How to Create and Deploy a Scenario

***Task**:*

In the following we want to describe how a scenario is created and deployed in the JEngine.

Realization:

Step 1: Modeling

1.1 Get the Modeling Tool

First you have to get a modeling tool. In our project we decided to use the ProcessEditor, which you can check out on the following link: http://github.com/BP2014W1/processeditor.git

After you got it, we recommend using IntelliJ to compile and start it.

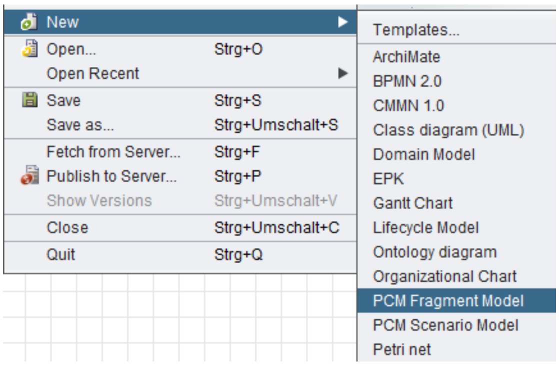
1.2 Start the Modeling Tool

For this to work you should have installed Ant on your system. After making sure you have Ant, you can run "ant clean init-ivy deps compile-wo-deps jar-workbench run-workbench.jar" or choose the corresponding methods in IntelliJ to run. Then the workbench should open and a PCM-Scenario can be modeled.

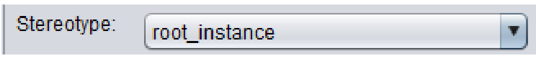
For troubleshooting, if something went wrong with the ProcessEditor, please check the corresponding documentation.

1.3 The Correct Way of Modeling PCM

First you should open a new PCM-Fragment or as many as you want to model and model them corresponding to the general modeling guidelines of the ProcessEditor. If you are finished with your Fragments you should publish them to the model versioning server.



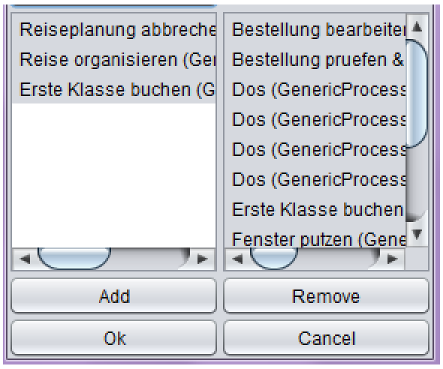
If you have finished publishing them it is time to create the domain model. For that open a new domain model. For every data object, there has to be a data class containing data attributes. Aggregations can be used to connect the data classes to each other. If you have a main data object, you can mark it as a root_instance. If you have finished the domain model, publish it too.



Stereotype: root_instance

Lastly we have to combine all the pieces to one Scenario. So create a new PCM-Scenario. Now you get two columns. Right-click on a node and choose the "*Add Fragments*"-option and choose the Fragments you created before and add them to the scenario. All the Fragments are now listed in the white column. The green column contains all data objects. You can right-click on a data object and open its properties to refer the data class.

After adding the termination condition and referencing the DomainModel, you can finally publish the scenario.



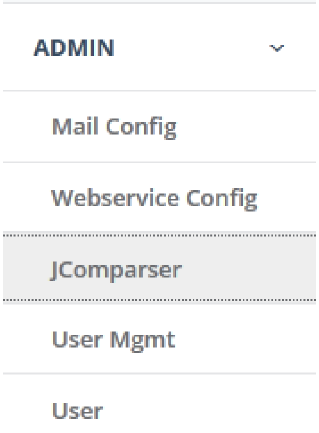
Step 2: Loading and Deploying the Scenario

There are two ways of using our Engine. Either you use our Frontend which is not open source or you write your own. In both ways the communication is held via REST.

The most important calls are explained below. Please check out our documentation for further information on the REST Interface.

2.1 Starting the JComparser

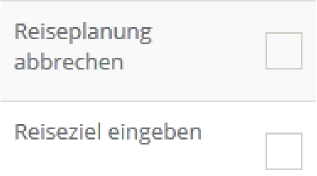
First of all, you need to communicate to our JComparser REST Interface. If you use our Frontend you just need to click on the JComparser tab. In that view, all Scenarios from the model-repository of the ProcessEditor are listed, so you can decide which to deploy. For that you have to just press "Launch" in the Frontend. Now your model should be saved into the database.



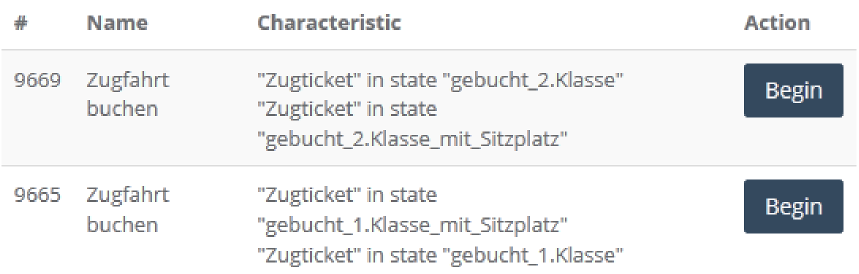
2.2 Create Instances of a Scenario and Executing Tasks

To start working with your Scenario you have to create Instances first. In the Frontend you just have to choose your Scenario and press the "Start Instance" button.

After that you can choose which name the new instance should have and ultimately create it. Now you get a list of enabled tasks from which you can freely choose which one to run.



