OpenRules®

Open Source Business Rules and Decision Management System

Release 7.0.1

Getting Started

OpenRules, Inc.

www.openrules.com

May-2019

Table of Contents

INTRODUCTION

OpenRules® is an open source Business Rules and Decision Management System developed by OpenRules, Inc. in 2003. It is oriented to subject matter experts (business analysts) allowing them to represent, test, and maintain their business logic using MS Excel or Google Docs. Today OpenRules® is a winner of several software awards and is used worldwide by multi-billion corporations, major banks, insurers, health care providers, government agencies, online stores, universities, and many other institutions.

OpenRules® offers the following decision-making components:

- Rule Repository for management of enterprise-level decision rules
- Rule Engine for rules execution
- Rule Dialog for building rules-based Web questionnaires.
- Rule Solver for solving constraint satisfaction and optimization problems

Additional components are described here. Integration of these components with executable decisions has effectively made OpenRules® a reliable and easy to use product oriented to "decision-centric" application development.

This document helps a user to get started with OpenRules®. It describes how to install OpenRules® and use it starting with simple examples. This document is aimed at business analysts and software developers who may assist them in the development of decision-making applications. More information is available in the User Manual.

EVALUATION AND INSTALLATION

You may analyze and execute sample decision projects without any downloads from here. To start working with OpenRules® locally, you need to download and unzip only one folder "**openrules.models**" from http://openrules.com/download.htm. You may start with a simple decision model "Hello" (described below) by looking at different Excel files in the folder "Hello/rules/". To execute this model from a Windows Explorer, double-click on "run.bat". To be able to execute OpenRules, you need to make sure that you have free Java Platform (JDK) installed - see the Installation Guide.

INTRODUCTORY EXAMPLE

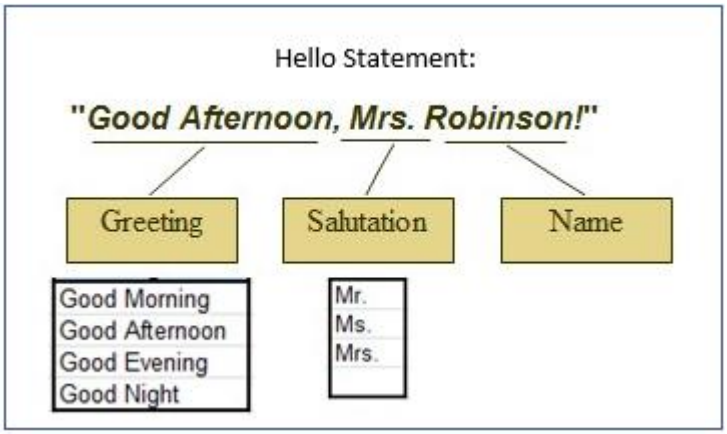
This section will demonstrate how to build simple executable decision projects with underlying decision tables using only OpenRules® and Excel. We will take a very simple example (similar to a traditional "Hello World!") and will explain how to implement it as an executable decision project.

Decision Model "Hello"

We will start with a very simple decision model.

Problem Description

The following example demonstrates how to develop a very simple decision model that should decide how to greet a customer during different times of the day. The proper decision might be a part of interactive voice response (IVR) system. For example, if a customer Robinson is a married woman and local time is 14:25, we want our decision to produce a hello statement like "*Good Afternoon, Mrs. Robinson!*" The following figure shows different variations of possible hello statements:



We may assume that "Name" and other related information about the customer is already known. Our decision model should represent business decision logic that determines the following decision variables called "Goals":

1. Goal "**Greeting**" with possible values "Good Morning", "Good Afternoon", etc.
2. Goal "**Salutation**" with possible values "Mr.", "Ms." And "Mrs."
3. Goal "**Hello Statement**" that combines Greeting, Salutation and Customer's Name in one hello-statement.

