Course Exercises Guide

Developing Rule Solutions in IBM Operational Decision Manager V8.10

Course code WB404 / ZB404 ERC 1.2

IBM Training

June 2019 edition

Notices

This information was developed for products and services offered in the US.

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 Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America

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 jurisdictions 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 websites 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 provide in any way it believes appropriate without incurring any obligation to you.

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 actual people or business enterprises is entirely coincidental.

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.

© Copyright International Business Machines Corporation 2019.
This document may not be reproduced in whole or in part without the prior written permission of IBM.

US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Trademarks ix

Exercises description x

General exercise information xiii
How to follow the exercise instructions xiii
File references for exercises xv
Projects for exercises xv
Sample server port xvii
Using the product documentation xvii
Checking for course corrections xviii

Exercise 1. Operational Decision Manager in action 1-1
Section 1. Running the Miniloan web application 1-4
1.1. Setting up your environment 1-4
1.2. Running the application 1-5
Section 2. Managing business rules in Decision Center 1-7
2.1. Editing rules in Decision Center Business console 1-7
2.2. Reviewing the rule history 1-13
2.3. Changing the credit score requirements 1-15
Section 3. Validating rule changes 1-19
Section 4. Deploying updated rules from Decision Center to Rule Execution Server 1-21
4.1. Deploying the decision service 1-21
Section 5. Monitoring the deployed rules in Rule Execution Server 1-25
Section 6. Viewing the effects in the Miniloan application 1-28
Section 7. Exploring the development environments in Rule Designer 1-29
7.1. Opening Rule Designer 1-29
7.2. Taking a quick tour of Rule Designer 1-31
7.3. Synchronizing Rule Designer and Decision Center environments 1-34
Section 8. Shutting down the modules 1-39

Exercise 2. Setting up decision services 2-1
Section 1. Creating a standard rule project 2-3
1.1. Setting up your environment for this exercise 2-3
1.2. Creating a standard rule project with a BOM 2-4
Section 2. Importing the XOM and setting up the BOM 2-7
2.1. Importing the execution object model (XOM) 2-7
2.2. Creating a reference to the XOM in the rule project 2-9
2.3. Creating the BOM 2-10
2.4. Exploring the BOM 2-13
Section 3. Creating the main rule project for the decision service 2-16
3.1. Identifying the next tasks in decision service development 2-19
Section 4. Creating and defining the decision operation 2-20
4.1. Creating the decision operation 2-20
4.2. Creating variables for parameters 2-22
4.3. Binding the variables to parameters 2-23
4.4. Creating a ruleset variable 2-25
Section 5. Creating rule packages 2-27
Section 6. Publishing from Rule Designer to Decision Center 2-29
6.1. Publishing the rule project to Decision Center 2-29

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

6.2. Opening the project in the Decision Center Business console . . . 2-32
6.3. Synchronizing Rule Designer with Decision Center . . . 2-34
Section 7. Deleting a decision service from Decision Center . . . 2-37
Section 8. Importing a decision service from Decision Center . . . 2-38
8.1. Creating a project in Rule Designer from Decision Center . . . 2-38
8.2. Finalizing the rule project in Rule Designer . . . 2-41

Exercise 3. Modeling decisions . . . 3-1
Section 1. Creating a diagram . . . 3-2
1.1. Open Decision Center Business console . . . 3-2
1.2. Create a model . . . 3-3
1.3. Design the Match Race to Runner decision model . . . 3-4
1.4. Model the node structure . . . 3-4
1.5. Create custom data types . . . 3-6
1.6. Set the output types for the data nodes . . . 3-7
Section 2. Author and test the decision logic . . . 3-8
2.1. Define the Runner category decision node and logic . . . 3-8
2.2. Define the Matching race logic . . . 3-9
2.3. Test the model . . . 3-10
2.4. Add a rule to handle cases when no race matches the runner . . . 3-13
2.5. Validate the model . . . 3-13
2.6. Editing rule sequence . . . 3-14
Section 3. Exporting and importing a decision model service . . . 3-16
3.1. Sign in to IBM Decision Composer . . . 3-16
3.2. Export the project . . . 3-16
3.3. Import the project . . . 3-17
Section 4. Deploy the decision model service . . . 3-19

Exercise 4. Working with the BOM . . . 4-1
Section 1. Finalizing the XOM . . . 4-2
1.1. Setting up your environment for this exercise . . . 4-2
1.2. Understanding the problem . . . 4-3
1.3. Finish implementing the Borrower class . . . 4-4
1.4. Finish implementing the Test class . . . 4-10
Section 2. Creating a rule project with a BOM . . . 4-11
Section 3. Working with the vocabulary that is required to author rules . . . 4-13
3.1. Creating the default vocabulary . . . 4-13
3.2. Examining the verbalization for members of the Borrower class . . . 4-16
3.3. Editing the default verbalization . . . 4-18

Exercise 5. Refactoring . . . 5-1
Section 1. Refactoring rule artifacts after vocabulary changes . . . 5-2
1.1. Setting up your environment for this exercise . . . 5-2
1.2. Exploring the default verbalization . . . 5-2
1.3. Changing the default verbalization of the getBankruptcyAge method . . . 5-3
1.4. Changing the default verbalization of the duration member . . . 5-5
Section 2. Maintaining consistency between the BOM and the XOM . . . 5-7
2.1. Adding a XOM class that is missing in the BOM . . . 5-7
2.2. Adding a XOM method that is missing in the BOM . . . 5-9
2.3. Creating a rule to use this new attribute . . . 5-10
2.4. Deprecating BOM members . . . 5-11

Exercise 6. Working with ruleflows . . . 6-1
Section 1. Exploring a ruleflow diagram . . . 6-2
1.1. Setting up your environment for this exercise . . . 6-2
1.2. Exploring the ruleflow . . . 6-3

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

1.3. Exploring action tasks and transitions . . . 6-7
1.4. Using parameters and variables in rules . . . 6-10
Section 2. Creating a ruleflow . . . 6-11
Section 3. Defining a main ruleflow . . . 6-15

Exercise 7. Exploring action rules . . . 7-1
Section 1. Exploring rule structure . . . 7-2
1.1. Setting up your environment for this exercise . . . 7-2
1.2. Exploring the rules . . . 7-3
Section 2. Exploring rule behavior . . . 7-5
Section 3. Working with automatic variables . . . 7-8

Exercise 8. Authoring action rules . . . 8-1
Section 1. Authoring action rules in the Intellirule editor . . . 8-2
1.1. Setting up your environment for this exercise . . . 8-2
1.2. Authoring rules with the Intellirule editor . . . 8-3
Section 2. Authoring actions rules in the Guided editor . . . 8-8
Section 3. Authoring rules with parameters . . . 8-11

Exercise 9. Authoring decision tables . . . 9-1
Section 1. Authoring a decision table . . . 9-2
1.1. Setting up your environment for this exercise . . . 9-2
1.2. Creating the table . . . 9-3
1.3. Completing the table cells with values . . . 9-8

Exercise 10. Working with static domains . . . 10-1
Section 1. Exploring a collection domain . . . 10-2
1.1. Setting up your environment for this exercise . . . 10-2
1.2. Exploring the domain . . . 10-3
1.3. Testing the rule . . . 10-5
Section 2. Working with an enumeration of literals . . . 10-8
2.1. Creating the domain . . . 10-8
2.2. Testing the domain in a rule . . . 10-11
Section 3. Defining an enumeration of static references . . . 10-13
3.1. Creating the static references Java class . . . 10-13
3.2. Authoring an action rule that uses a domain of static references . . . 10-20

Exercise 11. Working with dynamic domains . . . 11-1
Section 1. Creating a dynamic domain in Rule Designer . . . 11-3
1.1. Setting up your environment for this exercise . . . 11-3
1.2. Import the domain file . . . 11-4
1.3. Create the domain . . . 11-7
1.4. Explore the dynamic domain . . . 11-12
Section 2. Using the dynamic domain in a rule . . . 11-14
Section 3. Updating the dynamic domain . . . 11-19
Section 4. Updating the XOM . . . 11-23
4.1. **Identifying the inconsistency problem . . . 11-23**
4.2. **Correcting the inconsistency problem . . . 11-23**
Section 5. Publishing the BOM and rule projects to Decision Center . . . 11-24
5.1. Publishing the decision service . . . 11-24
Section 6. Examining rules in Decision Center . . . 11-26
6.1. Opening the published projects in Decision Center . . . 11-26
Section 7. Modifying the dynamic domain in Decision Center . . . 11-29
7.1. Viewing and modifying the values of the dynamic domain . . . 11-29
7.2. Modifying the domain value in the rule . . . 11-34
Section 8. Updating Rule Designer from Decision Center . . . 11-36

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

8.1. Synchronizing the rule project 11-36

Exercise 12. Working with queries 12-1
Section 1. Searching for rule artifacts 12-2
1.1. Setting up your environment for this exercise 12-2
1.2. Searching for specific criteria across the rules 12-2
1.3. Searching for dependencies between rules 12-4
Section 2. Querying rule projects 12-7
2.1. Searching for rules that may become applicable 12-7
2.2. Searching for rules that may lead to a state 12-9
2.3. Searching for rules that use a BOM member 12-10
2.4. Applying actions with queries 12-11
Section 3. Working with queries in Decision Center 12-13

Exercise 13. Executing rules locally 13-1
Section 1. Executing rules locally by using a launch configuration 13-2
1.1. Setting up your environment for this exercise 13-2
1.2. Creating a launch configuration 13-2
1.3. Debugging with a launch configuration 13-5

Exercise 14. Debugging a ruleset 14-1
Section 1. Running the debug launch configuration 14-2
1.1. Setting up your environment for this exercise 14-2
1.2. Running the launch configuration 14-2
1.3. Applying automatic exception handling 14-3
1.4. Adding exception handing logging 14-4
Section 2. Setting breakpoints 14-6
2.1. Searching for the problem source 14-6
2.2. Preparing the debug session 14-7
2.3. Running the debugger on a ruleflow 14-7
Section 3. Debugging a rule 14-10
3.1. Setting a breakpoint in a rule 14-10
3.2. Debugging the action rule 14-11
3.3. Fixing the error in the rule 14-11
Section 4. Debugging a decision table 14-13
4.1. Finding the source of the rejection message 14-13
4.2. Setting breakpoints in a decision table 14-14
4.3. Debugging the decision table 14-15
Section 5. Debugging the ruleflow 14-16
5.1. Debugging the ruleflow 14-16
5.2. Modifying the ruleflow 14-17

Exercise 15. Enabling rule validation 15-1
Section 1. Validating the rule project 15-3
1.1. Setting up your environment for this exercise 15-3
1.2. Validating the rule project 15-4
1.3. Choosing a constructor 15-5
1.4. Editing column headings for the scenario file template 15-6
Section 2. Removing columns from the scenario file template 15-8
Section 3. Enabling testing from Business console 15-9
Section 4. Testing from Business console 15-10
4.1. Restore the server 15-13

Exercise 16. Managing deployment 16-1
Section 1. Creating deployment configurations 16-2
1.1. Setting up your environment for this exercise 16-2

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

1.2. Creating a deployment configuration 16-2
Section 2. Deploying a RuleApp from Rule Designer 16-6
2.1. Deploying the RuleApp 16-6
Section 3. Adding a ruleset property to a deployment configuration 16-9
Section 4. Deploying the managed XOM from Rule Designer 16-11
4.1. Deploying the managed XOM from the rule project 16-11
4.2. Redeploying the XOM 16-12
4.3. Deploying a RuleApp associated with a managed XOM 16-13
Section 5. Exporting and importing deployment server definitions 16-15
5.1. Exporting deployment configurations 16-15
5.2. Importing deployment configurations 16-16

Exercise 17. Using Build Command to build RuleApps 17-1
Section 1. Setting up and installing Maven Build Command 17-2
1.1. Creating a workspace folder 17-2
1.2. Creating a local Maven repository 17-2
1.3. Installing the Maven Build Command plug-in 17-2
Section 2. Setting up the project structure 17-4
2.1. **Creating the aggregator POM file 17-4**
2.2. **Creating the decision service POM file 17-5**
2.3. Creating the XOM POM file 17-6
2.4. Building the project 17-6
Section 3. Deploying your RuleApp archive with Ant 17-8

Exercise 18. Exploring the Rule Execution Server console 18-1
Section 1. Exploring the Rule Execution Server console 18-2
1.1. Setting up your environment for this exercise 18-2
1.2. Exploring the Rule Execution Server console pages 18-2
Section 2. Exploring the deployed RuleApps 18-4
Section 3. Working with deployed resources 18-6
Section 4. Exploring the Diagnostics and Server Info tabs 18-7
Section 5. Exploring the REST API tab 18-8
5.1. Exploring REST API commands 18-8
5.2. Deploying with REST 18-9
5.3. Adding references to the XOM 18-11
Section 6. Managing RuleApps and rulesets 18-13
6.1. Creating a RuleApp 18-13
6.2. Adding a ruleset to your RuleApp 18-13
6.3. Deleting the RuleApp 18-14

Exercise 19. Auditing ruleset execution through Decision Warehouse 19-1
Section 1. Enabling ruleset monitoring 19-3
1.1. Enabling ruleset execution monitoring 19-3
1.2. Testing the RuleApp to generate execution traces 19-5
Section 2. Retrieving decision traces in the Rule Execution Server console 19-8
Section 3. Optimizing Decision Warehouse 19-11
3.1. Working with monitoring options 19-11
3.2. Specifying filters on the trace data 19-12
3.3. Removing BOM serialization 19-13
Section 4. Deleting trace information from the database 19-15
4.1. Clearing traces 19-15
4.2. Removing the RuleApp 19-16

Exercise 20. Executing rules as a hosted transparent decision service (HTDS) 20-1
Section 1. Deploying the decision service to Rule Execution Server 20-2
1.1. Setting up your environment for this exercise 20-2

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

1.2. Deploying the RuleApp 20-2
Section 2. Getting the HTDS WSDL file from the deployed ruleset 20-4
2.1. Viewing the deployed ruleset 20-4
2.2. Retrieving the WSDL 20-6
2.3. Building and running the Axis client 20-6
2.4. Reviewing decision service performance 20-7
Section 3. Testing the decision service with REST 20-8
Section 4. Retrieving an OpenAPI description and calling the service in a generated client 20-11
4.1. Retrieving the YAML file 20-11
4.2. Generating the Java client 20-11
4.3. Adding a test to the Java client 20-12
4.4. Running the Java client test 20-13

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Trademarks

The reader should recognize that the following terms, which appear in the content of this training document, are official trademarks of IBM or other companies:

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

The following are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide:

Bluemix®	CICS®	Express®
IBM API Connect®	IBM Cloud™	ILOG®
Insight®	Notes®	Orchestrate®
Redbooks®	SPSS®	Tivoli®
WebSphere®	z/OS®	

Intel, Intel Xeon and Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

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

VMware is a registered trademark or trademark of VMware, Inc. or its subsidiaries in the United States and/or other jurisdictions.

Social® is a trademark or registered trademark of TWC Product and Technology, LLC, an IBM Company.

Other product and service names might be trademarks of IBM or other companies.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercises description

Exercise objectives

After completing the exercises, students should be able to:

- Design a business rule solution that is based on decision services, including the implementation model (or XOM) for rule execution, and the business version of that model (or BOM) for rule authoring
- Orchestrate the flow of rule execution through ruleflows and by understanding rule engine algorithms
- Support business rule authors by setting up the rule authoring environment, collaborating on rule vocabulary changes, and maintaining synchronization between the business and development environments
- Set up the test and simulation environment for business users who work in Business console
- Package and deploy rule artifacts and the XOM to a managed execution environment in Rule Execution Server
- Use launch configurations to run and debug the rulesets
- Deploy business rules as a web service, or hosted transparent decision service (HTDS), in Rule Execution Server
- Test and monitor ruleset execution, and audit decision traces

Exercise list

This course includes the following exercises:

- Exercise 1, "Operational Decision Manager in action"
- Exercise 2, "Setting up decision services"
- Exercise 3, "Modeling decisions"
- Exercise 4, "Working with the BOM"
- Exercise 5, "Refactoring"
- Exercise 6, "Working with ruleflows"
- Exercise 7, "Exploring action rules"
- Exercise 8, "Authoring action rules"
- Exercise 9, "Authoring decision tables"
- Exercise 10, "Working with static domains"
- Exercise 11, "Working with dynamic domains"
- Exercise 12, "Working with queries"
- Exercise 13, "Executing rules locally"

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

- Exercise 14, "Debugging a ruleset"
- Exercise 15, "Enabling rule validation"
- Exercise 16, "Managing deployment"
- Exercise 17, "Using Build Command to build RuleApps"
- Exercise 18, "Exploring the Rule Execution Server console"
- Exercise 19, "Auditing ruleset execution through Decision Warehouse"
- Exercise 20, "Executing rules as a hosted transparent decision service (HTDS)"

Exercise structure

The exercises can be categorized into these groups:

Overview

Exercise 1, "Operational Decision Manager in action"

The first exercise is an overview of Operational Decision Manager components for business rules, including Decision Center, Rule Execution Server, and Rule Designer. This lab sets the stage for all the other labs by introducing you to the workflow between ODM modules. During the course, you work mainly with Rule Designer and Rule Execution Server console, but you also do some exercises with the Decision Center tools.

Enabling business rule authoring and governance

- Exercise 2, "Setting up decision services"
- Exercise 3, "Modeling decisions"
- Exercise 15, "Enabling rule validation"

Rule authoring

- Exercise 6, "Working with ruleflows"
- Exercise 7, "Exploring action rules"
- Exercise 8, "Authoring action rules"
- Exercise 9, "Authoring decision tables"

Configuring rule vocabulary

- Exercise 4, "Working with the BOM"
- Exercise 5, "Refactoring"
- Exercise 10, "Working with static domains"
- Exercise 11, "Working with dynamic domains"

Deployment and ruleset execution

- Exercise 12, "Working with queries"
- Exercise 13, "Executing rules locally"
- Exercise 14, "Debugging a ruleset"
- Exercise 16, "Managing deployment"
- Exercise 17, "Using Build Command to build RuleApps"
- Exercise 18, "Exploring the Rule Execution Server console"
- Exercise 19, "Auditing ruleset execution through Decision Warehouse"
- Exercise 20, "Executing rules as a hosted transparent decision service (HTDS)"

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Most of the exercises are independent and do not require you to complete a previous exercise. Exercises that you do in Rule Designer have “start” files and “answer” (or “solution”) files. You can use the “answer” files when you run out of time or are unable to complete the exercise correctly. If necessary, the start and answer files make it possible to skip exercises or do them out of sequence. However, you are encouraged to complete them in the order in which they are presented. In the exercise instructions, you can check off the line before each step as you complete it to track your progress.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

General exercise information

This section provides general information about the exercises in this course. Review this section before starting the exercises.

User IDs and passwords

The following table lists user ID and password information for this course.

Entry point	User ID	Password
VMware image	`administrator`	`passw0rd`
Windows 2008 R2	`administrator`	`passw0rd`
Single-sign-on ID for ODM installation and user ID for WebSphere Application Server and Decision Server	`odm`	`odm`
Decision Center administrator	`rtsAdmin`	`rtsAdmin`
Decision Center business user	`rtsUser1`	`rtsUser1`
Decision Center configuration user	`rtsConfig`	`rtsConfig`
Decision Server administrator	`resAdmin`	`resAdmin`
Business console manager user	`Paul`	`Paul`
Business console rule author user	`Bea`	`Bea`
Business console test specialist user	`Abu`	`Abu`

How to follow the exercise instructions

Exercise structure

Each exercise is divided into sections with a series of numbered steps and lettered substeps:

- The numbered steps (1, 2, 3) represent actions to be done.
- The lettered substeps (a, b, c) provide detailed guidance on how to complete the action.

Information

If you already understand how to do the action in the numbered step, you can skip the specific guidance in the lettered substeps.

The following example comes from Exercise 1 of this course.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Example

Excerpt from Exercise 1

__ 1. Edit the rule and change the debt-to-income ratio from 0.3 to 0.5.

__ a. Click **Edit** to open the rule editor.

__ b. In the **if** part of the rule, change `0.3` to: `0.5`

In this example, the numbered instructions prompt you to edit a rule. The “a” and “b” substeps provide specific guidance on how to edit the rule.

Text highlighting in exercises

Different text styles indicate various elements in the exercises.

Words that are highlighted in **bold** represent GUI items that you interact with, such as:

- Menu items
- Field names
- Icons

Words that are highlighted with a `fixed font` include the following items:

- Text that you type or enter as a value
- System messages
- Directory paths
- Code

Tracking your progress

As shown in the example step, you can see that an underscore precedes each numbered step and lettered substep.

You are encouraged to use these markers to track your progress by checking off each step as you complete it. Tracking your progress in this manner might be useful if you are interrupted while working on an exercise.

Required exercise sections

Most exercises include required sections that should always be completed. It might be necessary to complete these sections before you can start subsequent exercises.

Dependencies between exercises are listed in the exercise introduction.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Optional exercise sections

Some exercises might also include optional sections that you can complete when you have sufficient time and want an extra challenge.

File references for exercises

Exercise steps contain references to files or projects to open or import. Two directories are used in these references:

- *<InstallDir>*: This directory is the IBM Operational Decision Manager installation directory.
- *<LabfilesDir>*: This directory contains the files that are required during demonstrations, exercises, and the workshop steps, such as samples of code that you can copy and paste.

If you are using the VMware image that is provided with this course:

- *<InstallDir>* is: `C:\IBM\ODM810`

  This folder is the default IBM Operational Decision Manager installation directory on Windows.
- *<LabfilesDir>* is: `C:\labfiles`

If you are not using the VMware image that is provided with this course, ask the installer of your environment, or your instructor, where to find the *<InstallDir>* and *<LabfilesDir>* directories.

Stop

Make sure that you identify the *<InstallDir>* and *<LabfilesDir>* directories before you proceed with the exercises in this course.

Projects for exercises

Most of the exercises for this course are done in Rule Designer, which uses the Rule perspective of Eclipse.

The exercise projects are provided for you to import into Eclipse by using the Import wizard or the Samples Console perspective. For most of the exercises, you use the Samples console perspective, as described here.

To open Rule Designer, you use the **Rule Designer 8.10.0** shortcut on the Taskbar or on the Windows **Start** menu.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

When prompted for a workspace, you can type the path directly in the Workspace Launcher, for example:

`C:\labfiles\workspaces\myWorkspace`

When you type a path, an empty workspace is created in the Rule perspective.

To start an exercise, you use the Samples Console perspective.

You expand the **Rule Designer > Training** node to see a list of all exercises for this course, and click **Import projects** to import the projects for the exercise.

Both “start” and “solution” projects are provided for most exercises.

© Copyright IBM Corp. 2019
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

When you import the projects, Eclipse automatically switches to the Rule perspective. To give yourself more area to work in, you can close the Help view by clicking the **X**.

Sample server port

This course uses the default installation of Operational Decision Manager where Decision Center and Rule Execution Server are hosted on the sample server of Operational Decision Manager.

With the default installation, you use the following URLs to access the console of these two modules through a web browser:

- `http://localhost:`*`PORT`*`/decisioncenter`: to access the Decision Center Business console
- `http://localhost:`*`PORT`*`/teamserver`: to access the Decision Center Enterprise console
- `http://localhost:`*`PORT`*`/res`: to access the Rule Execution Server console

`PORT` is the required port. The port might be different in your environment.

If **you are using the VMware** image that is provided with this course:

- The value of *`PORT`* is: **9090**

  This value is the default port with the default installation of Operational Decision Manager. This course assumes that this default value of *`PORT`* is used, and uses the following URLs:

  - `http://localhost:`**`9090`**`/decisioncenter`: to access the Decision Center Business console
  - `http://localhost:`**`9090`**`/teamserver`: to access the Decision Center Enterprise console
  - `http://localhost:`**`9090`**`/res`: to access the Rule Execution Server console

Stop

If you are not using the VMware image that is provided with this course, make sure that you identify the value of *`PORT`* before you proceed with the exercises in this course.

Using the product documentation

The product documentation is installed locally on the VMware image that is provided with this course.

To access the local documentation while working in Rule Designer, you must first start it. From the Windows **Start** menu, click the down arrow to get to the **Apps** page, and in the **IBM Operational Decision Manager V8.10.0** section, click **Start Knowledge Center (local)**.

You can also access the product documentation from a web browser at this web address:

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

www.ibm.com/support/knowledgecenter/en/SSQP76_8.10.x/welcome/kc_welcome_odmV.html

Information

If you are not using the VMware image that is provided with this course, you must either install the local help as described in the product documentation, or use the online version.

Checking for course corrections

Important

Online course material updates might exist for this course. To check for updates, see the Instructor wiki at ibm.biz/CloudEduCourses.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 1. Operational Decision Manager in action

Estimated time

01:30

Overview

In this exercise, you see how the Operational Decision Manager modules work together to provide comprehensive decision management across the business and development environments.

Objectives

After completing this exercise, you should be able to:

- Explain the general workflow in Operational Decision Manager for working with business rule projects
- Identify the Operational Decision Manager tasks that apply to your role in your organization

Introduction

The exercise is based on a fictional financial institution, Miniloan, which provides online quotations for loan requests.

To see how Operational Decision Manager facilitates collaboration between business and IT teams, you take on business and technical roles to perform these tasks:

- Section 1, "Running the Miniloan web application"
- Section 2, "Managing business rules in Decision Center"
- Section 3, "Validating rule changes"
- Section 4, "Deploying updated rules from Decision Center to Rule Execution Server"
- Section 5, "Monitoring the deployed rules in Rule Execution Server"
- Section 6, "Viewing the effects in the Miniloan application"
- Section 7, "Exploring the development environments in Rule Designer"
- Section 8, "Shutting down the modules"

This exercise provides you with an introduction to the different Operational Decision Manager modules, and how they work together. Both IT and business tasks are included in this exercise to demonstrate the collaboration that occurs between the technical and business organizations.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The exercise workflow is shown here.

You start by running the Miniloan enterprise application. Then, you work in the business user environment by using Decision Center to author, validate, and deploy rules. Finally, you move to the IT user environment to monitor the rule changes, see how the changes affect the application. By completing tasks in the business and IT environments, you can see how both business and IT work together within Operational Decision Manager to manage business rules.

Requirements

This exercise has no requirements.

User accounts

Type	User ID	Password
Access to VMware image	`administrator`	`passw0rd`
Decision Center administrator	`rtsAdmin`	`rtsAdmin`
Business user	`rtsUser1`	`rtsUser1`
Rule Execution Server administrator	`resAdmin`	`resAdmin`

Important

The exercises in this course use a set of lab files that are installed in the product installation directory:

```
<InstallDir>\studio\training
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The default directory for `<InstallDir>` is `C:\IBM\ODM810`. If you are not using the provided lab environment for this course, make sure that you know the location of `<InstallDir>`.

Additional lab support files are stored in the `C:\labfiles` directory.

The exercises point you to the lab files as you need them.

Note

For IBM ODM on Cloud users

This exercise uses some features that are not supported in IBM ODM on Cloud. For ODM on Cloud, follow the instructions that are provided in the IBM ODM on Cloud documentation:

www.ibm.com/support/knowledgecenter/SS7J8H/welcome/kc_welcome_cloud.html

The documentation points you to software downloads and projects that are available on GitHub:

https://github.com/ODMDev/odm-on-cloud-getting-started

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Running the Miniloan web application

Before you explore the tools, you first look at how rules affect the user application.

Scenario

Joe is planning to buy a house and wants a loan for $500,000. To find out whether he is eligible, he goes to the Miniloan website.

1.1. Setting up your environment

This exercise uses the sample server that is provided with Operational Decision Manager.

For IBM ODM on Cloud users

To set up your environment in ODM on Cloud, follow the directions in the “First steps in IBM ODM on Cloud” tutorial in the IBM Knowledge Center for IBM ODM on Cloud.

In the “First steps” tutorial, you work with the `Miniloan Service` decision service and install the `miniloan-webapp` web application.

__ 1. Start the sample server.

__ a. Click the **Start** menu, and click the **Start Sample Server** shortcut.

__ b. Wait until the server is started.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The command window shows server trace messages as the server starts. A feedback message indicates when the server start is complete.

__ c. After the server is started, click in the command window, and press any key on your keyboard to close it.

1.2. Running the application

__ 1. Open a web browser and enter the following URL:

`http://localhost:9090/miniloan-server`

The Miniloan application opens.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Scenario

Joe earns $80,000 a year, has a credit score of 600, and would like to take a loan for $500,000. He intends to repay the loan over a 20-year period.

Is Joe eligible for this loan?

__ 2. Click **Validate Loan**.

Based on the information that Joe submitted, the rule engine returns a decision to reject the loan.

Rejected

```
{
    "decision": "Your loan is rejected",
    "messages": "[The debt-to-income ratio is too big.]"
}
```

Scenario

Joe's loan request is rejected. Miniloan responds by reporting that the ratio of debt to income is too high.

__ 3. You can either close the browser window or, if you prefer, you can leave it open because you come back to this application later in the exercise.

As you can see, decisions that are made by an application affect both your business and the lives of ordinary people. For that reason, it is important that experts in business policy, not just the IT team, can see and maintain the rules that are implemented. Instead of hardcoding the rules into the application, and requesting technical developers to make rule changes in the code, Operational Decision Manager separates the decision logic from the code. It empowers business users to access and manage the rules through Decision Center. Rules are stored in the Decision Center repository, and can be shared among various users through permission and locking mechanisms.

Next, you see how easily business policies can be updated in Decision Center.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Managing business rules in Decision Center

Scenario

Several customers, including Joe, complained about having their loan requests rejected. You do not change the rules merely so that Joe can get a house, but a significant number of loan rejections might be an indication of errors in the rules. Some types of errors can be corrected early on, while in other situations, modifying rules might be a maintenance issue, such as ongoing rule adjustment for promotions or for different types of customers.

Miniloan decided to review its eligibility policies about debt-to-income ratios. Miniloan uses Decision Center to store and manage business rules, so next, you open Decision Center to see which rule to change to solve the loan rejection issue.

Decision Center supports auditability to trace the who, what, where, when, and why of rule updates. The rule administrators define roles and permissions in Decision Center to control access to the rules.

For this step, you sign in with the business user role of Rule Author to review the eligibility rules. You use Decision Center Business console, which is a collaborative rule authoring environment, to edit the rules in this section.

2.1. Editing rules in Decision Center Business console

__ 1. Open the Business console.

__ a. From the **Start** menu, click the **Decision Center Business console** shortcut.

Information

You can also open the Business console by entering the following URL in a browser:

`http://localhost:9090/decisioncenter`

Make sure that you use the correct URL and port for your environment.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. If the Privacy message opens, click **Agree and Proceed**.

Troubleshooting

If you receive the Privacy message again, either later in this exercise or in other exercises, you can click **Agree and Proceed**.

__ c. Sign in as a business user with the following values for the **Username** and **Password** fields.

- **Username**: `rtsUser1`
- **Password**: `rtsUser1`

Note

These values are case-sensitive.

__ d. Click **Log in**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

For IBM ODM on Cloud users

You start the Business console by logging in to the ODM on Cloud User Portal, and clicking **Launch** under **Decision Center Business console**. The Business console is configured to open to the URL of your ODM on Cloud instance, so you do not need to enter a URL or port number.

Note

If you see a message to store your password, you can click **Not for this site**.

__ 2. On the Decision Center menu, click **Library**.

__ 3. On the Decision Services page, you see the list of decision services that you can work on in Decision Center.

__ 4. Click the **Miniloan Service** box (in the empty white space) to expand the box, and then click **main**.

The **Decision Artifacts** tab opens, and in the navigator pane, you see two rule folders: **eligibility** and **validation**. These folders contain the rule artifacts for Miniloan Service.

Information

By default, the Business console shows only rule artifacts, such as rules and decision tables. However, you can select various decision artifact types for display in the navigator pane.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. View all of the decision artifacts that are associated with Miniloan Service.

__ a. On the **Decision Artifacts** tab, click **Types**.

__ b. In the Types menu, select **All Types**, and then click **Apply**.

__ c. The navigator pane now shows all decision artifact types of Miniloan Service.

The decision artifacts in Miniloan Service include:

- **Operations** (artifact that defines input and output parameters for decision services)
- **Resources** (artifact that stores decision service resources, such as the execution object model, or XOM)
- **miniloan** (ruleflow)

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

- **Miniloan ServiceParameters** (artifact that defines the input and output parameters for a decision service)
- **eligibility** (folder that contains rules)
- **validation** (folder that contains rules

__ 6. View the rules.

__ a. Click the **eligibility** folder to see the rules that it contains.

__ b. Click **minimum income** to review the minimum income policy, which is:

```
if
   the yearly repayment of 'the loan' is more than the yearly income of
   'the borrower' * 0.3
then
   add "Too big Debt-To-income ratio" to the messages of 'the
   loan' ;
   reject 'the loan' ;
```

Scenario

The `minimum income` rule implements the debt-to-income ratio policy that caused the loan rejection for Joe. Currently, if the debt is more than 30% of the borrower's income, the loan cannot be approved.

Miniloan business analysts decided to update their policy and change the ratio from 30% to 50%. Now, if the loan causes the borrower's debt to be up to 50% of the borrower's income, the loan can be approved.

__ 7. Edit the rule and change the debt-to-income ratio from 0.3 to 0.5.

__ a. Click **Edit this rule** (the pencil icon) to open the rule editor.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. In the ***if*** part of the rule, change `0.3` to: `0.5`

```
if
  the yearly repayment of 'the loan' is more than the yearly income of 'the borrower' * 0.5
```

__ 8. Add a condition to the rule that tests whether the yearly income of the borrower is less than 500,000.

__ a. Place the cursor after `0.5`, and press Enter to create a line.

```
if
  the yearly repayment of 'the loan' is more than the yearly income of 'the borrower' * 0.5
|
then
```

__ b. Enter the following condition text into the editor:

```
and the yearly income of 'the borrower' is less than 500000
```

Note

As you type, the automatic completion menu might suggest terms that you can use to build your rule. You can also click these completion menu items to build the condition.

```
if
  the yearly repayment of 'the loan' is more than the yearly income of 'the borrower' * 0.5
and the yearly income of 'the borrower' is less than 500000
then
```

__ 9. When you are finished with your changes, save your work and add a comment to provide version information for your changes.

__ a. Click **Save**.

__ b. In the **Create New Version** field, type a comment, such as:

```
Reject loans for low to mid-range income with debt-to-income ratio above 50%
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. Click **Create New Version**.

The new rule details are shown in the “minimum income (v1.1)” window.

You can now check the history of the rule that you modified, and compare the differences between the two versions.

2.2. Reviewing the rule history

__ 1. Compare versions 1.0 and 1.1 of the `minimum income` rule.

__ a. On the toolbar, click **Compare**.

The “Compare minimum income Versions” window lists all versions of the rule. Since the rule has only two versions, both versions are selected by default.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. Click **Compare**.

The two rule versions are shown side by side, with the newer version on the right. The summary lists the differences between the two versions.

A red triangle indicates modified values, and additions to the rule are shown in purple, underlined text.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. Return to the `Miniloan Service` decision service and open the **eligibility** folder.

__ a. In the upper-right corner of the browser window, click the **main** breadcrumb.

__ b. In the left pane, click the **eligibility** folder to list its contents.

2.3. Changing the credit score requirements

Scenario

After reviewing the eligibility business policies, Miniloan decided to adjust its credit score requirements. Borrowers with a debt-to-income ratio that falls within the range of 45 and 50 are rejected when their credit score is less than 500. The credit score policies are implemented in a decision table.

__ 1. Edit the `repayment and score` decision table to modify the credit score values.

__ a. In the **eligibility** folder, hover your mouse on **repayment and score** and click the **Edit** icon to open the `repayment and score` decision table.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. In the “repayment and score” window, leave the default row order set to **Automatic** and click **OK**.

__ c. In row 5, double-click the cell in the credit score column with the value: `[0; 600[`

__ d. Change `600` to `500` and press Enter.

	debt to income		credit score	
	min	max	min	max
1	0 %	30 %	0	200
2	0 %	30 %	200	800
3	30 %	45 %	0	400
4	30 %	45 %	400	800
5	45 %	50 %	0	500
6	45 %	50 %	600	800
7	≥ 50 %		0	800

__ e. Click **Save**.

__ f. In the Create New Version window, add a comment about the update that you made, such as:

```
Row 5 value changed from 600 to 500
```

__ g. Click **Create New Version**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

After you create the version, your changes are saved and a gap error is detected in the table. The problem cells are highlighted with a gold triangle in the lower-left corner.

__ 2. Edit the table to resolve the gap error.

__ a. Click **Edit** to open the rule editor.

__ b. In row 6, select the cell in the credit score column with the value `[600; 800[`.

__ c. Change the value from `600` to `500`, and press Enter.

__ d. Click **Save**.

__ e. In the Create New Version window, click **Create New Version**.

The table no longer shows any errors.

	debt to income		credit score		message	loan a
	min	max	min	max		
1	0 %	30 %	0	200	debt-to-income too high compared to credit score	
2	0 %	30 %	200	800	⊘	
3	30 %	45 %	0	400	debt-to-income too high compared to credit score	
4	30 %	45 %	400	800	⊘	
5	45 %	50 %	0	500	debt-to-income too high compared to credit score	
6	45 %	50 %	500	800	⊘	
7	≥ 50 %		0	800	debt-to-income too high compared to credit score	

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. Click the **Stream** tab on the right view and notice that the rule stream was automatically updated. By clicking the **plus** icon , you can see the details of the version that you created.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Validating rule changes

Scenario

Before making the updated business rules available to the Miniloan web application, the policy manager wants to make sure that the rules behave as expected in a test environment.

The business analyst works with the development team to identify what must be tested and to create the test scenarios. The scenarios are stored, along with their expected results, in a Microsoft Excel spreadsheet.

In this section, you run tests and simulations from the Decision Center Business console.

To run a scenario in Decision Center, you must create a test suite. For this exercise, the scenario file for the test suite is already created for you. The `miniloan-test.xls` scenario file is in this directory:

```
<InstallDir>\gettingstarted\BusinessConsole
```

The scenario file contains two scenarios with their corresponding expected results. The scenarios are listed on the **Scenarios** tab, and the anticipated results on the **Expected Results** tab.

Running the test

__ 1. Click the **main** breadcrumb to return to the Miniloan Service artifacts.

__ 2. Click the **Tests** tab.

A prepared test suite, `Miniloan test suite`, is available to test all the rules in the decision service. You can run this test to see whether your changes affect the outcome.

__ 3. Hover your mouse over **Miniloan test suite**, and click the **Run** icon.

__ 4. In the Run Test Suite window, click **OK** to close the window.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. Wait for the test to complete and then click the report link to open the report.

Because you changed the rules, you should not expect a 100% success rate. The results show that 1 test failed.

__ 6. In the upper right, click **Close** to close the report.

Information

Later in this course, you learn how to define test suites and interpret the results.

__ 7. Log out of the Business console.

__ a. Click the down arrow next to your user name in the upper-right corner of the browser window.

__ b. Click **Log out**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 4. Deploying updated rules from Decision Center to Rule Execution Server

Scenario

The tested rules are now ready to be deployed to the Miniloan web application.

Users with the appropriate access rights can deploy rulesets directly from the Decision Center. You can also deploy rulesets from Rule Designer.

In this task, you take the role of Rule Administrator to define and deploy the updated rules. The rules are packaged as a RuleApp.

For IBM ODM on Cloud users

For this section, make sure that you run the steps on a Development Rule Execution Server. For information about the ODM on Cloud roles and environment, see:

www.ibm.com/support/knowledgecenter/SS7J8H/com.ibm.odm.cloud.admin/topics/con_work_env.html

Information

A RuleApp is a container for rulesets. The RuleApp is deployed to Rule Execution Server, making the rulesets available for execution.

