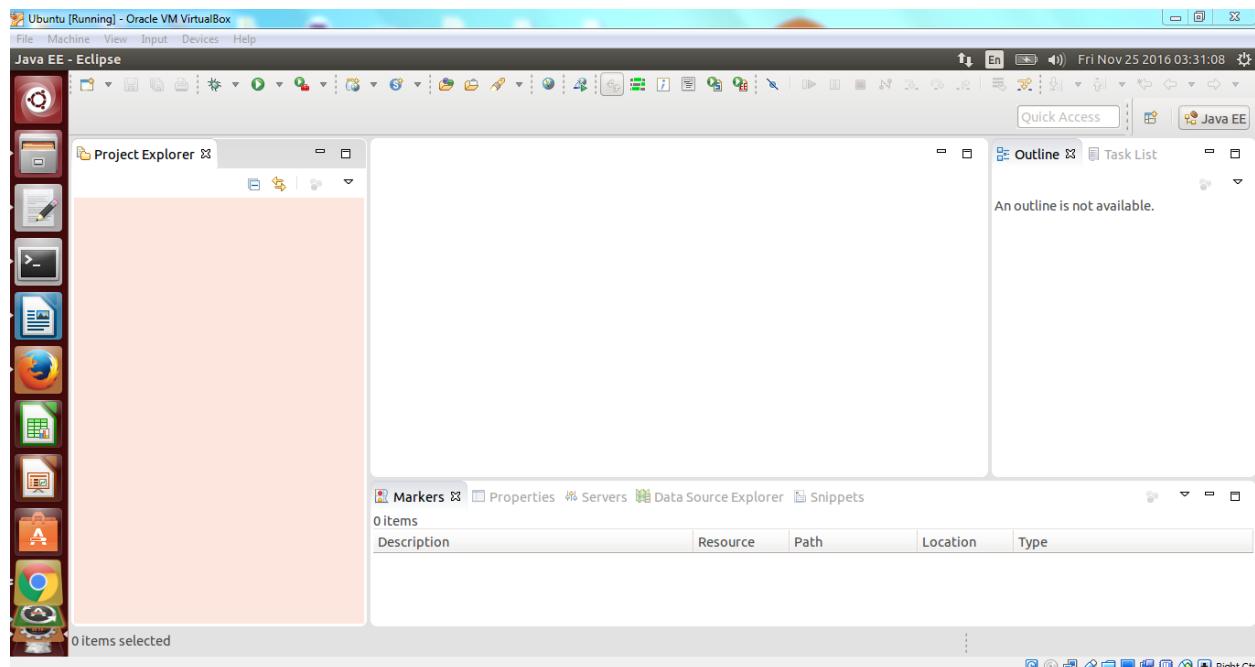
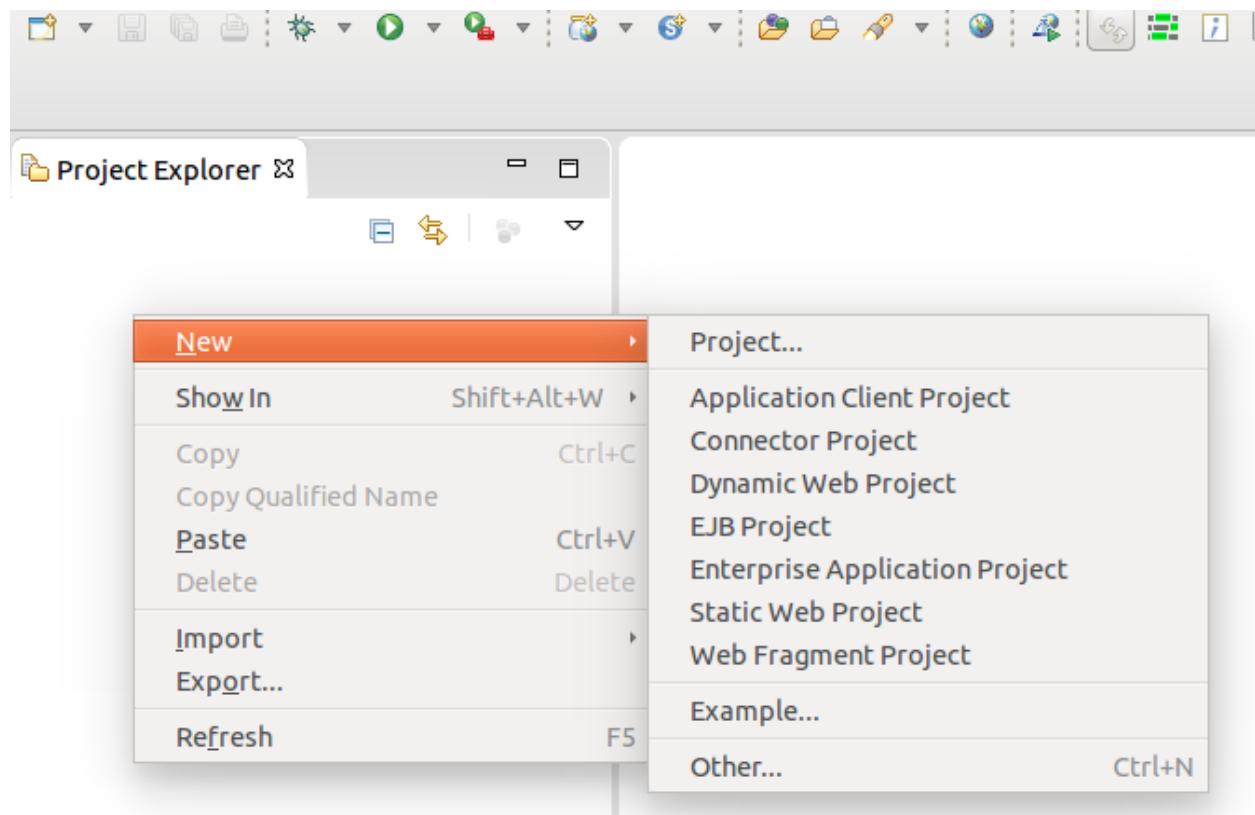


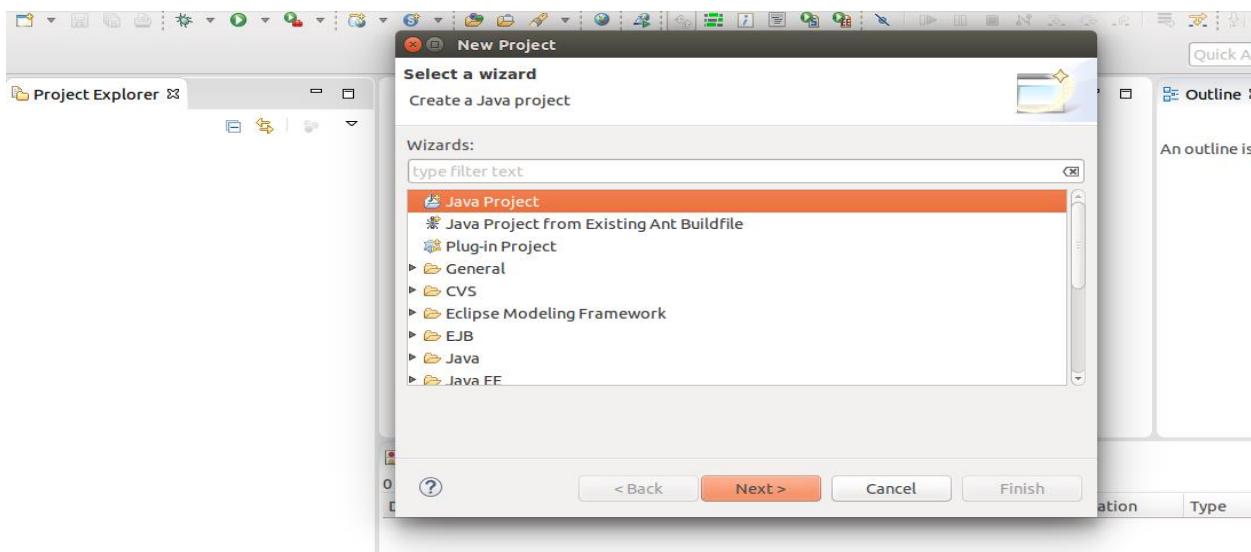
Start eclipse, once it is started.



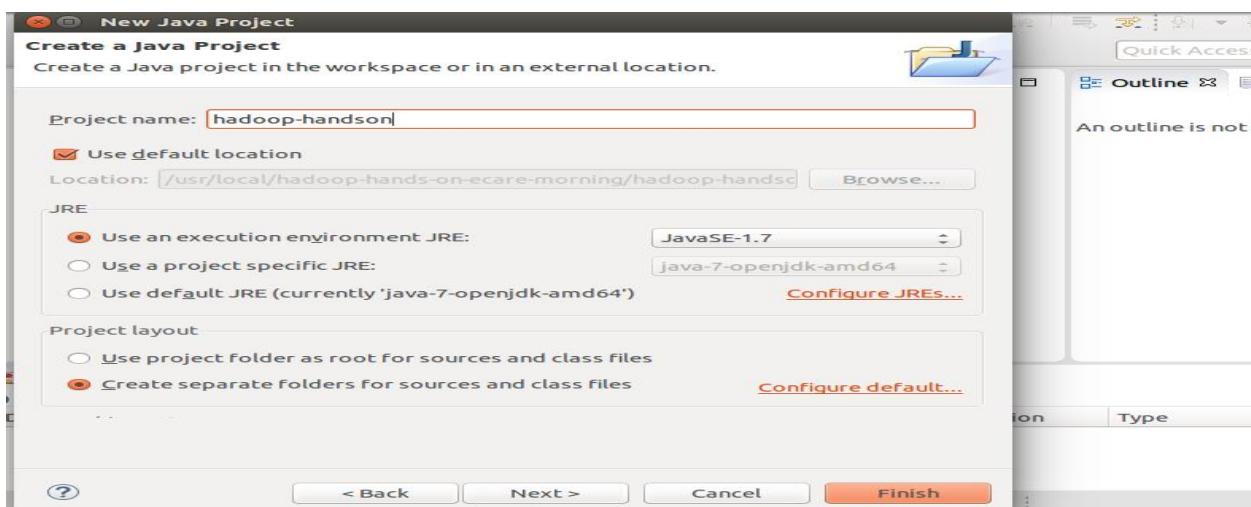
Create java project. Right click on the project explorer → select New → project → java project.

Follow below screens to create project

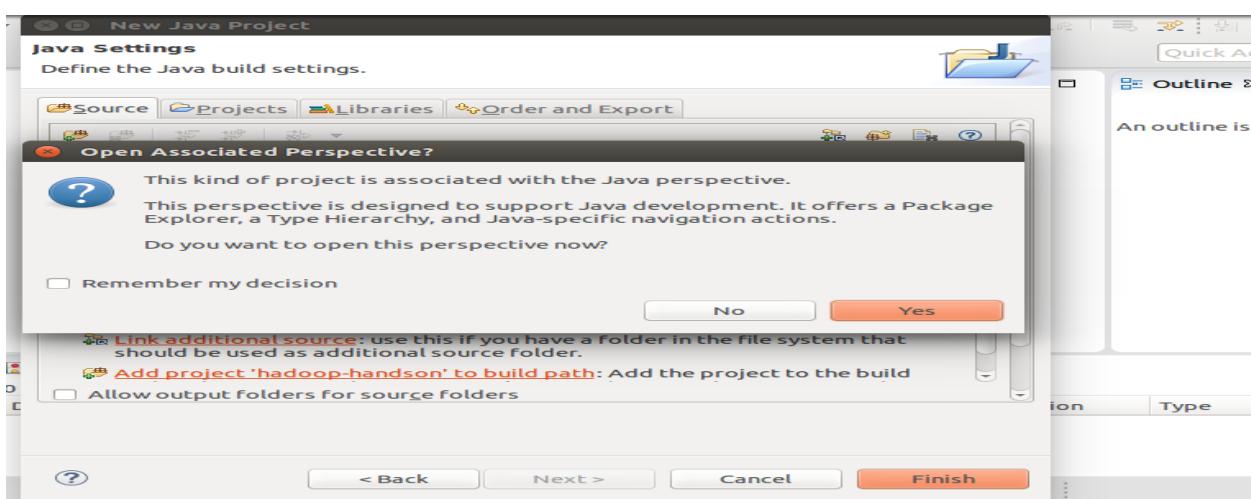




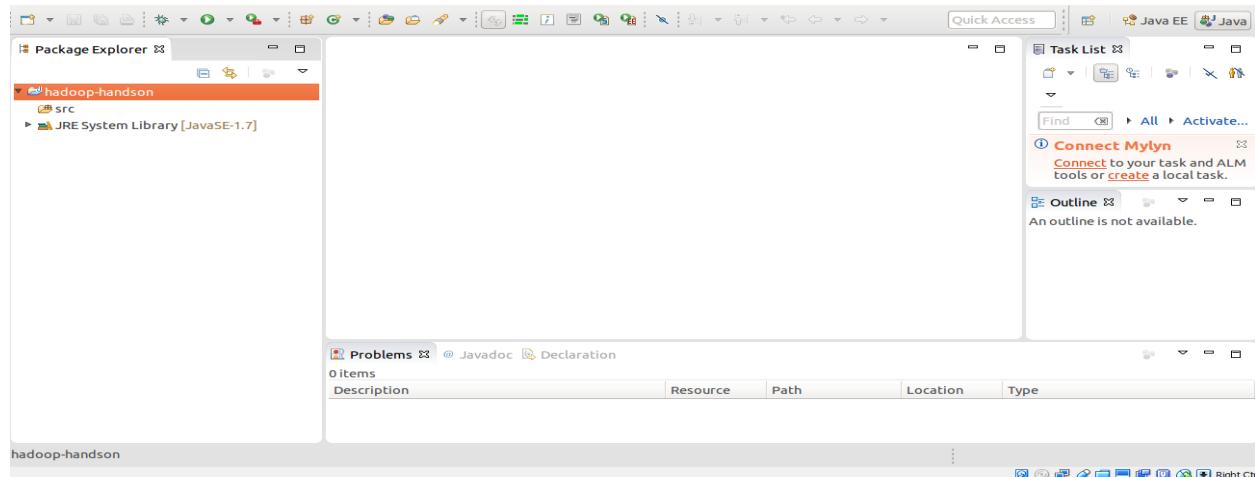
Give the project Name.. ex: mapreduce-hands-on and click on **finish** button



If it ask for prospective change—click on yes button.