Specify Business Goals and Sub-Goals

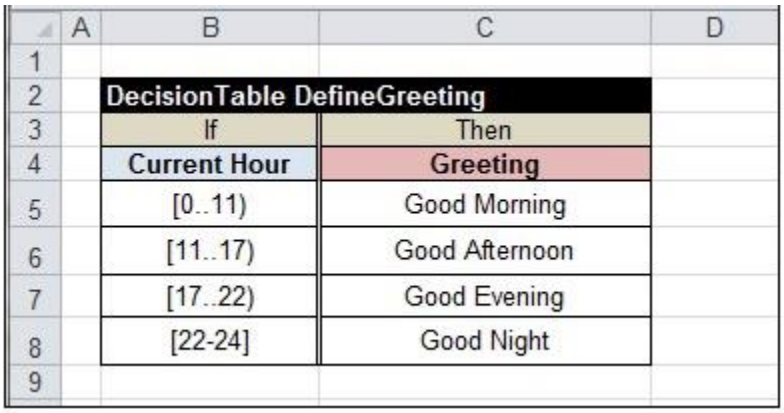
Our "ultimate" goal is to define "Hello Statement", but it requires to define two sub-goals: "Greeting" and "Salutation":

- **Hello Statement**
  - **Greeting**
  - **Salutation**

We will use decision tables to define these goals.

Decision Tables

Here is our first decision table:



	A	B	C	D
1				
2		DecisionTable DefineGreeting		
3		If	Then	
4		Current Hour	Greeting	
5		[0..11)	Good Morning	
6		[11..17)	Good Afternoon	
7		[17..22)	Good Evening	
8		[22-24]	Good Night	
9				

Hopefully, this decision table is self-explanatory. There are 4 rules:

1. If Current Hour is between 0 and 11, the Greeting is "Good Morning"
2. If Current Hour is between 11 and 17, the Greeting is "Good Afternoon".
3. If Current Hour is between 17 and 122, the Greeting is "Good Evening".
4. If Current Hour is between 22 and 24, the Greeting is "Good Night".

The first column is a condition specified by the OpenRules keyword "**If**", and the second column is a conclusion specified by the keyword "**Then**". If Current Hour is exactly 11, the second rule will be satisfied because the first interval [0..11) does not include 11 while the second interval [11..17) does. Instead of [0..11) you also may write:

< 11

Less than 11

More or equal 0 and less than 11

It is important to notice that all columns in the very first "black" row were merged in Excel. Why was that necessary? To help OpenRules to recognize the table bounds. If you do not merge the columns B and C in the Excel's row 2, OpenRules would "think" that this table has only one column (B).

It is important to surround a decision table with empty cells. If we place several decision tables inside the same Excel worksheet, these tables should not touch each other. Even if you want to put some text comments near some rows or columns of the decision table, make sure they do not touch the table. You may insert Excel's own comments anywhere – OpenRules will ignore them, but it will help other people who will analyze your decision table.

Text and background colors, fonts, borders, and other visual elements are used just for consistency and standardization. For example, many OpenRules customers have for years preferred to use a black background and a white foreground in the very first "signature" row. However, some customers prefer other coloring conventions. Contrary to "merging", it does not carry any semantic meaning for OpenRules. Coloring conditions in blue and conclusions in purple and separating them by a doubled line comes from the DMN standard samples.

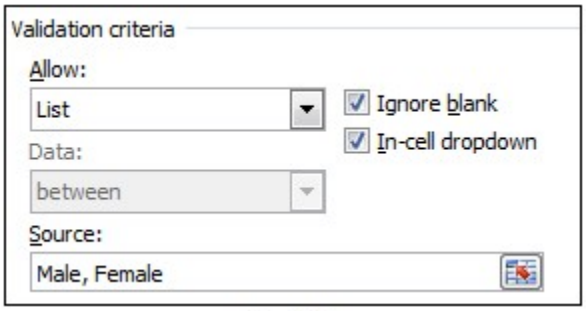
Here is a similar decision table "DefineSalutation":

DecisionTable DefineSalutation		
If	If	Then
Gender	**Marital Status**	**Salutation**
Male		Mr.
Female	Married	Mrs.
Female	Single	Ms.

