**一、jdk环境配置**

从官网下载最新的jdk1.8，双击安装即可。注意mac系统自带的jdk最高是1.6，

1. **全局配置：在/etc/profile中导入java路径：**

$ sudo su

#输入密码

$ vi /etc/profile

#配置JAVA\_HOME，此处路径根据自己的版本填写

JAVA\_HOME="/System/Library/Java/JavaVirtualMachines/1.8.0\_25.jdk/Contents/Home/"

CLASS\_PATH="$JAVA\_HOME/lib"

#把JAVA添加到到环境变量PATH中

PATH=".:$PATH:$JAVA\_HOME/bin"

export JAVA\_HOME

1. **为当前用户配置（只对当前用户有效）**

$ cd ~

$ vi .bash\_profile

在打开的文件最后导入java路径

export JAVA\_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0\_25.jdk/Contents/Home

命令行查看java版本：

$ java –version

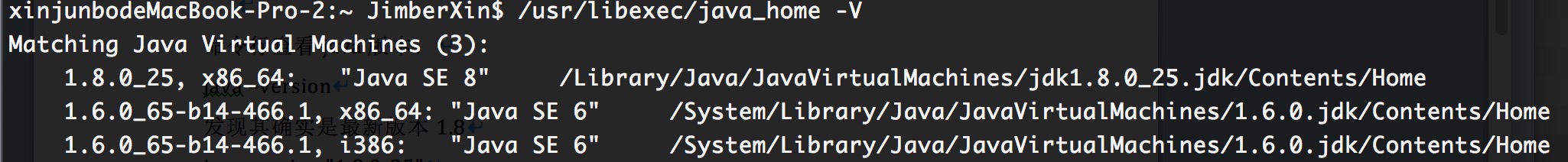
发现其确实是最新版本1.8

java version "1.8.0\_25"

Java(TM) SE Runtime Environment (build 1.8.0\_25-b17)

Java HotSpot(TM) 64-Bit Server VM (build 25.25-b02, mixed mode)

也可以用 /usr/libexec/java\_home –V查看所有的java版本



**二、安装ssh**

(1)安装ssh，方法网上自己找

(2)配置为免密码登陆本机

|  |
| --- |
| $ ssh-keygen -t dsa -P '' -f ~/.ssh/id\_dsa  其中，ssh-keygen代表生成密匙；-t（小写）表示指定生成的密匙类型；dsa是dsa密匙认证  的意思，即密钥类型；-P用于提供密语，这里表示免密码登陆，所以是’’, -f指定生成的密钥文件，  放在用户根文件夹下。此处~为用户目录/Users/JimberXin.  该命令后，在.ssh文件夹下创建id\_dsa以及id\_dsa.pub两个文件。这是ssh的一对私钥和公钥  类似钥匙和锁，接着，把id\_dsa.pub(公钥)追加到授权的key中去  $ cat ~/.ssh/id\_dsa.pub >> ~/.ssh/authorized\_keys |

该命令把公钥加到用于认证的公钥文件中，这里指的是authorized\_keys

1. 验证SSH是否已经安装成功过，以及是否可以免密码登陆

$ ssh –version

显示结果

OpenSSH\_6.2p2, OSSLShim 0.9.8r 8 Dec 2011

Bad escape character 'rsion'.

输入命令

$ ssh localhost

显示结果

Last login: Wed Mar 4 16:46:36 2015 from localhost

至此，ssh安装成功

**三、安装并运行hadoop**

Hadoop 分别从3个角度将主机分为两种角色

1. 最基本的划分，Master与Slave
2. 从HDFS的角度， 将主机分为NameNode 和 DadaNode。在分布式文件系统中，目录管理很重要，管理目录相当于主人，NameNode就是目录管理者
3. 从MapReduce的角度，将主机分为JobTracker和TaskTracker

Hadoop有3中运行方式：单机模式、伪分布式以及全分布式。其中单机模式无需配置，hadoop被认为是一个单独的java进程，主要用于调试。伪分布式的hadoop可以看成只有一个节点的集群，在这个集群中，该节点既是Master也是Slave， 既是NameNode也是DataNode， 既是JobTracker也是TaskTracker，这里搭建的环境就是指的伪分布式。

1. **指定JDK的安装位置**

打开etc/hadoop/hadoop-env.sh，增加如下代码

# The java implementation to use.

export JAVA\_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0\_25.jdk

**（2）Hadoop核心配置文件，配置HDFS(Hadoop分布式文件系统)地址与端口**

打开etc/hadoop/core-site.xml，增加如下代码

<configuration>

<property>

<name>hadoop.tmp.dir</name>

<value>/Users/JimberXin/hadoop-2.6.0/temp</value>

<description>A base for other temporary directory</description>

</property>

<property>

<name>fs.default.name</name>

<value>hdfs://localhost:9000</value>

</property>

</configuration>

其中，hadoop.tmp.dir指定hadoop的临时文件；fs.default.name是HDFS的地址与端口号，单机是file:///; 伪分布式是 hdfs://localhost:9000; 全分布式是hdfs://namenode

