Ojdbc.jar

Junit jar

JAR or Java Archive file is an executable file with collection of libraries associated metadata and resources.

It is actually a ZIP file having extension .JAR file.

As it also contains compiled main class (.class format) , other compiled classes , so it is used to compress

and use the code without execution even without using IDE.

It is used to distribute a software program over the network.

#### A Build Lifecycle is Made Up of Phases

Each of these build lifecycles is defined by a different list of build phases, wherein a build phase represents a stage in the lifecycle.

For example, the default lifecycle comprises of the following phases (for a complete list of the lifecycle phases, refer to the [Lifecycle Reference](http://maven.apache.org./guides/introduction/introduction-to-the-lifecycle.html#Lifecycle_Reference)):

* validate - validate the project is correct and all necessary information is available
* compile - compile the source code of the project
* test - test the compiled source code using a suitable unit testing framework. These tests should not require the code be packaged or deployed
* package - take the compiled code and package it in its distributable format, such as a JAR.
* verify - run any checks on results of integration tests to ensure quality criteria are met
* install - install the package into the local repository, for use as a dependency in other projects locally
* deploy - done in the build environment, copies the final package to the remote repository for sharing with other developers and projects.

These lifecycle phases (plus the other lifecycle phases not shown here) are executed sequentially to complete the defaultlifecycle. Given the lifecycle phases above, this means that when the default lifecycle is used, Maven will first validate the project, then will try to compile the sources, run those against the tests, package the binaries (e.g. jar), run integration tests against that package, verify the integration tests, install the verified package to the local repository, then deploy the installed package to a remote repository.

[*[top]*](http://maven.apache.org./guides/introduction/introduction-to-the-lifecycle.html)*.*

#### Usual Command Line Calls

In a development environment, use the following call to build and install artifacts into the local repository.

mvn install

This command executes each default life cycle phase in order (validate, compile, package, etc.), before executing install. You only need to call the last build phase to be executed, in this case, install:

In a build environment, use the following call to cleanly build and deploy artifacts into the shared repository.

mvn clean deploy

The same command can be used in a multi-module scenario (i.e. a project with one or more subprojects). Maven traverses into every subproject and executes clean, then executes deploy (including all of the prior build phase steps).

You have application,want to create a jar file and give it client

Steps:

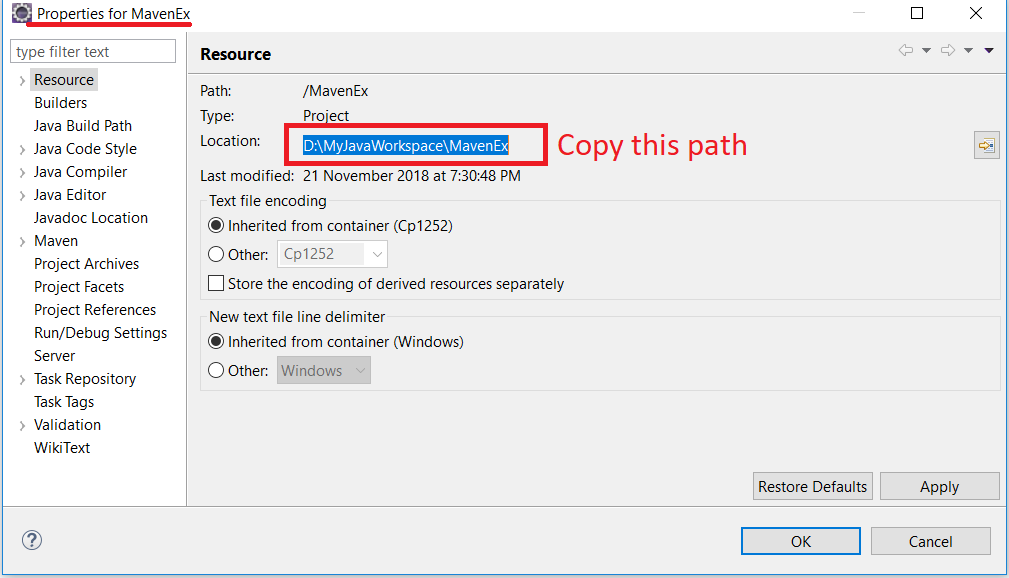
1> compile all source .java files to .class files

2> Compile all AppTest.java files to .class files

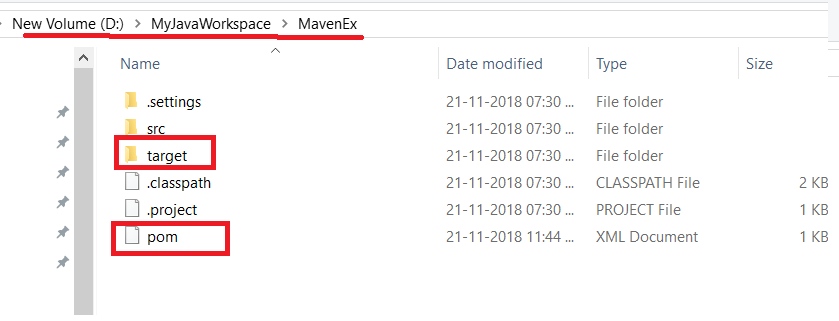
3> Run Tests - unless and until u get zero errors work on your source codes