4.1. Deploying the decision service

__ 1. Sign in to the Business console as a user with administrative privileges:

- **User name**: `rtsAdmin`
- **Password**: `rtsAdmin`

__ 2. Click **Library** and reopen the **main** branch of the Miniloan Service decision service.

__ 3. On the **Decision Artifacts** tab, enable all types of decision artifacts for display in the navigation pane by clicking the **Types** tab, selecting **All Types**, and clicking **Apply**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. In the navigation pane, click **Operations**, and click **Miniloan ServiceOperation**.

__ 5. In the upper-right area of the page, click the **Open Decision Operation View** icon.

__ 6. In the upper right, click **Edit** (the pencil icon).

__ 7. Change the value in the **Ruleset name** field to: `my_operation`

__ 8. Save your changes.

__ a. Click **Save** to exit the editor.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. In the Create New Version window, click **Create New Version**.

__ 9. Click the **main** breadcrumb to return to the main Decision Service page.

__ 10. Click the **Deployments** tab and make sure that you are on the Configurations page.

For IBM ODM on Cloud users

For this section, you use the Development configuration with the Development Environment server as the target.

__ 11. Click the **Miniloan** deployment configuration.

__ 12. In the upper-right area of the page, click the **Edit** icon.

__ 13. On the **General** tab, edit the configuration by changing the **RuleApp name** field to: `my_deployment`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 14. Click the **Targets** tab and make sure that **Decision Service Execution** is selected.

__ 15. Click **Save** and click **Create New Version**.

__ 16. In the toolbar, click the **Deploy** icon.

__ 17. In the “Deploy main” window, click **Deploy**.

__ 18. In the “Deployment status” window, click **OK**.

__ 19. After deployment is finished, click the **Report** link.

__ 20. View the report details, and when you are finished, click **Close**.

__ 21. Log out of the Business console and close it.

Information

Later in this course, you learn how to define deployment configurations, define target servers, and manage deployment in Rule Designer and Rule Execution Server console.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 5. Monitoring the deployed rules in Rule Execution Server

You can now go to the Rule Execution Server console and see whether the RuleApp was properly deployed.

Rule Execution Server 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 that are associated with the execution of your rules.

Viewing the deployed RuleApp

__ 1. Open Rule Execution Server by clicking the Rule Execution Server shortcut on the Start menu or by entering the following URL in a browser:

`http://localhost:9090/res`

Information

You can also open the Rule Execution Server by using the **Start** menu shortcut.

For IBM ODM on Cloud users

Instead of going through your operating system, you start the Rule Execution Server console by logging in to the User Portal, then in the Development Environment, clicking **Launch** under **Rule Execution Server console**.

The Rule Execution Server console is configured to open to the URL of your ODM on Cloud instance, so you do not need to enter a URL or port number.

__ 2. Sign in to the Rule Execution Server console with the following credentials.

- **User name**: `resAdmin`
- **Password**: `resAdmin`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. Click the **Explorer** tab, and in the Navigator pane, expand **RuleApps > my_deployment**.

You see the ruleset that you deployed.

__ 4. In the RuleApps list, click **my_deployment** to see the details in the RuleApp view.

__ 5. In the list of rulesets, click **my_operation** to open it in the Ruleset view.

Notice that the status of the ruleset is **enabled**, which means that your newly deployed ruleset can be executed. The next time that the application calls for a decision, this ruleset is considered the most recent version, and will be used.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 6. Scroll down and click the various options, such as Show Managed URIs and **Show Properties**.

These properties are just some of the details that are available in Rule Execution Server.

Information

Later in this course, you work with Rule Execution Server console.

__ 7. Sign out and close the Rule Execution Server console window.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 6. Viewing the effects in the Miniloan application

Scenario

After testing and deploying the updated business rules to the Miniloan application, you can now see how the business policy changes that you made are reflected in the Miniloan application. You can also see how they affect the original request from Joe.

To run the Miniloan application with the updated ruleset:

__ 1. Reopen the Miniloan application in a web browser window.

`http://localhost:9090/miniloan-server`

__ 2. Click **Execution**, and select **Use Rules,** and then click **Validate Loan**.

The loan is now approved because of the deployed rule changes.

Notice that the **Information** section lists the decision table row that you edited as the rule that was used for this loan decision.

__ 3. Close the browser.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 7. Exploring the development environments in Rule Designer

For IBM ODM on Cloud users

The steps in this section do not apply to ODM on Cloud.

For more information about installing and using Rule Designer with ODM on Cloud, see the documentation.

www.ibm.com/support/knowledgecenter/SS7J8H/com.ibm.odm.cloud.tutorials/topics/tsk_develop_rule_apps.html

7.1. Opening Rule Designer

__ 1. On the taskbar, click the **Rule Designer** shortcut.

You can also click the **Rule Designer 8.10.0** shortcut on the **Start** menu.

__ 2. When prompted for a workspace, type the following path and click **Launch**:

`C:\labfiles\workspaces\intro`

__ 3. Close the Welcome view.

By default, the Rule Designer perspective opens in Eclipse.

__ 4. Switch to the Samples Console and import the project for this exercise.

__ a. In the upper-right area of the window, click the **Open Perspective** icon to switch from the Rule perspective.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. In the Open Perspective list, select **Samples Console**, and click **OK**.

__ c. In the **Rule Designer** section, expand **Training > Ex 01: Operational Decision Manager in action > 01 - start**, and click **Import projects**.

The Eclipse perspective automatically switches back to the Rule perspective.

__ d. Close the Help view by clicking the **X**.

Hint

To give yourself more working area, you can close the views (such as Help or Welcome views) that open by default in Eclipse.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.2. Taking a quick tour of Rule Designer

In Eclipse, the window for the Rule Designer development environment is called the *Rule perspective*.

__ 1. Look at the various windows and panes that are open in the Rule perspective, including these views:

- Rule Explorer: View and access the contents of the projects that you are working with
- Rule Project Map view: Use the Decision Service Map to guide you through development of a decision service

The Rule Explorer contains these projects:

- **Miniloan Service**: The main rule project that contains Miniloan business rules.
- **miniloan-xom**: The Java project of the Miniloan application that is composed of the **borrower** and the **loan** Java classes.

__ 2. In Rule Explorer, expand **Miniloan Service > rules** to see the rule artifacts.

__ a. Expand the **eligibility** and **validation** folders to see the rules that they contain, and notice that these folders are the same folders that you saw in Decision Center.

__ b. In the **eligibility** folder, double-click **repayment and score** to open the decision table.

__ c. Notice that the values in the **credit score** columns for row 5 and 6 do not reflect the changes that you made in Decision Center, where you changed 600 to 500.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Close the decision table.

__ 3. Expand the **bom > miniloan > miniloan** folder to see **Borrower** and **Loan**.

The **bom** folder contains all the vocabulary that is used in the rules.

__ a. Expand **Borrower** and **Loan** to see their members in Rule Explorer.

__ b. Double-click **Borrower** to open it in the BOM editor and view its properties and members. For example, here you see that the Borrower class has a name, a credit score and a yearly income.

__ 4. Open the **Outline** view (under Rule Explorer), and note the connection between this view and the contents of the **bom** folder.

The **Outline** view lists all the members of the BOM.

__ a. In the Outline view, expand **Borrower** and **Loan** to see their members.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Click the **Vocabulary** tab and expand **borrower** to see the vocabulary expressions that are assigned to the BOM members.

__ c. Notice the similarities and differences between the expressions that are listed in the **Vocabulary** view and the **Outline** view.

__ 5. In Rule Explorer, go back to the **rules** folder to see how the rules use the vocabulary.

__ a. Go back to the **eligibility** folder and double-click **minimum income** to open this rule in the rule editor.

Questions

Does the wording in the rule seem to match the **Outline** view or the **Vocabulary** view?

Remember this comparison as a first look at the connection between rules and vocabulary.

__ b. Notice the rule statement. This rule also does not include your edits from Decision Center.

To get the most recent version of the rules from Decision Center, you must synchronize the Rule Designer and Decision Center environments.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.3. Synchronizing Rule Designer and Decision Center environments

__ 1. In Rule Explorer, right-click **Miniloan Service**, and click **Decision Center > Connect**.

__ 2. Enter the following values in the Decision Center Configuration window fields.

- **URL**: `http://localhost:9090/teamserver`
- **User name**: `rtsAdmin`
- **Password**: `rtsAdmin`
- **Data source**: (Leave the field empty)

__ 3. Click **Connect**.

When the connection is established, the warning message disappears and the Project configuration area becomes active.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. Click **Next**.

__ 5. In the **Select a branch or release** field, choose **main** and click **Finish**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Troubleshooting

If you are prompted about password recovery for Secure Storage, click **No**.

__ 6. When the window prompts you to change to Team Synchronizing perspective, click **Yes**.

After synchronization is complete, the Synchronize view shows the `miniloan-rules` rule project, and indicates that changes were made to rules in that rule project.

__ 7. In the Synchronize view, expand the `Miniloan Service - main` rule project to see which resources and rules were changed.

__ a. In particular, expand the **rules/eligibility** folder to see the rules that you updated in Decision Center.

- `minimum income.brl`
- `repayment and score.dta`

__ b. Double-click either of these rules to open it in the Text Compare view and see the differences that now require synchronization.

__ c. Close the Text Compare view for the rule that you selected.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Right-click **Miniloan Service - main** and click **Update**.

After the update is complete, the rules in the **rules/eligibility** folder are removed from the Synchronize view, which means that your local Rule Designer workspace is correctly updated with the edited rules from Decision Center.

During synchronization, not all artifacts require synchronization. Some artifacts are generated, such as artifacts in the **output** folder, and should not be synchronized and stored.

__ 8. Return to the Rule perspective.

Hint

To return to the Rule perspective, click **Rule** at the upper-right corner of the perspective.

__ 9. If the minimum income rule is closed, reopen it by expanding the **miniloan-rules > rules > eligibility** folder in Rule Explorer, and double-clicking **minimum income**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The condition statement should now match your edits in Decision Center and state:

```
if
  the yearly repayment of 'the loan' is more than the yearly income of 'the
borrower' * 0.5
  and the yearly income of 'the borrower' is less than 500000
```

__ 10. Close Rule Designer, and confirm closing when prompted to confirm.

Information

Later in this course, you work more with the Rule Designer synchronization tools.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 8. Shutting down the modules

For IBM ODM on Cloud users

The steps in this section do not apply to ODM on Cloud.

These steps describe how to shut down the sample server and modules.

__ 1. Close all browsers for the Miniloan application, Rule Execution Server, and Decision Center consoles.

__ 2. Close Rule Designer and click **OK** when prompted to confirm your exit from Eclipse.

__ 3. Stop the sample server.

  __ a. From the Windows **Start** menu, click the **Stop Sample Server** shortcut.

  __ b. After the sample server is stopped, press any key to close the terminal window.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

This exercise showed the general workflow of Operational Decision Manager for business rules.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 2. Setting up decision services

Estimated time

01:30

Overview

In this exercise, you learn how to set up decision services in Rule Designer.

Objectives

After completing this exercise, you should be able to:

- Create main and standard decision service projects
- Set up the decision service to reference the execution object model (XOM)
- Generate a business object model (BOM) and a default vocabulary
- Create a decision operation
- Define ruleset variables and ruleset parameters
- Create rule packages
- Synchronize decision services with Decision Center

Introduction

In this exercise, you use Rule Designer to create the initial elements that are required to set up a business rule application. You set up the decision service structure, define the business and execution object models, and create a decision operation, which includes variables and parameters. Parameters are used to pass objects between the external application and the rule engine.

You also outline the basic structure for organizing rule artifacts by creating rule packages, which are also called folders in the business user environment.

This exercise includes the following sections:

- Section 1, "Creating a standard rule project"
- Section 2, "Importing the XOM and setting up the BOM"
- Section 3, "Creating the main rule project for the decision service"
- Section 4, "Creating and defining the decision operation"
- Section 5, "Creating rule packages"
- Section 6, "Publishing from Rule Designer to Decision Center"
- Section 7, "Deleting a decision service from Decision Center"
- Section 8, "Importing a decision service from Decision Center"

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Requirements

This exercise uses the following files, which are installed in the `<InstallDir>\studio\training` directory:

- Start project: `Dev 02 - Decision services\01-start`
- Solution project: `Dev 02 - Decision services\02-answer`

You can use the solution project as described in "Exercise review and wrap-up" on page 2-43.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Creating a standard rule project

In this section of the exercise, you set up the environment for the exercise and create a standard rule project for the decision service.

1.1. Setting up your environment for this exercise

__ 1. Open Rule Designer by clicking the **Rule Designer** shortcut on the taskbar.

__ 2. In the Workspace Launcher window, create a workspace.

__ a. In the **Workspace** field, type a workspace name:
`<LabfilesDir>\workspaces\decision_service`

By default, `<LabfilesDir>` is the `C:\labfiles` directory.

__ b. Make sure that the **Use this as the default and do not ask again** check box is *not* selected.

__ c. Click **Launch**.

The `<LabfilesDir>\workspaces\decision_service` workspace is created.

__ 3. Close the Welcome view.

The Rule Designer workspace opens and you see the Decision Service Map in the lower part of the workspace, in the Rule Project Map view. The steps for setting up a decision service are outlined in the Decision Service Map tasks.

- **Set up a Decision Service**
- **Define decision operation**
- **Deploy and Integrate**

In this exercise, you go through these steps to create and deploy a decision service.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Hint

You can maximize the view and reset it to the default size and location by using the icons in the upper-right corner of the view.

If you close the Rule Project Map view, you can open it by clicking **Window > Show View > Rule Project Map**.

1.2. Creating a standard rule project with a BOM

In this section of the exercise, you create an empty standard rule project with a BOM in preparation for the next section of the exercise, where you import the XOM and create the BOM.

The rule project that you create in this section eventually stores the BOM for the decision service.

__ 1. Go to the **Set up a Decision Service** part of the Decision Service Map, and click **Create standard rule project**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. In the **Decision Service Rule Projects** section of the New Rule Project window, click **Rule Project with a BOM**, and click **Next**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. In the **Project name** field, type a name such as: mybom

__ 4. Click **Finish**.

The **mybom** folder is now in the Rule Explorer pane.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Importing the XOM and setting up the BOM

2.1. Importing the execution object model (XOM)

One of the first tasks when designing your decision service is to import a XOM into your project.

By importing a XOM, you create a reference between your rule project and a XOM project in your workspace. To be able to create such a reference, you must first have the XOM in your Rule Designer workspace.

Information

The XOM is the code implementation of the model against which your business rules are executed. For this exercise, the XOM is provided for you. You learn more about XOMs later in this course.

For this exercise, the required XOM is the `loan-xom` Java project and is provided for you.

__ 1. In the upper-right area of the Rule perspective, click the **Open Perspective** icon.

__ 2. Click **Samples Console** and then click **OK**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. In the “Samples and Tutorials” page, go to the Rule Designer section and expand **Training**.

__ 4. Expand **Ex 02: Setting up decision services**, and under **01-start**, click **Import projects**.

Rule Designer automatically imports the content of the selected project into your workspace. In this case, Rule Designer imports the `loan-xom` Java project.

The `loan-xom` Java project is now available in Rule Explorer.

__ 5. After the workspace builds, close the Help view.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Important

For each exercise, you import projects into your workspace. The procedure for importing is also available at the beginning of this document, in "Projects for exercises" on page xv.

2.2. Creating a reference to the XOM in the rule project

Now that you have a XOM in your workspace, use the Decision Service Map to import it into your decision service.

__ 1. Make sure that the `mybom` project is selected in the Rule Explorer.

__ 2. In the **Set up a Decision Service** part of the Decision Service Map, click the **Import XOM** link to import the XOM into your decision service.

__ 3. On the Import XOM page, select **Java execution object model**, and click **OK**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. In the “Properties for mybom” window, under **Java Execution Object Model**, select the **loan-xom** check box, and click **Apply and Close**.

By selecting **loan-xom**, the `mybom` project now references the `loan-xom` Java project.

You can expand the `loan-xom` project to view the Java code implementation.

2.3. Creating the BOM

In the **Set up a Decision Service** part of the Decision Service Map, the **Create BOM** link is enabled.

This link indicates that the next required task in designing your ruleset is to define the BOM and the vocabulary that you and business users need to author the business rules.

You now use this link to set up the BOM in your rule project.

__ 1. In the Rule Explorer, make sure that the **mybom** project is selected.

__ 2. In the **Set up a Decision Service** task, click the **Create BOM** link to create a BOM entry.

The **New BOM Entry** wizard opens.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Information

BOM entries are the building blocks of the BOM. Each BOM entry defines a set of business elements in the BOM. You learn more about BOMs and BOM entries later in this course.

__ 3. Make sure that the **Create a BOM entry from a XOM** option is selected, keep the default values for the other fields, and click **Next**.

__ 4. On the BOM Entry page, click **Browse XOM**.

© Copyright IBM Corp. 2019
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. In the Browse XOM window, select **platform:/loan-xom**, and click **OK**.

__ 6. Make sure that your BOM entry fully reflects the classes and methods in the XOM as follows:

__ a. Expand **training_loan** and select all classes by clicking **Select All**.

__ b. Make sure that the **Load getters and setters as attributes** check box is selected.

This step ensures that all Java classes, attributes, and methods in the XOM are mapped to or have a corresponding BOM member.

__ c. Click **Next**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The BOM Verbalization page opens.

__ 7. Click **Finish** to accept the default verbalization.

Information

Verbalization attaches natural-language terms to the BOM elements to create the vocabulary for your business rules. You learn more about verbalization later in this course.

In your **mybom** project, the BOM entry that is called `model` is now available with a default vocabulary. You can view it in the Rule Explorer by going to **mybom > bom > model**.

2.4. Exploring the BOM

Next, you explore your BOM entry to see how the XOM elements are translated into the BOM entry.

__ 1. In the `mybom` project, expand **bom > model > training_loan**.

The BOM entry contains a series of classes, including the `Borrower` class.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. Double-click the **Borrower** class in the Rule Explorer to open it in the BOM editor.

Hint

You can also right-click the **Borrower** class in the Rule Explorer, and click **Open With > BOM Editor**.

__ 3. In the **Class Verbalization** section, review the default verbalization of the `Borrower` class.

__ 4. In the **Members** section, select the **firstName** member of the **Borrower** class, and click **Edit** to open it.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. In the **Member Verbalization** section, note the verbalization for `firstName` that is used in the **Navigation** section:

`the first name of a borrower`

__ 6. Compare the navigation phrase to the **Template** field, and try to understand the use of braces.

`{first name} of {this}`

Note

You learn more about the BOM, the BOM editor, the verbalization, and how to use the braces (`{}`) later in this course.

__ 7. Close the BOM editor.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Creating the main rule project for the decision service

The main decision service project serves as the entry point to the decision service, and it references all of the other rule projects that are in the decision service.

In this section of the exercise, you create a main rule project for the decision service.

__ 1. In the “Set up a Decision Service” task in the Decision Service Map pane, click **Create main rule project**.

Note

You can also open the New Rule Project window by going to **File** > **New > Rule Project** or by right-clicking anywhere in the Rule Explorer to open the menu and clicking **New > Rule Project**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. In the New Rule Project window, create a main rule project for a decision service.

__ a. Select **Decision Service Rule Projects > Main Rule Project**, and click **Next**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. In the **Project name** field, type `myloan-rules` and click **Next**.

__ c. On the Main Rule Project References page, select **mybom**, and click **Finish**.

© Copyright IBM Corp. 2019
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The `myloan-rules` project is now visible in the Rule Explorer.

3.1. Identifying the next tasks in decision service development

The decision service is the starting point of your business rule application development. Consider the next tasks of developing the application.

__ 1. In Rule Explorer, select the `myloan-rules` decision service, and notice the newly enabled links:

- In the **Define decision operation** part: **Add decision operation** and **Go to operation map**
- In the **Deploy and Integrate** part: **Create deployment configuration**

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 4. Creating and defining the decision operation

In this section, you work with the “Define decision operation” part of the Decision Service Map.

A *decision operation* defines the decision-making logic and the input and output data (or parameters) for a decision. The parameters of the decision service function as the interface between your decision service and the calling application.

Requirements

Discussions with the business analysts confirm that the business decision should be based on borrower and loan information. The decision output should update the loan information and provide a report.

You are to implement the following parameters to pass the data between the calling application and the rule engine:

Name	Type	Direction	Verbalization
`borrower`	`training_loan.Borrower`	`IN`	`the borrower`
`loan`	`training_loan.Loan`	`IN_OUT`	`the loan`
`report`	`training_loan.Report`	`OUT`	`the loan report`

These parameters do not have default values.

4.1. Creating the decision operation

__ 1. Make sure that the `myloan-rules` folder is selected.

__ 2. In the “Define decision operation” part of the Decision Service Map, click **Add decision operation**.

__ 3. In the New Decision Operation wizard, in the **Name** field, type `my operation` and click **Next**.

__ 4. In the Source Rule Project window, ensure that **myloan-rules** is selected, and then click **Finish**.

You can see the **my operation** decision operation in the **myloan-rules > deployment** folder.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. In the “Define decision operation” part of the Decision Service Map view, click **Go to operation map**.

__ 6. Select **myloan-rules > deployment > my operation.dop** and click **OK**.

The Decision Service Map switches to the Operation Map. The Operation Map has the following parts:

- **Design signature**
- **Orchestrate**
- **Author**
- **Test**

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

4.2. Creating variables for parameters

In this section of the exercise, you create variables for the decision service. You map these variables to parameters later in this exercise.

__ 1. In the “Design signature” part of the Operation Map, click **Add variable set**.

__ 2. In the New Variable Set wizard, in the **Name** field, type: `my parameters`

__ 3. Click **Finish**.

The Variable Set editor opens.

__ 4. In the Variable Set editor, define the `borrower` parameter.

__ a. Click **Add**.

A new row is displayed with default values. You can type over the values as described next.

__ b. In the **Name** column, overwrite the value (`myVar`) by typing: `borrower`

__ c. In the **Type** column, click the cell, and then click the ellipsis (**...**) to open the Types window.

__ d. Start typing `borrower` to find the Borrower type in the Matching Types list, and then double-click **Borrower** to select it.

__ e. In the **Verbalization** field, overwrite the value by typing: `the borrower`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. Define the `loan` parameter.

__ a. Click **Add**.

__ b. Change the variable values to:

- **Name**: `loan`
- **Type**: `Loan`
- **Verbalization**: `the loan`

Hint

When you set the **Type** value, you can start typing `Loan` and Rule Designer finds the `training_loan.Loan` type for you.

__ 6. Define the `report` parameter.

__ a. Click **Add**.

__ b. Change the variable values to:

- **Name**: `report`
- **Type**: `Report`
- **Verbalization**: `the report`

Hint

When you set the **Type** value, you can start typing `Report` and Rule Designer finds the `training_loan.Report` type for you.

__ 7. Save your work by pressing Ctrl+S.

4.3. Binding the variables to parameters

In this section, you bind the variables that you defined to parameters.

__ 1. In the “Design signature” part of the Operation Map, click **Bind variables**.

The **Signature** tab of the Decision Operation editor opens.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. Expand **my parameters** in the **Eligible variables** section of the Decision Operation Signature editor to see the `borrower`, `loan`, and `report` variables.

__ 3. Map the variables to the input and output parameters.

__ a. Drag `borrower` to the **Input Parameters** table.

__ b. Drag `loan` to the **Input-Output** Parameters table.

__ c. Drag `report` to the **Output Parameters** table.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The decision operation signature now has `borrower` in the Input Parameters table, `loan` in the Input-Output Parameters table, and `report` in the Output Parameters table.

Input Parameters

Define the parameters required to call the execution.

Parameter name	Verbalization	Type
borrower	the borrower	training_loan.Borrower

Input - Output Parameters

Define the parameters that are required, modified, and then returned by the execution.

Parameter name	Verbalization	Type
loan	the loan	training_loan.Loan

Output Parameters

Define the parameters that are initialized and returned by the execution.

Parameter name	Verbalization	Type
report	the report	training_loan.Report

__ 4. Save your work (Ctrl+S).

4.4. Creating a ruleset variable

Parameters carry the objects that are passed between the calling application and the rule engine. The rules evaluate these objects during rule execution.

To avoid directly handling these objects, you can define variables to manipulate the objects during rule execution. When rule execution is complete, the final value of the variable can be assigned back to an output parameter and returned to the external application.

Requirements

In discussion with the business analysts, they clarify that the decision results should be returned in the `training_loan.Report` parameter (which is defined as an OUT parameter).

Rather than manipulating this parameter directly during rule execution, you implement the business request by creating a variable that is called `loanApproved` that can be updated during rule evaluation. When rule execution is complete, the value that is stored in `loanApproved` can be transferred to the `Report` parameter and passed back to the calling application.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

To meet these requirements, you create a ruleset variable that is called `loanApproved`, which has the following attributes:

- **Name**: `loanApproved`
- **Type**: `boolean`
- **Verbalization**: `the loan is approved`
- **Initial Value**: `true`

Note

To define the `loanApproved` variable, you add the variable to the `my parameters` variable set.

However, you do not bind `loanApproved` to a parameter.

__ 1. In Rule Explorer, in **myloan-rules > rules** folder, double-click **my parameters** to open the variable set.

__ 2. Add the `loanApproved` variable.

  __ a. In the Variable Set editor, click **Add**.

  __ b. In the **Name** column, type over `myVar` to enter: `loanApproved`

  __ c. Set the **Type** to `boolean` (lowercase) by clicking the ellipses (...) and in the Types window, typing: `boolean`.

  __ d. In the **Verbalization** column, type over **myVar** to enter: `the loan is approved`

  __ e. In the **Initial Value** column, type: `true`

__ 3. Save your work (Ctrl+S).

__ 4. Close the variable set editor by clicking the **X** for that window.

Name	Type	Verbalization	Initial Value
borrower	training_loan.Borrower	the borrower	
loan	training_loan.Loan	the loan	
report	training_loan.Report	the report	
loanApproved	boolean	the loan is approved	true

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 5. Creating rule packages

Recall in the Operation Map, the Orchestrate part has a link that is called **Add rule package**.

Your next step is to add rule packages to the `myloan-rules` project.

Requirements

The business analysts provide you with an initial outline of how the rules can be organized and the naming conventions to use. Based on these requirements, the rule artifacts should be logically managed within the following packages and subpackages:

- `computation`
- `eligibility`
- `insurance`
- `validation`
  - `validation.borrower`
  - `validation.loan`

These rule packages also outline the smaller decisions that must be taken before determining whether to approve the loan.

You can add packages by using the Operation Map link or by clicking the **New > Rule Package** menu item.

__ 1. Click the **myloan-rules** project and make sure that the Operation Map is open.

__ 2. Create the **computation** package.

__ a. In the Orchestrate part of the Operation Map, click the **Add rule package** link.

The New Rule Package window opens.

__ b. In the **Package** field of the Rule Package window, enter `computation` and click **Finish** (or press Enter).

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The `computation` package is created in your rule project.

__ 3. Add the remaining packages by using the instructions in Step 2:

- `eligibility`
- `insurance`
- `validation`

__ 4. Create the `borrower` and `loan` subpackages of the `validation` package.

__ a. In the Orchestrate part of the Operation Map, click the **Add rule package** link.

__ b. In the **Package** field, type `validation.borrower` and click **Finish**.

__ c. Add another package, type `validation.loan` in the **Package** field, and click **Finish**.

Note

The name of a package is its full path, with a **period** (.) to separate subpackages.

The rule project now contains all the required rule packages.

Notice the next link in the Orchestrate part of the Rule Project Map: **Add ruleflow**.

You work with ruleflows in Exercise 6, "Working with ruleflows".

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 6. Publishing from Rule Designer to Decision Center

In this part of the exercise, you publish the decision service from Rule Designer to Decision Center.

Notice the links in the Author part of the Operation Map.

- **Add action rule**
- **Add decision table**

The decision service is now ready for the rule authors to start working with rules for this project.

By making the decision service available to business users through the Decision Center consoles, rule authors can begin their work on the rules, and thus start the feedback loop on your implementation.

6.1. Publishing the rule project to Decision Center

In this section, you publish the `myloan-rules` decision service to Decision Center to create the corresponding project in Decision Center.

__ 1. Start the sample server by clicking the **Start Sample Server** shortcut in the Windows **Start** menu.

Reminder

Decision Center consoles run on the sample application server that is provided for this course. The sample server must be running before you can access Decision Center.

To check whether the sample server is running, open one of the Decision Center consoles in a browser, for example: `http://localhost:9090/teamserver`

If the console page opens correctly, the server is running and you can close the browser.

For IBM ODM on Cloud users

When you synchronize between Rule Designer and the ODM on Cloud environment, enter the URL of your IBM ODM on Cloud instance in the **URL** field instead of `localhost`.

__ 2. In the Rule Designer Rule Explorer view, right-click the **myloan-rules** decision service, and click **Decision Center > Connect**.

The Decision Center configuration wizard opens.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. Enter the following values in the Decision Center Configuration window fields.

- **URL**: `http://localhost:9090/teamserver`
- **User name**: `rtsAdmin`
- **Password**: `rtsAdmin`
- **Data source**: (Leave the field empty)

Important

Make sure that you use the correct URL and port for your environment.

For IBM ODM on Cloud users

When you synchronize between Rule Designer and the ODM on Cloud environment, enter the URL of your IBM ODM on Cloud instance in the **URL** field instead of `localhost` and make sure to use "`https`".

You must also specify which environment you are publishing to, either development, test, or production. The URL might look like the following example:

`https://odmdemo1.bpm.ibmcloud.com/odm/dev/teamserver`

__ 4. Click **Connect**.

__ 5. After the connection is successfully established, click **Next**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Troubleshooting

Sometimes, when you connect to Decision Center, the Decision Center Configuration window disappears behind other windows. You can click your Rule Designer instance or minimize other windows to see the Decision Center Configuration window.

__ 6. In the Synchronization Settings window, click **Next**.

Note: You do not select the **Use Decision Governance Framework** option for this exercise.

__ 7. In the Decision Service Dependent Projects window, make sure that **mybom** is selected, and click **Finish**.

Troubleshooting

If you see a Secure Storage window, click **No**.

After synchronization completes, you are notified that no changes were found.

© Copyright IBM Corp. 2019
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 8. Click **OK** to close the window.

__ 9. When prompted to switch to the Team Synchronizing perspective, click **No**.

You do not need to switch to the Team Synchronizing perspective because it is empty. However, you can see the Synchronize view in the lower part of the perspective.

Because you just created this project in Decision Center, no differences exist between the project in Rule Designer and the project in Decision Center.

Important

After you publish the `myloan-rules` decision service to Decision Center, do not disconnect from Decision Center so that you do not have to establish this connection again in later steps.

6.2. Opening the project in the Decision Center Business console

Sign in to the Business console to view the rule artifacts that you published from Rule Designer.

__ 1. Open the Decision Center Business console.

__ a. From the Windows **Start** menu, click the **Decision Center Business console** shortcut.

Information

Alternatively, you can open a web browser at the address that you used to connect Rule Designer and Decision Center:

```
http://localhost:9090/decisioncenter
```

Make sure that you use the correct URL and port for your environment.

__ b. Sign in with `rtsAdmin` as both the **User name** and the **Password**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. If you are not on the **Library** tab, click the **Library** tab.

__ 3. On the **Library** tab, notice your `myloan-rules` decision service is now listed.

__ 4. Click the **myloan-rules** link.

The `myloan-rules` decision service opens on the **Releases** tab. Because `myloan-rules` is not a governed decision service, it has no releases.

__ 5. Click **Branches**, and then click **main** to open the main branch of `myloan-rules`.

__ 6. If you are not on the **Decision Artifacts** tab, click **Decision Artifacts**.

Troubleshooting

You might need to use the scrolling arrows on the toolbar to access different tabs.

__ 7. On the left pane of the **Decision Artifacts** page, set the list of artifacts to **All Types**, and expand **myloan-rules**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 8. Click **my parameters** and click the **Edit** (pencil) **icon**.

__ 9. Change the value of the **loanApproved** variable to **false**.

__ 10. Click **Save** and **Create new version**.

__ 11. Leave Business console open.

The next task for a rule author is to complete the decision service by adding the rules. However, before doing anything further in this console, you switch to the Enterprise console to see the decision service in that environment.

6.3. Synchronizing Rule Designer with Decision Center

__ 1. Go back to Rule Designer and make sure that you are in the Rule perspective.

__ 2. Synchronize the `myloan-rules` decision service with Decision Center.

__ a. Right-click `myloan-rules` and click **Decision Center > Synchronize with Decision Center**.

__ b. In the Synchronization Settings window, click **Finish**.

__ c. When prompted to switch to the Team Synchronizing perspective, click **Yes**.

__ 3. In the Synchronize view, expand **myloan-rules - main > rules** to see that your variable set is changed.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. Double-click **my parameters.var** to open it in the Text Compare view and check your changes.

__ 5. After you see the differences, close the Text Compare view.

Information

In the Synchronize view, entries show whether rule artifacts were modified locally, remotely, or both concurrently. The color and direction of the arrow in the entry icon indicates where the modification occurred, and what type of action is possible.

Entry	Modification source	Expected actions
Black entries with an arrow that points to the right	A change occurred in Rule Designer.	Publish to Decision Center what is in Rule Designer: Right-click the rule artifact, and click **Publish**. Select **Override and Update** when you want to override the changes, keep the version from Decision Center, and update Rule Designer.
Blue entries with an arrow that points to the left	A change occurred in Decision Center.	Update Rule Designer with what is in Decision Center: Right-click the rule artifact, and click **Update**. Select **Override and Publish** when you want to override the changes, keep the version from Rule Designer, and publish it again to Decision Center.
Red entries with double arrows	Changes occurred in both Rule Designer and in Decision Center.	Automatic merging is not possible. Decide how to handle this conflict. If you want to update Rule Designer with the changes that are made in Decision Center, select **Override and Update**. If you want to keep the changes from Rule Designer and publish them to Decision Center, select **Override and Publish**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In this case, after consulting with the rule author to confirm the reasons for this change, you agree that the default value should return to `true`.

__ 6. Update your local copy of the project in Rule Designer by right-clicking **my parameters.var** and clicking **Update**.

__ 7. When prompted about conflicts, click **Yes** to confirm that you want to overwrite the changes.

After the publication is finished, no changes are listed in the Synchronize view.

__ 8. Close the Text Compare view.

__ 9. Return to the Rule perspective by clicking **Rule** on the toolbar.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 7. Deleting a decision service from Decision Center

When synchronization is not a viable option, or for some reason you must delete an entire decision service, you can erase the decision service from the Decision Center database.

__ 1. Open the Decision Center Business console, and sign in again with `rtsAdmin` as the user name and password.

__ 2. On the **Library** tab, click the arrow beside myloan-rules and click **Delete**.

__ 3. When prompted to confirm deletion of the decision service, click **Delete**.

__ 4. Sign out and close the Business console.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 8. Importing a decision service from Decision Center

In this part of the exercise, you import a decision service in to Rule Designer from an existing decision service in Decision Center.

Requirements

Business analysts indicate that they worked with another team and developed a complementary decision service, called Miniloan Service. The most recent version of Miniloan Service is in Decision Center. Business analysts ask you to import this decision service in Rule Designer to look at it.

8.1. Creating a project in Rule Designer from Decision Center

__ 1. Go back to Rule Designer and make sure that you are in the Rule perspective.

__ 2. Click **File > New > Project**, select **Rule Project from Decision Center**, and click **Next**.

The New Rule Project from Decision Center wizard opens.

__ 3. Verify the connection entries, which should be prepopulated from the previous exercise.

- **URL**: `http://localhost:9090/teamserver`
- **User name**: `rtsAdmin`
- **Password**: `rtsAdmin`

__ 4. Click **Connect**.

When the connection is established, the list of remote projects becomes available.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. In the **Project configuration** section, select **Miniloan Service** from the **Remote project** list.

Note

If you cannot see the project in the list, click **Refresh list**.

The **Remote project** list shows all the rule projects in Decision Center for which you have the **View** permission. This permission depends on the **user name** that you used to connect.

Here, because you are connected as `rtsAdmin`, the Decision Center administrator, you can see all the rule projects in Decision Center.

__ 6. In the **Select a branch or release** menu, select the **main** branch.

__ 7. Click **Finish** to import Miniloan Service into Rule Designer.

Rule Designer imports all the items in the rule project into your workspace and creates Miniloan Service in Rule Designer based on its content in Decision Center.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Troubleshooting

You can ignore any errors for now. When you get a synchronization error message, click **OK** to close the window. You see the synchronization error message because of errors in the BOM. You learn about BOM errors in the next section of this exercise.

__ 8. When prompted to switch to the Team Synchronizing perspective, click **No** to remain in the Rule perspective.

__ 9. When the Synchronize Complete window opens to notify you that no changes are found, click **OK** to close this window.

__ 10. Look at the created decision service in Rule Designer, and relate its content to the content of the corresponding decision service in Decision Center.

__ 11. In Miniloan Service, disconnect Rule Designer from Decision Center.

__ a. Right-click **Miniloan Service** and click **Decision Center > Disconnect**.

The “Disconnect from Decision Center” window opens.

__ b. Select the **Keep the Decision Center entries files** option, and click **Yes**.

__ c. Click **OK** to close the Decision Center synchronization error window.

Reminder

You learn about the BOM errors in the next section of this exercise.

__ 12. Stop the sample server by clicking **Start** and then the **Stop Sample Server** shortcut.

You can also stop the sample server through the app list by clicking **Start**, clicking the down arrow, and in the IBM Operational Decision Manager V8.10.0 section, clicking **Stop Sample Server**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

8.2. Finalizing the rule project in Rule Designer

After you successfully retrieve a rule project from Rule Designer, your work is not necessarily complete. In many cases, as you learn now with Miniloan Service, you still have a few steps to follow.

__ 1. In the Rule perspective, select the Miniloan Service (after it is finished building in the Rule Designer workspace), and click **Problems** to open the Problems view.

__ 2. Look at the messages in the **Problems** view, and notice that the issues is about the project path for the `miniloan-xom`, which is the XOM project for the Miniloan Service decision service.

__ 3. Open the Miniloan Service properties.

__ a. In the Rule Explorer, right-click **Miniloan Service** and click **Properties** from the menu.

The Java Execution Object Model page opens and you see that `miniloan-xom` is selected.

__ b. In the Properties window, select **Business Object Model**.

__ c. In the Business Object Model page, expand the list in the Required BOM entries to see the origin or source for the Miniloan Service BOM.

Questions

Why does your new project have XOM errors?

Answer

To build a rule project in Rule Designer that is created from Decision Center, you must also have its referenced projects. These referenced projects also include any executable elements (such as XOMs, `.jar` files, or libraries) that are not in Decision Center.

Questions

Which executable elements are required?

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Answer

The Miniloan Service decision service cannot build because it requires the `miniloan-xom` project, which is not present in your Rule Designer workspace.

Stop

For this exercise, you are not required to retrieve the missing XOM.

__ d. Click **Cancel** to close the Properties window.

You successfully created the initial elements that are required to start developing a business rule application, and learned how the Rule Project Map guides you through development. You also learned how to publish and synchronize the rule authoring environment with Decision Center.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

This exercise demonstrated how you start developing a business rule application. You set up a decision service in Rule Designer, and published the rule environment to Decision Center to make it accessible to business users through the Business console or the Enterprise console.

As an optional step when you are finished with this exercise, you can import the exercise solution file into a new workspace by using the Samples Console and review the solution.

__ 1. (*Optional*) To view the exercise solution, switch to a new workspace.

__ a. Go to **File > Switch Workspace > Other**.

__ b. In the **Workspace** field, enter a name for the solution workspace, such as:

`<LabfilesDir>\workspaces\decision_service_answer`

__ c. When the new workspace opens, close the **Welcome** tab.

__ 2. Open the Samples Console perspective.

__ a. Click the **Open Perspective** icon.

__ b. Select **Samples Console**, and click **OK**.

__ 3. Import the solution project for this exercise.

__ a. Go to **Rule Designer > Training > Ex 02: Setting up decision services > 02-answer**.

__ b. Click **Import projects**.

__ c. When the import is complete, close the Help pane.

__ 4. Review the solution for this exercise.

Note

The solution project does not include the imported `Miniloan Service` decision service.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 3. Modeling decisions

Estimated time

01:00

Overview

This exercise demonstrates how to create, test, and deploy a decision model service.

