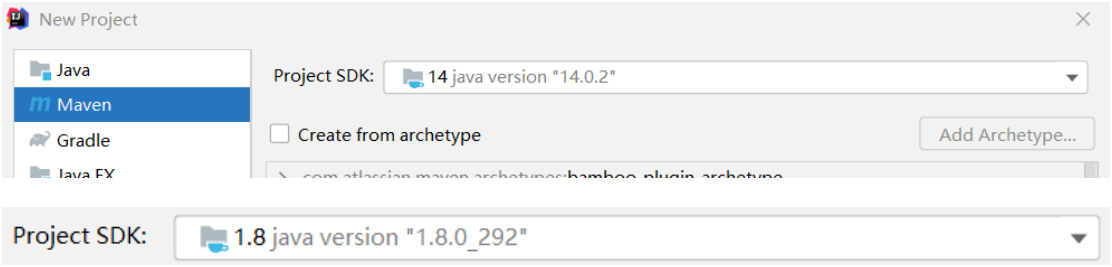
1、打开 IDEA，创建一个新的项目，在引导界面左侧选择 maven，java 需要选择 1.8 的版本，而不能选择 jdk14，否则会出现问题。因为需要在 linux 虚拟机中运行，所以需要选择可以与 hadoop 版本兼容的 java。而且 hadoop 命令运行的是一个 jar 包，所以需要使用 maven 将编写好的程序最后根据要求打成 jar 包，然后放到 linux 上面进行调试。



2、之后给项目命名为 WordCountTopN。

3、之后创建的项目当中，在左侧资源栏里面相对应 project 名字文件夹下会产生默认的 pom.xml 文件。将 groupid 与 artifactid 改成想要的名称。资源路径不用改变。

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>


    <groupId>dblab</groupId>
    <artifactId>WordCountTopN</artifactId>
    <version>1.0-SNAPSHOT</version>

</project>
```

4、由于 jar 包使用 hadoop 命令运行，且 IDEA 运行在 windows 环境下没有办法直接使用虚拟机中安装的 hadoop 包，所以需要添加运行 hadoop 命令的一些包，

提供给java程序可以调用的依赖和一些hadoop方法。否则没有办法使用hadoop分布式计算平台进行数据处理。一种方式是可以直接把虚拟机中的一些文件拷贝过来，作为 libraries 引入，但是比较麻烦。这里选择直接配置 pom 文件，利用IDEA 提供的功能自动 import 并解析所需资源与依赖。首先添加代码指明 hadoop版本。

```
<properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <hadoop.version>2.6.0</hadoop.version>
</properties>
```

5、添加所需依赖。

```
<dependencies>
    <!-- Hadoop -->
    <dependency>
        <groupId>org.apache.hadoop</groupId>
        <artifactId>hadoop-common</artifactId>
        <version>${hadoop.version}</version>
    </dependency>
    <dependency>
        <groupId>org.apache.hadoop</groupId>
        <artifactId>hadoop-hdfs</artifactId>
        <version>${hadoop.version}</version>
    </dependency>
    <dependency>
        <groupId>org.apache.hadoop</groupId>
        <artifactId>hadoop-mapreduce-client-core</artifactId>
        <version>${hadoop.version}</version>
    </dependency>
    <dependency>
        <groupId>org.apache.hadoop</groupId>
        <artifactId>hadoop-mapreduce-client-jobclient</artifactId>
        <version>${hadoop.version}</version>
    </dependency>
</dependencies>
```

6、给 pom 文件添加打 jar 包的功能，即 build 选项。选择 java1.8，同时指明使用的编译器 maven-compiler-plugin 的版本号。

```xml
    <build>
        <plugins>
            <!-- Java 1.8 -->
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.6.0</version>
                <configuration>
                    <source>1.8</source>
                    <target>1.8</target>
                </configuration>
            </plugin>
        </plugins>
    </build>
```

7、在 src 中添加 WordCountTopN 类，类中设置主函数。首先添加 MyMapper 类，以实现将文章分割成一个个单词的功能，这个类实现功能与 wordcount 中一样，可以拷贝代码。

```java
public static class MyMapper extends Mapper<LongWritable, Text, Text, IntWritable>
{

    private final IntWritable ONE = new IntWritable( value: 1);
    private Text outkey = new Text();
    private String[] infos;

    @Override
    protected void map(LongWritable key, Text value, Mapper<LongWritable, Text, Text, In
            throws IOException, InterruptedException {
        infos = value.toString().split( regex: " ");
        for (String word : infos) {
            outkey.set(word);
            context.write(outkey, ONE);
        }
    }

