

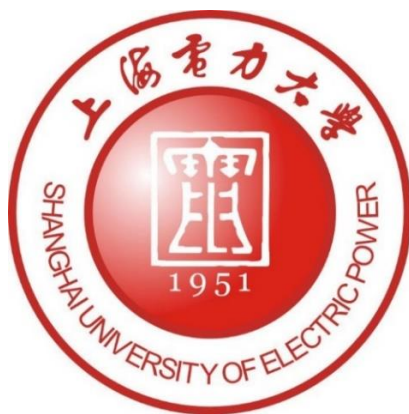


上海電力大學

SHANGHAI UNIVERSITY OF ELECTRIC POWER

大数据技术原理及应用

实验报告



学 院： 计算机科学与技术学院

专 业： 电力信息技术

报告名称： Hadoop MapReduce 编程实践

学生姓名： 崔荣成 学号： 19108002

指导老师： 周 平

时 间： 2020 年 4 月 17 日

目 录

一、	实验目的.....	3
二、	实验内容.....	3
三、	实验步骤.....	3
1.	Hadoop 整体介绍及集群搭建	3
1.1.	大数据简介.....	3
1.2.	Hadoop 整体概念	4
1.3.	Windows 下安装 Hadoop.....	5
2.	HDFS 简介及常用操作	15
2.1.	HDFS 相关概念和特性.....	15
2.2.	HDFS Shell 操作（命令行操作）	17
2.3.	HDFS 的 Java API 操作.....	18
3.	Eclipse 编程进行 hdfs 常用操作	24
3.1.	创建 TestHDFS 程序.....	24
3.2.	创建 TestHDFS 类.....	25
3.3.	添加依赖库.....	25
3.4.	添加 junit 测试单元.....	28
3.5.	创建 fs 对象.....	29
3.6.	创建文件夹.....	30
3.7.	删除文件.....	30
3.8.	上传文件.....	30
3.9.	下载文件.....	31
3.10.	查看文件及文件夹信息.....	31
4.	使用 eclipse 创建并运行 wordcount 程序.....	32
4.1.	创建 MapReduce 程序.....	32
4.2.	编写 WordCount 程序.....	34
4.3.	配置参数.....	36
4.4.	运行结果.....	37

一、 实验目的

掌握 Hadoop 平台的安装配置方法，认识 Hadoop 平台的构成；熟悉 Hadoop 平台基本命令操作；

掌握 HDFS 文件系统结构，熟悉 HDFS 的基本命令操作和编程操作。

二、 实验内容

- 1、安装配置 Hadoop 平台；
- 2、熟悉 Hadoop 常用指令；
- 3、安装配置 Eclipse-Hadoop 环境；
- 4、运行 HDFS 基本示例；

三、 实验步骤

1. Hadoop 整体介绍及集群搭建

1.1. 大数据简介

数据

数据就是数值，也就是我们通过观察、实验或计算得出的结果。数据有很多种，最简单的就是数字。数据也可以是文字、图像、声音等。在计算机系统中，数据以二进制信息单元 0,1 的形式表示。

大数据

大数据（big data），指的是在一定时间范围内不能以常规软件工具处理（存储和计算）的大而复杂的数据集。

数据衡量单位

- 从小到大排列：bit、Byte、KB、MB、GB、TB、PB、EB、 ZB、YB、BB、NB、DB

1 Byte = 8 bit
1 KB = 1,024 Bytes = 8192 bit
1 MB = 1,024 KB = 1,048,576 Bytes (普通用户数据级别)
1 GB = 1,024 MB = 1,048,576 KB
1 TB = 1,024 GB = 1,048,576 MB
1 PB = 1,024 TB = 1,048,576 GB (企业级数据级别)
1 EB = 1,024 PB = 1,048,576 TB
1 ZB = 1,024 EB = 1,048,576 PB (全球数据总量级别)
1 YB = 1,024 ZB = 1,048,576 EB
1 BB = 1,024 YB = 1,048,576 ZB
1 NB = 1,024 BB = 1,048,576 YB
1 DB = 1,024 NB = 1,048,576 BB

据国际数据公司(IDC)统计, 全球数据总量预计 2020 年达到 44ZB, 中国数据量将达到 8060EB, 占全球数据总量的 18%

大数据的特点

- 容量 (Volume): 数据的大小决定所考虑的数据的价值和潜在的信息
 - 新浪微博, 3 亿用户, 每天上亿条微博
 - 朋友圈, 8 亿用户, 每天亿级别朋友圈
- 种类 (Variety): 数据类型的多样性, 包括文本, 图片, 视频, 音频
 - 结构化数据: 可以用二维数据库表来抽象, 抽取数据规律
 - 半结构化数据: 介于结构化和非结构化之间, 主要指 XML, HTML 等, 也可称非结构化
 - 非结构化数据: 不可用二维表抽象, 比如图片, 图像, 音频, 视频等
- 速度 (Velocity): 指获得数据的速度以及处理数据的速度
 - 数据的产生呈指数式爆炸式增长
 - 处理数据要求的延时越来越低
- 价值 (Value): 合理运用大数据, 以低成本创造高价值
 - 单条数据记录无价值, 无用数据多
 - 综合价值大, 隐含价值大

一句话: 容量大, 种类多, 速度快, 价值高

1.2. Hadoop 整体概念

产生背景

- Hadoop 是 Apache Lucene 创始人 Doug Cutting 创建的, 最早起源于 Apache Nutch 项目。Nutch 的设计目标是构建一个大型的全网搜索引擎, 包括网页抓取、索引、查询等功能, 但随着抓取网页数量的增加, 遇到了严重的可扩展性问题 —— 如何解决数十亿网页的存储和索引问题
- 2003 年、2004 年谷歌发表的三篇论文为该问题提供了可行的解决方案
 - 分布式文件系统 GFS, 可用于处理海量网页的存储
 - 分布式计算框架 MapReduce, 可用于处理海量网页的索引计算问题

-
- 分布式数据库 BigTable，每一张表可以存储上 billions 行和 millions 列
 - Nutch 的开发人员完成了相应的开源实现 HDFS 和 MapReduce，并从 Nutch 中剥离成为独立项目 Hadoop，到 2008 年 1 月，Hadoop 成为 Apache 顶级项目，迎来了它的快速发展期

什么是 Hadoop

Hadoop 官网: <https://hadoop.apache.org/>

Hadoop 是 Apache 旗下的一套开源分布式软件平台，用户可以在不了解分布式底层细节的情况下，开发分布式程序，充分利用集群的威力进行高速运算和存储。你可以把 Hadoop 理解为一个分布式的操作系统。

什么是分布式程序？

- 该软件系统会划分成多个子系统或模块，各自运行在不同的机器上，子系统或模块之间通过网络通信进行协作，实现最终的整体功能，这多个机器就构成了**集群**。**通俗的讲，分布式系统就是利用集群的多个节点共同协作完成一项或多项具体业务功能的系统。**

Hadoop 的**核心组件**

- Common（基础功能组件：工具包，RPC 框架）
- HDFS（Hadoop Distributed File System 分布式文件系统）
- MapReduce（分布式运算编程框架）
- YARN（Yet Another Resources Negotiator 运算资源调度系统）

广义的 Hadoop 指一个更广泛的概念——**Hadoop 生态圈**：

- HDFS: Hadoop 的分布式文件存储系统
- MapReduce: Hadoop 的分布式程序运算框架，也可以叫做一种编程模型
- YARN: Hadoop 的资源调度系统
- Hive: 基于 Hadoop 的类 SQL 数据仓库工具
- HBase: 基于 Hadoop 的列式分布式 NoSQL 数据库
- ZooKeeper: 分布式协调服务组件
- Oozie/Azkaban: 工作流调度引擎
- Sqoop: 数据迁入迁出工具
- Flume: 日志采集工具
-

1.3. Windows 下安装 Hadoop

安装 jdk1.8.0 并配置环境

下载地址:

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

下载对应的 jdk



配置环境变量:

新建 JAVA_HOME


```
C:\Users\崔荣成>java -version
java version "1.8.0_181"
Java(TM) SE Runtime Environment (build 1.8.0_181-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.181-b13, mixed mode)
```

Cmd 命令测试，安装成功。

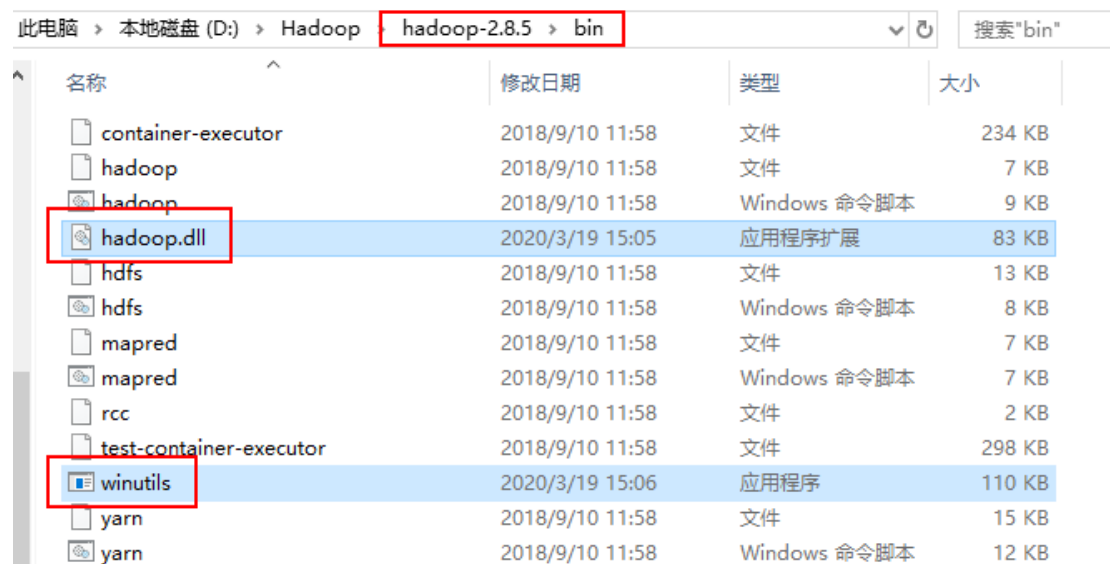
安装 Hadoop2.8.5 并配置环境

1. 下载 Hadoop2.8.5 并解压到本地

下载链接 <https://mirrors.tuna.tsinghua.edu.cn/apache/hadoop/common/>

2. 下载对应版本的配置文件

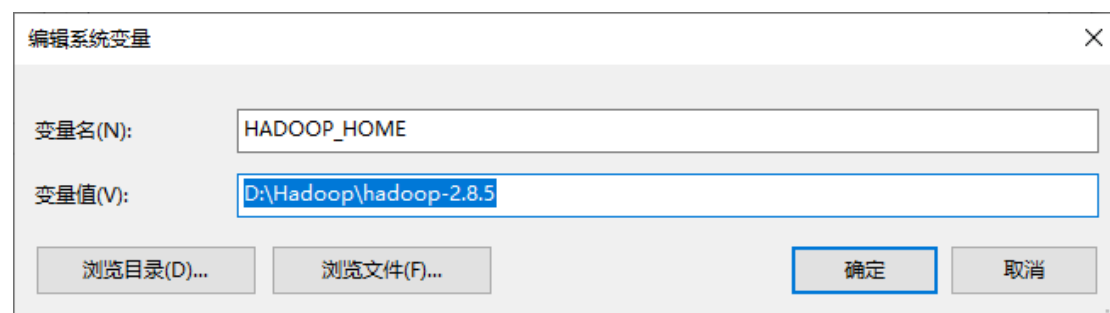
下载链接 <https://github.com/cdarlint/winutils>



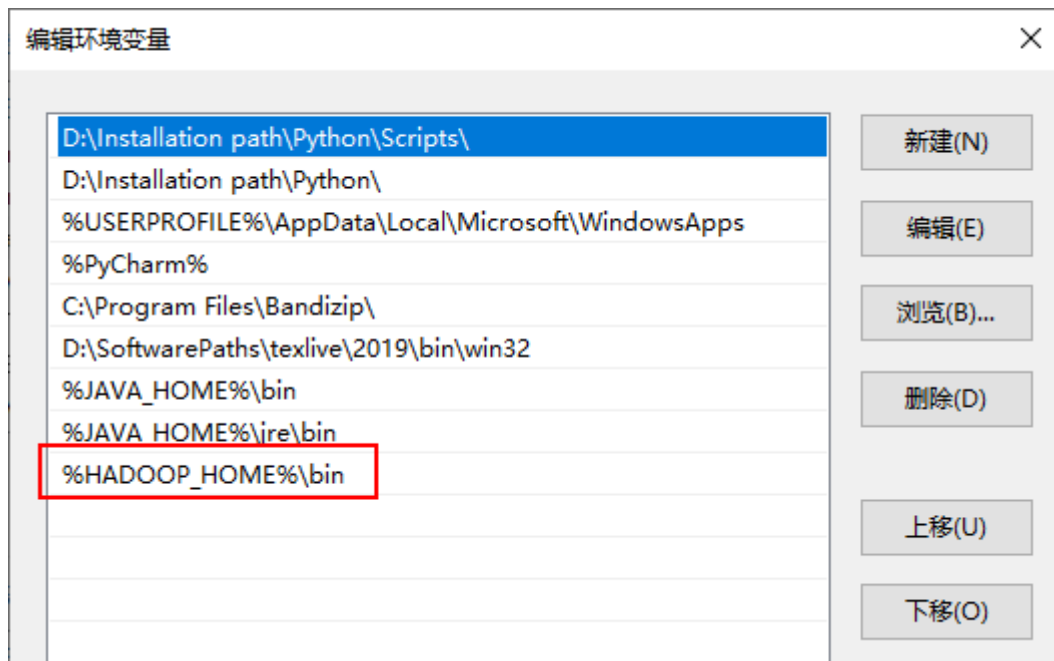
关键是将 hadoop.dll 和 winutils.exe 两个文件放入到/bin 目录下

