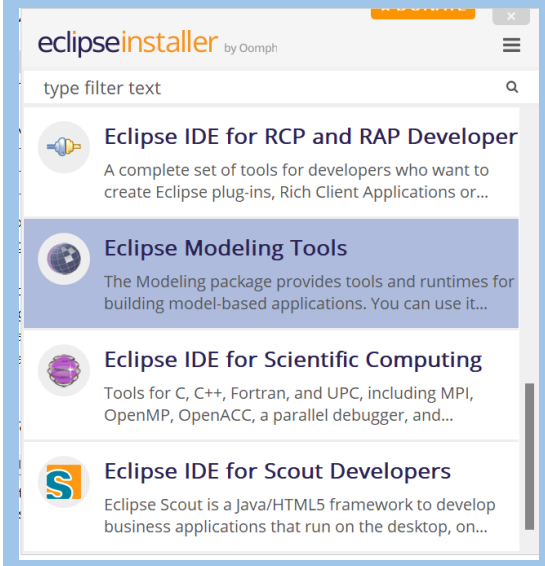
1. Summary

This document, besides exercises, contains details about the setup of the required software and the code artefacts for Manipulating Models and Metamodels using a Model-Driven Approach using EMF and Eclipse Framework in the Eclipse IDE as well as dealing with Sirius Visual Editors. The text is to be used as a collection of practical examples to support the lab lectures of the course on Model-Driven Engineering at Faculdade de Ciências e Tecnologia at Universidade Nova de Lisboa (FCT/UNL) in Portugal.

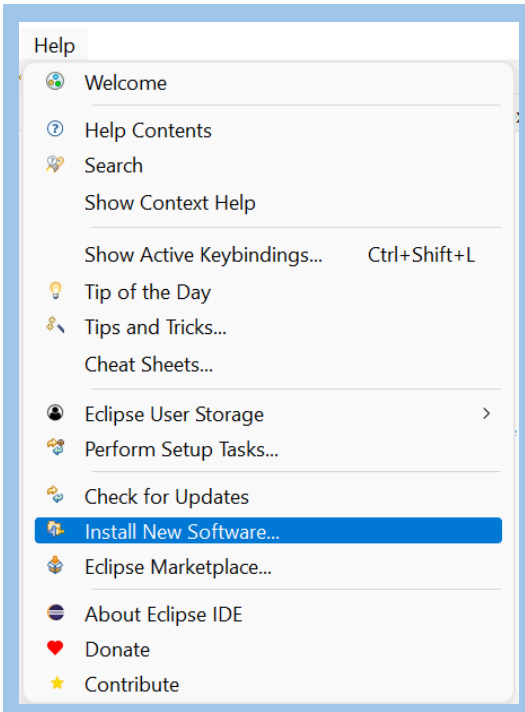
2. Eclipse Setup

The detailed descriptions in this section, and the following ones, are using **Eclipse IDE 2022-09** downloaded from https://www.eclipse.org/epsilon/download/

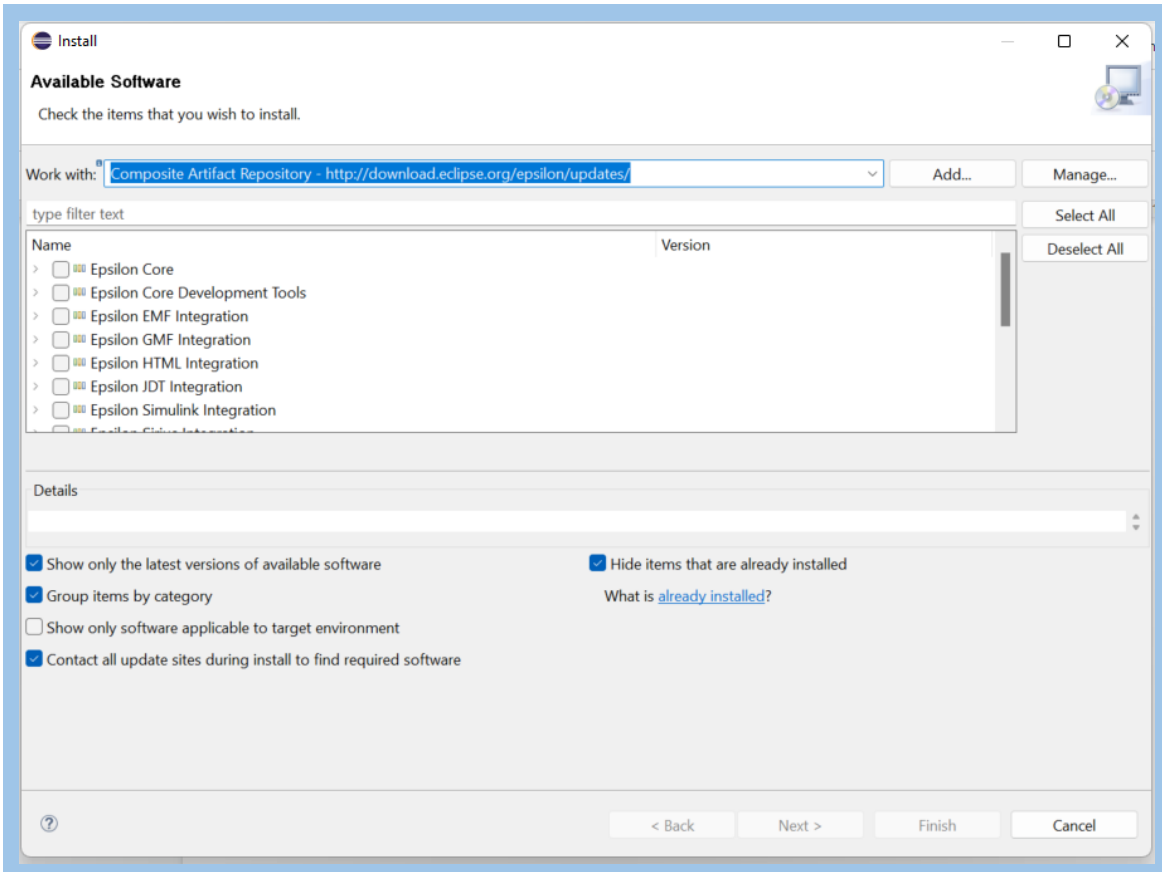
Make sure that when you run the installer, you select the Eclipse Modeling Tools, as in the Figure:



Install the Epsilon 2.4

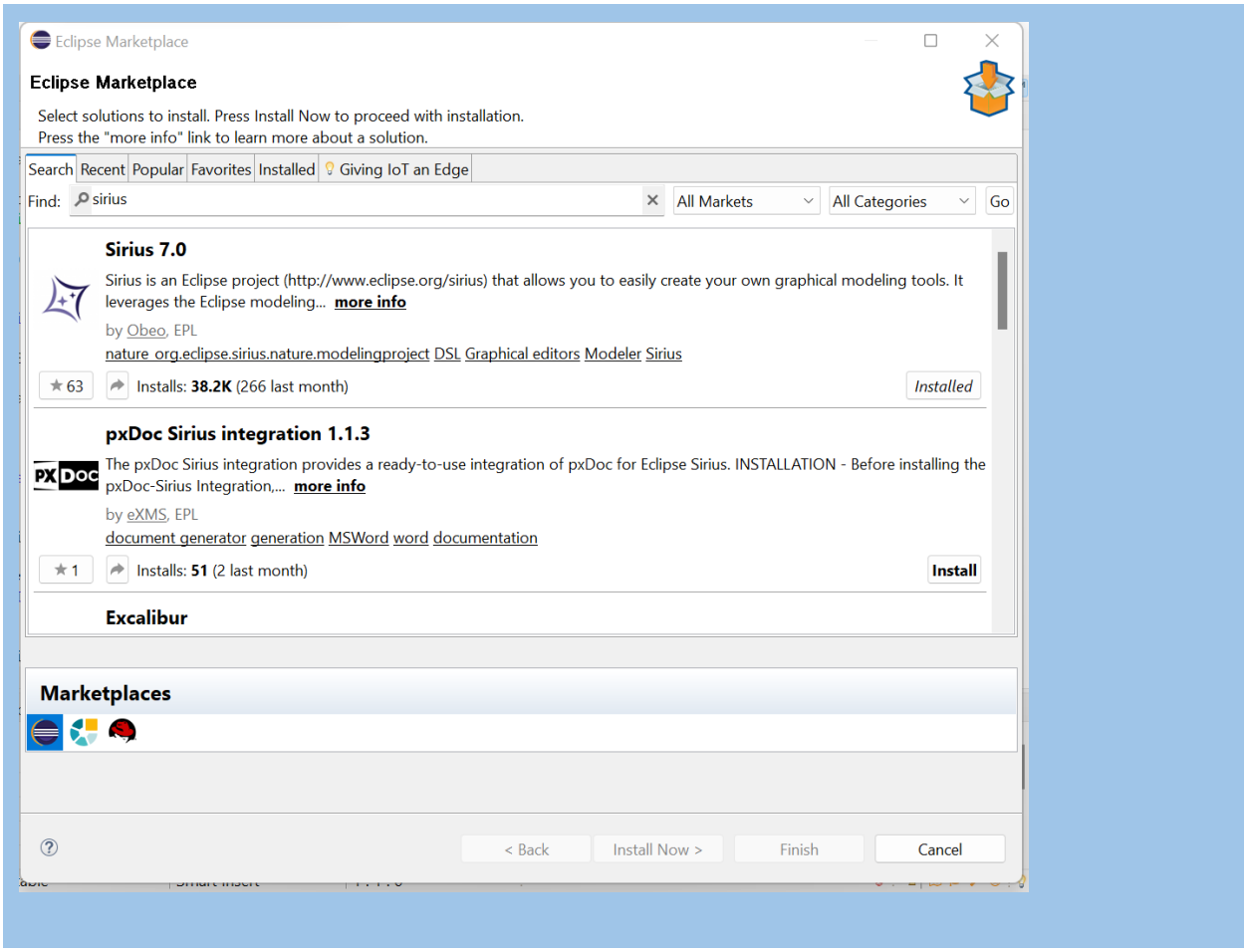


When running Eclipse after installing it, you will have to install Epsilon 2.4 by introducing the following path http://download.eclipse.org/epsilon/updates/ in the dialogue box you get when you select **Help->Install New Software** as follows:



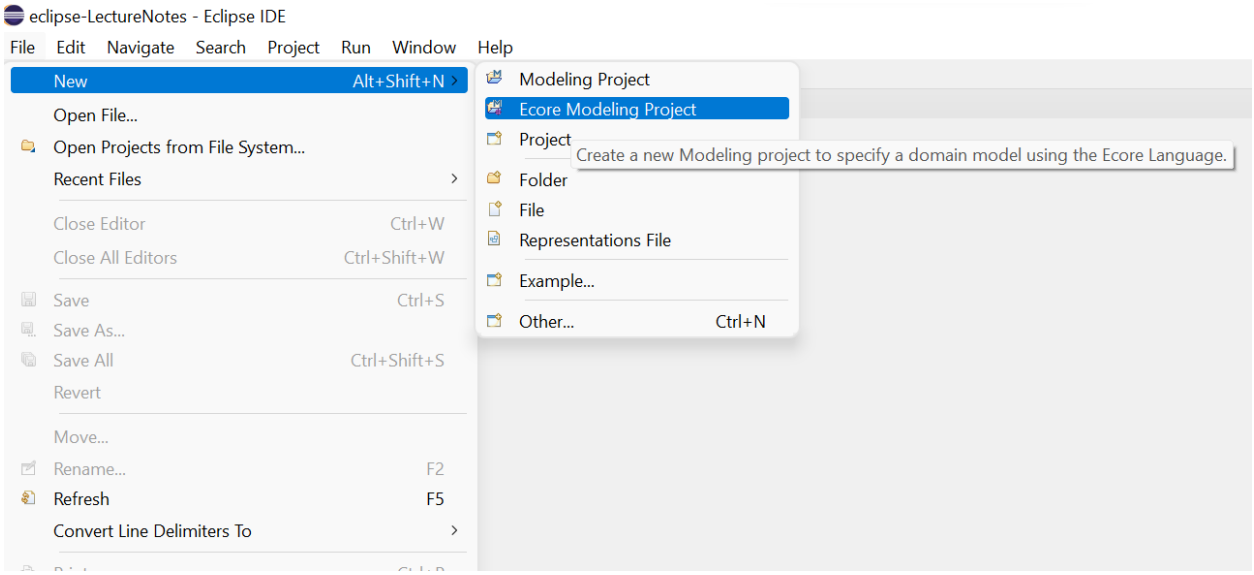
Select all the checkboxes mentioning Epsilon and then select Finish.

To install Sirius you can select Help->Eclipse Marketplace and install Sirius 7.0:

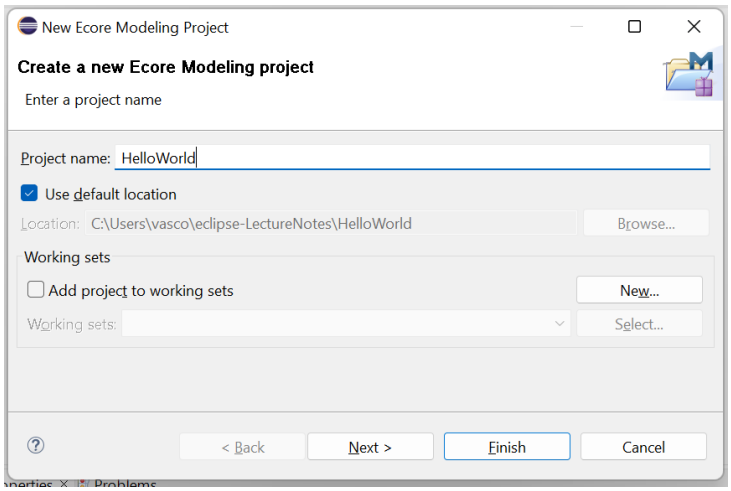

Eclipse Marketplace
Eclipse Marketplace
Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.
Search
Recent
Popular
Favorites
Installed
Giving IoT an Edge
Find:
sirius
All Markets
All Categories
Go
Sirius 7.0
Sirius is an Eclipse project (http://www.eclipse.org/sirius) that allows you to easily create your own graphical modeling tools. It leverages the Eclipse modeling... more info
by Obeo, EPL
nature_org.eclipse.sirius.nature.modelingproject DSL Graphical editors Modeler Sirius
★ 63
Installs: 38.2K (266 last month)
Installed
pxDoc Sirius integration 1.1.3
The pxDoc Sirius integration provides a ready-to-use integration of pxDoc for Eclipse Sirius. INSTALLATION - Before installing the pxDoc-Sirius Integration,... more info
by eXMS, EPL
document generator generation MSWord word documentation
★ 1
Installs: 51 (2 last month)
Install
Excalibur
Marketplaces
< Back
Install Now >
Finish
Cancel

3. Creating my First Modelling Project

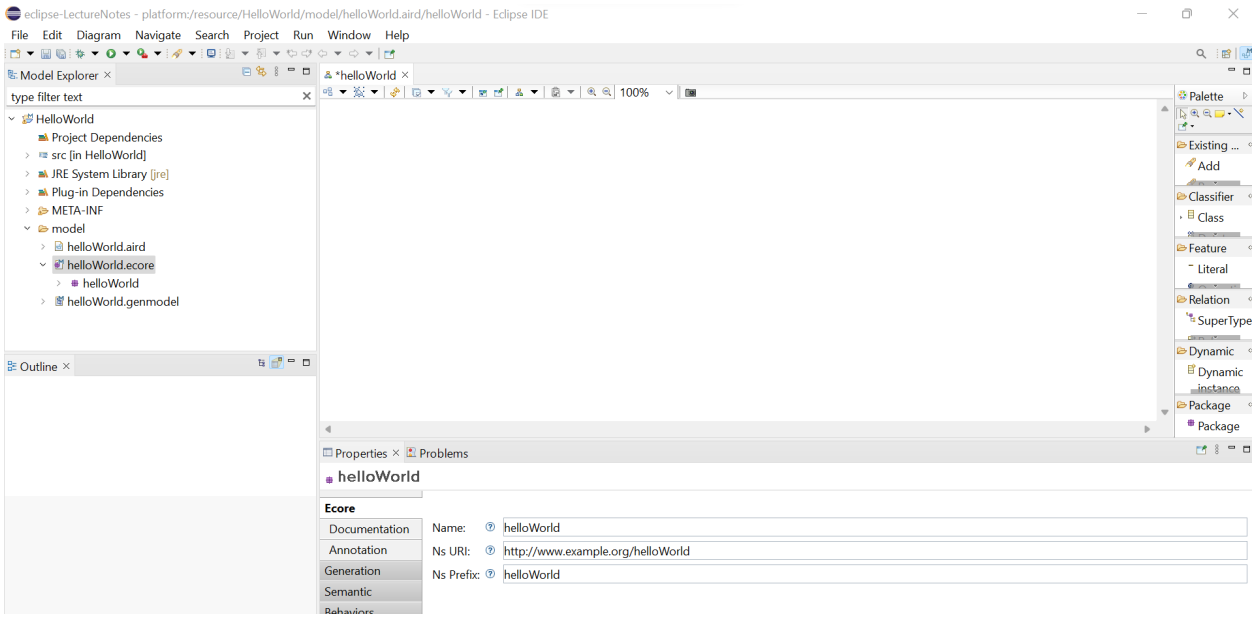
The first thing to do is to follow **File->New->Ecore Modelling Project** :



