**Experiment No:11**

**Aim: Write a code in JAVA for a simple Word Count application that counts the number of occurrences of each word in a given input set using the Hadoop Map Reduce framework on local-standalone set-up.**

#### Prerequisites: [Hadoop](https://www.geeksforgeeks.org/hadoop-an-introduction/) and [MapReduce](https://www.geeksforgeeks.org/map-reduce-in-hadoop/)

**Theory:** Counting the number of words in any language is a piece of cake like in C, C++, Python, Java, etc. MapReduce also uses Java but it is very easy if you know the syntax on how to write it. It is the basic of MapReduce. You will first learn how to execute this code similar to “Hello World” program in other languages. So here are the steps which show how to write a MapReduce code for Word Count.  
**Example:**  
*Input:* 

Hello I am GeeksforGeeks

Hello I am an Intern

*Output:*

GeeksforGeeks 1

Hello 2

I 2

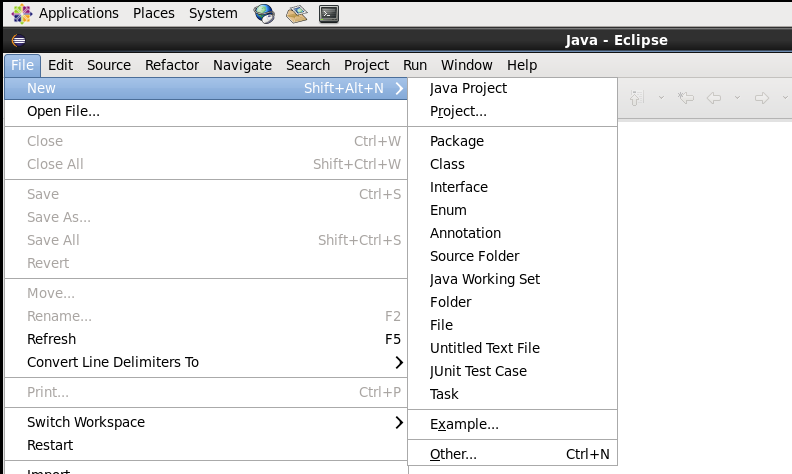
Intern 1

am 2

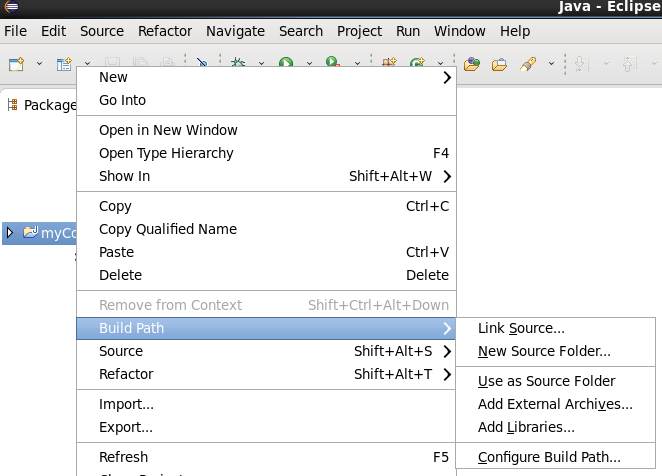
an 1

**Steps:** 

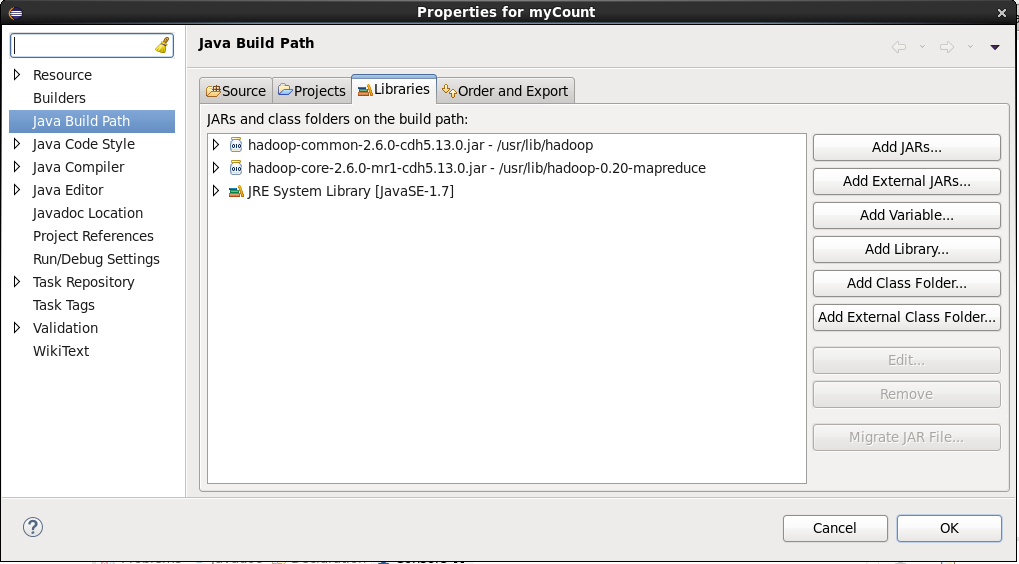
* First Open **Eclipse**-> then select **File**-> **New**-> **Java Project** ->Name it **WordCount**-> then **Finish**.



* Create Three Java Classes into the project. Name them **WCDriver**(having the main function), **WCMapper**, **WCReducer**.
* You have to include two Reference Libraries for that:  
  Right Click on **Project**-> then select **Build Path**-> Click on **Configure Build Path**