Objectives

After completing this exercise, you should be able to:

- Create a model diagram
- Define the decision and data node structure
- Create custom data types
- Author the business logic in decision modeling language
- Test the model
- Export and import models from IBM Decision Composer
- Deploy and run a decision model service

Introduction

In this exercise, you learn how to model decisions in Decision Center Business console. You also see how to test your model and deploy it as a decision service to an execution environment.

This exercise includes the following sections:

- Section 1, "Creating a diagram"
- Section 2, "Author and test the decision logic"
- Section 3, "Exporting and importing a decision model service"
- Section 4, "Deploy the decision model service"

Requirements

The second part of this exercise uses IBM Decision Composer online, and requires that you have an IBM ID. If you do not have an IBM ID, you can follow the instructions in the exercise to obtain one.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Creating a diagram

In this section of the exercise, you set up the environment for the exercise and create a standard rule project for the decision service.

1.1. Open Decision Center Business console

__ 1. Start the sample server.

__ a. Click the **Start** menu, and click the **Start Sample Server** shortcut.

__ b. Wait until the server is started.

__ 2. Open the Business console from the **Start** menu, by clicking the **Decision Center Business console** shortcut.

__ 3. Sign in as a business user with the following values for the **Username** and **Password** fields.

- **Username**: `rtsAdmin`
- **Password**: `rtsAdmin`

__ 4. Click **Log in**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. Click the Home tab, and notice the **Start modeling decisions** section, which links you to the IBM Knowledge Center for more information about modeling decisions.

1.2. Create a model

__ 1. Click the **Library** tab.

The Library includes regular decision services and decision services that are based on models, or *decision model services*. Decision model services are indicated by the model icon.

__ 2. Click the **Create new decision service model** (plus sign) icon.

__ 3. In the name field, enter: `my decision model` and click **Create**.

__ 4. Click **main**.

A new model diagram opens with a decision node and a data node as your starting point.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. Click the **Edit** icon.

Next, you create a model that matches runners to races that suit their abilities.

1.3. Design the Match Race to Runner decision model

To model a decision, first consider these questions:

Design questions	Answers
What is the actual decision you want to make?	Match a runner to the appropriate race
What is the business data that is involved in the decision?	The **runner** The **races**
What data influences the outcome?	The **location** and **ability** of the runner The **location** and **difficulty level** of the races
Is the main decision dependent on other decisions?	Yes, determining the runner's level of ability
What data is involved in the subdecision?	**How fast** and **how far** this runner runs regularly

Next, you translate the design into a diagram.

1.4. Model the node structure

__ 1. Define the Matching race decision node.

__ a. In the diagram, click the Decision 1 node.

__ b. On the **Model** tab, change the name from **Decision 1** to `Matching race` and press Enter.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. In the **Output type** list, choose **string**.

__ d. Select the **Output is a list** option.

To match a race to a runner, the decision depends on the race and the runner data. Next, you add the dependency to these data nodes.

__ 2. Click the Input Data 1 node and set the name to: `Runner`

__ 3. Add another data node for Races.

__ a. Hover your mouse on the Matching race node and click the plus sign.

__ b. Click **Input Data**.

__ c. Set the name to: `Races`

© Copyright IBM Corp. 2019
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

You see errors on your data nodes because they need to be set to a type. For this model, you create custom types.

1.5. Create custom data types

__ 1. Define the custom category type.

__ a. Click the **Types** tab.

__ b. In the **Data Types** section, click the plus sign and click **Add enumeration type**.

__ c. In the **Name** field, type `category` and press Enter.

__ d. In the **List of values** section, click the plus sign to add these values.

advanced
intermediate
beginner

__ 2. Define the custom runner type.

__ a. In the **Data Types** section on the **Types** tab, click the plus sign and click **Add type**.

__ b. In the **Name** field, type `runner` and press Enter.

__ c. Add the following attributes by clicking the plus sign.

name	string
area	string
average speed	number
weekly mileage	number
category	category

Note that the category attribute is based on the custom `category` type that you created.

__ 3. Define the custom runner type.

__ a. In the **Data Types** section on the **Types** tab, click the plus sign and click **Add type**.

__ b. In the **Name** field, type `race` and press Enter.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. Add the following attributes by clicking the plus sign.

name	string
area	string
category	category

1.6. Set the output types for the data nodes

__ 1. Click the **Model** tab to return to the diagram.

You can now set the type for the Races and Runner nodes.

__ 2. Set the Output type for the Races and Runner nodes.

__ a. Click the **Runner** data node and set the **Output type** to **runner**.

__ b. Click the **Races** data node and set the **Output type** to **race**, and select **Output is a list** option.

The data nodes do not show errors anymore.

__ 3. Save your work.

__ a. On the toolbar, click **Save and Continue**.

__ b. Click **Create new version**.

Note

Save often to avoid losing your work in case of session timeouts. Each time that you save, you create new version of the model.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Author and test the decision logic

The **Matching race** decision needs as input the location and ability of the runner, plus the location and difficulty level of the races. The runner's ability is defined by the **category** attribute.

Before a match can be made, a subdecision must determine the runner's category based on how fast and how far the runner runs each week. For this subdecision, you add another decision node called **Runner category** as input to the **Matching race** decision node

2.1. Define the Runner category decision node and logic

__ 1. Create the **Runner category** decision node.

__ a. On the toolbar, click the **Decision** plus sign.

__ b. Hover your mouse over the new node, click the plus sign, and use your mouse to drag it to the **Matching race** node.

Hint

By dragging the plus sign on one node to another node, you create a dependency between those nodes.

__ c. This node requires data input from the **Runner** data node, so hover your mouse over the **Runner** data node, click the plus sign, and use your mouse to drag it to the **Runner category** decision node.

__ 2. Set the name of the node to: `Runner category`

__ 3. Set the **Output type** to **category**.

__ 4. Define the **Runner category** logic.

__ a. In the **Decision logic** section, click **Add table**.

__ b. In the **Select the condition columns for your decision table** section, choose:

- **the weekly mileage of the runner**
- **the average speed of the runner**

These selections become condition columns in your table.

__ c. Click **Create table**.

__ d. Keep the default table name.

__ e. Set the values in the table to match these values.

weekly mileage		average speed		Runner category
min	**max**	**min**	**max**	
0	10	3	6	beginner

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

10	20	6	9	intermediate
20	30	9	12	advanced

__ 5. Change the operator in the **weekly mileage** columns.

__ a. Right-click the cell in the min or max column of row 1

__ b. Click **Change Operator** and select **]..]**

The result is equivalent to: more than 0 and at most 10

__ c. Change the operator for rows 2 and 3.

Your decision table should match this screen capture.

	weekly mileage		average speed		Runner category
	min	max	min	max	
1	]0	10]	3	6	beginner
2	]10	20]	6	9	intermediate
3	]20	30]	9	12	advanced

__ 6. Close the table.

2.2. Define the Matching race logic

__ 1. Click the **Matching race** decision node.

__ 2. In the **Decision logic** section of the **Model** tab, click **Add rule**.

__ 3. Choose the following criteria in the **Select the criteria for your rule** section, and click **Create rule**.

- the races
- the category of the runner
- the name of the runner

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. Copy the following text for this rule:

```
for each race called 'the race' , in 'the races'
if
    the area of 'the runner' is the area of 'the race'
    and the category of 'the race' is 'the runner category'
then
    add "Here is a race that we recommend for you: " + the name of 'the race'
+" . Are you in?" to decision;
```

Your rule should match this screen capture.

```
for each race called 'the race' , in 'the races'
if
   the area of 'the runner' is the area of 'the race'
   and the category of 'the race' is 'the runner category'
then
   add "Here is a race that we recommend for you: " + the name of 'the race' +" . Are you in?" to decision;
```

Available inputs | Output

Name	Type
Races	*List of race*
Runner	*runner*
category	*category (enum)*

__ 5. Close the rule.

2.3. Test the model

__ 1. Click the **Validate** tab, and click the plus sign to create a new data set.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. For the **New input** field, click the **Edit** (pencil) icon and set the name to: `Mark`

__ 3. Define input data for the **Races** section

__ a. In the **race** fields, enter the following data:

- **name**: `Paris Marathon`
- **area**: `Paris`
- **category**: `advanced`

__ b. Click **Add race**, and enter the following data:

- **name**: `Citadel Trail`
- **area**: `Montsegur`
- **category**: `intermediate`

__ c. Click **Add race** again, and enter the following data:

- **name**: `Terry Fox Run`
- **area**: `Toronto`
- **category**: `beginner`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. In the **Runner** section, enter the following data:

- **name**: `Mark`
- **area**: `Toronto`
- **weekly mileage**: `7`
- **average speed**: `3`
- **category**: `beginner`

__ 5. Click **Run**.

The results are correct.

__ 6. In the **Runner** section, change **area** to **`Montreal`** and click **Run**.

The test with `area` set to `Montreal` does not provide any output.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Questions

Why is there no output when there is no matching race?

Answer: Because the decision logic doesn't handle cases when no races match the runner. You must author a rule to handle a "no match" situation.

2.4. Add a rule to handle cases when no race matches the runner

__ 1. Return to the diagram and click the **Matching race** node.

__ 2. Add a second rule to handle the case when no races match the runner.

__ a. In the **Decision logic** section, click **Add rule**.

__ b. In the **Select the criteria for your rule** section, choose:

- **the races**
- **the name of the runner**

__ c. Click **Create rule**.

__ 3. Overwrite the rule contents with the following text:

```
if the number of elements in 'the matching races' equals 0
then
  print "Hi, " + the name of 'the runner' + "! We have no recommendation for
your category and area , please come back later" ;
```

2.5. Validate the model

__ 1. Click the **Validate** tab,

__ 2. Make sure that `area` is set to `Montreal`, and click **Run**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The Decision outputs section correctly shows that no race matched.

__ 3. In the **Runner** section on the Validate tab, change `area` to **`Toronto`**, and click **Run**.

The Decision outputs section correctly shows a match. However, the Messages section also includes a message that no match was found.

Questions

Why does the output show conflicting results?

2.6. Editing rule sequence

__ 1. Return to the diagram and click the **Matching race** decision node.

Notice the setting: Rules are applied in sequence. That means `rule-2` executes before `rule-1` matches a race to the runner.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. Select **rule-1** and drag it above **rule-2**.

__ 3. Return to the **Validate** tab and click **Run**.

This time, the results are correct.

__ 4. Click **Save and Close** on the toolbar.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Exporting and importing a decision model service

3.1. Sign in to IBM Decision Composer

__ 1. Open a browser window, and open IBM Decision Composer at this URL:

`https://decision-composer.ibm.com`

__ 2. If you have an IBM ID, click **Sign in**.

Note

If you do not have an IBM ID, click **sign up** on this page to register for one.

3.2. Export the project

__ 1. On the Decision Composer home page, click **Samples**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. Select **Runner** and click **Import**.

__ 3. Click the Runner project menu (3 dots) and click **Export**.

The `Runner.dproject` file is saved to your **Downloads** folder.

3.3. Import the project

__ 1. Return to the browser window running Business console, and click the **Library** tab.

__ 2. Click the **Import** icon.

__ 3. In the Import Decision Service window, click **Choose**, browse to the `Runner.dproject` file in the **Downloads** folder, and click **Open**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. Click **Import**.

__ 5. Click **main**.

__ 6. Click the **Validate** tab and click **Run**.

__ 7. Take a few minutes to explore the data set and the model to see how the logic is defined in this version of the decision model service.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 4. Deploy the decision model service

__ 1. In Business console, click the **Deployment** tab and notice that a deployment configuration exists.

__ 2. Click the **Deploy** icon.

__ 3. Open Rule Execution Server by clicking the shortcut on the **Start** menu.

__ 4. In the Navigator view, expand **RuleApps** to see your **Runner_RuleApp**.

Your decision model service is ready to be invoked for execution.

Information

Decision model services are deployed and executed exactly like decision services, because they *are* decision services.

Later in this course, you learn more about how decision services are executed against real data.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. Delete this decision service.

__ a. In the RuleApps table, select **Runner_RuleApp**, and click **Remove**.

__ b. When prompted to confirm deletion, click **Confirm**.

__ 6. Delete the decision model service resource file.

__ a. In the Navigator view, click **Resources**.

__ b. Select **Runner-decisionmodel.jar** and click **Remove**.

__ c. Click **Confirm**.

__ 7. Delete the decision model service library file.

__ a. In the Navigator view, click **Libraries**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Select **Runner-RuleApp_library** and click **Remove**.

__ c. Click **Confirm**.

__ 8. Close all the browsers running Rule Execution Server console, Business console, and Decision Composer.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

This exercise demonstrated how to create, test, and deploy a decision model service.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 4. Working with the BOM

Estimated time

00:30

Overview

This exercise describes how to create a BOM from a XOM.

Objectives

After completing this exercise, you should be able to:

- Generate a BOM from an existing XOM
- Verbalize the BOM with natural-language vocabulary

Introduction

In this exercise, you work with the Java code to finalize the implementation of the XOM. Next, you create a BOM and vocabulary that is based on the completed XOM.

The exercise includes these tasks:

- Section 1, "Finalizing the XOM"
- Section 2, "Creating a rule project with a BOM"
- Section 3, "Working with the vocabulary that is required to author rules"

Requirements

This exercise uses the following files, which are installed in the `<InstallDir>\studio\training` directory:

- Start project: `Dev 03 - BOM\01-start`
- Solution project: `Dev 03 - BOM\02-answer`
  - You use the solution project in "Exercise review and wrap-up" on page 4-20

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Finalizing the XOM

As part of your role as a developer, you create the implementation code, including the XOM, for the business rule project. The `loan-xom` Java project does not compile, so your task is to complete the XOM implementation.

Note

The purpose of this exercise is not to teach you how to create Java classes, but to illustrate the types of tasks that developers are required to do when implementing the XOM.

1.1. Setting up your environment for this exercise

__ 1. In Rule Designer, switch to a new workspace by following these steps:

__ a. From the **File** menu, click **Switch Workspace > Other**.

__ b. In the Workspace Launcher window, enter the path:

`<LabfilesDir>\workspaces\bom`

__ c. Click **Launch**.

__ 2. Close the Welcome view.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. Use the Samples Console to import the exercise start project.

__ a. In the upper-right area of the Rule perspective, click the **Open Perspective** icon.

__ b. In the Open Perspective window, click **Samples Console** and click **Open**.

__ c. In the **Rule Designer** section, expand **Training > Ex 03: Working with BOMs** > **01-start**, and click **Import projects**.

__ 4. When the workspace finishes building, close the Help view.

__ 5. In the Rule Explorer, notice that you now have the `loan-xom` Java project available in your workspace.

The `loan-xom` project shows the warning icon.

1.2. Understanding the problem

__ 1. In the Rule perspective, click the **Problems** tab to open the Problems view (at the bottom of the perspective) and expand **Warnings** to review the problem.

Notice the description of the problem states that the `spouse` field of the `Borrower` class is not used.

__ 2. Click the **Tasks** tab to open the Tasks view.

The Tasks view indicates a series of tasks that are required to finalize this XOM.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Next, you complete the tasks that are listed in the Tasks view to finish implementing the classes in the `loan-xom` Java project.

1.3. Finish implementing the Borrower class

__ 1. In Rule Explorer, expand **loan-xom > src > training_loan** and see the `Borrower.java` class file.

__ 2. Double-click **Borrower.java** to open this file in the code editor.

Next, you generate the “getter” code for some of the attributes in this class.

Example

This code shows you an example of what a getter method should look like for the `firstName` attribute:

```
/**
* @return the firstName
*/
public int getFirstName() {
   return firstName;
}
```

Rule Designer can automatically generate these types of methods for you, as you see next.

__ 3. In the `Borrower` class, create the getters (not the setters) for these attributes:

- `firstName`
- `lastName`
- `SSN`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ a. In the `Borrower` class code, select the `Borrower` class name, and right-click to click **Source > Generate Getters and Setters.**

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. In the “Generate Getters and Setters” window, expand the **firstName**, **lastName**, and **SSN** attributes and select the “get” method for each.

__ c. Select **Generate method comments**.

__ d. Click **OK**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ e. In the `Borrower.java` file, look for the newly added getter methods.

```
15      /**
16       * @return the firstName
17       */
18      public String getFirstName() {
19          return firstName;
20      }
21
22      /**
23       * @return the lastName
24       */
25      public String getLastName() {
26          return lastName;
27      }
28
29      /**
30       * @return the sSN
31       */
32      public SSN getSSN() {
33          return SSN;
34      }
```

Note

The new methods were appended where your mouse was positioned in the file.

__ 4. In the `Borrower` class, create both the getters and the setters for the following attributes:

- `creditScore`
- `spouse`
- `yearlyIncome`
- `zipCode`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Example

The following examples show what the getter and setter methods should look like for the `yearlyIncome` attribute.

```
/**
* @return the yearlyIncome
*/
public int getYearlyIncome() {
return yearlyIncome;
}

/**
* @param yearlyIncome the yearlyIncome to set
*/
public void setYearlyIncome(int yearlyIncome) {
this.yearlyIncome = yearlyIncome;
}
```

__ a. Reopen the “Generate Getters and Setters” window from the `Borrower.java` source code editor by right-clicking the `Borrower` class name, and clicking **Source > Generate Getters and Setters**.

__ b. Select **creditScore**, **spouse**, **yearlyIncome**, and **zipCode**, which selects both the “get” and “set” methods for these attributes.

__ c. Click **OK**, and review the source code to see these newly added methods.

__ d. Save your work by pressing Ctrl+S.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Note

The new methods for the `spouse` attribute should resolve the warning that you saw earlier.

__ 5. In the `Borrower` class, create a getter that is called `getBirthDate` for the `birth` attribute.

This method must return `birth.getTime()`, which is an instance of the `java.util.Date` class.

Rule Designer cannot automatically generate this method, so you must manually edit the `Borrower.java` file.

__ a. Scroll to the end of the file to find the final closing brace (`}`).

__ b. Just before the final brace, add the following lines of code:

```
/**
* @return the birth date
*/
public Date getBirthDate() {
return birth.getTime();
}
```

__ 6. In the `Borrower` class, look for the following comments and delete them:

```
/**
* TODO: Create the getters (not the setters) for the following attributes:
firstName, lastName, and SSN.
* TODO: Create the getters and the setters for the following attributes:
zipCode, yearlyIncome, creditScore, and spouse.
* TODO: Create a getter called getBirthDate for the birth attribute.
*/
```

__ 7. Save the `Borrower.java` file by pressing Ctrl+S.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

1.4. Finish implementing the Test class

__ 1. In Rule Explorer, double-click **Test.java** to open it in the code editor and follow the instructions of the `TODO` comments.

__ a. Look through the `Borrower` class to find the following method calls and delete the forward slashes (`//`) to obtain these lines of code:

```
b1.setCreditScore(600);
b1.setYearlyIncome(100000);
r1.setCorporateScore(b1.getCreditScore());
```

```
Borrower b1 = new Borrower("John", "Doe", DateUtil.makeDate(1968, Calendar.MAY,
//TODO : Uncomment when the getters and setters of the Borrower's attributes are
b1.setCreditScore(600);
b1.setYearlyIncome(100000);
b1.setLatestBankruptcy(DateUtil.makeDate(1990, Calendar.JANUARY, 01), 7, "Unemp
System.out.println(b1);

Borrower b2 = new Borrower("John", "Doe", DateUtil.makeDate(1970, Calendar.MAY,
System.out.println(b2);

Borrower b3 = new Borrower("John", "Doe", DateUtil.makeDate(1970, Calendar.MAY,
System.out.println(b3);

Loan l1 = new Loan(DateUtil.makeDate(2005, Calendar.JUNE, 1), 60, 100000, 0.70),
System.out.println(l1);

Report r1 = new Report(b1, l1);
//TODO : Uncomment when the getters and setters of the Borrower's attributes are
r1.setCorporateScore(b1.getCreditScore());
r1.addCorporateScore(12);
```

__ b. Delete the `TODO` comments in the `Test` class.

__ 2. Press Ctrl+Shift+S (“Save All”) to save the `Borrower` class and the `Test` class.

__ 3. Close the editor windows for both the Borrower and the Test classes.

__ 4. Verify that the Problems view and the Tasks view are empty, which indicates that the `loan-xom` Java project compiled successfully.

__ 5. Close the **Test.java** and **Borrower.java** editing windows.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Creating a rule project with a BOM

Now that you have a complete XOM, you create a standard rule project, and then you create a BOM from the XOM.

__ 1. Create a standard rule project with a BOM.

__ a. Right-click anywhere in the Rule Explorer and click **New > Rule Project**.

__ b. On the “Select a template” page, in the **Decision Service Rule Projects** list, select **Standard Rule Project** and click **Next**.

__ c. In the **Project name** field, type `loan-rules-bom` and click **Finish**.

__ 2. Create a reference to the `loan-xom` project.

__ a. Right-click **loan-rules-bom** and click **Properties**.

The new rule project opens in the rule editor.

__ b. In the Properties wizard, click **Java Execution Object Model**, select **loan-xom**, and click **OK**.

__ 3. Create a BOM entry in the `loan-rules-bom` project.

__ a. In Rule Explorer, right-click **loan-rules-bom > bom** and click **New > BOM Entry**.

__ b. In the New BOM Entry window, keep `model` in the **Name** field, select the **Create a BOM entry from a XOM** option, and click **Next**.

__ c. For the **Choose a XOM entry** field, click **Browse XOM**.

__ d. In the Browse XOM window, select **platform:/loan-xom**, and click **OK** to close the window.

__ e. In the **Select classes** section, expand **training_loan**, select the check boxes for all classes except `Test`, and click **Next**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ f. On the BOM Verbalization page, click **Deselect All** to clear all the check boxes, select **Borrower**, and click **Finish**.

The `bom` folder of the `loan-rules` project now contains a BOM entry called `model`. This `model` is the BOM that you generated from the XOM.

Only the `Borrower` class is verbalized.

__ 4. Make sure that everything is saved by pressing Ctrl+Shift+S.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Working with the vocabulary that is required to author rules

In the previous part of the exercise, you learned how to create the default vocabulary of your BOM at the time you created the BOM itself. In this part of the exercise, you learn more about the BOM editor and create the vocabulary that is required to author rules by using the Verbalization wizard in Rule Designer. You also review the various concepts that are involved, including business terms, phrases, and placeholders.

3.1. Creating the default vocabulary

Requirements

Business analysts examined the BOM that you created for the project and found that some elements must be reworked. They now ask for a complete verbalization of the `Report` class and of the `Loan` class. They also ask that you revise the vocabulary that is associated with the `Borrower` class.

In this section, you learn now how to create or modify the vocabulary in the BOM editor.

__ 1. In Rule Explorer, expand **loan-rules-bom > bom > model > training_loan**.

__ 2. Double-click **Report** to see this class in the BOM editor, and note that this member does not yet have a verbalization.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. In the BOM editor, verbalize the `Report` class and all its members with the default verbalization.

__ a. Click the **Package** tab to open the Package page of the BOM editor, and click the **Verbalize** task.

The Verbalize BOM wizard opens.

__ b. On the Configure Verbalization page of the Verbalize BOM wizard, click **Deselect All**, and then select **Report**.

You want to verbalize the `Report` BOM class and all its members.

__ c. Click **Finish**.

__ d. Save your work (Ctrl+S).

__ e. In the **Business Object Model Entry** section of the **Package** tab of the BOM editor, expand **training_loan** and double-click **Report** to open it in the Class page of the BOM editor.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ f. In the **Members** section of the Class page, double-click **approved** to open it in the Member page and view the default verbalization.

The default verbalization for the setter of the Boolean `approved` member is:

```
make it {approved} that {this} is approved
```

__ 4. Return to the **Package** tab and double-click **Loan** to open it in the **Class** tab.

__ 5. In the **Class Verbalization** section, click **Create a default verbalization**.

Note

The `Loan` class itself is now verbalized, but not its members.

In the next steps, you verbalize some members individually in the corresponding **Member Verbalization** section of the BOM editor.

__ 6. Create the default verbalization for the `duration` member of the Loan class.

__ a. On the Class page for the `Loan` class, double-click the **`duration`** member to edit it in the Member page of the BOM editor.

The Member page opens. In the **Member Verbalization** section of the Member page, you can see that the `duration` member is not verbalized.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. In the **Member Verbalization** section of the Member page, click **Create a default verbalization**.

__ c. Notice the default verbalization:

```
Navigation: "the duration of a loan"
Template: {duration} of {this}
```

__ 7. Save your work (Ctrl+S).

3.2. Examining the verbalization for members of the Borrower class

__ 1. On the **Package** tab of the BOM editor, double-click the `Borrower` class to open it in the **Class** tab.

__ 2. Click **Edit term** in the **Class Verbalization** section.

The Edit Term window opens.

__ a. Notice that you can edit the `borrower` term to set the vocabulary for singular and plural forms, and for the definite and indefinite articles.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Click **Cancel** to close the Edit Term window.

__ 3. From the **Class** tab, open the `hasLatestBankrupcy` method of the `Borrower` class and examine its default verbalization:

```
{this} is has latest bankrupcy
```

Questions

Do you think that the default verbalization of the `hasLatestBankrupcy` method is suitable vocabulary for business rules?

Answer

No, the verbalization ("`{this} is has latest bankrupcy`") would not make sense in a sentence, so a rule that uses this phrase would be confusing. The sentence would not be grammatically correct.

Also, the word "bankruptcy" is incorrectly spelled as: *bankrupcy*

Important

When you create a BOM from a XOM, or when you define the default verbalizations, you retain the way that the method is written in the XOM.

In the XOM for this exercise, "bankruptcy" is incorrectly written as "*bankrupcy*" in the `hasLatestBankrupcy` method name. As a result, the name and the default verbalization of the `hasLatestBankrupcy` method in the BOM also have the same incorrect spelling.

__ 4. From the **Class** tab, open the `setLatestBankruptcy` method of the `Borrower` class and examine its default verbalization in the **Template** field.

```
{this}.setLatestBankruptcy({0}, {1}, {2})
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Questions

Do you think that the default verbalization of the `setLatestBankruptcy` method is suitable for business rules?

Answer

No, the verbalization of the `setLatestBankruptcy` method is not suitable for business rules because it does not look like natural language. In a business rule, this type of programming-language vocabulary makes the rule awkward and unreadable.

Questions

Can you identify what the placeholders in the default verbalization of the `setLatestBankruptcy` method stand for?

Answer

The placeholders correspond to the three arguments of this method, as shown in the **Arguments** section on the **Member** tab.

3.3. Editing the default verbalization

The default verbalization of the `hasLatestBankrupcy` method is incorrect. The default verbalization of the `setLatestBankruptcy` method also does not look like natural language.

You must correct this situation by changing the vocabulary that is associated with these methods.

__ 1. Return to the Member page for the `hasLatestBankrupcy` method, and change the verbalization in the **Template** field of the **Member Verbalization** section to the following text:

```
{this} has latest bankruptcy
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Note

The new verbalization is more natural and also corrects the way “bankruptcy” is written so that you and business users can easily author rule artifacts later.

You can leave the name of the BOM method unchanged and consistent with the name of the corresponding XOM method, without any effect on authored rules.

__ 2. From the Class page of the BOM editor, double-click the `setLatestBankruptcy` method.

__ 3. Change the verbalization in the **Template** field to match the following text:

```
section the latest bankruptcy of {this} to date {0}, chapter {1} and reason
{2}
```

Notice that the BOM editor notifies you of missing placeholders as you edit the **Template** field.

__ 4. Save your work.

__ 5. Close the BOM editor.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

To see a solution to this exercise, switch to a new workspace and import the project **Ex 03: Working with BOMs > 02-answer**.

The first part of this exercise looked at how you can create a BOM from an existing XOM, and create the default vocabulary at the same time.

The second part of this exercise looked at how you can use the BOM editor to create or modify the vocabulary that is associated with a specific BOM element.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 5. Refactoring

Estimated time

00:45

Overview

This exercise describes how to manage inconsistencies within the project as the XOM, BOM, and vocabulary evolve.

Objectives

After completing this exercise, you should be able to:

- Refactor vocabulary changes
- Manage inconsistency issues after updating the XOM and BOM

Introduction

In this exercise, you modify the XOM, BOM, and vocabulary to support rule authoring requirements from the business users. You also learn how these changes can affect the project in terms of consistency, and you resolve the consistency issues.

The exercise includes these tasks:

- Section 1, "Refactoring rule artifacts after vocabulary changes"
- Section 2, "Maintaining consistency between the BOM and the XOM"

Requirements

This exercise uses the following files, which are installed in the `<InstallDir>\studio\training` directory:

- Start project: `Dev 04 - Refactoring\01-start`
- Solution project: `Dev 04 - Refactoring\02-answer`

You can use the solution project as described in "Exercise review and wrap-up" on page 5-15.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Refactoring rule artifacts after vocabulary changes

In this part of the exercise, you see how a change in the vocabulary affects rule artifacts, and learn more about the relationship between the rule artifacts and the vocabulary in the BOM.

1.1. Setting up your environment for this exercise

__ 1. In Rule Designer, switch to a new workspace:

__ a. From the **File** menu, click **Switch Workspace > Other**.

__ b. In the Workspace Launcher window, enter the path:

`<LabfilesDir>\workspaces\refactoring`

__ c. Click **Launch**.

__ 2. Close the Welcome view.

__ 3. Use the Samples Console to import the exercise start project.

__ a. Click the **Open Perspective** icon.

__ b. In the Open Perspective window, click **Samples Console** and click **Open**.

__ c. In the **Rule Designer** section, expand **Training > Ex 04: Refactoring > 01-start**, and click **Import projects**.

__ 4. Close the Help view.

1.2. Exploring the default verbalization

__ 1. In Rule Explorer, expand **loan-rules > bom > model > training_loan > Borrower**.

__ 2. Double-click the **getBankruptcyAge** method to see its default verbalization:

```
{bankruptcy age} of {this}
```

__ 3. To see how this verbalization is displayed in a rule, open the `checkLatestBankruptcy` rule in the **rules** folder of the `loan-rules` project.

__ a. Expand **loan-rules > rules**.

__ b. Double-click **checkLatestBankruptcy**.

The condition statement uses this vocabulary:

```
if the bankruptcy age of 'the borrower' is less than 365
```

Questions

Is this rule condition easy to understand and unambiguous?

Answer

The `checkLatestBankruptcy` condition is ambiguous because a borrower might have more than one bankruptcy. Rule authors might be unsure about how to maintain this rule.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Questions

How can you solve this ambiguity issue?

Answer

Change the vocabulary to use clear language.

Important

You must discuss usability requirements with the business analysts and rule authors before changing vocabulary, since they are best placed to explain vocabulary requirements.

Requirements

The business analysts request that you change the "`bankruptcy age`" term to indicate the *latest* bankruptcy.

1.3. Changing the default verbalization of the getBankruptcyAge method

After you change the default verbalization in the next steps, you also learn the effects of such a change and how to resolve them.

__ 1. Edit the `getBankruptcyAge` BOM member verbalization to become: `age of the latest bankruptcy`

__ a. Switch back to the **model** tab in the editing window.

__ b. Click the **Edit the subject used in phrases** link in the **Member Verbalization** section of the `getBankruptcyAge` method.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. In the **Singular** field of the Edit Term window, enter:

`age of the latest bankruptcy`

__ d. Click **OK**.

Note the changes to the navigation phrase and the template of `Borrower.getBankruptcyAge`.

A warning is displayed that indicates which rules the new vocabulary affects.

__ 2. Save your work (Ctrl+S).

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The Refactor Vocabulary window opens.

Because the previous vocabulary term is no longer valid, you must update any rule artifact that uses that BOM member.

The Refactor Vocabulary window automatically lists the rules that are affected so you can update them all at the same time.

__ 3. In the Refactor Vocabulary window, click **Select All**, and click **Yes**.

__ 4. Wait for the workspace to finish building, then switch back to the `checkLatestBankruptcy` rule to see how it was modified.

1.4. Changing the default verbalization of the duration member

__ 1. In Rule Explorer, in the **rules** folder, double-click the **checkDuration** rule.

The ***if*** statement checks whether the duration is between 1 and 30.

Requirements

The business analysts request that you change the navigation template for the `duration` member of the `Loan` class should indicate that the duration is in *years*.

__ 2. In Rule Explorer, expand **bom > model > training_Loan > Loan**, double-click the **`duration`** BOM member, and look at the current verbalization.

__ 3. Edit the **Template** field in the **Navigation** section by inserting `(in years)` after `{duration}`

```
{duration} (in years) of {this}
```

The Navigation and Template phrases are updated.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. Save your BOM.

The Refactor Vocabulary window opens again for you to select the business rule elements that you want to update.

__ 5. In the Refactor Vocabulary window, click **Select All**, and click **Yes** to automatically refactor the selected rule artifacts.

__ 6. Open the `checkDuration` rule to see how it was modified.

__ 7. Close the editors for the `checkLatestBankruptcy` and `checkDuration` rules.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Maintaining consistency between the BOM and the XOM

In this part of the exercise, you make sure that the XOM and the BOM of the rule project are consistent.

2.1. Adding a XOM class that is missing in the BOM

In this section, you use the BOM Update view to determine which classes in the XOM are not present in the BOM, or vice versa, and to make sure that the two object models are consistent.

__ 1. In Rule Explorer, expand **loan-xom > src > training_loan** to see the list of classes.

__ 2. Expand the BOM in the `loan-rules` project (**loan-rules > bom > model > training_loan**).

Questions

Do the lists match?

Notice that the `training_loan.Test` class exists in the `loan-xom` Java project, but is not present in the BOM of the `loan-rules` project.

__ 3. Right-click **`model`** in the `bom` folder of the `loan-rules` project, and click **BOM Update**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The BOM Update view opens in the lower part of the perspective.

__ 4. In the BOM Update view, expand **Differences and Actions**.

Notice the lines that state that the `training_loan.Test` XOM class is not found in the BOM.

Only one suggested action is listed: `Import the XOM class training_loan.Test`.

__ 5. Select the suggested action, and click **Perform and save** to apply it.

The BOM is automatically updated, and the `training_loan.Test` BOM class is created.

Rule Designer also prompts you to create a default verbalization for this new class in the Verbalize BOM window.

__ 6. In the Verbalize BOM window, click **Select All**, and click **Finish** to create a default verbalization for the new class.

__ 7. Notice the presence of the new `training_loan.Test` class in the BOM of the `loan-rules` decision service.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

2.2. Adding a XOM method that is missing in the BOM

Requirements

Following a meeting with business analysts, you must now create a method that is called `getFullName` in the `Borrower` XOM class of the `loan-xom` project. You must also author an action rule that lists the full names of all borrowers.

Applying this requirement affects consistency between the XOM and the BOM. In this section, you learn how to resolve this type of inconsistency.

__ 1. In Rule Designer, switch to the Java perspective.

__ a. Click the **Open Perspective** icon.

__ b. Select **Java (default)** from the perspectives list, and click **Open**.

__ 2. In the Package Explorer of the Java perspective, expand **loan-xom > src > training_loan** and double-click **`Borrower.java`** to edit it.

__ 3. Add a method called `getFullName` to the `Borrower` class of the `loan-xom` project.

The method should concatenate the first name and the last name. You use this code to implement the method:

```
public String getFullName() {
   return getFirstName() + " " + getLastName();
}
```

You can append this method at the end of the `Borrower` class before the final closing brace (`}`).

__ 4. Save your work.

__ 5. Switch back to the Rule perspective.

__ 6. Go to the BOM Update view, and manage the new difference between the BOM and the XOM.

__ a. Refresh the BOM Update view by clicking **Refresh** ( ) in the upper-right area of the pane.

__ b. Expand **Differences and Actions** and notice the new line that states that the XOM attribute `training_loan.Borrower.fullName` was not found in the `training_loan.Borrower` BOM class.

__ c. In the **Actions** field, select the following line and click **Perform and save**.

```
Update the BOM class "training_loan.Borrower"
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. When the Verbalize BOM window opens, click **Select All** and click **Finish** to create a default verbalization for the new member.

The BOM is automatically updated, and the `Borrower` BOM class contains a new member called `fullName`.

__ e. Refresh the BOM Update view, and notice that no differences between the XOM and the BOM are listed.

2.3. Creating a rule to use this new attribute

__ 1. Add an action rule to the **rules** folder of the `loan-rules` project:

__ a. Right-click the **rules** folder, and click **New > Action Rule**.

__ b. In the **Name** field, type: `displayFullName`

__ c. Click **Finish**.

The Intellirule editor opens.

__ 2. In the **Content** section of the rule editor, type the following rule:

```
then
   print the full name of the borrower of 'the loan report';
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. Notice that the new “full name” attribute is immediately available as vocabulary for rule authoring.

Information

For this exercise, you want to test only your new BOM member, so no ***if*** statement is required in your rule. As you learn later, condition statements are optional in rules.

__ 4. Save your work.

2.4. Deprecating BOM members

The BOM Update view also shows the differences when the BOM has a member that is not in the XOM, and also suggests different corrective actions.

Requirements

Following a subsequent meeting, business analysts now indicate that the `getFullName` method in the `Borrower` class of the `loan-xom` project is superfluous and must be removed.

Applying this requirement can affect consistency between the XOM and the BOM, and affect the rule project. In this section, you learn how to resolve these consistency issues.

__ 1. Delete the `getFullName` method that you created in "Adding a XOM method that is missing in the BOM" on page 5-9.

__ a. In Rule Explorer, expand **loan-xom > src > training_loan**, and double-click the **`Borrower.java`** file.

__ b. Delete the code that you added for the `getFullName` method.

__ c. Save your work.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. Open the Problems view, and notice that the `Borrower` class in the BOM of the `loan-rules` project now has an error:

```
[B2X] Cannot find attribute "fullName" in execution class
"training_loan.Borrower"
```

__ 3. Refresh the BOM Update view and look at the description of the difference.

The difference states that the `training_loan.Borrower.fullName` BOM attribute cannot be found within the `training_loan.Borrower` XOM class.

__ 4. Open the list of proposed actions to solve it.

With the possible actions, you can either *delete* or *deprecate* the BOM attribute: `training_loan.Borrower.fullName`

__ 5. Select: **Deprecate the BOM attribute "`training_loan.Borrower.fullName`"** and click **Perform and save**.

Information

You can ignore the error icon for the `Borrower.fullName` BOM member.

You learn how to resolve this error in later steps.

__ 6. View the deprecation result in the `Borrower.fullName` BOM member in the BOM editor.

__ a. Expand **loan-rules > bom > model > training_loan > Borrower**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Double-click the **fullName** attribute.

Notice how this `fullName` attribute is marked as deprecated.

__ 7. View the deprecation result in the `displayFullName` rule.

__ a. Expand **loan-rules > rules**.

__ b. Reopen the `displayFullName` action rule.

__ c. Hover your mouse over the **Warning** icon.

__ 8. View the result in the Problems view.

__ a. Notice the error:

```
Cannot find attribute "fullName" in execution class
"training_loan.Borrower"
```

__ b. Notice the warning messages and the resources to which they apply:

- The deprecated message for the `displayFullName` action rule
- The deprecated message for the `fullName` attribute of the `Borrower` BOM class

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Questions

Can you identify how to update the projects in your workspace to resolve the problems and warnings that are currently identified?

Answer

Several options exist to resolve this problem:

- Delete the `fullName` attribute of the `Borrower` BOM class, and delete the `displayFullName` action rule.
- Implement the Getter method of the `fullName` attribute of the `Borrower` BOM class in its **BOM to XOM mapping** section to replace the missing XOM method.

However, these options are business decisions, so before you implement a solution, you must first consult with the business analysts.

Requirements

The decision is made to implement the first listed solution.

