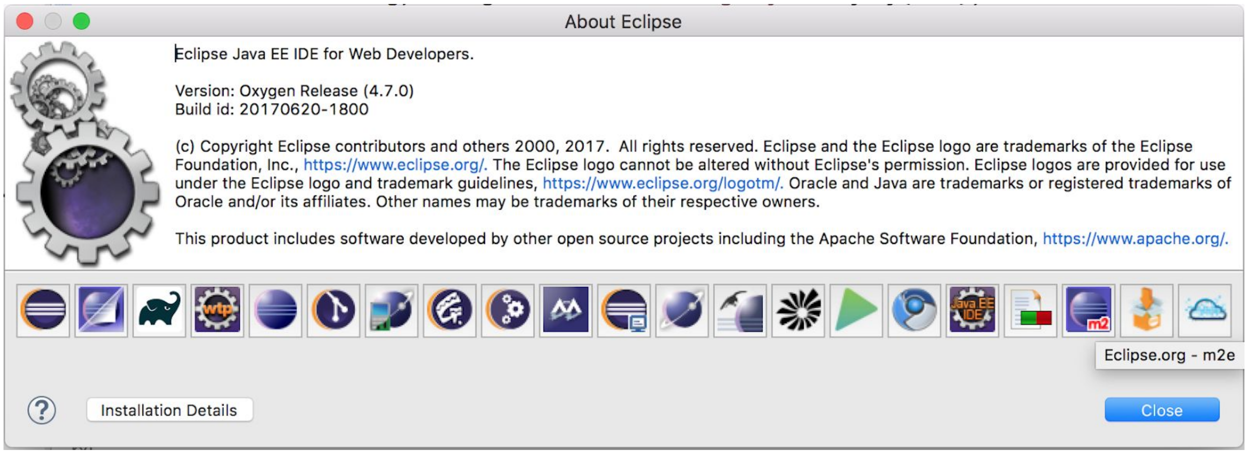
Creating a MAVEN Project in Eclipse Oxygen

- Its recommended to use Eclipse Oxygen IDE because Maven plugin is already present in this version of eclipse.



Eclipse Java EE IDE for Web Developers.

Version: Oxygen Release (4.7.0)
Build id: 20170620-1800

(c) Copyright Eclipse contributors and others 2000, 2017. All rights reserved. Eclipse and the Eclipse logo are trademarks of the Eclipse Foundation, Inc., https://www.eclipse.org/. The Eclipse logo cannot be altered without Eclipse's permission. Eclipse logos are provided for use under the Eclipse logo and trademark guidelines, https://www.eclipse.org/logotm/. Oracle and Java are trademarks or registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

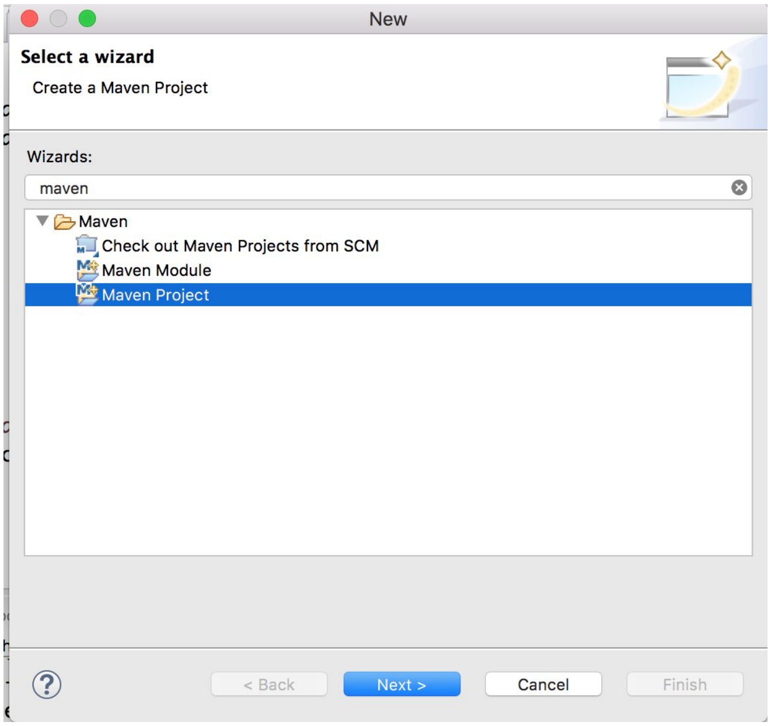
This product includes software developed by other open source projects including the Apache Software Foundation, https://www.apache.org/.