* In the above figure, you can see the Add External JARs option on the Right Hand Side. Click on it and add the below mention files. You can find these files in */usr/lib/*  
  1. /usr/lib/hadoop-0.20-mapreduce/hadoop-core-2.6.0-mr1-cdh5.13.0.jar   
  2. /usr/lib/hadoop/hadoop-common-2.6.0-cdh5.13.0.jar





**Mapper Code:** You have to copy paste this program into the WCMapper Java Class file.

* Java

|  |
| --- |
| // Importing libraries  import java.io.IOException;  import org.apache.hadoop.io.IntWritable;  import org.apache.hadoop.io.LongWritable;  import org.apache.hadoop.io.Text;  import org.apache.hadoop.mapred.MapReduceBase;  import org.apache.hadoop.mapred.Mapper;  import org.apache.hadoop.mapred.OutputCollector;  import org.apache.hadoop.mapred.Reporter;    public class WCMapper extends MapReduceBase implements Mapper<LongWritable,                                                  Text, Text, IntWritable> {        // Map function      public void map(LongWritable key, Text value, OutputCollector<Text,                   IntWritable> output, Reporter rep) throws IOException      {            String line = value.toString();            // Splitting the line on spaces          for (String word : line.split(" "))          {              if (word.length() > 0)              {                  output.collect(new Text(word), new IntWritable(1));              }          }      }  } |

