IBM Operational Decision Manager
Version 8 Release 6

Getting Started with Business Rules

Note

Before using this information and the product it supports, read the information in “Notices” on page 41.

This edition applies to version 8, release 6, modification 0 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 is an introduction to some of the features of Operational Decision Manager V8.7 and includes exercises to help you become familiar with creating, running, and executing rules. In this tutorial, you learn how to create and run a rule-based application in Rule Designer, and how to execute rules in Rule Execution Server. If you are doing this tutorial for the first time, ensure that you follow the links to read the related sections.

Learning objectives

By completing the tutorial, you learn about the following aspects of Operational Decision Manager:

- Designing a rule project.
- Orchestrating the sequence in which rules are processed.
- Defining a flow of execution.
- Writing business rules, and then test and debug the rules.
- Deploying rules to the environment in which the rules are run.
- Monitoring and auditing the rules.
- Publishing the rule project for business users to access.

Time required

This tutorial takes 3 - 4 hours to complete.

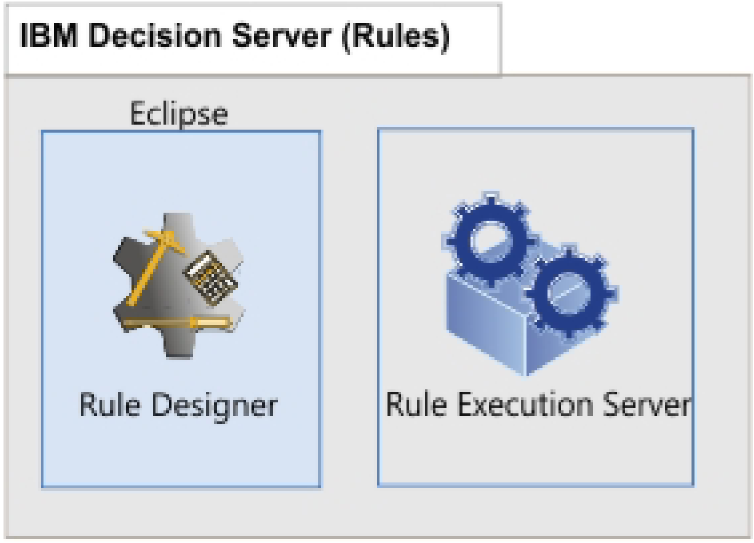
Tutorial scenario and prerequisites

Review the scenario and prerequisites before you begin the tutorial.

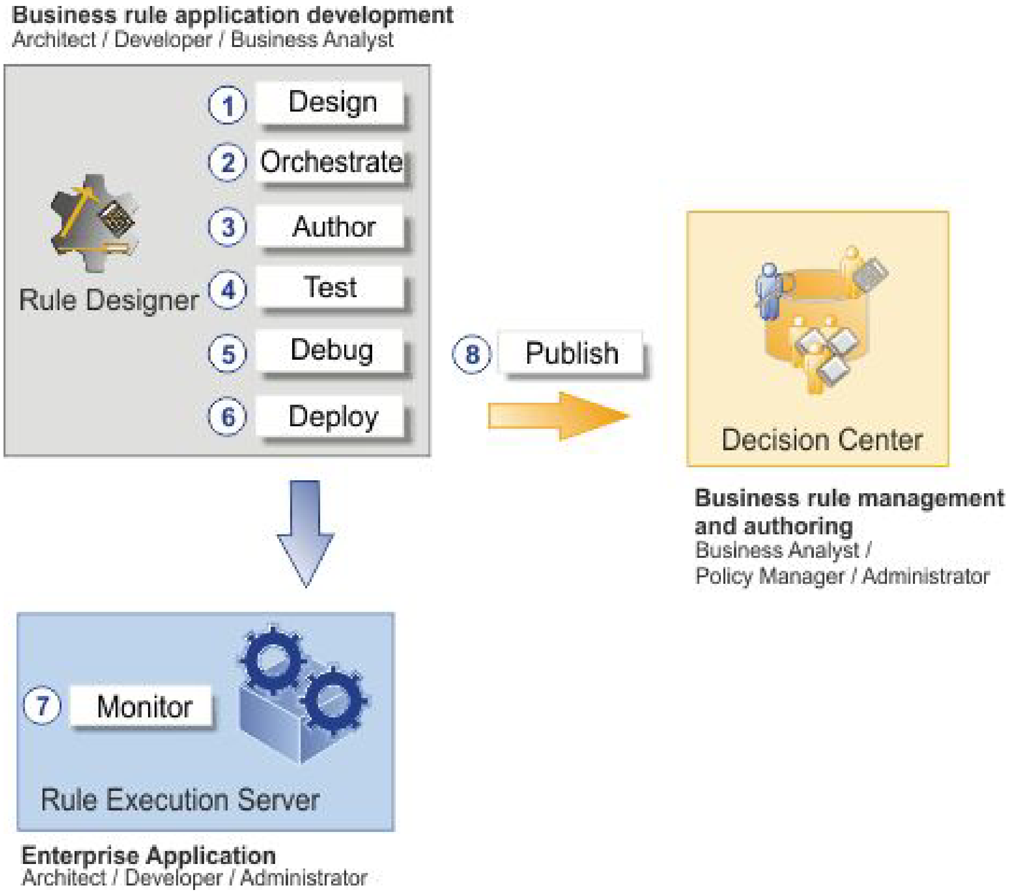
For this tutorial, you create a web-based application for a fictitious online lender. The web-based application is called Miniloan. Miniloan determines whether a customer is eligible for a loan according to specific criteria. The criteria include the amount of the loan, the yearly income of the borrower, and the duration of the loan.

When you create Miniloan, you learn how to develop a rule-based application in Rule Designer and how to execute rules in Rule Execution Server.

In the following diagram, you can see that Rule Designer and Rule Execution Server are two components of Decision Server Rules. Rule Designer is the rule development and authoring environment, and Rule Execution Server is the rule execution environment.



When you create Miniloan, you follow the typical rule development workflow. You start from a version of the application that is hardcoded in Java, and then you replace the hardcoded logic with business rules. By doing so, you discover all of the steps that are required to develop, deploy, and maintain a rule-based application. The following diagram shows the rule development workflow, from design to publication.


Business rule application development
Architect / Developer / Business Analyst
1 Design
2 Orchestrate
3 Author
4 Test
5 Debug
6 Deploy
Rule Designer
8 Publish
Decision Center
Business rule management and authoring
Business Analyst / Policy Manager / Administrator
7 Monitor
Rule Execution Server
Enterprise Application
Architect / Developer / Administrator

For more information about a particular step that you complete, see the summary for a list of related information and related tutorials.

Prerequisites

Before you start the tutorial, ensure that your workstation meets the following requirements.

Note: On Windows 7, if you install 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.

If your application server runs with JDK 6, you must modify the settings in Eclipse to use JDK 6.

Review the product overview to understand the main goals of the product, some of its modules, and rule-oriented terminology.

Knowledge of Java is helpful.

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

To do this tutorial, make sure that you install the required products:

- Decision Server Rules: contains both Rule Designer and Rule Execution Server.
- Decision Center (optional): required if you want to do “Task 8: Publishing to Decision Center” on page 36.
- A supported version of Microsoft Excel to do “Task 4: Testing rules” on page 20.

To do 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 with the launchpad
- By using the features that are installed by default with Installation Manager. For more information, see Installing using Installation Manager

Ensure that the required products, getting started projects, and the sample server are installed. For more information, 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 intended for developers and product architects.

Learning objectives

By completing the tutorial, you learn about the following aspects of Operational Decision Manager:

- Designing a rule project.
- Orchestrating the sequence in which rules are processed.
- Defining a flow of execution.
- Writing business rules, and then test and debug the rules.
- Deploying rules to the environment in which the rules are run.
- Monitoring and auditing the rules.
- Publishing the rule project for business users to access.

Time required

This tutorial takes 3 - 4 hours to complete.

Starting the Miniloan web application

Test the Miniloan application in a web browser.

Miniloan has default business logic that is hardcoded in Java. The default business logic determines whether a borrower who enters specific information is eligible for a loan. You can test the business logic by opening Miniloan in a web browser.

To start the Miniloan application, you must first start the sample server. The sample server contains the samples that you can use to learn about Operational Decision Manager V8.7.

After you start the sample server, you can open the Miniloan application in a web browser to test the hardcoded business logic. By default, the borrower information and the loan information fields are pre-filled with a loan request for a fictional borrower, Joe. When you submit the loan request, Miniloan approves or denies the loan request based on the default business logic.

To start the Miniloan application:

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 V8.7.

   On Windows 7, if you install the product in one of the `Program Files` or `Program Files (x86)` directories, you must start the sample server with administrator write privileges. To run the sample server as an administrator, right click the **Start server** icon, and then select **Run as administrator**.
2. Wait for the sample server to start.

