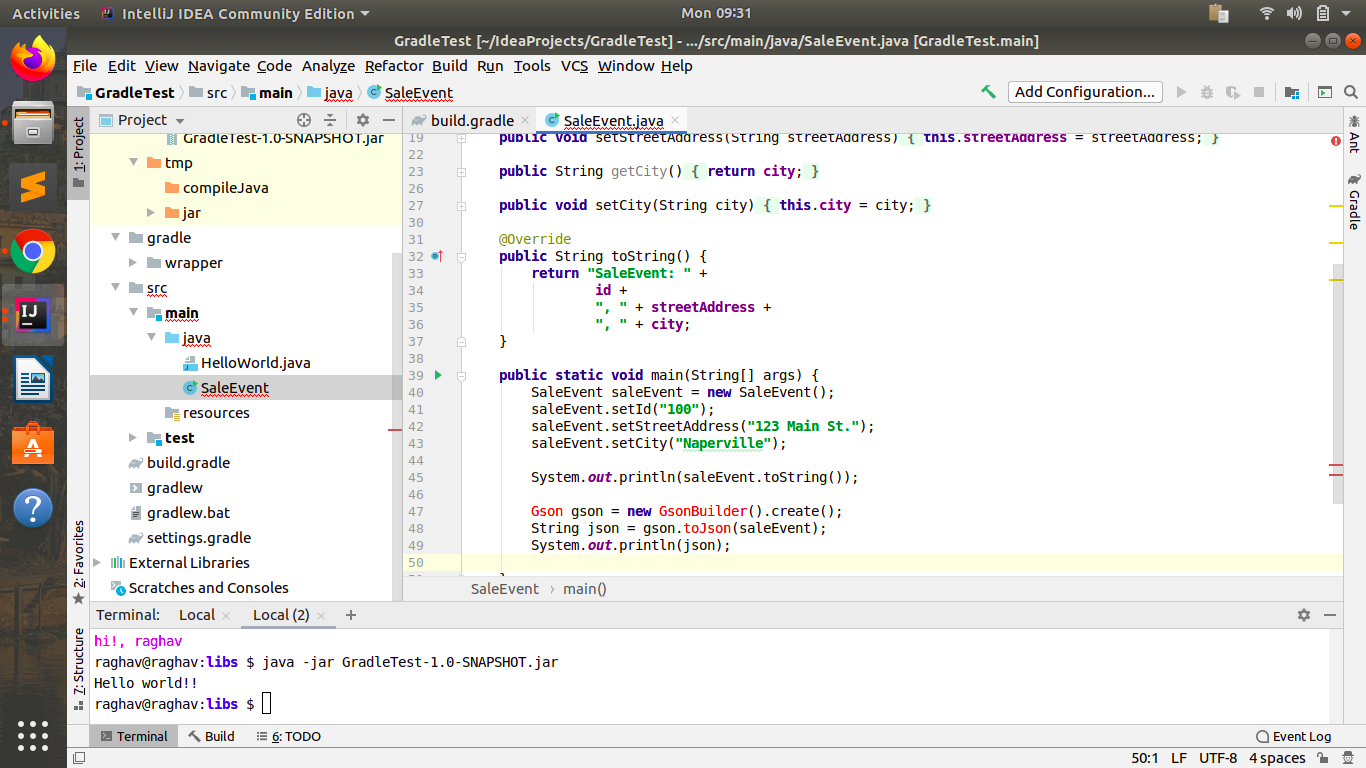
**GRADLE ASSIGNMENT**

**RAGHAV GUPTA**

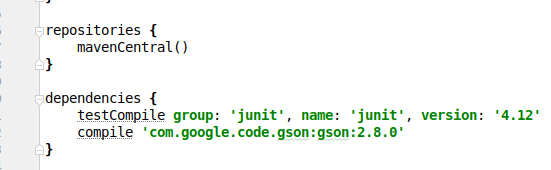
**------------------------------------------------------------------**

1. **Add a gradle dependency and its related repository url.**

I have used the GSON repository. Initially it was giving error in the source code but after adding the dependency in the build.gradle file and importing the classes, the error went away.

