



Red Hat Process Automation Manager 7.0

テストシナリオを使用したデシジョンサービスの
テスト

Red Hat Process Automation Manager 7.0 テストシナリオを使用したデシジョンサービスのテスト

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

法律上の通知

Copyright © 2018 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

概要

本書は、Red Hat Process Automation Manager 7.0 で、テストシナリオを使用してデシジョンサービス进行测试する方法を説明します。

目次

前書き 3

第1章 テストシナリオ 4

第2章 データオブジェクト 5

 2.1. データオブジェクトの作成 5

第3章 テストシナリオの作成および実行 7

 3.1. テストシナリオへの GIVEN ファクトの追加 9

 3.2. テストシナリオへの EXPECT 結果の追加 11

第4章 次のステップ 13

付録A バージョン情報 14

前書き

ビジネス分析者またはビジネスルールの開発者は、Business Central でテストシナリオを使用して、プロジェクトをデプロイする前にデシジョンサービスをテストできます。デシジョンサービスをテストすると、プロジェクトのルールアセットが適切に、想定通りに機能することが確認できます。デシジョンサービスは、プロジェクトの開発時にいつでもテストできます。

前提条件

- デシジョンサービスのチームおよびプロジェクトが Business Central に作成されています。詳細は『[デシジョンサービスの使用ガイド](#)』を参照してください。
- デシジョンサービスに、ビジネスルールおよび関連するデータオブジェクトが定義されています。詳細は『[ガイド付きデシジョンテーブルを使用したデシジョンサービスの作成](#)』を参照してください。



注記

テストシナリオでは、ビジネスルールを構成するように定義したデータをテストできるため、ビジネスルールを先に定義しておくことは、テストシナリオにおける技術的な前提条件ではありません。ただし、先にルールを作成しておくこと、テストシナリオでルール全体をテストすることができ、かつ意図するデシジョンサービスにシナリオがより近づくため便利です。

第1章 テストシナリオ

Red Hat Process Automation Manager のテストシナリオでは、ビジネスルールを実稼働環境にデプロイする前に、ビジネスルールの機能とデータの妥当性を確認できます。このテストシナリオでは、プロジェクトのデータを使用して、指定した条件と、定義した 1 つ以上のビジネスルールで想定される結果を設定できます。シナリオを実行する際は、想定した結果と、ルールのインスタンスから実際に得られた結果を比較します。想定した結果が実際の結果に一致するとテストに成功し、一致しない場合はテストに失敗します。

テストシナリオは、一度に 1 つずつ、またはグループ単位で実行できます。グループで実行する場合は、1 つのパッケージに含まれる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ているため、別のシナリオに影響を及ぼしたり修正したりすることはありません。テストシナリオは、Business Central のプロジェクト開発時にいつでも実行できます。テストシナリオを実行するために、デシジョンサービスをコンパイルまたはデプロイする必要はありません。

テストシナリオに関連するすべてのデータオブジェクトは、テストシナリオと同じプロジェクトパッケージに置く必要があります。同じパッケージに含まれるアセットはデフォルトでインポートされます。必要なデータオブジェクトとテストシナリオを作成したら、テストシナリオデザイナーの **Data Objects** タブを使用して、必要なデータオブジェクトがすべてリストされていることを検証するか、**アイテムを追加**して既存のデータオブジェクトをインポートします。

第2章 データオブジェクト

データオブジェクトは、作成するルールアセットの構成要素です。データオブジェクトは、プロジェクトで指定したパッケージに Java オブジェクトとして実装されているカスタムのデータタイプです。たとえば、データフィールド **Name**、**Address**、および **DateOfBirth** を使用して **Person** オブジェクトを作成し、ローン申請ルールに詳細な個人情報を指定できます。このカスタムのデータタイプは、アセットとデシジョンサービスがどのデータに基づいているかを指定します。

2.1. データオブジェクトの作成

以下の手順は、データオブジェクトを作成する際の一般的な概要で、特定のビジネスプロセスに固有のものではありません。

手順

1. Business Central で、**Menu** → **Design** → **Projects** に移動して、プロジェクト名をクリックします。
2. **Add Asset** → **Data Object** の順にクリックします。
3. 一意の **データオブジェクト** 名を入力し、**パッケージ** を選択します。これにより、その他のルールアセットでもデータオブジェクトを利用できるようになります。同じパッケージに、同じ名前のデータオブジェクトを複数作成することはできません。指定するパッケージは、そのデータオブジェクトを必要とするルールアセットが割り当てられている、もしくはこれから割り当てるパッケージにする必要があります。