In the above mentioned image you can see in the right corner just above the "close" button there is a text highlight "Eclipse.org - m2e". This indicates that maven is bundled with Eclipse Oxygen.

The link to download Eclipse Oxygen is mentioned below for your reference:

https://www.eclipse.org/oxygen/

- Open Eclipse Oxygen IDE and Goto *File* ⟶ *New* ⟶ *Other* ⟶ *<In the text area type Maven>* ⟶ <Click the drop down arrow to the left of "Maven" icon which appears after search> ⟶ <Select "Maven Project"> ⟶ Click Next



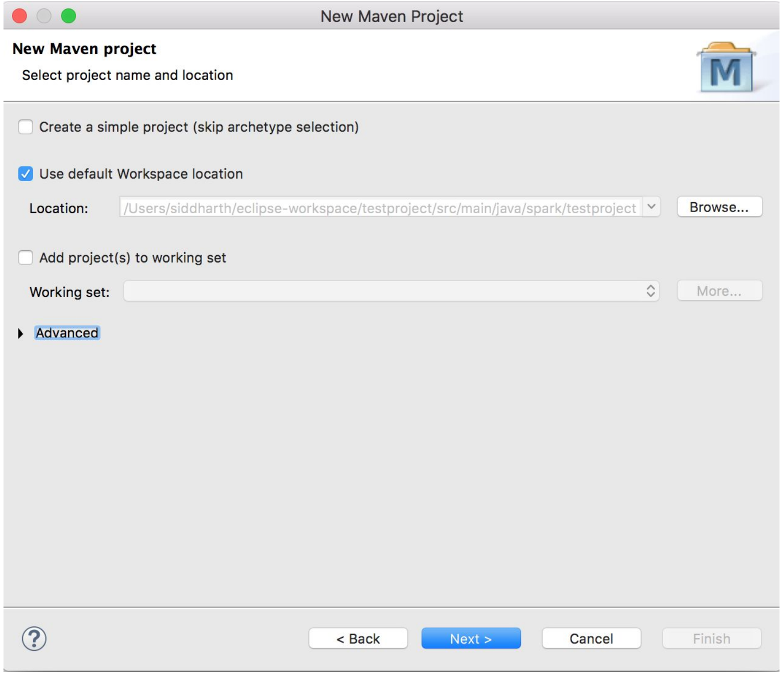
New

Select a wizard

Create a Maven Project

Wizards: maven

- Select the default workspace location and click next.



New Maven Project

New Maven project

Select project name and location

☐ Create a simple project (skip archetype selection)

