

作业五 利用莎士比亚文集实现WordCount

首先配置Maven及Eclipse实现

因为数据结构课为了图方便选了eclipse作为java的IDE（电脑快满了放弃IntelliJ IDEA），发现同样可以实现maven管理。具体操作步骤如下：

- 在安装和配置完成maven之后（运行结果如下），首先利用archetype创建maven项目

```
mvn archetype:generate
```

```
C:\Users\lenovo>mvn -version
Apache Maven 3.6.3 (cecedd343002696d0abb50b32b541b8a6ba2883f)
Maven home: D:\Maven\apache-maven-3.6.3\bin\..
Java version: 14.0.2, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk-14.0.2
Default locale: zh_CN, platform encoding: GBK
OS name: "windows 10", version: "10.0", arch: "amd64", family: "windows"
```

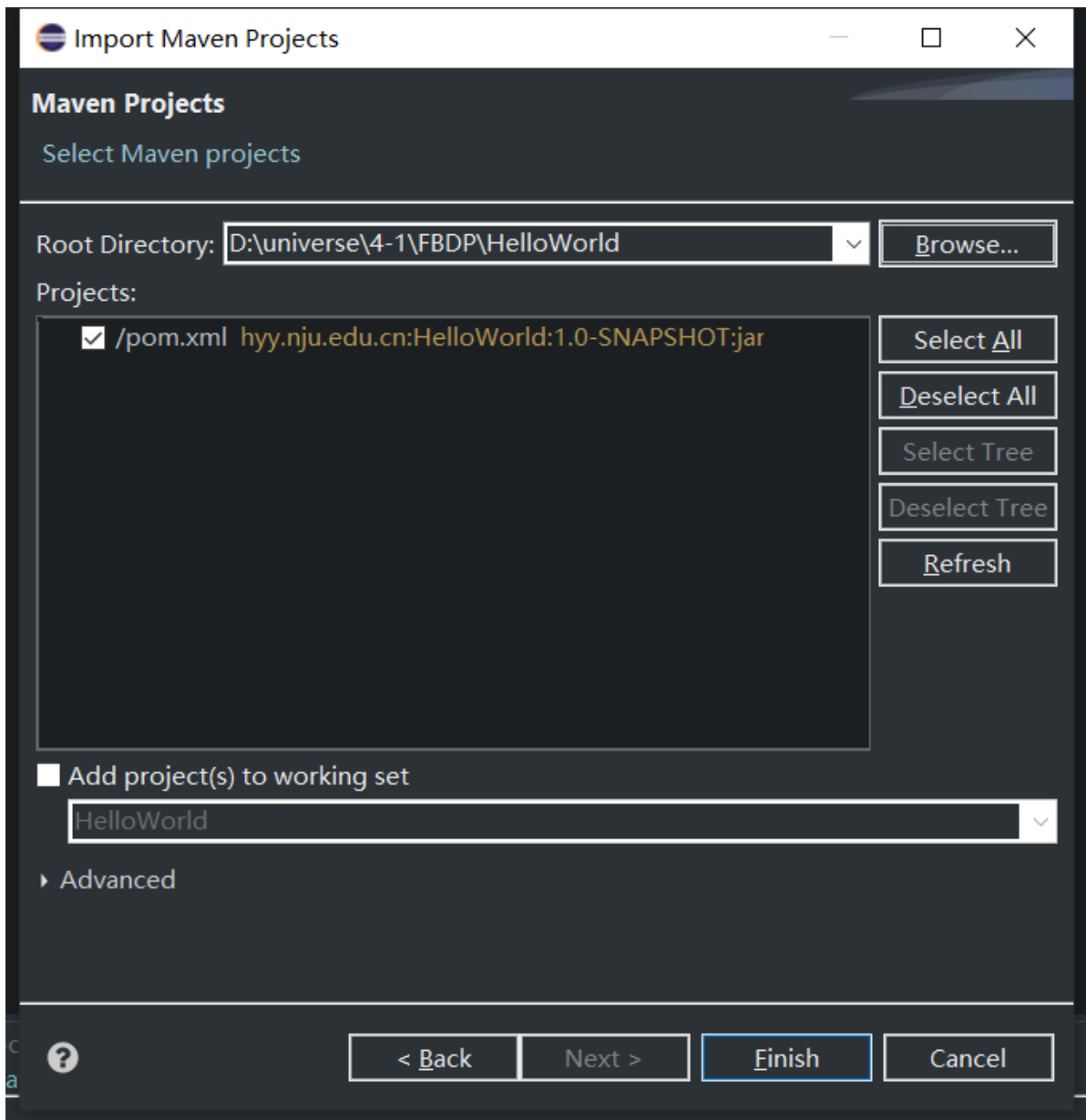
- 选择quickStart 7 作为maven测试：

```
Choose a number or apply filter (format: [groupId:]artifactId, case sensitive contains): 7:
Define value for property 'groupId': hyy.nju.edu.cn
Define value for property 'artifactId': HelloWorld
Define value for property 'version' 1.0-SNAPSHOT: :
Define value for property 'package' hyy.nju.edu.cn: :
Confirm properties configuration:
groupId: hyy.nju.edu.cn
artifactId: HelloWorld
version: 1.0-SNAPSHOT
package: hyy.nju.edu.cn
Y: :
[INFO] -----
[INFO] Using following parameters for creating project from Old (1.x) Archetype: maven-archetype-quickstart:1.1
[INFO] -----
[INFO] Parameter: basedir, Value: D:\universe\4-1\FBDP
[INFO] Parameter: package, Value: hyy.nju.edu.cn
[INFO] Parameter: groupId, Value: hyy.nju.edu.cn
[INFO] Parameter: artifactId, Value: HelloWorld
[INFO] Parameter: packageName, Value: hyy.nju.edu.cn
[INFO] Parameter: version, Value: 1.0-SNAPSHOT
[INFO] project created from Old (1.x) Archetype in dir: D:\universe\4-1\FBDP\HelloWorld
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 01:01 min
[INFO] Finished at: 2020-10-26T21:30:47+08:00
[INFO] -----
```