It uses customer's attributes "Gender" and "Marital Status" as input decision variables to define the goal "Salutation". Note that in the first rule the cell for Marital Status left empty because whether Marital Status is "Married" or "Single" the Male's salutation is always "Mr." You also may put a hyphen to stress that this value is not relevant.

Instead of retyping the same values like Male or Female you may take advantage of

Excel's ranges. For example, you may select 3 cells with values Male, Female, Female in the first column and then click on the menu-item Data+Data Validation and fill out the displayed Excel's dialog:



Now you will be able to choose a value from the dropdown list. It makes sense to similarly define dropdown lists for the Marital Status, Salutation and Greeting columns.

OpenRules allows you to use a different design of the decision table "Define Salutation":

DecisionTable DefineSalutation					
Condition		Condition		Conclusion	
Gender		**Marital Status**		**Salutation**	
Is	Male			Is	Mr.
Is	Female	Is	Married	Is	Mrs.
Is	Female	Is	Single	Is	Ms.

It contains special sub-columns for operators "Is" to each table column. It also uses different keywords: "**Condition**" instead of "If" and "**Conclusion**" instead of "Then". Note that these keywords are placed in the merged cells now. While this decision table design may look a little bit more complex, it makes the table more readable when it utilizes different operators like "Is" or "=", "Is Not", "Less", "More or Equal", "Include", "Exclude", and many others.

You can mix columns of the types "Condition", "If", "Conclusion", and "Then" based on your preferences in each particular case. You can use the keyword "Action" as a synonym for "Then".

Now we will define a decision table for our goal-variable "Hello Statement":

DecisionTable DefineHelloStatement	
Conclusion	
Hello Statement	
Is	Greeting + ", " + Salutation + " " + Name + "!"

This decision table contains only one "Conclusion" and no conditions. It will assign the result of the concatenation of several strings to the decision variable "Hello Statement". If Greeting is "Good Afternoon", Salutation is "Mrs." and Name is "Robinson", then this formula will produce: Good Afternoon, Mrs. Robinson!

In the DMN standard's terms this formula is an example of a so-called FEEL expression, where FEEL stands for "Friendly Enough Expression Language".

To finalize our decision model, we also need to define a glossary and test cases.

Glossary

A glossary contains a list of all decision variables (both input and output) categorized by the business concepts they belong to. In our case it is quite simple because we use only the following decision variables:

- **Current Hour**
- **Greeting**
- **Gender**
- **Marital Status**
- **Salutation**
- **Hello Statement**
- **Name**

