**Maven Plugin Guide**

OpenL Tablets BRMS

Release 5.18

[](http://creativecommons.org/licenses/by/3.0/us/)

*OpenL Tablets Documentation is licensed under a* [*Creative Commons Attribution 3.0 United States License*](http://creativecommons.org/licenses/by/3.0/us/)*.*

Table of Contents

[1 Preface 4](#_Toc433898900)

[1.1 Related Information 4](#_Toc433898901)

[1.2 Typographic Conventions 4](#_Toc433898902)

[2 Introduction 5](#_Toc433898903)

[2.1 Goals Overview 5](#_Toc433898904)

[2.2 Usage Overview 5](#_Toc433898905)

[3 Goals 7](#_Toc433898906)

[3.1 openl:generate 7](#_Toc433898907)

[Required Parameters 7](#_Toc433898908)

[Optional Parameters 9](#_Toc433898909)

[Parameter Details 10](#_Toc433898910)

[3.2 openl:compile 12](#_Toc433898911)

[Optional Parameters 12](#_Toc433898912)

[Parameter Details 13](#_Toc433898913)

[3.3 openl:test 13](#_Toc433898914)

[Optional Parameters 13](#_Toc433898915)

[Parameter Details 13](#_Toc433898916)

[3.4 openl:help 14](#_Toc433898917)

[Optional Parameters 14](#_Toc433898918)

[Parameter Details 15](#_Toc433898919)

[4 Usage 16](#_Toc433898920)

[4.1 Directory Structure 16](#_Toc433898921)

[4.2 Configure Interface, Domain Classes and Project Descriptor Generation 16](#_Toc433898922)

[4.3 Configure OpenL Project Compilation and Validation 17](#_Toc433898923)

[4.4 Configure OpenL Project Testing 17](#_Toc433898924)

[5 Examples 20](#_Toc433898925)

[5.1 Configuration with all OpenL Maven Plugin Goals 20](#_Toc433898926)

[5.2 Creating a Project with a Working Example of OpenL Maven Plugin Usage 21](#_Toc433898927)

# Preface

This preface is an introduction to the *OpenL Tablets Maven Plugin Guide.*

The following topics are included in this preface:

* [Related Information](#_Related_Information)
* [Typographic Conventions](#_Typographic_Conventions)

## Related Information

The following table lists sources of information related to contents of this guide:

| Related information | |
| --- | --- |
| Title | Description |
| <http://openl-tablets.org/> | OpenL Tablets home page. |

## Typographic Conventions

The following styles and conventions are used in this guide:

| Typographic styles and conventions | |
| --- | --- |
| Convention | Description |
| **Bold** | * Represents user interface items such as check boxes, command buttons, dialog boxes, drop-down list values, field names, menu commands, menus, option buttons, perspectives, tabs, tooltip labels, tree elements, views, and windows. * Represents keys, such as **F9** or **CTRL+A.** * Represents a term the first time it is defined. |
| Courier | Represents file and directory names, code, system messages, and command-line commands. |
| Courier Bold | Represents emphasized text in code. |
| Select **File > Save As** | Represents a command to perform, such as opening the **File** menu and selecting **Save As.** |
| *Italic* | * Represents any information to be entered in a field. * Represents documentation titles. |
| < > | Represents placeholder values to be substituted with user specific values. |
| Hyperlink | Represents a hyperlink. Clicking a hyperlink displays the information topic or external source. |
| [name of guide] | Reference to another guide that contains additional information on a specific feature. |

# Introduction

Access to rules and data in Excel tables is realized through OpenL Tablets API. OpenL Tablets provides wrappers to developers to facilitate easier usage.

This plugin is used to generate interface to access the rules, to validate rules during compilation phase, and to run OpenL Tablets tests.

The following topics are included in this section:

* [Goals Overview](#_Goals_Overview)
* [Usage Overview](#_Usage)

To acquire a better understanding of plugin name usage, see the following examples:

* [Configuration with all OpenL Maven Plugin Goals](#_Configuration_with_all)
* [Creating a Project with a Working Example of OpenL Maven Plugin Usage](#_Creating_a_Project)

## Goals Overview

General information about the goals is as follows:

| General information about the goals | |
| --- | --- |
| Link to the goal section | Description |
| [openl:generate](#_openl:generate) | Generates OpenL Tablets interface, domain classes, and project descriptor. |
| [openl:compile](#_openl:compile) | Compiles the OpenL Tablets project. |
| [openl:test](#_openl:test) | Runs OpenL Tablets tests. |
| [openl:help](#_openl:help) | Displays help information on openl-maven-plugin. |

## Usage Overview

General instructions on how to use the plugin name can be found in [Usage](#_Usage_1). This section provides specific configuration example.