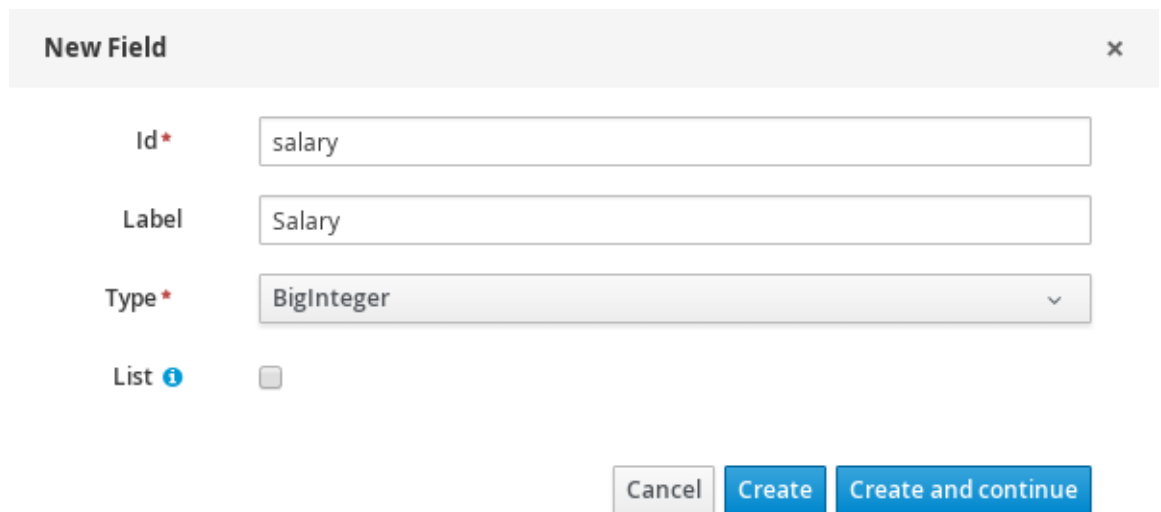


別のパッケージからのデータオブジェクトのインポート

別のパッケージから、ルールアセットのパッケージに、既存のデータオブジェクトをインポートすることもできます。プロジェクトに関連するルールアセットを選択し、アセットデザイナーで **Data Objects** → **New item** に移動して、インポートするオブジェクトを選択します。

4. データオブジェクトを永続化するには、**Persistable** チェックボックスを選択します。永続型データオブジェクトは、JPA 仕様に準じてデータベースに保存できます。デフォルトの JPA は Hibernate です。
5. **OK** をクリックします。
6. データオブジェクトデザイナーで **add field** をクリックして、**Id** 属性、**Label** 属性、**Type** 属性を使用するオブジェクトにフィールドを追加します。必須属性にはアスタリスク (*) マークが付いています。
 - **Id:** フィールドの一意の ID を入力します。
 - **Label:** (任意) フィールドのラベルを入力します。
 - **Type:** フィールドのデータ型を入力します。
 - **List:** このチェックボックスを選択すると、このフィールドで、指定したタイプのアイテムを複数保持できるようになります。

図2.1 データオブジェクトへのデータフィールドの追加




New Field x

Id* salary

Label Salary

Type* BigInteger v

List  ☐

Cancel Create Create and continue

7. **Create** をクリックして、新しいフィールドを追加します。**Create and continue** をクリックすると、新しいフィールドが追加され、別のフィールドを引き続き作成できます。



注記

フィールドを編集するには、フィールド行を選択し、画面右側の **general properties** を使用します。

第3章 テストシナリオの作成および実行

ビジネスルールデータをデプロイする前に、Business Central にテストシナリオを作成して、その機能をテストできます。基本的なテストシナリオには、少なくとも以下のデータが必要です。