- 项目创建完成之后，cd到项目内利用 `mvn eclipse:eclipse` 转化为eclipse工程项目

```
D:\universe\4-1\FBDP>cd HelloWorld
D:\universe\4-1\FBDP\HelloWorld>mvn eclipse:eclipse
[INFO] Scanning for projects...
[INFO] -----< hyy.nju.edu.cn:HelloWorld >-----
[INFO] Building HelloWorld 1.0-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO] >>> maven-eclipse-plugin:2.10:eclipse (default-cli) > generate-resources @ HelloWorld >>>
[INFO] <<< maven-eclipse-plugin:2.10:eclipse (default-cli) < generate-resources @ HelloWorld <<<
[INFO] --- maven-eclipse-plugin:2.10:eclipse (default-cli) @ HelloWorld ---
[INFO] Using Eclipse Workspace: D:\universe\4-1\FBDP
[WARNING] Workspace defines a VM that does not contain a valid jre/lib/rt.jar: C:\Program Files\Java\jdk-14.0.2
[INFO] Adding default classpath container: org.eclipse.jdt.launching.JRE_CONTAINER
[INFO] Not writing settings - defaults suffice
[INFO] Wrote Eclipse project for "HelloWorld" to D:\universe\4-1\FBDP\HelloWorld.
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 1.813 s
[INFO] Finished at: 2020-10-26T21:32:28+08:00
[INFO] -----
```

之后就可以在eclipse中导入Maven project并打开啦！



- 之后执行 `mvn clean`
- `mvn clean compile` 报错如下:

```
命令提示符
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ HelloWorld ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 1 source file to D:\universe\4-1\FBDP\HelloWorld\target\classes
[INFO] COMPILATION ERROR :
[INFO] -----
[ERROR] 不再支持源选项 5。请使用 7 或更高版本。
[ERROR] 不再支持目标选项 5。请使用 7 或更高版本。
[INFO] 2 errors
[INFO] -----
[INFO] BUILD FAILURE
[INFO] -----
[INFO] Total time: 1.605 s
[INFO] Finished at: 2020-10-26T21:36:47+08:00
[INFO] -----
[ERROR] Failed to execute goal org.apache.maven.plugins:maven-compiler-plugin:3.1:compile (default-compile) on project HelloWorld: Compilation failure:
[ERROR] 不再支持源选项 5。请使用 7 或更高版本。
[ERROR] 不再支持目标选项 5。请使用 7 或更高版本。
[ERROR] -> [Help 1]
[ERROR]
[ERROR] To see the full stack trace of the errors, re-run Maven with the -e switch.
[ERROR] Re-run Maven using the -X switch to enable full debug logging.
[ERROR]
[ERROR] For more information about the errors and possible solutions, please read the following articles:
[ERROR] [Help 1] http://cwiki.apache.org/confluence/display/MAVEN/MojoFailureException
D:\universe\4-1\FBDP\HelloWorld>
```

