

# hadoop shell & java API

## 实验目的

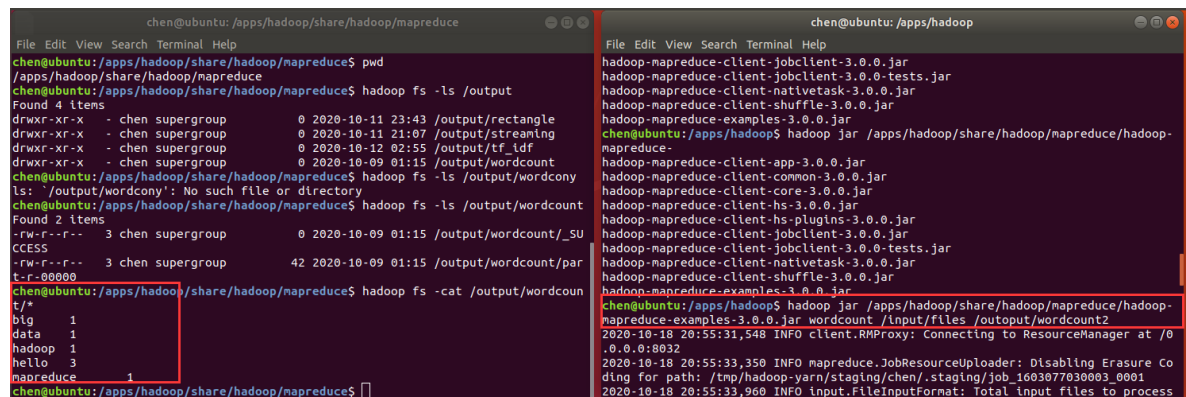
熟悉hadoop 的shell 和java API

## 实验过程

### 一、shell api

```
hadoop fs -cp /input/testfile1 /input/test/
hadoop fs -mv /input/testfile1 /input/test/
hadoop fs -rm /input/testfile1
hadoop fs -rm -r /input/testfile1
hadoop fs -put /data/testfile /input/
hadoop fs -get /input/testfile1 /data/
hadoop fs -mkdir -p /input/test/testfile1
hadoop fs -test -e /input/testfile1
hadoop fs -test -d /input/
echo $? # 返回上条命令的输出，是返回0，否返回1
```

### 使用hadoop自带的Wordcount



The image shows two terminal windows. The left window shows the execution of Hadoop fs commands to list files, check directory existence, and run the wordcount job. The right window shows the execution of the Hadoop jar command to run the wordcount job, followed by the output of the job, which lists the files in the output directory and the wordcount results for each file.

```
chen@ubuntu: /apps/hadoop/share/hadoop/mapreduce
File Edit View Search Terminal Help
chen@ubuntu: /apps/hadoop/share/hadoop/mapreduce$ pwd
/apps/hadoop/share/hadoop/mapreduce
chen@ubuntu: /apps/hadoop/share/hadoop/mapreduce$ hadoop fs -ls /output
Found 4 items
drwxr-xr-x - chen supergroup 0 2020-10-11 23:43 /output/rectangle
drwxr-xr-x - chen supergroup 0 2020-10-11 21:07 /output/streaming
drwxr-xr-x - chen supergroup 0 2020-10-12 02:55 /output/tf_idf
drwxr-xr-x - chen supergroup 0 2020-10-09 01:15 /output/wordcount
chen@ubuntu: /apps/hadoop/share/hadoop/mapreduce$ hadoop fs -ls /output/wordcount
ls: /output/wordcount: No such file or directory
chen@ubuntu: /apps/hadoop/share/hadoop/mapreduce$ hadoop fs -ls /output/wordcount
Found 2 items
-rw-r--r-- 3 chen supergroup 0 2020-10-09 01:15 /output/wordcount/_SUCCESS
-rw-r--r-- 3 chen supergroup 42 2020-10-09 01:15 /output/wordcount/part-r-00000
chen@ubuntu: /apps/hadoop/share/hadoop/mapreduce$ hadoop fs -cat /output/wordcount/t/*
big 1
data 1
hadoop 1
hello 3
mapreduce 1
chen@ubuntu: /apps/hadoop/share/hadoop/mapreduce$
```

```
chen@ubuntu: /apps/hadoop
File Edit View Search Terminal Help
hadoop-mapreduce-client-jobclient-3.0.0.jar
hadoop-mapreduce-client-jobclient-3.0.0-tests.jar
hadoop-mapreduce-client-native-task-3.0.0.jar
hadoop-mapreduce-client-shuffle-3.0.0.jar
hadoop-mapreduce-examples-3.0.0.jar
chen@ubuntu: /apps/hadoop$ hadoop jar /apps/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.0.0.jar wordcount /input/files /output/wordcount2
2020-10-18 20:55:31,548 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2020-10-18 20:55:33,350 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/chen/.staging/job_1603077030003_0001
2020-10-18 20:55:33,960 INFO input.FileInputFormat: Total input files to process
```

