1. File Structure

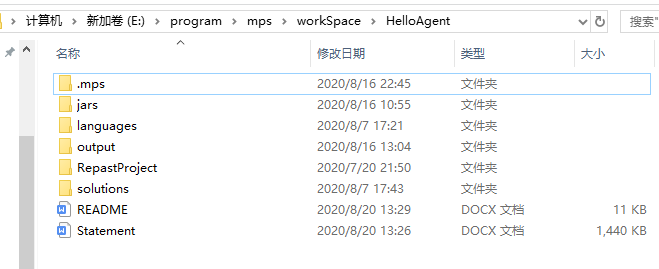


Figure 1.1 Project structure

The project structure is shown as the Figure 1.1. Fundamentally it is a MPS project( with .mps ) directory, so it can be directly imported by Jetbrains MPS IDE, as Figure 2.1 shows. The “jars” directory contains necessary external jar files, “output” directory records all log files output of each simulation, “RepastProject” is an embedded Repast Simphony project demo for replacing files and executing simulations( they are automatically done by the project ). “languages” and “solutions” directories are MPS project directories containing languages and solutions of the project separately.

2. Configurations

Please download and configure Jetbrains MPS IDS from <https://www.jetbrains.com/mps/> before executing. “HelloAgent” folder is a MPS project folder and can be imported as Figure 2.1 shows.

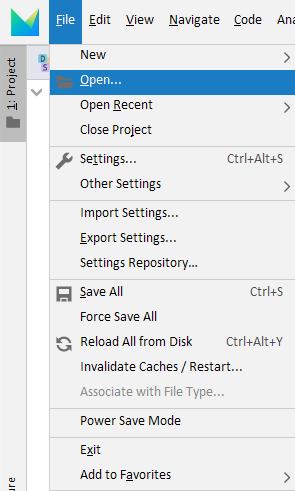
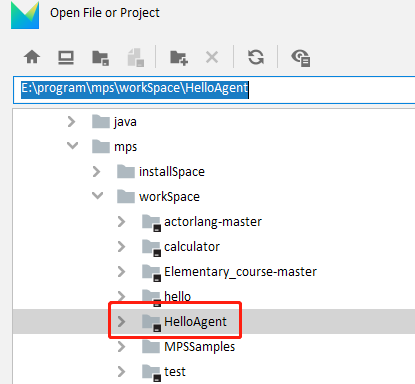
 

Figure 2.1 MPS import

The project structure should look like Figure 2.2 now. Before executing, please configure “RepastMainProvider” Solution in “dependencies” solutions(because only absolute path can be used in MPS Solution configurations), to make dependencies reference correct jar files.

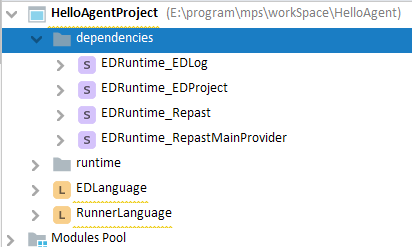


Figure 2.2 MPS interface

Select “RepastMainProvider\_EDLog” solution and press “Alt + Enter”, or right click and choose “Module Properties”, to open a configuration interface as Figure 2.3 shows.

