Exercise 04: Map Reduce applications for Word Counting

Previous exercise described how to count repeated words in the input file. This exercise practice the students to do MapReduce process using word counting application with elimination words.

Prerequisites

Ensure that Hadoop is installed, configured and is running. More

details: Single Node Setup for first-time users.

Cluster Setup for large, distributed clusters.

Inputs and Outputs

i. **Input file should be in : /wcsw/in00/**

data.txt

Copy the content text from Shakespeare.txt, Which is attached in Google classroom.

sw.txt

Add following elimination words into sw.txt file.

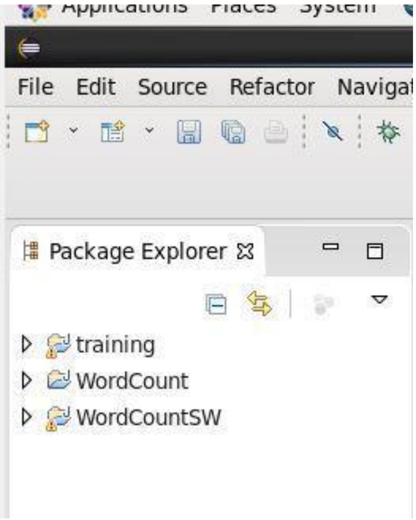
all
is
the
our
I
It

ii. **Output file should be in /wcsw/out00/**

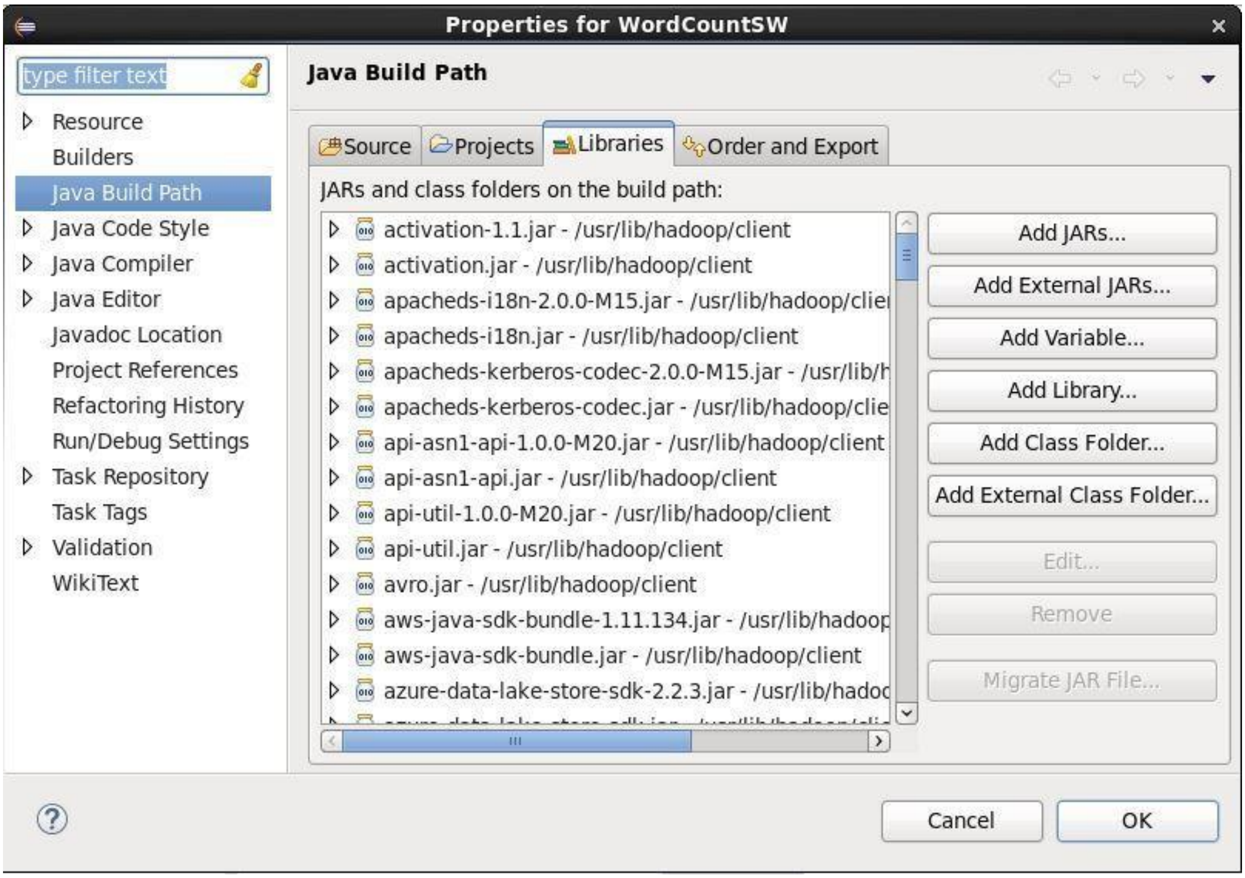
Step 1

Compile `WordCountSW.java` and create a WordCountSW.jar:

(i) Create WordCountSW.java project.



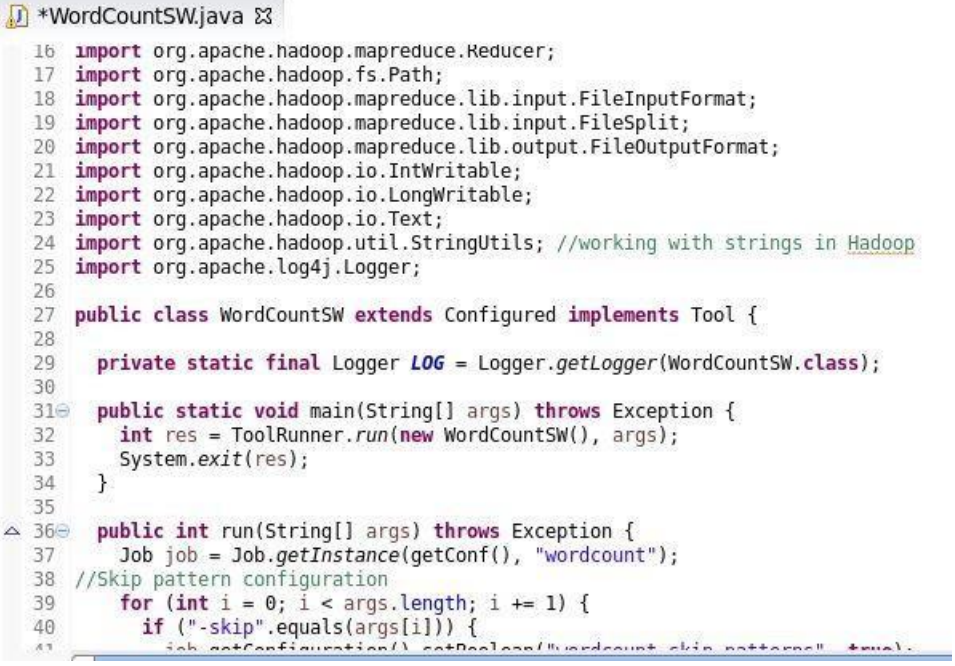
(ii) Import external .jar files



JARs and class folders on the build path:

