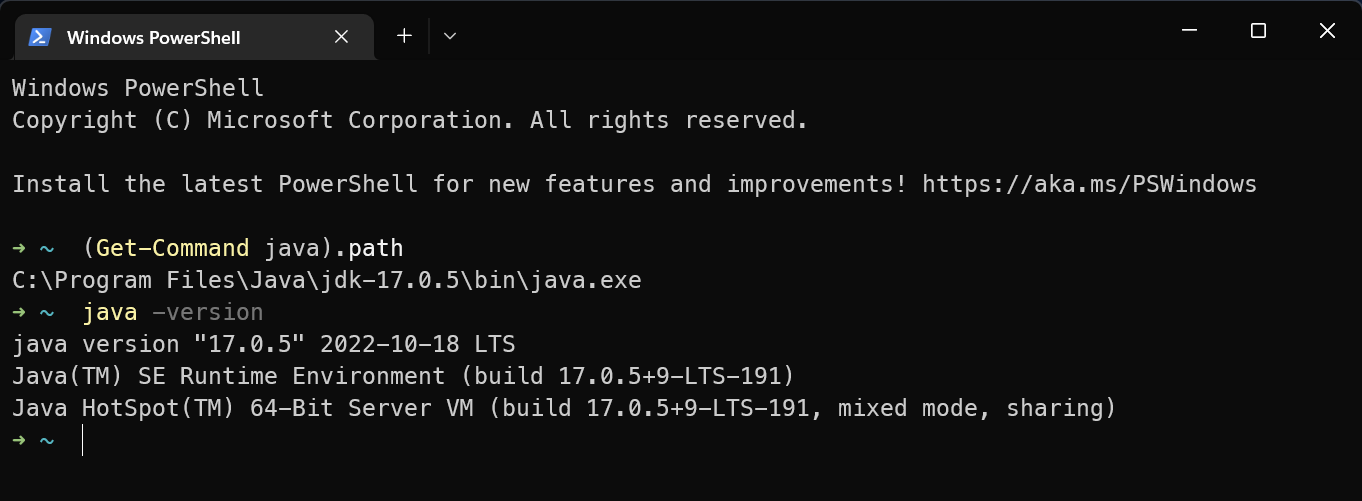
# LIDL Code Editor developer manual

## Configuring the development environment

### Download Java

In this manual, we use the Java environment information as shown below.

Specifically, the Java version is 17.0.5, and the path is the default installation path of Java.



### Unzip the source code archive

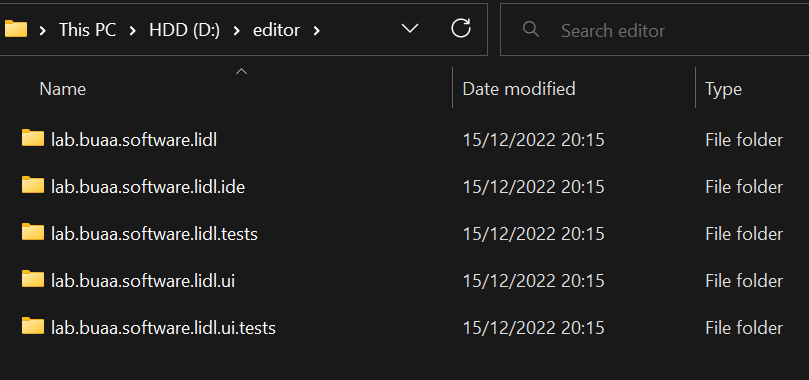
Path: /LIDL MEC Toolkit/source code/Code Editor/src.zip

GitHub: /LIDL-MEC-Toolkit/source code/Code Editor/editor/

There is only one editor folder in the root directory of the zip archive, and it contains the entire source code of Code Editor.

Unzip the zip package, and in the following steps, we will set the Eclipse IDE workspace to the editor folder.

In this manual, we extract the source code to the root directory of the D drive directory, as shown in the following figure.



### Download Eclipse IDE

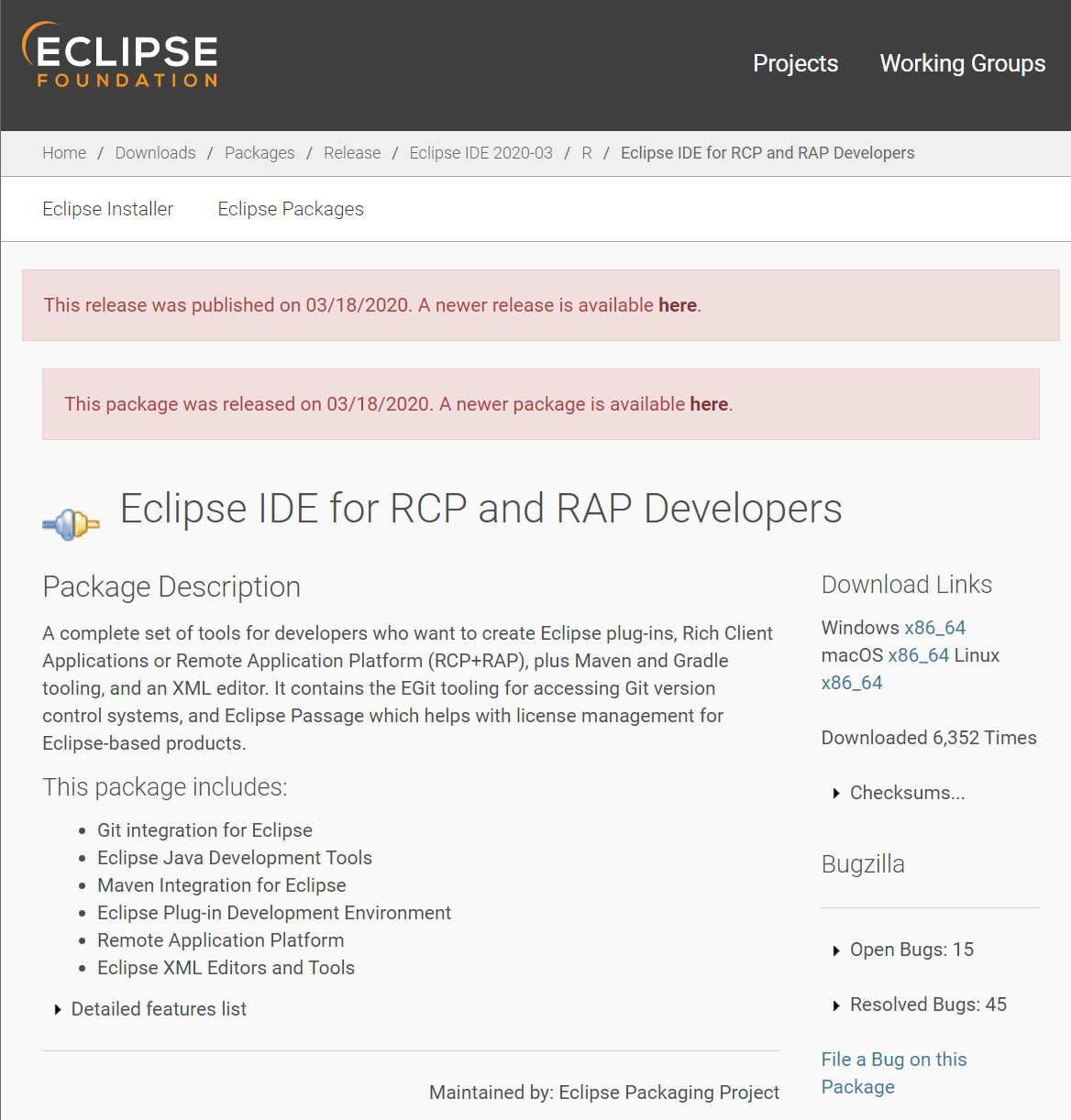
Download **Eclipse IDE for RCP and RAP Developers 20-03** at the following link

<https://www.eclipse.org/downloads/packages/release/2020-03/r/eclipse-ide-rcp-and-rap-developers>

Note: we must download this version, otherwise it will introduce unpredictable problems.

For Windows, the downloaded file is: eclipse-rcp-2020-03-R-win32-x86\_64.zip

The link refers to the following web page, and it provides three versions of download links for Windows, macOS, and Linux operating systems.

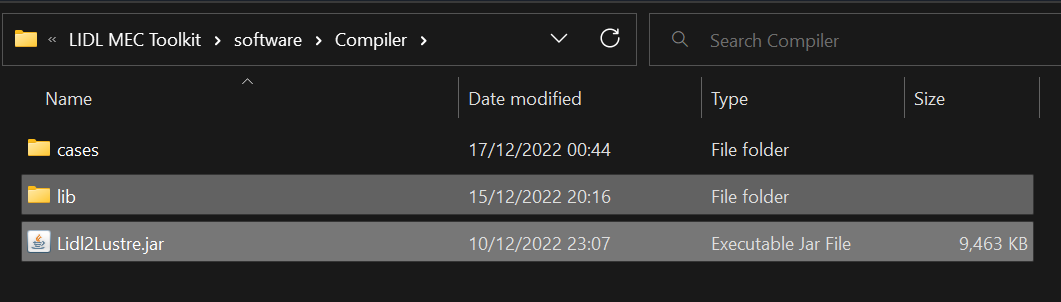


### Add Compiler Plug-in

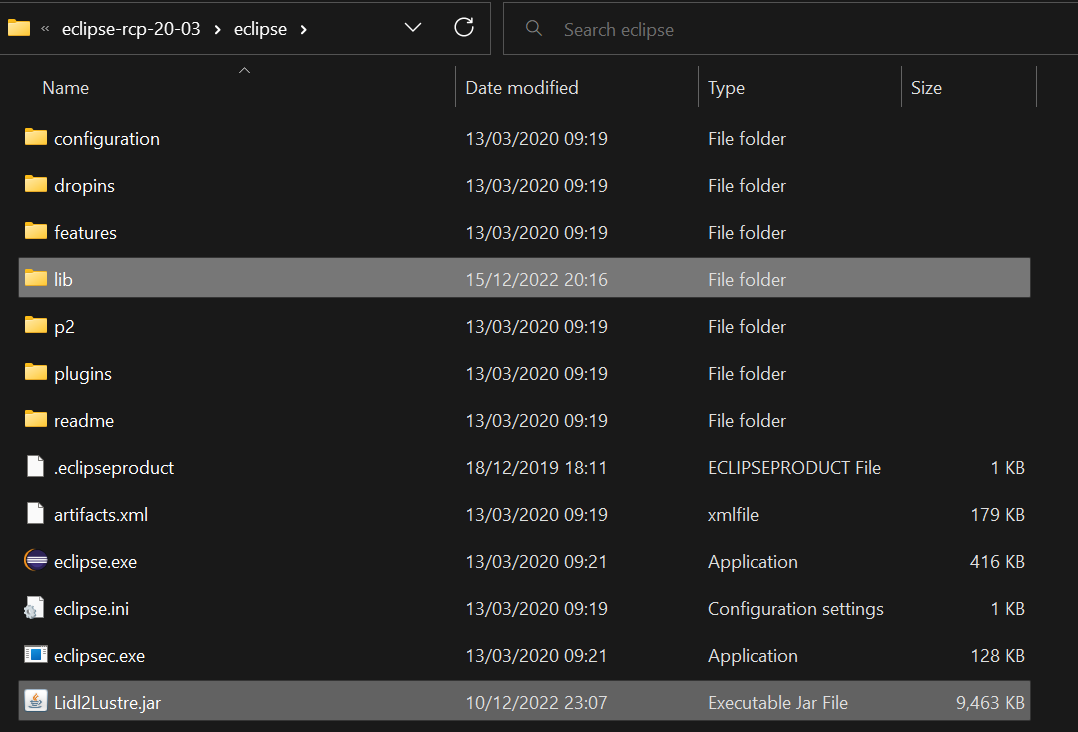
Unzip the downloaded Eclipse IDE package and copy the lib directory and Lidl2Lustre.jar file in the /LIDL MEC Toolkit/software/Compiler/ directory to the eclipse folder in the root directory of the unzipped IDE.

The path on GitHub is: /LIDL-MEC-Toolkit/software/Compiler/

The lib directory and Lidl2Lustre.jar are in the /LIDL MEC Toolkit/software/Compiler/ directory:



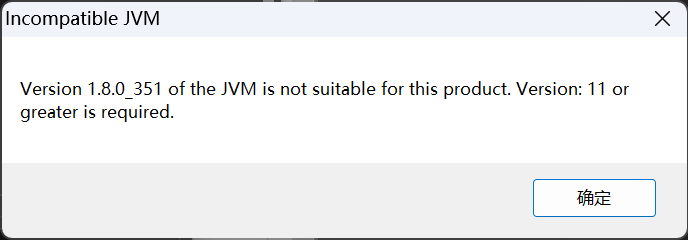
Copy to the eclipse folder:



### Run Eclipse IDE

Run eclipse.exe in the folder eclipse in the root directory of Eclipse IDE.

If you cannot start Eclipse due to the low Java version on the computer (as shown in the figure below),

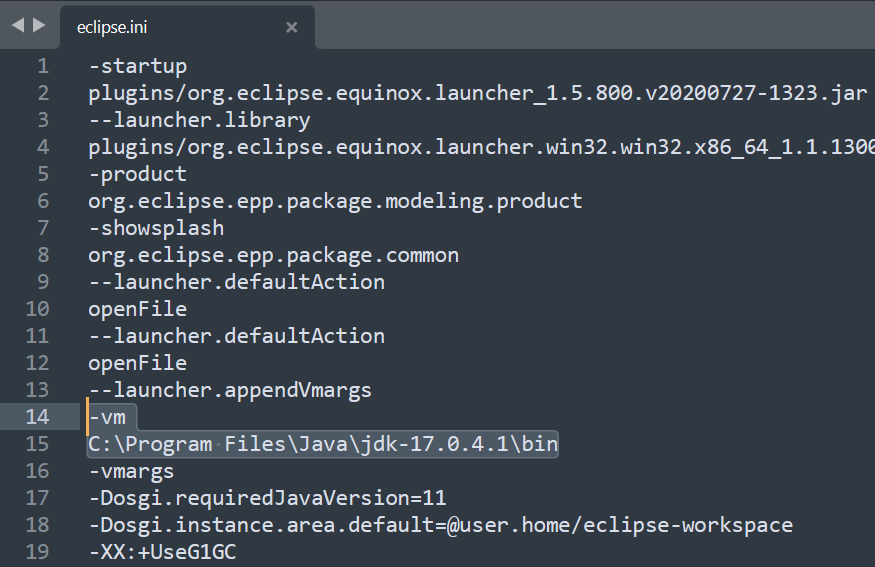


Then edit the eclipse.ini file in the eclipse directory and add two lines before -vmargs.

-vm

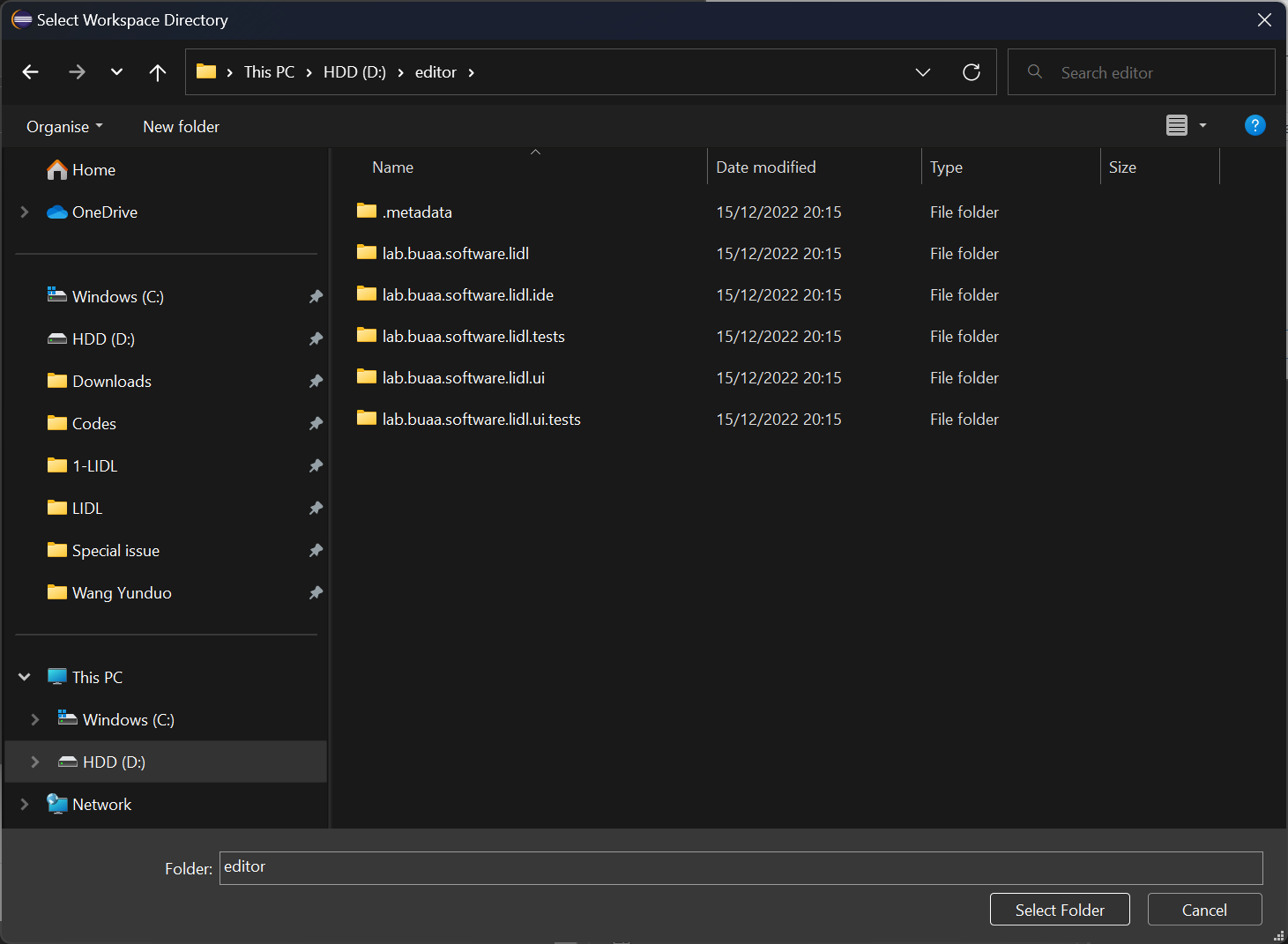
<path to the java executable for Java 11 or higher version>

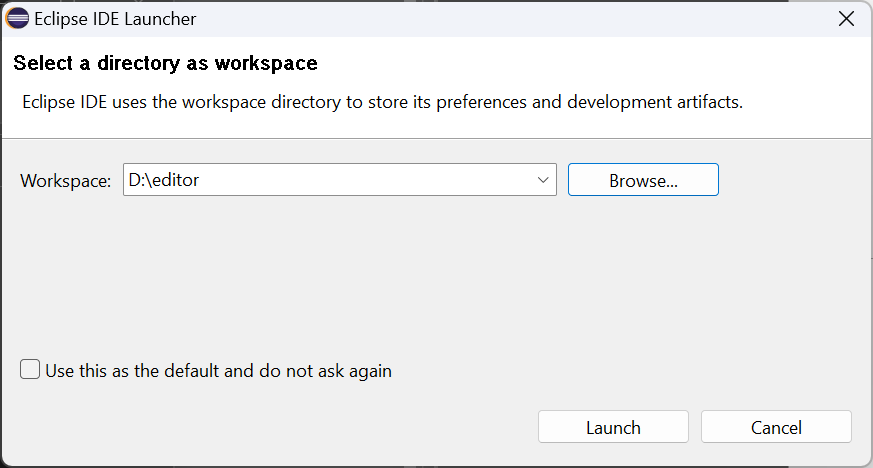
As shown in the figure below



### Configure workspace

Configure Workspace to the editor folder as follows.