4> create a jar

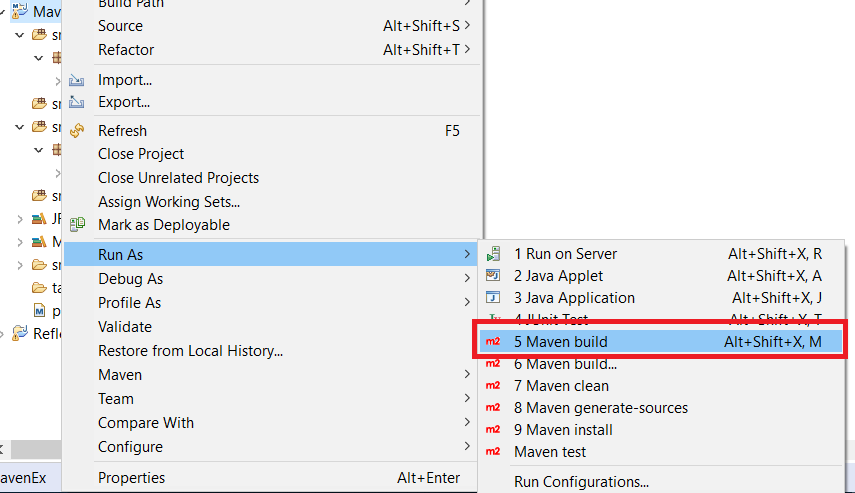
Open your project : right click -> properties

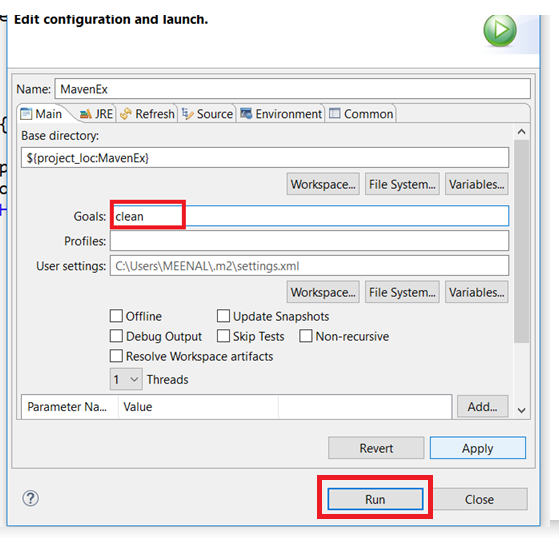


Open it in explorer , see you have target folder and pom.xml file



Next step is clean :

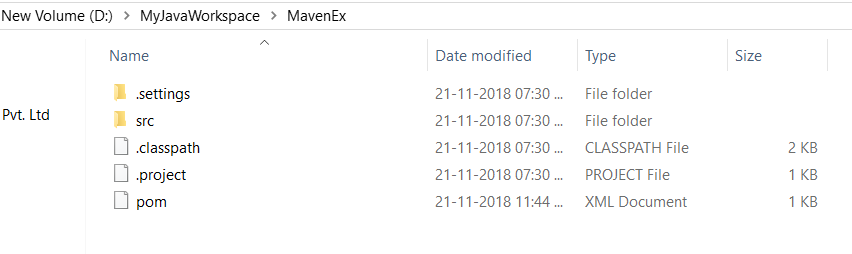




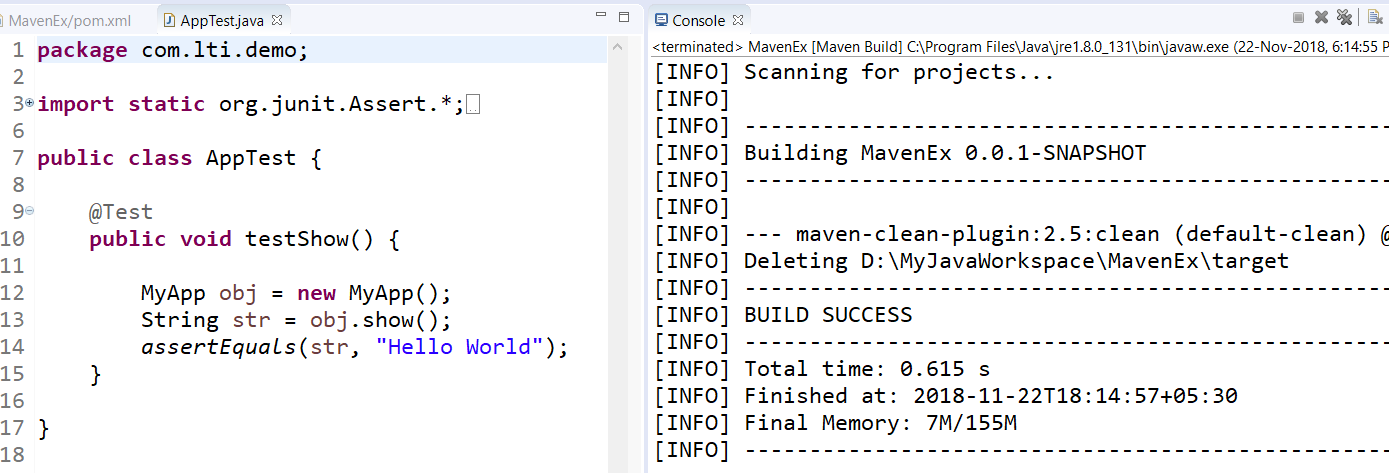
Observer Target Folder is removed:

All the compiled files and jar files after compilation are saved here,

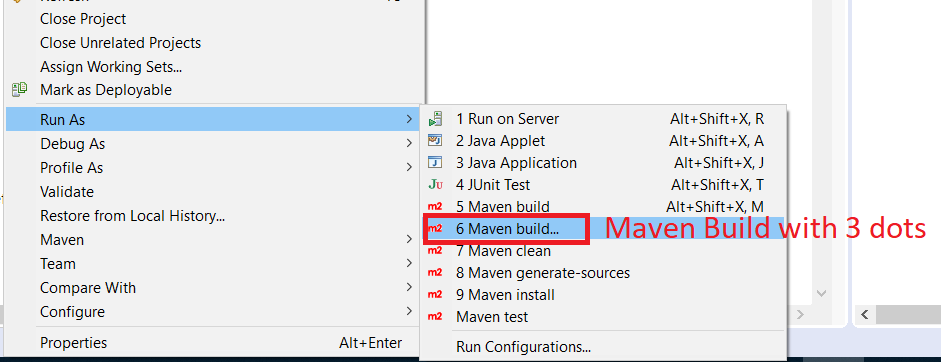
Clean – cleans everything .

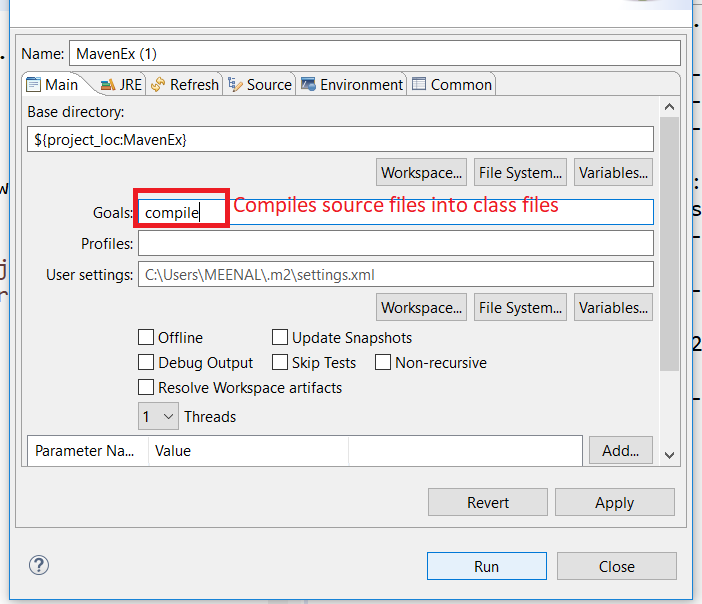


See the console :

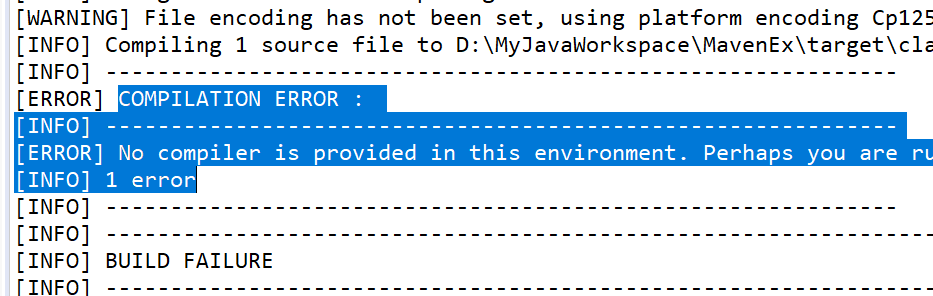


Ready to compile our app :



