

Kennesaw State University
College of Computing and Software Engineering
Department of Computer Science

Developing Simulation Models with OOSimL Using Eclipse
For Java Code

José M. Garrido

Updated November 2016

Introduction

OOSimL is a high-level programming language that was developed for implementing discrete-event simulation models using the object-oriented approach. Using an Integrated Development Environment (IDE), such as Eclipse greatly facilitates the editing, compilation, and running of simulation models.

Building an Eclipse project enhances the OOSimL model development. An Eclipse plug-in was developed recently by Kevin Sealy for use with OOSimL; it can be imported and set up to work with OOSimL projects. The features of this plugin are:

1. OOSimL keyword syntax highlighting
2. OOSimL comments & description highlighting including important OOSimL functions and important keywords.
3. OOSimL content assist to prompt programmer with relevant choices
4. Include functions and variables in the Content Outliner including code highlighting when selecting an item in the content outliner.

This report describes the installation of Eclipse and the OOSimL editor plug-in. Additional information is provided that describes the process for configuring Eclipse to work with OOSimL projects. Finally, the features that are built into the OOSimL editor plug-in are described in detail.

Installing The Eclipse OOSimL Editor Plug-in

The archive file to download, *oosiml.zip* on Windows and *oosiml_soft.gz* on Linux. The archives include the Eclipse plugin, the OOSimL compiler, and the OOSimL library.

The Eclipse version 3.8 plug-in file is included in the archive and can be downloaded from: <http://ksuweb.kennesaw.edu/~jgarrido/oosiml/>. After downloading, unzip the *oosiml.zip*. The Eclipse version 3.5 plug-in file included is *edu.kennesaw.oosiml_1.0.1.jar*.

Copy the plug-in file into the *dropins* directory of an Eclipse installation to install the plug-in. Restarting Eclipse after copying the file will install the plug-in. The editor will automatically launch when opening a file with the *osl* extension.

The OOSimL editor plug-in for Eclipse requires Java SE runtime environment version 1.7 and above when running the Eclipse IDE. The OOSimL editor plug-in for Eclipse will run on any platform that supports the Eclipse IDE version 3.8 with Java 1.8. Follow the procedure outlined below to begin using the OOSimL editor plug-in for Eclipse.

As mentioned previously, the OOSimL Editor plug-in will launch automatically when any file with an *osl* is opened. The editor provides comment and OOSimL code syntax highlighting, content assist and an

outline of the functions and variables in the content outliner to assist the OOSimL developer when creating simulation models.

Installing the OOSimL Compiler and Libraries

1. Create a new OOSimL workspace for Eclipse for your OOSimL projects. After starting Eclipse choose File and select Switch Workspace, then Other from the Eclipse menu bar.
2. Create a new Java project. Choose File then select New. Select Project and then select the Java Project wizard. Then click next and type in a project name like *Carwash* and click Finish.
3. Next, setup the OOSimL compiler and the project directory structure. Extract the OOSimL compiler *oosiml.exe* for Winows or *oosiml.out* for Linux and library file *oosiml.jar* from the archive file.

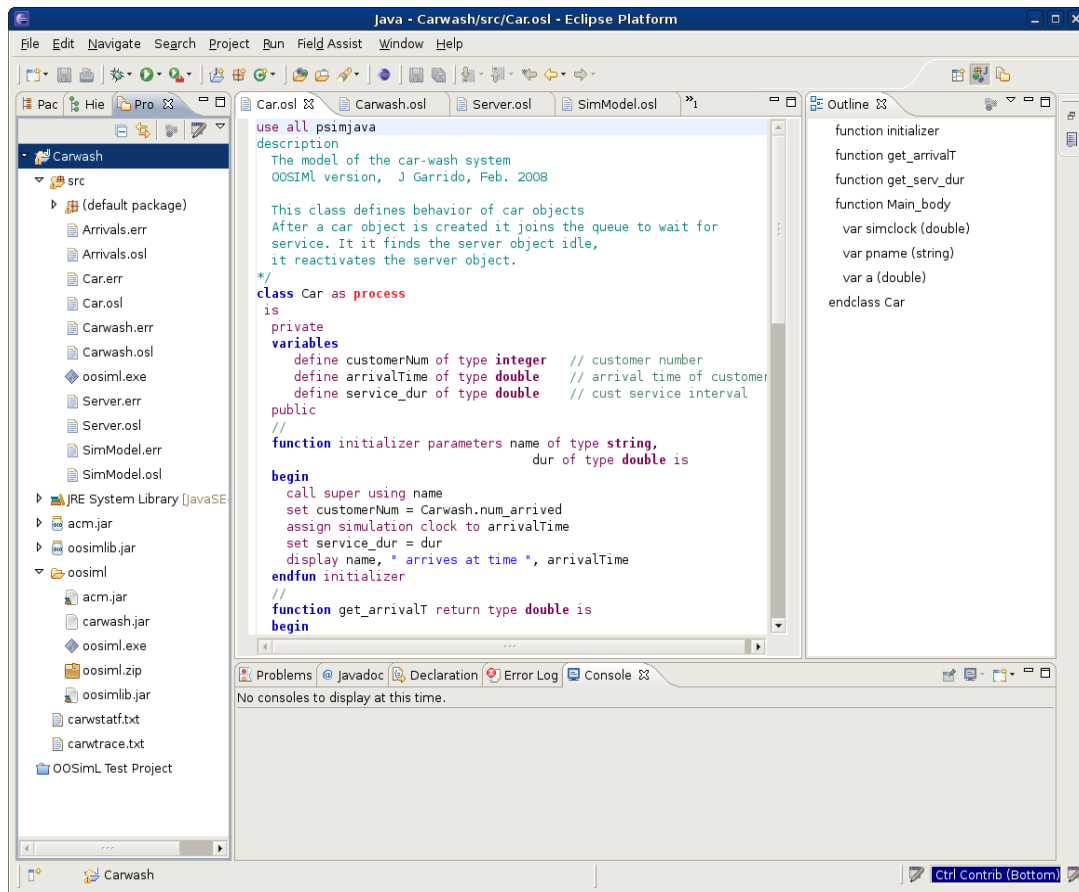
Installing the OOSimL compiler libraries:

1. Refresh the Eclipse project by right clicking the project in the view and choosing Refresh from the popup menu.
2. Open the project properties dialog by selecting the project name, choose the Project menu or right-clicking the project in the view and choose Properties.
3. Select *Java Build Path* on the left and choose the Libraries tab. Then click the Add External Jars button and select *oosimlib.jar* from the appropriate directory.
4. Finally, click the OK button to save your changes.

To setup the OOSimL compiler:

1. Choose Run and select External Tools, then select External Tools Configurations.
2. Choose the Program option on the left side followed by the new button to create a new configuration. In the Name field, type *OOSimL Compiler*
3. Select the Browse File System button below the Location field, navigate to the appropriate directory and choose the directory with *oosiml.exe*.
4. Click the Browse Workspace button below the Working Directory field and choose the **/src* directory.
5. Finally, click the Variables button below the Argument field and choose *selected_resource_name*.

6. Click the Apply button.



View of Eclipse with the OOSimL Editor Plug-in

Compiling and Running the Carwash Sample Project

1. Extract the files from *carwash.jar*, save them in the Carwash/src directory without creating a carwash directory inside the src directory.
2. If you are creating a project using existing OOSimL source files, then select Import from the File menu, and save the files inside the src directory in the project folder.

Extracting files from a JAR archive

1. On the current project, select the *src* folder.
2. Right-click on the SRC folder and select Import, or activate the File menu and select Import.
3. On the Import window, select General then Archive File.
4. Click the Next button.
5. In the Archive File window, on the input field "From archive file:", click the Browse button to select the source folder and the (JAR) archive file. Click the Open button. On the input field "Into folder:", make sure the *src* folder is selected on the current project folder.
6. On the left pane, expand the tree folder structure and click on the folder (for example, *carwash*) that contains the desired files. On the right pane, select the desired files.
7. Move the files from the folder copied to the *src* folder of the project.

The Eclipse is now set up to compile and run the simulation project.

1. To compile an *os/* file, you can open a file or select a file in the view.
2. Then run the OOSimL compiler by choosing the Run menu, then selecting External Tools, then selecting OOSimL Compiler.
3. The output from the compiler will show in the console view in the bottom of the Eclipse desktop.
4. Repeat this procedure for the each of the *os/* files.

Once all of the *os/* files are compiled, turn off the automatic build feature by selecting Build Automatically from the Project menu (this action will only need to be completed once). Refresh the project again (right click the project and choose refresh from the popup menu) and the compiled java files will now be visible in the default package folder. Now choose Build Project from the Project menu to compile all of the java files. The resulting class files will not be shown in the default view of the project but they can be found in a bin directory located inside the Carwash directory in the file system. To run a model, activate the Run menu and select Run As, then select Application.

Viewing the Carwash Simulation Results

After running the model, the output will show in the console window and the final dialog will display some of the simulation run statistics. Refresh the project one more time (right click the project and choose Refresh from the popup menu) and the *carwstatf.txt* and *carwtrace.txt* files will be visible in the view. The files can then be opened to see the results of the simulation run.

Syntax Highlighting

The OOSimL Editor plug-in provides syntax highlighting for both comments, OOSimL keywords, OOSimL classes and some keywords for classes, functions/methods and the `main` and `Main_body` function names. Single line comments and description comments will show in two different shades of green. All keywords show in a purple color and the different OOSimL types (integer, double, string, etc) show in bold in the same color. All OOSimL library classes are highlighted in a bold red color and a few of the OOSimL keywords like *function*, *endfun*, *variables*, *class* and *endclass* are shown in bold blue. In addition the *main* and *Main_body* functions are highlighted in bold blue.

Content Assist

When editing an *osl* file content assist will provide a list of all of the OOSimL keywords to assist without having to type the word. To trigger content assist, press the enter key while holding the <ctrl> key. Then select a word from the list to insert into the code.

Content Outline

The content outline view to the right side of the OOSimL editor view works together with the syntax highlighting feature to assist the developer when modifying existing code. The content outline will show all functions/methods and any class and function/method variables and object references along with the beginning class statement and the *endclass* statement. When selecting an item in the content assist view, the code in the editor view will show the selected item highlighted in the left side of the view. When selecting the class, the entire view will be highlighted from the class statement to the *endclass* statement. When a function is selected, the code for only that method is highlighted and when a variable/object reference is selected, just the line for that variable or object reference will be highlighted.