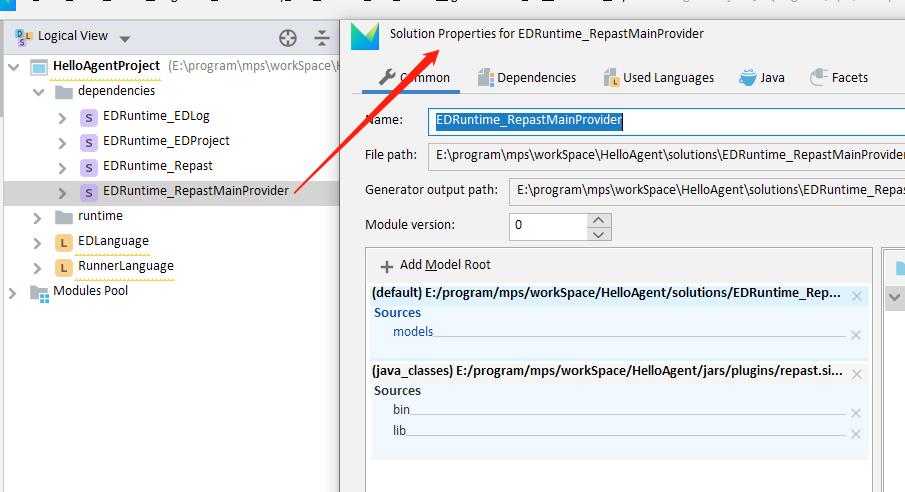
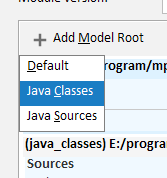
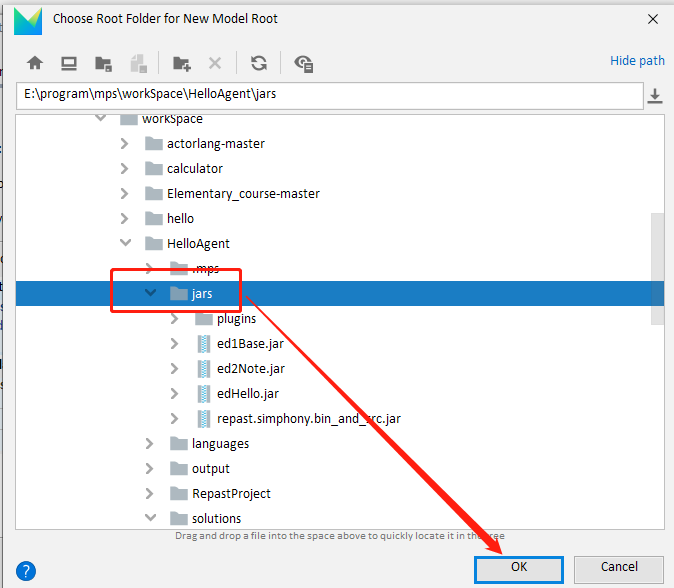


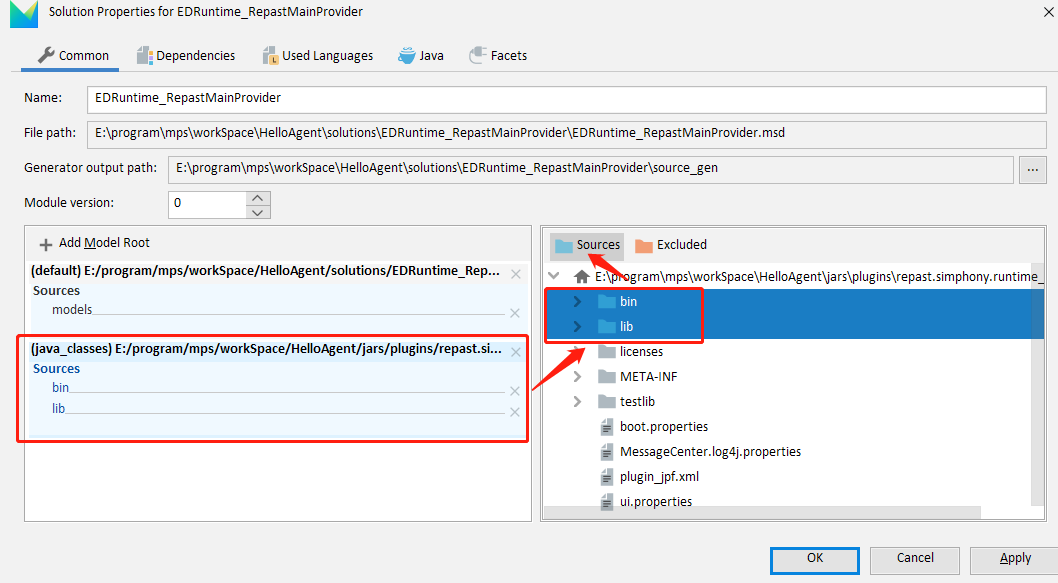
Figure 2.3 Configuration interface

Press “Add Model Root” and select “Java Classes”, then find and select “jars” directory of project path, click “OK”.