}
```

8、添加类 MyReducer，其实现的功能与单纯的 wordcount 有一定的区别。他除了将单个的同组单词组合起来写入内存中之外，还添加了一个判断的功能。在获取了参数 N 之后，如果内存空间 topN 中不满 N 个元素的话，可以直接往里面添加新的 key；如果 key 值有相同的，reducer 负责合并；如果有>=N 个不同的 key，先放进去一个，然后删除掉一个。由于使用了 treeMap，一个按照 key 进行排序排序的 map，所以可以始终保证删去最后一个后仅仅剩下 TopN 个。

```
public static class MyReducer extends Reducer<Text, IntWritable, Text, IntWrita 10 3
{

    private int sum;
    private Text outkey = new Text();
    private IntWritable outvalue = new IntWritable();
    // 开辟内存空间保存topN
    // treeMap是一个排序的map.按照key进行排序，这样不用认为排序，方便topN实现
    private TreeMap<Integer, String> TopN = new TreeMap<Integer, String>();

    @Override
    protected void reduce(Text key, Iterable<IntWritable> values, Reducer<Text, IntWrit
        sum = 0;
        for (IntWritable value : values) {
            sum += value.get();
        }
        // 把计算结果放入topN
        // 如果topN中不满N个元素的话,可以直接往里面添加
        // 先看topN中有没有相同的key,如果有吧topN中相同key对相应的value单词穿在一起,如果没有的话,就直
        if (TopN.size() < context.getConfiguration().getInt( name: "N",  defaultValue: 10)) {
            if (TopN.get(sum) != null) {
                TopN.put(sum, TopN.get(sum) + "-----" + key.toString());
            } else {
                TopN.put(sum, key.toString());
            }
        } else {
            // 如果有》=N个不同的key,放进去一个,然后删除掉一个
            // 始终保持topN中有N个元素
            if (TopN.get(sum) != null) {
                TopN.put(sum, TopN.get(sum) + "------" + key.toString());
                // 因为有同key,是归并操作,因此没有增也不用删

            } else {
                TopN.remove(TopN.firstKey());
                TopN.put(sum, key.toString());
                //TopN.remove(TopN.lastKey());

            }

        }
        // 放进去后treemap会自动排序,这时候把最后一个删掉,保证topN中只有n
    }
```

9、添加 clean up 函数，可以拷贝 wordcount 的代码。

```
protected void cleanup(Reducer<Text, IntWritable, Text, IntWritable>.Context context) thr
    if (TopN != null && !TopN.isEmpty()) {
        Set<Integer> keys = TopN.keySet();
        for (Integer key : keys) {
            outkey.set(TopN.get(key));
            outvalue.set(key);
            context.write(outkey, outvalue);
        }
    }
}
```

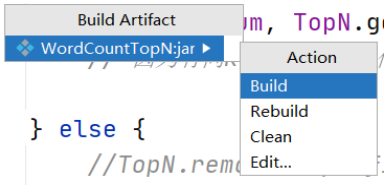
10、在 main 函数中，需要将传递的参数 N 拿出来放入 conf 当中，方便其他函数使用的时候拿取。

```
public static void main (String args[])throws Exception
{
    System.out.println("HADOOP_TEST_WordCountTopN");
    Configuration conf = new Configuration();
    String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();

    conf.setInt( name: "N", new Integer(otherArgs[0]));

    if(otherArgs.length < 3){

        System.err.println("error_input");

        System.exit( status: 2);

    }
```

11、设置好刚刚编写的 map 类与 reduce 类，传入对应的 job.set()中，并且设置好用户传入的输入位置与输出位置，这个可以参考 wordcount 代码。这里考虑了多输入情况，使得 WordCountTopN 适用于多输入。

```
113         Job job = Job.getInstance(conf, jobName: "WordCountTopN");
114         job.setJarByClass(WordCountTopN.class);
115         job.setMapperClass(MyMapper.class);
116         job.setCombinerClass(MyReducer.class);
117         job.setMapOutputKeyClass(Text.class);
118         job.setMapOutputValueClass(IntWritable.class);
119
120         job.setReducerClass(MyReducer.class);
121         job.setOutputKeyClass(NullWritable.class);
122         job.setOutputValueClass(IntWritable.class);
123
124         for (int i = 1; i < otherArgs.length-1; i++) {
125             FileInputFormat.addInputPath(job, new Path(otherArgs[i]));
126         }
127
128         FileOutputFormat.setOutputPath(job, new Path(otherArgs[otherArgs.length-1]));
129
130         System.exit(job.waitForCompletion( verbose: true) ? 0 : 1);
```

12、点击 build，build artifact，然后选择 build，等待提示成功后就可以在 out 文件夹里找到 jar。



13、将 jar 包放入虚拟机，使用 hadoop 命令运行。由于在打包的时候没有设定主类，在运行时需要申明入口类 WordCountTopN。

```
prestyan@hadoop-master:/opt/hadoop/sbin$ hadoop jar ~/WordCountTopN.jar WordCoun
tTopN 5 /input/passage /output-wdcount/test4
```

14、运行成功。