- 関連するデータオブジェクト
- **GIVEN** (指定した) ファクト
- **EXPECT** (想定される) 結果

テストシナリオでは、このデータを使用し、定義したファクトに基づいて、そのルールインスタンスに対して想定した結果と実際の結果の妥当性を検証できます。**CALL METHOD** と利用可能な **globals** をテストシナリオに追加することもできますが、これは必須ではありません。

手順

1. **Menu** → **Design** → **Projects** に移動して、プロジェクト名をクリックします。
2. **Add Asset** → **Test Scenario** の順にクリックします。
3. テストシナリオ 名を入力し、適切な **パッケージ** を選択します。指定するパッケージは、必要なデータオブジェクトとルールアセットが割り当てられている、またはこれから割り当てるパッケージにする必要があります。
4. **OK** をクリックして、テストシナリオを作成します。
Project Explorer の **Test Scenarios** パネルに、新しいテストシナリオが追加されました。
5. **Data Objects** タブをクリックして、テストするルールに必要なデータオブジェクトがすべてリストされていることを検証します。追加されていない場合は、**New item** をクリックして別のパッケージから必要なデータオブジェクトをインポートするか、パッケージに **データオブジェクトを作成** します。
6. データオブジェクトをすべて配置したら、テストシナリオデザイナーの **Model** タブに戻り、利用可能なデータオブジェクトのシナリオに、**GIVEN** データと **EXPECT** データを定義します。

図3.1 テストシナリオデザイナー


The screenshot shows the Test Scenario Designer interface with the following components:

- + GIVEN** button
- Block 1: Insert 'Applicant' [a]
 - age: 17
 - 'Applicant' facts
- Block 2: Insert 'LoanApplication' [application]
 - amount: 1
 - 'LoanApplication' facts
- Block 3: Insert 'IncomeSource' [incomeSource]
 - Add a field
 - 'IncomeSource' facts
- + CALL METHOD** button
- Text: Add input data and expectations here.
- + EXPECT** button
- Block 4: LoanApplication 'application' has values:
 - approved: equals false
 - 'application'
- Delete one scenario block above** button
- More...** button
- + (globals)** button

GIVEN セクションには、テストする入力ファクトを定義します。たとえば、プロジェクトの **Underage** ルールで、ローン申請者の年齢が 21 歳未満であれば承認しない場合は、テストシナリオの **GIVEN** ファクトで、**Applicant** の **age** に、21 より小さい数字に設定する必要があります。

EXPECT セクションには、**GIVEN** に入力したファクトに基づいて想定される結果を定義します。つまり、入力ファクトを **GIVEN** (指定) すると、その他のファクトが有効であること、またはルール全体が有効であることを **EXPECT** (想定) します。たとえば、このシナリオで、申請者が 21 歳未満の場合に **想定される** 結果は、(申請者の年齢が基準を満たさないため) **LoanApplication** の **approved** が **false** になるか、**Underage** ルール全体が有効になります。

7. **CALL METHOD** と **globals** をテストシナリオに追加することもできます。

- **CALL METHOD:** ルールの実行を開始する際に、別のファクトからメソッドを呼び出します。**CALL METHOD** をクリックし、ファクトを選択し、 をクリックして呼び出すメソッドを選択します。(可能な場合は) プロジェクトに対してインポートした Java ライブラリー、または JAR から (ArrayList のメソッドなどの) Java クラスメソッドを呼び出すことができます。
- **globals:** テストシナリオで妥当性を確認するプロジェクトにグローバル変数を追加します。**globals** をクリックして、妥当性を確認する変数を選択し、テストシナリオデザイナーでグローバル名をクリックして、グローバル変数に適用するフィールド値を定義します。グローバル変数が利用できない場合は、Business Central に新しいアセットとして作成する

必要があります。グローバル変数は、プロセスエンジンに表示されますが、ファクトに対するオブジェクトとは異なるオブジェクトの名前です。グローバルのオブジェクトを変更しても、ルールの再評価は行われません。

