IBM Operational Decision Manager
Version 8 Release 5

Getting Started with Business Rules

Note
Before using this information and the product it supports, read the information in "Notices" on page 43.

This edition applies to version 8, release 5, modification 1 of Operational Decision Manager and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Tutorial: Getting started with business rules

This tutorial helps you to take your first steps with Operational Decision Manager V8.5.1. In this tutorial, you learn how to use Rule Designer to create and run a rule-based application, and Rule Execution Server to execute the rules. If you are performing this tutorial for the first time, read the related sections as well.

Learning objectives

In this tutorial, you learn how to:

- Design a rule project.
- Orchestrate the rules and define a flow of execution.
- Write business rules, and then test and debug the rules.
- Deploy the rules to the execution environment.
- Monitor and audit the rules.
- Publish the rule project to the web-based business environment.

Time required

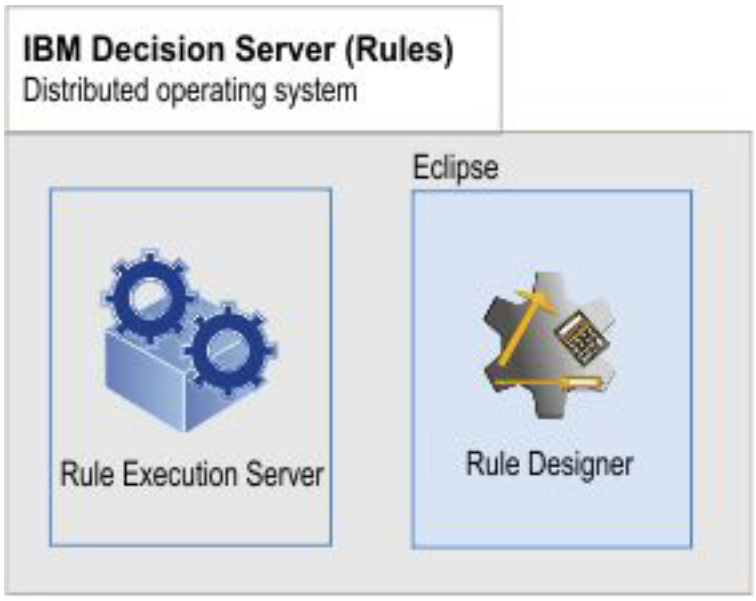
 This tutorial takes between 3 and 4 hours to complete.

Introduction: Getting started with business rules

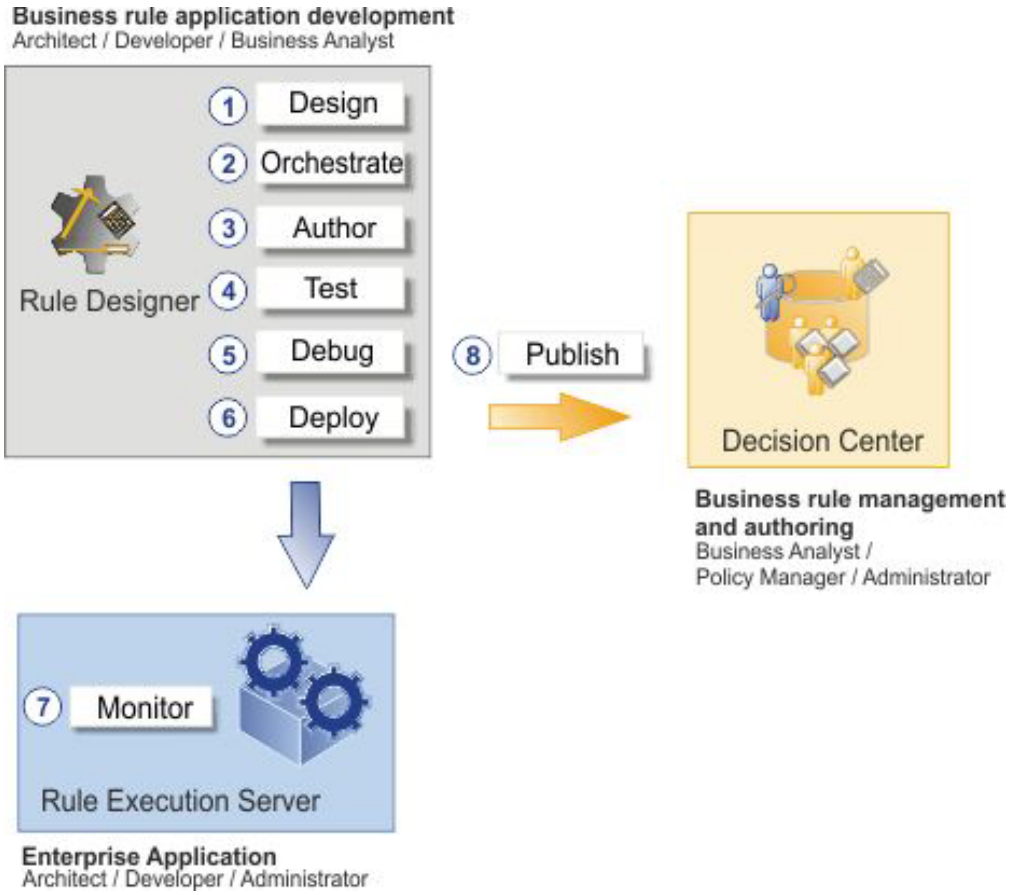
In this tutorial, you learn how to use Rule Designer to create a rule-based application, and Rule Execution Server to execute the rules.

This tutorial demonstrates the fictitious web-based application of an online lender. The application is called Miniloan. The Miniloan application decides whether a customer is eligible for a loan according to several criteria. The criteria include the amount of the loan, the yearly income of the borrower, and the duration of the loan.

The rules development and authoring environment is provided by Rule Designer. The rules execution environment is provided by Rule Execution Server.



The following figure shows the workflow and interactions between the modules.



Prerequisites

A knowledge of Eclipse workspaces, perspectives, and views is helpful. If you are not familiar with Eclipse, go through the short Eclipse Hello World cheat sheet.

To perform this tutorial make sure that you installed the required products:

- Decision Server Rules: contains both Rule Designer and Rule Execution Server.
- Decision Center (optional): only required if you want to perform "Task 8: Publishing to Decision Center" on page 37.
- A supported version of Microsoft Excel to perform "Task 4: Testing rules" on page 20.

To perform this tutorial, the sample server is also required. The sample server is installed in one of the following ways:

- By using the launchpad sample server installation. For more information, see Installing the product and the sample server
- By using the default set of features installed with Installation Manager. For more information, see Installing using Installation Manager

To check that the required products, getting started projects, and the sample server have been installed, see Checking your installation.

For more information about the required products and their modules, and how to install them, see Installing Operational Decision Manager.

Important: You must open Rule Designer in American English. The rule project that contains the rules to import in this tutorial is only provided in American English (en_US).

Audience

This tutorial is aimed at developers and architects.

Learning objectives

In this tutorial, you learn how to:

- Design a rule project.
- Orchestrate the rules and define a flow of execution.
- Write business rules, and then test and debug the rules.
- Deploy the rules to the execution environment.
- Monitor and audit the rules.
- Publish the rule project to the web-based business environment.

Time required

 This tutorial takes between 3 and 4 hours to complete.

Starting the Miniloan web application

To start the Miniloan application, which demonstrates this Getting started tutorial, you start a server and then view it with your browser.

The scenario of the Getting started tutorial is based on the fictitious web-based application of an online lender.

In this tutorial, you start from a version of this application that is hard-coded, and you replace its logic with business rules. By doing so, you discover all the steps required to develop, deploy, and maintain a rule-based application.

First, take a look at the Miniloan application in its initial state.

To start Miniloan:

1. On the **Start** menu, click **All Programs** > **IBM** > *package_group* > **Sample Server** > **Start server**.

   package_group refers to the package group specified in IBM® Installation Manager during installation. The default package group is Operational Decision Manager V8.5.1.

   Note: On Windows 7, if you have installed the product in the `Program Files` or `Program Files (x86)` directories, you must be an administrator to start the sample server. You can run the sample server as an administrator, or obtain write permissions on the Operational Decision Manager installation directory.
2. Wait until the server has started.

   It can take a while for the server startup procedure to complete. The command window displays server trace messages as the server starts. A feedback message indicates when the server has started successfully:

   ```
   [samples.echo] GBRPS0029I: start.server is completed.

   BUILD SUCCESSFUL
   Total time: 20 minutes 3 seconds
   Press any key to continue . . .
   ```

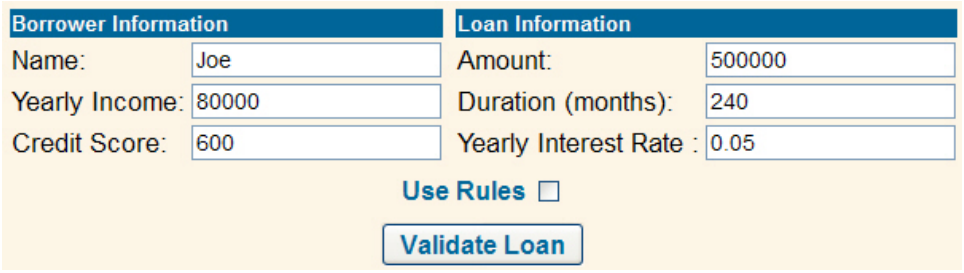
   If you encounter difficulties starting the sample server, see Using the sample server.
3. Enter the following URL with the correct port number in a browser:

   `http://localhost:<PORT>/miniloan-server`

   Important:

   In the URLs to the Miniloan application and Rule Execution Server, you must replace `<PORT>` with the correct port number. The default port number for WebSphere® Application Server is *9080*, but it can be different depending on your installation. For more information, see Checking the server port number.

   The Miniloan application is displayed:



   By default, the Miniloan application runs using a business logic implemented in Java™.

   Important:

   Do not select the **Use Rules** check box yet. You will select it later in the tutorial, after creating the rules.

4. Click **Validate Loan**.

   The results of the validation show that the loan was rejected because the borrower's debt-to-income ratio is too big:

   ```
   The loan is rejected
   Messages:
   The debt-to-income ratio is too big.
   ```

5. Close the Miniloan application.

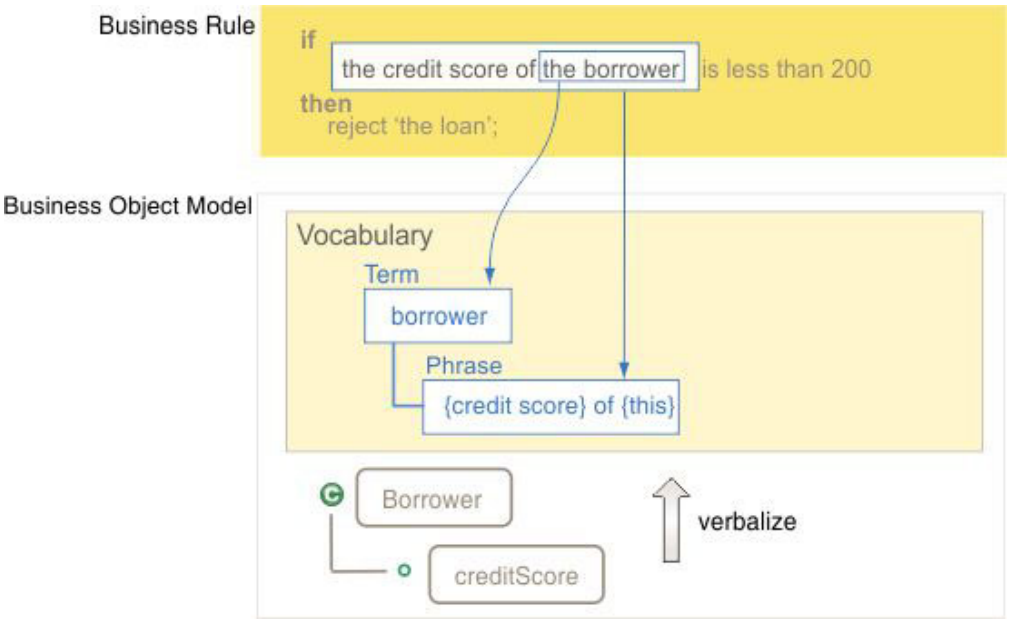
To start creating an application with rules, go to "Task 1: Designing the rule project"

Task 1: Designing the rule project

To help business users write rules easily, you must create a meaningful vocabulary for them.

In this task, you use Rule Designer to create a vocabulary directly from the object model of the existing Miniloan application.

Business users need to write and edit rules using familiar terms. As the rule project developer, you must create a business rule vocabulary for them. The process for creating this vocabulary is called "verbalization". You create a Business Object Model (BOM) based on an object model defined in a Java project. The classes and members of the BOM map to the terms and phrases familiar to the business user, as follows:



This task should take you about 20 to 30 minutes to complete.

