WORD COUNT

1. Open Eclipse> File > New > Java Project >(Name it – MRProgramsDemo) > Finish.
2. Right Click > New > Package (Name it - PackageDemo) > Finish.
3. Right Click on Package > New > Class (Name it - WordCount).
4. Add Following Reference Libraries:
    1. Right Click on Project > Build Path> Add External
        1. */usr/lib/hadoop-0.20/**hadoop-core.jar***
        2. *Usr/lib/hadoop-0.20/lib/**Commons-cli-1.2.jar***
5. Type the following code:

```
package PackageDemo;

import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;

public class WordCount {
    public static void main(String [] args) throws Exception
    {
            Configuration c=new Configuration();
            String[] files=new GenericOptionsParser(c,args).getRemainingArgs();
            Path input=new Path(files[0]);
            Path output=new Path(files[1]);
            Job j=new Job(c,"wordcount");
            j.setJarByClass(WordCount.class);
            j.setMapperClass(MapForWordCount.class);
            j.setReducerClass(ReduceForWordCount.class);
            j.setOutputKeyClass(Text.class);
            j.setOutputValueClass(IntWritable.class);
            FileInputFormat.addInputPath(j, input);
            FileOutputFormat.setOutputPath(j, output);
            System.exit(j.waitForCompletion(true)?0:1);
    }
```

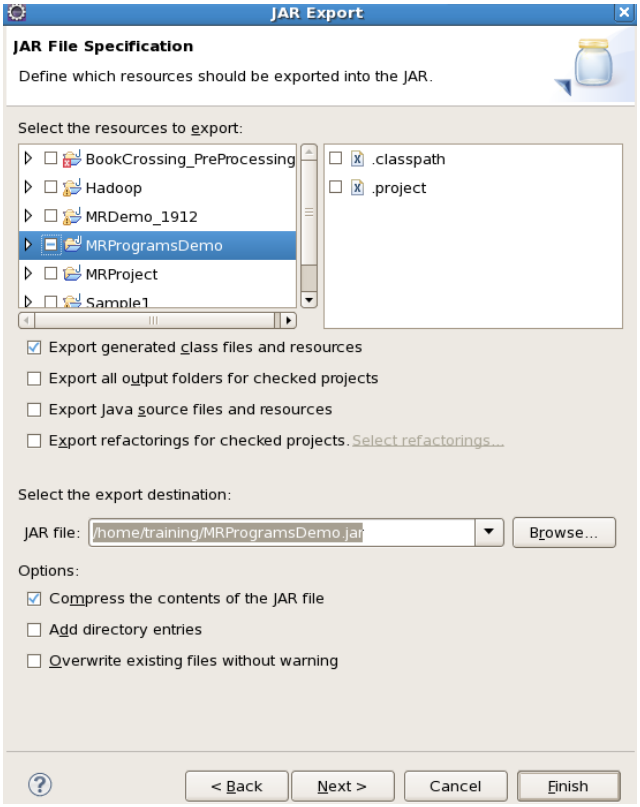
```
    public static class MapForWordCount extends Mapper<LongWritable, Text, Text,
    IntWritable>{
        public void map(LongWritable key, Text value, Context con) throws IOException,
        InterruptedException
        {
                String line = value.toString();
                String[] words=line.split(",");
                for(String word: words )
                {
                  Text outputKey = new Text(word.toUpperCase().trim());
                  IntWritable outputValue = new IntWritable(1);
                  con.write(outputKey, outputValue);
                }
        }
    }

    public static class ReduceForWordCount extends Reducer<Text, IntWritable, Text,
    IntWritable>
    {
        public void reduce(Text word, Iterable<IntWritable> values, Context con) throws
        IOException, InterruptedException
        {
                int sum = 0;
                  for(IntWritable value : values)
                  {
                          sum += value.get();
                  }
                con.write(word, new IntWritable(sum));
        }
     }
}
```

The above program consists of three classes:

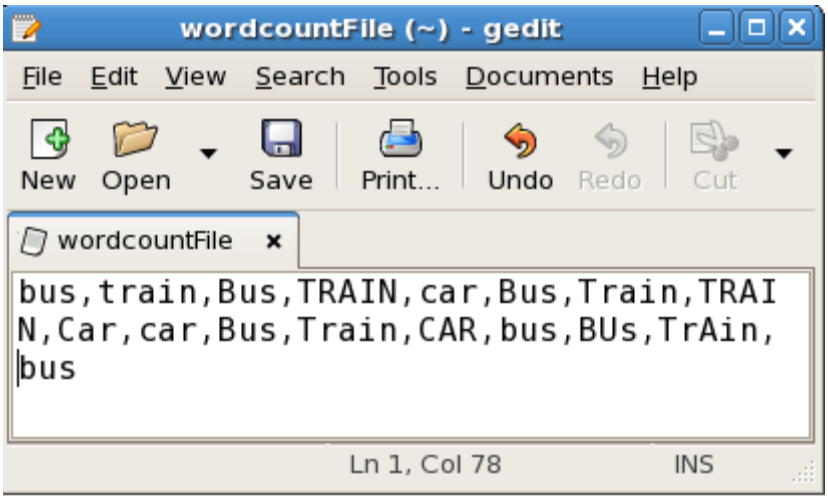
- Driver class (Public, void, static, or main; this is the entry point).
- The Map class which **extends** the public class Mapper<KEYIN,VALUEIN,KEYOUT,VALUEOUT> and implements the Map function.
- The Reduce class which extends the public class Reducer<KEYIN,VALUEIN,KEYOUT,VALUEOUT> and implements the Reduce function.

6. Make a jar file

Right Click on Project> Export> Select export destination as **Jar File >** next> Finish.



7. Take a text file and move it into HDFS format:



To move this into Hadoop directly, open the terminal and enter the following commands:

[training@localhost ~]$ **hadoop fs -put wordcountFile wordCountFile**

8. Run the jar file:

(Hadoop jar jarfilename.jar packageName.ClassName PathToInputTextFile PathToOutputDirectry)

[training@localhost ~]$ **hadoop jar MRProgramsDemo.jar PackageDemo.WordCount wordCountFile MRDir1**

8. Open the result:

[training@localhost ~]$ **hadoop fs -ls MRDir1**

Found 3 items

-rw-r--r-- 1 training supergroup 0 2016-02-23 03:36 /user/training/MRDir1/_SUCCESS

drwxr-xr-x - training supergroup 0 2016-02-23 03:36 /user/training/MRDir1/_logs

-rw-r--r-- 1 training supergroup 20 2016-02-23 03:36 /user/training/MRDir1/part-r-00000

[training@localhost ~]$ **hadoop fs -cat MRDir1/part-r-00000**

BUS 7

CAR 4

TRAIN 6