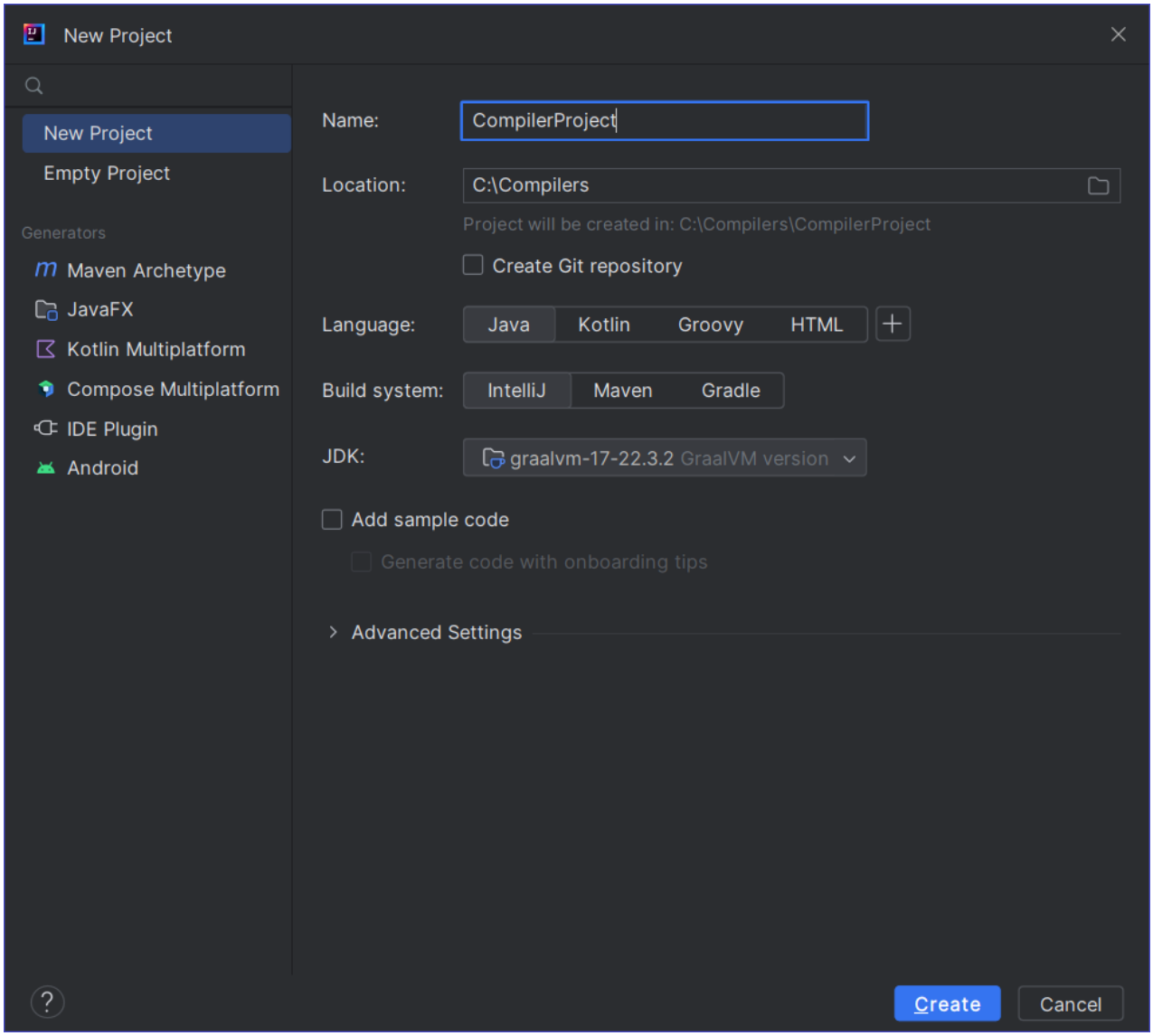
Tips on Creating a Multi-Module Project Using IntelliJ IDEA