Step 1: Start Rule Designer

Rule Designer is the development environment for business rule applications. Developers can take advantage of its integration with Eclipse to develop Java projects along with rule projects.

In this tutorial, the rule project containing the rules to import is provided in American English (en_US) only. Therefore, you must start Rule Designer in American English (en_US) to edit and create business rules. To start Rule Designer in American English (en_US) and to import the projects for this tutorial, you open the samples console.

To open the samples console:

1. From the **Start** menu, click **All Programs** > **IBM** > *package_group* > **Sample Server** > **Samples Console (en_US)** . *package_group* refers to the package group specified in IBM Installation Manager during installation. The default package group is Operational Decision Manager V8.5.1.

   The Workspace Launcher dialog shows your default workspace. If the workspace is not empty, switch to a new and empty workspace.
2. To create a new workspace, change the name of the workspace to a new name, and click **OK**.

   Tip: If you do not see the Workspace Launcher dialog when starting Rule Designer, change the workspace by clicking **File** > **Switch Workspace**.
3. Click **Open Perspective** > **Samples Console**. The Samples Console perspective opens on the Samples and Tutorials view which contains the samples and tutorials to discover the components of Operational Decision Manager. You can view the instructions and import the projects for each sample or tutorial.
4. Under **Getting Started**, expand **Decision Server**, and under **start**, click **Import projects**.

   Rule Designer imports the projects for the Getting Started tutorial, and switches to the 

Rule perspective.

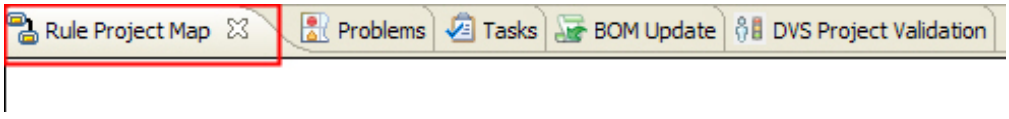
The Rule perspective contains various views that you will discover in this tutorial:

The Rule Explorer currently contains two projects:

- `miniloan-server-webapp`: the project for the Miniloan web application from which you retrieve the hard-coded rules.

  You can inspect these rules in the `validateWithJava` method of the `miniloan-server-webapp/src/miniloanweb/MiniloanBean.java` class.
- `miniloan-xom`: the Java project for the Miniloan application, which is composed of the `miniloan` package that includes the `Borrower` and the `Loan` Java classes.

The Rule Project Map guides you through the different steps for setting up a rule project. It is currently empty because you have not created a rule project yet.



If the Rule Project Map is not displayed, click **Window** > **Show View** > **Rule Project Map** to open it.

Step 2: Create a rule project

The Miniloan business object model is composed of two classes, one for the borrower and one for the loan. Before creating the rule project itself, make sure that the `miniloan-xom` Java project is displayed in the Rule Explorer.

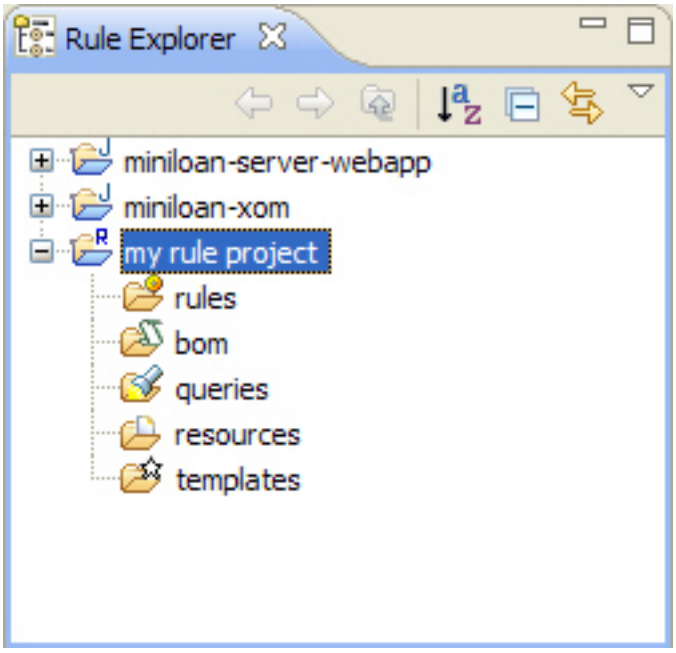
In Rule Designer, you store the business logic of your application in a rule project. The rule project contains rule artifacts, business object model (BOM), vocabulary, and reference to the execution object model (XOM). This project enables you to manage, build, and debug the items that comprise the business logic of your application.

To create the rule project:

1. Make sure that you are in the Rule **Rule** perspective.

   Tip: To switch to the Rule perspective, click the **Window** menu, click **Open Perspective** > **Other**, then select **Rule**, and then click **OK**.
2. Click **File** > **New** > **Rule Project**.
3. Select **Standard Rule Project** and click **Next**.
4. In the **Project name** field, type `my rule project`.
5. Click **Finish**.

   `my rule project` opens in the Rule Explorer, and the Rule Project Map is now active.



For now, the rule project only contains empty folders. During the tutorial you will use the `rules` and `bom` folders to store your rules and your BOM.

Step 3: Attach the Java project

Now that you have an empty rule project, you can use the Rule Project Map to guide you through the steps of building the project.

The first thing that your rule project needs is the object model of the Miniloan Java project, which you imported into your workspace. This is referred to as the execution object model (XOM).

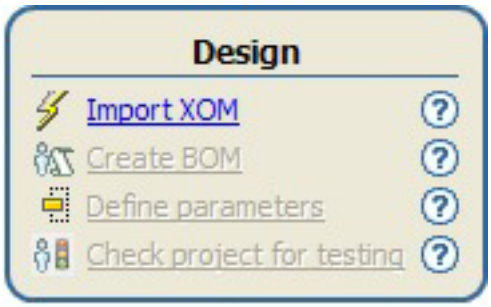
Note: Alternatively, you can use an XML Schema for the execution object model.

To import the XOM into your rule project:

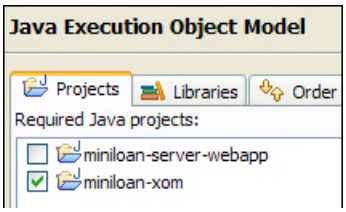
1. In the Rule Explorer, click `my rule project` to select it.

The Rule Project Map displays the steps to follow to design your rule project.

2. In the Design part of the Rule Project Map, click **Import XOM**.



3. In the Import XOM dialog, select **Java Execution Object Model** and click **OK**.
4. Under **Required Java projects**, select `miniloan-xom`.



5. Click **OK**.
   The Rule Project Map shows that you now have one XOM in your rule project.

Step 4: Create the BOM

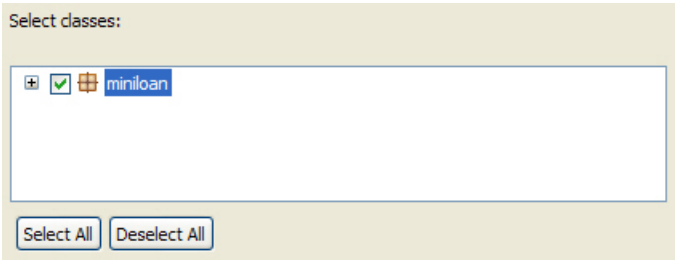
Before you create and edit rules, you need to define a business object model (BOM). You can create a BOM from scratch or create it automatically by parsing your execution object model (XOM).

Here you use Rule Designer to parse your Java classes (XOM) automatically and create the BOM from their methods and properties. Then, you can write rules from the vocabulary terms that are contained in the BOM.

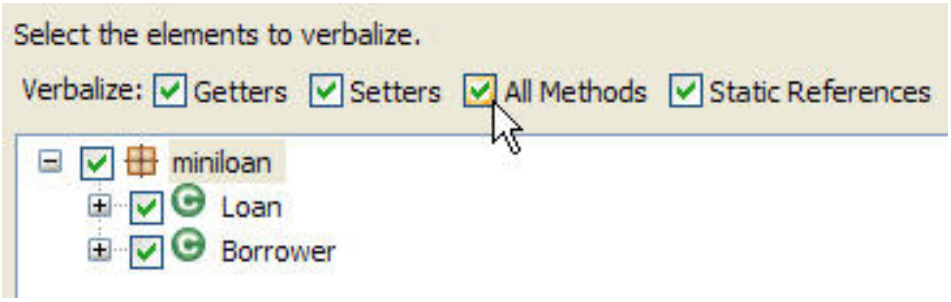
To create a BOM from the XOM:

1. In the Design part of the Rule Project Map, click **Create BOM**.

   Tip: You can also right-click the `bom` folder in the Rule Explorer and click **New** > **BOM Entry**.
2. In the New BOM Entry wizard, in the **Name** field, type `miniloan`.
3. Make sure that **Create a BOM entry from a XOM** is selected, and then click **Next**.
4. In the **Choose a XOM entry** field, click **Browse XOM**, select `platform:/miniloan-xom`, and then click **OK**.
5. Under **Select classes**, select the `miniloan` package. Selecting the package selects all the classes in the package.

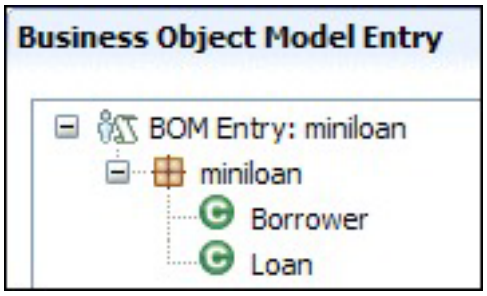


6. Click **Next**.
7. In the BOM Verbalization page, you must select the **All Methods** check box. This ensures that all methods are verbalized in addition to the elements already selected.



8. Click **Finish**.
9. In the Rule Explorer, double-click `bom > miniloan` to open the BOM Editor.

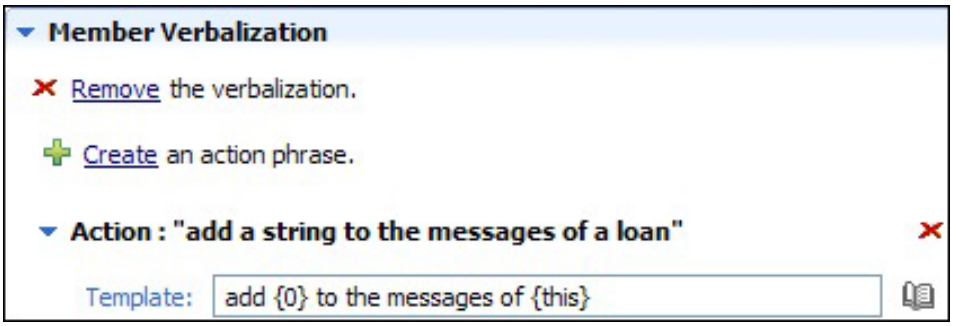
   Take a few moments to look at your BOM. In the BOM Editor, expand the `miniloan` entry:



   You now have in your BOM two classes equivalent to those in the XOM: one for the borrower and one for the loan.
10. Double-click the `Loan` class to open it in the BOM editor.

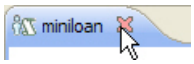
    All Java members and methods have been converted and assigned a default verbalization, which you can change.
11. In the Members section, double-click the `addToMessages(String)` method.

    The BOM editor switches to the Member tab. In the Member Verbalization section, you can see that the verbalization of this method is `add a string to the messages of a loan`:



This verbalization will also be used in the Rule Editor.

12. Close the `miniloan` tab to close the BOM Editor:



Step 5: Declare ruleset parameters

A ruleset is an executable package that includes rule artifacts and other elements required for execution. It contains a set of rules that can be executed by the rule engine. Ruleset parameters are part of the design of the project because they define the data that is sent to the rule engine and the type of information that can be retrieved. Rules can then use these parameters to manipulate objects passed to the rule engine, as you will see later.

Ruleset parameters are equivalent to Java method parameters. They are references that you can use when you write your rules.

To enable a decision to be made on the status of a loan, you need to create ruleset parameters for a borrower and a loan:

- The `borrower` must be an `IN` parameter. The value of the `IN` parameter is provided as input to the ruleset on execution.
- The `loan` must be an `IN_OUT` parameter. The value of the `IN_OUT` parameter is provided as input to the ruleset on execution, and can be modified by the ruleset and provided as output at execution completion.

To declare ruleset parameters:

1. In the Design part of the Rule Project Map, click **Define parameters**.

   Tip: You can also right-click the `my rule project` project in the Rule Explorer and click **Properties**.
2. In the Properties dialog, make sure that **Ruleset Parameters** is selected.
3. To define the `borrower` parameter, click **Add**.

