Exp:8 WRITE A PROGRAM TO USE THE API’s OF HADOOP TO INTERACT WITH IT

AIM:

To write a java program to use the API’s of hadoop to interact with it.

PROCEDURE:

Step 1:Check hadoop version

```
data@data-HP-Notebook:~$ hadoop version
Hadoop 3.0.3
Source code repository https://yjzhangal@git-wip-us.apache.org/repos/asf/hadoop.git -r
37fd7d752db73d984dc31e0cdfd590d252f5e075
Compiled by yzhang on 2018-05-31T17:12Z
Compiled with protoc 2.5.0
From source with checksum 736cdcefa911261ad56d2d120bf1fa
This command was run using /home/hadoop/hadoop/share/hadoop/common/hadoop-common-3.0.3.jar
```

Step 2:Create directory named itdept.

```
data@data-HP-Notebook:~$ hadoop fs -mkdir /itdept
2018-08-11 08:29:18,291 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your
platform... using builtin-java classes where applicable
```

Step 3:Showing files of hadoop filesystem description.

```
data@data-HP-Notebook:~$ hadoop fs -ls /
2018-08-11 08:30:34,633 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your
platform... using builtin-java classes where applicable
found 3 items
drwxr-xr-x          -hadoop supergroup             0  2018-08-11 06:00 /itdept
drwxr-xr-x          -hadoop supergroup             0  2018-08-11 06:00 /user
```

CREATION OF JAR FILES:

Step 1:

- Creating jar files using netbeans:
- Create new project named as MyHadoopApplication
- Add the following libraries to the application
  i. projects->libraries(right-click)->add jar/folder
  ii. browse to *hadoop*/local/hadoop/lib/hadoop-common-3.0.3.jar
  iii. click ok.

Edit the program as follows:

```
package myhadoopapplication;

import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.FSDataInputStream;
import org.apache.hadoop.fs.FSDataOutputStream;
import org.apache.hadoop.fs.Path;
public class MyHadoopApplication {
    public static final String filename="dummy.txt";
    public static final String message="This is the dummy text for test the write to file operation of HDFS";
    public static void main(String[] args) throws IOException{
        Configuration conf=new Configuration();
        FileSystem fs=FileSystem.get(conf);
        Path filenamePath=new Path(filename);
        try{
            if(fs.exists(filenamePath)){
                fs.delete(filenamePath,true);
            }
            FSDataOutputStream out=fs.create(filenamePath);
            out.writeUTF(message);
            out.close();
            FSDataInputStream in=fs.open(filenamePath);
            String messageIn=in.readUTF();
            System.out.println(messageIn);
            in.close();
            if(fs.exists(filenamePath)){
                Path renameFilenamePath=new Path("renamed_"+filename);
                fs.rename(filenamePath,renameFilenamePath);
            }

        }
        catch(IOException ex){
            System.out.println("Error:"+ex.getMessage());
        }
    }

}
```

Step 2:Create a java arhieve (.jar) file using netbeans as follows:

1. Right click on the project name.
2. Select properties.
3. Click packaging
4. Check Build jar after Compiling.
5. Check Compress JAR File
6. Click ok to accept changes.
7. Right click on a project name.
8. Select build or clean and build.

The JAR file is built with in the project directory.To view it inside Netbeans:

Click the Files tab
Expand Project name>>dist

```
hadoop@data-HP-Notebook:~/NetBeansProjects/MyHadoopApplication/dist$ hadoop jar
MyHadoopApplication.jar

2018-08-11 08:58:00,385 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your
platform... using builtin-java classes where applicable
This is the dummy text for test the write to file operation of HDFS
```

RESULT:

Thus the java program to use the API’s of hadoop to interact with it was executed and output was verified successfully.