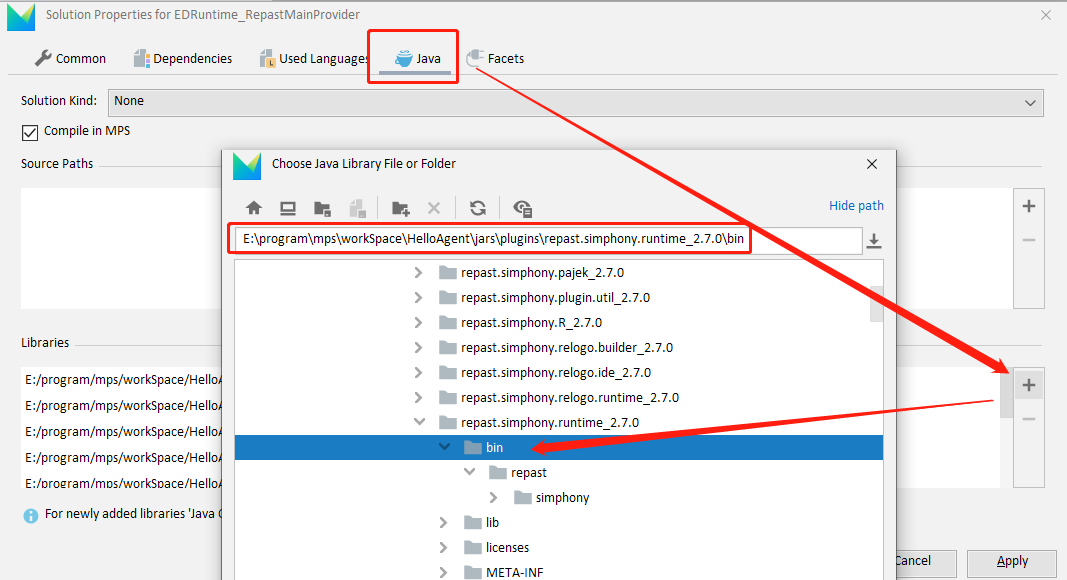




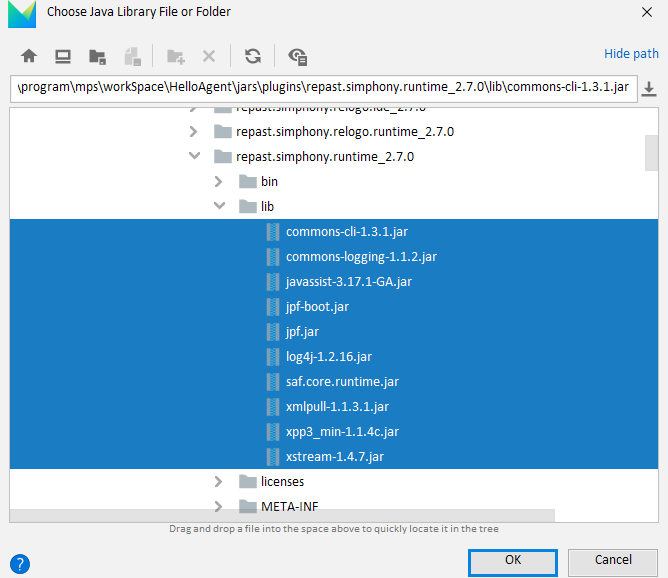
Select added Model Root, select “bin” and “lib” directories, click “Sources” button, until the two directories become blue.



