Assignment 25:

Give your own input in output-screenshot of report.

Task 1 As discussed in class integrate Spark Hive Task

Steps to integrate Spark hive:

---

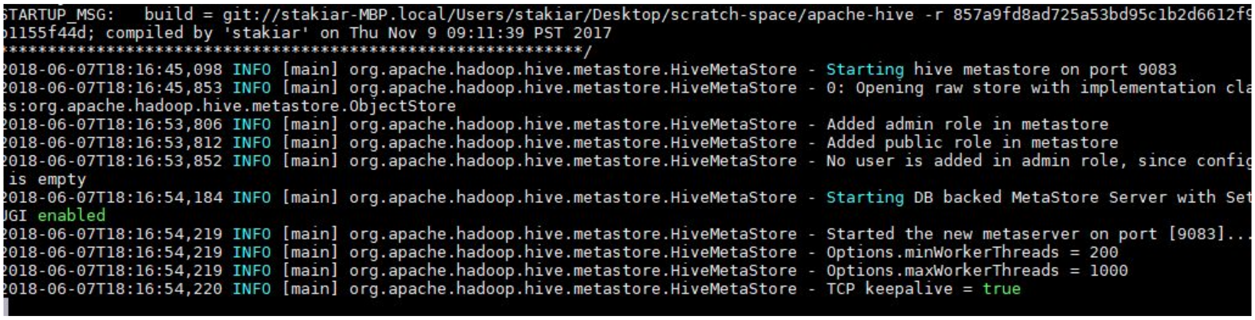
2. Copy the hive-site.xml file from $HIVE_HOME/conf to $SPARK_HOME/conf
3. Add the following properties to hive-site.xml on spark side :

- <property>
- <name>hive.metastore.uris</name>
- <value>thrift://localhost:9083</value>
- <description>password for connecting to mysql server</description>
- </property>

4. Write the code in Scala IDE to list the Databases in the hive. Source code is uploaded separately.
5. Make sure that your hadoop is started
6. Start hive metastore by executing hive —service metastore command.
7. Run the code from the IDE. You should be able to see all the hive databases.

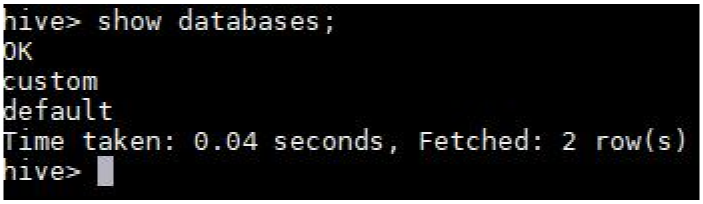
Output Screen Shot:

Starting metastore:



```
STARTUP_MSG:   build = git://stakiar-MBP.local/Users/stakiar/Desktop/scratch-space/apache-hive -r 857a9fd8ad725a53bd95c1b2d6612f9
b1155f44d; compiled by 'stakiar' on Thu Nov 9 09:11:39 PST 2017
************************************************************/
2018-06-07T18:16:45,098 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Starting hive metastore on port 9083
2018-06-07T18:16:45,853 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - 0: Opening raw store with implementation cla
ss:org.apache.hadoop.hive.metastore.ObjectStore
2018-06-07T18:16:53,806 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Added admin role in metastore
2018-06-07T18:16:53,812 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Added public role in metastore
2018-06-07T18:16:53,852 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - No user is added in admin role, since config
 is empty
2018-06-07T18:16:54,184 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Starting DB backed MetaStore Server with Set
UGI enabled
2018-06-07T18:16:54,219 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Started the new metaserver on port [9083]..
2018-06-07T18:16:54,219 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Options.minWorkerThreads = 200
2018-06-07T18:16:54,219 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - Options.maxWorkerThreads = 1000
2018-06-07T18:16:54,220 INFO [main] org.apache.hadoop.hive.metastore.HiveMetaStore - TCP keepalive = true
```

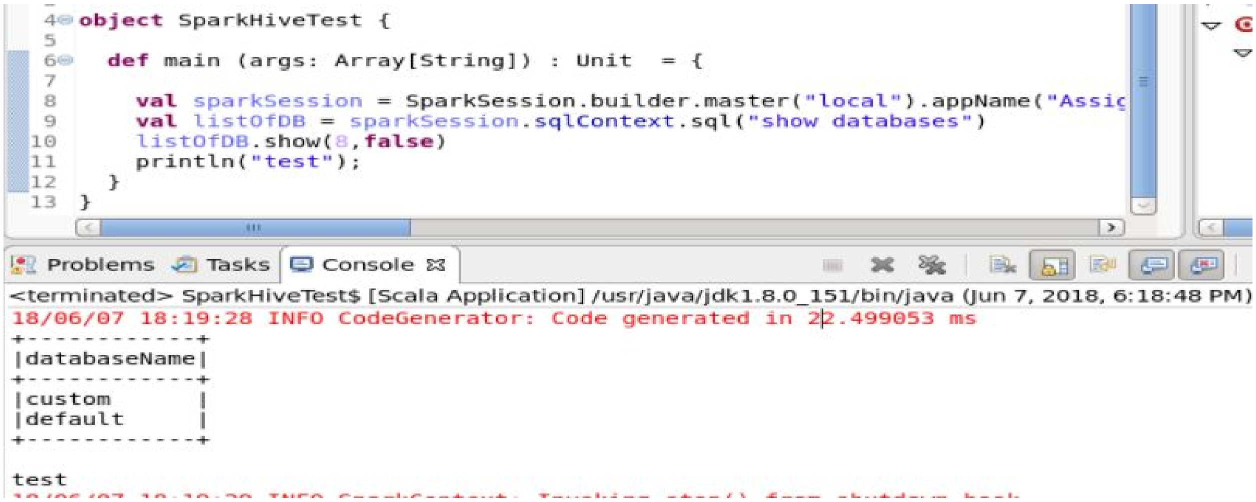
Hive database in the terminal:



```
hive> show databases;
OK
custom
default
Time taken: 0.04 seconds, Fetched: 2 row(s)
hive> 
```

Output after running the code in Scala Ide

--



```
object SparkHiveTest {

  def main (args: Array[String]) : Unit  = {

    val sparkSession = SparkSession.builder.master("local").appName("Assig
    val listOfDB = sparkSession.sqlContext.sql("show databases")
    listOfDB.show(8,false)
    println("test");
  }
}
```

```
<terminated> SparkHiveTest$ [Scala Application] /usr/java/jdk1.8.0_151/bin/java (Jun 7, 2018, 6:18:48 PM)
18/06/07 18:19:28 INFO CodeGenerator: Code generated in 22.499053 ms
+------------+
|databaseName|
+------------+
|custom      |
|default     |
+------------+

test
18/06/07 18:19:28 INFO SparkContext: Invoking stop() from shutdown hook
```

2. As discussed in class integrate Spark Hbase Task
Steps:

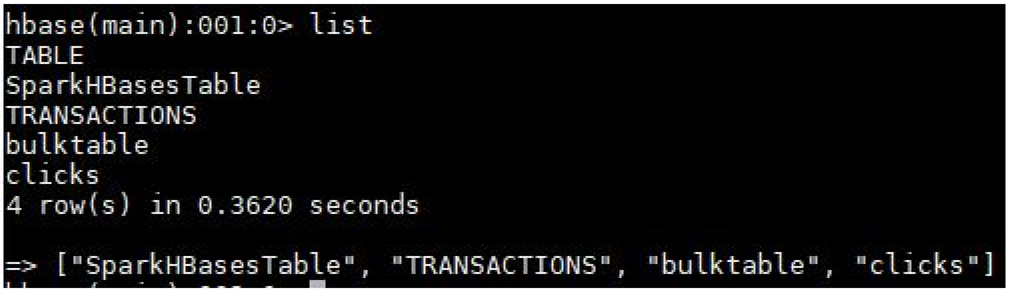
1.Write an API code in scala ide to create a new table in hbase. Source code is uploaded separately.