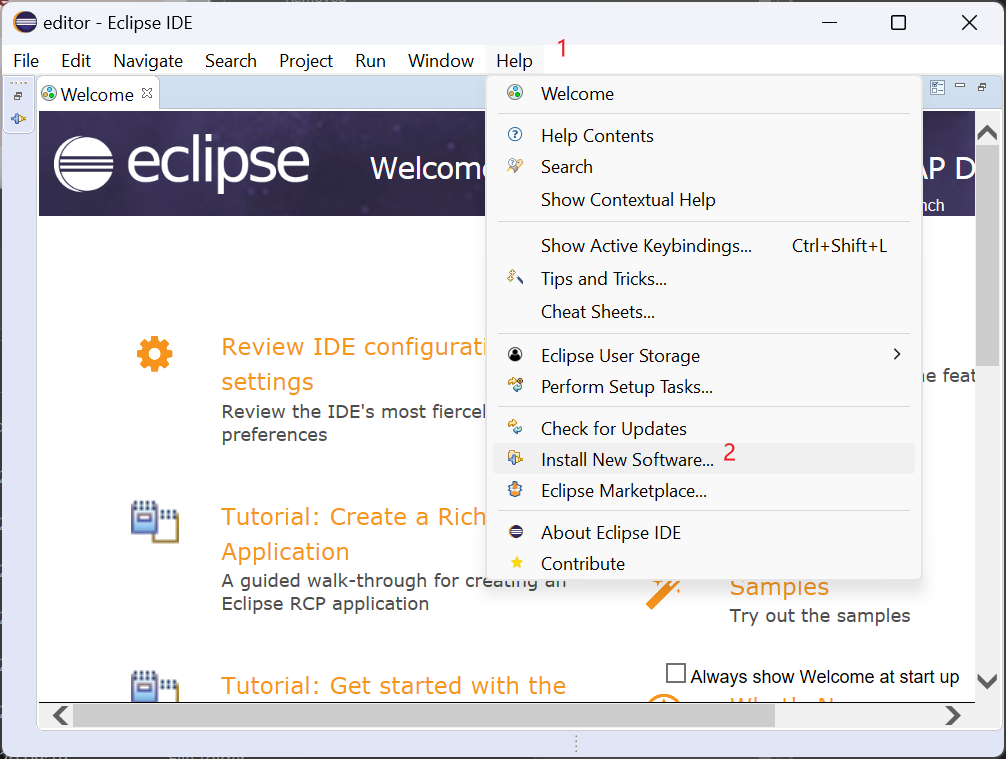




Click Launch to start the Eclipse IDE.

### Install Plug-in

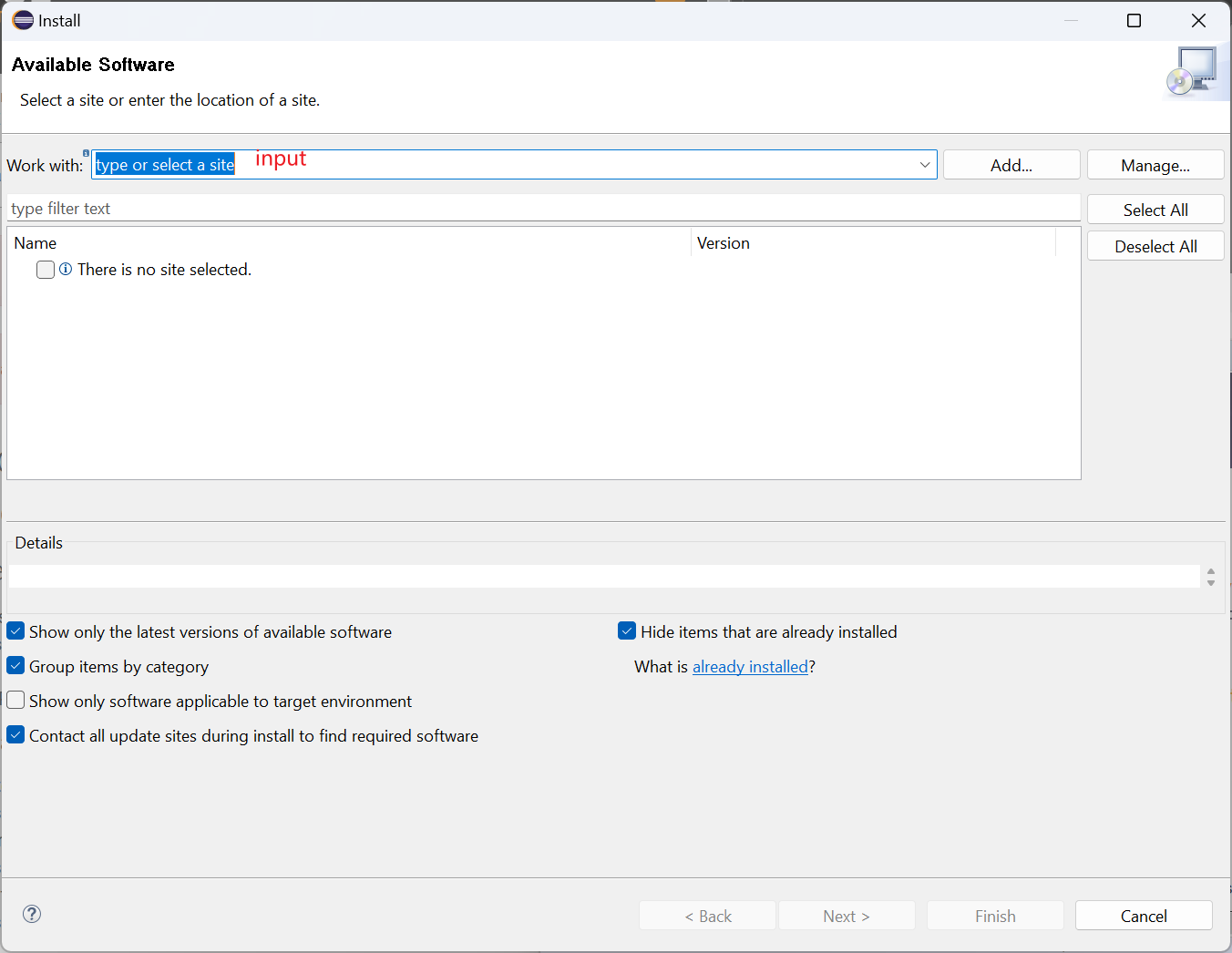
After the Eclipse IDE starts, click Help, then click Install New Software...

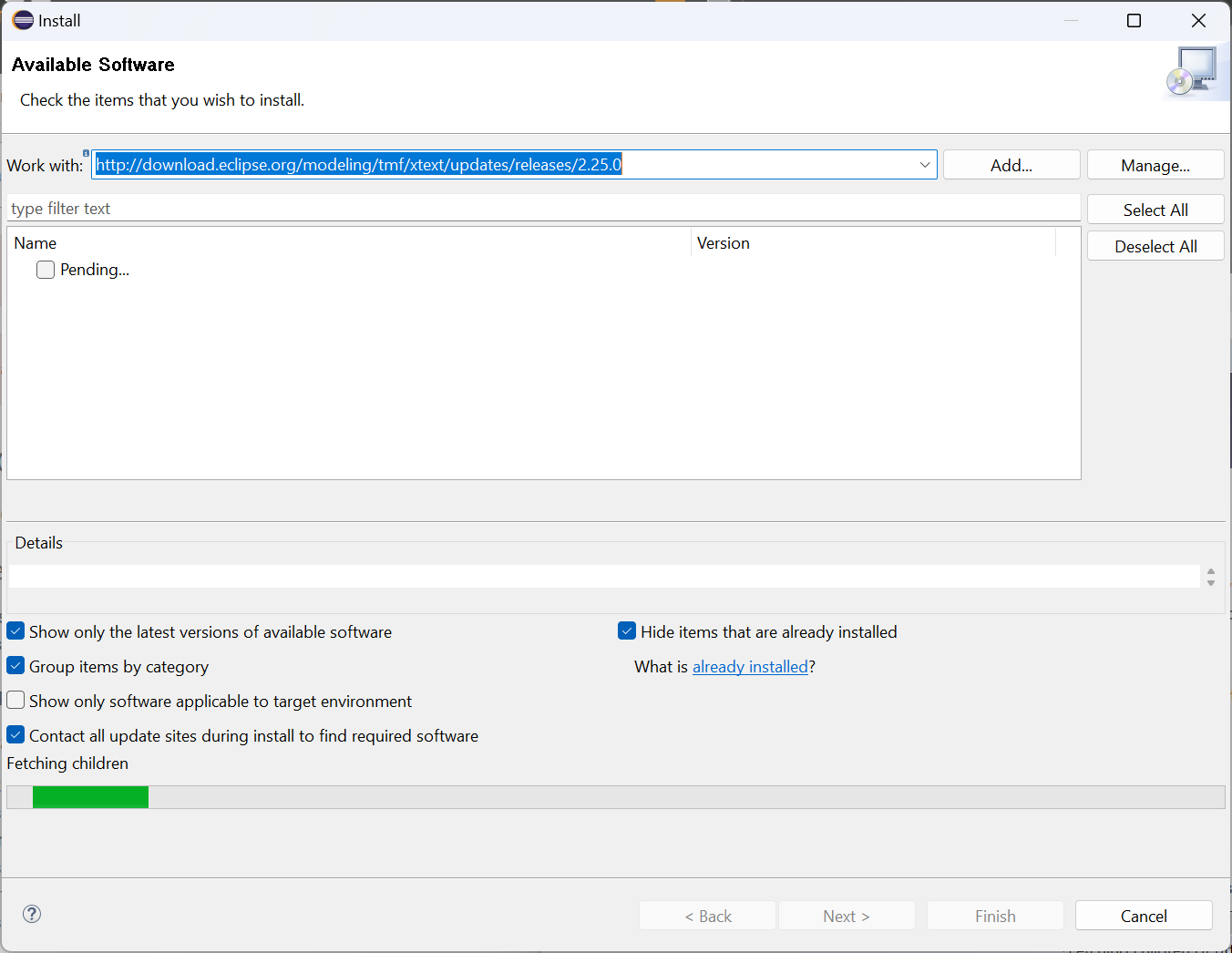


In the pop-up window, input in the input box after Work with:

<http://download.eclipse.org/modeling/tmf/xtext/updates/releases/2.25.0>

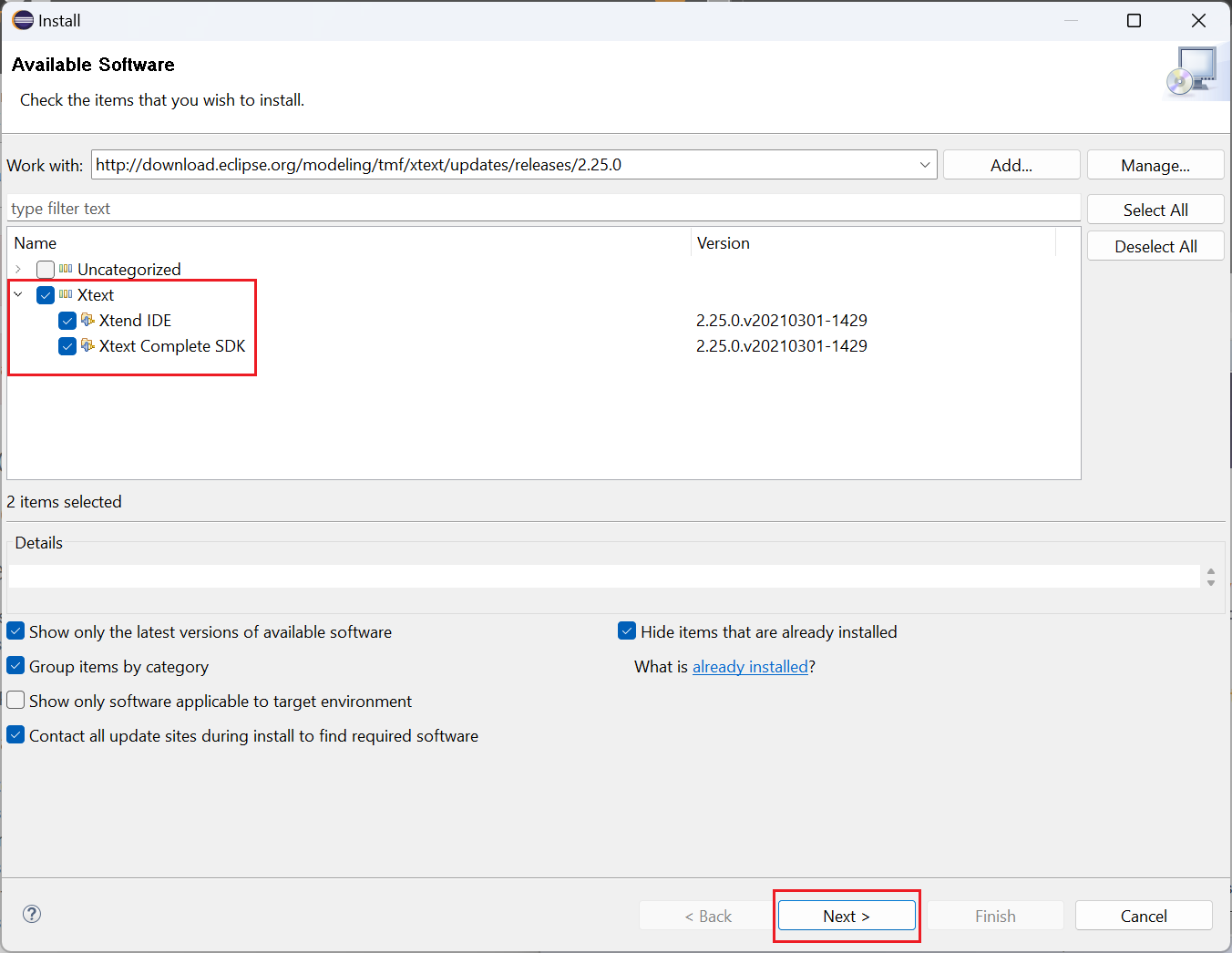
Press the Enter key.