   If you start the sample server for the first time, it can take 10 - 15 minutes for the server to start. The command window displays server trace messages as the server starts. When complete, a feedback message indicates that the build was successful.

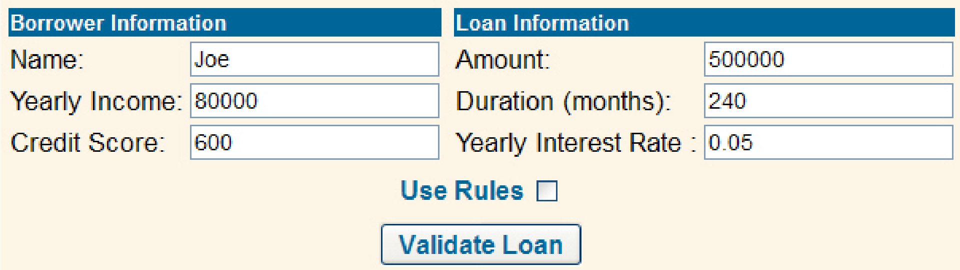
   ```
   [samples.echo] GBRPS0029I: start.server is completed.

   BUILD SUCCESSFUL
   Total time: 13 minutes 3 seconds
   Press any key to continue . . .
   ```

For more information about 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`. Ensure that you replace the *<PORT>* variable with the correct port number. For more information, see Checking the server port number. By default, the default port number is `9080`, but the port number that you use depends on how you install the product.

   The Miniloan application is displayed. The loan request information for Joe, a fictitious borrower, is pre-filled:



Borrower Information		Loan Information	
Name:	Joe	Amount:	500000
Yearly Income:	80000	Duration (months):	240
Credit Score:	600	Yearly Interest Rate :	0.05

   Important: Do not select the **Use Rules** check box. You select the check box later in the tutorial when you create the business rules.

4. Click **Validate Loan**.

   The Miniloan application screens Joe's borrower and loan information against the hardcoded business logic. As you can see, the default business logic rejects Joe's loan request because the borrower's debt-to-income ratio is too significant. In other words, Joe's income is insufficient for the amount that he is requesting.

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

5. Close the Miniloan application.

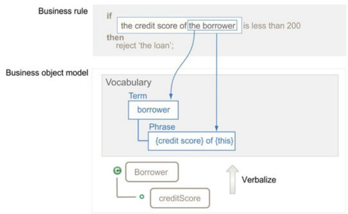
Now that you have tested the default business logic for Miniloan, you can begin creating a new rule application. The rule application that you create replaces the hardcoded business logic with the business logic that you implement.

Task 1: Designing the rule project

Create a meaningful vocabulary that helps users to write business rules in natural language. In this task, you use Rule Designer to create a vocabulary from the object model of the existing Miniloan application.

As a developer, you want to create a business rule vocabulary that business users can use to write and edit rules. The business rule vocabulary must consist of terms that are familiar to the business user. The process of creating a vocabulary is called "verbalization."

You create a business object model (BOM) based on an object model that is defined in a Java™ project. The following diagram shows how the classes and members of the BOM relate to the vocabulary with which business users are familiar.



```
if
   the credit score of the borrower is less than 200
then
   reject 'the loan';
```

This task takes 20 - 30 minutes to complete.

Step 1: Start Rule Designer and import the Getting Started projects

You can access the samples console from Rule Designer. The samples console contains the Getting Started projects that you use to verbalize business rules.

Rule Designer is an Eclipse development environment for business rule applications. Because Rule Designer is an Eclipse development environment, you can develop both rule projects and Java projects in the same environment.

The rule project that you import for this tutorial is available in American English (en_US) only. Therefore, you must start Rule Designer in American English. When you start the samples console from the **Start** menu, Rule Designer is displayed by default in American English.

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 V8.7.

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

   Tip: If you do not see the Workspace Launcher window when you start Rule Designer, you can 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 **Rule** perspective.

The Rule perspective contains various views that you discover in this tutorial, such as the Rule Explorer view and the Rule Project Map view.

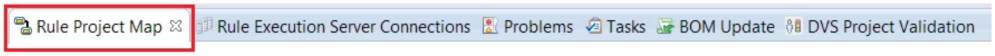
The Rule Explorer view now contains the `miniloan-server-webapp` and `miniloan-xom` projects.

`miniloan-server-webapp` is the project from which you retrieve the hardcoded rules for the Miniloan web application. Inspect the rules in the `validateWithJava` method of the `miniloan-server-webapp/src/miniloanweb/MiniloanBean.java` class to understand the rules with which Miniloan assesses loan requests.

`miniloan-xom` is the Java project for the Miniloan application. The `miniloan-xom` contains the `miniloan` package that contains the `Borrower` and the `Loan` Java classes.

Notice the Rule Project Map view. The Rule Project Map shows you the different steps of a rule project. You can use the Rule Project Map to guides you through the a rule project process. If the Rule Project Map view is not displayed, click **Window** > **Show View** > **Rule Project Map** to open it.

The following screen capture shows the Rule Project Map tab.



Step 2: Create a rule project

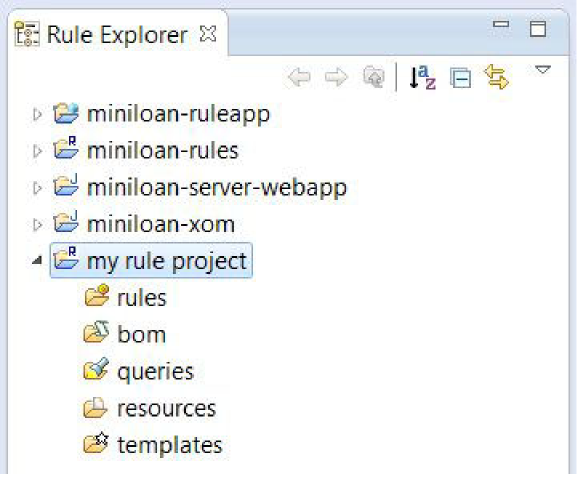
In Rule Designer, you store the business logic of your rule application in a rule project. The rule project contains rule artifacts, business object model (BOM), vocabulary, and a reference to the execution object model (XOM) against which rules are run. With this project, you can manage, build, and debug the items that comprise the business logic of your application.

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 a reference to the execution object model (XOM). With this project, you can 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. To switch to the Rule perspective, click the **Window** menu, click **Open Perspective** > **Other**. Select **Rule**, and then click **OK**.
2. Click **File** > **New** > **Rule Project**.
3. Under **Classic Rule Projects**, select **Standard Rule Project** and click **Next**.
4. In the **Project name** field, type `my rule project`.
5. Click **Finish**.

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



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

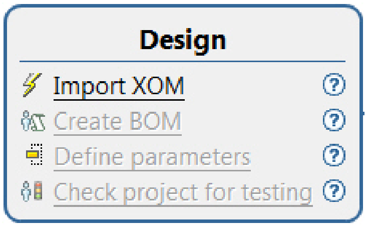
Step 3: Import the XOM in to your rule project

You have an empty rule project in the Rule Explorer view. Your new rule project now needs an object model of the Miniloan Java project, which you import in "Step 1: Start Rule Designer and import the Getting Started projects" on page 6 of this getting started tutorial. The object model that you need is called an execution object model (XOM).

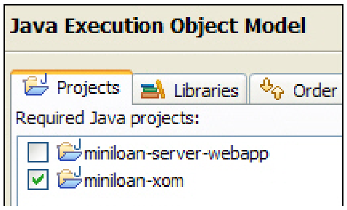
To import the XOM into your rule project:

1. In the Rule Explorer view, click `my rule project`.

   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`.

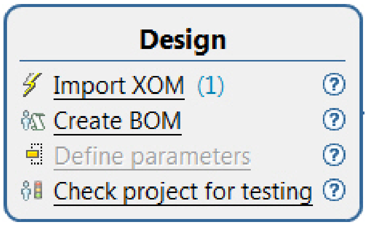


Required Java projects:

□ miniloan-server-webapp
☑ miniloan-xom

5. Click **OK**.

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



Design

- Import XOM (1)
- Create BOM
- Define parameters
- Check project for testing

Step 4: Create the business object model

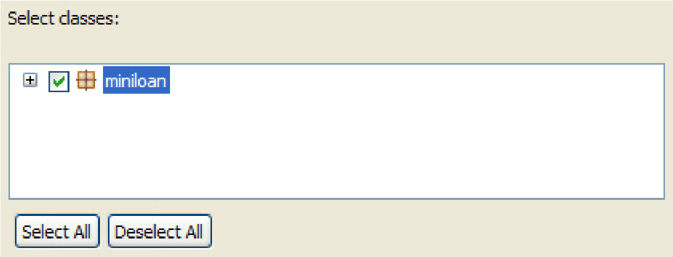
Before you can create and edit business rules, you must define a business object model (BOM). You can create a BOM manually or create it automatically by parsing your execution object model (XOM).