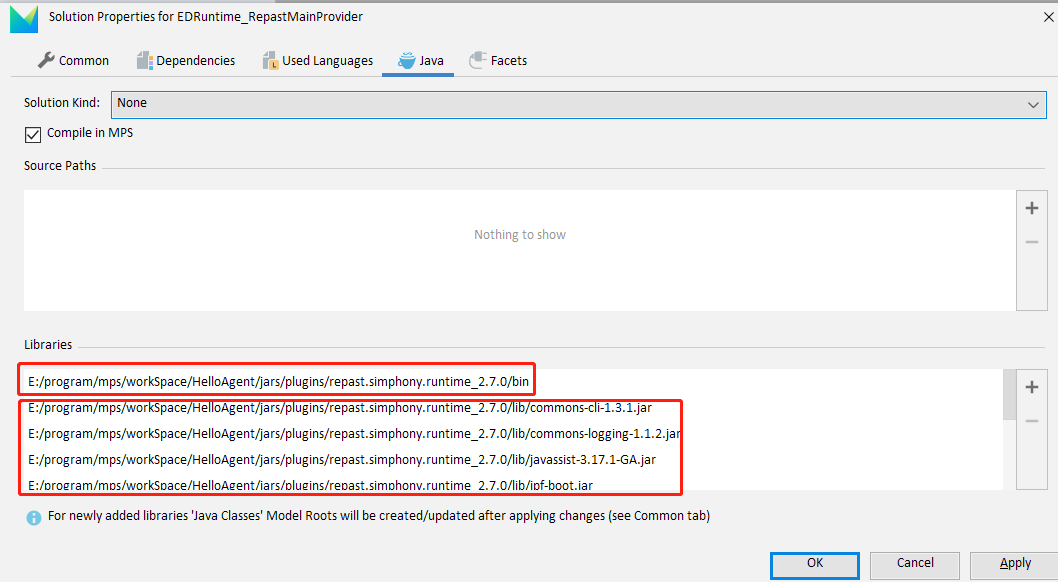
Add java Libraries from this interface. Click “+” and find directory “..\HelloAgent\jars/plugins\repast.simphony.runtime\_2.7.0\bin”, select the “bin” directory, click “OK”.



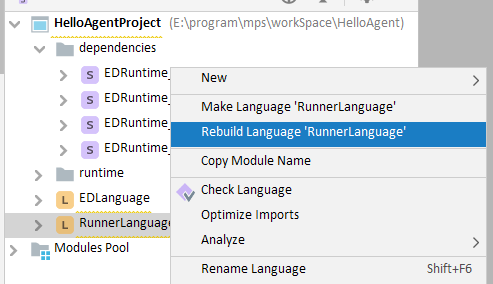
Click “+” again, select all jar files under “lib” directory( unlike previous step, select the jar files NOT the lib directory ), click “OK”.



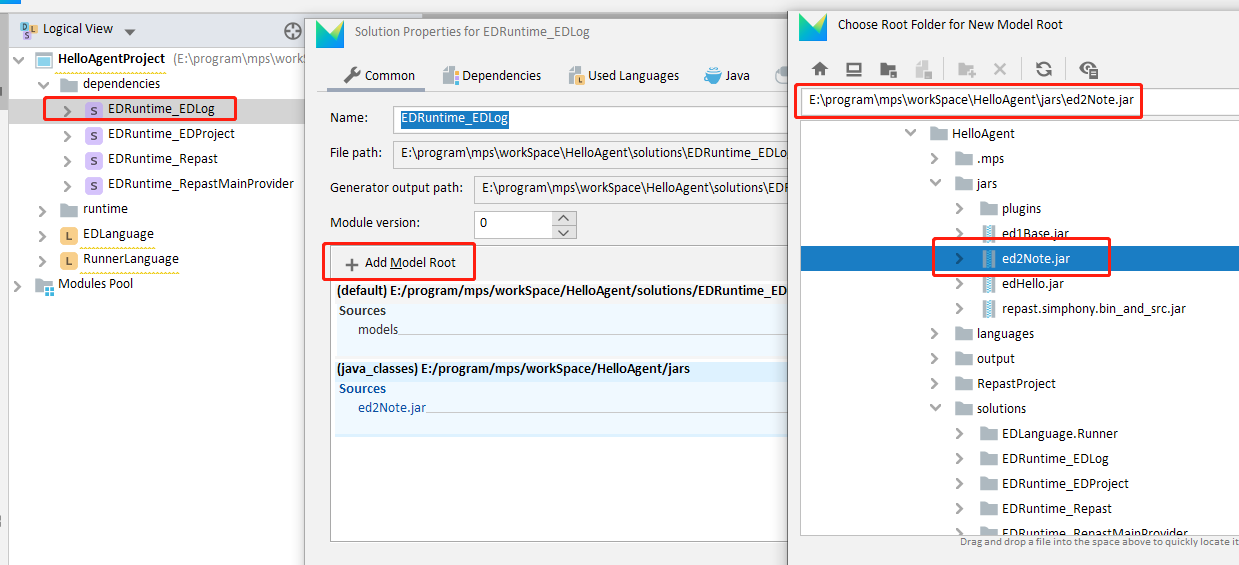
Configuration of “RepastMainProvider” solution should look like this:

