SDCLib & CMake

Eclipse Setup

This document describes how to create an Eclipse CDT Project from the SDCLib CMake Project with cmake-gui.

Requirements

- Cloned git repository of SDCLib
- Installed all required packages to build the SDCLib (as described in the README.md)
- Cmake-gui
- Eclipse

CMake to Eclipse: 2 Steps

- Generate the Makefiles
- Import into Eclipse

The first step covers the generation of the Makefiles with the cmake-gui in an out-of-source approach.
Out-of-source: The binaries are **not** built in the same folder as your source and CMake files. You can easily „cleanup“ by deleting the generated files from the filesystem without touching the source code and manage different (external) binary folders for different kind of build types (DEBUG, RELEASE, etc.).

The second step covers the import of the generated Makefiles into the Eclipse IDE.

Generating the Makefiles

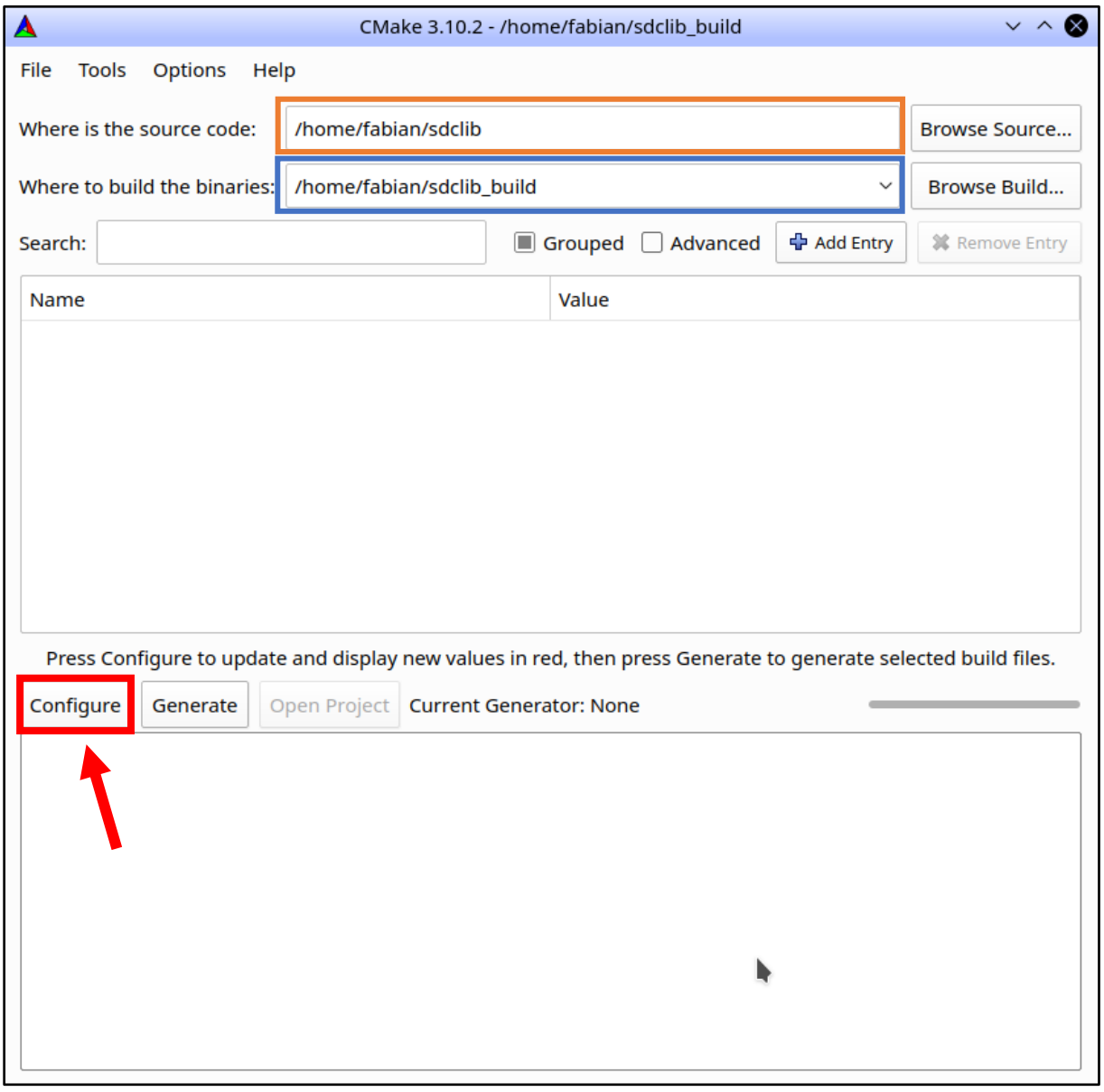
Open cmake-gui.

Where is the source code	[…]	is your cloned folder.
Where to build the binaries	[…]	should be another folder . (out-of-source build recommended!)

In this case the cloned folder is /home/fabian/sdclib and the build folder will be /home/fabian/sdclib_build.

If the build folder does not exist, cmake will create it for you.

