**What is a gradle?**

-> open source

-> build automation tool

-> focused on flexibility and performance

-> Gradle build scripts are written in groovy or kotlin DSL(domain specific language).

-> Easier to maintain when compared to XMLs

-> It has become official build tool for android studio.

-> Andriod SDK tools mainly provide

-> gradle android plugin

-> Android studio as official tools

-> It's initial release was in the year 2007.

**Why we need gradle?**

-> Highly customizable (we can customize based on the technologies, ex: if we are working on java

and need gradle as build tool, then we have to use plugin java. And if we are using it in Android

studio we need to use andriod plugin.) so this is the reason where the gradle can be applied to many of the programming languages hence it is highly customizable.

-> Fast(it has high performance, because, it uses the outputs from the previous executions, and processes only the inputs that are changed, and executes tasks parallel).

-> powerful(it is so powerful as its been used by/ supports many popular languages and technologies).

**Projects and tasks in gradle:**

-> There are mainly two basic concepts on gradle

-> projects

-> tasks

-> each of the gradle build is made up of one or more projects.

**Gradle projects:**

-> project could be a jar or a web application

-> it could also be a ZIP file that contain the jars produced by other projects.

-> the gradle project could be deploying the application to production environment.

-> each and every project has one or more tasks.

-> each task are atomic(they have specific purpose)

**Gradle tasks:**

-> compiling class could be a task of a project

-> creating a jar could be a task

-> generating javadoc

-> publishing the archives into the repositories.

**Installing gradle and setting up in our system:**

**step-1:**

-> go to the link “https://gradle.org/install/ “

-> scroll down to see "Installing manually" section

-> Under it choose "Binary-only".

-> then installation starts automatically

**step-2:**

-> once installed extract the zip file.

-> then goto control panel

-> goto system properties and select advance

-> in this select environment variable

-> click on new under user variables section, then give variable name as "GRADLE\_HOME"

-> and variable value as path where the gradle installed[note: you need to place the path expect opening the bin folder.

-> Then select "path" and click "edit" and click on "new"

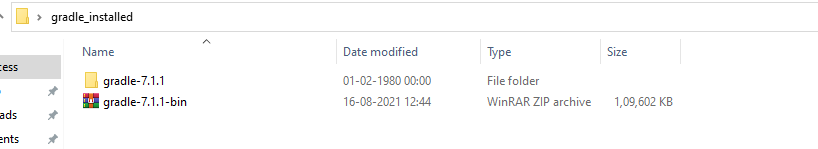
-> add the path including the bin folder here.

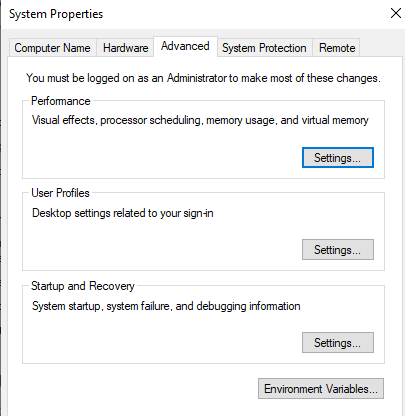
**Screen shots of these steps:**

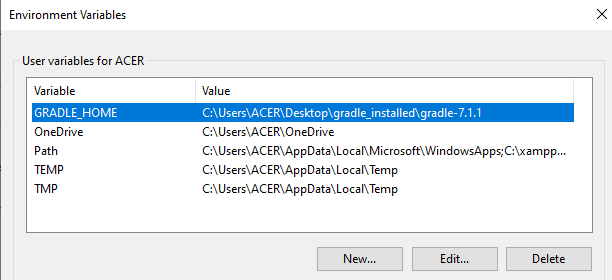
**Step-1:**

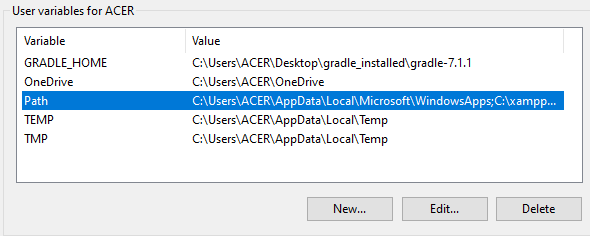


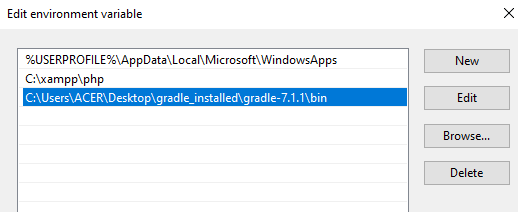
**Step-2:**











-> now everything is done.