   A new row is displayed with default values. Change the values as follows:
   a. In the **Name** column, type `borrower`.
   b. Click the cell in the **Type** column, click the **...** button to display the Types dialog, and then double-click the **Borrower** type in the Matching types box.

      The `miniloan.Borrower` type is displayed in the **Type** column.
   c. In the **Direction** column, select the **IN** direction.
   d. In the **Verbalization** column, type `the borrower`.
4. To define the `loan` parameter, click **Add**.
   a. In the **Name** column type `loan`.

b. Click the cell in the **Type** column, click the **...** button to display the Types dialog, and then double-click the **Loan** type in the Matching types box.
   The `miniloan.Loan` type is displayed in the **Type** column.
c. In the **Direction** column, keep the default **IN_OUT** direction.
d. In the **Verbalization** column, type `the loan`.
   Your ruleset parameters are displayed as follows:

Ruleset Parameters

Define ruleset parameters.

Name	Type	Direction	Default Value	Verbalization
borrower	miniloan.Borrower	IN		the borrower
loan	miniloan.Loan	IN_OUT		the loan

5. Click **OK** to save.

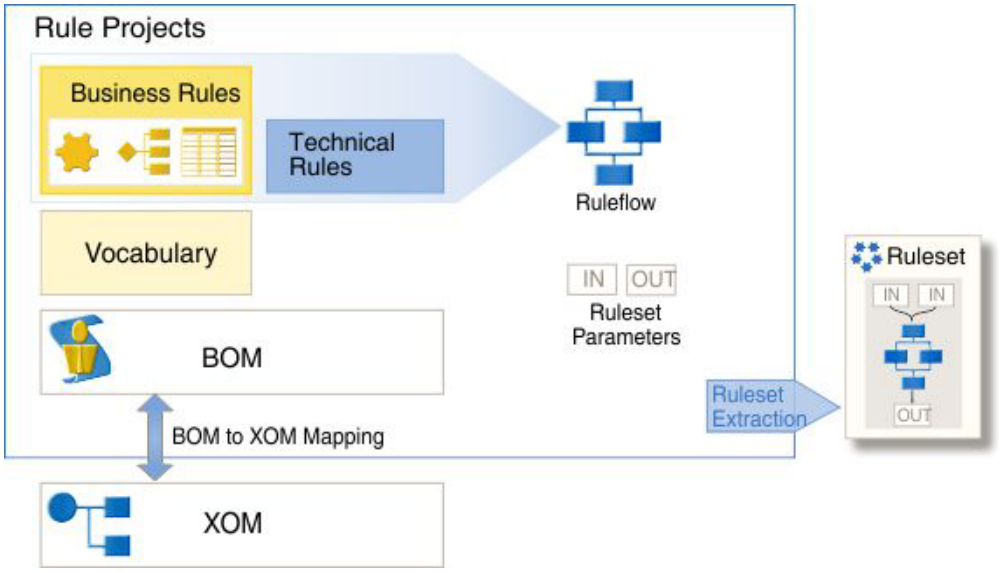
You now have a rule project with a vocabulary, and ruleset parameters. You have completed the design part of your rule project.

Before writing the actual rules in Rule Designer, you will orchestrate how your rules will be executed. You do this with a ruleflow in the next task.

Task 2: Orchestrating

In this task, you use a ruleflow to specify the order in which rules are executed.

A ruleflow is a way to organize the sequence in which rules are processed by the rule engine. In Rule Designer you orchestrate rule execution using a ruleflow.



This task should take you about 15 to 25 minutes to complete.

Step 1: Create rule packages

The Miniloan application first validates the data about the loan and the borrower, and, if the data is valid, assesses whether the borrower is eligible for the loan.

When defining the flow of execution, you organize your rules into packages that contain related rules. In this case, you have a package of rules related to validation, and another related to eligibility. You then treat these rule packages as tasks in the ruleflow.

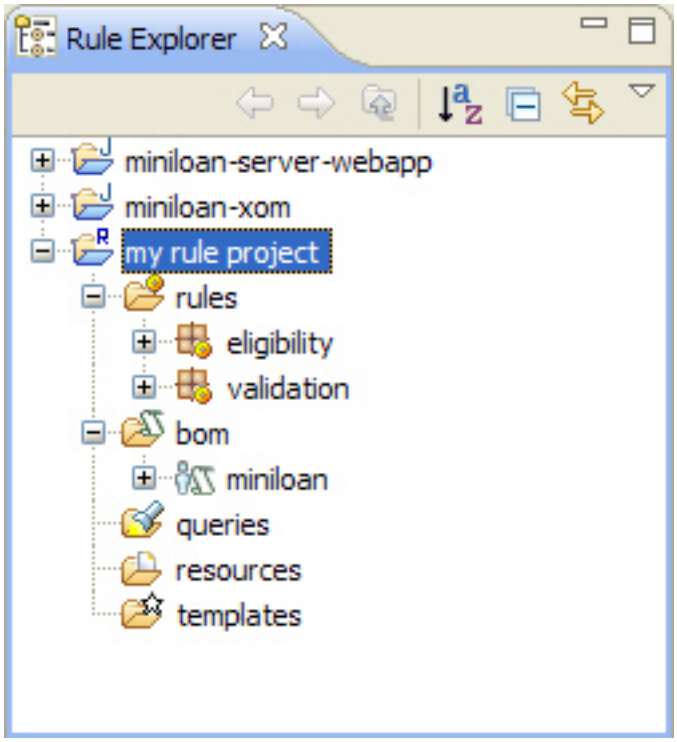
To create a rule package:

1. In Rule Designer, in the Orchestrate part of the Rule Project Map, click **Add rule package**.

   Tip: You can also right-click the `my rule project/rules` folder in the Rule Explorer and click **New** > **Rule Package**.
2. In the New Rule Package wizard, enter `validation` in the **Package** field, and then click **Finish**.

   The new `validation` rule package opens in the Rule Explorer.
3. Create another package named `eligibility`.

Your rule project now contains two packages for storing your rules:



Step 2: Create the ruleflow diagram

To define the high level flow of execution of business rules, you create a ruleflow. A ruleflow consists of rule tasks and logical links between these tasks. You need two tasks in the ruleflow: one for validation and one for eligibility.

To create a ruleflow:

1. In the Orchestrate part of the Rule Project Map, click **Add ruleflow**.

   Tip: You can also right-click the `my rule project/rules` folder in the Rule Explorer and click **New** > **Ruleflow**.

2. In the New Ruleflow wizard, make sure that the **Source folder** field is set to `/my rule project/rules`, and that the **Package** field is empty.
3. In the **Name** field, type `miniloan`.
4. Click **Finish**.

   The ruleflow editor opens, and enables you to construct the flow of tasks graphically. You specify how tasks are chained together: how, when, and under what conditions they will be executed.

5. Click  **Create a start node** and then click in the ruleflow editor.

   The ruleflow start node is displayed in the ruleflow editor.

6. Click  **Create an end node** and then click in the ruleflow editor.

   You now have a start node and an end node for your ruleflow.

Step 3: Define rule tasks

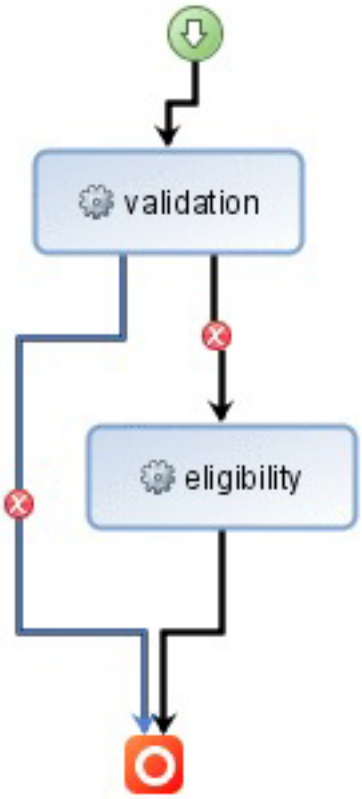
To define the ruleflow, you select the rule packages to include, and then create the transitions between them.

To define the rule tasks:

1. Drag the `validation` rule package from the Rule Explorer and drop it into the ruleflow editor.

   The `validation` rule package becomes a rule task in the ruleflow. By dropping this package into the ruleflow editor, any rule that you create in the package will be part of the execution, unless you specify otherwise.
2. Drag the `eligibility` rule package from the Rule Explorer and drop it into the ruleflow editor.
3. Click  **Create a transition** and create the following transitions (shown as arrows) by clicking the first item and then clicking the second item:
   a. The **start node** and the `validation` task.
   b. The `validation` task and the `eligibility` task.
   c. The `eligibility` task and the **end node**.
   d. The `validation` task and the **end node**.

   Some errors are displayed on the transitions to indicate that the conditions are missing:



4. Click  **Create a transition** to deselect the transition tool.
5. Click  **Layout All Nodes** to format the ruleflow diagram.
6. Save your work.

Step 4: Define the main transition

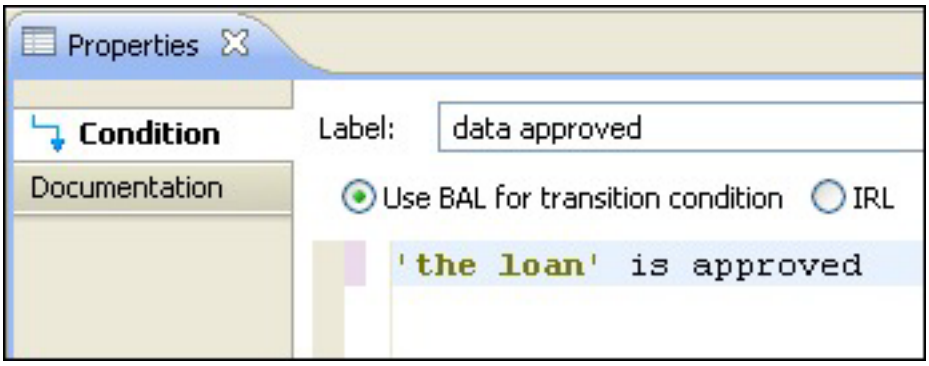
You can define conditions on transitions in the ruleflow. In this ruleflow, you set a transition condition so that rules in the eligibility package are run only when data is validated.

To define the main transition:

1. Click the transition from `validation` to `eligibility`.
   The Properties view opens and shows the condition for this transition.

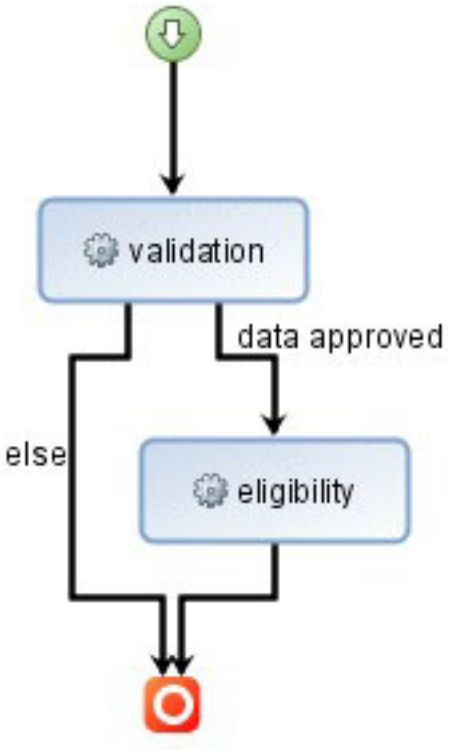
   Tip: If you cannot see the Properties view, click **Window** > **Show View** > **Properties** to open the Properties view.
2. In the **Label** field, type `data approved`.
3. Make sure **Use BAL for transition condition** is selected, to write the condition using the Business Action Language (BAL).
4. In the text area, type a space to display the Content Assist box, and double-click the items to form the following statement: `'the loan' is approved`.

   Tip: You can also type the statement directly in the text area.
   The Properties view should look like this:



The transition from `validation` to the **end node** is automatically set to `else`.

5. Save the changes.
   Your ruleflow should now look like this:



Step 5: Define the final action

You can also define a final action to display a message at the end of the execution.

To define the final action:

1. Click the **end node**.
   The Properties view lets you enter the final action.
2. In the **Final Action** section, make sure **Use BAL for action** is selected.
3. In the text area, type a space to display the Content Assist box, and enter the following final action:

```
print the approval status of 'the loan' ;
```

Note: You must add a semicolon (`;`) at the end of the line.

At execution time, this final action displays a message in the Console indicating the status of the loan at the end of rule execution.