2. Run the code in scala ide and check the hbase for the newly created table.

3. Make sure to start the hbase shell using the below commands
Start-hbase.sh
Hbase shell

Output Screenshots:

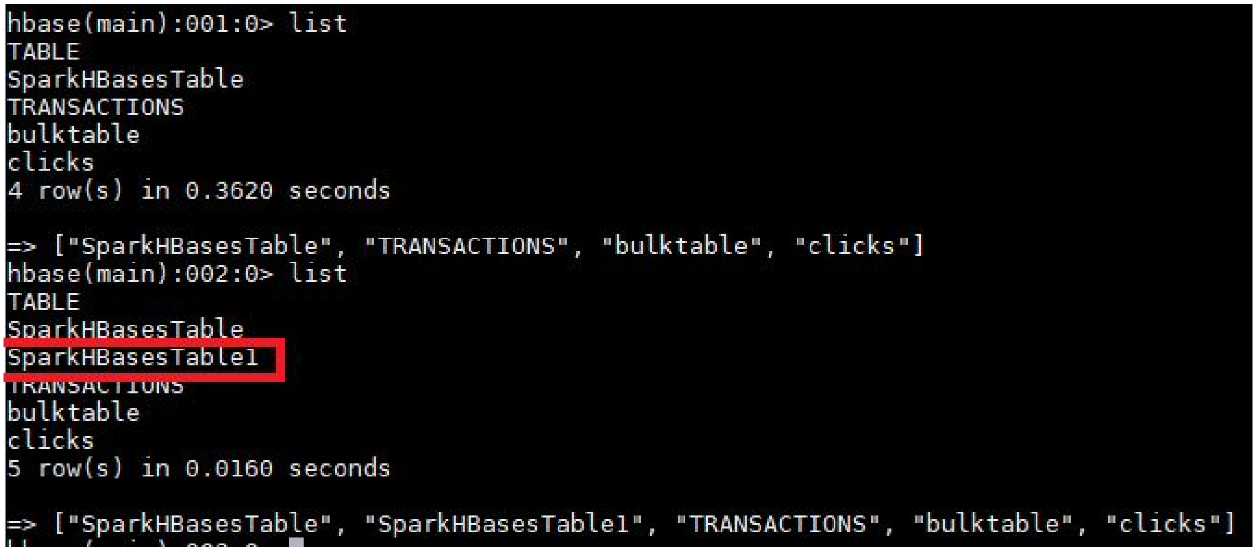
List of tables before running the code:



```
hbase(main):001:0> list
TABLE
SparkHBasesTable
TRANSACTIONS
bulktable
clicks
4 row(s) in 0.3620 seconds

=> ["SparkHBasesTable", "TRANSACTIONS", "bulktable", "clicks"]
```

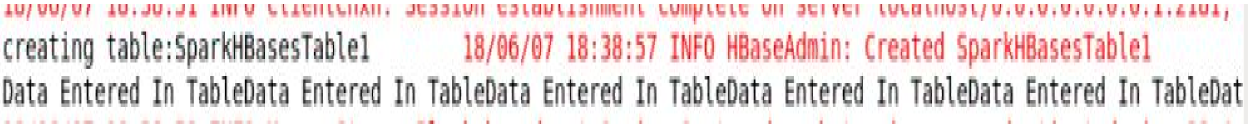
List of tables after running the code:

Newly created table highlighted in red



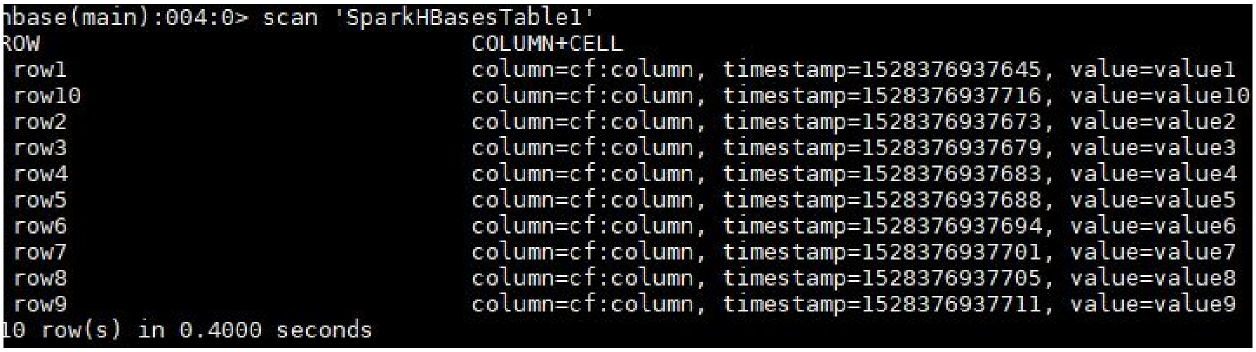
```
hbase(main):001:0> list
TABLE
SparkHBasesTable
TRANSACTIONS
bulktable
clicks
4 row(s) in 0.3620 seconds

=> ["SparkHBasesTable", "TRANSACTIONS", "bulktable", "clicks"]
hbase(main):002:0> list
TABLE
SparkHBasesTable
SparkHBasesTable1
TRANSACTIONS
bulktable
clicks
5 row(s) in 0.0160 seconds

=> ["SparkHBasesTable", "SparkHBasesTable1", "TRANSACTIONS", "bulktable", "clicks"]
```

Console output:



```
18/06/07 18:38:51 INFO ClientCnxn: Session establishment complete on server localhost/0:0:0:0:0:0:0:1:2181,
creating table:SparkHBasesTable1       18/06/07 18:38:57 INFO HBaseAdmin: Created SparkHBasesTable1
Data Entered In TableData Entered In TableData Entered In TableData Entered In TableData Entered In TableDat
```