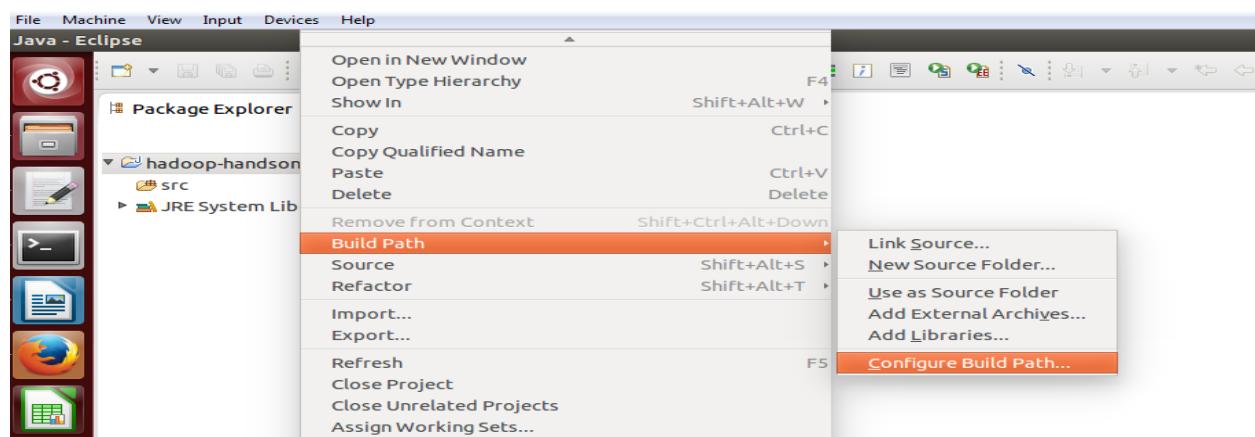


It will create new java-project on project Explorer

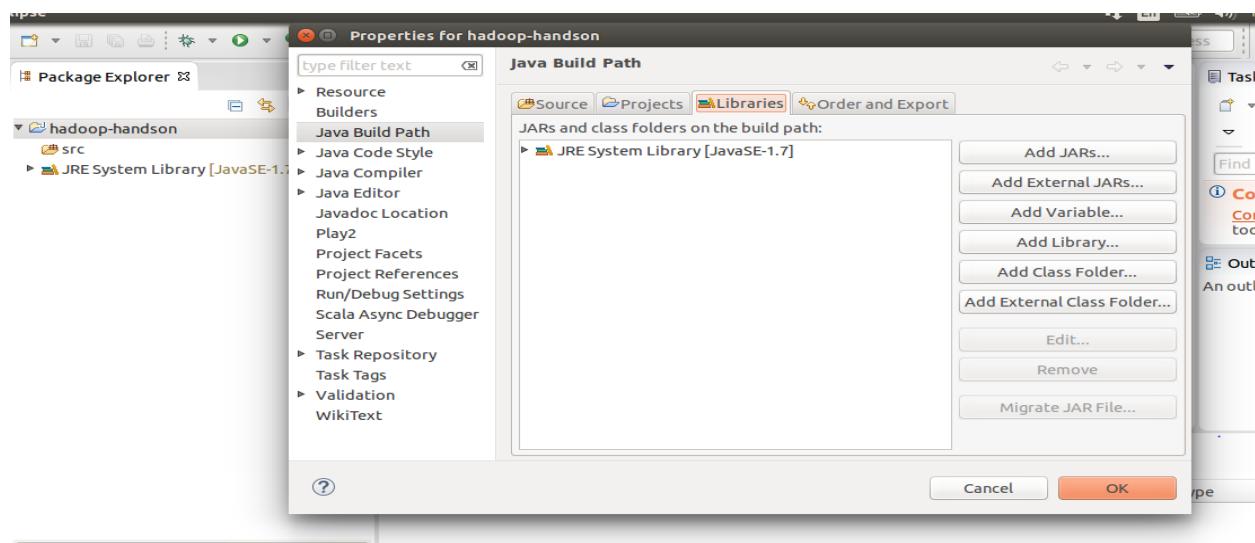


Set the Build Path:

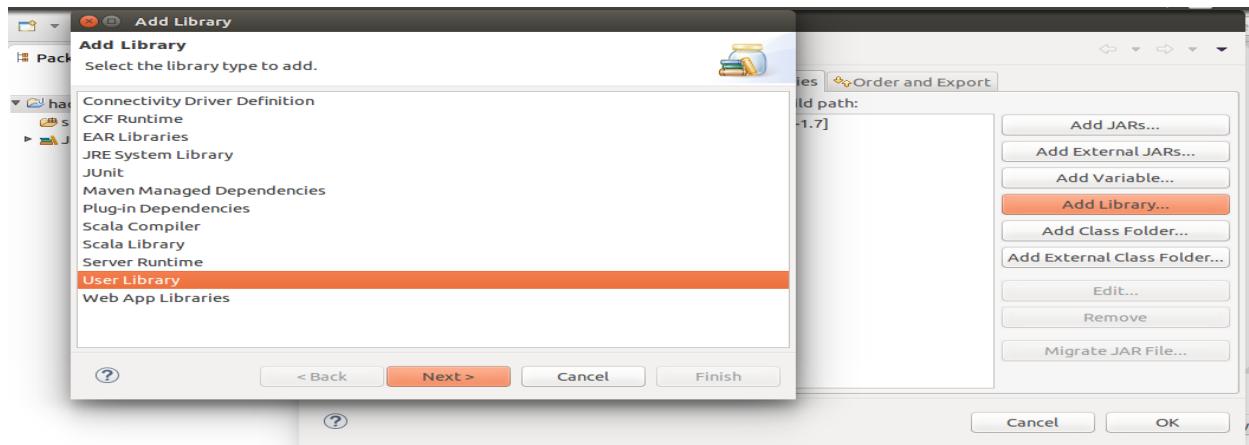
Right click on project → select **Build path** → **configure Build path**.



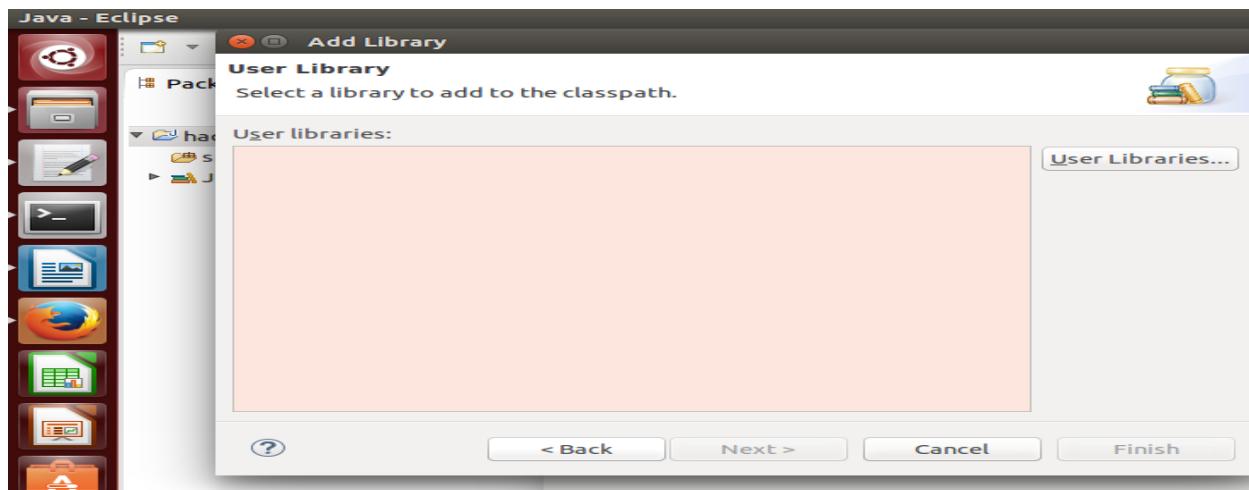
Click on “Libraries” from top pane and click on **Add library** from the right pane.



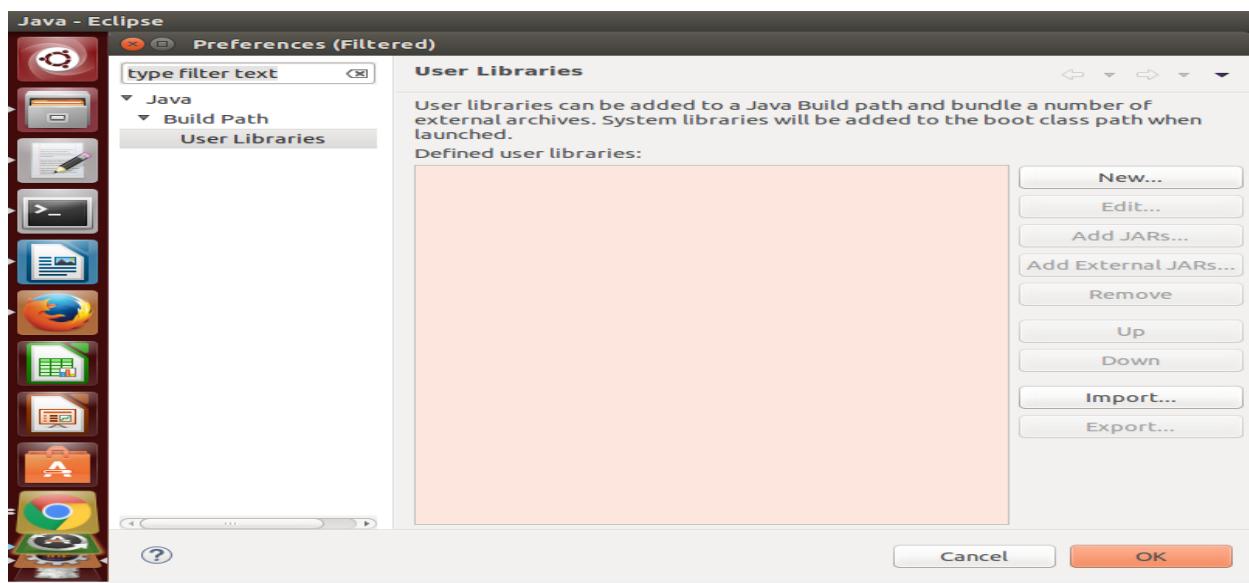
Select **User library** from left menu. See in below window and click next



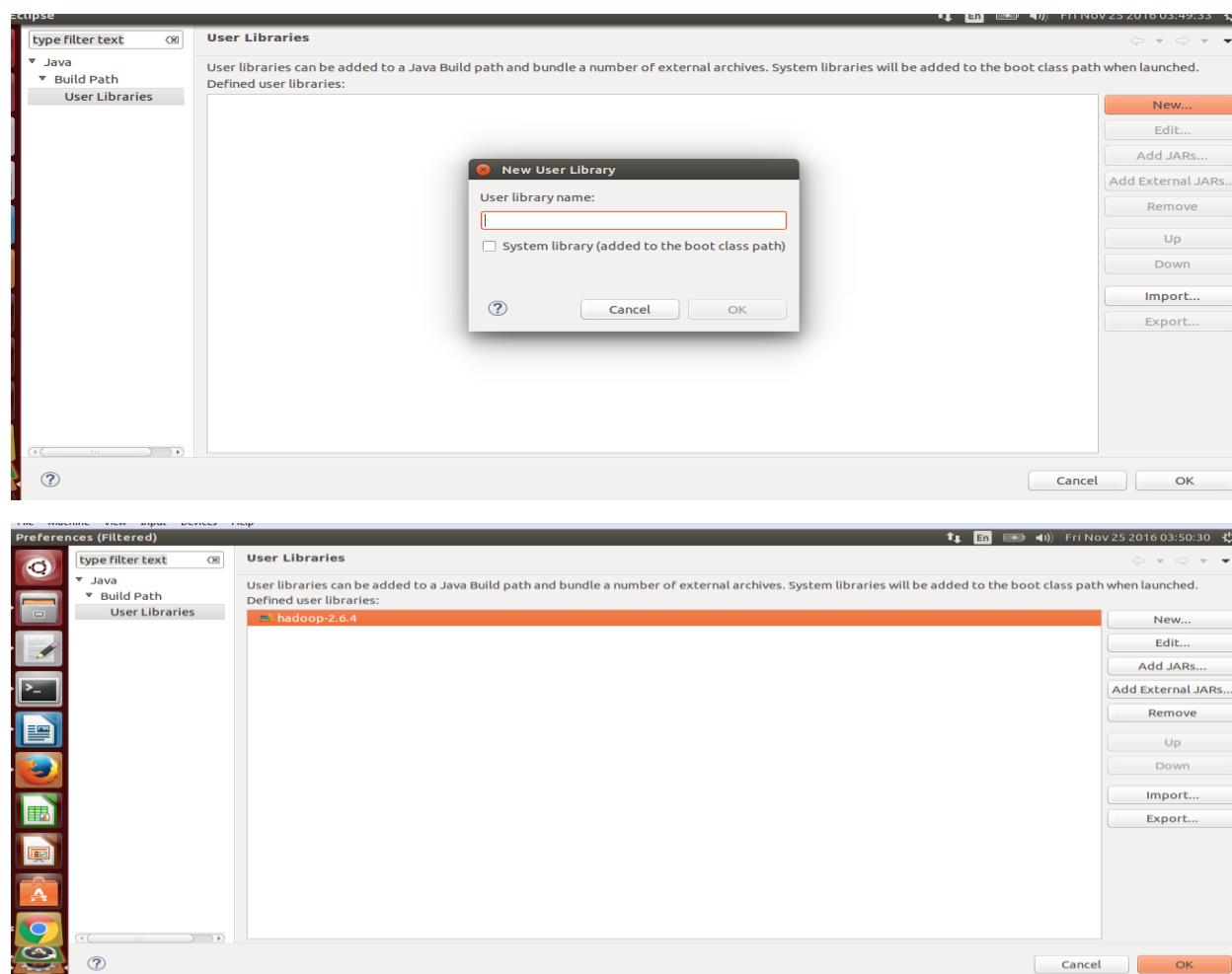
Click on **User-Libraries** from right pane



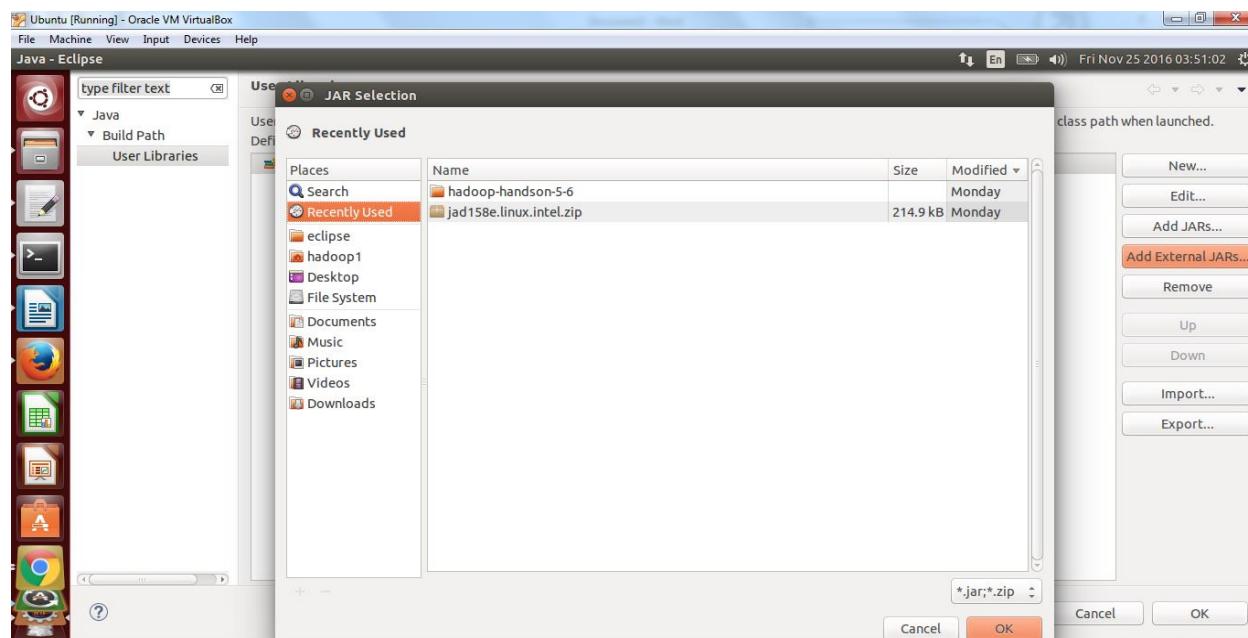
Click on new button from right pane for creating new userlibrary



Give a name to user library and click on **OK**



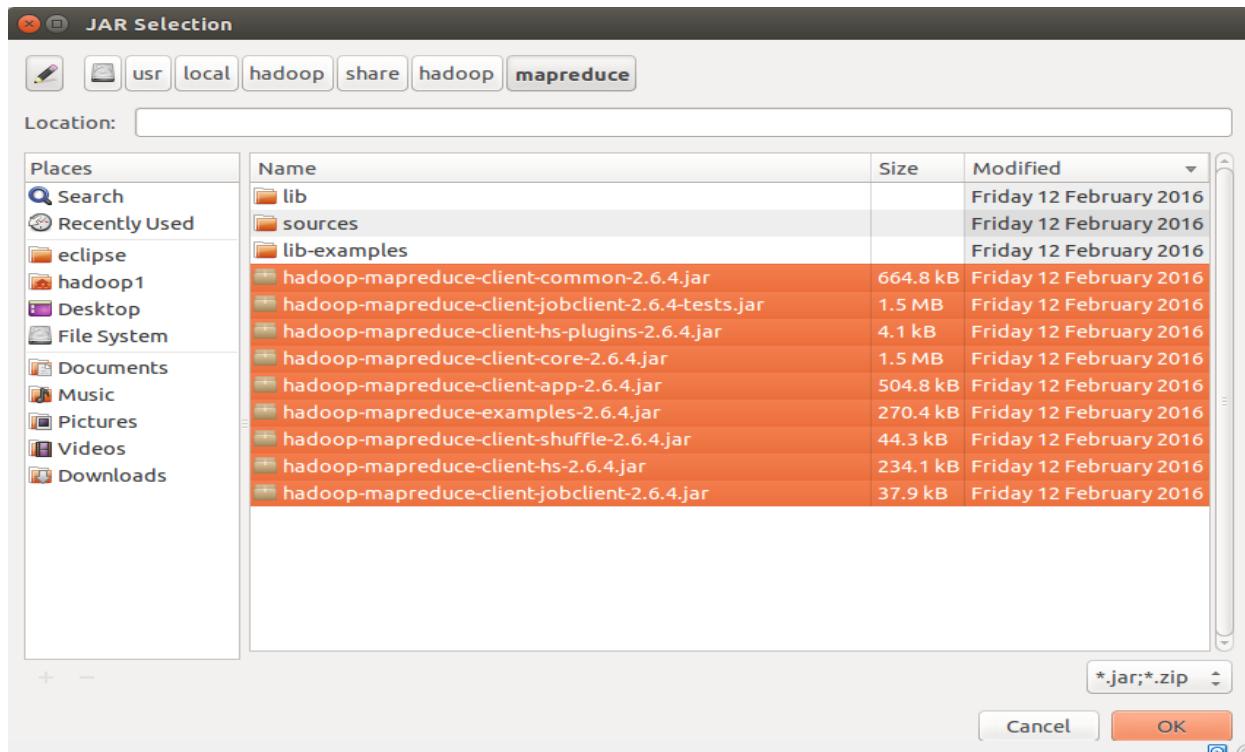
Click on Add external jars and it will display below window, please navigate go to Hadoop home dir from left pane.