8. 必要に応じて、テストシナリオデザイナーの下部で **More** をクリックし、同じシナリオファイルに別のデータブロックを追加します。
9. シナリオに対して、**GIVEN**、**EXPECT**、その他のデータをすべて定義したら、テストシナリオデザイナーで **Save** をクリックして、設定した内容を保存します。
10. 右上の **Run scenario** をクリックして、この **.scenario** ファイルを実行します。プロジェクトパッケージに **.scenario** ファイルを複数保存している場合は、**Run all scenarios** をクリックして、すべてのシナリオを実行します。**Run scenario** オプションでは、対象の **.scenario** ファイルを保存する必要はありませんが、**Run all scenarios** オプションを使用する場合は、すべての **.scenario** ファイルを保存する必要があります。
テストに失敗したら、ウィンドウ下部の **レポート** メッセージに記載されている問題に対応し、シナリオの全コンポーネントを見直し、エラーが表示されなくなるまで妥当性確認を行います。
11. 変更がすべて終了したら、テストシナリオデザイナーで **Save** をクリックして、設定した内容を保存します。

GIVEN ファクトをテストシナリオに追加する方法は [「テストシナリオへの GIVEN ファクトの追加」](#) を参照してください。

EXPECT 結果をテストシナリオに追加する方法は [「テストシナリオへの EXPECT 結果の追加」](#) を参照してください。

3.1. テストシナリオへの GIVEN ファクトの追加

GIVEN セクションには、テストする入力ファクトを定義します。たとえば、プロジェクトの **Underage** ルールで、ローン申請者の年齢が 21 歳未満であれば承認しない場合は、テストシナリオの **GIVEN** ファクトで、**Applicant** の **age** に、21 より小さい数字を設定する必要があります。

前提条件/事前作業

テストシナリオに必要なデータオブジェクトがすべて作成、またはインポートされていて、テストシナリオデザイナーの **Data Objects** タブにリストされています。

手順


1. テストシナリオデザイナーで **GIVEN** をクリックすると、利用可能なファクトが追加されている **New input** ウィンドウが開きます。

図3.2 テストシナリオへの GIVEN 入力追加

リストには以下のオプションが含まれます。表示されるオプションは、テストシナリオデザイナーの **Data Objects** タブで利用可能なデータオブジェクトによって異なります。

- **Insert a new fact:** ファクトを追加して、フィールド値を修正します。**Fact name** にファクトの変数を入力します。
 - **Modify an existing fact:** (別のファクトが追加されている場合に限り表示) シナリオの実行と実行の間に、プロセスエンジンにすでに挿入されていて、修正したいファクトを指定します。
 - **Delete an existing fact:** (別のファクトが追加されている場合に限り表示) シナリオの実行と実行の間に、プロセスエンジンにすでに挿入されていて、削除したいファクトを指定します。
 - **Activate rule flow group:** そのグループ内にあるすべてのルールをテストできるように、有効にするルールフローグループを指定します。
2. 目的の入力オプションに対するファクトを選択し、**Add** をクリックします。たとえば、**Insert a new fact:** に **Applicant** を設定し、**Fact name** に対して **a** または **app**、もしくは別の変数を入力します。
 3. テストシナリオデザイナーのファクトをクリックし、修正するフィールドを選択します。

図3.3 ファクトフィールドの修正

4. 編集アイコン () をクリックし、以下のフィールド値を選択します。
 - **Literal value:** 特定のリテラル値を入力するオープンフィールドを作成します。

- **Bound variable:** フィールドの値を、選択した変数にバインドするファクトに設定します。フィールドタイプが、バインドした変数型に一致する必要があります。
 - **Create new fact:** 新しいファクトを作成し、そのファクトを親ファクトのフィールド値として割り当てます。テストシナリオデザイナーで子ファクトをクリックし、同じようにフィールド値を割り当てるか、別のファクトをネストできます。
5. 続いて、シナリオに別の **GIVEN** 入力データを追加し、テストシナリオデザイナーで **Save** をクリックして、設定した内容を保存します。

3.2. テストシナリオへの EXPECT 結果の追加

EXPECT セクションには、**GIVEN** に入力したファクトに基づいて想定される結果を定義します。つまり、入力ファクトを **GIVEN (指定)** すると、その他のファクトが有効であること、またはルール全体が有効であることを **EXPECT (想定)** します。たとえば、シナリオで、申請者が 21 歳未満の場合に **想定される結果は、(申請者の年齢が基準を満たさないため) LoanApplication の approved が false** になるか、**Underage** ルール全体が有効になります。

前提条件/事前作業

テストシナリオに必要なデータオブジェクトがすべて作成、またはインポートされていて、テストシナリオデザイナーの **Data Objects** タブにリストされています。