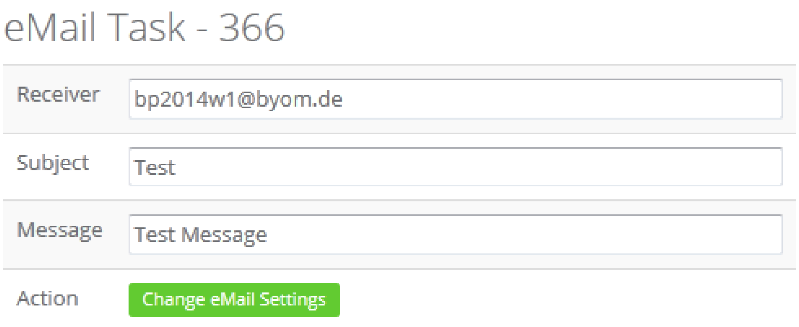
In case of referenced tasks with different output sets you have to decide which one you want before you can start it. When terminating the tasks you may have to choose which output set you want if there are multiple possibilities. It is represented in a similar way just that you have to press “Terminate” instead of “Begin”.



#	Name	Characteristic	Action
9669	Zugfahrt buchen	"Zugticket" in state "gebucht_2.Klasse" "Zugticket" in state "gebucht_2.Klasse_mit_Sitzplatz"	Begin
9665	Zugfahrt buchen	"Zugticket" in state "gebucht_1.Klasse_mit_Sitzplatz" "Zugticket" in state "gebucht_1.Klasse"	Begin

2.3 Configuration of E-Mail and Web-Service Tasks

You have to configure E-Mail and Web-Service tasks before you can execute them. To do that, you have to check the menu point “Admin” where you can choose either “Mail Config” or “Webservice Config”. For E-Mail tasks you first have to choose your corresponding scenario. Then you can freely specify the sender, subject and message. By clicking on the “Change eMail Settings” button you will save your configuration. In this form you can also make use of process variables.

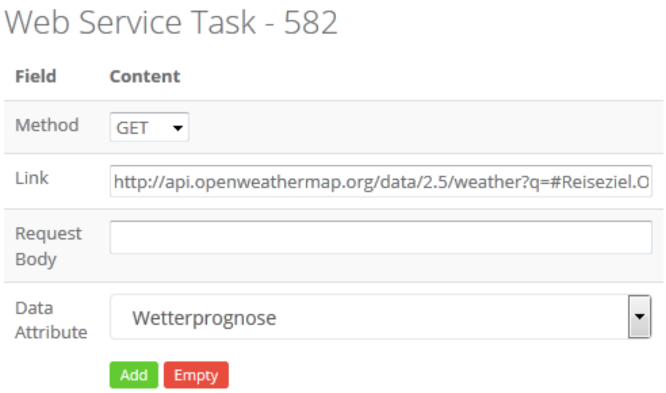


eMail Task - 366

Receiver	bp2014w1@byom.de
Subject	Test
Message	Test Message
Action	Change eMail Settings

For Web-Service tasks you have to choose the scenario as well. Here you can choose which type of request you want to send (GET/PUT/POST). In case you choose either PUT or POST you have to specify the request body as the REST standard demands.

You can save the response in a process variable (Data Attribute) by choosing the corresponding Data Attribute and pressing “Add”.



Web Service Task - 582

Field	Content
Method	GET ▾
Link	http://api.openweathermap.org/data/2.5/weather?q=#Reiseziel.O
Request Body	
Data Attribute	Wetterprognose ▾
	Add Empty

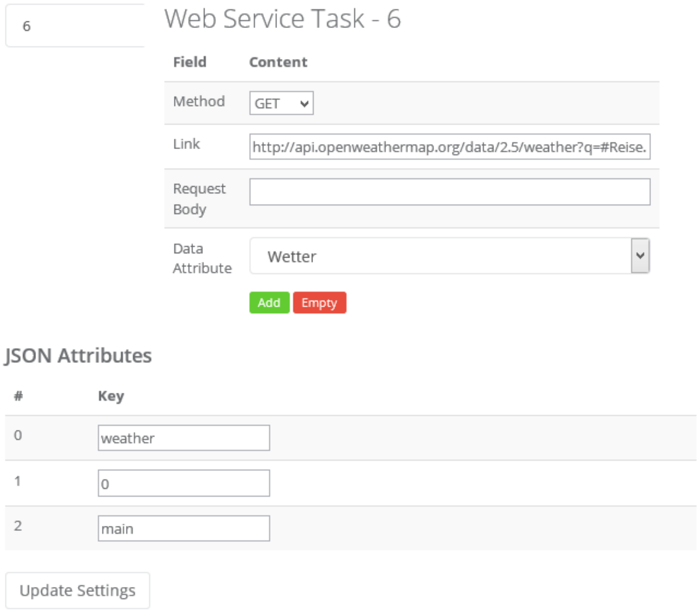
After that you have to specify the JSON parameters. These are used to parse the answer your Web Service task will get and identify the correct value you want to write into the variable. For that you will have to add at least one parameter.

The figure to the right shows how a specific value of the response of a GET-request is saved into a Data Attribute. The response that is returned by the specified link might look like this:

```
{"coord":{"lon":-0.13,"lat":51.51},
"weather":[{"id":800,"main":"Clear",
"description":"Sky is Clear",
"icon":"01d"}],"base":"stations"}
```

The value “Clear” from the response will be saved into the Data Attribute “Wetter”.

You can specify more parameters by pressing the “Add” button again. Ultimately, to save your changes, you have to click the “Update Settings” button.



6

Web Service Task - 6

Field	Content
Method	GET
Link	http://api.openweathermap.org/data/2.5/weather?q=#Reise.
Request Body	
Data Attribute	Wetter

JSON Attributes

#	Key
0	weather
1	0
2	main