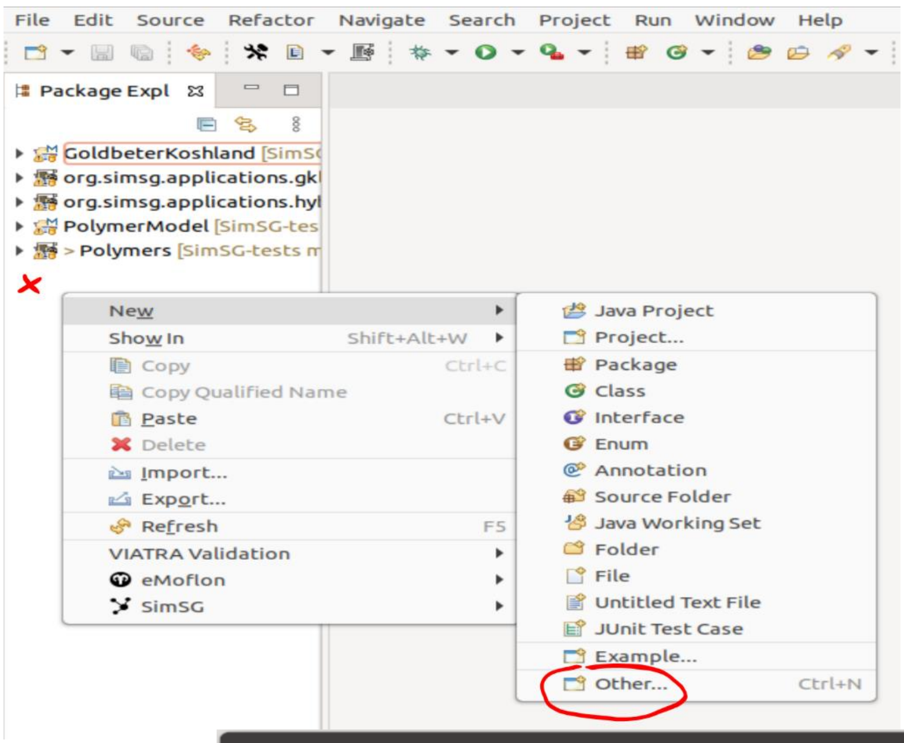
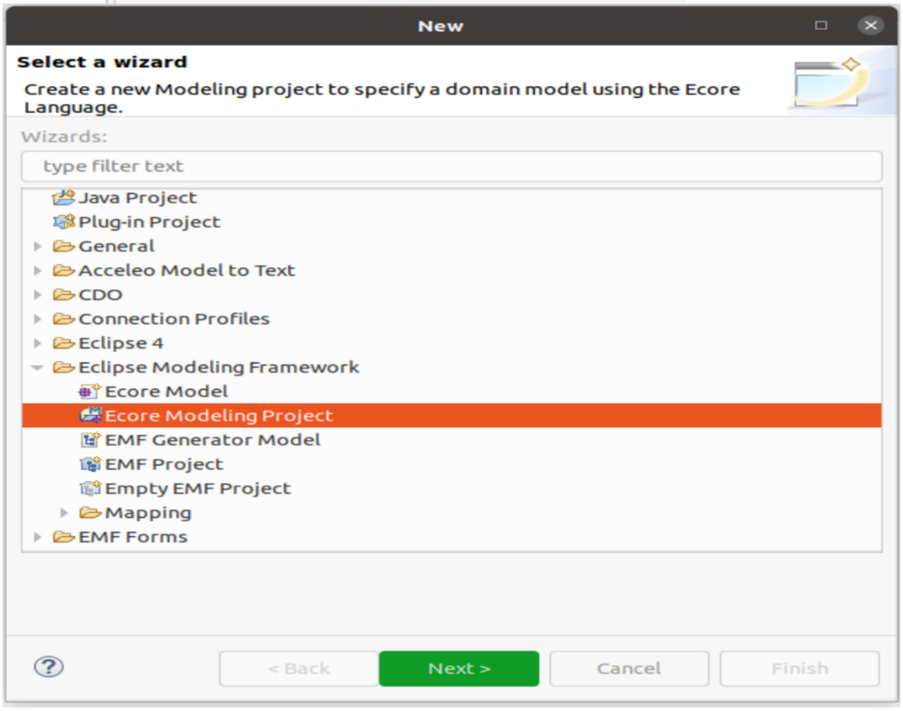
Ecore-Modeling for eMoflon GT-Projects