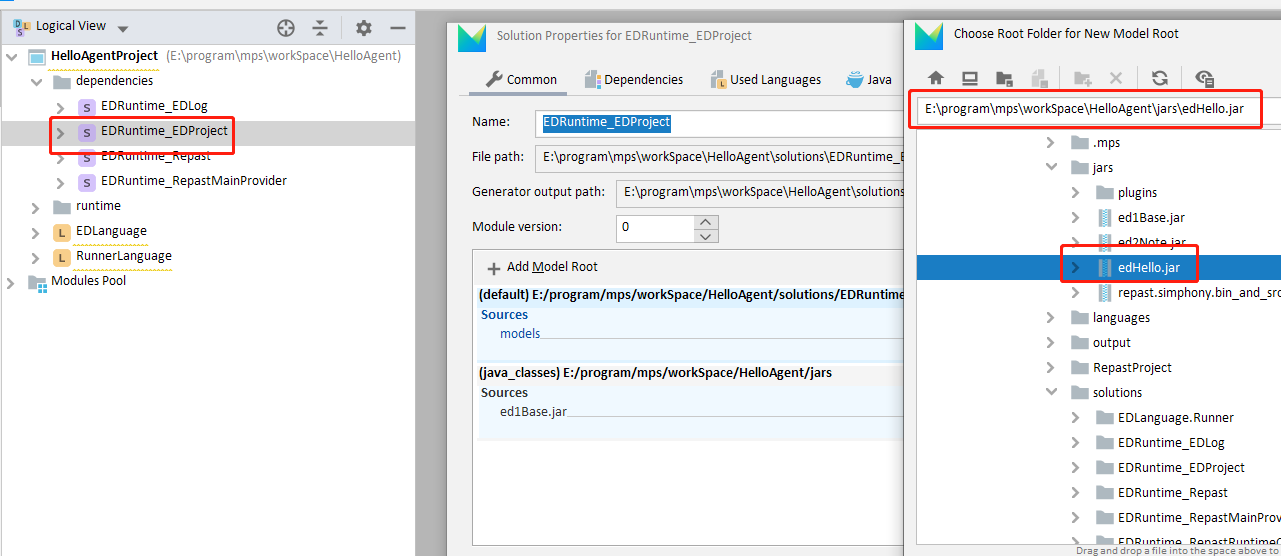


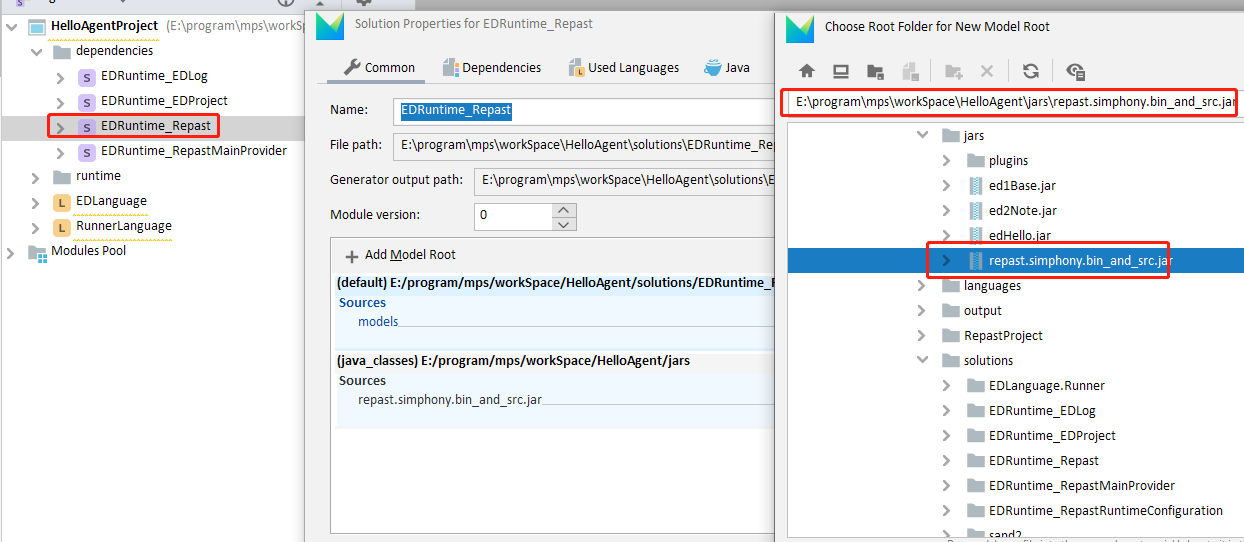
Then click “Apply” and rebuild “RunnerLanguage”, it should work now.



If anything went wrong, please configure the “Model Root” and “java Libraries” of