__ 9. Delete the `fullName` attribute of the `Borrower` BOM class.

__ a. Open the `Borrower` BOM class in the BOM editor.

__ b. In the Members list, select **fullName** and click **Delete**.

__ c. When prompted to confirm removal of this BOM member, click **Yes**.

__ d. Save your work.

__ 10. Delete the **displayFullName** action rule.

__ a. In the Rule Explorer, right-click the **displayFullName** action rule.

__ b. Click **Delete**.

__ c. When prompted to confirm deletion, click **Yes**.

All errors and warnings should now be removed from the Problems view.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

The first part of the exercise looked at how a change in the vocabulary affects rule artifacts, and how the rule artifacts relate to the vocabulary in the BOM.

The second part of the exercise looked at how to keep the XOM and the BOM of the rule project consistent with each other upon changes in the XOM.

(Optional) To see the solution to this exercise, switch to a new workspace and import the project **Ex 04: Refactoring > 02-answer**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 6. Working with ruleflows

Estimated time

00:30

Overview

In this exercise, you learn how to create a ruleflow.

Objectives

After completing this exercise, you should be able to:

- Describe the parts of a ruleflow
- Create a ruleflow
- Orchestrate rule selection and execution through the ruleflow

Introduction

In this exercise, you explore an existing ruleflow to see how ruleflows are designed and how they orchestrate rule execution. You then create your own ruleflow in the Ruleflow editor.

The exercise involves these tasks:

- Section 1, "Exploring a ruleflow diagram"
- Section 2, "Creating a ruleflow"
- Section 3, "Defining a main ruleflow"

Requirements

This exercise uses the files that are installed in the `<InstallDir>\studio\training\Dev 05 - Ruleflows\01-start` directory.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Exploring a ruleflow diagram

In this section, you explore the parts of an existing ruleflow in the Ruleflow editor.

1.1. Setting up your environment for this exercise

__ 1. In Rule Designer, switch to a new workspace:

__ a. From the **File** menu, click **Switch Workspace > Other**.

__ b. In the Workspace Launcher window, enter the path:

`<LabfilesDir>\workspaces\ruleflow`

__ c. Click **Launch**.

__ 2. Close the Welcome view.

__ 3. Use the Samples Console to import the exercise start project.

__ a. Click the **Open Perspective** icon.

__ b. In the Open Perspective window, click **Samples Console** and click **Open**.

__ c. In the Rule Designer section of the “Samples and Tutorials” page, expand **Training > Ex 05: Ruleflows > 01-start**, and click **Import projects**.

__ 4. When the workspace finishes building, close the Help view.

Note

Ignore any warnings that appear on the project.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

1.2. Exploring the ruleflow

__ 1. In Rule Explorer, expand **loan-rules > rules** and double-click the **`loanvalidation`** ruleflow to open it in the Ruleflow editor.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. Notice the outline of tasks and the transitions between them.

Questions

Do you recognize the ruleflow tasks?

Do you understand the paths that can be taken through the ruleflow?

Can you determine what the ruleflow does by looking at how it is organized?

Answer

The ruleflow organizes the flow of rule execution in a ruleset to produce a decision.

Within the ruleflow, rules are evaluated in groups or *rule tasks*. Each rule task is equivalent to a rule package.

Evaluation of each rule task produces a result or decision. For example, after executing the rules in the `validation` task, the result would be either *valid* or *not valid*. If the result is *valid*, the ruleflow follows the path to the `computation` task. Otherwise, the ruleflow goes directly to the end and the rule execution terminates.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Transitions between tasks define the path through the ruleflow. You set conditions on transitions to define which path to take according to the results of evaluating a rule task.

__ 3. Double-click the **`validation`** rule task in the diagram to open its properties in the Properties view.

__ a. On the **Rule Task** tab of the Properties view, look at how you can set the **Algorithm**, **Exit criteria**, and **Ordering** rule execution properties of this rule task.

These execution properties dictate how instances of the rules in this rule task are executed.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. On the **Rule Selection** tab of the Properties view, expand the **`validation`** package and subpackages to see the contents.

__ c. On the **Rule Selection** tab of the Properties view, click **Edit** to open the Select Rules window.

__ d. Take some time and experiment with selecting which rules to include in this task.

Notice that all the validation rules are included. However, this window provides the option of removing rules or changing the order in which they are evaluated by moving them up or down.

__ e. Click **Cancel** to close the Select Rules window.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ f. Take some time to explore the **Initial Action** tab, the **Final Action** tab, and the **Documentation** tab of the `validation` task properties.

- On the **Initial Action** tab, you define the actions to take before this task executes. You can write these actions either in Business Action Language (BAL) or in ILOG Rule Language (IRL).
- On the **Final Action** tab, you define the actions to take after this task executes. You can write these actions either in BAL or in IRL.
- On the **Documentation** tab, you can write comments to describe this task.

1.3. Exploring action tasks and transitions

See now how action tasks and transition conditions are expressed in the ruleflow, and how they can use parameters and variables.

Questions

Can you identify an action task in the ruleflow?

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Answer

The `initResult` task is an action task.

__ 1. In the `loanvalidation` ruleflow, select the **`initResult`** task to see its properties in the Properties view.

__ 2. On the **Action Task** tab of the Properties view, look at the code of the `initResult` task.

The function code of this `initResult` action task is written in ILOG Rule Language (IRL), which is similar to Java code.

In the present case, the `initResult` action task is used to create the `report` output parameter.

Questions

The function code does not set the initial value of the `loanApproved` variable. Why not?

Answer

You do not have to set the initial value of the `loanApproved` variable in the ruleflow. This initial value is already given as part of the variable definition, in the `local var` variable set.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. View the properties for the `data valid` transition.

__ a. In the `loanvalidation` ruleflow diagram, click the `data valid` transition.

__ b. In the Properties view, click the **Condition** tab to see the condition properties.

Questions

Do you see how to label a transition and how to write the condition of a transition?

Answer

You set the label and the condition of a transition on its **Condition** tab. The condition might be written in either Business Action Language (BAL) or ILOG Rule Language (IRL). You learn what BAL and IRL are later in this course.

Often, this transition is expressed by using a Boolean parameter, a Boolean variable, one of their Boolean attributes, or a logical combination of them. In the present case, the transition is based on the value of the Boolean `validData` attribute of `report`, the output parameter of the `training_loan.Report` class.

This transition condition is written in BAL.

__ 4. Take some time to explore the relationship between the `validData` attribute of the `training_loan.Report` class, its verbalization that you can see in the BOM of the `loan-rules` decision service. Also, take time to explore the way this transition condition is expressed.

__ 5. View the properties for the end node.

__ a. In the `loanvalidation` ruleflow diagram, click the end node.

__ b. In the Properties view, click the **Final Action** tab.

__ c. View the **Final Action** properties.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The Final Action makes sure that the `approved` attribute of the `report` parameter is set according to the latest values of the `loanApproved` variable and the `validData` attribute of the `report` parameter.

This Final Action is written in IRL:

```
report.approved=loanApproved&&report.validData;
note(borrower.toString());
note(loan.toString());
note(report.toString());
```

1.4. Using parameters and variables in rules

Rule authoring is described in detail later, but for now, take some time to explore how the rules manipulate the parameters and the variables that are defined in the rule project.

__ 1. In the `loan-rules` decision service, expand the **eligibility** package.

__ 2. Double-click the **`approval`**, **`checkCreditScore`**, and **`checkIncome`** rules of the **eligibility** package to open them in the rule editor.

Questions

How do these action rules manipulate the `loanApproved` variable and the `report` parameter that are defined for this project?

Answer

The action rules of the `eligibility` package are expressed in BAL, and their expression is based on the BOM verbalization.

- These action rules use the `loanApproved` variable, which is verbalized as: `'the loan is approved'`
- These action rules use the `report` parameter, which is verbalized as: `'the loan report'`

__ 3. Close the rule editor windows for these rules.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Creating a ruleflow

Now that you explored a ruleflow in its entirety, create a ruleflow with the Ruleflow editor.

__ 1. In the `loan-rules` project, create a ruleflow:

__ a. Right-click the `rules` folder in the `loan-rules` project and click **New > Ruleflow**.

__ b. In the **Name** field of the New Ruleflow wizard, type `loanvalidation_2` and click **Finish**.

The `loanvalidation_2` ruleflow automatically opens in the Ruleflow editor.

In the next steps, you re-create parts of the `loanvalidation` ruleflow.

Hint

Keep the `loanvalidation` ruleflow open while you work on the `loanvalidation_2` ruleflow in a second Ruleflow editor. By switching between windows and tabs, you can see what you must update when you work through the next steps.

__ 2. Add the start node of the `loanvalidation_2` ruleflow.

__ a. Click **Create a start node**.

__ b. In the Ruleflow editor main area, click anywhere on the canvas to place the node.

__ 3. Add the end node of the `loanvalidation_2` ruleflow.

__ a. Click **Create an end node**.

__ b. In the Ruleflow editor main area, click anywhere on the canvas to place the node.

__ 4. Create the `initResult` action task of the `loanvalidation_2` ruleflow with the same IRL function code as in the `initResult` action task of the `loanvalidation` ruleflow.

__ a. Click **Create an action task**.

__ b. In the Ruleflow editor main area, click anywhere on the canvas to place the action task.

__ c. Click the created action task, and in the Properties view, click the **Action task** tab.

__ d. On the **Action Task** tab, in the **ID** field, enter: `initResult`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ e. Select the **IRL** option and enter the following IRL function code:

```
report = new training_loan.Report(borrower,loan);
report.approved = true;
```

__ f. Save your work (Ctrl+S).

Troubleshooting

You can ignore the errors that you see.

__ 5. Create the `validation`, `computation`, `eligibility`, and `insurance` rule tasks by dragging each of those rule packages from Rule Explorer onto the Ruleflow editor.

__ 6. Set the **Rule Task** properties of each rule task in the `loanvalidation_2` ruleflow to match the corresponding **Rule Task** properties of the `loanvalidation` ruleflow.

__ a. Select one of the rule tasks.

__ b. In the Properties view, set the task properties on the **Rule Task** tab to match the properties in the `loanvalidation` ruleflow.

- **validation**
  - Algorithm: **Fastpath**
  - Exit Criteria: **None**
  - Ordering: **Literal**
- **computation**
  - Algorithm: **RetePlus**
  - Exit Criteria: **None**
  - Ordering: **Default**
- **eligibility**
  - Algorithm: **Sequential**
  - Exit Criteria: **None**
  - Ordering: **Priority**

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

- **insurance**
  - Algorithm: **RetePlus**
  - Exit Criteria: **None**
  - Ordering: **Default**

__ 7. Create the transitions between the tasks.

__ a. In the Ruleflow editor palette, click **Create a transition**. You can now create multiple transitions in the diagram.

__ b. Click the start node and then click the **`initResult`** action task.

__ c. Click the **`initResult`** action task, and then click the **`validation`** rule task.

__ d. Click the **`validation`** rule task, and then click the **`computation`** rule task.

__ e. Click the **`computation`** rule task, and then click the **`eligibility`** rule task.

__ f. Click the **`eligibility`** rule task, and then click the **`insurance`** rule task.

__ g. Click the **`insurance`** rule task, and then click the end node.

__ h. Create another transition between the **`validation`** rule task and the end node.

__ 8. Disable the transition creation mode in the Ruleflow editor palette, by clicking **Create a transition** again.

__ 9. To adjust the ruleflow diagram layout, go to the Ruleflow editor toolbar and click the **Layout All Nodes** icon.

__ 10. Set the Condition properties of the transition between the validation and computation rule tasks to match the corresponding transition in the **`loanvalidation`** ruleflow.

__ a. Click the transition between the **`validation`** and **`computation`** rule tasks.

__ b. In the Properties view, the **Condition** tab should be open, and in the **Label** field, enter: `data valid`

__ c. Select **Use BAL for transition condition**.

__ d. In the condition editor, enter: `'the loan report' is valid data`

Notice that the transition from **validation** to the **end node** is automatically labeled **else**.

Troubleshooting

If you find it difficult to select transitions and other ruleflow elements for editing in the Properties window, try closing the ruleflow and reopening it in the ruleflow editor.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 11. Save your work.

__ 12. Compare your ruleflow to the `loanvalidation` ruleflow.

Questions

Are you missing any tasks or transitions?

What tasks must you do to complete the `loanvalidation_2` ruleflow diagram so that it matches the `loanvalidation` diagram?

Answer

To complete the `loanvalidation_2` diagram so that it matches the `loanvalidation` diagram, you must complete the following tasks.

__ 1. Add a transition between the **`eligibility`** rule task and the end node.

__ 2. Add a label and a condition to the transition between the eligibility and insurance tasks.

- __ a. Select the transition between the **`eligibility`** and **`insurance`** rule tasks.
- __ b. On the **Condition** tab, in the **Label** field, enter: `loan approved`
- __ c. Select **Use BAL for transition condition**.
- __ d. In the condition editor, enter: `'the loan is approved'`

__ 3. Save your work.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Defining a main ruleflow

An application can have several ruleflows, but one of them must be defined as the main ruleflow. The main ruleflow is defined in the decision operation for the decision service.

In this task, you learn how to define a main ruleflow.

__ 1. Open the decision operation for the `loan-rules` decision service.

__ a. In the Rule Explorer, expand **loan-rules > deployment**.

__ b. Double-click **loan-rulesOperation** to open it in the decision operation editor.

In the **Ruleflow** section of the decision operation editor, you see the following options:

- **Use main ruleflow**
- **Do not use a ruleflow**

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In this exercise, the `loanvalidation` ruleflow is already selected as the main ruleflow.

__ 2. Change the main ruleflow to the `loanvalidation_2`.

__ a. In the **Ruleflow** section, next to the **Use main ruleflow** option, click the **loanvalidation** link.

__ b. In the Ruleflow window, select **loanvalidation_2** and click **OK**.

The `loanvalidation_2` ruleflow is now set as the main ruleflow.

__ 3. Change the main ruleflow back to the `loanvalidation` ruleflow.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. Save your work (Ctrl+S).

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

This exercise looked at how to design a ruleflow.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 7. Exploring action rules

Estimated time

00:30

Overview

In this exercise, you learn how to write action rules.

Objectives

After completing this exercise, you should be able to:

- Identify the parts of an action rule
- Explain the difference between using automatic variables or rule variables

Introduction

To learn how to author action rules, you review the Trucks and Drivers rule project, which is a complete project with rules that use various BAL constructs. You explore and run these rules to understand rule behavior during rule execution.

The exercise involves these tasks:

- Section 1, "Exploring rule structure"
- Section 2, "Exploring rule behavior"
- Section 3, "Working with automatic variables"

Requirements

This exercise uses the following files, which are installed in the `<InstallDir>\studio\training` directory:

- Start project: `Dev 06 - Exploring rules\01-start`
- Solution project: `Dev 06 - Exploring rules\02-answer`

You can use the solution project as described in "Exercise review and wrap-up" on page 7-12.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Exploring rule structure

In this section, you explore the behavior of rules in the Trucks and Drivers rule project. This rule project is designed to demonstrate specific BAL constructs. You can open each rule, review the constructs that are used, and then view the results that are produced after rule execution.

1.1. Setting up your environment for this exercise

__ 1. In Rule Designer, switch to a new workspace:

__ a. From the **File** menu, click **Switch Workspace > Other**.

__ b. In the Workspace Launcher window, enter the path:

`<LabfilesDir>\workspaces\explore`

__ c. Click **Launch**.

__ 2. Close the Welcome view.

__ 3. Use the Samples Console to import the exercise start project.

__ a. Click the **Open Perspective** icon.

__ b. In the Open Perspective window, click **Samples Console** and click **Open**.

__ c. In the **Rule Designer** section, expand **Training > Ex 06: Exploring rules > 01-start**, and click **Import projects**.

__ 4. When the workspace finishes building, close the Help view.

Note

Ignore any warnings that appear on the project.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

1.2. Exploring the rules

The action rules to explore are provided in different rule packages.

__ 1. In Rule Explorer, expand: **trucksAndDrivers-rules > rules**.

__ 2. Notice the `condition` package, which includes the following subpackages to illustrate these BAL constructs in condition statements:

- **Arithmetic operators**: Action rules in this rule package show how to use mathematical operators to carry out calculations in the condition of an action rule.
- **Collections**: Action rules in this rule package show how to define a variable to retrieve a list of objects (`java.util.Collection`).
- **Comparisons**: Action rules in this rule package show how to use BAL operators to compare the values of strings or the number of objects in working memory.
- **Count tests**: Action rules in this rule package show how to use operators that count the number of occurrences of an object.
- **Existence tests**: Action rules in this rule package show how to use BAL operators to test the existence of objects with certain members in working memory.
- Conditions are qualified with the **`where`** keyword.
- **Grouped conditions**: Action rules in this rule package show how to use the `all`, `any`, and `none` BAL constructs to combine conditions, as an alternative to logical operators (`and`, `it is not true that`, `or`) between condition statements.
- **Tests for membership in a set**: Action rules in this rule package show how to write conditions that test whether a member of an object in working memory is part of a set (`java.util.Collection`).

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. In the Rule Explorer, open the BOM by expanding **trucksAndDrivers-bom > bom > model**.

__ 4. Expand **drivers** and review the objects and vocabulary.

Questions

Consider these questions as you review the ***rules*** and ***vocabulary*** in this rule project:

- Can you identify the objects on which the action rules work?
- Are there any objects for which you cannot identify the origin?
- Based on your experience with Java programming, can you determine what the terms *definitions*, *conditions*, *actions*, and *variables* mean in relation with the design of the rules?

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Exploring rule behavior

In this part of the exercise, you execute at least one action rule from each rule package in the `trucksAndDrivers-rules` project.

To execute a rule:

__ 1. In Rule Explorer, expand **trucksAndDrivers-tests > rules**.

__ 2. Double-click the **`testFlow`** ruleflow to open it in the Ruleflow editor.

__ 3. In the ruleflow diagram, double-click the **`Execute Rule`** rule task to open the Properties view.

__ 4. In the Properties view, click the **Rule Selection** tab, and click **Edit** to choose a rule to run.

__ 5. In the Select Rules window, you can choose any rule to start with, and you can also repeat these steps to try other rules.

__ a. Browse through the list of rules in the left section of the Select Rules window to see the available rules.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Expand **condition > count**, and double-click **01_count_theNumberOf** to add that rule to the task.

__ c. Click **OK**.

__ 6. Save your work (Ctrl+S).

__ 7. From the **Run** menu, click **Run Configurations** to set up rule execution.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 8. In the Run Configurations window, expand **Decision Operation** and click **Test a rule - Decision service**.

__ 9. In the `Test a rule - Decision service` launch configuration, click **Run** to start the execution of the `trucksAndDrivers-tests` ruleflow.

The console opens and you can see that the rule executed correctly by looking at the message, which should match the action statement in the rule.

__ 10. Try selecting different rules and running them, one at a time, to view their behavior.

__ 11. After you finish testing, remove the rules that you added, so that only the `01_action_phrase` rule remains.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Working with automatic variables

In this part of the exercise, you create automatic variables in your BOM and use them in your rule artifacts. By doing so, you learn the differences in using an automatic variable instead of a rule variable in action rules.

__ 1. In Rule Explorer, expand **trucksAndDrivers-bom > bom > model > drivers** and double-click the `Truck` BOM class to edit it in the BOM editor.

__ 2. In the **Class Verbalization** section, select **Generate automatic variable**.

A variable that is called `the truck` is automatically made available in the rule editors. You can use the automatic variable to author rules based on objects in the working memory that are instances of the `Truck` BOM class.

__ 3. Save the BOM (Ctrl+S) and wait for the workspace to rebuild.

An error message appears on the **trucksAndDrivers-rules** project.

__ 4. In the Problems view, double-click the error to open the problem rule.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Questions

Why does this rule no longer compile?

Answer

The `01_setMembership_isOneOf` action rule cannot compile because it declares a variable that is called `truck`, which conflicts with the name of the automatic variable that you created.

Information

In the following steps, you gain hands-on experience with the rule editor by trying to intuitively use its authoring functions.

Questions

How can you rewrite the rule so that it compiles?

Answer

- To avoid conflicting variables, you can delete the definitions part of the `01_setMembership_isOneOf` rule.
- As indicated in the BOM, the automatic variable for the `Truck` class is verbalized as **`the truck`**, meaning that everywhere you previously had **`truck`** only, you must now use: **`the truck`**

__ 5. Select the ***definitions*** statement and delete it.

__ 6. Change the `01_setMembership_isOneOf` action rule to use the automatic variable by clicking the **Quick Fix** icons, and then double-clicking the proposed fix: **Insert word 'the'**

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The resulting rule should match this rule:

```
if
  the model of the truck is one of { the F150 , the MACK TRUCK }
then
  display the message "The truck model of " + the serial number of the truck
+ " is F150 or MACK_TRUCK";
```

Hint

You can also double-click **`truck`** and select the correct variable from the list, but if you use this option, be careful not to overwrite any other part of the rule statement.

__ 7. Save the rule and close it in the editing window.

__ 8. In Rule Explorer, expand **trucksAndDrivers-rules > rules > action** and double-click the **`01_action_phrase`** rule to open it.

Questions

Can you figure out how you can simplify this rule with the new automatic variable?

Do not change it yet.

Answer

- The definition in the `01_action_phrase` action rule is useless.
- Where the `01_action_phrase` action rule uses the rule variable **`'the truck'`** (with single quotation marks), it can use the automatic variable **`the truck`** (without quotation marks) instead.

__ 9. To use the automatic variable, delete the definition statement and modify the `01_action_phrase` rule in the `action` rule package to use the automatic vservicearіable.

Your rule should match the rule statement here:

```
if
  the current load of the truck is more than the capacity of the model of the
truck
then
  remove the smallest delivery of the truck;
  display the message "the truck overloaded " + the serial number of the
truck + " has its smallest delivery removed";
```

__ 10. Save the `01_action_phrase` action rule and close it.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 11. Rerun the `Test a rule - Decision service` configuration by clicking the **Run** icon on the toolbar, and clicking **Test a rule - Decision service** from the list.

The traces in the Console view are:

```
the truck overloaded TRUCK-T2000 has its smallest delivery removed
```

These traces should match the traces you had in Section 2, "Exploring rule behavior," on page 7-5.

Optional

Optionally, you can make similar changes in the other action rules in the `trucksAndDrivers-rules` project.

- Delete all definitions of the rule variable called `'the truck'` (with single quotation marks).
- Where you had the rule variable `'the truck'` (with single quotation marks), you can use the automatic variable `the truck` instead (without quotation marks).

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

The first part of the exercise looked at how the action rules are structured and demonstrated their behavior during rule execution. You also saw the difference between using automatic variables or rule variables.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

(Optional) To see the solution to this exercise, switch to a new workspace and import the project **Ex 06: Exploring rules > 02-answer**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 8. Authoring action rules

Estimated time

00:45

Overview

In this exercise, you learn how to author action rules.

Objectives

After completing this exercise, you should be able to:

- Use the Intellirule editor and Guided editor to author action rules
- Use rule variables, automatic variables, and parameters in rule statements

Introduction

In this exercise, you author the action rules that use rule variables, automatic variables, and parameters.

The exercise includes these tasks:

- Section 1, "Authoring action rules in the Intellirule editor"
- Section 2, "Authoring actions rules in the Guided editor"
- Section 3, "Authoring rules with parameters"

Requirements

You should complete these exercises before proceeding:

- Exercise 2, "Setting up decision services"
- Exercise 7, "Exploring action rules"

This exercise uses the following files, which are installed in the `<InstallDir>\studio\training` directory:

- Start project: `Dev 07 - Authoring rules\01-start`
- Solution project: `Dev 07 - Authoring rules\02-answer`

You can use the solution project as described in "Exercise review and wrap-up" on page 8-15.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Authoring action rules in the Intellirule editor

In this exercise, you author rules in the main Rule Designer rule editors:

- Intellirule editor
- Guided editor

Note

You can also edit the action rules as pure text, in the Eclipse Text editor.

1.1. Setting up your environment for this exercise

__ 1. In Rule Designer, switch to a new workspace:

__ a. From the **File** menu, click **Switch Workspace > Other**.

__ b. In the Workspace Launcher window, enter the path:

`<LabfilesDir>\workspaces\author`

__ c. Click **Launch**.

__ 2. Close the Welcome view.

__ 3. Use the Samples Console to import the exercise start project.

__ a. Click the **Open Perspective** icon.

__ b. In the Open Perspective window, click **Samples Console** and click **Open**.

__ c. In the **Rule Designer** section, expand **Training > Ex 07: Authoring rules > 01-start**, and click **Import projects**.

__ 4. When the workspace finishes building, close the Help view.

The workspace includes a predefined `loan-xom` XOM project and a separate `loan-bom` project.

__ 5. In Rule Designer, create a main rule project named: `loan-rules`

__ a. In the “Set up a Decision Service” part of the Decision Service Map, click **Create main rule project**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. In the **Decision Service Rule Projects** section of the New Rule Project wizard, make sure that **Main Rule Project** is selected, and click **Next**.

__ c. In the **Project name** field, enter `loan-rules` and click **Next**.

__ d. In the Main Rule Project References page, select **loan-bom**.

This reference page indicates that the `loan-rules` project references the `loan-bom` project. The `loan-rules` project uses the vocabulary from this BOM project.

__ e. Click **Finish**.

__ 6. Create a decision operation for the `loan-rules` decision service.

__ a. In the Rule Explorer, select the **`loan-rules`** main rule project.

__ b. In the “Define decision operation” part of the Decision Service Map, click **Add decision operation**.

__ c. In the New Decision Operation window **Name** field, enter: `loan-rules operation`

__ d. Click **Next**.

__ e. In the Source Rule Project window, select **loan-rules** and click **Finish**.

__ 7. Add a rule package that is named `loan` to the `loan-rules` decision service.

__ a. In the Rule Explorer, select the **`loan-rules`** main rule project.

__ b. In the “Define decision operation” part of the Decision Service Map, click **Go to operation map**.

__ c. In the “Select an operation” window, select **loan-rules operation.dop**, and click **OK**.

__ d. In the Orchestrate part of the Operation Map, click **Add rule package**.

__ e. In the **Package** field of the New Rule Package window, enter: `loan`

__ f. Click **Finish**.

__ 8. Close the **loan-rules operation** tab.

1.2. Authoring rules with the Intellirule editor

Requirements

***Scenario*:**

A loan company must implement its loan policies, which evolve regularly, and must provide a loan application to its agents.

First, the application must verify the input data from a form. Then, it must check customer eligibility, which is based on personal profile, score, and the type of loan requested. The application can also compute some insurance rates, as a function of the computed score.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Use the Intellirule editor to author some of the action rules that implement the Loan scenario.

__ 1. In the **loan** rule package, add a rule that is called: `grade`

__ a. In Rule Explorer, right-click the **loan** rule package, and click **New > Action Rule**.

The New Action Rule window opens.

__ b. In the **Name** field, enter: `grade`

__ c. Click **Finish**.

The empty rule opens in the Intellirule editor.

Note

To verify whether you are using the Intellirule editor or the Guided editor, look at the tab of the editor window:

- **Intellirule** for the Intellirule editor

- **Guided** for the Guided editor

If the Intellirule editor is not the default editor that opens when you create an action rule, right-click this action rule in the Rule Explorer and click **Open With > Intellirule Editor**.

__ 2. Use the Intellirule editor to start authoring the `grade` rule.

The `grade` rule must define a rule variable to manipulate the Report object that is passed to the rule engine by the `report` parameter.

__ a. Press Ctrl+Space to open the Content Assist menu and double-click **definitions**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Content Assist continues to prompt you, so double-click **set <variable>**, and then double-click **variable1**.

__ c. Put the cursor in the space after `set 'variable1'` and press Ctrl+Space to open Content Assist.

__ d. Double-click **to <binding type>**.

The Content Assist menu now prompts you with a vocabulary list.

__ e. You want the variable to be of type Report, so scroll through the vocabulary list to find and double-click **a report**.

__ f. In the Content Assist menu, double-click the semicolon (;).

The semicolon is required in the Intellirule editor, or to add another clause to your statement. For this rule, the variable definition is complete.

__ g. When prompted, select **if <condition>**.

The editor automatically formats the condition statement to start on a new line.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ h. Before you continue with the condition statement, go back to the variable definition and type over **'variable1'** to change it to: `'the report'`

Important

Make sure that you keep your variable name enclosed with single quotation marks (`'`).

__ 3. Now that you see how Content Assist can guide you to build the rule, complete the rule statement to match this content:

```
definitions
  set 'the report' to a report;
if
  the amount of the loan of 'the report' is more than 1000000
then
  set the grade of 'the report' to "GOLD";
```

__ 4. Save your work and close the editing window.

__ 5. In the `loan` rule package, create a second action rule called: `invalid corporate score`

This rule must refuse a loan application when the corporate score is less than 1000.

__ 6. For this rule, instead of using the Content Assist menu, type the rule content directly into the editor to match this content:

```
definitions
   set 'the report' to a report;
if
   the corporate score in 'the report' is less than 1000
then
   in 'the report', refuse the loan with the message "Corporate score less
   than 1000";
```

Hint

You can copy the rule content from the `author.txt` file and paste directly into the Intellirule editor. The `author.txt` file is in the `<LabfilesDir>\code` directory.

Press Ctrl+Shift+F to format the rule.

Note

Notice that you must use the correct punctuation to avoid errors in the rule statement.

__ 7. Save the rule (Ctrl+S).

__ 8. Create another action rule in the `loan` rule package called: `too big loan`

This rule should refuse a loan application when the loan is more than 10,000,000.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Questions

How do you write this action rule in the Intellirule editor?

Answer

The body of the action rule that is called `too big loan` must be:

```
definitions
  set 'the report' to a report;
if
  the amount of the loan of 'the report' is more than 10000000
then
  in 'the report', refuse the loan with the message "The loan amount is too big";
```

Hint

You can find the text of the `too big loan` rule in the `author.txt` file in the `<LabfilesDir>\code` directory.

__ 9. Save your work and close the editing windows.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Authoring actions rules in the Guided editor

Use the Guided editor to author the remaining action rules to fully implement the Loan scenario. As in the previous section, you author these action rules by using rule variables and automatic variables.

__ 1. Create an action rule in the `loan` rule package, named: `too long loan duration`

The rule must refuse a loan application when the loan duration is strictly greater than 200.

By default, the rule opens in the Intellirule editor.

__ 2. Close the editing window for the rule, and then in Rule Explorer, right-click the rule and click **Open With > Guided Editor**.

__ 3. Define the rule content to match this text:

```
definitions
   set the report to a report
if
   the duration (in years) of the loan of the report is more than 200
then
   in the report, refuse the loan with the message The loan duration is too
   long
```

__ a. Start with the variable definition statement by clicking **definitions**.

__ b. Click **variable1** and set the variable name by typing: `the report`

__ c. Click **<select a choice>** and click **an <object>** from the list.

__ d. Click **<select an object>** and select **a report** from the list.

__ e. Continue editing the condition and action statements to complete the rule.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Hint

To access the **is more than** BAL construct, click **is**.

For the refusal message, you can type directly in the **enter a string** field.

Your rule should match the following statements:

```
definitions
   set the report to a report
if
   the duration (in years) of the loan of the report is more than 200
then
   in the report, refuse the loan with the message The loan duration is too
   long.
```

Questions

What differences did you identify between the Intellirule editor and the Guided editor?

© Copyright IBM Corp. 2019
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Answer

With the Intellirule editor, you must use punctuation, such as semicolons (`;`) at the end of definitions and actions, single quotation marks (`'`) around variables, and double quotation marks (`"`) around String constants.

The Guided editor does not require these extra characters.

__ 4. Save your work.

__ 5. Close the editing window.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Authoring rules with parameters

For this part of the exercise, you use a parameter instead of the local rule variable. Recall that to define a parameter:

- You first create a variable set.
- Then, you bind the variables to parameters.

Reminder

For more information about defining parameters, see Section 4.2, "Creating variables for parameters" and Section 4.3, "Binding the variables to parameters" in Exercise 2, "Setting up decision services".

__ 1. Expand **loan-bom > bom > model > training_loan > Report**.

Notice the `Report.approved` member. This member can be used to record the decision that the rule engine returns.

Questions

Which members of the `Report` class might be useful for recording output results from the rule engine?

Answer

This list includes some examples:

- `messages`
- `insuranceRequired`
- `insuranceRate`
- `validData`

Because objects of type `Report` are designed to carry output information, you can create an output parameter that is based on this object type.

__ 2. Add a variable set called `my parameters` to the `loan-rules` decision service.

__ a. In Rule Explorer, select the **loan-rules** decision service.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Make sure the Rule Project Map view is open, and in the “Design signature” part of the Operation Map, click **Add variable set**.

__ c. In the New Variable Set wizard, in the **Name** field, type: `my parameters`

__ d. Click **Finish**.

The Variable Set editor opens.

__ 3. Create a variable called `report`.

__ a. In the Variable Set editor, click **Add**.

__ b. Overwrite the default values in the **Name**, **Type**, and **Verbalization** columns by entering the following information:

- **Name**: `report`
- **Type**: `training_loan.Report`

Reminder

To select the `training_loan.Report` type:

- In the **Type** column, click `java.lang.String`, and click the ellipses (**...**) to open the Types window.
- In the **Choose a type** field, type: `report` and in the **Matching types** field, double-click **Report**.
- Click **OK**.

- **Verbalization**: `the loan report`

__ c. Save your work and close the **my parameters** tab.

__ 4. Bind the report variable to an output parameter.

__ a. In the “Design signature” part of the Operation Map, click **Bind variables**.

The Decision Operation Signature editor opens.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. In the **Eligible variables** section, expand **my parameters**.

__ c. Select **report** and drag it to the Output Parameters table.

__ d. Save your work and close the **loan-rules operation** tab.

__ 5. Rewrite the `grade` rule to use the `report` parameter.

You can use your existing local rule variable to manipulate the `report` parameter (which shows up in the vocabulary list as “the loan report”).

You can also delete the local rule variable and use the parameter directly.

__ a. In Rule Explorer, expand **loan-rules > rules > loan** and double-click the **`grade`** rule.

__ b. Delete the **definitions** part of the rule, which includes these lines:

```
definitions
  set 'the report' to a report
```

You should now see errors in the rule.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. In the condition and action statements, double-click **'the report'** and double-click **'the loan report'** to replace it with the new parameter.

The errors should now be gone.

__ 6. Save your work (Ctrl+Shift+S).

__ 7. When you are finished editing rules, close the rule editors.

__ 8. In the Problems view, verify that no messages are listed.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

This exercise looked at how to author action rules in the Intellirule editor and in the Guided editor, by using parameters, rule variables, or automatic variables. You also saw how to create and work with rule templates to support the rule authors in Decision Center.

(Optional) To see the solution to this exercise, switch to a new workspace and import the project Ex 07: Authoring rules > 02-answer.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 9. Authoring decision tables

Estimated time

00:45

Overview

In this exercise, you learn how to author decision tables.

Objectives

After completing this exercise, you should be able to:

- Use the decision table editor to create a decision table

Introduction

In this exercise, you work with patterns of rules to implement them as decision tables.

This exercise includes these tasks:

- Section 1, "Authoring a decision table"

Requirements

You should complete these exercises before proceeding:

- Exercise 8, "Authoring action rules"

This exercise uses the following files, which are installed in the `<InstallDir>\studio\training` directory:

- Start project: `Dev 08 - Authoring decision tables\01-start`
- Solution project: `Dev 08 - Authoring decision tables\02-answer`

You can use the solution project as described in "Exercise review and wrap-up" on page 9-16.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Authoring a decision table

In this part of the exercise, you author the following decision table, which was given as an example during the lectures.

This decision table defines the following five rules:

- If the duration of the loan is less than five years
  then set the yearly interest rate of the loan to 5.0
- If the duration of the loan is between five years and eight years
  then set the yearly interest rate of the loan to 5.8
- If the duration of the loan is at least nine years and less than 12 years
  then set the yearly interest rate of the loan to 6.7
- If the duration of the loan is between 12 years and 16 years
  then set the yearly interest rate of the loan to 7.4
- If the duration of the loan is at least 17 years
  then set the yearly interest rate of the loan to 7.9

By creating this table, you also learn how the operators that you can use in the decision table editor (`<`, `[..]`, and others) correspond to BAL operators (`is less than`, `is between`, and others).

1.1. Setting up your environment for this exercise

__ 1. In Rule Designer, switch to a new workspace:

__ a. From the **File** menu, click **Switch Workspace > Other**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. In the Workspace Launcher window, enter the path:

`<LabfilesDir>\workspaces\dtable`

__ c. Click **Launch**.

__ 2. Close the Welcome view.

__ 3. Use the Samples Console to import the exercise start project.

__ a. Click the **Open Perspective** icon.

__ b. In the Open Perspective window, click **Samples Console** and click **Open**.

__ c. In the **Rule Designer** section, expand **Training > Ex 08: Authoring decision tables > 01-start**, and click **Import projects**.

__ 4. When the workspace finishes building, close the Help view.

1.2. Creating the table

__ 1. In the `loan` rule package, create a decision table named: `rate`

__ a. In Rule Explorer, expand the **`loan-rules`** > **`rules`** project.

__ b. Right-click the **`loan`** rule package, and click **New > Decision Table**.

The New Decision Table wizard opens.

__ c. In the **Name** field, type: `rate`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Click **Finish**.

The new decision table opens in the decision table editor, with three condition columns and one action column.

The editor tab is called **Decision Table**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. Open the **General** tab of the decision table editor, and enter the following definitions in the **Preconditions** section:

```
definitions
   set 'the loan' to the loan of 'the loan report';
```

Later, when you define the condition tests and action statements, you use **`the loan`** variable instead of the longer term: `the loan of 'the loan report'`

Hint

You can press Ctrl+Space to open Content Assist to build your precondition, or you can type directly in the editor. If you type directly, you can use Ctrl+Shift+F to format the content.