Next click **Configure**

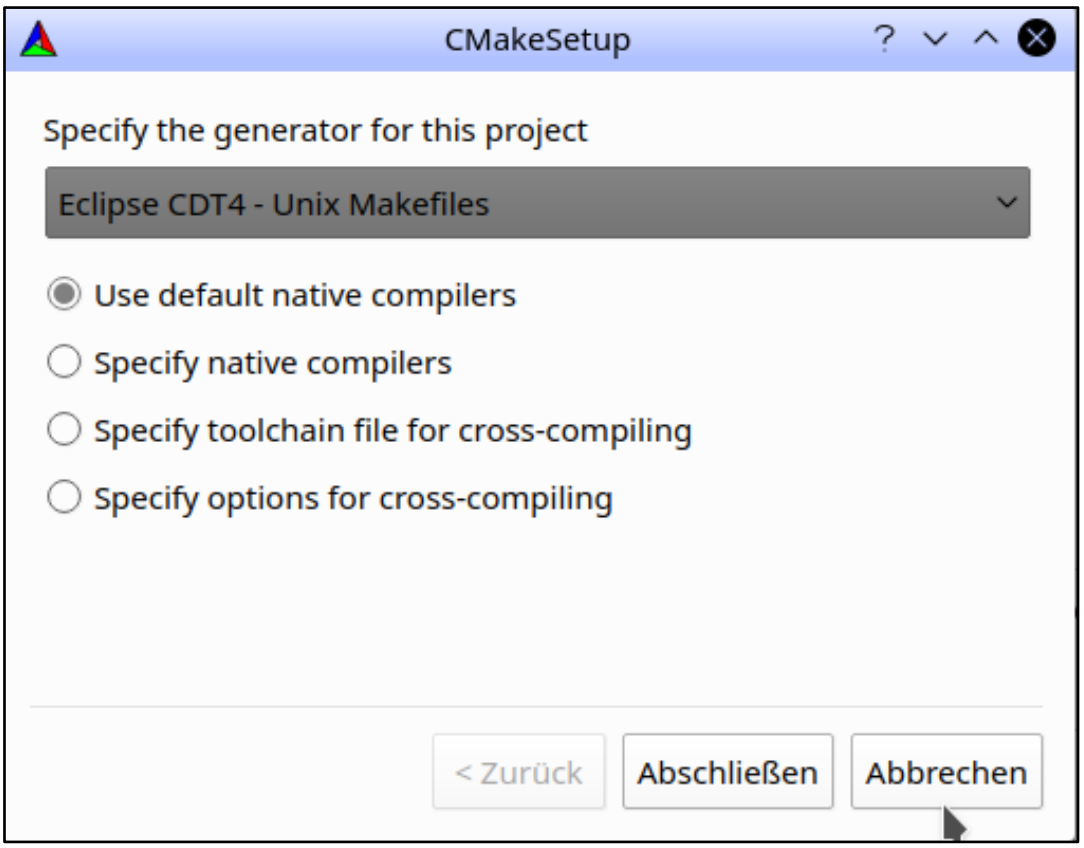


When asked which generator to use, select:

Eclipse CDT4 – Unix Makefiles

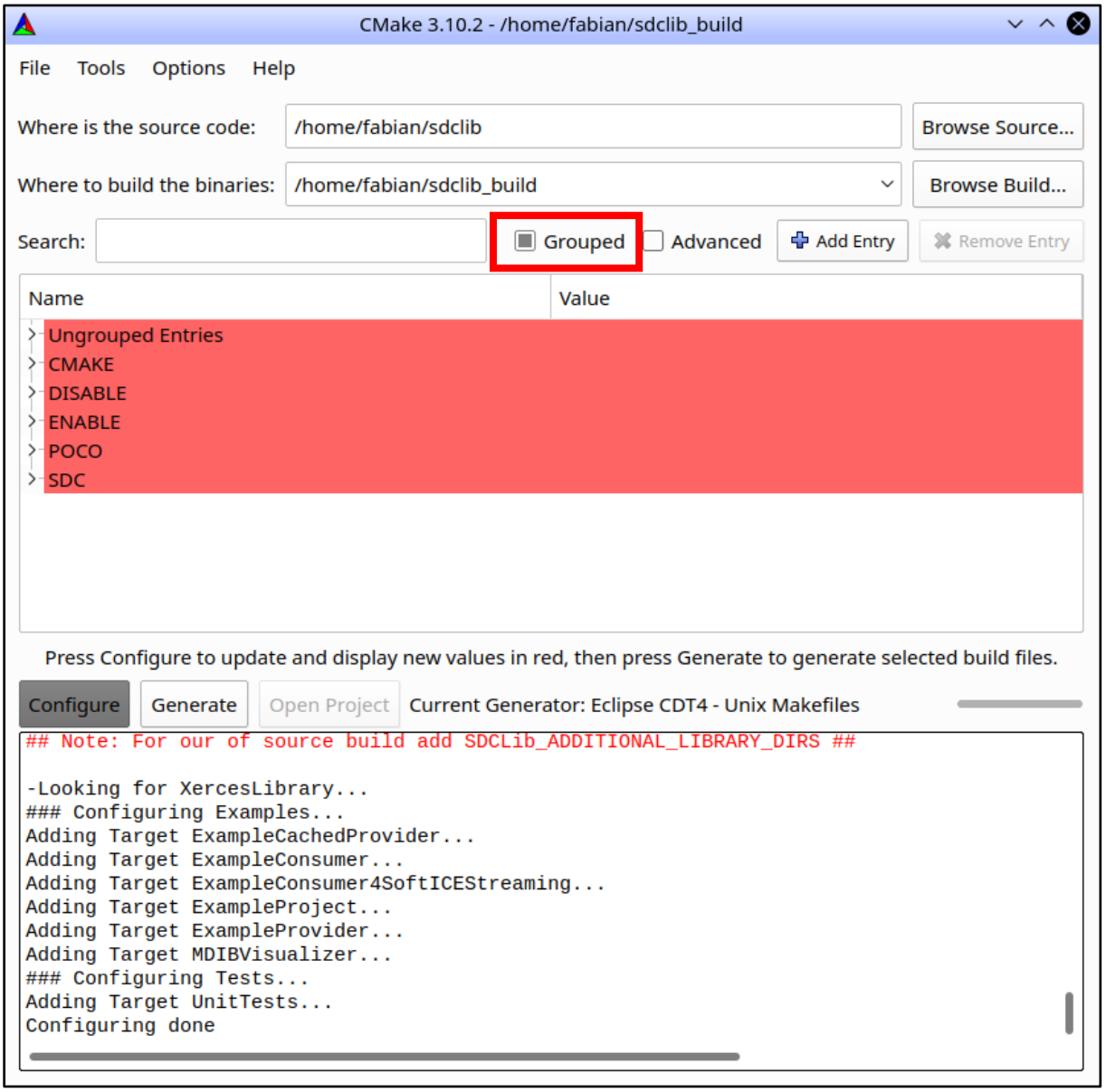
and proceed.

(**Note**: You can „Specify native compilers“, but you should be fine with the „Use default native compilers“ option for most of the time here.)



The generating process should start and you should see something similar to this:

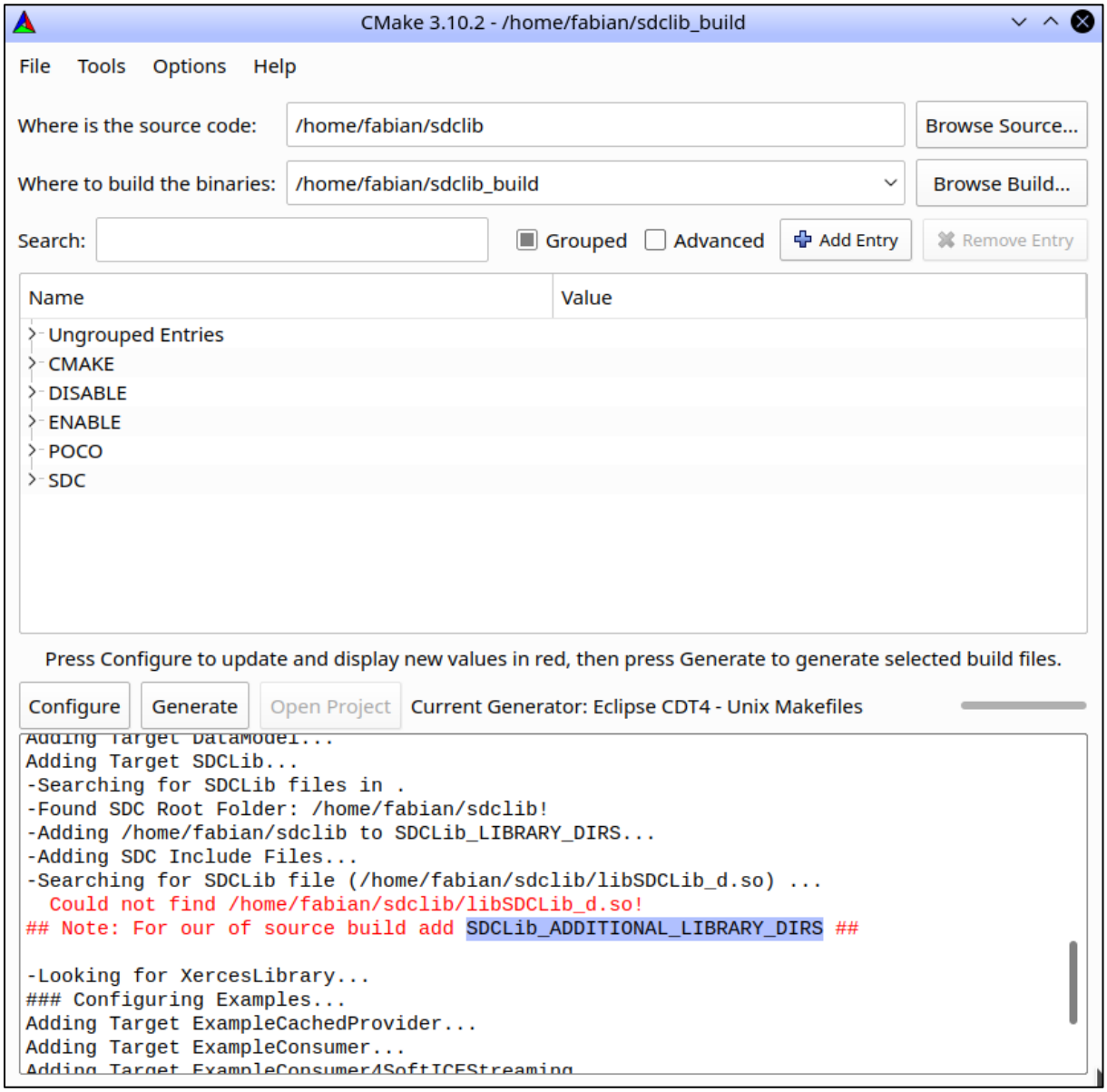
Hint: You should check the „Grouped“ checkbox for a better overview.



Variables highlighted red were „updated“ in the last configure step.

Another click on *Configure* will remove this mark. Keep configuring until there are no more highlighted variables.

You will notice a warning (warnings are red, errors would abort the whole process):

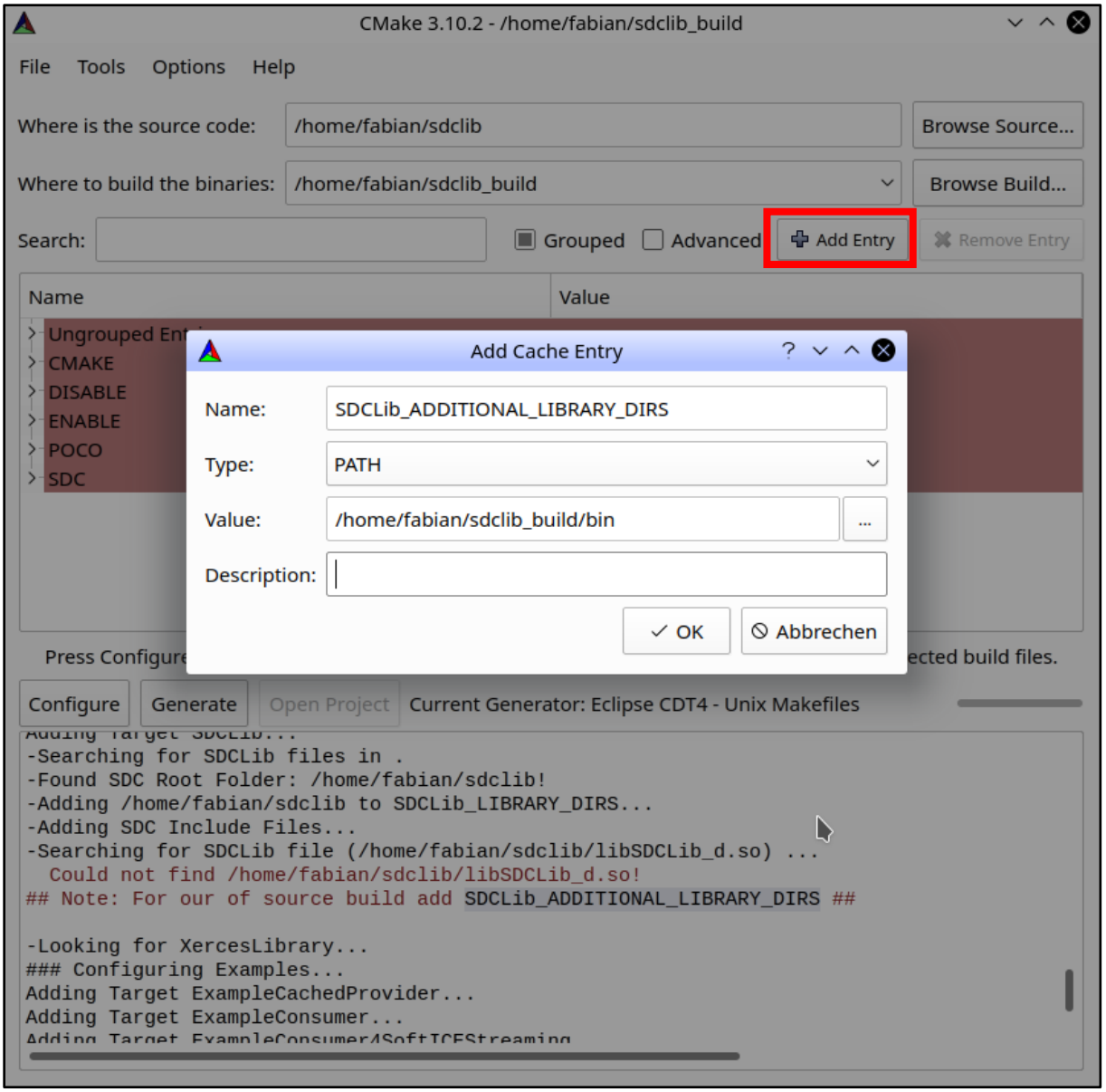


When doing an out-of-source build you can add this variable to tell the FindSDCLib module where to search for the built SDCLib binary.

This warning tells you, that it could not find the binary (yet!), but lets you specify a path where to search for it later.