You use Rule Designer to parse the Java classes in the XOM automatically and create the BOM from their methods and properties. Then, you can write rules that use the verbalized 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**.

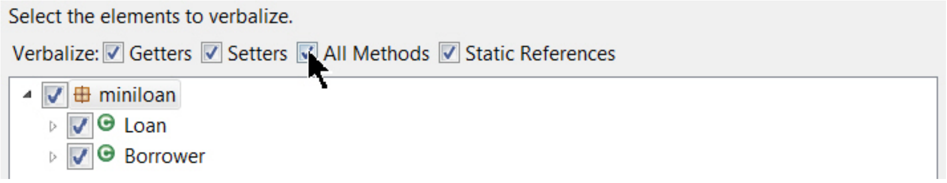
   Tip: You can also right-click the bom folder in the Rule Explorer and click **New** > **BOM Entry**.
2. In the New BOM Entry wizard, in the Name field, type `miniloan`.
3. Ensure that **Create a BOM entry from a XOM** is selected, and then click **Next**.
4. In the Choose a XOM entry field, click **Browse 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.

   Select classes:

   ☑ miniloan

   The `miniloan` packages contains the `borrower` and `Loan` classes.
6. Click **Next**.
7. In the BOM Verbalization page, you must select the **All Methods** check box. The **All Methods** checkbox ensures that all the methods are verbalized in addition to the elements already selected.

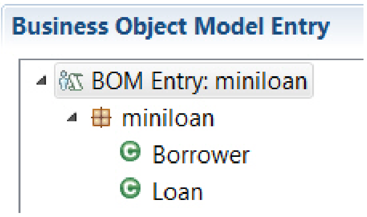
   

   Select the elements to verbalize.

   Verbalize: ☑ Getters ☑ Setters ☑ All Methods ☑ Static References

   ☑ miniloan
   - ☑ Loan
   - ☑ Borrower
8. Click **Finish**.
9. In the Rule Explorer, double-click `bom` > `miniloan` to open the BOM Editor, and take a moment to look at the BOM.

In the BOM Editor, expand the miniloan entry:



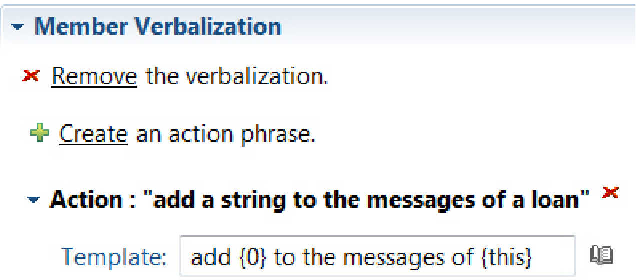
You now have in your BOM two classes that are equivalent to those in the XOM: one for the borrower and one for the loan.

10. Double-click the `Loan` class to open it in the BOM editor.

    All Java members and methods are converted and assigned a default verbalization.

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`:

    
    Member Verbalization
Remove the verbalization.
Create an action phrase.
Action : "add a string to the messages of a loan"

    Template: add {0} to the messages of {this}

    The verbalization is also used in the Rule Editor.

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

Step 5: Define ruleset parameters

A ruleset is an executable package that includes rule artifacts and the other elements, and 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 the objects that are passed to the rule engine.

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

To enable a decision to be made on the status of a loan, you must 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**.
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. In the Type column, click the **...** button, 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 **IN**.

   d. In the Verbalization column, enter `the borrower`.
4. To define the `loan` parameter, click **Add**.

   a. In the Name column type `loan`.

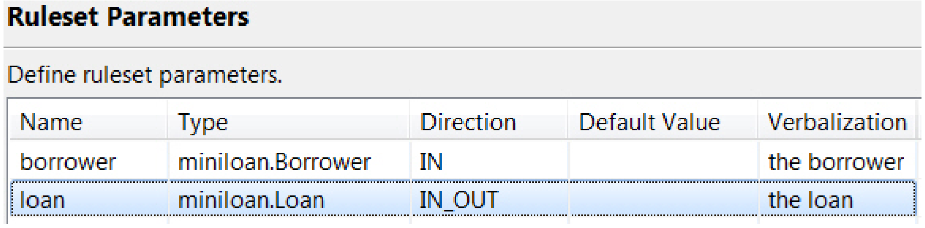
   b. In the Type column, click the **...** button, and then double-click the **Loan** type in the Matching types box.

      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 display 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 your work.

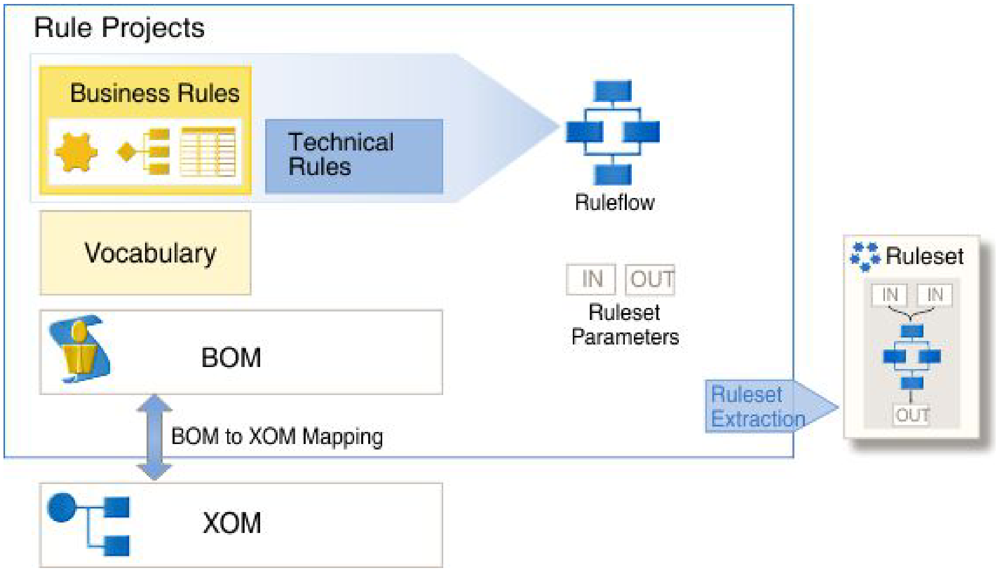
You now have a rule project with a vocabulary, and ruleset parameters. The design of your rule project is complete.

Before you can begin to write the business rules in Rule Designer, you must orchestrate how your rules are executed. In the next task, you use a ruleflow to specify the order in which the rules are executed.

Task 2: Orchestrating

Orchestrate how rules are executed. To orchestrate rule execution, you use a ruleflow to specify the order in which the 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 takes 15 - 25 minutes to complete.

Step 1: Creating rule packages

Before you define the flow of execution, you must organize your rules into packages that contain related rules. In the this step, you create two rule packages. You create rule packages for validation and eligibility. Later, you define the ruleflow of these packages.

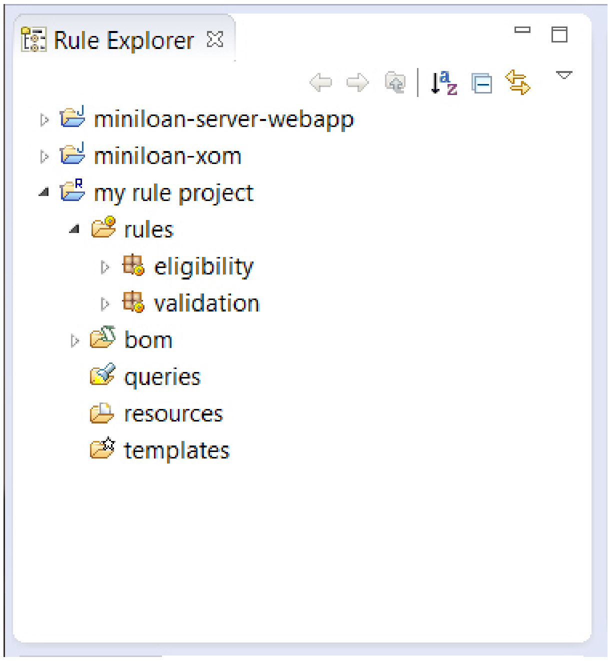
Create a rule package:

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 an `eligibility` rule package.
   a. In the Rule Project Map, click **Add rule package**.
   b. In the New Rule Package wizard, enter `eligibility` in the Package field.
   c. Click **Finish**.

