

RA Generator for

OpenSCENARIO

[RA Generator for 1](#_Toc52884154)

[OpenSCENARIO 1](#_Toc52884155)

[1 Introduction 5](#_Toc52884156)

[1.1 Overview 5](#_Toc52884157)

[2 Delivery 6](#_Toc52884158)

[3 Installation 7](#_Toc52884159)

[3.1 Prerequisites 7](#_Toc52884160)

[3.1.1 Apache Maven installation 7](#_Toc52884161)

[3.2 Eclipse Installation 7](#_Toc52884162)

[3.2.1 Install the latest “Eclipse IDE for Java Developers” 7](#_Toc52884163)

[3.2.2 Install Groovy 7](#_Toc52884164)

[3.2.3 Install Template Editor Plugin 9](#_Toc52884165)

[4 Adding Projects to Eclipse 11](#_Toc52884166)

[4.1 Adding the generator project 11](#_Toc52884167)

[4.2 Adding the basic project 12](#_Toc52884168)

[As said, the basic project is not needed to run the generator, but if you want to view the groovy source code and explore the UML helper functions, you might also want to import this project into your workspace. 12](#_Toc52884169)

[5 Generating code 13](#_Toc52884170)

[5.1 The executable generator classes 13](#_Toc52884171)

[5.1.1 Input directoy 13](#_Toc52884172)

[5.1.2 Output directory 13](#_Toc52884173)

[5.1.3 Setting the arguments in eclipse 13](#_Toc52884174)

[5.2 Running the generator from command line 15](#_Toc52884175)

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Changes |
| 1.0 | 12/16/2020 | Andreas Hege | Initial Documentation |
|  |  |  |  |

Table : Change log

# Introduction

This documentation describes installation and usage of the RA Consulting Code Generator utilized for the OpenSCENARIO project. All rights reserved by RA Consulting.

## Overview

The RA Consulting code generator utilizes models to generate source code. Models are usually complex UML models, but also textual models and other proprietary models may be used to produce source code.

The generator uses templates to produce textual output. The results of a generator run is usually executable, compilable or interpretable source code for documentation or software development.

# Installation

The generator uses groovy and is developed and executed on the java platform. It is executable with [Oracle java11 OpenJDK](https://openjdk.java.net/install/)..

It is recommended to gain some knowledge in these technologies:

* The Java platform
* The groovy programming language.
* Maven build system
* Eclipse IDE.

## Prerequisites

Make sure that you have installed OpenJDK 11 when running the generator. For compilation Java8 is sufficient.

Download the OpenJDK 11 [here](https://openjdk.java.net/install/) and install it on your PC.

### Apache Maven installation

It is recommended to install Apache Maven. See here for [instructions](https://maven.apache.org/install.html).

## Eclipse Installation

Though eclipse it is not necessary to build and to run the generator, it is recommended to install eclipse. The IDE supports template editing and integrates well when generating code for java projects.

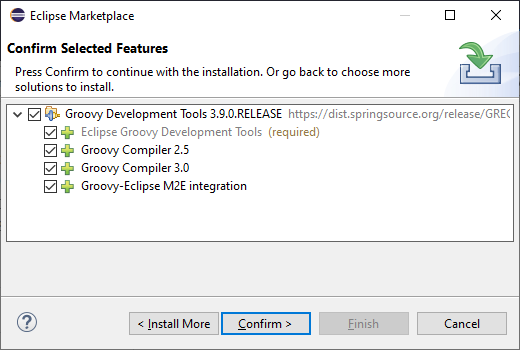
Installing eclipse includes three steps

* Install the latest “Eclipse IDE for Java Developers”
* Install latest groovy form the Eclipse Marketplace
* Install template editor plugin.