To add a new variable, click on „Add Entry“, next to the Grouped and Advanced box, a window will open where you can specify name and type and value of the new variable:

(Hint: copy paste the name from the output)



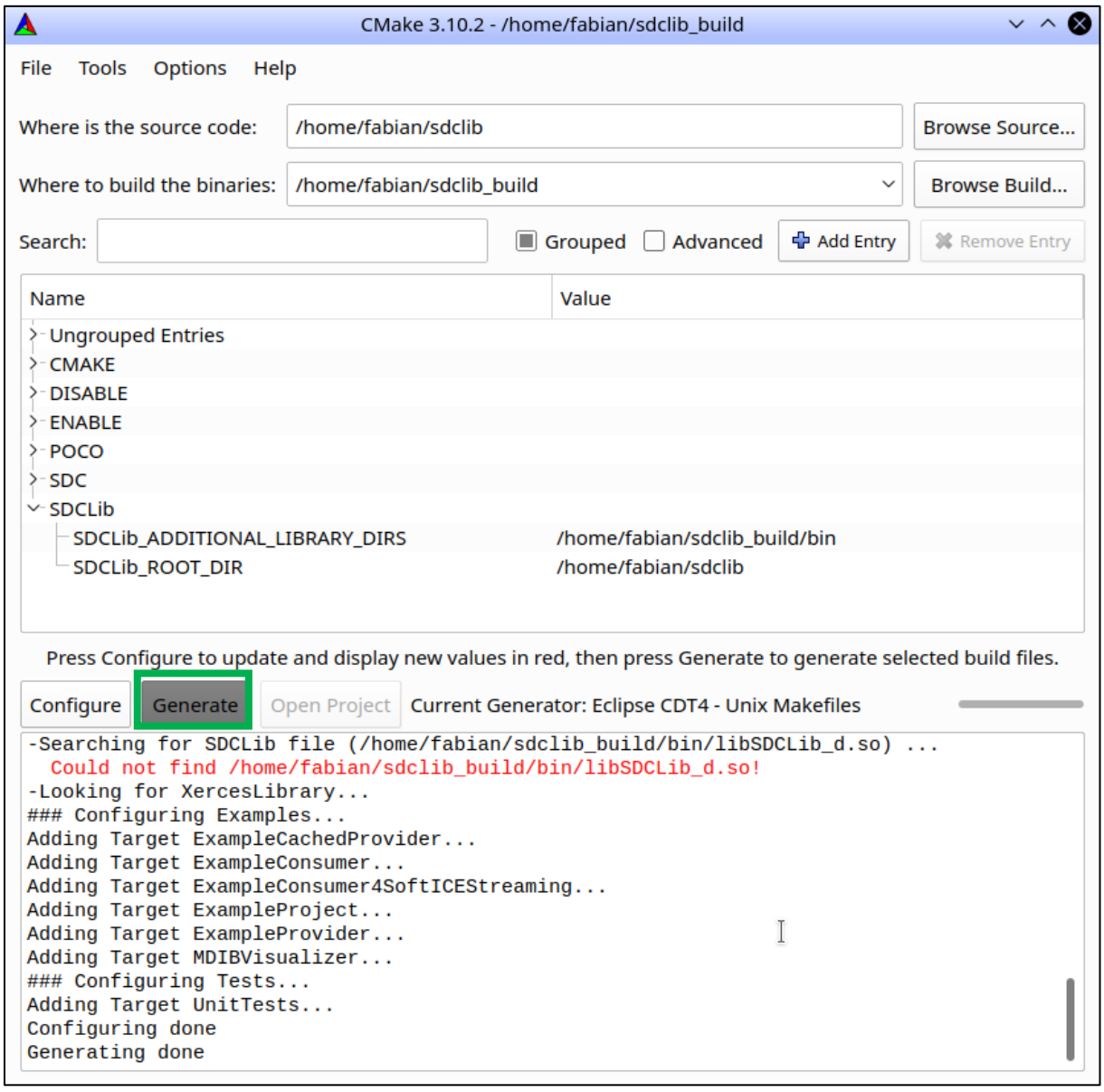
What to set here:

Type: PATH

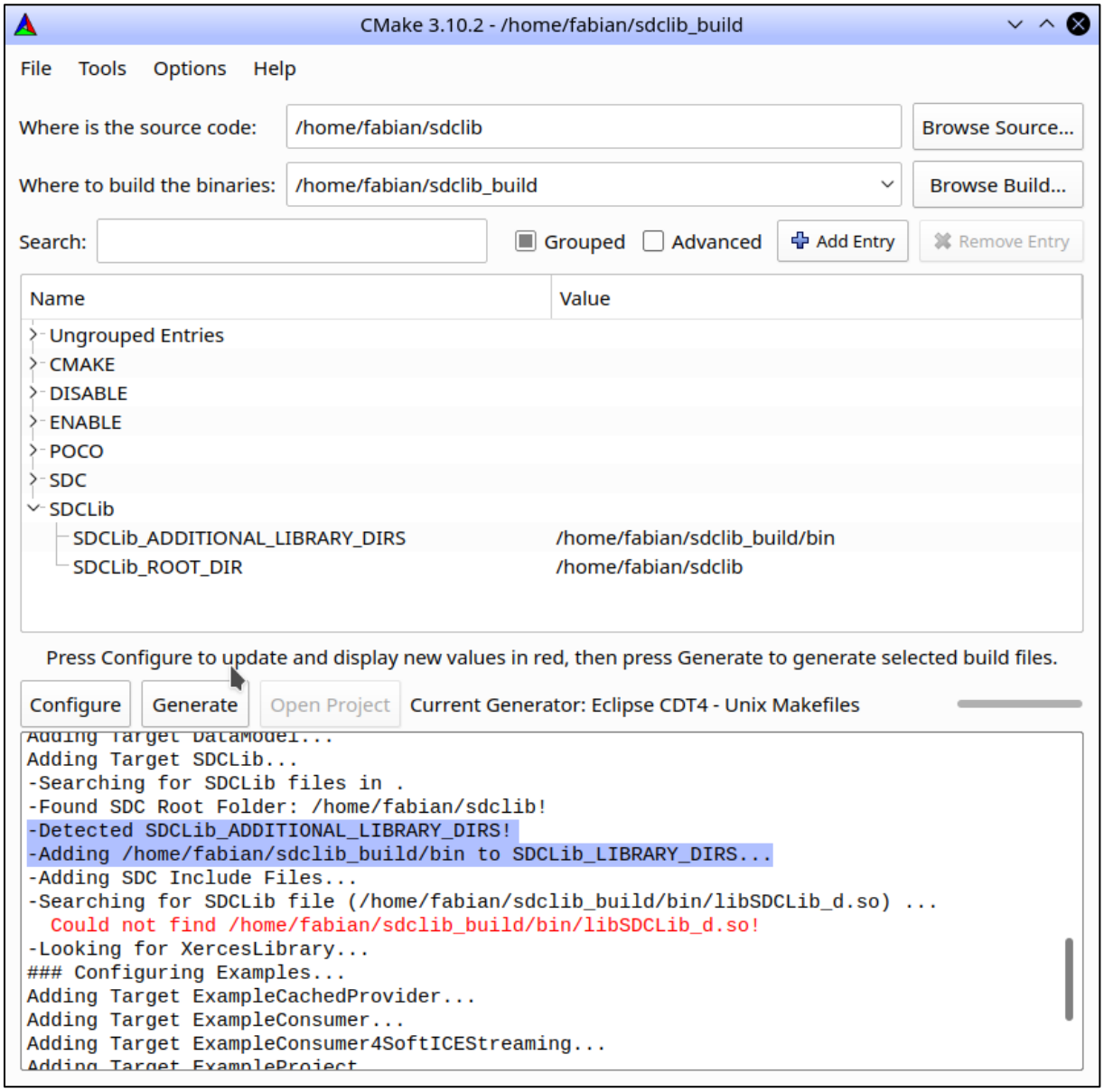
Value: <binary_build_folder>/bin

The new variable will appear (highlighted as mentioned before), click **Configure** again. You should see the new variable grouped to the other SDC variables.

Click **Generate**.



Of course we have not built the SDCLib yet, so the warning will persist. But you get a note that the new path has been added:



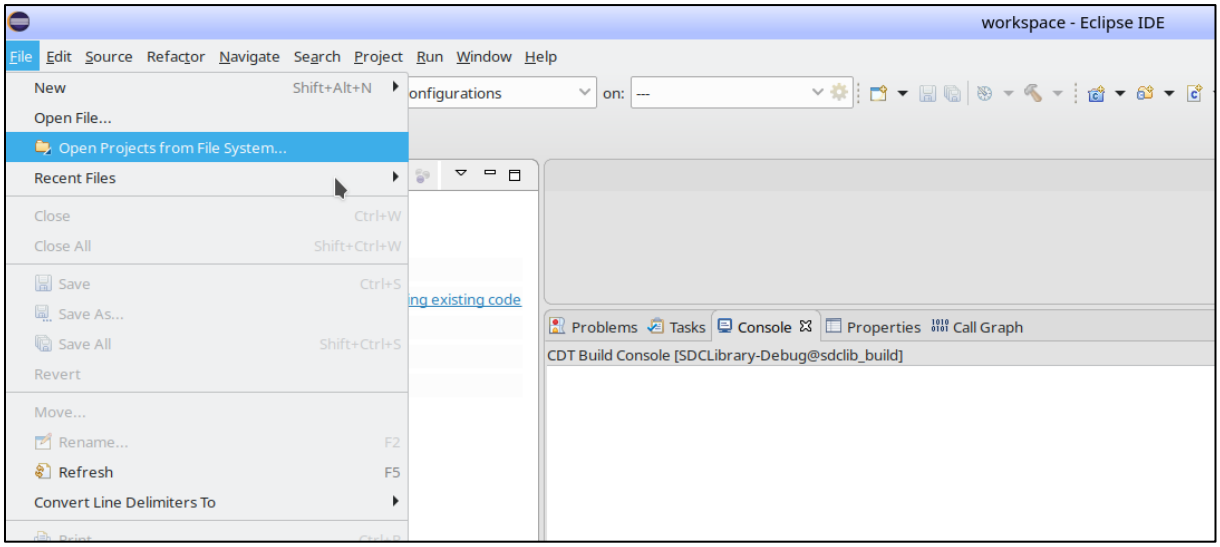
The Makefiles are now generated inside the specified build directory.

Note: If you specify *Unix Makefiles* as generator (right at the beginning), you could now navigate to the build folder and execute **make** to build the project.

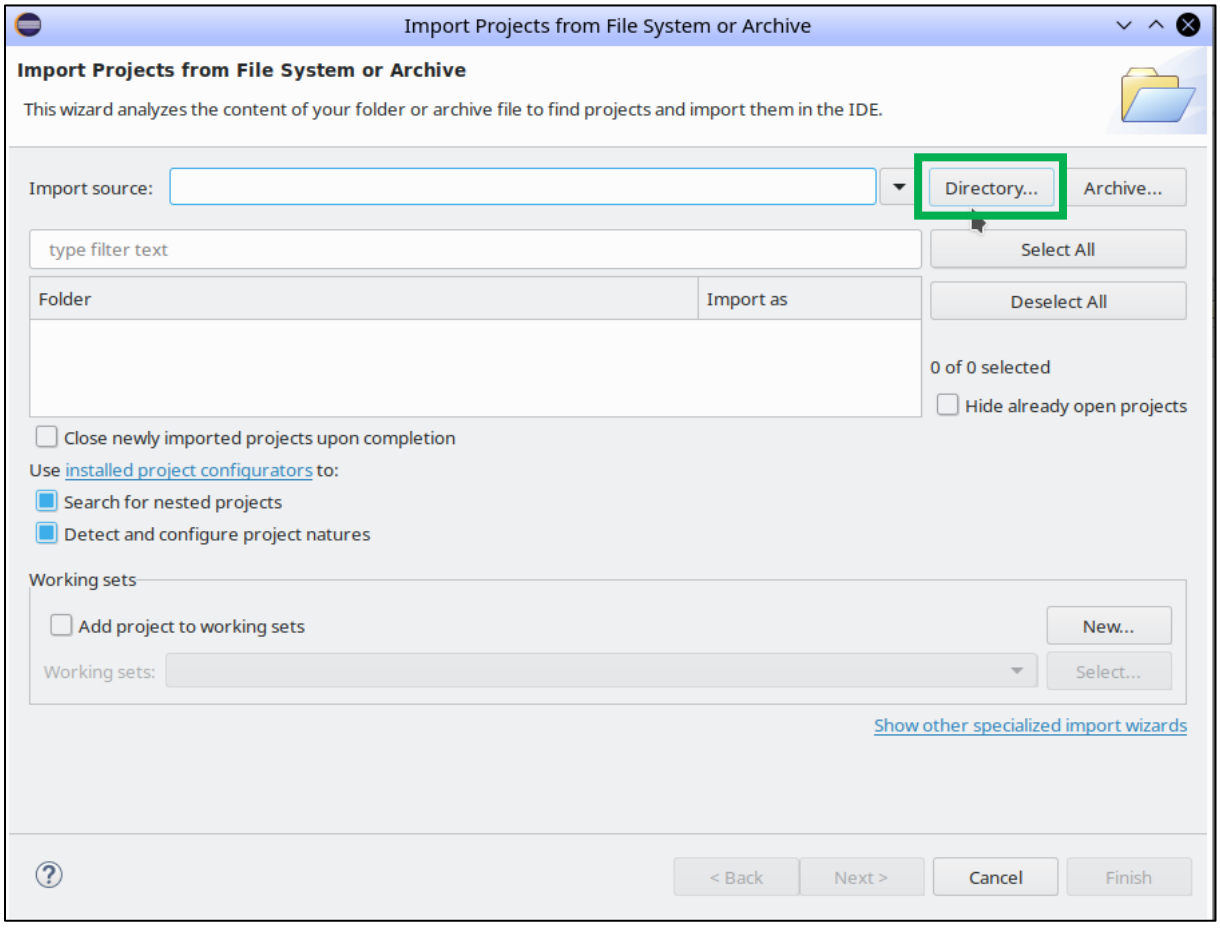
Importing the Makefiles

Open Eclipse [Here used Eclipse Version: 2019-03 (4.11.0)]