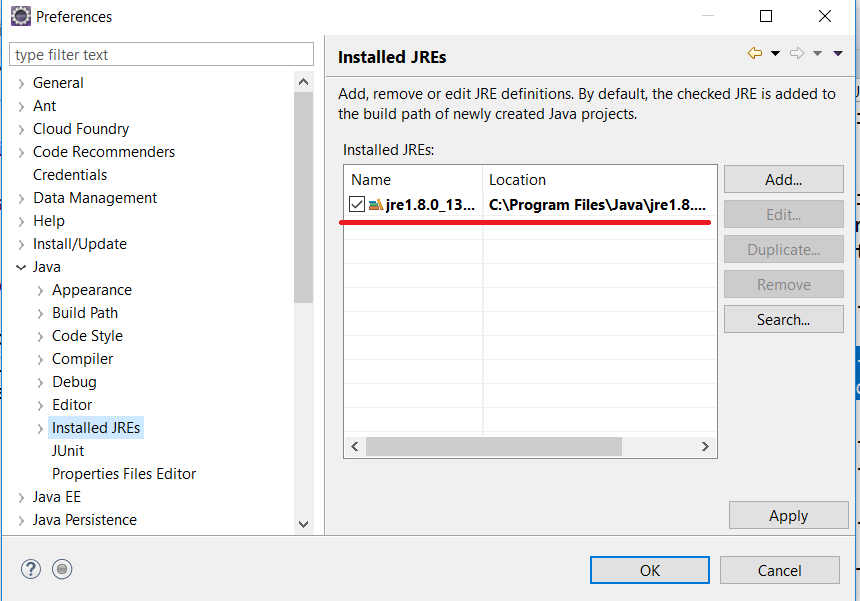


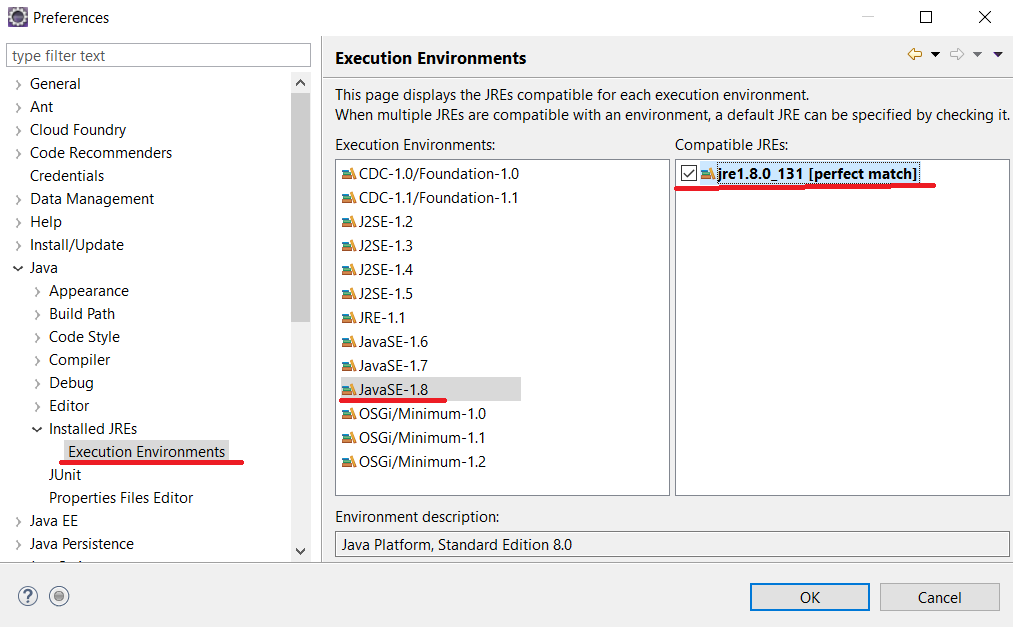
If You get this error :

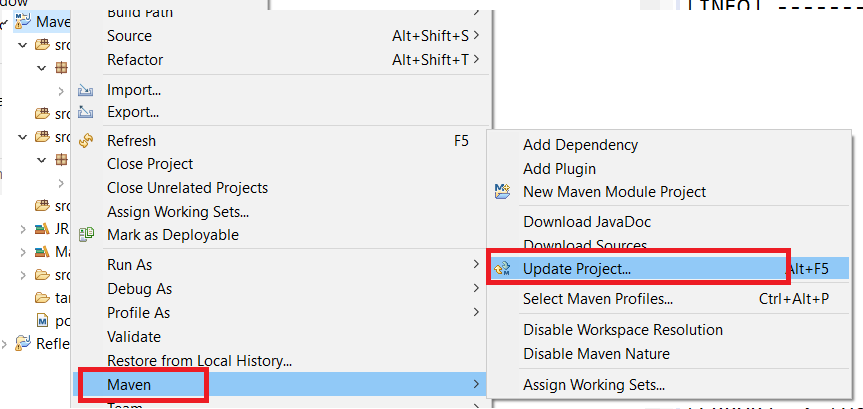


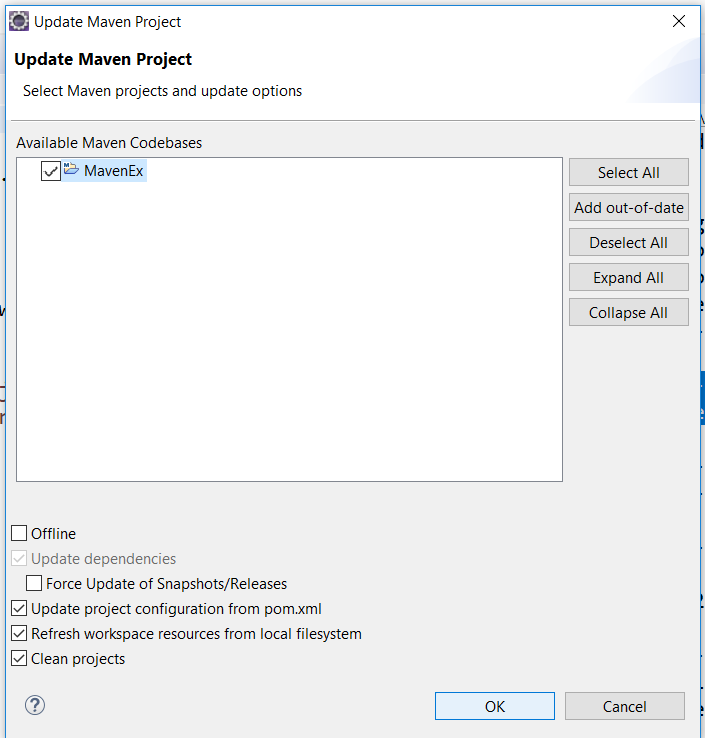
Windows -> preferences -> Java -> installed JRE-