**Adding gradle plugin in eclipse:**

-> After setting everything on the system, you have to go to "c:" and inside which select "users" folder

within that select check that ".gradle" folder is generated automatically.

-> This is the "local repository" of gradle.[what all dependencies jar files that will be downloading will be stored

some where inside .gradle.

-> Then if we want gradle to be used in eclipse we have to add the gradle plugin inside the

eclipse.

-> i.e the steps in adding gradle plugin inside the eclipse are:

step-1:

-> open eclipse

step-2:

-> goto "help" at the top

step-3:

-> select "eclipse marketplace" [NOTE: you need to have internet while this process needs to be carried out]

-> Then inside this goto find and type "gradle"

setp-4:

-> you will be displayed with one pludin named "Buildship gradle integration"

-> all you need to do is install it by pressing onstall button gien below.

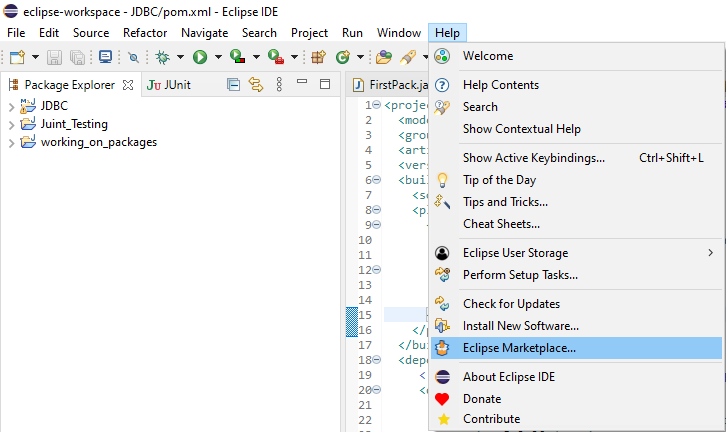
-> so now gradle plugin is incorporated inside the eclipse.

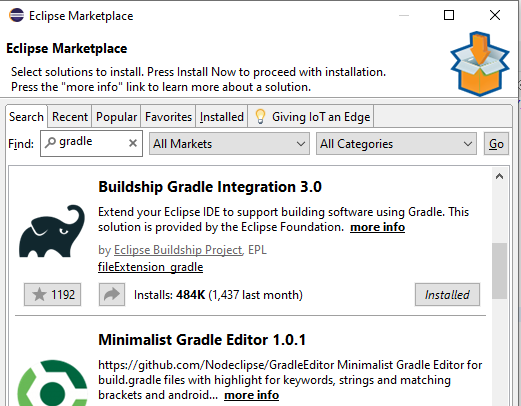
-> to check whether the gradle plugin is installed or not, goto "files" then select "new"

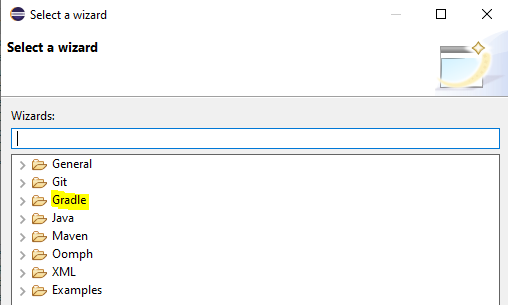
-> then select "others" you can see a wizard

-> that wizard should have the "Gradle" folder.