4. In the ruleflow editor, click outside the diagram, and in the Properties view, make sure that the `main flow task` is set to `true`.
5. Save your work and close the ruleflow editor.

You have now defined the flow of execution. In the next task, you will write an action.

Task 3: Authoring rules

You can now author rules using the vocabulary created at the beginning of the tutorial.

When developing a rule project, the developer writes the initial rules, designs rule templates, and organizes the folders used to manage the rules. Later, the business user writes and edits these rules in a web environment, though a developer might be asked to write more complex rules.

In this task, you first create an action rule, and then import the rules already prepared for you. In Rule Designer, you use the completion menu to write the following action rule that is based on the vocabulary that you created:

```
if
  the amount of 'the loan' is more than 1,000,000
then
   add "The loan cannot exceed 1,000,000" to the messages of 'the loan';
   reject 'the loan' ;
```


This task should take you about 15 to 30 minutes to complete.

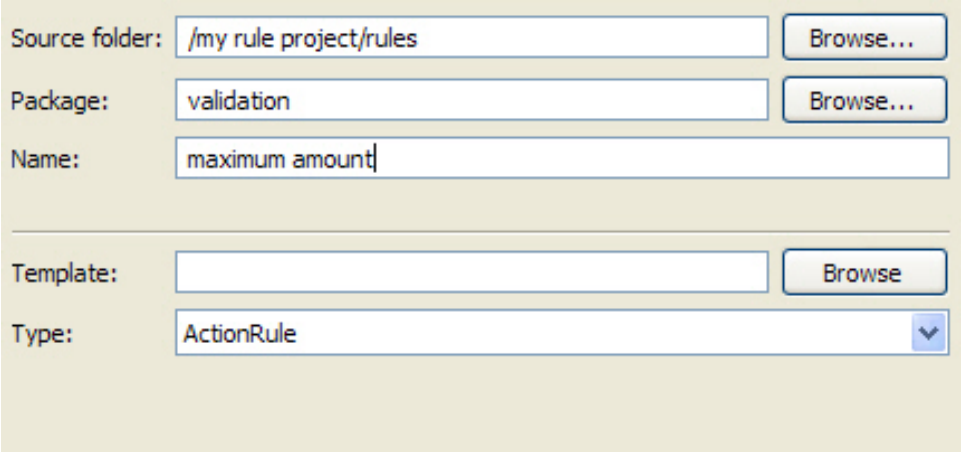
Step 1: Create an action rule

The first rule that you are going to write rejects the loan if the requested amount is greater than 1,000,000.

To create an action rule:

1. In Rule Designer, in the Author part of the Rule Project Map, click **Add action rule**.

   Tip: You can also right-click the `validation` package in the Rule Explorer and click **New** > **Action Rule**.
2. In the **Package** field, type `validation` (or click **Browse** to select it) and in the **Name** field, name the rule `maximum amount`.



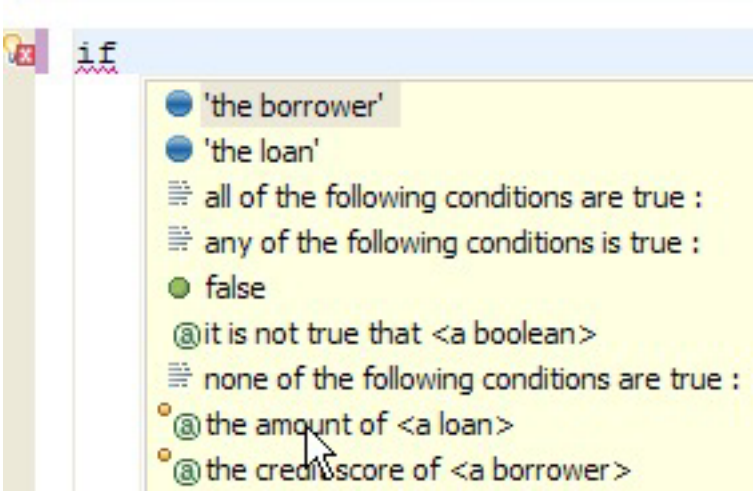
3. Click **Finish**.
   The Intellirule editor opens.

Step 2: Complete the action rule

You now use the code completion mechanisms of the Intellirule Editor to help you create the rule.

To complete the action rule:

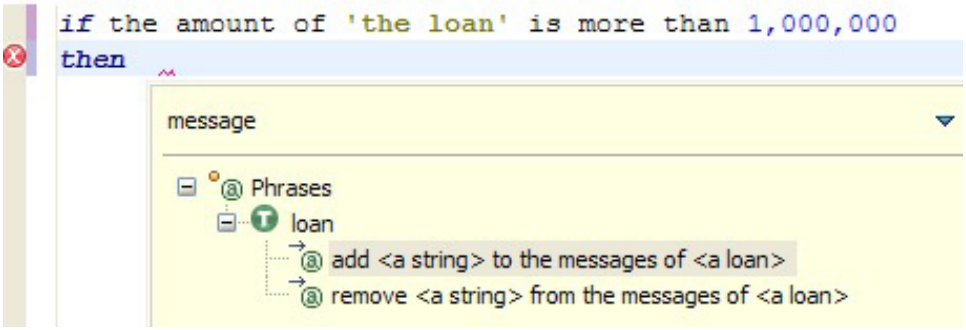
1. In the Intellirule editor, type `if`, and then press the space bar. The Content Assist box opens:



   Select terms and phrases from the Content Assist box to build the following expression:

   ```
   the amount of 'the loan' is more than 1,000,000
   ```

2. On the next line, type `then`, press the space bar, and then press **`Ctrl+Shift+space`** bar to activate the Tree Completer.
3. In the Tree Completer box, type `message` in the field at the top to display only terms and phrases about messages:



4. Double-click `add <a string> to the messages of <a loan>` to insert it, then use the Content Assist box to finish building the following expression:

   ```
   add "The loan cannot exceed 1,000,000" to the messages of 'the loan' ;
   ```

   Important:

   You must include a semicolon (`;`) at the end of the line.
5. Press **Esc** after entering the semicolon, if you are still in the Content Assist box.
6. Press **Enter** to create a new line, type a space, and write the following statement using the Content Assist box:

   ```
   reject 'the loan' ;
   ```
7. Press **Ctrl+Shift+F** to format the rule.

   Your rule should now be:

   ```
   if
     the amount of 'the loan' is more than 1,000,000
   then
      add "The loan cannot exceed 1,000,000" to the messages of 'the loan';
      reject 'the loan' ;
   ```
8. Save your work and close the Intellirule editor.

   For more information on using the rule editors, see Working with action rules.

Step 3: Import remaining rules

To complete the policy, you must add the rules that determine whether a borrower is eligible for a loan. For this tutorial, you import the eligibility rules into your rule project.

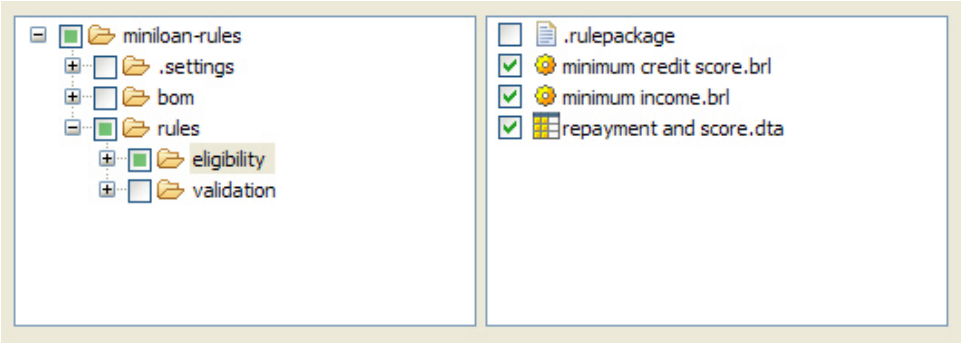
To import the remaining rules:

1. Click **File** > **Import**.
2. In the Import wizard, select **General** > **File System**, and click **Next**.
3. In the **From directory** field, click **Browse** and select `<InstallDir>/gettingstarted/DecisionServer/answer/miniloan-rules`, and then click **OK**.

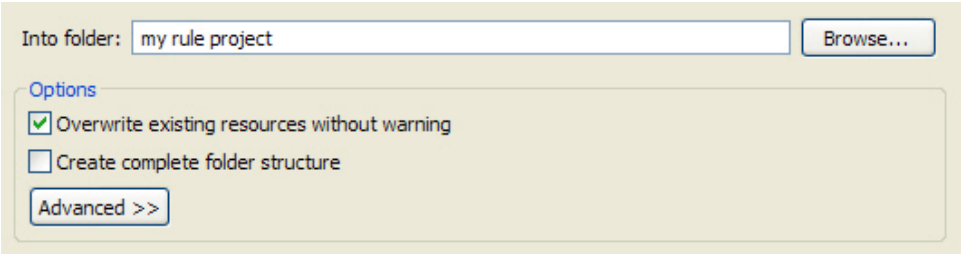
   Note:

   Ignore the message `There are no resources currently selected for import`. This is an Eclipse message prompting you to carry out the action you are about to perform.
4. The `miniloan-rules` folder opens in one of the panes in the Import wizard. Expand the folder to `miniloan-rules/rules` and select the check box beside `eligibility`.

5. Select the `eligibility` folder and make sure `eligibility` is highlighted, then clear the `.rulepackage` check box.



6. In the **Into folder** field, click **Browse**, select `my rule project`, and then click **OK**.
7. Select the **Overwrite existing resources without warning** option:



8. Click **Finish**.

   The `eligibility` rules are added to your project in the Rule Explorer.

Step 4: View the imported rules

To view the rules that you imported:

1. In the Rule Explorer, double-click the eligibility rules, and take a moment to look at them:

   minimum credit score

   Rejects the loan if the credit score is too low.

   minimum income

   Rejects the loan if the income is too low for the yearly repayments.

   repayment and score

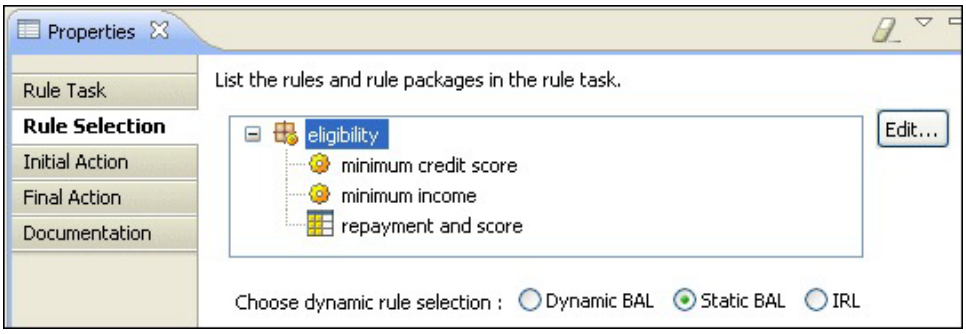
   In a decision table, rejects the loan based on various values of the debt-to-income ratio and credit score.

   You use decision tables to represent rules that share conditions and actions. Each row in a decision table represents a rule. By placing your cursor over the number of a row, you can view the text of the corresponding rule as hover help.

   Note:

   Rows with no actions display a message warning you about invalid or incomplete rows. You can ignore these warnings because the only purpose of these rows is to fill gaps in the table. These rows are ignored when you run the project.

2. Double-click the `rules/miniloan` ruleflow in the Rule Explorer to open it.
3. In the Ruleflow Editor, double-click the `eligibility` task.
4. In the Properties view, click the Rule Selection tab, and expand the `eligibility` package.



   By default, all the rules participate when the task is executed by the rule engine.
5. Close the Ruleflow Editor.

You have created rules and defined the flow to execute them. You have used ruleset parameters as data to be processed by the ruleset. In the next task, you create an Excel file in which you enter scenarios to test your rules.

Task 4: Testing rules

To validate rules and make sure that the changes have the effects that you want, you can test rules against scenarios. In this task, you create a scenario file to run tests on your rules.

Decision Validation Services (DVS) scenarios are use cases to validate the behavior of your rules. The scenarios and their expected results are stored in an Excel file called “scenario file”. The rules are executed against the scenarios and a report compares the expected results with the results that are obtained at execution time. The Excel scenario file contains two sheets:

Scenarios
: To enter the test data in the columns created from the input parameters.

Expected Results
: To define the results that you expect to get from the tests.

Business users can also create scenario files in Decision Center, but you must first prepare the rule project in Rule Designer.

To run the tests in Rule Designer or Decision Center, you must perform the following steps:

- Validate the rule project and create an Excel scenario file to check the correctness of the output.
- Enter test data in the Excel scenario files.
- Run the tests in Rule Designer to ensure that the scenario file works as expected.