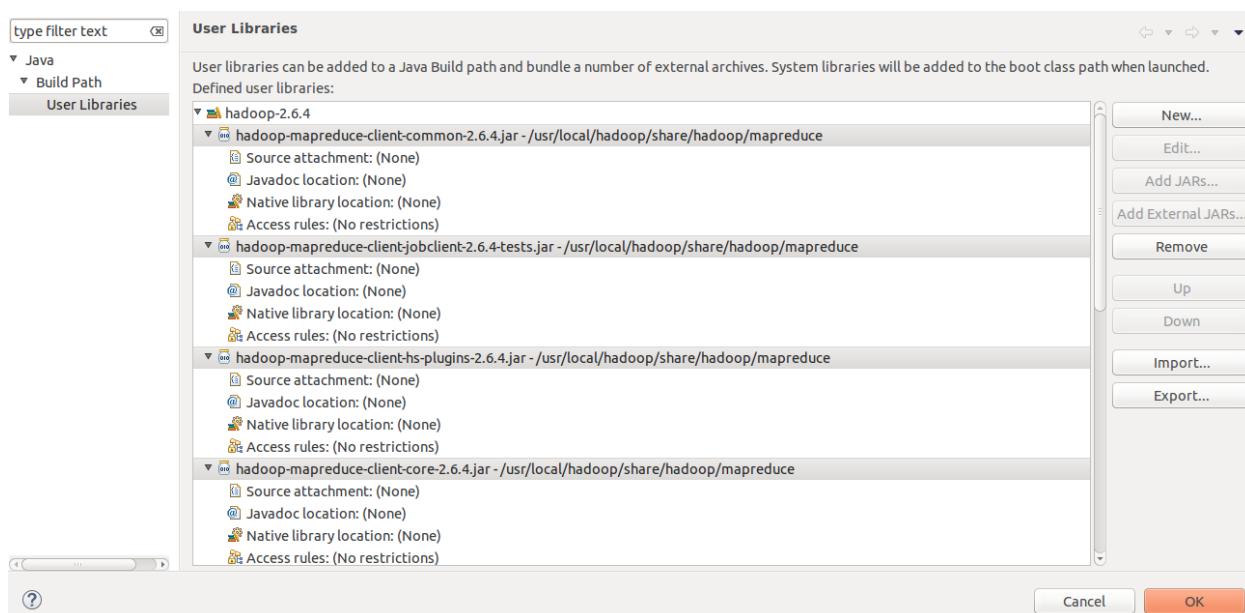
Go to Hadoop home dir (/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/mapreduce) select all the jar..

Note: to run mapreduce, we do not need all the jar files, but it's good to add all the jar file for practice purpose.. But in real-time we will select only required jar file.

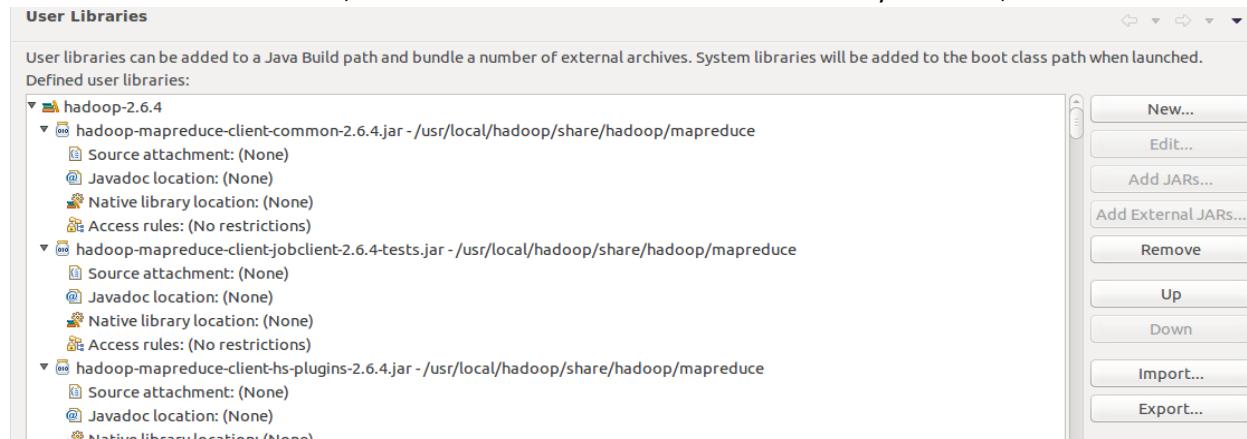
Adding MapReduce jar (/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/mapreduce)



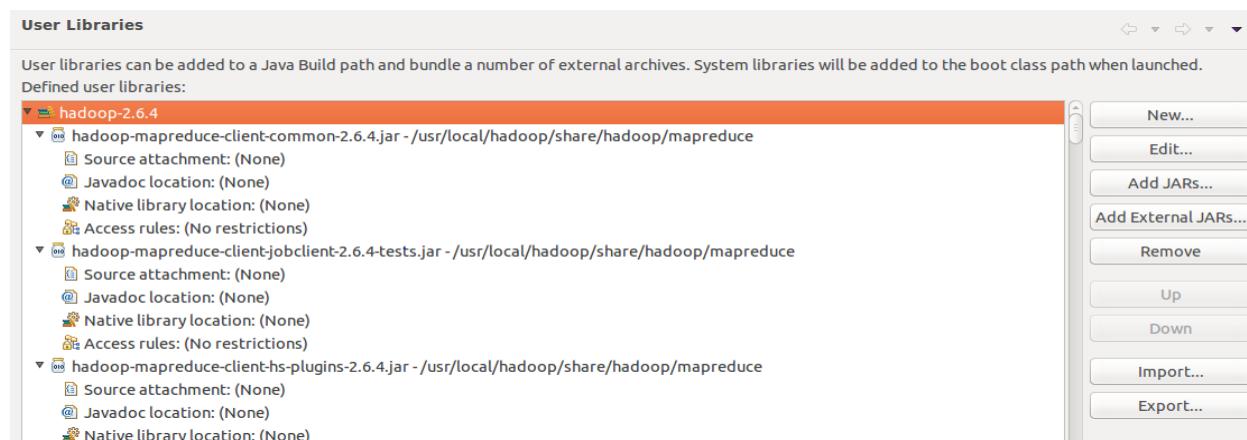
Click ok..



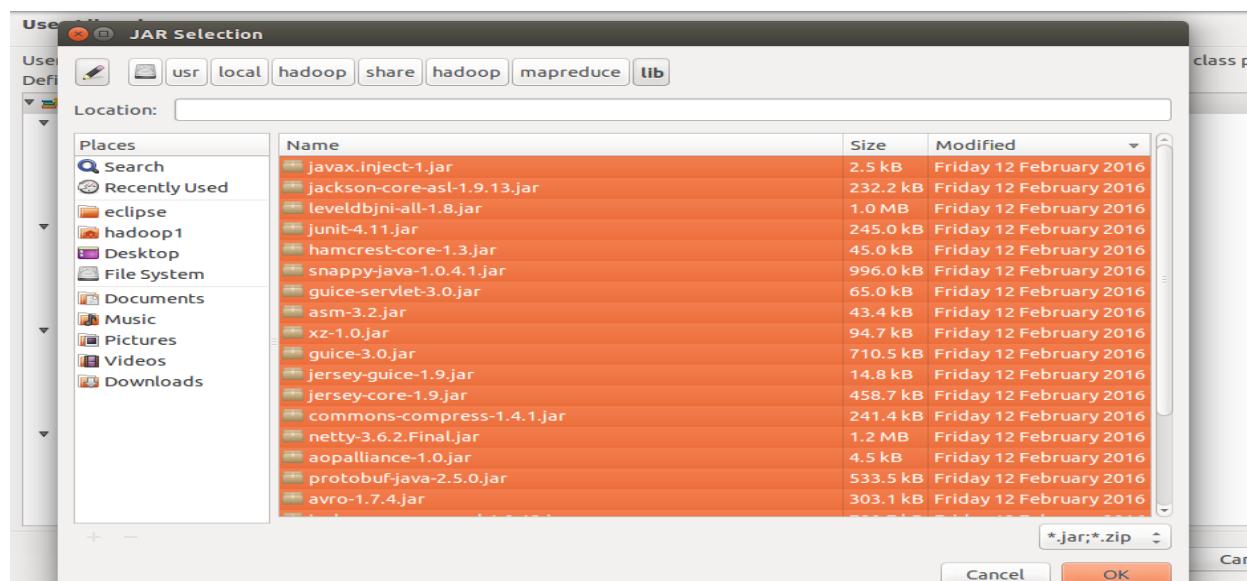
If the add external is disable, it means we did not selected the user library folder so, select created user library.



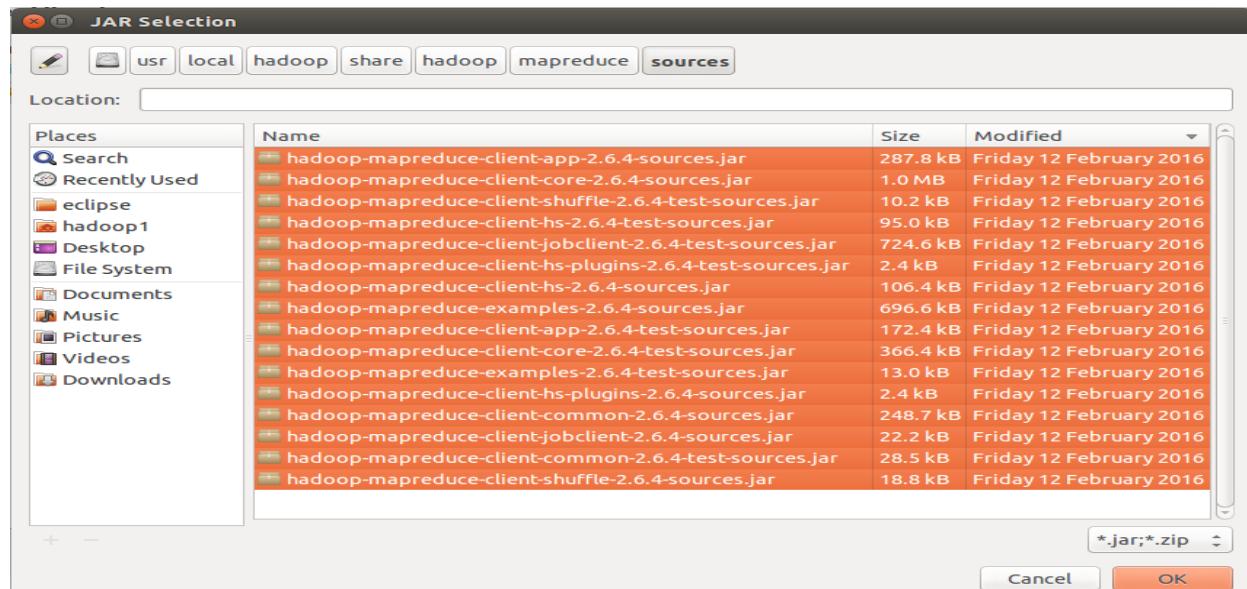
When you select the user library folder, the add external will enabled:



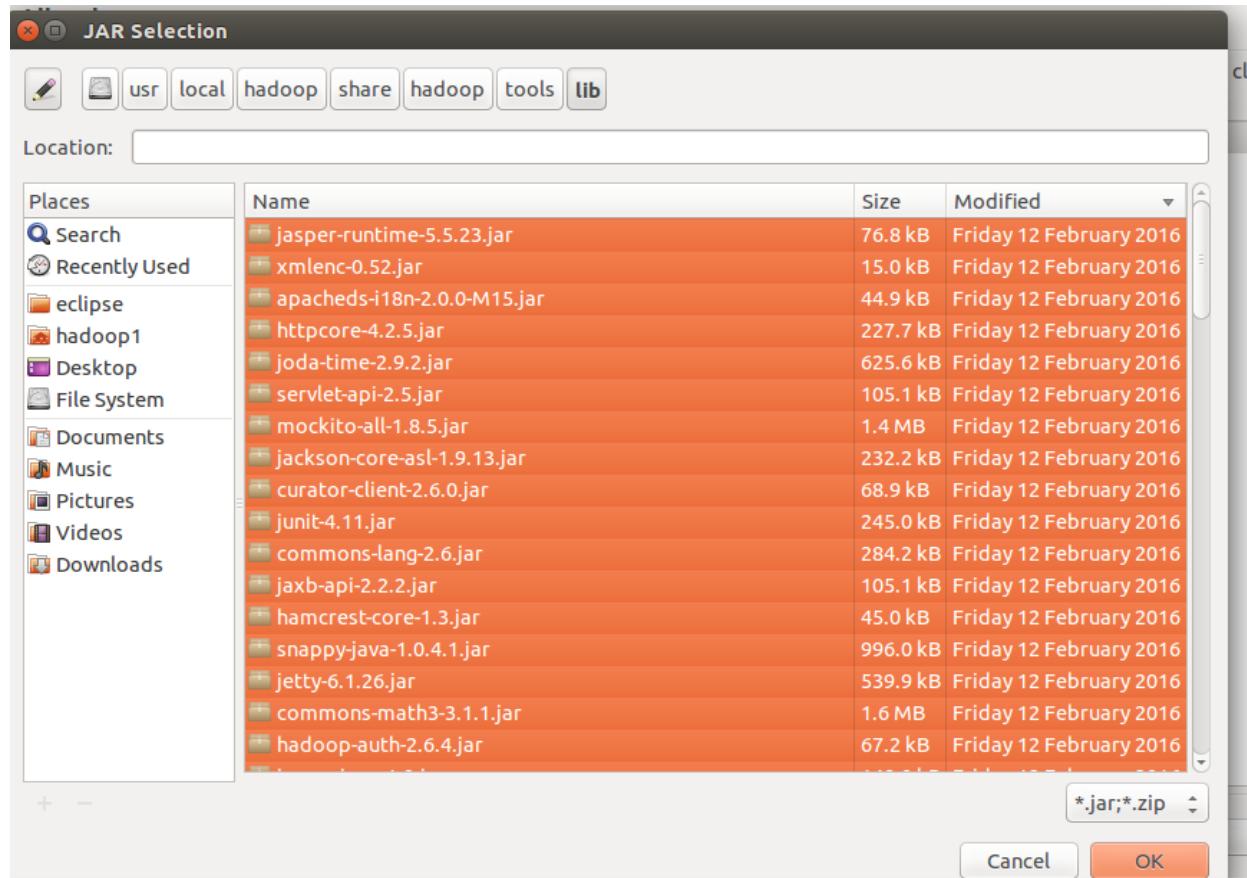
Again need to select few more jar file. So select user library name (should select) and click on add external jars (/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/mapreduce/lib) Click OK



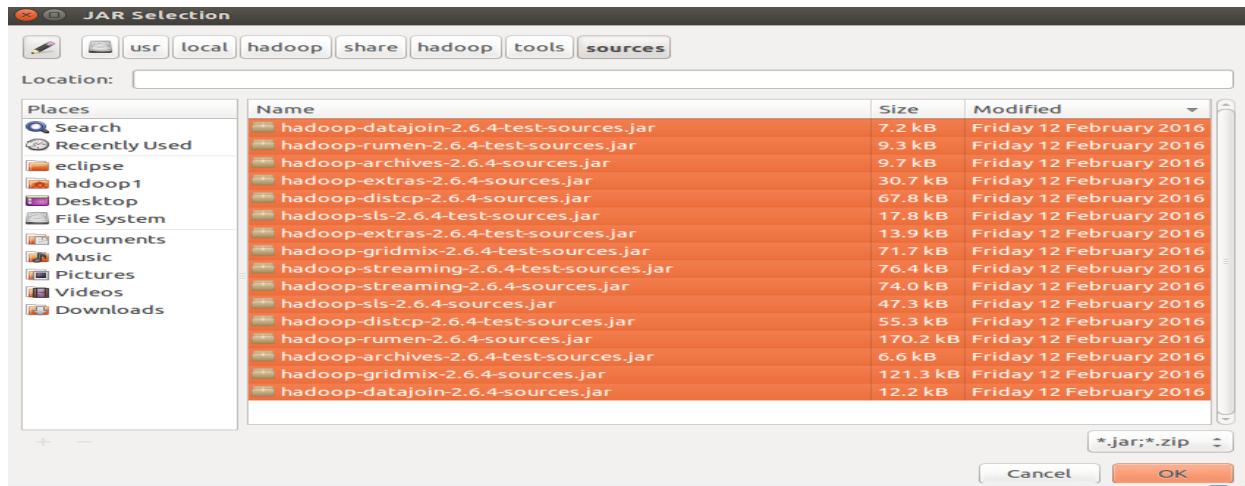
Repeat same step for add all the jars from Hadoop(/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/mapreduce/source) and click OK