Newly created table with column family, column and value



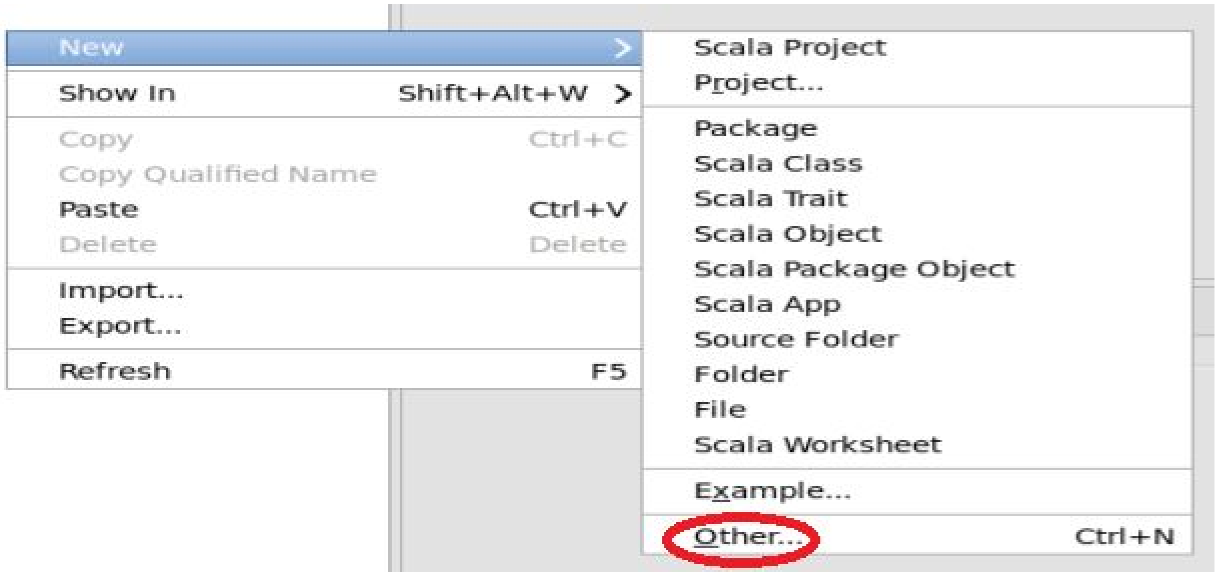
```
hbase(main):004:0> scan 'SparkHBasesTable1'
ROW                               COLUMN+CELL
 row1                             column=cf:column, timestamp=1528376937645, value=value1
 row10                            column=cf:column, timestamp=1528376937716, value=value10
 row2                             column=cf:column, timestamp=1528376937673, value=value2
 row3                             column=cf:column, timestamp=1528376937679, value=value3
 row4                             column=cf:column, timestamp=1528376937683, value=value4
 row5                             column=cf:column, timestamp=1528376937688, value=value5
 row6                             column=cf:column, timestamp=1528376937694, value=value6
 row7                             column=cf:column, timestamp=1528376937701, value=value7
 row8                             column=cf:column, timestamp=1528376937705, value=value8
 row9                             column=cf:column, timestamp=1528376937711, value=value9
10 row(s) in 0.4000 seconds
```

3 As discussed in class integrate Spark Kafka

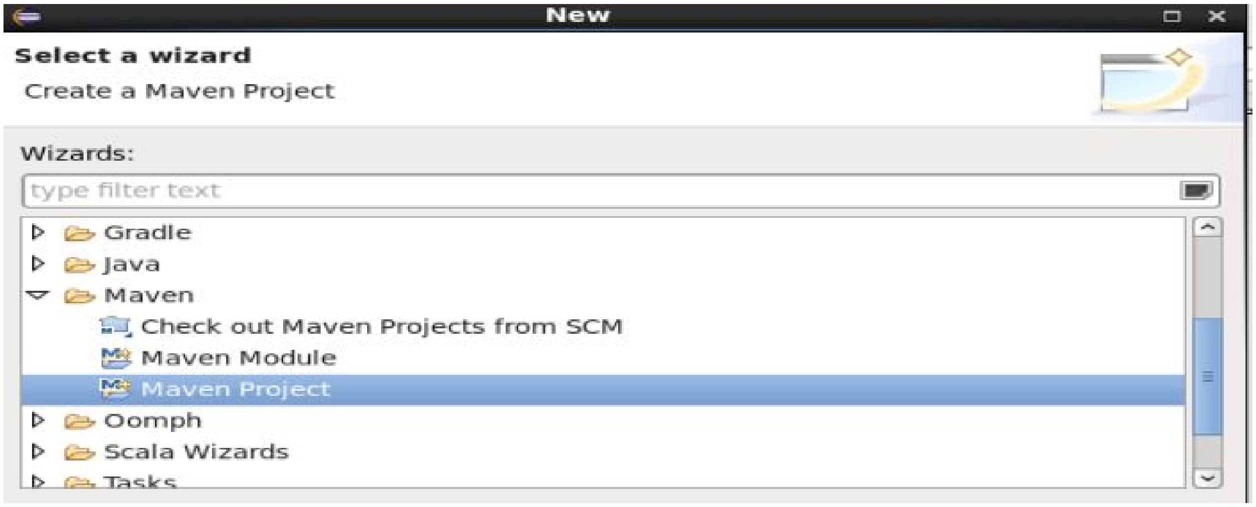
Steps:

Create a Maven project as given in the below screenshots:

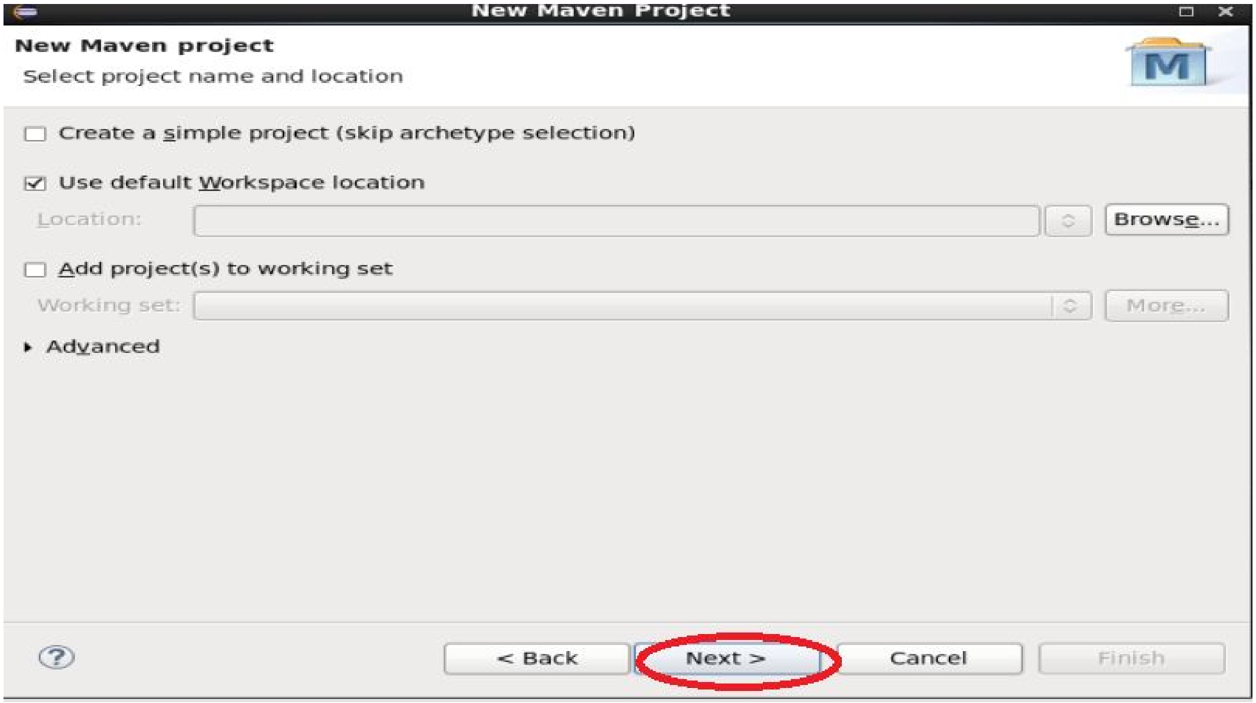
Right click on the package explorer and select others as below:



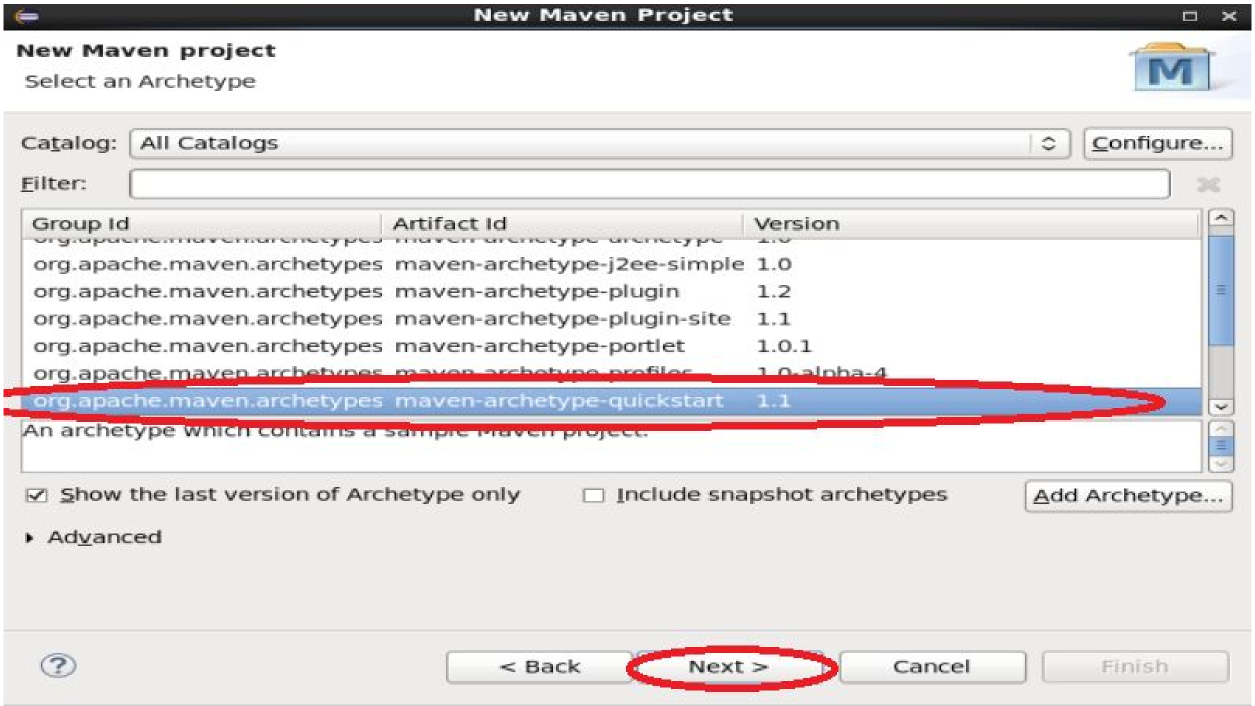
Select maven project as shown below:



In the new Maven project wizard select next as below


New Maven Project
New Maven project
Select project name and location
□ Create a simple project (skip archetype selection)
☑ Use default Workspace location
Location:
Browse...
□ Add project(s) to working set
Working set:
More...
Advanced
< Back
Next >
Cancel
Finish

Select the below option highlighted in red and click next:



Catalog: All Catalogs

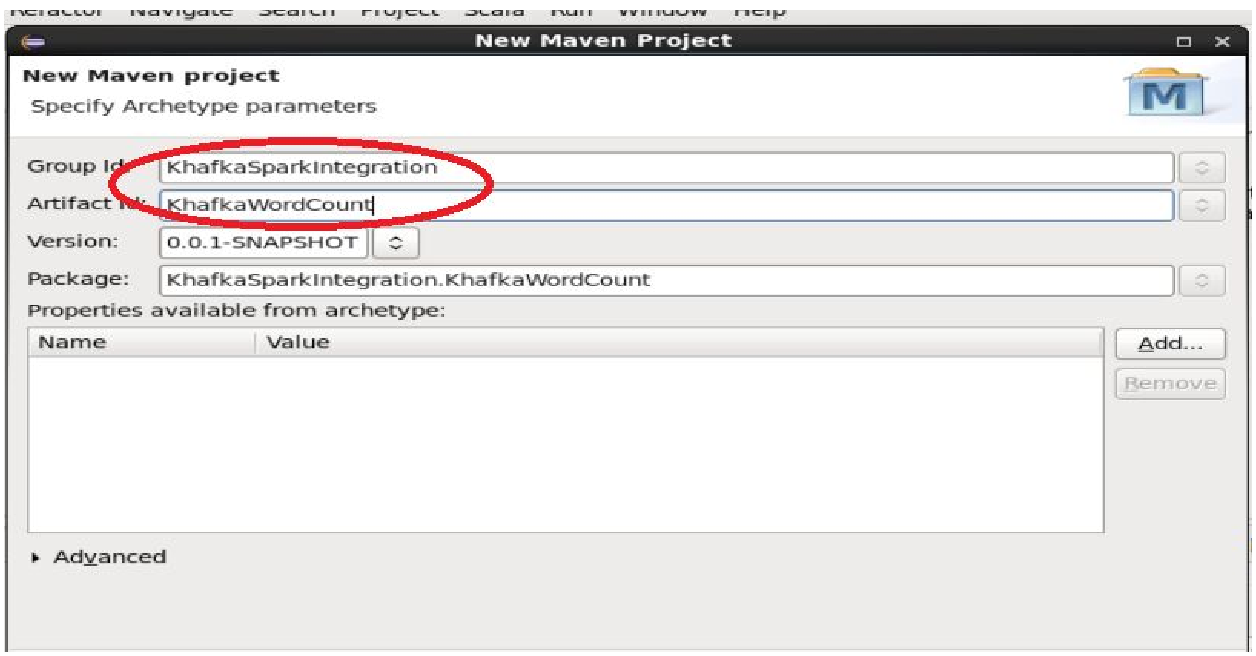
Filter:

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

An archetype which contains a sample Maven project.

☑ Show the last version of Archetype only □ Include snapshot archetypes

Provide the GroupId and Artifact Id as given below


Refactor Navigate Search Project Scala Run Window Help
New Maven Project
New Maven project
Specify Archetype parameters
Add...
Remove
Advanced