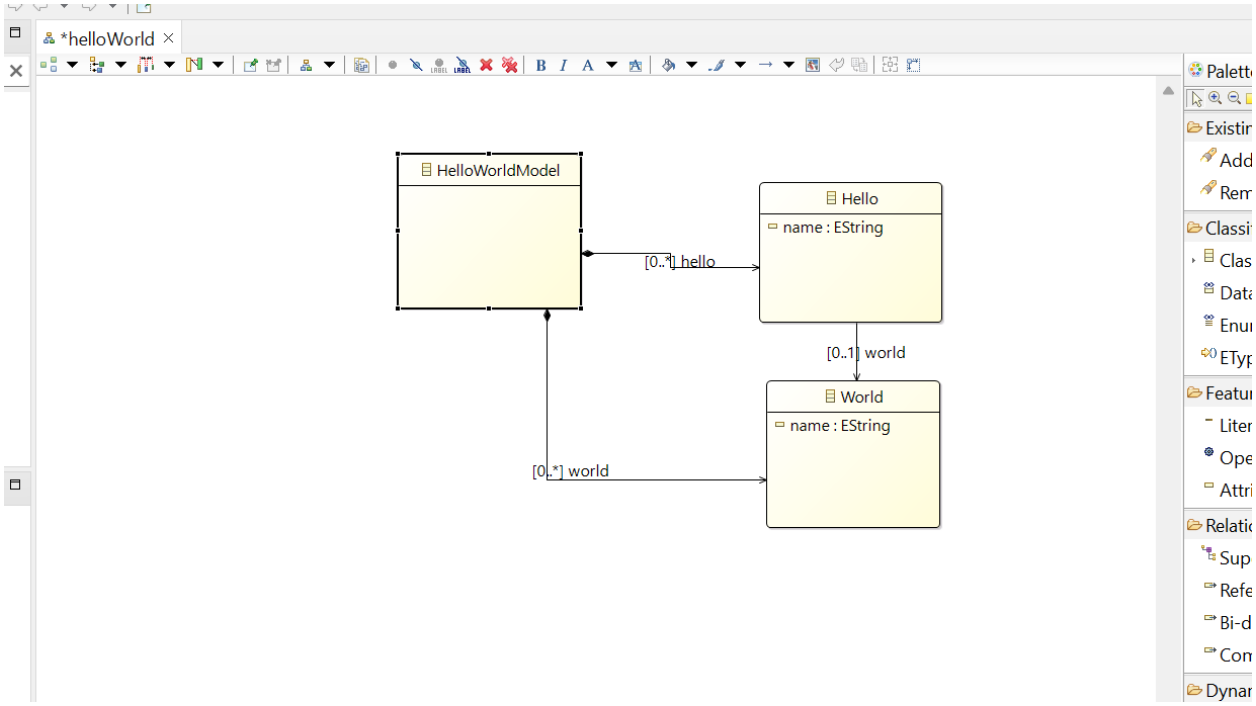
Then introduce a name like, for instance, “HelloWorld”:



After pressing Finish whole set of files and folders will be automatically created:



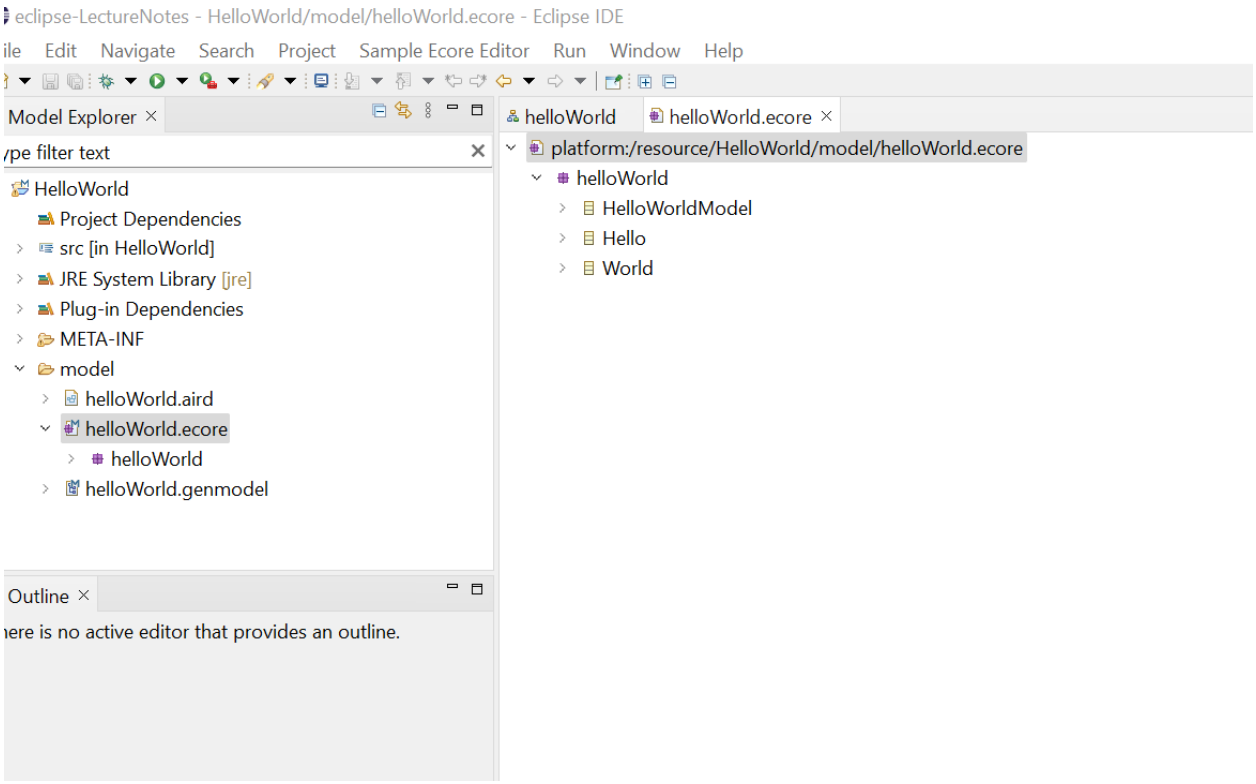
When navigating on the model explorer you will see the model folder, which is the place where you are going to edit all the models and metamodels manipulation files. An empty helloWorld.ecore empty file was created, and automatically the helloWolrd.aird is opened in the visual Diagram editor. You can start immediately modelling your first metamodel with the Ecore UML-like Class models notation. You can Model the following HelloWorld metamodel. By convention, you should always name the classes starting with capital letters and the names of attributes should be started with a lower-case character.