1. Create a new Ecore Modeling Project in your (dynamic) workspace by right-clicking in the package explorer: New -> Other... -> Ecore Modeling Project- > Next

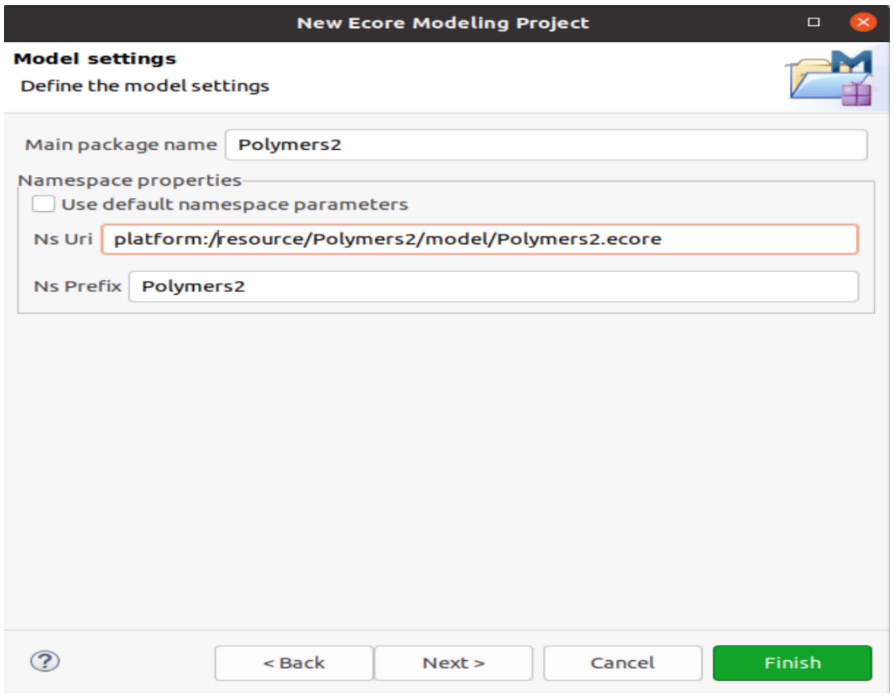




Create a new Modeling project to specify a domain model using the Ecore Language.

2. Select a Name for your Project and, if necessary, set your custom project location. Please do not click on “Finish” but click on “Next” for more options.

3. We recommend naming your package the same as your project and set namespace parameters manually. For example, if your project is named Polymers2, you set your Ns URI to the following value: platform:/resource/Polymers2/model/Polymers2.ecore



