Hat JBoss BRMS では**サポートされていません**。

7.9. ガイド付きデシジョンツリー

デシジョンツリーは、意思決定モデルのツリー形式によるグラフィック表示です。フラットなデータオブジェクトモデルを使用して、単純なデシジョンツリーを Business Central で作成することができます。エディターはネスト化されたデータオブジェクトをサポートしません。

エディターの左側にはパレットがあり (利用可能なデータオブジェクト、フィールドおよび対応するアクションが含まれる)、右側には、デシジョンツリーを作成するためにデータオブジェクトをドラッグアンドドロップできる作業エリアがあります。エディターでプロンプト表示されるコネクターや子オブジェクトを使用してツリーを完成することができます。各ノードは、**Delete**、**Edit**、および **Collapse** アイコンを使用して操作できます。

デシジョンツリーの作成時に、以下の点に留意してください。

- データオブジェクトはツリーのルートになければならない。
- ツリーには 1 つのルートしか設定できない。
- データオブジェクトには、子として他のデータオブジェクトや、フィールドの制約、またはアクションを持たせることができる。
- フィールドの制約は、親ノードと同じデータオブジェクトのフィールドに設定される必要がある。
- フィールドの制約には、子として他の制約またはアクションのいずれかを持たせることができる。
- フィールドの制約は、親ノードと同じデータオブジェクトのフィールドに設定される必要がある。
- アクションには、子として他のアクションのみを持たせることができる。

注記

ガイド付きデシジョンツリーはテクノロジープレビューとして提供しているため、現時点で Red Hat JBoss BRMS では**サポートされていません**。

7.10. ガイド付きデシジョンテーブルの確認および検証

Business Central では、ガイド付きデシジョンテーブルは、条件付きロジック (ルール) を詳細に表現する 1 つの方法であり、それはビジネスレベルのルールに適しています。

7.10.1. はじめに

Business Central の確認および検証機能は、ガイド付きデシジョンテーブルを完成し、エラーがない状態にするのに役立ちます。この機能は、ガイド付きデシジョンテーブルのロジックにギャップがあるかどうかをチェックし、異なるセル間の関係を検証します。Business Central の確認および検証機能で報告される問題のほとんどは、作成者のロジックのギャップについてです。確認および検証機能は、この確認および検証通知を無視することを選択した場合でも作業の保存を防ぐことはありません。

ガイド付きデシジョンテーブルを編集する場合、この確認および検証機能は操作を誤った場合に通知を送信します。たとえば、ガイド付きデシジョンテーブルの行にアクションを設定するのを忘れた場合や、重複した行がある場合には、新規パネルの Analysis がこれらの問題についての通知と共に Business Central に開かれます。Analysis パネルはガイド付きデシジョンテーブルで見つかる問題の一覧を表示し、それぞれの項目は影響を受けるガイド付きデシジョンテーブルの行をポイントします。問題の一覧から項目を選択すると、問題のさらに詳細な説明を参照することができます。

検証および確認機能は、以下の問題の解決に役立ちます。

- 冗長性 (Redundancy)
  冗長性は、デシジョンテーブルの 2 つの行で同じファクトセットの同じ結果を実行する際に生じます。たとえば、顧客の誕生日をチェックし、誕生日割引を提供する 2 つ行があると、割引が 2 倍になる可能性があります。

- 包摂 (Subsumption)
  包含は冗長性と似ており、2 つのルールが同じ結果を実行する場合に生じますが、1 つのルールが別のルールのファクトのサブセットに対して実行される場合に生じます。たとえば、以下の 2 つのルールを想定してください。

  - when Person age > 10 then Increase Counter
  - when Person age > 20 then Increase Counter

  この場合、対象者の年齢が 15 歳の場合は 1 つのルールのみが実行され、対象者の年齢が 20 歳の場合には、2 つのルールが実行されます。これにより、冗長性の場合と同様の問題が実行時に生じます。

- 競合 (Conflict)
  2 つの似ている条件に 2 つの異なる結果が設定されている場合には、競合する状況が生じます。デシジョンテーブルの 2 つの行 (ルール) または 2 つのセルの間に競合が見られる場合があります。

  以下の例は、デシジョンテーブルの 2 つの行の間の競合を示しています。

  - when Deposit > 20000 then Approve Loan
  - when Deposit > 20000 then Refuse Loan
    この場合、ローンが承認されるかどうかについて確認することはできません。

    以下の例は、デシジョンテーブルの 2 つのセル間にある競合を示しています。

  - When Age > 25
  - When Age < 25

  競合するセルを持つ列が実行されることはありません。

- 欠陥 (Deficiency)

欠陥は競合と似ており、デシジョンテーブルのルールのロジックが未完成な状態である場合に生じます。たとえば、以下の 2 つの欠陥のあるルールを見てみましょう。

- when Age > 20 then Approve Loan
- when Deposit < 20000 then Refuse Loan

これらの 2 つのルールにより、21 歳以上で預金が 20000 未満の対象者について混乱が生じる可能性があります。ルールのロジック完成を求める警告に出された後に、競合を防ぐために制約を追加できるかもしれません。

- 列の欠落 (Missing Column)
列を削除するとロジックが未完成または不正確になり、ルールが適切に実行されない場合があります。この問題が検出されることにより、欠落した列への対応が可能になり、削除された条件やアクションにロジックが意図的に依存しないよう調整することができます。

7.10.2. レポート

Business Central の確認および検証機能は、ガイド付きデシジョンテーブルの更新中に Analysis パネルで異なるレベルの問題をレポートします。

- Error: これは、ガイド付きデシジョンテーブルが実行時に設計された通りに機能しなくなるような重大な障害があることを意味します。たとえば、競合はエラーとしてレポートされます。
- Warning: これも重大な障害である可能性がありますが、これによってガイド付きデシジョンテーブルが機能しなくなることはありませんが、再確認が必要になります。包含などの場合に、この警告が出されます。
- Information: この種類の問題は、ガイド付きデシジョンテーブルの作成者の設計上の決定によるものや単純な事故である可能性があります。たとえば、条件のない行がある場合に使用されます。