Your rule project now contains two rule packages in which you can store rules.



Step 2: Creating the ruleflow diagram

In this step, you create a ruleflow. A ruleflow is a diagram that consists of several rule tasks that are connected by logical links. A ruleflow shows the sequence of execution of business rules.

In step 1, you create `validation` and `eligibility` rule packages. Now, you must design the ruleflow that creates a logical connection between the rule packages.

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, enter `miniloan`.
4. Click **Finish**. The ruleflow editor opens.

   Now, you can construct the ruleflow. In the ruleflow diagram, you specify how tasks are related. In other words, how, when, and under what conditions rules are executed.
5. Click **Create a start node** and then click 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: Defining rule tasks

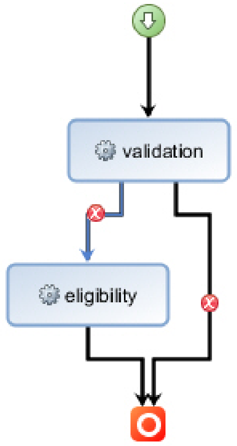
Now that you have the start and end notes in the ruleflow editor, you must define the ruleflow. To define the ruleflow, you select the rule packages to include in the diagram, and then you create the transitions between the rule packages.

To define the rule tasks:

1. Drag and drop the `validation` rule package from the Rule Explorer view to the ruleflow editor.

   The `validation` rule package becomes a rule task in the ruleflow. When you add the package into the ruleflow editor, any rule that you create in the package is part of the execution by default.
2. Drag and drop the `eligibility` rule package from the Rule Explorer view to the ruleflow editor.
3. Click the **Create a transition** button ↓ and create the following transitions (shown as arrows) by clicking the first item and then clicking the second item:
   a. Connect the start node to the `validation` task.
   b. Connect the `validation` task to the `eligibility` task.
   c. Connect the `eligibility` task to the **end node**.
   d. Connect the `validation` task to 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: Defining the main transition

You can define conditions on transitions in the ruleflow. In this step, 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`.

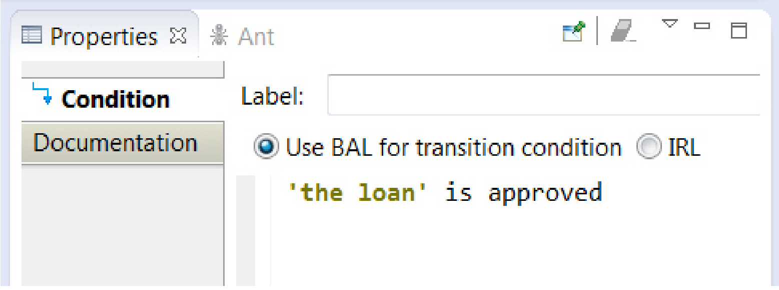
   The Properties view opens and shows the condition for this transition.

   Tip: If you cannot see the Properties view, click **Window** > **Show View** > **Properties** to open the Properties view.
2. In the Properties view, enter `data approved` in the **Label** field.

3. Ensure that **Use BAL for transition condition** is selected. Business Action Language (BAL) is the language in which you write your conditions.
4. In the text area, enter 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 enter the statement directly in the text area.

   The Properties view should look like the following:



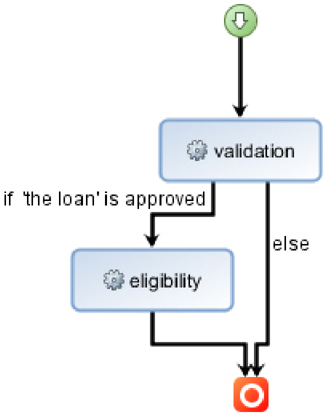
Label:

● Use BAL for transition condition ○ IRL

```
'the loan' is approved
```

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

   Your ruleflow should now look like the following:



Step 5: Defining the final action

You now have transitions between your rule projects. You can also define a final action to display a message at the end of rule execution. At execution time, the final action displays a message in the console that indicates the status of the loan at the end of rule execution.

To define the final action:

1. Click the **end node**.

   You can enter the final action in the Properties view.
2. In the **Final Action** section, make sure **Use BAL for action** is selected.
3. In the text area, enter a space to display the Content Assist box, and enter the following final action:

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

   Ensure that you have a semicolon (;) at the end of the line.
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 now have a ruleflow that describes the sequence of execution for your rule projects. In the next task, you will write an action rule.

Task 3: Authoring rules

Write rules by using the vocabulary that you verbalized in task one.

As a developer, your role is to write the initial business rules and design the rule templates. Your objective is to give business users to power to write and edit business rules in a web environment.

In this task, you create an action rule, and then you import the remaining rules into your project. An action rule defines the specific actions that are taken if particular conditions are met. The action rule that you write in this task is based in part on the vocabulary that you create earlier in this tutorial. The following is an example of an action rule.

```
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 takes 15 - 30 minutes to complete.

Step 1: Creating an action rule

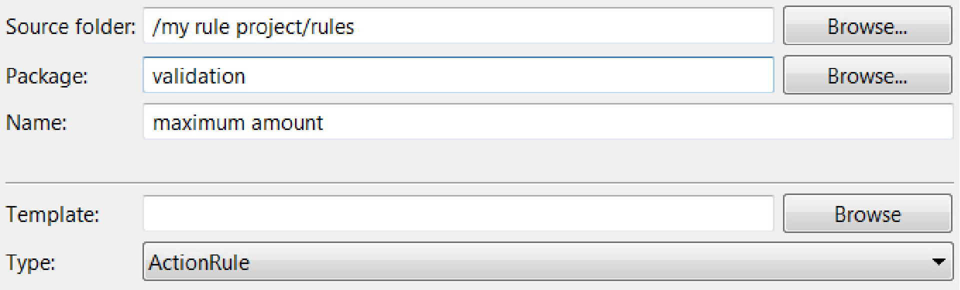
You want to create an action rule that rejects all loan requests greater than the amount of 1,000,000.

To create an action rule that rejects loan requests greater than 1,000,000:

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`, and in the **Name** field, name the rule `maximum amount`.



Source folder: /my rule project/rules
Package: validation
Name: maximum amount
Template:
Type: ActionRule

3. Click **Finish**.

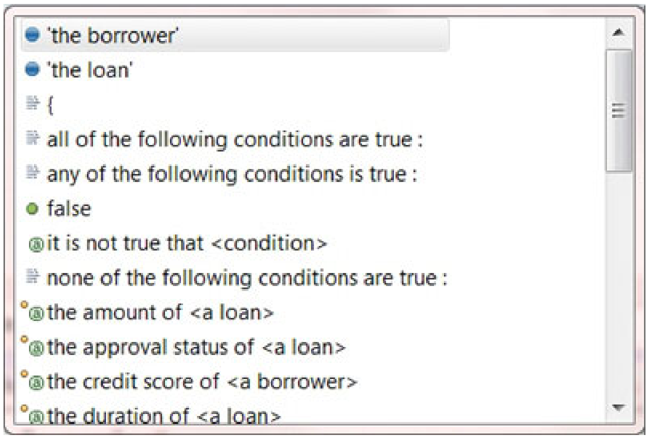
   The Intellirule editor opens. Intellirule is an text editor in which you can create and edit business rules. When you use the Intellirule editor to build a rule, you must create the parts that the rule requires. You can use a completion mechanism to select from a list of rule fragments that can be used with the word that you enter.

Step 2: Completing the action rule

Now that you opened the Intellirule editor, you can use the code completion mechanisms of the Intellirule editor to help you create the action rule.

To complete the action rule with the completion mechanism:

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

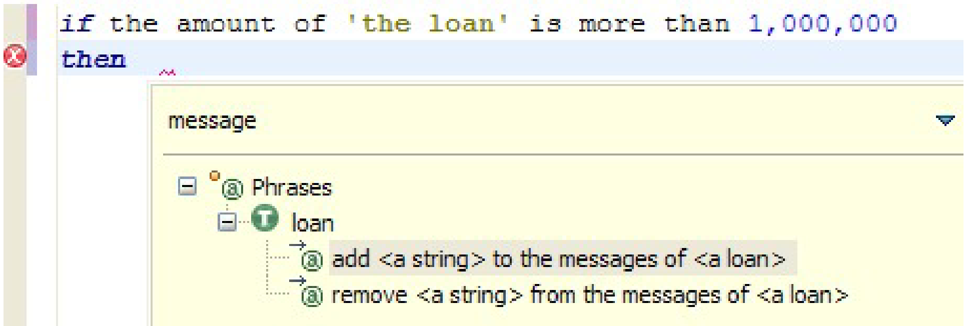
   

   'the borrower'
   'the loan'
   {
   all of the following conditions are true :
   any of the following conditions is true :
   false
   it is not true that <condition>
   none of the following conditions are true :
   the amount of <a loan>
   the approval status of <a loan>
   the credit score of <a borrower>
   the duration of <a loan>

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

   ```
   if 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 to activate the tree view of the completion menu. The tree view displays possible phrases from which you can select to build your rules.
3. In the tree view, type `message` in the field at the top to display only terms and phrases about messages:

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

   message

   Phrases
   loan
   add <a string> to the messages of <a loan>
   remove <a string> from the messages of <a loan>

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. If you are still in the Content Assist Box, press Esc.
6. Press Enter to create a new line, enter a space, and then write the following statement. You can use the Content Assist box:

```
reject 'the loan' ;
```

7. Press Ctrl+Shift+F to format the rule.

   Your final rule is the following:

```
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 about using the rule editors, see Working with action rules.

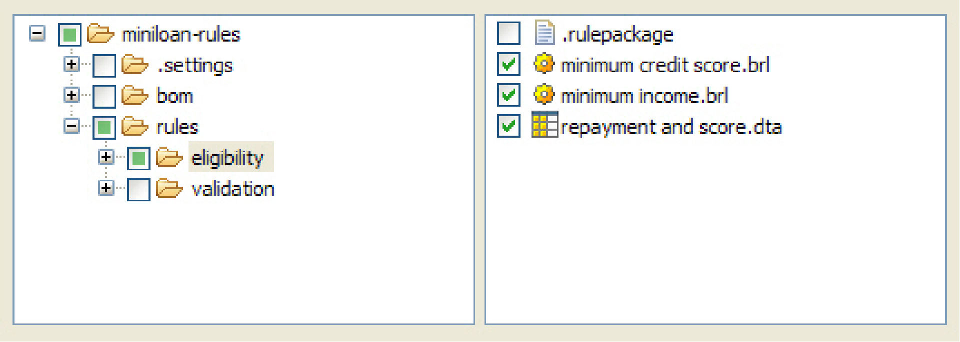
Step 3: Importing remaining rules

You created the validation rule. Now, you must add the rules that determine whether a borrower is eligible for a loan. For this step, you can import the eligibility rules into your rule project.

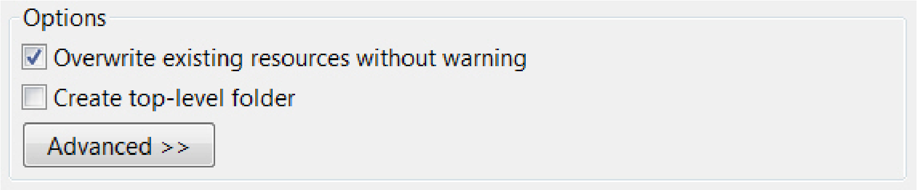
To import the remaining rules into your rule project:

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.
6. Ensure that `eligibility` is highlighted, and then clear the `.rulepackage` check box.



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



Options

☑ Overwrite existing resources without warning

□ Create top-level folder

Advanced >>

9. Click **Finish**.

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

Step 4: Viewing 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 review 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.

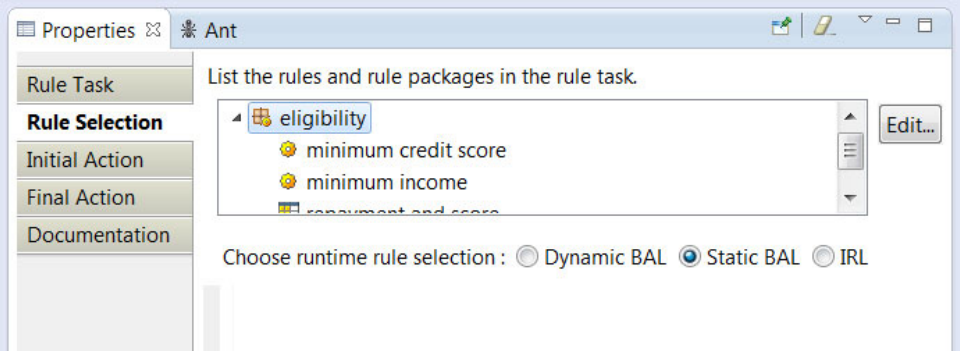
   repayment and score

   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.

   Properties ☒ Ant

   Rule Task

   Rule Selection

   Initial Action

   Final Action

   Documentation

   List the rules and rule packages in the rule task.

   eligibility
   - minimum credit score
   - minimum income
   - repayment and score

   Edit...

   Choose runtime rule selection : ○ Dynamic BAL ● Static BAL ○ IRL

   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

Test the rules that you write. In this task, you can create a scenario file to compare the outcome of the rules that you write with your expectation of how the rules should behave.

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. 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 do the following:

- 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 takes 20 - 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 you create 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. The borrower class is under `my rule project > bom > miniloan > miniloan > Borrower`.

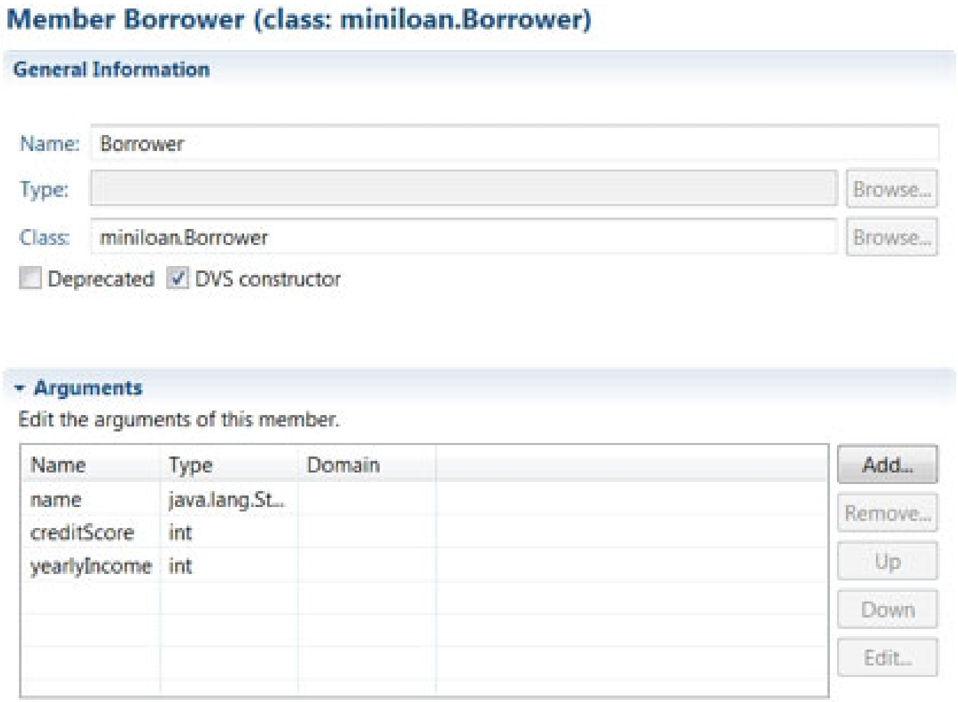
   The `Borrower` class opens in the BOM Editor. In the Members section, you can see that the `Borrower` class has two constructors:

   - `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:



Member Borrower (class: miniloan.Borrower)

General Information

Name: Borrower

Type:

Class: miniloan.Borrower

□ Deprecated ☑ DVS constructor

Arguments

Edit the arguments of this member.

Name	Type	Domain
name	java.lang.St...	
creditScore	int	
yearlyIncome	int	

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 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, look at miniloan-xom/src/miniloan/Borrower.java and miniloan-xom/src/miniloan/Loan.java. For more information about 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 might 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. Your project does not have errors. If no errors or warnings are listed, the project is valid and you can create a scenario file. If your project has errors, you must resolve them before you resume this tutorial.

Step 3: Create a scenario file

After you ensure that your project is free of errors and warnings, you can 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 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 is approved.

To populate the Excel scenario file:

1. Outside of Eclipse, go to *`<MyEclipseWorkspace>`*`/my rule project` and open `miniloan-test.xls`.

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

   Do not use the Excel editor that is embedded in Eclipse because 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.

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

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

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

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

To test the Excel scenario file locally:

1. Click **Run** > **Run Configurations**.
2. Configure your test:
   a. In the side pane of the Run Configurations window, right-click **DVS Excel File**, and click **New**.
   b. In the Name field, enter `Miniloan Test`. This is the name of 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 ensure that the **Local execution** option is selected.
3. Click **Apply**, and then click **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:

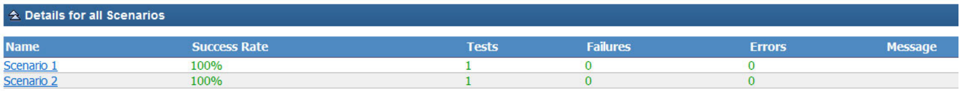
```
--- Output for scenario 'Scenario 1' :
false [Too big Debt-To-Income ratio]

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

Execution finished
Starting generation of DVS HTML report now
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	

   The preceding screen capture shows that Scenario 1 and Scenario 2 performed as expected. In Scenario 1, the loan is rejected (the loan is approved equals: `FALSE`) because the amount of the loan is superior to 1,000,000. In Scenario 2, the loan is approved (the loan is approved equals: `TRUE`) because the amount of the loan request is inferior to 1,000,000. The business logic that you wrote executes as expected.