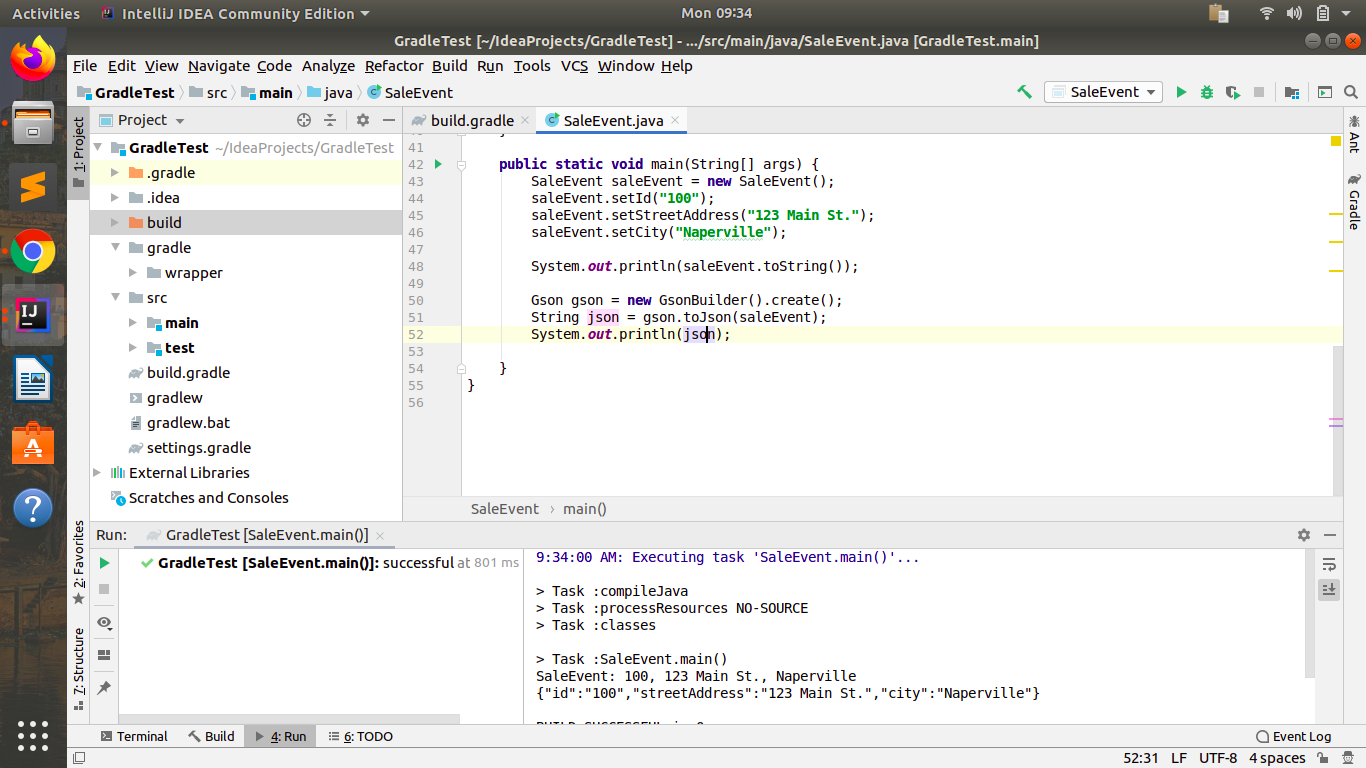


Then,

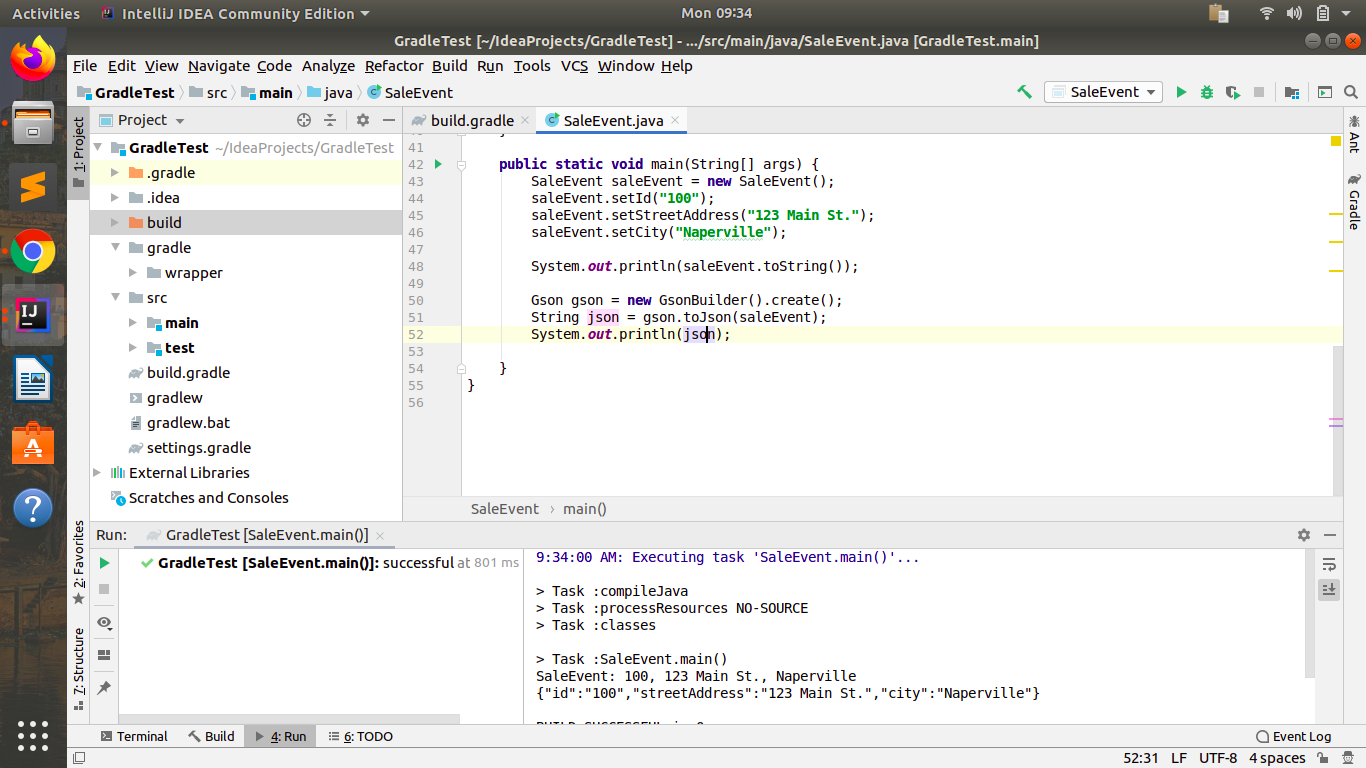


Then do ‘gradle build’

And now error is gone.

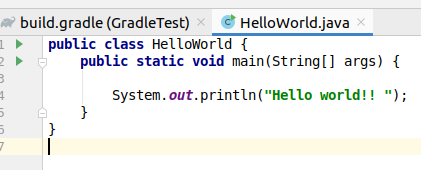


OUTPUT ->

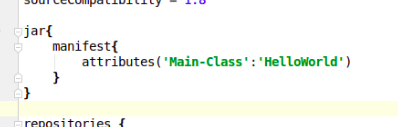


1. **Using java plugin, make changes in the manifest to make the jar executable. Using java -jar JAR\_NAME, the output should be printed as "Hello World"**

* I made a HelloWorld class.



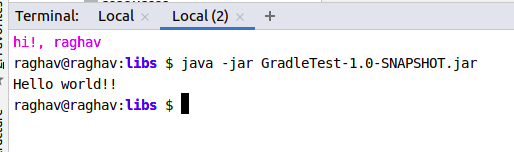
* prepared a jar file by doing ‘gradle build’
* Setting the configuration for jar file



Then run it by the

* ‘java -jar fileName’ command

Output ->



1. **Differentiate between the different dependency scopes: compile, runtime, testCompile, testRuntime using different dependencies being defined in your build.gradle.**

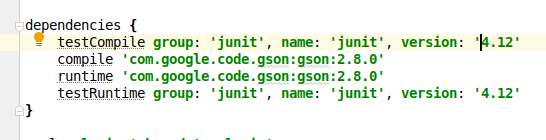
**compile:** Dependencies required at compile time only such as source-only annotations or annotation processors. Compile dependencies are available in all classpaths of a project.

They are automatically available in runtime also.

**runtime:** The dependencies with this scope are required at runtime, but they're not needed for compilation of the project code. Because of that, dependencies marked with the runtime scope will be present in runtime and test classpath, but they will be missing from compile classpath.

**testCompile:**  testCompile is a group of dependencies that might be needed only for testing while compile is the group of dependencies that might be needed to build the application . The testCompile configuration contains the dependencies which are required to compile the tests of the project.