 This task should take you about 20 to 30 minutes to complete.

Step 1: Select a constructor

The test data of the Excel scenario file is created from the ruleset input parameters. The BOM classes that make up the input parameters have constructors that define the columns in the Excel scenario file.

Before creating a scenario file, you must define the DVS constructor for the `Borrower` class. The DVS constructor that you select defines the mandatory columns.

In this tutorial, the name, credit score, and yearly income of the borrower are mandatory to test the rules.

To select a DVS constructor:

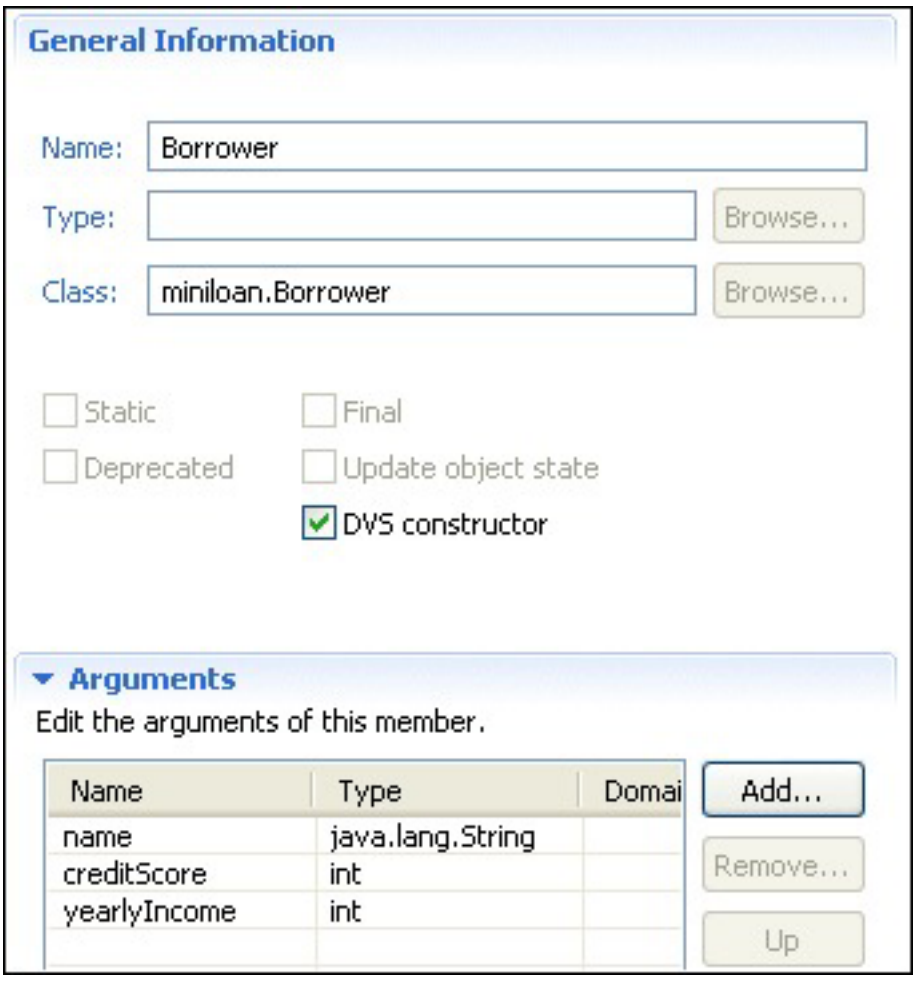
1. In the Rule Explorer, double-click the `Borrower` class: `my rule project > bom > miniloan > miniloan > Borrower`.

   The `Borrower` class opens in the BOM Editor. In the Members section, you can see that the `Borrower` class has two constructors:

   - `Borrower()`
   - `Borrower(String,int,int)`

   The `Borrower(String,int,int)` is the constructor that contains the arguments that correspond to the name, credit score, and yearly income of the borrower. These arguments are used to create the columns in the Excel scenario file.
2. Double-click the **Borrower(String,int,int)** constructor to edit it.
3. In the General Information section, select **DVS constructor**.

   This option specifies that the constructor must be used to create the columns of the Excel scenario file. The `Borrower(String,int,int)` constructor looks as follows in the BOM Editor:



4. Save the changes and close the BOM editor.

You have defined the constructor to create the mandatory input columns for the borrower in the Scenarios sheet.

Note:

For the purpose of this tutorial, the DVS constructor of the `Loan` class is already selected for you.

The miniloan XOM contains annotations that specify the names of the constructor arguments, and indicates that `Loan(int, int, double)` is the constructor to use for testing. To view the annotations used, take a look at `miniloan-xom/src/miniloan/Borrower.java` and `miniloan-xom/src/miniloan/Loan.java`. For more information on annotations, see Adding annotations to the XOM.

Step 2: Validate the project

Before you generate the Excel scenario file template, you must check that your project does not contain any errors or warnings that could prevent the generation of the Excel file.

To check your project:

1. In the Rule Explorer, select `my rule project`.

2. In the Rule Project Map, in the Design part, click **Check project for testing**.
   The DVS Project Validation view opens.
3. Make sure that there are no errors or warnings.
   If no errors or warnings are listed, the project is valid and you can create a scenario file.

Step 3: Create a scenario file

After you checked that your project does not contain any errors or warnings, you create an Excel scenario file template to validate the behavior of your rules.

To create the Excel scenario file:

1. Switch back to the Rule Project Map and in the Deploy and Integrate part, click **Create testing scenario file**.
2. In the **Rule Project** field, make sure that `my rule project` is selected, and then click **Next**.
3. Select **2003** as the Excel version to use.
4. Keep the **Default Excel Format** option.
5. Select **English (United States)** as the language to use in the Excel file.
6. In the **Excel Scenario File Name** field, change the name to `/my rule project/miniloan-test.xls`, and then click **Next**.
7. On the Expected Results page, expand `'the loan'`, and select **approved**.
   The `equals` operator is displayed next to **approved**. The scenarios will test if the loan is approved.
8. Click **Next**.
9. Leave the Expected Execution Details page empty, and click **Finish**.
   In the Rule Explorer, the `miniloan-test.xls` is displayed under `my rule project`.

   Tip: If `miniloan-test.xls` is not displayed under `my rule project`, right-click the project in the Rule Explorer, and click **Refresh**.

Step 4: Populate the Excel scenario file

To check that the rule project is valid and that the Excel scenario file is correct, you enter two simple scenarios that you test later in this task:

- **Scenario 1** shows the original data from the Miniloan web application. The debt-to-income ratio is too high and the loan is rejected.
- **Scenario 2** shows that the amount of the loan is lower than in Scenario 1. The expected result is that the loan should be approved.

To populate the Excel scenario file:

1. Outside of Eclipse, navigate to *<MyEclipseWorkspace>*`/my rule project` and open `miniloan-test.xls`.
   <MyEclipseWorkspace> refers to your workspace directory on the file system.
   If you use the Excel editor embedded in Eclipse, you might encounter difficulties to save the file.

   Tip: You can also right-click the file in the Rule Explorer, and click **Open With** > **System Editor**.
2. Complete the **Scenarios** sheet of the Excel scenario file template as follows:

Tip:

- To add rows, copy and paste the first one. Remember to change the name of the scenario that you have pasted.
- In `<InstallDir>/gettingstarted/DecisionServer/answer/miniloan-rules/`, the `miniloan-test.xls` file is already completed with the following information.

Table 1. Scenarios sheet

		the borrower			the loan		
Scenario ID	description	**name**	**credit score**	**yearly income**	**amount**	**duration**	**yearly interest**
Scenario 1		Joe	600	80000	500000	240	0.05
Scenario 2		Joe	600	80000	**250000**	240	0.05

The amount of the loan is different between scenario 1 and scenario 2.

3. Complete the **Expected Results** sheet as follows:

Table 2. Expected Results sheet

Scenario ID	the loan is approved equals
Scenario 1	FALSE
Scenario 2	TRUE

4. Save and close the file.
5. In Rule Designer, right-click `my rule project` in the Rule Explorer and click **Refresh** to update the file.

Step 5: Test the Excel scenario file locally

To test that the scenario works as expected, you run the Excel scenario file locally in Rule Designer.

To test the Excel scenario file locally:

1. On the **Run** menu, click **Run Configurations**.
2. Create the configuration:
   a. In the side pane of the Run Configurations dialog, right-click **DVS Excel File**, and click **New**.
   b. In the **Name** field, enter `Miniloan Test` as the name for the launch configuration.
   c. In the **Excel File** field, click **Browse**, select `my rule project/miniloan-test.xls`, and click **OK**.
   d. In the **Rule Project** field, click **Browse**, select `my rule project`, and click **OK**.
   e. In the **HTML Report** field, click **Browse**, select `my rule project`, and click **OK**.
   f. Click the **DVS Configuration** tab, and make sure that the **Local execution** option is selected.
3. Click **Apply**, and then **Run**.

   If the Console view does not open automatically, click **Window** > **Show View** > **Other**, then select **General** > **Console**, and click **OK**.

   The Console view shows a log of the build process, and the following result:

```
--- Output for scenario 'Scenario 1' :
false [Too big Debt-To-Income ratio]

--- Output for scenario 'Scenario 2' :
true []

Execution finished
Now starting generation of DVS HTML report
Finished generating DVS HTML report
```

4. In the Rule Explorer, right-click `my rule project` and click **Refresh**.

   The `report.html` file is displayed under the project.

5. In the Rule Explorer, right-click `report.html`, and click **Open With** > **Web Browser**.

   The report opens and shows the results of the tests: the execution results are the same as the expected results. The tests are successful.

Details for all Scenarios

Name	Success Rate	Tests	Failures	Errors	Message
Scenario 1	100 %	1	0	0	
Scenario 2	100 %	1	0	0	

   In Scenario 1, the loan is rejected, as specified in the expected results (the loan is approved equals: FALSE).

   In Scenario 2, the loan is approved, as specified in the expected results (the loan is approved equals: TRUE).

6. Close the report.

Now that you have made some changes to the BOM, in the next task you will see how to debug the rule project using the Excel file.

Task 5: Debugging

In this task, you debug the ruleset, by using the test data from the Excel scenario file that you created in the previous task.

Before you deploy the rules to the execution environment and integrate your work into the Miniloan application, you want to make sure that there are no defects and that the rules execute as expected. In Rule Designer, you can execute rules in a sandbox for testing and debugging purposes.

This task should take you about 15 minutes to complete.

Step 1: Inserting a breakpoint

You can use an Excel scenario file to debug the execution of the rules. Debugging in Rule Designer allows you to step into both rule code and Java code. You insert breakpoints into rules to stop the execution at the location where the breakpoint is set.

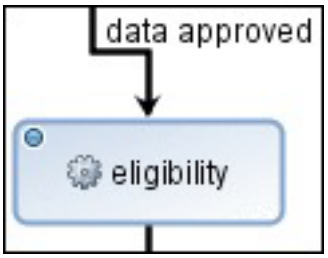
To set a breakpoint:

1. Make sure that the Console view is opened.

Tip: To open the Console view, on the **Window** menu, click **Show View** > **Other**. In the Show View dialog, select **General** > **Console** and then click **OK**.

2. In the Rule Explorer, double-click the `rules/miniloan` ruleflow to display it in the Ruleflow Editor.
3. Select and right-click the `eligibility` task in the ruleflow diagram, and click **Toggle Breakpoint**.

   A breakpoint marker is displayed next to the `eligibility` task.



Step 2: Debugging rule execution

Now that you have inserted a breakpoint, you can start debugging the execution.

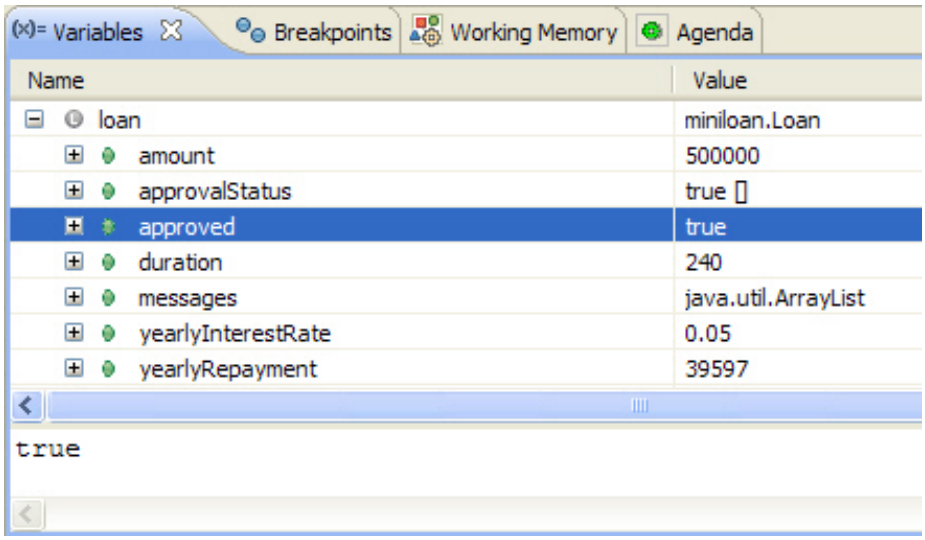
To debug the execution:

1. Start the debugger:
   a. Click **Run** > **Debug Configurations**.
   b. In the side pane of the Debug Configurations dialog, select **DVS Excel File** > **Miniloan Test**.