6. Close the report.

Now that you understand how to test rules, you can learn how to debug the rule project in an Excel file.

Task 5: Debugging

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

The rules that you wrote execute as expected. Now, you can execute the rules in a sandbox environment for testing and debugging purposes. You can use Rule Designer to test and debug your rule project.

This task takes 15 minutes to complete.

Step 1: Insert a breakpoint

You can use an Excel scenario file to debug the execution of the rules. You can step into the code when you debug in Rule Designer. 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: Debug the rule execution

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

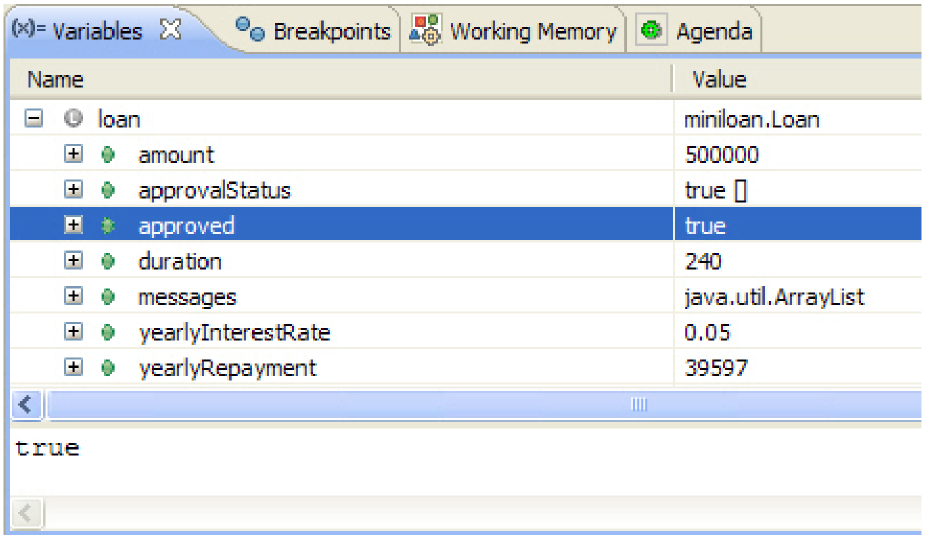
To debug the execution:

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**.

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

      When a dialog is displayed, click **Yes** to open the Debug perspective. The debugging commands are available from the Debug view and 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 the **Step Into** button ( ). 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`.

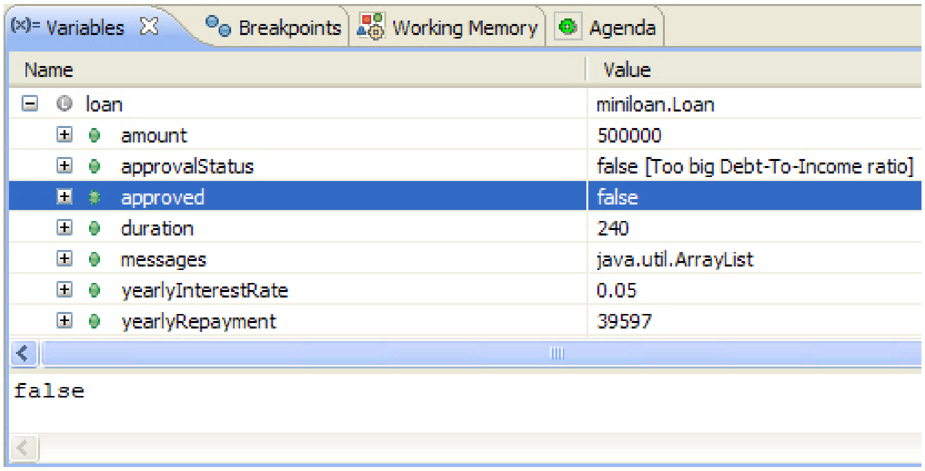


Name	Value
loan	miniloan.Loan
amount	500000
approvalStatus	true []
approved	true
duration	240
messages	java.util.ArrayList
yearlyInterestRate	0.05
yearlyRepayment	39597

```
true
```

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



Name	Value
loan	miniloan.Loan
amount	500000
approvalStatus	false [Too big Debt-To-Income ratio]
approved	false
duration	240
messages	java.util.ArrayList
yearlyInterestRate	0.05
yearlyRepayment	39597

```
false
```

3. Click the **Resume** button (▶) 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 the **Resume** button (▶) to complete the execution of Scenario 2. When execution ends, the console view shows the following 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.
6. Close the ruleflow.

You created a scenario file to test your rules, and you used it for debugging. Now, you can deploy your rules to Rule Execution Server.

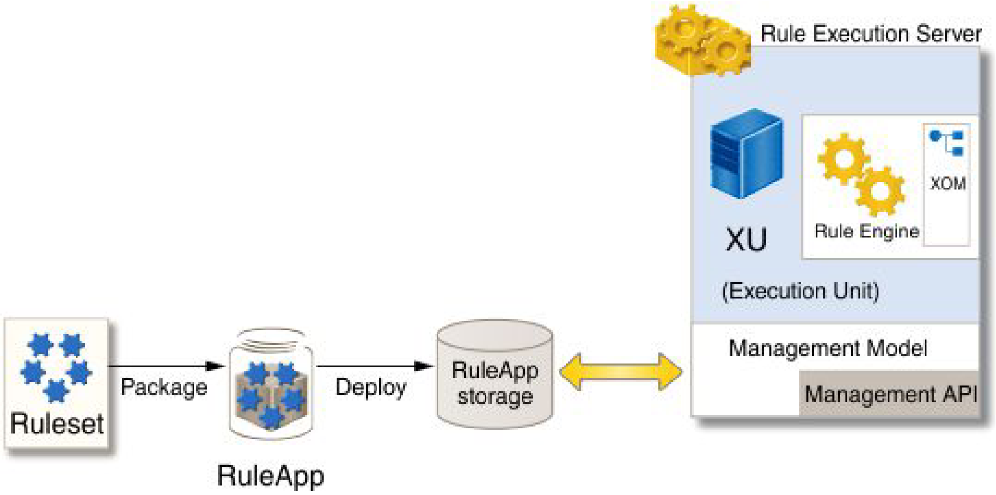
Task 6: Deploying rules

Deploy your ruleset to the Rule Execution Server. Rule Execution Server is the runtime environment that contains the rule engine to execute rules.

Now that the rules are verbalized and tested, business users can more easily write business rules in natural language. Next, you must integrate your rules in to the Miniloan application. You can integrate rules into the Miniloan application in two steps.

First, 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 everything necessary for execution (rules, ruleflow, and so on).

Then, call the integration code in the Miniloan web application so that the business logic that you implement and validate in previous tasks is 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 Web Services Description Language (WSDL) file from your ruleset, and then test it in Rule Designer.

This task takes 15 - 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:

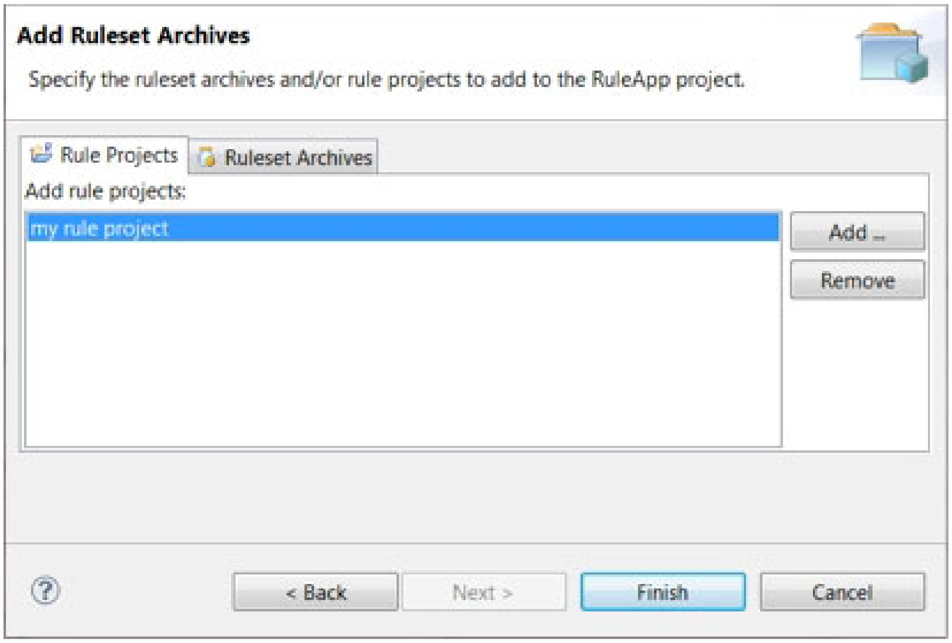
1. Ensure that the sample server is started.