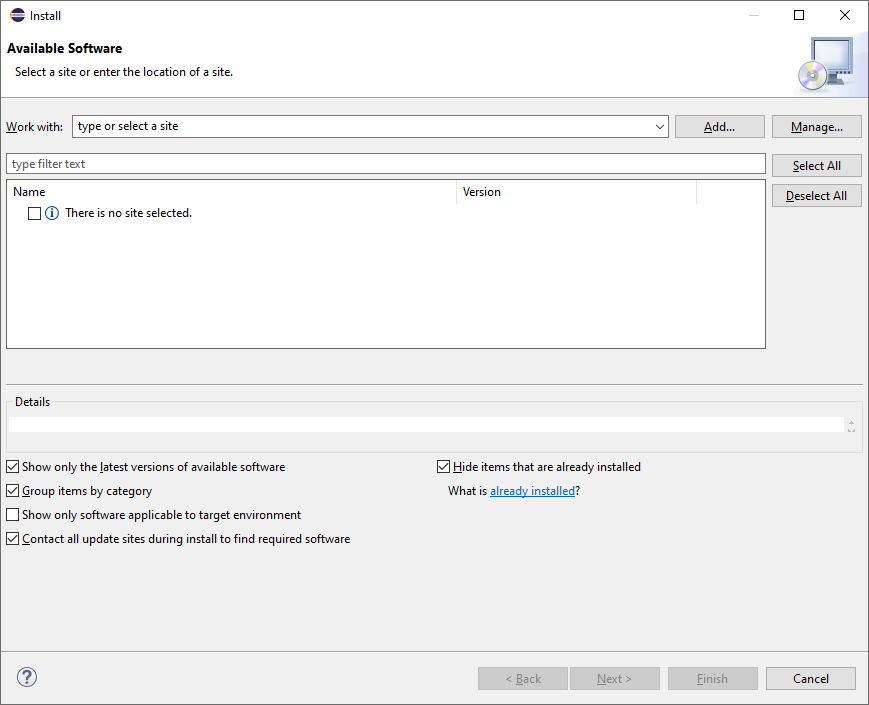
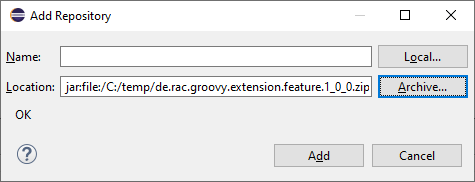
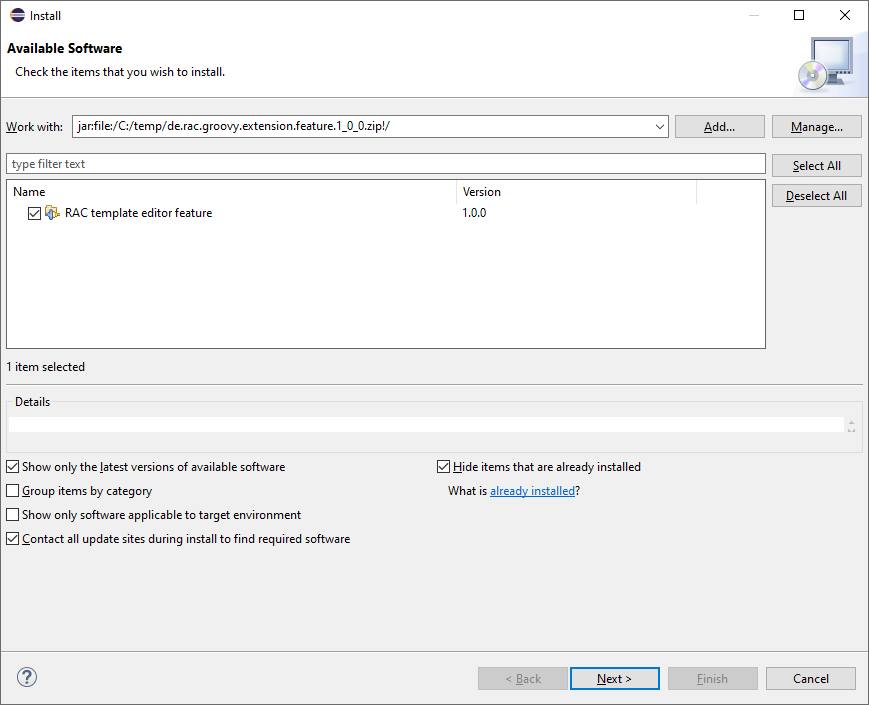
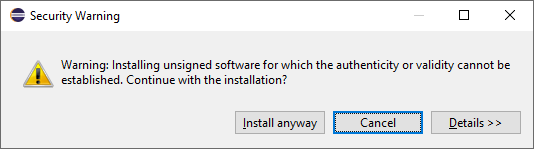
### Install the latest “Eclipse IDE for Java Developers”

Follow the instructions [here](https://www.eclipse.org/downloads/packages/installer). You might install everything with the default options (No eclipse projects included, default bootstrap). The only thing you might want to edit is the JRE.