__ 3. Save your work.

__ 4. Click the **Decision Table** tab to return to the table editor and define the first condition column.

__ a. Double-click the header of the first condition column, which is labeled `A`, to open the Condition Column window.

__ b. In the **Test** section, click **<condition>** and select the appropriate vocabulary from the vocabulary list to build this condition:

```
the duration (in years) of 'the loan' is between <min> and <max>
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. When you are finished editing the test, click **Apply**.

__ d. In the **Title** field, replace `A` with: `Loan duration (years)`

Notice that the placeholders in the condition test are now listed in the **Expression Placeholders** list.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ e. Click **OK**.

__ 5. Right-click each of the next two condition columns and remove them.

__ 6. Define the action column.

__ a. Double-click the action column header, which is now labeled `B`, to open the Action Column window.

__ b. In the **Action** field, click **<action>**, and select the appropriate vocabulary from the vocabulary list to write this action statement:

```
set the yearly interest rate of 'the loan' to <a number>
```

__ c. In the **Action** section, click **Apply**.

__ d. In the **Title** field, replace `B` with: `Yearly rate (%)`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ e. Click **OK** to close the window.

1.3. Completing the table cells with values

The cells in the `rate` decision table are currently empty. To complete the `rate` decision table, you work with operators in the next step.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Hint

To show the list of available operators in the decision table editor, right-click a decision table cell and click **Operator**. To enter an operator in a cell of the decision table, you can select the required operator from this list.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Questions

Can you relate the operators in the decision table editor (such as >, <) to BAL operators?

Answer:
The operators in the decision table editor and the corresponding BAL constructs are as follows:

Decision table editor operators	Corresponding BAL number operators
=	equals
!=	does not equal
in	is one of { ... }
!in	is not one of { ... }
<	is less than
<=	is at most
>	is more than
>=	is at least
[..]	is between ... and ...
[..[	is at least ... and less than ...
]..]	is more than ... and at most ...
]..[	is strictly between ... and ...

You are now ready to complete the `rate` decision table.

__ 1. Review the five rules that you must write and the corresponding lines in the decision table, recalled at the beginning of this exercise. This table lists the values and operators that you use to complete the table.

__ 2. In the decision table editor, first enter the values (not the operator) in the cells of the `rate` decision table:

Loan duration (years)	Yearly rate (%)
< 5	5
>= 5 AND <= 8	5.8
>= 9 AND < 12	6.7
>= 12 AND <= 16	7.4
>=17	7.9

For rows that have only a single value, enter that value in the first column only.

__ 3. Next, you edit the operators that are used for each row in the condition.

__ a. Right-click the first row of the condition column, and in the menu, click **Operator**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. From the **Operator** list, select the “less than” (**<**) operator.

The subcolumns merge automatically to match the operator. Notice that the condition test changes from **is between** to **is less than**.

__ c. Click the second row of the condition column, and consider the condition test (**is between**), which is equivalent to the BAL operator: `[..]`

You want the row to state: `>= 5 AND <= 8`

In this case, these operators are equivalent to **is between** or `[..]`.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Right-click the second row and click `[..]` from the **Operator** menu.

Notice that the cells do not change.

__ e. Right-click the condition column for row 3, and set the condition to **is at least and less than**, which is represented in BAL as: `[..[`

__ f. Complete the remaining rows of the condition column.

__ 4. Delete the empty rows of the decision tables.

__ a. Select all of the empty rows.

__ b. Click **Remove selected row(s)** on the toolbar.

Your table should have five rows, and have the following values.

Loan duration (years)		Yearly rate (%)
Min	Max	
< 5		5
5	8	5.8
[9	12[	6.7
12	16	7.4
>=17		7.9

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Information

In the decision table editor, when you select a row of a decision table, you can see the corresponding rule at the top of the decision table. You can use this useful function to verify what you are writing.

__ 5. Verify the rule statement for each row in the decision table.

__ a. Click **1** in row 1, and hover your mouse to see the full rule statement, including the precondition.

__ b. Click the remaining rows to verify that the rule conditions are correct for each row (or rule).

__ c. Save your work.

__ 6. Take some time to examine the Decision Table Properties window.

__ a. Edit the decision table properties by clicking **Open Decision Table Properties Editor** on the toolbar.

The Decision Table Properties window opens.

With this window, you can specify some general or lock properties of the decision table.

__ b. Open each of the **General**, **Lock**, and **Format** tabs to see the available options.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. On the **General** tab, select **Show rules** and click **OK**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Click **1** in row 1 again to see the rule in the rule pane that is now open In the decision table editor.

Questions

Can you determine how you can lock the preconditions of the decision table to prevent changes?

Answer

To lock the preconditions of the decision table, you must:

- Select the **Lock Preconditions** check box on the **Lock** tab of the Decision Table Properties window.
- Then, select the **Enforce locking rules on this table** check box on the **Lock** tab for locking to take effect.

__ 7. Save your work.

__ 8. Close the decision table editor.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

The exercise looked at how to author a decision table.

(Optional) To see the solution to this exercise, switch to a new workspace and import the project **Ex 08: Authoring decision tables and trees > 02-answer**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 10.Working with static domains

Estimated time

00:45

Overview

In this exercise, you learn how to simplify rule authoring by defining static domains in the BOM.

Objectives

After completing this exercise, you should be able to:

- Create various types of static domains
- Use domains in rules

Introduction

In this exercise, you learn how to create and use static domains.

This exercise includes these sections:

- Section 1, "Exploring a collection domain"
- Section 2, "Working with an enumeration of literals"
- Section 3, "Defining an enumeration of static references"

Requirements

This exercise uses the following files, which are installed in the `<InstallDir>\studio\training` directory:

- Start project: `Dev 09 - Static domains\01-start`
- Solution project: `Dev 09 - Static domains\02-answer`
  - You use the solution project in "Exercise review and wrap-up" on page 10-21.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Exploring a collection domain

In this section, you explore a collection domain to learn how you can use it in a business rule.

Hint

In this exercise, you must write some code and some rule statements. You can find the code snippets and the rules in the `<LabfilesDir>\code\create_domains.txt` file, and you can copy and paste them in Rule Designer.

1.1. Setting up your environment for this exercise

__ 1. In Rule Designer, switch to a new workspace:

__ a. From the **File** menu, click **Switch Workspace > Other**.

__ b. In the Workspace Launcher window, enter the path:

`<LabfilesDir>\workspaces\static_domains`

__ c. Click **Launch**.

__ 2. Close the Welcome view.

__ 3. Use the Samples Console to import the exercise start project.

__ a. Click the **Open Perspective** icon.

__ b. In the Open Perspective window, click **Samples Console** and click **Open**.

__ c. In the **Rule Designer** section, expand **Training > Ex 09: Static domains > 01-start**, and click **Import projects**.

__ 4. When the workspace finishes building, close the Help view.

In your workspace, you now have the following projects in the `trucksAndDrivers-tests` decision service:

- `trucksAndDrivers-bom`
- `trucksAndDrivers-rules`
- `trucksAndDrivers-tests` (main rule project)
- `trucksAndDrivers-xom`

These projects define the business rule solution in which you explore and create domains.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

1.2. Exploring the domain

__ 1. In Rule Explorer, expand **trucksAndDrivers-bom > bom > model > drivers** and double-click the `Driver` class to open it in the BOM editor.

__ 2. In the **Members** section of the Class page, double-click the `drivingAssignments` member.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. On the Member page, notice that the `drivingAssignments` member is defined as a vector.

__ 4. The vector is defined with a collection domain, and each member of the collection is of type `drivers.DrivingAssignment`.

This collection domain was automatically defined when the BOM was generated from the XOM because this XOM attribute is defined as: `Vector<DrivingAssignment>`

__ 5. In Rule Explorer, expand **trucksAndDrivers-rules > rules > domains**.

__ 6. Double-click the `01_collection_domain` rule to open it and see how this rule uses the collection domain.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 7. When you are done, close the rule.

1.3. Testing the rule

With the next steps, you test the `domains\01_collection_domain` action rule.

First, you verify that objects in the working memory exist that match the definitions of this rule. Then, you use the `trucksAndDrivers-tests` project to test this rule in the rule project.

__ 1. Expand the **trucksAndDrivers-xom > src > feeder** folder and double-click the **`WMFeeder.java`** file.

__ 2. In the `WMFeeder.java` file, scroll towards the end of the file to see the `DrivingAssignment` object for the `Driver` with the name: `John Jongle`

__ 3. Use the `trucksAndDrivers-tests` project to test the `01_collection_domain` rule.

__ a. In the Rule Explorer, expand the **trucksAndDrivers-tests > rules** folder and double-click the `testFlow` ruleflow to open it.

__ b. Double-click the **Execute Rule** rule task to edit its properties.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. In the Rule Task properties, click **Rule Selection** and click **Edit**.

The Select Rules window opens.

__ d. In the Select Rules window, replace the content on the right side with the `domains` package and click **OK**.

Now, only the rules of the `domains` package are executed in the `testFlow` ruleflow.

__ e. Save the ruleflow (Ctrl+S).

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ f. In the `trucksAndDrivers-tests` project, right-click the **`Test a rule - Decision service.launch`** configuration, and click **Run As > Decision Operation** to execute the `testFlow` ruleflow.

__ g. Verify that you have the following result in the Console view:

```
Added the pending driving assignment to the list of driving assignments of
John Jongle
```

Note

The following rule is another example of how you can use this collection domain.

```
definitions
   set 'the driver' to a driver ;
if
   there is at least one driving assignment in the driving assignments of 'the
   driver'
then
   ...
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Working with an enumeration of literals

In this section, you define a domain as a list of literal values `{A, B, C, D}` for the `licenseClass` member of the `Driver` and `TruckModel` classes, and then use it to author an action rule.

2.1. Creating the domain

__ 1. In the `trucksAndDrivers-bom` project, expand the **`drivers.Driver`** class and double-click the **`licenseClass`** BOM member to open it in the BOM editor.

__ 2. In the BOM editor, in the **Domain** section, create a domain of literal values:

__ a. Click **Create a domain** to open the Domains window.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. On the Domain Type page of the Domains windows, select **Literals** in the list and click **Next**.

__ c. On the Literals page of the Domains window, click **Add** and type over the new domain value to rename it to: `A`

Hint

If you want, you can click **Add** several times to add the placeholders for your domain values, and then type over the placeholder names later.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Add these values to the domain: `B`, `C`, and `D`

__ e. Click **Finish** to close the Domains window.

You created the domain of type Literals `{A, B, C, D}`.

Domain type: Literals

- A
- B
- C
- D

__ 3. Create the domain of type Literals `{A, B, C, D}` for the `drivers.TruckModel.licenseClass` BOM member as well.

__ 4. Save the BOM (Ctrl+S).

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

2.2. Testing the domain in a rule

In this section, you use your new domains to author a rule.

__ 1. In the `domains` rule package of the `trucksAndDrivers-rules` project, create the action rule named: `02_literal_domain`

__ 2. Enter this rule statement in the rule editor:

```
definitions
   set 'the truck' to a truck where the driver of this truck is not null;
if
   the license class of the driver of 'the truck' is "A" and the license
   class of the model of 'the truck' is one of {"B", "C", "D"}
then
   display the message "The driver (" + the name of the driver of 'the truck'
   + ") is not eligible to drive the truck " + the serial number of 'the
   truck' ;
```

Hint

Recall that you can type directly in the Intellirule editor or you can paste the code from the code snippet in the *<LabfilesDir>*\code\create_domains.txt file.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Because the `licenseClass` BOM member type is a domain of Literals, an enumeration of values is available to you when you author an action rule that uses this `licenseClass` member.

__ 3. After you finish editing, save the rule.

__ 4. Rerun the test project from the toolbar by clicking **Run > Test a rule - Decision service** on the toolbar.

Information

The `testFlow` ruleflow is already correctly configured to execute all rules of the `domains` package. You do not have to explicitly add your new rule to the Execute task.

__ 5. Verify that, in addition to the result from the previous section, you also have the following result:

```
The driver (John Jongle) is not eligible to drive the truck TRUCK-F150
```

__ 6. Close the rule.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Defining an enumeration of static references

A domain that is set as an enumeration of static references specifies a list of references to constants, for example:

```
{static GroupA, static GroupB, static GroupC}
```

You can define attribute types, method return types, and arguments as follows:

- If you have an attribute of type A, you can define a domain of static references on it by using the static attributes of the class A (classic Java enumeration pattern)
- If you have an attribute of a primitive type, you can define a literal domain on it

3.1. Creating the static references Java class

In this section, you change the attribute `gender` in the `Driver` class to use the static attributes of a new class called `GenderType`.

__ 1. Switch to the Java perspective.

__ a. Click the **Open Perspective** icon in the toolbar, which is in the upper-right corner of the Rule Designer window.

__ b. Select **Java (default)** from the list of perspectives, and click **OK**.

__ 2. Create a package and class in the `trucksAndDrivers-xom > src` project.

__ a. Expand **trucksAndDrivers-xom > src**, right-click **src**, and click **New > Package**.

__ b. Name the package: `domains`

__ c. Click **Finish**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Right-click the `domains` package, and click **New > Class**.

__ e. Name the class: `GenderType`

__ f. Leave the other default values and click **Finish**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. When the `GenderType.java` file opens in the editor, define the static attributes in the Java file with this content:

```
package domains;
public class GenderType {
   private final String name;
   public static final GenderType MALE = new GenderType("male");
   public static final GenderType FEMALE = new GenderType("female");
   public static final GenderType UNKNOWN = new GenderType("unknown");
   private GenderType(String _name){
      this.name = _name;
   }
   public String toString(){
      return this.name;
   }
}
```

Hint

You can find this code snippet in the `<LabfilesDir>\code\create_domains.txt` file.

__ 4. Save your work (Ctrl+S).

__ 5. Change the `drivers.Driver` class to use the new `GenderType` for the following class members:

- `gender` attribute
- `getGender` method
- `setGender` method

__ a. Expand **trucksAndDrivers-xom > src > drivers.Driver**.

__ b. Double-click **drivers.Driver.gender**.

__ c. Find the following line:

```
private String gender;
```

```
public class Driver {
    private int age;
    private int nbAccidents;
    private String name;                    // name
    private String licenseClass;            // driver's license class
    private Vector<DrivingAssignment> drivingAssignments;  // curren
    private Vector<VacationRequest> vacationRequests;    // vacation
    private String gender;
```

__ d. Edit the line by changing `String` to `GenderType`.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

It should now read:

```
private GenderType gender;
```

```
public class Driver {
    private int age;
    private int nbAccidents;
    private String name;                    // name
    private String licenseClass;            // driver's license class
    private Vector<DrivingAssignment> drivingAssignments;  // curren
    private Vector<VacationRequest> vacationRequests;    // vacation
    private GenderType gender;
```

__ e. Change the `String` attribute for `drivers.Driver.getGender` and `drivers.Driver.setGender` to `GenderType`.

__ f. Save your work (Ctrl+S).

Notice that you get an error on `GenderType`.

__ 6. To resolve the problem, click the error icon and double-click the **Import Gender Type (domains)** quick fix.

__ 7. Save your work (Ctrl+Shift+S).

__ 8. Also in the `src` folder, update the `feeder.WMFeeder` class that uses the setter method.

__ a. In the `WMFeeder` Java file, replace all occurrences of `"MAN"` with: `GenderType.MALE`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. In the `WMFeeder` class, replace all occurrences of `"UNKNOWN"` with: `GenderType.UNKNOWN`

```
private void createModel() {
    // Definitions of the drivers
    Driver dA20 = new Driver("George Smith", TruckModel.MACK_TRUCK.getLicenseC
    dA20.setAge(20);
    dA20.setNbAccidents(3);
    dA20.setGender(GenderType.MALE);

    Driver dA18 = new Driver("John Jongle", TruckModel.MACK_TRUCK.getLicenseCl
    dA18.setAge(18);
    dA18.setNbAccidents(0);
    dA18.setGender(GenderType.MALE);

    Driver dB23 = new Driver("Marc Lansen", TruckModel.F150.getLicenseClass())
    dB23.setAge(23);
    dB23.setNbAccidents(0);
    dB23.setGender(GenderType.MALE);

    Driver dB37 = new Driver("Jill Northwood", TruckModel.T2000.getLicenseClas
    dB37.setAge(37);
    dB37.setNbAccidents(1);
    dB37.setGender(GenderType.UNKNOWN);
```

__ c. If you have errors on these values, use the **Import Gender Type (domains)** quick fix as you did in Step 4.

After you save the XOM, the Problems view lists errors on the BOM. You correct these errors now by using the BOM Update view.

__ 9. Save your work and close the editing windows for the Java files.

__ 10. Switch back to the Rule perspective.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 11. In the **trucksAndDrivers-bom > bom** folder, right-click `model` and click **BOM Update**.

__ a. In the BOM Update tab in the bottom pane, expand the **Differences and Actions** section.

__ b. Notice the differences that were detected between the XOM and BOM classes:

- The XOM class `domains.GenderType` is not found in the BOM.
- The definition of the attribute `drivers.Driver.gender` differs between the BOM and the XOM.

__ c. In the Actions menu, select **Import the XOM class** and click **Perform and save**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. In the Verbalize BOM window, make sure that you click **Select All** and that you select the **All Methods** check box to verbalize all members and methods of the BOM class `GenderType`.

__ e. Click **Finish**.

__ 12. Update the BOM class.

__ a. Go back to the BOM Update view.

__ b. Expand **Differences and Actions**.

__ c. Select the **Bulk: Update BOM Class (2)** action and click **Perform and save**.

__ 13. In the Configure Verbalization window, click **Finish**.

__ 14. After the BOM is updated, verify the following points:

- A new `GenderType` BOM class exists under `domains` in the BOM and is defined as a domain of type Static References with these values: `FEMALE`, `MALE`, and `UNKNOWN`

- The **Type** field of `drivers.Driver.gender` BOM member is changed to: `GenderType`

__ 15. Close the BOM editor.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

3.2. Authoring an action rule that uses a domain of static references

In this section, you use the `GenderType` static domain of type Static References to author an action rule.

__ 1. In the `domains` rule package of the `trucksAndDrivers-rules` project, create the action rule `03_static_references_domain`:

```
definitions
   set 'the driver' to a driver;
if
   the gender of 'the driver' is UNKNOWN
then
   display the message "Update the personal data of " + the name of 'the
   driver';
```

Reminder

You can type directly in the Intellirule editor or you can paste the code from the code snippet in the *<LabfilesDir>*`\code\create_domains.txt` file.

__ 2. Save and close the rule.

__ 3. Rerun the `trucksAndDrivers-tests` project from the toolbar by clicking **Run > Test a rule - Decision service** on the toolbar.

__ 4. Verify that the following line is now included in the result:

```
Update the personal data of Jill Northwood
```

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

This exercise looked at how to create static domains in the BOM, and how to work with them in rules. You also saw the relationship between domains and the XOM. You explored an existing domain of type Collection and learned how you can use it in an action rule.

(Optional) To see the solution to this exercise, switch to a new workspace and import the project **Ex 09: Static domains > 02-answer**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 11. Working with dynamic domains

Estimated time

01:30

Overview

In this exercise, you learn how to define and use dynamic domains with Microsoft Excel spreadsheets.

Objectives

After completing this exercise, you should be able to:

- Create dynamic domains in Microsoft Excel spreadsheets
- Update and use dynamic domains in rules
- Access and update dynamic domains in Decision Center
- Synchronize dynamic domains between Rule Designer and Decision Center

Introduction

In this exercise, you work in Rule Designer to create a dynamic domain with Microsoft Excel, update the values of a domain, and discover the effects of such updates.

You learn how to resolve inconsistencies across the BOM, XOM, and rules after modifying domain classes. You also learn how to synchronize updated domain values between Rule Designer and Decision Center.

The exercise involves these tasks:

- Section 1, "Creating a dynamic domain in Rule Designer"
- Section 2, "Using the dynamic domain in a rule"
- Section 3, "Updating the dynamic domain"
- Section 4, "Updating the XOM"
- Section 5, "Publishing the BOM and rule projects to Decision Center"
- Section 6, "Examining rules in Decision Center"
- Section 7, "Modifying the dynamic domain in Decision Center"
- Section 8, "Updating Rule Designer from Decision Center"

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Requirements

You should complete these exercises before proceeding:

- Exercise 4, "Working with the BOM"
- Exercise 7, "Exploring action rules"
- Exercise 10, "Working with static domains"

This exercise uses a series of Excel files that are stored in the `<LabfilesDir>\code` directory.

It also uses the following files, which are installed in the `<InstallDir>\studio\training` directory:

- Start project: `Dev 10 - Dynamic domains\01-start`
- Solution project: `Dev 10 - Dynamic domains\02-answer`
  - You use the solution project in "Exercise review and wrap-up" on page 11-39

Important

For this exercise, you work with a series of Microsoft Excel spreadsheets to define the values of the dynamic domain, and their updates. The Excel spreadsheets are stored in the `<LabfilesDir>\code` directory.

During the exercise, you open and read these Excel spreadsheets, but you do not have to modify them.

You can read the Microsoft Excel spreadsheets by using either:

- Microsoft Excel Viewer, which is installed on the computer lab environment that is developed for this course
- Microsoft Excel, which is **not** installed on the computer lab environment that is developed for this course

If you have access to Microsoft Excel, you can *optionally* edit the Microsoft Excel spreadsheets.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Creating a dynamic domain in Rule Designer

In this section, you create a dynamic domain in the BOM to represent the values that are listed in an Excel spreadsheet.

Hint

In this exercise, you must write some pieces of code. You can find the pieces of code to create in the `<LabfilesDir>\code\create_domains.txt` file, and you can copy and paste them in Rule Designer.

1.1. Setting up your environment for this exercise

__ 1. In Rule Designer, switch to a new workspace:

__ a. From the **File** menu, click **Switch Workspace > Other**.

__ b. In the Workspace Launcher window, enter the path:

`<LabfilesDir>\workspaces\dynamic_domain`

__ c. Click **Launch**.

__ 2. Close the Welcome view.

__ 3. Use the Samples Console to import the exercise start project.

__ a. Click the **Open Perspective** icon.

__ b. In the Open Perspective window, click **Samples Console** and click **Open**.

__ c. In the **Rule Designer** section, expand **Training > Ex 10: Dynamic domains > 01-start**, and click **Import projects**.

__ d. When the workspace finishes building, close the Help view.

Information

In your workspace, you now have the `trucksAndDrivers-tests` decision service, which has the following projects:

- `trucksAndDrivers-bom`
- `trucksAndDrivers-rules`
- `trucksAndDrivers-tests`
- `trucksAndDrivers-xom`

These projects define the business rule solution to which you must add the dynamic domain.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Scenario:

In the Trucks and Drivers rule application, drivers might submit requests for vacations. In their current state, requests for vacations are not associated with any specific type.

After a discussion with the human resources department, business analysts tell you that the type of vacation request must be differentiated for statistical purposes.

To start the work, the human resources department provides you with an Excel spreadsheet that lists their current list of possible types of vacation requests. They also indicate that this list might change regularly, and ask that it remain easy to update when they have changes.

Because of potential change, you cannot use a static domain to implement the requirement. Instead, you use a dynamic domain, which is an enumerated domain (or list of values) that can change dynamically.

1.2. Import the domain file

__ 1. Open the `<LabfilesDir>\code\vacationRequestTypes-1.xls` file and verify that each row of this file contains three columns, where:

- The first column represents the name to the domain value.

  Example: `JuryDuty`

- The second column represents the code in the BOM-to-XOM mapping.

  Example: `return "V78";`

- The third column represents the verbalization of the domain value.

  Example: `Jury Duty`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. In Rule Designer, import the Excel domain file into the `resources` folder of your BOM project.

__ a. Expand the **`trucksAndDrivers-bom`** project, right-click the **`resources`** folder of the `trucksAndDrivers-bom` project, and click **Import**.

The Import window opens.

__ b. In the Select pane of the Import window, select **General > File System**, and click **Next**.

__ c. In the File System pane, click **Browse**, which is next to the **From directory** field, select the `<LabfilesDir>\code` folder, and click **OK**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Back in the Import window, select **vacationRequestTypes-1.xls** and click **Finish**.

The `vacationRequestTypes-1.xls` file is now visible under the `resources` folder of the `trucksAndDrivers-bom` project.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. In your workspace, rename the `vacationRequestTypes-1.xls` file to: `vacationRequestTypes.xls`

__ a. Right-click the file and click **Refactor > Rename**.

__ b. In the Rename Resource window, in the **New name** field, change the value to: `vacationRequestTypes.xls`

__ c. Click **OK** to close the window.

1.3. Create the domain

__ 1. In Rule Explorer, expand **trucksAndDrivers-bom > bom > model** and double-click **domains** to open the `domains` package in the BOM editor.

__ 2. Create a BOM class called `VacationRequestType` in the **domains** package.

__ a. In the **Business Object Model Entry: model** section, make sure that **domains** is selected.

__ b. Click **New Class**.

__ c. In the New BOM Class window, in the **Name** field, enter: `VacationRequestType`

__ d. Click **Finish**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. Edit the new `VacationRequestType` BOM class in the BOM editor to associate a dynamic domain with that class.

__ a. In the **Business Object Model Entry** section, make sure that **VacationRequestType** is selected, and click **Edit**.

__ b. In the **Domain** section, click **Create a domain**.

The Domains window opens.

__ c. In the Domains window, go to the **Dynamic Domains** section, select **Excel**, and click **Next**.

The Excel pane of the Domains window opens.

__ d. In the Excel pane, set the **Excel file** field to `vacationRequestTypes.xls`, which is the only available choice in the list.

Note

This choice is available in this window because the `vacationRequestTypes.xls` file is present in the `resources` folder of the BOM project.

The `vacationRequestTypes.xls` file contains only the `vacationRequestTypes` sheet. Rule Designer automatically sets the value of the **Sheet** field to the name of that unique sheet.

__ e. Select **Table with header** to indicate that the first row in the `vacationRequestTypes` sheet is the header for the columns in that sheet.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Information

It is a good practice to set a header row in the Excel spreadsheet for dynamic domains. This header helps distinguish the purpose of each column.

Also, Rule Designer uses the header values as indications for you to select the appropriate columns when you configure the dynamic domain, as you see next.

__ f. In the **Value column** field, select **Value**, which is the header name for the column that gives the values of the dynamic domain in the spreadsheet.

__ g. In the **BOM to XOM column** field, select **BOM to XOM**, which is the header name for the column that provides the BOM-to-XOM mapping in the spreadsheet.

__ h. In the table, click the cell at the intersection of the row **English (default)** and the column **Label column**, and select **Label** in the list.

Label is the header name for the column that gives the labels of the dynamic domain in the spreadsheet.

The `vacationRequestTypes.xls` file does not define a documentation column, so you can leave the cells of the **Documentation column** empty.

__ i. Click **Next**. (If the **Next** button is not enabled, click anywhere in another cell to make sure that the **Label** header is set.)

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The next pane opens, where you can add custom properties. For this domain, you have no properties.

__ j. Click **Finish** to close the Domains window.

__ 4. Save the BOM (Ctrl+S).

The dynamic domain is created and its values are available in Rule Designer. The **Synchronize with dynamic values** link is now visible in the **Domain** section of the Class page.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

However, the Problems view indicates an error because the XOM does not include a `domains.VacationRequestType` class to translate your new BOM class.

Questions

How can you resolve this problem?

Answer

The `domains.VacationRequestType` class in the XOM does not exist. To resolve this problem, set the **Execution name** in the BOM-to-XOM mapping of the `domains.VacationRequestType` BOM class to an existing class.

Questions

Can you figure out which class you must use to set the **Execution name** in the BOM-to-XOM mapping?

Hint

Look at the content of the cells in the **BOM to XOM** column in the `vacationRequestTypes.xls` file.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Answer

The class that you require depends on the type for the values that the BOM-to-XOM mapping returns. In the `vacationRequestTypes.xls` file, the **BOM to XOM** column defines a BOM-to-XOM mapping. For example, the mapping for Jury Duty is:

```
return "V78";
```

The execution values associated with the values in the BOM dynamic domain are `String` objects, such as `"V78"`. The **Execution name** in the BOM-to-XOM mapping must be:

```
java.lang.String
```

__ 5. To correct the problem in the `VacationRequestType` BOM class, define the BOM-to-XOM mapping.

__ a. Scroll down to the **BOM to XOM Mapping** section and expand this section.

__ b. Next to the **Execution name** field, click **Browse**.

__ c. In the **Choose a type** field, type `String`

__ d. Select **String** from the list, and click **OK**. The type field is now set to: `java.lang.String`

__ 6. Save your work.

__ 7. Verify that the Problems view no longer indicates any errors on this new class. You can ignore the warnings.

1.4. Explore the dynamic domain

In this section, you continue working in the BOM editor to examine your new dynamic domain.

__ 1. In the **Custom Properties** section of the `VacationRequestType` BOM class, verify that two custom properties were defined.

- The `domainProviderResource` property gives the name of the Excel spreadsheet that is used as the source for the values of the dynamic domain.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

You must modify this value when you want to change the source Excel spreadsheet for your domain.

- The `domainValueProviderName` property gives the name of the classes that are used to manipulate this spreadsheet and transform its values into the dynamic domain.

__ 2. In the **Members** section, verify that you can see all the values that are defined in the `vacationRequestTypes.xls` file.

__ a. Double-click the **VolunteerFireAndRescue** BOM member to open it.

__ b. Verify that this BOM member is both static and final.

__ c. Verify that the **Type** of this BOM member is `domains.VacationRequestType`.

__ d. Verify that the verbalization of this BOM member is `Volunteer Fire and Rescue` as shown in the **Label** field of Microsoft Excel file.

__ e. Expand the **BOM to XOM Mapping** section and verify that the getter method of this BOM member is as shown in the Microsoft Excel file:

```
return "K29";
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Using the dynamic domain in a rule

Now that the dynamic domain exists in the BOM of the rule project, use it to complement the `VacationRequest` BOM class per the request from the human resources department. You then author rules that are based on this type.

__ 1. Update the `VacationRequest` BOM class by adding a `type` BOM attribute that business users can read and that is based on the dynamic domain.

__ a. Return to the Package page of the BOM editor, expand the drivers package, and double-click the `VacationRequest` class to open it on the Class page.

__ b. In the **Members** section, add a BOM attribute named: `type`

__ c. Set the **Type** field to `domains.VacationRequestType` by clicking **Browse** and typing `VacationRequestType` and clicking **OK**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Hint

As you start typing, the you click **Browse** to find the type, you can filter the list of options by starting to type the class name that you are looking for.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Click **Finish**.

__ 2. Set `type` to **Read Only** and create a default verbalization for it.

__ a. From the Members list, double-click **type** to open the Member page.

__ b. In the **General Information** section, select **Read Only**.

__ c. Create a default verbalization for `type`.

__ 3. Save the BOM.

Questions

Are you done preparing this attribute?

Answer

Your new `type` attribute for the `VacationRequest` BOM class is not associated with any attribute in the XOM. You cannot execute this business rule solution without an error at run time.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The Problems view shows this error:

```
Cannot find attribute 'type' in execution class 'drivers.VacationRequest'
```

You solve this problem later in this exercise, in "Updating the XOM" on page 11-23. For now, ignore the error and try to use the new dynamic domain in a rule.

__ 4. Now, write an action rule that uses this new `type` BOM attribute in Rule Designer.

__ a. In Rule Explorer, expand the **`trucksAndDrivers-rules`** project.

__ b. In the **domains** package, add an action rule that is named `05_dynamic_domain` with the following code:

```
definitions
   set 'the driver' to a driver;
   set 'a vacation request' to a vacation request in the vacation requests
   of 'the driver';
if
   the type of 'a vacation request' is one of { Administrative,
   Compensatory, Educational }
then
   print "The driver " + the name of 'the driver' + " requested vacation
   for Administrative, Compensatory, or Educational reason";
```

Hint

You can find the code for this rule in the `create_domains.txt` file in the `<`*`LabfilesDir`*`>\code` directory.

Press Ctrl+Shift+F to format the rule.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. Notice how the rule editor shows the verbalization of the values in the dynamic domain, while you edit `one of { }`.

__ 5. Save the rule.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Updating the dynamic domain

Requirements

The human resources department indicates that the `Compensatory` value is no longer valid and must be replaced with the `Other Vacation` value. You must update the dynamic domain by modifying the Excel file and dynamically updating the dynamic domain of the BOM in Rule Designer. You also verify the consequences of the update.

Modifying the source Excel spreadsheet and update the dynamic domain

__ 1. In Rule Designer, import the `<LabfilesDir>\code\vacationRequestTypes-2.xls` file into the `resources` folder of the `trucksAndDrivers-bom` project.

__ 2. In the `resources` folder and double-click **`vacationRequestTypes.xls`**. It opens in Excel Viewer.

__ 3. In Excel Viewer, notice that line 3 includes these values:

Value: `Compensatory`

BOM to XOM: `return "B34";`

Label: `Compensatory`

__ 4. In Rule Explorer, double-click the **`vacationRequestTypes-2.xls`** file, and verify that its content is the same as the content of the `vacationRequestTypes.xls` file, except for line 3, which has these values:

Value: `Other`

BOM to XOM: `return "O22";`

Label: `Other Vacation`

__ 5. Close both the `vacationRequestTypes.xls` file and the `vacationRequestTypes-2.xls` file.

__ 6. In Rule Designer, rename the `vacationRequestTypes.xls` file back to: `vacationRequestTypes-1.xls`

__ 7. In Rule Designer, rename the `vacationRequestTypes-2.xls` file to: `vacationRequestTypes.xls`

When you rename the new Excel spreadsheet, Rule Designer uses it as the source for the values in the dynamic domain.

__ 8. On the **Package** tab of the BOM editor, double-click the `domains.VacationRequestType` BOM class to open it.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 9. Click **Synchronize with dynamic values** in the **Domain** section.

The removed `Compensatory` value is still visible, both under **Synchronize with dynamic values** and in the **Members** section. However, this value is marked as *deprecated* in the BOM editor (as you verify next).

The added `Other` value is also visible, both under **Synchronize with dynamic values** and in the **Members** section.

Important

The **Members** section lists all the static references that are, or were, part of this dynamic domain. The references that correspond to values that are no longer present in the dynamic domain are marked as *deprecated*.

Deprecated values are not removed and might still be used to author rule artifacts. However, because they are marked as deprecated, these values are underlined in the edited rule artifacts that use them, and associated warnings are added in the Problems view of Rule Designer.

__ 10. Save the BOM.

__ 11. Verify that the `Compensatory` value is deprecated.

__ a. In the Problems view, verify that the following warning messages exist:

- `'Compensatory' is deprecated.`
- `[BOM] GBRMO0011W: Member domains.VacationRequestType.Compensatory is deprecated.`

__ b. In the **Members** section of the `domains.VacationRequestType` BOM class, double-click **`Compensatory`**.

__ c. On the Member page for the `Compensatory` BOM member, verify that the **Deprecated** check box is selected.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

After modifying the values of the dynamic domain, you must verify that the rule artifacts based on this dynamic domain are still correct. In the present case, you must verify only the `05_dynamic_domain` rule.

__ 12. Reopen the `05_dynamic_domain` rule in the `domains` rule package of the `trucksAndDrivers-rules` project.

__ a. Verify that the term `Compensatory` is underlined, and that the associated warning is the same as in the Problems view:

```
'Compensatory' is deprecated
```

__ b. In the ***if*** part of the rule, click `Compensatory` and replace it with `Other Vacation`.

__ c. In the then part of the rule, replace Compensatory in the message by typing `Other Vacation`.

Note

In the action statement, this term is part of a string.

You obtain the following rule:

```
definitions
  set 'the driver' to a driver ;
  set 'a vacation request' to a vacation request in the vacation requests of
'the driver' ;
if
  the type of 'a vacation request' is one of { Administrative, Other
Vacation, Educational }
then
  print "The driver " + the name of 'the driver' + " requested vacation for
Administrative, Other Vacation, or Educational reason";
```

__ 13. Save the `05_dynamic_domain` rule.

__ 14. Verify that the `'Compensatory' is deprecated` warning is no longer present in the Problems view.

__ 15. Close the rule.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Information

If you define multiple dynamic domains in a BOM entry, you can update or repopulate the values of all of them from the Package page of the BOM editor.

__ 1. On the Package page of the BOM editor, click **Update the dynamic domains of this BOM entry** in the **Tasks** section.

The Dynamic Domains window opens.

__ 2. Click **Finish** and save your work.

The Dynamic Domains window closes, and Rule Designer updates the selected dynamic domains.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 4. Updating the XOM

You must now make sure that the complete business rule solution is executable by updating its XOM to reflect the additions in the BOM.

4.1. Identifying the inconsistency problem

First, you identify the inconsistency in the rule project.

__ 1. Reopen the Problems view. (If it is closed, you can click **Window > Show View > Other > General > Problems**.)

__ 2. Verify that the Problems view contains the following error:

```
[B2X] GBREX0021E: Cannot find attribute 'type' in execution class
'drivers.VacationRequest'
```

This message indicates that the `type` attribute cannot be found in the XOM class that is associated with the `drivers.VacationRequest` BOM class.

__ 3. If you cannot see this error message in the Problems view, make sure that the BOM is selected against the XOM while you build the project, and rebuild the project.

__ a. Go to **Window > Preferences**, and then go to **Rule Designer > Build**.

__ b. On the Build page, make sure that the **Perform BOM to XOM checks during build** check box is selected, and then click **Apply and Close**.

__ c. On the **Project** menu, click **Clean** and then click **Clean** to clean and rebuild all projects in your workspace.

4.2. Correcting the inconsistency problem

__ 1. Expand **trucksAndDrivers-xom > src > drivers** and open the `VacationRequest` class.

__ 2. Add a private attribute:

- **Name**: `type`
- **Type**: `String`

__ 3. Create the corresponding setter and getter methods.