Select: **File -> Open Projects from File System…**



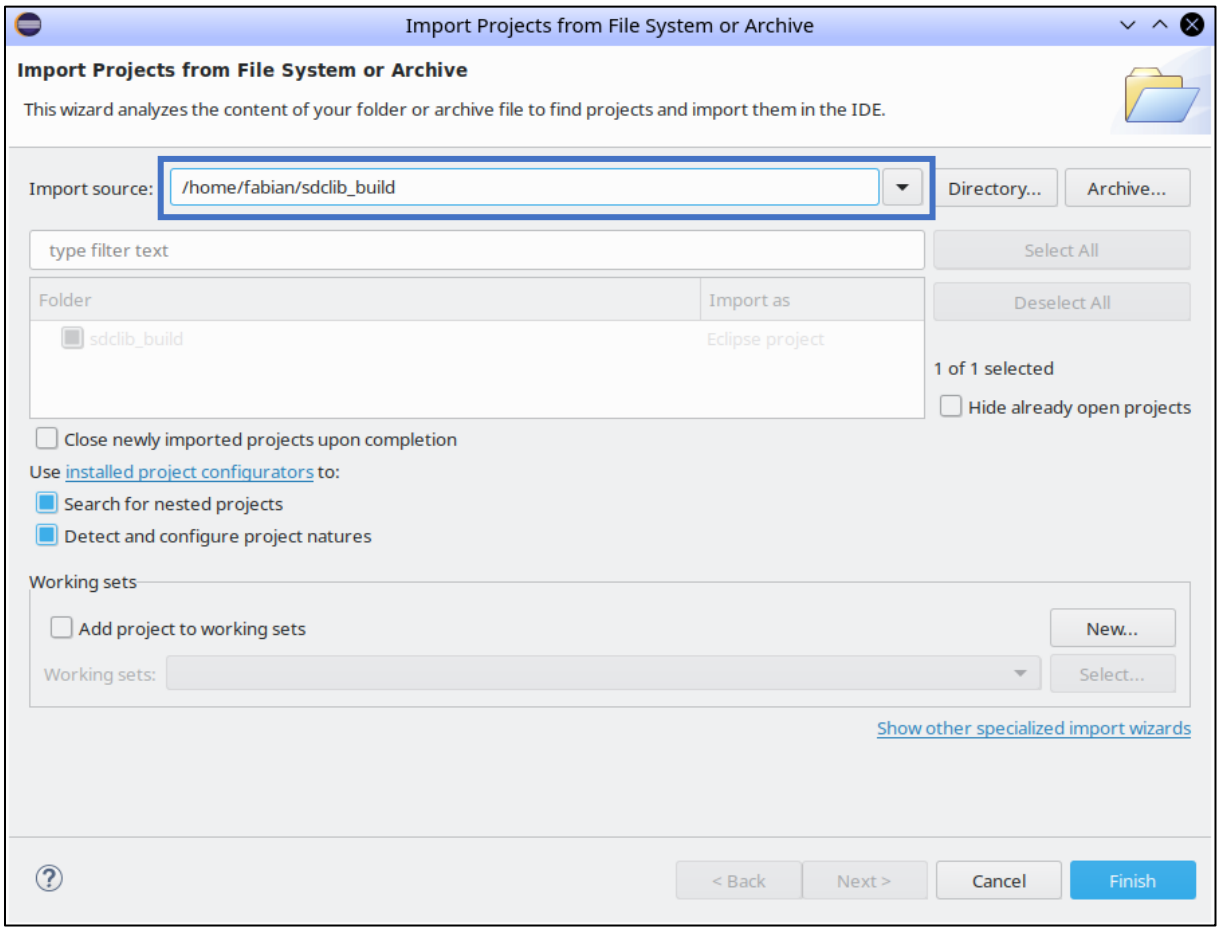
A Dialog will open:



Select **Directory**.

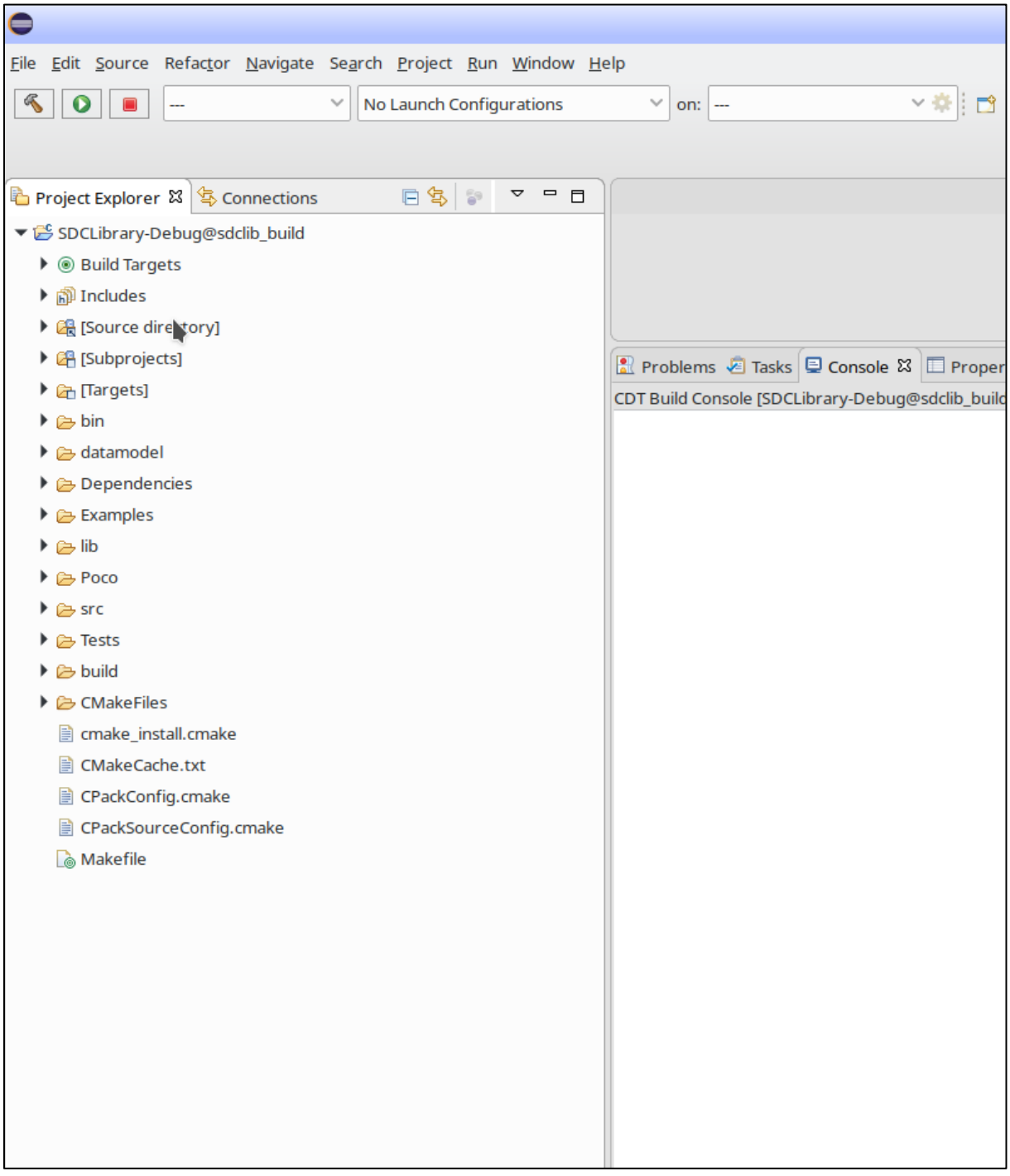
Select the folder where you **generated the Makefiles**.