3. 配置 hadoop 环境变量

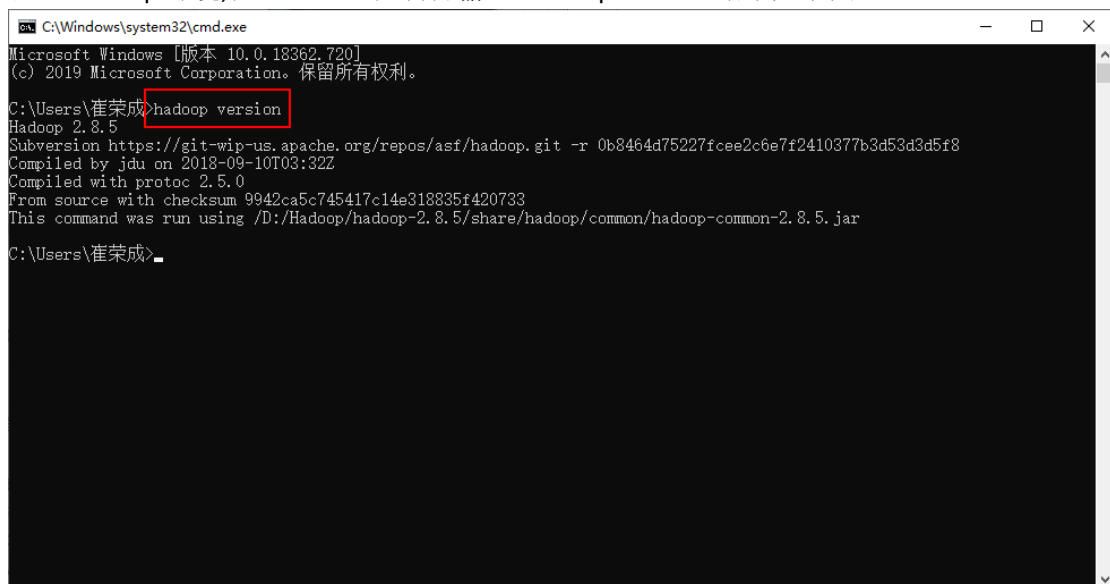
新建 HADOOP_HOME



添加 path 属性,将;%HADOOP_HOME%\bin;添加到 path 环境变量中



验证 hadoop 环境,在 windows 控制台输入: `hadoop version` 结果如下图:



4.修改 Hadoop 配置文件

打开 Hadoop 文件夹下的\etc\hadoop 目录对配置文件进行修改,

hadoop-env.cmd

找到如下代码 (26 行), 将路径修改为你自己的 JAVA_HOME 路径

```
@rem The java implementation to use. R
set JAVA_HOME=D:\Java\jdk1.8.0_181|
```

hdfs-site.xml

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:/D:/Hadoop/hadoopdata/namenode</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:/D:/Hadoop/hadoopdata/datanode</value>
  </property>
</configuration>
```

namenode 的文件路径:

D:/Hadoop/hadoopdata/namenode

datanode 的文件路径:

D:/Hadoop/hadoopdata/datanode

core-site.xml

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://localhost:9000</value>
  </property>
</configuration>
```

mapred-site.xml

```
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

yarn-site.xml

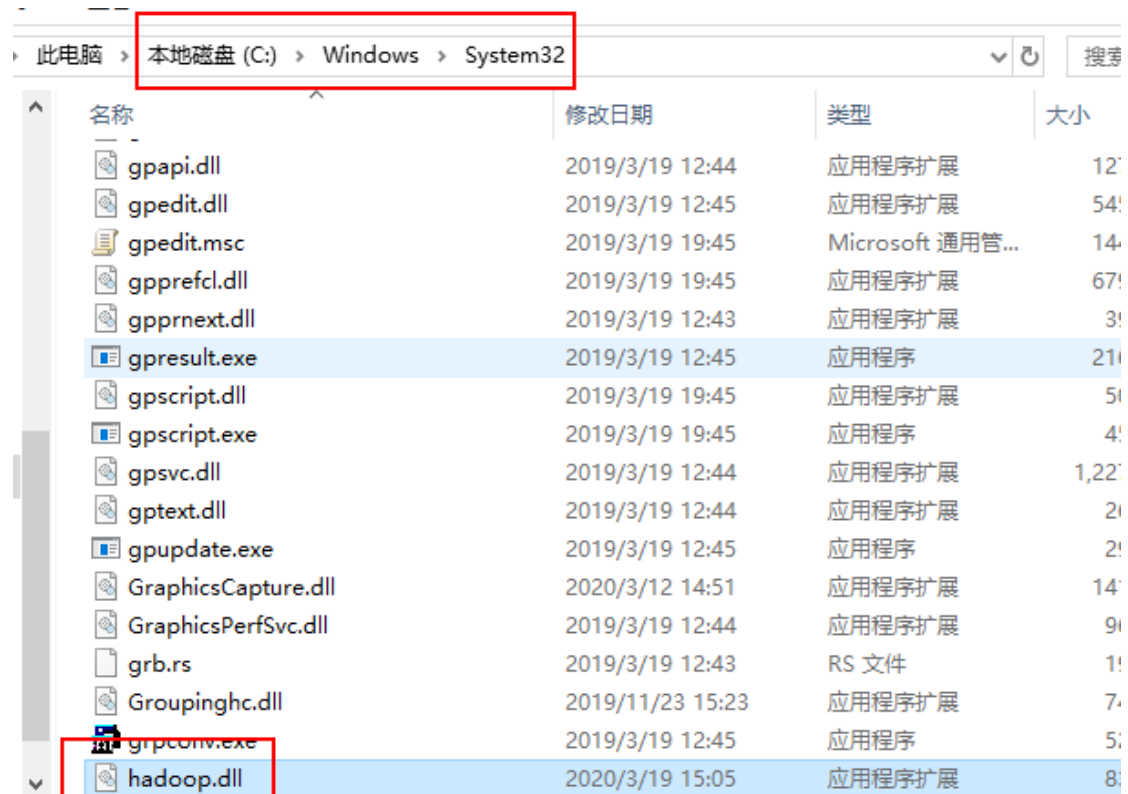
```
<configuration>

<!-- Site specific YARN configuration properties -->

  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>
      yarn.nodemanager.aux-services.mapreduce.shuffle.class
    </name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>

</configuration>
```

5. 将 Hadoop 文件夹\bin 下的 hadoop.dll 复制到 C:\Windows\System32



6.初始化 HDFS

运行如下命令:

```
hdfs namenode -format
```

Most commands print help when invoked w/o parameters.

```
C:\Users\崔荣成>hdfs namenode -format
20/03/19 15:31:58 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   user = 崔荣成
STARTUP_MSG:   host = DESKTOP-U5MIIOV/192.168.132.1
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 2.8.5
STARTUP_MSG:   classpath = D:\Hadoop\hadoop-2.8.5\etc\hadoop;D:\Hadoop\hadoop-2.8.5\share\hadoop\comm
.1.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\apacheds-ldap-2.0.0-M15.jar;D:\Hadoop\hadoop-2.
common\lib\apacheds-kerberos-codec-2.0.0-M15.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\api-asn
;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\api-util-1.0.0-M20.jar;D:\Hadoop\hadoop-2.8.5\share\h
m-3.2.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\avro-1.7.4.jar;D:\Hadoop\hadoop-2.8.5\share\
commons-beanutils-1.7.0.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\commons-beanutils-core-1.8.
oop-2.8.5\share\hadoop\common\lib\commons-cli-1.2.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\
ar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\commons-collections-3.2.2.jar;D:\Hadoop\hadoop-2.8.
on\lib\commons-compress-1.4.1.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\commons-configuratio
\hadoop-2.8.5\share\hadoop\common\lib\commons-digester-1.8.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\co
-2.4.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\commons-lang-2.6.jar;D:\Hadoop\hadoop-2.8.5\sh
lib\commons-logging-1.1.3.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\commons-math3-3.1.1.jar;
8.5\share\hadoop\common\lib\commons-net-3.1.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\curator
D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\curator-framework-2.7.1.jar;D:\Hadoop\hadoop-2.8.5\sh
b\curator-recipes-2.7.1.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\gson-2.2.4.jar;D:\Hadoop\h
adoop\common\lib\guava-11.0.2.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\hadoop-annotations-2
hadoop-2.8.5\share\hadoop\common\lib\hadoop-auth-2.8.5.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common
1.3.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\htrace-core4-4.0.1-incubating.jar;D:\Hadoop\ha
adoop\common\lib\httpclient-4.5.2.jar;D:\Hadoop\hadoop-2.8.5\share\hadoop\common\lib\httpcore-4.4.4.jar
*****/
```

初始化成功

```
20/03/19 15:32:03 INFO namenode.FSNamesystem: dfs.namenode.safemode.extension = 30000
20/03/19 15:32:03 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
20/03/19 15:32:03 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
20/03/19 15:32:03 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
20/03/19 15:32:03 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
20/03/19 15:32:03 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time
is 600000 millis
20/03/19 15:32:03 INFO util.GSet: Computing capacity for map NameNodeRetryCache
20/03/19 15:32:03 INFO util.GSet: VM type = 64-bit
20/03/19 15:32:03 INFO util.GSet: 0.0299999999329447746% max memory 889 MB = 273.1 KB
20/03/19 15:32:03 INFO util.GSet: capacity = 2^15 = 32768 entries
20/03/19 15:32:03 INFO namenode.FSImage: Allocated new BlockPoolId: BP-190122261-192.168.132.1-1584603123341
20/03/19 15:32:03 INFO common.Storage: Storage directory D:\Hadoop\hadoopdata\namenode has been successfully formatted.
20/03/19 15:32:03 INFO namenode.FSImageFormatProtobuf: Saving image file D:\Hadoop\hadoopdata\namenode\current\fsimage.c
kpt_00000000000000000000 using no compression
20/03/19 15:32:03 INFO namenode.FSImageFormatProtobuf: Image file D:\Hadoop\hadoopdata\namenode\current\fsimage.chkpt_000
0000000000000000000 of size 325 bytes saved in 0 seconds.
20/03/19 15:32:03 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
20/03/19 15:32:03 INFO util.ExitUtil: Exiting with status 0
20/03/19 15:32:03 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at DESKTOP-U5MIIOV/192.168.132.1
*****/
C:\Users\崔荣成>
```

7.启动 Hadoop

进入 hadoop 目录下 sbin，执行启动命令：

start -all

localhost:50070/dfshealth.html#tab-overview

应用 百度一下, 你就知道 微软必应搜索 - 全... 工具 微信 充电 铁大 程序猿 电大 雅思 论文 机器学习 剁手 无人机

Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities

Overview 'localhost:9000' (active)

Started:

Thu Mar 19 15:46:44 +0800 2020

Version:

2.8.5, r0b8464d75227fcee2c6e7f2410377b3d53d3d5f8

Compiled:

Mon Sep 10 11:32:00 +0800 2018 by jdu from branch-2.8.5

Cluster ID:

CID-0c0b02cf-11cc-4e89-9e92-d9b66aa02b84

Block Pool ID:

BP-190122261-192.168.132.1-1584603123341

Summary

Security is off.

Safemode is off.

1 files and directories, 0 blocks = 1 total filesystem object(s).

Heap Memory used 62.65 MB of 139.5 MB Heap Memory. Max Heap Memory is 889 MB.


Non Heap Memory used 41.02 MB of 42.22 MB Committed Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity: 300 GB

localhost:8088

localhost:8088/cluster

应用 百度一下, 你就知道 微软必应搜索 - 全... 工具 微信 充电 铁大 程序猿 电大 雅思 论文 机器学习 剁手 无人机



All Applications

Cluster

About

Nodes

Node Labels

Applications

NEW

NEW SAVING

SUBMITTED

ACCEPTED

RUNNING

FINISHED

FAILED

KILLED

Scheduler

Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total
0	0	0	0	0	0 B	8 GB	0 B	0	8

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
1	0	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Maximum Cluster /
Capacity Scheduler	(MEMORY)	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	% of Queue	% of Cluster	Progress
No data available in table															

Showing 0 to 0 of 0 entries

8.关闭 Hadoop

关闭命令: stop-all

```
Microsoft Windows [版本 10.0.18362.720]
(c) 2019 Microsoft Corporation. 保留所有权利。

