package com.mapreduce.wc;

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();

// Ensure correct input arguments

if (files.length < 2) {

System.err.println("Usage: WordCount <input path> <output path>");

System.exit(-1);

}

Path input = new Path(files[0]);

Path output = new Path(files[1]);

Job j = Job.getInstance(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);

}

// Mapper Class

public static class MapForWordCount extends Mapper<LongWritable, Text, Text, IntWritable> {

private final static IntWritable one = new IntWritable(1);

private Text wordText = new Text();

public void map(LongWritable key, Text value, Context con) throws IOException, InterruptedException {

String line = value.toString().trim();

String[] words = line.split("\\s+"); // Handles multiple spaces

for (String word : words) {

if (!word.isEmpty()) { // Avoid empty strings

wordText.set(word.trim().toUpperCase());

con.write(wordText, one);

}

}

}

}

// Reducer Class

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));

}

}

}