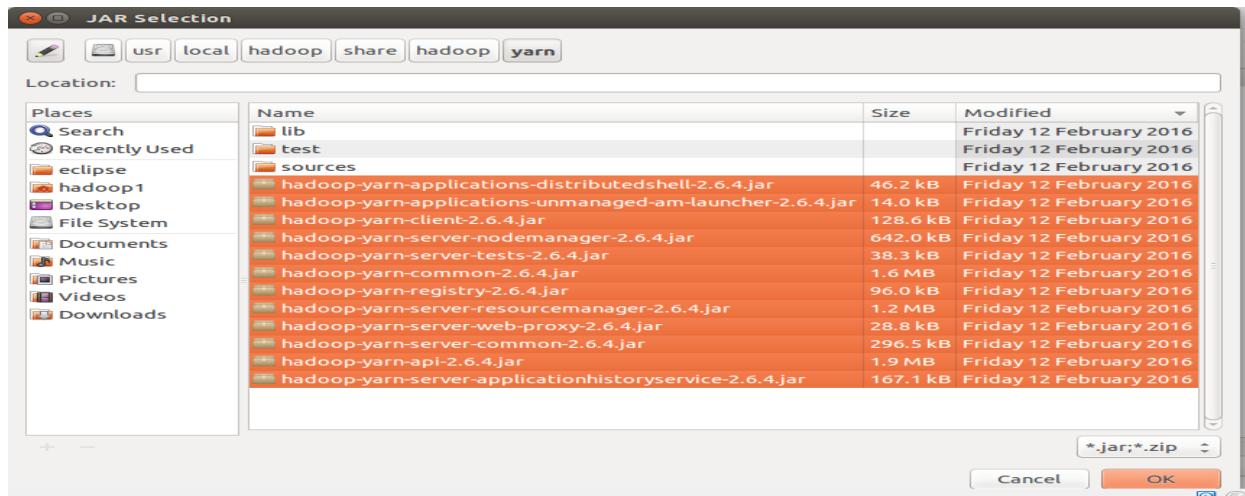
Repeat the same steps for other dirs(/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/tools/lib) click OK



Every time you need to select the user library and add external jars file(/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/sources) Click OK



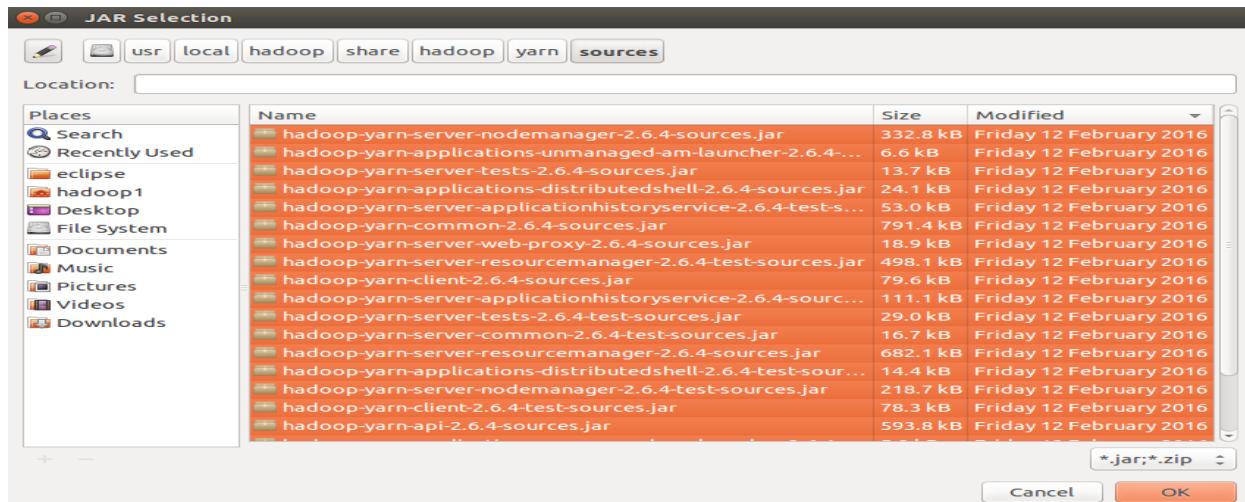
Selecting YARN related jars(/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/yarn) click OK



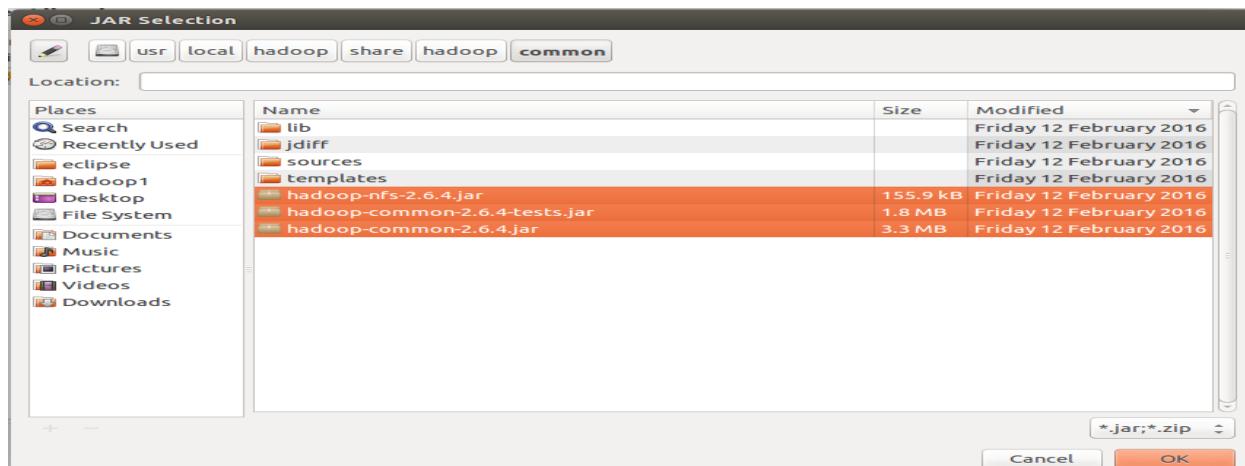
Select jar from Yarn lib folder (/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/yarn/lib). Click OK



Yarn source folder: (/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/yarn/sources).click OK



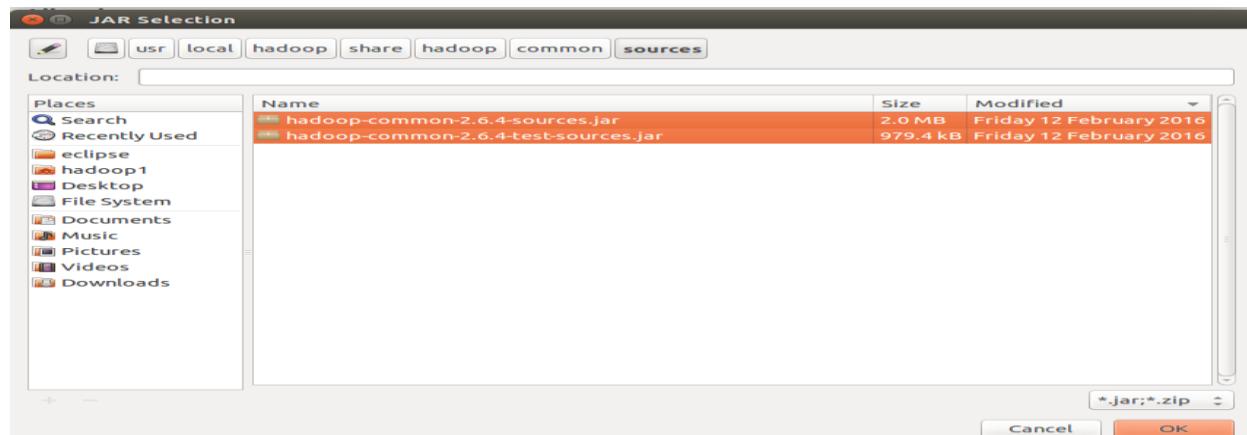
Hadoop-Common folder (/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/Hadoop-common). Click OK



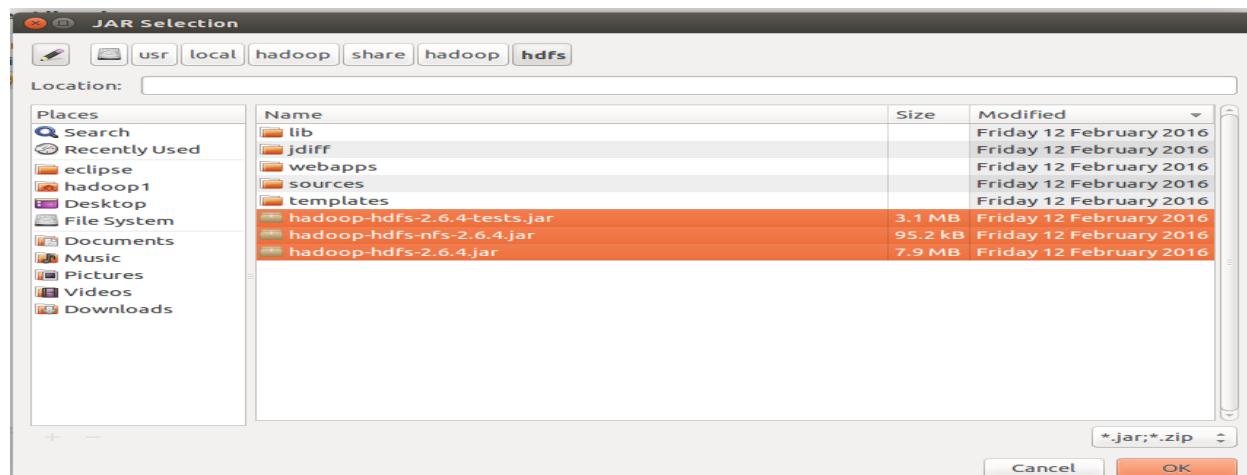
Hadoop-common-lib: (/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/Hadoop-common/lib).click OK



Common-source: (/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/Hadoop-common/sources). Click OK

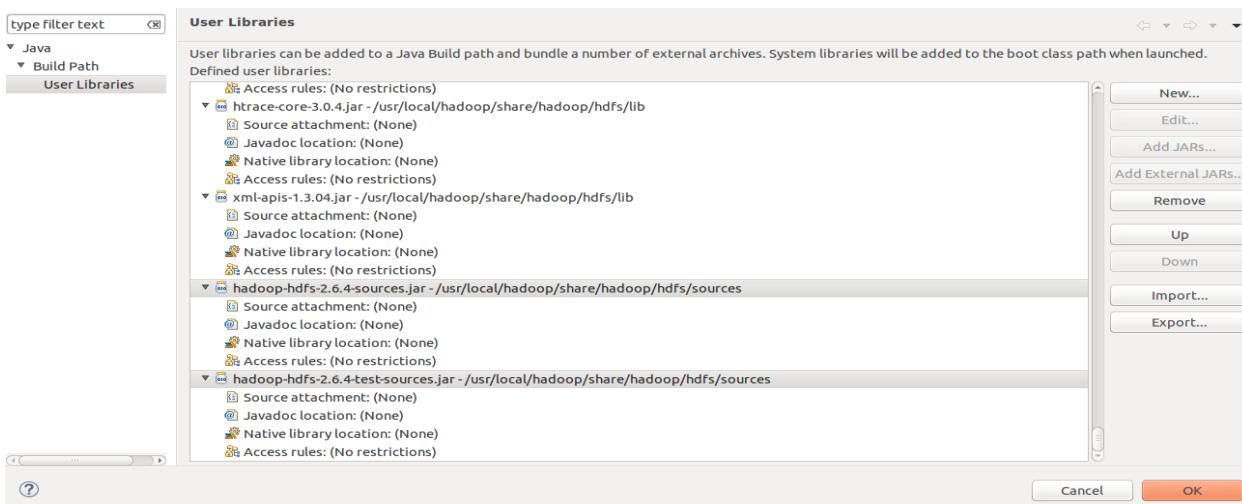


HDFS folder libs(/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/hdfs). Click OK

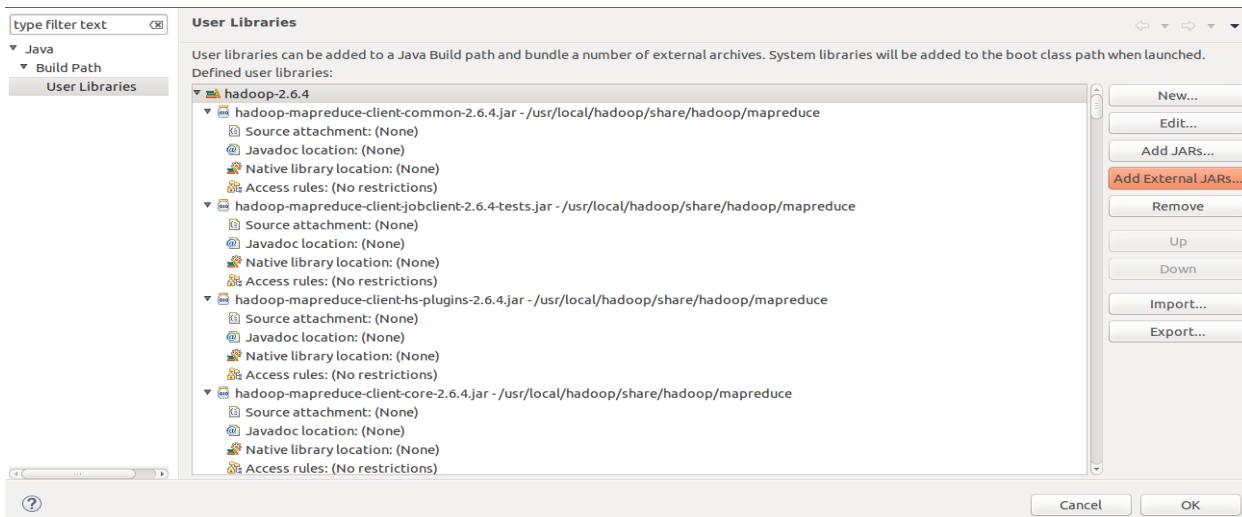


HDFS/lib: (/usr/local/Hadoop-env/Hadoop-2.6.4/share/Hadoop/HDFS/lib). Click OK

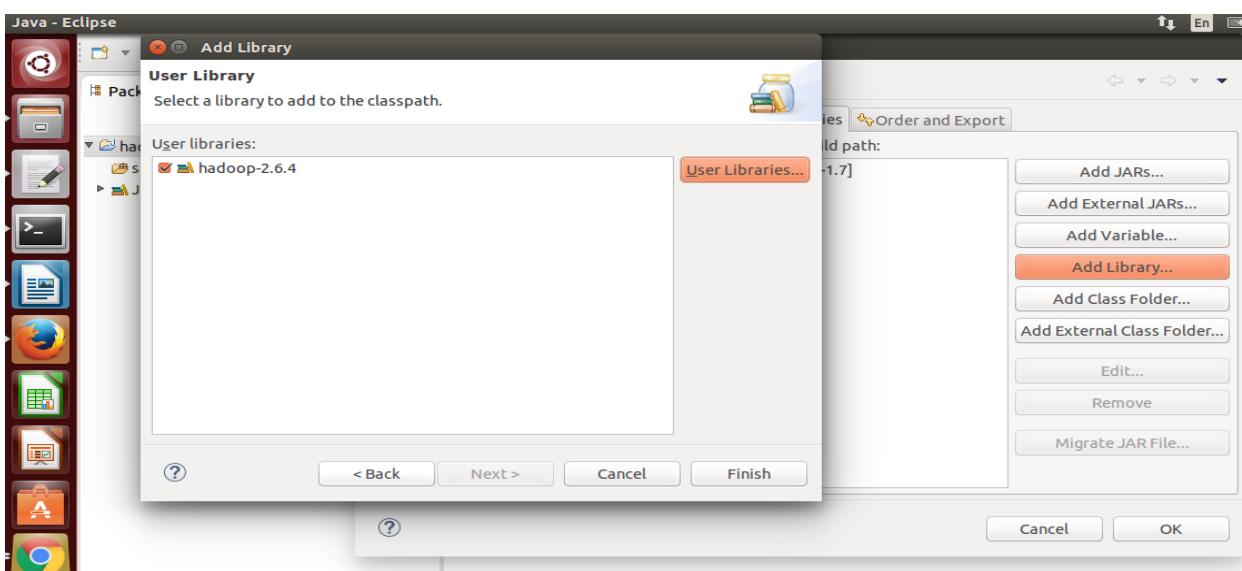




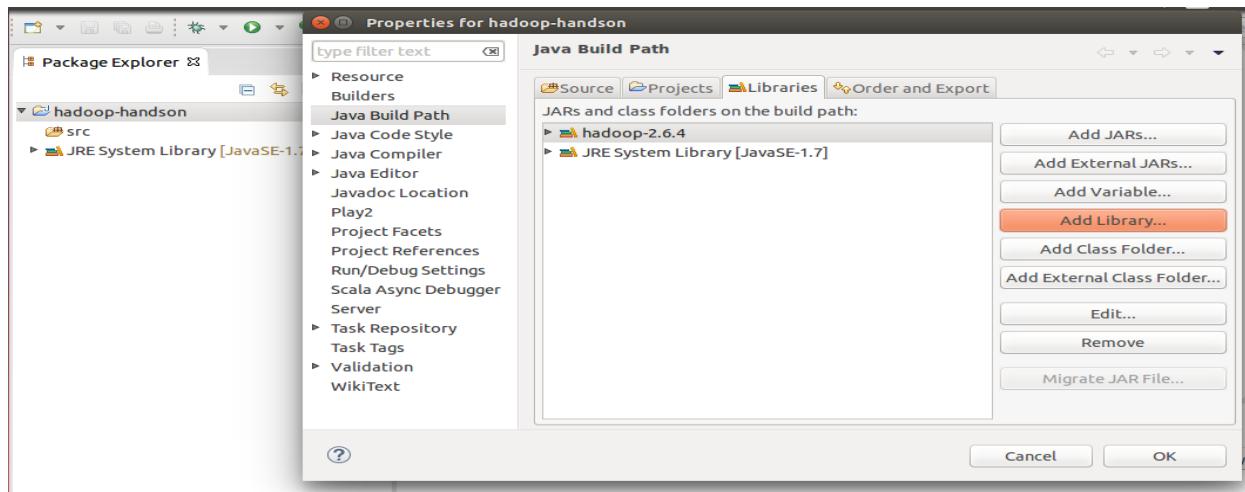
After adding all the libraries or jar files, click on OK