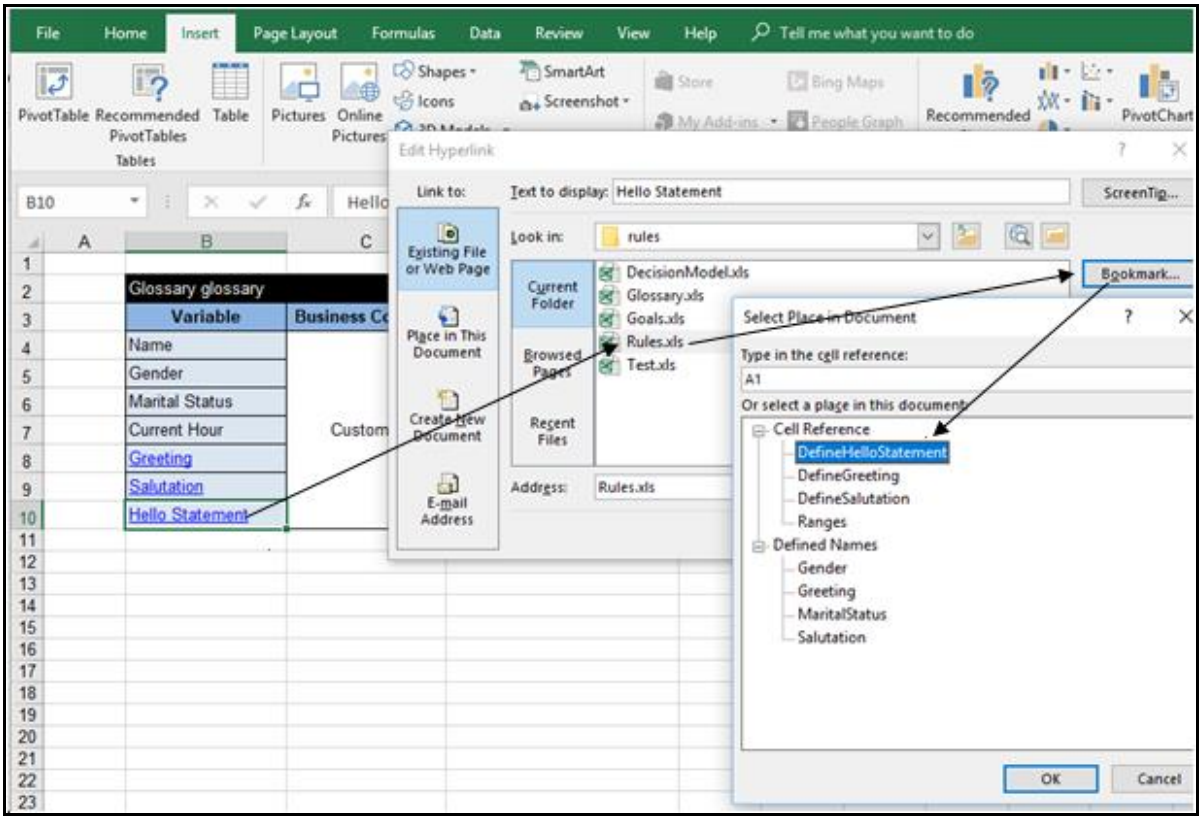
We may assume that in this simple case all decision variables belong to a business concept that we may call "Customer". Here is our table of the type "Glossary":

Glossary glossary		
Variable	**Business Concept**	**Attribute**
Name	Customer	name
Gender		gender
Marital Status		maritalStatus
Current Hour		currentHour
Greeting		greeting
Salutation		salutation
Hello Statement		helloStatement

The first column contains decision variables copied & pasted from already defined decision tables. All these 7 decision variables belong to the business concept "Customer", so all 7 rows are merged in the second column. OpenRules also requires the third column to provide technical names (called "attributes") for every decision variable. The attribute names cannot contain spaces as they will be used in the future to integrate our decision model with an actual IT system. The attribute names start with a small letter and then use capital letters for all consecutive words following so-called "Camel" naming convention. The reason is simple: in real-world applications, these attributes will correspond to the underlying Java objects.

Connecting Glossary with Decision Tables

Real-world decision models usually include a glossary surrounded by many decision tables which implement business logic for various goals and Subgoals. You usually keep a decision table in the file "Glossary.xls" and decision tables in the file "Rules.xls" or in multiple files distributed between different directories. To better manage large decision models, we recommend utilizing **Excel hyperlinks** to connect all decision tables to the glossary. Usually, you need to specify hyperlinks only for decision variable inside glossary that represent goals. For example, in our case we may insert hyperlink as in the following figure:



This way the Glossary really becomes a focal point of any decision model, from which you may access all related files and tables inside them.