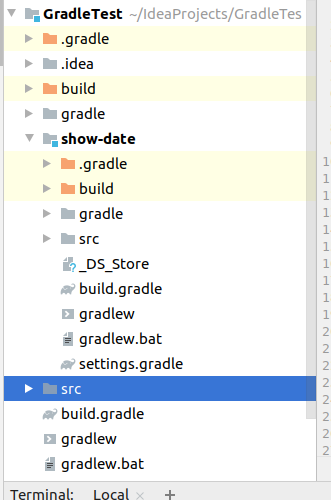
**testRuntime:** The testRuntime contains the dependencies which are required when our tests are run. This configurations contains the dependencies added to compile, runtime, and testCompile configurations.

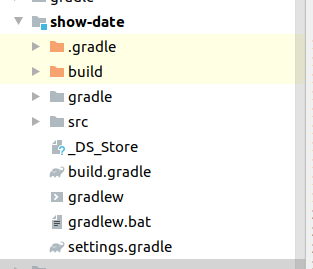


I need no such specific testRuntime dependency, so i added the same dependency in all to show the syntax only.

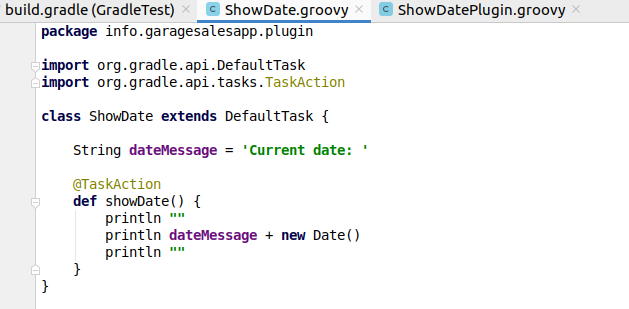
1. **Create a custom plugin which contains a custom task which prints the current date-time. Using that plugin in your project, execute that task after the jar task executes.**

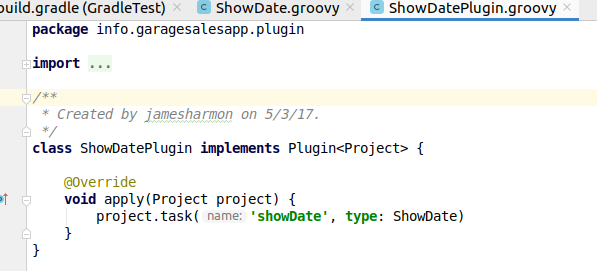
* First of all we create a new module in the root level of the project.



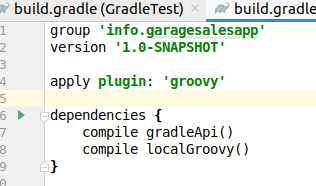


* Then we write the plugin ShowDate plugin

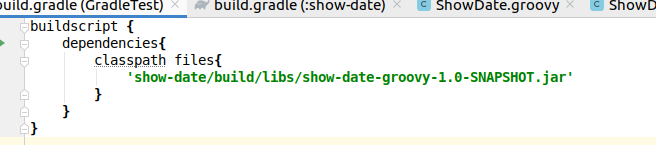




* Add dependencies in the build.gradle file of the plugin ShowDate

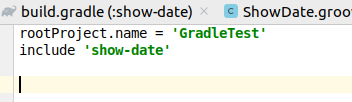


* apply plugin in the build.gradle of the main project

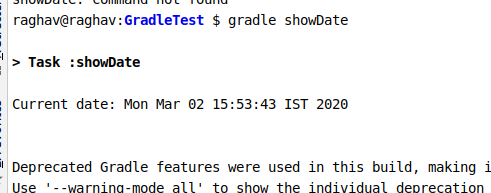




* include the plugin to gradle.settings of main project.

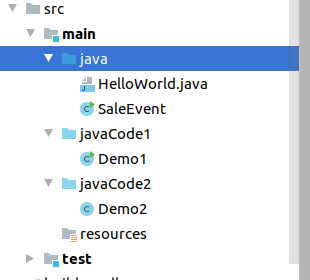


* Build the plugin first and then the main project.
* Run the task showDate of the plugin from main project terminal.