### hadoop 安全模式

```
进入安全模式
hdfs dfsadmin -safemode enter
离开安全模式
hdfs dfsadmin -safemode leave
```

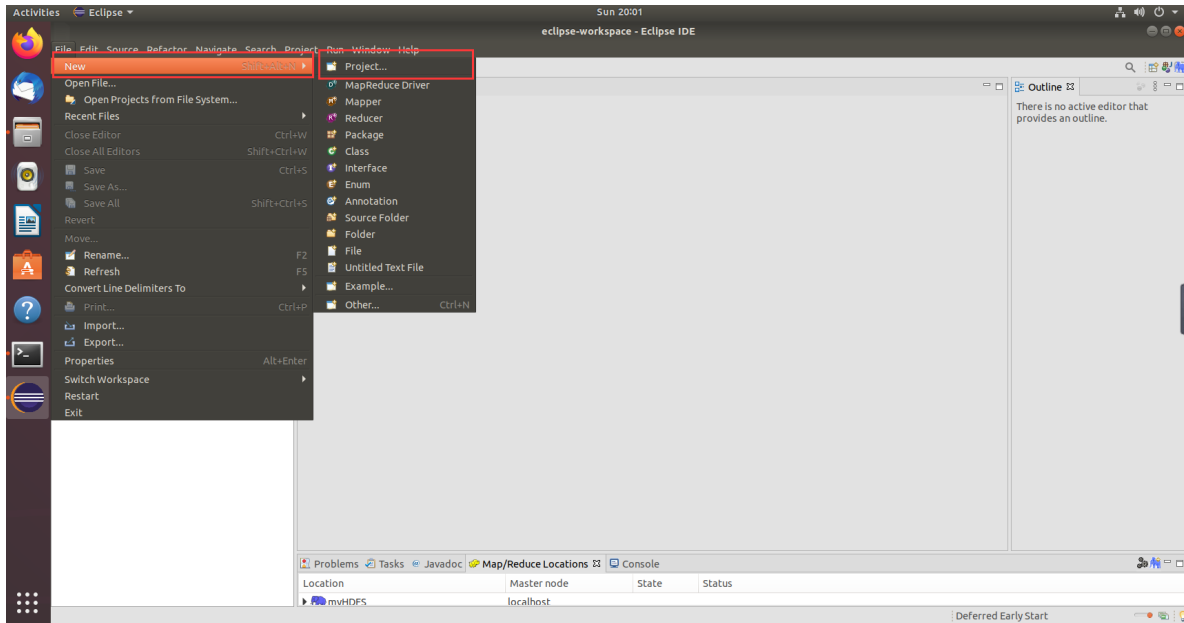
## 二、java api

### 1.准备需要的jar文件

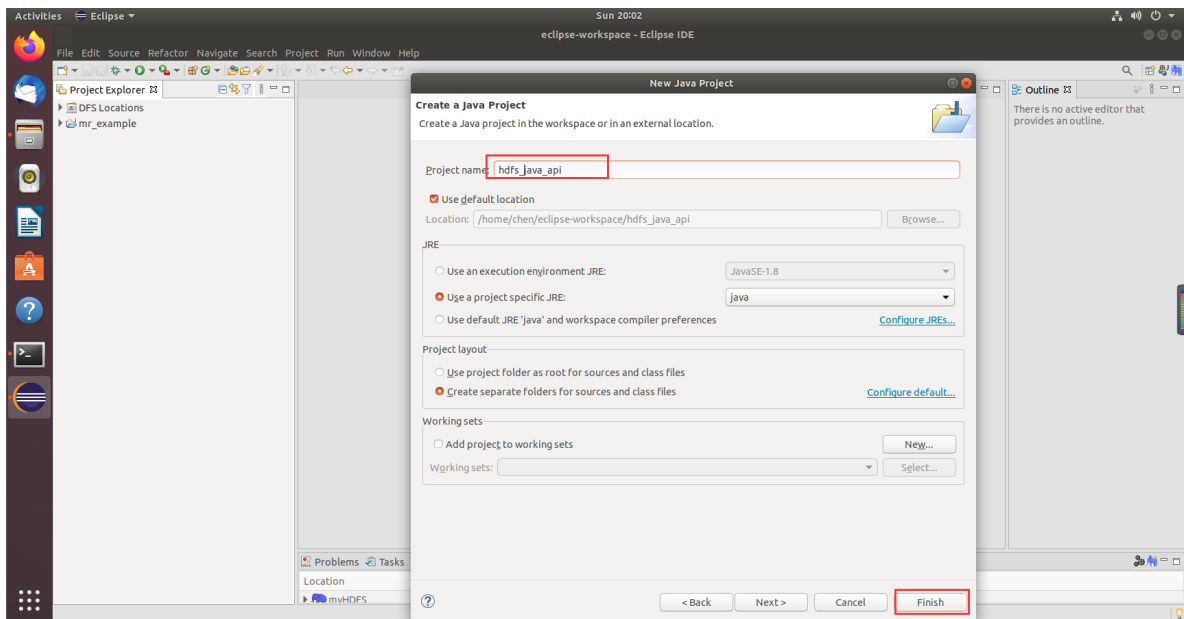
将/apps/hadoop/share/hadoop 目录下的 common, hdfs, mapreduce, yarn 4 个子目录中的 jar 文件以及这 4 个子目录下 lib 文件夹中的所有 jar 文件复制到~/big\_data\_tools/hadoop3lib 中。这些 jar 文件将作为外部的 jar 文件添加到工程中。分别切换到那几个目录下，执行一下这条命令

```
cp *.jar ~/big_data_tools/hadoop3lib/
```