Test Cases

OpenRules offers you a quite powerful, yet simple mechanism to create test cases directly in Excel tables. First, we define a so-called Datatype table for our business concept Customer:

Datatype Customer	
String	name
String	gender
String	maritalStatus
int	currentHour
String	greeting
String	salutation
String	helloStatement

This table starts with the keyword “Datatype” followed by the name of the datatype after space – in this case, “Datatype Customer”. Then we list different attributes. The

first column contains attribute types, e.g. "String" for text variables, "int" for integer variables, "double" for real variables, "Date" for dates. The second column contains the names of attributes exactly as they were defined in our Glossary.

Note. Capitalization like in "String" is important here. If you write "string" instead of "String" or "Int" instead of "int", OpenRules will point you to a syntax error "Unknown type" (indicating the exact cell where this error occurred).

Now we will define test-customers using the following table of the type "Data":

Data Customer customers						
name	gender	maritalStatus	currentHou	greeting	salutation	helloStatement
Name	**Gender**	**Marital Status**	**Current Hour**	**Greeting**	**Salutation**	**Hello Statement**
Robinson	Female	Married	20	?	?	?
Green	Male	Single	11	?	?	?
Kaye	Female	Single	22	?	?	?

This table starts with the keyword "Data" following the datatype and the name of the array of customers – in this case, "Data Customer customers". The second row contains the technical names of Customer's attributes, and the third column contains the business names. The business names are usually the names of variables, but you can use any name as the technical names already provide a mapping to the decision variables from the glossary. Then we may define as many rows as we want to specify customers.

OpenRules also allows you to define the expected results for each test case. Then it automatically checks if the actual results correspond to the expected results and shows mismatches. Here is the proper table of the type "DecisionTableTest":

DecisionTableTest testCases			
#	ActionUseObject	ActionExpect	ActionExpect
Test ID	**Customer**	**Greeting**	**Salutation**
Test 1	:= customers[0]	Good Evening	Mrs.
Test 2	:= customers[1]	Good Afternoon	Mr.
Test 3	:= customers[2]	Good Night	Ms.

The name "testCases" is the name used by the default test launcher. Each test has its own ID defined in the first column of the type "#". This table can use different business

objects (concepts) defined in the column of the type "ActionUseObject" – in this case, we have only one object "Customer". We also can add different expected test results using the columns of the type "ActionExpect" and the corresponding variable names (like Greeting and Salutation). For example, the first test-customer defined by

:= customers[0]

is expected to produce Greeting "Good Afternoon" and Salutation "Mrs."

Note. The use of ":=" may look confusing for a beginner. The real reason is that this is an example of a Java snippet that OpenRules allows you to put in any cell of any OpenRules table. The sign ":=" indicates that this is a Java snippet and the index [0] indicates that you want to use the very first element of the array "customers" defined in the above table "customers".

Executing Decision Model

To run these test-cases we will use the standard OpenRules configuration that provides two batch-files "**built.bat**" and "**run.bat**". First, you need to double-click on the file "build.bat" to prepare our model for execution, and then you can double-click on the file "run.bat" to actually execute it. Here is the execution protocol:

```
RUN TEST: Test 1
  Assign: Greeting = Good Evening
  Conclusion: Salutation Is Mrs.
  Assign: Hello Statement = Good Evening, Mrs. Robinson!
Validating results for the test <Test 1>
Test 1 was successful
RUN TEST: Test 2
  Assign: Greeting = Good Afternoon
  Conclusion: Salutation Is Mr. [Mr.]
  Assign: Hello Statement = Greeting + ", " + Salutation + Name + "!"
[Good Afternoon, Mr. Green!]
Validating results for the test <Test 2>
Test 2 was successful
RUN TEST: Test 3
  Assign: Greeting = Good Night [Good Night]
  Conclusion: Salutation Is Ms. [Ms.]
  Assign: Hello Statement = Greeting + ", " + Salutation + Name + "!"
[Good Night, Ms. Kaye!]
Validating results for the test <Test 3>
Test 3 was successful

All 3 tests succeeded!
```

This protocol shows the results for each Test Run and the execution steps. As you can see, first OpenRules assigned "Good Evening" to the variable Greeting. It's done by our decision table "DefineGreeting". Then we can see the Conclusion that "Salutation Is Mrs". It was made by our decision table "DefineSalutation". And finally, our formula in the table "DefineHelloStatement" assigned "Good Evening, Mrs. Robinson!" to the decision variable "Hello Statement". OpenRules validated that all produced correspond to those that were expected by out test cases otherwise, it will show mismatches.

