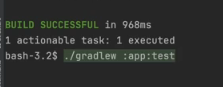
<https://dpeuniversity.gradle.com/courses/012de84f-fcd3-45d4-9c4c-284382eb3f3f/activities/dee43e9e-5d8c-407e-b8c3-55bc64c7cc34>

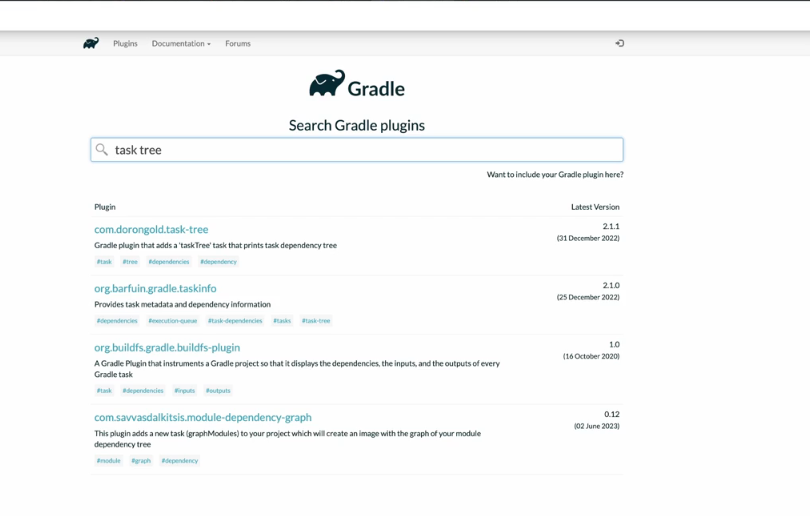
Applying plugins – 3:27 minute

Learn how to search for plugins in the plugin portal and apply them.

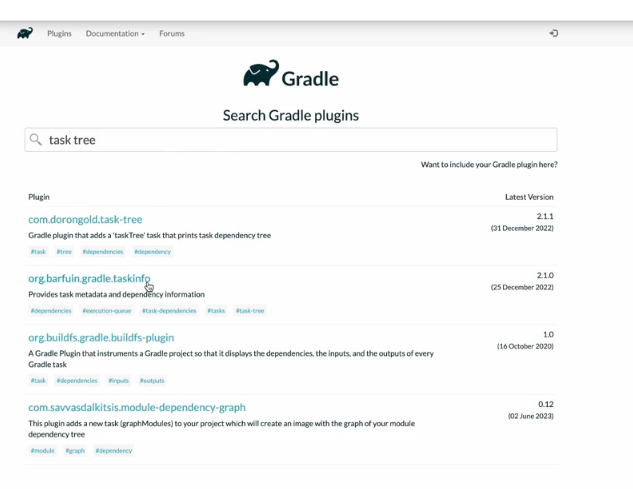
Now let’s take a look at an example of searching for and applying a community plugin. If you run the test task,



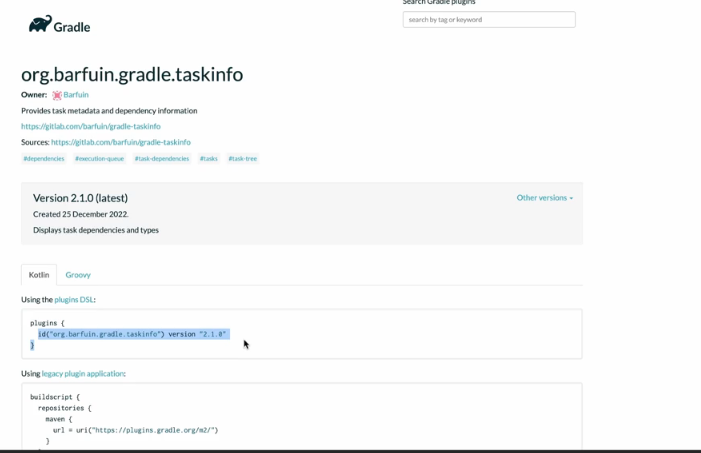
we can see a number of tasks had to run before including the compiled Java task. It’s not clear, however, what the dependencies are between the prerequisite tasks. It would be nice to see a graphical or tree representation of the task dependencies. Gradle build tool does not have a task that can show this. Let’s search for a community plugin that can do this.