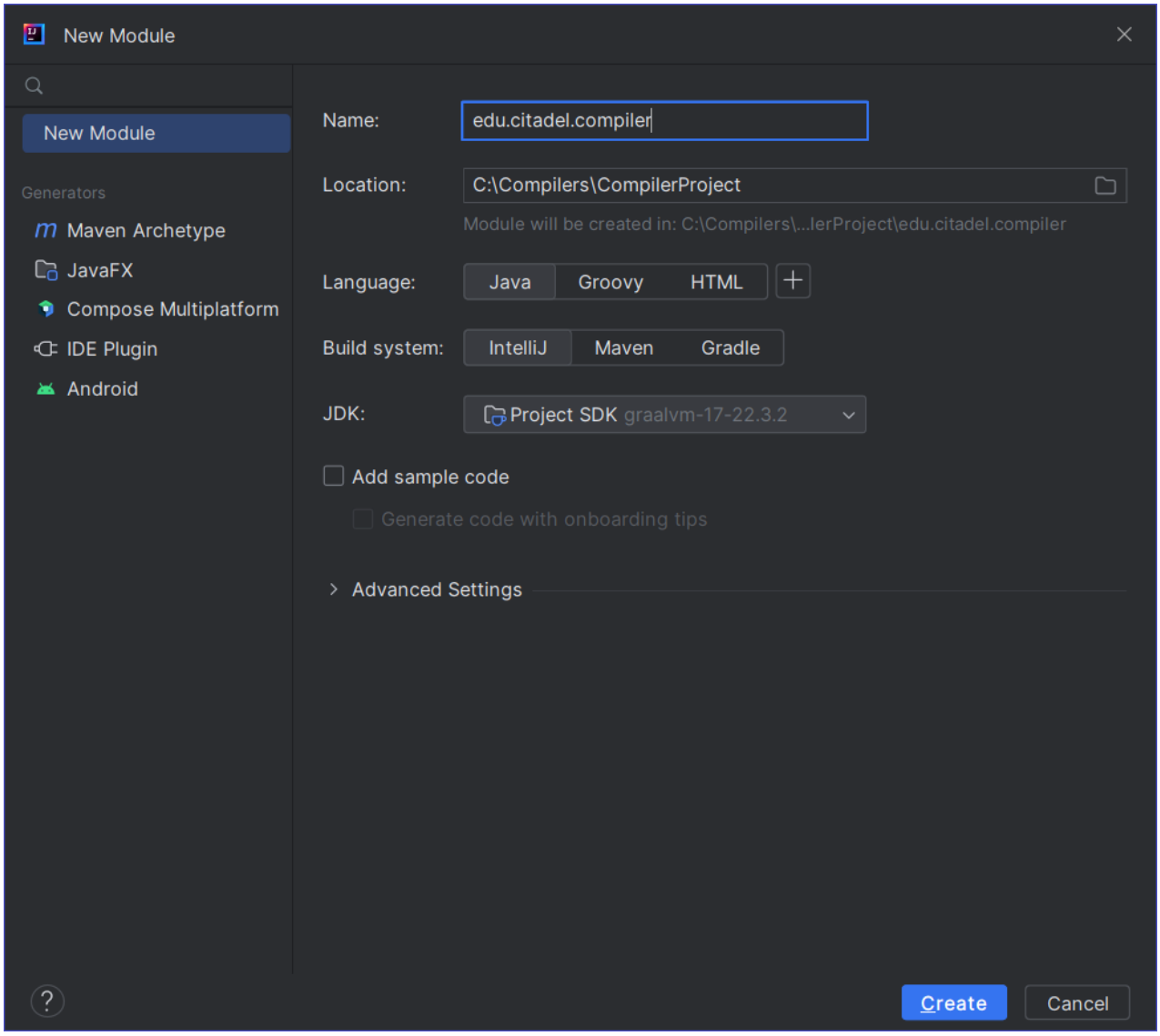
This handout illustrates *one* way to create a multi-module project for your compiler using IntelliJ IDEA. It creates the Compiler (`edu.citadel.compiler`) and CVM (`edu.citadel.cvm`) modules as described in Appendix A of the book, but other modules are created similarly.

1. Create a new Java project at your preferred location using IntelliJ as the build system. (If you are programming in Kotlin you can create a Kotlin project instead of a Java project, but there is no real advantage since each module will be created as a Java module configured to use Kotlin.)