**(3) 配置HDFS的配置，默认复制个数为3， 修改为1**

打开etc/hadoop/hdfs-site.xml, 增加以下代码：

<configuration>

<property>

<name>dfs.replication</name>

<value>1</value>

</property>

</configuration>

**（4）配置MapReduce的配置**

hadoop2.6.0默认是没有mapred-site.xml的，自己创建 mapred-site.xml，配置JobTracker的地址及端口号，增加如下代码：

<configuration>

<property>

<name>mapred.job.tracker</name>

<value>hdfs://localhost:9001</value>

</property>

</configuration>

其中，mapred.job.tracker是JobTracker的地址以及端口，单机版是local，伪分布式是localhost:9001, 全分布式是jobtracker:9001。

**(5)配置yarn。Hadoop2.x以后MapReduce框架并入yarn框架，需要进行配置**

打开etc/hadoop/yarn-site.xml，增加如下代码

<configuration>

<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

<property>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce\_shuffle</value>

</property>

</configuration>

至此，基本的配置文件都已经修改完毕，准备启动Hadoop。在此之前，需要格式化Hadoop的文件系统HDFS。进入Hadoop文件夹，输入如下命令：

$ bin/hadoop namenode –format

此命令格式化文件系统，接下来启动Hadoop。

输入命令，启动所有进程

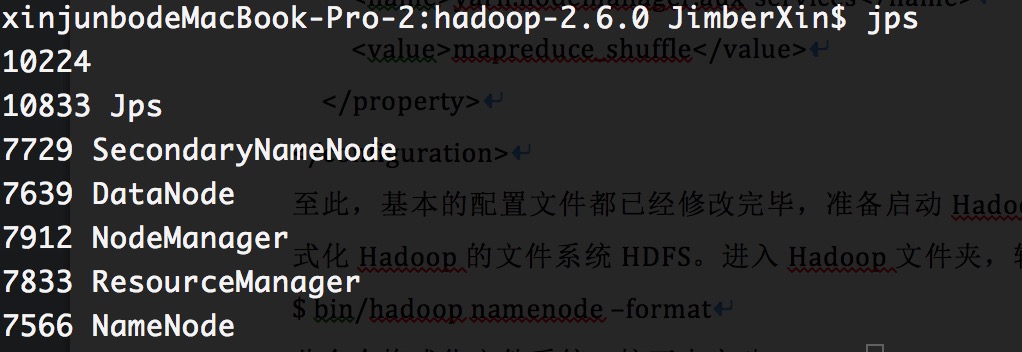
bin/start-all.sh

关闭所有进程

bin/stop-all.sh

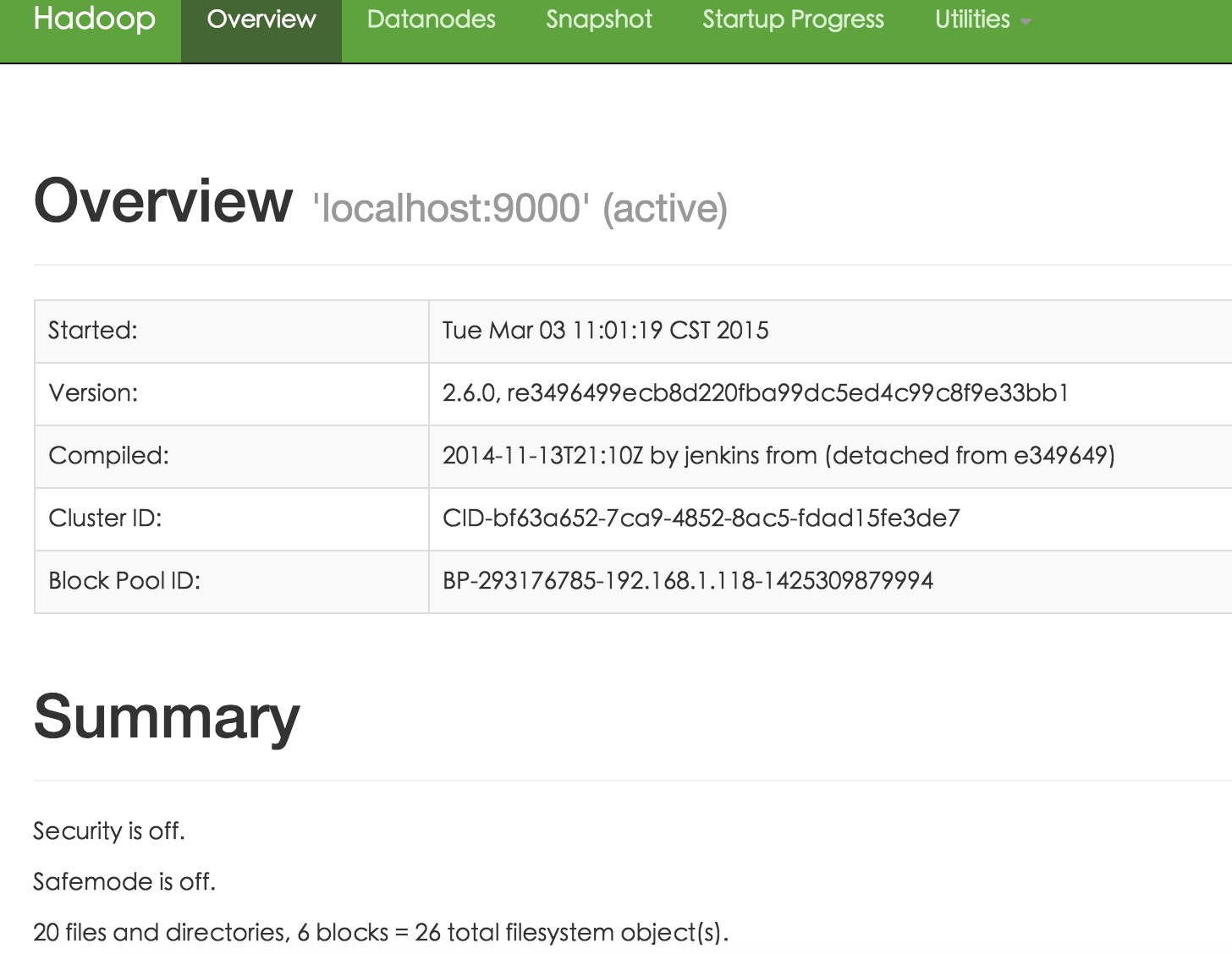
也可以单独开启或者关闭某些进程，例如sbin/start-dfs.sh，sbin/start-yarn.sh，sbin/stop-yarn.sh等。