注記

Business Central の確認および検証機能は、誤った変更の保存を防ぐものではありません。この機能は、編集時に問題のみをレポートするので、これらの問題を無視して変更を保存することができます。

7.10.3. ガイド付きデシジョンテーブルの確認および検証の無効化

Business Central のデシジョンテーブルの確認および検証機能はデフォルトで有効にされています。これは、**org.kie.verification.disable-dtable-realtime-verification** システムプロパティー値を **true** に設定することで無効にできます。

たとえば、JBoss EAP をアプリケーションサーバーとして使用している場合、**$EAP_HOME** ディレクトリーに移動して以下のコマンドを実行します。

```
./standalone.sh -Dorg.kie.verification.disable-dtable-realtime-verification=true
```

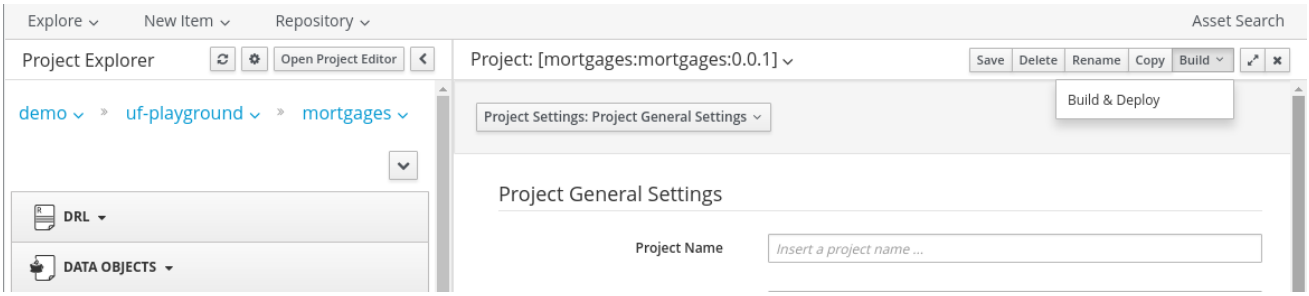
または、このプロパティーを **standalone.xml** ファイルで設定します。

```
<property name="org.kie.verification.disable-dtable-realtime-verification" value="true"/>
```

第8章 アセットのビルドおよびデプロイ

以下の画面に示されるように **Build & Deploy**をクリックしてパッケージおよびアセットのビルドおよびデプロイを実行します。

図8.1 ビルド & デプロイオプション



プロジェクト画面の下にある **Problems** ウィンドウは、ビルドで障害を発生させる可能性のあるエラーメッセージ (コンパイルエラーなど) を表示します。

パッケージ全体またはそのサブセットのビルドを選択できます。

プロジェクトに多数のアセット (ルール、デシジョンテーブルその他) が含まれる場合、ビルドプロセスは通常よりも遅くなる場合があります。ビルドに成功すると、**Authoring → Artifact Repository** でバイナリーパッケージを JAR ファイルとしてダウンロードできます。

8.1. 重複した GAV の検出

以下に記載の操作のいずれかを実行するたびに、すべての Maven リポジトリーで、**GroupId**、**ArtifactId**、および **Version** の重複の有無がチェックされます。重複がある場合は、実行された操作が取り消されます。

重複した GAV の検出は、以下を実行するたびに実行されます。

- 新規の管理リポジトリーの作成。
- プロジェクト定義のプロジェクトエディターへの保存。
- 新規モジュールの管理された複数モジュールリポジトリーへの追加。
- **pom.xml** ファイルの保存。
- プロジェクトのインストール、ビルドまたはデプロイ。

以下の Maven リポジトリーで重複の有無がチェックされます。

- **pom.xml** ファイルの **<repositories>** and **<distributionManagement>** 要素で指定されたリポジトリー。
- Maven の **settings.xml** 設定ファイルで指定されたリポジトリー。

admin ロールを持つユーザーは、影響を受けるリポジトリーの一覧を変更できます。これを実行するには、プロジェクトエディターでプロジェクトを開き、**Project Settings: Project General Settings → Validation** をクリックします。

