







Model Engineering Lab	Assignment 4
188.923 IT/ME VU, WS 2018/19	
Deadline: Upload (ZIP) in TUWEL until Sunday, January 13 th , 2019, 23:55 Assignment Review: Wednesday, January 16 th , 2019	25 points

Code Generation

The goal of this assignment is to develop a code generator for the *RoverML language* using Xtend. The code generator generates HTML and JavaScript code to visualize and simulate a Rover System. This lab is organized in two goals: (A) Graphically display the rovers of a system and their components, and (B) to simulate the execution of rover programs.

Assignment Resources

	ME_WS18_Lab4_Resources.zip
	at.ac.tuwien.big.roverml (Modeling project for STL; solution of Assignment 1)
	at.ac.tuwien.big.roverml.codegen (Xtend project for developing the code generator)
	at.ac.tuwien.big.roverml.codegen.lib (Library used by the code generator)
	at.ac.tuwien.big.roverml.codegen.examples (Examples for the simulation)
	lab4.pdf (this document)

Setting up your workspace

Before starting this assignment, make sure that you have all necessary components installed in your Eclipse. A detailed installation guide can be found in the TUWEL course¹.

Import the projects provided in the assignment resources into your workspace. Therefore select *File → Import → General/Existing Projects into Workspace → Select archive file → Browse*. Choose the downloaded archive *ME_WS18_Lab4_Resources.zip* and import at least the *.roverml* and *.codegen* project into your workspace. The other two projects do not need to necessarily be imported. The *.lib* project provides the JavaScript library used for the simulator and the *.examples* projects contains only some examples. Please refer to the *.examples* project for the usage of the *.lib* project.

It is recommended that you read the complete assignment specification at least once. If there are any parts of the assignment specification or the provided resources that are ambiguous to you, don't hesitate to ask in the forum for clarification.

¹ Eclipse Setup Guide: <https://tuwel.tuwien.ac.at/mod/page/view.php?id=388945>

Part A: Visualizing the RoverSystem

In the first part of the assignment, the objective is to generate the HTML/JS code that creates the *RoverSystem simulator*, a simulation of a simple rover's program. To do so, you need to identify which parts of the code depend on the model, i.e. must be computed from the model, and which parts are independent of the model, i.e., static, and define your code generation templates accordingly.

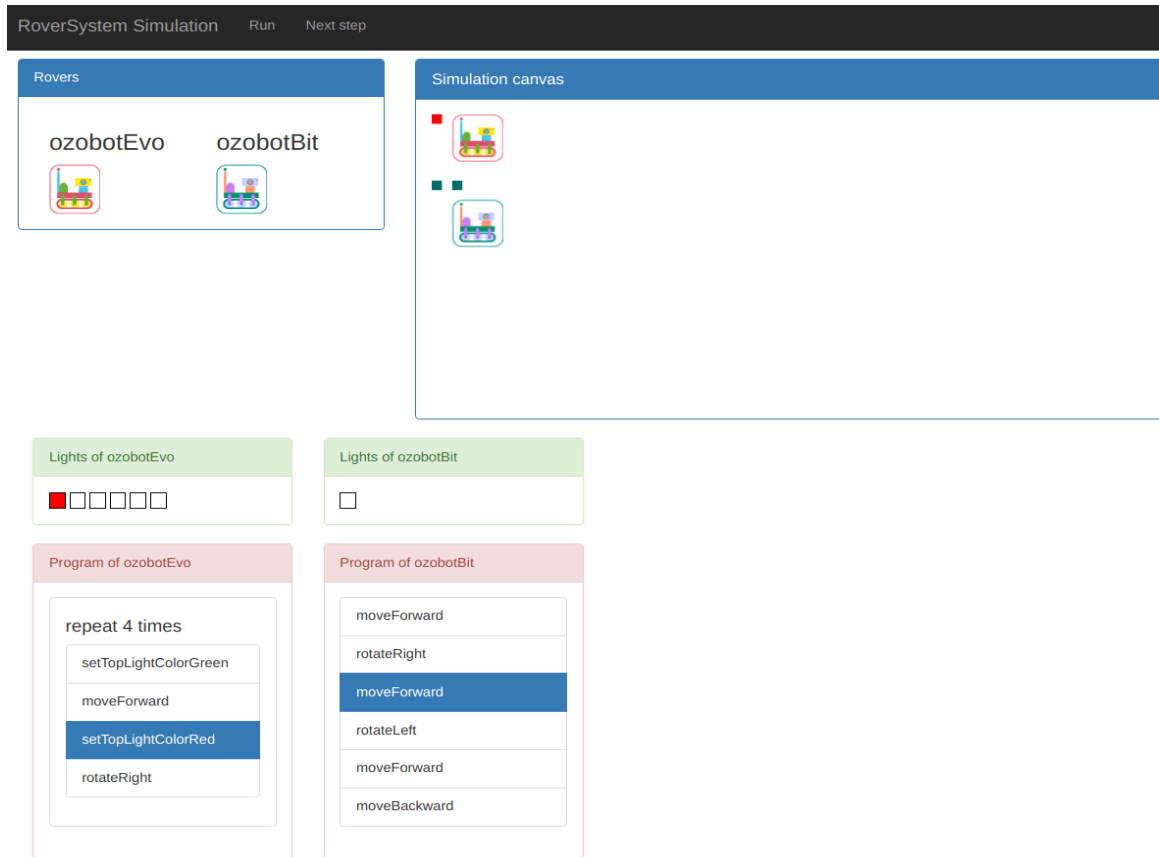


Figure 1: RoverSystem simulator

Figure 1 shows an example for the design of the simulator. It contains the following elements:

Component	Description
Run buttons	A menu that contains the buttons which control the simulation: Run and next step.
Rovers List	Here, you have to display the rovers (name + image)
Simulation canvas	On this canvas, the rover programs are simulated. It shows the rovers and simulated execution of their programs.
Rover Lights	Each rover has a small panel that displays its lights
Rover program	Each rover has a panel that shows the list of commands of its program. During simulation, the current command is highlighted.

HTML example

The assignment resources provide you an example of simulator. Examine this example carefully. This will help you to write the code generator in Xtend.

Code generation

The **only** file that you have to modify in this assignment, is the **src/SimulatorGenerator.xtend**. Here we describe the steps for code generation:

1. Create the Xtend class for generating the code of a given rover system. This class should generate an HTML (roverSystem.html) file that contains the whole code for the simulation. For each rover and his components (lights, program, image on the simulation canvas) you must create the corresponding HTML elements. These elements should contain unique ID. This is very important for the simulation step.
2. Generate with Xtend a .js file that contains the *onload* function for drawing the initial position of the rovers.
3. Your generated code must be coherent with the given lib (CSS & JS). Carefully name the HTML elements and assign IDs to them.

Mandatory features

Your generated code must contain the following features:

1. Display all the rovers of a system
2. Display all the lights of a rover
3. Display the sequential list of commands for each rover.
 1. This includes repeat commands.
 2. As shown in the example, identification of the sub-commands of a block is shown using sub-lists in html.
4. Assign different image filter to identify the rovers (also displayed on the generated code)
5. Draw the rovers on an html canvas (onload function).
6. Create a menu to run the simulation

As means of simplification, infinite repeats or move commands are not allowed in this lab. Furthermore, only normal transitions (no triggered transitions) are allowed. Finally, each command has at most one incoming and one outgoing transition.

Tip: Check the validity of your html code

Once your html code is generated, use the w3 validator: <https://validator.w3.org/> to check its validity

Part B: Executing a Rover Program

In the second part of the assignment, the goal is to write the simulator that executes the programs that belong to the different rovers. The implementation must be done in the same file as part A, **src/SimulatorGenerator.xtend**.

The general ideas are the following:

1. Run button launches the whole execution at once.
2. Next step button allows to execute commands one by one.
3. The executed commands are highlighted (in the frames on the left).

Example

The given example shows the full execution of a rover system containing 2 rovers. Examine the `example.js` file and the `simulation.js` (lib) carefully.

The code generated with your Xtend project must be operational for simulation without any manual modification.

Mandatory Features

1. An **onload** function that draws the first position of the rovers (pay attention to draw each rover at a different position).
2. A **CSS filter** is assigned to each rover. This helps to distinguish the icons of rovers and the paths that the rovers follow.
3. The **program** of each rover consists of a JS function.
4. The following list of commands are mandatory:
 1. **move**: calling `moveRover` function
 2. **setLight**: calling `setLight` function
 3. **rotate**: calling `rotate` function
5. Two execution modes:
 1. **run**: executes the functions that describe the programs of rovers.
 2. **Next step**: executes the next command of each rover.
6. In **next step mode**, the current command is highlighted: calling `highlight` function.

Your generator

The code generator you develop must produce the following files:

- `system.html` a file for each rover system.
- `system.js` file that contains the programs execution and all function calls.
 - Reference the `simulation.css` library.
 - Reference the given `simulation.js` library. All the commands are encoded in this library.

Submission & Assignment Review

Upload the following components in TUWEL:

You must upload one archive file, which contains the following projects:



For exporting these projects, select *File* ☰ *Export* ☰ *General/Archive File* and select the projects. Make sure that all files contained by the projects are selected.

Assignment Review:

At the assignment review, you will have to present your Xtend code generator. You also must show that you understand the theoretical concepts underlying the assignment.

All group members must be present at the assignment review. The registration for the assignment review can be done in TUWEL. The assignment review consists of two parts:

- Submission and **group evaluation:** 20 out of 25 points can be reached.
- **Individual evaluation:** Every group member is interviewed and evaluated separately. The remaining 5 points can be reached. If a group member fails in the individual evaluation, the points reached in the group evaluation are also revoked for this student, resulting in a negative grade for the entire course.