启动了所有进程后，命令行输入jps，用于查看当前的java进程，显示如下：

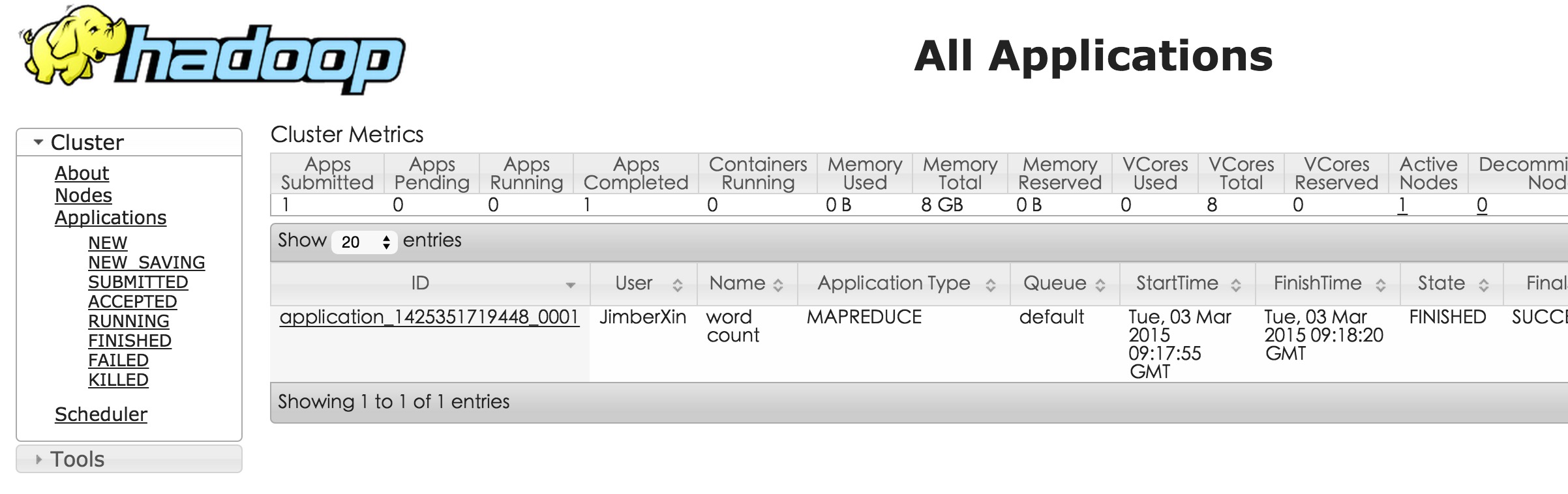


最后，验证Hadoop是否安装成功。打开浏览器，分别查看HDFS以及MapReduce的页面是否能打开

HDFS的页面：http://localhost:50070



MapReduce的页面:



（6）查看集群状态

输入命令

bin/hdfs dfsadmin –report

显示结果如下：

15/03/04 19:18:08 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

Configured Capacity: 120154296320 (111.90 GB)

Present Capacity: 22928572416 (21.35 GB)

DFS Remaining: 22928371712 (21.35 GB)

DFS Used: 200704 (196 KB)

DFS Used%: 0.00%

Under replicated blocks: 0

Blocks with corrupt replicas: 0

Missing blocks: 0

-------------------------------------------------

Live datanodes (1):

Name: 127.0.0.1:50010 (localhost)

Hostname: 192.168.1.118

Decommission Status : Normal

Configured Capacity: 120154296320 (111.90 GB)

DFS Used: 200704 (196 KB)

Non DFS Used: 97225723904 (90.55 GB)

DFS Remaining: 22928371712 (21.35 GB)

DFS Used%: 0.00%

DFS Remaining%: 19.08%

Configured Cache Capacity: 0 (0 B)

Cache Used: 0 (0 B)

Cache Remaining: 0 (0 B)