      This is the launch configuration that you created in the previous task. You reuse it to debug rule execution.
   c. Click **Debug**.

      A dialog opens, asking you if you want to change to the Debug perspective. Click **Yes**. The Debug perspective opens. The debugging commands are available from the Debug view, or from the **Run** menu.

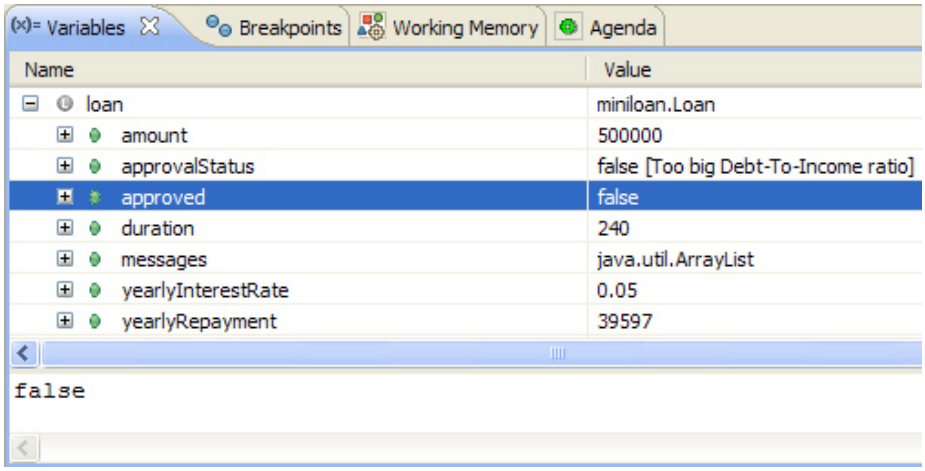
      Debugging stops at the beginning of the `eligibility` task, where you inserted the breakpoint.
2. Step through the rule code:
   a. Click  **Step Into**. Debugging stops at the first action of the `minimum income` rule.
   b. In the Variables view, expand the `loan` object. The value of the `approved` attribute is `true`.



c. Click  **Step Over** to go to the next action statement in the `minimum income` rule. This action calls a method that rejects the loan.

d. Click  **Step Into**. You can now see the Java code of the `reject` method that is called from the rule. This method is defined in the Java XOM project.

e. Click  **Step Return** to return to the rule. In the Variables view, the `approved` attribute of the `loan` object is now `false`, which means that the loan is rejected.



3. Click  **Resume** to complete the execution of Scenario 1.
   The execution of Scenario 2 starts and is stopped at the beginning of the `eligibility` task.

4. Click  **Resume** to complete the execution of Scenario 2.
   When execution terminates, the Console view shows the message:

```
--- Output for scenario 'Scenario 1' :
false [Too big Debt-To-Income ratio]

--- Output for scenario 'Scenario 2' :
true []
```

```
Execution finished
Now starting generation of DVS HTML report
Finished generating DVS HTML report
```

5. On the **Window** menu, click **Open Perspective** > **Other** > **Rule** to return to the Rule perspective.

   Alternatively, the side of the Eclipse toolbar displays the current perspective name and buttons to quickly access the other available perspectives.

6. Close the ruleflow.

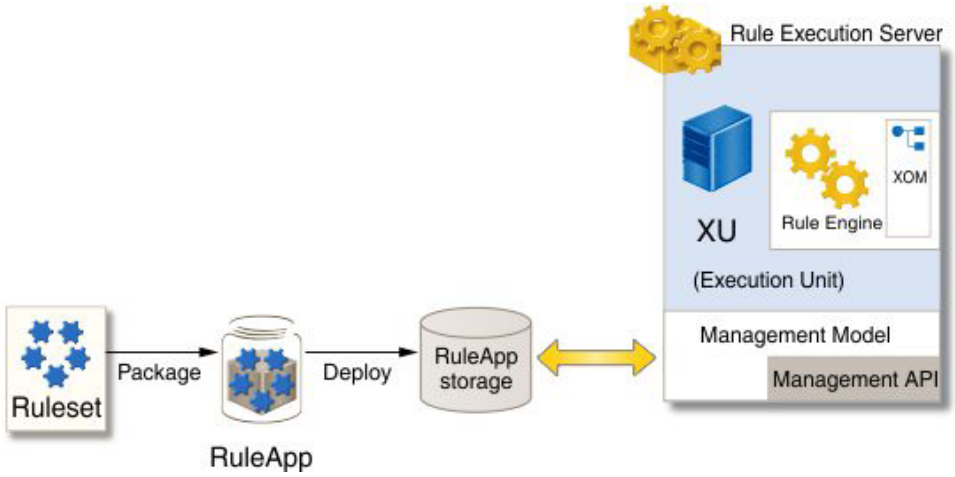
You have created a scenario file to test your rules, and you used it for debugging. Now, you can deploy your rules to Rule Execution Server, which is the runtime execution environment that provides a rule engine for your application.

Task 6: Deploying rules

In this task, you deploy your ruleset to Rule Execution Server. Rule Execution Server is the runtime environment that contains the rule engine to execute the rules.

Up to this point, you have written some rules by using vocabulary that a business user can understand, and associated these rules with a rule flow. Now, you integrate your work into the Miniloan application in two stages:

1. Deploy a RuleApp to Rule Execution Server. The RuleApp is the format expected by Rule Execution Server. It contains the ruleset. In the same way that Java classes are packaged in a JAR file, a ruleset is packaged in a JAR file and contains all that is necessary for the execution (rules, ruleflow, and so on).
2. Call the integration code in the miniloan web application so that the business logic you implemented and validated in previous tasks is now executed in Rule Execution Server.



In this task, you can also test your deployed ruleset using a hosted transparent decision service (HTDS). After deploying the RuleApp in Rule Execution Server, you can generate a WSDL file from your ruleset, and then test it in Rule Designer.

This task should take you about 15 to 30 minutes to complete.

Step 1: Deploy from Rule Designer

To deploy the rules from Rule Designer, you must first create a RuleApp project. Then, you set some properties to enable the monitoring of the ruleset execution that you will do in the next task.

To create a RuleApp project and deploy the RuleApp:

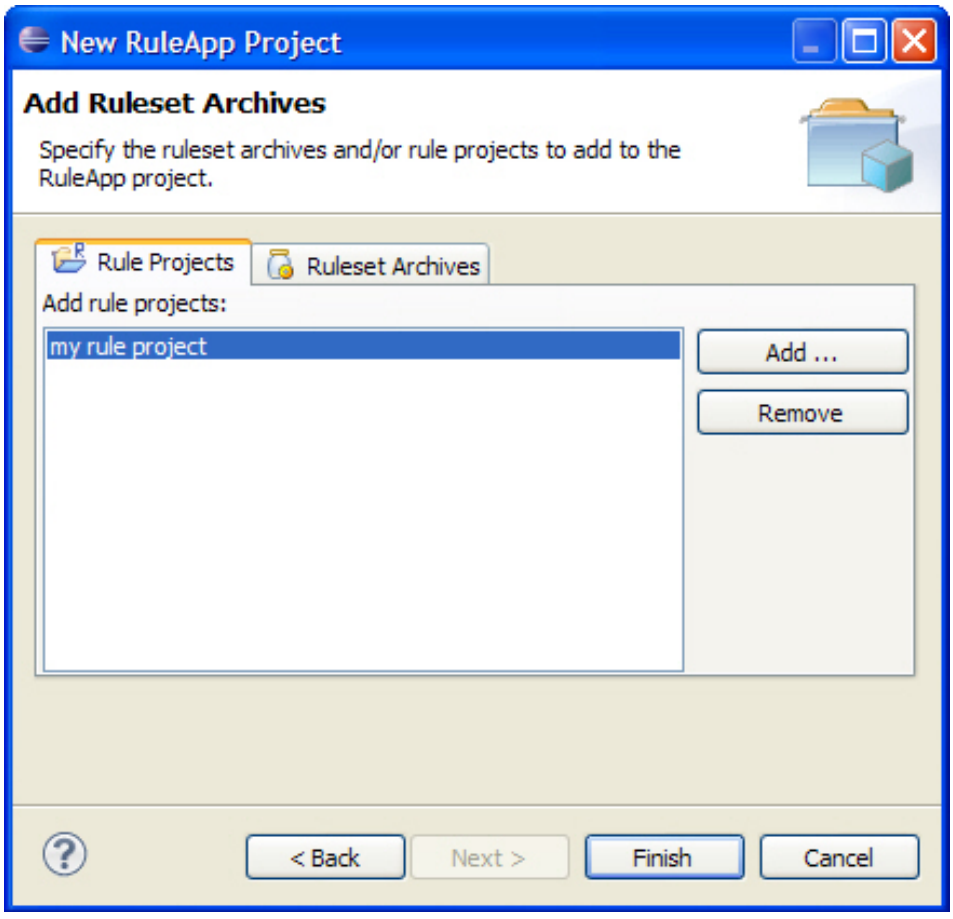
1. Make sure that the sample server is started.

   If you have stopped the sample server, click **Start** > **All Programs** > **IBM** > ***package_group*** > **Sample Server** > **Start server**. *package_group* refers to the package group specified in Installation Manager during installation. The default package group is Operational Decision Manager V8.5.1.
2. In the Deploy and Integrate part of the Rule Project Map, click **Create RuleApp project**.

   Tip: You can also use the **File** menu, then click **New** > **Project** and select **RuleApp Project**.
3. In the New RuleApp Project wizard, in the **Project name** field, type `my ruleapp`.
4. Click **Next**.

   The `my rule project` is displayed on the Add Ruleset Archives page.

   Tip: If you cannot see your rule project, click **Add**, select `my rule project`, and click **OK**.

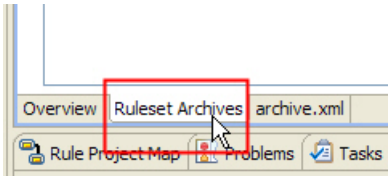


5. Click **Finish**.

You created a RuleApp project that contains a ruleset archive generated from the project `my rule project`.

`my ruleapp` is displayed in the Rule Explorer and the RuleApp editor opens to let you deploy the RuleApp to Rule Execution Server.

6. Click the **Ruleset Archives** tab in the editor.



7. Under Ruleset Archives, select the `myruleproject` archive.
8. Add the ruleset property to enable ruleset monitoring:
   a. In the Ruleset Properties section, make sure that `ruleset.bom.enabled` is absent or set to `true`.
   b. Click **New** to create a new property for the ruleset.
   c. In the Edit Property dialog, select the predefined property `monitoring.enabled` from the list, type `true` in the **Value** field, and then click **OK**.
   d. Save the changes.
9. Click the **Overview** tab, and click **Deploy** in the Deployment section.
10. In the Deploy RuleApp Archive wizard, keep **Increment RuleApp major version** selected, and click **Next**.

    Note: A warning is displayed if you are using the default Eclipse with JDK 7. If your application server runs with JDK 6, you must modify the settings in Eclipse to use JDK 6.
11. On the next wizard page, make sure that **Create a temporary Rule Execution Server configuration** is selected, and type the following configuration details:

    URL: `http://localhost:<PORT>/res`

    Important: Enter the correct port number in the URL. For more information, see Checking the server port number.

    Login: `resAdmin`

    Password: `resAdmin`
12. Click **Finish**.

    The Console displays a message indicating that the 1.0 version of the RuleApp has been deployed.
13. Close the RuleApp editor.

Step 2: View the deployed RuleApp

You now view the deployed RuleApp in Rule Execution Server, which is an execution environment for rules (Java SE and Java EE) that interacts with the rule engine. Rule Execution Server handles the management, performance, security, and logging capabilities associated with the execution of your rules.

From your application, you access Rule Execution Server using either web services, EJBs, or, in this case, pure Java objects (POJO).

To view the deployed RuleApp:

1. Click **Start** > **All Programs** > **IBM** > *package_group* > **Sample Server** > **Rule Execution Server Console**.

   package_group refers to the package group specified in IBM Installation Manager during installation. The default package group is Operational Decision Manager V8.5.1.