D:\Hadoop\hadoop-2.8.5\sbin>start-all
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons
D:\Hadoop\hadoop-2.8.5\sbin>stop-all
This script is Deprecated. Instead use stop-dfs.cmd and stop-yarn.cmd
成功: 给进程发送了终止信号, 进程的 PID 为 2528。
成功: 给进程发送了终止信号, 进程的 PID 为 2920。
stopping yarn daemons
成功: 给进程发送了终止信号, 进程的 PID 为 15880。
成功: 给进程发送了终止信号, 进程的 PID 为 19364。
信息: 没有运行的带有指定标准的任务。
D:\Hadoop\hadoop-2.8.5\sbin>
```

2. HDFS 简介及常用操作

HDFS (Hadoop Distributed File System)，Hadoop 分布式文件系统，主要用来解决海量数据的存储问题，在大数据系统架构中为各类分布式运算框架（MapReduce，Spark，Tez，Flink，…）提供数据存储服务

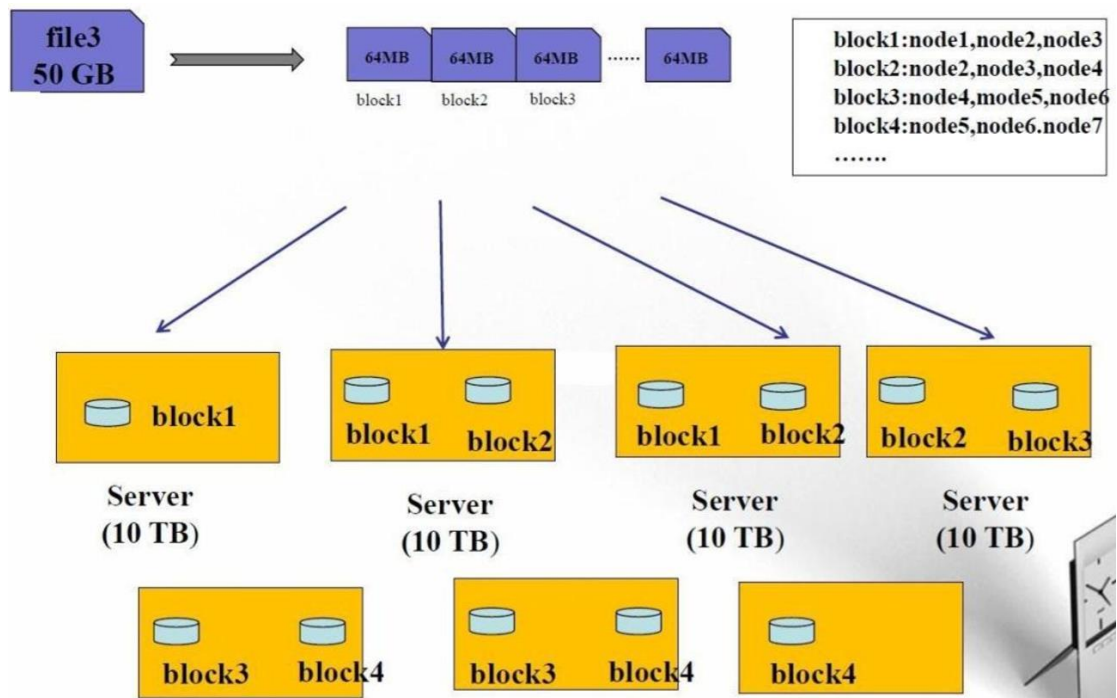
- 设计思想
 - 分而治之：将大文件，大批量文件，分布式的存放于大量服务器上。以便于采取分而治之的方式对海量数据进行运算分析
- 重要概念
 - 数据块/副本、负载均衡、心跳机制、副本存放策略、元数据/元数据管理、安全模式……

2.1. HDFS 相关概念和特性

HDFS 设计思路

HDFS 被设计成用来使用低廉的服务器来进行海量数据的存储，那是怎么做到的呢？

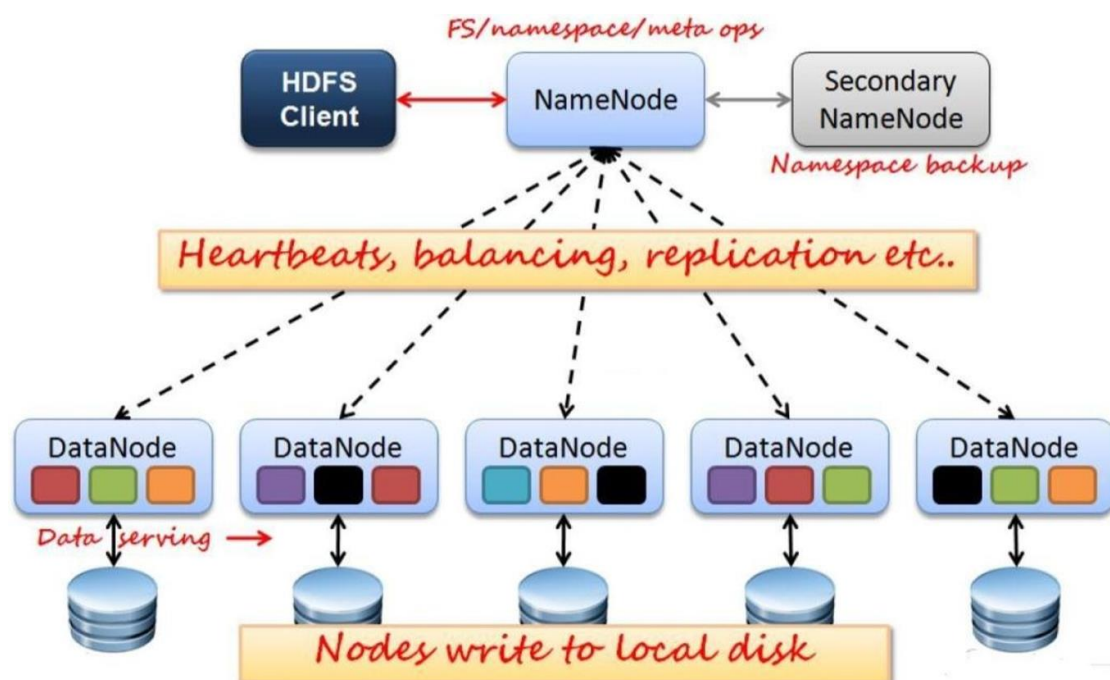
- 1、大文件被切割成小文件，使用分而治之的思想让很多服务器对同一个文件进行联合管理
- 2、每个小文件做冗余备份，并且分散存到不同的服务器，做到高可靠不丢失



HDFS 架构

主从架构设计：

- 主节点：NameNode（集群老大，管理整个文件系统的元数据，处理客户端读且请求负责）
 - 普通分布式集群还有一个 Secondary NameNode（用来帮 NameNode 分担压力）
- 从节点：DataNode（存储整个集群所有数据块，真正处理数据读写）



HDFS 特性

- HDFS 中的文件在物理上是分块存储 (block)，块的大小可以通过配置参数 (dfs.blocksize) 来规定，默认大小在 hadoop2.x 版本中是 128M，1.x 是 64M，3.x 是 256M
- HDFS 文件系统会给客户端提供一个统一的抽象目录树，客户端通过路径来访问文件，形如：hdfs://hadoop1:9000/soft/hadoop-2.7.5-centos-6.7.tar.gz
- 目录结构及文件分块位置信息 (metadata, 元数据) 的管理由 namenode 节点承担。namenode 是 HDFS 集群主节点，负责维护整个 hdfs 文件系统的目录树，以及每一个路径 (文件) 所对应的 block 块信息 (block 的 id, 及所在的 datanode 服务器)
- 文件的各个 block 的存储管理由 datanode 节点承担。datanode 是 HDFS 集群从节点，每一个 block 都可以在多个 datanode 上存储多个副本 (副本数量也可以通过参数设置：dfs.replication，默认是 3)
- HDFS 是设计成适应一次写入，多次读出的场景，且不支持文件的随机修改

HDFS 优缺点

- 优点
 - 可构建在廉价机器上，通过多副本提高可靠性，提供了容错和恢复机制
 - 高容错性，数据自动保存多个副本，副本丢失后，自动恢复
 - 适合大数据处理，GB、TB、甚至 PB 级数据，百万规模以上的文件数量，10K+节点规模
 - 流式文件访问，一次性写入，多次读取，保证数据一致性
- 缺点
 - 延迟高 (涉及到多个节点之间相互通信)
 - 不适合存储小文件 (因为元信息存储在 NameNode 内存中，一个 block 元信息消耗大约 150 byte，存储 1 亿个 block，大约需要 15GB 内存来存储对应的元数据，浪费资源)

2.2. HDFS Shell 操作 (命令行操作)

HDFS 提供 shell 命令行客户端，使用方法如下：