Cache Used%: 100.00%

Cache Remaining%: 0.00%

Xceivers: 1

Last contact: Wed Mar 04 19:18:09 CST 2015

可知DataNode的个数以及运行状态是对的

四、运行hadoop例子WordCount

1、准备工作

（1）创建本地示例文件

首先在hadoop文件夹下，创建输入文件夹input

$ mkdir input

接着在input文件夹下创建两个文本文件f1和f2

输入命令

$ echo “Hello World Bye World” > f1

$ echo “Hello Hadoop Bye Hadoop” > f2

1. 创建HDFS上的文件夹

  Hdfs是hadoop Distributed file system的缩写，是hadoop的分布式文件系统。

       Hdfs由hadoop来管理，它不同于普通的文件系统，不能直观的查找文件，必须要通过hadoop命令操作hdfs。

**HDFS是一个主从结构的体系，一个HDFS集群是由一个名字节点，它是一个管理文件的命名空间和调节客户端访问文件的主服务器**，当然还有的数据节点，一个节点一个，它来管理存储。HDFS暴露文件命名空间和允许用户数据存储成文件。

　　内部机制是将一个文件分割成一个或多个的块，这些块存储在一组数据节点中。名字节点操作文件命名空间的文件或目录操作，如打开，关闭，重命名，等等。它同时确定块与数据节点的映射。数据节点来负责来自文件系统客户的读写请求。

　　数据节点同时还要执行块的创建，删除，和来自名字节点的块复制指示。

一些常见HDFS的命令

hadoop fs -mkdir /tmp/input              在HDFS上新建文件夹

hadoop fs -put input1.txt /tmp/input  把本地文件input1.txt传到HDFS的/tmp/input目录下

   hadoop fs -get  input1.txt /tmp/input/input1.txt  把HDFS文件拉到本地

    hadoop fs -ls /tmp/output                  列出HDFS的某目录  
    hadoop fs -cat /tmp/ouput/output1.txt  查看HDFS上的文件

    hadoop fs -rmr /home/less/hadoop/tmp/output  删除HDFS上的目录

    hadoop dfsadmin -report 查看HDFS状态，比如有哪些datanode，每个datanode的情况

   hadoop dfsadmin -safemode leave  离开安全模式

    hadoop dfsadmin -safemode enter  进入安全模式

这里查看这个文件还有一个方法就是在网页中查看:进入[http://127.0.0.1:50070](http://127.0.0.1:50070/),也就是hdfs的管理系统，然后点击相应的文件目录就可以了.

这里需要注意:当查看hadoop文件系统的时候需要用命令hadoop fs –ls  这样前面加上hadoop fs 因为假如直接用ls的话,那么就是指模拟出来的linux文件系统中的东西

现在，可以开始在HDFS上创建文件夹了，输入命令

$ bin/hadoop fs –mkdir /temp

$ bin/hadoop fs –mkdir /temp/input

1. 从本地文件夹下HDFS文件夹上传文件

输入命令

$ bin/hadoop –put input /temp

该命令将本地文件input下的文件拷贝到HDFS文件夹上得/temp上

1. 检查HDFS是否已经存在拷贝后的文件

输入命令

$ bin/hadoop fs –ls /temp/input

显示结果

15/03/03 17:16:48 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

Found 2 items

-rw-r--r-- 1 JimberXin supergroup 22 2015-03-03 17:16 /temp/input/f1

-rw-r--r—1 JimberXin supergroup 28 2015-03-03 17:16 /temp/input/f2

可以发现，f1和f2确实已经在HDFS的文件夹/temp/input上了

1. 运行WordCount程序

利用hadoop2.6.0自带的hadoop-mapreduce-examples-2.6.0.jar，给定输入，来运行WordCount程序

bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar wordcount /temp/input/ /output/wordcount3

其中 bin/hadoop jar为执行jar命令；

share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar为WordCount所在的jar包；

wordcount为生成的程序主类名，自定义

/temp/input为程序的输入，这里包括f1和f2

/output/wordcount3为程序的输出文件

输出结果如下所示：

15/03/03 17:17:52 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

15/03/03 17:17:53 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032

15/03/03 17:17:54 INFO input.FileInputFormat: Total input paths to process : 2

15/03/03 17:17:54 INFO mapreduce.JobSubmitter: number of splits:2

15/03/03 17:17:54 INFO mapreduce.JobSubmitter: Submitting tokens for job: job\_1425351719448\_0001

15/03/03 17:17:55 INFO impl.YarnClientImpl: Submitted application application\_1425351719448\_0001

15/03/03 17:17:55 INFO mapreduce.Job: The url to track the job: http://xinjunbodeMacBook-Pro-2.local:8088/proxy/application\_1425351719448\_0001/

15/03/03 17:17:55 INFO mapreduce.Job: Running job: job\_1425351719448\_0001

15/03/03 17:18:08 INFO mapreduce.Job: Job job\_1425351719448\_0001 running in uber mode : false

15/03/03 17:18:08 INFO mapreduce.Job: map 0% reduce 0%

15/03/03 17:18:16 INFO mapreduce.Job: map 100% reduce 0%

15/03/03 17:18:22 INFO mapreduce.Job: map 100% reduce 100%

15/03/03 17:18:22 INFO mapreduce.Job: Job job\_1425351719448\_0001 completed successfully