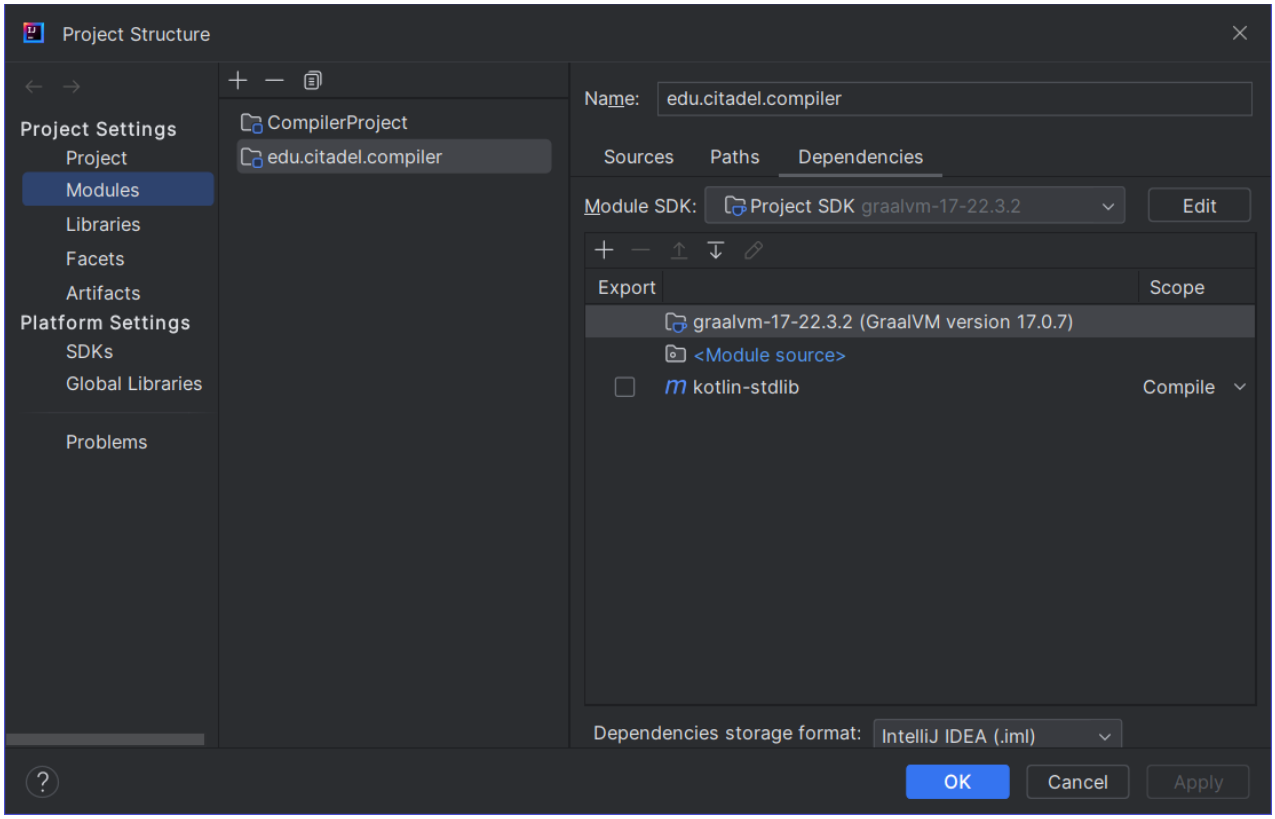


After creating the project, note that there is a folder named "`src`" immediately under `CompilerProject`. We will not put any Kotlin source code in this folder. Instead, we will create modules within the project, and each module will have its own separate "`src`" folders for Kotlin source code.

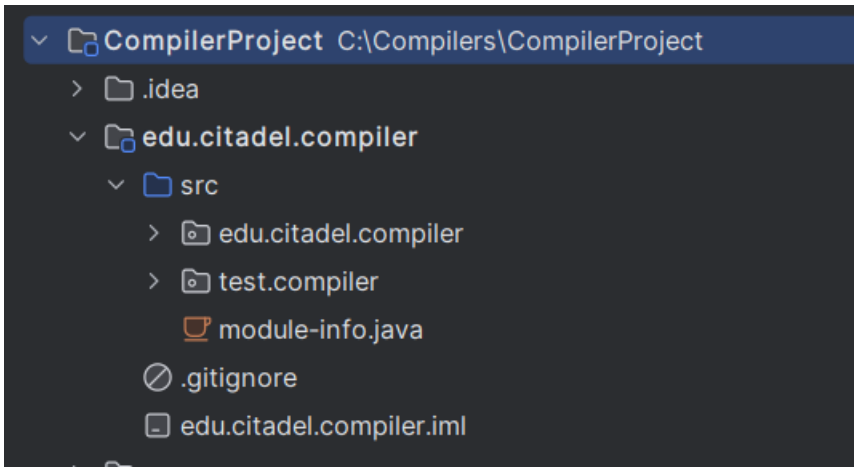
2. Right click on the newly created project, and select New - > Module ... Name the module `edu.citadel.compiler`.



