Hadop案例之WordCount

代码如下：

**package** hadopp\_wordCount;

**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;

**import** org.apache.hadoop.util.GenericOptionsParser;

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

//map

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

{

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

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

@Override

**protected** **void** map(LongWritable **key**, Text **value**, **Context** **context**)

**throws** IOException, InterruptedException {

StringTokenizer **iter** = **new** StringTokenizer(value.toString());

**while** (iter.hasMoreTokens()) {

word.set(iter.nextToken());

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

}

}

}

//reduce

**public** **static** **class** reduce **extends** Reducer<Text, IntWritable, Text, IntWritable>

{

**private** IntWritable result = **new** IntWritable();

@Override

**protected** **void** reduce(Text **key**, Iterable<IntWritable> **value**,

**Context** **cont**) **throws** IOException, InterruptedException {

**int** **sum** = 0;

**for** (IntWritable **i** : value) {

sum += i.get();

}

result.set(sum);

cont.write(key, result);

}

}

//main

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

{

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

String[] **otherArgs** = **new** GenericOptionsParser(conf,args).getRemainingArgs();

**if** (otherArgs.length < 2) {

System.***out***.println("Usage: wordcount <in> [<in>...] <out>");

System.*exit*(2);

}

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

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

job.setMapperClass(Map.**class**);

job.setCombinerClass(reduce.**class**);

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

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

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

**for**(**int** **i** = 0; i < otherArgs.length -1; i++)

{

**FileInputFormat**.*addInputPath*(job, **new** Path(otherArgs[i]));

}

**FileOutputFormat**.*setOutputPath*(job, **new** Path(otherArgs[otherArgs.length - 1]));

System.*exit*(job.waitForCompletion(**true**) ? 0 : 1);

}

}

代码比较简单，网上也有很多介绍，本文不再详细描述。

需要注意的一点是命名空间问题：

如果按照如下方式执行WordCount，会报错：

root@node1:/usr/local/hadoop/hadoop-2.5.2/myJar# hadoop jar WordCount.jar WordCount /usr/local/hadooptempdata/input/wc /usr/local/hadooptempdata/output/wc

Exception in thread "main" java.lang.ClassNotFoundException: WordCount

at java.net.URLClassLoader.findClass(URLClassLoader.java:381)

at java.lang.ClassLoader.loadClass(ClassLoader.java:424)

at java.lang.ClassLoader.loadClass(ClassLoader.java:357)

at java.lang.Class.forName0(Native Method)

at java.lang.Class.forName(Class.java:348)

at org.apache.hadoop.util.RunJar.main(RunJar.java:205)

原因是默认命名空间问题，本文中使用的包是 package hadopp\_wordCount;

按照如下方式执行就没问题：

hadoop jar WordCount.jar hadopp\_wordCount.WordCount /usr/local/hadooptempdata/input/wc /usr/local/hadooptempdata/output/wc