   Click **Start** > **All Programs** > **IBM** > *package_group* > **Sample Server** > **Start server**.
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**.

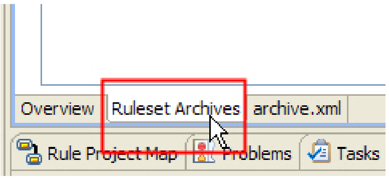


5. Click **Finish**.

   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 so that you can 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 set to `true`.
   b. Click **New** to create a new property for the ruleset.
   c. In the Edit Property window, 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 version 1.0 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 that interacts with the rule engine. Rule Execution Server handles the management, performance, security, and logging capabilities that are associated with the execution of your rules.

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

To view the deployed RuleApp:

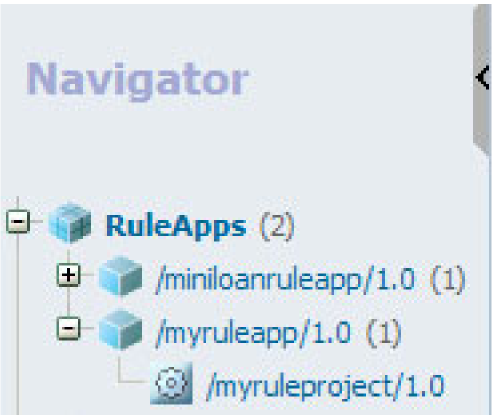
1. Click **Start** > **All Programs** > **IBM** > *package_group* > **Sample Server** > **Rule Execution Server Console**.

   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`

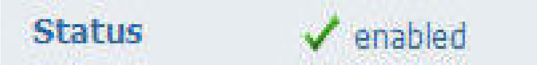
   Password: `resAdmin`
3. Click the **Explorer** tab.
4. In the Navigator, expand **RuleApps**, and then `/myruleapp/1.0`.

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

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

   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

With the Miniloan web application, you can choose whether the business logic that is embedded in the application is pure Java code or coded in a ruleset. When you first started the Miniloan application, you did not have 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 that the application now calls the ruleset.

The `validateWithJRules` method of the Miniloan bean validates the rules.

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 now relies on the rules that you wrote.

   Some fields are temporarily added under Ruleset Information to show you, 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.

   Now, you see the following response:

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

   The loan is approved and no rule is executed.
5. Close the Miniloan web application. You 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 transparent decision service component passes input parameters to the rule engine and accesses the return values. The transparent decision service support includes traceability from transparent decision services to rules, runtime monitoring, and version management.

You can retrieve the Web Services Description Language (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

If you retrieved the WSDL file and saved it to your workspace, you can use Rule Designer to test the web service. In the test environment, you enter some test data and call the WSDL operation.

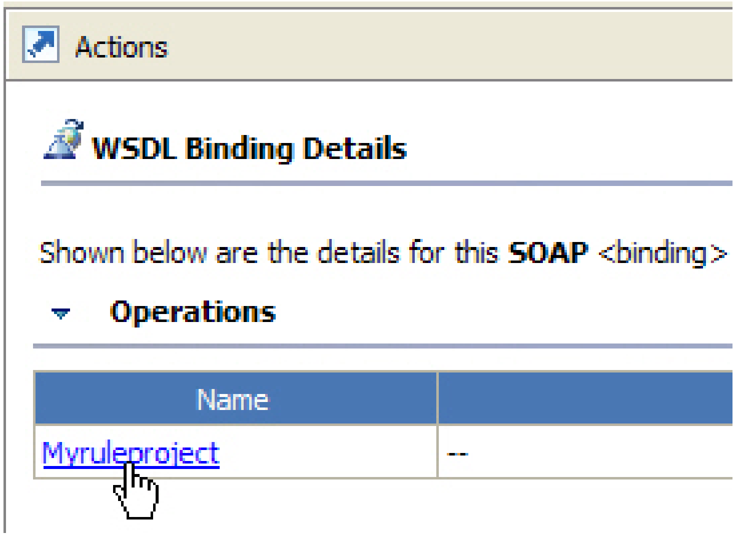
To test the web service:

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 Sample Object Access Profocol (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**.



WSDL Binding Details

Shown below are the details for this **SOAP** <binding>

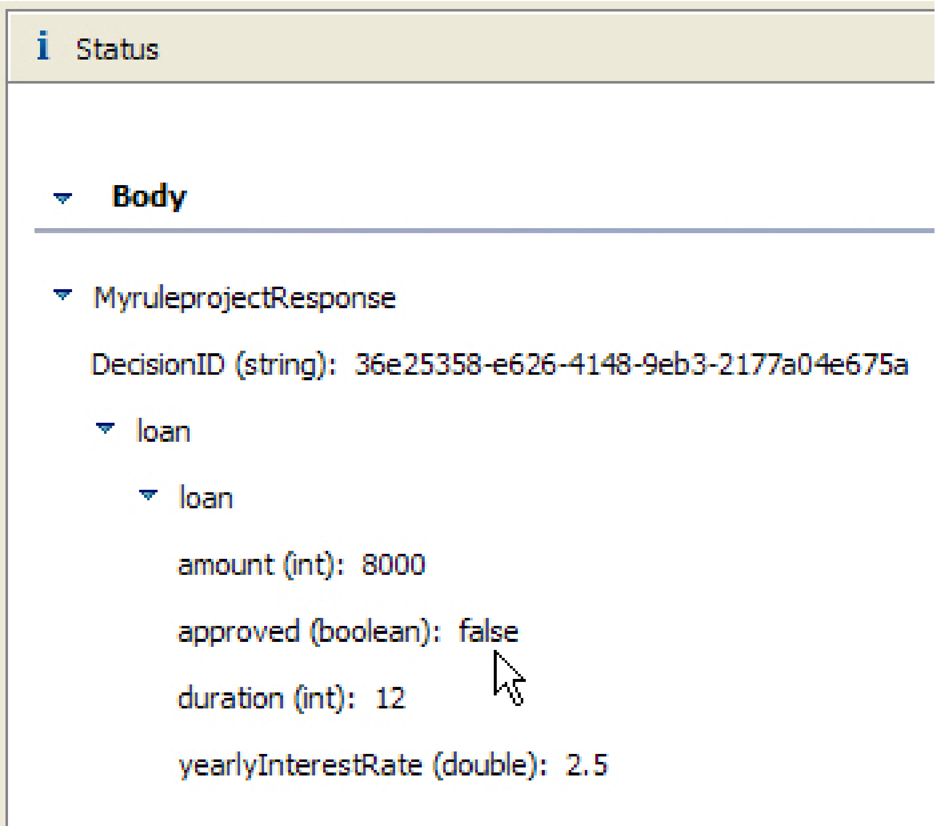
Operations

Name	
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 you can expect the loan to be rejected.
10. Click **Go** to call the web service on Rule Execution Server.
11. In the Status section, review the response.

    The loan was not approved.



Body

MyruleprojectResponse

DecisionID (string): 36e25358-e626-4148-9eb3-2177a04e675a

loan

loan

amount (int): 8000

approved (boolean): false

duration (int): 12

yearlyInterestRate (double): 2.5

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

Your ruleset is now deployed to Rule Execution Server. You tested the ruleset 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 the 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, you can monitor the execution of your rulesets with Rule Execution Server.

Auditors can analyze the execution performance of your ruleset, and troubleshoot any problematic transactions. 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 takes 10 - 15 minutes to complete.

Step 1: Run Rule Execution Server diagnostics

Run the Rule Execution Server diagnostics to help you identify errors with the execution environment.

To run the Rule Execution Server diagnostics:

1. Open the Rule Execution Server console.

   Click **Start** > **All Programs** > **IBM** > *package_group* > **Sample Server** > **Rule Execution Server Console**.
2. Sign in to the Rule Execution Server console with the following credentials:

   Username: `resAdmin`

   Password: `resAdmin`
3. Click the **Diagnostics** tab, and then click **Run Diagnostics**.