**Reducer Code:** You have to copy paste this program into the WCReducer Java Class file.

* Java

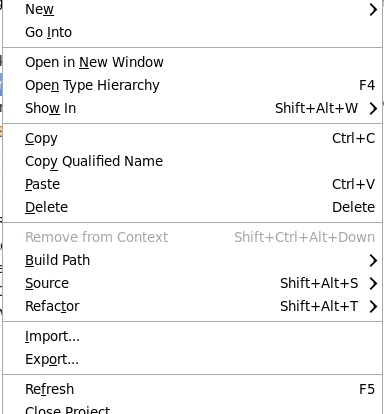
|  |
| --- |
| // Importing libraries  import java.io.IOException;  import java.util.Iterator;  import org.apache.hadoop.io.IntWritable;  import org.apache.hadoop.io.Text;  import org.apache.hadoop.mapred.MapReduceBase;  import org.apache.hadoop.mapred.OutputCollector;  import org.apache.hadoop.mapred.Reducer;  import org.apache.hadoop.mapred.Reporter;    public class WCReducer extends MapReduceBase implements Reducer<Text,                                      IntWritable, Text, IntWritable> {        // Reduce function      public void reduce(Text key, Iterator<IntWritable> value,                     OutputCollector<Text, IntWritable> output,                              Reporter rep) throws IOException      {            int count = 0;            // Counting the frequency of each words          while (value.hasNext())          {              IntWritable i = value.next();              count += i.get();          }            output.collect(key, new IntWritable(count));      }  } |

**Driver Code:** You have to copy paste this program into the WCDriver Java Class file.

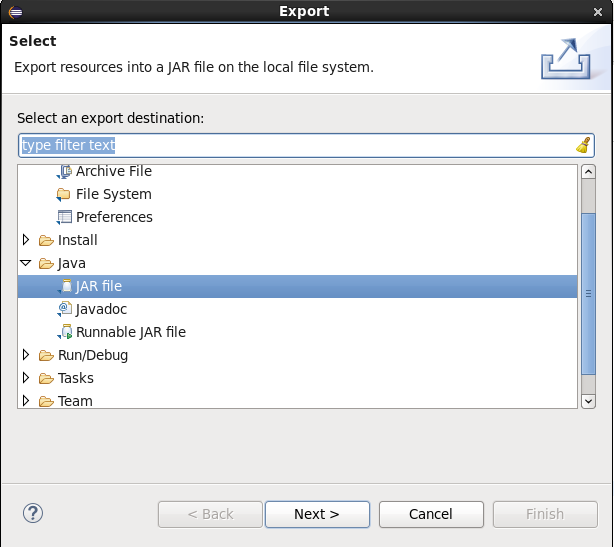
* Java

|  |
| --- |
| // Importing libraries  import java.io.IOException;  import org.apache.hadoop.conf.Configured;  import org.apache.hadoop.fs.Path;  import org.apache.hadoop.io.IntWritable;  import org.apache.hadoop.io.Text;  import org.apache.hadoop.mapred.FileInputFormat;  import org.apache.hadoop.mapred.FileOutputFormat;  import org.apache.hadoop.mapred.JobClient;  import org.apache.hadoop.mapred.JobConf;  import org.apache.hadoop.util.Tool;  import org.apache.hadoop.util.ToolRunner;    public class WCDriver extends Configured implements Tool {        public int run(String args[]) throws IOException      {          if (args.length < 2)          {              System.out.println("Please give valid inputs");              return -1;          }            JobConf conf = new JobConf(WCDriver.class);          FileInputFormat.setInputPaths(conf, new Path(args[0]));          FileOutputFormat.setOutputPath(conf, new Path(args[1]));          conf.setMapperClass(WCMapper.class);          conf.setReducerClass(WCReducer.class);          conf.setMapOutputKeyClass(Text.class);          conf.setMapOutputValueClass(IntWritable.class);          conf.setOutputKeyClass(Text.class);          conf.setOutputValueClass(IntWritable.class);          JobClient.runJob(conf);          return 0;      }        // Main Method      public static void main(String args[]) throws Exception      {          int exitCode = ToolRunner.run(new WCDriver(), args);          System.out.println(exitCode);      }  } |