```
[hadoop@hadoop1 ~]$ hadoop fs -ls /
Found 2 items
drwx-wx-wx - hadoop supergroup 0 2018-12-22 17:06 /tmp
drwxr-xr-x - hadoop supergroup 0 2018-12-22 17:08 /user
[hadoop@hadoop1 ~]$
```

常用命令参数介绍：

- -help 功能：输出这个命令参数手册
 - hadoop -help

- `hadoop fs -help`
- `hadoop fs -help ls`
- `-ls` 功能: 显示目录信息
 - `hadoop fs -ls hdfs://hadoop1:9000/`
 - `hadoop fs -ls /` (二者等效)
- `-mkdir` 功能: 在 hdfs 上创建目录
 - `hadoop fs -mkdir -p /aa/bb/cc/dd`
- `-put` 功能: 等同于 `-copyFromLocal`, 进行文件上传
 - `hadoop fs -put /aaa/jdk.tar.gz /bbb/jdk.tar.gz`
- `-get` 功能: 等同于 `-copyToLocal`, 就是从 hdfs 下载文件到本地
 - `hadoop fs -get /aaa/jdk.tar.gz`
- `-getmerge` 功能: 合并下载多个文件
 - `hadoop fs -getmerge /aaa/log.* ./log.sum` (假如 hdfs 的目录 /aaa/ 下有多个文件: log.1, log.2, log.3, ...)
- `-cp` 功能: 从 hdfs 的一个路径拷贝 hdfs 的另一个路径
 - `hadoop fs -cp /aaa/jdk.tar.gz /bbb/jdk.tar.gz.2`
- `-mv` 功能: 在 hdfs 目录中移动文件
 - `hadoop fs -mv /aaa/jdk.tar.gz /`
- `-rm` 功能: 删除文件或文件夹
 - `hadoop fs -rm -r /aaa/bbb/`
- `-appendToFile` 功能: 追加一个文件到已经存在的文件末尾
 - `hadoop fs -appendToFile ./hello.txt /hello.txt`
- `-cat` 功能: 显示文件内容
 - `hadoop fs -cat /hello.txt`
- `-tail` 功能: 显示一个文件的末尾
 - `hadoop fs -tail /weblog/access_log.1`
- `-text` 功能: 以字符形式打印一个文件的内容
 - `hadoop fs -text /weblog/access_log.1`
- `hdfs dfsadmin -report` 查看 dfs 集群工作状态的命令

2.3. HDFS 的 Java API 操作

利用 Eclipse 可视化 HDFS 上的文件信息

下载 eclipse

官网下载链接: <https://www.eclipse.org/downloads/>

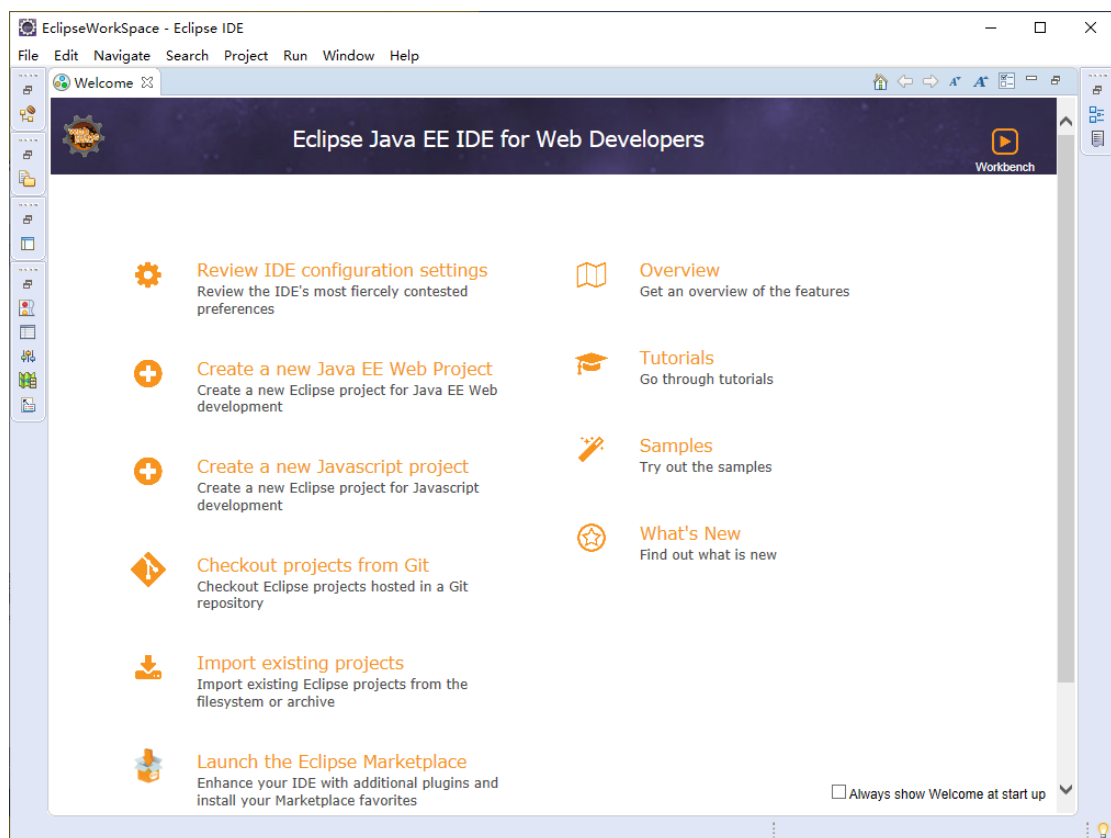