図8.2 チェックするリポジトリーの一覧

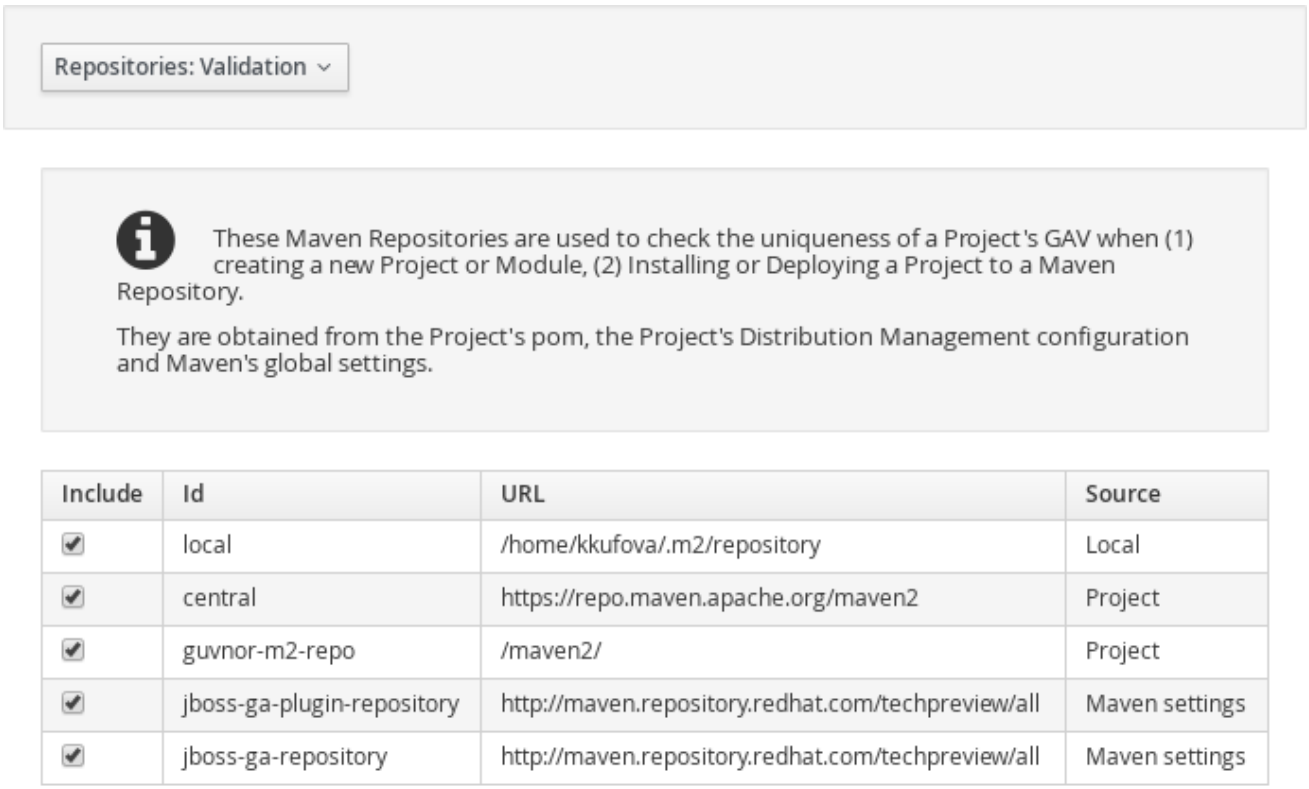
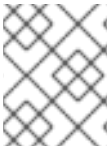


図8.3 検出された重複 GAV



注記

この機能を無効にするには、**org.guvnor.project.gav.check.disabled** システムプロパティーを **true** に設定します。

第9章 アセットの管理

9.1. バージョンおよびストレージ

Business Central で作成されるビジネスルール、プロセス定義、およびその他のアセットおよびリソースは、Realtime Execution Server でアクセスされるアーティファクトリポジトリー (ナレッジストア) に保存されます。

ナレッジストアは、ビジネスの知識を一元化したリポジトリーのことです。複数のリポジトリー (現時点では GIT リポジトリーのみがサポート対象) に接続するので、各種のナレッジおよびアーティファクトをそれぞれ異なる場所に保存できる状態で、単一の環境から異なるコンテンツにアクセスできます。Business Central は web フロントエンドを提供し、ここからユーザーはストアコンテンツを表示したり、更新したりできます。ナレッジストアは Red Hat JBoss BRMS の統一された環境からアクセスでき、**Authoring Perspective** の **Project Editor** および **Project Explorer** からアクセスできます。

Git は分散バージョン管理システムであり、リビジョンをコミットオブジェクトとして実装します。変更をリポジトリーにコミットするたびに、Git リポジトリーに新規コミットオブジェクトが作成されます。同様に、ユーザーは既存リポジトリーをコピーすることもできます。通常、このコピープロセスはクローンと呼ばれており、結果として作成されるリポジトリーはクローンと呼ばれます。すべてのクローンにはファイルのコレクションの完全な履歴が含まれており、そのコンテンツは元のリポジトリーと同じです。