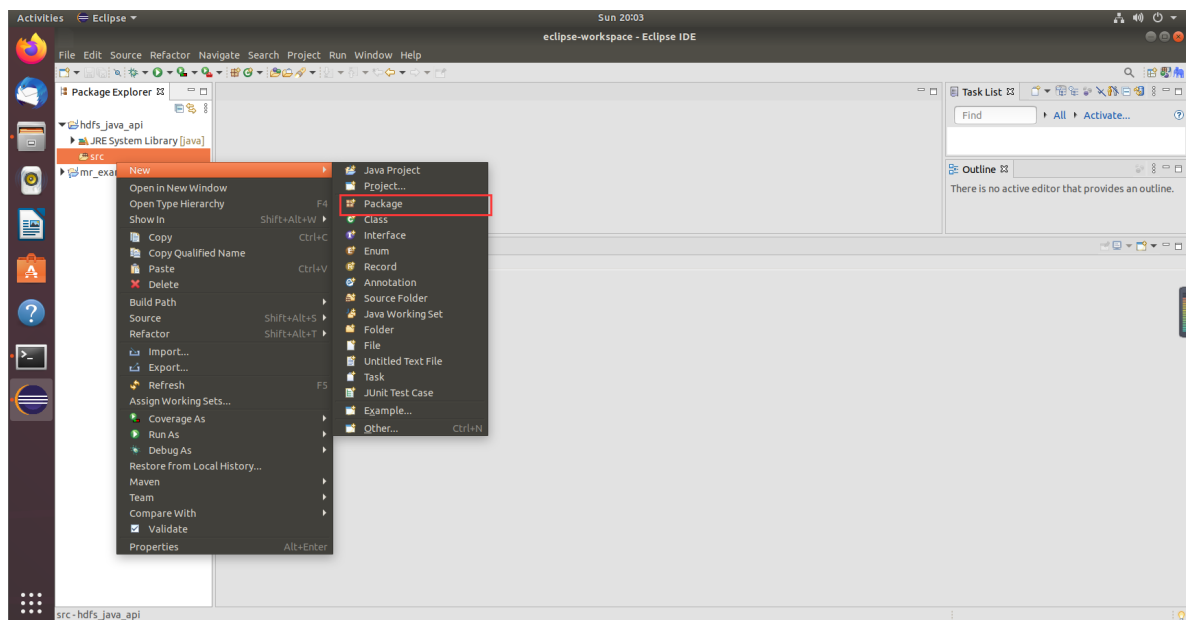
## 2.新建javaproject



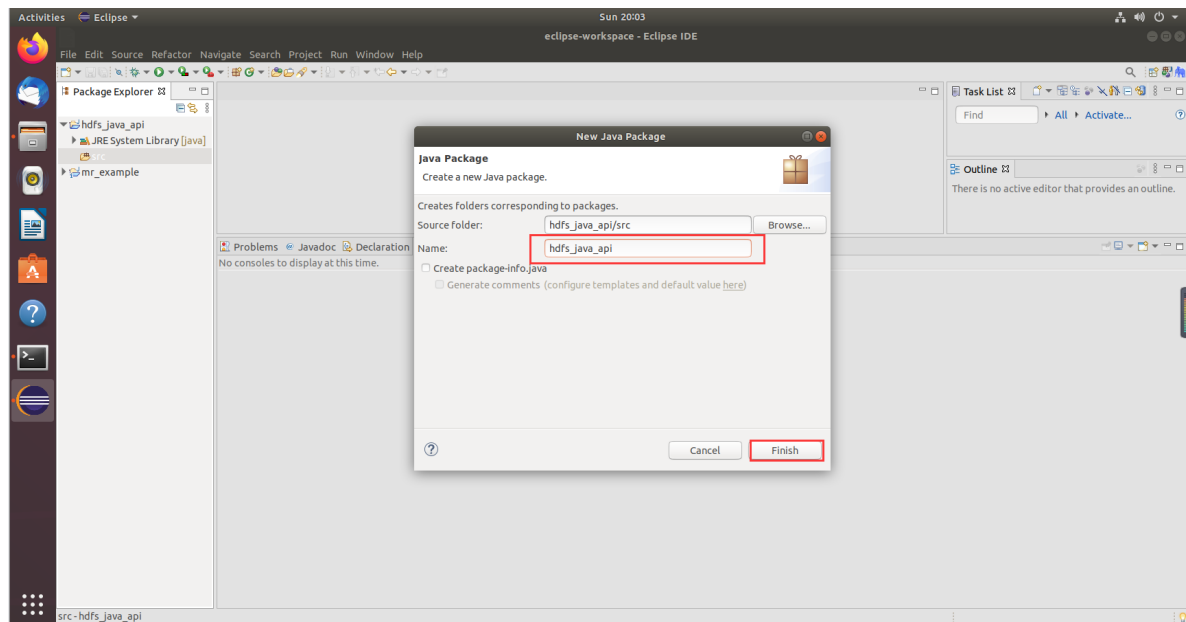