官网下载较慢, 可以使用镜像下载:

<https://mirrors.tuna.tsinghua.edu.cn/eclipse/technology/epp/downloads/release/>

eclipse-jee-2020-03-R-incubation-macosx-cocoa-x86_64.dmg.sha1	99 B	2020-03-14 11:41
eclipse-jee-2020-03-R-incubation-macosx-cocoa-x86_64.dmg.sha512	187 B	2020-03-14 11:41
eclipse-jee-2020-03-R-incubation-win32-x86_64.zip	400.4 MiB	2020-03-14 11:41
eclipse-jee-2020-03-R-incubation-win32-x86_64.zip.md5	84 B	2020-03-14 11:41
eclipse-jee-2020-03-R-incubation-win32-x86_64.zip.sha1	92 B	2020-03-14 11:41
eclipse-jee-2020-03-R-incubation-win32-x86_64.zip.sha512	180 B	2020-03-14 11:41

选择 eclipse-jee-xxx-win32-xxx 版本
下载解压，然后运行 exe

configuration	2020/3/30 19:10
dropins	2020/3/13 8:52
features	2020/3/30 19:00
p2	2020/3/30 19:28
plugins	2020/3/30 19:06
readme	2020/3/13 8:52
.eclipseproduct	2019/12/18 18:11
artifacts.xml	2020/3/13 8:51
eclipse.exe	2020/3/13 8:55
eclipse.ini	2020/3/13 8:52
eclipsesec.exe	2020/3/13 8:55

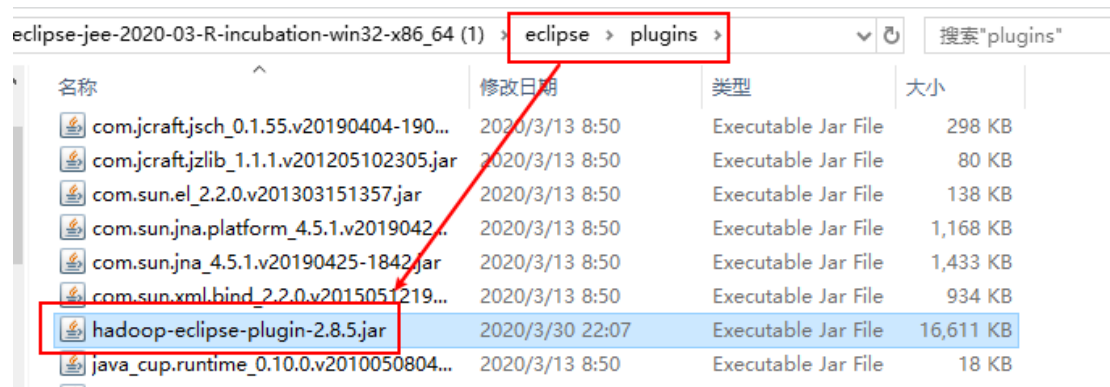


编译 hadoop-eclipse 插件

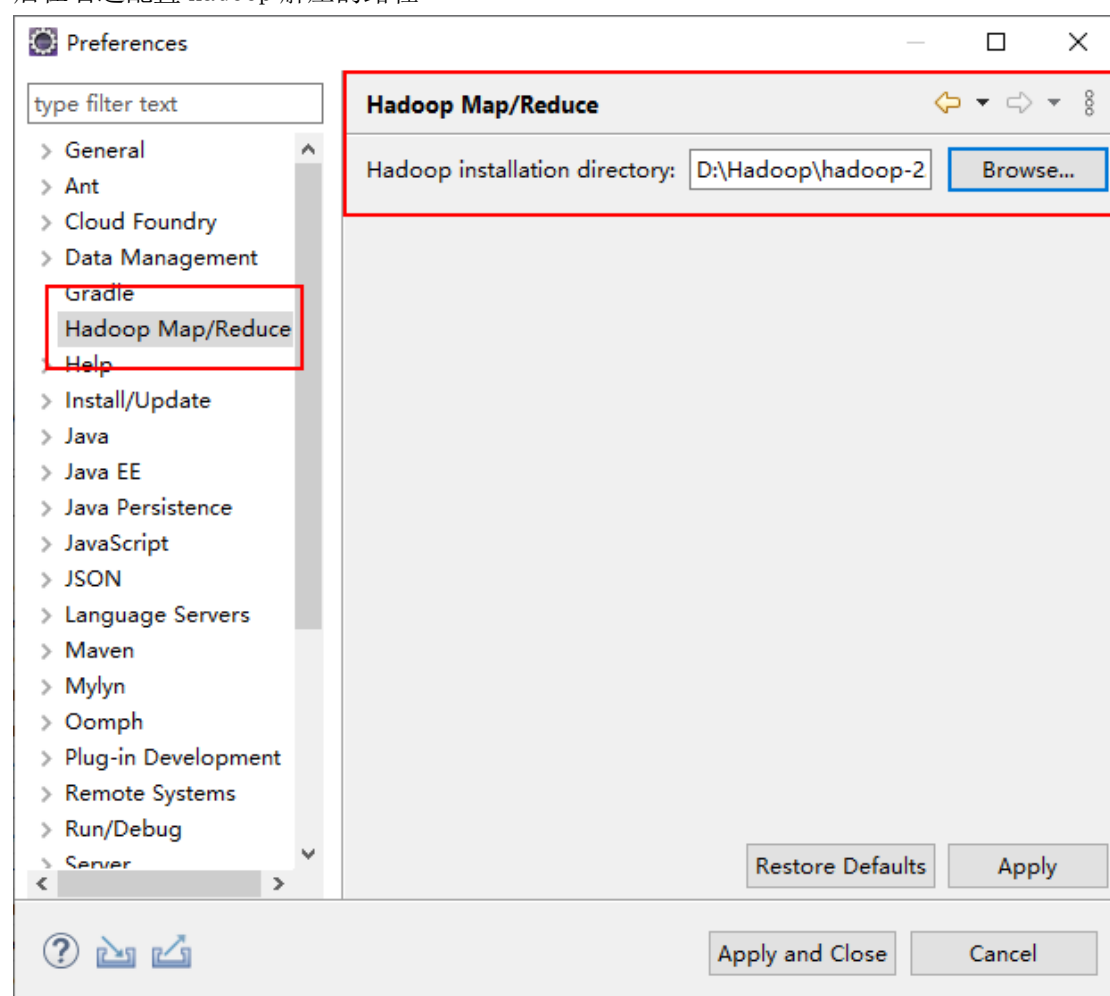
可以下载 ant 自行编译对应环境的插件，本次用的编译好的插件。

下载链接: <https://github.com/DoubleBirdsU/Hadoop-eclipse-plugin>

将 hadoop-eclipse-plugin-2.8.5.jar 插件包放到 eclipse/plugins 文件夹下

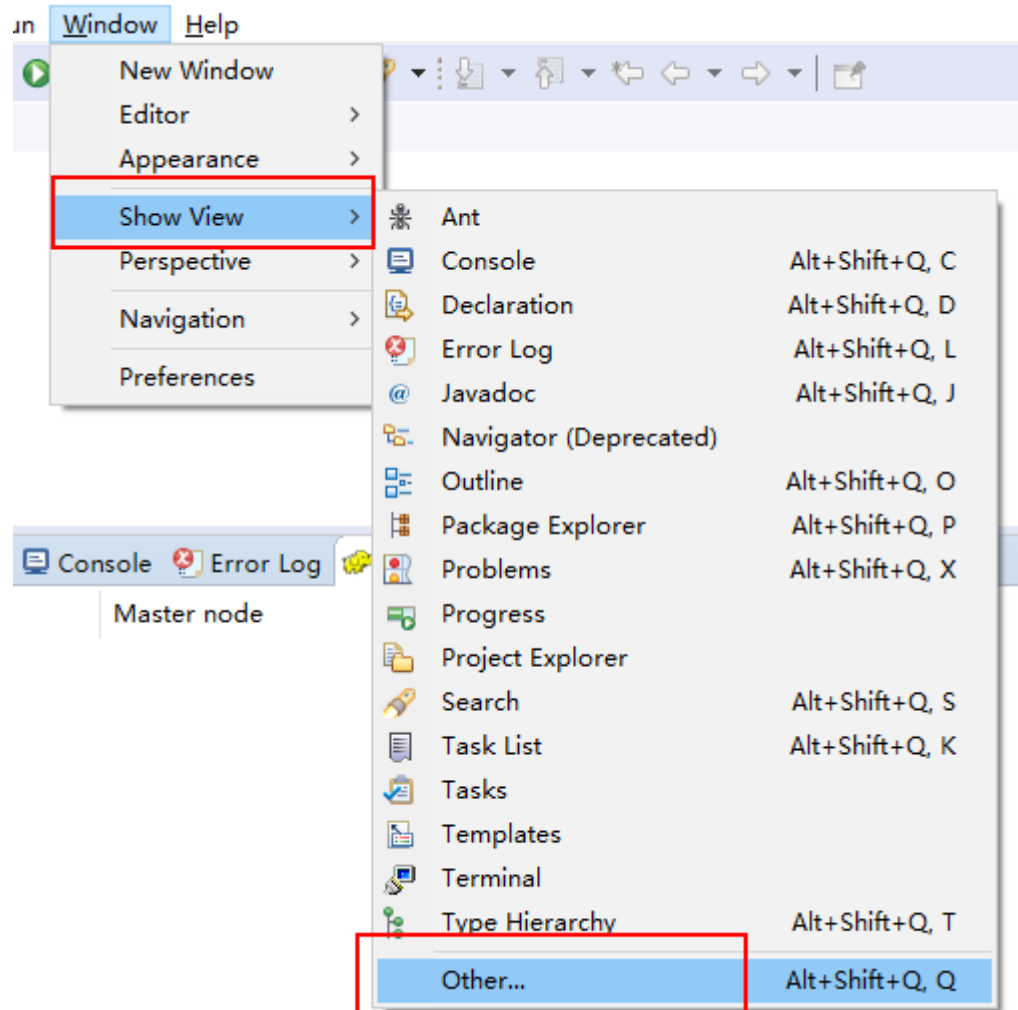


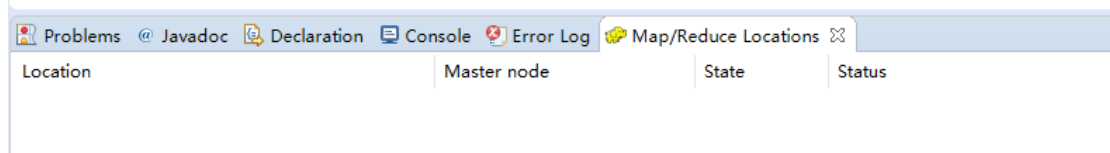
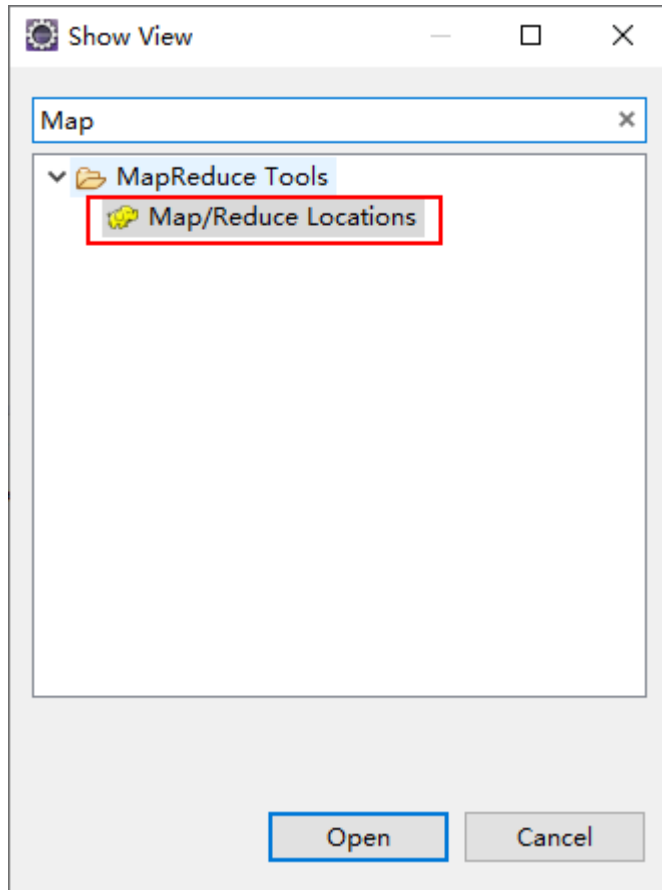
启动 eclipse ，点击 Window -> Preferences ，在弹出的框选中 Hadoop Map/Reduce ，然后在右边配置 hadoop 解压的路径



切换"Map/Reduce"工作目录