In this example: /home/fabian/sdclib_build

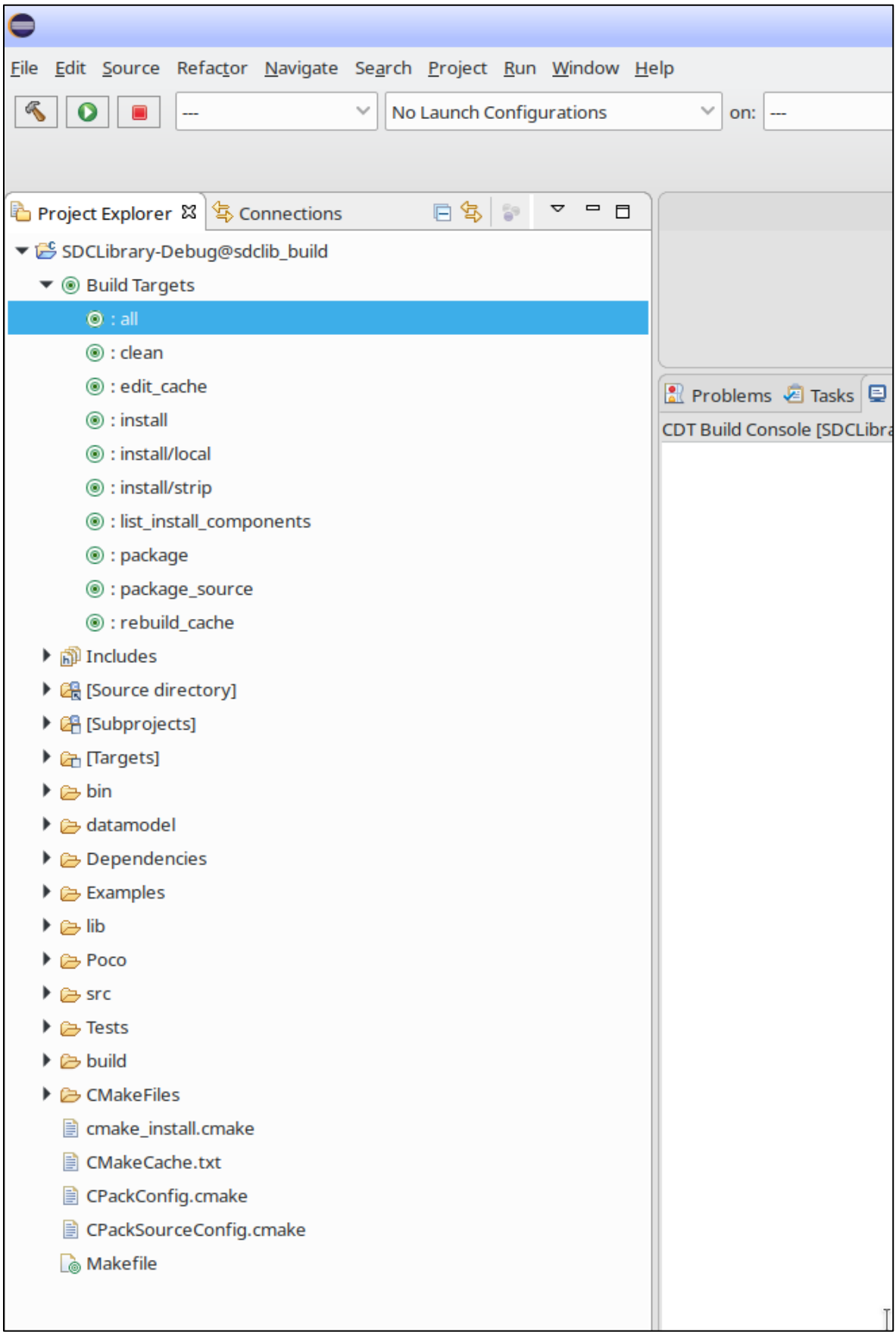


Click **Finish**.