## 3.命名为hdfs\_java\_api



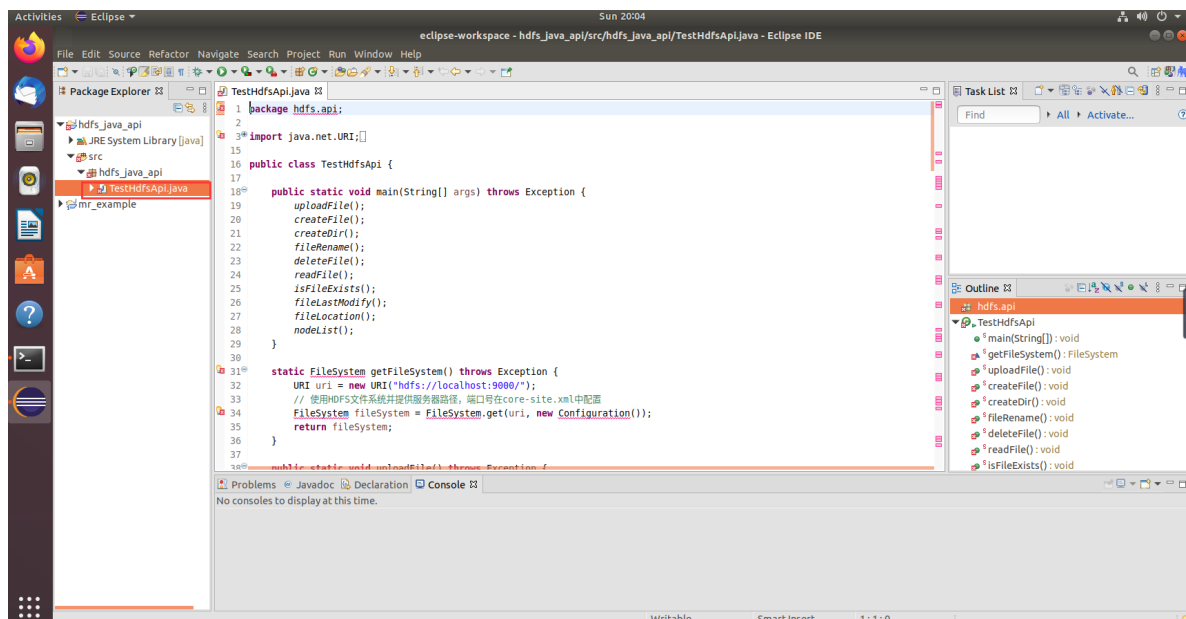
## 4.