### Install Groovy

* Open Eclipse. *Choose Help->Eclipse Marketplace*.
* Search for groovy
* Choose “Groovy Development Tools (...)” and press *Install*
* Add “Groovy-Eclipse M2E integration” and press *Confirm*  
  
* Restart Eclipse

### Install Template Editor Plugin

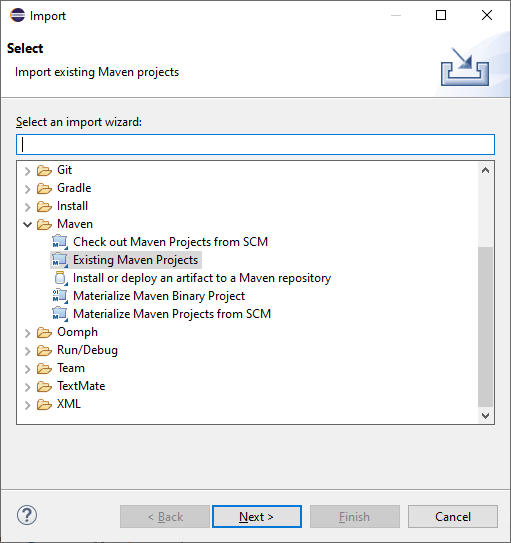
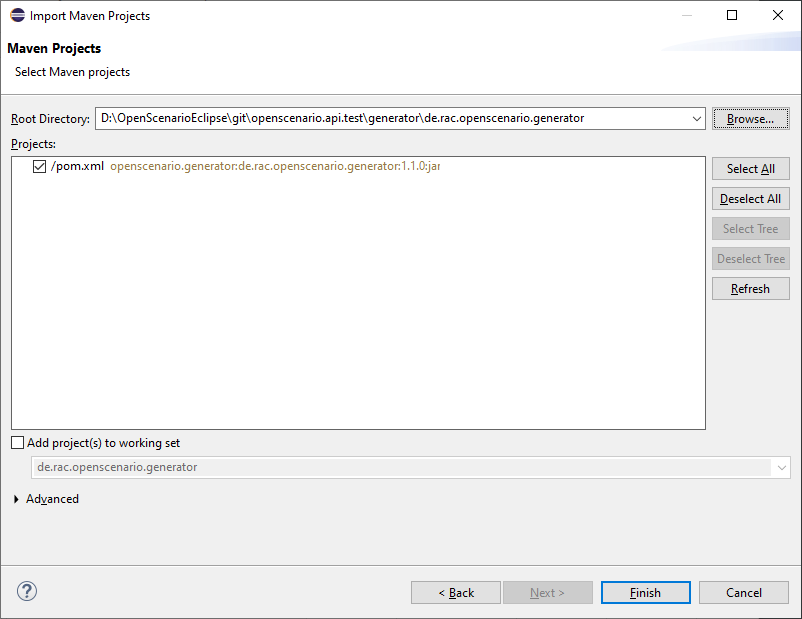
* Open Eclipse. *Choose Help->Install New Software*. Choose *Add...*  
  
* Choose *Archive...* and add *de.rac.groovy.extension.feature.1\_0\_0.zip* from your local hard drive.  
  
* Deselect *Group items by category* and select *RAC template editor feature*. Then choose *Next*
* Select *Install anyway* (if you trust RA Consulting)  
  
* Restart Eclipse as suggested.

# Adding Projects to Eclipse

Checkout the source code together with the OpenSCENARIO API

## Adding the generator project

So, let’s add the OpenSCENARIO generator to Eclipse.

* Open Eclipse with a new Workspace (e.g. c:\temp\test-workspace)
* Import the project in Eclipse with *File->Import->Maven->Existing Maven Projects* into Workspace.  
  
* Browse into the workspace folder and choose the suggested project  
  
* Try to build the project in Eclipse. Select pom.xml in the project, right click and choose *Run As->Maven install*.
* Alternatively with command line: Change into the project directory (de.rac.openscenario.generator-1.1.0) and execute *mvn install*
* Everything should work so far.

# Generating code

Now, as everything is installed and the project is added, let’s get familiar with running the generator.

## The executable generator classes

There are two generator classes that are executable. One for each supported platform:

* de.rac.openscenario.generator.cpp.GeneratorCpp.groovy
* de.rac.openscenario.generator.java.GeneratorJava.groovy

Start each of them by right-click on the class and choose *Run As.->Java Application*

It will give you the following output:

Argument must be <outputDir>

We have to hand over the putput dir to the generator:

* **Output Directory**: The directory for the generated code

The input comes as a part of the project and contains the following files:

### Input directory

The input directory is included in the project (src/main/resources/input:

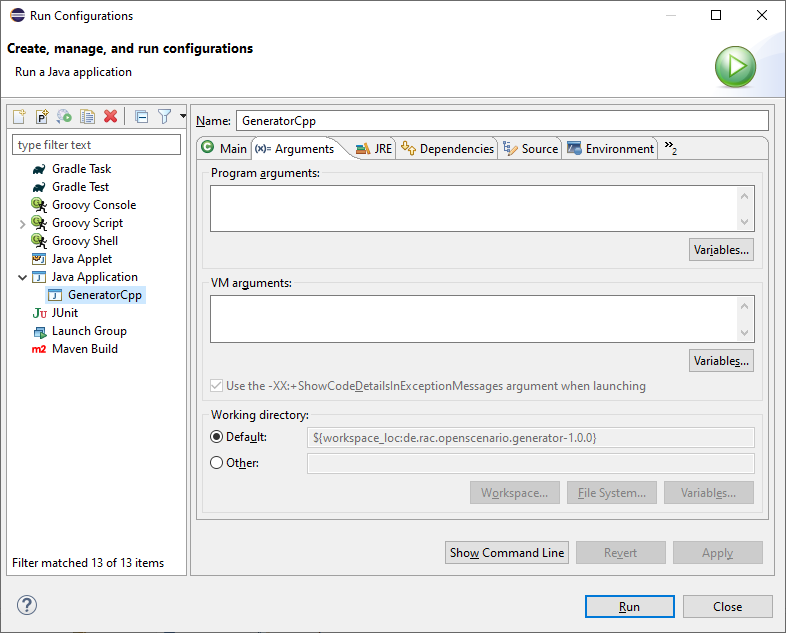
* **OpenSCENARIO\_Ea\_1.0.0.xmi**: The UML snapshot model from OpenSCENARIO 1.0.0 as an Enterprise Architect XMI 2.1 export.
* **RangeCheckerRules.json**: A structured file that describes the range constraints in a formal way.

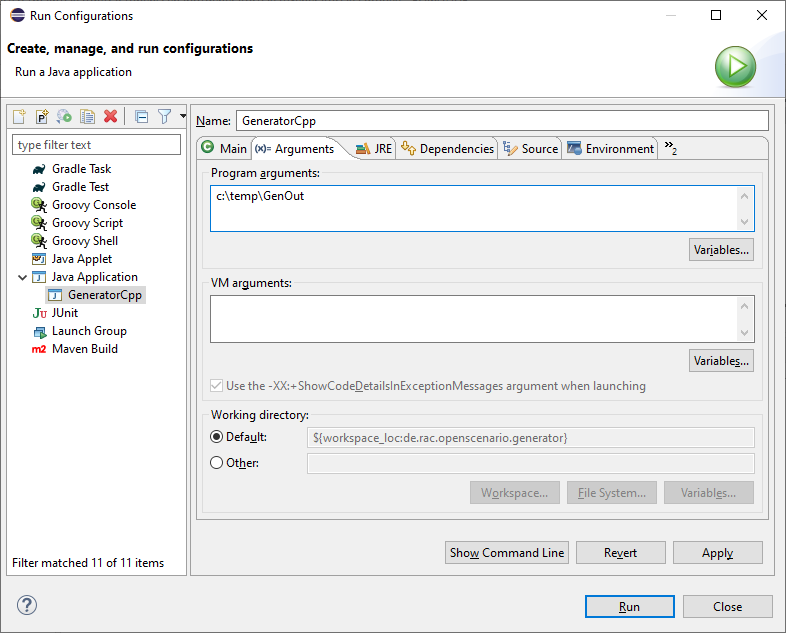
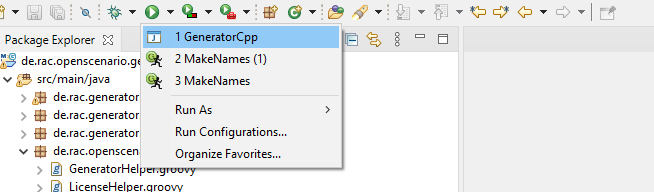
### Output directory

Be aware that everything in the output directory is deleted prior to a generator run. The content in the output directory should be completely reproducible by the generator.

### Setting the arguments in eclipse

To set input parameters in eclipse choose *Run->Run Configurations* from the main menu.



* Choose *GeneratorCpp* in the left window under *Java Applications*. Go to tab *(x)= Arguments.* Be aware that you first have to run the *GeneratorCpp* to create a run configuration (with *Run As.->Java Application)* before it appears in the list.
* Now add the output dir (e.g. c:\temp\GenOut)  
  
* Press *Run*. Your Generator should produce code.
* Your run configuration has been saved. You can run it either *via Run->Run Configuration* in the main menu or by selecting it in the list of the *Run* button in the toolbar.  
    
  

## Running the generator from command line

To run the generator from the command line we first have to package the results in a self-contained jar with any dependencies.

* Select the pom.xml right click and choose *Run As->Maven install* (if not done before)
* Now, in the *target* folder of the project a self-contained jar is created. de.rac.openscenario.generator-1.1.0-jar-with-dependencies.jar
* With your command interpreter: change into the *target* directory
* Add the self-contained package to the classpath.  
  set CLASSPATH=de.rac.openscenario.generator-1.1.0-jar-with-dependencies.jar
* Run the main class with java and the command line arguments  
  java de.rac.openscenario.generator.cpp.GeneratorCpp C:\temp\GenOut
* 