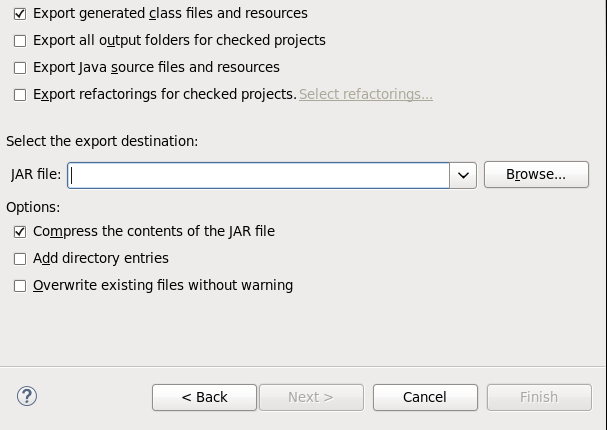
* Now you have to make a jar file. Right Click on **Project**-> **Click on Export**-> **Select export destination as Jar File**-> **Name the jar File**(WordCount.jar) -> **Click on next** -> at last **Click on Finish**. Now copy this file into the Workspace directory of Cloudera



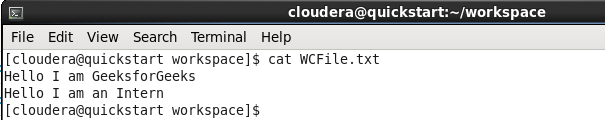








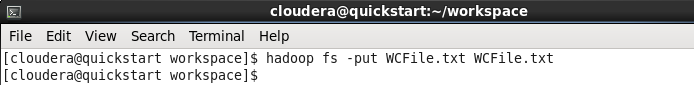
* Open the terminal on CDH and change the directory to the workspace. You can do this by using “cd workspace/” command. Now, Create a text file(**WCFile.txt**) and move it to HDFS. For that open terminal and write this code(remember you should be in the same directory as jar file you have created just now).



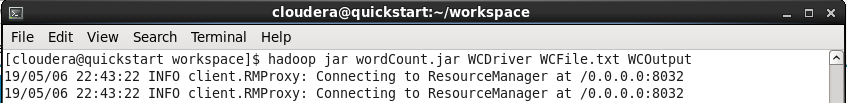
* Now, run this command to copy the file input file into the HDFS.

hadoop fs -put WCFile.txt WCFile.txt





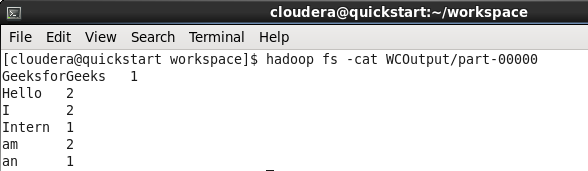
* Now to run the jar file by writing the code as shown in the screenshot.



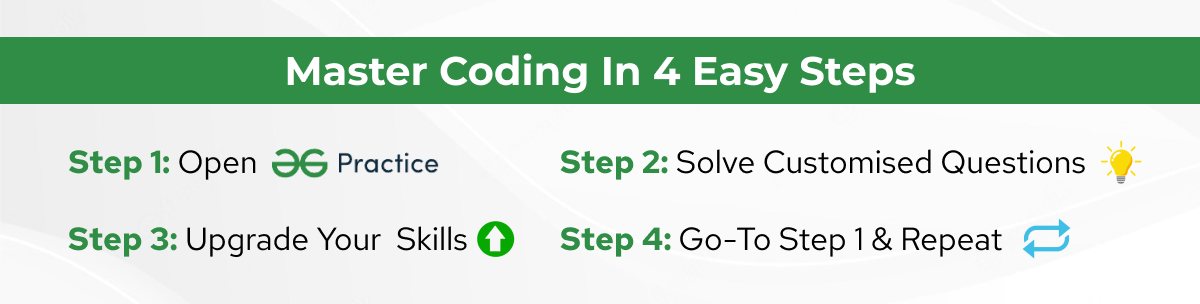
* After Executing the code, you can see the result in *WCOutput*file or by writing following command on terminal.

hadoop fs -cat WCOutput/part-00000







**[](https://practice.geeksforgeeks.org/explore?page=1&sortBy=submissions&utm_source=gfg&utm_medium=in-article&utm_campaign=in-article)**