GRADLE

LAB ASSIGNMENTS

Pre-Requisitions:

1. Create one account with GitHub (https://github.com/)
    a. Create one repository in the name of "**demoRepo**"
2. JDK 1.6 or higher
3. Install Eclipse or STS (Spring Tool Suite)
4. Install BuildShip plugins in your Eclipse IDE
    a. Help → Install new Sofwares → Add the below link
        i. **http://download.eclipse.org/buildship/updates/e45/releases/1.0**
    b. Click next to install the plugin. Once install Restart the Eclipse / STS

Lab 1:

Download the ProductApp_Gradle application from GitHub and do the followings:

- Specify the default java compiler as jdk1.8
- Change Repository as Maven
- Add below dependencies in that project
  - junit-4.12.jar
  - mockito-all-1.10.19.jar
  - commons-lang3-3.3.2.jar

Steps:

1. Open GitHub URL (https://github.com/Training2017/MCD)
2. Click **ProductApp_Gradle.zip folder,** next page click download link. The zip file will be downloaded.
3. Unzip the folder
4. Go to Eclipse IDE, choose import ->Gradle -> Gradle Project into Workspace
5. Under select root directory click browse button.
6. Go to the unzip folder location and choose the folder in the name of ProductApp_Gradle(root directory of your project). Click next.
7. Select local installation directory, click browse button to select Gradle HOME directory.
8. Click Next →Finish
9. Now the Product Application project will be imported in your local machine.
10. Once the project imported into Eclipse IDE, locate build.gradle file.

```
// Apply the java plugin to add support for Java
apply plugin: 'java'

group = 'ProductApp_Gradle'
version = '0.0.1-SNAPSHOT'

description = 'Product Application Using Gradle'



sourceCompatibility = 1.7
targetCompatibility = 1.7


// In this section you declare where to find the dependencies of your project
repositories {
   //Configure maven Repositories here
}
```

```
dependencies {
   //Configure All dependencies Here

}

//Wrapper task for Gradle Version
task wrapper(type: Wrapper){
       gradleVersion = '2.6'
}
```

11. Read the problem statement and then implement your code in the highlighted places.
12. Right click build.gradle file select gradle→refresh gradle project.
13. Now we could see all dependencies have been included under **Project and External Dependencies.**

Conclusion:

From the above example, we learnt to include java plug-in, repositories and dependencies configurations in build.gradle file.

Note:

In case in the current workspace if you have ProductApp_Gradle already, please remove the project. Then import the project in the workspace. If the same name of the project exists in the workspace, eclipse will not allow you to import the project.

Lab 2:

Write one new task under Jar to specify your main class in manifest file, so that once the jar has been invoked, it will start execute your main method.

Steps:

- Use your previous project downloaded from GitHub URL (https://github.com/Training2017/MCD)
- In build.gradle file add the below task

```
jar{

        manifest.attributes “Main-Class” : “<<yourMainClass name with Package>>”

}
```

- Go to command prompt
- Build your project using **gradle build** command.
- Once the projects successfully build, then invoke below command in your command prompt.

```
java –jar build/libs/GradleApp.jar <<commandline args if any>>
```

- It will start executing the application.

Conclusion:

From the above example, we learnt jar packaging with main method configuration.

Note:

In case in the current workspace if you have ProductApp_Gradle already, please remove the project. Then import the project in the workspace. If the same name of the project exists in the workspace, eclipse will not allow you to import the project.

Lab 3:

Write the below task in your project which has been downloaded from GitHub URL. You can continue with Lab2. The tasks are below:

- Create one task called greetings (it should contain proper group and description), add simple message for initial configuration phase.
- In the greetings task mention doFirst, doLast method with appropriate messages.
- In **run** task mention main,classpath and command line args.
- In **runJar** task add executable and args with proper inputs.
  ***Note**: In **run and runJar** tasks you should mention the main method to start the execution.*

Steps:

1. Open GitHub URL (https://github.com/Training2017/MCD)
2. Click **ProductApp_Gradle.zip** **folder,** next page click download link. The zip file will be downloaded.
3. Unzip the folder
4. Go to Eclipse IDE, choose import ->Gradle -> Gradle Project into Workspace
5. Under select root directory click browse button.
6. Go to the unzip folder location and choose the folder in the name of ProductApp_Gradle(root directory of your project). Click next.
7. Select local installation directory, click browse button to select Gradle HOME directory.
8. Click Next →Finish
9. Now the Product Application project will be imported in your local machine.
10. Once the project imported into Eclipse IDE, locate build.gradle file.
11. Copy the below mention **build.gradle** file, and add your code.

build.gradle

```
// Apply the java plugin to add support for Java
apply plugin: 'java'
group = 'GradleApp'
version = '0.0.1-SNAPSHOT'

description = 'Product Application Using Gradle'


sourceCompatibility = 1.8
targetCompatibility = 1.8

repositories{
      maven { url 'http://repo.maven.apache.org/maven2' }
}
```

```
dependencies{
	compile 'junit:junit:4.12'
	compile 'org.mockito:mockito-all:1.10.19'

}

jar{
	manifest.attributes 'Main-Class' : 'com.capgemini.demo.Main'
}

//Wrapper task for Gradle Version
task wrapper(type: Wrapper){
	gradleVersion = '2.6'
}

task hello{
	//Write your code here

task run(type:JavaExec, dependsOn:classes){
	//Write your code here
}
task runJar(type:Exec, dependsOn:jar){
	//Write your code here
}
```

Conclusion:

From the above example, we learnt task customization in gradle.

Please refer the below link to get more information about Gradle tasks.

https://docs.gradle.org/current/userguide/java_plugin.html#N15056

Lab 4:

Write a task called '**writeMyName'** in build.gradle. In this task create a file called '**myName'** and then write your name in that file by using doLast task. (You can continue with Lab2 project)

Steps:

1. Open GitHub URL (https://github.com/Training2017/MCD)
2. Click **ProductApp_Gradle.zip folder,** next page click download link. The zip file will be downloaded.
3. Unzip the folder
4. Go to Eclipse IDE, choose import ->Gradle -> Gradle Project into Workspace
5. Under select root directory click browse button.
6. Go to the unzip folder location and choose the folder in the name of ProductApp_Gradle(root directory of your project). Click next.
7. Select local installation directory, click browse button to select Gradle HOME directory.
8. Click Next →Finish
9. Now the Product Application project will be imported in your local machine.
10. Once the project imported into Eclipse IDE, locate build.gradle file.
11. Copy the below mention **build.gradle** file, and add your code.

build.gradle

```
task writeMyName{
      //Keep your code here
}
```

Conclusion:

From the above example, we learnt creating a file using gradle with one message. Similarly we can customize any task in gradle.