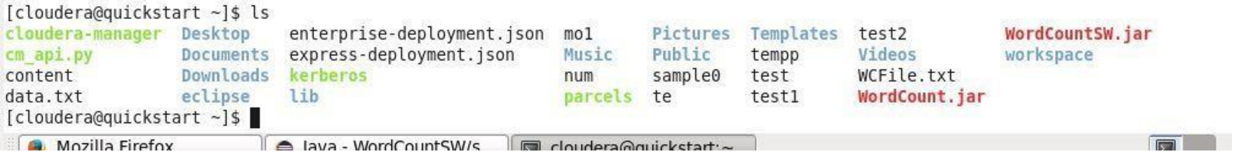
```
activation-1.1.jar - /usr/lib/hadoop/client
activation.jar - /usr/lib/hadoop/client
apacheds-i18n-2.0.0-M15.jar - /usr/lib/hadoop/clie
apacheds-i18n.jar - /usr/lib/hadoop/client
apacheds-kerberos-codec-2.0.0-M15.jar - /usr/lib/h
apacheds-kerberos-codec.jar - /usr/lib/hadoop/clie
api-asn1-api-1.0.0-M20.jar - /usr/lib/hadoop/client
api-asn1-api.jar - /usr/lib/hadoop/client
api-util-1.0.0-M20.jar - /usr/lib/hadoop/client
api-util.jar - /usr/lib/hadoop/client
avro.jar - /usr/lib/hadoop/client
aws-java-sdk-bundle-1.11.134.jar - /usr/lib/hadoop
aws-java-sdk-bundle.jar - /usr/lib/hadoop/client
azure-data-lake-store-sdk-2.2.3.jar - /usr/lib/hadoo
```

(iii) Create WordCount class file using Google classroom attached WordCount.java file.



```java
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.FileSplit;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.util.StringUtils; //working with strings in Hadoop
import org.apache.log4j.Logger;

public class WordCountSW extends Configured implements Tool {

  private static final Logger LOG = Logger.getLogger(WordCountSW.class);

  public static void main(String[] args) throws Exception {
    int res = ToolRunner.run(new WordCountSW(), args);
    System.exit(res);
  }

  public int run(String[] args) throws Exception {
    Job job = Job.getInstance(getConf(), "wordcount");
//Skip pattern configuration
    for (int i = 0; i < args.length; i += 1) {
      if ("-skip".equals(args[i])) {
        job.getConfiguration().setBoolean("wordcount.skip.patterns", true);
```

(iv) Create WordCountSW.jar file

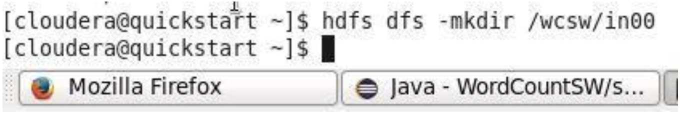


```
[cloudera@quickstart ~]$ ls
cloudera-manager  Desktop    enterprise-deployment.json  mo1      Pictures  Templates  test2       WordCountSW.jar
cm_api.py         Documents  express-deployment.json     Music    Public    tempp      Videos      workspace
content           Downloads  kerberos                    num      sample0   test       WCFile.txt
data.txt          eclipse    lib                         parcels  te        test1      WordCount.jar
[cloudera@quickstart ~]$ 
```

Step 2

Create following folders in HDFS:

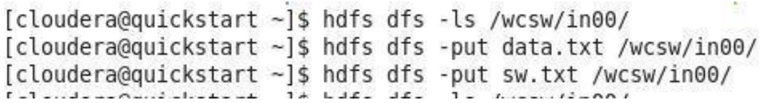
- `/wcsw/in00` - input directory in HDFS
- `/wcsw/out00` - output directory in HDFS



```
[cloudera@quickstart ~]$ hdfs dfs -mkdir /wcsw/in00
[cloudera@quickstart ~]$ █
```

Step 3

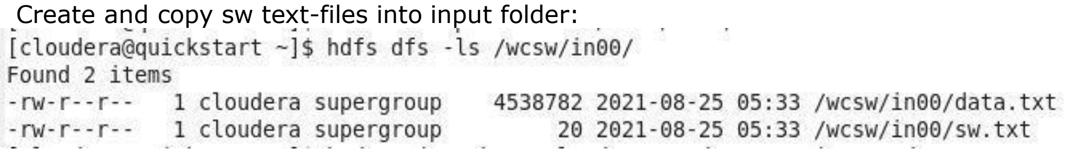
Create and copy data text-files into input folder:



```
[cloudera@quickstart ~]$ hdfs dfs -ls /wcsw/in00/
[cloudera@quickstart ~]$ hdfs dfs -put data.txt /wcsw/in00/
[cloudera@quickstart ~]$ hdfs dfs -put sw.txt /wcsw/in00/
```

