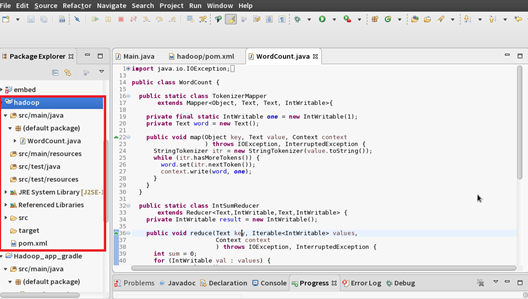
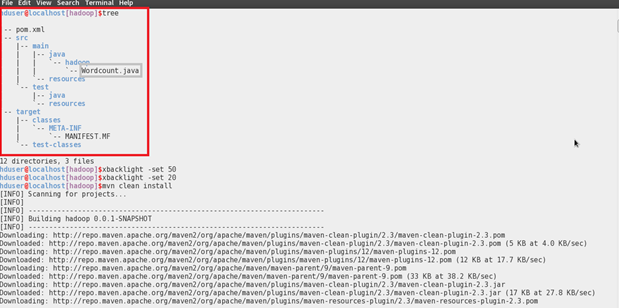
**QUE-15.4- Build and Deploy hadoop word count program using Maven.**

**ANS-15.4-**

**Build and Deploy Hadoop word count program using Maven-**



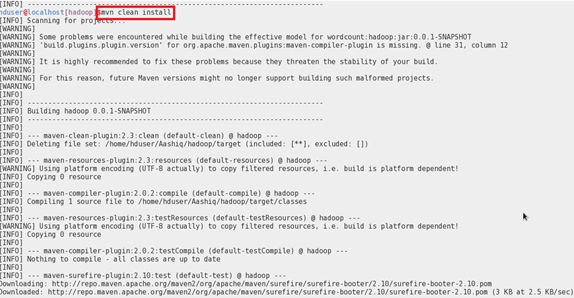
**Tree Diagram- After Creating the maven project the pom.xml file is created**



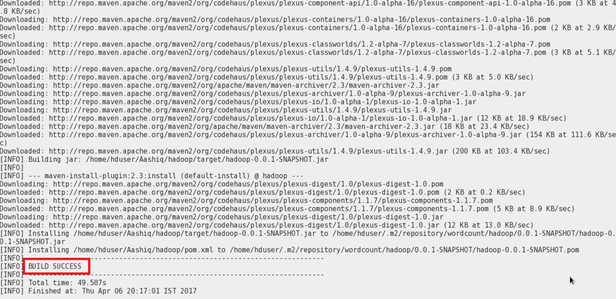
**Adding the dependencies to the Pom.xml-**



**Building the project by giving the mvn clean install in the project folder-**



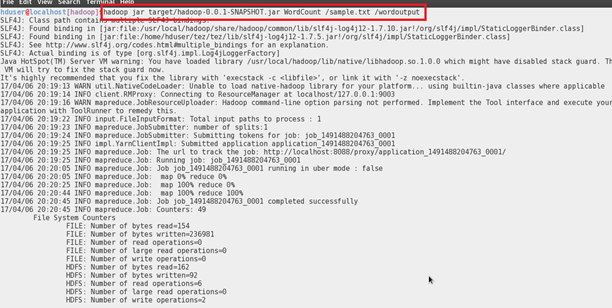
**Project successfully build message-**



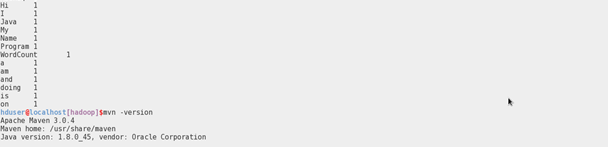
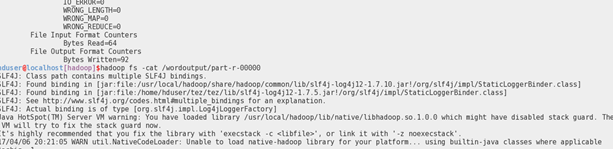
**Snapshot.jar file created after the Build-**



**Running the Snapshot.jar file-**



**WordCount Output-**



**FOR REFERENCE-**

**Pom.xml file-**

<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>wordcount</groupId>

<artifactId>hadoop</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<!-- JDK Tools -->

<dependency>

<groupId>jdk.tools</groupId>

<artifactId>jdk.tools</artifactId>

<scope>system</scope>

<version>1.8.0\_45</version>

<systemPath>${JAVA\_HOME}/lib/tools.jar</systemPath>

</dependency>

<!-- https://mvnrepository.com/artifact/org.apache.hadoop/hadoop-core -->

<dependency>

<groupId>org.apache.hadoop</groupId>

<artifactId>hadoop-core</artifactId>

<version>1.2.1</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.apache.hadoop/hadoop-mapreduce-client-core -->

<dependency>

<groupId>org.apache.hadoop</groupId>

<artifactId>hadoop-mapreduce-client-core</artifactId>

<version>2.7.3</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

</plugins>

</build>

</project>

**JAVA FILE -**

importjava.io.IOException;

importjava.util.StringTokenizer;

importorg.apache.hadoop.conf.Configuration;

importorg.apache.hadoop.fs.Path;

importorg.apache.hadoop.io.IntWritable;

importorg.apache.hadoop.io.Text;

importorg.apache.hadoop.mapreduce.Job;

importorg.apache.hadoop.mapreduce.Mapper;

importorg.apache.hadoop.mapreduce.Reducer;

importorg.apache.hadoop.mapreduce.lib.input.FileInputFormat;

importorg.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class WordCount {

public static class TokenizerMapper

extends Mapper<Object, Text, Text, IntWritable>{

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

private Text word = new Text();

public void map(Object key, Text value, Context context

) throws IOException, InterruptedException {

StringTokenizeritr = new StringTokenizer(value.toString());

while (itr.hasMoreTokens()) {

word.set(itr.nextToken());

context.write(word, one); }

}

}

public static class IntSumReducer

extends Reducer<Text,IntWritable,Text,IntWritable> {

privateIntWritable result = new IntWritable();

public void reduce(Text key, Iterable<IntWritable> values,

Context context

) throws IOException, InterruptedException {

int sum = 0;

for (IntWritableval : values) {

sum += val.get(); }

result.set(sum);

context.write(key, result);

} }

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

Configuration conf = new Configuration();

Job job = Job.getInstance(conf, "word count");

job.setJarByClass(WordCount.class);

job.setMapperClass(TokenizerMapper.class);

job.setCombinerClass(IntSumReducer.class);

job.setReducerClass(IntSumReducer.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

FileInputFormat.addInputPath(job, new Path(args[0]));

FileOutputFormat.setOutputPath(job, new Path(args[1]));

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

}}