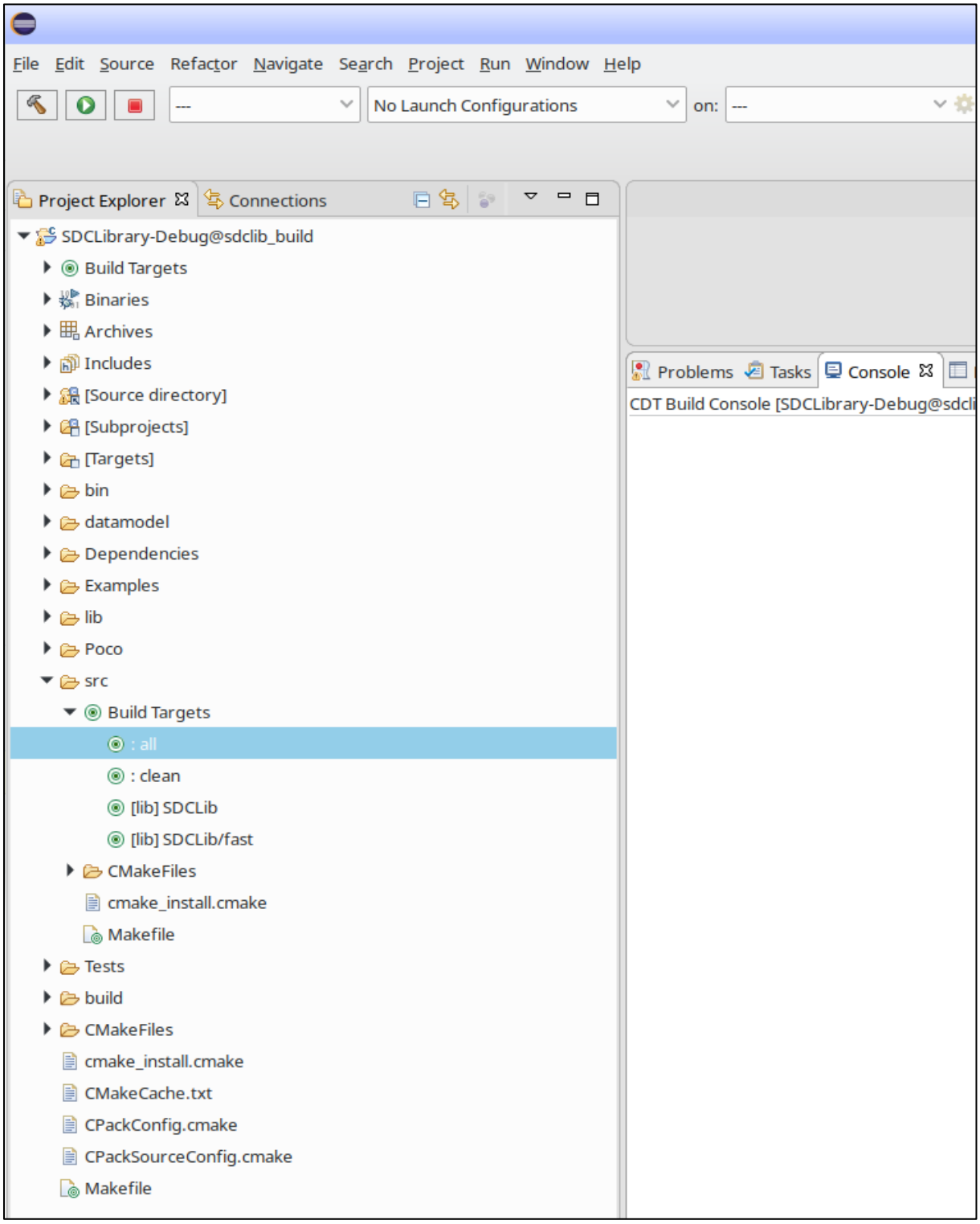
You should see the SDCLib Project with the corresponding build targets.



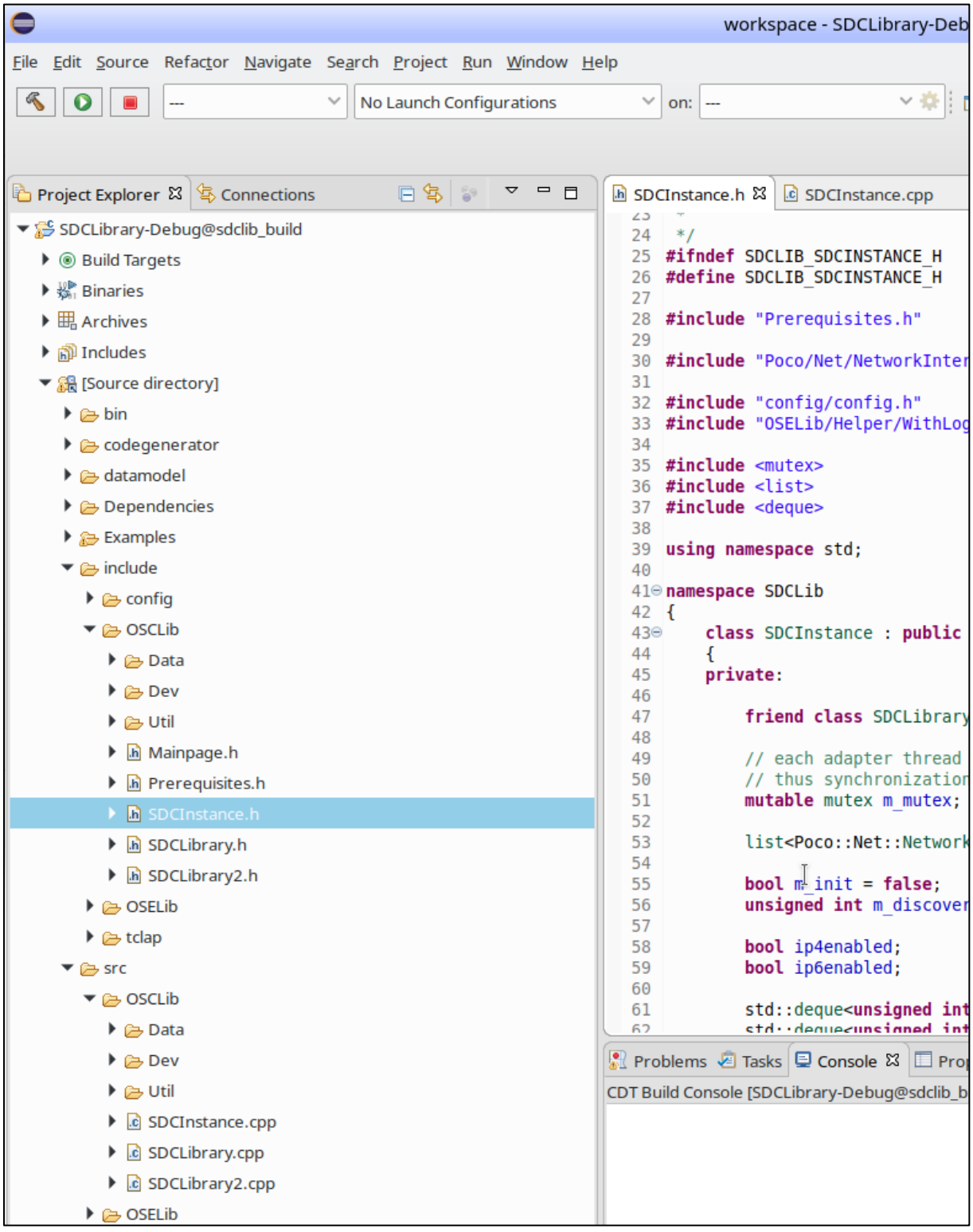
If you built Examples and / or Tests they will be built too at the top level target called „**all**“.



If you navigate into the src (source) folder, you can find the Build Targets for the SDCLib only:



Source and Include files can be found under **[Source directory]**.



Note: As long as you only change the file contents (work on the source code files) and don‘t make any major changes to the CMakeLists files or the whole project structure (moving/deleting files), there is no need to reconfigure or regenerate the project with cmake-gui!

Configuring the Indexer for c++11

To configure the source code indexer to work with the c++11 standard go to:

- *Project -> Properties -> C/C++ General -> C/C++ Include Path and Symbols*
- Add (or overwrite if already defined) Preprocessor Symbol:
    - Symbol: **_cplusplus**
    - Value: **201103L**