You should have jdk specified here : I have JRE ( NOT JDK )

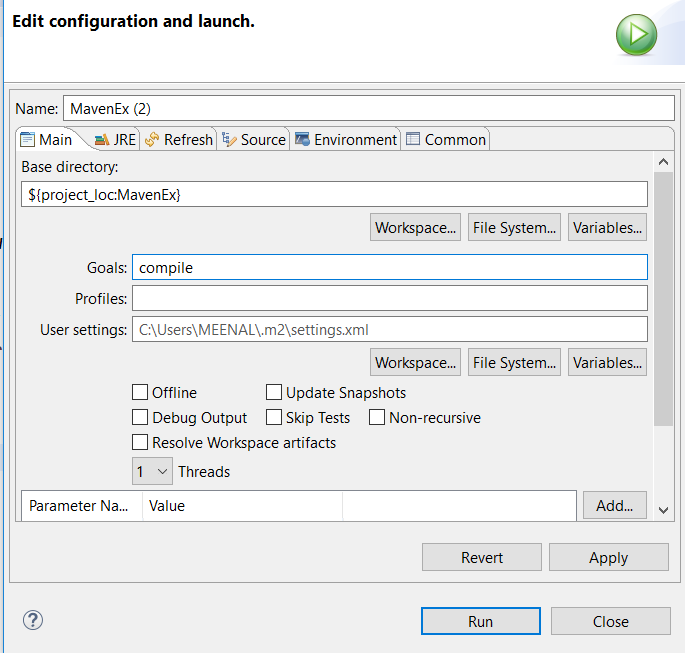


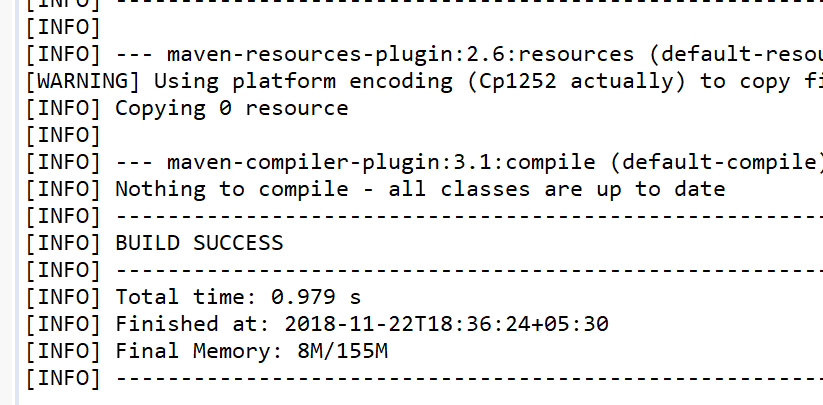


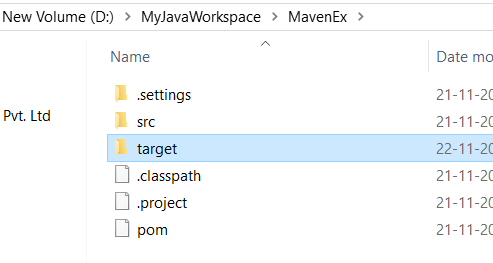


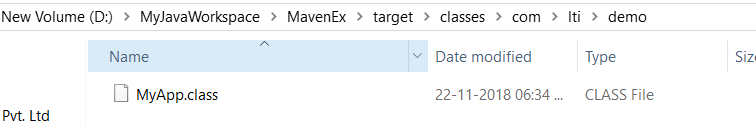


