**import** java.io.IOException;

**import** java.util.StringTokenizer;

**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;

**public** **class** WordCount {

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

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

**private** Text word = **new** Text();

**protected** **void** map(LongWritable key, Text value, Context context) **throws** IOException, InterruptedException {

String line = value.toString();

StringTokenizer tokenizer = **new** StringTokenizer(line);

**while** (tokenizer.hasMoreTokens()) {

word.set(tokenizer.nextToken());

context.write(word, ***one***);

}

}

}

**public** **static** **class** ReduceForWordCount **extends** Reducer<Text, IntWritable, Text, IntWritable> {

**private** IntWritable value = **new** IntWritable(0);

**protected** **void** reduce(Text key, Iterable<IntWritable> values, Context context) **throws** IOException, InterruptedException {

**int** sum = 0;

**for** (IntWritable value : values)

sum += value.get();\-ơ

context.write(key, value);

}

}

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

{

Configuration conf = **new** Configuration();

@SuppressWarnings("deprecation")

Job job = **new** ~~Job~~(conf, "word count");

job.setJarByClass(WordCount.**class**);

job.setMapperClass(MapForWordCount.**class**);16/9

//job.setCombinerClass(ReduceForWordCount.class);

job.setReducerClass(ReduceForWordCount.**class**);

job.setOutputKeyClass(Text.**class**);

job.setOutputValueClass(IntWritable.**class**);

FileInputFormat.*addInputPath*(job, **new** Path(".\\input"));

FileOutputFormat.*setOutputPath*(job, **new** Path("D:\\MapReduce\\output"));

**if** (job.waitForCompletion(**true**))

{

System.***out***.println("success");

System.*exit*(0);

}

}

}