the other 3 runtime solutions in “dependencies”pack in a same way, following the correspond relationships below.



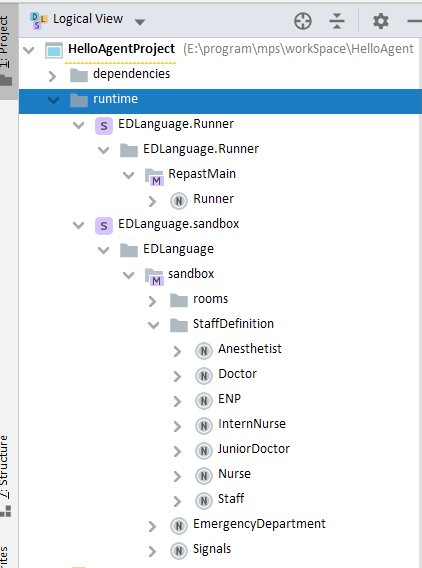




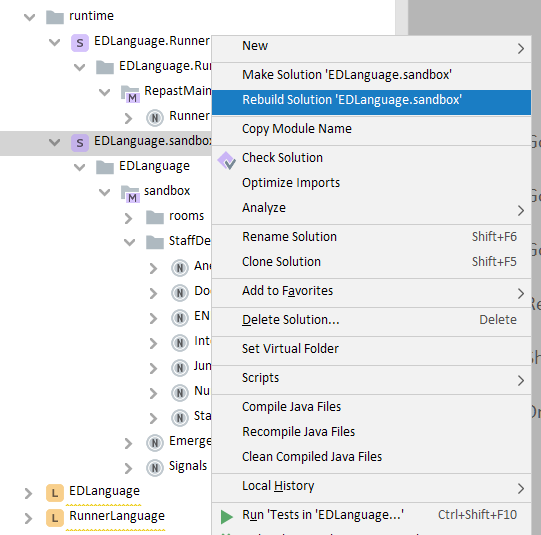
Then rebuild the two Languages.

