**Wordmapper.java**

package org.abc.WC;

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 WordMapper extends MapReduceBase implements

Mapper<LongWritable, Text, Text, IntWritable>{

@Override

public void map(LongWritable key, Text value,

OutputCollector<Text, IntWritable> output, Reporter reporter)

throws IOException {

String s=value.toString();

for (String word : s.split("\\w+\*")){

if (word.length()>0){

output.collect(new Text(word), new IntWritable(1));

}

}

}

}

**Sumreducer.java**

package org.abc.WC;

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.Reducer;

import org.apache.hadoop.mapred.OutputCollector;

import org.apache.hadoop.mapred.Reporter;

public class SumReducer extends MapReduceBase implements

Reducer<Text, IntWritable, Text, IntWritable>{

@Override

public void reduce(Text key, Iterator<IntWritable> values,

OutputCollector<Text, IntWritable> output, Reporter reporter)

throws IOException {

int wordCount=0;

while (values.hasNext()){

IntWritable value= values.next();

wordCount += value.get();

}

output.collect(key, new IntWritable(wordCount));

}

}

**Wordcount.java**

package org.abc.WC;

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.conf.Configured;

import org.apache.hadoop.util.Tool;

import org.apache.hadoop.util.ToolRunner;

public class WordCount extends Configured implements Tool{

@Override

public int run(String[] args) throws Exception{

if (args.length !=2){

System.out.printf("Usage: %s [generic options] <input dir> <output dir>\n", getClass().getSimpleName());

ToolRunner.printGenericCommandUsage(System.out);

return -1;

}

JobConf conf= new JobConf(getConf(), WordCount.class);

conf.setJobName(this.getClass().getName());

FileInputFormat.setInputPaths(conf, new Path(args[0]));

FileOutputFormat.setOutputPath(conf, new Path(args[1]));

conf.setMapperClass(WordMapper.class);

conf.setReducerClass(SumReducer.class);

conf.setMapOutputKeyClass(Text.class);

conf.setMapOutputValueClass(IntWritable.class);

conf.setOutputKeyClass(Text.class);

conf.setOutputValueClass(IntWritable.class);

JobClient.runJob(conf);

return 0;

}

public static void main(String[] args) throws Exception{

int exitCode= ToolRunner.run(new WordCount(), args);

System.exit(exitCode);

}

}