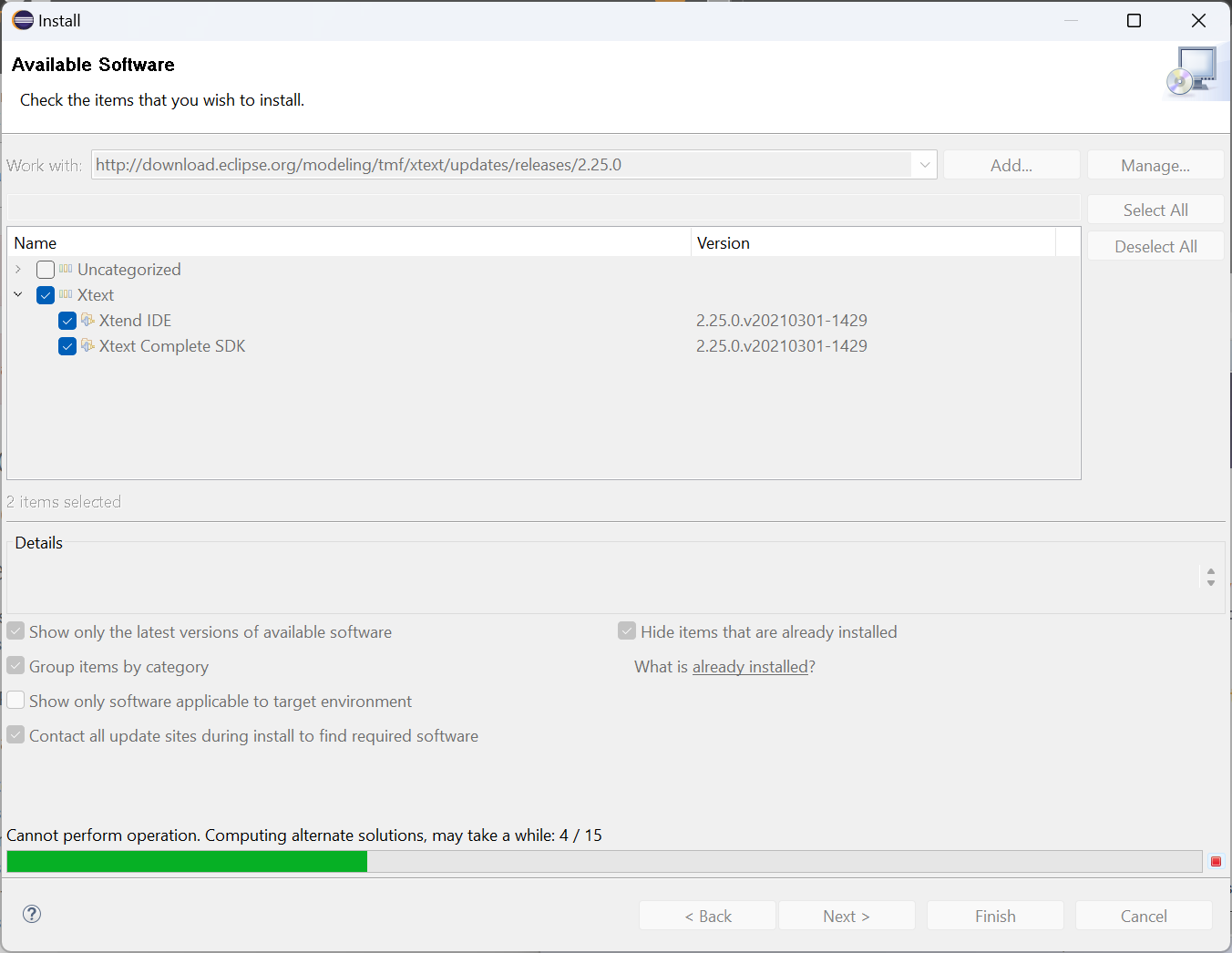


Check Xtext when the loading is complete, as shown in the figure below.

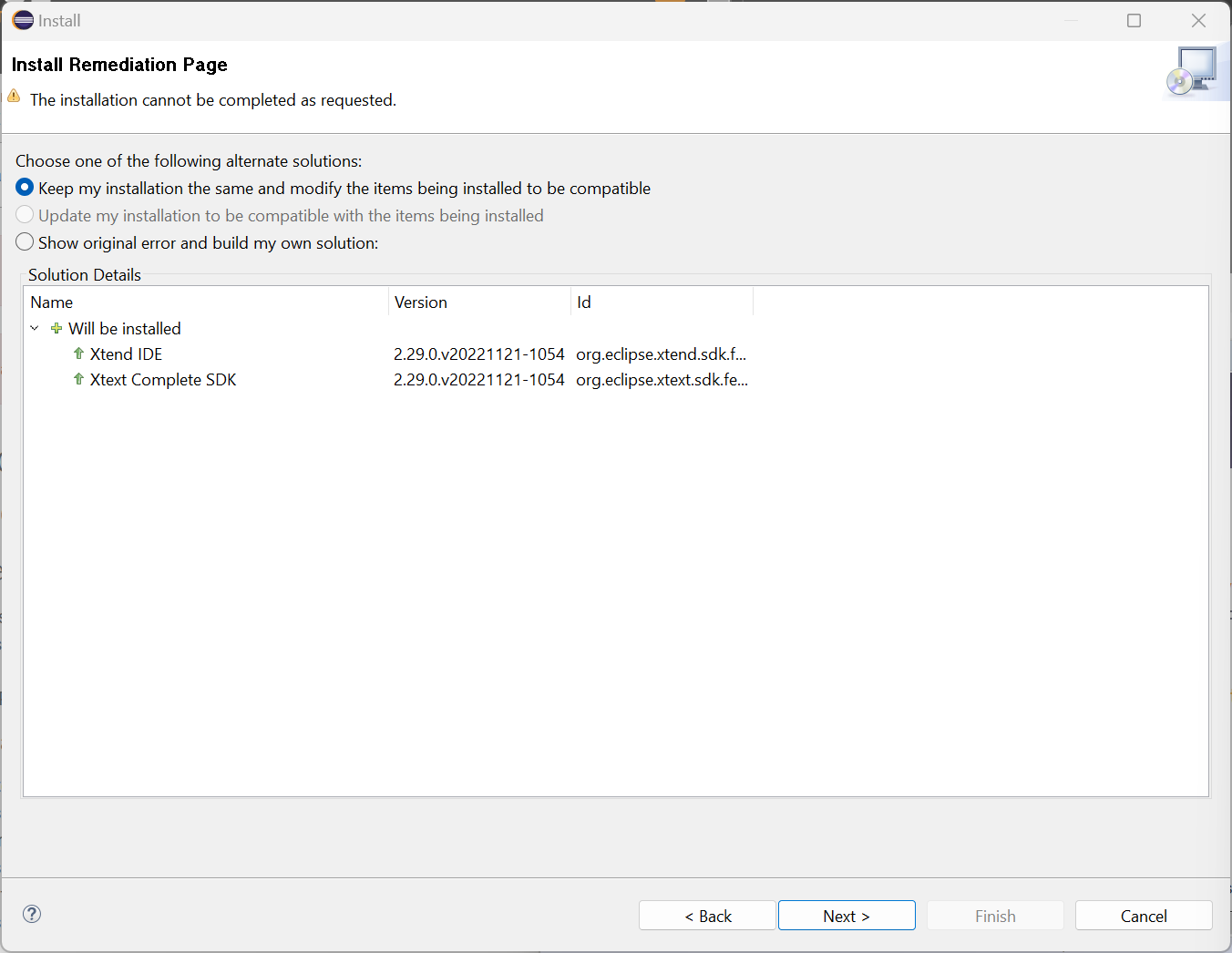
Then click the Next button at the bottom of the page.



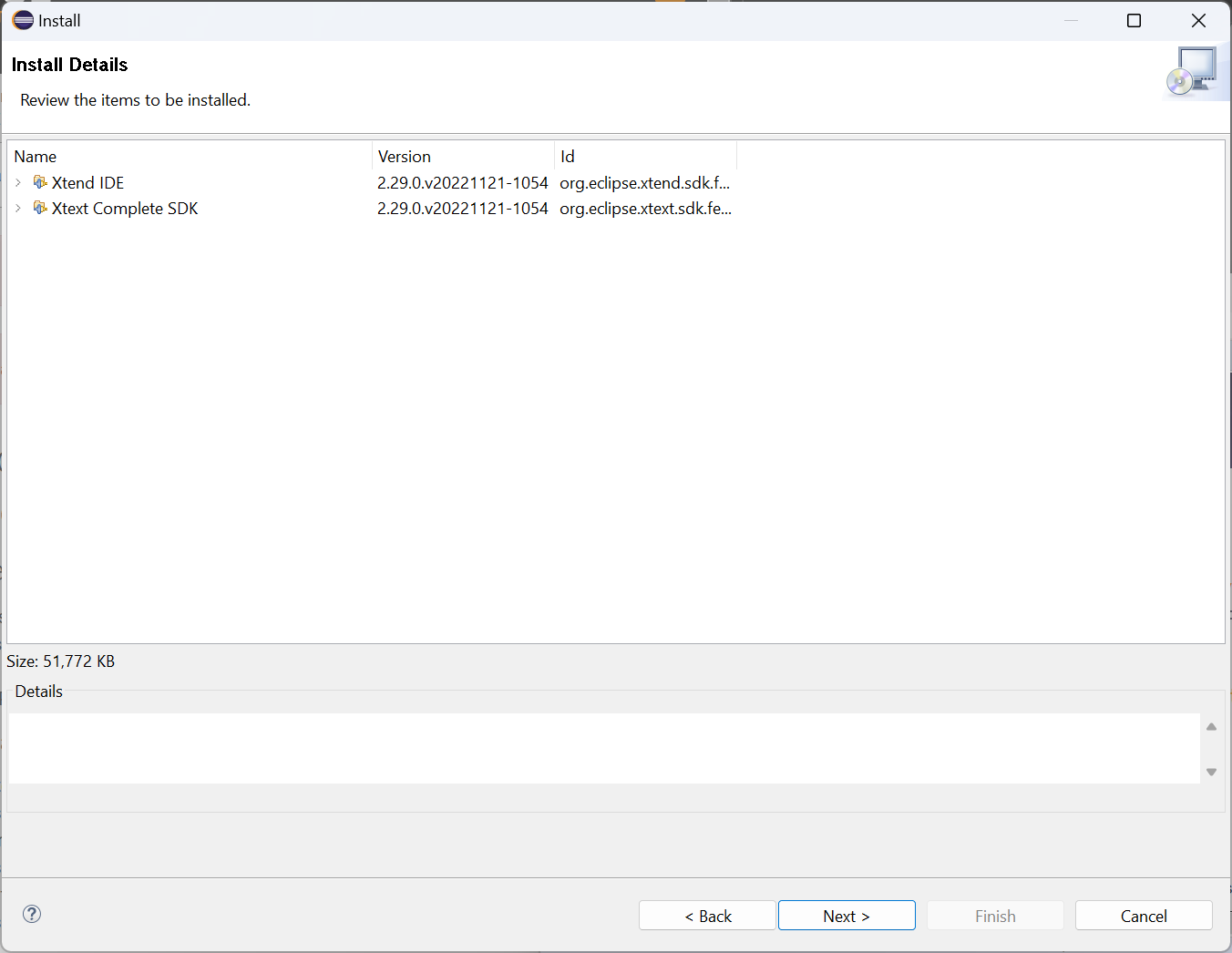
Wait for loading.



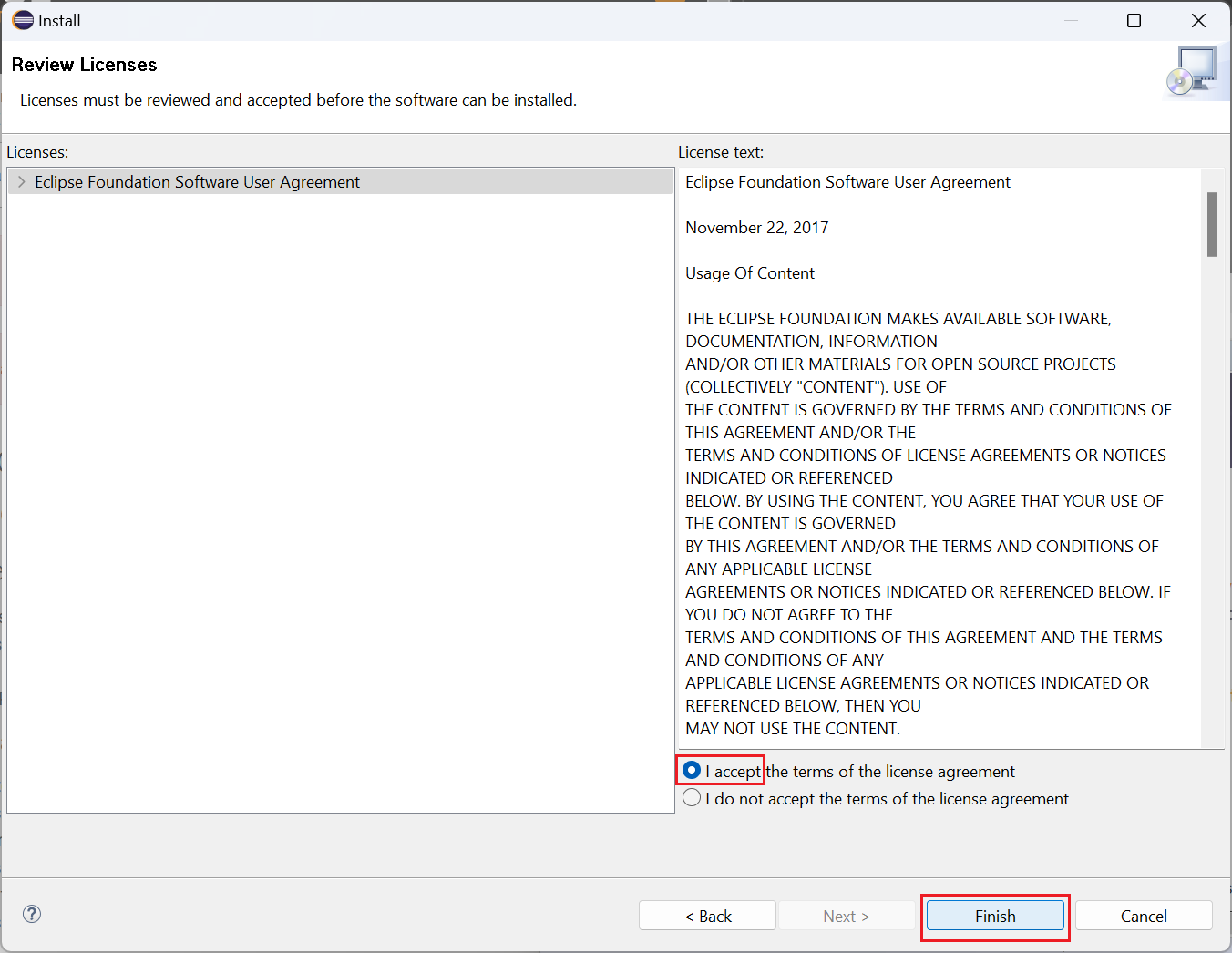
Click Next when the loading is complete.



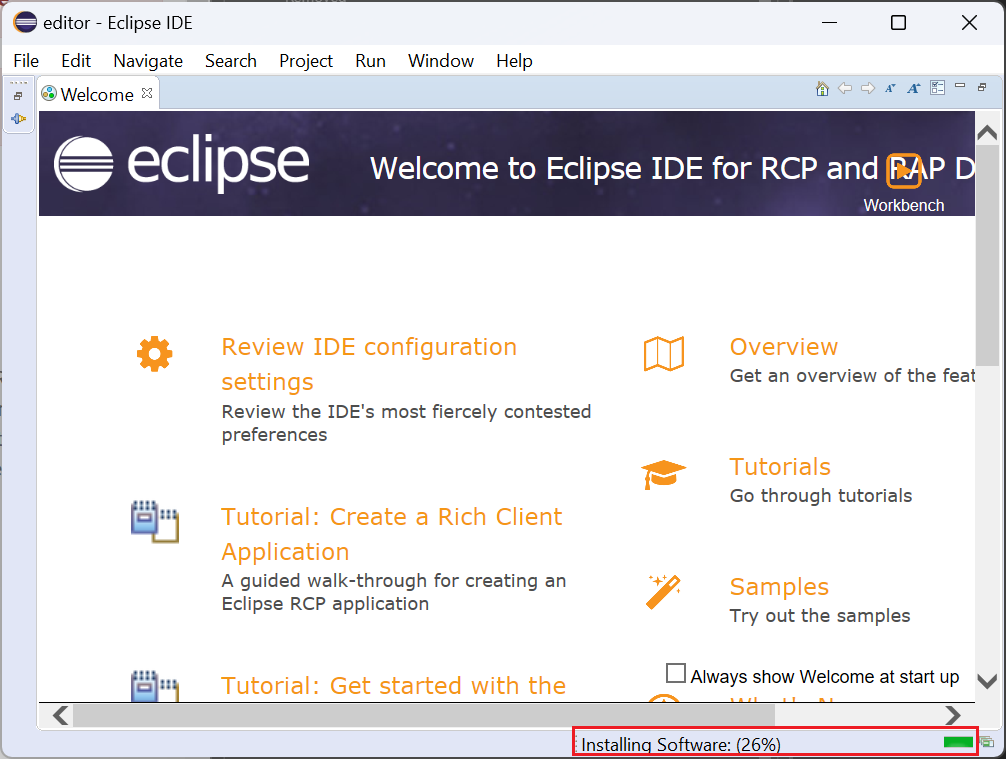
Click Next.

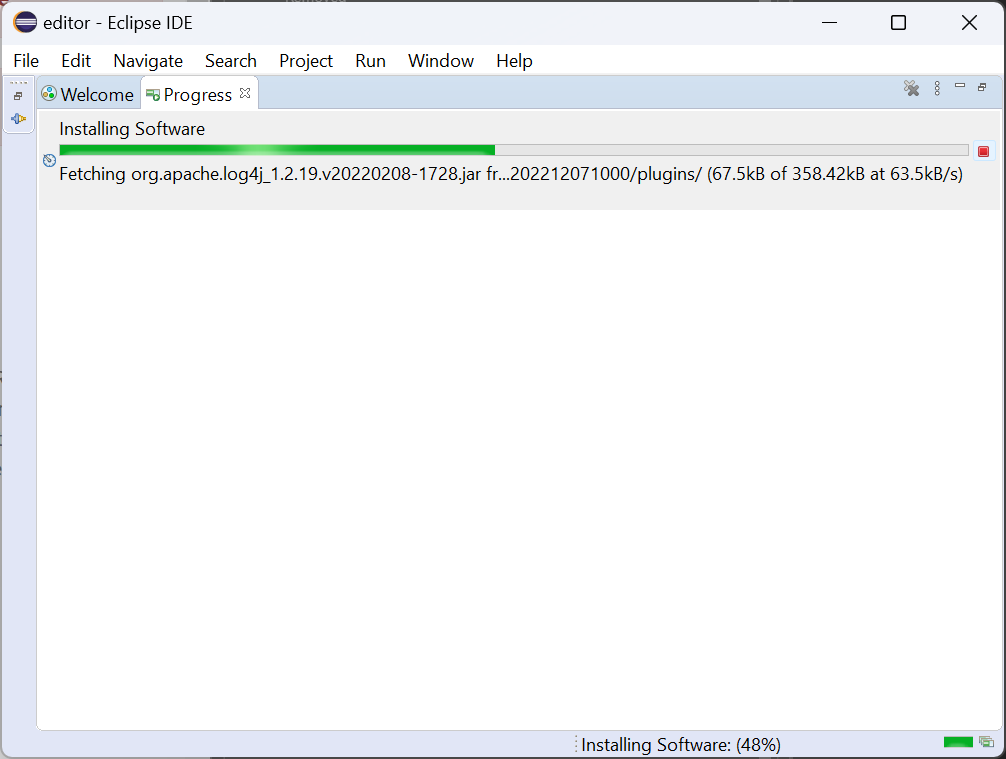


Check the box I accept, and click Finish.