3. How To Use

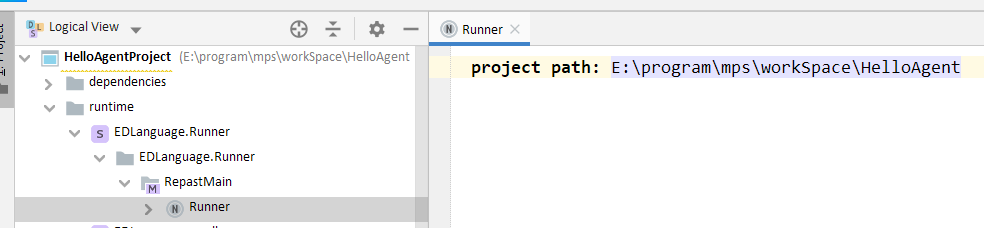
Model designing and simulating is done in the two Solutions in “runtime” pack. The design is done in “EDLanguage.sandbox” solution, the invocation of simulation is done in “EDLanguage.Runner” solution, as specified in the report.



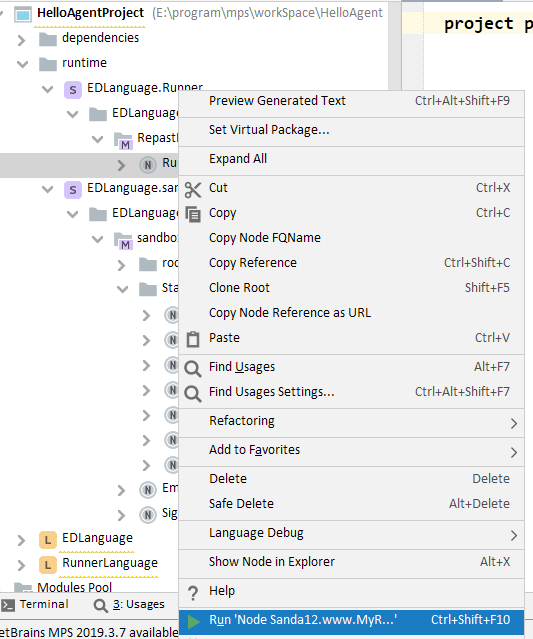
After editing the model, Rebuild the sandbox solution:



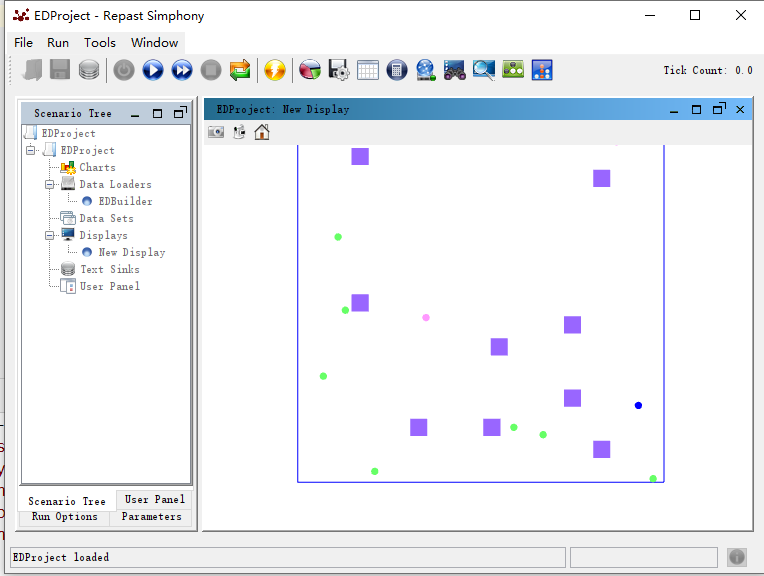
Click Runner node in Runner solution and configure the project path, which correspond to path in Figure 1.1:



Rebuild “Runner” solution, right click on “Runner” node, select “Run” instruction:



Simulation interface representing the ED model should be seen now.



Please rebuild the “sandbox” solution if ED model is modified, and rebuild “Runner” solution if project path configuration in Runner node is modified, before Running simulation.

The uploaded project only contains my part of work, so the move action of agents is still naive and constant places are concretely defined, only to demonstrate the work of agent workflow design. The constant places here are only for demonstrating and cannot be modified, they correspond to purple squares in ED map in the way graphs below shows.