问题：目前jdk版本不支持

解决方案：修改pom中properties，增加maven编译的 java.version jdk版本设置，以及 maven.compiler.source 资源编译jdk版本设置和 maven.compiler.target 资源构建jdk版本设置

```
<properties>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  <maven.compiler.encoding>UTF-8</maven.compiler.encoding>
  <java.version>14</java.version>
  <maven.compiler.source>14</maven.compiler.source>
  <maven.compiler.target>14</maven.compiler.target>
</properties>
```

- 重新进行clean和compile即可

```
D:\universe\4-1\FBDP\HelloWorld>mvn compile
[INFO] Scanning for projects...
[INFO] -----< hyy.nju.edu.cn:HelloWorld >-----
[INFO] Building HelloWorld 1.0-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ HelloWorld ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory D:\universe\4-1\FBDP\HelloWorld\src\main\resources
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ HelloWorld ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 1 source file to D:\universe\4-1\FBDP\HelloWorld\target\classes
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 2.050 s
[INFO] Finished at: 2020-10-26T21:41:22+08:00
[INFO] -----
```

- mvn clean test

```
-----
T E S T S
-----
Running hyy.nju.edu.cn.AppTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.018 sec


Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 3.894 s
[INFO] Finished at: 2020-10-26T21:42:13+08:00
[INFO] -----
```

- mvn clean package 及 mvn clean install

target文件下已经生成jar包

classes	2020/10/26 21:43	文件夹	
generated-sources	2020/10/26 21:43	文件夹	
generated-test-sources	2020/10/26 21:43	文件夹	
maven-archiver	2020/10/26 21:43	文件夹	
maven-status	2020/10/26 21:43	文件夹	
surefire-reports	2020/10/26 21:43	文件夹	
test-classes	2020/10/26 21:43	文件夹	
 HelloWorld-1.0-SNAPSHOT.jar	2020/10/26 21:43	Executable Jar File	3 KB

- 在命令行中试运行jar包报错（没有主清单属性，即 jar 中的 /META-INF/ MANIFEST.MF缺少项目启动项，即没有Main-Class）

```
D:\universe\4-1\FBDP\HelloWorld>java -jar HelloWorld-1.0-SNAPSHOT.jar
Error: Unable to access jarfile HelloWorld-1.0-SNAPSHOT.jar
```

（上网搜索发现如果要运行需要配置plugin插件，此处少了一步）

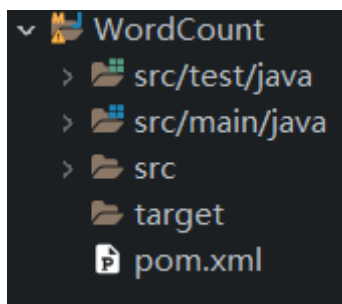
解决方案：在pom.xml的project节点中添加plugin配置，保存修改后重新从package实现，成功~

```
<build>
  <finalName>${project.artifactId}</finalName><!--修改编译出来的jar包名，仅为
{artifactId}.jar-->
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-jar-plugin</artifactId>
      <version>3.2.0</version>
      <configuration>
        <archive>
          <manifest>
            <addClasspath>true</addClasspath>
            <mainClass></mainClass> <!-- 此处为主入口，需要修改为对应
的main类-->
          </manifest>
        </archive>
      </configuration>
    </plugin>
  </plugins>
</build>
```

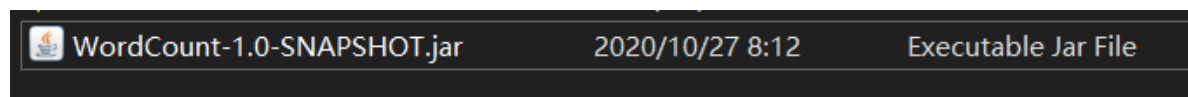
```
D:\universe\4-1\FBDP\HelloWorld>java -jar target\HelloWorld.jar
Hello World!
```

对WordCount进行jar包实现

- 首先同上述步骤创建WordCount项目



- 修改pom对Hadoop进行配置，编写wordcount主程序，调整测试并利用 `maven package` 打包



- 加入input文件，将本地仓库上传到github

main
FBDP / WordCount /
Go to file
Add file

Hannahalover add input file
#903629 4 minutes ago History

..		
input	add input file	4 minutes ago
src	add WordCount Project	1 hour ago
target	add WordCount Project	1 hour ago
.gitignore	add WordCount Project	1 hour ago
README.md	add WordCount Project	1 hour ago
pom.xml	add WordCount Project	1 hour ago