New Ecore Modeling Project

Model settings

Define the model settings

Main package name: Polymers2

Namespace properties

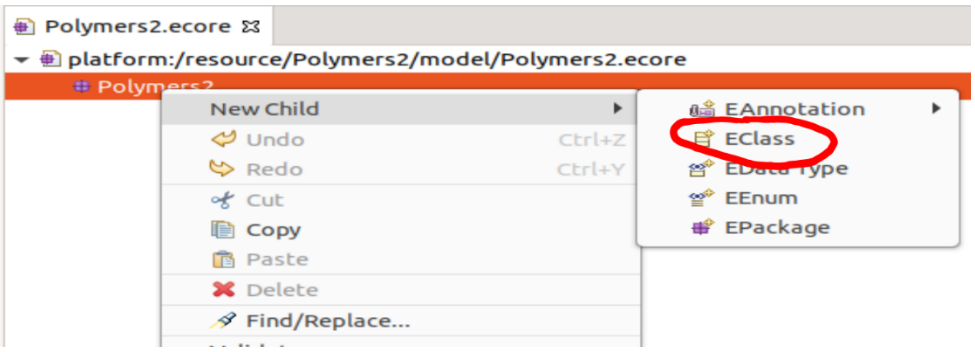
☐ Use default namespace parameters

Ns Uri: platform:/resource/Polymers2/model/Polymers2.ecore

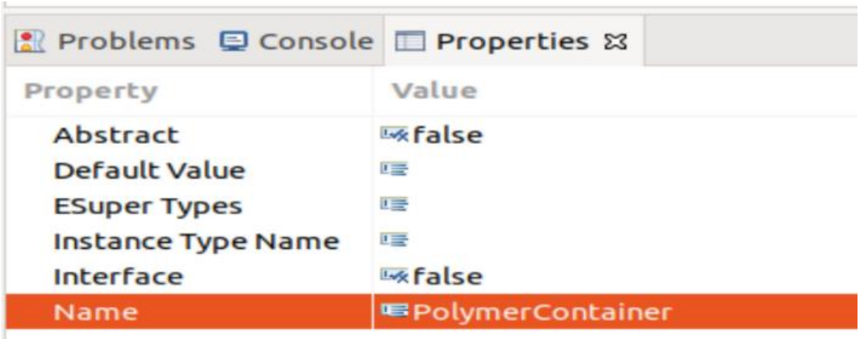
Ns Prefix: Polymers2

4. Press on “Finish” and don’t open the recommend modeling perspective (it’s useless anyways), by pressing “No”. Close the popped-up *.aird-file and discard changes. To create and edit your metamodel, it’s easier (and a lot less buggy) to use the *.ecore-file.
5. **Example: Creating a new metamodel**
   a. An important EMF/Ecore rule is, that every element must be contained within the containment hierarchy of a model. Meaning, that any element must have at least one reference with the containment flag (set to true) pointing to it. We recommend creating a container class as a top-level element in your diagram, which makes model instantiation easier and prevents chaos in your models. To keep it simple every element of your model should be contained within the container element, but can be contained within other elements if the need arises. But be careful, not every reference must necessarily be a containment reference. In fact, making every reference in your model a containment reference will most likely lead to undesired behaviour, since the behaviour of containment references differs from the behaviour of non-containment references. If you change a containment reference to an element, it will be removed from the original containment and moved to the new container. This will automatically lead to deleted containment edges. If you don’t want that, simply use a “normal” reference.

b. Add a new Class to your metamodel by right-clicking on your model: New Child -> EClass



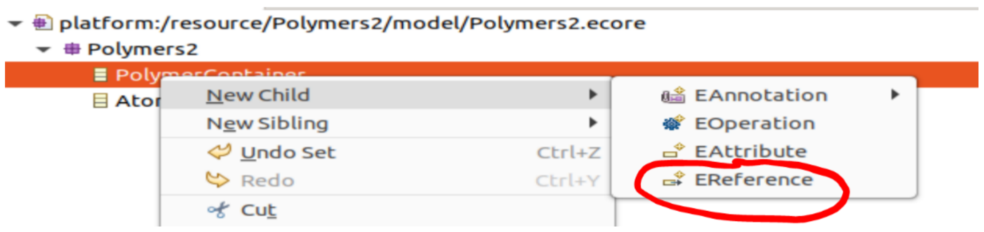
c. Name your class by editing the entry in the “Properties” tab.