第10章 テスト

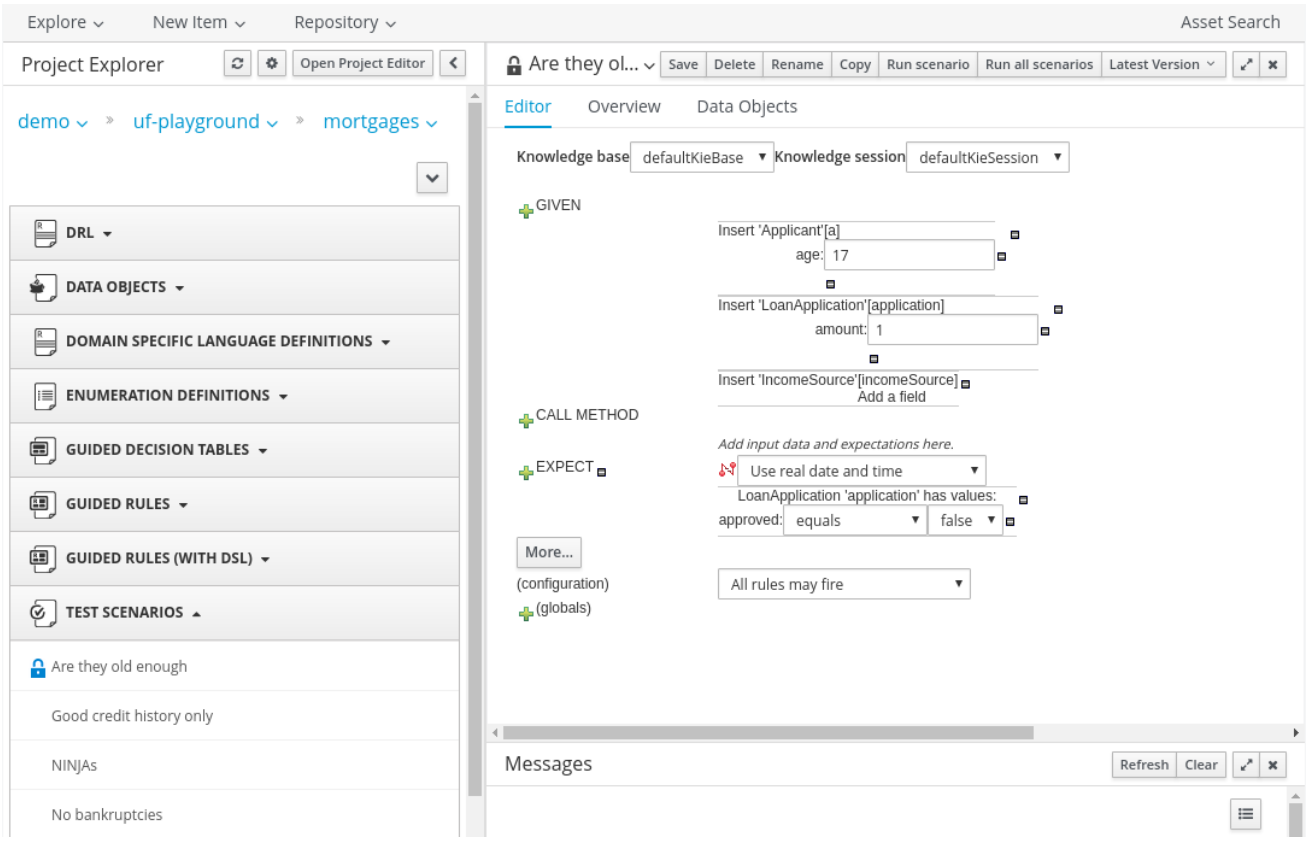
10.1. テストシナリオ

テストシナリオは、開発者によるルール、モデル、イベントの機能検証を可能にする強力な機能です。テストシナリオは、ナレッジベースのデプロイおよび実稼働環境への実装前のテスト機能を提供します。

テストシナリオは、1度に1回またはグループ単位で実行できます。グループの実行では、1つのパッケージのすべてのシナリオが対象になります。テストシナリオは独立したものであり、1つのシナリオが他のシナリオを変更したり、影響を与えたりすることはありません。

すべてのテストシナリオの実行後に、レポートパネルが表示されます。これには、実行されたテストシナリオについての成功または失敗を示すメッセージが含まれます。

図10.1 テストシナリオ画面



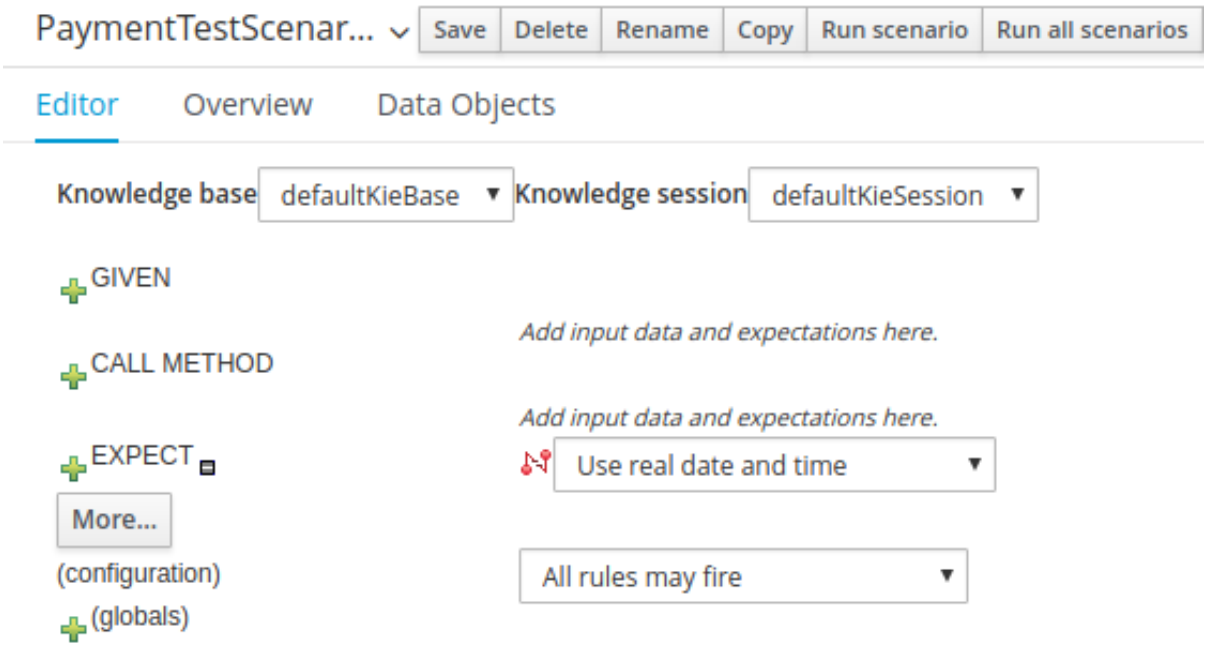
10.2. テストシナリオの作成

テストシナリオの作成には、ファクトまたはプロジェクトモデルのインスタンスと同様の条件のデータを指定する必要があります。テストシナリオは所定のルールセットに対応付けられ、予想される結果が実際の結果に対応付けられる場合は、テストシナリオはパスしたものと見なされます。

新規テストシナリオの作成

1. Business Central で、**Authoring → Project Authoring** をクリックして **Projects** ビューを開きます。
2. テストシナリオのプロジェクトを選択します。
3. **New Item → Test Scenario** をクリックします。

4. 名前を入力し、パッケージを選択して **OK** をクリックします。
5. テストシナリオの編集画面が表示されます。



テストシナリオのモデルのインポート

同じパッケージのデータオブジェクトはデフォルトで利用可能になります。たとえば、パッケージ構造 **org.company.project** については、以下が該当します。