手順

1. テストシナリオデザイナーで **EXPECT** をクリックし、利用可能なファクトが追加されている **New expectations** ウィンドウを開きます。

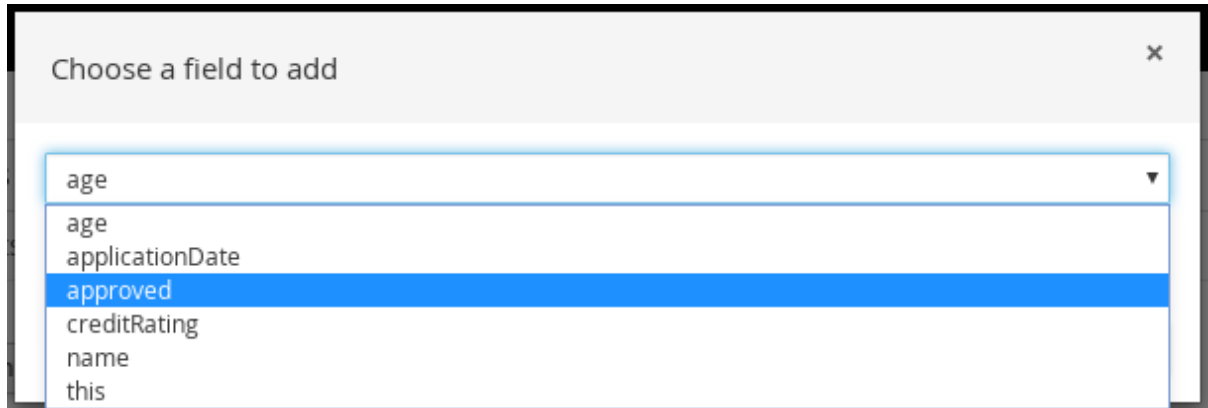
図3.4 テストシナリオへの EXPECT 結果の追加

リストには以下のオプションが含まれます。表示されるオプションは、**GIVEN** セクションのデータや、テストシナリオデザイナーの **Data Objects** タブで利用可能なデータオブジェクトによって異なります。

- **Rule:** プロジェクトに、**GIVEN** に指定した内容に対して有効になることが想定される特定のルールを指定します。ルールの名前を入力するか、ルールリストから選択します。次に、テストシナリオデザイナーで、ルールが有効になるべき回数を指定します。
- **Fact value:** ファクトを選択し、**GIVEN** セクションに定義したファクトに対して有効になることが想定される値を定義します。ファクトは、**GIVEN** の入力に対して事前に定義した **Fact name** でリストされます。
- **Any fact that matches:** **GIVEN** に指定した内容に対して、指定した値を持つファクトが最低 1 つ存在するかどうかの妥当性を確認します。

2. (**Fact value: application** など) 期待される結果のファクトを選択し、**Add** または **OK** を選択します。
3. テストシナリオデザイナーでファクトをクリックし、追加または修正するフィールドを選択します。

図3.5 ファクトフィールドの修正



4. フィールド値に、**GIVEN** に指定した内容に対して、有効になると想定される値 (**approved | equals | false** など) を設定します。
5. 続いて、シナリオに別の **EXPECT** 入力データを追加し、テストシナリオデザイナーで **Save** をクリックして、設定内容を保存します。
6. シナリオに **GIVEN**、**EXPECT**、その他のデータを定義して保存したら、右上の **Run scenario** をクリックしてこの **.scenario** ファイルを実行します。プロジェクトパッケージに保存した **.scenario** が複数ある場合は、**Run all scenarios** をクリックして保存したすべてのシナリオを実行します。**Run scenario** オプションでは、対象の **.scenario** ファイルを保存する必要はありませんが、**Run all scenarios** オプションを使用する場合は、すべての **.scenario** ファイルを保存する必要があります。
テストに失敗したら、ウィンドウ下部の **レポート** メッセージに記載されている問題に対応し、シナリオの全コンポーネントを見直し、エラーが表示されなくなるまで妥当性確認を行います。
7. 変更がすべて終了したら、テストシナリオデザイナーで **Save** をクリックして、設定した内容を保存します。

第4章 次のステップ

Packaging and deploying a Red Hat Process Automation Manager project

付録A バージョン情報

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