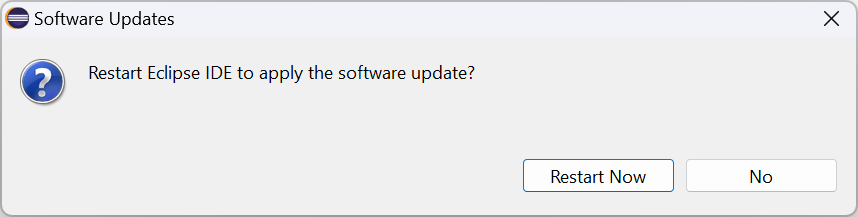


Click in the bottom right corner to see the Progress tab and wait for the download to complete.





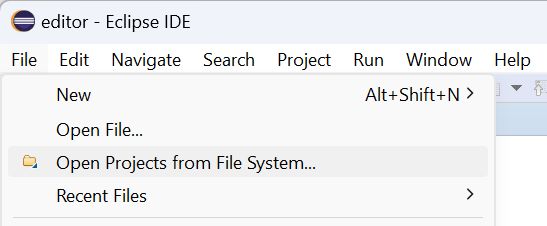
After downloading, click Restart Now.



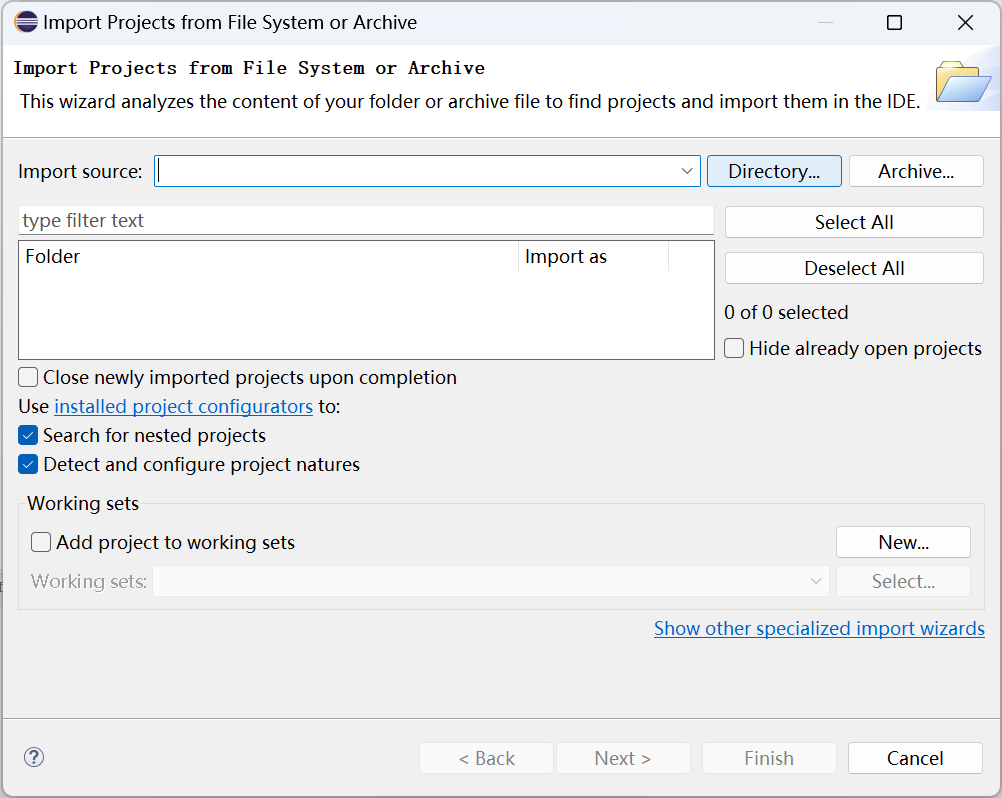
## Run the software project

### Import the software project

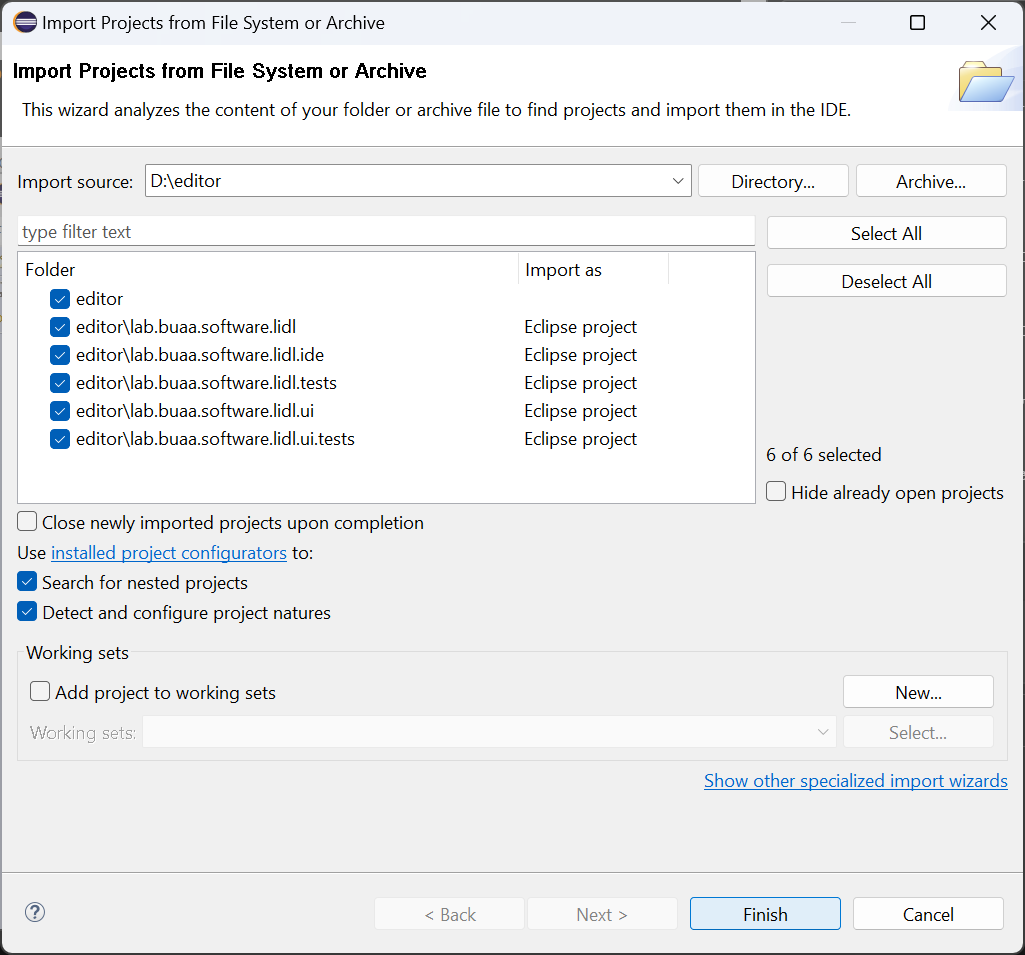
Click File > Open Projects from File System...



Click Directory in Import Projects from File System or Archive.

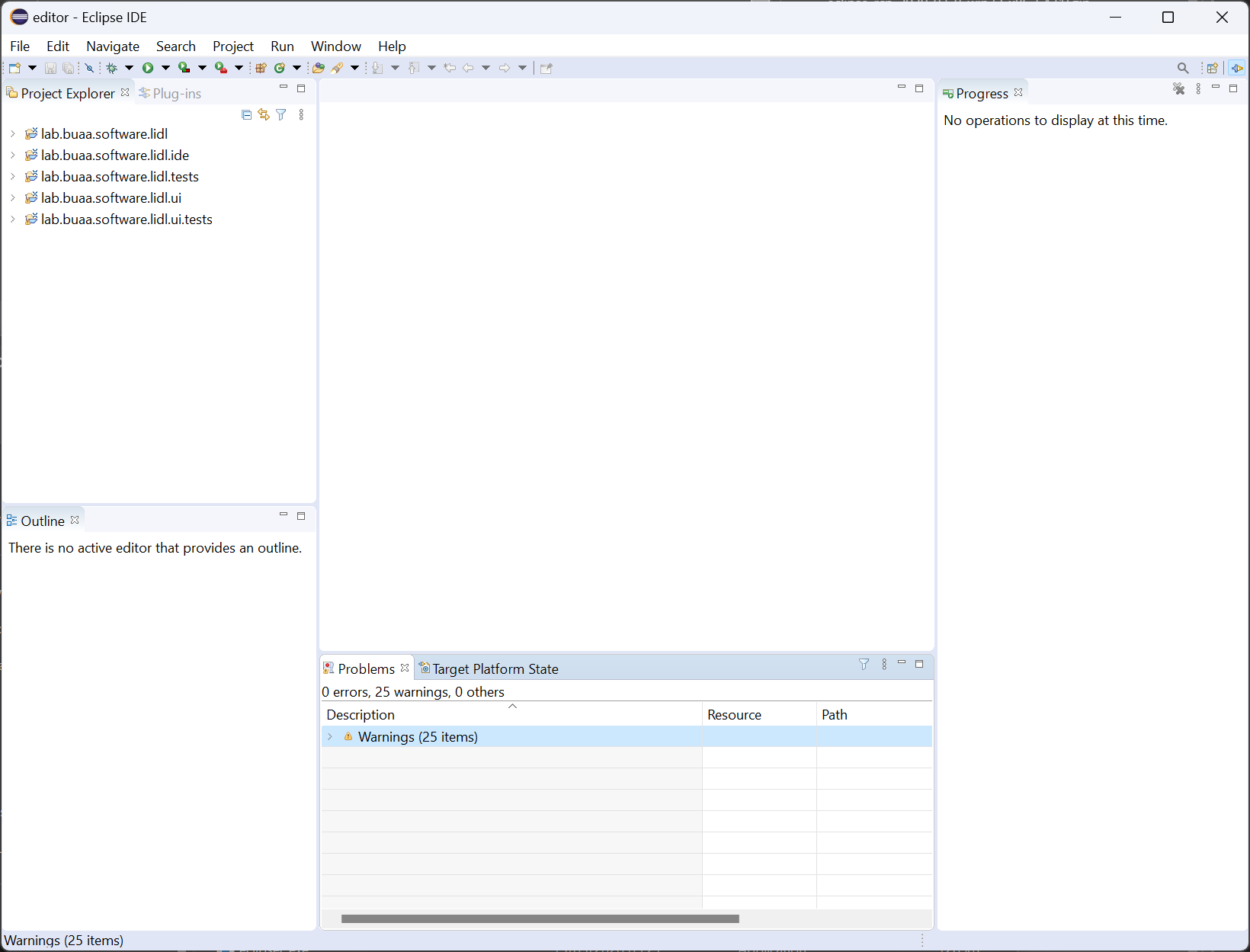


Select the editor folder previously extracted from src.zip, and we select the editor folder in the root directory of the D drive in the manual.



Eclipse will automatically check all Eclipse projects under the Folder, and you can click Finish.

After importing the software project, it shows Warnings (25 items) in the Problems View below Eclipse. According to our experience, you can ignore these Warnings.

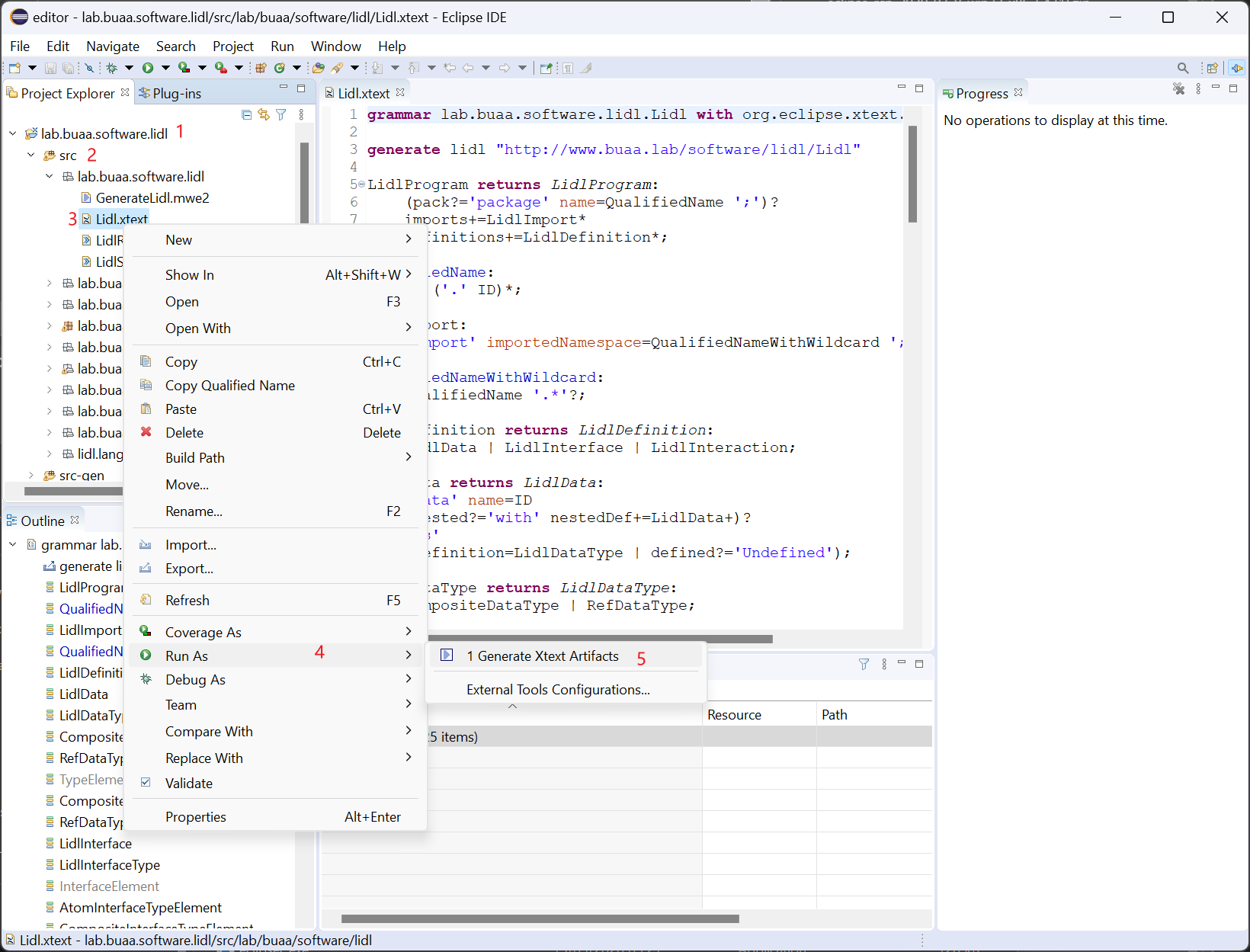


### Generate Xtext Artifacts

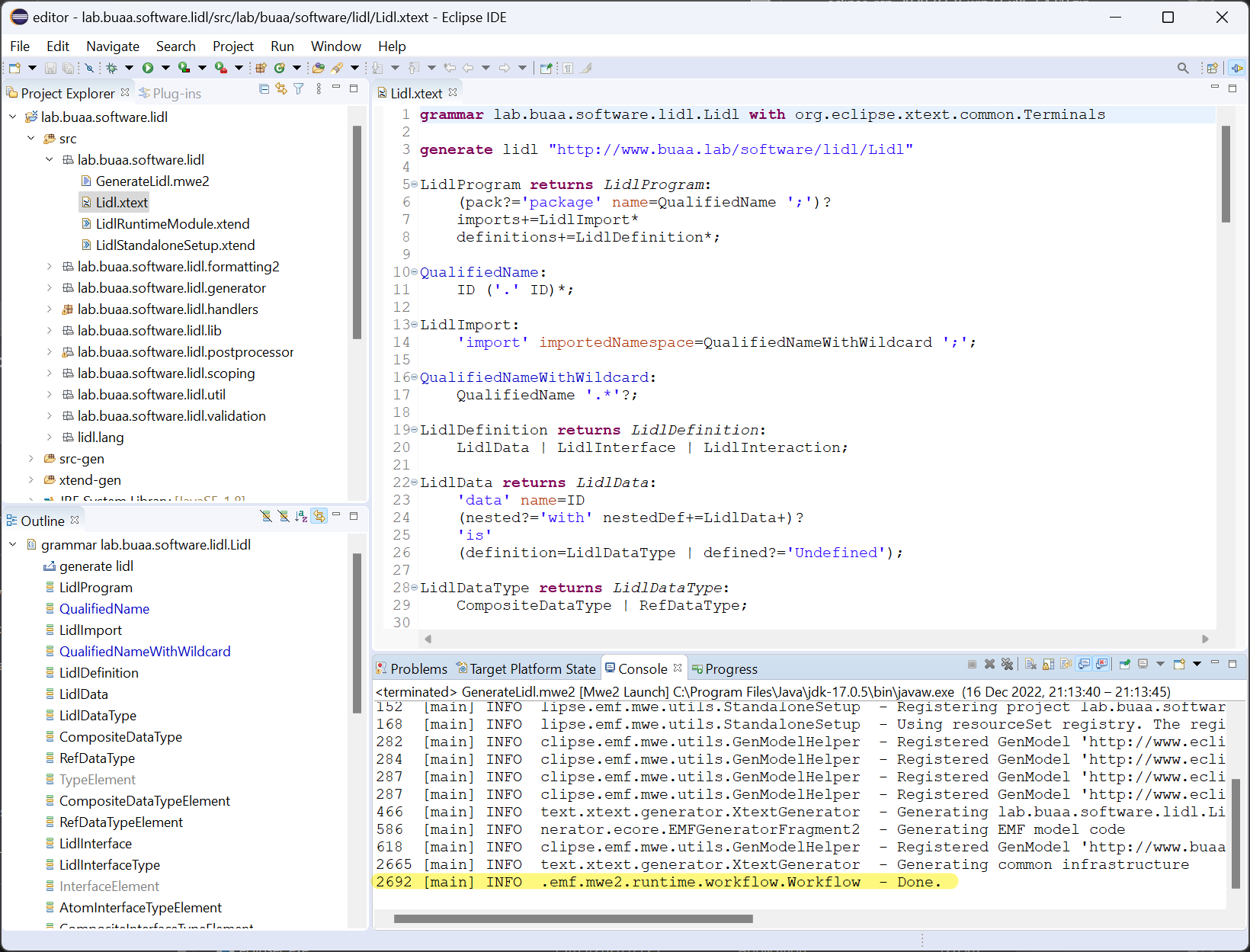
Open /lib.buaa.software.lidl/src/lab.buaa.software.lidl/Lidl.xtext