- パッケージ **org.company** のデータオブジェクト Fact1
- パッケージ **org.company.project** の Fact2

org.company でテストシナリオを作成する場合、**org.company.Fact1** は利用可能ですが、**org.company.Fact2** をインポートする必要があります。データオブジェクトをインポートするには、以下を実行します。

1. テストシナリオを開きます。
2. **Data Objects** タブをクリックします。
3. **New Item** をクリックし、インポートを選択して **Ok** をクリックします。インポートはプロジェクトのデータモデルに固有な場合もあれば、**String** または **Double** オブジェクトなどのように汎用的なものである場合もあります。

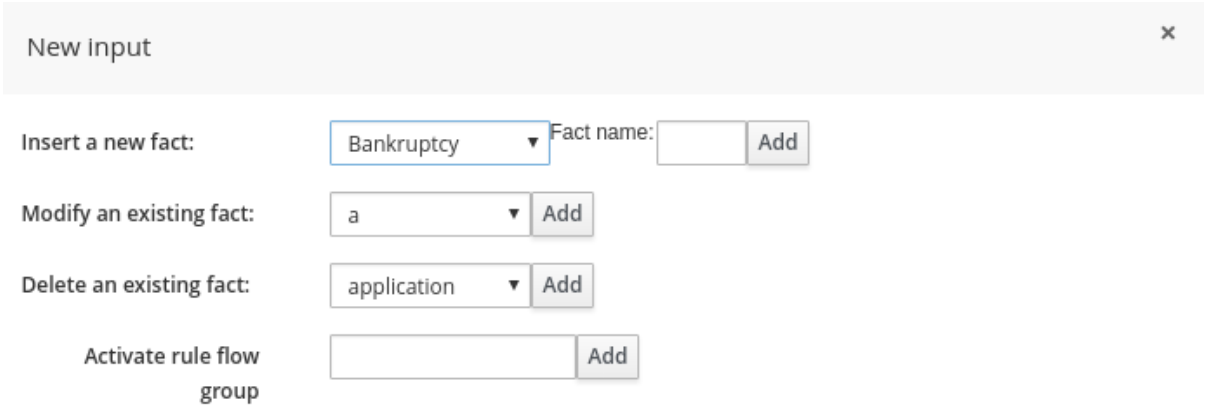
テストシナリオファクトの指定

1. データオブジェクトのインポート後に、**Editor** タブをクリックします。最低でも、**GIVEN** および **EXPECT** などの 2 つのセクションで入力が必要になります。
   - **GIVEN**: テストの入力ファクトです。
   - **EXPECT**: 入力ファクトに基づいて予想される結果です。
     入力パラメーターを **仮定した場合 (GIVEN)**、これらのルールが有効にされるか、または実行されることが **予想されます (EXPECT)**。また、ファクトが表示され、特定フィールドの値を持つことを **予想** できる場合や、ルールが全く実行されないことを **予想** できる場合もあります。

予想される内容が一致する場合、テストシナリオはパスし、ルールが正しいとされます。そうでない場合にテストは失敗します。

仮定 (Given) ファクトの指定

1. 新規ファクトを追加するには、**GIVEN** ラベルの横にある  をクリックします。**Data Objects** タブでインポートしたデータモデルに基づいて、ファクトデータを **New Input** ダイアログウィンドウに指定します。



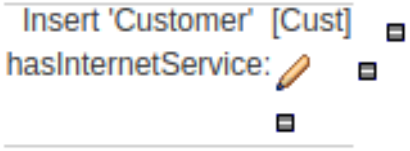
ウィンドウ内で、モデルから特定のデータオブジェクトを選択し、これに **Fact Name** という変数名を付けるか、または代わりにルールフローグループのアクティブ化を選択することもできます。ルールフローグループをアクティブにすると、事前にグループをアクティブして指定されたルールフローグループのルールをテストすることができます。仮定されるファクトを追加し、ルールフローグループをアクティブ化するには、以下を実行します。

   a. 仮定ファクトを追加します。

   b.  を再度クリックして、ルールフローグループのアクティブ化を追加します。

2. オプションで、挿入するオブジェクトに制限を追加します。

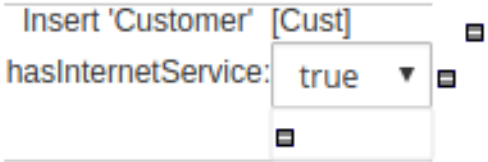
   a. **Add a field** をクリックしてオブジェクトのプロパティーを選択します。



   b. プロパティーの横にある  をクリックします。

      - プロパティータイプが別のファクトオブジェクトの場合は、**Create a new fact** をクリックします。
      - または、**Literal value** をクリックします。
        詳細は、「追加のテストシナリオ機能」 を参照してください。

   c. 値を指定します。以下は例になります。

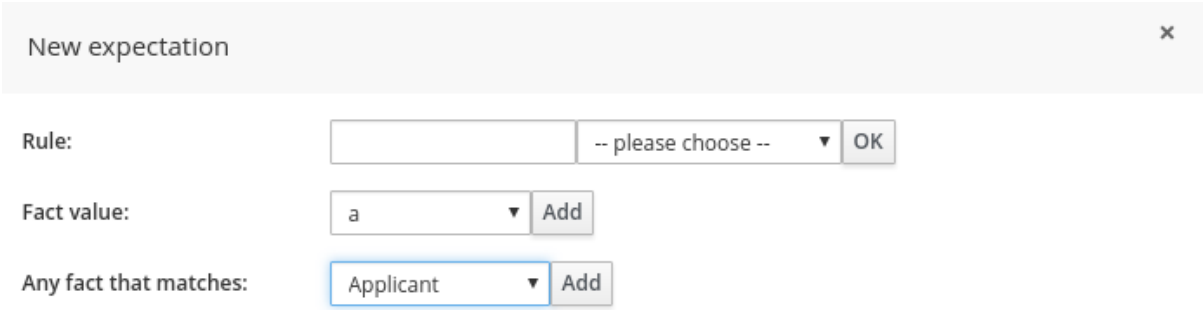


上記の例は、以下に相当します。

```
Customer fact1 = new Customer();
fact1.setHasInternetService(true);
insert(fact1);
```