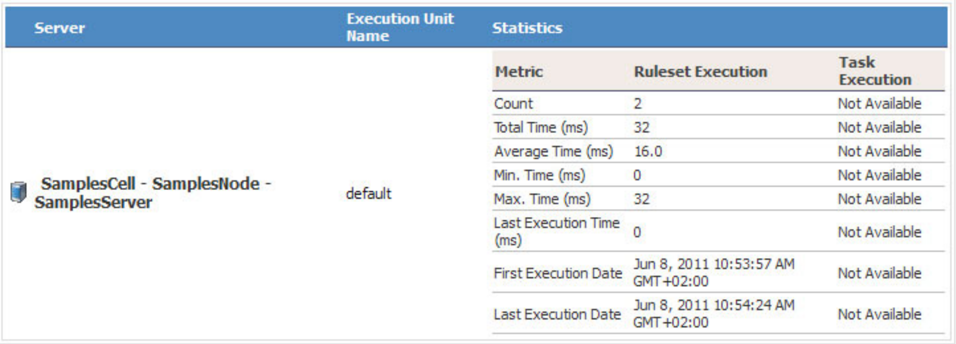
   The green check marks indicate that rule execution was successful.
4. Click **Expand All** to show the details of each test. The diagnostics test various aspects that relate to the execution environment. Review the diagnostics, and then review the statistics on deployed RuleApps.

Step 2: View statistics on deployed RuleApps

When the execution environment is functioning correctly but performance problems are being reported, you can consult statistics from the Rule Execution Server console.

To view statistics on deployed RuleApps:

1. In the Rule Execution Server console, 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 is 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

   In the previous screen capture, you can see the results under the Ruleset Execution column. The Task Execution column is "Not Available" because the statistics display either the Ruleset Execution or the Task Execution modes.
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 increases 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 that you simulate for the ruleset is now stored and logged in Decision Warehouse.

To simulate a transaction in the Miniloan application:

1. Ensure that the Miniloan application is started. 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.

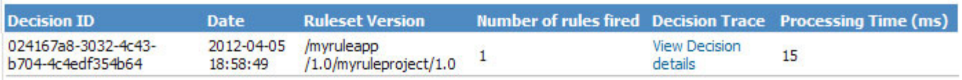
Step 4: Search for past transactions in Decision Warehouse

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 executed in the Miniloan application. For example, the decision for the transaction that is 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

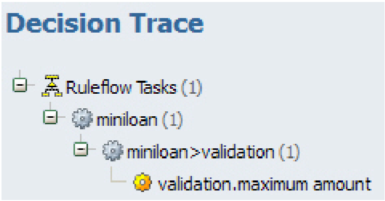
You can use Decision Warehouse to understand why the loan is rejected. You check the execution details for a decision and view the rules that were executed.

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.



Decision Trace

Ruleflow Tasks (1)
miniloan (1)
miniloan>validation (1)
validation.maximum amount

The loan was rejected because the amount entered in “Step 3: Execute a transaction in the Miniloan application” on page 35 exceeds the maximum amount of 1000000.

3. In the Input Parameters section, review 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 that you develop 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 do this task, you must have Decision Center installed.

This task takes 10 - 15 minutes to complete.

Step 1: Publish the rule project to Decision Center

Ensure that the sample server is started. For more information about starting the sample server, see “Starting the Miniloan web application” on page 4.

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.

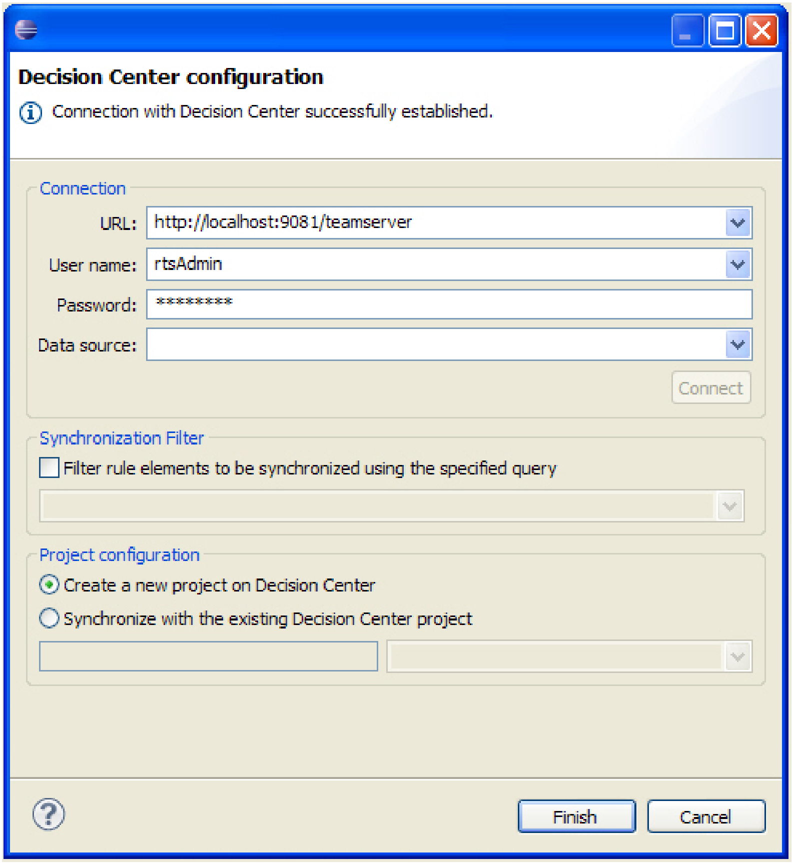
   URL: `http://localhost:<PORT>/teamserver`

   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**.



Decision Center configuration

ⓘ Connection with Decision Center successfully established.

Connection

URL: http://localhost:9081/teamserver

User name: rtsAdmin

Password: *********

Data source:

Synchronization Filter

□ Filter rule elements to be synchronized using the specified query

Project configuration

● Create a new project on Decision Center

○ Synchronize with the existing Decision Center project

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 the 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:

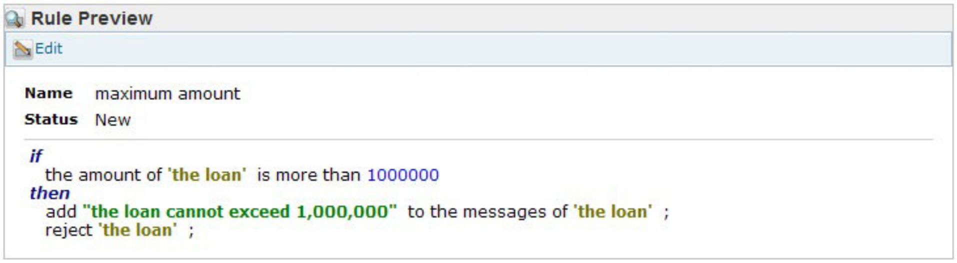
1. From the **Start** menu, click **All Programs** > **IBM** > *package_group* > **Sample Server** > **Decision Center Enterprise Console**.

   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 with the following credentials:

   Username: `rtsUser1`

   Password: `rtsUser1`
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.



Name maximum amount

Status New

```
if
  the amount of 'the loan' is more than 1000000
then
  add "the loan cannot exceed 1,000,000" to the messages of 'the loan' ;
  reject 'the loan' ;
```

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.

Summary

You have now completed the getting started tutorial.

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.

You saw how to 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 Setting up rule projects	Tutorial: Defining a vocabulary
"Task 2: Orchestrating" on page 11	Orchestrating ruleset execution	Tutorial: Creating your first ruleflow

Tasks	Related information	Related tutorials
"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 24	Executing using the classic rule engine	Tutorial: Debugging an Excel scenario file Tutorial: Debugging a ruleset
"Task 6: Deploying rules" on page 26	Deploying and exporting RuleApps	Tutorial: Managing RuleApps
"Task 7: Monitoring" on page 33	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 36	Synchronizing from Designer	

Restarting the tutorial

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

Stopping the sample server

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

Tutorial: Getting started with business rules

This tutorial is an introduction to some of the features of Operational Decision Manager V8.7 and includes exercises to help you become familiar with creating, running, and executing rules. In this tutorial, you learn how to create and run a rule-based application in Rule Designer, and how to execute rules in Rule Execution Server. If you are doing this tutorial for the first time, ensure that you follow the links to read the related sections.

Learning objectives

By completing the tutorial, you learn about the following aspects of Operational Decision Manager:

- Designing a rule project.
- Orchestrating the sequence in which rules are processed.
- Defining a flow of execution.
- Writing business rules, and then test and debug the rules.

- Deploying rules to the environment in which the rules are run.
- Monitoring and auditing the rules.
- Publishing the rule project for business users to access.

Time required

This tutorial takes 3 - 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