Right-click on the Lidl.xtext file in Project Explorer

Click Run As, and click Generate Xtext Artifacts



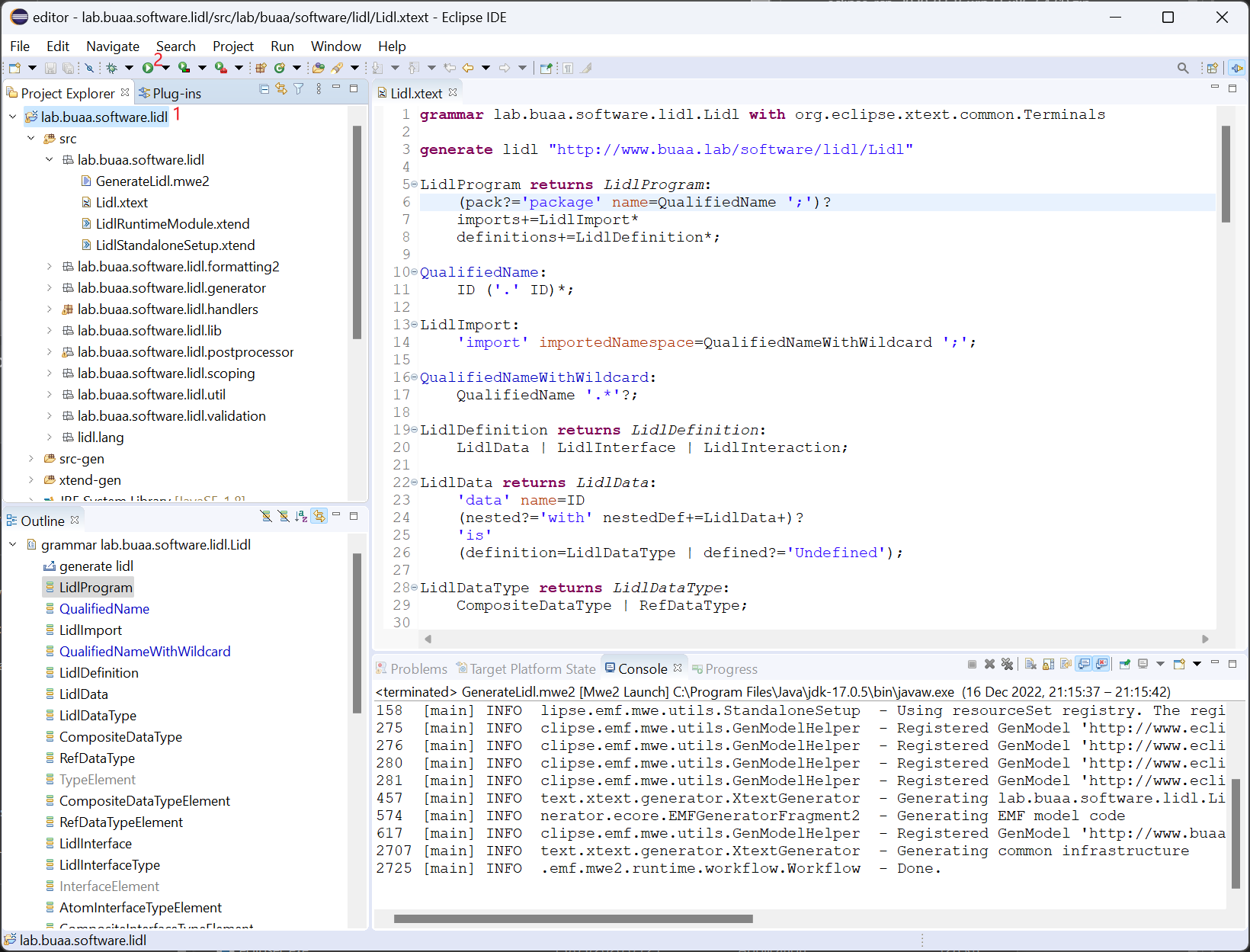
When Console shows Done, generation completes.



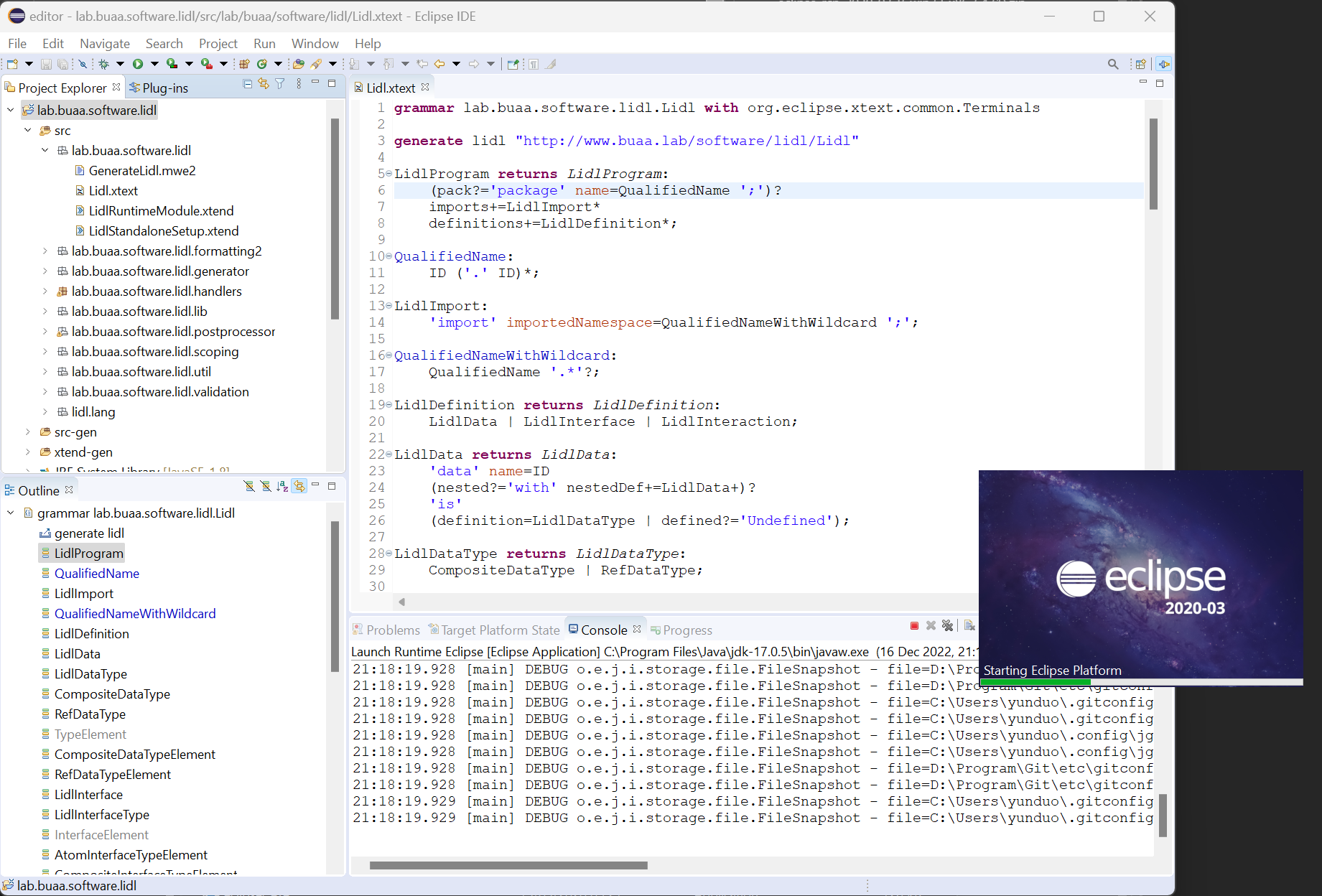
### Run Code Editor

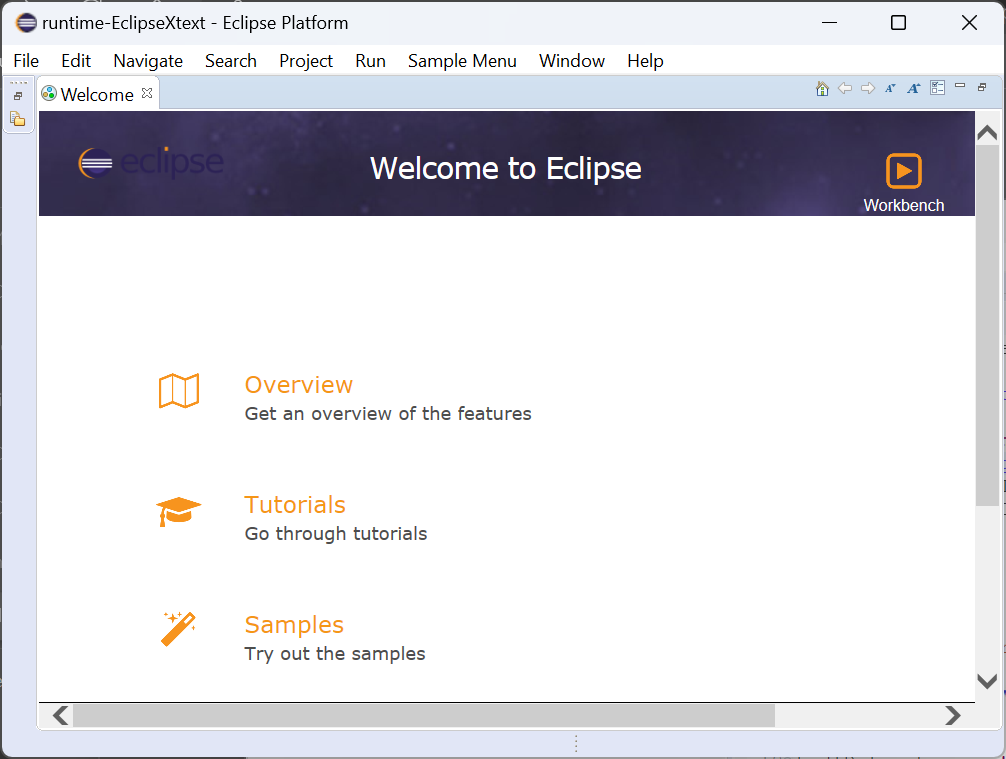
Click on lab.buaa.software.lidl in Project Explorer.

Then click the run button in the toolbar.



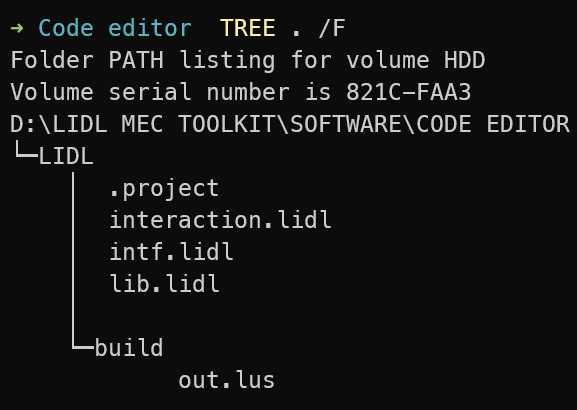
It will start a new Eclipse runtime.





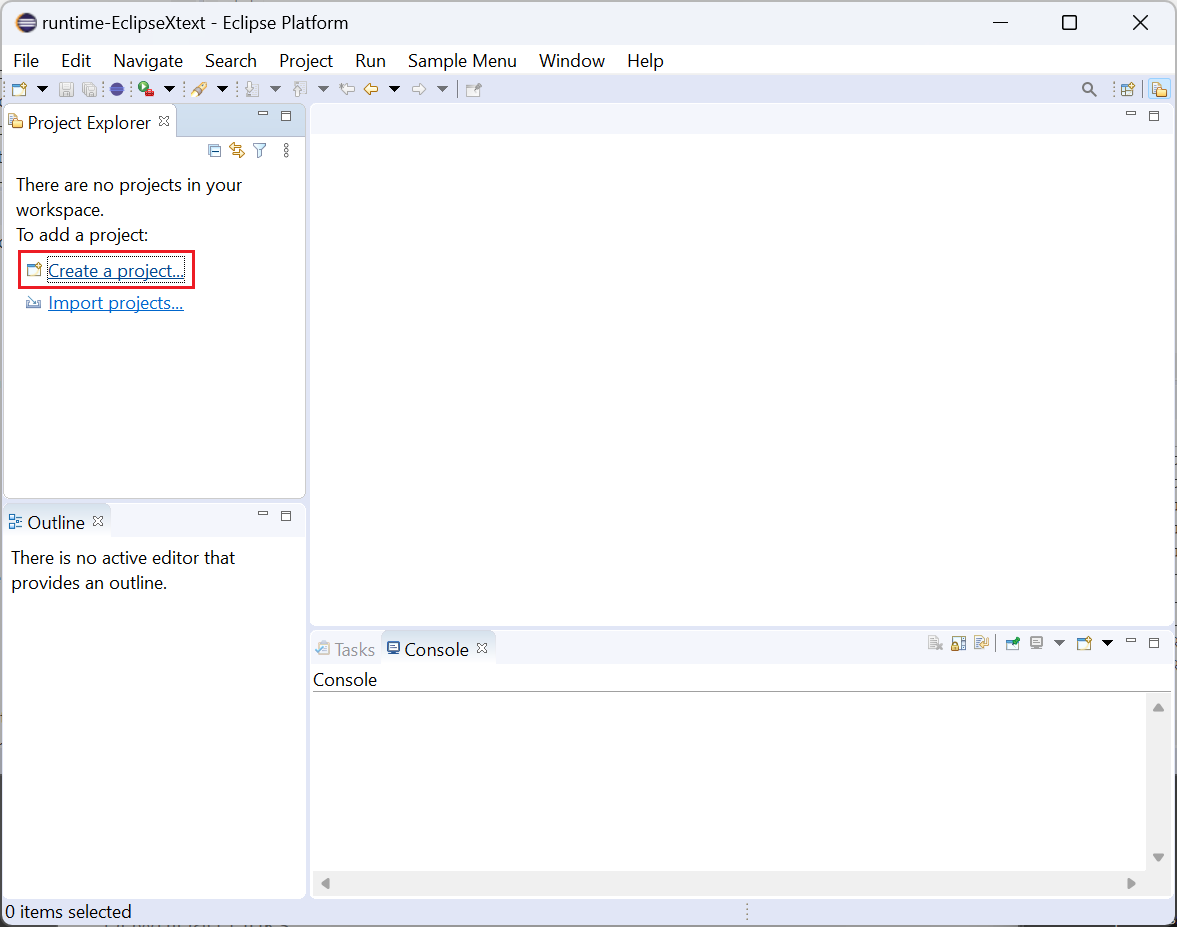
## Edit LIDL code in Code Editor

We have provided the running examples created in this section at  
Zenodo: /LIDL MEC Toolkit/software/Code editor/LIDL/  
GitHub: /LIDL-MEC-Toolkit/software/Code editor/LIDL/

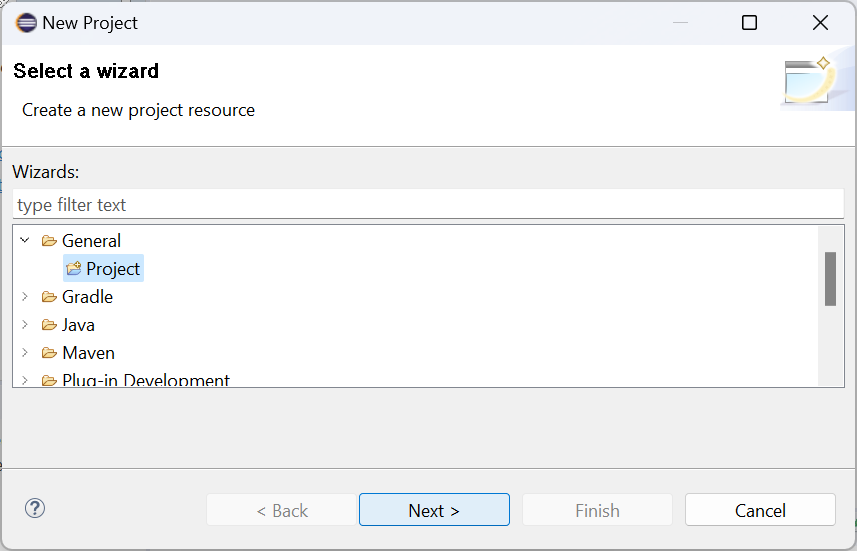


### Create a project

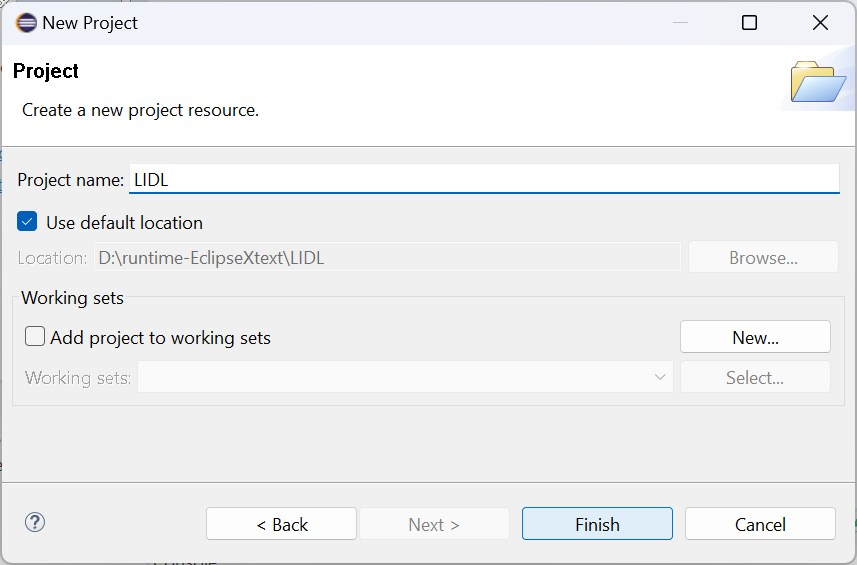
Click Create a project...