Specify the version in the project plugin configuration as follows:

<project>

...

<build>

<!-- To define the plugin version in your parent POM -->

<pluginManagement>

<plugins>

<plugin>

<groupId>org.openl.rules</groupId>

<artifactId>openl-maven-plugin</artifactId>

<version>${openl.version}</version>

</plugin>

...

</plugins>

</pluginManagement>

<!-- To use the plugin goals in your POM or parent POM -->

<plugins>

<plugin>

<groupId>org.openl.rules</groupId>

<artifactId>openl-maven-plugin</artifactId>

<version>${openl.version}</version>

</plugin>

...

</plugins>

</build>

...

</project>

# Goals

This section includes the following topics:

* [openl:generate](#_openl:generate)
* [openl:compile](#_openl:compile)
* [openl:test](#_openl:test)
* [openl:help](#_openl:help)

## openl:generate

**Full name:**

org.openl.rules:openl-maven-plugin:5.X.X:generate

**Description:**

Generates OpenL Tablets interface, domain classes, project descriptor, and unit tests.

**Attributes:**

Requires a Maven project to be executed.

By default, binds to the **generate-sources** lifecycle phase.

### Required Parameters

| Required parameters for openl:generate | | | |
| --- | --- | --- | --- |
| Name | Type | Since | Description |
| generateInterfaces | JavaAntTask[] | - | Tasks that will generate classes.  **Object Properties**   |  |  |  |  | | --- | --- | --- | --- | | Name | Type | Required | Description | | srcFile | String | true | Reference to the Excel file for which an interface class must be generated. | | targetClass | String | true | Full name of the interface class to be generated.  OpenL Tablets WebStudio recognizes modules in projects by interface classes and uses their names in UI.  If there are multiple wrappers with identical names, only one of them is recognized as a module in OpenL Tablets WebStudio. | | displayName | String | false | End user oriented title of the file that appears in OpenL Tablets WebStudio.  **Default value**: the Excel file name without extension. | | targetSrcDir | String | false | Folder where the generated interface class must be placed. An example is src/main/java. **Default value:** ${project.build.sourceDirectory}. | | openlName | String | false | OpenL configuration to be used.  For OpenL Tablets, the org.openl.xls value must always be used.  **Default value**: org.openl.xls. | | userHome | String | false | Location of user-defined resources relative to the current OpenL Tablets project.  **Default value:**  **.**  The dot stands for the current folder. | | userClassPath | String | false | Reference to the folder with additional compiled classes imported by the module when the interface is generated.  **Default value**: null. | | ignoreTestMethods | boolean | false | Parameter which denotes, if set to true, that test methods must not be added to the interface class. It is used only in JavaInterfaceAntTask.  **Default value**: true. | | generateUnitTests | boolean | false | Parameter that overwrites the base generateUnitTests value. | | unitTestTemplatePath | String | false | Parameter that overwrites the base unitTestTemplatePath value. | | overwriteUnitTests | boolean | false | Parameter that overwrites the base overwriteUnitTests value. | |

### Optional Parameters

| Optional parameters for openl:generate | | | |
| --- | --- | --- | --- |
| Name | Type | Since | Description |
| classpaths | String[] | - | Default classpath entries in rules.xml.  **Default value**:  **.**  The dot stands for the current folder.  It is used only if createProjectDescriptor == true. |
| createProjectDescriptor | boolean | - | Parameter which denotes, if set to true, that rules.xml will be generated if it does not exist.  **Default value:** true. |
| generateUnitTests | Boolean | - | Parameter which denotes, if set to true, that JUnit tests for OpenL Tablets Test tables will be generated.  **Default value:** false. |
| openlOutputDirectory | String | - | Folder used by OpenL to compile rules. An example is ${project.build.directory}/openl.  **Default value:** ${project.build.directory}/openl. |
| openlResourcesDirectory | String | - | Folder that contains all OpenL-related resources, such as OpenL Tablets rules and project descriptor. An example is ${project.basedir}/src/main/openl.  **Default value:** ${project.basedir}/src/main/openl. |
| overwriteProjectDescriptor | boolean | - | Parameter which denotes, if set to true, that rules.xml will be overwritten on each run. Makes sense only if createProjectDescriptor == true. **Default value:** true. |
| overwriteUnitTests | Boolean |  | Parameter which denotes, if set to true, that existing JUnit tests will be overwritten. If set to false, only absent tests will be generated, and others will be skipped.  **Default value:** false. |
| projectName | String | - | Default project name in rules.xml. If omitted, the name of the first module in the project is used. The parameter is used only if createProjectDescriptor == true. |
| unitTestTemplatePath | String | - | Path to the Velocity template for generated unit tests. If omitted, default template is used. It is available in the following template variables:   |  |  | | --- | --- | | Name | Description | | openlInterfacePackage | Package of generated interface class. | | openlInterfaceClass | Generated interface class name. | | testMethodNames | Available test method names. | | projectRoot | Root directory of OpenL project. | | srcFile | Reference to the Excel file for which an interface class must be generated. | | StringUtils | Apache commons utility class. |   **Default value:** org/openl/rules/maven/JUnitTestTemplate.vm. |