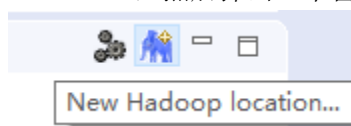
选择"Window"菜单下选择"Show view"，选择 others，弹出一个窗体，搜索 Map，从中选择"Map/Reduce"选项即可进行切换





建立与 Hadoop 集群的连接

在 Eclipse 软件下面的“Map/Reduce Locations”进行右击，弹出一个选项，选择“New Hadoop Location”，然后弹出一个窗体。



New Hadoop location...

Define Hadoop location

Define the location of a Hadoop infrastructure for running MapReduce applications.

General

Advanced parameters

Location name: Hadoop

Map/Reduce(V2) Master

Host: localhost

Port: 9001

DFS Master

☒ Use M/R Master host

Host: localhost

Port: 9000

User name: 崔荣成

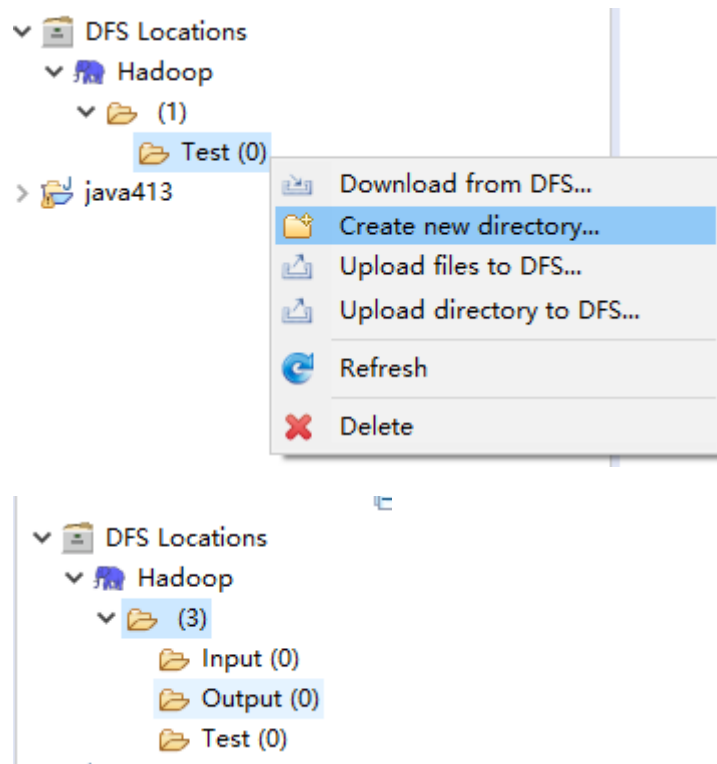
SOCKS proxy

☐ Enable SOCKS proxy

Host: host

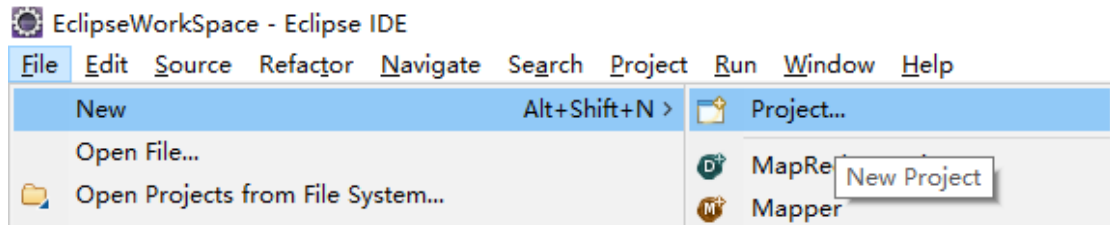
Port: 1080

连接成功后可以看到 HDFS 结构，因为第一次连接所以没有东西，可以右击闯进文件夹。

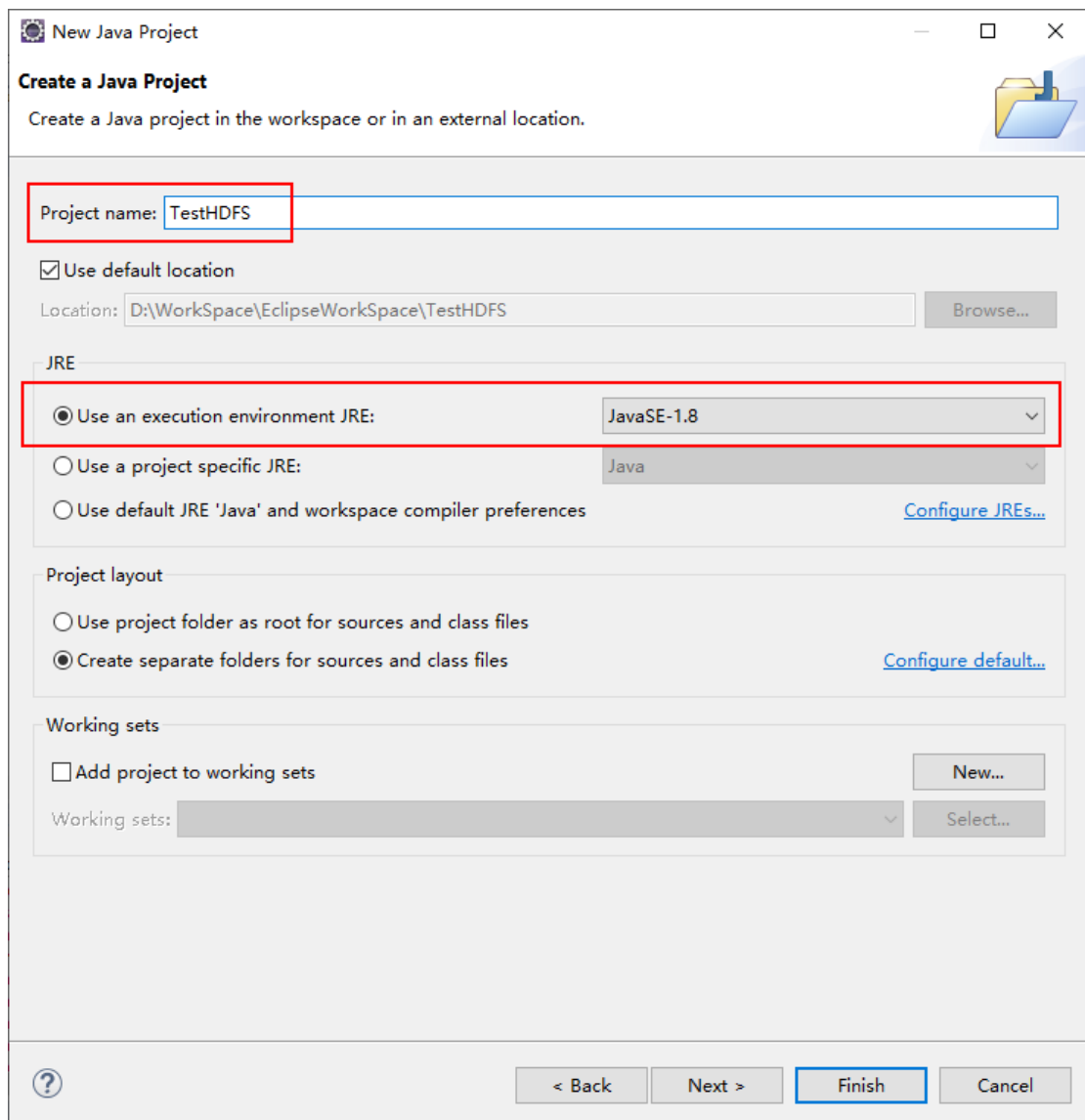


3. Eclipse 编程进行 hdfs 常用操作

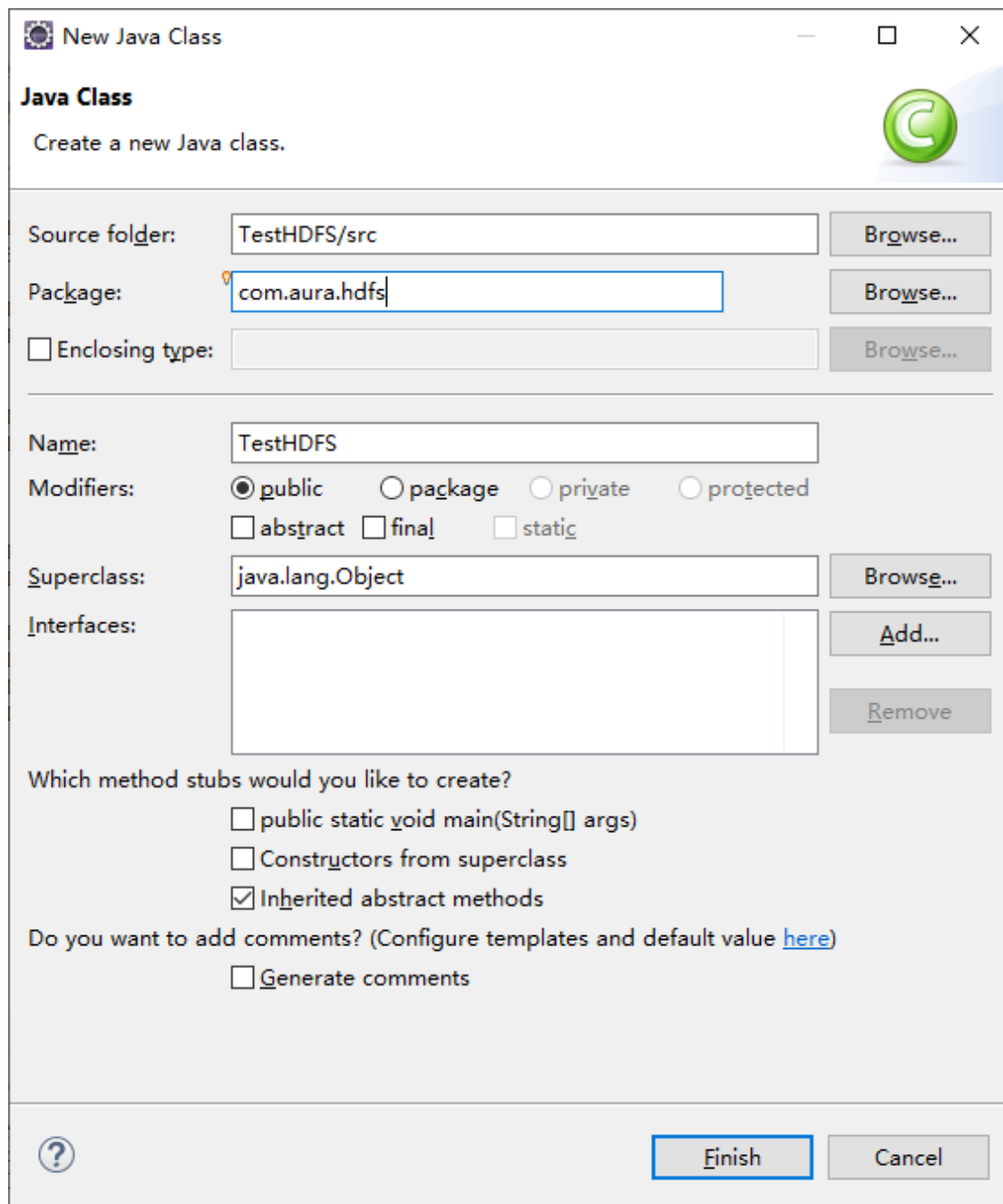
3.1. 创建 TestHDFS 程序



选择 Java 程序

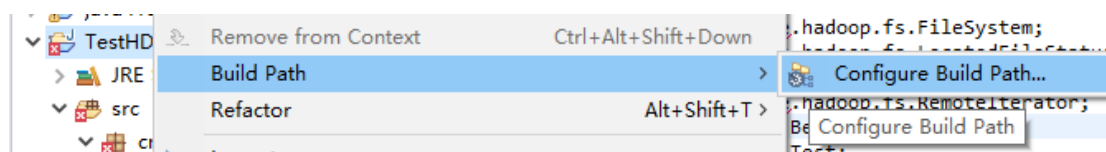


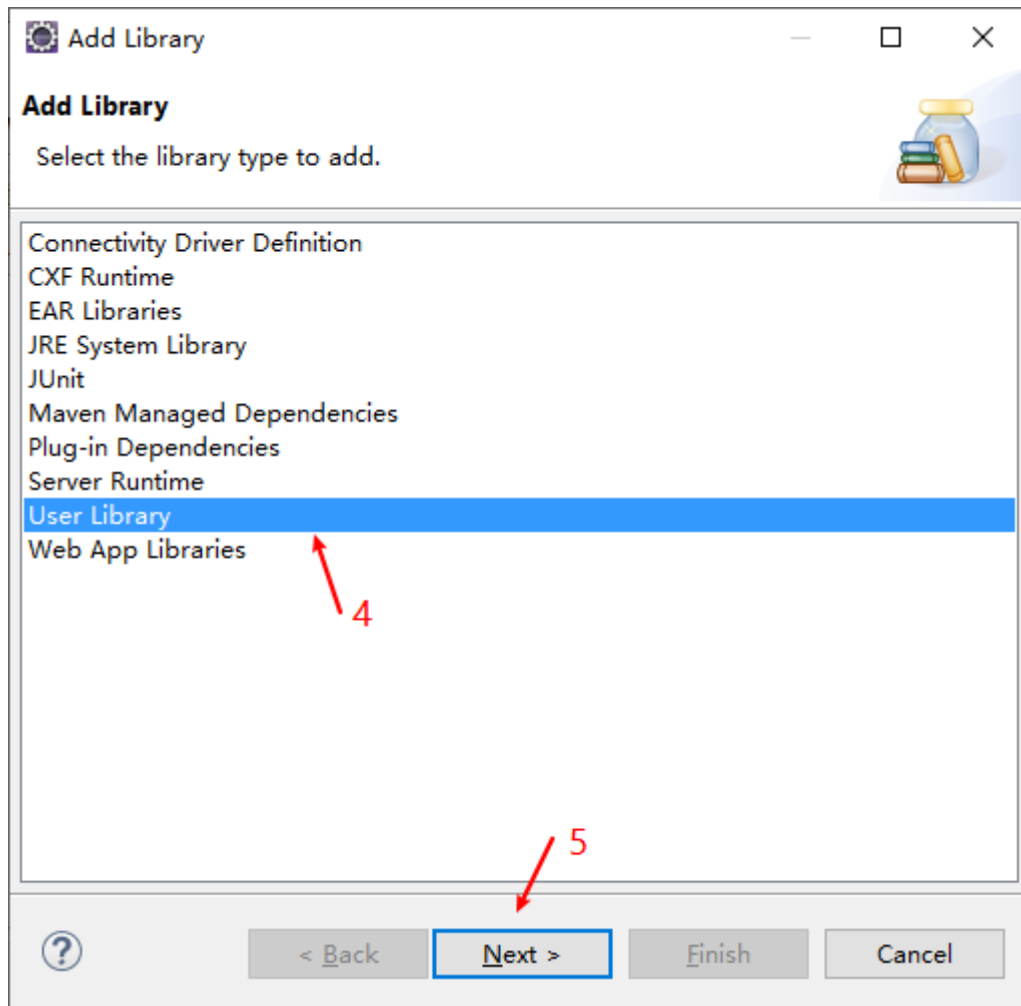
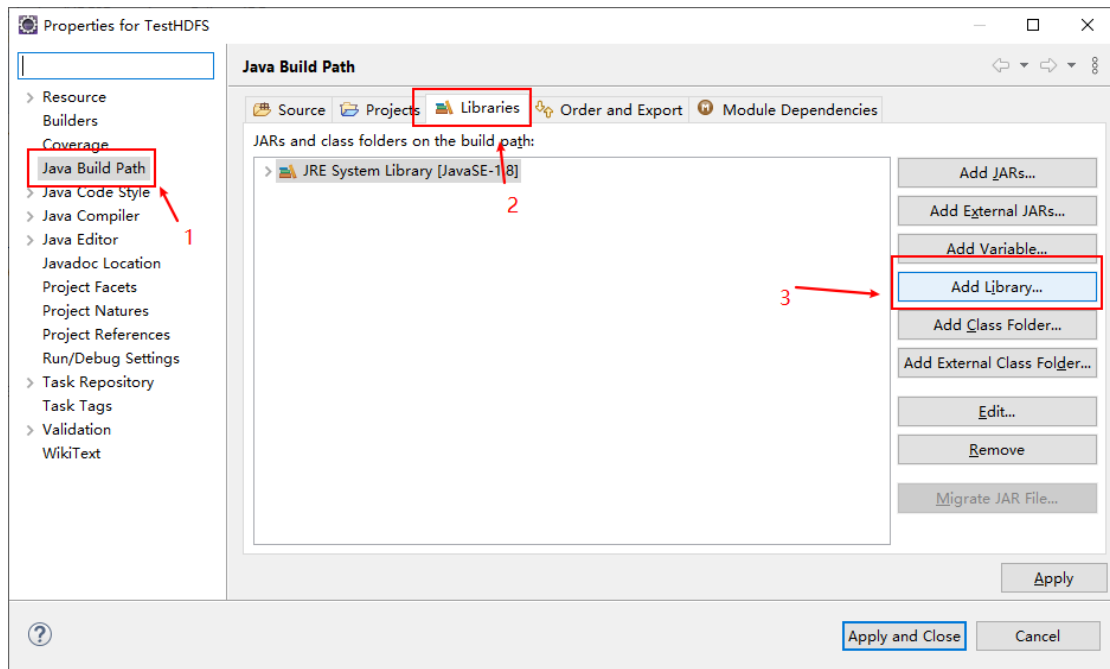
3.2. 创建 TestHDFS 类



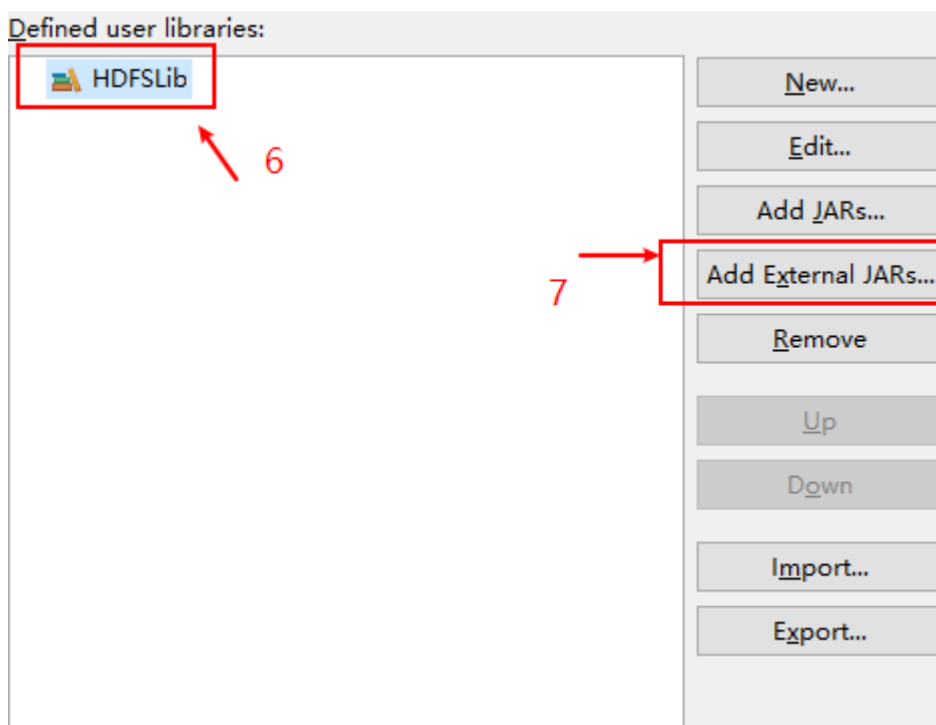
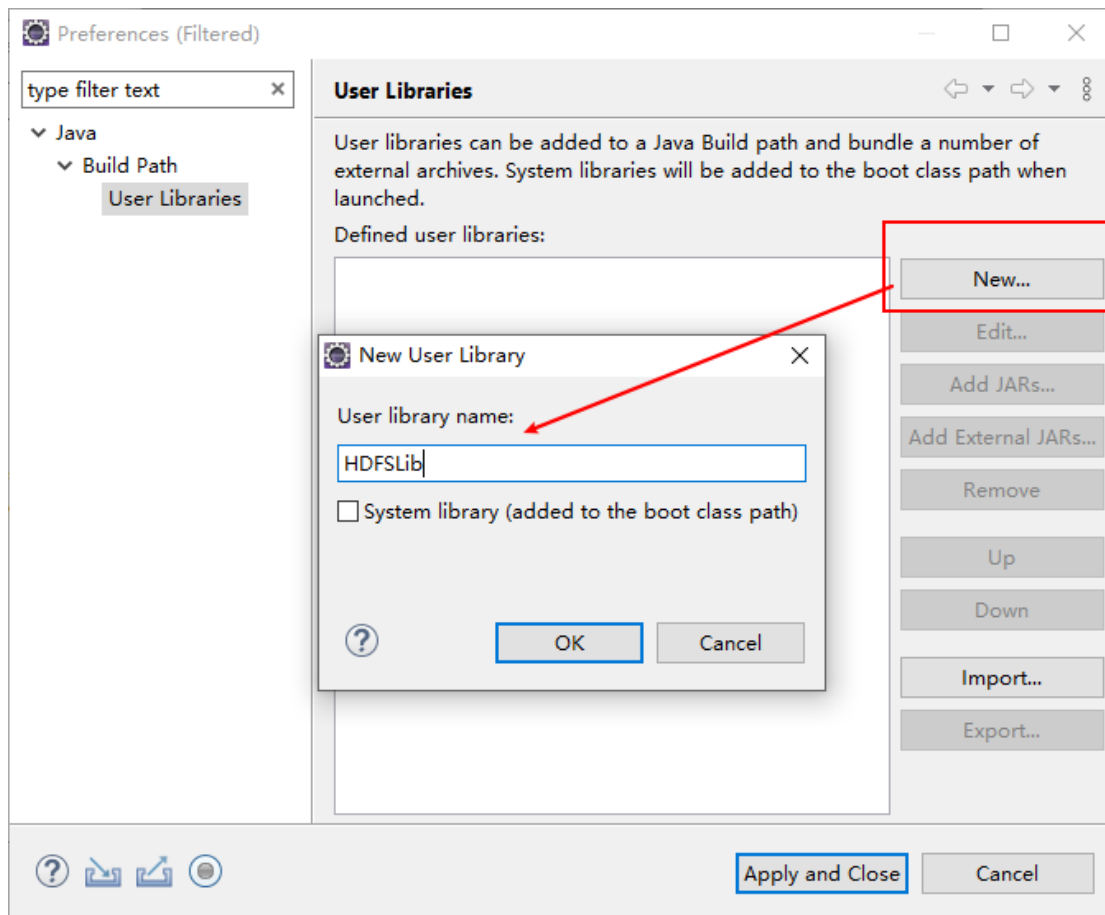
3.3. 添加依赖库

右击项目，选择 build path，进行配置

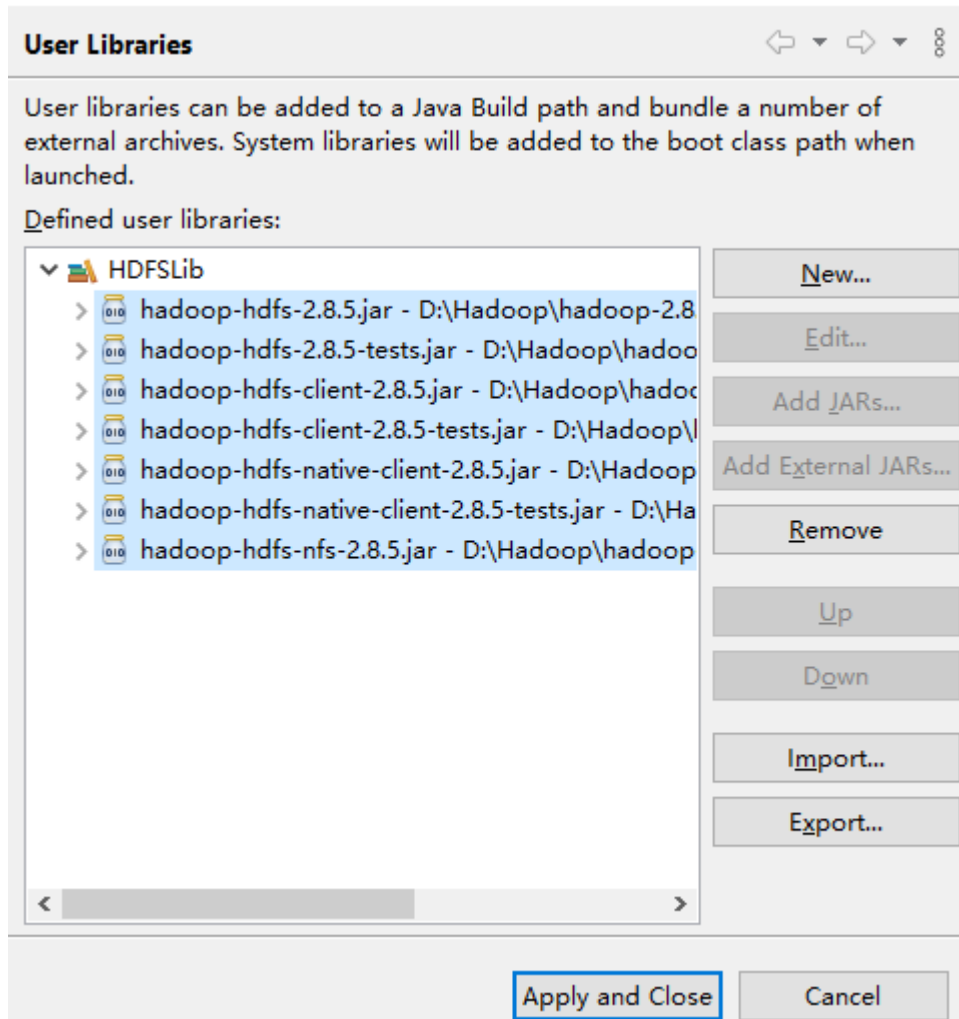




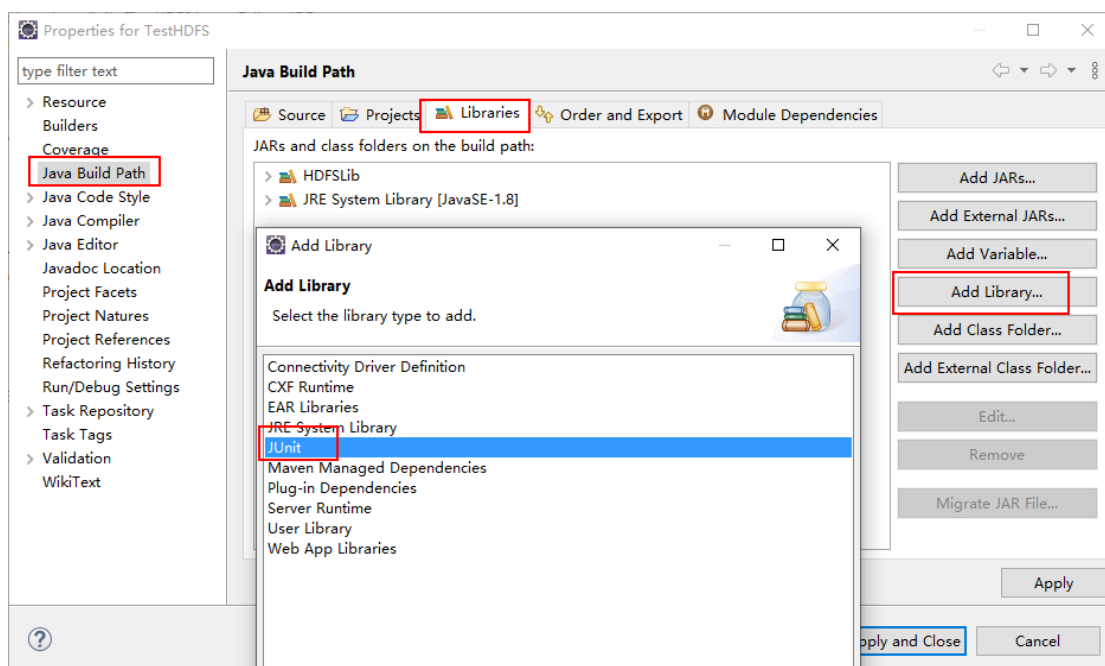
接下来选中 User Libraries -> New，输入 HDFS 库的名字

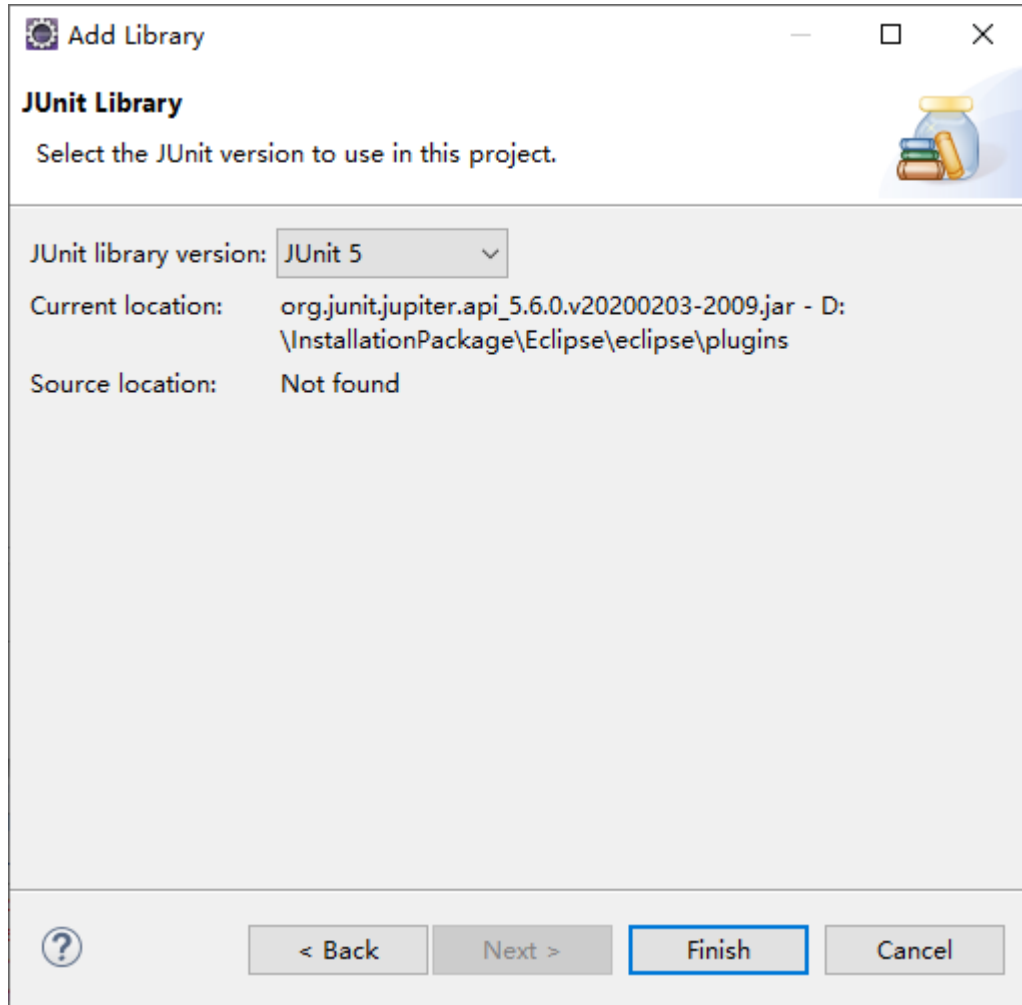


然后把 hadoop 目录下/share/hadoop/hdfs 的所有 jar 包放进去



3.4. 添加 junit 测试单元





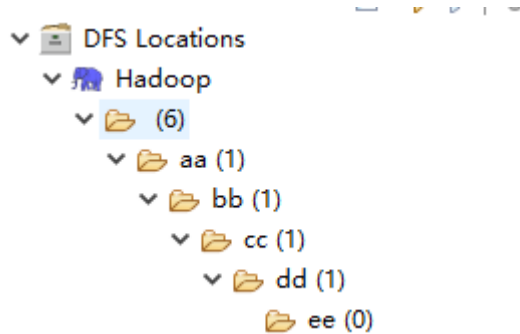
3.5. 创建 fs 对象

