cd hadoop-2.7.3

bin/hadoop namenode -format

./hadoop-daemon.sh start namenode

./hadoop-daemon.sh start datanode

./yarn-daemon.sh start resourcemanager

./yarn-daemon.sh start nodemanager

./mr-jobhistory-daemon.sh start historyserver

jps

localhost:50070/dfshealth.html

mkdir words

Put that downloaded jar file into words folder.

code

import java.io.IOException;

import java.util.\*;

import org.apache.hadoop.conf.\*;

import org.apache.hadoop.fs.\*;

import org.apache.hadoop.io.\*;

import org.apache.hadoop.mapreduce.\*;

import org.apache.hadoop.mapreduce.lib.input.\*;

import org.apache.hadoop.mapreduce.lib.output.\*;

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

public class WordCount extends Configured implements Tool

{

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

{

int res = ToolRunner.run(new WordCount(), args);

System.exit(res);

}

public int run(String[] args) throws Exception

{

Path inputPath = new Path(args[0]);

Path outputPath = new Path(args[1]);

Configuration conf = getConf();

Job job = new Job(conf, this.getClass().toString());

job.setJarByClass(WordCount.class);

FileInputFormat.setInputPaths(job, inputPath);

FileOutputFormat.setOutputPath(job, outputPath);

job.setJobName("WordCount");

job.setMapperClass(Map.class);

job.setCombinerClass(Reduce.class);

job.setReducerClass(Reduce.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(IntWritable.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

return job.waitForCompletion(true) ? 0 : 1;

}

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

{

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

private Text word = new Text();

@Override

public void map(LongWritable key, Text value, Mapper.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 Reduce extends Reducer<Text, IntWritable, Text, IntWritable>

{

@Override

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

{

int sum = 0;

for(IntWritable value : values)

{

sum += value.get();

}

context.write(key, new IntWritable(sum));

}

}

}

Create input1.txt on home directory with some random text

javac -classpath /home/vijay/words/hadoop-core-1.2.1.jar /home/vijay/words/WordCount.java

Step 7) jar -cvf words.jar -C words/ .

Step 8) cd .. then use following commands

hadoop fs -mkdir /input

hadoop fs -put input1.txt /input

hadoop fs -ls /input

hadoop jar /home/vijay/words/words12.jar WordCount /input/input1.txt /out321

hadoop fs -ls /out321

hadoop fs -cat /out321/part-r-00000