15/03/03 17:18:22 INFO mapreduce.Job: Counters: 49

File System Counters

FILE: Number of bytes read=79

FILE: Number of bytes written=317173

FILE: Number of read operations=0

FILE: Number of large read operations=0

FILE: Number of write operations=0

HDFS: Number of bytes read=250

HDFS: Number of bytes written=41

HDFS: Number of read operations=9

HDFS: Number of large read operations=0

HDFS: Number of write operations=2

Job Counters

Launched map tasks=2

Launched reduce tasks=1

Data-local map tasks=2

Total time spent by all maps in occupied slots (ms)=10968

Total time spent by all reduces in occupied slots (ms)=3183

Total time spent by all map tasks (ms)=10968

Total time spent by all reduce tasks (ms)=3183

Total vcore-seconds taken by all map tasks=10968

Total vcore-seconds taken by all reduce tasks=3183

Total megabyte-seconds taken by all map tasks=11231232

Total megabyte-seconds taken by all reduce tasks=3259392

Map-Reduce Framework

Map input records=2

Map output records=8

Map output bytes=82

Map output materialized bytes=85

Input split bytes=200

Combine input records=8

Combine output records=6

Reduce input groups=5

Reduce shuffle bytes=85

Reduce input records=6

Reduce output records=5

Spilled Records=12

Shuffled Maps =2

Failed Shuffles=0

Merged Map outputs=2

GC time elapsed (ms)=215

CPU time spent (ms)=0

Physical memory (bytes) snapshot=0

Virtual memory (bytes) snapshot=0

Total committed heap usage (bytes)=513802240

Shuffle Errors

BAD\_ID=0

CONNECTION=0

IO\_ERROR=0

WRONG\_LENGTH=0

WRONG\_MAP=0

WRONG\_REDUCE=0

File Input Format Counters

Bytes Read=50

File Output Format Counters

Bytes Written=41

Hadoop命令会启动一个JVM来运行这个MapReduce程序，并自动获得Hadoop的配置，同时把类的路径（及其依赖关系）加入到Hadoop的库中。以上就是Hadoop Job的运行记录，从这里可以看到，这个Job被赋予了一个ID号：job\_1425351719448\_0001，而且得知输入文件有两个（Total input paths to process : 2），同时还可以了解map的输入输出记录（record数及字节数），以及reduce输入输出记录。比如说，在本例中，map的task数量是2个，reduce的task数量是一个。map的输入record数是2个，输出record数是4个等信息。

1. 查看运行结果

输入命令

$ bin/hadoop dfs –ls /output/wordcount3/\*

显示结果如下：

15/03/04 22:22:07 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

Found 2 items

-rw-r--r-- 1 JimberXin supergroup 0 2015-03-03 17:18 /output/wordcount3/\_SUCCESS

-rw-r--r-- 1 JimberXin supergroup 41 2015-03-03 17:18 /output/wordcount3/part-r-00000