WordCount1.0

- 在BDKIT创建集群并上传任务

hy171098175-cluster

Status: ● 就绪
CreationTime: 2020-10-27 09:44:18

SlaveNum: 1

端口

Name	Port	Node Port	操作
ssh	22	32130	访问
hdfs-client	9000	32575	访问
resource-manager	8088	31559	访问

- 将input文件上传至HDFS（此处踩坑）

第一次先在VS Code中terminal进行文件复制，发现报错无NameNode可用，运行jps发现没有启动节点。。但SSH终端的节点是可用的

解决方案：助教小哥哥解释vscode只和真正的master共享代码文件，其运行是在不同的pod上的，直接在集群终端运行就可以了（菜鸡落泪

```

root@hy171098175-master:~# jps
256 JobHistoryServer
178 ResourceManager
579 Master
83 NameNode
693 Jps
488 HistoryServer
root@hy171098175-master:~# ls
hdfs run-wordcount.sh start-hadoop.sh start.sh stop.sh
root@hy171098175-master:~# cd workspace
bash: cd: workspace: No such file or directory
root@hy171098175-master:~# cd /workspace
root@hy171098175-master:/workspace# ls
FBDP dataset1 spark-events
root@hy171098175-master:/workspace# ls
FBDP dataset1 spark-events
root@hy171098175-master:/workspace# cd FBDP
root@hy171098175-master:/workspace/FBDP# ls
README.md WordCount
root@hy171098175-master:/workspace/FBDP# cd WordCount
root@hy171098175-master:/workspace/FBDP/WordCount# ls
README.md input pom.xml src target
root@hy171098175-master:/workspace/FBDP/WordCount# cd ..
root@hy171098175-master:/workspace/FBDP# hdfs dfs -mkdir /wordcount
root@hy171098175-master:/workspace/FBDP# cd WordCount
root@hy171098175-master:/workspace/FBDP/WordCount# hdfs dfs -put input/ /wordcount
root@hy171098175-master:/workspace/FBDP/WordCount# hdfs dfs -ls /wordcount/input
Found 1 items
-rw-r--r--  2 root supergroup    9767967 2020-10-27 02:51 /wordcount/input/wordcount.txt
root@hy171098175-master:/workspace/FBDP/WordCount#

```

- hadoop运行jar包再次报错


```

./0-SNAPSHOT.jar /wordcount/input /wordcount/output hadoop jar target/WordCount-1
Exception in thread "main" java.lang.UnsupportedClassVersionError: hyj/nju/edu/cn/WordCount has been compiled by a more recent version of the Java Runtime (class file version 58.0), this version of the Java Runtime only recognizes class file versions up to 52.0

```

上网搜索发现是由于使用了java14（class file version 58）编译，但是只能运行java8（class file version 52），于是修改pom中指定的java版本重新编译。。

- 重新修改java版本后重复上述操作，成功！



FINISHED Applications Logged in as: dr.who

- Cluster
- About
- Nodes
- Node Labels
- Applications
- NEW
- SAVING
- SUBMITTED
- ACCEPTED
- RUNNING
- FINISHED
- FAILED
- KILLED
- Scheduler
- Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
1	0	0	1	0	0 B	16 GB	0 B	0	16	0	2	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:8>

Show 20 entries

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI	Blacklisted Nodes
application_1603766386737_0001	root	word count 1.0	MAPREDUCE	default	Tue Oct 27 13:06:24 +0800 2020	Tue Oct 27 13:06:39 +0800 2020	FINISHED	SUCCEEDED		History	N/A

Showing 1 to 1 of 1 entries First Previous 1 Next Last

- 查看hdfs中的文件，将output同步至git仓库

仓库中的output就是最后的结果啦~（此处截图前十名）

65448 lines (65448 sloc)		690 KB
1	&	101
2	'	12
3	'Gamut'	2
4	'Od's	2
5	'Tis	4
6	'tis	2
7	'twas	2
8	'--O	2
9	'?	1
10	'A	32