**Screen shots of the steps:**

****





**Creating a gradle poject:**

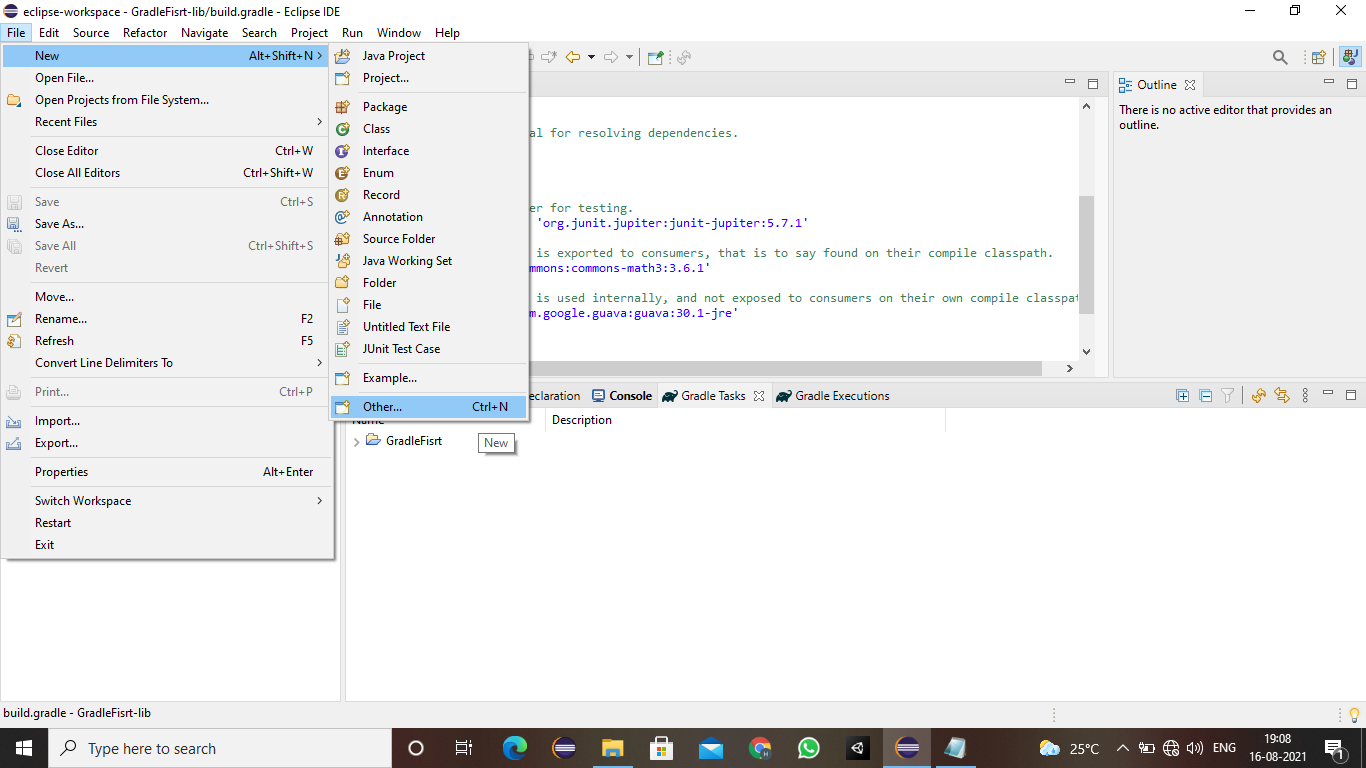
steps:

step-1:

-> goto "files"

-> select "new"

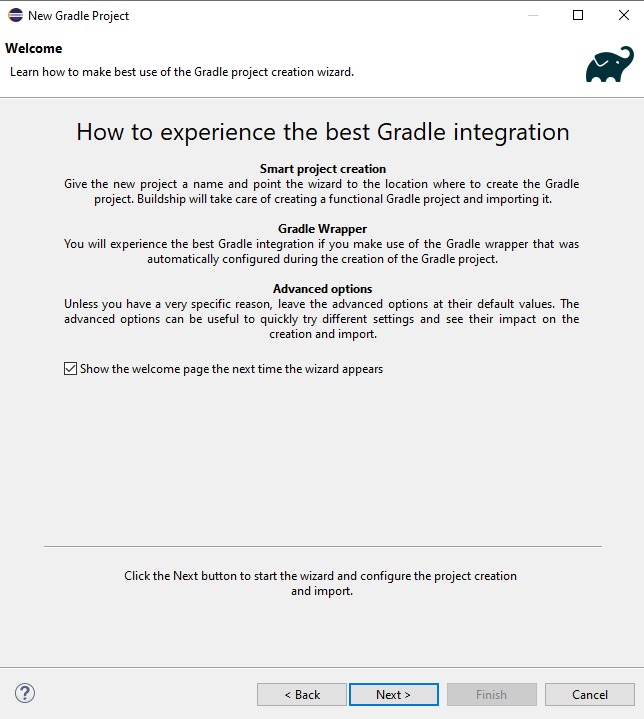
-> goto "others"



step-2:

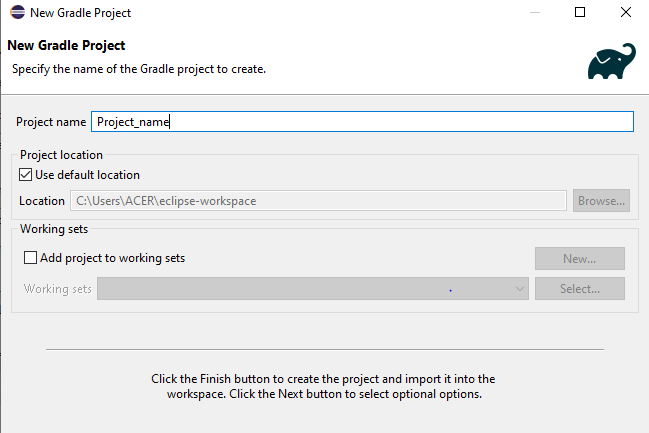
-> choose the "gradle folder" and click on "gradle project"

-> then you will see a wizard showing some information, click on next

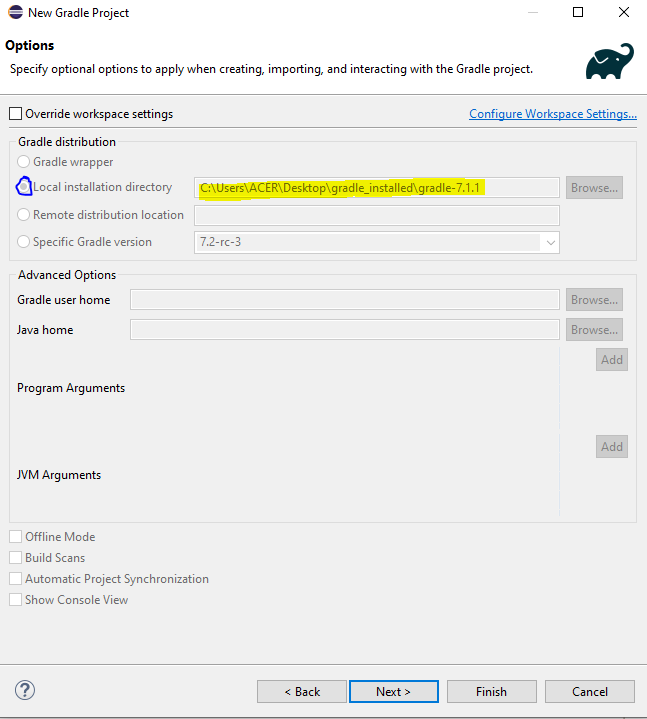


step-3:

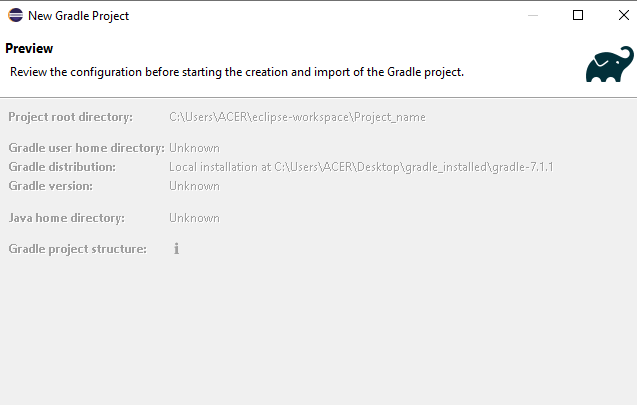
-> give the project name and click on next



-> you will have to enter the option as shown in the image below



-> after this you will be shown with the preview before the project is built



-> after this you will see a project that is created and visible on the eclipse.

**Gradle tasks:**

Default gradle task-> the tasks that gradle is already providing.

we can also write our own customesed tasks.

so we have two types of tasks:

1. default tasks

2. custom task

Default gradle tasks in gradle.