```
Configuration conf = new Configuration();
FileSystem fs;

/**
 * 创建fs对象
 */
@Before
public void getfs() throws Exception {
    conf = new Configuration();
    // conf.set("fs.defaultFS", "hdfs://localhost:9000");
    // System.setProperty("HADOOP_USER_NAME", "crc");
    // fs = FileSystem.get(conf);
    fs = FileSystem.get(new URI("hdfs://local:9000"), conf, "crc");
}
```

3.6. 创建文件夹

```
/**
 * 创建文件夹
 */
@Test
public void testMkdir() throws Exception {
    System.out.println(fs.mkdirs(new Path("/aa/bb/cc/dd/ee")));
}
```



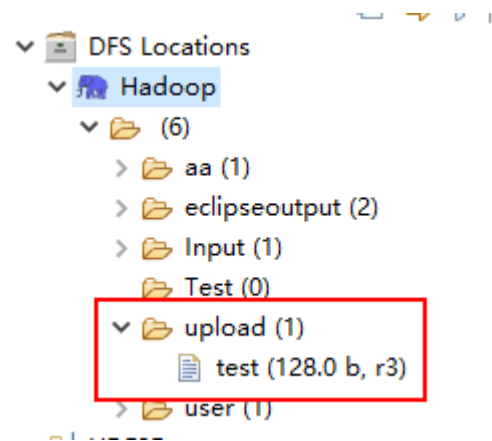
3.7. 删除文件

```
/**
 * 删除文件
 */
@Test
public void testDelete() throws Exception {
    boolean delete = fs.delete(new Path("/aa/bb/cc/dd/ee"), true);
    System.out.println(delete ? "yes" : "no");
    fs.close();
}
```

```
<terminated> Demo (1
true
yes
```

3.8. 上传文件

```
/**
 * 上传文件
 */
@Test
public void upload() throws Exception {
    fs.copyFromLocalFile(new Path("C:\\Users\\李荣成\\Desktop\\words.txt"), new Path("/upload/test"));
    fs.close();
}
```



3.9. 下载文件

```
/**
 * 下载文件
 */
@Test
public void download() throws Exception {
    fs.copyToLocalFile(new Path("/upload/test"), new Path("c:/"));
    fs.close();
}
```

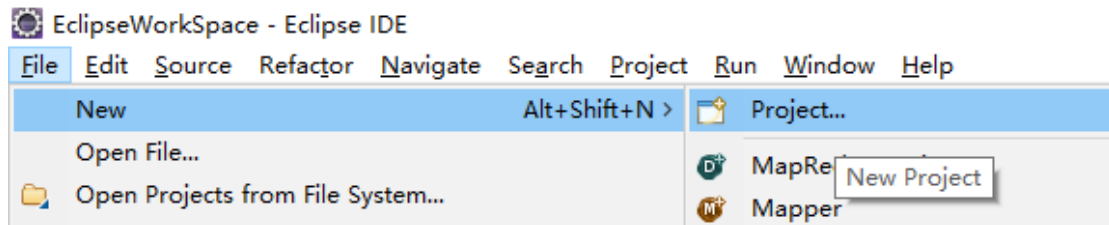
3.10. 查看文件及文件夹信息

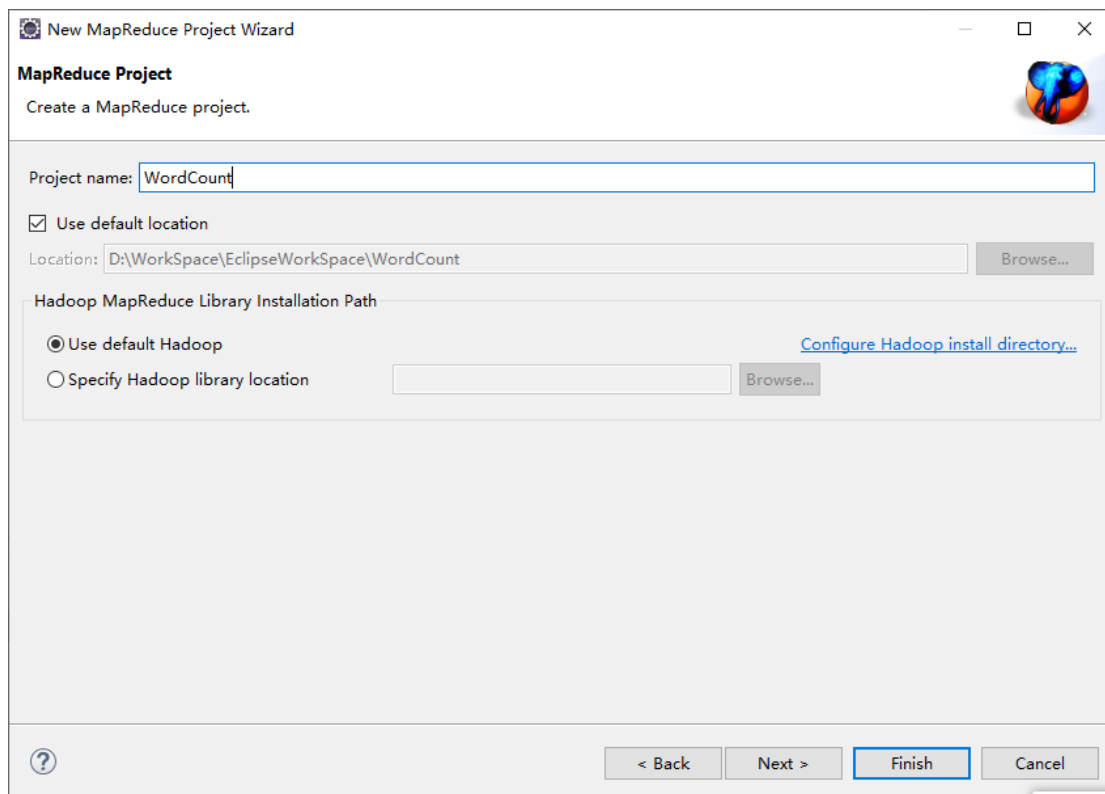
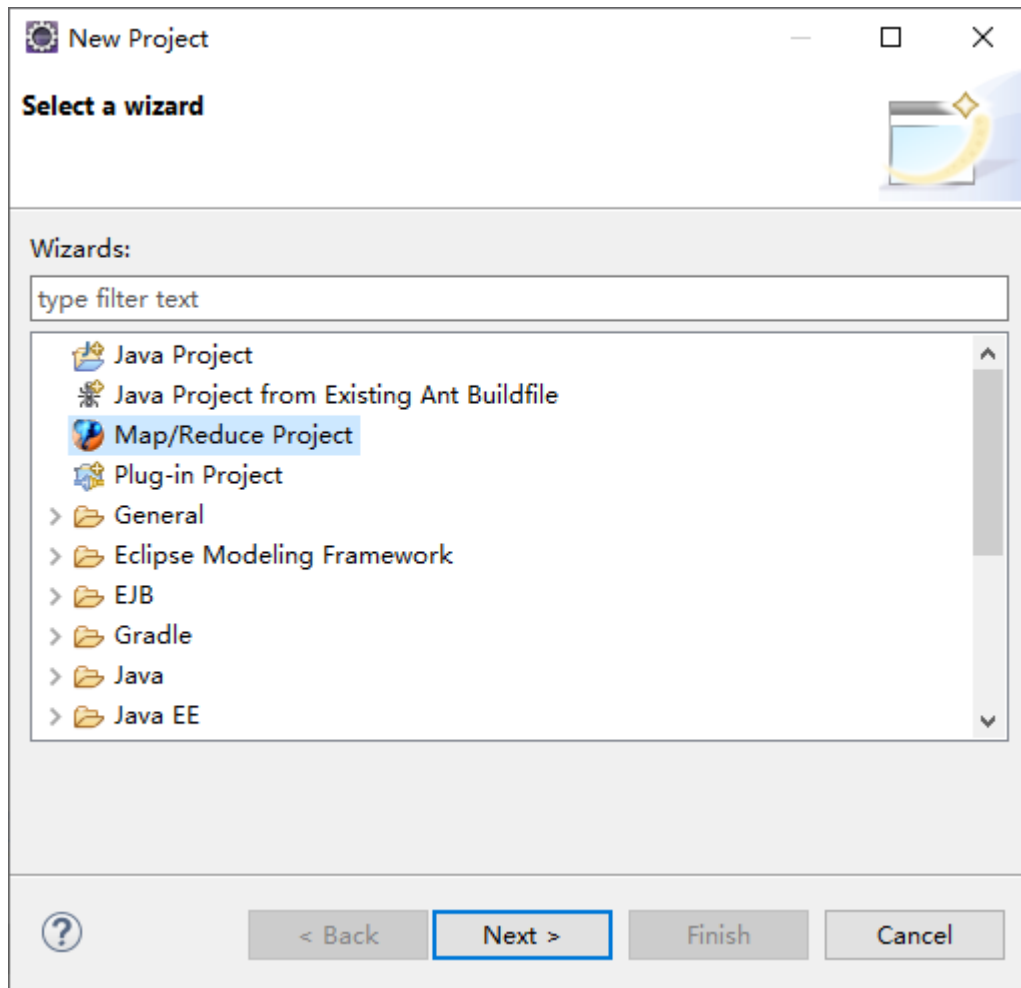
```
/**
 * 查看文件及文件夹信息
 */
@Test
public void testListStatus() throws Exception {
    FileStatus[] listStatus = fs.listStatus(new Path("/"));
    String flag = "";
    for (FileStatus status : listStatus) {
        if (status.isDirectory()) {
            flag = "文件夹";
        } else {
            flag = "文件";
        }
        System.out.println(status.getPath().getName() + "\t" + flag);
    }
    fs.close();
}
```