Select General > Project, and click Next.

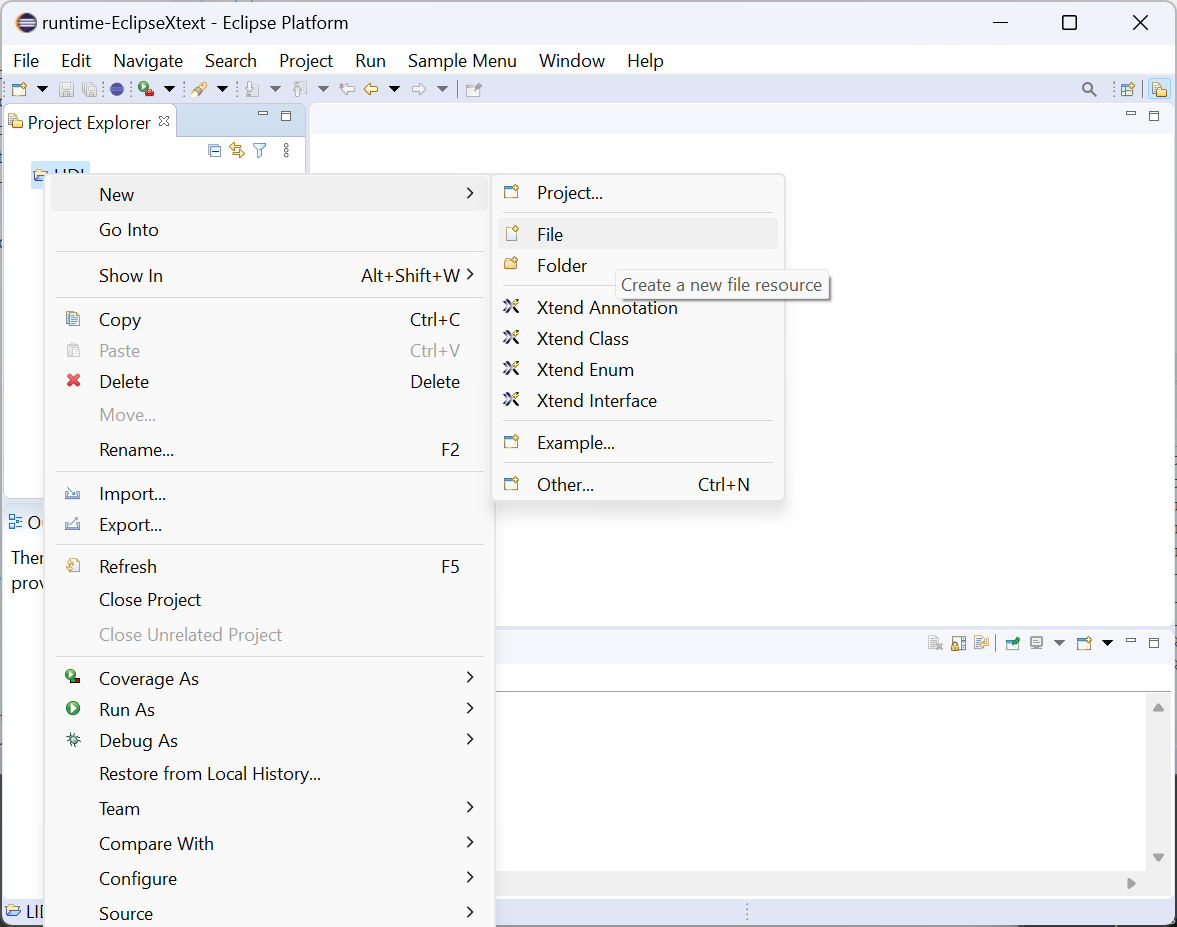


Name the project without spaces. Here it is named LIDL, and click Finish.

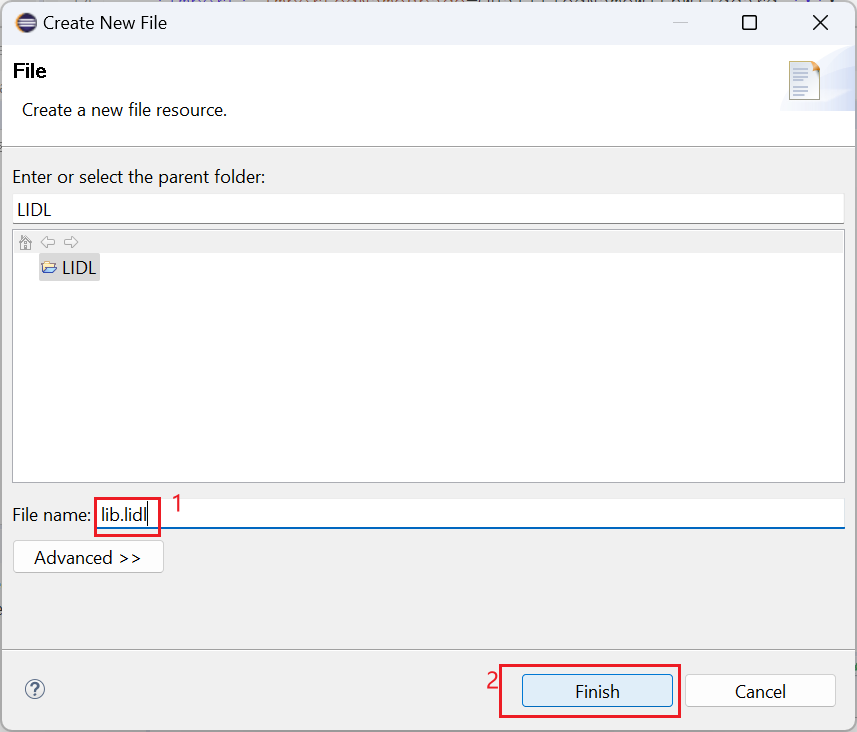


### Edit LIDL code

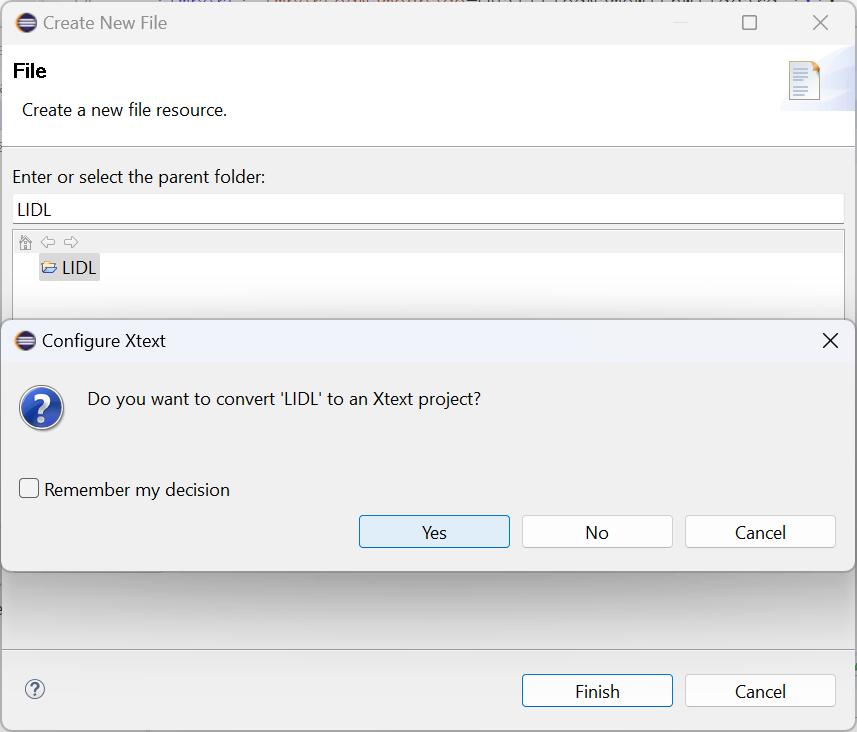
Right-click on the created LIDL project and create a new file.



Name the file in the pop-up window. Name it lib.lidl and click Finish.



Click Yes in the pop-up window.



Input the following in the newly created lib.lidl and save it.

**package** lidl.lang;

**data** Text

**is** **Undefined**

**data** Activation

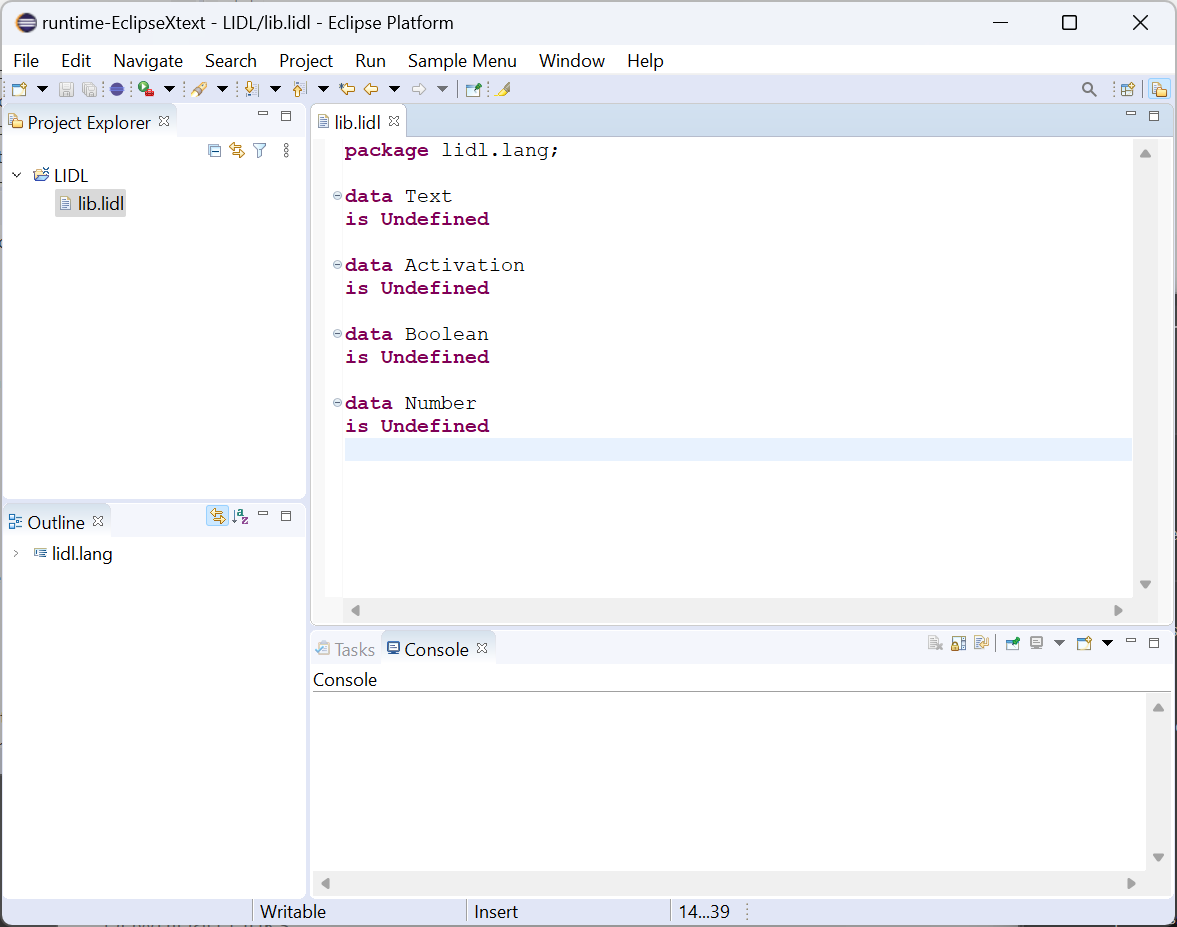
**is** **Undefined**

**data** Boolean

**is** **Undefined**

**data** Number

**is** **Undefined**



Then create a new intf.lidl file with the following content and save it.

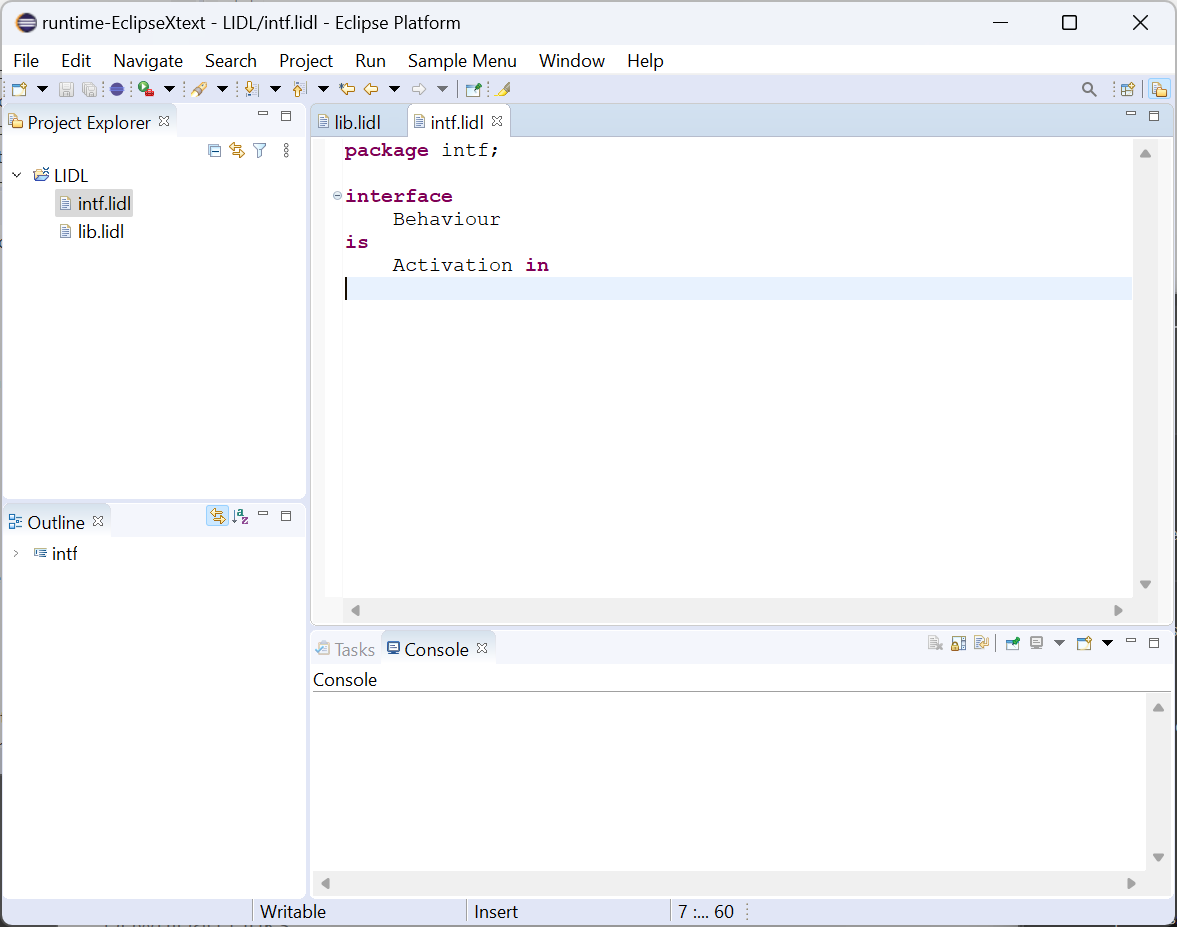
**package** intf;

**interface**

Behaviour

**is**

Activation **in**



Finally, create a new interaction.lidl file with the following content and save it.