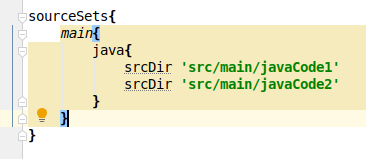


1. **Instead of using default source set, use src/main/javaCode1, src/main/javaCode2 to be taken as code source. Make sure that the JAR created contains files from both the directories and not from src/main/java.**

* Create 2 directories inside src/main in the main project
* Add 1 class in each of them

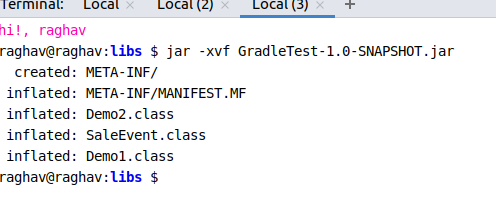


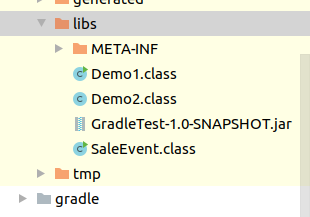
* Modify the sourceSets property in the build.gradle so as to change the sources to be included in the jar file.



* Run gradle build and the jar will be created
* Extract the jar file form the build/lib folder and see the output

jar tf GradleTest-1.0-SNAPSHOT.jar

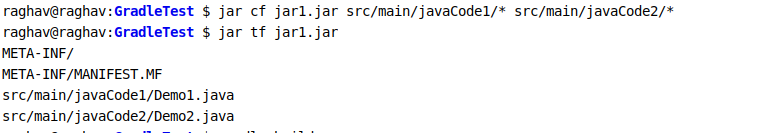




==============ANOTHER METHOD ==================

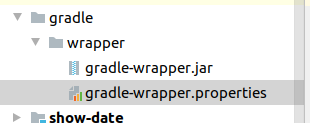
* Use the command

jar cf jar1.jar src/main/javaCode1/\* src/main/javaCode2/\*

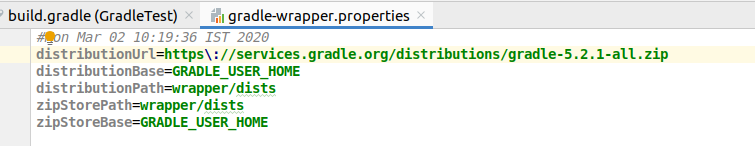


1. **Override the Gradle Wrapper task to install a different version of gradle. Make sure that the task written in Q4 also executes with it.**

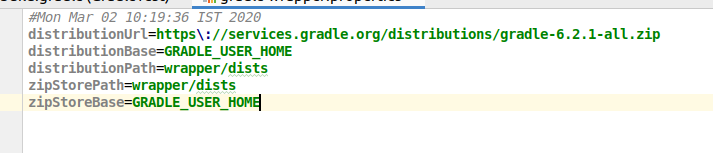
* Go to this file first



* Intially the version is 5.2.1 -



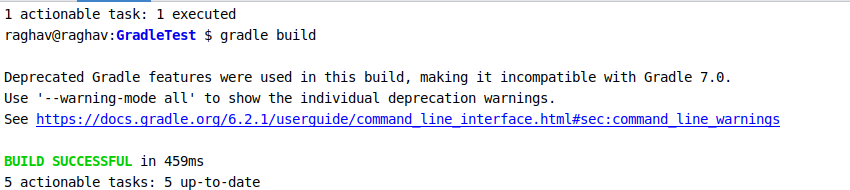
* Change the version to 6.2.1



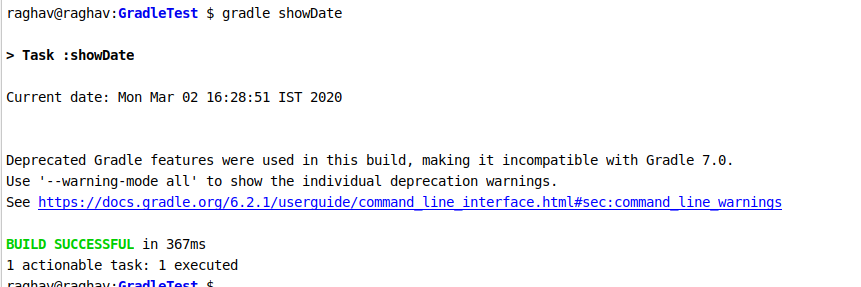
* Now run ’./gradlew’



* We see that it downloads the new version.
* Build again now



* Run the showDate task of the plugin and it runs successfully.



1. **Run the gradle profile command and attach the resulting files.**