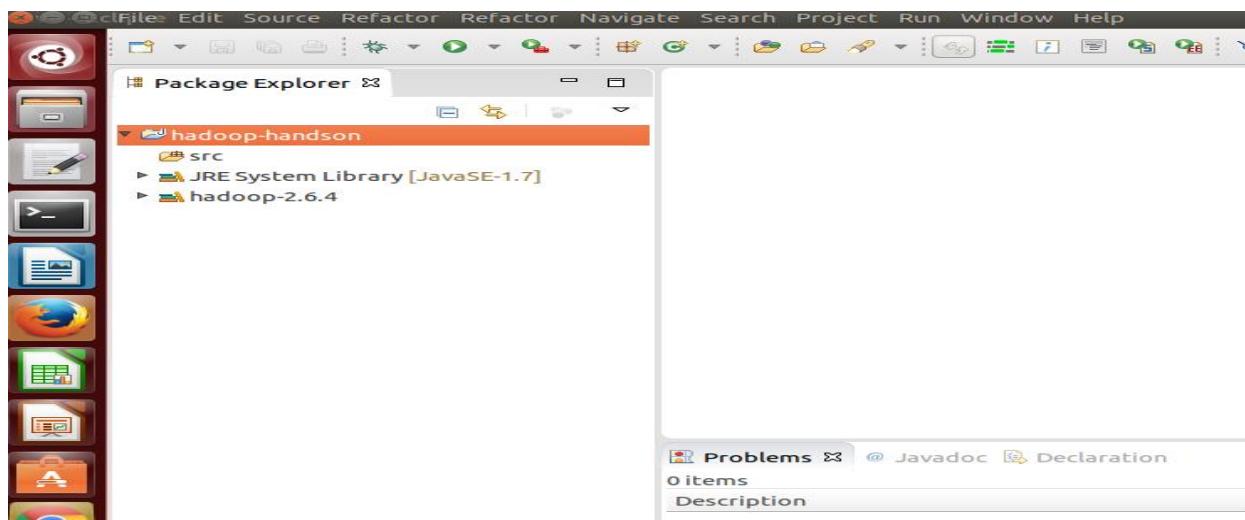
It will create library folder and click finish



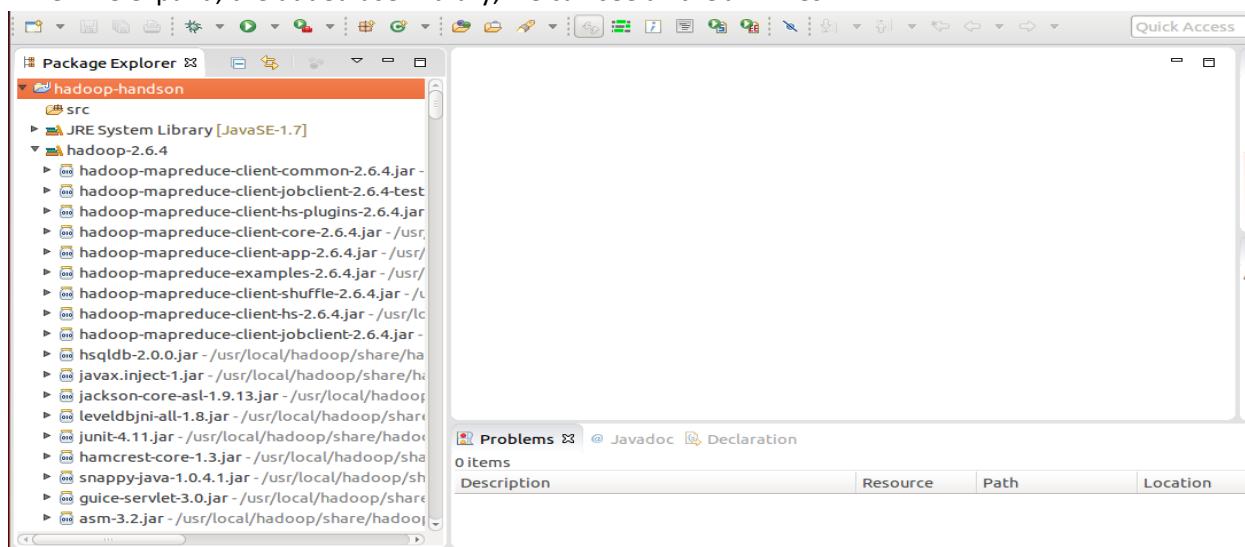
When clicked finish, it is added to Build path libraries and click OK



After clicking OK, it is added to your project in eclipse

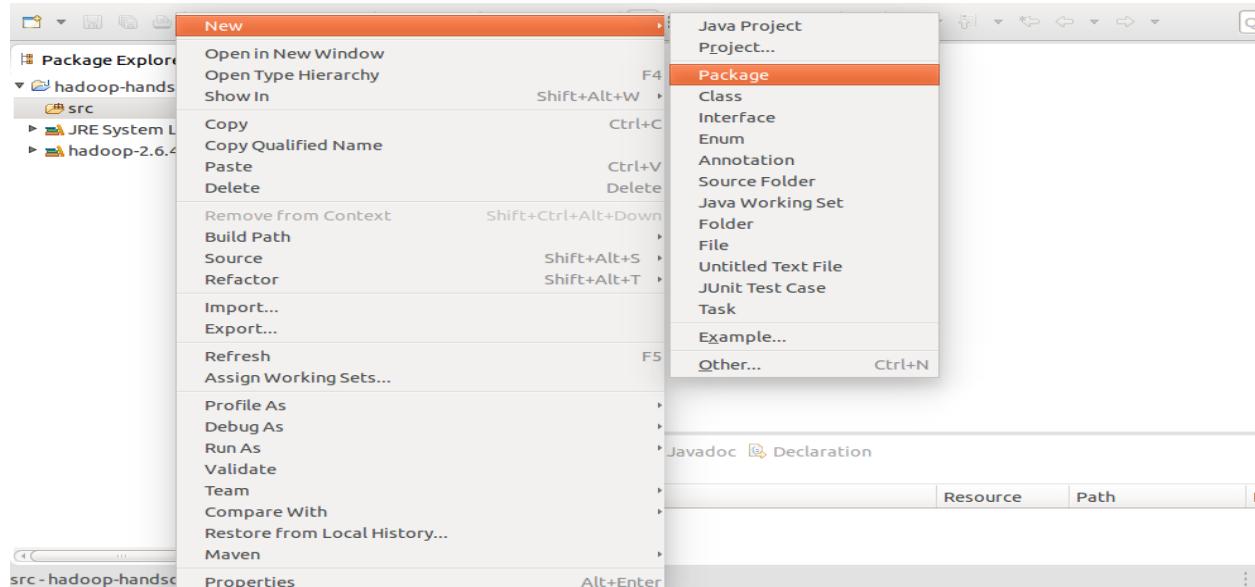


When we expand, the added user library, we can see all the JAR files

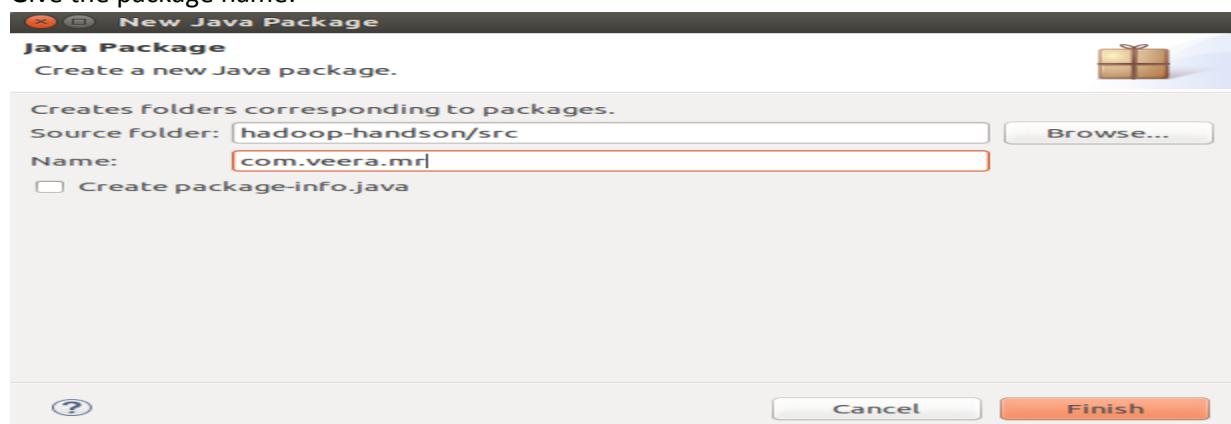


Now your project is ready to code for mapreduce programs.

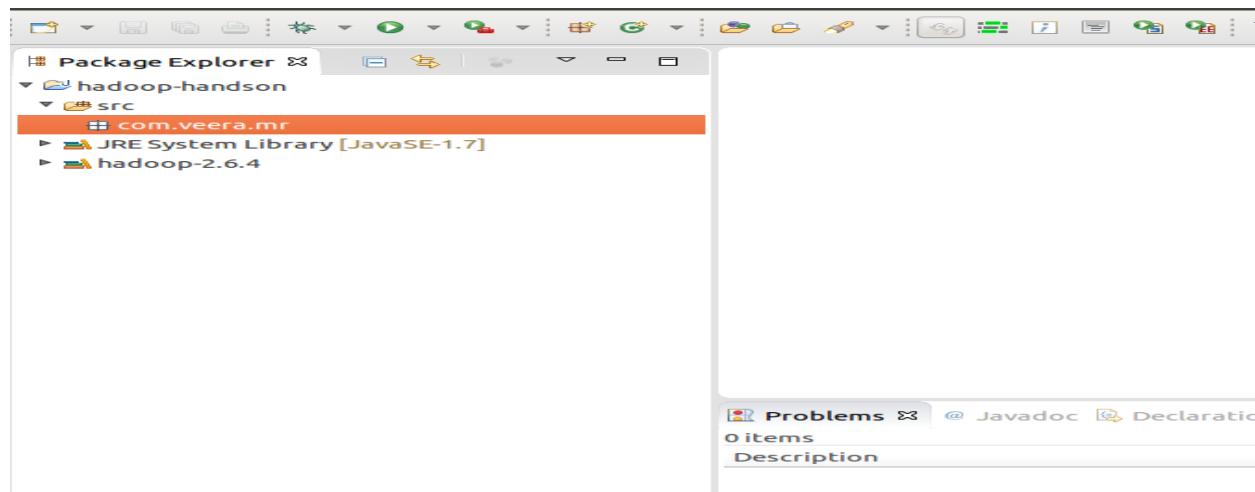
Example: Create package...



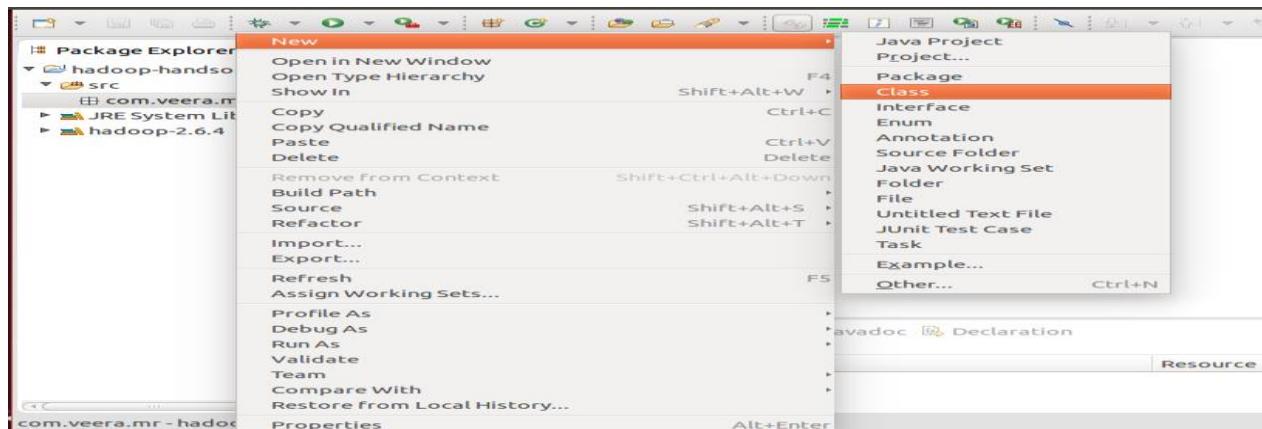
Give the package name:



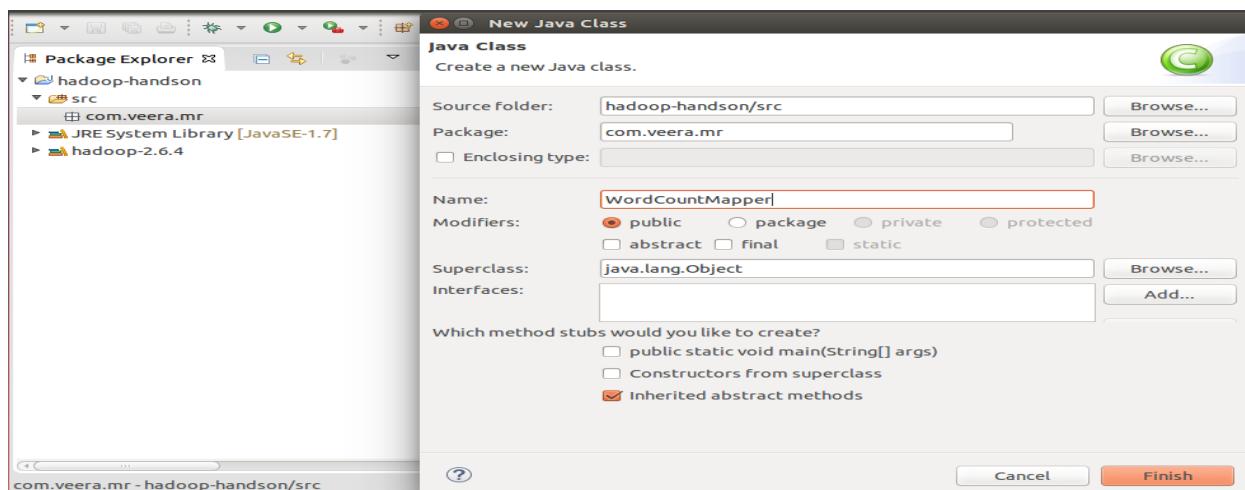
Package is created:



After creating the package, create class for Mapper



Give the class name and click finish



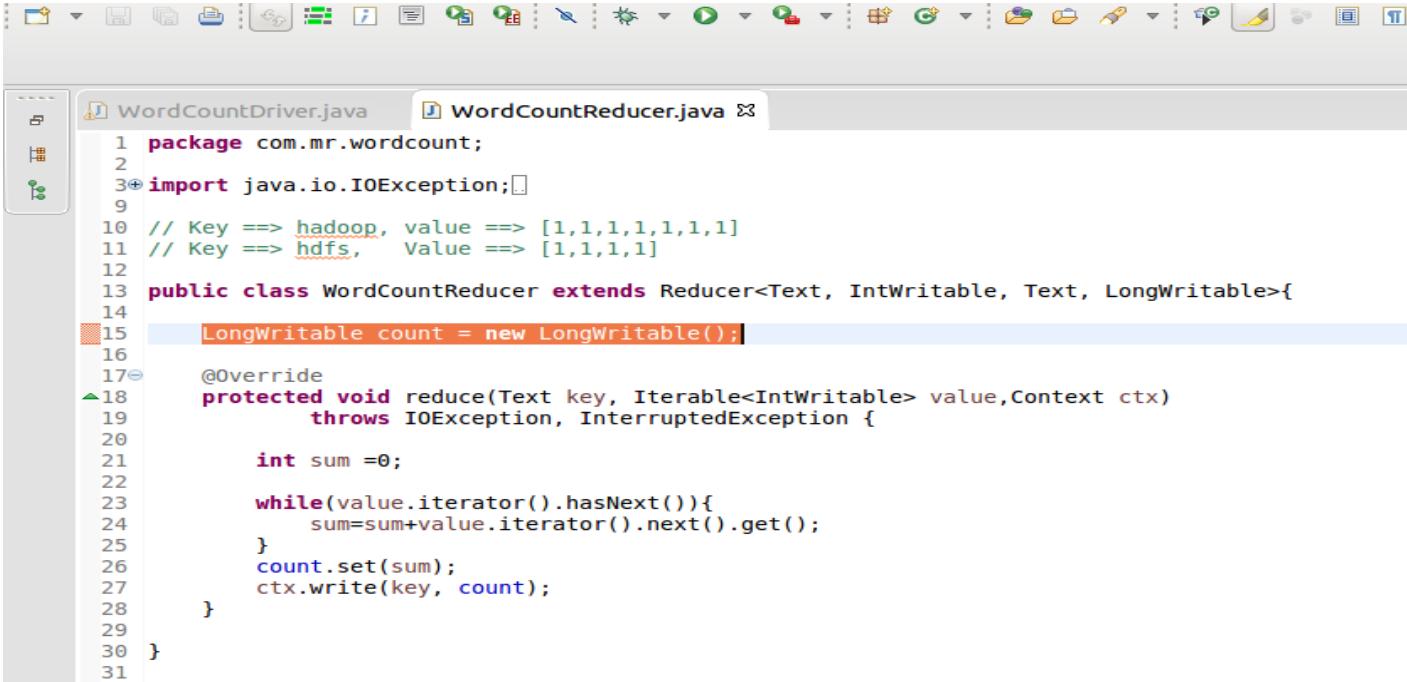
WordCountMapper.java

```

1 package com.veera.mr;
2
3 import java.io.IOException;
4 import java.util.StringTokenizer;
5
6 import org.apache.hadoop.io.LongWritable;
7 import org.apache.hadoop.io.Text;
8 import org.apache.hadoop.mapreduce.Mapper;
9
10 public class WordCountMapper extends Mapper<LongWritable, Text, Text, LongWritable> {
11
12     private Text temp = new Text();
13     private final static LongWritable one = new LongWritable(1);
14
15     @Override
16     protected void map(LongWritable key, Text value, Context context)
17             throws IOException, InterruptedException {
18         String string = value.toString();
19         StringTokenizer strTock = new StringTokenizer(string, " ");
20         while (strTock.hasMoreTokens()) {
21             temp.set(strTock.nextToken());
22             context.write(temp, one);
23         }
24     }
25 }
26

```

WordCountReducer.java

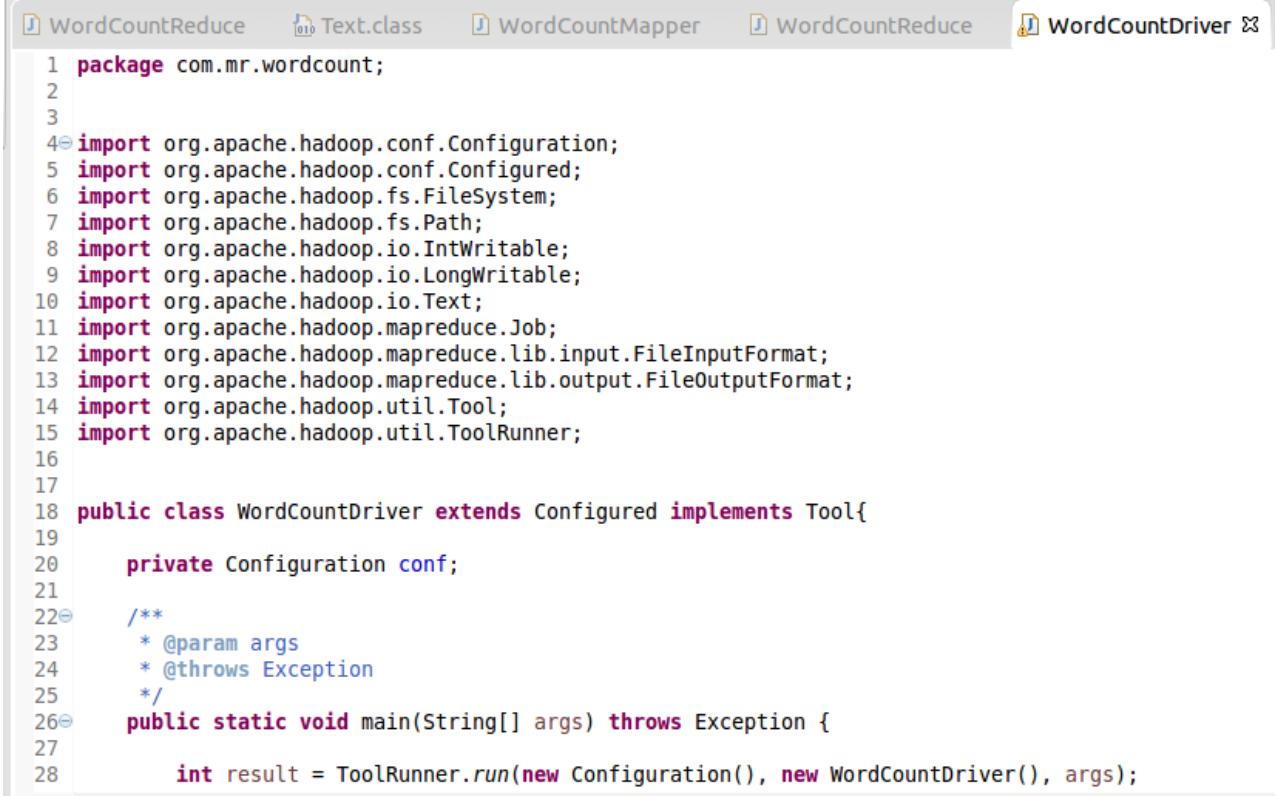


```

1 package com.mr.wordcount;
2
3 import java.io.IOException;
4
5 // Key ==> hadoop, value ==> [1,1,1,1,1,1,1]
6 // Key ==> hdfs,   Value ==> [1,1,1,1]
7
8 public class WordCountReducer extends Reducer<Text, IntWritable, Text, LongWritable>{
9
10     LongWritable count = new LongWritable();
11
12     @Override
13     protected void reduce(Text key, Iterable<IntWritable> value, Context ctx)
14         throws IOException, InterruptedException {
15
16         int sum =0;
17
18         while(value.iterator().hasNext()){
19             sum=sum+value.iterator().next().get();
20         }
21         count.set(sum);
22         ctx.write(key, count);
23     }
24
25 }
26
27
28
29
30 }
31

```

WordCountDriver.java



```

1 package com.mr.wordcount;
2
3
4 import org.apache.hadoop.conf.Configuration;
5 import org.apache.hadoop.conf.Configured;
6 import org.apache.hadoop.fs.FileSystem;
7 import org.apache.hadoop.fs.Path;
8 import org.apache.hadoop.io.IntWritable;
9 import org.apache.hadoop.io.LongWritable;
10 import org.apache.hadoop.io.Text;
11 import org.apache.hadoop.mapreduce.Job;
12 import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
13 import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
14 import org.apache.hadoop.util.Tool;
15 import org.apache.hadoop.util.ToolRunner;
16
17
18 public class WordCountDriver extends Configured implements Tool{
19
20     private Configuration conf;
21
22     /**
23      * @param args
24      * @throws Exception
25      */
26     public static void main(String[] args) throws Exception {
27
28         int result = ToolRunner.run(new Configuration(), new WordCountDriver(), args);
29
30     }
31
32 }
33

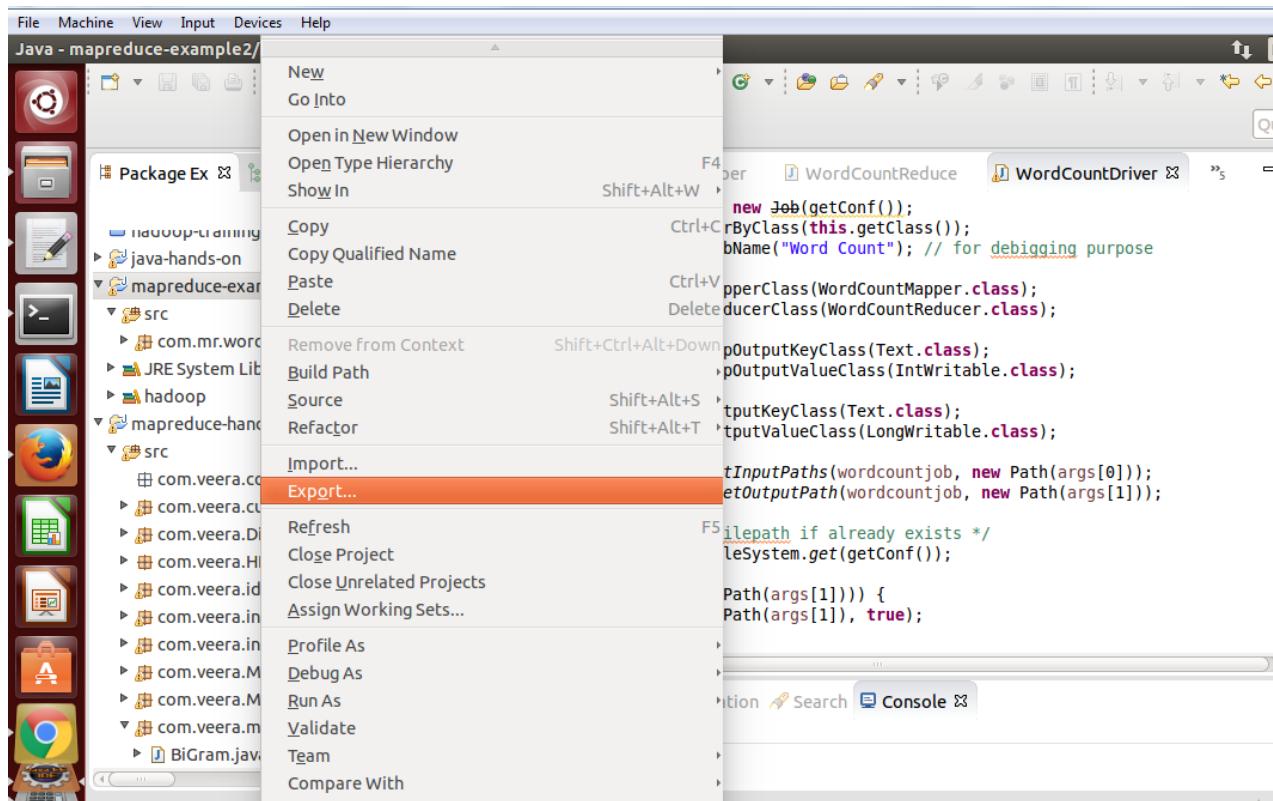
```

```
30     System.out.println("Result of wordcount job " + result);
31
32     if (result == 0) {
33         System.out.println("Result of wordcount job Success");
34     }
35     else {
36         System.out.println("Result of wordcount job Failure");
37     }
38 }
39
40@override
41 public void setConf(Configuration conf) {
42     this.conf = conf;
43 }
44
45@Override
46 public Configuration getConf() {
47     return conf;
48 }
49
50@Override
51 public int run(String[] args) throws Exception {
52
53     Job wordcountjob = new Job(getConf());
54     wordcountjob.setJarByClass(this.getClass());
55     wordcountjob.setJobName("Word Count"); // for debugging purpose
56
57     wordcountjob.setMapperClass(WordCountMapper.class);
58     wordcountjob.setReducerClass(WordCountReducer.class);
59
60     wordcountjob.setMapOutputKeyClass(Text.class);
61     wordcountjob.setMapOutputValueClass(IntWritable.class);
62
63     wordcountjob.setOutputKeyClass(Text.class);
64     wordcountjob.setOutputValueClass(LongWritable.class);
65
66     FileInputFormat.setInputPaths(wordcountjob, new Path(args[0]));
67     FileOutputFormat.setOutputPath(wordcountjob, new Path(args[1]));
68
69     /* Delete output filepath if already exists */
70     FileSystem fs = FileSystem.get(getConf());
71
72     if (fs.exists(new Path(args[1]))) {
73         fs.delete(new Path(args[1]), true);
74     }
75
76     return wordcountjob.waitForCompletion(true) == true ? 0 : 1;
77 }
78
79 }
80 }
```

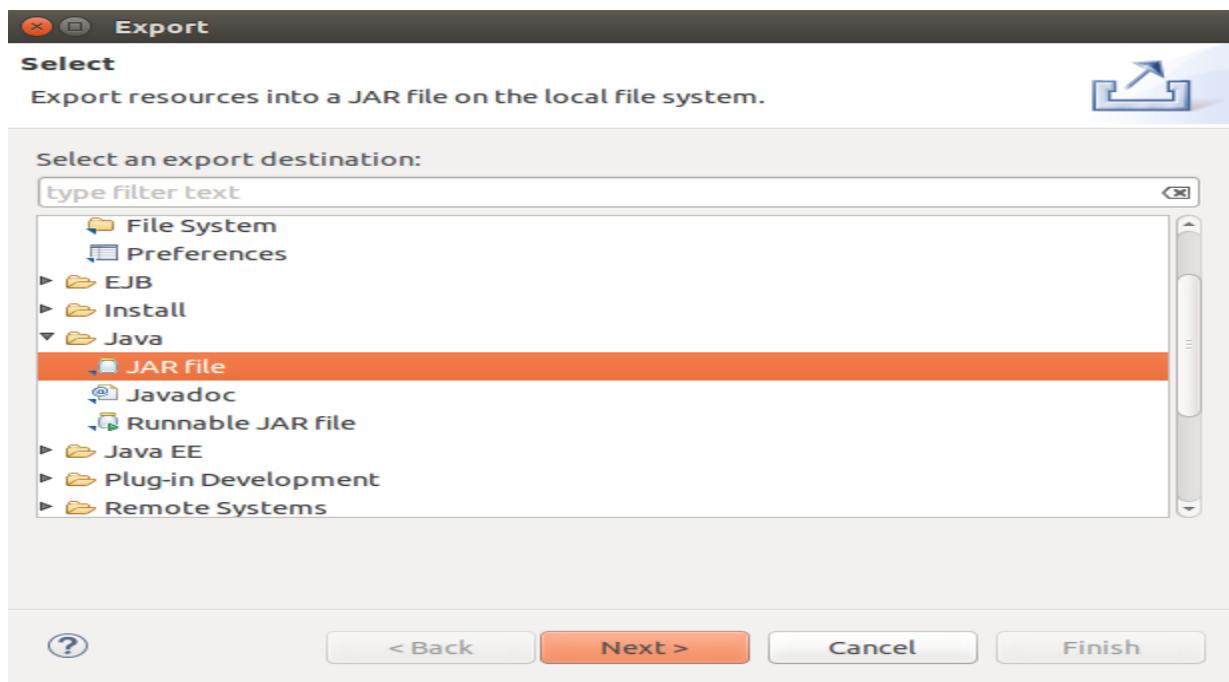
Steps to Run in MR-mode:

- 1) Create JAR.