可知生成了两个文件，我们的结果存在part-r-00000中，显示其内容

输入命令：

$ bin/hadoop dfs –cat /output/wordcount3/\*

显示结果如下

15/03/03 17:19:12 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

Bye 1

Goodbye 1

Hadoop 2

Hello 2

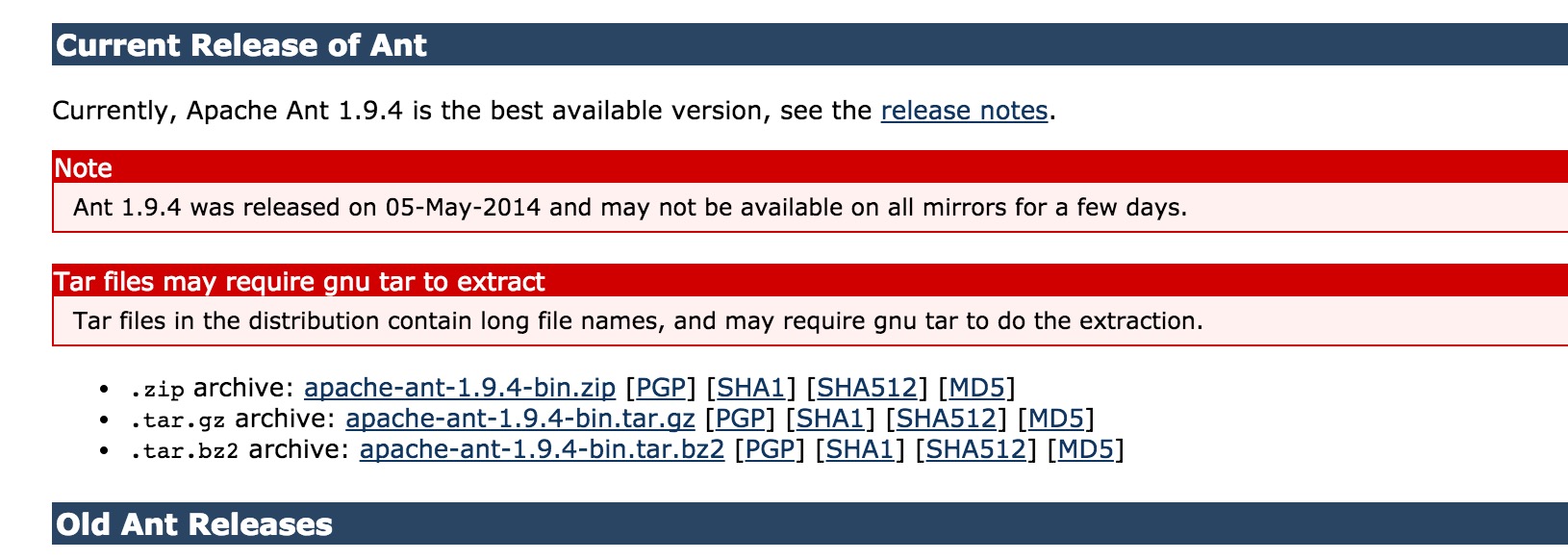
World 2

**四、eclipse for hadoop 配置**

**1、安装 ant**

官网下载最新版本： apache-ant-1.9.4

<http://ant.apache.org/bindownload.cgi>



选择 apache-ant-1.9.4.zip，解压到自己指定的某个文件夹下，例如我的是：

/Users/JimberXin/apache-ant-1.9.4

接着，在根目录下，找到 .bash\_profile（如没有自己创建一个）添加ant的路径

cd /

vi .bash\_profile

在打开的文件里，在末尾最后添加如下命令：

export ANT\_HOME=/Users/JimberXin/apache-ant-1.9.4

export PATH=$PATH:$ANT\_HOME/bin

按ecs退出， ZZ保存后退出

source .bash\_profile

现在，可以查看版本信息了

ant -v

Apache Ant(TM) version 1.9.4 compiled on April 29 2014

Trying the default build file: build.xml

Buildfile: build.xml does not exist!

Build failed

如果还是发现ant没找到，输入如下命令，确定下载的ant没问题

/Users/JimberXin/apache-ant-1.9.4/bin/ant –version

Apache Ant(TM) version 1.9.4 compiled on April 29 2014

1. **编译hadoop-eclipse 插件**

由于自己的hadoop版本是2.6.0, eclipse版本是4.4.0，比网上很多参考文章的版本都要高，所以在eclipse配置hadoop时自己编译了hadoop

首先下载源代码

git clone https://github.com/winghc/hadoop2x-eclipse-plugin.git

接着编译源码

首先进入源码位置

cd source/contrib/eclipse-plugin

查看下路径

/Users/JimberXin/hadoop-2.6.0/hadoop2x-eclipse-plugin/src/contrib/eclipse-plugin

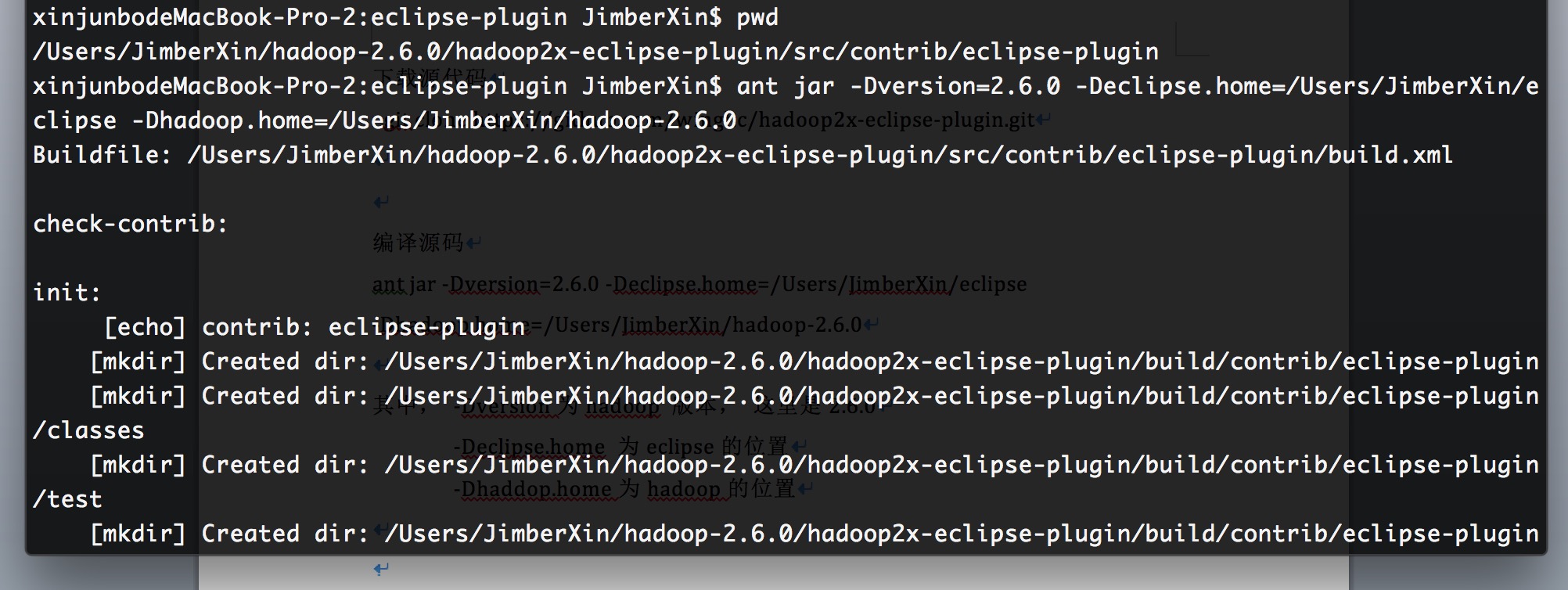
编译源码：

ant jar -Dversion=2.6.0 -Declipse.home=/Users/JimberXin/eclipse -Dhadoop.home=/Users/JimberXin/hadoop-2.6.0