### Parameter Details

**classpaths:**

Default classpath entries in rules.xml. The default value is an array containing one string with a dot . where the dot stands for the current folder. It is used only if createProjectDescriptor == true.

* **Type:** java.lang.String[]
* **Required:** No

**createProjectDescriptor:**

Parameter which denotes, if set to true, that rules.xml will be generated if it does not exist. The default value is true.

* **Type:** boolean
* **Required:** No
* **Default:** true

**generateInterfaces:**

Tasks that generates classes.

**Object Properties**

| Object properties | | | |
| --- | --- | --- | --- |
| Name | Type | Required | Description |
| srcFile | String | true | Reference to the Excel file for which an interface class must be generated. |
| targetClass | String | true | Full name of the interface class to be generated. OpenL Tablets WebStudio recognizes modules in projects by interface classes and uses their names in the user interface. If there are multiple wrappers with identical names, only one of them is recognized as a module in OpenL Tablets WebStudio. |
| displayName | String | false | End user oriented title of the file that appears in OpenL Tablets WebStudio.  **Default value:** the Excel file name without extension. |
| targetSrcDir | String | false | Folder where the generated interface class must be placed. An example is src/main/java.  **Default value:** ${project.build.sourceDirectory}. |
| openlName | String | false | OpenL configuration to be used. For OpenL Tablets, the org.openl.xls value must always be used.  **Default value:** org.openl.xls. |
| userHome | String | false | Location of user-defined resources relative to the current OpenL Tablets project.  **Default value:**  .  The dot stands for the current folder. |
| userClassPath | String | false | Reference to the folder with additional compiled classes imported by the module when the interface is generated.  **Default value:** null. |
| ignoreTestMethods | boolean | false | Parameter which denotes, if set to true, that test methods will not be added to interface class. It is used only in JavaInterfaceAntTask. **Default value:** true. |
| generateUnitTests | boolean | false | Parameter that overwrites the base generateUnitTests value. |
| unitTestTemplatePath | String | false | Parameter that overwrites the base unitTestTemplatePath value. |
| overwriteUnitTests | boolean | false | Parameter that overwrites the base overwriteUnitTests value. |

* **Type**: org.openl.conf.ant.JavaAntTask[]
* **Required**: Yes

**generateUnitTests:**

If set to true, JUnit tests for OpenL Tablets Test tables will be generated. The default value is false.

* **Type**: java.lang.Boolean
* **Required**: No
* **Default**: false

**openlOutputDirectory:**

Folder used by OpenL to compile rules. An example is ${project.build.directory}/openl.

* **Type**: java.lang.String
* **Required**: No
* **Default**: ${project.build.directory}/openl

**openlResourcesDirectory:**

Folder that contains all OpenL Tablets-related resources, such as OpenL Tablets rules and project descriptor. An example is ${project.basedir}/src/main/openl.

* **Type**: java.lang.String
* **Required**: No
* **Default**: ${project.basedir}/src/main/openl

**overwriteProjectDescriptor:**

If set to true, rules.xml will be overwritten on each run. If set to false, rules.xml generation will be skipped if it exists. Makes sense only if createProjectDescriptor == true. The default value is "true".

* **Type**: boolean
* **Required**: No
* **Default**: true

**overwriteUnitTests:**

If set to true, existing JUnit tests will be overwritten. If set to false, only absent tests will be generated, and others will be skipped.

* **Type**: java.lang.Boolean
* **Required**: No
* **Default**: false

**projectName:**

Default project name in rules.xml. If omitted, the name of the first module in the project is used. The parameter is used only if createProjectDescriptor == true.