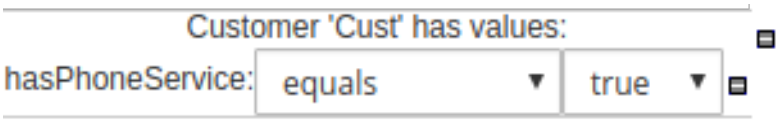
予想されるルールの指定

1. GIVEN 条件に問題がなければ、実行されるルール、作成されるファクト、または変更された既存ファクトのフィールド値を予想できます。EXPECT ラベルの横にある  をクリックして予想される結果の追加を開始します。

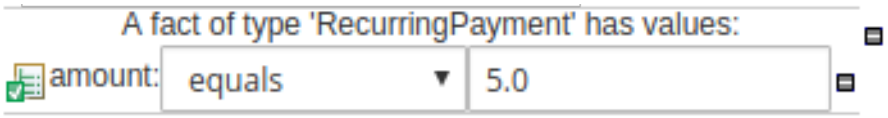


2. Given セクションで作成されたデータのセットに基づいて、3 つの予想される内容のいずれかを指定できます。

- **Rule:** 特定ルールの実行についてチェックできます。実行が予想されるルールの名前を入力するか、またはルールの一覧からこれを選択します。終了したら、**OK** をクリックします。
- **Fact value:** 特定のオブジェクトインスタンスおよびその値をチェックできます。以下の例では、顧客オブジェクトで **hasInternetService** ブール値が **true** に設定されている場合に、同じオブジェクトの **hasPhoneService** ブール値が **true** に設定されることが予想されます。



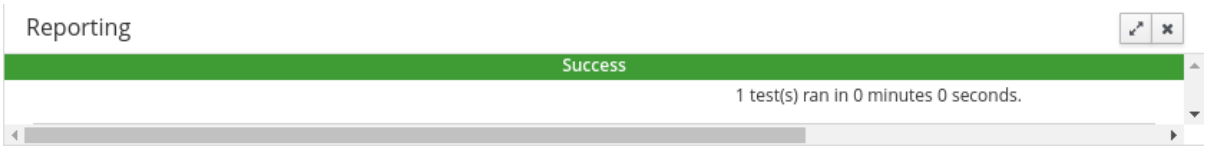
- **Any fact that matches:** 作業メモリーのオブジェクトおよびそれらのフィールドの値をチェックできます。以下の例では、インターネットサービスを有する顧客オブジェクトの場合、新規オブジェクトの RecurringPayment が **amount** フィールドが **5** にされた作業メモリーに挿入されることが予想されます。



シナリオの確認、保存および実行

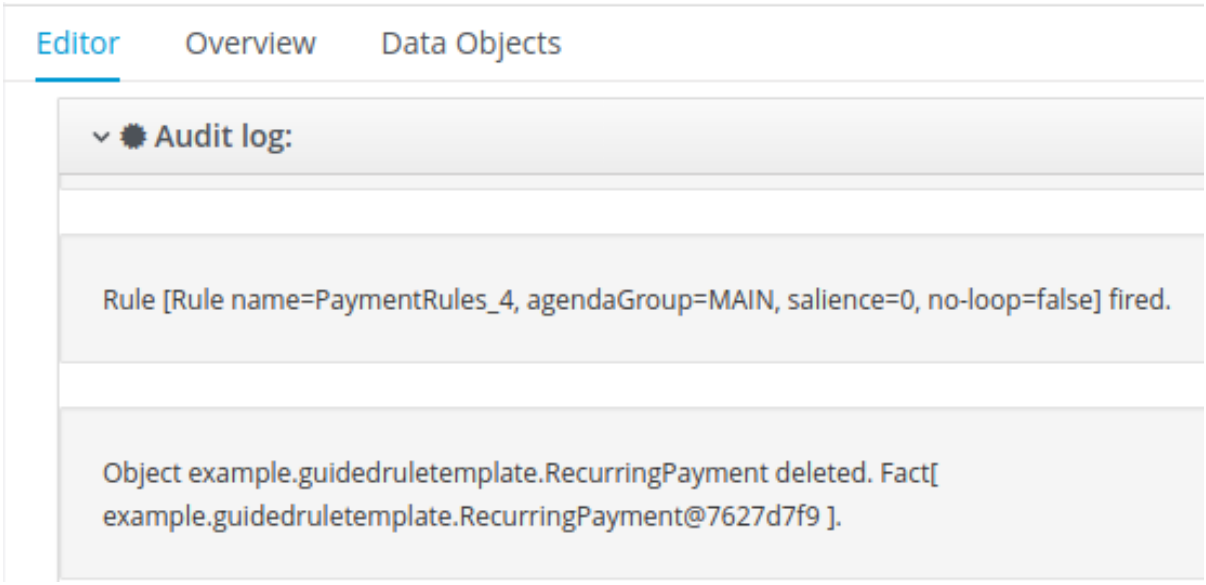
1. 右上にある **Save** をクリックしてシナリオを保存します。シナリオは定期的に保存し、確認するようにしてください。

2. 右上の **Run scenario** をクリックしてテストを実行します。結果は、この画面下の **Reporting** という新規パネルに表示されます。