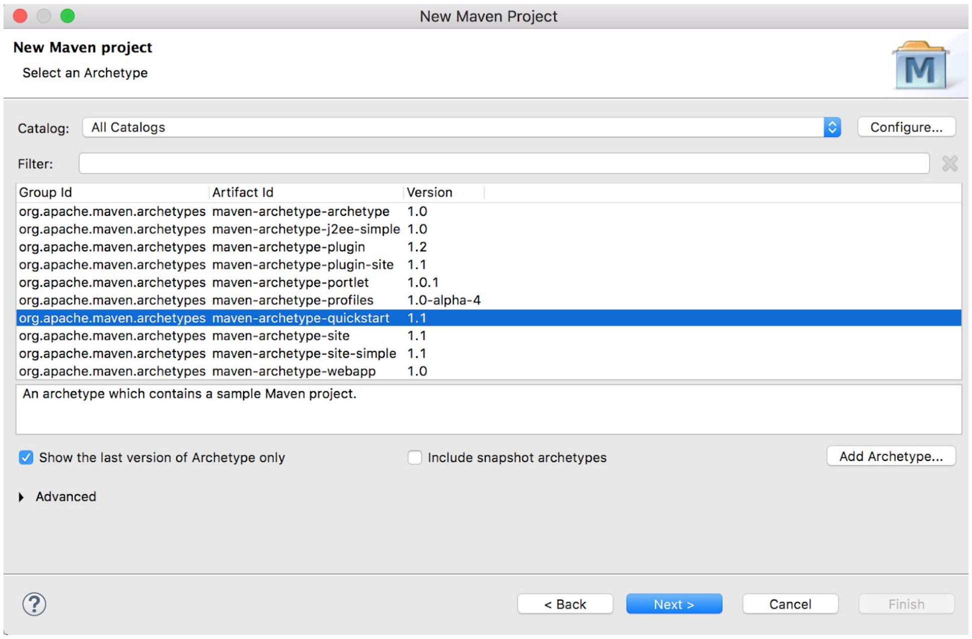
☑ Use default Workspace location

Location: /Users/siddharth/eclipse-workspace/testproject/src/main/java/spark/testproject

☐ Add project(s) to working set

Working set:

- Select the default Maven version type and click next.



New Maven Project

New Maven project

Select an Archetype

Catalog: All Catalogs

Filter:

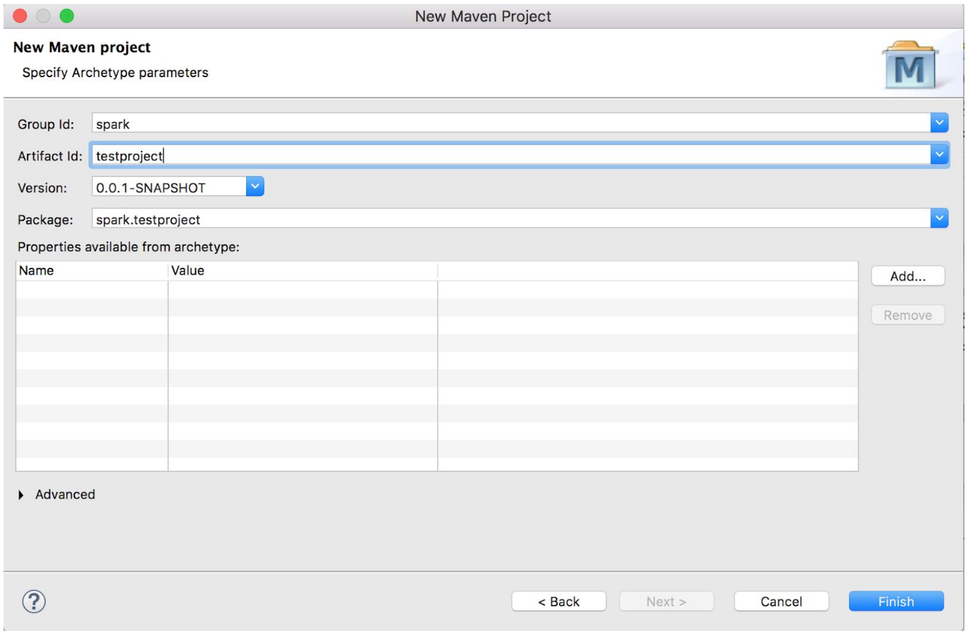
Group Id	Artifact Id	Version
org.apache.maven.archetypes	maven-archetype-archetype	1.0
org.apache.maven.archetypes	maven-archetype-j2ee-simple	1.0
org.apache.maven.archetypes	maven-archetype-plugin	1.2
org.apache.maven.archetypes	maven-archetype-plugin-site	1.1
org.apache.maven.archetypes	maven-archetype-portlet	1.0.1
org.apache.maven.archetypes	maven-archetype-profiles	1.0-alpha-4
org.apache.maven.archetypes	maven-archetype-quickstart	1.1
org.apache.maven.archetypes	maven-archetype-site	1.1
org.apache.maven.archetypes	maven-archetype-site-simple	1.1
org.apache.maven.archetypes	maven-archetype-webapp	1.0

An archetype which contains a sample Maven project.