```
public void setType(String type) {
   this.type = type;
}

public String getType() {
   return type;
}
```

__ 4. Save the XOM.

The error that is related to this inconsistency in the BOM-to-XOM mapping is no longer visible in the Problems view.

__ 5. Close the `VacationRequest.java` file.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 5. Publishing the BOM and rule projects to Decision Center

In this section, you publish the `trucksAndDrivers-bom` project and the `trucksAndDrivers-rules` project to Decision Center to create the corresponding projects in Decision Center.

5.1. Publishing the decision service

__ 1. If the sample server is not started, start it now (by clicking the Windows **Start** menu > **Start Sample Server** shortcut).

Starting the server might take a few minutes.

__ 2. Publish `trucksAndDrivers-tests` as an ungoverned decision service.

__ a. In the Rule Explorer, right-click **`trucksAndDrivers-tests`** and click **Decision Center > Connect**.

The Decision Center configuration wizard opens.

__ b. Enter the connection entries:

- **URL**: `http://localhost:9090/teamserver`
- **User name**: `rtsAdmin`
- **Password**: `rtsAdmin`

__ c. Click **Connect** and when the connection is established, click **Next**.

__ d. In the Synchronization Settings window, do not select the **Use Decision Governance** check box, but in the **Select a branch or release field**, choose **main**, and then click **Next**.

__ e. On the Decision Service Dependent Projects page, make sure that **trucksAndDrivers-bom** and **trucksAndDrivers-rules** are selected, and click **Finish**.

Information

If you see a Secure Storage window, click **No**.

__ f. When synchronization completes, no changes are found, so you can click **OK** to close the Synchronize Complete window.

__ g. Click **No** when prompted to switch to the Team Synchronizing perspective.

You do not have to open the Team Synchronizing perspective because it is empty.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Important

After you publish the `trucksAndDrivers-rules` project to Decision Center, do not disconnect. By keeping Rule Designer and Decision Center connected, you do not have to establish this connection again in the next steps.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 6. Examining rules in Decision Center

In this section, acting as a business user in Decision Center, you work with the dynamic domain and the rule artifacts that you published from Rule Designer.

6.1. Opening the published projects in Decision Center

__ 1. Sign in to the Decision Center Business console as an administrator.

__ a. Open the Business console from the Windows **Start** menu, by clicking the **Decision Center Business console** shortcut.

__ b. Sign in by entering `rtsAdmin` in both the **Username** field and the **Password** field.

__ 2. Make sure that you are on the **Library** tab.

__ 3. Open the main branch of the `trucksAndDrivers-tests` decision service, and enable all decision artifacts for viewing in the navigator.

__ a. In the Decision Services list, find the `trucksAndDrivers-tests` decision service and click the white space to expand the section.

__ b. Click **main** to open the `main` branch of `trucksAndDrivers-tests`.

__ c. In the **Decision Artifacts** tab, click **Types (1/5)**, click **All Types**, and click **Apply**.

__ 4. Open the `05_dynamic_domain` action rule in the rule editor.

__ a. In the **Decision Artifacts** tab navigator pane, expand **trucksAndDrivers-rules** and click **domains**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. In the rule list for the **domains** folder, hover the mouse pointer over the `05_dynamic_domain` action rule and click the **Edit this rule** icon (pencil).

Information

You can also open the rule in the rule editor by clicking the rule to open the rule preview, and then in the toolbar, clicking **Edit this rule**.

The `05_dynamic_domain rule` is now open in the editor.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. In the ***if*** part of the rule, double-click one of the vacation request values, such as **Administrative**, to see the list of domain values.

__ 6. Click **Cancel** to exit the rule editor.

__ 7. Click the **main** breadcrumb to return to the **Decision Artifacts** tab.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 7. Modifying the dynamic domain in Decision Center

You learned how to use Excel source files for dynamic domains in rules that are authored in Rule Designer. Next, you see how the domain can be modified and used in rules that are authored in Decision Center.

Note

If you are using the VMware image that was developed for this course, you cannot modify Excel files. Instead, the modified domain values are provided in the `vacationRequestTypes-3.xls` file in the `<LabfilesDir>\code` directory. You delete the Excel file that is in use and replace it with: `vacationRequestTypes-3.xls`

7.1. Viewing and modifying the values of the dynamic domain

__ 1. View the current dynamic domain data on the **Model** tab.

__ a. On the toolbar, click **Model**. If you do not see the Model tab, click the right arrow to show the other available tabs.

__ b. In the Dynamic Domains page, under **Domain**, expand **`domains.VacationRequestType`** to see the list of vacation request types from the `vacationRequestTypes.xls` spreadsheet.

Information

After you modify the `vacationRequestTypes.xls` spreadsheet, you return to the **Model** tab to apply the changes.

__ 2. View and prepare the `vacationRequestTypes-3.xls` file for use in Decision Center.

__ a. In Windows Explorer, in the `<LabfilesDir>\code` directory, open the `vacationRequestTypes-3.xls` Excel file to view the modified domain values.

The first line, which defined the `Administrative` value, is replaced with the `Training` value.

__ b. Make a backup of this file by copying it and pasting it to get the file: `vacationRequestTypes-3 - Copy.xls`

__ c. Rename the `vacationRequestTypes-3 - Copy.xls` file as: `vacationRequestTypes.xls`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. In Decision Center, return to the **Decision Artifacts** tab and delete the existing `vacationRequestTypes.xls` file in the `trucksAndDrivers-bom` project.

__ a. In the Business console, click the left arrow on the tab bar and click the **Decision Artifacts** tab.

__ b. In the Decision Artifacts navigator pane, expand **trucksAndDrivers-bom** and click the **Resources** folder.

In the Resources pane, you can see the domain Excel spreadsheets that you imported earlier in this exercise.

__ c. In the Resources list, hover the mouse pointer over the `vacationRequestTypes.xls` file and click the **Delete this resource** icon (trash can).

Information

You can also select the **vacationRequestTypes.xls** check box and click the **Delete the selected items** icon at the top of the Resources list.

__ d. When prompted to confirm the deletion, click **Yes**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. Import the new file to Decision Center.

__ a. In the Resources pane, click **Create an artifact** (the plus [+] sign), and then click **New Resource**.

__ b. In the New Resource pane, click **Choose** next to the **File** field.

__ c. In the File Upload window, go to `<LabfilesDir>\code`, select the `vacationRequestTypes.xls` Excel spreadsheet, and click **Open**.

__ d. In the upper-right area of the New Resource pane, click the **Save changes** icon.

__ e. Verify that the `vacationRequestTypes.xls` file is now visible in the **Resources** list.

Information

You can also replace the file by editing the file options:

- In the Resources list, click the file name to open the preview pane.
- Click **Edit this resource** (the pencil icon).

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

- Click **Refresh**.

- In the File Upload window, browse to the updated file, select it, and click **Open**.
- Back in the preview pane, click **Save** (the disk icon).

__ 5. Apply the changes from the `vacationRequesttypes.xls` file to the BOM.

__ a. Click the **Model** tab.

__ b. Notice that the Changes column lists **domains.VacationRequestType**.

__ c. Under Changes, expand **domains.VacationRequestType**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In the Changes list, you can see that the `Administrative` type was removed and the `Training` type was added.

__ d. Select the **Domain** check box, and click **Apply changes**.

__ e. When you are asked to confirm the update, click **Yes**.

Troubleshooting

If Decision Center hangs when you try to apply the changes, close the browser and reopen the Business console, and then retry this step.

__ 6. Under Domain, expand **domains.VacationRequestType** again to see the updated list.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.2. Modifying the domain value in the rule

In this section, you examine the consequences of your changes to the BOM in the `trucksAndDrivers-rules` project.

__ 1. Go back to the **Decision Artifacts** tab.

__ 2. In the Decision Artifacts navigator pane, expand **`trucksAndDrivers-rules`** and click **domains**.

__ 3. Click `05_dynamic_domain` to open the rule preview.

__ 4. Confirm that the rule has a warning:

`'Administrative' is deprecated.`

Questions

Why is there a warning?

Answer

The `Administrative` value no longer exists in the dynamic domain, so it is marked as `deprecated` (similar to what occurs in Rule Designer).

__ 5. Click **Edit this rule** to edit the ***if*** and ***then*** parts of the `05_dynamic_domain` rule.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 6. Replace `Administrative` with: `Training`

Reminder

In the action statement, `Administrative` is part of a string.

__ 7. The rule should state:

```
definitions
   set 'the driver' to a driver ;
   set 'a vacation request' to a vacation request in the vacation requests of
   'the driver' ;
if
      the type of 'a vacation request' is one of { Training, Other Vacation,
      Educational }
then
      print "The driver " + the name of 'the driver' + " requested vacation
      for Training, Other Vacation or Educational reason";
```

__ 8. Click **Save** and then click **Create New Version** to save your changes to the `05_dynamic_domain` rule.

__ 9. Verify that the rule no longer has a warning.

__ 10. Sign out and close the Business console.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 8. Updating Rule Designer from Decision Center

In Decision Center, you changed both the rule project and the BOM. You must now update the Rule Designer copy of the BOM and rule project by synchronizing with Decision Center.

8.1. Synchronizing the rule project

__ 1. In Rule Explorer, synchronize the `trucksAndDrivers-tests` decision service with Decision Center.

   __ a. Right-click the `trucksAndDrivers-tests` decision service, and click **Decision Center > Synchronize with Decision Center**.

   __ b. In the Synchronization Settings window, click **Finish**.

__ 2. When prompted to open the Team Synchronize perspective, click **Yes**.

__ 3. In the Team Synchronize perspective, view the artifacts that must be synchronized.

   Notice that both the **trucksAndDrivers-bom - main** and **trucksAndDrivers-rules - main** projects are listed.

Information

The **trucksAndDrivers-bom - main** project is listed because you added `Training` in the dynamic domain and used this value in the rule in Decision Center.

The **trucksAndDrivers-rule - main** project is listed because you added the `05_dynamic_domain` rule in the Business console.

If you expand both rule projects, the blue arrows that point left on the various decision artifacts indicate that Decision Center has a more recent version than Rule Designer.

The suggested action for the changes to both of these rule projects is to *update* Rule Designer.

__ 4. View the changes to the BOM and to the `05_dynamic_domain.brl` rule.

   __ a. In the Synchronize pane, expand **trucksAndDrivers-bom - main > bom**.

   __ b. Double-click **model.bom** to open the compare view.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. Notice that `Training` in the Remote File (Decision Center) replaces `Administrative` in the Local File (Rule Designer).

__ d. Expand **trucksAndDrivers-rules - main > rules/domains**.

__ e. Double-click **`05_dynamic_domain.brl`** to open it in the Text Compare view, and notice that `Training` replaces `Administrative`.

__ 5. Update the Rule Designer projects with the latest changes from Decision Center.

__ a. In the Team Synchronize perspective, right-click **trucksAndDrivers-bom - main** and click **Update**.

__ b. Repeat this step to update **trucksAndDrivers-rules - main**.

After you update the Decision Center projects, the Synchronize pane no longer lists any changes.

__ c. Close the Text Compare window.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 6. Return to the Rule perspective, and view the rule `05_dynamic_domain` in the rule editor.

__ a. In the Rule Explorer, expand **trucksAndDrivers-rules > rules > domains**.

__ b. Double-click `05_dynamic_domain` to open it in the rule editor.

__ 7. Verify that the value `Training` is in the rule.

```
Content
1 definitions
2     set 'the driver' to a driver;
3     set 'a vacation request' to a vacation request in the vacation requests
4 if
5     the type of 'a vacation request' is one of { Training , Other Vacation
6 then
7     print "The driver " + the name of 'the driver' + " requested vacation fo
8
```

__ 8. Open the Problems view, and note the following warning message:

`Member domains.VacationRequestType.Administrative is deprecated.`

Rule Project Map | Rule Execution Server ... | Problems | Tasks | BOM Update | DVS Project Va

0 errors, 8 warnings, 0 others

Description	Resource
Warnings (8 items)	
[BOM] GBRMO0011W: Member domains.VacationRequestType.Administrative is deprecated.	model.bom
[BOM] GBRMO0011W: Member domains.VacationRequestType.Compensatory is deprecated.	model.bom

The `Administrative` domain value was deprecated when you updated the domain to include the `Training` value.

__ 9. Disconnect Rule Designer and Decision Center.

__ a. Right-click the **`trucksAndDrivers-test`** main rule project, click **Decision Center > Disconnect**.

__ b. In the Disconnect from Decision Center window, select **Keep the Decision Center entries files**, and click **Yes**.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

This exercise looked at how to define and use a dynamic domain, how to update dynamic domains in Decision Center, and how to synchronize them between Rule Designer and Decision Center. You also saw the effects of changing dynamic domain values in both Decision Center and Rule Designer.

(Optional) To see the solution to this exercise, switch to a new workspace and import the project **Ex 10: Dynamic domains > 02-answer**.

Information

As you learned, the values of the dynamic domains are shared between business users and developers, and can be updated independently. To ensure consistency across projects, you must establish governance guidelines:

- Clarify who can make these types of updates, and whether to enforce restrictions on when and how updates are made.
- Make sure that all roles that are involved in the business rule solution are aware of these changes, and update their working environment to reflect the changes.

Important

If you want to redo this exercise, you must delete the decision service that you published to Decision Center.

__ 1. Open the Decision Center Enterprise console by clicking **Start**, and then in the **Start** menu, clicking the **Decision Center Enterprise console** shortcut.

__ 2. Sign in with `rtsAdmin` as the user name and password.

__ 3. Delete the `trucksAndDrivers-tests` decision service.

   __ a. On the **Home** tab, select **Work on a decision service**.

   __ b. From the **Decision service in use** menu, select **`trucksAndDrivers-tests`**.

   __ c. On the **Configure** tab, under Administration, click **Erase Current Decision Service**.

   __ d. Click **Yes** when prompted to confirm the project deletion.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 12.Working with queries

Estimated time

00:45

Overview

This exercise teaches you how to define queries and rule extractors on rule projects. You also learn how to synchronize queries between Rule Designer and Decision Center.

Objectives

After completing this exercise, you should be able to:

- Search for rule artifacts and find rules according to their dependencies
- Define and run queries and apply actions on query results
- Synchronize queries between Rule Designer and Decision Center

Introduction

In many cases, you must analyze your ruleset to find dependencies between your rules. In this exercise, you search your rule project for rule artifacts that have specific properties, or rules that can enable or disable some other rule artifacts. You also learn how to create and run queries on rule projects, and to synchronize queries between Rule Designer and Decision Center.

The exercise involves these tasks:

- Section 1, "Searching for rule artifacts"
- Section 2, "Querying rule projects"
- Section 3, "Working with queries in Decision Center"

Requirements

This exercise uses the following files, which are installed in the `<InstallDir>\studio\training` directory:

- Start project: `Dev 11 - Queries\01-start`
- Solution project: `Dev 11 - Queries\02-answer`
  - You use the solution project in "Exercise review and wrap-up" on page 12-18.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Searching for rule artifacts

In this part of the exercise, you search for rule artifacts within a rule project. You also search for which rules are enabled or disabled by others, or which ruleflows might select them.

1.1. Setting up your environment for this exercise

__ 1. In Rule Designer, switch to a new workspace:

__ a. From the **File** menu, click **Switch Workspace > Other**.

__ b. In the Workspace Launcher window, enter the path:

`<LabfilesDir>\workspaces\queries`

__ c. Click **Launch**.

__ 2. Close the Welcome view.

__ 3. Use the Samples Console to import the exercise start project.

__ a. Click the **Open Perspective** icon.

__ b. In the Open Perspective window, click **Samples Console** and click **Open**.

__ c. In the **Rule Designer** section, expand **Training > Ex 11: Queries > 01-start**, and click **Import projects**.

__ d. When the workspace finishes building, close the Help view.

1.2. Searching for specific criteria across the rules

__ 1. From the **Search** menu, click **Search** to open the Search window.

__ 2. In the Search window, click the **Rule Search** tab. Take several minutes and figure out what the different elements on this tab are used for.

__ 3. Search all rules in the workspace that contain the `corporate score` string in their names or bodies.

__ a. In the **Search string** field, type: `corporate score`

__ b. Make sure that **Rule** is selected in the **Search for** section.

__ c. Make sure that **Body** is selected in the **Limit to** section.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Make sure that **Workspace** is selected in the **Scope** section.

__ e. Click **Search**.

All the results are listed in the Search view.

__ f. In the Search view results, double-click **initialCorporateScore**.

The `initialCorporateScore` action rule opens. The `corporate score` string that you searched for is highlighted in the rule artifact.

__ 4. Close the `initialCorporateScore` rule.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. From the **Search > Search** menu, do a new search for the `insur*` string in all packages in the workspace.

The `insurance` package is returned as a match.

__ 6. Open the Search window again and take some time to discover the purpose of the **XOM Search** tab and the **BOM Search** tab.

1.3. Searching for dependencies between rules

In this section, you search for dependencies between rules by using the **Find Rule Dependencies** menu.

__ 1. In the `loan-rules` decision service, expand **rules > insurance**.

__ 2. Right-click the **`insurance`** decision table, and click **Find Rule Dependencies > Rules which may enable or disable this rule**.

The Search view lists the `grade` decision table.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Questions

Why can the `grade` decision table enable or disable the `insurance` decision table that you selected?

Answer

The actions that are defined in the `grade` decision table set the grade of `'the loan report'`, which in turn determines which of the rows of the `insurance` decision table can be executed.

__ 3. Expand the `eligibility` rule package, right-click the **`grade`** decision table, and click **Find Rule Dependencies > Rules which may be enabled or disabled by this rule**.

The search results show the `insurance` decision table, which you expected, along with the `approval` action rule.

Questions

Can you figure out why the `grade` decision table that you selected might enable or disable the `insurance` decision table and the `approval` action rule?

Answer

The `grade` decision table might enable or disable the `insurance` decision table because it sets the `'the loan report'` grade, which is later used in the conditions of the `insurance` decision table.

The same mechanism applies to the `approval` action rule, where the condition is also based on this grade:

```
the loan grade in 'the loan report' is one of { "A" , "B" , "C" }
```

__ 4. Expand the `computation` rule package, right-click the **`initialCorporateScore`** action rule in the rule project, and click **Find Rule Dependencies > Ruleflows which may select this rule**.

The Search view lists the `loanvalidation` ruleflow and the `computation` rule task.

Questions

Can you figure out why the `loanvalidation` ruleflow and its `computation` rule task are the answer?

Answer

The Search view shows the `loanvalidation` ruleflow because this ruleflow selects all the rule artifacts in the rule project.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The ruleflow includes rule tasks that correspond directly to all the rule packages that are defined in the project, and can thus select all the rule artifacts of this rule project. If the path through the `loanvalidation` ruleflow includes the `computation` rule task, then the `initialCorporateScore` rule is executed.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Querying rule projects

In this part of the exercise, you create queries on rule projects to search for rules. You create different queries by using different criteria and run an action on the rules that the query returns.

2.1. Searching for rules that may become applicable

In this section, you search for rules that might become applicable if another rule is executed.

__ 1. Create a query in the `loan-rules` decision service.

__ a. In the Rule Explorer, right-click the **`queries`** folder and click **New > Query**.

__ b. In the **Name** field, type: `may become applicable`

__ c. Click **Finish**.

__ 2. In the new query, define the ***such that*** statement.

__ a. Click **enter a condition** and double-click **each business rule** to select it from the vocabulary list.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Complete the statement to match this content:

```
Find all business rules
  such that each business rule may become applicable
when [ 'a borrower' has filed a bankruptcy ]
```

__ c. Save the query.

Hint

To see the list of choices, press Ctrl+Space at the location in the rule where you want to add text, or type directly in the Query editor.

__ 3. In the **Run** section, click **Run query** to run the query on the rules.

The Search view indicates all rows of the `bankruptcyScore` decision table.

Questions

Do you know why you get this result?

Answer

The Search view indicates all rows of the `bankruptcyScore` decision table. All rows in this decision table have conditions that are based on the age of the latest bankruptcy of `the borrower`. The conditions are met only if `the borrower` filed a bankruptcy (which is the query condition).

Why is the `neverBankruptcy` rule not listed? This rule also relies on the bankruptcy status of `the borrower` for its conditions. However, if `the borrower` filed a bankruptcy, the `neverBankruptcy` action rule is not applicable because its conditions start with: `it is not true that`

No other rule artifact in the rule project relies on the bankruptcy status of `the borrower`.

__ 4. Close the query.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

2.2. Searching for rules that may lead to a state

In this section, you search for rules that can lead to a specific state of the application objects.

__ 1. Create a query that is called `may lead to a state` with the following code:

```
Find all business rules
   such that each business rule may lead to a state
      where [ 'a report' is insurance required ]
```

__ 2. Save the query and click **Run query**.

The Search view lists the `defaultInsurance` action rule and some of the rows in the `insurance` decision table.

Questions

Can you figure out why you obtain this result?

Answer

The Search view includes the `defaultInsurance` action rule because this action rule sets the `insuranceRequired` member of `'the loan report'` to `true`.

The `insurance` decision table defines rules that set the `insuranceRequired` member of `'the loan report'` to either `true` or `false`. All rows that are indicated in the Search view set this member to `true`.

The other rows (1 and 5) set this member to `false` and are not indicated in the Search view.

__ 3. Verify that the results in the Search view are accurate and that all rows of the `insurance` decision table, except 1 and 5, set the `insuranceRequired` member of `'the loan report'` to `true`.

__ a. In the Search view, right-click the `insurance` decision table and click **Show In > Rule Explorer**. Notice that the `insurance` decision table is highlighted in the Rule Explorer.

__ b. In the Rule Explorer, double-click the `insurance` decision table, which is highlighted, to open it in the decision table editor.

__ c. Read the values that are given in the `Insurance required` column and verify that all rows contain `true` except for rows 1 and 5, which contain `false`.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Hover the mouse over a row (for example, row 3), and read the complete action rule that corresponds to this row of the `insurance` decision table.

__ 4. Close the decision table and the query.

2.3. Searching for rules that use a BOM member

In this section, you search for rules that use a specific BOM member.

__ 1. Create a query that is called: `using BOM member`

The query should state:

```
Find all business rules
   such that each business rule is using the BOM Member
   "training_loan.Loan.amount"
```

__ 2. Save the query and run it.

The Search view indicates the `checkAmount` action rule, the `insurance` decision table, and the `repayment` action rule.

Questions

Can you figure out why you obtain this result?

Answer

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The Search view indicates the `checkAmount` action rule because the conditions of this action rule are based on the value of `the amount of 'the loan'`, that is, on the `training_loan.Loan.amount` BOM member.

The Search view also indicates the `insurance` decision table because the conditions for all the rows in this decision table are based on the same value.

Finally, the Search view indicates the `repayment` rule because the actions of this rule use the same value to compute the monthly repayment.

No other rule artifact in the rule project relies on the value of `the amount of 'the loan'`.

2.4. Applying actions with queries

In this section, you run a query and apply actions to the results. First, you search for rules that have a high priority, and modify their priority as an action that is associated with the query.

__ 1. Create a query that is called: `the priority of the rules`

The query should state:

```
Find all business rules
   such that the priority of each business rule is "1000000"
Do set the priority of each business rule to "-1000000"
```

Important

The query action is introduced with the `Do` keyword and modifies the priority of the business rules that match the query criteria.

Before you run a query that potentially modifies your rule project, you first want to determine whether your query is correct, and verify which rule artifacts might be affected. For this reason, the **Run query actions** task is separate from the **Run query** task, which provides an opportunity to review the query results before applying actions.

__ 2. Save your work.

__ 3. Select the **Run query** option of the query “`the priority of the rules`”.

You have two results:

- The `grade` decision table
- The `initialCorporateScore` action rule

Questions

Are these results correct?

Answer

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In the `loan-rules` project, only the `grade` decision table and the `initialCorporateScore` action rule have their **priority** property that is defined with the value `1000000`.

__ 4. Now, apply the actions to the found rule artifacts by clicking **Run query actions** in the **Run** section.

__ 5. Validate your results.

__ a. In Rule Explorer, expand **loan-rules > rules > computation**.

__ b. Open the `initialCorporateScore` rule.

__ c. In the Properties view (which is in the lower-right pane), verify that the **priority** property of the `initialCorporateScore` action rule is now equal to `-1000000`.

__ d. Do the same verification for the `eligibility.grade` decision table.

__ e. Close the rule and decision table.

__ 6. Run the `priority of the rules` query again.

No business rules match.

Questions

Why are there no matches?

Answer

The action of this query replaces the priority of `1000000` with a priority of `-1000000`. After this action is applied, the two rule artifacts found initially now have a priority of `-1000000`, and are thus not found if you run the query again.

__ 7. Close the query and the rules.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Working with queries in Decision Center

In this section, you work with queries in Decision Center. You also learn how to synchronize queries between Rule Designer and Decision Center.

Publishing and viewing the decision service in Decision Center

__ 1. Publish the `loan-rules` decision service to Decision Center.

__ a. In the Rule Explorer, right-click **`loan-rules`** and go to **Decision Center > Connect**.

__ b. In the Decision Center Configuration window, enter the following connection details and click **Connect**:

- **URL**: `http://localhost:9090/teamserver`
- **User name**: `rtsAdmin`
- **Password**: `rtsAdmin`

__ c. After the connection is established, click **Next**.

__ d. In the Synchronization Settings window, click **Finish**.

__ e. In the Synchronize Complete window, click **OK** to close it.

__ f. In the Confirm Open Perspective window, click **No**. You do not need to switch to the Team Synchronizing perspective.

__ 2. Log in to Decision Center as `rtsUser1`.

__ a. From the Windows **Start** menu, click the **Decision Center Business console** shortcut.

__ b. Enter `rtsUser1` in the **Username** and **Password** fields, and click **Log in**.

__ 3. If you are not on the Library page, click the **Library** tab to open the list of Decision Services.

__ 4. Open the `main` branch of the `loan-rules` decision service that you published.

__ a. In the Decision Services list, locate `loan-rules`.

__ b. Click the white space of the `loan-rules` box to expand it.

__ c. Click the **main** link.

Alternatively, you can click the **loan-rules** link, then click **Branches > main**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Working with queries in Decision Center

You can work with queries in the Business console by using the **Queries** tab.

__ 1. Click the **Queries** tab.

__ 2. In the Queries List, you see the queries that you defined earlier in this exercise:

- `may become applicable`
- `may lead to a state`
- `the priority of the rules`
- `using BOM member`

__ 3. Click each query to view its details in the main pane.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

In the main pane, you can run the query, and you can also edit the query and save your changes.

Information

The `using BOM member` query has the following error message:

```
This query contains constructions that are invalid for this environment.
```

Severity	Line	Message
	1	This query contains constructions that are invalid for this environment.

This error appears in Decision Center because the `using BOM member` construct is specific to the Rule Designer module. This query cannot be run in Decision Center.

If a query must be run in both modules, make sure that you use query constructs that can be used in both Rule Designer and Decision Center.

__ 4. Create and run a query to check the status of the rules.

__ a. At the bottom of the main query pane, click **New Query**.

__ b. In the name field, enter: `the status of a rule`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. In the Query Expression editor, build the following query:

```
Find all business rules
  such that the status of each business rule is new
```

__ d. Click **Save**, and then click **Run**.

The Query Results page opens, and lists the results of your new query. All of the rule artifacts in the `loan-rules` decision service are listed because their status is new.

__ 5. Leave the Business console window open. You return to this console later.

Synchronizing queries in Rule Designer

In this section, you synchronize the decision service in Rule Designer with Decision Center so that it can retrieve the query that you created.

__ 1. Switch back to Rule Designer.

__ 2. In the Rule Explorer, right-click **`loan-rules`** and go to **Decision Center > Synchronize with Decision Center**.

__ 3. In the Decision Center Configuration window, click **Finish**.

__ 4. In the Confirm Open Perspective window, click **Yes** to open the Team Synchronizing perspective.

__ 5. In the Synchronize pane, expand **loan-rules - main > queries** to see `the status of a rule` query.

__ 6. Right-click **`the status of a rule`** and click **Update**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 7. Return to the Rule perspective, and in the Rule Explorer, go to **loan-rules > queries**. The new query `the status of a rule` is now listed.

Deleting the loan-rules decision service from Decision Center

__ 1. Return to the Decision Center Business console.

__ 2. Sign out of Business console, and sign in again with `rtsAdmin` as the user name and password.

__ 3. In the upper left of the window, click the **loan-rules** breadcrumb, then click the **All Decision Services** breadcrumb.

__ 4. Click the arrow beside **loan-rules** and click **Delete**.

__ 5. Click **Delete**.

__ 6. Sign out of the Business console and close the browser window.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

The first part of the exercise looked at how to search for rule artifacts within a rule project and how to find which rules are enabled or disabled by others, or which ruleflows might select them.

The second part of the exercise looked at how to create and run queries on rule projects.

The third part of the exercise looked at the Queries feature in Decision Center, and also demonstrated how to synchronize queries between Rule Designer and Decision Center.

(Optional) To see the solution to this exercise, switch to a new workspace and import the project **Ex 11: Queries > 02-answer**.

Attention

In the answer project, the priorities for the `grade` decision table and for the `initialCorporateScore` action rule are still `1000000`. This value was left intentionally for you to be able to see the results of running the `priority of the rules` query actions on the answer project.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 13.Executing rules locally

Estimated time

00:30

Overview

This exercise teaches you how to run rule projects locally to ensure the correctness of rulesets.

Objectives

After completing this exercise, you should be able to:

- Create launch configurations to run rulesets locally

Introduction

In this exercise, you create a launch configuration to execute the rule artifacts locally in Rule Designer, without writing code.

The exercise includes these tasks:

- Section 1, "Executing rules locally by using a launch configuration"

Requirements

This exercise uses the following file, which is installed in the `<InstallDir>\studio\training` directory:

- Start project: `Dev 12 - Executing rules locally\01-start`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Executing rules locally by using a launch configuration

In this part of the exercise, you create a launch configuration. You then use it to execute the rule artifacts in your rule project locally in Rule Designer.

1.1. Setting up your environment for this exercise

__ 1. In Rule Designer, switch to a new workspace:

__ a. From the **File** menu, click **Switch Workspace > Other**.

__ b. In the Workspace Launcher window, enter the path:

`<LabfilesDir>\workspaces\execute_locally`

__ c. Click **Launch**.

__ 2. Close the Welcome view.

__ 3. Use the Samples Console to import the exercise start project.

__ a. Click the **Open Perspective** icon.

__ b. In the Open Perspective window, click **Samples Console** and click **Open**.

__ c. In the **Rule Designer** section, expand **Training > Ex 12: Executing rules locally > 01-start**, and click **Import projects**.

__ d. When the workspace finishes building, close the Help view.

Hint

In this exercise, you write some code. To help you, you can copy and paste the code snippets from the `<LabfilesDir>\code\execute_rules_locally.txt` file into Rule Designer.

1.2. Creating a launch configuration

In this section, you create a launch configuration for the `loan-rules` project.

__ 1. From the **Run** menu, click **Run Configurations**.

The Run Configurations window opens.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. In the Run Configurations wizard, double-click **Decision Operation**.

__ 3. In the **Name** field, enter: `loan run`

__ 4. On the **Decision Operation** tab, configure the following properties.

__ a. To set the **Decision Operation** field, click **Browse**, select **loan-rulesOperation**, and click **OK**.

__ b. Make sure that the **Stop at first rule statement** check box is *not* selected.

This check box is used only when you debug rules, which you do later.

__ c. In the **Working directory** section, keep **Default** selected.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. Click the **Parameters & Arguments** tab and configure the parameters that are required to execute your rule project.

__ 6. Define the **`borrower`** parameter.

__ a. Select the **`borrower`** parameter in the **Name** column, and click **Edit Value**.

__ b. Select the **Function body** option.

__ c. Overwrite the initial value with the following code to overwrite and click **OK**:

```
training_loan.Borrower borrower = new
training_loan.Borrower("John","Doe",training_loan.DateUtil.makeDate(1968,
java.util.Calendar.MAY,12),"123456789");
borrower.zipCode = "91320";
borrower.creditScore = 600;
borrower.yearlyIncome = 100000;
return borrower;
```

Hint

You can find all code snippets for this exercise in the *<LabfilesDir>*`\code\execute_rules_locally.txt` file.

__ 7. Define the **`loan`** parameter.

__ a. Select the **`loan`** parameter and click **Edit Value**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Select **Function body** and overwrite the initial value with the following code and click **OK**:

```
training_loan.Loan loan = new
training_loan.Loan(training_loan.DateUtil.makeDate(2016,
java.util.Calendar.JANUARY,15),72,100000,0.7d);
return loan;
```

__ 8. Click **Apply** to save the changes.

__ 9. Click **Run**.

Rule Designer executes the rules in your rule project locally by using the parameters that are defined in the `loan run` launch configuration. The Console view shows the following execution traces:

```
Borrower( firstName: John lastName: Doe born: May 12, 1968 SSN: 123-45-6789
zip code: 91320 income: 100000 credit score: 600)
Loan( amount: 100000 starting: Jun 1, 2016 duration: 72 month LTV: 70% rate:
5.7% monthly repayment: 1,643.16)
Report( validData: true approved: true score: 820 grade: B insuranceRequired:
true insuranceRate: 2% message: Low risk loan
Congratulations! Your loan has been approved
)
```

1.3. Debugging with a launch configuration

In the previous section, you created the `loan run.launch` configuration to execute your ruleset locally without the Rule Debugger. In this section, you modify this launch configuration to use the Rule Debugger.

__ 1. From the **Run** menu, click **Debug Configurations**.

The Debug Configurations window opens.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Questions

Do you see any differences between the Debug Configurations window and the Run Configurations window that you worked in earlier?

Answer

The Debug Configurations window and Run Configurations window share content except that **Debug** replaces **Run**.

__ 2. On the **Decision Operation** tab, select **Stop at first rule statement**.

The **Stop at first rule statement** option is used when you run the debug configuration. If this check box is selected, the Rule Debugger stops before any business rule is executed. You can then complete any steps that are required for your debugging session (such as setting breakpoints or modifying parameters).

__ 3. Click **Apply**.

__ 4. Click **Close** to exit.

Note

On the **Overview** tab for the decision operation (**loan-rules > deployment > loan-rulesOperation**), you can click the **Run** or **Debug** icons to rerun the configuration.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

This exercise looked at how to execute rules locally by using a launch configuration in Rule Designer.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 14.Debugging a ruleset

Estimated time

00:30

Overview

This exercise teaches you how to debug a ruleset in Rule Designer.

Objectives

After completing this exercise, you should be able to:

- Use automatic exception handling
- Set breakpoints in rules, decision tables, and ruleflows
- Run a debugging session

Introduction

In many cases, executing rules locally in Rule Designer is not enough for you to find out why the ruleset does not create the expected decision. In this exercise, you debug a ruleset with the Rule Debugger.

The exercise includes these tasks:

- Section 1, "Running the debug launch configuration"
- Section 2, "Setting breakpoints"
- Section 3, "Debugging a rule"
- Section 4, "Debugging a decision table"
- Section 5, "Debugging the ruleflow"

The steps that are described here are extracted from the *Debugging a decision service* tutorial, which is distributed with the product documentation.

Requirements

Note

For IBM ODM on Cloud users

This exercise requires that you use the on-premises version of ODM. For ODM on Cloud, you can use the projects and instructions that are provided at this URL:

`https://github.com/ODMDev/odm-on-cloud-debug`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Running the debug launch configuration

In this section, you use a launch configuration to debug the ruleset execution.

1.1. Setting up your environment for this exercise

__ 1. In Rule Designer, switch to a new workspace:

__ a. From the **File** menu, click **Switch Workspace > Other**.

__ b. In the Workspace Launcher window, enter the path:

`<LabfilesDir>\workspaces\debug`

__ c. Click **Launch**.

__ 2. Close the Welcome view.

__ 3. Use the Samples Console to import the exercise start project.

__ a. Click the **Open Perspective** icon.

__ b. In the Open Perspective window, click **Samples Console** and click **Open**.

__ c. In the **Rule Designer** section, expand **Tutorials > Debugging a decision service > start**, and click **Import projects**.

__ d. After the workspace finishes building, close the Help view.

The Rule perspective opens with these projects:

- `Loan Validation Base`
- `Loan Validation Check`
- `Loan Validation Determination`
- `Loan Validation Scoring`
- `Loan Validation Service`
- `loan-validation-xom`

1.2. Running the launch configuration

__ 1. From the menu, click **Run > Run Configurations**.

__ 2. Expand **Decision Operation** and select the **`run loan validation`** launch configuration.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. Click **Run**.

The Console view shows an exception error.

To handle this type of error, you implement automatic exception handling.

1.3. Applying automatic exception handling

__ 1. In the Rule Explorer, right-click **Loan Validation Service**, and click **Properties**,

__ 2. In the left column of the Properties dialog, select **Rule Engine**, and select **Automatic exception handling in conditions**.

__ 3. Click **Apply and Close**:

__ 4. Rerun the `run loan validation` launch configuration by clicking the run arrow on the toolbar.

The Console view no longer shows an exception error. The decision engine automatically handles the exception.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

1.4. Adding exception handing logging

Next, to determine where the exception takes place, you add more logging information to the results.

__ 1. In Rule Explorer, expand **Loan Validation Service** and double-click **logging.properties** to open it.

__ 2. Notice the line in the file that sets the Console as a logging handler:

```
handlers = java.util.logging.ConsoleHandler
```

__ 3. Note also that the log level is set to **`FINE`**.

__ 4. Open **Run > Run Configurations**, and select the **run loan validation** configuration.

__ 5. Open the **Parameters and Arguments** tab, and in the **VM arguments** section, add the following argument:

```
-Djava.util.logging.config.file="${workspace_loc}/Loan Validation
Service/logging.properties"
```

__ 6. Click **Apply**, and click **Run**.

The Console shows the results, which include a log message for the automatic exception handling. The results show that the exception takes place in a condition in the `checkZipcode` rule.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Note

The Console shows the `AEH_LOG` message twice. Because the `checkZipcode` rule contains a ***then*** action and an ***else*** action, its condition part is evaluated twice, producing two `AEH_LOG` messages.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Setting breakpoints

2.1. Searching for the problem source

Based on the information in the log, you know that the exception takes place in a condition in the `checkZipcode` rule. The results also point to the `validation` task in the `loanvalidation` ruleflow:

`ruleflow.loanvalidation$003validation`

You can set a breakpoint on the ruleflow to search for the source of the error.

__ 1. On the **Search** menu, click **Search**.

__ 2. Open the **Rule Search** tab.

__ 3. In the **Search string** field, enter: `loanvalidation`

__ 4. Select **Ruleflow**, and then click **Search**.

The `loanvalidation` ruleflow is displayed in the Search view.

__ 5. Double-click the `loanvalidation` ruleflow in the Search view to open it.

__ 6. Click the validation task to select it.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 7. Right-click the task and click **Toggle Breakpoint**.

2.2. Preparing the debug session

__ 1. Double-click the `validation` task in the `loanvalidation` ruleflow.

__ 2. In the **Properties** tab, click **Rule Selection** and expand the `validation` package.

__ 3. Expand `borrower`, and double-click the `checkZipcode` action rule to open it in the rule editor.

You see that the condition in the rule uses the length of the zip code. You want to know the value of the zip code, so you add trace information to print the value.

__ 4. Go back to the `loanvalidation` ruleflow in the ruleflow editor, and double-click the `validation` task.

__ 5. In the Properties view, select **Initial Action**.

__ 6. Keep **Use BAL for action** selected, and add the following code to display the borrower zip code:

```
print "The zip code of the borrower is " + the zip code of the address of
'the borrower' ;
```

__ 7. Save your work.

2.3. Running the debugger on a ruleflow

__ 1. On the Run menu, click **Debug Configurations**.

__ 2. Click the `run loan validation` configuration to open it, and click **Debug**.

__ 3. If the Save and Launch dialog opens, click **OK** to save your changes before running the debugger.

__ 4. When the Confirm Perspective Switch dialog box opens, click **Yes** to open the debug perspective.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The debugger stops at the beginning of the `validation` rule task, at the breakpoint that you set.

__ 5. Expand borrower in the **Variables** tab.

__ 6. Notice the null values for `latestBankruptcy`, `spouse`, and `zipCode`.

__ 7. On the toolbar, click the **Resume** icon.

__ 8. Notice the results in the Console view.

The first line is the trace that you added to the initial action of the `validation` task:

```
The zip code of the borrower is null
```

You see that the automatic exception handling catches the null pointer exception because of the null zip code in the `checkZipcode` rule. The condition part has an unknown status, so the rule is ignored. You confirm with the rule authors about the null zip code for the borrower in the configuration. The rule authors confirm that the zip code rule can be ignored.

Now that you confirmed that the automatic exception handling correctly addresses the problem, you can remove the trace for the logging information that comes from the automatic exception handling.

__ 9. Switch back to the Rule perspective.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 10. On the **Run** menu, click **Run Configurations**, and open the `run loan validation` configuration.

__ 11. On the **Parameters and Arguments** tab, delete the VM arguments.

__ 12. Click **Apply**, and then **Run**.

The automatic exception handling trace information is no longer shown in the Console view. However, the loan is still not approved, so more errors need to be fixed.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Debugging a rule

In this section, you set breakpoints in a rule and run a debug session. When you find an error, you make changes to correct it.

When you run the decision service, it does not generate the correct monthly repayment rate. You want to find the source of this value to determine how to fix it. First, you create a query to find the rule that uses the variable `monthly repayment`. Then, you add a breakpoint to track the rule.

3.1. Setting a breakpoint in a rule

__ 1. In the Rule Explorer, expand **Loan Validation Service**.

__ 2. Right-click the **queries** folder, and click **New > Query**.

__ 3. In the **Name** field of the New Query dialog, enter `Monthly`, and click **Finish**.

__ 4. Set the query to state:

```
Find all business rules
  such that the definition of each business rule contains "monthly"
```

__ 5. Save your work (Ctrl+S).

__ 6. Click **Run query**.

The query returns the `repayment` rule in the Search view.

__ 7. Double-click the `repayment` rule in the Search view to open it in the rule editor.

You see that the first action, line 7, sets the value of the monthly repayment variable.

__ 8. Click the **ARL** tab to see how the monthly repayment is calculated in the rule.

__ 9. Note that in the ***then*** statement, the rule calls the `loan.LoanUtil.getMonthlyRepayment` method to calculate the monthly repayment rate. You must step into this method.

__ 10. Reopen the **Intellirule** tab, right-click line 7, and click **Toggle Breakpoint**.

A breakpoint is displayed next to the line.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

3.2. Debugging the action rule

Next, you run a debugging session. When it stops at breakpoints, you look for reasons for the monthly repayment error.

__ 1. From the **Run** > **Debug Configurations** menu open the `run loan validation` configuration and click **Debug**.

__ 2. Click **Yes** if prompted to switch to the Debug perspective. Otherwise, you can switch manually.

The debugger stops at the `validation` task in the `loanvalidation` ruleflow.

__ 3. On the toolbar, click **Resume**.

The debugger stops at the “set the monthly repayment line” of the `repayment` rule.

__ 4. On the **Variables** tab, expand **report**. The `monthlyRepayment` variable shows `0.0`.

__ 5. Click the **Step Into** icon.

The debugger stops at `LoanUtil.java`. The Java code shows the following method at the breakpoint:

```
double i = yearlyRate / MonthsInYear;
```

The function computes a monthly repayment rate by dividing `yearlyRate` by `MonthsInYear`. In the **Variables** view, you see that the value of `MonthsInYear` is `0`. The error is that the function is dividing by 0, when it should divide by 12.

__ 6. Stop the debugging session by clicking the **Terminate** icon.

3.3. Fixing the error in the rule

You change the value of the `MonthsInYear` variable to `12`, and run the decision service to check the results.

__ 1. Switch to the Rule perspective.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. In the editing window for `LoanUtil.java`, change the value of the `MonthsInYear` variable to 12.