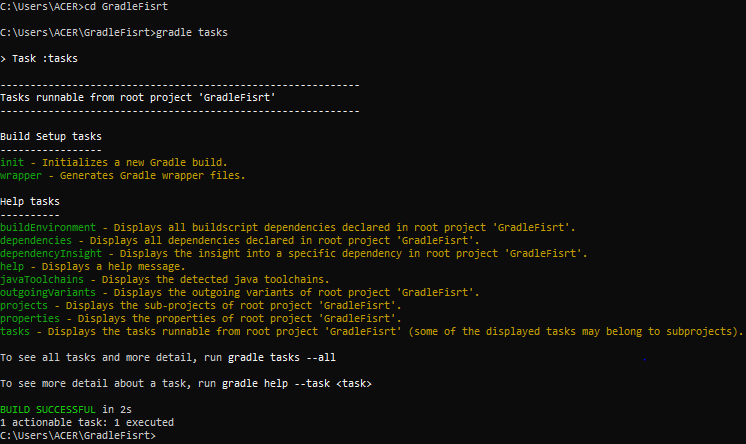
-> open cmd

-> type the command gradle tasks

-> when enter is pressed the gradle will list down all the tasks available in gradle.

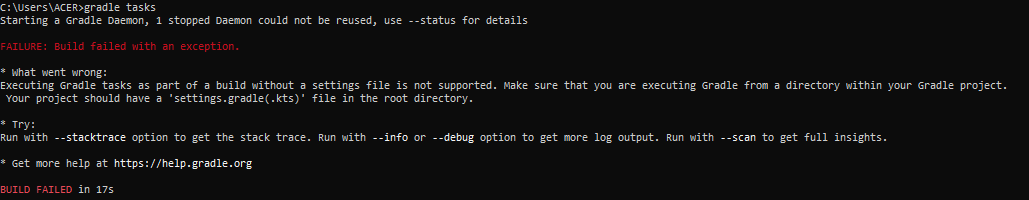
-> [NOTE: before typing the command "gradle tasks" you basically have to specify the root project

i.e it expects the settings.gradle file to be present in order to display the default or built in tasks]



This is how it basically looks when you type the command “gradle tasks”.

* I also made a point that we get an build failed exception i.e shown below:

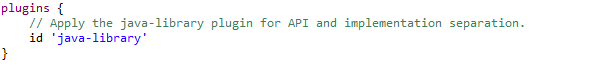


* To know the complete tasks in the gradle we need to use the command “ gradle tasks –all”.
* This will display all the tasks that are default in the gradle and i.e is shown below.

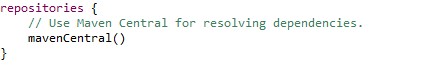


**Custom gradle tasks:**

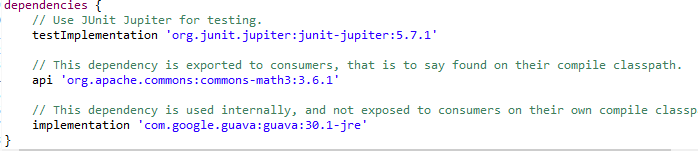
* We can specify our own tasks to be executed apart from the default tasks.
* To do that we have to go to the eclipse and create a gradle project (use the guide lines to create the gradle project given above in the same doc.).
* Once the project is built go to the build.gradle file.
* You can see there are different sections like plugin, repository and dependencies.
* Where plugin are used to specify the plugin for the language that we want to use the gradle i.e say if we want to use gradle for java then we need to add the java-libraries plugin inside the plugin section. i.e



* Then repository section hold stuff, from where the dependencies are downloaded i.e



* Then dependencies section, we use this section to add certain dependencies that our project may require



**Now we will see how to write our own task:**

* So if you are writing your task, note you need to write in the build.gradle file only.
* The syntax for to write the custom task is:

task task\_name{

group “group\_name” //specifies under this group name the task need to be/shown.

Description “write the description if the task”

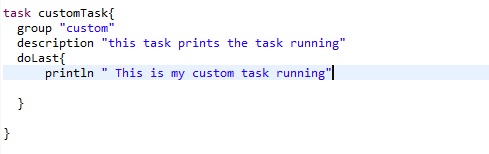
doLast{

//write the logic that is to be performed i.e the task you need to perform as customized

}

}

* Example : I have written a task that just prints some statement



* So after wiring the task we need to run the task.
* Then command to run the task is “gradle task\_name”.