☑ Show the last version of Archetype only

☐ Include snapshot archetypes

- Enter a GroupID and artifact and press “Finish”. Note that the package name is the combination of GroupId and Artifact ID. This package structure is created in the “src” folder of the project



New Maven Project

New Maven project

Specify Archetype parameters

Group Id: spark

Artifact Id: testproject

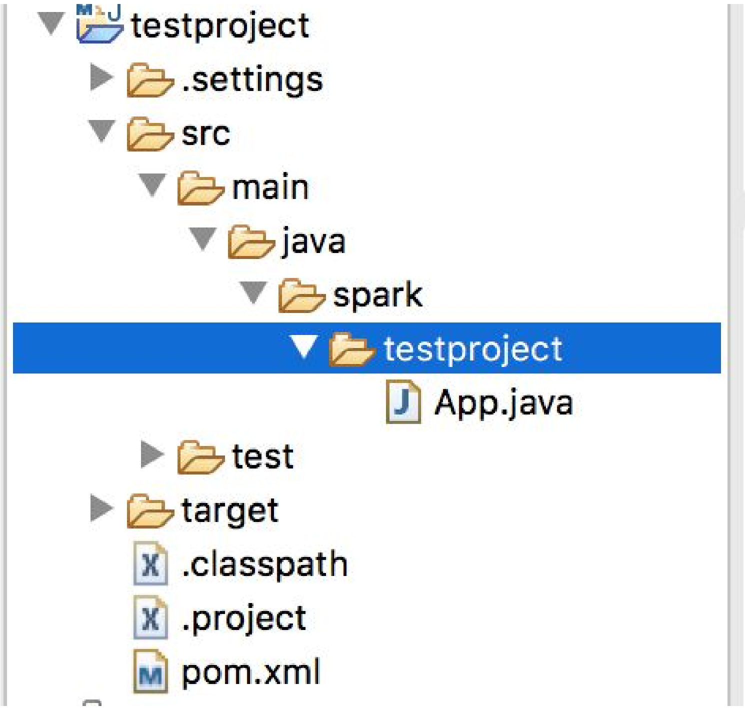
Version: 0.0.1-SNAPSHOT

Package: spark.testproject

Properties available from archetype:

Name	Value

- A maven project is created with following following folder structure:



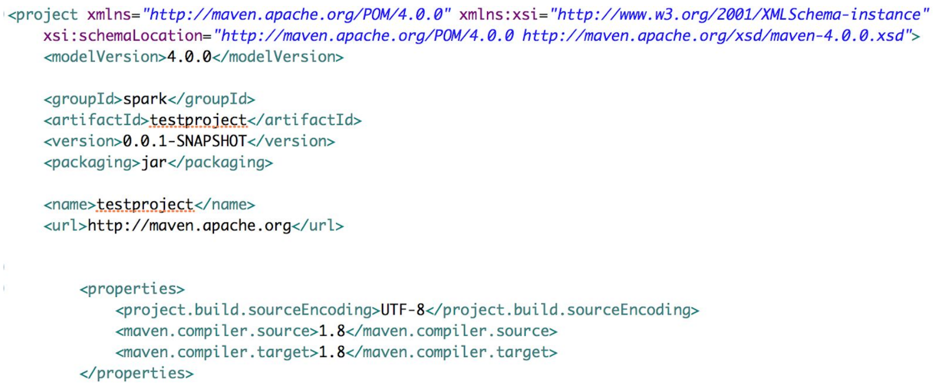
- App.java file is already present in location src.main.<groupid>.<artifactid>. Use this class file for creating the spark jobs or delete it and create your own class file.

Editing the “pom.xml” File for Compiling and Creating the Executable Jar

In the pom.xml file we will need to add the following entries:

○ For instructing maven to compile the project using Java 1.8 add the below mentioned entries in pom.xml. Replace the entire existing information contained in between the properties tags i.e. <properties></properties> with the below mentioned entries