   Tip: You can also enter `http://localhost:<PORT>/res` in a browser. Enter the correct port number for the URL.
2. Sign in to the Rule Execution Server console using the following details:

   Username: `resAdmin`

   Password: `resAdmin`
3. Click the **Explorer** tab.
4. In the Navigator, expand **RuleApps**, and then `/myruleapp/1.0`.

   You see that Rule Execution Server contains your 1.0 version of `myruleapp`, which contains the 1.0 version of the ruleset as expected:

5. Click `/myruleproject/1.0` to view the details of the ruleset in the Ruleset View.

   Note that the status of the ruleset is **enabled**, indicating that it can be executed.

6. Click the **Show Properties** link to view the ruleset properties.

   The property that you added in the previous step is set to `true`.

Step 3: Run the Miniloan web application with rules

The Miniloan web application is designed to let you choose whether the business logic is embedded in the application as pure Java code or is coded in a ruleset. When you first started the Miniloan application, you had not yet created the ruleset, so you ran the application using the Java code. Now that you have created the ruleset and deployed it to Rule Execution Server, you can select the **Use Rules** check box. The application now calls for the execution of the ruleset.

The validation is done using the `validateWithJRules` method of the Miniloan bean.

To call the integration code in the Miniloan application:

1. Open a new browser window, and enter the following URL with the correct port number:

   `http://localhost:<PORT>/miniloan-server`
2. Select the **Use Rules** check box.

This check box activates the code that calls the execution of the ruleset. From this point, when you click **Validate Loan**, the miniloan web application executes the rules that you deployed to Rule Execution Server. The behavior of Miniloan is the same, except that the business logic is no longer part of the application code.

Note that some fields have been temporarily added under Ruleset Information to let you see, as a developer, how the rules are being executed.

3. Click **Validate Loan**. The results of the validation are as before:

```
The loan is rejected
Messages:
Too big Debt-To-Income ratio
```

The Rules Execution Summary shows that the `eligibility.minimum income` rule was executed in the rule task `miniloan#eligibility`.

A rule is executed when the conditions of the rule are met and the actions of the rule are triggered. This rule sets the `approved` status of the loan to `false`.

4. Change the amount to 300000 and click **Validate Loan** again.

This time you should see the following response:

```
The loan is approved
The yearly repayment is 23758
```

The loan is approved, no rule has been fired.

5. Close the Miniloan web application. You will open it again later to monitor the application.

(Optional) Step 4: Retrieve the HTDS WSDL file

A hosted transparent decision service (HTDS) is a web service that provides an interface to access a deployed ruleset. The Decision Service component passes input parameters to the rule engine and accesses the return values. The transparent decision service support includes traceability from decision services to rules, runtime monitoring and version management.

You can retrieve the WSDL file for the `myruleproject` ruleset, from the Rule Execution Server console.

To retrieve the WSDL file:

1. In the Rule Execution Server console, make sure that you are still on the `myruleproject` ruleset page, and click **Retrieve HTDS Description File** in the toolbar at the top.
2. Keep the **SOAP** option selected, and then select **Latest ruleset version** and **Latest RuleApp version**, and click **Download**.
3. Save the WSDL file to *<MyEclipseWorkspace>*`/my rule project` and rename it to `MyDecisionService.wsdl`.

<MyEclipseWorkspace> refers to your workspace directory on the file system.

Tip: To import the WSDL file into Rule Designer, you can also use the Import wizard:

a. On the **File** menu, click **Import**.
b. Click **General** > **File System**, and then click **Next**.
c. Browse to the folder where you saved the WSDL file, and select the WSDL file to import.
d. In the **Into folder** field, select `my rule project`, and then click **Finish**.

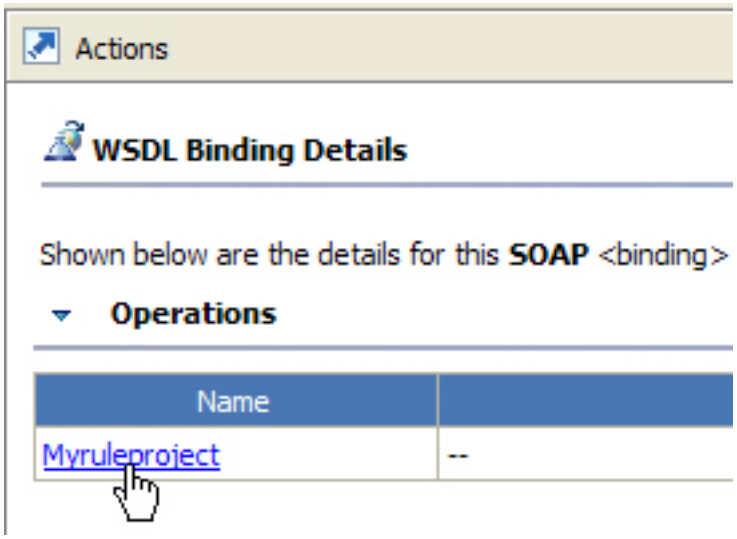
4. Sign out of the Rule Execution Server console.

(Optional) Step 5: Test the HTDS in Rule Designer

In Rule Designer, you use the WSDL file that you retrieved from Rule Execution Server and saved to your workspace, to test the web service. In the test environment, you enter some test data and call the WSDL operation.

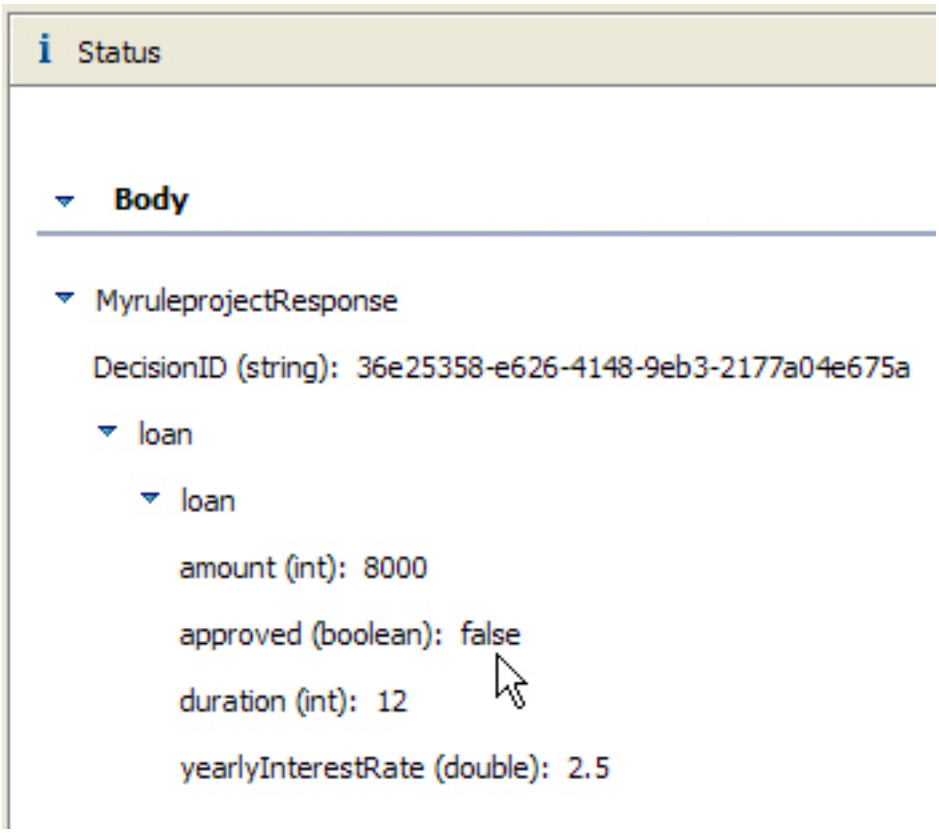
To test the web service:

1. In the Rule Explorer, right-click `my rule project`, and click **Refresh**.
   The `MyDecisionService.wsdl` file is displayed in the rule project.
2. Double-click the `MyDecisionService.wsdl` to open it in the WSDL editor.
   There is one operation (`MyRuleProject`) of request-response type with an input message, an output message, and a SOAP fault.
3. Close the `MyDecisionService.wsdl` file.
4. Switch to the **Java** perspective:
   a. On the **Window** menu, click **Open Perspective** > **Other** > **Java**.
   b. Click **OK**.
5. In the Package Explorer, expand `my rule project`.
6. Right-click `MyDecisionService.wsdl`, and click **Web Services** > **Test with Web Services Explorer**.
7. In the Web Services Explorer, under Operations, click **Myruleproject**.



8. Enter the following values in the empty fields for the `borrower`:
   - **creditScore**: `100`
   - **yearlyIncome**: `70000`
9. Enter the following values in the empty fields for the `loan`:
   - **amount**:`8000`
   - **duration**:`12`
   - **yearlyInterestRate**:`2.5`

   The credit score is below the minimum limit so the loan should be rejected.
10. Click **Go** to call the web service on Rule Execution Server.
11. In the Status section, take a look at the response.
    The loan was not approved.



12. Close the Web Services Explorer, and switch back to the **Rule** perspective.

You have now deployed your ruleset to Rule Execution Server and tested the result in the Miniloan web application, and as a web service. In the next task, you use Rule Execution Server to monitor and audit the execution of the rules.

Task 7: Monitoring

In this task you learn how to monitor ruleset execution using Rule Execution Server Console. You also use Decision Warehouse to audit and view stored decision traces.

As an IT professional responsible for computer applications within your company, you must ensure that all rules-enabled applications are functioning correctly. In addition to providing an environment to manage the execution of your rules, Rule Execution Server lets you monitor the execution of your rulesets.

Auditors can analyze the execution performance of your rulesets, and troubleshoot any problematic transaction that may be reported. To identify the problem when a transaction fails, auditors and analysts need to know the business policies that were applied, and the transactional data that was used at execution time. Decision Warehouse is available from Rule Execution Server and stores ruleset execution traces that can be used for auditing purposes.

This task should take you about 10 to 15 minutes to complete.

Step 1: Run Rule Execution Server diagnostics

To help you identify errors with the execution environment, for example, network failures that make a database unavailable, Rule Execution Server provides diagnostic capabilities.

To run the Rule Execution Server diagnostics:

1. Open the Rule Execution Server console.

   Tip:

   - Click **Start** > **All Programs** > **IBM** > *package_group* > **Sample Server** > **Rule Execution Server Console**. *package_group* refers to the package group specified in IBM Installation Manager during installation. The default package group is Operational Decision Manager V8.5.1.
   - You can also enter `http://localhost:<PORT>/res` in a browser, with the correct port number.

2. Sign in to Rule Execution Server using the following details:

   Username: `resAdmin`

   Password: `resAdmin`

3. Click the **Diagnostics** tab, and then click **Run Diagnostics**.

   The green check marks indicate that rule execution was successful.

4. Click **Expand All** to show the details of each test.

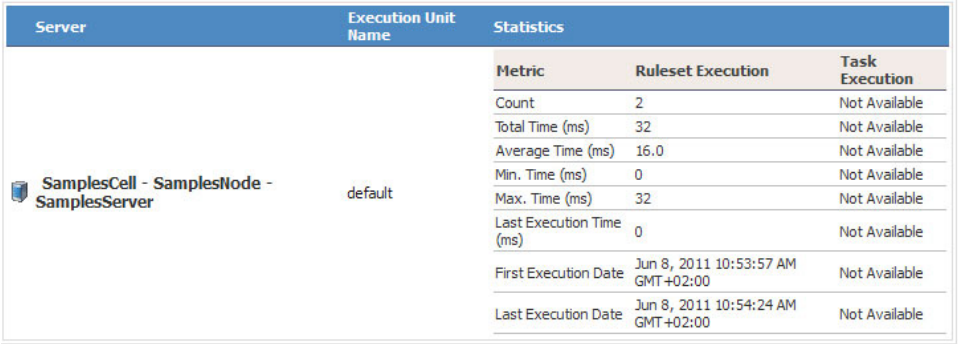
   The diagnostics test various aspects relating to the execution environment: connection, resource adapter information, creation of the RuleApp and ruleset, execution and update of the ruleset, and removal of the RuleApp and ruleset.

Step 2: View statistics on deployed RuleApps

When the execution environment is functioning correctly but performance problems are being reported, the Rule Execution Server console lets you obtain statistics on the execution of your rules.

To view statistics on deployed RuleApps:

1. In Rule Execution Server, click the **Explorer** tab.
2. Under Navigator, expand **RuleApps** and select the ruleset `/myruleapp/1.0/myruleproject/1.0`.
3. Click **View Statistics** in the Ruleset View toolbar to see ruleset execution statistics such as how many times the ruleset has been executed, and runtime statistics such as `Average Time` and `Maximum Time`.