- 可以看到wordcount1.0没有对标点符号和停词进行优化，因此在wordcount2.0中进行下一步调整

WordCount2.0

- 优化实现目标：
 - 忽略标点符号、大小写区分\数字及长度<3的单词
 - 返回排名前100的单词
- 首先修改pom文件为WordCount2.0 (hhhhh)

```
<project xmlns="http://maven.apache.org/POM
  xsi:schemaLocation="http://maven.apache.o
  <modelVersion>4.0.0</modelVersion>

  <groupId>hyy.nju.edu.cn</groupId>
  <artifactId>WordCount</artifactId>
  <version>2.0-SNAPSHOT</version>
  <packaging>jar</packaging>
```

- 最开始想到的分词方式即利用split方法先对文本进行分词，再replace全部标点符号并进行大小写转化，但split处理后的字符串存在String[]中很明显十分占用内存

试图优化：

- 优化1.0：发现示例代码中的分词使用的是StringTokenizer（上网查阅发现StringTokenizer的分隔符不需要使用转义字符，且StringTokenizer本身就是不支持正则表达式的），所以准备在分词之前对文本进行预处理，先用空格替代全部标点符号和数字后统一大小写，再利用StringTokenizer对文本文件进行分词处理，但需要把标点符号和数字以字符串的形式存储，考虑到停词需要读入文件，于是继续寻求可以在mapper中读入文件的接口和方式，参考hadoop给出的[mapreduce tutorial](#)（果然还是官方文档yyds），可以先将标点符号文件和停词文件读入DistributedCache class，具体操作如下
- 获取停词和标点符号文件的路径（需要先上传至HDFS），将其增加到Cache中


```
job.addCacheFile(new Path(remainingArgs[++i]).toUri());
```
- 在Mapper中使用 `.getCacheFiles()` 得到cache中的文件

在此处偷懒将标点符号和停词作为一个文件读入StopWords中了

下面实现reducer

- 查阅网上资料发现可以通过两个MapReduce过程实现排序，第一个reduce对所有单词及出现次数进行键值对的汇总；第二个mapper中实现键值的交换，reducer中实现排序；
- 第二步中，mapper通过InverseMapper实现键值对的交换，reducer通过重写setSortComparatorClass实现倒排，最后因为要进行全部键值对的排序，因此将reduce节点数设为1 setNumReduceTasks(1)

MapReduce的Reduce阶段会按照key-value对中的key进行排序，如果key为封装int的IntWritable类型，那么MapReduce按照数字大小对key排序，如果key为封装为String的Text类型，那么MapReduce按照字典顺序对字符串排序。

IntWritable的默认排序机制通过compare方法实现，默认升序排列，源码如下

```
public static class Comparator extends WritableComparator {
    public Comparator() {
        super(IntWritable.class);
    }
    public int compare(byte[] b1, int s1, int l1, byte[] b2, int s2, int l2) {
        int thisValue = readInt(b1, s1);
        int thatValue = readInt(b2, s2);
        return thisValue < thatValue ? -1 : (thisValue == thatValue ? 0 : 1);
    }
}
```

若要实现降序排列，就必须通过重载机制重写compare方法，重写之后通过Job的setSortComparatorClass为计算任务指定排序类，即可实现。

```
static class IntWritableDescComparator extends IntWritable.Comparator {
    public int compare(WritableComparable a, WritableComparable b) {
        return -super.compare(a, b);
    }
    public int compare(byte[] b1, int s1, int l1, byte[] b2, int s2, int l2) {
        return -super.compare(b1, s1, l1, b2, s2, l2);
    }
}
```

Quote: [MapReduce应用案例4: 简单数据倒排](#)

- 出现的问题：报错如下

```
20/10/27 16:14:19 INFO mapreduce.Job: Task Id : attempt_1603766386737_0005_m_000000_2, Status : FAILED
Error: java.lang.RuntimeException: java.lang.NoSuchMethodException: hyy.nju.edu.cn.WordCount$CountSumReducer.<init>
    at org.apache.hadoop.util.ReflectionUtils.newInstance(ReflectionUtils.java:134)
    at org.apache.hadoop.mapred.Task$NewCombinerRunner.combine(Task.java:1678)
    at org.apache.hadoop.mapred.MapTask$MapOutputBuffer.sortAndSpill(MapTask.java:1637)
    at org.apache.hadoop.mapred.MapTask$MapOutputBuffer.flush(MapTask.java:1489)
    at org.apache.hadoop.mapred.MapTask$NewOutputCollector.close(MapTask.java:723)
    at org.apache.hadoop.mapred.MapTask.runNewMapper(MapTask.java:793)
    at org.apache.hadoop.mapred.MapTask.run(MapTask.java:341)
    at org.apache.hadoop.mapred.YarnChild$2.run(YarnChild.java:164)
    at java.security.AccessController.doPrivileged(Native Method)
    at javax.security.auth.Subject.doAs(Subject.java:422)
    at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1657)
    at org.apache.hadoop.mapred.YarnChild.main(YarnChild.java:158)
Caused by: java.lang.NoSuchMethodException: hyy.nju.edu.cn.WordCount$CountSumReducer.<init>()
    at java.lang.Class.getConstructor0(Class.java:3082)
    at java.lang.Class.getDeclaredConstructor(Class.java:2178)
    at org.apache.hadoop.util.ReflectionUtils.newInstance(ReflectionUtils.java:128)
    ... 11 more
```