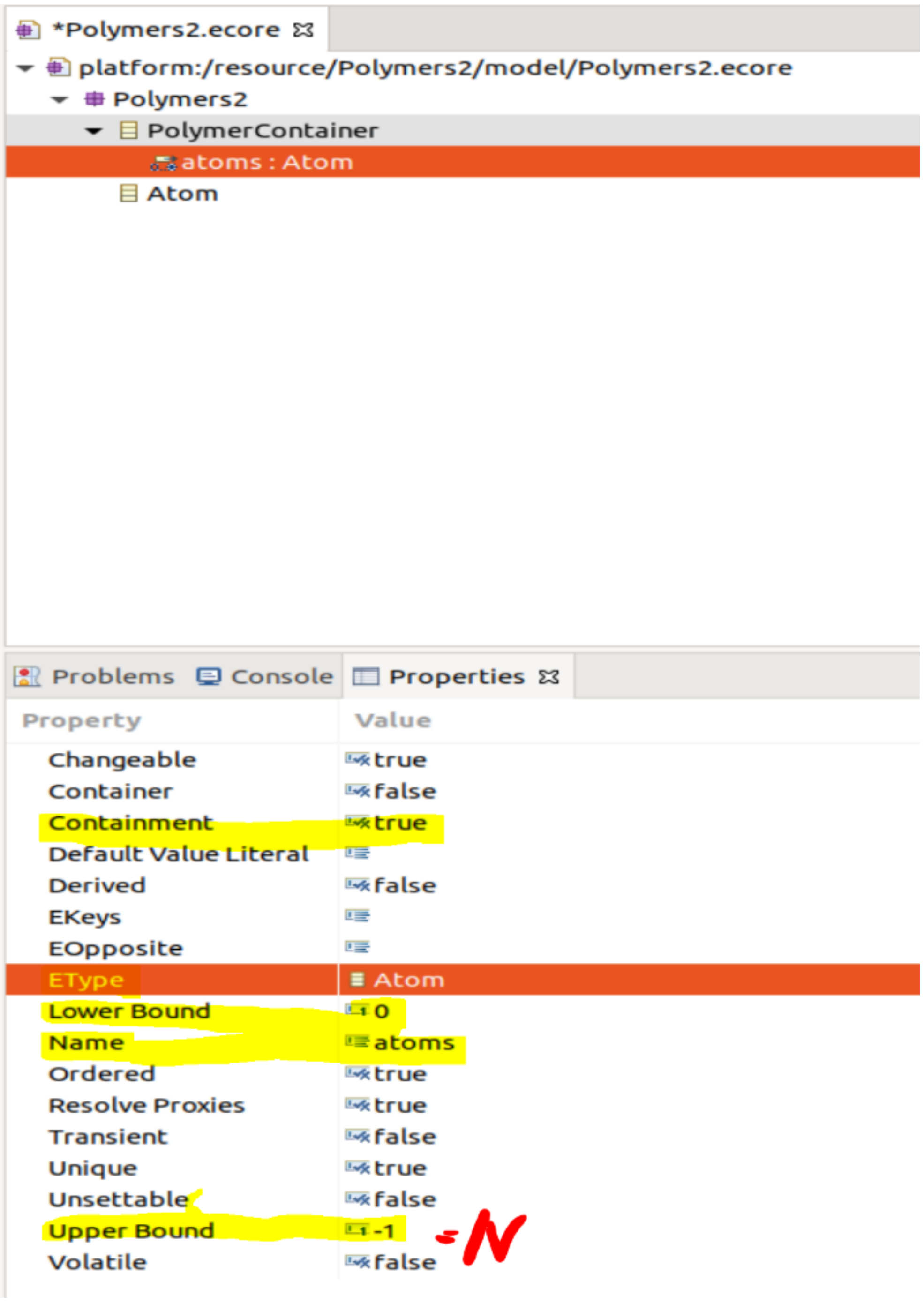
Property	Value
Abstract	false
Default Value	
ESuper Types	
Instance Type Name	
Interface	false
Name	PolymerContainer

d. Add further classes to your model by following Steps b) to c) and create containment references from your containment class to your added classes.

e. (Containment)References can be created by right-clicking on a class: New Child -> EReference.

