GRADLE EXERCISE

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

CODE:-

plugins **{**

id **'java'**

**}**

group **'org.example'**

version **'1.0-SNAPSHOT'**

sourceCompatibility = 1.8

println **"Hello World"**

repositories **{**

maven **{**

url **'https://repo1.maven.org/maven2/'**

**}**

**}**

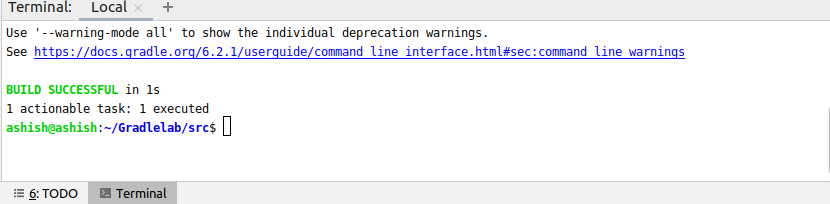
dependencies **{**

testCompile **group**: **'junit'**, **name**: **'junit'**, **version**: **'4.12'**

compile **'com.google.code.gson:gson:2.8.0'**

**}**

OUTPUT:-



2.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"

CODE:-

apply **plugin** : **'java'**

version **'1.0-SNAPSHOT'**

sourceSets**{**

main**{**

java.srcDirs = [**'src/main/java'**]

**}**

**}**

jar **{**

manifest **{**

manifest**{**

attributes( **'main-Class'**: **'Hello'**,

**"class-path"**: configurations.**compile**.collect **{**it.getName()**}**.join(**' '**))

**}**

**}**

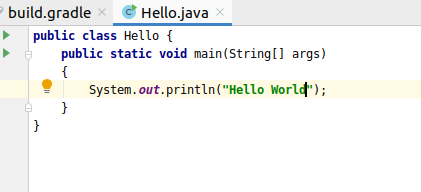
**}**

sourceCompatibility = 1.8

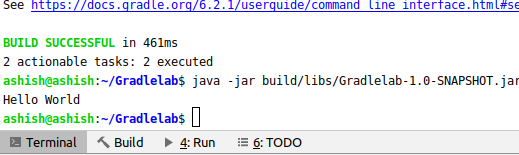
repositories **{**

mavenCentral()

**}**



OUTPUT:-



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

Ans:-

a)compile

This is where you should declare dependencies which are required at compile time, but should not leak into the runtime. Compile is the group of dependencies you need to build your application.

b)runtime

This is where you should declare dependencies which are only required at runtime, and not at compile time.

c)testcompile

This is where you should declare dependencies which are required at test compile time, but should not leak into the runtime. TestCompile is a group of dependencies that you need only for testing

d)testruntime

This is where you should declare dependencies which are only required at test runtime, and not at test compile time.apply plugin : 'java'

group 'org.example'

version '1.0-SNAPSHOT'

sourceCompatibility = 1.8

println "Hello World"

repositories {

maven {

url 'https://repo1.maven.org/maven2/'

}

}

dependencies {

testCompile group: 'junit', name: 'junit', version: '4.12'

compile 'com.google.code.gson:gson:2.8.0'

runtime group: 'org.springframework', name: 'spring-core', version: '2.5'

}

4.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.

Code:-

apply **plugin** : **'java'**

group **'org.example'**

version **'1.0-SNAPSHOT'**

sourceCompatibility = 1.8

jar **{**

manifest **{**

attributes **'Main-Class'**: **'Hello'**

**}**

**}**

println **"Hello World"**

repositories **{**

maven **{**

url **'https://repo1.maven.org/maven2/'**

**}**

**}**

dependencies **{**

testCompile **group**: **'junit'**, **name**: **'junit'**, **version**: **'4.12'**

compile **'com.google.code.gson:gson:2.8.0'**

**}**

**class** GreetingPlugin **implements** Plugin<Project> {

**void** apply(Project project) {

project.task(**'hello'**) **{**

doLast **{**

println **new** Date()

**}**

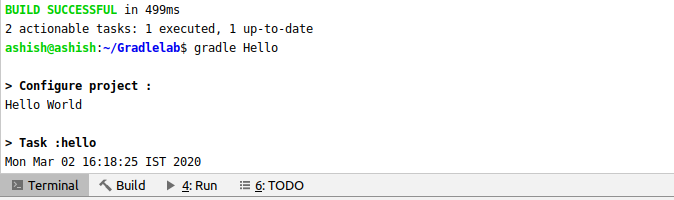
**}**

}

}

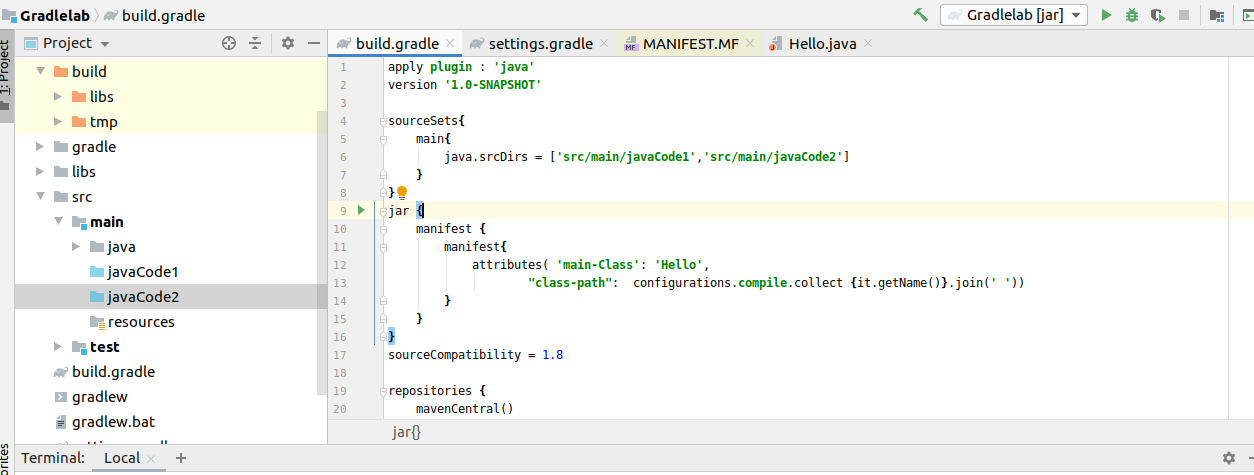
apply **plugin**: GreetingPlugin

Output:-



5.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.

Code :-



6.directories and not from src/main/java. Override the Gradle Wrapper task to install a different version of gradle. Make sure that the task written in Q4 also executes with it.

Code:-

apply **plugin** : **'java'**

sourceSets**{**

main**{**

java.srcDirs = [**'src/main/javaCode1'**]

**}**

**}**

jar **{**

manifest **{**

attributes **'Main-Class'**: **'Hello'**

**}**

**}**

println **"Hello World"**

dependencies**{**

*/\*// https://mvnrepository.com/artifact/com.google.code.gson/gson*

*compile group: 'com.google.code.gson', name: 'gson', version: '2.8.6'\*/*

runtimeOnly **group**: **'org.springframework'**, **name**: **'spring-core'**, **version**: **'2.5'**

**}**

**class** GreetingPlugin **implements** Plugin<Project> {

**void** apply(Project project) {

project.task(**'hello'**) **{**

doLast **{**

println **new** Date()

**}**

**}**

}

}

*// Apply the plugin*

apply **plugin**: GreetingPlugin

repositories**{**

mavenCentral()

**}**

Output:-



7.Run the gradle profile command and attach the resulting files.

Gradle --profile