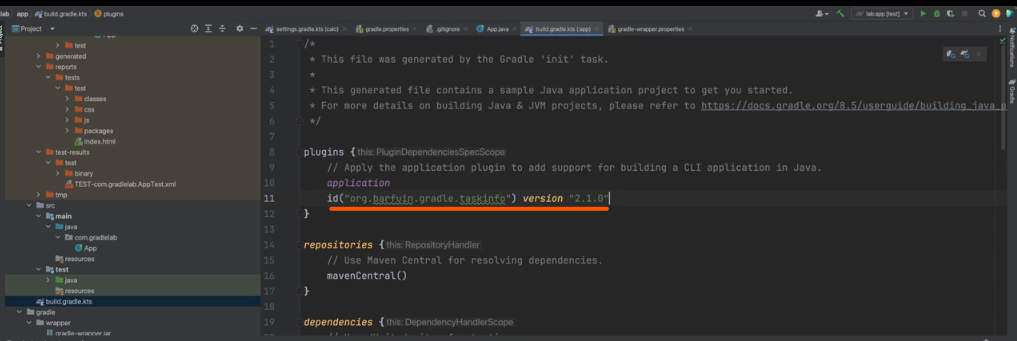
In the Gradle plugin portal, we can search for task tree plugins. There are a few results that come up, I like to use the plugin that is maintained and therefore recently updated. After looking at the dates and descriptions, this plugin seems to provide the functionality we want.



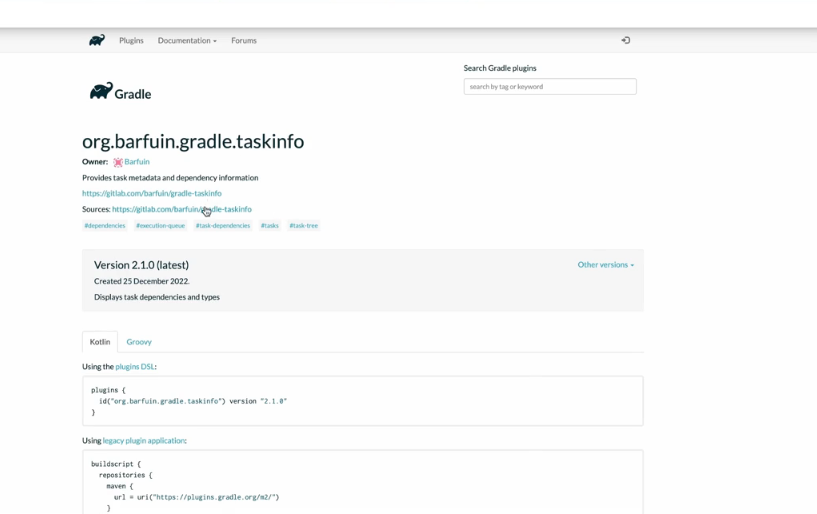
We can see how to apply the plugin in either Kotlin or Groovy configuration. Let’s apply it.