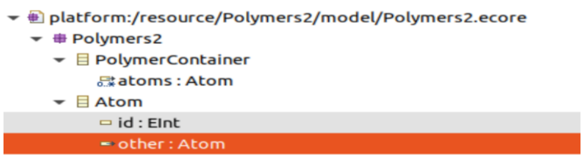


Set the name (type), target type, multiplicty and containment flag of your reference in the "Properties" tab.

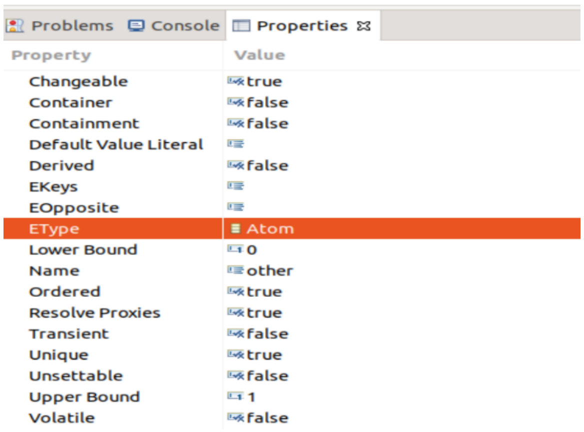


Property	Value
Changeable	true
Container	false
Containment	true
Default Value Literal	
Derived	false
EKeys	
EOpposite	
EType	Atom
Lower Bound	0
Name	atoms
Ordered	true
Resolve Proxies	true
Transient	false
Unique	true
Unsettable	false
Upper Bound	-1 =N
Volatile	false

f. You may add other features to classes in the same fashion, e.g., EAttributes or non-containment EReferences.



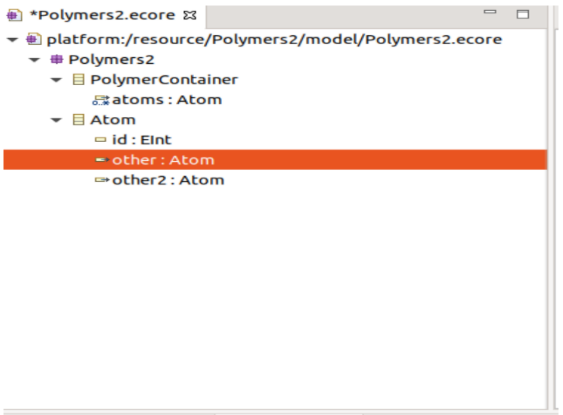
```
▾ platform:/resource/Polymers2/model/Polymers2.ecore
  ▾ Polymers2
    ▾ PolymerContainer
        atoms : Atom
    ▾ Atom
        id : EInt
        other : Atom
```



Problems | Console | Properties

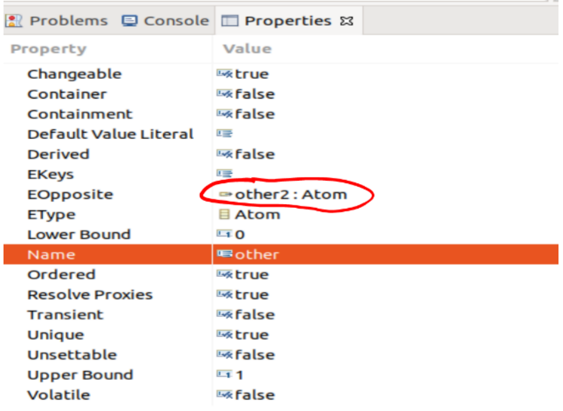
Property	Value
Changeable	true
Container	false
Containment	false
Default Value Literal	
Derived	false
EKeys	
EOpposite	
EType	Atom
Lower Bound	0
Name	other
Ordered	true
Resolve Proxies	true
Transient	false
Unique	true
Unsettable	false
Upper Bound	1
Volatile	false

g. If you want to set bidirectional references, you have to create two references and add an entry to EOpposite, to let emf know that these references are each other's opposites. If you set the EOpposite of one reference, the other reference's EOpposite will be set automatically.