新建包



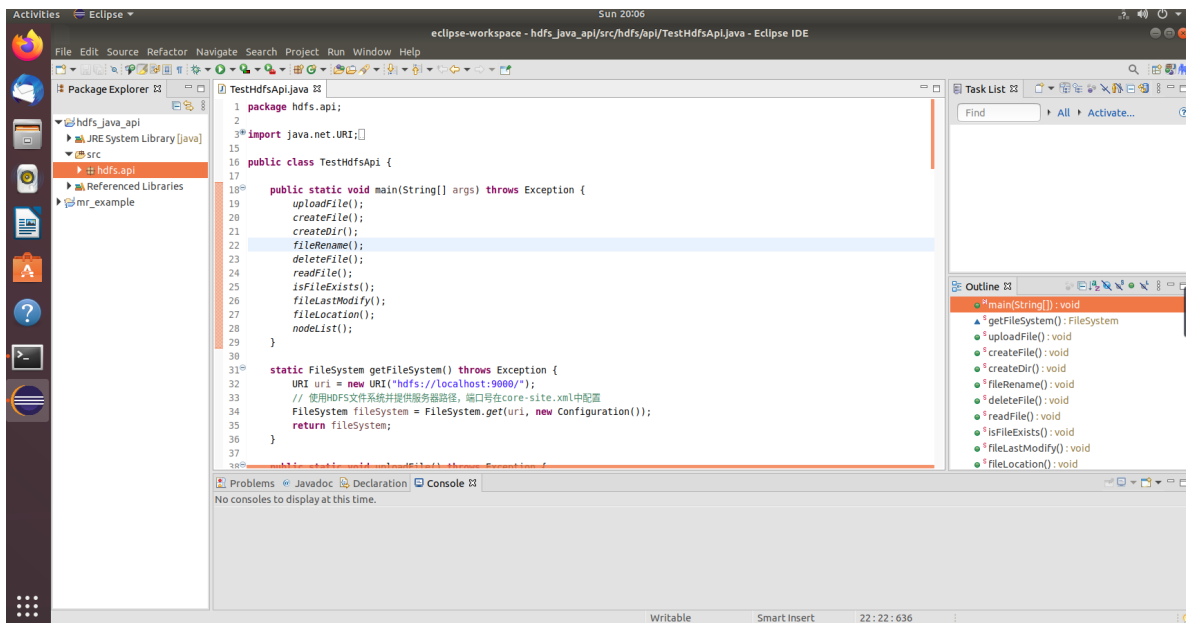
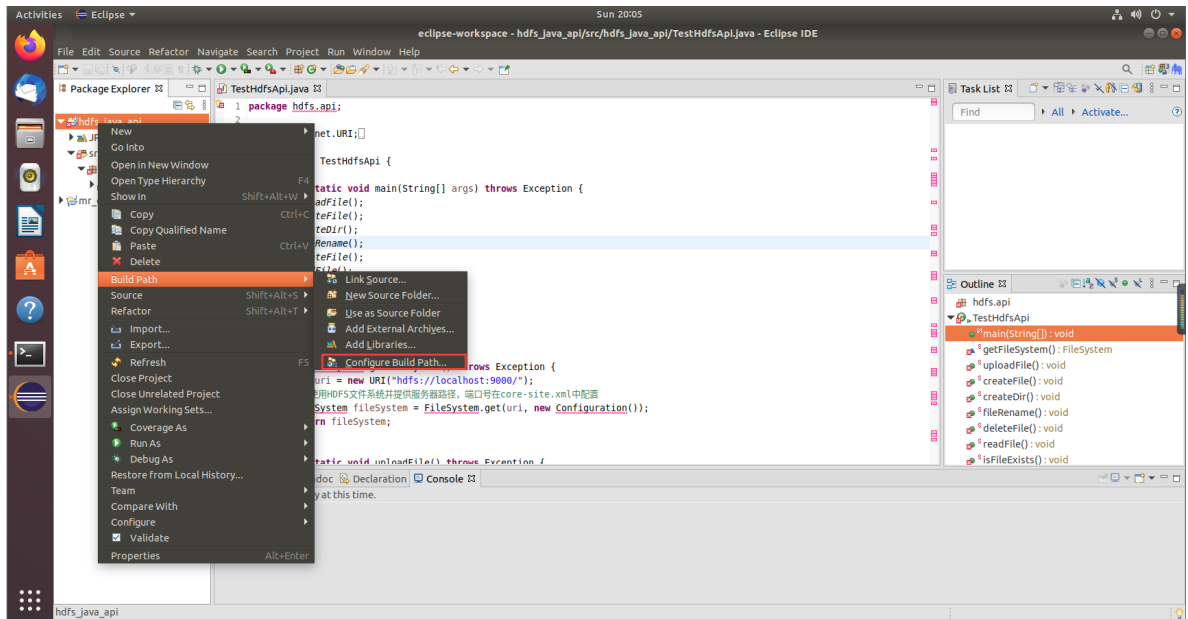
## 5.命名为 hdfs.api