**import** lidl.lang.\*;

**import** intf.\*;

**interaction**

(Counter Start From (theConst: Number **in**) Output (theResult: Number **out**)): Behaviour

**is**

(

(All

((theResult) = (y))

((y) = ((Previous(y)) + (1)))

)

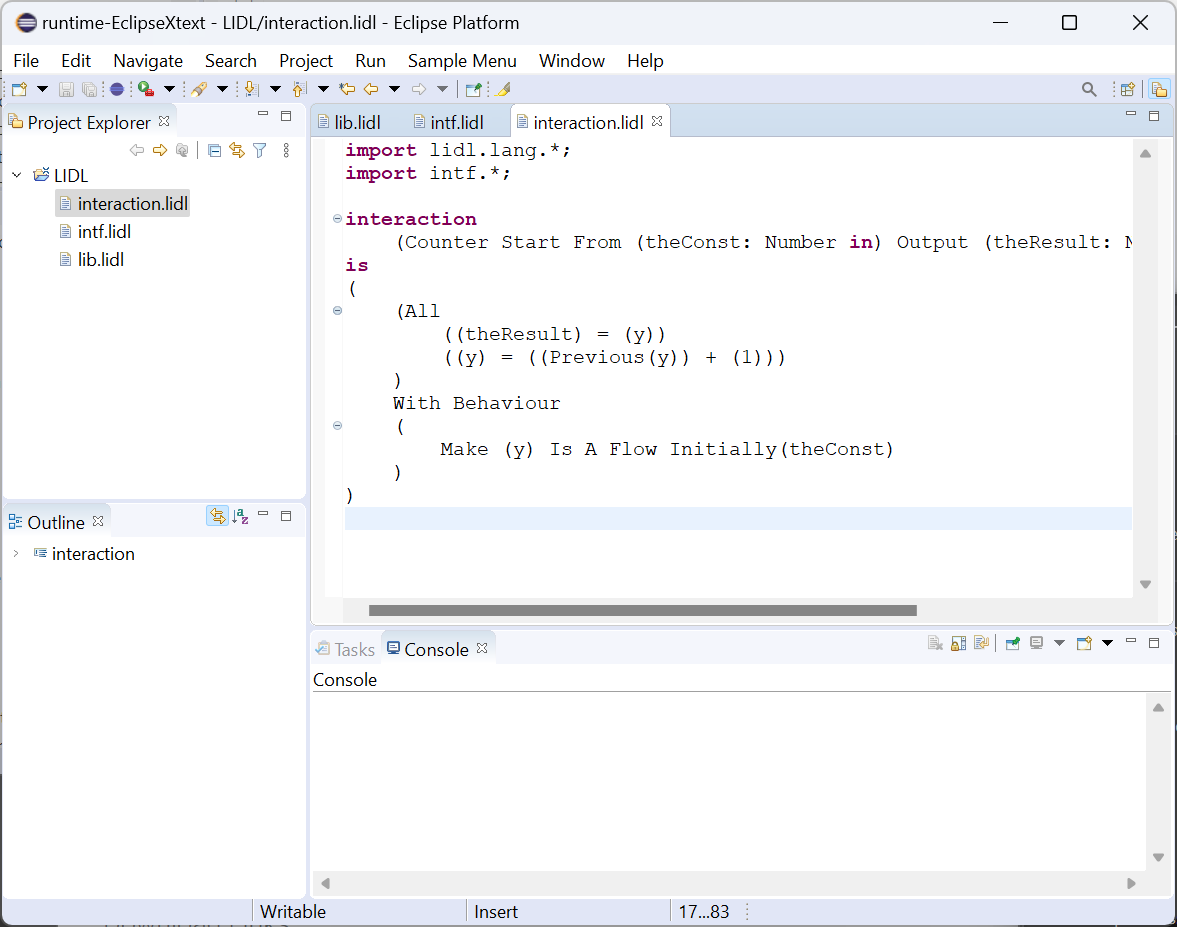
With Behaviour

(

Make (y) Is A Flow Initially(theConst)

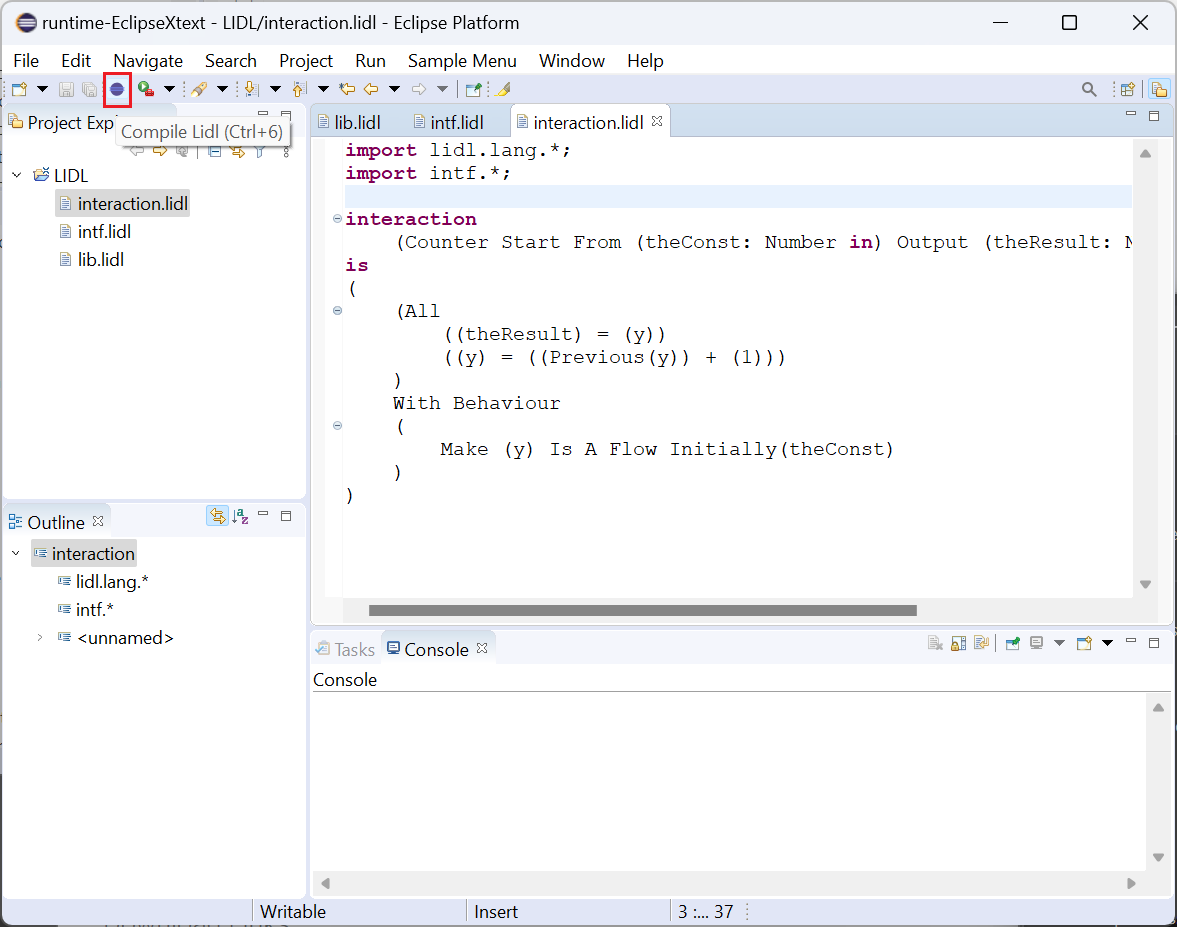
)

)



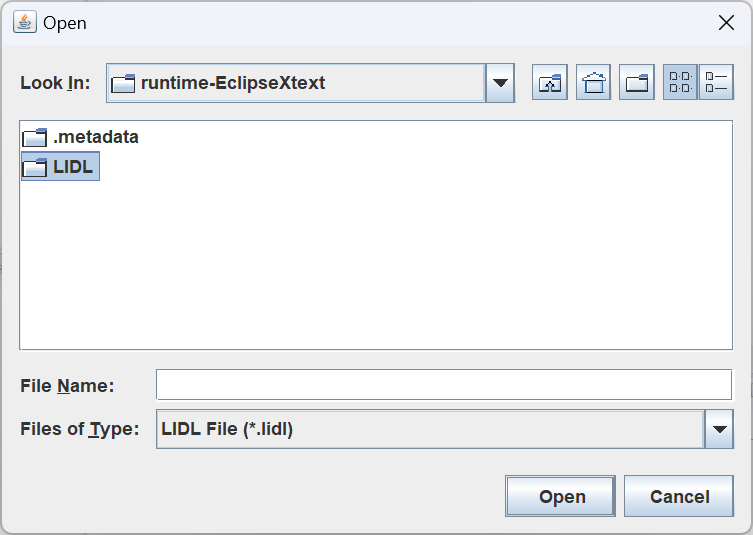
### Compile LIDL code

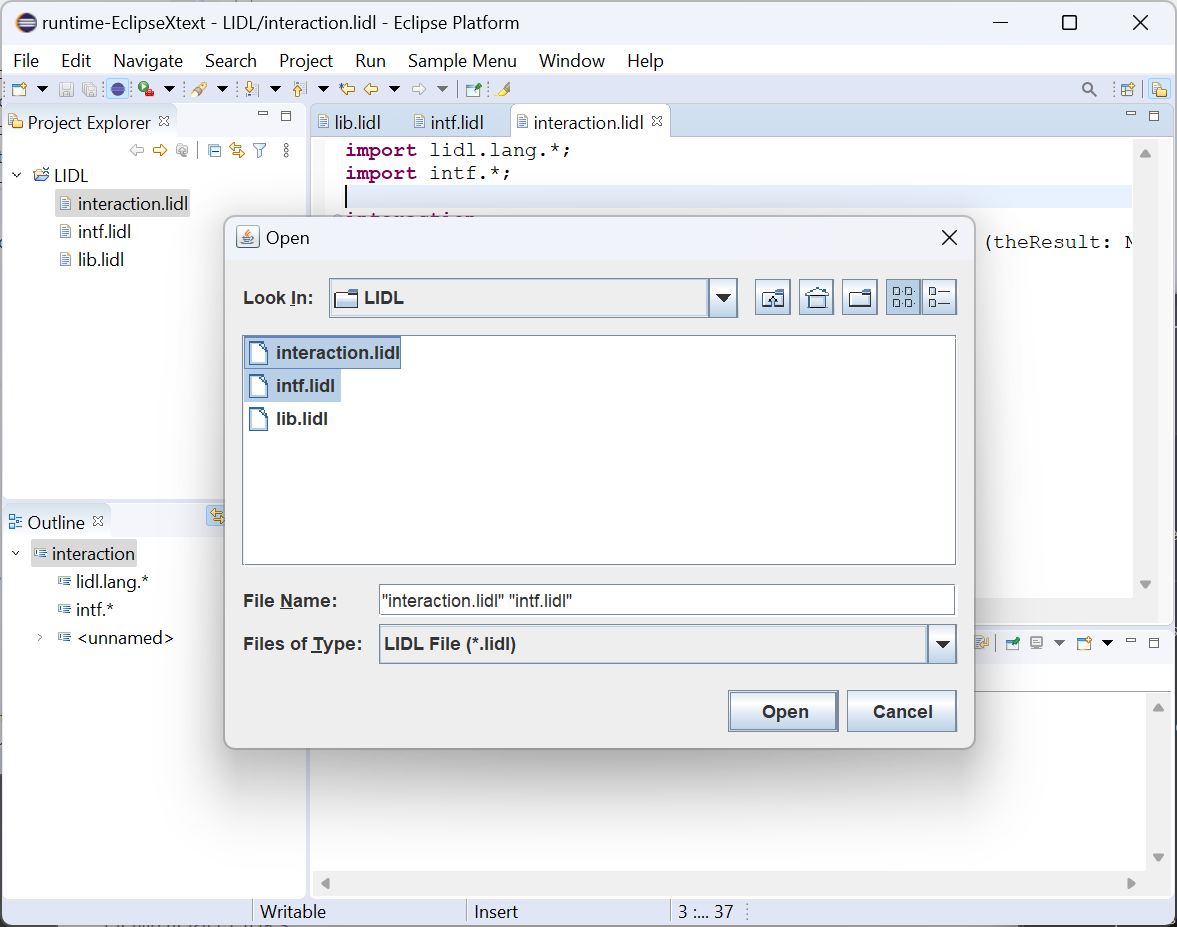
Click Compile Lidl button on the toolbar.



In the pop-up window, select the intf.lidl and interaction.lidl files (hold down the shift key and click the mouse).

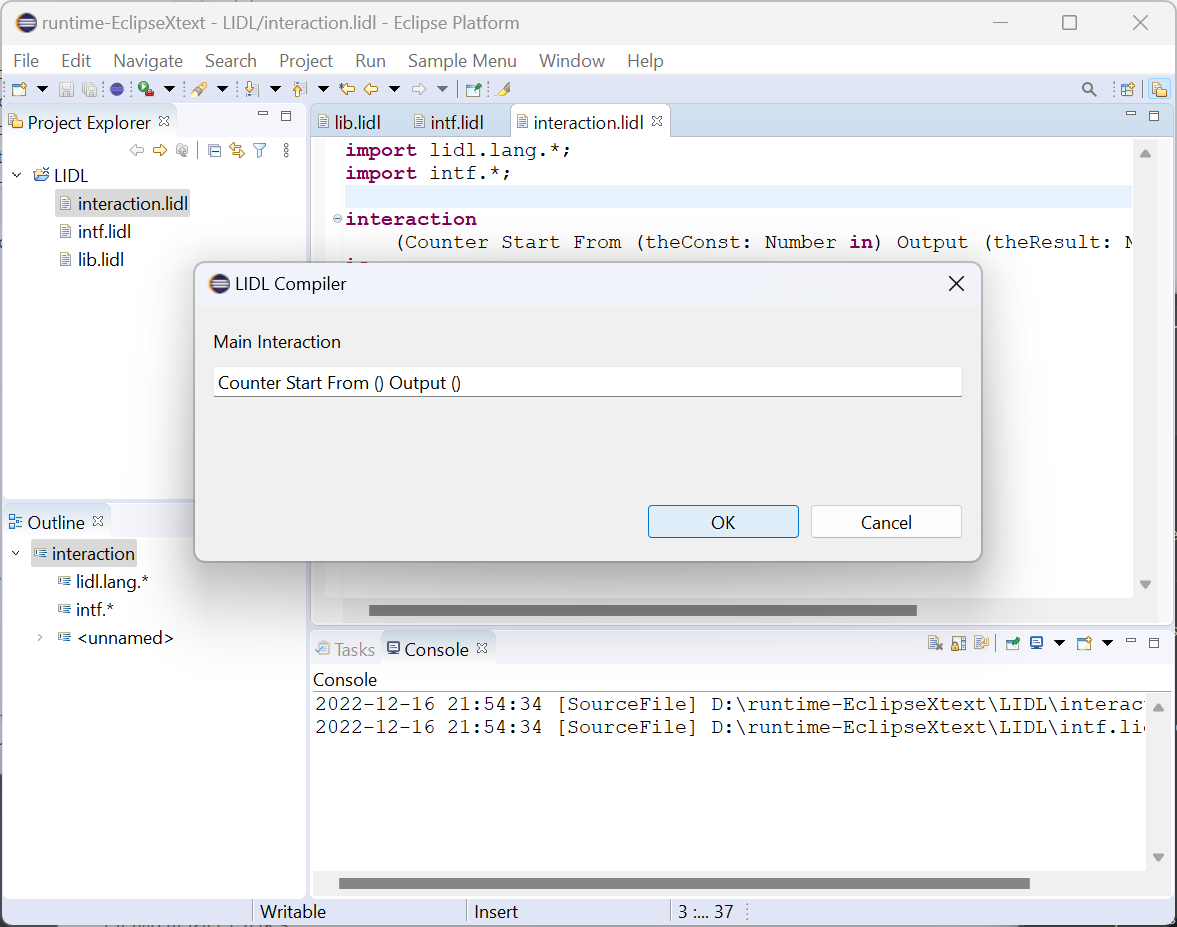
Then click Open.



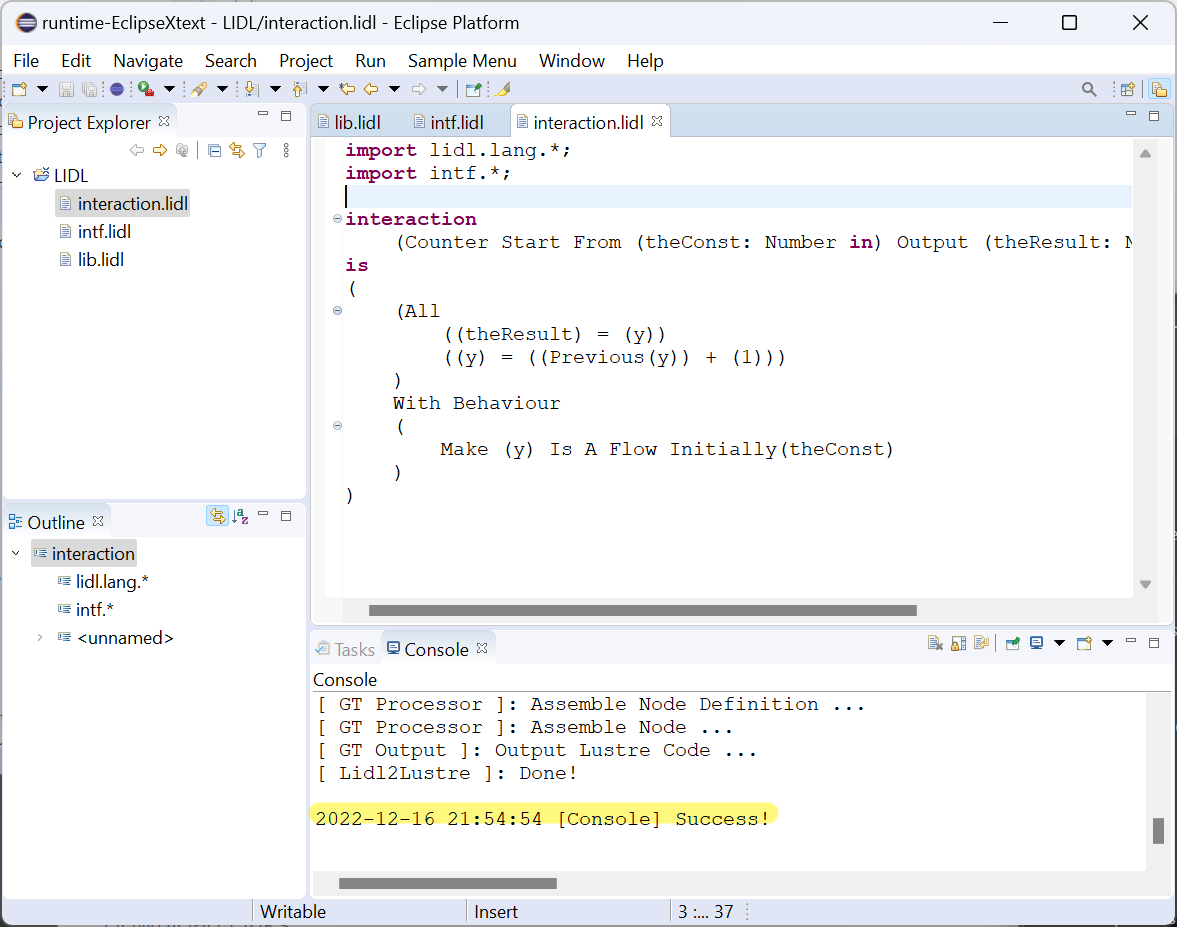


In the LIDL Compiler pop-up window, input Counter Start From () Output ().