```
22/03/30 20:38:32 INFO mapreduce.Job:  map 0% reduce 0%
22/03/30 20:38:37 INFO mapreduce.Job:  map 100% reduce 0%
22/03/30 20:38:43 INFO mapreduce.Job:  map 100% reduce 100%
22/03/30 20:38:44 INFO mapreduce.Job: Job job_1648643519950_0002 completed successfully
22/03/30 20:38:44 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=66
                FILE: Number of bytes written=215947
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=3558
                HDFS: Number of bytes written=45
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=2219
                Total time spent by all reduces in occupied slots (ms)=3131
                Total time spent by all map tasks (ms)=2219
                Total time spent by all reduce tasks (ms)=3131
                Total vcore-milliseconds taken by all map tasks=2219
                Total vcore-milliseconds taken by all reduce tasks=3131
                Total megabyte-milliseconds taken by all map tasks=4544512
                Total megabyte-milliseconds taken by all reduce tasks=6412288
        Map-Reduce Framework
                Map input records=45
                Map output records=584
                Map output bytes=5747
                Map output materialized bytes=66
                Input split bytes=105
                Combine input records=584
                Combine output records=5
                Reduce input groups=5
                Reduce shuffle bytes=66
                Reduce input records=5
                Reduce output records=5
                Spilled Records=10
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=139
                CPU time spent (ms)=1250
                Physical memory (bytes) snapshot=391921664
                Virtual memory (bytes) snapshot=4248604672
                Total committed heap usage (bytes)=242745344
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=3453
        File Output Format Counters
                Bytes Written=45
prestyan@hadoop-master:/opt/hadoop/sbin$
```

15、查看输出结果，可以看到当 N=5、10 的时候，分别输出了 Top5、Top10 个统计个数，说明 WordCountTopN 达到了目标要求。

```
prestyan@hadoop-master:/opt/hadoop/sbin$ hdfs dfs -cat /output-wdcount/test4/part-r-00000
to------was     11
Allison 13
and     14
a       18
the     36
```

```
prestyan@hadoop-master:/opt/hadoop/sbin$ hdfs dfs -cat /output-wdcount/test5/part-r-00000
on------she------with   5
her------said.  6
He------in------of      7
Clark   8
She     9
to------was     11
Allison 13
and     14
a       18
the     36
```

16、改动代码中删除 key 的位置，还可以实现 WordCountBottomN。

```
---evening------exhausted24.------expected------extremely------faces.------facia
l------far------features.------few------fingers.------fixed23------flames.------
floorboards.------fluttering------fool,"------foot.------foyer,------funniest,--
----garage------garish10------gift------glider6,------good------greater------gui
de,------guts22.------had,------hell,"------her,------her.------his,------holdin
g------hood8------house.------husband,------if------included------inside------in
to------is------its------jack-o-lanterns.------just------kicked------kids."-----
-knees------lap.------last------leaves------let------letter------lids------like-
-----liked------limping------line------list------lit------lit,"------little.----
--locked.------long------look------lowered------maid's------mail:------married.-
-----me------me,"------meant------mess.------mine,"------missed------moment-----
-moon------more.------morning.------most------moving------my------natural-hair--
----new------newspapers.------next------night,------nothing------nothing.------n
ow.------off,"------old------older------older-seventy-eight------once------one--
----opened,------orange------over------own------packet,------padded------papers-
-----pay-TV------perspective------phone------plaguing------porch.------predicted
,------program------pulse------pumpkin2,------pumpkins------pumpkins.------pumpk
ins3------pushed------quickly------quite------railing.------raised------read,---
---reeked21------regular------relations------rest------results------round------r
ow------sat------season------see------settled,------shawl.------she,------signed
------slippered7------smaller------smell------speaking------special------spend--
----squishy------stayed.------stood------struggled------supper.------surreal,"--
----sweetly------talent,------tall------telephone,------tell------them.------the
n------they're------thick------thing,------thing;------thirty-five.------through
------time,"------today.------together------tomorrow."------took------trees-----
-twig-and-leaf-littered------twilight4------unbothered.------uncashable,------un
der------unkind------until------usually,------view------vigil------volunteered--
----warm.------was,------weeks------weight------when------which------while------
white------who------wide------wife------wig------wig5.------wool------worst-----
-wrong.------yellow------you------yourself      1
"Don't-----It------been------candle------candles.------each------face.------foun
d------get------he------him.------jack-o-lanterns------looked------much,------ni
ght------only------out------porch------pumpkin------put------said------something
------their------time,------told------too------watched      2
"You're-----Clark's-----The-----They------are------away------back------being----
--for------his------it------little------much------than------that------wanted----
--went------were------wore      3
an------at------from------had------up   4
on------she------with   5
her------said.  6
He------in------of      7
Clark   8
She     9
to------was     11
```

代码另附文件：WorCountTopN