解决方案：mapper和reducer的类都需要设置为static才能调用啊...(流下Java没学好的泪水)

- 最终得到的output如下：

58507 lines (58507 sloc) 625 KB		
1	51553	the
2	44756	and
3	18822	you
4	17762	that
5	13366	with
6	13027	for
7	12875	not
8	12032	your
9	11659	his
10	10263	this
11	10226	scene
12	10079	but
13	10068	have
14	8659	thou
15	8054	will
16	6646	what
17	6588	thy
18	6163	shall
19	5868	all
20	5696	are
21	5439	him
22	5432	our
23	5385	her
24	4751	king
25	4631	good

两个问题需要继续解决：

- 停词似乎没有读入 -- 因为input的class中会自动对文件夹内的文件进行解析而stopwords不会，因此-skip之后的文件路径要一直精确到.txt
- 需要保证最终结果只输出前100 --在排序后重新定义reducer并通过设置全局变量进行rank
- 输出格式应该是<排名><单词><出现次数> --将排名作为最终输出的key，后两项合并为字符串作为value

```

/**放一下参考代码
    * reduce将输入中的key复制到输出数据的key上，然后根据输入的value-list中元素的个数决定
key的输出次数
    * 用全局linenum来代表key的序 ---可以用于生成排名
    */
    public static class SortReducer extends Reducer<IntWritable, IntWritable,
IntWritable, IntWritable> {
        private static IntWritable linenum = new IntWritable(1);
        public void reduce(IntWritable key, Iterable<IntWritable> values,
Context context)
            throws IOException, InterruptedException {
            for (IntWritable val : values) {
                context.write(linenum, key);
                linenum = new IntWritable(linenum.get() + 1);
            }
        }
    }
}
/* 最后的reducer代码

```

- 最终输出结果: [完整project地址](#)

100 lines (100 sloc) | 2.39 KB

1	1	word:scene times:10241
2	2	word:thou times:9438
3	3	word:thy times:6592
4	4	word:shall times:6398
5	5	word:king times:6254
6	6	word:lord times:5702
7	7	word:sir times:5530
8	8	word:thee times:5381
9	9	word:good times:5123
10	10	word:come times:4473
11	11	word:enter times:4252
12	12	word:act times:4109
13	13	word:let times:4084
14	14	word:love times:3596
15	15	word:man times:3565
16	16	word:hath times:3379
17	17	word:like times:3346
18	18	word:henry times:3304
19	19	word:say times:3057
20	20	word:know times:3028
21	21	word:make times:2891
22	22	word:did times:2844
23	23	word:shakespeare times:2664
24	24	word:homepage times:2652
25	25	word:previous times:2458
26	26	word:tis times:2406
27	27	word:duke times:2260
28	28	word:speak times:2063
29	29	word:tell times:1918
30	30	word:father times:1909
31	31	word:think times:1890
32	32	word:exeunt times:1864
--	--	-----

特别鸣谢:

我的集群 (截图留念hh)

集群



hy171098175-cluster

Status: ● 就绪 ⓘ

CreationTime: 2020-10-27 10:37:55

SlaveNum: 2



Quote: [\[Maven实战-许晓斌\]-\[第三章\] Maven使用入门](#)

[\[Hadoop基础学习（一）分析、编写并执行WordCount词频统计程序\]](#)
(<https://my.oschina.net/u/4311561/blog/3552054>)

[MapReduce Tutorial](#)

[MapReduce-WordCount实现按照value降序排序、字符小写、识别不同标点](#)

[7个实例全面掌握Hadoop MapReduce](#)

[执行mapreduce报错java.lang.Exception: java.lang.RuntimeException:
java.lang.NoSuchMethodException:](#)

[MapReduce应用案例4：简单数据倒排](#)