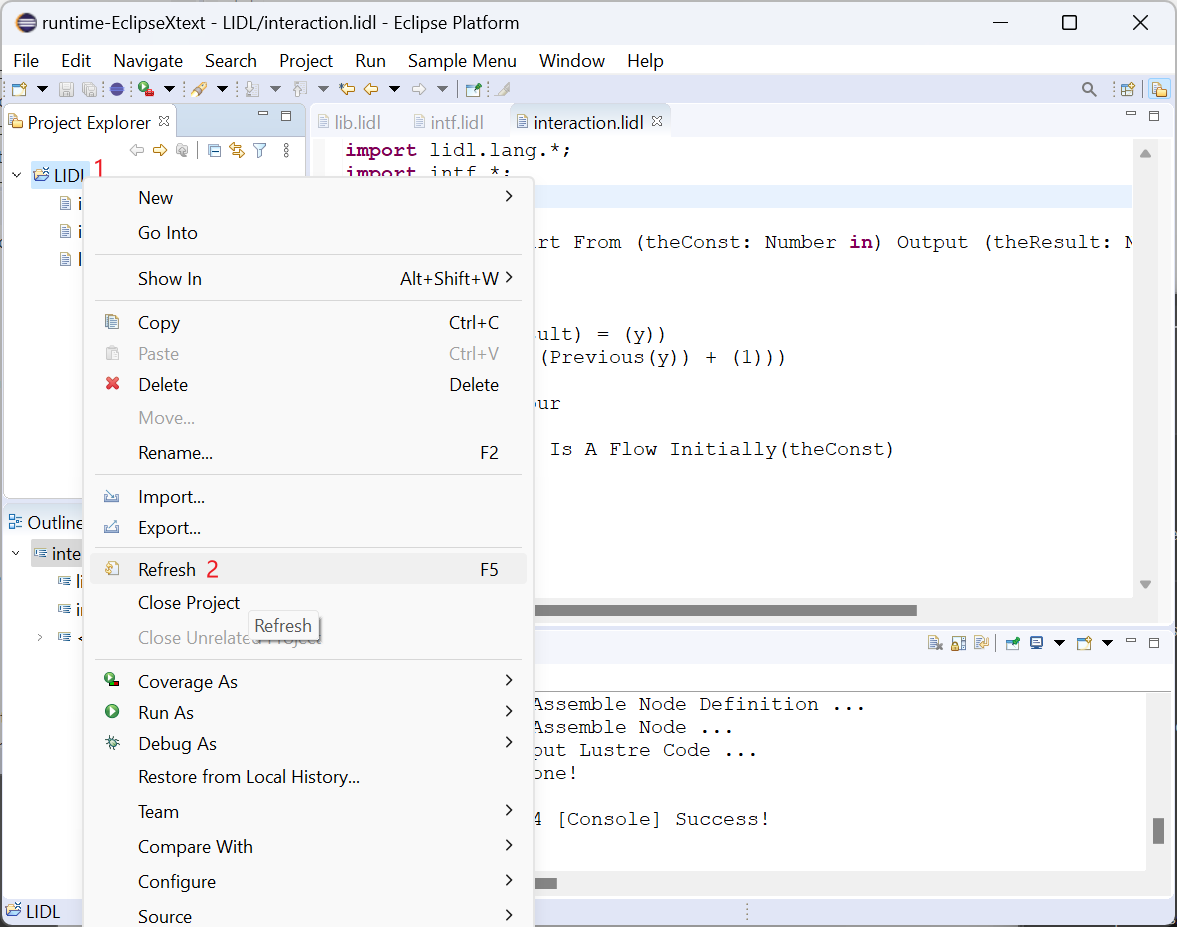
Click OK.



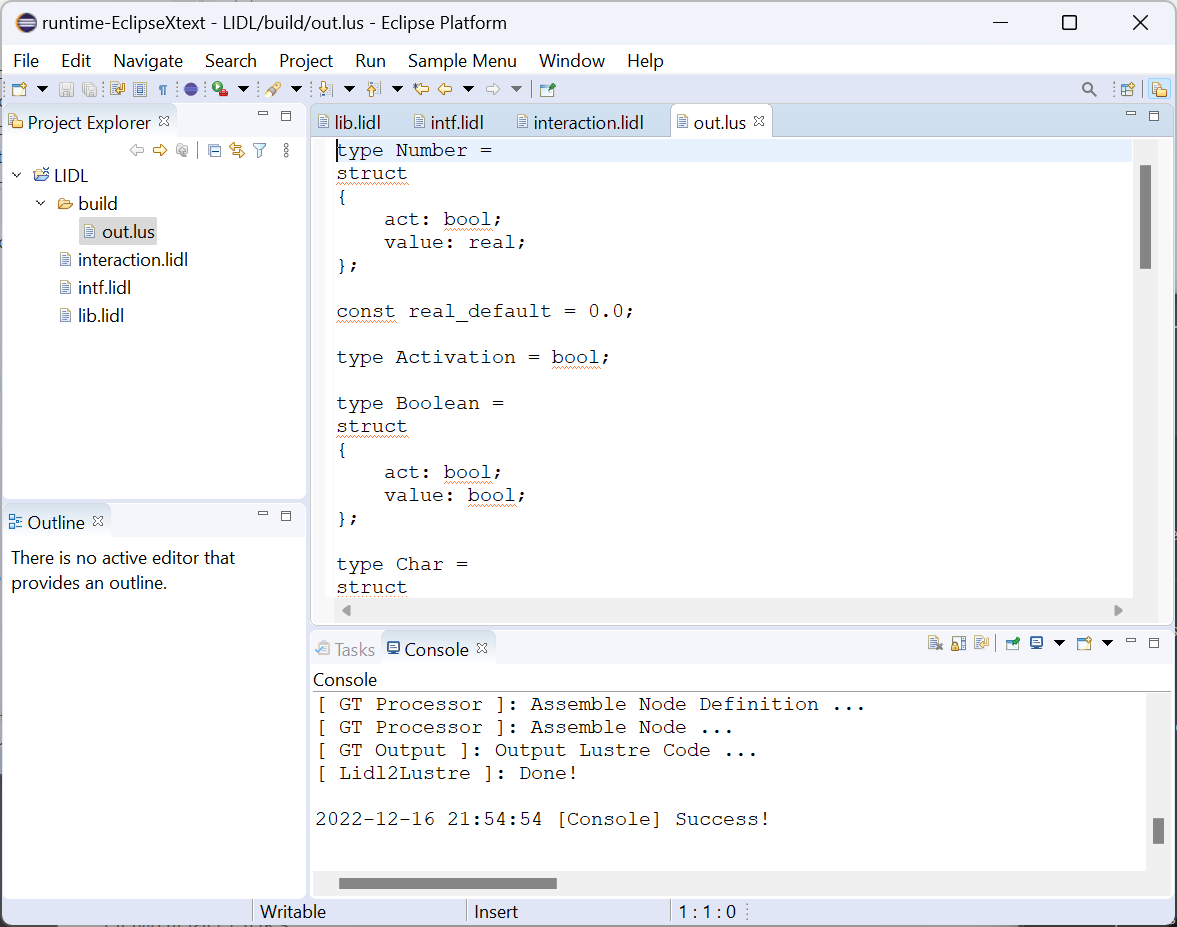
As you can see, the end of the Console shows Success.



Right-click on the LIDL folder and click Refresh.



You can see the compiled Lustre code out.lus file in the build directory generated by the compiler.



## Code Editor source code snippets analysis

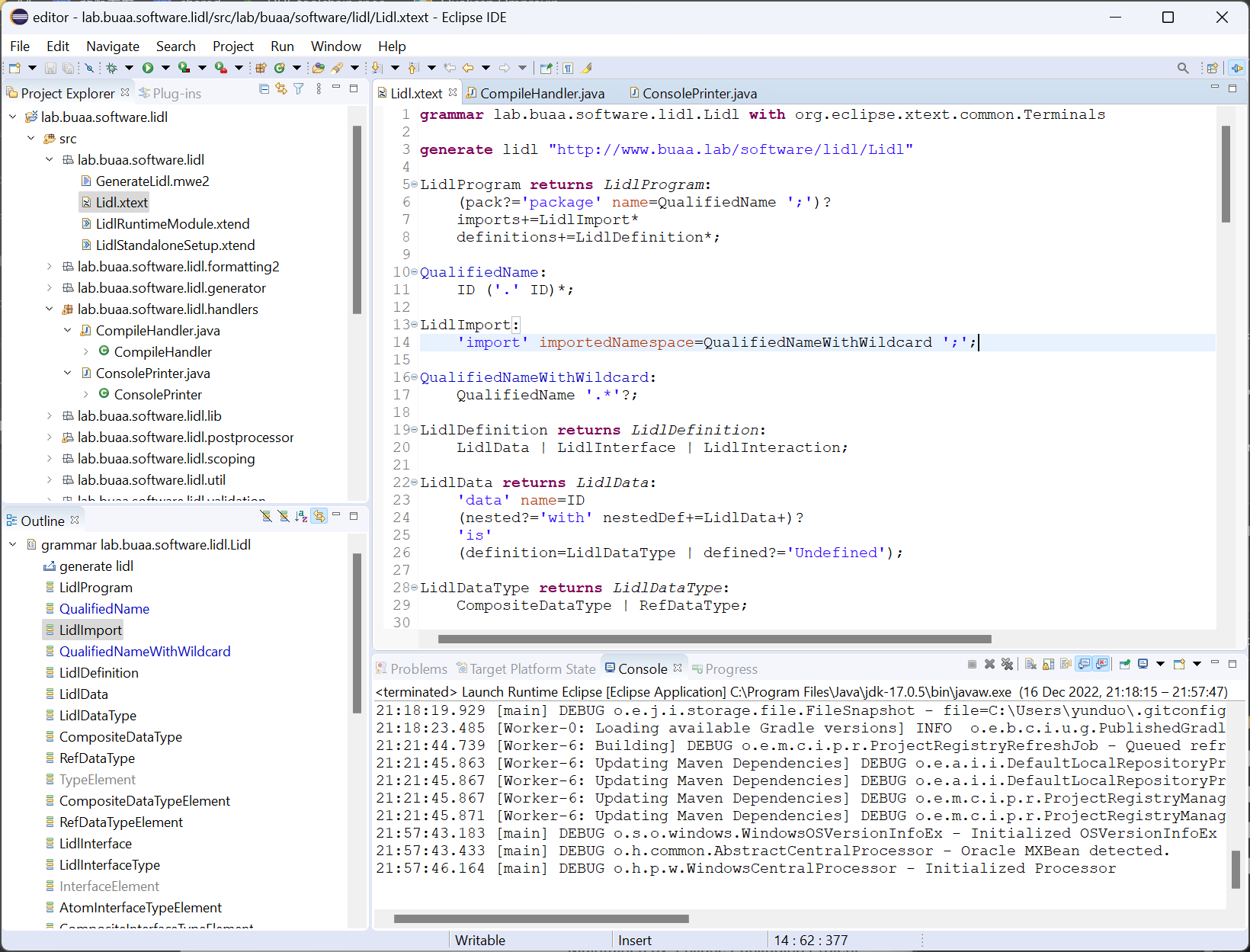
The code highlighting and definition jumping functions in Code Editor are implemented in the Lidl.xtext file.

The path to the Lidl.xtext file is.

/lab.buaa.software.lidl/src/lab/buaa/software/lidl/Lidl.xtext

The specific implementation techniques can be found in:

1. Lorenzo Bettini. 2016. Implementing domain-specific languages with Xtext and Xtend. Packt Publishing Ltd.
2. [Xtext - Documentation (eclipse.org)](https://www.eclipse.org/Xtext/documentation/index.html)   
   https://www.eclipse.org/Xtext/documentation/index.html

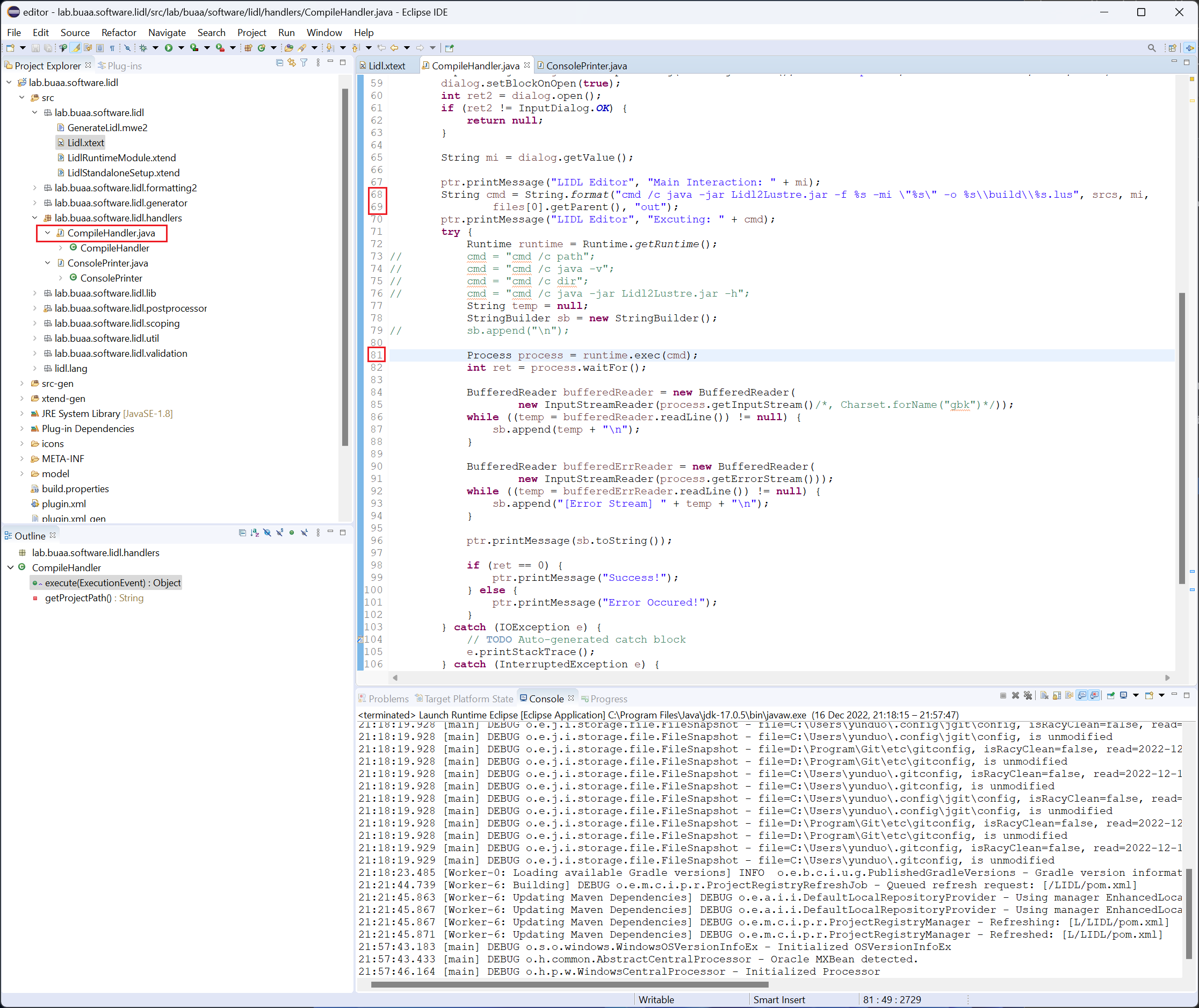


The processing function for the Compiler embedded in the Code Editor is in the CompileHandler.java file

The file path is.

/lab.buaa.software.lidl/src/lab/buaa/software/lidl/handlers/CompileHandler.java

Lines 68-69 of the code set up the console command that calls the compiler jar and it executes the command on line 81 of the code.



The ConsolePrinter.java file configures the format in which the console displays log messages in Code Editor.

The file path is:

/lab.buaa.software.lidl/src/lab/buaa/software/lidl/handlers/ConsolePrinter.java