*Polymers2.ecore

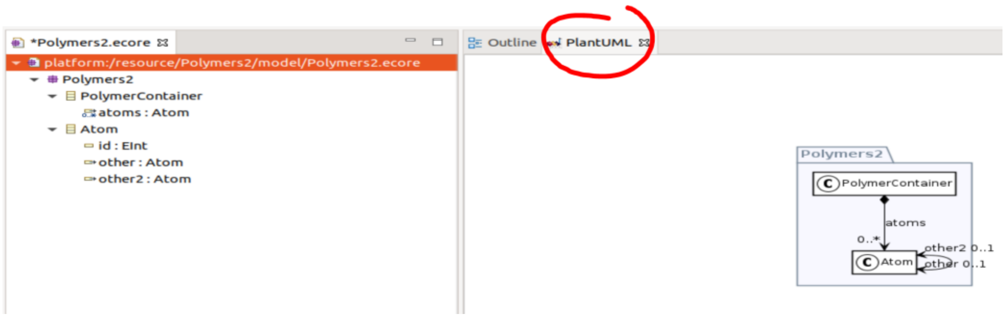
```
▾ platform:/resource/Polymers2/model/Polymers2.ecore
  ▾ Polymers2
    ▾ PolymerContainer
        atoms : Atom
    ▾ Atom
        id : EInt
        other : Atom
        other2 : Atom
```



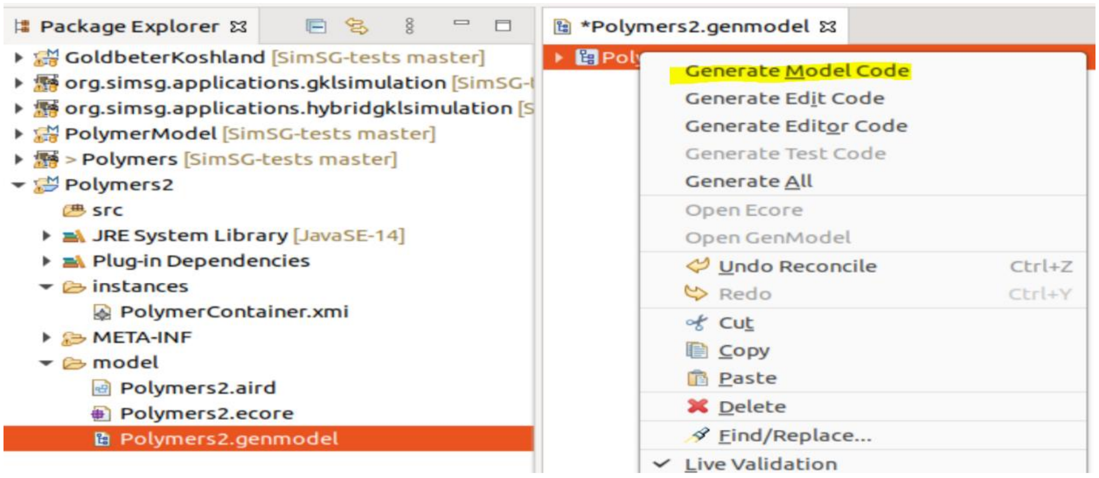
Problems | Console | Properties

Property	Value
Changeable	true
Container	false
Containment	false
Default Value Literal	
Derived	false
EKeys	
EOpposite	other2 : Atom
EType	Atom
Lower Bound	0
Name	other
Ordered	true
Resolve Proxies	true
Transient	false
Unique	true
Unsettable	false
Upper Bound	1
Volatile	false

h. Don’t forget to save your metamodel (Ctrl+s). You can look at the class diagram of your finished metamodel using the PlantUML view.