Run As -> Maven Build with 3 dots -> complie ->repeat steps

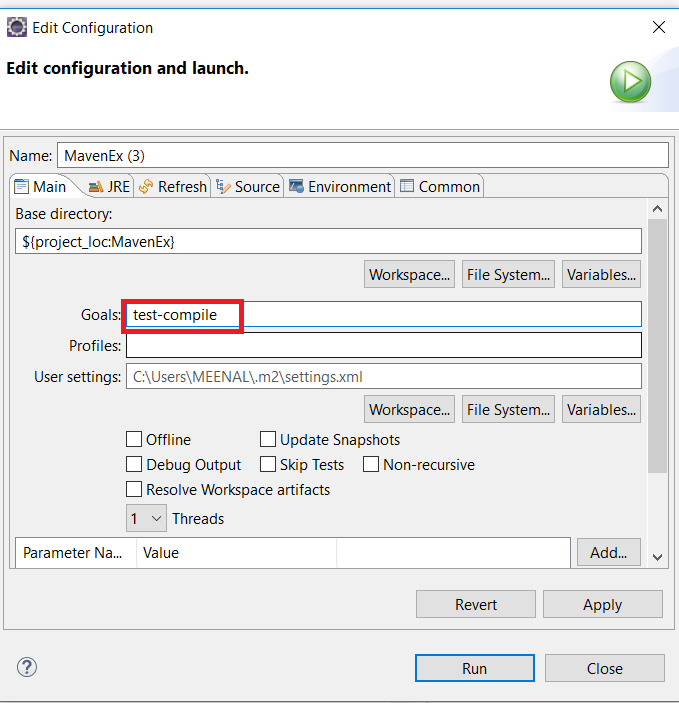


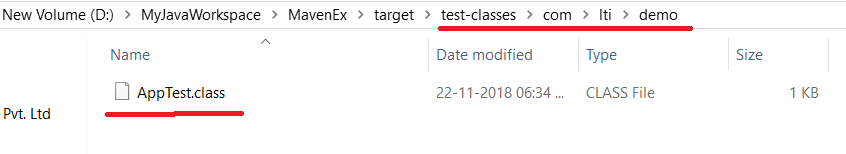




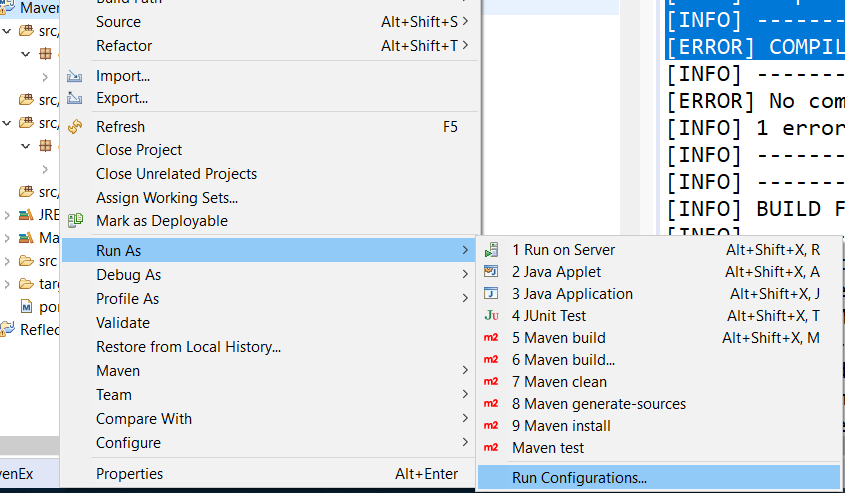


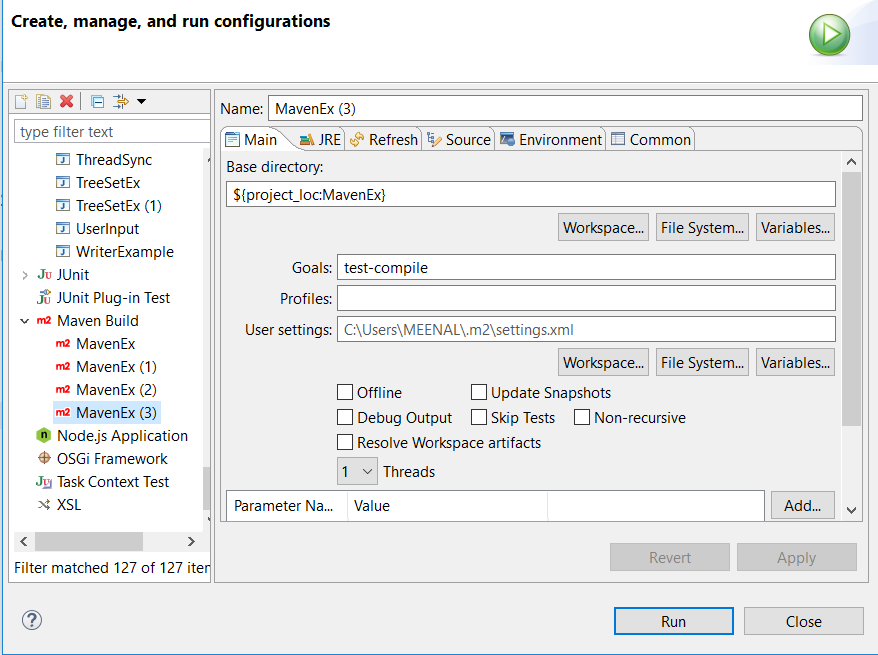
Compile your Test Files :