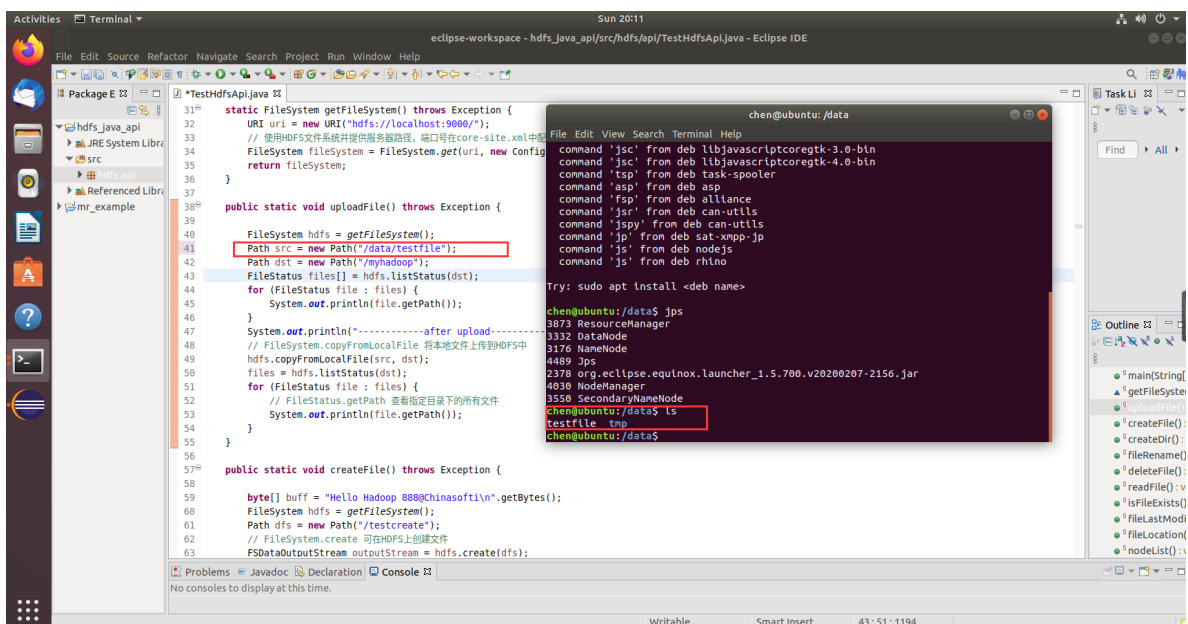
## 5.新建java类,TestHdfsApi.java



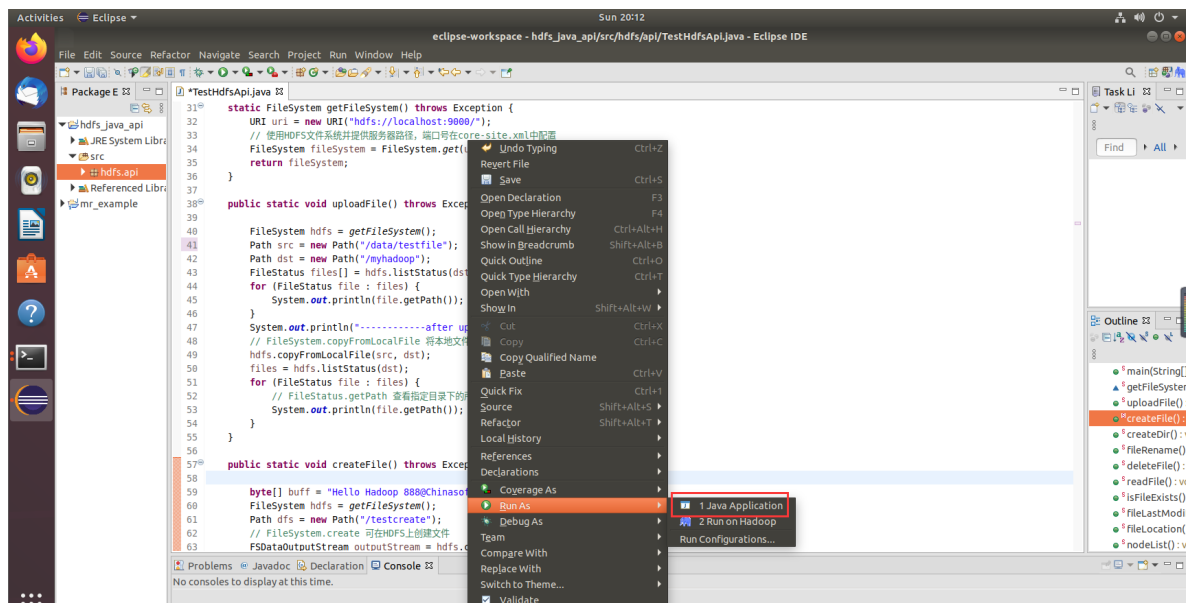