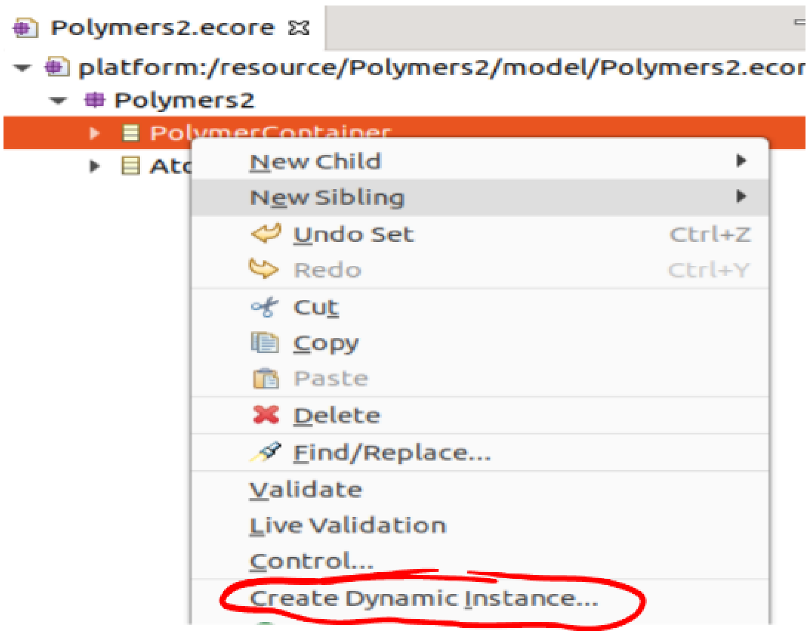


6. With the modeling step being complete you may now generate java code from the metamodel. Simply open your *.genmodel-file and in the editor: right-click -> Generate Model Code.

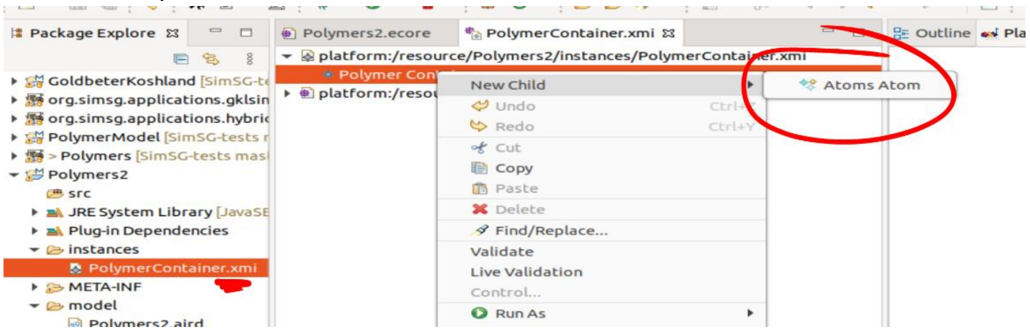


7. **Example: Instantiate a model conforming to your metamodel (via UI)**

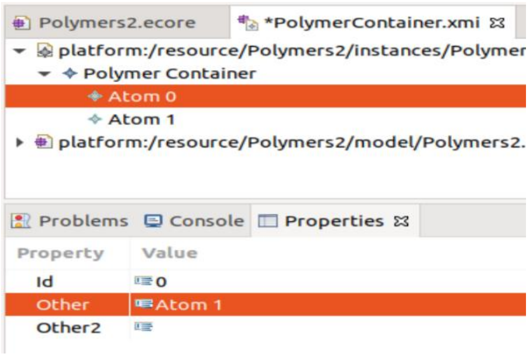
a. By right-clicking on your top-level container element, you can create a dynamic instance (aka a new empty model conforming to the metamodel) and store it to location of your choice.



b. Using the graphical editor, you can now edit the model by opening the *.xmi-file you just created. By right-clicking on the container element you can add new child elements to your model.



c. References between elements may be added by editing the "Properties" tab of each element.