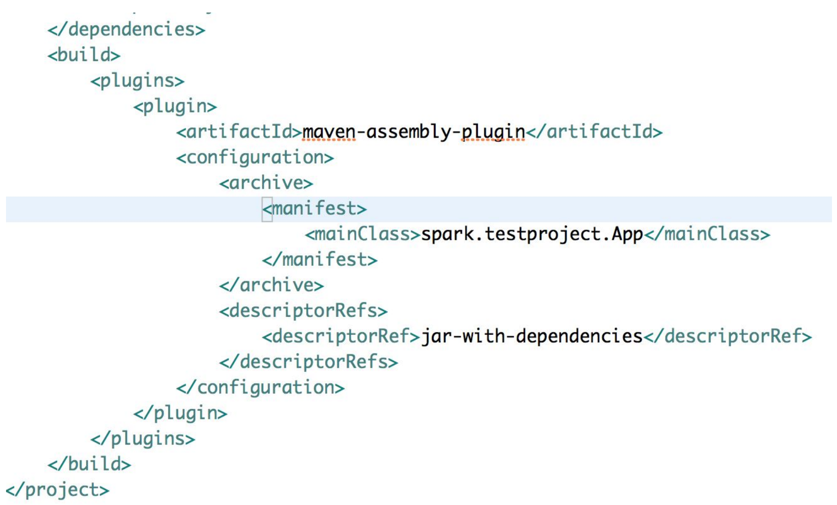
```
<properties>
      <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
      <maven.compiler.source>1.8</maven.compiler.source>
      <maven.compiler.target>1.8</maven.compiler.target>
</properties>
```



```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>spark</groupId>
    <artifactId>testproject</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <packaging>jar</packaging>

    <name>testproject</name>
    <url>http://maven.apache.org</url>


        <properties>
            <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
            <maven.compiler.source>1.8</maven.compiler.source>
            <maven.compiler.target>1.8</maven.compiler.target>
        </properties>
```

- For creating an executable jar with all the dependencies present in it add the following entries into the pom.xml file. Add this entry exactly where the dependencies tags ends

```
<build>
    <plugins>
        <plugin>
        <artifactId>maven-assembly-plugin</artifactId>
            <configuration>
            <archive>
                <manifest>
                    <mainClass>spark.testproject.App</mainClass>
                </manifest>
            </archive>
            <descriptorRefs>
                <descriptorRef>jar-with-dependencies</descriptorRef>
            </descriptorRefs>
            </configuration>
            </plugin>
        </plugins>
    </build>
```

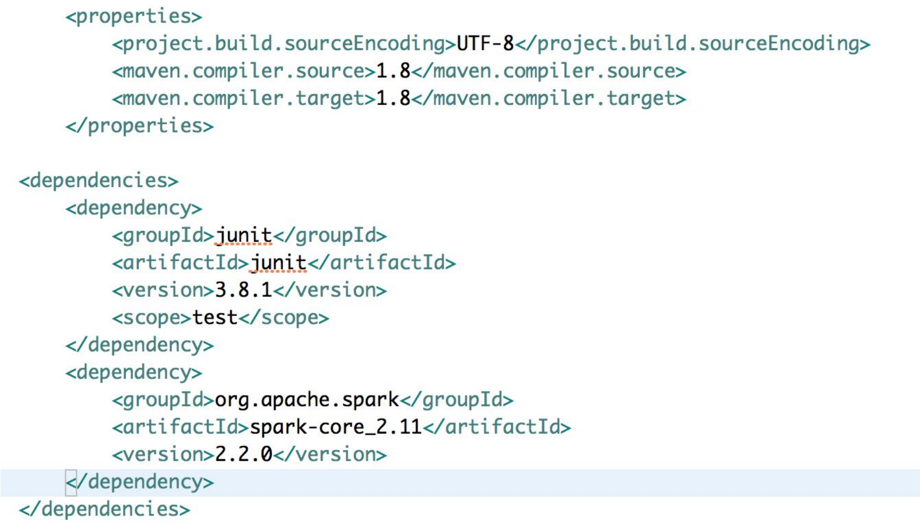


```
    </dependencies>
    <build>
        <plugins>
            <plugin>
                <artifactId>maven-assembly-plugin</artifactId>
                <configuration>
                    <archive>
                        <manifest>
                            <mainClass>spark.testproject.App</mainClass>
                        </manifest>
                    </archive>
                    <descriptorRefs>
                        <descriptorRef>jar-with-dependencies</descriptorRef>
                    </descriptorRefs>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>
```

The complete main class path is mentioned in between the “<mainClass></mainClass>” tags. Main class is configurable. It is the main class of your project which contains the main method.