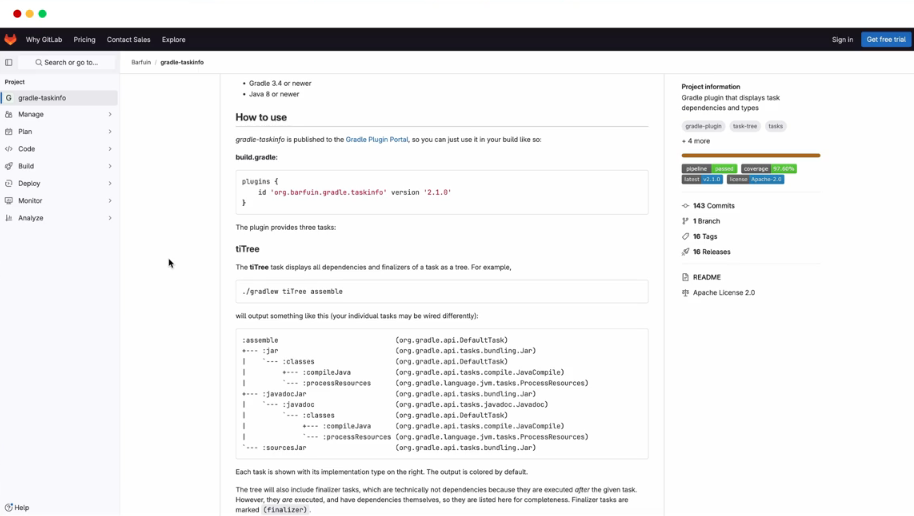


In the app sub project build file, we can go to the plugin section and update it.

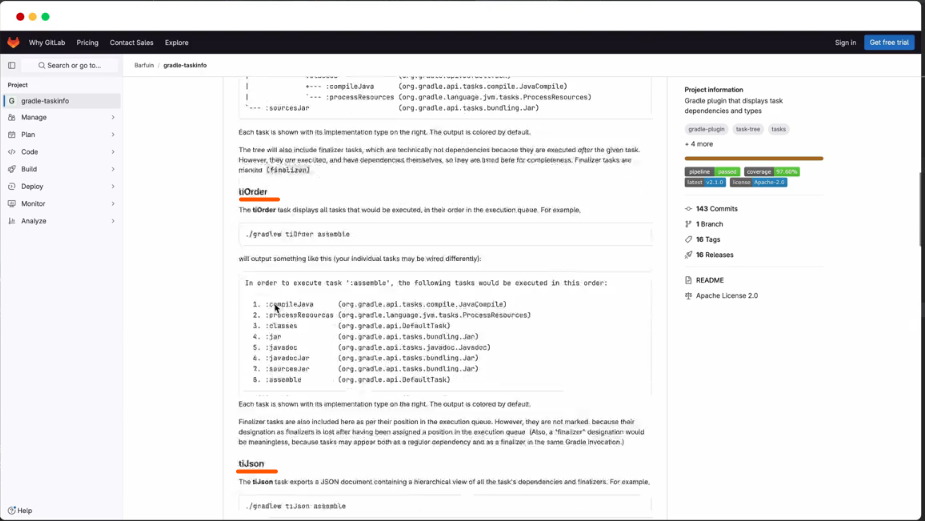


We can read the documentation for the plugin to learn about the tasks it adds. Note that the documentation here also shows how to apply the plugin.