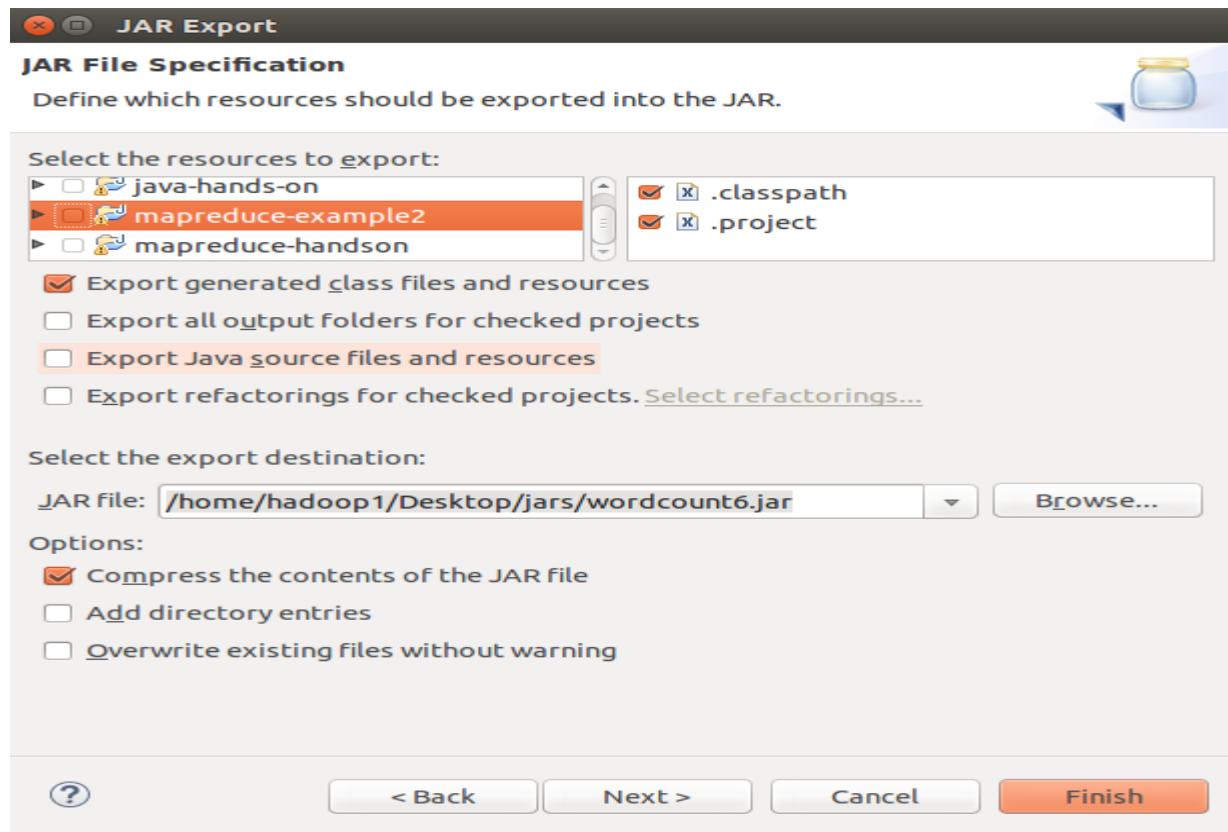
Right click on project (or package or select the files) → select **Export**



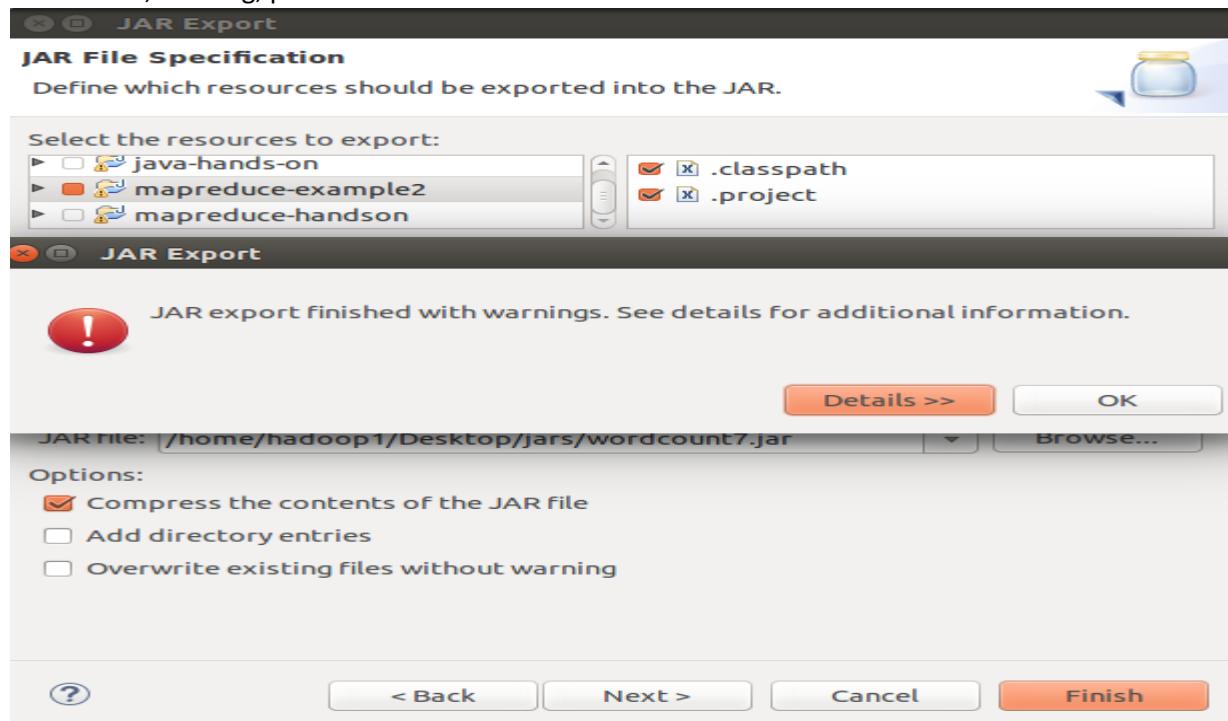
Select JAR from java and click on **Next** button



Select path to save JAR path and give name(/home/hadoop1/Desktop/jars/wordcount6.jar) and click on **Finish**: follow below screen



It will show, warning, please click on OK



- 2) Start all the Hadoop services.
 - 3) Execute below command to launch MR job.

syntax: `hadoop jar <path of the saved jar> <fully qualified driver class name> <hdfs input path> <hdfs output path> ...`

<hdfs input path> : Input file location in HDFS

<hdfs output path> : non existing(MR frame work will create dir in spciified path) output dir location in HDFS

ex: hadoop jar /home/hadoop1/Desktop/jars/wordcount6.jar com.mr.wordcount.wordCountDriver /veera/biggerFile.txt /veera/wc1

```
hadoop1@ubuntu:~$  
hadoop1@ubuntu:~$  
hadoop1@ubuntu:~$ hadoop jar /home/hadoop1/Desktop/jars/wordcount6.jar com.mr.wordcount.WordCountDriver /veera/biggerfile2.txt /veera/wc1
```

```
hadoop1@ubuntu:~$  
hadoop1@ubuntu:~$  
hadoop1@ubuntu:~$ hadoop jar /home/hadoop1/Desktop/jars/wordcount6.jar com.mr.wordcount.WordCountDriver /veera/biggerfile2.txt /veera/wc1  
16/11/30 11:51:59 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable  
16/11/30 11:52:00 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032  
16/11/30 11:52:00 INFO input.FileInputFormat: Total input paths to process : 1  
16/11/30 11:52:00 INFO mapreduce.JobSubmitter: number of splits:9  
16/11/30 11:52:01 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1480468065521_0007  
16/11/30 11:52:01 INFO impl.YarnClientImpl: Submitted application application_1480468065521_0007  
16/11/30 11:52:01 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1480468065521_0007/  
16/11/30 11:52:01 INFO mapreduce.Job: Running job: job_1480468065521_0007  
16/11/30 11:52:09 INFO mapreduce.Job: Job job_1480468065521_0007 running in uber mode : false  
16/11/30 11:52:09 INFO mapreduce.Job: map 0% reduce 0%  
16/11/30 11:52:44 INFO mapreduce.Job: map 2% reduce 0%
```

```
hadoop1@ubuntu: ~          x  hadoop1@ubuntu:  
16/11/27 05:16:00 INFO mapreduce.Job: map 60% reduce 0%  
16/11/27 05:16:03 INFO mapreduce.Job: map 61% reduce 0%  
16/11/27 05:16:07 INFO mapreduce.Job: map 63% reduce 0%  
16/11/27 05:16:11 INFO mapreduce.Job: map 66% reduce 0%  
16/11/27 05:16:15 INFO mapreduce.Job: map 67% reduce 0%  
16/11/27 05:16:53 INFO mapreduce.Job: map 67% reduce 11%  
16/11/27 05:16:57 INFO mapreduce.Job: map 67% reduce 19%  
16/11/27 05:17:00 INFO mapreduce.Job: map 67% reduce 22%  
16/11/27 05:17:02 INFO mapreduce.Job: map 68% reduce 22%  
16/11/27 05:17:07 INFO mapreduce.Job: map 69% reduce 22%  
16/11/27 05:17:10 INFO mapreduce.Job: map 70% reduce 22%  
16/11/27 05:17:14 INFO mapreduce.Job: map 71% reduce 22%  
16/11/27 05:17:21 INFO mapreduce.Job: map 72% reduce 22%  
16/11/27 05:17:25 INFO mapreduce.Job: map 73% reduce 22%  
16/11/27 05:17:31 INFO mapreduce.Job: map 74% reduce 22%  
16/11/27 05:17:35 INFO mapreduce.Job: map 75% reduce 22%  
16/11/27 05:17:38 INFO mapreduce.Job: map 76% reduce 22%  
16/11/27 05:17:44 INFO mapreduce.Job: map 77% reduce 22%  
16/11/27 05:17:48 INFO mapreduce.Job: map 78% reduce 22%  
16/11/27 05:17:51 INFO mapreduce.Job: map 79% reduce 22%  
16/11/27 05:17:57 INFO mapreduce.Job: map 80% reduce 22%  
16/11/27 05:18:01 INFO mapreduce.Job: map 81% reduce 22%  
16/11/27 05:18:04 INFO mapreduce.Job: map 82% reduce 22%  
16/11/27 05:18:14 INFO mapreduce.Job: map 83% reduce 22%  
16/11/27 05:18:21 INFO mapreduce.Job: map 84% reduce 22%  
16/11/27 05:18:28 INFO mapreduce.Job: map 88% reduce 22%  
16/11/27 05:18:31 INFO mapreduce.Job: map 88% reduce 26%  
16/11/27 05:18:40 INFO mapreduce.Job: map 89% reduce 26%  
16/11/27 05:18:53 INFO mapreduce.Job: map 90% reduce 26%  
16/11/27 05:19:06 INFO mapreduce.Job: map 91% reduce 26%  
16/11/27 05:19:16 INFO mapreduce.Job: map 92% reduce 26%  
16/11/27 05:19:22 INFO mapreduce.Job: map 93% reduce 26%
```

Job finished: see the counter:

```

hadoop1@ubuntu:~          x | hadoop1@ubuntu:~
16/11/27 05:20:24 INFO mapreduce.Job: Job job_1480223274108_0001 completed successfully
16/11/27 05:20:24 INFO mapreduce.Job: Counters: 50
  File System Counters
    FILE: Number of bytes read=1008070035
    FILE: Number of bytes written=1529373125
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=286296619
    HDFS: Number of bytes written=43776
    HDFS: Number of read operations=30
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=2
  Job Counters
    Killed map tasks=3
    Launched map tasks=11
    Launched reduce tasks=1
    Data-local map tasks=11
    Total time spent by all maps in occupied slots (ms)=2500254
    Total time spent by all reduces in occupied slots (ms)=245654
    Total time spent by all map tasks (ms)=2500254
    Total time spent by all reduce tasks (ms)=245654
    Total vcore-milliseconds taken by all map tasks=2500254
    Total vcore-milliseconds taken by all reduce tasks=245654
    Total megabyte-milliseconds taken by all map tasks=2560260096
    Total megabyte-milliseconds taken by all reduce tasks=251549696
  Map -Reduce Framework
    Map input records=4557361
    Map output records=40520729
    Map output bytes=439189453
    Map output materialized bytes=520230965
    Input split bytes=972
    Combine input records=0
  
```

```

hadoop1@ubuntu:~          x | hadoop1@ubuntu:~
  Map input records=4557361
  Map output records=40520729
  Map output bytes=439189453
  Map output materialized bytes=520230965
  Input split bytes=972
  Combine input records=0
  Combine output records=0
  Reduce input groups=3378
  Reduce shuffle bytes=520230965
  Reduce input records=40520729
  Reduce output records=3378
  Spilled Records=119039344
  Shuffled Maps =9
  Failed Shuffles=0
  Merged Map outputs=9
  GC time elapsed (ms)=7654
  CPU time spent (ms)=296400
  Physical memory (bytes) snapshot=2328567808
  Virtual memory (bytes) snapshot=8018878464
  Total committed heap usage (bytes)=1739718656
  Shuffle Errors
    BAD_ID=0
    CONNECTION=0
    IO_ERROR=0
    WRONG_LENGTH=0
    WRONG_MAP=0
    WRONG_REDUCE=0
  File Input Format Counters
    Bytes Read=286295647
  File Output Format Counters
    Bytes Written=43776
Result of wordcount job 0
Result of wordcount job Success
  
```

- 4) Go to HDFS output dir path, it will contain, 2 files **_SUCCESS** and **part-r-00000**. The **part-r-00000** is final output file.

```

hadoop1@ubuntu:~$ 
hadoop1@ubuntu:~$ 
hadoop1@ubuntu:~$ 
hadoop1@ubuntu:~$ hadoop fs -ls /veera/wc1
16/11/30 12:33:15 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform
        Found 2 items
-rw-r--r--  1 hadoop1 supergroup      0 2016-11-30 11:56 /veera/wc1/_SUCCESS
-rw-r--r--  1 hadoop1 supergroup  35751 2016-11-30 11:56 /veera/wc1/part-r-00000
hadoop1@ubuntu:~$ 
  
```

- 5) View the output of mapreduce job.