- For Spark RDDs include the below mentioned dependencies in the dependencies tag. Please note you dont have to replace the existing content, you just have to add the below mentioned dependency along with existing content:

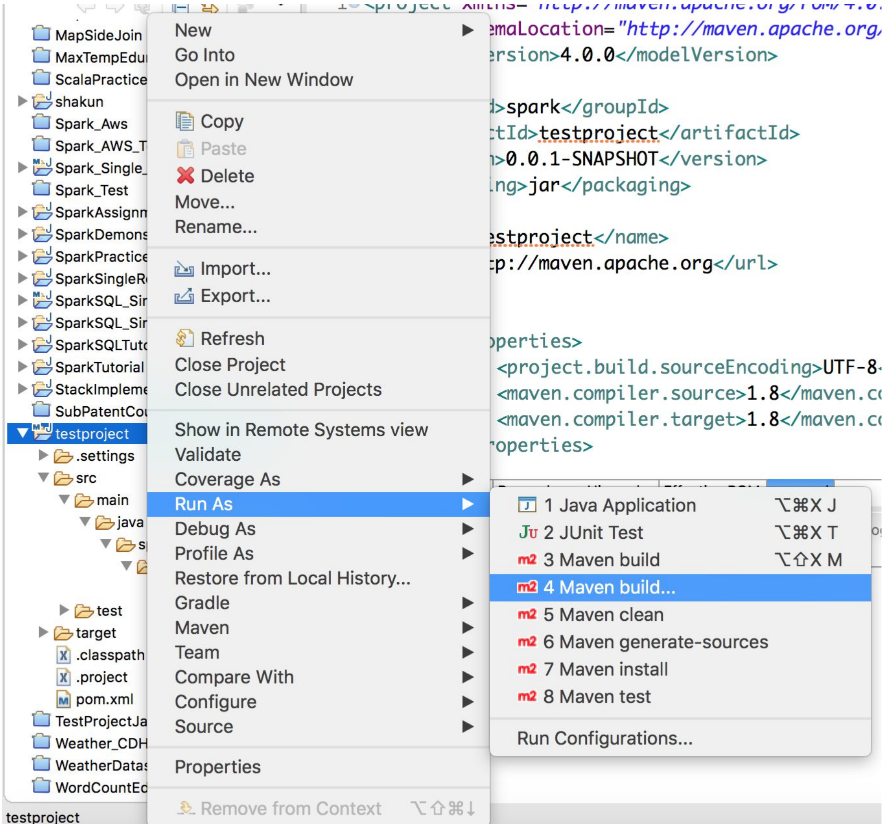
```
<dependency>
        <groupId>org.apache.spark</groupId>
        <artifactId>spark-core_2.11</artifactId>
        <version>2.2.0</version>
</dependency>
```



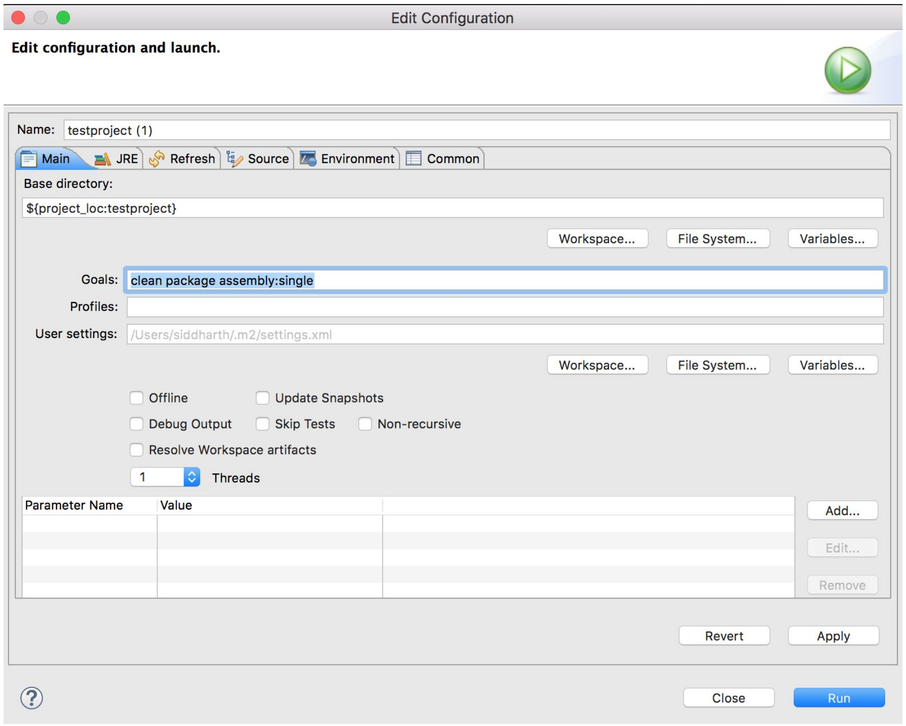
```
    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>1.8</maven.compiler.source>
        <maven.compiler.target>1.8</maven.compiler.target>
    </properties>

<dependencies>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>3.8.1</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.apache.spark</groupId>
        <artifactId>spark-core_2.11</artifactId>
        <version>2.2.0</version>
    </dependency>
</dependencies>
```