3. 1 つのファイルに複数のテストを作成した場合、すべてのテストを順番に実行できます。これを実行するには、**Run all scenarios** をクリックします。

4. **Audit log:** では、挿入されたファクトおよび実行されたルールについての通知が表示されることにも留意してください。



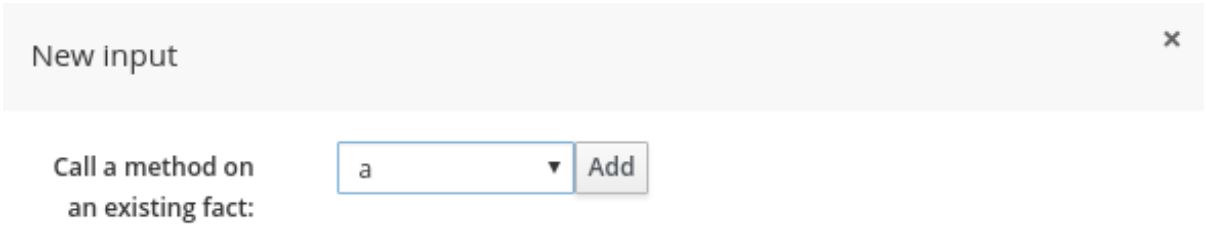
10.3. 追加のテストシナリオ機能

前述のテストシナリオ機能に加えて、テストシナリオにはその他のさまざまな機能が含まれます。

メソッドの呼び出し

1. **Call Method** により、ルールの実行開始時に既存ファクトでメソッドを呼び出すことができます。この機能には、**CALL METHOD** ラベルの横にある  をクリックしてアクセスできます。

図10.2 メソッドの呼び出し



2. ドロップダウンリストから既存ファクトを選択した後に、**Add** をクリックします。緑の矢印ボタン  を使って、ファクトに対してメソッドを呼び出すことができます。

図10.3 メソッドの呼び出し

テストシナリオでのグローバル (Global) の使用

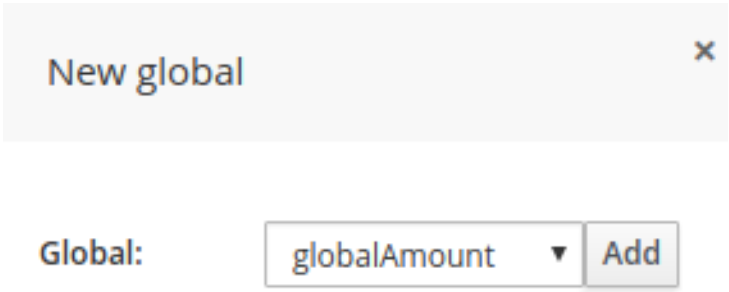
グローバル (Global) は、ルールエンジンに表示される名前付きオブジェクトですが、ファクトのオブジェクトとは異なります。そのため、グローバルのオブジェクトの変更によってルールの再評価はトリガーされません。テストシナリオでグローバルフィールドを使用し、検証することができます。

テストシナリオでグローバル変数を利用可能にするには、以下を実行します。

1. **New Item → Global Variable(s)** をクリックしてグローバル定義を作成します。
2. グローバル変数の名前およびタイプを定義します。
3. オブジェクトタイプをテストにインポートします。グローバル変数のタイプをインポートしない場合、変数はテストで利用可能になりません。

新規グローバルの追加

1. **(globals)** ラベルの横にある  をクリックしてグローバルを追加し、**Add** をクリックします。

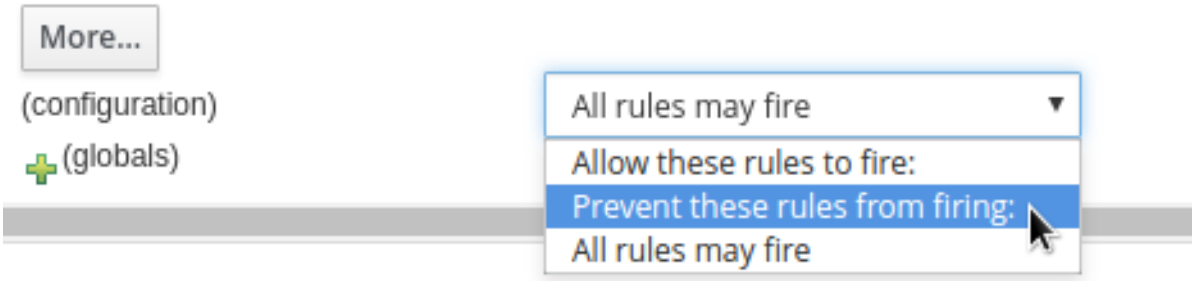


フィールドに制限を追加することは、フィールドや制限を **Given** セクションに追加することに似ています。詳細は、仮定 (Given) ファクトの指定 を参照してください。

ルールの設定

1. **(configuration)** ラベルを使用すると、以下のオプションを指定してルールの実行についての追加の制約を設定することができます。
   - **Allow these rules to fire:** 実行を許可するルールを選択できます。
   - **Prevent these rules from firing:** テストシナリオで一部のルールが実行されないようにできます。
   - **All rules may fire**: 特定のテストですべてのルールの実行を許可します。