其中， -Dversion为hadoop 版本， 这里是2.6.0

-Declipse.home 为eclipse的位置

-Dhaddop.home为hadoop的位置



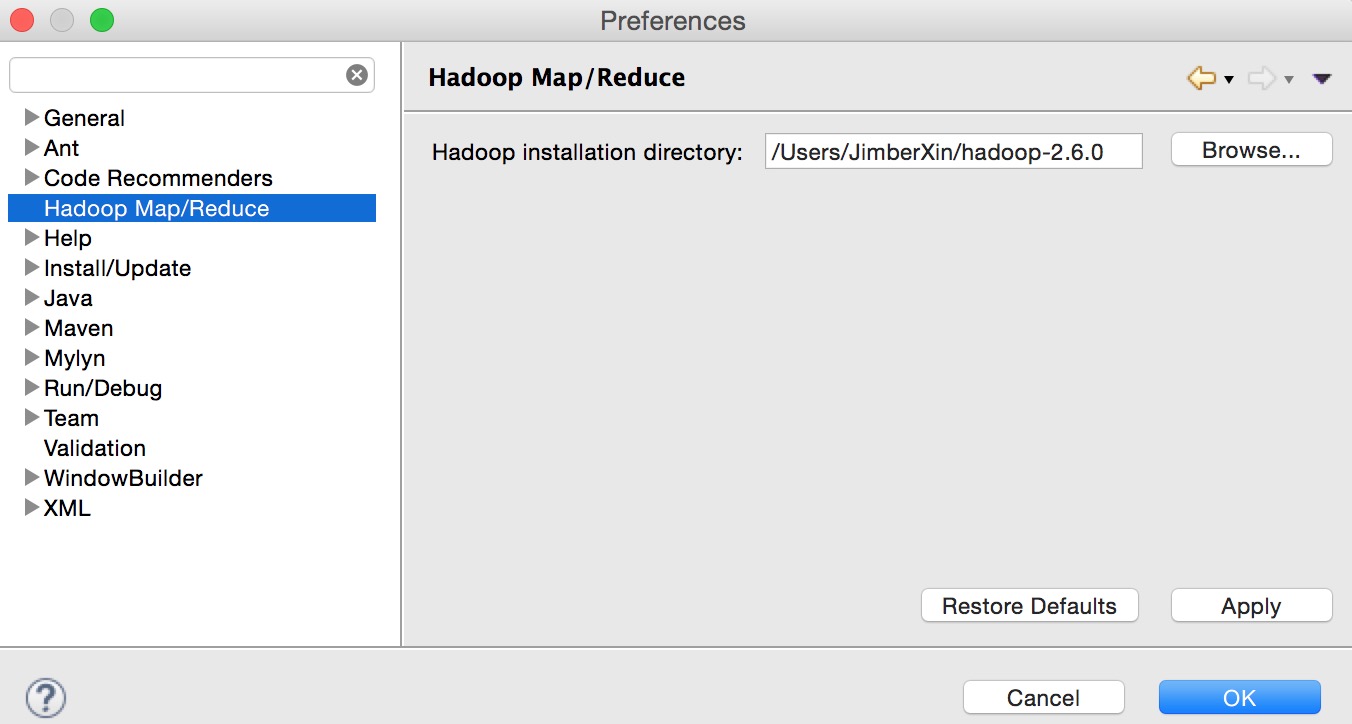
hadoop2.6.0开始编译

编译过程依赖ivy，需要的时间有点久，编译完了以后，文件在/build/contrib/eclipse-plugin/hadoop-eclipse-plugin-2.6.0.jar下，具体为：

/Users/JimberXin/hadoop-2.6.0/hadoop2x-eclipse-plugin/build/contrib/eclipse-plugin/hadoop-eclipse-plugin-2.6.0.jar

**3、eclipse hadoop插件的配置**

1. 将上述编译生成的hadoop-eclipse-plugin.jar拷贝放在eclipse目录下的plugin文件夹下，重启eclipse
2. 打开eclipse后，在eclipse的偏好设置里（mac系统在此处，如果是windows或者linux系统，在window->preference里）选择Map/Reduce的安装位置，即是hadoop的安装位置 /Users/JimberXin/hadoop-2.6.0