Compiling and Creating the Executable Jar

- Right click on project ”testproject” → *Run as* → *4 Maven Build ..* → <Type “clean package assembly:single” in Goals> → *Run* . This will clean all the previously created class files or jar files and compile the project again.



```
emaLocation="http://maven.apache.org
ersion>4.0.0</modelVersion>

d>spark</groupId>
tId>testproject</artifactId>
n>0.0.1-SNAPSHOT</version>
ing>jar</packaging>

estproject</name>
tp://maven.apache.org</url>

operties>
  <project.build.sourceEncoding>UTF-8
  <maven.compiler.source>1.8</maven.c
  <maven.compiler.target>1.8</maven.c
roperties>
```



Edit configuration and launch.

Name: testproject (1)

Base directory: ${project_loc:testproject}

Goals: clean package assembly:single

Profiles:

User settings: /Users/siddharth/.m2/settings.xml

□ Offline □ Update Snapshots

□ Debug Output □ Skip Tests □ Non-recursive

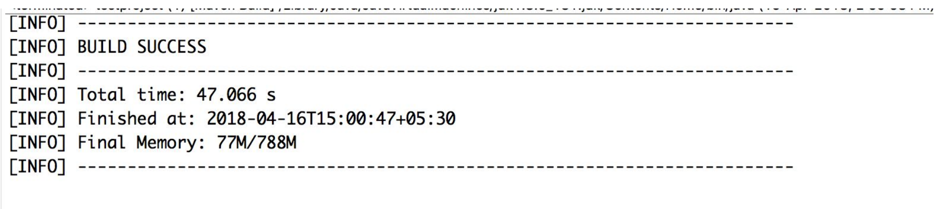
□ Resolve Workspace artifacts

1 Threads

Parameter Name	Value

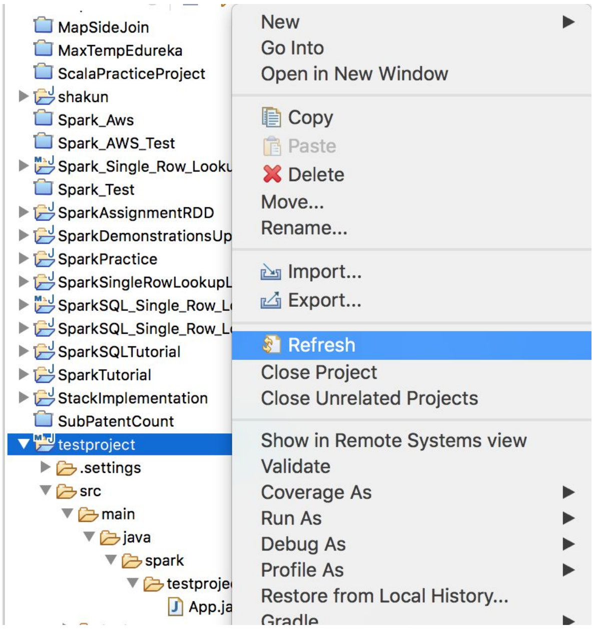
Compilation of code using maven starts. All the necessary dependency jars will be downloaded into your computer hence make sure you are connected to internet during this activity.

- Once the compilation is finished you will see the following message in your console.