図10.4 設定

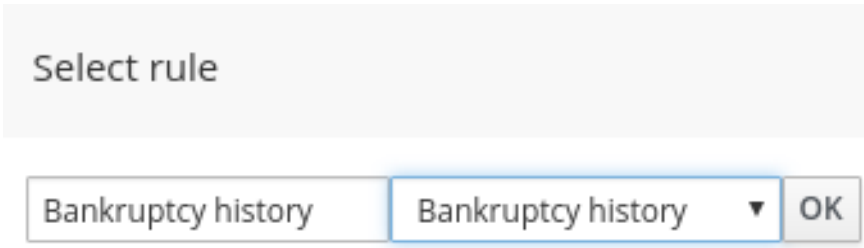


2. 以下のいずれかを選択します。

   - **Allow these rules to fire:**
   - **Prevent these rules from firing:**

     ルールを空のフィールドに入力します。空のフィールドの横にある をクリックして、条件の影響を受けるルールを選択します。

     図10.5 ルールの選択

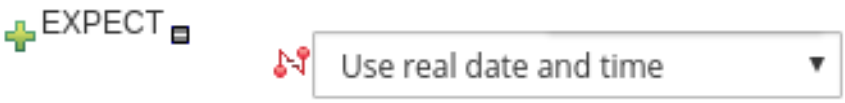


3. ドロップダウンリストからルールを選択し、**OK** をクリックします。選択されたルールが、ルール設定オプションの横にあるフィールドに表示されます。

日付と時刻の設定

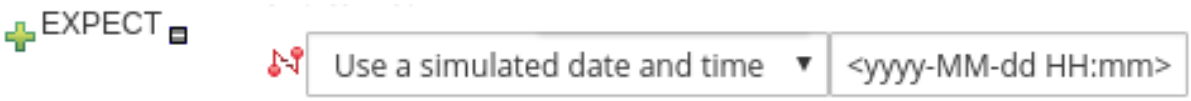
1. **Use real date and time**オプションは、テストシナリオの実行時に実際の時間を使用します。

   図10.6 実際の日時



2. **Use a simulated date and time**オプションでは、テストシナリオに関連した年、月、日、時間、分を指定することができます。

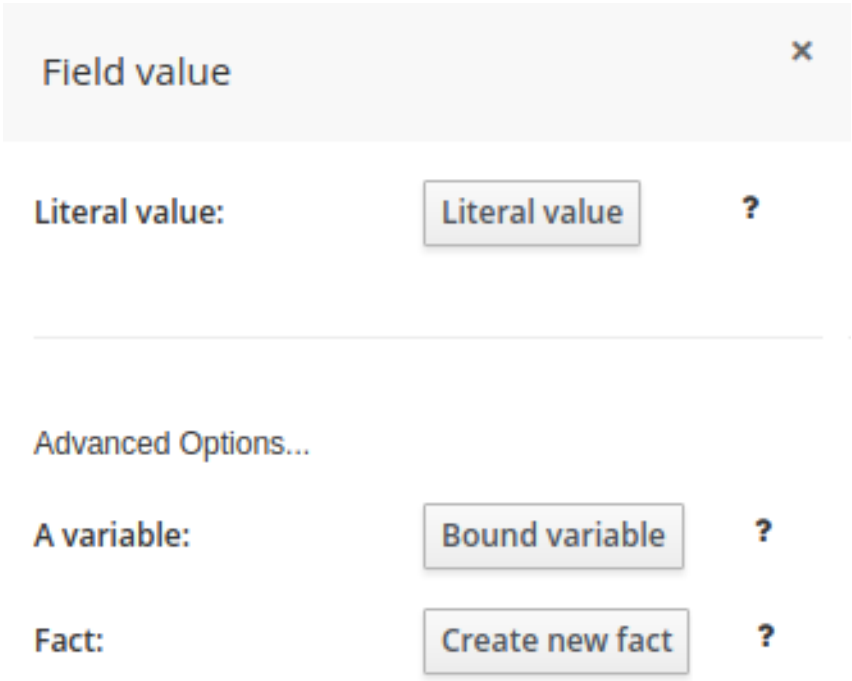
   図10.7 タイトル



詳細のファクトデータ

1. フィールドを、作成されたファクトの一部として編集可能なプロパティーに指定した後に、 をクリックして **Field value** ダイアログを開きます。リテラル値を編集するか、または詳細なファクトデータを指定することができます。

図10.8 詳細オプション



2. **Advanced Options...** セクションでは、作成されるファクトや特定のテストシナリオで使用されるモデルオブジェクトに基づいて以下のいずれかを選択できます。

- **Bound variable**: フィールドの値を、選択された変数にバインドされたファクトに設定します。フィールドタイプは、バインドされた変数タイプに一致している必要があります。
- **Create new fact** 新規ファクトを作成し、これを親ファクトのフィールド値として割り当てることができます。各種フィールド値のドロップダウンで指定されるフィールド値として割り当てられるファクトをクリックします。これらの値には、詳細なフィールド値が指定される場合があります。

その他のセクションの追加

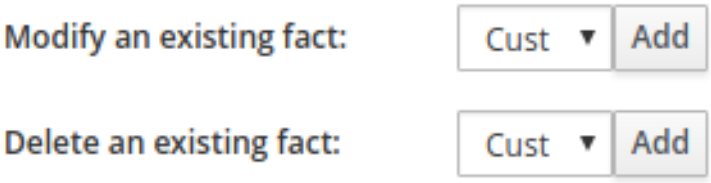
- **Editor** タブを使って、**GIVEN**、**CCALL METHOD**、 and **EXPECT** セクションをシナリオに追加できます。これを実行するには、**EXPECT** セクションの下にある **More** をクリックします。これにより、 をクリックして削除できる 3 つのすべのセクションを含むブロックが開かれます。

既存ファクトの変更または削除

1 つのファイルに複数のテストを作成する場合、以前のテストで挿入されたファクトを削除することが推奨されます。新規の **GIVEN** ファクトを挿入する際には、以下のフィールドに留意してください。

- **Modify an existing fact** ナレッジベースの実行間でファクトを編集できます。
- **Delete an existing fact** 実行間でファクトを削除できます。

図10.9 既存ファクトの変更および削除



付録A バージョン情報

Documentation last updated on: Wednesday, Oct 23, 2019.