## 6.导入外部依赖的Jar包



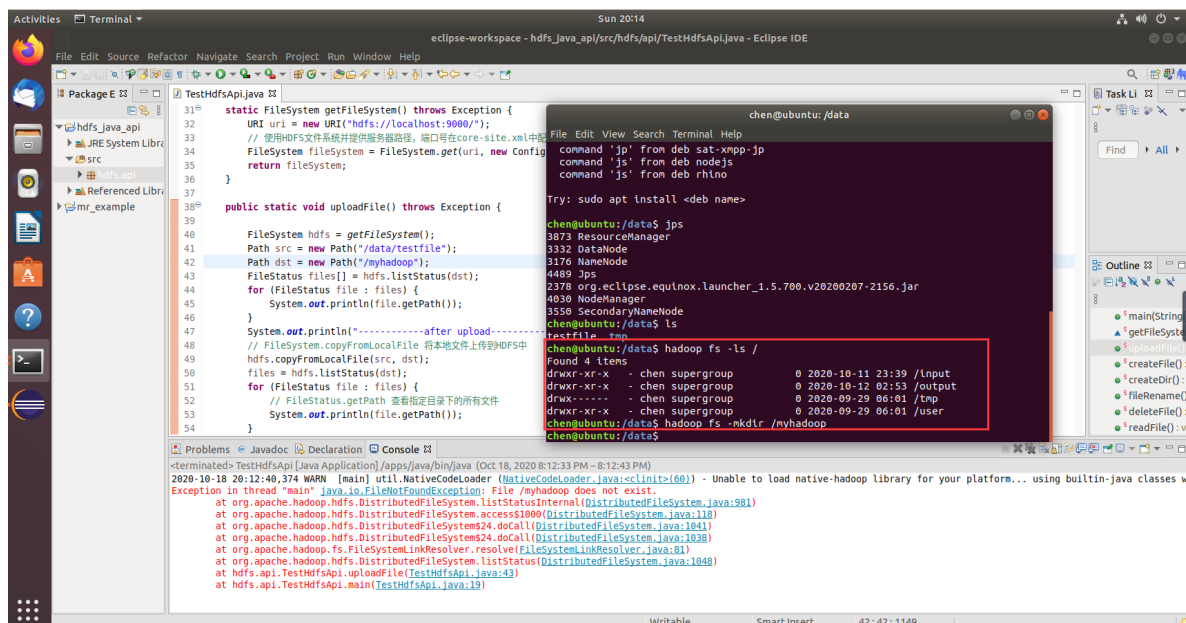
## 7.创建testfile



## 8.运行程序



## 9.创建/myhadoop目录



## 10.再次运行as java application