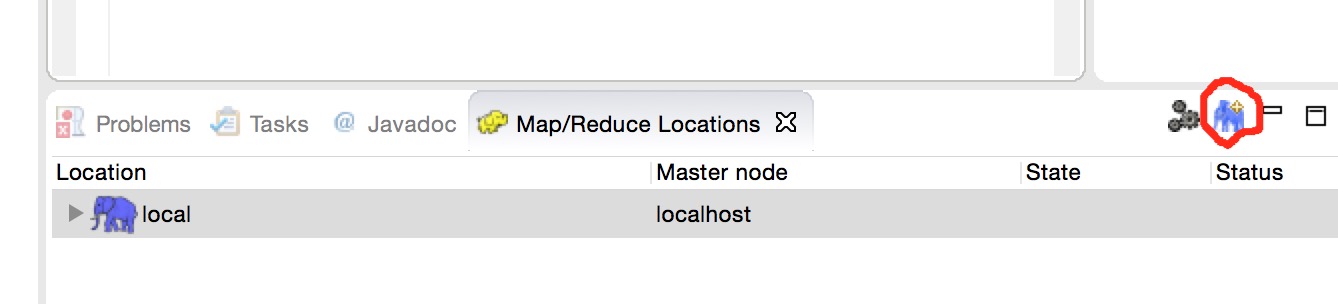
1. 配置Map/Reduce视图

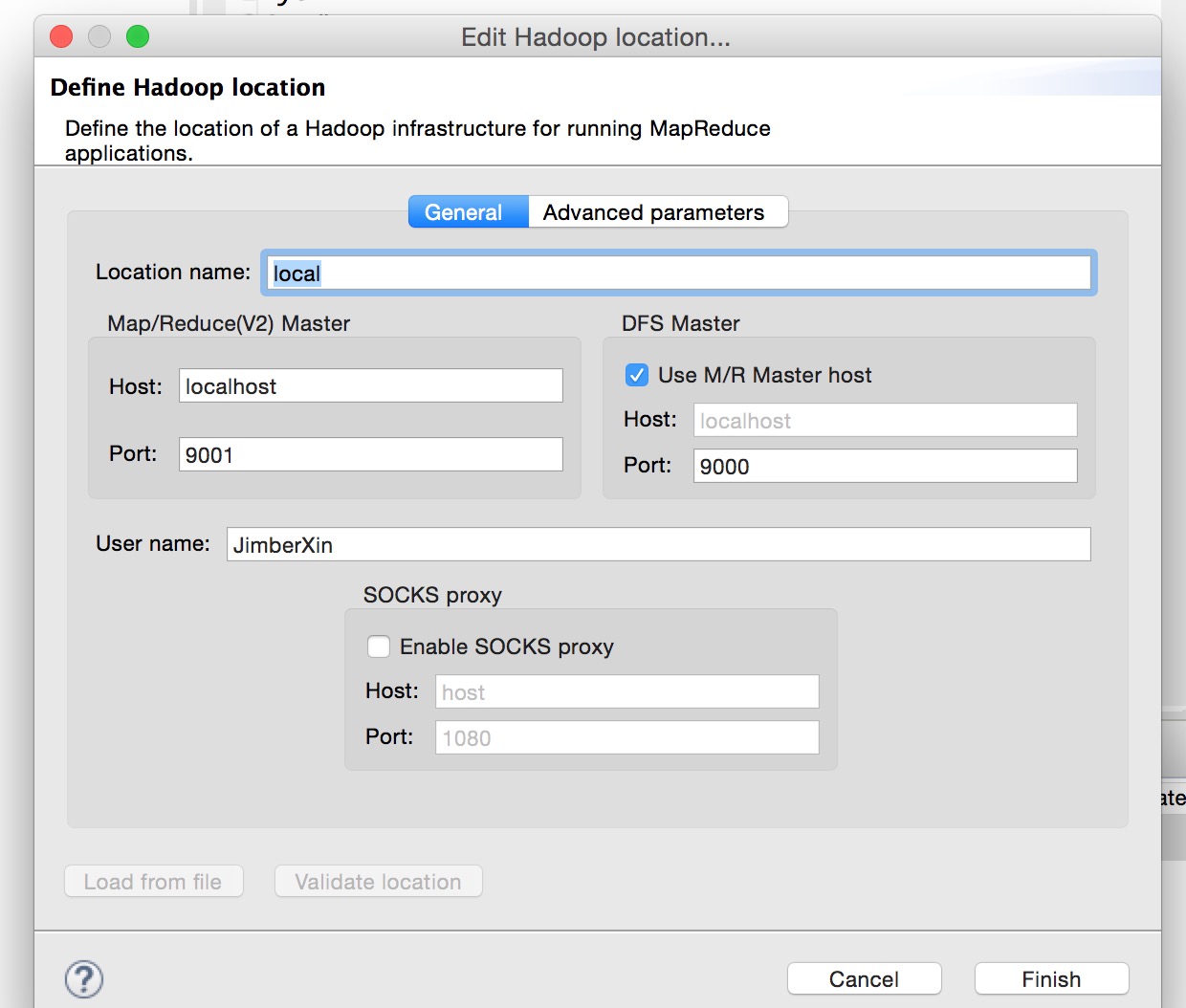
window ->Open Perspective -> other->Map/Reduce -> 点击“OK”

windows → show view → other->Map/Reduce Locations-> 点击“OK”

1. 配置 Map/Reduce Locations选项

在“Map/Reduce Locations” Tab页 点击有房图标<大象+>或者在空白的地方右键，选择“New Hadoop location…”，弹出对话框“New hadoop location…”





参数说明如下：

1. Location name: 任意
2. map/reduce master：与mapred-site.xml里mapred.job.tracker设置一致
3. DFS master：与core-site.xml里fs.default.name设置一致。
4. User name: 服务器上运行hadoop服务的用户名，这里是JimberXin