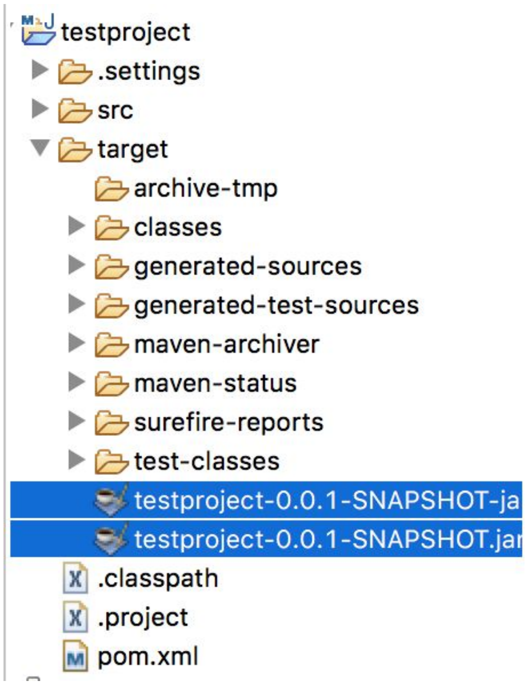


```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 47.066 s
[INFO] Finished at: 2018-04-16T15:00:47+05:30
[INFO] Final Memory: 77M/788M
[INFO] ------------------------------------------------------------------------
```

- Right click on the project and click refresh

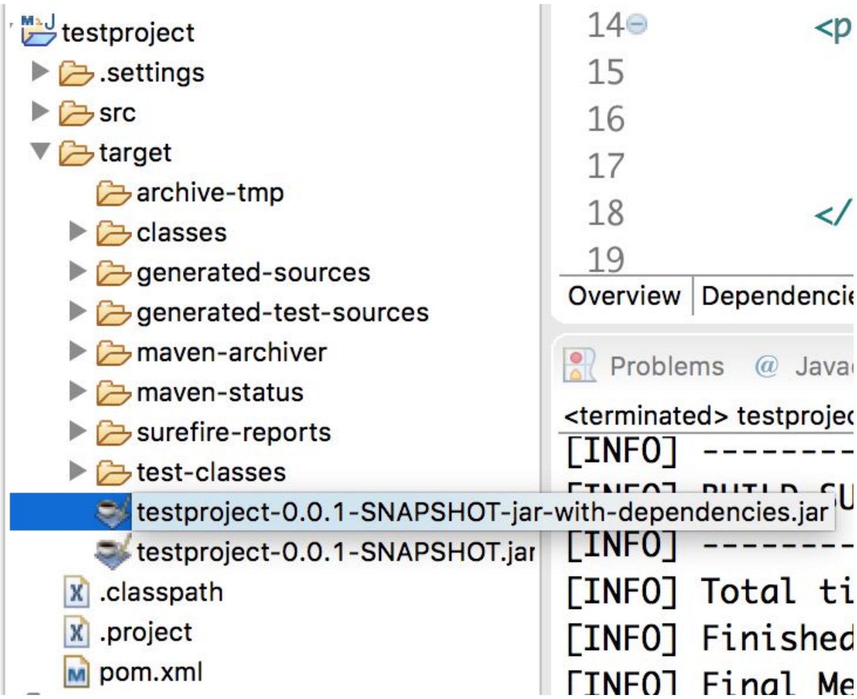


- In the project “testproject” go to the folder named “target” where you can see all the required jars are created.



```
testproject
  .settings
  src
  target
    archive-tmp
    classes
    generated-sources
    generated-test-sources
    maven-archiver
    maven-status
    surefire-reports
    test-classes
    testproject-0.0.1-SNAPSHOT-ja
    testproject-0.0.1-SNAPSHOT.jar
  .classpath
  .project
  pom.xml
```

The jar having the text “jar-with-dependencies” in its name is the executable jar which can be executed in the shell.



```
testproject
  .settings
  src
  target
    archive-tmp
    classes
    generated-sources
    generated-test-sources
    maven-archiver
    maven-status
    surefire-reports
    test-classes
    testproject-0.0.1-SNAPSHOT-jar-with-dependencies.jar
    testproject-0.0.1-SNAPSHOT.jar
  .classpath
  .project
  pom.xml
```

```
<terminated> testprojec
[INFO] --------
[INFO] BUILD SU
[INFO] --------
[INFO] Total ti
[INFO] Finished
[INFO] Final Me
```