Server	Execution Unit Name	Statistics		
		Metric	Ruleset Execution	Task Execution
SamplesCell - SamplesNode - SamplesServer	default	Count	2	Not Available
		Total Time (ms)	32	Not Available
		Average Time (ms)	16.0	Not Available
		Min. Time (ms)	0	Not Available
		Max. Time (ms)	32	Not Available
		Last Execution Time (ms)	0	Not Available
		First Execution Date	Jun 8, 2011 10:53:57 AM GMT+02:00	Not Available
		Last Execution Date	Jun 8, 2011 10:54:24 AM GMT+02:00	Not Available

4. In a separate browser window, open the Miniloan web application (`http://localhost:<PORT>/miniloan-server`).

5. Make sure that the **Use Rules** check box is selected, and click **Validate Loan**.
6. Switch back to the Rule Execution Server console and click **Refresh**.

   In the statistics, the count of rule executions has increased by the number of times you validated the loan.

Step 3: Execute a transaction in the Miniloan application

In the previous task, you added the `monitoring.enabled` ruleset property to keep a trace of the decision history. Every transaction you simulate for the ruleset is now stored and logged in Decision Warehouse.

To simulate a transaction in the Miniloan application:

1. To start the Miniloan application, enter the following URL with the correct port number in a browser:

   `http://localhost:<PORT>/miniloan-server`
2. Change the amount of the loan to `2000000`.
3. Make sure that the **Use Rules** check box is selected, and click **Validate Loan**.

   The loan is rejected.

Step 4: Search for past transactions in Decision Warehouse

You search for past transactions and decision traces in Decision Warehouse, to find the decision that led to the failed transaction.

To search for past transactions:

1. In the Rule Execution Server console, click the **Decision Warehouse** tab.
2. On the Search Decisions page, leave the fields blank, and click **Search**.

   Decision Warehouse displays the decisions for the transactions that you have executed in the Miniloan application. For example, the decision for the transaction executed in "Step 3: Execute a transaction in the Miniloan application," shows the date and the processing time, and indicates that one rule was executed.

Decision ID	Date	Ruleset Version	Number of rules fired	Decision Trace	Processing Time (ms)
024167a8-3032-4c43-b704-4c4edf354b64	2012-04-05 18:58:49	/myruleapp /1.0/myruleproject/1.0	1	View Decision details	15

Step 5: View the rules executed

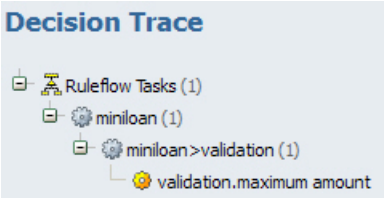
You can use Decision Warehouse to understand why the loan has been rejected. You check the execution details for a decision and view the rules that were executed.

To view the execution details for a decision:

1. In the table that lists the decisions found, in the Decision Trace column, click **View Decision details** for the decision where one rule was executed.

   The decision trace details open in a new window.
2. In the Decision Trace section, expand **Ruleflow Tasks** > **miniloan** > **miniloan** > **validation**.

   The decision trace shows that the rule `validation.maximum amount` was executed.



The loan was rejected because the amount entered in "Step 3: Execute a transaction in the Miniloan application" on page 36 exceeds the maximum amount of 1000000.

3. In the Input Parameters section, take a look at the input parameters listed.

   You can see that the input parameter for the amount of the loan is 2000000:

   ```
   <amount>
   <int>2000000</int>
   </amount>
   ```

4. Close the decision trace window, and sign out of Rule Execution Server.

In the next task, you publish the rule project to Decision Center so that the business rules become accessible to the business user in a shared environment.

Task 8: Publishing to Decision Center

You now publish the rule project to Decision Center.

In this task, you make the rule project you developed in Rule Designer available to business users in Decision Center. Decision Center is a web-based environment that allows business users to view, create, and modify rules. From Rule Designer you publish your rule project to Decision Center, and then periodically synchronize the work of the business users with your Rule Designer copy.

Important: This task is optional. To perform this task, you must have Decision Center installed.

 This task should take you about 10 to 15 minutes to complete.

Step 1: Publish the rule project to Decision Center

Note: Start the sample server, as described in "Starting the Miniloan web application" on page 3, if you have stopped it.

In order to make your rule project available to business users, you need to connect Rule Designer to Decision Center and then publish to it.

To publish the rule project to Decision Center:

1. In Rule Designer, right-click `my rule project`, and click **Decision Center** > **Connect**.
2. Complete the Decision Center Configuration dialog as follows.

   URL: `http://localhost:<PORT>/teamserver`

   Enter the correct port number in the URL. For more information, see Checking the server port number.

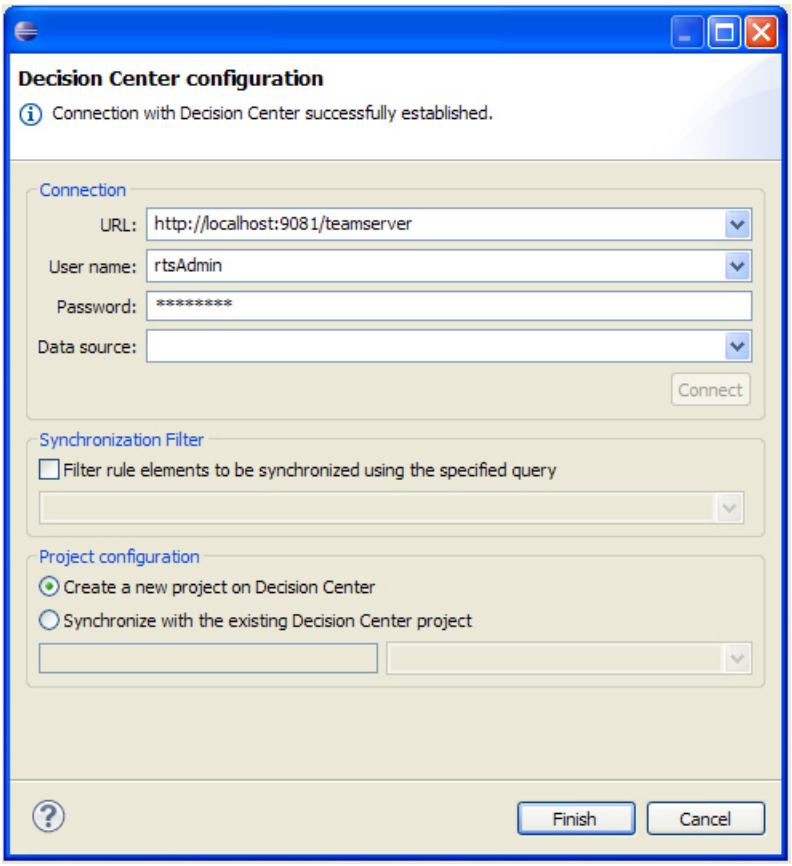
   User name: `rtsAdmin`

Password: `rtsAdmin`

3. Click **Connect**.

   When the connection is established, the message `Connection with Decision Center successfully established` is displayed, and the Project configuration area becomes active.

4. In the Project configuration area, make sure that **Create a new project on Decision Center** is selected, and then click **Finish**.



5. The Synchronize Complete dialog opens when the publishing process is complete. Click **OK** to close this dialog.
6. A dialog opens asking you if you want to change to Team Synchronizing perspective. Click **Yes**.

   An empty Synchronize view opens, indicating that there are no changes in the project. This means that your rules are now published to Decision Center.

Step 2: Explore the rule project in Decision Center

Now that you have published the rule project, you can open Decision Center and see your rules in the business user environment.

To explore the rule project in Decision Center:

1. From the **Start** menu, click **All Programs** > **IBM** > *package_group* > **Sample Server** > **Decision Center Enterprise Console**.

   package_group refers to the package group specified in IBM Installation Manager during installation. The default package group is Operational Decision Manager V8.5.1.

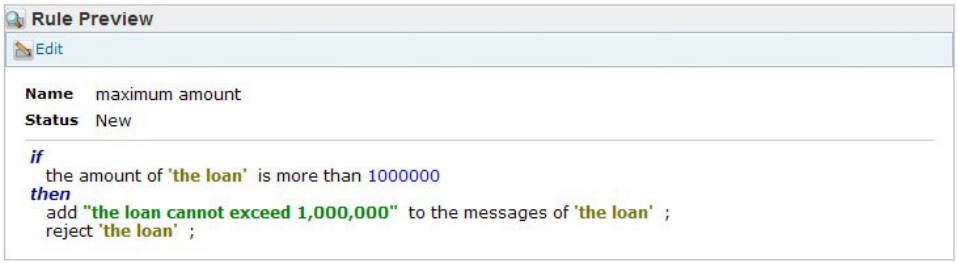
Tip: You can also enter the following URL with the correct number in a browser: `http://localhost:<PORT>/teamserver/`.

2. Sign in to Decision Center using the following details:

   Username: `rtsUser1`

   Password: `rtsUser1`

   Decision Center can handle different user profiles. Here you sign in as a regular business user.
3. On the Decision Center **Home** tab, in the **Project in use** field, select `my rule project`.
4. Click the **Explore** tab.
5. Under **Business Rules**, click the `validation` folder.
6. Preview the content of the `maximum amount` rule by clicking  **Preview** beside the name of the rule in the table.



7. Click the `eligibility` folder, and then click  Preview next to the `repayment and score` decision table to preview its content.
8. Click **Ruleflows**, and then click  Preview next to the `miniloan` ruleflow to preview its content.
9. Sign out of Decision Center.

Your business logic is now available to business users. You have a rules-enabled application, where the business logic is in the hands of the business user and the execution is monitored by IT. For more information on the tasks that you performed in the tutorial, see “Summary.”

Summary

You have completed the getting started tutorial, and you have discovered the Decision Server modules and how they interact.

During this tutorial you became familiar with the following modules:

- Rule Designer to design and develop the business rule application.
- Rule Execution Server to execute and monitor the business logic.

You also learned how to publish the rule project to Decision Center to make the rules available to business users.

We hope that this tutorial helped you understand how you can use Decision Server to externalize the business logic from your own application and place it in the hands of the business users.

If you want to know more about what you did in this tutorial, you can find useful pointers in the following table. For each task in this tutorial, the following table provides links to related information in the rest of the documentation.

Tasks	Related information	Related tutorials
"Task 1: Designing the rule project" on page 5	Designing business object models Developing rule projects	Tutorial: Defining a vocabulary
"Task 2: Orchestrating" on page 11	Orchestrating ruleset execution	Tutorial: Creating your first ruleflow
"Task 3: Authoring rules" on page 16	Authoring business rules	Tutorial: Creating action rules Tutorial: Editing decision tables
"Task 4: Testing rules" on page 20	Testing and simulating rulesets	Tutorial: Configuring the BOM for Excel testing
"Task 5: Debugging" on page 25	Executing rules using the rule engine	Tutorial: Debugging an Excel scenario file Tutorial: Debugging a ruleset
"Task 6: Deploying rules" on page 28	Deploying and exporting RuleApps	Tutorial: Managing RuleApps
"Task 7: Monitoring" on page 34	Monitoring and managing the server Monitoring and managing Decision Warehouse Monitoring ruleset execution	Tutorial: Executing a hosted transparent decision service on Java or .NET
"Task 8: Publishing to Decision Center" on page 37	Synchronizing from Designer	

Restarting the tutorial

If you want to perform this tutorial again, switch to a new and empty workspace, and see Restoring the sample databases.

Stopping the sample server

To stop the sample server, click **All Programs** > **IBM** > *package_group* > **Sample Server** > **Stop server**.

package_group is the package group specified in IBM Installation Manager during installation. The default package group is Operational Decision Manager.

Tutorial: Getting started with business rules

This tutorial helps you to take your first steps with Operational Decision Manager V8.5.1. In this tutorial, you learn how to use Rule Designer to create and run a rule-based application, and Rule Execution Server to execute the rules. If you are performing this tutorial for the first time, read the related sections as well.

Learning objectives

In this tutorial, you learn how to:

- Design a rule project.
- Orchestrate the rules and define a flow of execution.
- Write business rules, and then test and debug the rules.
- Deploy the rules to the execution environment.
- Monitor and audit the rules.
- Publish the rule project to the web-based business environment.

Time required

 This tutorial takes between 3 and 4 hours to complete.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM United Kingdom Laboratories,
Mail Point 151,
Hursley Park,
Winchester,
Hampshire,
England SO21 2JN

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those

websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM United Kingdom Laboratories,
Mail Point 151,
Hursley Park,
Winchester,
Hampshire,
England SO21 2JN

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or

imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Index

Printed in USA