```
import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
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.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class WordCount extends Configured implements Tool {

   public static void main(String[] args) throws Exception {
      int res = ToolRunner.run(new Configuration(), new WordCount(), args);
      System.exit(res);
   }

   public int run(String[] args) throws Exception {
      Configuration conf = getConf();
      Job job = Job.getInstance(conf, "WordCount");
      job.setJarByClass(WordCount.class);

      // Input and Output paths
      FileInputFormat.setInputPaths(job, new Path(args[0]));
      FileOutputFormat.setOutputPath(job, new Path(args[1]));

      // Mapper and Reducer classes
      job.setMapperClass(TokenizerMapper.class);
      job.setCombinerClass(SumReducer.class);
      job.setReducerClass(SumReducer.class);

      // Output types
      job.setOutputKeyClass(Text.class);
      job.setOutputValueClass(IntWritable.class);

      // Input and Output format
      job.setInputFormatClass(TextInputFormat.class);
```

```
        job.setOutputFormatClass(TextOutputFormat.class);

        return job.waitForCompletion(true) ? 0 : 1;
    }

    public static class TokenizerMapper extends Mapper<LongWritable, Text, Text,
IntWritable> {
        private final static IntWritable one = new IntWritable(1);
        private final Text word = new Text();

        public void map(LongWritable key, Text value, Context context)
                throws IOException, InterruptedException {
            StringTokenizer tokenizer = new StringTokenizer(value.toString());
            while (tokenizer.hasMoreTokens()) {
                word.set(tokenizer.nextToken());
                context.write(word, one);
            }
        }
    }

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

—----------

Input:-

```
hello world
hello hadoop
```

Output:-

```
hadoop 1
hello 2
world 1
```