Group Id: KhafkaSparkIntegration

Artifact Id: KhafkaWordCount

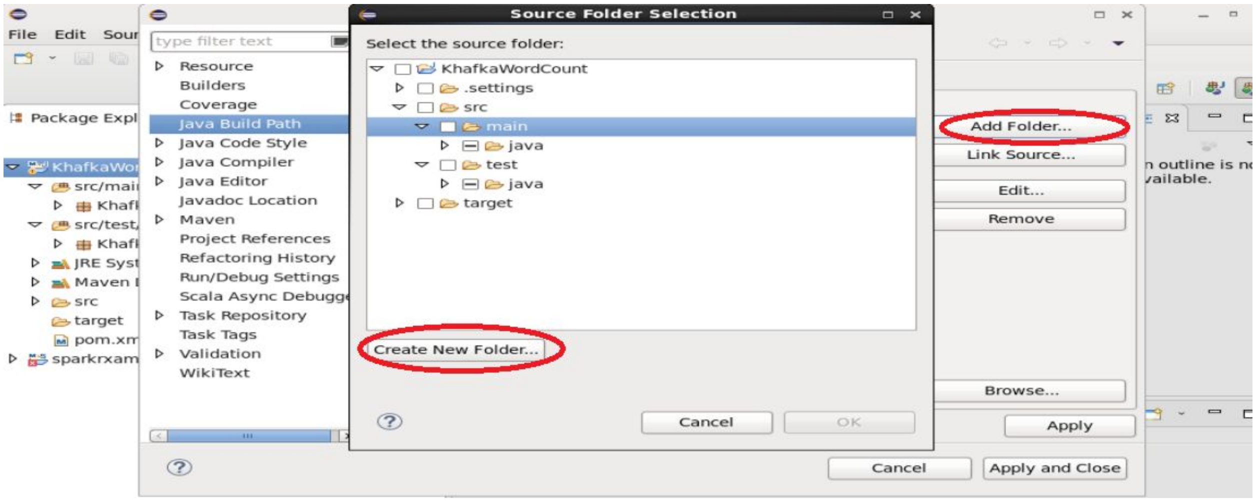
Version: 0.0.1-SNAPSHOT

Package: KhafkaSparkIntegration.KhafkaWordCount

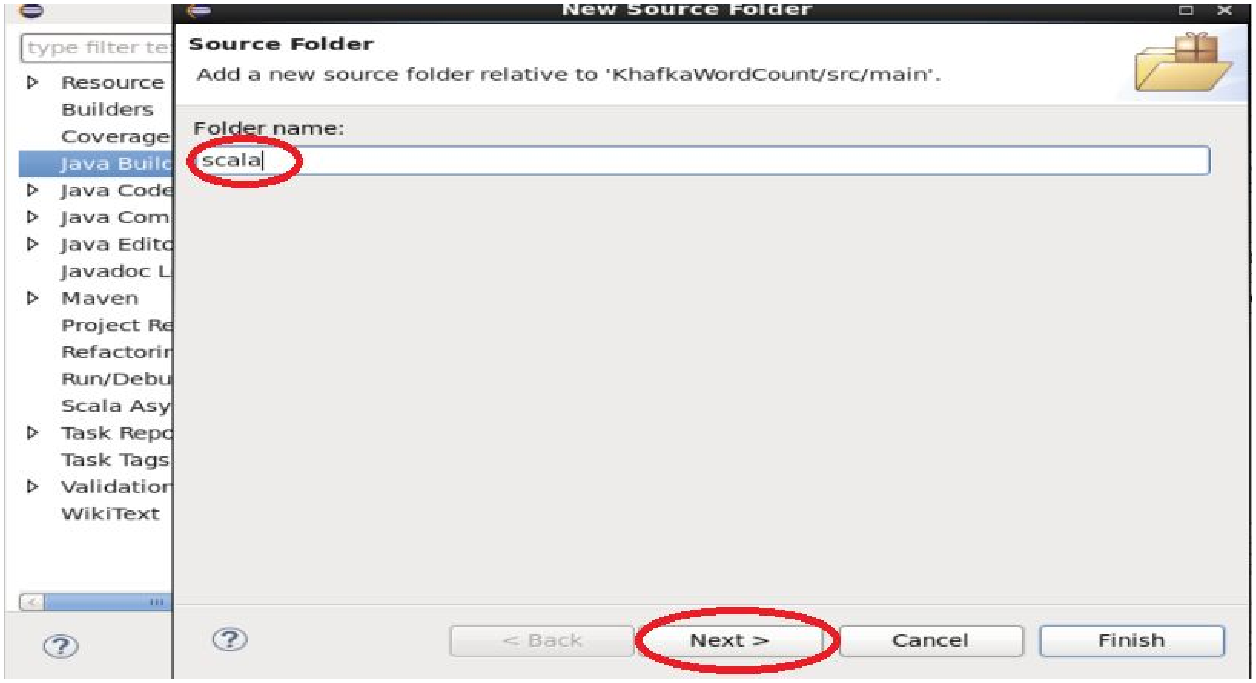
Properties available from archetype:

Name	Value

After creating Maven project create a new folder as given in the below screenshot



Create a folder named scala as below:

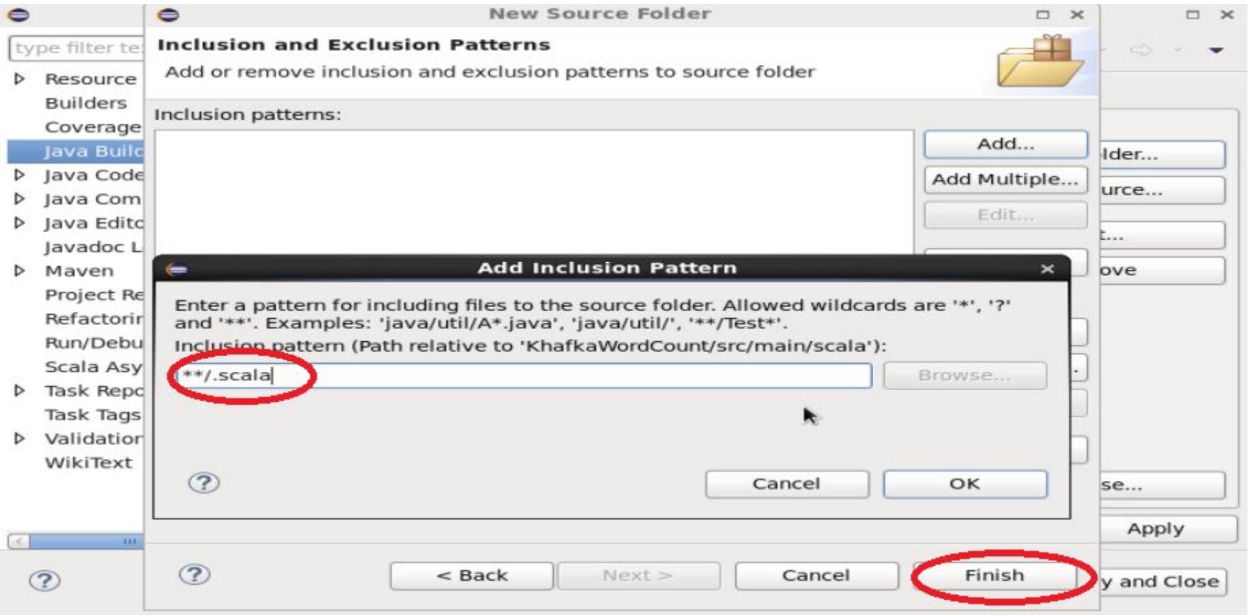


Source Folder

Add a new source folder relative to 'KhafkaWordCount/src/main'.

Folder name: scala

Give the inclusion and pattern as given below:



New Source Folder

Inclusion and Exclusion Patterns

Add or remove inclusion and exclusion patterns to source folder

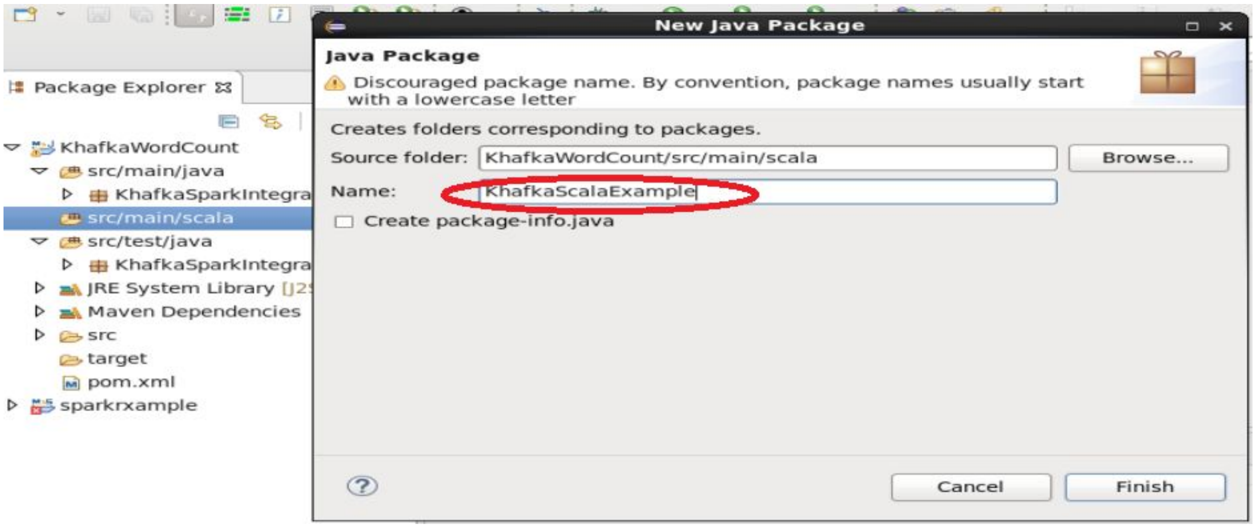
Inclusion patterns:

Add Inclusion Pattern

Enter a pattern for including files to the source folder. Allowed wildcards are '*', '?' and '**'. Examples: 'java/util/A*.java', 'java/util/', '**/Test*'.

Inclusion pattern (Path relative to 'KhafkaWordCount/src/main/scala'): **/.scala

Create a package under the newly created folder as given below:



New Java Package

Java Package

Discouraged package name. By convention, package names usually start with a lowercase letter

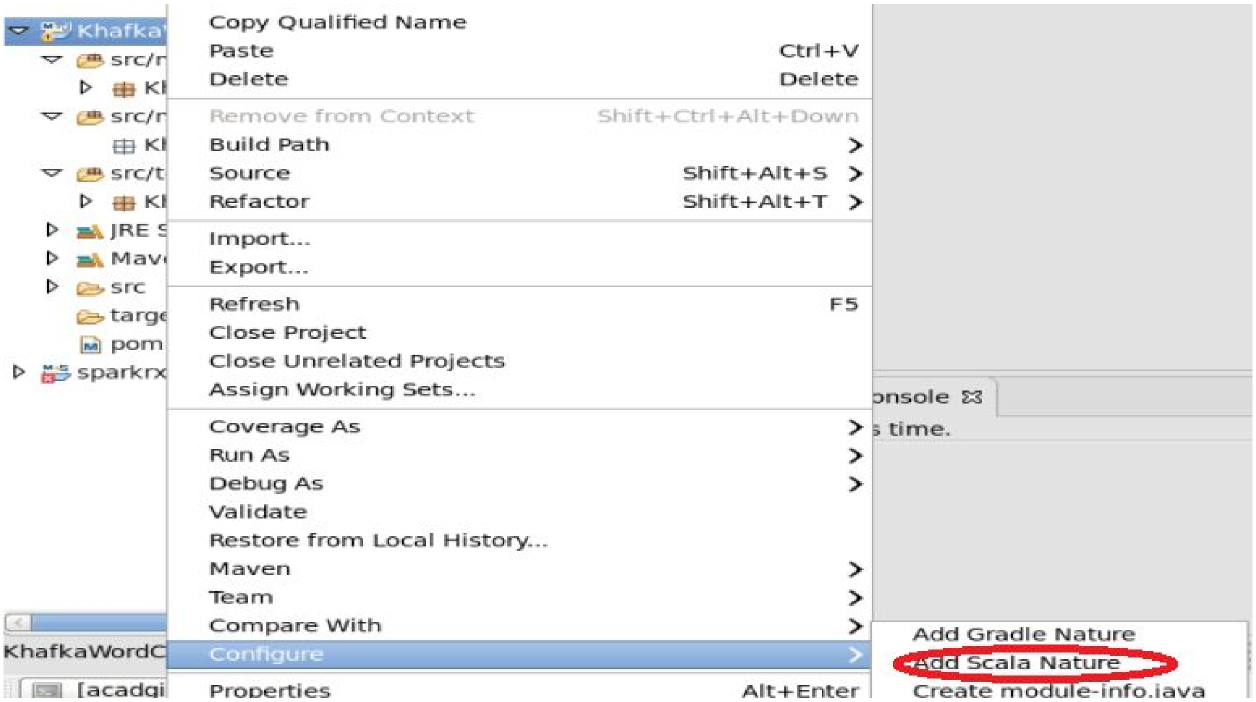
Creates folders corresponding to packages.

Source folder: KhafkaWordCount/src/main/scala

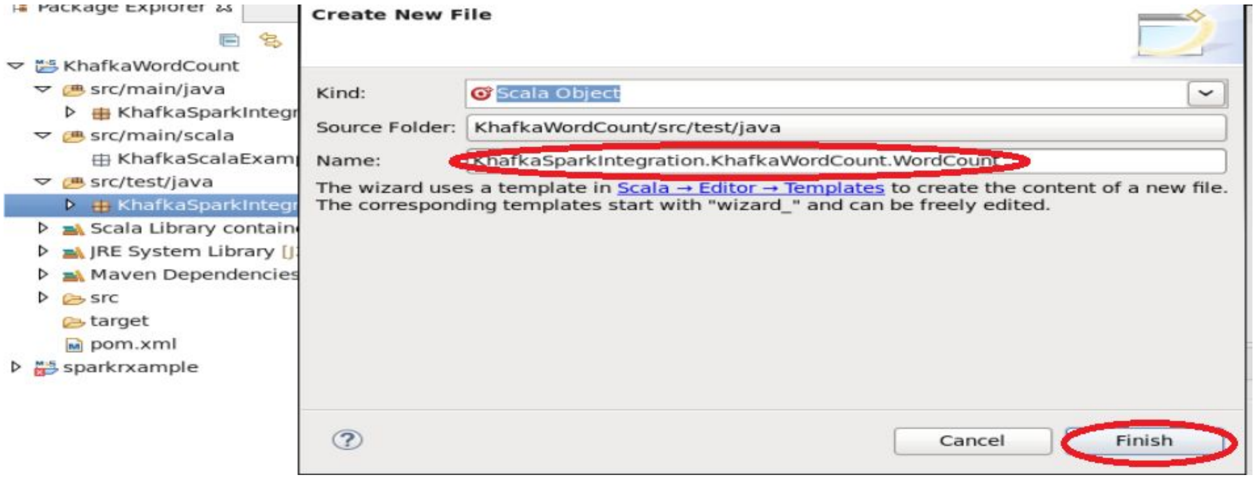
Name: KhafkaScalaExample

□ Create package-info.java

Select the project and configure as Scala nature as given below:



Create a scala object under the newly created package as below:



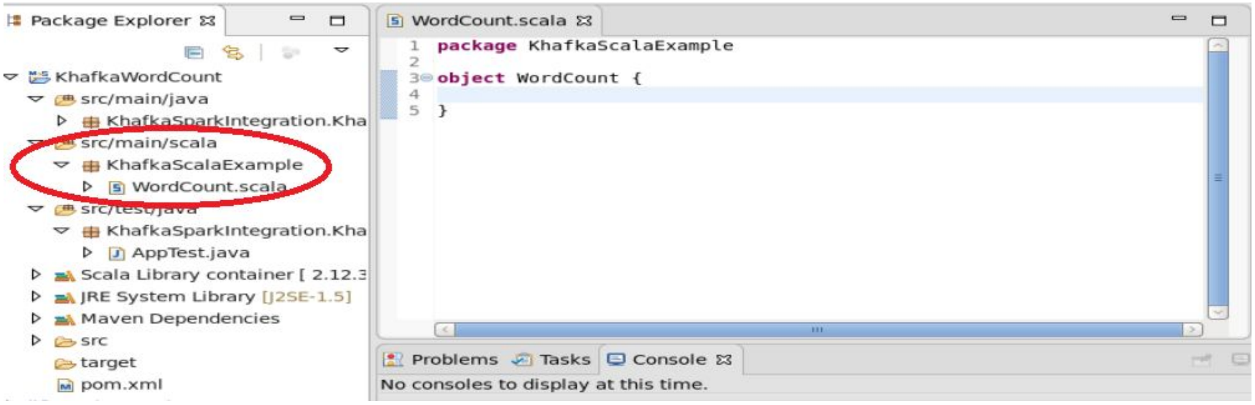
Kind: Scala Object

Source Folder: KhafkaWordCount/src/test/java

Name: KhafkaSparkIntegration.KhafkaWordCount.WordCount

The wizard uses a template in Scala → Editor → Templates to create the content of a new file. The corresponding templates start with "wizard_" and can be freely edited.

Find the project/folder/package structure below:

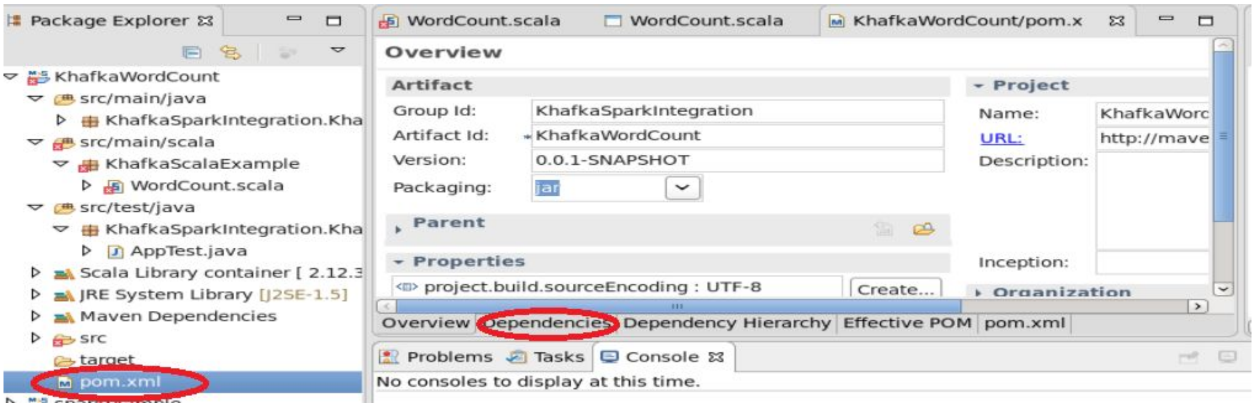


```
KhafkaWordCount
  src/main/java
    KhafkaSparkIntegration.Kha
  src/main/scala
    KhafkaScalaExample
      WordCount.scala
  src/test/java
    KhafkaSparkIntegration.Kha
      AppTest.java
  Scala Library container [ 2.12.3
  JRE System Library [J2SE-1.5]
  Maven Dependencies
  src
  target
  pom.xml
```

```
package KhafkaScalaExample

object WordCount {

}
```

No consoles to display at this time.

While creating a maven project an xml file named pom is created. Add the dependencies as given below:



```
KhafkaWordCount
  src/main/java
    KhafkaSparkIntegration.Kha
  src/main/scala
    KhafkaScalaExample
      WordCount.scala
  src/test/java
    KhafkaSparkIntegration.Kha
      AppTest.java
  Scala Library container [ 2.12.3
  JRE System Library [J2SE-1.5]
  Maven Dependencies
  src
  target
  pom.xml
```

Group Id: KhafkaSparkIntegration
Artifact Id: KhafkaWordCount
Version: 0.0.1-SNAPSHOT
Packaging: jar

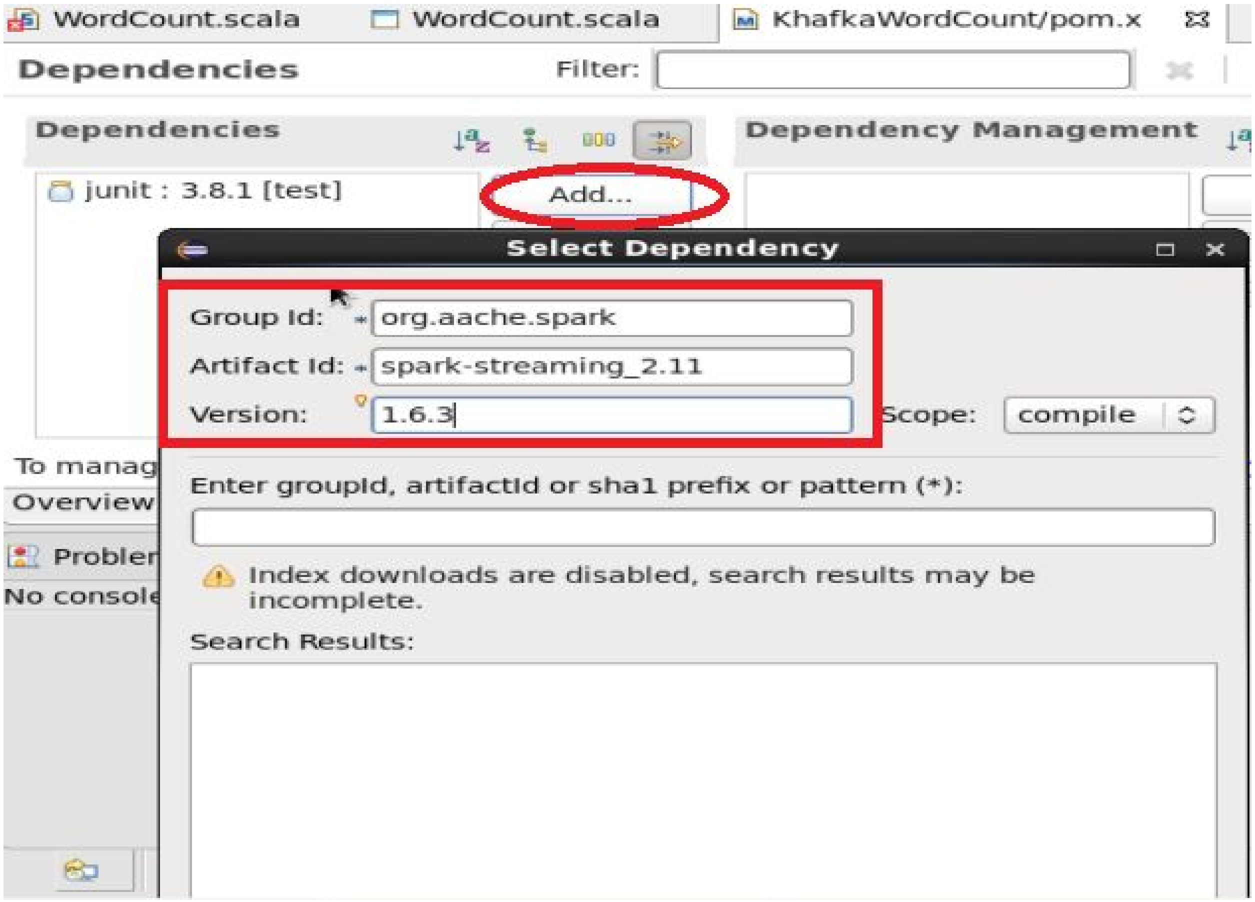
project.build.sourceEncoding : UTF-8

Name: KhafkaWorc
URL: http://mave
Description:
Inception:

No consoles to display at this time.