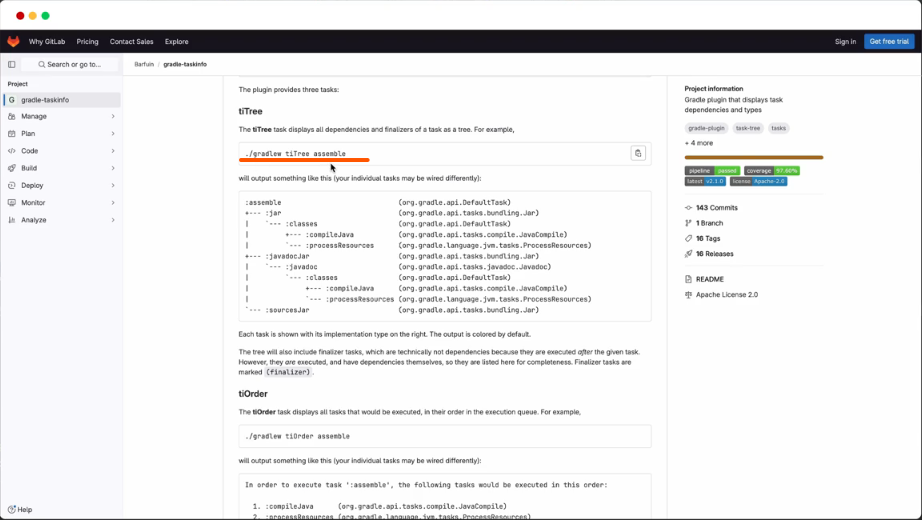




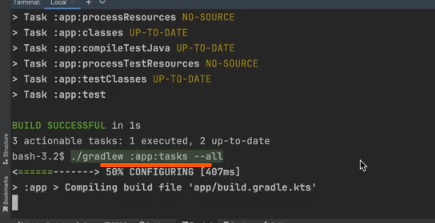
However, it’s only for Groovy, there is no sample provided for Kotlin. We can see the plugin provides three tasks tiTree, tiOrder and tiJson.

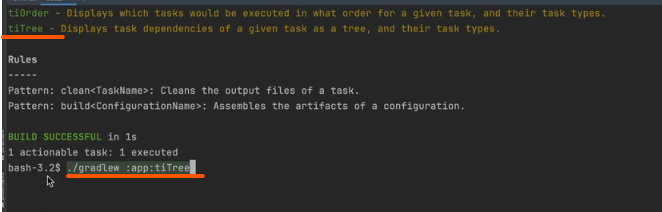


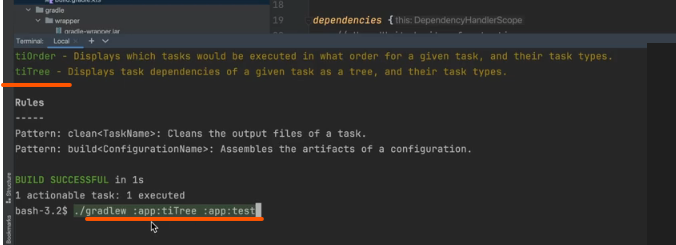
This is what we’re looking for. So the task is tiTree, we can pass in an argument, which is the task we wanna inspect.



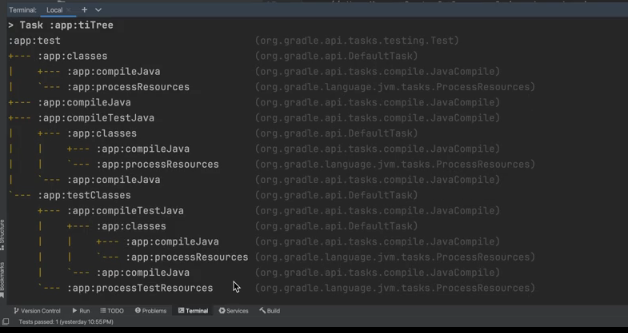
We can look at the available task that the plugin added, and we can run the tiTree task.

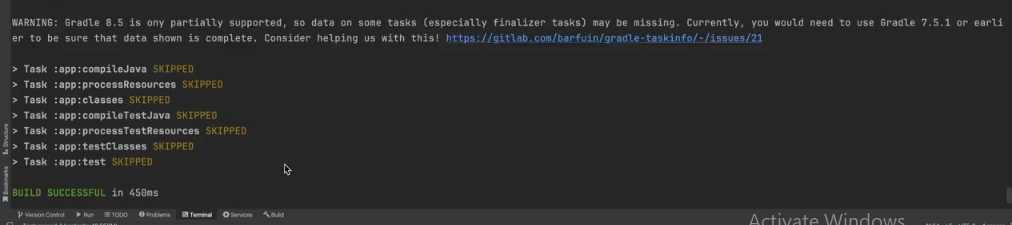






We’re gonna run the tiTree task on the app sub project, for the test task on the app sub project.





We can see the output and understand the task dependencies better. It is recommended to be cautious and do due diligence when applying community plugins. Popular plugins owned by trusted authors are generally safe, but one needs to be careful when applying less used plugins contributed by individual developers.