```
<terminated> Demo (1) [Java App]
true
yes
Input    文件夹
Test     文件夹
aa       文件夹
eclipseoutput  文件夹
upload   文件夹
user     文件夹
```

4. 使用 eclipse 创建并运行 wordcount 程序

4.1. 创建 MapReduce 程序

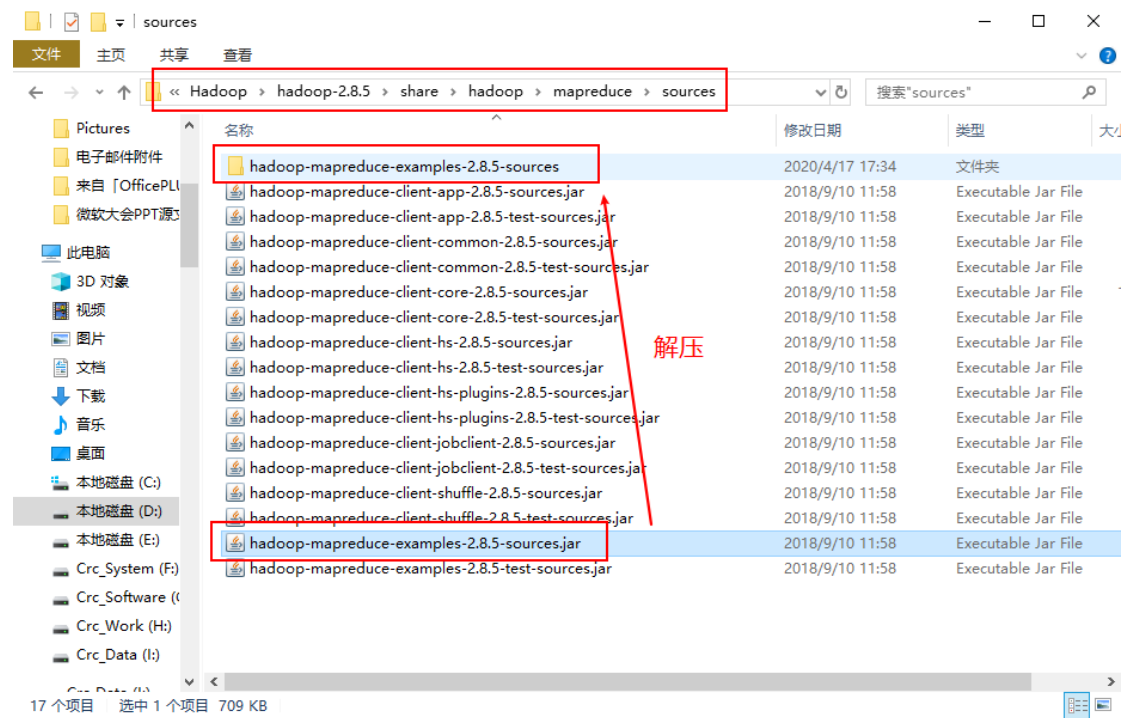




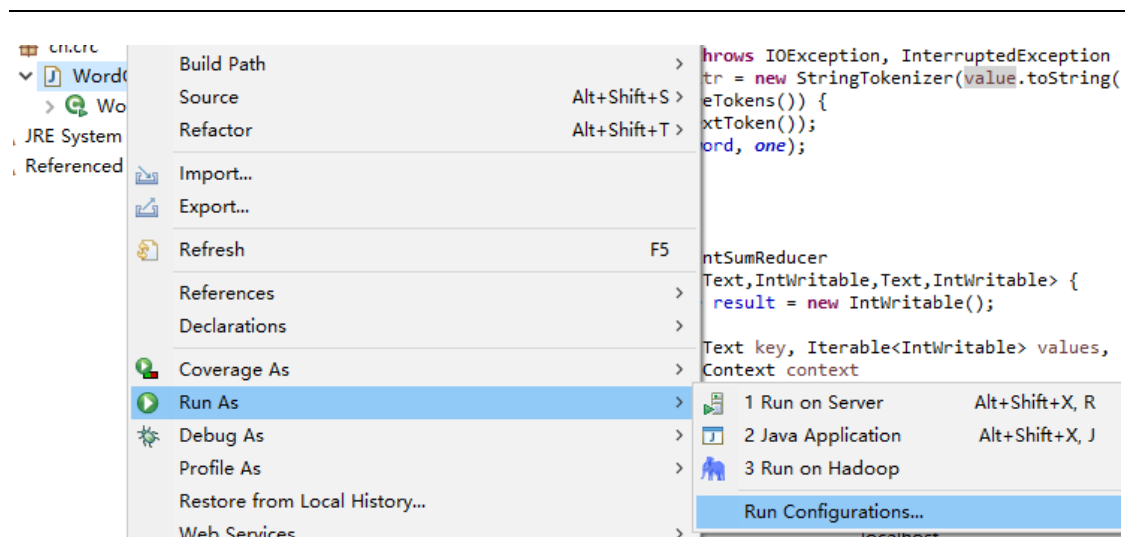
4.2. 编写 WordCount 程序

可以自己网上搜索代码

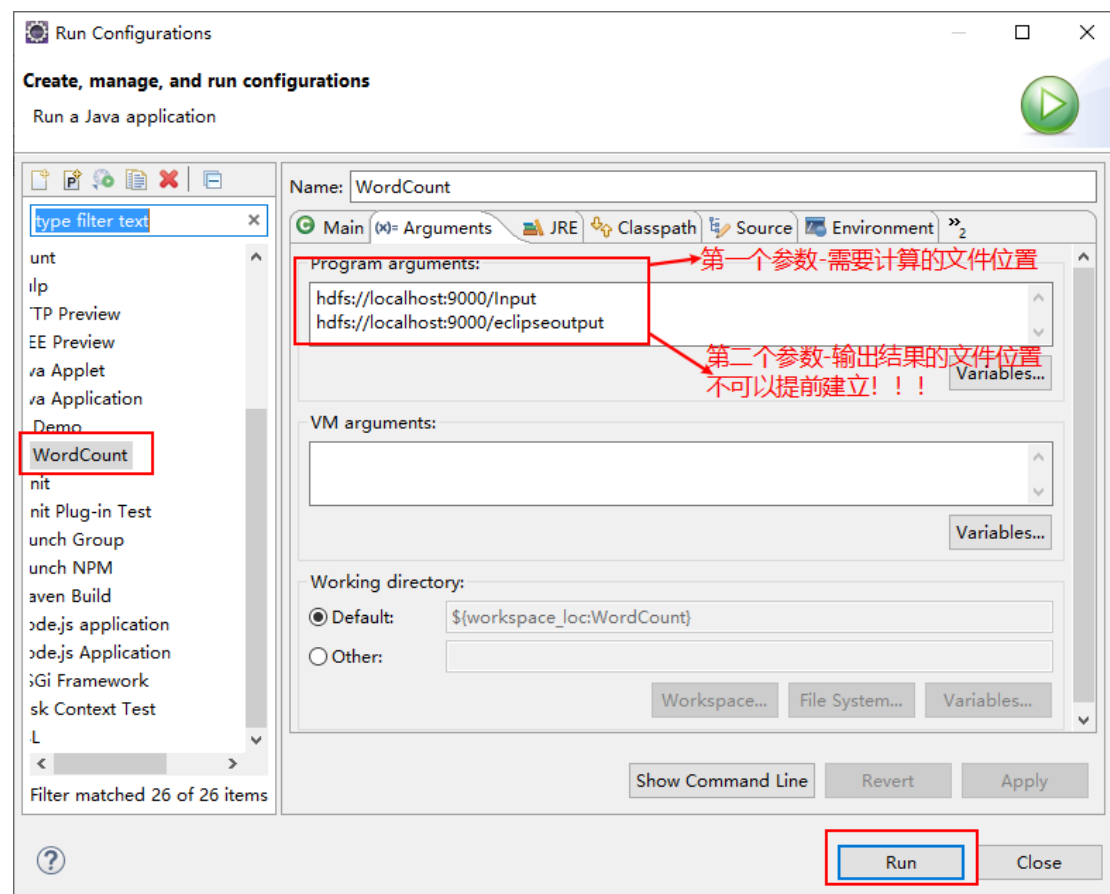
或者到 Hadoop 路径下 share/Hadoop/mapreduce/sources 下解压对应版本的 examples



在解压文件的 `org\apache\hadoop\examples` 这个文件夹下面找到 `wordcount` 程序



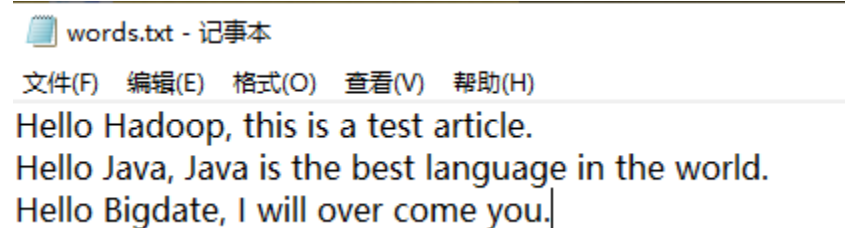
4.3. 配置参数



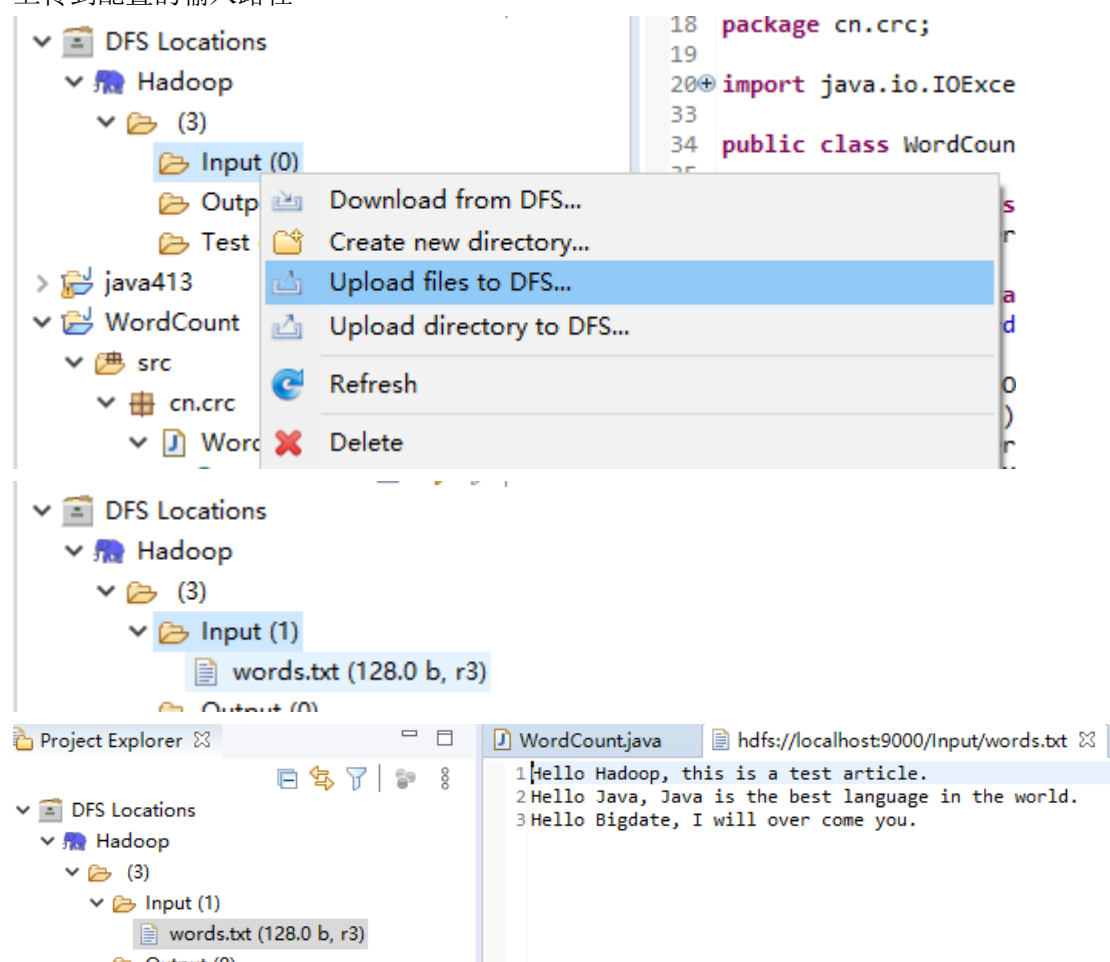
log4j:WARN No appenders could be found for logger (org.apache.hadoop.metrics2.lib.MutableMetricsFactory).
 log4j:WARN Please initialize the log4j system properly.
 log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.

运行显示没有输入文件。

创建一个待统计的文件



上传到配置的输入路径



4.4. 运行结果

运行之后可以看到在 HDFS 文件下创建了结果文件夹，同时出现了统计结果。

Project Explorer

- DFS Locations
 - Hadoop
 - (3)
 - eclipseoutput (2)
 - _SUCCESS (0.0 b, r3)
 - part-r-00000 (148.0 b, r3)
 - Input (1)
 - Test (0)
- java413
- WordCount
 - src
 - cn.crc
 - WordCount.java
 - WordCount
 - JRE System Library [Java]
 - Referenced Libraries

hdfs://loca...

```
1 bigdate, 1
2 Hadoop, 1
3 Hello 3
4 I 1
5 Java 1
6 Java, 1
7 a 1
8 article. 1
9 best 1
10 come 1
11 in 1
12 is 2
13 language 1
14 over 1
15 test 1
16 the 2
17 this 1
18 will 1
19 world. 1
20 you. 1
21
```