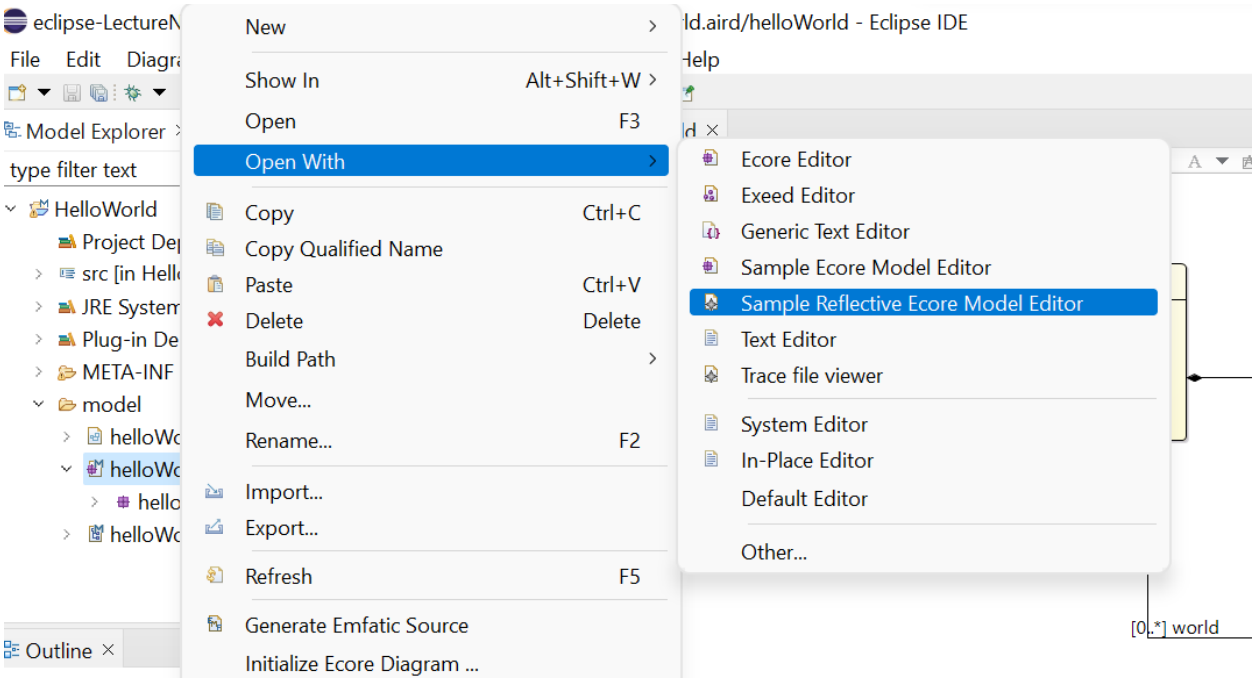
What we are saying here is that HelloWorld models have Hello's and Worlds. Each instance of those classes should have a name (or text, if you will) and a language. Each Hello is related to zero or more worlds.

Now save everything you've done so far (this should be a common procedure when you are dealing with these tools to keep consistency as this automatically should synchronize/update with your ecore file).

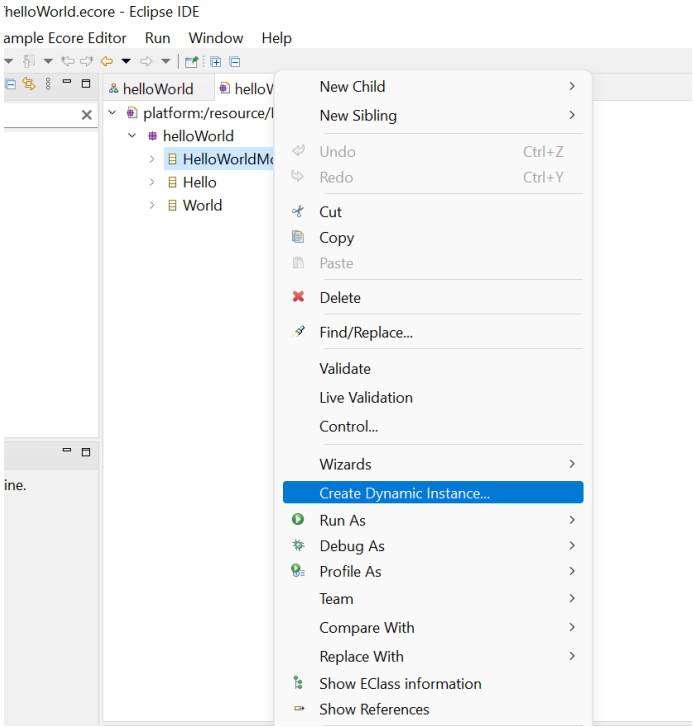
The next thing to do is to try out some instances of this metamodel. So let us create an example model by double-clicking on helloWorld.ecore file on the Model Explorer:



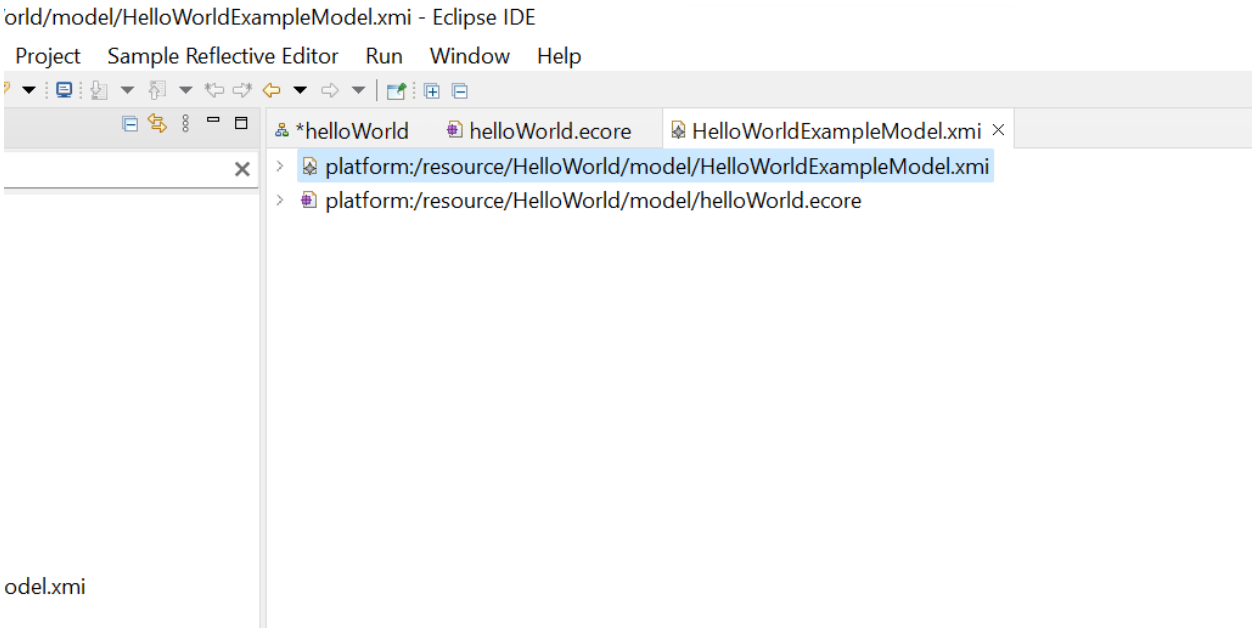
If you don't see the window like editor like the figure before, you can also right-click on the helloWorld.ecore file and select Open With->Sample Reflective Ecore Model Editor like this:



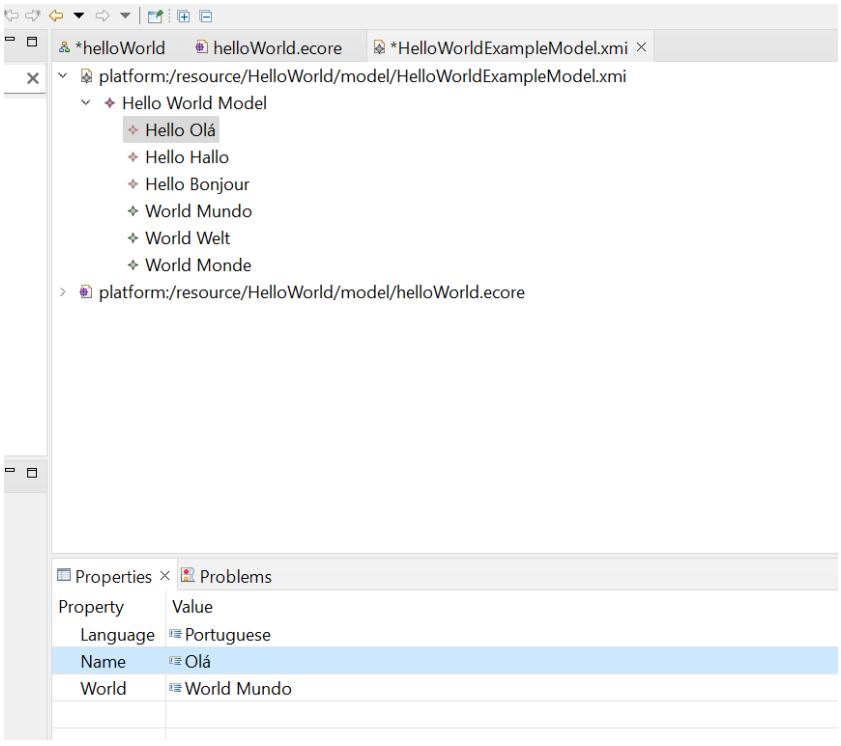
The next step is to create the model instances. For that, you need to right-click on the class HelloWorldModel. It should be always the class that contains the other elements, also called root node (and that usually, we name XXXModel to be clear). Select Create Dynamic Instance like in the following figure:



A dialogue box will appear asking for the name you want to persist the instance model you want to build as an example. You can call here, for instance, HelloWorldExampleModel.xmi:



Now you are able to add to the XMI file the instances in a explorer-tree-like fashion. Let us create the following example:

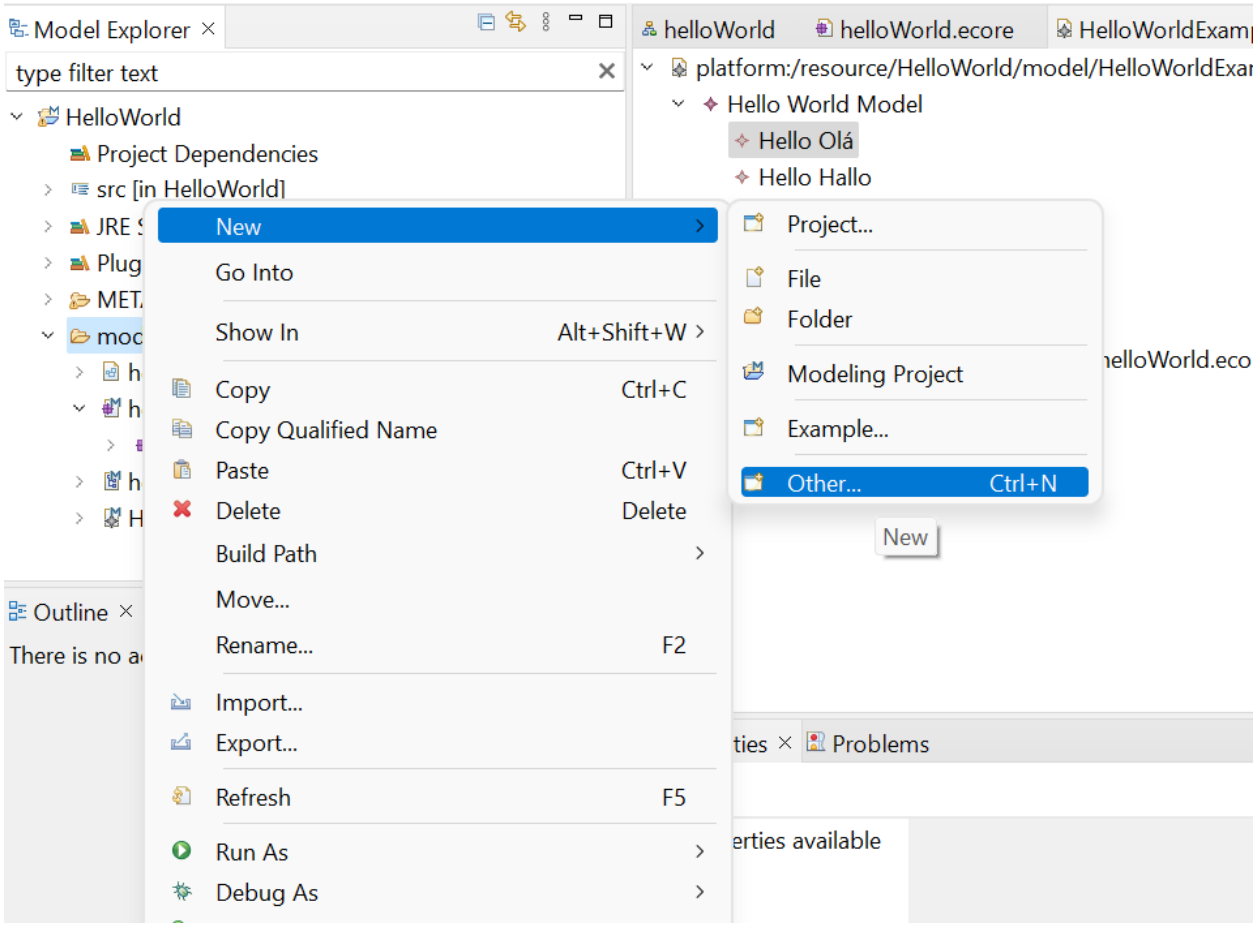


You can save it now. And that is it! You now have instances of your meta-model persisted in a XMi file.

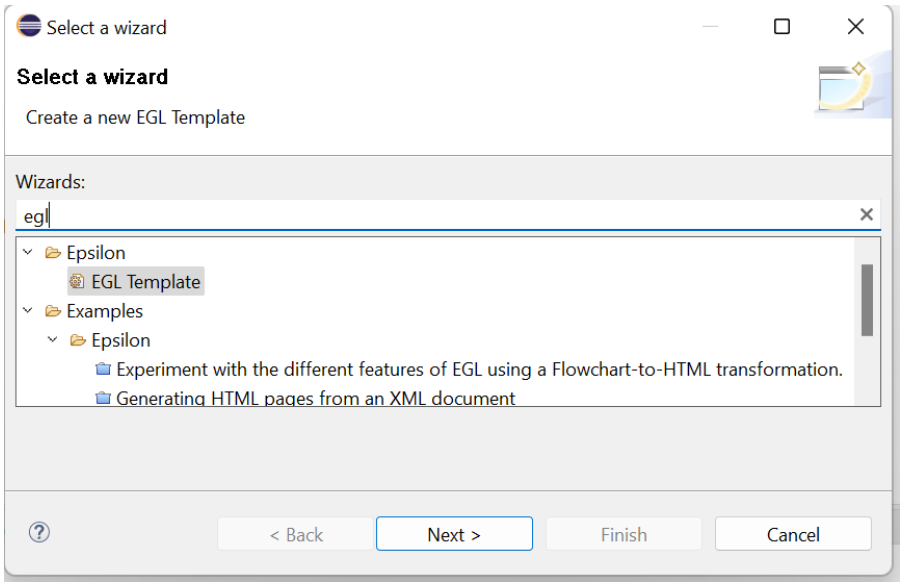
The next step is to make something with the instances.

Let us now translate the model instance to text, which is called model-to-text transformation. To do this we will use Epsilon. More specifically, we will use the Epsilon Generation Language (EGL) language (for more details about the language see the epsilon Book http://https://www.eclipse.org/epsilon/doc/book/).

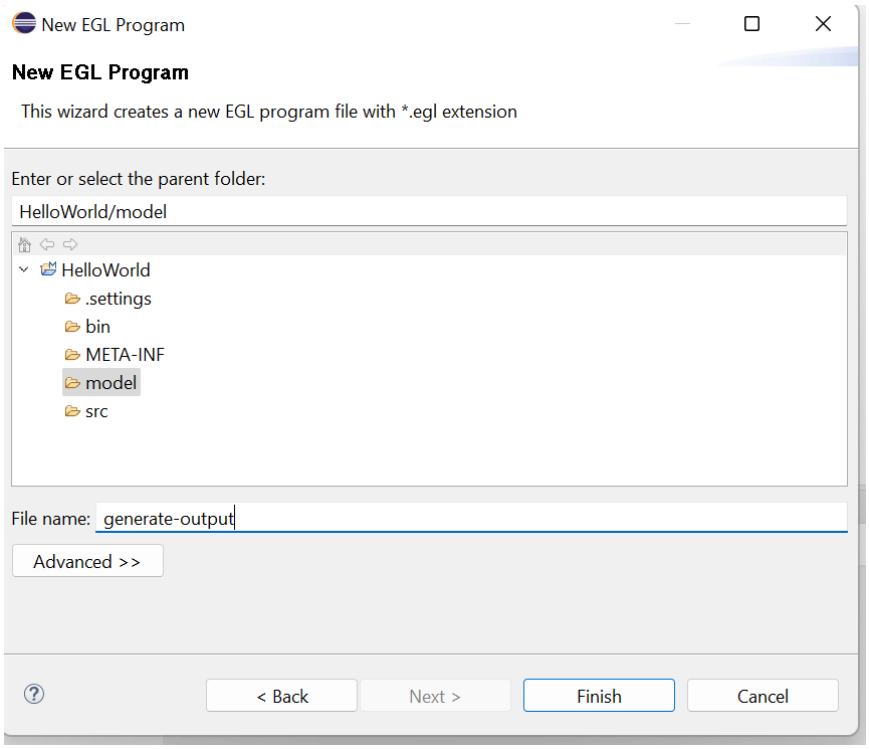
For that, you have to create in the same model folder a EGL file. Just follow the sequence **File->New->Other**:



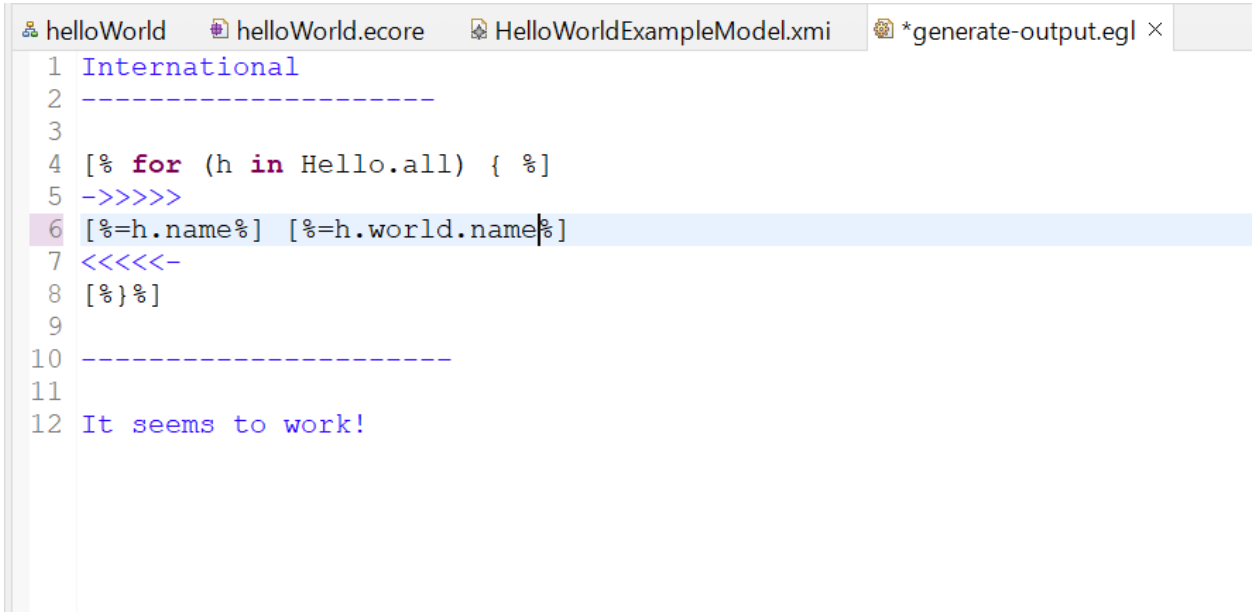
The next step is to look for egl:



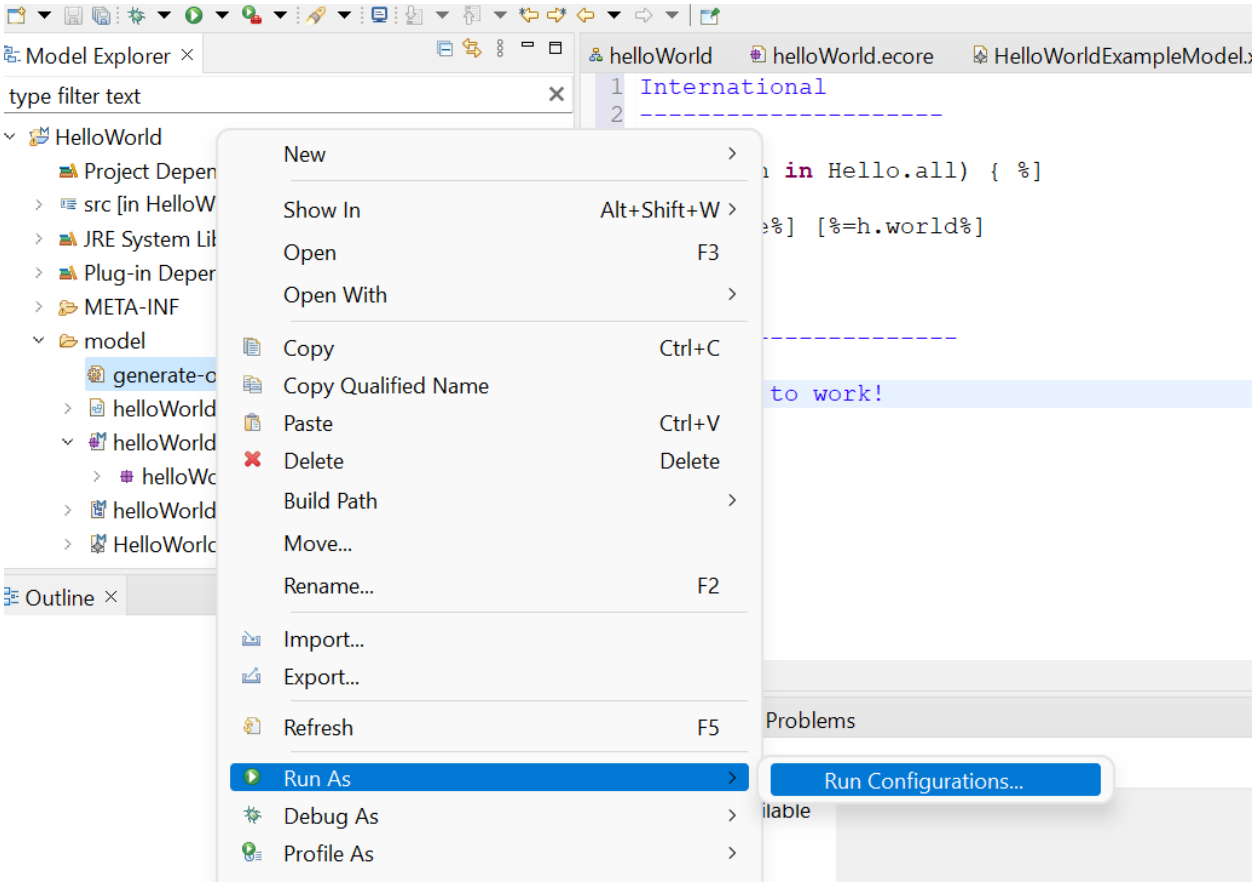
Give a name to the file you want to store the egl code (e.g. generate-output.egl) and press **Finish**:



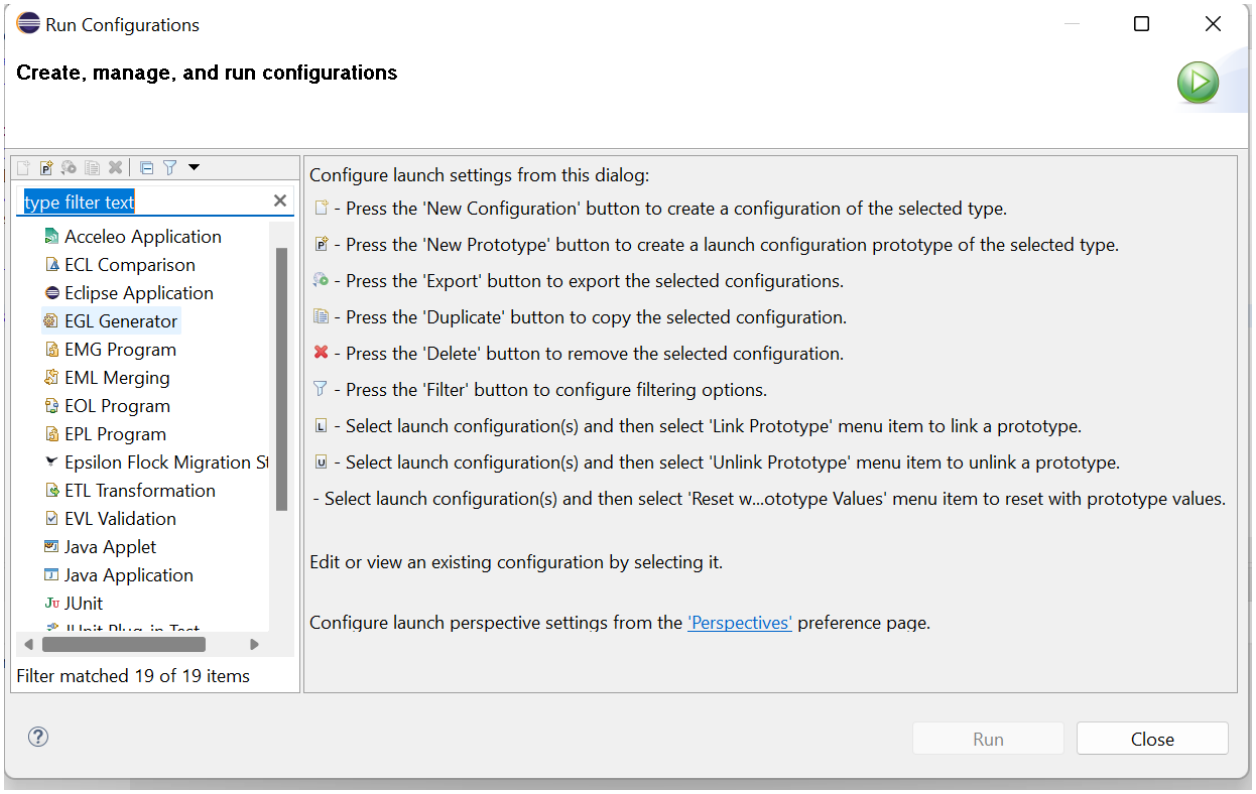
You can write the following code:



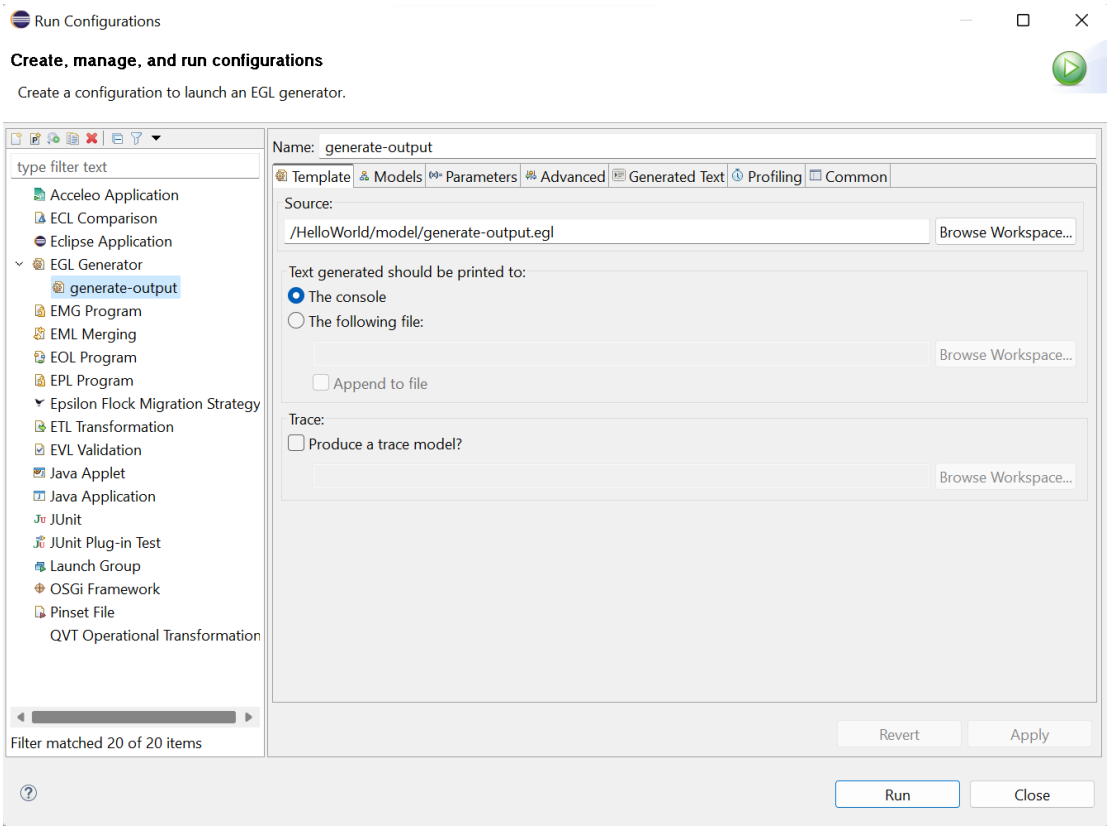
Finally lets do the last steps to run it. Start by right-clicking on the egl file on the model explorer tree. Next, follow **Run As -> Run Configurations**:



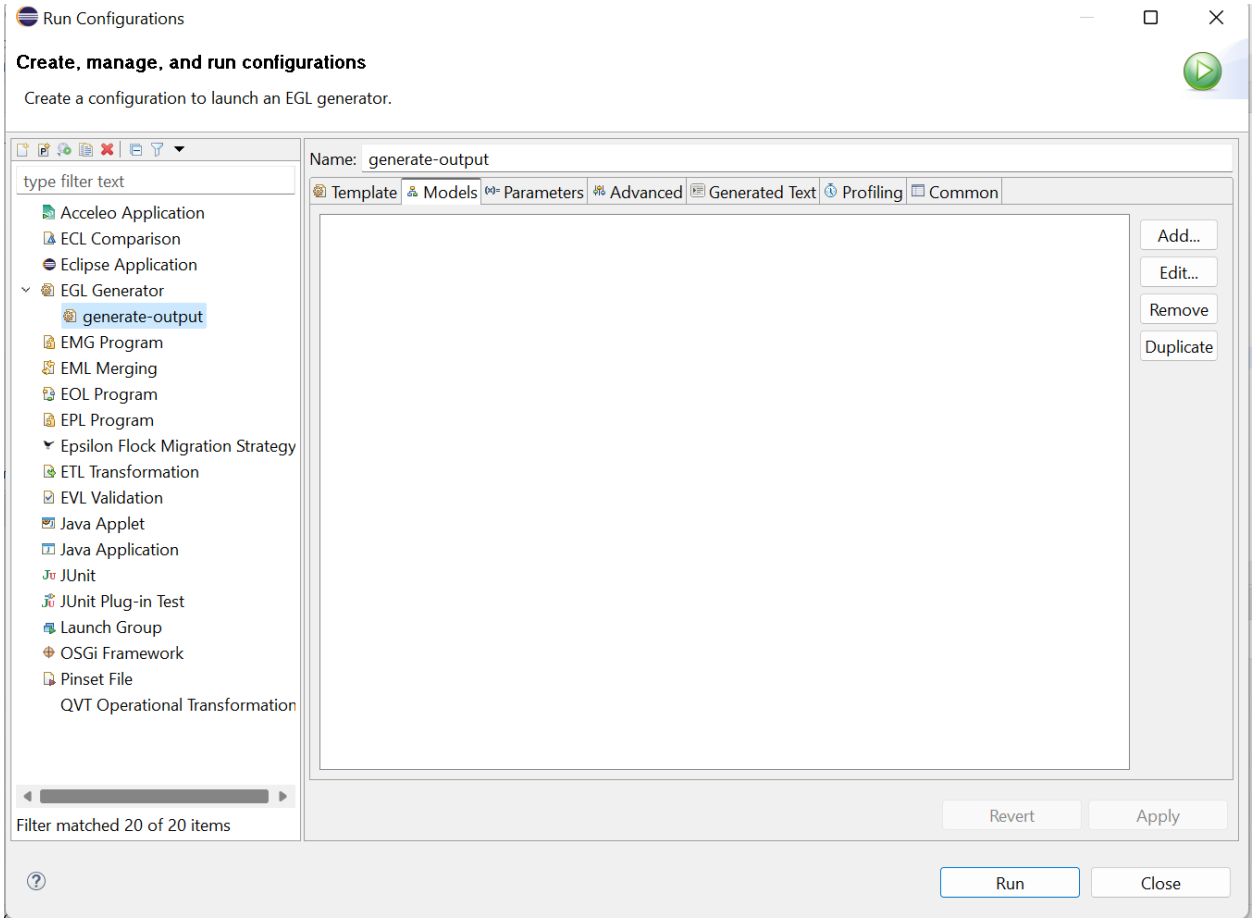
Then double click EGL Generator:



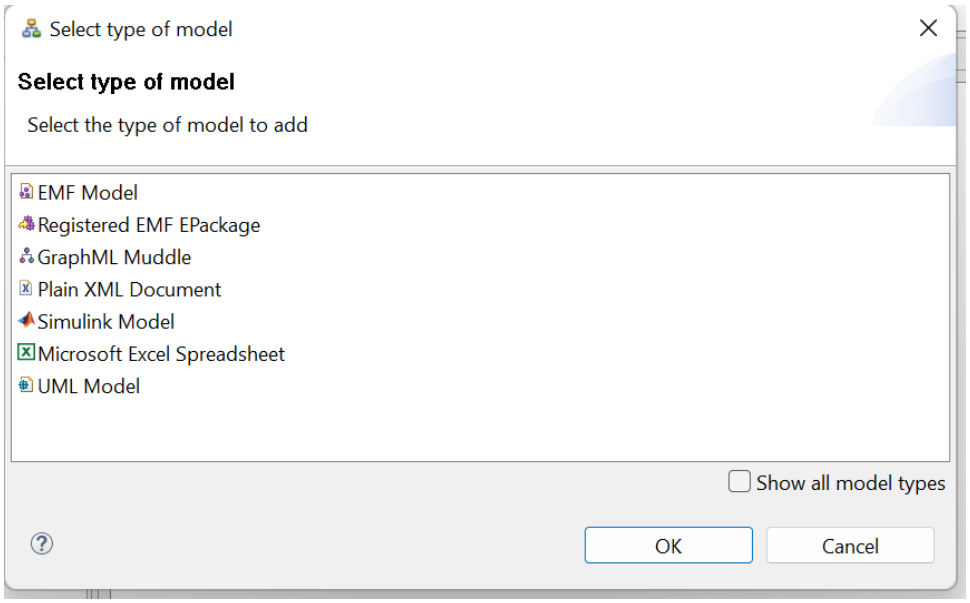
And you should see:



Select the **Models** tab and click on the "**Add…**" button, so that we can now associate the input instance model so that the egl code can manipulate it:

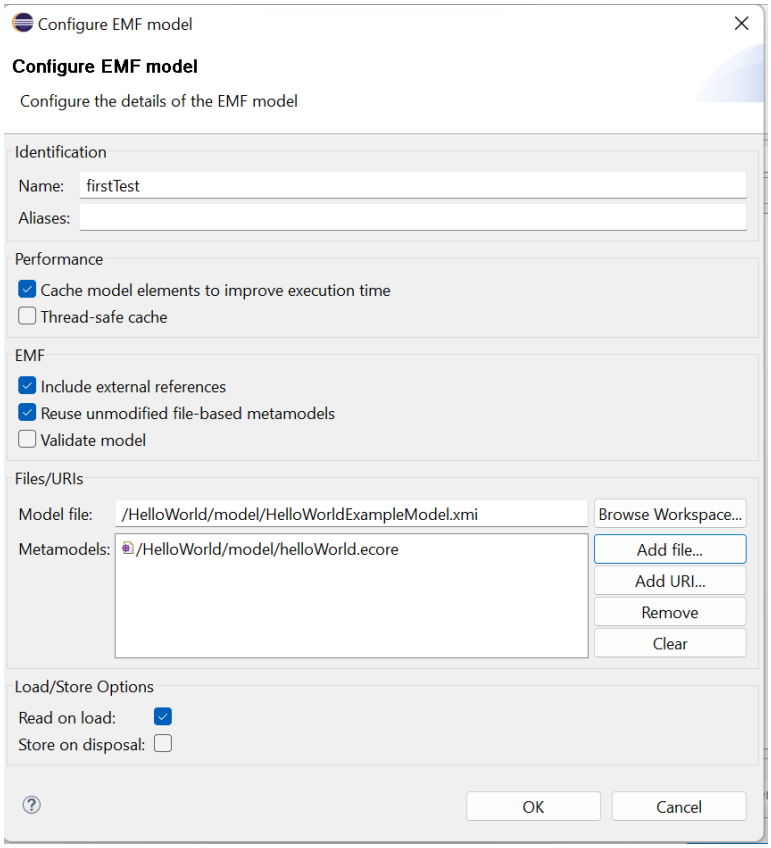


The next step is to select the option **EMF Model**:

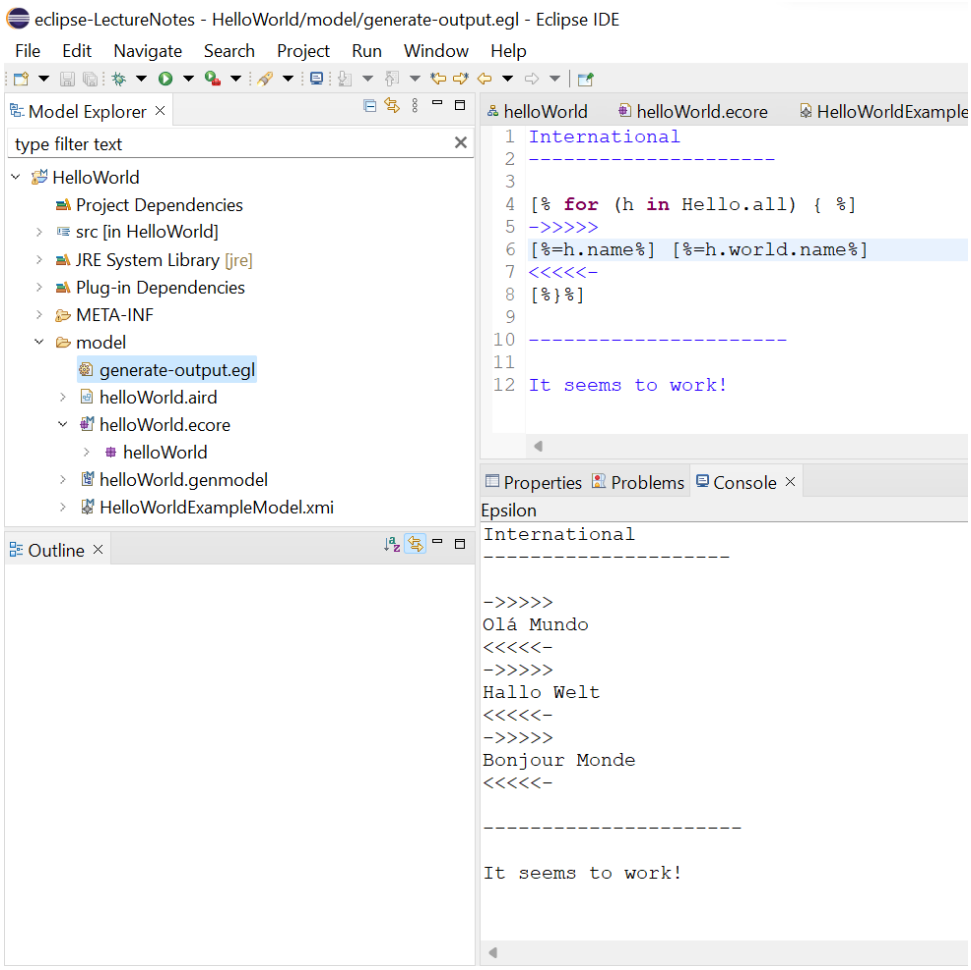


Give a name (e.g. firstTest) and then press the Browse Workspace for finding the input file, that in this case we named it **HelloWorldExampleModel.xmi** . It will find by itself some metamodels. Remove them and press "**Add File…**" and look for the **helloWorld.ecore** file (our metamodel). Before finishing unselect

the “**Store on disposal**” checkbox to prevent that some wrong manipulation by egl can corrupt the **HelloWorlExampleModel.xmi** file. Everything should look like this:



You just need to press “**OK**” then “**Apply**” and then “**Run**”. The result should be your first Model-Driven Hello World in the console:



Congratulations!