Assignment 6: Improving the GUI Responsivity using Threads

Submission date: 21st of May 2018, 09:00am

Objective: Java Threads and Swing.

1. Overview

You must have noticed that, in assignment 5, when running the simulator the GUI is updated only once when the execution ends. In fact, it is updated several times by we only see the last update since all are executed at the end. This is because the simulator's `run` method is called from the *events dispatch thread* of swing, which is the one on which we execute the GUI updates as well. In addition, currently there is no way to stop the simulator once it is started, we must way until it executes all time units that we have specified.

We have seen this problem in class already, and also discussed how to solve it using multi-threaded programming. In this assignment, we will apply this solution to assignment 5, and thus make the GUI responsive. In particular, we will modify assignment 5 such that it starts the simulator's main loop from a different thread, and thus see how the traffic evolves more lively. In addition, we will add components to allow the user to stop the simulator at any time and to set a *delay* between the iterations in order see the progress slowly.

2. Modifying the GUI View/Controller

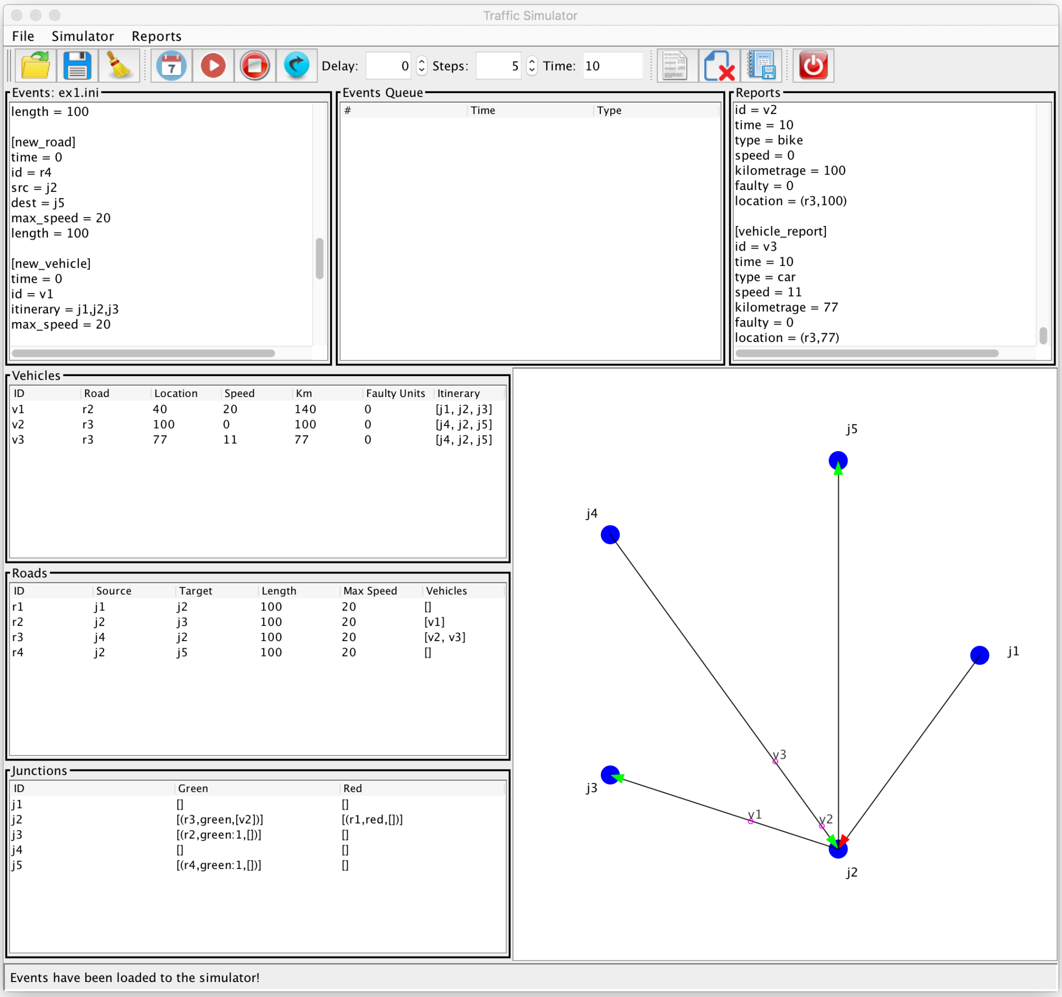
You are required to do the following modifications to the traffic simulator GUI view/-controller:

- add a button to be used for stopping the simulator and a component for setting the *delay* (see Figure 1);

- when pressing the `Run` button, instead of calling the `run` method of the simulator/controller directly, create a new thread in which you call `run` n times, where n is the provided number of time units, and each time execute the simulator's loop 1 time unit and sleep the thread for the selected amount of *delay* (the delay is in milliseconds).
- when pressing the `Stop` button interrupt the thread (of the previous point) to stop the execution of the simulator's main loop, etc.
- when the simulators is running, all menus and buttons, **except the `Stop` button**, should be disabled. They should be enabled again when the simulator's main loop stops.

3. Further Comments and Requirements

- You can either create the thread yourself, use executors, or Swing workers.



Events: ex1.ini

```
length = 100

[new_road]
time = 0
id = r4
src = j2
dest = j5
max_speed = 20
length = 100

[new_vehicle]
time = 0
id = v1
itinerary = j1,j2,j3
max_speed = 20
```

Events Queue

#	Time	Type

Reports

```
id = v2
time = 10
type = bike
speed = 0
kilometrage = 100
faulty = 0
location = (r3,100)

[vehicle_report]
id = v3
time = 10
type = car
speed = 11
kilometrage = 77
faulty = 0
location = (r3,77)
```

Vehicles

ID	Road	Location	Speed	Km	Faulty Units	Itinerary
v1	r2	40	20	140	0	[j1, j2, j3]
v2	r3	100	0	100	0	[j4, j2, j5]
v3	r3	77	11	77	0	[j4, j2, j5]

Roads

ID	Source	Target	Length	Max Speed	Vehicles
r1	j1	j2	100	20	[]
r2	j2	j3	100	20	[v1]
r3	j4	j2	100	20	[v2, v3]
r4	j2	j5	100	20	[]

Junctions

ID	Green	Red
j1	[]	[]
j2	[(r3,green,[v2])]	[(r1,red,[])]
j3	[(r2,green:1,[])]	[]
j4	[]	[]
j5	[(r4,green:1,[])]	[]

Events have been loaded to the simulator!

Figure 1: The Graphical User Interface.