* **Type**: java.lang.String
* **Required**: No

**unitTestTemplatePath:**

Path to Velocity template for generated unit tests. If omitted, a default template will be used. The parameter is available in template variables:

|  |  |
| --- | --- |
| Variables | |
| Name | Description |
| openlInterfacePackage | Package of generated interface class. |
| openlInterfaceClass | Generated interface class name. |
| testMethodNames | Available test method names. |
| projectRoot | Root directory of OpenL project. |
| srcFile | Reference to the Excel file for which an interface class must be generated. |
| StringUtils | Apache commons utility class. |

* **Type:** java.lang.String
* **Required:** No
* **Default**: org/openl/rules/maven/JUnitTestTemplate.vm

## openl:compile

**Full name:**

org.openl.rules:openl-maven-plugin:5.13.0:compile

**Description:**

Compiles and validates OpenL Tablets project.

**Attributes:**

* Requires a Maven project to be executed.
* Binds by default to the **compile** lifecycle phase.

### Optional Parameters

| Optional parameters for openl:compile | | | |
| --- | --- | --- | --- |
| Name | Type | Since | Description |
| openlOutputDirectory | String | - | Folder used by OpenL Tablets to compile rules. An example is ${project.build.directory}/openl.  **Default value:** ${project.build.directory}/openl. |
| openlResourcesDirectory | String | - | Folder that contains all OpenL Tablets-related resources, such as OpenL Tablets rules and project descriptor. An example is ${project.basedir}/src/main/openl.  **Default value:** ${project.basedir}/src/main/openl. |

### Parameter Details

**openlOutputDirectory:**

Folder used by OpenL to compile rules. An example is ${project.build.directory}/openl.

* **Type**: java.lang.String
* **Required**: No
* **Default**: ${project.build.directory}/openl

**openlResourcesDirectory:**

Folder that contains all OpenL Tablets-related resources, such as OpenL Tablets rules and project descriptor. An example is ${project.basedir}/src/main/openl.

* **Type**: java.lang.String
* **Required**: No
* **Default**: ${project.basedir}/src/main/openl

## openl:test

**Full name:**

org.openl.rules:openl-maven-plugin:5.13.0:test

**Description:**

Runs OpenL tests.

**Attributes:**

* Requires a Maven project to be executed.
* Binds by default to the **test** lifecycle phase.

### Optional Parameters

| Optional parameters for openl:test | | | |
| --- | --- | --- | --- |
| Name | Type | Since | Description |
| openlOutputDirectory | String | - | Folder used by OpenL Tablets to compile rules. An example is ${project.build.directory}/openl".  **Default value:** ${project.build.directory}/openl. |
| openlResourcesDirectory | String | - | Folder that contains all OpenL Tablets-related resources, such as OpenL Tablets rules and project descriptor. An example is "${project.basedir}/src/main/openl".  **Default value:** ${project.basedir}/src/main/openl. |
| skipTests | boolean |  | Parameter which denotes, if set to true, to skip running OpenL Tablets tests.  **User property:** skipTests. |

### Parameter Details

**openlOutputDirectory:**

Folder used by OpenL Tablets to compile rules. An example is ${project.build.directory}/openl.

* **Type**: java.lang.String
* **Required**: No
* **Default**: ${project.build.directory}/openl

**openlResourcesDirectory:**

Folder that contains all OpenL Tablets-related resources, such as OpenL Tablets rules and project descriptor. An example is ${project.basedir}/src/main/openl.

* **Type**: java.lang.String
* **Required**: No
* **Default**: ${project.basedir}/src/main/openl

**skipTests:**

Parameter which denotes, if set to true, to skip running OpenL Tablets tests.

* **Type**: boolean
* **Required**: No
* **User Propert**y: skipTests

## openl:help

**Full name:**

org.openl.rules:openl-maven-plugin:5.13.0:help

**Description:**

Displays help information on openl-maven-plugin.

Calls mvn openl:help -Ddetail=true -Dgoal=<goal-name> to display parameter details.

**Attributes:**

* The goal is thread-safe and supports parallel builds.