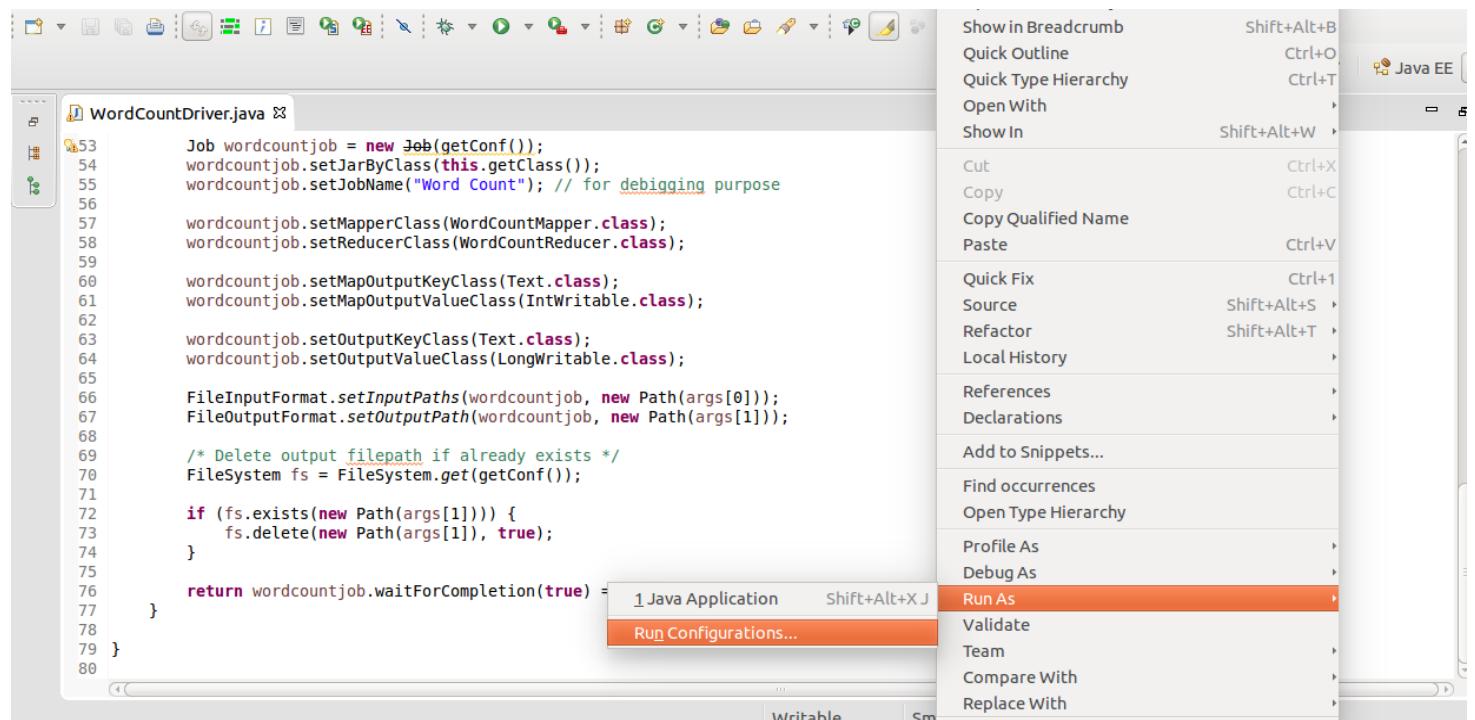
```

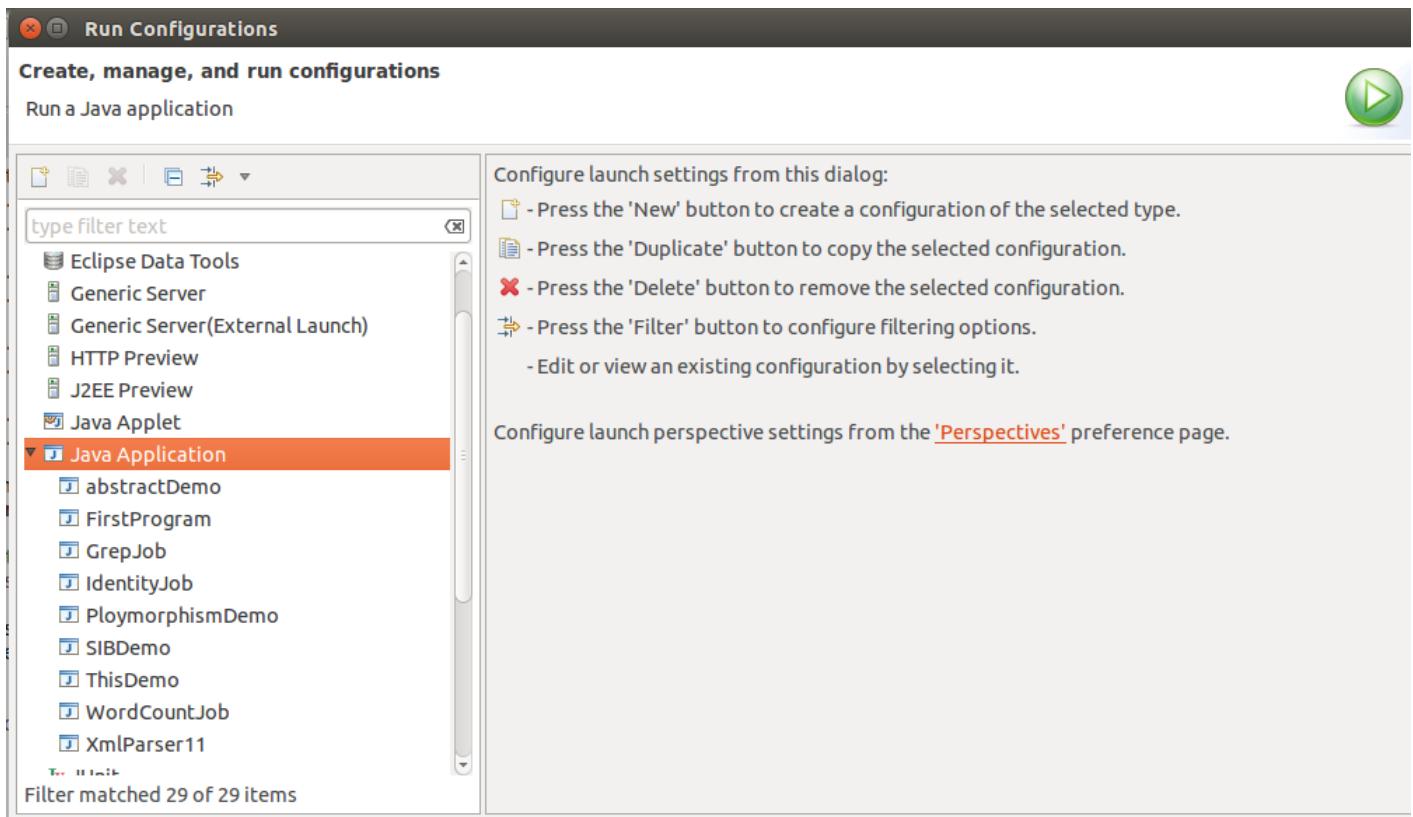
hadoop1@ubuntu:~$ hadoop fs -cat /veera/wc1/part-r-00000 | tail -20
16/11/30 12:50:38 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform
able
zero      3
zip       1
{-m|-x}  2
{-n}      4
|        20177
Čeština  6720
Монгол  6720
Русский 6720
Српски  6720
Українська 6720
עברית  6720
ไทย     6720
فارسی   6720
தமிழ்  6720
മലയුලු 6720
-        107520
-        40320
中文     6720
日本語   6720
한국어   6720
hadoop1@ubuntu:~$ 

```

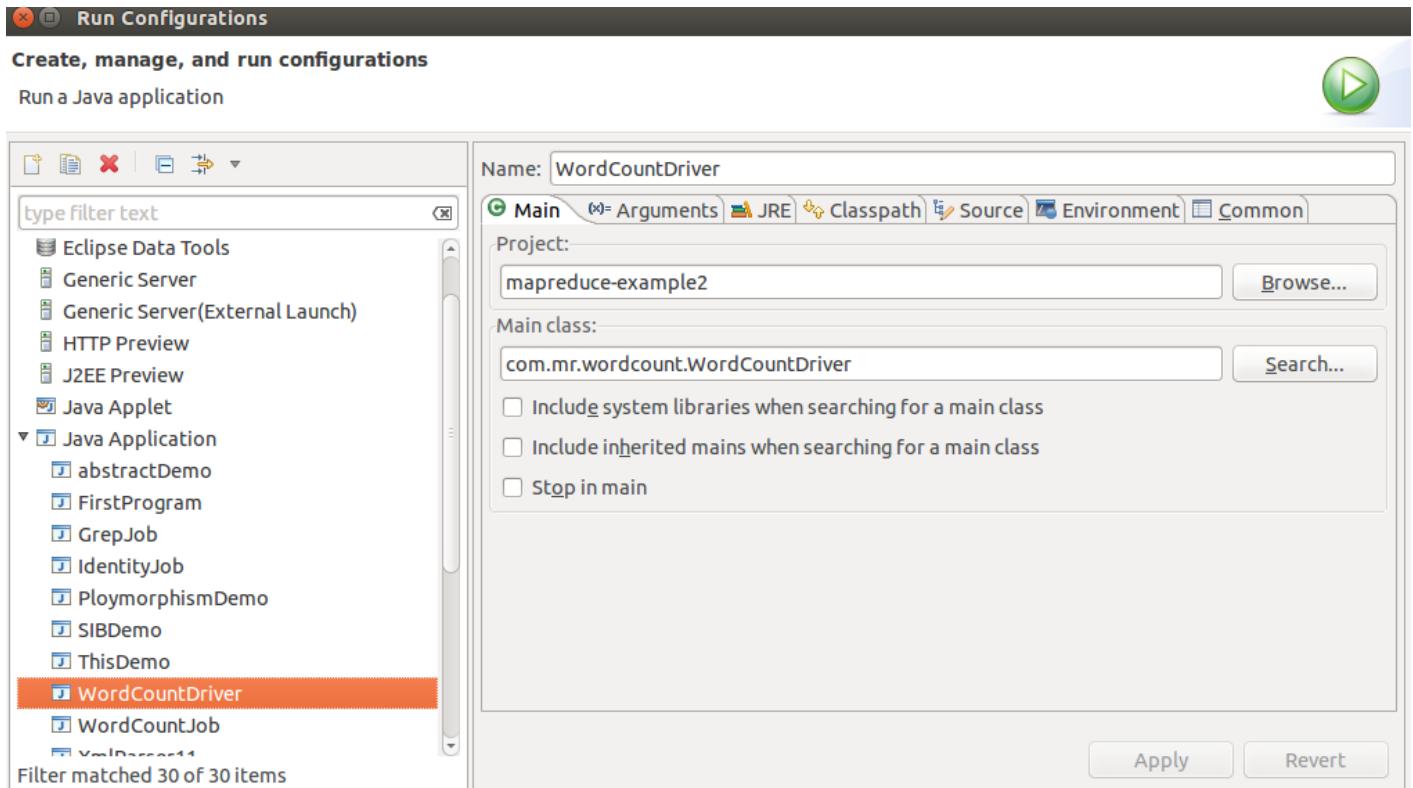
Steps to Run in Local Runner-mode:

- 1) Right click on Driver class ex: WordCountDriver.java
- 2) Select **Run As → Run configurations** and **Double click on Java Application**

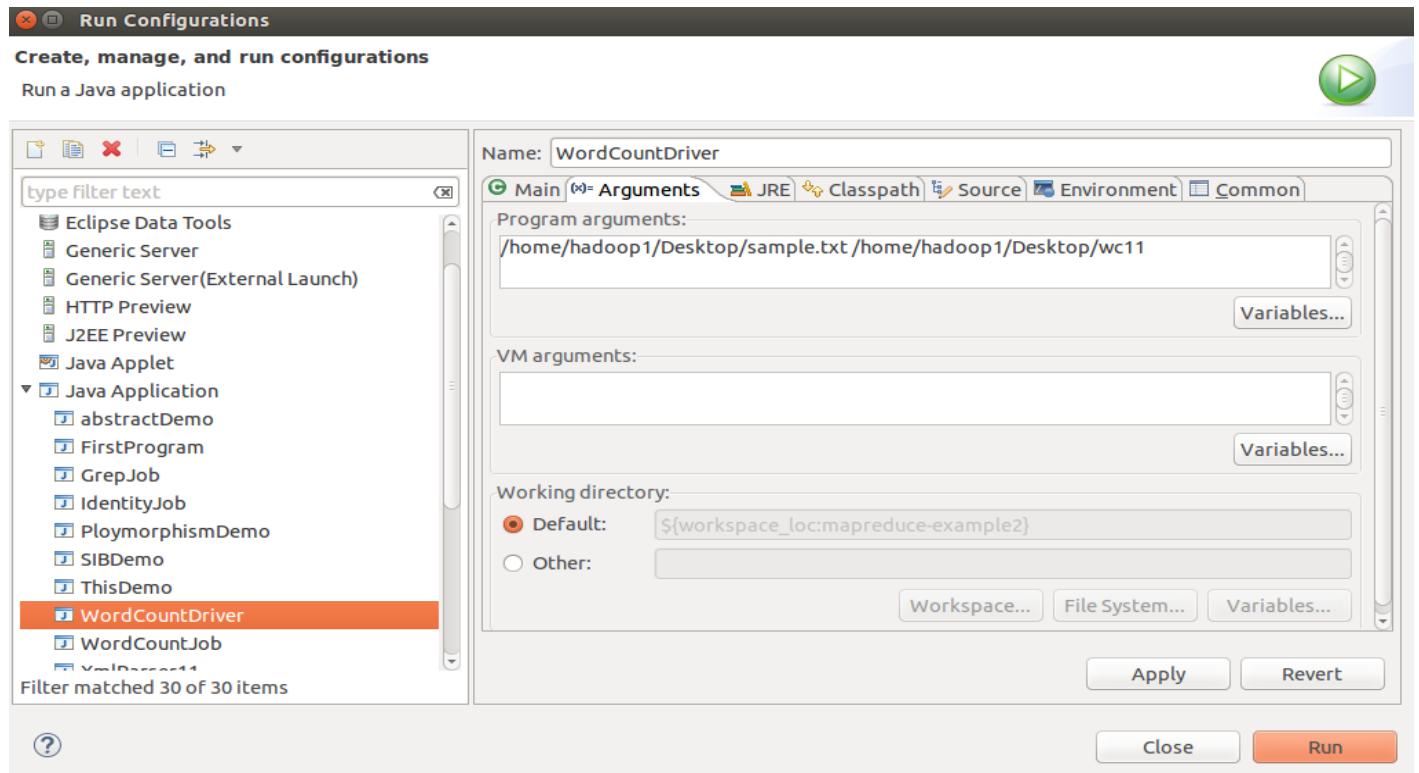




3) It will list out the respective driver class name (which you selected driver class)

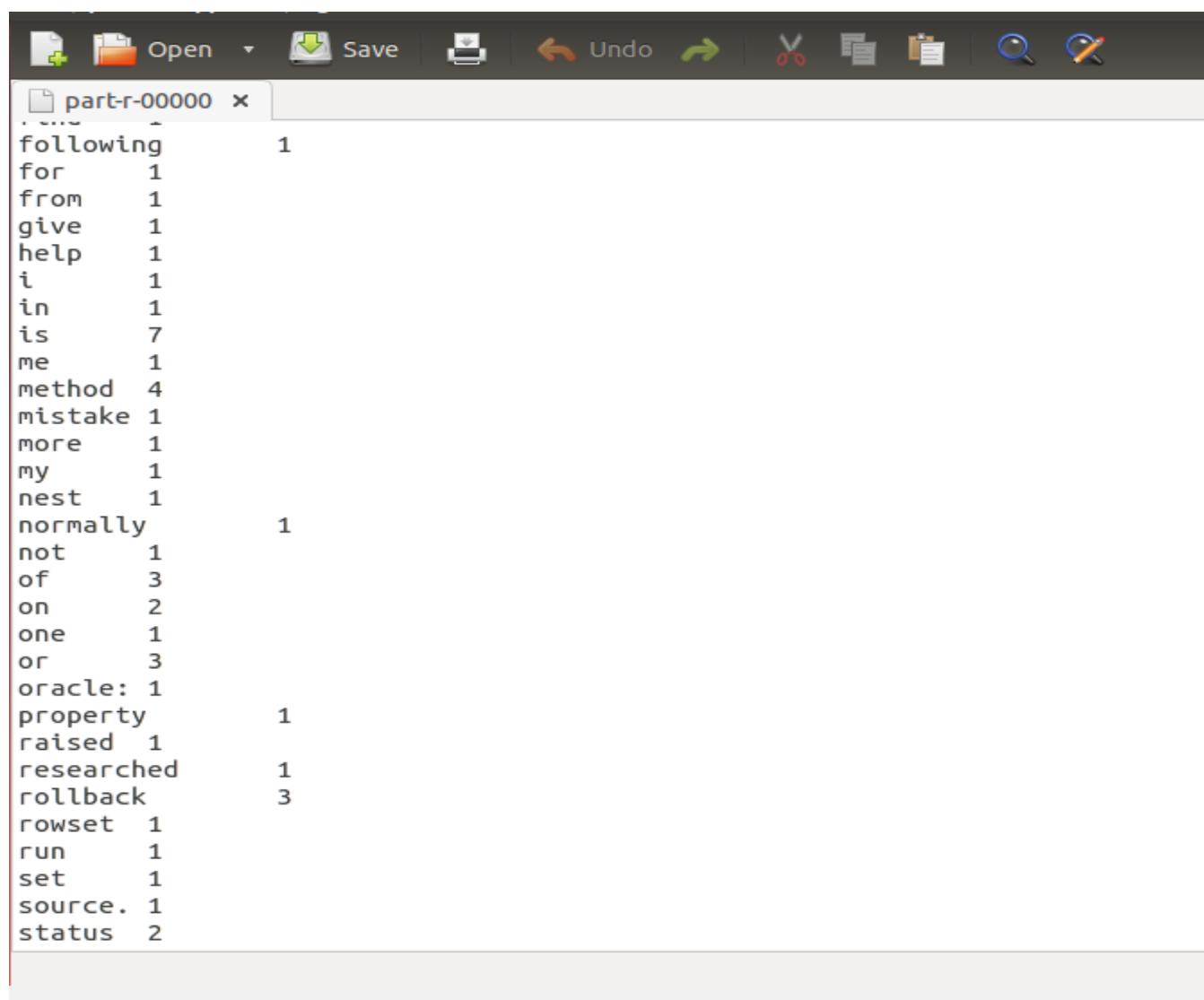


- 4) Click on **Arguments** tab from right pane and give the **local input path** (not HDFS) , **local out path dir** (the dir should not exists) and click on **RUN**



- 5) Go to output dir in local file system (/home/hadoop1/Desktop/wc1), it will have **_SUCCESS** file and **part_00000** files. The **part_00000** is final output (reducer output)





The screenshot shows the Eclipse IDE interface with a text editor window open. The window title is "part-r-00000". The content of the editor is a word count output, likely from a MapReduce program, showing various words and their counts:

```
following      1
for            1
from           1
give           1
help           1
i              1
in              1
is              7
me              1
method         4
mistake        1
more           1
my              1
nest            1
normally       1
not             1
of              3
on              2
one             1
or              3
oracle:        1
property       1
raised          1
researched     1
rollback        3
rowset          1
run             1
set             1
source.         1
status          2
```