Decision Explanations

To explain why certain rules were executed. OpenRules produces an HTML report, one for every test-case. Here is a report for Test 1:

Decision Table : Rule#	Executed Rule	Variables and Values
DefineGreeting:3	**IF** Current Hour = [17..22) **THEN** Greeting = Good Evening	Current Hour=20 Greeting=Good Evening
DefineSalutation:2	**IF** Gender Is Female **AND** Marital Status Is Married **THEN** Salutation Is Mrs.	Gender=Female Marital Status=Married Salutation=Mrs.
DefineHelloStatement:1	Hello Statement Is Greeting + ", " + Salutation + " " + Name + "!"	Name=Robinson Greeting=Good Evening Salutation=Mrs. Hello Statement=Good Evening, Mrs. Robinson!

It shows only actually executed decision tables and rules in the order they were executed. It also explains why these rules were executed by showing the values of the involved decision variables in the moment of the execution.

Let's summarize the major implementation steps.

Implementation Steps

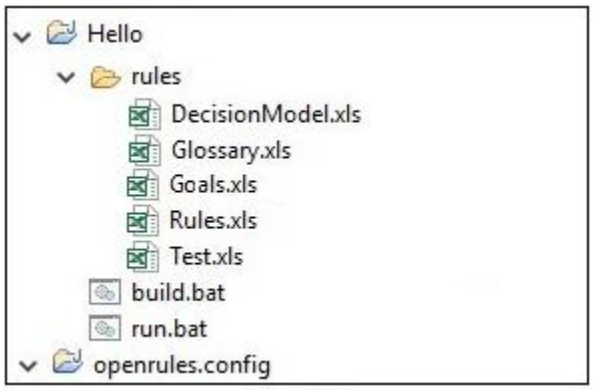
1. **Specify Goals and Sub-goals**
2. **Create Decision Tables** that implement these goals/sub-goals
3. **Create Glossary**
4. **Create Test Cases**
5. **Build the model**
6. **Executed Test Cases**.

These six steps represent a quite generic implementation schema for much more complex decision models as well. OpenRules also calls this approach "Goal-Oriented Decision Modeling".

Rules Repository

Let's look at how our decision model is organized. Different decisioning constructs such as decision tables, glossary, and tests can be placed in one of multiple Excel files maintained in different folders. We call it "Rules Repository".

By default, when you install OpenRules you will find many decision models in the workspace "openrules.models". Here is a view of the file structure for our decision model "Hello":



Everything related to our decision model is placed in the folder "Hello" – sometimes we call it "decision project". You also can find the folder "**openrules.config**" that comes with the standard OpenRules installation and contains all template files and libraries provided by OpenRules.

Our rules repository is the folder "rules" inside "Hello". You can see that now it contains several xls-files. The file "Rules.xls" contains our decision tables, the file "Glossary.xls" contains our glossary, and the file "Test.xls" contains out test cases. There are also two more files "DecisionModels.xls" and "Goals.xls" you should know about.

The Environment Table

The file "DecisionModels.xls" is considered as an entry point to a decision model. It usually contains the top-level description of our decision model (optional) and one special table of the type table "**Environment**" that informs OpenRules about the structure of our decision model. The following Environment table describes our decision model:

Environment	
include	Rules.xls
	Glossary.xls
	../../openrules.config/DecisionTemplates.xls

This table refers to all files included into our decision model relatively to this file. The files "Rules.xls" and "Glossary.xls" are at the same level as "DecisionModel.xls", while the standard file "openrules.config/DecisionTemplates.xls" is located two levels higher than "DecisionModels.xls" and thus is preceded by "../../".