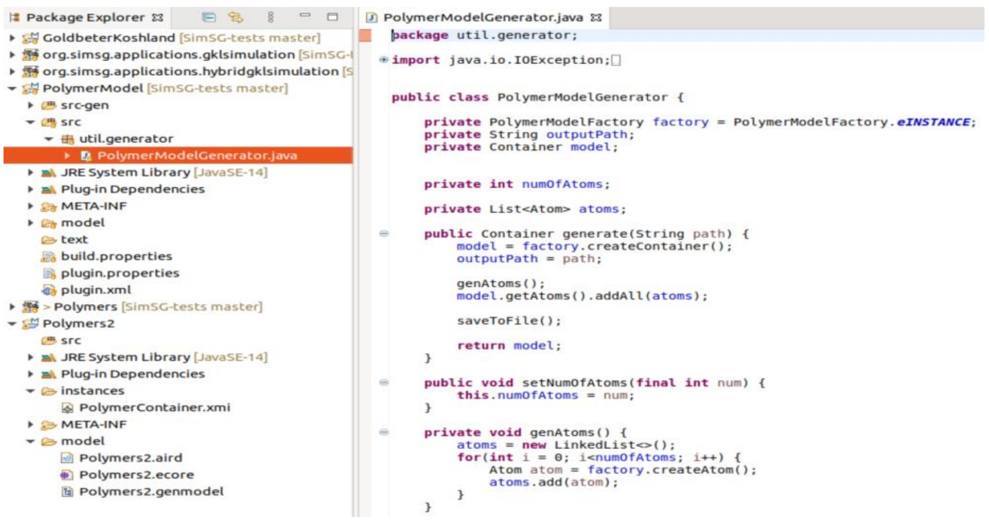

Polymers2.ecore
*PolymerContainer.xmi
platform:/resource/Polymers2/instances/Polymer
Polymer Container
Atom 0
Atom 1
platform:/resource/Polymers2/model/Polymers2.
Problems
Console
Properties

Property	Value
Id	0
Other	Atom 1
Other2	

8. **Example: Instantiate a model conforming to your metamodel (programmatically)**

a. As you might notice, creating large models using the graphical editor is a tedious process, hence, we recommend writing a simple model generator by hand.

b. For starters, just have a look at the PolymerModelGenerator.java in the PolymerModel project. The easiest way to implement a generator yourself, would be to copy & paste this class and modify it to suit your needs.


Package Explorer
GoldbeterKoshland [SimSG-tests master]
org.simsg.applications.gklsimulation [SimSG-t
org.simsg.applications.hybridgklsimulation [S
PolymerModel [SimSG-tests master]
src-gen
src
util.generator
PolymerModelGenerator.java
JRE System Library [JavaSE-14]
Plug-in Dependencies
META-INF
model
text
build.properties
plugin.properties
plugin.xml
> Polymers [SimSG-tests master]
Polymers2
src
JRE System Library [JavaSE-14]
Plug-in Dependencies
instances
PolymerContainer.xmi
META-INF
model
Polymers2.aird
Polymers2.ecore
Polymers2.genmodel
PolymerModelGenerator.java

```java
package util.generator;

import java.io.IOException;

public class PolymerModelGenerator {

    private PolymerModelFactory factory = PolymerModelFactory.eINSTANCE;
    private String outputPath;
    private Container model;


    private int numOfAtoms;

    private List<Atom> atoms;

    public Container generate(String path) {
        model = factory.createContainer();
        outputPath = path;

        genAtoms();
        model.getAtoms().addAll(atoms);

        saveToFile();

        return model;
    }

    public void setNumOfAtoms(final int num) {
        this.numOfAtoms = num;
    }

    private void genAtoms() {
        atoms = new LinkedList<>();
        for(int i = 0; i<numOfAtoms; i++) {
            Atom atom = factory.createAtom();
            atoms.add(atom);
        }
    }
```