```
32 public class LoanUtil implements Serializable {
33     private static final long serialVersionUID = 7764736178720624835L;
34     private static final shor MonthsInYear= 12;
```

__ 3. Save your changes.

__ 4. Run the decision service by clicking the **Run** icon on the toolbar.

The Console now shows the correct monthly repayment rate of 570.83.

However, the decision service still does not approve the loan, so you have more work to do.

__ 5. Remove all the breakpoints.

__ a. On the **Window** menu, click **Show view > Other**.

__ b. Expand **Debug** and double-click **Breakpoints**, and then click **Open**.

__ c. Click the **Remove All Breakpoints** icon on the toolbar of the Breakpoints view, and click **Yes** to confirm removal.

__ 6. Save your work.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 4. Debugging a decision table

4.1. Finding the source of the rejection message

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

__ 2. Open the **Run > Run Configurations** menu item, and click the `run loan validation` configuration to open it.

__ 3. Click **Run**.

The results in the Console view show that the loan is low risk, but the decision service still does not approve the loan.

You decide to search for the rule that makes the rejection message by running a query for the message.

__ 4. In the Rule Explorer, expand **Loan Validation Service**, right-click the **queries** folder, and click **New > Query**.

__ 5. Create a query to find the rule that uses the following rejection message:

```
Your loan has not been approved
```

__ a. Set the query name to `Approval` and click **Finish**.

__ b. Set the query to:

```
Find all business rules
  such that the definition of each business rule contains "Your loan has
not been approved"
```

__ c. Save your work.

__ d. Click **Run query**.

The query returns the `approval` action rule.

__ 6. Double-click the `approval` rule in the Search view to open it in the rule editor.

The action depends on the grade that is given to the loan. If the grade does not equal A, B or C, the loan is rejected:

```
if
    'the grade' is one of { "A" , "B" , "C" }
then
    in 'the loan report', accept the loan with the message "Congratulations!
Your loan has been approved" ;
else
    in 'the loan report', refuse the loan with the message "We are sorry.
Your loan has not been approved" ;
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 7. Place breakpoints on both actions (lines 4 and 6) in the `approval` rule.

```
1 if
2     'the grade' is one of { "A" , "B" , "C" }
3 then
4     in 'the loan report', accept the loan with the message "Congratulations
5 else
6     in 'the loan report', refuse the loan with the message "We are sorry. Y
```

Now you search for the rule that computes the grade condition for the approval rule.

4.2. Setting breakpoints in a decision table

The approval action rule applies a decision that is based on the grade that is given to the loan. You decide to locate the rule that assigns grades to the loans. You create a query that finds a decision table, and then you add breakpoints to the decision table to observe it in a debugging session.

__ 1. Create another query on the **Loan Validation Service** project.

__ a. Right-click the **Loan Validation Service > queries** folder, and click **New > Query**.

__ b. Name the query `gradeSet`, and click **Finish**.

__ c. Set the query to:

```
Find all business rules
  such that each business rule may lead to a state where [ 'the grade' is
not one of { "A", "B", "C" } ]
```

__ d. Save your work.

__ e. Click **Run query**.

The query looks for a rule that can assign a grade that is not A, B or C, and therefore, causes the decision service to reject the loan.

The Search view shows the grade decision table, and lists the rows that do not contain grade A, B, or C.

__ 2. Double-click one of the `grade` decision table results in the Search view to open it in the decision table editor.

__ 3. Select the **Grade** column by clicking the column header cell, right-click the A cell in row 1, and click **Toggle Breakpoint**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Every Grade cell now has a breakpoint.

4.3. Debugging the decision table

__ 1. Switch to the Debug perspective.

__ 2. Click the **Run > Debug Configurations** menu item, open the `run loan validation` configuration, and click **Debug**.

The debugger stops at row 3, which shows grade D and the message `Low risk loan`.

__ 3. Click the **Resume** icon.

The debugger stops at the second breakpoint in the `approval` action rule. The loan is rejected because the grade is D. The error is coming from row 3 of the `grade` decision table.

__ 4. Click the **Terminate** icon to stop the debugging session.

You contact the business user, who tells you to change the grade in row 3 to B.

__ 5. Open the **grade** decision table in the Rule perspective, and change the grade in row 3 to B.

__ 6. Save your changes, and run the decision service normally by selecting `run loan validation` in the **Run Configurations**.

The results show that the loan is low risk, but the decision service still rejects the loan.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 5. Debugging the ruleflow

The decision service still rejects the loan. The rules run correctly now, so you decide to test the flow of decisions among the rules.

5.1. Debugging the ruleflow

Run a debugging session to determine the source of the error.

__ 1. Run the `run loan validation` configuration in the **Debug Configurations**.

The debugger stops at row 3 of the grade decision table.

__ 2. In the Debug perspective, open the **Agenda** tab.

The tab shows two rules that are in the same `eligibility` rule task. The first entry points to the rule that is formed by row 3 in the `grade` decision table, and the second entry points to the `approval` action rule.

In the `grade` decision table, you see that an arrow points to the grade cell in row 3. The grade is B, and the message for the row is `Low risk loan`.

__ 3. Click the **Resume** icon.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The debugger stops on the ***else*** line in the `approval` action rule, which rejects the loan.

Although the `grade` decision table set the `grade` value to B, the action rule still goes to the ***else*** condition. The approval action rule does not see the grade value set by the decision table. This disconnect between the rules indicates a ruleflow problem.

__ 4. Click the **Terminate** icon to stop the debugging session.

5.2. Modifying the ruleflow

__ 1. Switch to the Rule perspective and open the `loanvalidation` ruleflow.

__ 2. Select the `eligibility` rule task.

__ 3. In the **Properties** view, open the **Rule Selection** tab, and expand **eligibility**.

You see that the task contains both the `approval` action rule and the `grade` decision table.

__ 4. Open the **Rule Task** tab and note that the Fastpath algorithm is selected.

The Fastpath algorithm does not support inference. The agenda is computed once in the rule task, but it is not changed if variables are modified by other rules in the task. This means that the `approval` rule runs with an unset grade value, and the value is not changed by the `grade` decision table.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. Create a separate rule task for the `approval` action rule to resolve the error.

__ a. Click the **Create a rule task** icon and then click the ruleflow diagram next to the `insurance` task node to add the task node.

__ b. On the Rule Task tab in the Properties view, set the ID to: `approval`

__ c. On the **Rule Selection** tab, click **Edit**, double-click the **eligibility.approval** rule, and click **OK**.

__ 6. Change the transitions to flow between these nodes:

- `eligibility` to `approval`
- `approval` to `insurance`
- `approval` to end node

__ a. Click the transition line from the `eligibility` task to the end node, and reset it to run from the `approval` task to the end node.

__ b. Click the transition line from the `eligibility` task to the `insurance` task, and reset it to run from the `approval` task to the `insurance` task.

__ c. Create a transition line between the `eligibility` task and the `approval` task.

__ 7. Edit the `eligibility` task to remove the `approval` action rule from the **Rule Selection** tab.

__ a. Click the `eligibility` task on the ruleflow diagram, and in the Properties view, click the **Rule Selection** tab, and click **Edit**.

__ b. Double-click the **eligibility** package to remove it from the list of selected rules (right side of the Select Rules dialog).

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. In the left side of the Select Rules dialog, expand **eligibility**, and double-click each of the `eligibility` rules except the `approval` rule.

__ d. Click **OK**.

__ 8. Save your work.

__ 9. Run the decision service by using the `run loan validation` configuration in the Run Configurations.

Now, the grade value set in the `grade` decision table is set before the `approval` rule is evaluated in the `approval` task, and the loan is approved.

__ 10. Remove all the breakpoints.

__ a. Open the Breakpoints view.

__ b. Click the **Remove All Breakpoints** icon on the toolbar of the **Breakpoints** view, and click **Yes** to confirm removal.

__ 11. Save your changes.

__ 12. Run the decision service by using the `run loan validation` configuration in the Run Configurations.

The decision service runs correctly, and produces the expected report of an approved loan.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

This exercise looked at how to debug a ruleset with the Rule Debugger. You saw how to set breakpoints on an action rule, a decision table, and in a ruleflow, and you saw how to view variable values and rule instances in the agenda.

To see the solution projects for this exercise, switch to a new workspace and import the project **Rule Designer > Tutorials > Debugging a ruleset > answer > Import projects**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 15.Enabling rule validation

Estimated time

1:00

Overview

This exercise teaches you how to set up testing and simulation functionality for business users.

Objectives

After completing this exercise, you should be able to:

- Validate the BOM and generate scenario file templates
- Customize scenario file templates
- Validate remote testing conditions for business users in the Business console

Introduction

In this exercise, you prepare a decision service to make it ready for running tests and simulations. You also work with scenario file templates.

As developers, your role with testing is mainly to enable business users to run tests and simulations with minimal dependence on you. This exercise provides an overview of the tasks that are involved, including:

- Section 1, "Validating the rule project"
- Section 2, "Removing columns from the scenario file template"
- Section 3, "Enabling testing from Business console"
- Section 4, "Testing from Business console"

Requirements

This exercise uses files that are stored in the `<LabfilesDir>\code` directory. It also uses the following files, which are installed in the `<InstallDir>\studio\training` directory:

- Start project: `Dev 14 - Validate\01-start`
- Solution project: `Dev 14 - Validate\02-answer`

You can use the solution project as described in "Exercise review and wrap-up" on page 15-14.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Note

For IBM ODM on Cloud users

This exercise uses some features that are not supported in IBM ODM on Cloud.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Validating the rule project

To use the default Excel format for your scenarios, you must validate that the rule projects to be tested can correctly generate the appropriate columns in the Excel scenario file template.

You use the Testing and Simulation BOM Validation view when you validate your rule project. This view raises error and warning messages if the BOM or the input parameters of the rule project must be modified.

1.1. Setting up your environment for this exercise

__ 1. Make sure that the Sample Server is running.

__ 2. Open the Business console by clicking the **Decision Center Business console** shortcut on the Windows **Start** menu.

__ 3. Sign in with the `rtsAdmin` for the user name and password.

__ 4. Erase the `Loan Validation Service` decision service.

  __ a. On the **Library** page, click the arrow beside **Loan Validation Service**, and click **Delete**.

  __ b. When prompted to confirm, click **Delete**.

__ 5. Close the Business console.

__ 6. In Rule Designer, switch to a new workspace:

  __ a. From the **File** menu, click **Switch Workspace > Other**.

  __ b. In the Workspace Launcher window, enter the path:

  `<LabfilesDir>\workspaces\testing`

  __ c. Click **Launch**.

__ 7. Close the Welcome view.

__ 8. Use the Samples Console to import the exercise start project.

  __ a. Click the **Open Perspective** icon.

  __ b. In the Open Perspective window, click **Samples Console** and click **Open**.

  __ c. In the **Rule Designer** section, expand **Training > Ex 14: Enabling rule validation > 01-start**, and click **Import projects**.

__ 9. Close the Help view.

Hint

In this exercise, you must write some code. You can find the code snippets to create in the `<LabfilesDir>\code\enable_testing.txt` file, and you can copy and paste them in Rule Designer.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

1.2. Validating the rule project

__ 1. In Rule Explorer, make sure that the `Loan Service` project is selected and open the Rule Project Map view.

__ 2. On the Decision Service Map, in the **Define decision operation** part of the map, click **Go to operation map**, select **loan validation production.dop**, and click **OK**.

__ 3. In the **Design signature** part of the map, click **Check for testing**.

__ 4. When prompted by the Decision Operation Selection window, make sure that only **loan validation production** is selected and click **Finish**.

The Testing and Simulation BOM Validation window opens and reports a compliance error.

__ 5. Click **OK** to close the Testing and Simulation BOM Validation window.

__ 6. In the Testing and Simulation BOM Validation view, select the error for `loan.Borrower` to open and read the Solution Description.

An error is raised because no class constructor was defined or specified as the Decision Validation Services constructor.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

1.3. Choosing a constructor

BOM classes that are used to define ruleset input parameters must have at least one constructor. One of these constructors must be specified as the constructor for Decision Validation Services. The columns of the Microsoft Excel scenario file template then correspond to the argument of this BOM class constructor.

To choose a constructor:

__ 1. In Rule Explorer, expand **Loan BOM > bom > model > loan** and double-click **Borrower** to open the BOM editor.

The Class page opens for the `Borrower` class. Scroll through the Members list and notice that `Borrower` has two constructors.

When a class has multiple constructors, you must specify which one to use for validation.

__ 2. In the **Members** section of the Borrower class page, double-click the first constructor with four arguments, **Borrower(String, String, Date, String, int)**.

__ 3. On the Member page, select the **Testing and simulation constructor** check box.

__ 4. Save your work.

__ 5. Check the project again for compliance.

__ a. Right-click **`Loan Service`** and click **Testing and Simulation > Check Project**.

__ b. Make sure that only the **loan validation production** decision operation is selected and click **Finish**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Another compliance error is reported.

__ c. Click **OK** to close the error message window.

In the Testing and Simulation Project Validation view, you see a list of errors.

__ 6. Select an error to see the description.

You see that the constructor arguments use generic names that you must rename with meaningful names. The arguments (`arg1`, `arg2`, `arg3`, `arg4`, and `arg5`) correspond to the arguments of the selected constructor:

`Borrower (String, String, Date, String)`

__ 7. Double-click the first error message to open the Borrower page in the BOM editor.

Next, you edit the constructor argument names.

1.4. Editing column headings for the scenario file template

The constructor argument names become the column headings in the scenario file template. You must edit the argument names to match the corresponding BOM member names. For example, if `arg1` sets the value of the `firstName` attribute, then rename `arg1` as: `firstName`

To edit the arguments for the column headings:

__ 1. In the **Arguments** section of the `Borrower` class page, double-click **`arg1`**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. In the **Name** field, change `arg1` to `firstName` and click **OK**.

Important

The name must match the corresponding attribute name in the `Borrower` class.

When the template is generated, the column headings display the verbalized name of the corresponding attribute or argument. For example, the generated template column for the `firstName` attribute uses the heading: `first name`

__ 3. Edit the names of the remaining arguments to these values:

- `lastName`
- `birthDate`
- `SSNCode`

__ 4. Save your work.

__ 5. Check the project compliance again.

__ a. Right-click **`Loan Service`** and click **Testing and Simulation Services > Check Project**.

__ b. Make sure that only the **loan validation production** decision operation is selected and click **Finish**.

You see a message that the project is now compliant.

__ c. Click **OK** to close the Testing and Simulation BOM Validation window.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Removing columns from the scenario file template

Because of the relationship between the BOM and the scenario file, customizing the scenario file template requires modifying the BOM.

When preparing the BOM to generate a template, you can exclude BOM members that are not useful as input to tests or simulations. For example, attributes that are based on calculated values should not be included as input columns because they do not have input values.

To remove a column:

__ 1. In the Rule Explorer, expand **Loan BOM > bom > model > loan**, and double-click **Borrower** to open it in the BOM editor.

__ 2. In the **Members** section of the Class page, double-click **`zipCode`** to open it on the Member page.

On the Member page for the `zipCode` attribute, you can see that **Ignore for testing and simulation** is not selected, meaning that this attribute is included in the scenario file template.

__ 3. Select **Ignore for testing and simulation**.

__ 4. Save your work.

No input columns are created for this attribute on the **Scenarios** sheet of the template. However, when you generate the template, you can still choose to include these attributes in the **Expected Results** sheet to test that their values are calculated correctly as a result of rule execution.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Enabling testing from Business console

In this section, you publish the updated project to Decision Center so that business users can generate scenario files to run tests and simulations from Business console.

__ 1. Publish the decision service to Decision Center.

__ a. In Rule Explorer, right-click **Loan Services** and click **Decision Center > Connect**.

__ b. Enter the following values in the Decision Center Configuration window fields.

- **URL**: `http://localhost:9090/teamserver`
- **User name**: `rtsAdmin`
- **Password**: `rtsAdmin`
- **Data source**: (Leave the field empty)

__ c. Click **Connect**.

__ d. Click **Next**.

__ e. On the Synchronization Settings page, click **Next**.

__ f. On the Decision Service Dependent Projects page, keep **Loan BOM** selected and click **Finish**.

__ 2. When prompted to switch to the Synchronizing view, click **No** (because no conflicts exist).

__ 3. Click **OK** to close the Synchronize Complete window.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 4. Testing from Business console

__ 1. Open the Business console by double-clicking the **Decision Center Business console** shortcut on the **Start** menu.

__ 2. Sign in with the `rtsAdmin` for the user name and password.

__ 3. On the **Library** tab, click **Loan Service**.

__ 4. On the **Branches** tab, click **main**.

__ 5. Click the **Tests** tab and make sure that you are on the **Test Suites** page.

__ 6. Generate a scenario file that is called `My Scenario File`.

__ a. Click the **Generate Scenario File** icon.

__ b. In the “Select an operation” window, select **loan validation production**, and click **OK**.

__ c. In the **Filename** field, type: `Loan Scenario`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. In the **Tests** section, expand **the loan report** and select **approved**.

__ e. In the upper-right corner, click the **Download** icon and save the file to a temporary location, such as the **Downloads** folder.

__ f. Go to the folder that contains your scenario file and view it in Excel Viewer.

__ g. After you review the **Scenarios** tab and **Expected Results** tab, close the file.

__ 7. In Business console, click the **Loan Service** > **main** breadcrumb to return to the **Tests** tab.

__ 8. Create a test suite called: `My Test Suite`.

__ a. Click the plus (+) icon to add a new test suite.

__ b. When prompted to select a decision operation, select **loan validation production** and click **OK**.

__ c. In the **Name** field, change the name to: My Test Suite.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. In the **File to use** field, click **Choose** and browse to the `Loan Scenario-populated.xlsx` file that is provided for you in the **`C:\labfiles\code`** folder.

__ e. Click the link to the uploaded file, and select **Open with Microsoft Excel Viewer** to see the scenario data that is now provided in the file. You can leave this file open for later reference when you review the results of your test.

__ f. In the **Expected execution details to include in report** section, select **The list of rules fired**.

__ g. In the upper-right corner, click **Save and Run**.

__ h. When prompted, click **Create New Version** and click **OK** on the Run Test Suite window.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 9. After execution finishes, click the **Report** link on to view the results. The most recent execution report is at the top of the list.

The report shows that the tests ran without errors. The rules were executed correctly and produced a decision for each scenario. In this case, one scenario was successful and one failed.

The 50% success rate means that execution did not produce all the results that the business users expected. In such cases, business users must review the data that is provided in the scenario file to determine whether they should change the scenario or change the rules.

__ 10. Click **Close** to close the report

__ 11. Sign out of the Business console and close the browser window.

4.1. Restore the server

__ 1. In Rule Designer, switch to the Samples Console perspective.

__ 2. In the Sample Server view, click the **Restore the sample server** icon.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

To see the solution to this exercise, switch to a new workspace and import the project **Ex 14: Enabling rule validation > 02-answer**.

The exercise looked at how to validate decision service projects for business users by preparing the BOM and creating a customized scenario file template. It also demonstrated how business users generate test suites and run tests from Business console.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 16.Managing deployment

Estimated time

00:45

Overview

This exercise teaches you how to deploy rules and XOMs for managed execution with Rule Execution Server.

Objectives

After completing this exercise, you should be able to:

- Define a RuleApp and ruleset properties
- Use deployment configurations to deploy decision services
- Deploy the XOM for its management in Rule Execution Server

Introduction

After finalizing the content of the rule project, you are ready to deploy it to the execution environment in Rule Execution Server.

In this exercise, you create a RuleApp to package your ruleset for deployment to Rule Execution Server. You learn how to deploy from Rule Designer and how to manage the XOM in Rule Execution Server.

This exercise is divided into these sections:

- Section 1, "Creating deployment configurations"
- Section 2, "Deploying a RuleApp from Rule Designer"
- Section 3, "Adding a ruleset property to a deployment configuration"
- Section 4, "Deploying the managed XOM from Rule Designer"
- Section 5, "Exporting and importing deployment server definitions"

Requirements

This exercise uses the following files, which are installed in the `<InstallDir>\studio\training` directory.

- Start project: `Dev 15 - Deployment\01-start`
- Solution project: `Dev 15 - Deployment\02-answer`

You can use the solution project as described in "Exercise review and wrap-up" on page 16-18.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Creating deployment configurations

In this part of the exercise, you create a deployment configuration that can be used to deploy the decision service to Rule Execution Server or for a Java SE environment.

1.1. Setting up your environment for this exercise

__ 1. If the sample server is not started, start it now by clicking **Start** and then clicking the **Start Sample Server** shortcut.

Starting the server might take several minutes.

__ 2. In Rule Designer, switch to a new workspace:

__ a. From the **File** menu, click **Switch Workspace > Other**.

__ b. In the Workspace Launcher window, enter the path:

`<LabfilesDir>\workspaces\deploy`

__ c. Click **Launch**.

__ 3. Close the Welcome view.

__ 4. Use the Samples Console to import the exercise start project.

__ a. Click the **Open Perspective** icon.

__ b. In the Open Perspective window, click **Samples Console** and click **Open**.

__ c. In the **Rule Designer** section, expand **Training > Ex 15: Managing deployment** > **01-start**, and click **Import projects**.

__ d. When the workspace finishes building, close the Help view.

1.2. Creating a deployment configuration

__ 1. In the **Rule Project Map** view, in the **Deploy and Integrate** part of the Decision Service Map, click **Create deployment configuration**.

__ 2. Make sure that the **Deployment folder** field is set to: **/loan-rules/deployment**.

If it is not, click **Browse** and select the **loan-rules > deployment** folder, and click **OK**.

__ 3. In the **Name** field, type `loan-deploy` and click **Finish**.

The deployment configuration is now listed in Rule Explorer with errors because the configuration settings are not yet defined. The configuration editor opens to the **Overview**

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

tab. The editor also includes the **Decision Operations** tab and the **Target Servers** tab, which you use next.

__ 4. On the **Overview** tab, in the **General** section, in the **Deploy the XOM** option, select **No**.

__ 5. In the **Decision Operations** section, click **Include decision operations**.

The editor switches to the **Decision Operations** tab.

__ 6. Add the `loan-rulesOperation` decision operation to the configuration.

__ a. In the **Configured Decision Operations** section, click the plus sign (+).

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Choose **Select existing decision operations** and expand **loan-rules** to choose **loan-rulesOperation**.

__ c. Click **Finish**.

__ 7. Click the **Target Servers** tab and define the target server.

__ a. Click the plus sign (+) to add a server.

__ b. Leave **Create a Rule Execution Server connection** selected and click **Next**.

__ c. Define these fields:

- **Name**: `Sample`
- **URL**: `http://localhost:9090/res`
- **User name**: `resAdmin`
- **Password**: `resAdmin`

For IBM ODM on Cloud users

Use the name and URL of the Rule Execution Server for your IBM ODM on Cloud instance in the **URL** field instead of `Sample` and `localhost`.

__ d. Select **Save my login information**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ e. Click **Test Connection**.

You should see a `Connection succeeded` message.

__ f. Click **Finish**.

__ 8. Save your work by pressing Ctrl+S.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Deploying a RuleApp from Rule Designer

In this section, you deploy a RuleApp for its managed execution in Rule Execution Server.

2.1. Deploying the RuleApp

In this section, you deploy the RuleApp.

__ 1. Return to the **Overview** tab of the `loan-deploy` deployment configuration editor.

__ 2. In the **Deployment** section, click **Proceed to RuleApp deployment**.

__ 3. After reviewing the deployment summary, click **Next**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. After verifying the login credentials for the server, click **Next**.

__ 5. On the Version Summary page, click **Finish**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The Deployment Report opens and lists the deployed artifacts.

__ 6. Close the **Deployment Report** tab.

Questions

Do you notice a difference in the name of the deployment configuration? What is the name of the ruleset that was deployed?

Answer

The hyphen (-) character is not an authorized character in the name of a RuleApp or of a ruleset.

The RuleApp defined with the `loan-deploy` project is thus deployed as a `loan_deploy` file to Rule Execution Server. The same principle applies to the name of the deployed ruleset, which is modified as `loan_rulesRuleset`.

Important

This `/loan_deploy/1.0/loan_rulesRuleset/1.0` ruleset path, without the hyphen character, is what client applications must use to request the execution of the `loan_rulesRuleset` ruleset.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Adding a ruleset property to a deployment configuration

In this section, you add the `ruleset.sequential.trace.enabled` ruleset property to the `loan-deploy` deployment configuration.

__ 1. Reopen the `loan-deploy` deployment configuration and open the **Decision Operations** tab.

__ 2. In the **Configured Decision Operations** section, under **loan-rules**, select **loan-rulesOperation**.

__ 3. In the **Ruleset Properties** section, click the plus sign (+) to add a ruleset property.

__ 4. Define the ruleset property.

__ a. In the Ruleset Property Name And Value window, in the **Name** field list, type: `ruleset.sequential.trace.enabled`

__ b. In the **Value** field, type: `true`

__ c. Click **OK** to close the Ruleset Property Name And Value window.

__ d. Save your work.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. Open the **Source** tab to find the added XML code, and verify that the XML definition of the ruleset property is as follows:

```
<properties key="ruleset.sequential.trace.enabled">
  <value><![CDATA[true]]></value>
</properties>
```

__ 6. Return to the **Decision Operations** tab and delete the `ruleset.sequential.trace.enabled` property by selecting it and clicking the **X**.

You do not need this property for the rest of this exercise.

__ 7. Save your work (Ctrl+S).

The XML definition of the ruleset property is no longer visible in the XML code of the **Source** tab.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 4. Deploying the managed XOM from Rule Designer

In this part of the exercise, you deploy the XOM that the `loan-rules` project uses. When you deploy the XOM to Rule Execution Server, you can access and manage the XOM through the Rule Execution Server console.

4.1. Deploying the managed XOM from the rule project

In this section, you deploy the XOM directly from the rule project.

__ 1. Back in your Rule Designer workspace, in Rule Explorer, expand **loan-rules > resources > xom-libraries** to see the `loan-xom.zip` file.

__ 2. Deploy the XOM that the `loan-rules` project uses.

__ a. If the **`loan-deploy`** deployment configuration is closed, reopen it to the **Overview** tab.

__ b. In the **General** section, in the **Deploy the XOM** option, choose **Yes**.

__ c. Save your work (by pressing Ctrl+S).

__ d. In the **Deployment** section, click **Proceed to RuleApp deployment**.

__ e. If you see a “Java version notification” window, select the **Do not show this message again** check box and click **OK**.

__ f. The RuleApp Deployment window opens. Click **Next** until you reach the Version Summary page, and then click **Finish**.

__ 3. Look at the Deployment Report and notice the **XOM Deployment** section, which provides details on the deployment of the XOM.

Note

Depending on the order in which you completed the exercises, or the number of times you try deploying the XOM, you might see a different version number in your results.

The main point of this exercise is to recognize how you can use version numbers to manage deployments.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. In Rule Explorer, expand **loan-rules > resources > xom-libraries** and double-click the `loan-xom.zip` file to open it.

The `loan-xom.zip` file opens in Windows Explorer to the **training_loan** folder, which contains the classes.

4.2. Redeploying the XOM

In the next steps, you change the XOM, redeploy it, and verify that the version of the deployed XOM is incremented.

__ 1. In the Rule Explorer, expand **loan-xom > src > training_loan** and double-click the **`Test.java`** class.

__ 2. Edit the `Test.java` class by commenting the final line of the `main` method:

```
// System.out.println(r1);
```

__ 3. Save the `Test` class and wait for the workspace to rebuild.

If **Build Automatically** is selected on the **Project** menu, your workspace should build automatically after saving.

__ 4. Redeploy the XOM from the `loan-rules` project.

__ a. Right-click the **`loan-rules`** project and click **Rule Execution Server > Deploy XOM**.

The XOM Deployment wizard opens.

__ b. Click **Next**, and then click **Finish**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The **XOM Deployment** section of the Deployment Report shows that the `loan-xom` version is now incremented.

Because the XOM changed, its checksum also changed, as did the version of the managed XOM in Rule Execution Server.

Note

Depending on whether the XOM was deployed earlier during exploration of the Rule Execution Server, you might have a different version number. The main point of this exercise is to recognize how you can use version numbers to manage deployments.

4.3. Deploying a RuleApp associated with a managed XOM

In this section, you deploy the RuleApp again. But this time, you also indicate the managed XOM that is associated with this RuleApp.

__ 1. Undo the changes in the XOM that you made in Step on page 16-12.

__ a. In the `loan-xom` project, open the `Test` class and uncomment the final line of the `main` method:

```
System.out.println(r1);
```

__ b. Save the `Test` class and close the file and wait for your workspace to rebuild.

__ 2. Redeploy the XOM.

__ a. In Rule Explorer, right-click the `loan-rules` project and click **Rule Execution Server > Deploy XOM**.

__ b. The XOM Deployment window opens. Click **Finish**.

__ c. Look at the Deployment Report and look for this line:

```
The resource 'loan-zom-zip' is already deployed with version '1.0' on
Sample (http://localhost:9090/res). Skipping it.
```

The XOM is not redeployed because the XOM now matches the originally deployed XOM.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Information

The version in the URI of the managed XOM depends on the checksum that Rule Execution Server computes for this XOM. Because you restored the XOM to its initial state, the deployed XOM has the same checksum as the earlier version of the XOM.

As a consequence, Rule Execution Server considers that you deployed that XOM version again; so that URI is used.

Note

Depending on the number of times you tried deploying the RuleApp during this exercise, you might see a different version number in your results. The main point of this exercise is to recognize how you can use version numbers to manage deployments.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 5. Exporting and importing deployment server definitions

When you define a target server in a deployment configuration, that definition is stored at the workspace level. If you want to reuse the target server definition in another workspace, you can export the definition and save it. You then import it to the next workspace.

5.1. Exporting deployment configurations

__ 1. In Rule Designer, from the **File** menu, click **Export**.

__ 2. In the Export wizard, expand **Rule Designer**, select **Deployment Server Definition**, and click **Next**.

__ 3. Save the `server-list.xml` file to the `<LabfilesDir>/code` folder.

The file is now available for import to any workspace where you want to define a deployment configuration that uses the sample server.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

5.2. Importing deployment configurations

To see how to import a deployment server definition, you can switch to a new (empty) workspace.

__ 1. In Rule Designer, switch to a new workspace with the path:

`<LabfilesDir>\workspaces\deploy2`

__ 2. Use the Samples Console to import the exercise solution project.

__ a. Click the **Open Perspective** icon.

__ b. In the Open Perspective window, click **Samples Console** and click **Open**.

__ c. In the **Rule Designer** section, expand **Training > Ex 15: Managing deployment > 02-answer**, and click **Import projects**.

Note

The project imports with an error on the `loan-deploy` deployment configuration file.

__ 3. In Rule Explorer, expand **loan-rules > deployment** and double-click **loan-deploy** to see the error on the deployment configuration.

__ 4. Import the previously exported deployment server definition.

__ a. From the **File** menu, click **Import**.

__ b. Expand **Rule Designer**, select **Deployment Server Definition**, and click **Next**.

__ c. In the Import Deployment Server Definition window, set the **File** field to the `<LabfilesDir>/code/server-list.xml` file.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Select **Override existing servers** and click **Finish**.

__ 5. Open the **Project > Clean** menu and click **Clean**.

The errors should no longer be visible.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

The first part of the exercise looked at how to create a deployment configuration. You also learned how to add and remove a ruleset property. The next parts of the exercise showed you how to deploy a RuleApp and how to deploy a managed XOM.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 17.Using Build Command to build RuleApps

Estimated time

00:30

Overview

This exercise teaches you how to work with the Build Command tool, which is a Maven plug-in, to build projects into RuleApps for deployment.

Objectives

After completing this exercise, you should be able to:

- Define POM files
- Build a RuleApp archive from a set of projects

Introduction

In this exercise, you use Build Command tool to build a RuleApp.

The exercise includes these tasks:

- Section 1, "Setting up and installing Maven Build Command"
- Section 3, "Deploying your RuleApp archive with Ant"

Requirements

This exercise requires that the `JAVA_HOME` and the `ANT_HOME` environment variables be set. The Apache Maven plug-in and Node.js must also be installed. The Path environment variable must include `JAVA_HOME\bin`, `ANT_HOME\bin`, and `C:\IBM\apache-maven\bin`.

This exercise uses the files that are stored in the **`C:\labfiles\code\build`** folder.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Setting up and installing Maven Build Command

In this section, you prepare your environment to use Build Command, you write the POM files for your project, and then you build the project into a RuleApp.

1.1. Creating a workspace folder

__ 1. In the `C:\labfiles\workspaces` directory, create a folder called: `build`

__ 2. From the `C:\labfiles\code\build` directory, copy the **`PreTradeChecks`** and **`PreTradeChecks-xom`** folders to your `C:\labfiles\workspaces\build` folder.

1.2. Creating a local Maven repository

__ 1. In the `C:\IBM` directory, create a folder called: `repository`

__ 2. Edit the Maven configuration settings.

__ a. Open the `C:\IBM\apache-maven\conf` folder, right-click `settings.xml`, and open with WordPad.

__ b. Uncomment the `<localrepository>` tag and insert the path: `C:\IBM\repository`

```
  <!-- localRepository
   | The path to the local repository maven will use to store artifacts.
   |
   | Default: ${user.home}/.m2/repository
-->
  <localRepository>C:\IBM\repository</localRepository>
```

__ c. Save the file and close it.

Note

If you do not create a repository, Maven uses a default repository.

1.3. Installing the Maven Build Command plug-in

__ 1. Open a command prompt window and go to the Build Command compiler directory:

```
cd C:/IBM/ODM810/buildcommand/rules-compiler
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. Run the following command:

```
mvn install:install-file -Dfile=rules-compiler.jar
-DpomFile=rules-compiler-maven-plugin.pom
```

```
[INFO] Installing C:\IBM\ODM892\buildcommand\rules-compiler\rules-compiler-maven
-plugin.pom to C:\IBM\repository\com\ibm\rules\buildcommand\rules-compiler-maven
-plugin\8.9.2.0\rules-compiler-maven-plugin-8.9.2.0.pom
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 16.812 s
[INFO] Finished at: 2018-06-27T08:03:57-04:00
[INFO] ------------------------------------------------------------------------

C:\IBM\ODM892\buildcommand\rules-compiler>
```

This command adds the following plug-in in your local Maven repository:

```
com/ibm/rules/buildcommand/rules-compiler
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Setting up the project structure

To build the project into a RuleApp, you must write the following POM files for the project:

- Decision service POM for the deployment configuration in the decision service
- XML POM for the XOM
- Aggregator XOM

For this exercise, you use the sample POM files that are stored in the `<InstallDir>\buildcommand\samples\HelloWorld` folder as a starting point for your project POM files. You then edit the files so that they work with the PreTradeChecks application.

2.1. Creating the aggregator POM file

In the aggregator POM file, you set these properties:

- `groupId`
- `module`

__ 1. Open the `C:\IBM\ODM810\buildcommand\samples\HelloWorld` directory.

__ 2. Copy the aggregator `pom.xml` file from the `HelloWorld` root directory to your `C:\labfiles\workspaces\build` workspace folder.

__ 3. Right-click the `pom.xml` file, and click **Open with WordPad**.

__ 4. Set the `<groupId>` tag to: `<groupId>pretrade</groupId>`

__ 5. In the `<modules>` section, set the `<module>` tags to:

```
<module>PreTradeChecks-xom</module>
<module>PreTradeChecks</module>
```

```
<groupId>pretrade</groupId>
<artifactId>parent</artifactId>
<version>1.0.0</version>

<packaging>pom</packaging>

<modules>
    <module>PreTradeChecks-xom</module>
    <module>PreTradeChecks</module>
</modules>
```

__ 6. Save the POM file and close it.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

2.2. Creating the decision service POM file

In the aggregator POM file, you set these properties:

- `groupId`
- `artifactId`
- `deployments / name`
- `resolvers / kind | url | artifactKey`
- `dependencies / artifactId`

__ 1. Go to the `C:\IBM\ODM810\buildcommand\samples\HelloWorld\Hello Main Service` folder and copy the `pom.xml` file

__ 2. Paste the `pom.xml` file in the `C:\labfiles\workspaces\build\PreTradeChecks` folder.

__ 3. In the **PreTradeChecks** folder, right-click the `pom.xml` file that you copied, and open it with WordPad.

__ 4. In the `<parent>` section, set the `<groupId>` tag to: `<groupId>pretrade</groupId>`

__ 5. After the `<parent>` section, set the `<artifactId>` tag (which is set to `hello-main`) to: `<artifactId>PreTradeChecks</artifactId>`

__ 6. In the `<deployments>` section, set the `<name>` to: `<name>PreTradeChecksRuleApp</name>`

__ 7. Edit the `<resolvers>` section and the `<dependencies>` section to match the `.ruleproject` entries in the **PreTradeChecks** folder.

__ a. In the `C:\labfiles\workspaces\build\PreTradeChecks` folder, right-click the `.ruleproject` file and open it with Notepad.

__ b. Scroll to the paths section to find the XOM entry, and note that the values for `url` and `kind` are `platform:/PreTradeChecks-xom` and `JAVA_PROJECT`.

```
<rulesetProperties></rulesetProperties>
<paths xsi:type="ilog.rules.studio.model.xom:XOMPath" pathID="XOM">
  <entries xsi:type="ilog.rules.studio.model.xom:LibraryXOMPathEntry"
  name="org.eclipse.jdt.launching.JRE_CONTAINER"
  url="file:org.eclipse.jdt.launching.JRE_CONTAINER" kind="LIBRARY"
  exported="false"/>
  <entries xsi:type="ilog.rules.studio.model.xom:SystemXOMPathEntry"
  name="PreTradeChecks-xom" url="platform:/PreTradeChecks-xom"
  kind="JAVA_PROJECT" exported="true"/>
</paths>
```

__ c. In the `<resolvers>` section of the `pom.xml` file, set the `url` and `kind` values to match the `.ruleproject` file values:

```
<kind>JAVA_PROJECT</kind>
<url>platform:/PreTradeChecks-xom</url>
```

__ d. In the `<resolvers>` section, set the `<artifactKey>` value to:

```
<artifactKey>${project.groupId}:PreTradeChecks-xom</artifactKey>
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 8. In the `<dependencies>` section, set the `<artifactId>` tag to:
`<artifactId>PreTradeChecks-xom</artifactId>`

__ 9. Save the POM file and close it.

2.3. Creating the XOM POM file

In the XOM POM file, you set these properties:

- `groupId`
- `artifactId`

__ 1. In the `C:\IBM\ODM810\buildcommand\samples\HelloWorld\Hello XOM` folder, copy the `pom.xml` file, and paste it in to the `C:\labfiles\workspaces\build\PreTradeChecks-xom` folder.

__ 2. In the **PreTradeChecks-xom** folder, right-click the `pom.xml` file, and open it with Notepad or Wordpad.

__ 3. In the `<parent>` section, set the `<groupId>` tag to: `<groupId>pretrade</groupId>`

__ 4. After the `<parent>` section, set the `<artifactId>` tag to:
`<artifactId>PreTradeChecks-xom</artifactId>`

__ 5. Save the POM file and close it.

2.4. Building the project

__ 1. Open a command prompt window and go to your workspace directory:

```
cd C:\labfiles\workspaces\build
```

__ 2. Enter the following command:

```
mvn clean install
```

The build takes a few moments. After the build is done, you should see a `BUILD SUCCESS` message.

```
[INFO] Installing C:\labfiles\workspaces\build\PreTradeChecks\pom.xml to C:\IBM\
repository\pretrade\PreTradeChecks\1.0.0\PreTradeChecks-1.0.0.pom
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] parent 1.0.0 ..................................... SUCCESS [  1.968 s]
[INFO] PreTradeChecks-xom ............................... SUCCESS [ 45.313 s]
[INFO] PreTradeChecks 1.0.0 ............................. SUCCESS [ 30.687 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 02:13 min
[INFO] Finished at: 2018-06-27T08:31:51-04:00
[INFO] ------------------------------------------------------------------------

C:\labfiles\workspaces\build>
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Troubleshooting

If you run into build errors, try the following actions:

- Double-check the settings in all three of the POM files, and make sure that they match the exercise instructions.
- Make sure that all POM files are saved.
- Run the `mvn clean` command to clean the local repository.
- If you are still getting build errors, run the `install` command again with the `-e` and `-X` switches appended to the command:

  ```
  mvn clean install -e -X
  ```

  The `-e` switch enables the full stack trace for the errors, and the `-X` switch enables full logging for debugging purposes.