Note that our test file "Test.xls" is not included in this table. The reason is that the same decision model including this Environment table should be used not only for testing but for the actual production environment. So, it shouldn't depend on our test files.

There is also file "Goals.xls" that was automatically generated when we double-clicked on "build.bat".

Execution Path

The file "**Goals.xls**" contains an automatically generated an execution path that leads to the goal "Hello Statement".

Decision DecisionHelloStatement
ActionExecute
Decision Tables
DefineGreeting
DefineSalutation
DefineHelloStatement

The execution path is a special table of the type "Decision" that specifies an execution sequence of all decision tables. This path is used by OpenRules when we execute the decision model against different data sets. When we double-clicked on our "build.bat", it generated the file "Goals.xls".

The name of this table "DecisionHelloStatement" is composed by concatenation of the word "Decision" and the goal's name "Hello Statement" from which all spaces (and dashes if any) were removed. This decision will execute our 3 decision tables "DefineGreeting", "DefineSalutation" and "DefineHelloStatement" in the top-down order

described in this table.

While OpenRules provides a standard batch-file "**built.bat**", for each decision model you need to modify it to define the model-specific goal (keep in mind the same decision model may have different goals), and its input and output files. Here is the file "build.bat" adjusted to our example:

```
set GOAL="Hello Statement"
set INPUT_FILE_NAME=rules/DecisionModel.xls
set OUTPUT_FILE_NAME=rules/Goals.xls
cd %~dp0
call ..\openrules.config\projectBuild
```

When you double-click on "build.bat", OpenRules analyzes the decision model whose Environment table is defined in the INPUT FILE, and automatically defines an execution path for the specified GOAL and saves it in the OUTPUT FILE.

Note. You may create the file "Goals.xls" and the table "DecisionHelloStatement" in it manually. For some, very complex decision models it might be the way to go. However, a nice thing about auto-generation of the execution path is the fact that you don't have to worry about the execution order of all your decision tables – and real-world decision models may contain hundreds of them. You don't even have to look inside this file. Every time when you modify your decision model, e.g. adding new goals, you may run "build.bat" to re-generate this file.

The standard file "**run.bat**" file that executes our test cases from the file "Test.xls" also should be adjusted for each decision model:

```
set DECISION_NAME=DecisionHelloStatement
set FILE_NAME=rules/Test.xls
cd %~dp0
call ..\openrules.config\projectRun
```

Here we need to set only two parameters:

- DECISION_NAME that points to the table of the type "Decision" that contains our execution path
- FILE_NAME that contains our test data.

Tools We Used

To build and test a decision model we used only MS Excel and Windows Explorer! Where is actually OpenRules? OpenRules was installed in the folder "openrules.config" and contains some Java libraries and templates in the form of Excel files. Please note that we use MS Excel only as a table editor – you do not need it to run your decision model. Instead of Excel, you also may use Google Sheets.

Accepting Data from Java

In the real world, decisions are incorporated in business applications that supply the decisions with data and expect to receive back data-specific decisions. Above we have shown how to define data types and data instances in Excel tables. The standard OpenRules® installation workspace "openrules.models" includes the proper decision project "HelloJava" that explains how the same decision model can be used with data defined in Java objects.

Other Decision Examples

The standard OpenRules® installation includes many other simple and more complex examples in the folder "openrules.models". You may start analyzing, modifying, and executing these projects. More tutorials can be found at section Documentation. The User Manual covers the core OpenRules® concepts in greater depth. Additional OpenRules® components are described in separate user manuals: see Rule Learner, Rule Solver, and Rule Dialog.

Technical Support

Direct all your technical questions to support@openrules.com or to this Discussion Group. Read more at http://openrules.com/services.htm.