Step 4

Create and copy sw text-files into input folder:



```
[cloudera@quickstart ~]$ hdfs dfs -ls /wcsw/in00/
Found 2 items
-rw-r--r--   1 cloudera supergroup    4538782 2021-08-25 05:33 /wcsw/in00/data.txt
-rw-r--r--   1 cloudera supergroup         20 2021-08-25 05:33 /wcsw/in00/sw.txt
```

[cloudera@quickstart ~]$ hdfs dfs -ls

/wcsw/in00/ Found 2 items

-rw-r--r-- 1 cloudera supergroup 3309 2021-08-24 07:00 /wcsw/in00/data.txt

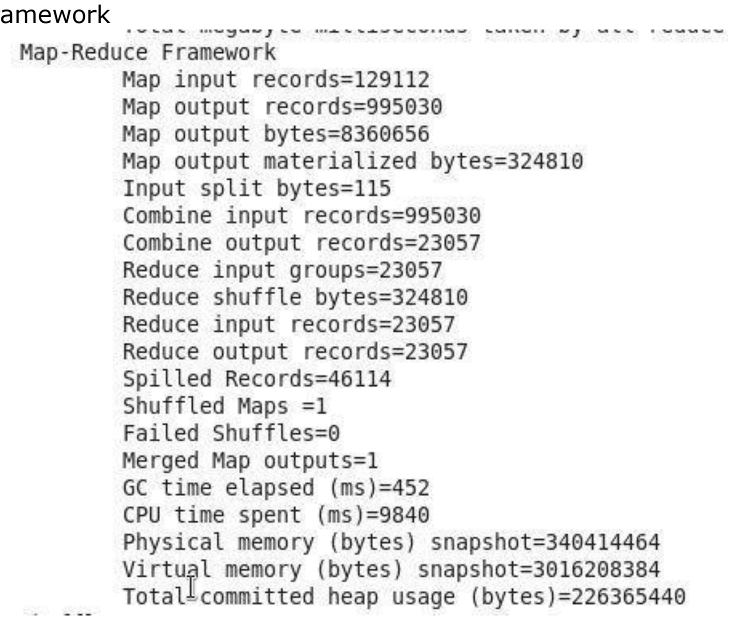
-rw-r--r-- 1 cloudera supergroup 15 2021-08-24 07:02 /wcsw/in00/sw.txt

Step 5

Run the MapReduce application with skip option:

[cloudera@quickstart ~]$ hadoop jar /home/cloudera/WordCountSW.jar /wcsw/in00/data.txt /wcsw/out00/ -skip /wcsw/in00/sw.txt

Show MapReduce Framework

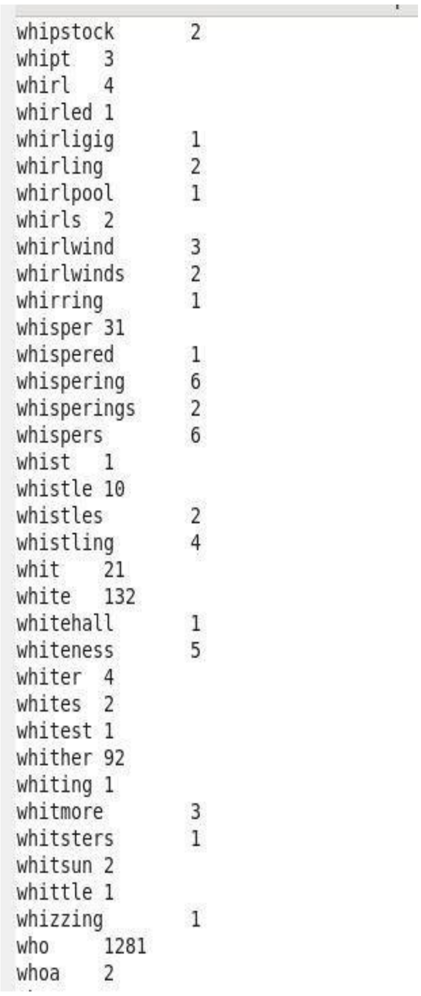


```
Map-Reduce Framework
        Map input records=129112
        Map output records=995030
        Map output bytes=8360656
        Map output materialized bytes=324810
        Input split bytes=115
        Combine input records=995030
        Combine output records=23057
        Reduce input groups=23057
        Reduce shuffle bytes=324810
        Reduce input records=23057
        Reduce output records=23057
        Spilled Records=46114
        Shuffled Maps =1
        Failed Shuffles=0
        Merged Map outputs=1
        GC time elapsed (ms)=452
        CPU time spent (ms)=9840
        Physical memory (bytes) snapshot=340414464
        Virtual memory (bytes) snapshot=3016208384
        Total committed heap usage (bytes)=226365440
```