Add the below 2 dependencies highlighted in red:

1st dependency:



Group Id: org.aache.spark

Artifact Id: spark-streaming_2.11

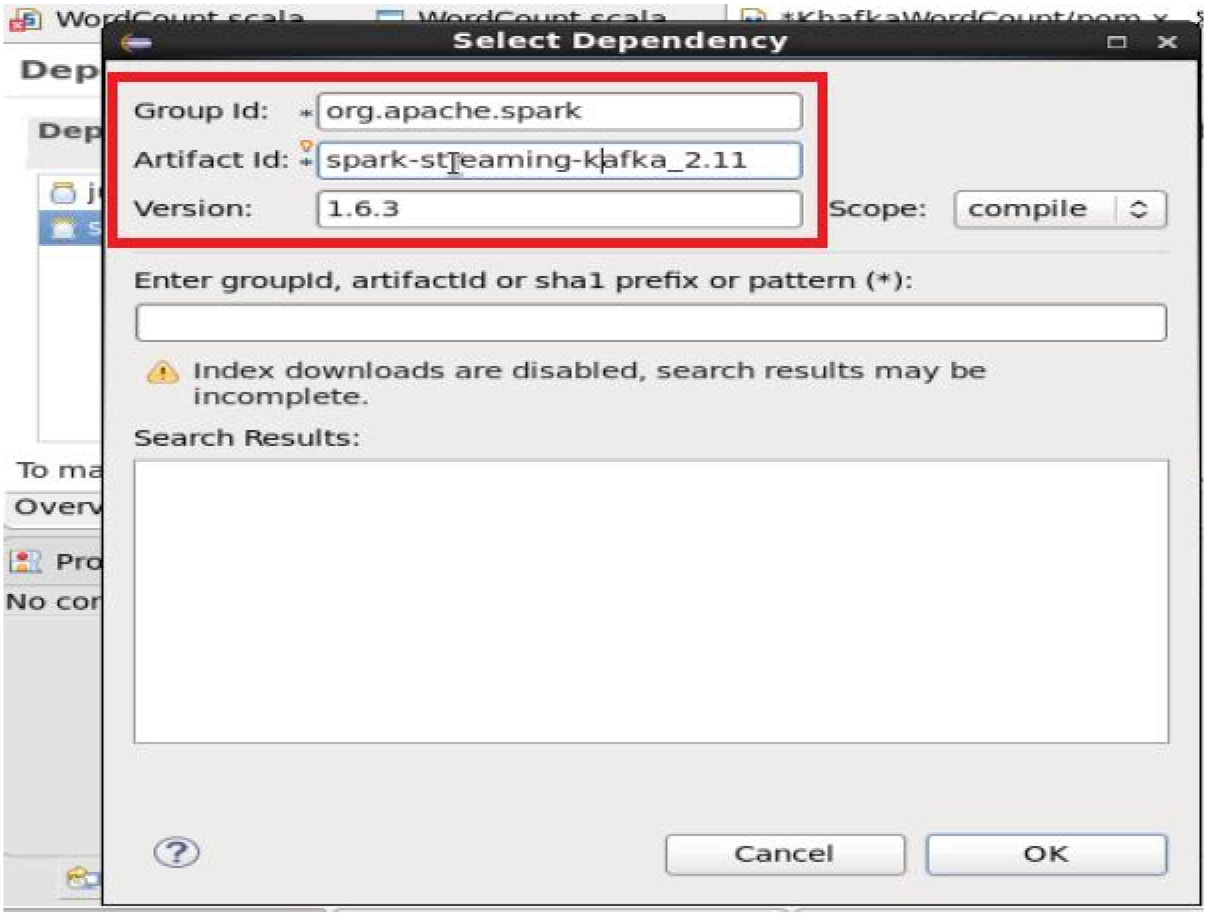
Version: 1.6.3

Scope: compile

Enter groupId, artifactId or sha1 prefix or pattern (*):

Index downloads are disabled, search results may be incomplete.

2nd dependency:



Group Id: org.apache.spark

Artifact Id: spark-streaming-kafka_2.11

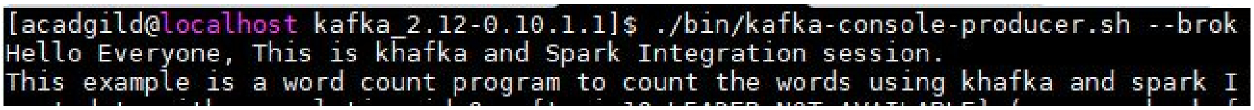
Version: 1.6.3

Scope: compile

Enter groupId, artifactId or sha1 prefix or pattern (*):

Index downloads are disabled, search results may be incomplete.

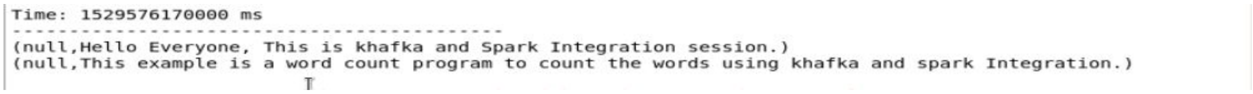
Input:



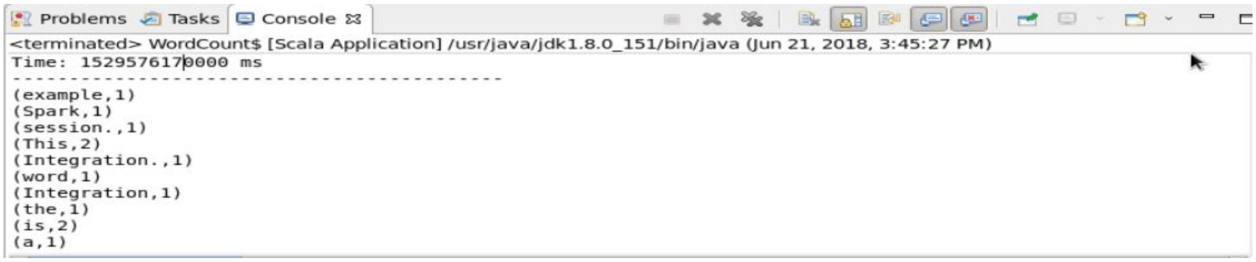
```
[acadgild@localhost kafka_2.12-0.10.1.1]$ ./bin/kafka-console-producer.sh --brok
Hello Everyone, This is khafka and Spark Integration session.
This example is a word count program to count the words using khafka and spark I
```

Output :

Wordcount screenshot:



```
Time: 1529576170000 ms
-------------------------------------------
(null,Hello Everyone, This is khafka and Spark Integration session.)
(null,This example is a word count program to count the words using khafka and spark Integration.)
```



```
<terminated> WordCount$ [Scala Application] /usr/java/jdk1.8.0_151/bin/java (Jun 21, 2018, 3:45:27 PM)
Time: 1529576170000 ms
-------------------------------------------
(example,1)
(Spark,1)
(session.,1)
(This,2)
(Integration.,1)
(word,1)
(Integration,1)
(the,1)
(is,2)
(a,1)
```