  You can then review the messages and information in the command prompt window to help troubleshoot your installation issues.

__ 3. Open the `C:\IBM\repository\pretrade` directory to find your RuleApp.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Deploying your RuleApp archive with Ant

In this section, you use an Ant task to deploy your RuleApp.

__ 1. Make sure that the sample server is running. If you need to start it, click the **Start Sample Server** shortcut on the Windows **Start** menu.

__ 2. Go to the `C:\labfiles\code\build` folder and open the `build.xml` file with Notepad or Wordpad.

__ 3. Make sure that the `ds.home` property is set to: `C:/IBM/ODM810`

__ 4. Copy the `build.xml` file from the `C:\labfiles\code\build` folder to the `C:\IBM\repository\pretrade` folder.

__ 5. Open a command prompt window and go to the `C:\IBM\repository\pretrade` folder.

```
cd C:\IBM\repository\pretrade
```

__ 6. Deploy the RuleApp archive by typing the following Ant command:

```
ant deploy -Dserver.port=9090
```

After the build is done, you should see a `BUILD SUCCESSFUL` message.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

This exercise looked at how to use the build command tool to build projects into RuleApps for deployment. You also saw how to use Ant to deploy the RuleApp.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 18. Exploring the Rule Execution Server console

Estimated time

00:30

Overview

This exercise teaches you how to work with the Rule Execution Server console.

Objectives

After completing this exercise, you should be able to:

- Work with Rule Execution Server console tools
- Manage RuleApps and rulesets through the Rule Execution Server console

Introduction

After you deploy RuleApps and XOMs to Rule Execution Server, you can manage them through the Rule Execution Server console.

In this exercise, you explore the Rule Execution Server console environment and features.

The exercise is divided into these sections:

- Section 1, "Exploring the Rule Execution Server console"
- Section 2, "Exploring the deployed RuleApps"
- Section 3, "Working with deployed resources"
- Section 4, "Exploring the Diagnostics and Server Info tabs"
- Section 5, "Exploring the REST API tab"
- Section 6, "Managing RuleApps and rulesets"

Requirements

You should complete Exercise 16, "Managing deployment" and Exercise 17, "Using Build Command to build RuleApps" before starting this exercise.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Exploring the Rule Execution Server console

1.1. Setting up your environment for this exercise

__ 1. Make sure that the sample server is started. (Click **Start Sample Server** on the **Start** menu.)

For IBM ODM on Cloud users

You do not need to do this step. Your Rule Execution Server instance is already running in your IBM ODM on Cloud environment.

1.2. Exploring the Rule Execution Server console pages

__ 1. Open the Rule Execution Server console.

__ a. On the Windows **Start** menu, click the **Rule Execution Server console** shortcut.

__ b. In both the **User Name** and **Password** fields, enter: `resAdmin`

For IBM ODM on Cloud users

Instead of going through your operating system, you start the Rule Execution Server console by logging in to the User Portal, then clicking **Launch** under **Rule Execution Server console**.

The Rule Execution Server console is configured to open to the URL of your ODM on Cloud instance, so you do not need to enter a URL or port number.

__ 2. Note the tabs: **Home**, **Explorer**, **Decision Warehouse**, **Diagnostics**, **Server Info**, and **REST API**.

Note

You work with the **Decision Warehouse** tab in Exercise 19, "Auditing ruleset execution through Decision Warehouse".

__ 3. Click the **Explorer** tab.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The Navigator pane gives access to the elements that you can manage in the Rule Execution Server console.

- The **RuleApps** link gives access to the RuleApps View pane, where you can manage the deployed RuleApps.
- The **Resources** link gives access to the Resources View pane, where you can manage resources.
- The **Libraries** link gives access to the Libraries View pane, where you can manage libraries.
- The **Service Information** link gives access to the Transparent Decision Services View pane, where you can manage the created decision services.

By default, the Explorer page opens with the RuleApps View pane visible, and you can see the RuleApps that you deployed in Exercise 16, "Managing deployment".

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Exploring the deployed RuleApps

In this section, you explore the RuleApps deployed in Rule Execution Server, and relate them to what you did in Exercise 16, "Managing deployment".

Note

Depending on the number of deployments that you tried before doing this exercise, the version numbers that you see in the Rule Execution Server console might differ from the versions in these instructions.

Exploring the RuleApp deployed from Rule Designer

__ 1. In the Navigator pane, expand **RuleApps > /loan_deploy/1.0**.

These artifacts are the ones that you deployed in Exercise 16, "Managing deployment".

__ 2. In the Navigator pane, click **loan_deploy/1.0** to open the deployed RuleApp on the RuleApp View page.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. In the Navigator pane, click **loan_rulesRuleset/1.0** to open the ruleset on the Ruleset View page.

The Ruleset View pane opens, where you can see the `borrower`, `loan`, and `report` parameters that are defined for this ruleset.

__ 4. In the Navigator page, click **loan_rulesRuleset/1.1** to open the other deployed ruleset, and in the Ruleset View page, click **Show Managed URIs**.

You can see that this ruleset is associated with a deployed XOM. The URI, `reslib://loan_deploy_1.0/1.0`, corresponds to the XOM that you deployed in Exercise 16, "Managing deployment".

__ 5. In the Navigator page, expand **PreTradeChecksRuleApp/1.0**, click **PreTradeChecks/1.0** to open the deployed ruleset, and in the Ruleset View page, click **Show Managed URIs**.

Notice that the RuleApp does not have a XOM that is associated with it. Later in the exercise, you deploy it.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Working with deployed resources

In this section, you explore the XOM resources that are deployed in Rule Execution Server, and relate them to what you did in Exercise 16, "Managing deployment".

__ 1. In the Navigator pane, click **Resources**.

__ 2. On the Resources View page, click the **loan-xom.zip** link for the first XOM that you deployed in Exercise 16, "Managing deployment". Depending on the number of times that you deployed, you might have several versions.

__ 3. Click **Show references from Rulesets** to see which rulesets reference this XOM. If no rulesets reference this XOM, click **Resources** again in the Navigation pane and click a different XOM link.

__ 4. Click **Show references from Libraries** to see which XOM libraries reference this XOM. If no rulesets reference this XOM, click **Resources** again in the Navigation pane and click a different XOM link.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 4. Exploring the Diagnostics and Server Info tabs

In this section, you explore the Diagnostics and Server Info pages.

__ 1. Explore the Diagnostics page.

__ a. Click the **Diagnostics** tab.

__ b. Click **Rerun**.

__ c. Click **Expand All** to view the results.

__ 2. Explore the Server Info page.

__ a. Click the **Server Info** tab.

__ b. Notice the location of the log file in the **Log File** field, the logging level, and the server that is being used for the execution unit. You can view errors that occur on that server by clicking the server name link.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 5. Exploring the REST API tab

The REST API tool is a web application that you can use to test the management capabilities of REST resources and their associated parameters. REST resources include RuleApps, rulesets, XOM libraries, and XOM resources.

5.1. Exploring REST API commands

__ 1. In the Rule Execution Server console, click the **REST API** tab to display the test tool.

__ 2. Explore the **REST API** tab by clicking the tab for each resource and noting which methods are available:

- `/ruleapps`
- `/rulesets`
- `/libraries`
- `/xoms`
- `/decisiontraces`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

5.2. Deploying with REST

__ 1. Deploy the PreTradeChecks XOM by using the REST API feature.

__ a. On the **REST API** tab, click the **/xoms** tab.

__ b. Select **POST - deployResource**.

__ c. In the Request Body section, click **Browse** and go to:
`C:\IBM\repository\pretrade\PreTradeChecks-xom\1.0.0`

__ d. Select `PreTradeChecks-xom-1.0.0.jar` and click **Open**.

__ e. Set the **xomname** field to: `PreTradeChecks-xom.jar`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ f. Click **Call method**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. In the Response section, make sure that you see `HTTP Status 201`.

__ 3. Go back the **Explorer** tab, and in the Navigator, expand the **Resources** folder.

`PreTradeChecks-xom.jar/1.0` is listed as a resource.

5.3. Adding references to the XOM

Both the RuleApp and XOM are deployed, but before you can test ruleset execution, the RuleApp and XOM need to “see” each other. Next, you add references to the URI of the managed XOM resource.

__ 1. Go back to the **REST API** tab.

__ 2. If the **/ruleapps** tab is not open, click it.

__ 3. Scroll down to **POST - addRulesetProperty** and click it.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. Set the request parameters:

- **Request Body**: `resuri://PreTradeChecks-xom.jar/1.0`
- **ruleappname**: `PreTradeChecksRuleApp`
- **ruleappversion**: `1.0`
- **rulesetname**: `PreTradeChecks`
- **rulesetversion**: `1.0`
- **propertyname**: `ruleset.managedxom.uris`

__ 5. Click **Call method**.

__ 6. Make sure that the Response section shows `HTTP Status 200`.

__ 7. Go back to the **Explorer** tab.

__ 8. In the Navigator, expand **RuleApps > PreTradeChecksRuleApp > 1.0** and select the **PreTradeChecks** ruleset.

__ 9. In the Ruleset View, click **Show Managed URIs**. The resource URI is set to the XOM resource.

Note

You work more with the REST API in Exercise 20, "Executing rules as a hosted transparent decision service (HTDS)".

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 6. Managing RuleApps and rulesets

In this section, you see how to manage resources through the Rule Execution Server console.

6.1. Creating a RuleApp

__ 1. Click the **Explorer** tab.

__ 2. In the RuleApps View, click **Add RuleApp**.

__ 3. In New RuleApp pane, leave the **Name** field set to: `NewRuleApp`

__ 4. In the **Version** field, keep: `1.0`

__ 5. In the **Display Name** field, enter: `Test RuleApp`

__ 6. Click **Add**.

The RuleApp View now shows the properties of the `NewRuleApp/1.0` RuleApp.

6.2. Adding a ruleset to your RuleApp

__ 1. Add a ruleset to the new `NewRuleApp`.

__ a. In the RuleApp View pane, click **Add Ruleset**.

The New Ruleset pane opens.

__ b. In the **Name** field, keep: `NewRuleset`

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. In the **Version** field, keep: `1.0`

__ d. In the **Display** Name field, enter: `Test Ruleset`

__ e. Click **Browse** next to **Local path of ruleset archive** and go to the `output` folder of the workspace that you used in section Section 5.2, "Importing deployment configurations" of Exercise 16, "Managing deployment":

`<LabfilesDir>\workspaces\deploy\loan-rules\output`

__ f. Select the **`loan_rulesRuleset.dsar`** file, and click **Open**.

__ g. On the New Ruleset page, leave the other fields unchanged, and click **Add**.

The `NewRuleApp/1.0/NewRuleset/1.0` ruleset is now created.

__ 2. On the Ruleset View page for the `NewRuleApp/1.0/NewRuleset/1.0` ruleset, scroll down the page and click **Show Archive Content**.

__ 3. Expand the folders to explore the content of the ruleset archive.

6.3. Deleting the RuleApp

__ 1. In the Navigator pane, click **RuleApps** to open the list of RuleApps in the RuleApps View.

__ 2. In the list of RuleApps, select **NewRuleApp** and click **Remove**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. When prompted to confirm the removal of the RuleApp, click **Confirm**.

This RuleApp is not required for the rest of the course.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

This exercise looked at how you can work with the Rule Execution Server console. You explored the different pages of the Rule Execution Server console. You also learned how to manage the RuleApps, the rulesets, and the managed XOMs.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 19. Auditing ruleset execution through Decision Warehouse

Estimated time

00:45

Overview

This exercise describes how to enable monitoring of ruleset execution and how to audit execution traces in Decision Warehouse.

Objectives

After completing this exercise, you should be able to:

- Enable monitoring for ruleset execution
- Retrieve decision traces through Decision Warehouse
- Optimize Decision Warehouse
- Delete trace data from Decision Warehouse

Introduction

After the rules are deployed in the execution environment, you must be able to audit which rules were executed to determine the decision. In this exercise, you learn how to enable monitoring for a ruleset and deploy it from Rule Designer. Next, you test rule execution to generate rule execution traces that you can review in Decision Warehouse.

You also configure monitoring properties and deploy from Decision Center to see how execution traces in Decision Warehouse can link you directly to the corresponding rule artifacts in Decision Center.

The exercise includes these sections:

- Section 1, "Enabling ruleset monitoring"
- Section 2, "Retrieving decision traces in the Rule Execution Server console"
- Section 3, "Optimizing Decision Warehouse"
- Section 4, "Deleting trace information from the database"

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Requirements

This exercise requires that you complete the following exercises before proceeding:

- Exercise 17, "Using Build Command to build RuleApps"
- Exercise 18, "Exploring the Rule Execution Server console"

Note

For IBM ODM on Cloud users

This exercise requires that you use the on-premises version of ODM. ODM on Cloud does not support Decision Warehouse.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Enabling ruleset monitoring

You can set properties on your ruleset to capture execution traces and determine the behavior of the ruleset during execution.

When you enable monitoring, you can retrieve information such as:

- How many tasks were executed
- How many rules were executed
- What events took place in the working memory
- Which version of the ruleset was executed

You set monitoring properties at ruleset level. You can set this ruleset property in the Rule Execution Server console. By default, all the information is retrieved, except working memory events.

In this section, you enable these monitoring properties on your ruleset.

- `monitoring.enabled` with its value set to `true`
- `ruleset.sequential.trace.enabled` with its value set to `true`

You then execute the ruleset to generate traces. After execution, you review the traces in Decision Warehouse.

Information

The `ruleset.sequential.trace.enabled` property when the ruleset contains tasks that use the sequential or the Fastpath execution modes.

1.1. Enabling ruleset execution monitoring

__ 1. If the Rule Execution Server console is not already open, sign in as an administrator.

__ a. On the Windows **Start** menu, click the **Rule Execution Server console** shortcut.

__ b. In the **Username** and **Password** fields, enter `resAdmin` and click **Sign In**.

__ 2. Open the **Explorer** tab, and in the Navigator page, expand **RuleApps > PreTradeChecksRuleApp/1.0**, click **PreTradeChecks/1.0** to open the Ruleset View page for your deployed ruleset.

__ 3. In the Ruleset View, click **Show Properties** and note that this ruleset does not have defined properties.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. Click **Show Monitoring Options**.

__ 5. Click the **Edit** icon next to **Tracing in Decision Warehouse is currently disabled**.

__ 6. Select **Enable tracing in Decision Warehouse** and click **Save** at the bottom of the **monitoring options** section.

The **monitoring options** section lists the monitoring properties that can now be retrieved in Decision Warehouse.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 7. Click **Show Properties** and note that four properties are now set on the ruleset to enable monitoring, including `monitoring.enabled` and `ruleset.sequential.trace.enabled`, which are both set to true.

1.2. Testing the RuleApp to generate execution traces

__ 1. On the toolbar of the `PreTradeChecks` Ruleset View, click **Retrieve HTDS Description File**.

__ 2. Select the following options, and click **Test**:

- Service protocol type: **REST**
- Format: **OpenAPI-JSON**

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. On the **REST Service** page, replace the template code inside the curly bracket with the following customer and order parameters:

```
"customerParameter": {
"name": "John Smith",
"owner": "George Smith@IBM",
"initialValue": 2000,
"currentValue": 800,
"sectorPosition": 800,
"bankStockPosition": 800,
"oilStockPosition": 800,
"railStockPosition": 800,
"bondPosition": 800,
"futurPosition": 800,
"stockPosition": 800,
"currency": "Euro"
},
"orderParameter": {
"id": "string",
"stock":
"Company X",
"amount": 90,
"exchange": "Nouveau Marche",
"currency": "Euro",
"destination": "France",
"action": "Buy",
"securityType": "Stock",
"dateTime": "2004-02-14T19:44:14.000+0000",
"sector": "Oil",
"status": "Pending",
"reasons": "BadClient"
```

Information

You can copy and paste the code snippet from the `<LabfilesDir>\code\dw.txt` file into the **Execution Request** field.

__ 4. Click **Execute Request**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. Check the Server Response, which shows the status as "BLOCKED":

Server Response:


```
{
  "__DecisionID__": "e73fdf2a-7ef5-4522-8cc3-3f45848619ef0",
  "reportParameter": {
    "id": "string",
    "stock": "Company X",
    "amount": 90,
    "exchange": "Nouveau Marche",
    "currency": "EURO",
    "destination": "France",
    "action": "BUY",
    "securityType": "STOCK",
    "dateTime": "2017-02-14T19:44:14.000+0000",
    "sector": "OIL",
    "status": "BLOCKED",
    "reasons": "BLACK_LIST_STOCKS"
  }
}
```


__ 6. Click **Execute Request** 3 or 4 more times.

__ 7. Keep the **REST Service** tab open, but go back to the **Rule Execution Server** console tab. You can return to the **REST Service** tab again later to retest the decision service.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Retrieving decision traces in the Rule Execution Server console

In this section, you work in the Rule Execution Server console. You examine the RuleApp that you deployed from Rule Designer. You then explore the decision traces that are associated with the execution of the ruleset that is packaged in this RuleApp.

__ 1. Return to the Rule Execution Server console window, and click the **Decision Warehouse** tab.

The **Decision Warehouse** tab includes these tasks:

- **Search Decisions**: To define filters on the data source to find only the events or decisions of interest.
- **Persistence Properties**: To select the data source for Decision Warehouse to query during your web session. Decision Warehouse is installed with a default data source. You can add more sources, such as a historical database. When your web session expires, the active data source reverts to the default data source.
- **Clear Decisions**: To delete existing decisions from the Decision Warehouse.

By default, the Search Decisions page opens. You can switch between these pages with the “Select a task” left pane of the **Decision Warehouse** tab.

__ 2. On the Search Decisions page, leave all fields empty and click **Search** at the bottom of the **Search Decisions** page.

__ 3. Scroll down the page to see the results.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

All decisions that are stored in Decision Warehouse are listed.

5 Decision(s) found

Decision ID	Date	Ruleset Version	Number of rules fired	Decision
5c82abe0-0d4c-433d-8c2b-e77c497df03d0	2018-06-27 15:00:03	/PreTradeChecksRuleApp/1.0/PreTradeChecks/1.0	4	View Decis details
3c46c23d-5b6b-43b3-99dd-e027b6f028ac0	2018-06-27 15:00:01	/PreTradeChecksRuleApp/1.0/PreTradeChecks/1.0	4	View Decis details
713eff05-2f0e-43d9-bb3b-7a930d647e960	2018-06-27 14:59:58	/PreTradeChecksRuleApp/1.0/PreTradeChecks/1.0	4	View Decis details
b23d3f00-2900-4763-a36d-6d258e223fa70	2018-06-27 14:59:46	/PreTradeChecksRuleApp/1.0/PreTradeChecks/1.0	4	View Decis details
4d985dbe-6b1f-404d-9453-2bb3bd17f4d70	2018-06-18 20:02:31	/my_deployment/1.0/my_operation/1.0	0	View Decis details

__ 4. In the **Decision Trace** column for the most recent decision, click the **View Decision details** link.

	Date	Ruleset Version	Number of rules fired	Decision Trace	Process
3d-8c2b-	2018-06-27 15:00:03	/PreTradeChecksRuleApp/1.0/PreTradeChecks/1.0	4	View Decision details	10
b3-99dd-	2018-06-27 15:00:01	/PreTradeChecksRuleApp/1.0/PreTradeChecks/1.0	4	View Decision details	10
9-bb3b-	2018-06-27 14:59:58	/PreTradeChecksRuleApp/1.0/PreTradeChecks/1.0	4	View Decision details	17
63- 70	2018-06-27 14:59:46	/PreTradeChecksRuleApp/1.0/PreTradeChecks/1.0	4	View Decision details	345
4d- 70	2018-06-18 20:02:31	/my_deployment/1.0/my_operation/1.0	0	View Decision details	547

The Decision Trace page opens, and shows the details of the execution.

__ 5. Look at the fields in the **Execution Details** section to relate them to the ruleset execution.

Execution Details

Decision ID:	2fa043f0-747f-44f2-931e-b722cfd21a220
Date:	2017-03-20 19:24:36
Executed ruleset path:	/PreTradeChecksRuleApp/1.0/PreTradeChecks/1.0
Engine type:	de
Processing Time (ms)	0
Number of rules fired	4
Number of tasks executed	5

Your RuleApp version and statistics might differ from this screen capture.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 6. Expand the Ruleflow Tasks tree in the **Decision Trace** section.

__ 7. Close the Decision Trace window.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Optimizing Decision Warehouse

When the trace is enabled, Decision Warehouse captures all execution trace data. You can optimize Decision Warehouse by filtering which parts of the ruleset execution trace you monitor and store, and by turning off BOM serialization.

3.1. Working with monitoring options

__ 1. In Rule Execution Server console, return to the Ruleset View for your `PreTradeChecks` ruleset.

__ a. Click the **Explorer** tab.

__ b. In the Navigator pane, expand **RuleApps > PreTradeChecksRuleApp/1.0** and click **PreTradeChecks/1.0**.

__ 2. Identify the connection between the **Enable tracing in Decision Warehouse** option and the `monitoring.enabled` ruleset property.

__ a. In the monitoring options section, click **Edit** to open the monitoring options.

__ b. Clear **Enable tracing in Decision Warehouse** and click **Save**.

__ c. If necessary, refresh Rule Execution Server console by pressing F5.

Troubleshooting

If you see a prompt from the web browser to confirm whether you want to resend information, click **Resend** or **Retry**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Look in the **properties** section and verify that the `monitoring.enabled` ruleset property is now set to: `false`

__ 3. Enable decision tracing again.

__ a. In the **monitoring options** section and click the **Edit** icon.

__ b. Select **Enable tracing in Decision Warehouse** and click **Save**.

Notice that the monitoring options section now lists specific execution traces.

3.2. Specifying filters on the trace data

To reduce the information that is stored in an execution trace in Decision Warehouse, you:

- Apply the `monitoring.filters` property to the ruleset
- Select which filters to set so that you can refine the data that is stored

You can set the `monitoring.filters` property filter values in Rule Designer or Decision Center before you deploy a ruleset. You can also set this property on the deployed ruleset in Rule Execution Server console, which you do next.

__ 1. In the **monitoring options** section, click **Edit** to reopen the monitoring properties.

__ 2. In the **Select the execution traces to store in the Decision Warehouse** list, notice the complete list of execution traces that are available and which traces are selected.

The selected traces are listed as values of the `monitoring.filters` ruleset property.

__ 3. Select *all* the remaining execution traces in the list, and click **Save**.

__ 4. Notice that the `monitoring.filters` property disappears from the list of properties.

Selecting all the optional filters is equivalent to turning off the filter property.

__ 5. Click **Edit** to reopen the monitoring options, clear all the selected traces, and then click **Save**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 6. Notice that the `monitoring.filters` property is in the list of properties again, but its value is empty.

__ 7. Turn on the default monitoring filters again.

__ a. Click **Edit** to reopen the monitoring options, clear **Enable tracing in Decision Warehouse** and click **Save**.

__ b. Again, click **Edit** to reopen the monitoring options, select **Enable tracing in Decision Warehouse** and click **Save**.

3.3. Removing BOM serialization

When the `ruleset.bom.enabled` property is set to `true`, it converts the list of objects in the parameters into a memory buffer by using BOM serialization.

Important

For large amounts of input and output data, BOM serialization can result in poor performance.

To optimize performance, you can turn off BOM serialization.

__ 1. If the **properties** section for this ruleset is closed, click **Show Properties**, and click the **Edit** icon for the `ruleset.bom.enabled` ruleset property.

__ 2. Set `ruleset.bom.enabled` to `false` and click the **Save** icon.

Note

You can edit only one property at a time. If you try to edit the values of more than one property, only one of the values is saved.

__ 3. In the **monitoring options** section, click **Edit** to reopen the monitoring options.

__ 4. Notice that **Native format** is selected as the option for storing the values of ruleset parameters.

When BOM serialization is turned off, the BOM objects that are passed as parameters are stored either in XML for dynamic XOMs or in a string representation for Java XOMs.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. Return to the REST Service window to retest the ruleset as described in Section 1.2, "Testing the RuleApp to generate execution traces".

__ 6. In Rule Execution Server console, open the **Decision Warehouse** tab, and click **Search** to get the latest traces.

__ 7. Click **View Decision details** for your latest trace and notice the format for the input and output parameters.

__ 8. Click **View Decision details** for the previous trace and take a moment to compare the formats of the input and output parameters between the two traces.

__ 9. Close the Decision Trace windows.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 4. Deleting trace information from the database

You can delete trace information from the Decision Warehouse database by specifying the ruleset paths or execution dates.

4.1. Clearing traces

__ 1. In the Rule Execution Server console, on the **Decision Warehouse** tab, click **Search Decisions > Search** and look at the list of decisions to see RuleApp version numbers and dates.

__ 2. In the “Select a task” pane, click **Clear Decisions**.

__ 3. On the Clear Decisions page, you can either leave all fields empty to delete all traces, or specify which traces to delete, according to the RuleApp versions and dates that you saw listed.

For example, if you want to delete traces for a specific RuleApp version, you can specify which version to delete.

__ a. Leave the **Executed ruleset path** field blank.

Information

You can leave this field empty to erase all traces. You can also specify a partial or complete ruleset path to limit which traces are erased.

__ b. In the **From Date** field, click the arrow beside the field to open the calendar, and specify a date, such as yesterday’s date, or leave the field blank.

__ c. In the **To Date** field, leave the field blank.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. Click **Clear** to delete the specified traces.

__ 5. When prompted to confirm the deletion, click **Confirm**.

__ 6. Click **Search Decisions** and click **Search** to verify the deletion.

4.2. Removing the RuleApp

__ 1. In the Rule Execution Server console, click the **Explorer** tab.

__ 2. In the RuleApp View, select **PreTradeChecks** and click **Remove**.

__ 3. When prompted to confirm the deletion, click **Confirm**.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

This exercise looked at how you can use Decision Warehouse to audit ruleset executions.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise 20. Executing rules as a hosted transparent decision service (HTDS)

Estimated time

01:00

Overview

This exercise teaches you how to execute rules as a hosted transparent decision service (HTDS).

Objectives

After completing this exercise, you should be able to:

- Retrieve a WSDL description and call the decision service in a generated client
- View and test a ruleset in REST by using an OpenAPI in the Rule Execution Server console
- Retrieve an OpenAPI description for API Connect, and run the decision service through API Connect

Introduction

In this exercise, you review all the steps that are required to use hosted transparent decision services in an SOA environment.

After you deploy the ruleset as a RuleApp to Rule Execution Server, you create an HTDS from this ruleset. You then create a web service client application that uses the HTDS. After you run the application, you monitor the execution results in Rule Execution Server console by using the statistics that are associated with the HTDS.

This exercise is divided into these sections:

- Section 1, "Deploying the decision service to Rule Execution Server"
- Section 2, "Getting the HTDS WSDL file from the deployed ruleset"
- Section 3, "Testing the decision service with REST"
- Section 4, "Retrieving an OpenAPI description and calling the service in a generated client"

Requirements

This exercise requires that the `JAVA_HOME` environment variable be set and be included in the Path environment variable. The Apache Maven plug-in and Node.js must also be installed. This exercise also requires that you have an IBM Cloud account. The instructions for signing up are provided in this exercise.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 1. Deploying the decision service to Rule Execution Server

In this section, you deploy the RuleApp to Rule Execution Server.

1.1. Setting up your environment for this exercise

__ 1. If the sample server is not running, start it now by the **Start Sample Server** shortcut on the **Start** menu.

__ 2. In Rule Designer, switch to a new workspace:

__ a. From the **File** menu, click **Switch Workspace > Other**.

__ b. In the Workspace Launcher window, enter the path:

`<LabfilesDir>\workspaces\htds`

__ c. Click **Launch**.

__ 3. Close the Welcome view.

__ 4. Import the projects.

__ a. Switch to the Samples Console perspective.

__ b. Expand the **Rule Execution Server > Tutorials > Running a hosted transparent decision service**, and click **Import projects**.

The projects open in the Rule perspective.

__ 5. After the workspace builds, close the Help and Sample Server views.

1.2. Deploying the RuleApp

__ 1. In Rule Designer, expand **PreTradeChecks > deployment** and double-click **PreTradeChecksRuleApp**.

__ 2. Define the sample server as the target server.

__ a. Click the **Target Servers** tab.

__ b. Click the plus sign (+) to add a server.

__ c. Leave **Create a Rule Execution Server connection** selected and click **Next**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Define these fields:

- **Name**: `Sample`
- **URL**: `http://localhost:9090/res`
- **User name**: `resAdmin`
- **Password**: `resAdmin`

For IBM ODM on Cloud users

Use the name and URL of the Rule Execution Server for your IBM ODM on Cloud instance in the **URL** field instead of `Sample` and `localhost`.

__ e. Select **Save my login information**.

__ f. Click **Test Connection**.

You should see a `Connection succeeded` message.

__ g. Click **Finish**.

__ 3. Save your work by pressing Ctrl+S.

__ 4. Deploy the RuleApp.

__ a. In the **Deployment** section of the **Overview** tab, click **Proceed to RuleApp deployment**.

__ b. On the Deployment Summary page, click **Next**.

__ c. On the Credentials page, click **Next**.

__ d. On the Version Summary page, click **Finish**.

After the deployment finishes, the Deployment Report page opens.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 2. Getting the HTDS WSDL file from the deployed ruleset

In this section, you retrieve the HTDS WSDL file, which is the definition of the web service that is associated with the deployed ruleset.

Now that the `htdsRuleApp` is deployed with its associated managed XOM in Rule Execution Server, open the Rule Execution Server console to retrieve the corresponding decision service, and its WSDL interface.

2.1. Viewing the deployed ruleset

__ 1. In Rule Designer, in the Sample Server view, click the **Rule Execution Server console** link to open Rule Execution Server console, and sign in with `resAdmin` as the user name and password.

__ 2. Click the **Explorer** tab and in the list of RuleApps, click the **`PreTradeChecksRuleApp`** RuleApp that you deployed.

__ 3. Click the **`PreTradeChecks`** ruleset to open it in the Ruleset View.

The Ruleset View page opens with the details of this ruleset.

__ 4. On the Ruleset View page, click **Show HTDS Options**.

You see a series of options. If you have specific requirements in your enterprise, you can edit these options. For this exercise, you use the default settings.

__ 5. On the toolbar of the Ruleset View page, click **Retrieve HTDS Description File**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Information

A transparent decision service is technically either a strongly typed web service (WSDL) or a REST API that provides an interface to access a deployed rule set. The Decision Service component can pass one or more input parameters to the rule engine and access the return values. The transparent decision service support includes traceability from decision services to rules, runtime monitoring, and version management.

The HTDS is available in two forms: SOAP and REST. You can retrieve the file to view or to download.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

You can retrieve the HTDS with the following options:

Latest ruleset version **Latest RuleApp version**	You select **Latest ruleset version**, **Latest RuleApp version**, or both, to create a WSDL file that either uses a canonical ruleset path or not. In the present case, you select both check boxes to create a WSDL file that is based on a non-canonical ruleset path (where both versions are ignored) to execute the most recent RuleApp and ruleset versions.
Decision trace information	Select **Decision trace information** to set filters in the request, and later retrieve decision traces in the response.
Inline types in separate XSD files	You select or clear this check box, depending on whether you have models with the same name and the same namespace, and on how you want to manage their potential conflict. For more information, see the product documentation. For this exercise, you can clear this check box.
Compatibility mode	You select or clear this check box, depending on whether you want to create a WSDL file that is compatible with a previous version of Operational Decision Manager. If you select it, you must also select the appropriate version. In this course, you do not try to be compatible with any previous version of the product, so clear this check box.

2.2. Retrieving the WSDL

__ 1. Return to Rule Execution Server console and go to the Ruleset view for the **`PreTradeChecks`** ruleset.

__ 2. Download the HTDS description file as SOAP.

  __ a. Click **Retrieve HTDS Description File**.

  __ b. For **Service protocol type**, choose **SOAP**.

  __ c. Select **Decision trace information** and click **Download**.

  __ d. Select **Save File** and click **OK**.

__ 3. Open the **Downloads** folder and rename the file as: `pretradecheck.wsdl`

__ 4. Copy the `pretradecheck.wsdl` to the **`<InstallDir>\executionserver\samples\decisionservice\data`** folder to replace the existing file.

2.3. Building and running the Axis client

__ 1. Go to Rule Designer and open the Samples Console perspective.

__ 2. Open the **Samples Commands** tab.

__ 3. Expand **Samples Commands > Rule Execution Server > Running a hosted transparent decision service**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 4. Double-click **build**.

The build command calls an Axis Ant task to generate proxy classes from the WSDL description file. The classes are used by the custom classes that call the decision service. After the Java class files are compiled, you see a `BUILD SUCCESSFUL` message.

__ 5. Double-click the **run.axisclient** target.

The task creates a customer parameter and an order parameter, and calls the transparent decision service with the parameters.

The result in the Console view shows execution information from the response, including the reason for rejecting the order and the number of rules fired.

```
Completed: build
Executing: run.axisclient
Nb rules fired 4
Report reason = BlackListStocks
Completed: run.axisclient
BUILD SUCCESSFUL
```

2.4. Reviewing decision service performance

__ 1. Return to the Rule Execution Server console and click the **Explorer** tab.

__ 2. In the Navigator pane, click **Service Information**.

The Transparent Decision Service Information View is displayed.

__ 3. Click **Refresh**.

__ 4. Click the **HTDS/PreTradeChecksRuleApp/1.0/PreTradeChecks/1.0** decision service in the Transparent Decision Service Information table.

__ 5. Click **View Statistics** to see the performance statistics for this decision service.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3. Testing the decision service with REST

In this section, you test the decision service with REST.

__ 1. In the Rule Execution Server console, go to the Ruleset view for the **`PreTradeChecks`** ruleset.

__ 2. Retrieve the OpenAPI description of your HTDS.

__ a. Click **Retrieve HTDS Description File**.

__ b. For **Service protocol type**, choose **REST**.

__ c. For the **Format**, choose **OpenAPI - JSON**.

__ d. Select **Decision trace information** and click **Test**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. In the **Execution Request** field, set the input values for these parameters:

- customerParameter
- orderParameter
- TraceFilter

__ a. Set the **customer** and **order** input with these values:

```
"customerParameter": {
    "name": "John Smith",
    "owner": "George Smith@IBM",
    "initialValue": 2000,
    "currentValue": 800,
    "sectorPosition": 800,
    "bankStockPosition": 800,
    "oilStockPosition": 800,
    "railStockPosition": 800,
    "bondPosition": 800,
    "futurPosition": 800,
    "stockPosition": 800,
    "currency": "Euro"
  },
"orderParameter": {
    "id": "string",
    "stock":
    "Company X",
    "amount": 90,
    "exchange": "Nouveau Marche",
    "currency": "Euro",
    "destination": "France",
    "action": "Buy",
    "securityType": "Stock",
    "dateTime": "2004-02-14T19:44:14.000+0000",
    "sector": "Oil",
    "status": "Pending",
    "reasons": "BadClient"
  },
  "__TraceFilter__": {
    "infoRulesetProperties": false,
    "infoOutputString": false,
    "infoInputParameters": false,
    "infoOutputParameters": false,
    "none": true,
    "infoExecutionEventsAsked": false,
    "workingMemoryFilter": "string",
    "infoBoundObjectByRule": false,
    "infoExecutionDuration": false,
    "infoExecutionDate": false,
    "infoExecutionEvents": false,
    "infoInetAddress": false,
```

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

```
"infoRules": false,
"infoRulesNotFired": false,
"infoSystemProperties": false,
"infoTasks": false,
"infoTasksNotExecuted": false,
"infoTotalRulesFired": true,
"infoTotalRulesNotFired": false,
"infoTotalTasksExecuted": false,
"infoTotalTasksNotExecuted": false,
"infoWorkingMemory": false,
"infoRulesFired": false,
"infoTasksExecuted": false,
"infoBoundObjectSerializationType": "ClassName"
```

Information

You can copy and paste the code snippet from the `<LabfilesDir>\code\htds.txt` file into the **Execution Request** field.

__ 4. Click **Execute Request**.

__ 5. In the Server Response section, the response should include these values:

```
{
"__decisionTrace__": {
"totalRulesFired": 4
},
"__DecisionID__": "string",
"reportParameter": {
"id": "string",
"stock": "Company X",
"amount": 90,
"exchange": "Nouveau Marche",
"currency": "EURO",
"destination": "France",
"action": "BUY",
"securityType": "STOCK",
"dateTime": "2004-02-14T19:44:14.000+0000",
"sector": "OIL",
"status": "BLOCKED",
"reasons": "BLACK_LIST_STOCKS"  }}
```

The trace shows that 4 rules were run. The order status was blocked because of `BlackListStocks`.

__ 6. Close the **REST Service** browser tab.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 4. Retrieving an OpenAPI description and calling the service in a generated client

In this section, you retrieve an OpenAPI description of your HTDS, and use it to generate a Java client.

4.1. Retrieving the YAML file

__ 1. In Rule Execution Server console, return to the Ruleset view for the `PreTradeChecks` ruleset.

__ 2. Retrieve the OpenAPI description of your HTDS.

__ a. Click **Retrieve HTDS Description File**.

__ b. For **Service protocol type**, choose **REST**.

__ c. For the **Format**, choose **OpenAPI - YAML**.

__ d. Select **Decision trace information**.

__ e. Click **Download** and save the file.

By default, the file is saved to the **Downloads** folder.

__ 3. Go to the **Downloads** folder and rename the file as: `pretradecheck.yaml`

__ 4. Use your downloaded file to replace the `pretradecheck.yaml` file in the `<InstallDir>\executionserver\samples\decisionservice\data` directory.

4.2. Generating the Java client

__ 1. Go to Rule Designer and open the Samples Console perspective.

__ 2. Open the **Samples Commands** view and expand **Samples Commands > Rule Execution Server > Running a hosted transparent decision service**.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. Double-click **generate.openapi.client**.

__ 4. In Windows Explorer, go to the `<InstallDir>\executionserver\samples\decisionservice` directory to see the newly generated `open_api_client` folder.

Information

When you generate the OpenAPI client, it also copies the `DefaultApiTest.java` class in the `<InstallDir>\executionserver\samples\decisionservice\open_api_client\src\test\java\io\swagger\client\api` directory.

4.3. Adding a test to the Java client

You add a JUnit test to the generated client.

__ 1. Return to the Samples Console perspective in Rule Designer, and in the **Samples Commands > Rule Execution Server > Running a hosted transparent decision service**, double-click **add.openapi.unit.test**.

This command copies the `SimpleApiTest` Java class to the `<InstallDir>\executionserver\samples\decisionservice\open_api_client\src\test\java\io\swagger\client\api` directory.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

4.4. Running the Java client test

__ 1. Open a command prompt and navigate to the `open_api_client` folder by typing this command:

```
cd C:\IBM\ODM810\executionserver\samples\decisionservice\open_api_client
```

__ 2. Run the test with Maven.

```
mvn clean test
```

You get an output similar to the output from running `io.swagger.client.api.DefaultApiTest`

```
    reportParameter: class Order {
        id: an_id
        stock: Company X
        amount: 90
        exchange: New Market
        currency: Euro
        destination: France
        action: Buy
        securityType: Stock
        dateTime: 2004-02-15T00:44:14.000Z
        sector: Oil
        status: Blocked
        reasons: BlackListStocks
    }
}
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 2.109 sec

Results :

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 01:10 min
[INFO] Finished at: 2018-06-27T19:10:55-04:00
[INFO] ------------------------------------------------------------------------

C:\IBM\ODM892\executionserver\samples\decisionservice\open_api_client>_
```

__ 3. Close the command prompt.

End of exercise

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

This exercise looked at how you can use hosted transparent decision services by retrieving, examining, and by using the HTDS files.

© Copyright IBM Corp. 2019

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

IBM Training

© Copyright International Business Machines Corporation 2019.