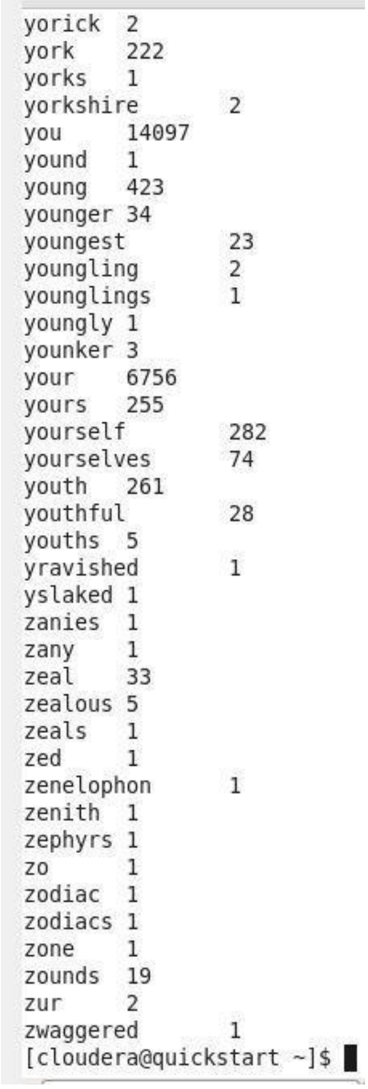
Step 6

Output:

```
[cloudera@quickstart ~]$ hdfs dfs -ls
/wcsw/out00/ Found 2 items
-rw-r--r--   1 cloudera supergroup     0 2021-08-24 07:05 /wcsw/out00/_SUCCESS
-rw-r--r--  1 cloudera supergroup  2384 2021-08-24 07:05 /wcsw/out00/part-r-
00000 [cloudera@quickstart ~]$ hdfs dfs -cat /wcsw/out00/part-r-00000
```



```
whipstock       2
whipt   3
whirl   4
whirled 1
whirligig       1
whirling        2
whirlpool       1
whirls  2
whirlwind       3
whirlwinds      2
whirring        1
whisper 31
whispered       1
whispering      6
whisperings     2
whispers        6
whist   1
whistle 10
whistles        2
whistling       4
whit    21
white   132
whitehall       1
whiteness       5
whiter  4
whites  2
whitest 1
whither 92
whiting 1
whitmore        3
whitsters       1
whitsun 2
whittle 1
whizzing        1
who     1281
whoa    2
```



```
yorick  2
york    222
yorks   1
yorkshire       2
you     14097
yound   1
young   423
younger 34
youngest        23
youngling       2
younglings      1
youngly 1
younker 3
your    6756
yours   255
yourself        282
yourselves      74
youth   261
youthful        28
youths  5
yravished       1
yslaked 1
zanies  1
zany    1
zeal    33
zealous 5
zeals   1
zed     1
zenelophon      1
zenith  1
zephyrs 1
zo      1
zodiac  1
zodiacs 1
zone    1
zounds  19
zur     2
zwaggered       1
[cloudera@quickstart ~]$
```