Now let’s have a discussion on the topic **COPY TASK**:

* Here we will write a task that copies the file from one folder to another.
* So how it is done?
* Here is the syntax:

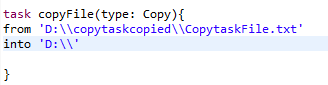
task task\_name(type : copy){

from “specify the location of the file”

to “specify where the file needs to be copied”

}

* Example:



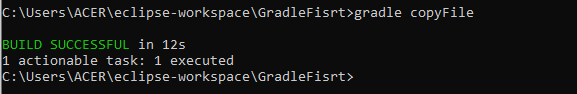
* So when this task is made run [NOTE: command to run task: “gradle copyFile” in case of above example]
* So here he most important parameter are:

type: Copy // this specifies that we are performing copy operation

from // this specifies from where the file needs to be copied

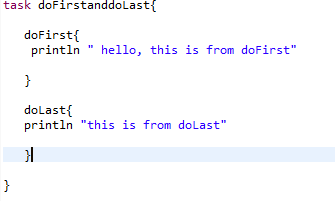
into // this specifies to where the file needs to be copied

* The output:

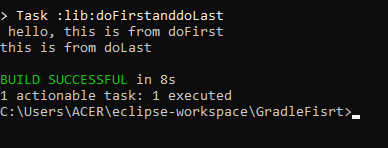


**doFirst and doLast:**

* Now we will discuss some simple topics named doFirst and doLast.
* So by the name itself it say, doFirst indicates you need to do this stuff first
* And doLast indicates you need to perform that particular task at the last
* Let us see this with an example:

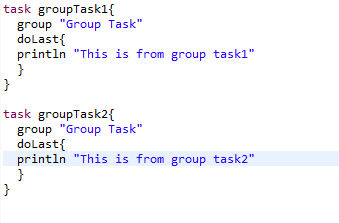


**Output:**

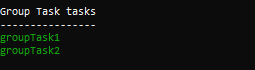


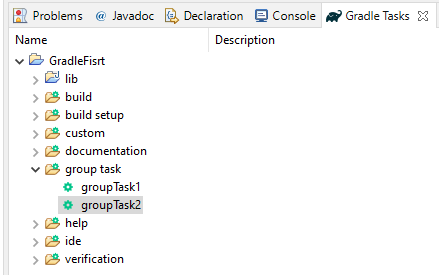
**Grouping tasks:**

* So here we will see how we can group different tasks under same group.
* This is basically done by giving same group name.
* Example for this:



**Output:**

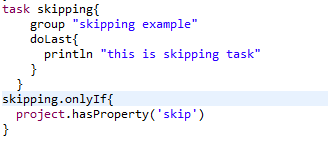




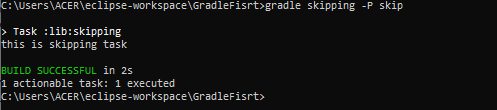
* Make sure that in projects the build automatically is ticked.
* Then once the above tasks are written you can see he groups in the gradle tasks section.

**Skipping Task:**

* Here we are going to talk about skipping some tasks.
* Where say we want to skip some task, then this option is also available in gradle.
* Let us see an example:



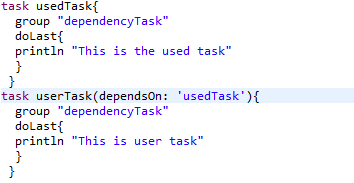
* In the above example we can see how we can skip some tasks.
* Where we have a task named skipping which is placed under the group skipping example.
* And the task will be skipped if the program argument is not specified
* In command prompt we execute this type of tasks as follows:



* Where –P stands for property and it should be followed by the property name.
* Here the property name given is skip.

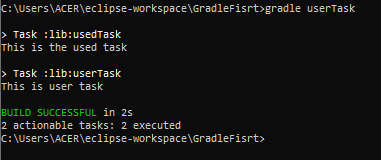
**Task dependency:**

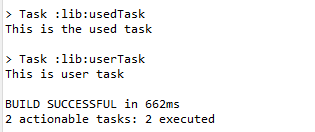
* Here we will be discussing about how dependencies are been handled.
* Say for example:



* In the above example we have two tasks, one is the userTask and other is usedTask, where the userTask is dependent on the usedTask.
* This is specified in the above example using [dependsOn: “task\_name”]
* Which basically mean that before executing the usedTask the userTask cannot be.
* So if you try to execute the userTask it will first execute the usedTask and then wil execute the userTask.

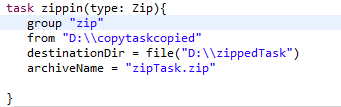
**Output:**





**Zipping task:**

* In this task what we need to do is, we have to choose a folder that is to be converted as zip file.
* And then we need to specify the destination directory to where the zipped file has to stored.
* And we also have to define the name of the zipped file.
* So let us see this with an example:



**Output:**