### Optional Parameters

| Optional parameters for openl:help | | | |
| --- | --- | --- | --- |
| Name | Type | Since | Description |
| detail | boolean | - | Parameter which denotes, if set to true, to display all settable properties for each goal.  **Default value:** false.  **User property:** detail. |
| goal | String | - | Name of the goal for which to display help. If left unspecified, all goals are displayed.  **User property:** goal. |
| indentSize | int | - | Number of spaces per indentation level. The value must be positive.  **Default value:** 2.  **User property:** indentSize. |
| lineLength | int | - | Maximum length of a display line. The value must be positive.  **Default value:** 80.  **User property:** lineLength. |

### Parameter Details

**detail:**

Parameter which denotes, if set to true, to display all settable properties for each goal.

* **Type:** boolean
* **Required:** No
* **User Property:** detail
* **Default:** false

**goal:**

The name of the goal for which help must be displayed. If left unspecified, all goals are displayed.

* **Type:** java.lang.String
* **Required:** No
* **User Property:** goal

**indentSize:**

The number of spaces per indentation level. The value must be positive.

* **Type:** int
* **Required:** No
* **User Property:** indentSize
* **Default:** 2

**lineLength:**

The maximum length of a display line. The value must be positive.

* **Type:** int
* **Required:** No
* **User Property:** lineLength
* **Default:** 80

# Usage

This section includes the following topics:

* [Directory Structure](#_Directory_Structure)
* [Configure Interface, Domain Classes and Project Descriptor Generation](#_Configure_Interface,_Domain)
* [Configure OpenL Project Compilation and Validation](#_Configure_OpenL_Project)
* [Configure OpenL Project Testing](#_Configure_OpenL_Project_1)

## Directory Structure

OpenL Maven Plugin expects the following directory structure:

|- your-project/ Project root folder

| |- pom.xml Maven project file

| |

| |- src/

| |

| | |- main/

| | |

| | | |- java/ Contains Java sources

| | | |

| | | |- resources/ Contains Java resources

| | | |

| | | |- openl/ Contains all OpenL Tablets-related resources (rules, xml)

| | | |

| | | |

| | | | |- rules.xml OpenL Tablets project descriptor (for OpenL Tablets only)

| | | | |- rules/

| | | | | |- TemplateRules.xls File with rules

Note that OpenL Tablets-related resources are located in the src/main/openl directory. It can be changed to fit user needs by modifying the openlResourcesDirectory parameter in Maven plugin configuration.

**Note:** It is not recommended to put OpenL Tablets-related resources to the src/main/resources folder. In this case, OpenL Tablets resources will be inside the JAR file alongside with the compiled Java classes, which most probably is not what was expected to do in production.

## Configure Interface, Domain Classes and Project Descriptor Generation

The simplest way to generate interface for rules is defined in the TemplateRules.xls file as follows:

<build>

[...]

<plugins>

[...]

<plugin>

<groupId>org.openl.rules</groupId>

<artifactId>openl-maven-plugin</artifactId>

<version>${openl.rules.version}</version>

<configuration>

<generateInterfaces>

<generateInterface>

<srcFile>src/main/openl/rules/TemplateRules.xls</srcFile>

<targetClass>org.company.gen.TemplateRulesInterface</targetClass>

</generateInterface>

</generateInterfaces>

</configuration>

<executions>

<execution>

<goals>

<goal>generate</goal>

</goals>

</execution>

</executions>

</plugin>

</plugins>

[...]

</build>

In this case, classes and rules.xml are generated on each Maven run during the **generate-sources** phase.

To invoke class generation manually, remove the executions node and run in the console when needed:

mvn openl:generate

For more information on configuration options, see [openl:generate](#_openl:generate).

## Configure OpenL Project Compilation and Validation

<build>

[...]

<plugins>

[...]

<plugin>

<groupId>org.openl.rules</groupId>

<artifactId>openl-maven-plugin</artifactId>

<version>${openl.rules.version}</version>

<executions>

<execution>

<goals>

<goal>compile</goal>

</goals>

</execution>

</executions>

</plugin>

</plugins>

[...]

</build>

For more information on configuration options, see [openl:compile](#_openl:compile).

## Configure OpenL Project Testing

The simplest way to invoke the OpenL Tablets test is as follows:

<build>

[...]

<plugins>

[...]

<plugin>

<groupId>org.openl.rules</groupId>

<artifactId>openl-maven-plugin</artifactId>

<version>${openl.rules.version}</version>

<executions>

<execution>

<goals>

<goal>test</goal>

</goals>

</execution>

</executions>

</plugin>

</plugins>

[...]

</build>

For more information on configuration options, see [openl:test](#_openl:test).

To gain more control over tests and invoke and debug them from the Java code, generate JUnit tests. Do not use the test goal. Instead, configure the generate goal as follows:

<build>

[...]