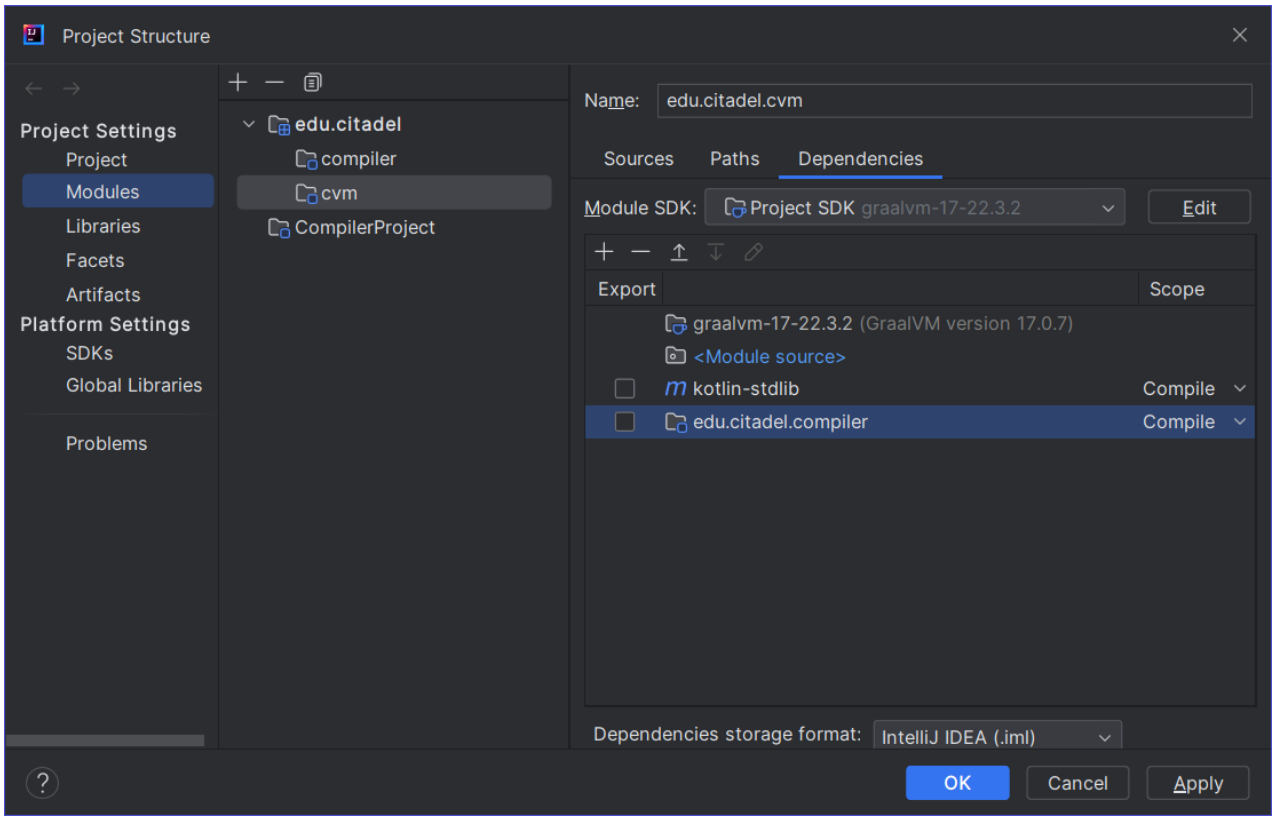
3. If you are implementing the compiler in Kotlin, right click on the newly created module and select Open Module Settings. Select the module `edu.citadel.compiler` and click + to add the Kotlin library. Click o.k. and then build the (empty) project to make sure that there are no errors.