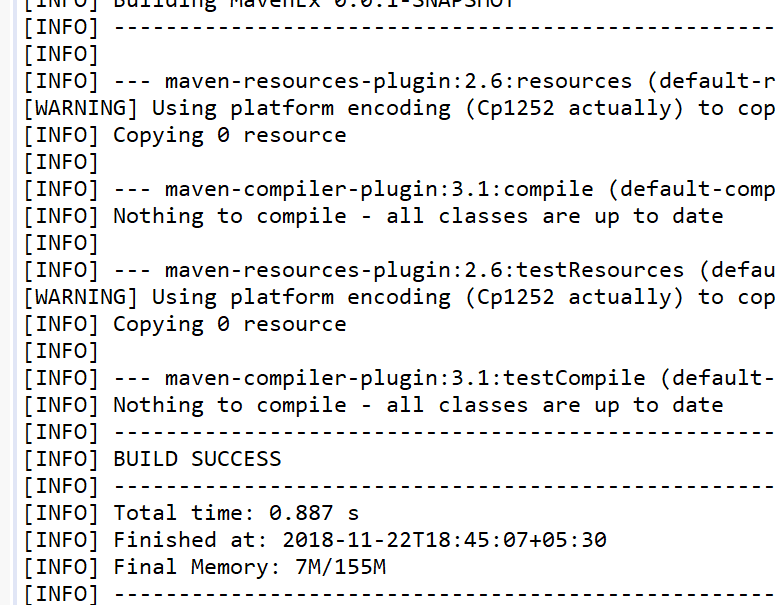


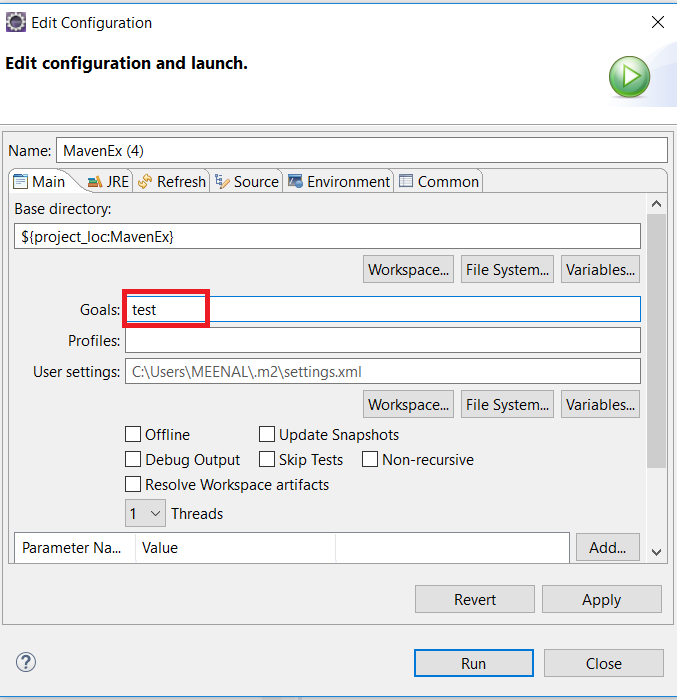


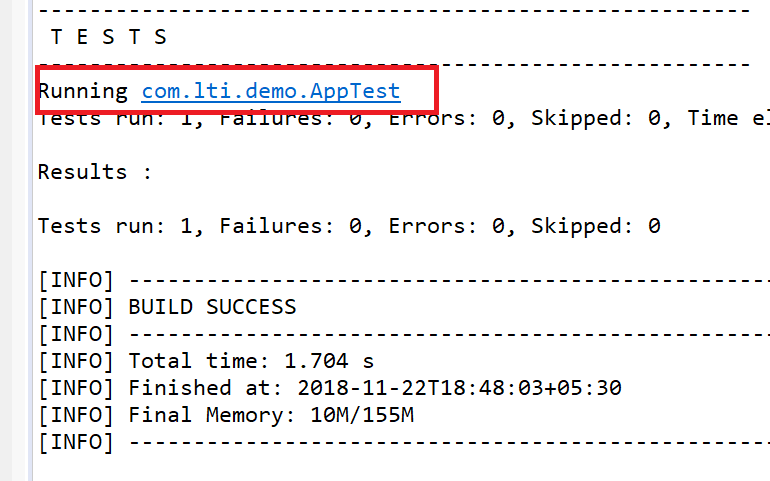
If you Get Error:

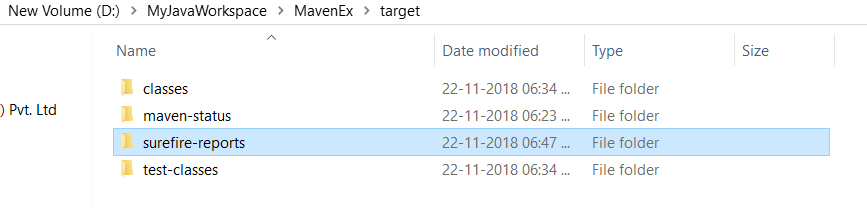


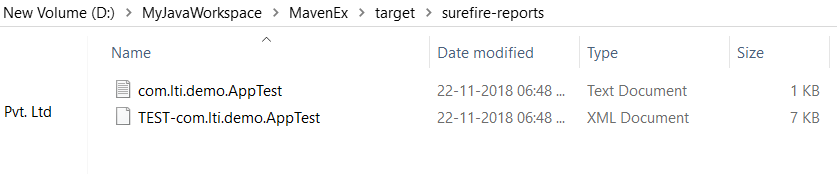


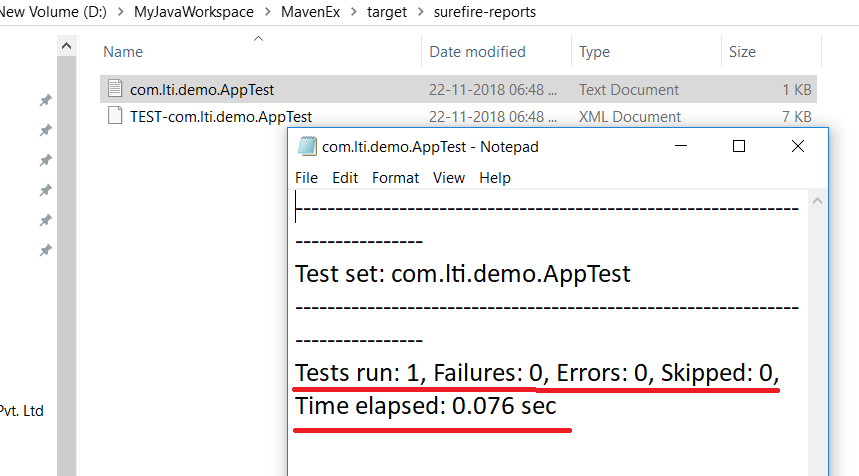










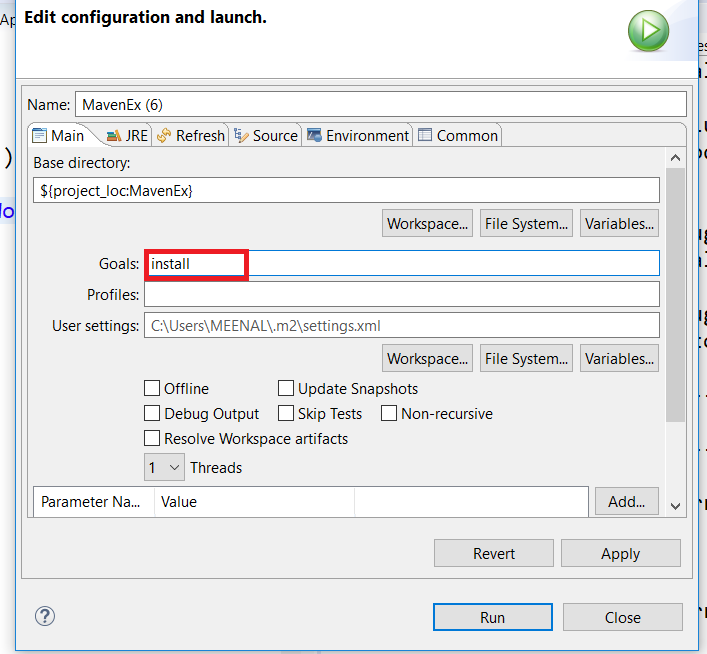


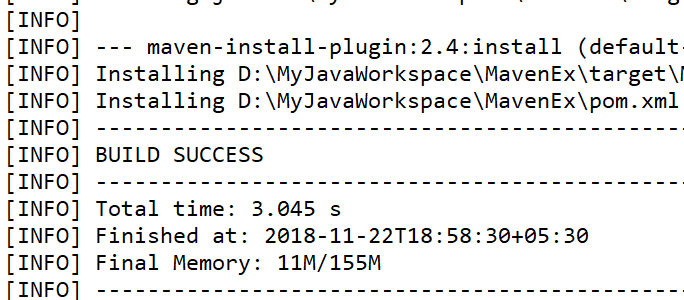
There are no errors in your source ,

All 3 steps are done

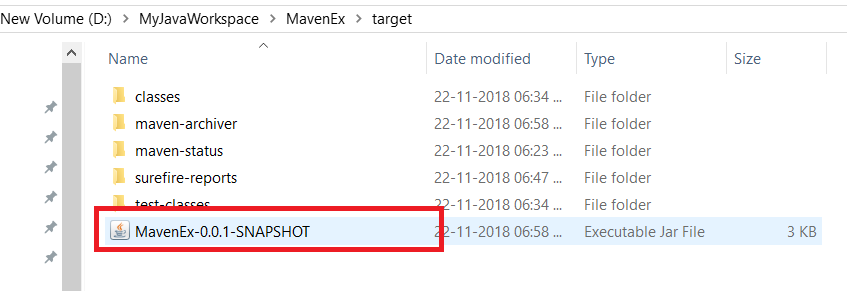
Source files compiled , test files compiled , tests are run

Create jar file





See Jar file is created here :



Name is: Your Projectname

1. JAR or Java Archive file is an executable file with collection of libraries associated metadata and resources.
2. It is actually a ZIP file having extension .JAR file.
3. As it also contains compiled main class (.class format) so it is used to compress
4. and use the code without execution even without using IDE.
5. It is used to distribute a software program over the network.