<plugins>

[...]

<plugin>

<groupId>org.openl.rules</groupId>

<artifactId>openl-maven-plugin</artifactId>

<version>${openl.rules.version}</version>

<configuration>

<generateUnitTests>true</generateUnitTests>

<generateInterfaces>

[...]

</generateInterfaces>

</configuration>

<executions>

<execution>

<goals>

<goal>generate</goal>

</goals>

</execution>

</executions>

</plugin>

</plugins>

[...]

</build>

To define a user custom template for JUnits tests, set the unitTestTemplatePath parameter with a path to the user's custom Velocity template. For example, consider <unitTestTemplatePath>src/test/resources/MyTemplate.vm</unitTestTemplatePath>. An example of such template is as follows:

#if ($openlInterfacePackage)

package $openlInterfacePackage;

#end

import org.junit.Before;

import org.junit.Test;

import org.openl.rules.runtime.RulesEngineFactory;

import org.openl.rules.testmethod.TestUnitsResults;

import java.io.File;

import static org.junit.Assert.assertTrue;

#set( $openlInterfaceClassWithTests = "${openlInterfaceClass}WithTests" )

public class ${openlInterfaceClass}Test {

private static interface $openlInterfaceClassWithTests extends $openlInterfaceClass {

#foreach( $testMethodName in $testMethodNames )

TestUnitsResults $testMethodName();

#end

}

private $openlInterfaceClassWithTests instance;

@Before

public void setUp() throws Exception {

File xlsFile = new File("$projectRoot", "$srcFile");

instance = new RulesEngineFactory<$openlInterfaceClassWithTests>(

xlsFile,

${openlInterfaceClassWithTests}.class

).newEngineInstance();

}

#foreach( $testMethodName in $testMethodNames )

@Test

public void test$StringUtils.capitalize($testMethodName)() throws Exception {

TestUnitsResults results = instance.$testMethodName();

assertTrue(results.toString(), results.getNumberOfFailures() == 0);

}

#end

}

# Examples

To acquire a better understanding of Plugin Name usage, see the following examples:

* [Configuration with all OpenL Maven Plugin Goals](#_Configuration_with_all)
* [Creating a Project with a Working Example of OpenL Maven Plugin Usage](#_Creating_a_Project)

## Configuration with all OpenL Maven Plugin Goals

To configure rules.xml generation, set its project ID, project name, and classpath values. An example is as follows:

<build>

[...]

<plugins>

[...]

<plugin>

<groupId>org.openl.rules</groupId>

<artifactId>openl-maven-plugin</artifactId>

<version>${openl.rules.version}</version>

<configuration>

<!-- Project name. -->

<projectName>OpenL Rules Simple Project</projectName>

<!-- Project's classpath. -->

<classpaths>

<param>.</param>

</classpaths>

<!-- OpenL project includes one or more modules. -->

<generateInterfaces>

<generateInterface>

<displayName>Template Rules</displayName>

<targetClass>template.Wrapper</targetClass>

<!-- Rules root document. Usually excel file on file system. -->

<srcFile>src/main/openl/rules/TemplateRules.xls</srcFile>

</generateInterface>

</generateInterfaces>

</configuration>

<executions>

<execution>

<goals>

<goal>generate</goal>

<goal>compile</goal>

<goal>test</goal>

</goals>

</execution>

</executions>

</plugin>

</plugins>

[...]

</build>

For more information on configuration options, see [openl:generate](#_openl:generate).

## Creating a Project with a Working Example of OpenL Maven Plugin Usage

OpenL Tablets has an archetype which can be used to create a simple OpenL Rules project containing an example of OpenL Maven Plugin usage. Proceed as follows:

1. Execute the following command in the command line:

mvn archetype:generate

Maven runs an archetype console wizard.

1. Select the openl-simple-project-archetype menu item.
2. Follow the wizard instructions to complete project creation.

When the creation is completed, a new Maven-based project appears in the file system. It is an OpenL Tablets Rules project which has one module with simple rules.

1. To compile the project, in the command line, execute the following command from the root of the project folder:

mvn install

After that, the following objects can be found in the target folder:

* A ZIP file with "-deployable" suffix, for importing a project to WebStudio.

For more information, see ***[EIS\_Suite\_OpenL\_Tablets\_WebStudio\_(UG)].***

* A ZIP file with "-runnable" suffix that can be executed after unpacking.

It demonstrates how OpenL Tablets rules can be invoked from the Java code.

* A JAR file that contains only compiled Java classes.