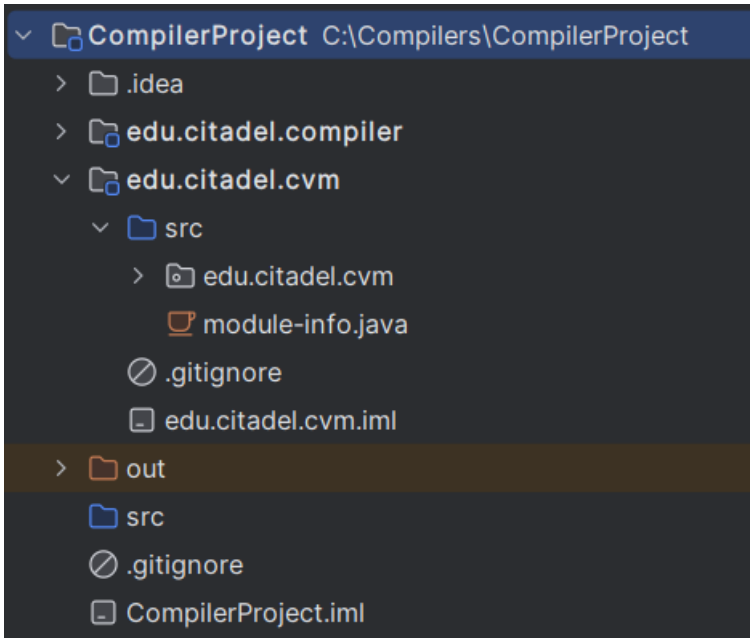
4. Once any module has been created, you can add new packages and Java or Kotlin source files to the module in the standard way, but all source code for this particular module is available in the GitHub repository as described in Appendix A. Using a file explorer (not IntelliJ), copy the `src` folder for this module (previously downloaded from GitHub) to the `src` folder in the newly created module. IntelliJ will detect the new code. If you expand the folder in IntelliJ it should look similar to the following. Again, build the project to make sure that there are no errors.



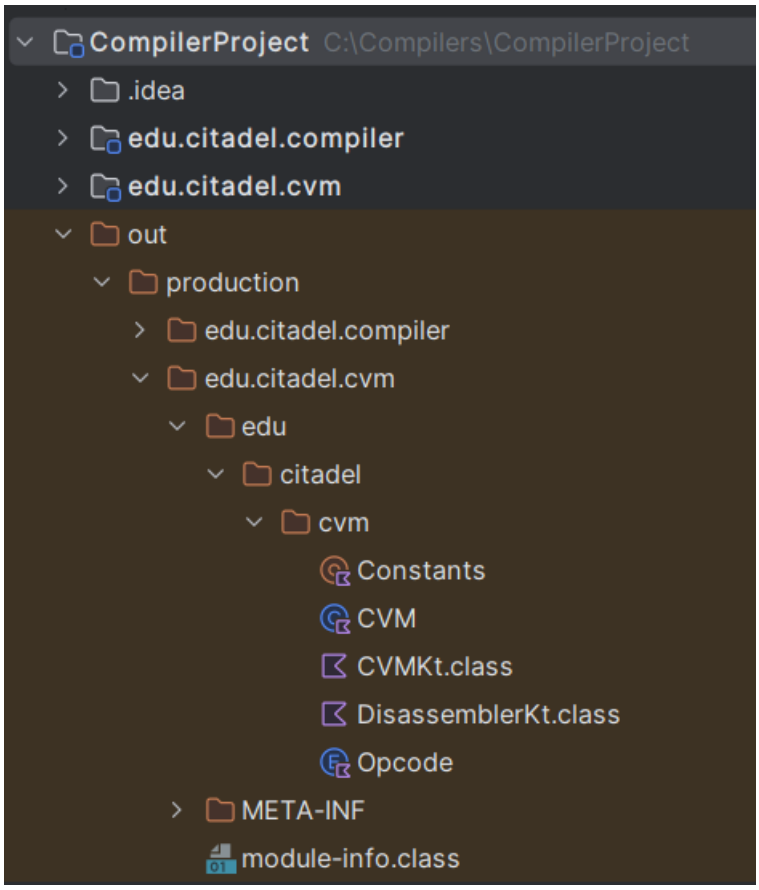
5. Use step 2 above (plus step 3 for Kotlin developers), create a new module named `edu.citadel.cvm`. This module has a dependency on module `edu.citadel.compiler`, so after adding the Kotlin library, click + again to add a module dependency for `edu.citadel.compiler`.



6. As before, use a file explorer to copy the `src` folder for edu.citadel.cvm (previously downloaded from GitHub) to the `src` folder in the newly created module. If you expand the folder in IntelliJ it should look similar to the following. Again, build the project to make sure that there are no errors.



IntelliJ puts the compiled class files in a folder named `out`. After rebuilding the project, you can expand subfolders of `out` to see the class files. The image below shows module `edu.citadel.cvm` expanded.



7. Using an approach similar to the above, create module `edu.citadel.assembler`, with dependencies on modules `edu.citadel.compiler` and `edu.citadel.cvm`. Then create module `edu.citadel.cprl`, also with dependencies on modules `edu.citadel.compiler` and `edu.citadel.cvm` (but no dependency on module `edu.citadel.assembler`).