

Java-Success.com

Prepare to fast-track, choose & go places with 800+ Java & Big Data Q&As with lots of code & diagrams.

[Home](#) [Why? ▾](#) [300+ Java FAQs ▾](#) [300+ Big Data FAQs ▾](#) [Courses ▾](#)

[👤 Membership ▾](#) [Your Career ▾](#)

[Home](#) › [bigdata-success.com](#) › [Tutorials - Big Data](#) › [TUT - Starting Spark & Scala](#) ›

08. Setting up & getting started with Hadoop on Mac

08. Setting up & getting started with Hadoop on Mac

 Posted on [November 13, 2018](#)

This tutorial outlines the basic steps to get started with Hadoop on Mac OS. This is a single-node cluster with YARN as the resource manager.

1. Install Xcode

Xcode can be installed via [Apple appstore](#). Xcode is Apple's Integrated Development Environment (IDE). Xcode is a large suite of software development tools and libraries from Apple.

2. Install the Apple command line tools

300+ Java Interview FAQs

300+ Java FAQs



16+ Java Key Areas Q&As



150+ Java Architect FAQs



80+ Java Code Quality Q&As



150+ Java Coding Q&As



300+ Big Data Interview FAQs

300+ Big Data FAQs



Tutorials - Big Data



TUT -  Starting Big Data

TUT - Starting Spark & Scala

Once Xcode is installed, install the command line tools via “Xcode menu” -> “preferences” -> “command lines tools”, and click the install button. This may take a while to install. Once installed you can verify with a **Terminal** window

```
1
2 $ xcode-select -h
3
```

The Xcode Command Line Tools are part of XCode. The Xcode Command Line Tools include a GCC compiler, and many common Unix-based tools require the GCC compiler

3. Install homebrew

Homebrew is a package manager for OS. On a Terminal window type

```
1
2 $ ruby -e "$(curl -fsSL https://raw.githubusercontent.com/mxcl/homebrew/master/brew-install.rb)"
3
```

Verify if brew is installed properly by typing the following on a Terminal window:

```
1
2 $ brew doctor
3
```

4. Install Java

Hadoop is an open source software built on Java.

```
1
```

TUT - Starting with Python

TUT - Kafka

TUT - Pig

TUT - Apache Storm

TUT - Spark Scala on Zeppelin

TUT - Cloudera

TUT - Cloudera on Docker

TUT - File Formats

TUT - Spark on Docker

TUT - Flume

TUT - Hadoop (HDFS)

TUT - HBase (NoSQL)

TUT - Hive (SQL)

TUT - Hadoop & Spark

TUT - MapReduce

TUT - Spark and Scala

TUT - Spark & Java

TUT - PySpark on Databricks

TUT - Zookeeper

800+ Java Interview Q&As

300+ Core Java Q&As



300+ Enterprise Java Q&As



150+ Java Frameworks Q&As



120+ Companion Tech Q&As



Tutorials - Enterprise Java



```
2 $ brew cask install java
3
```

Also, you will need Scala for Spark if you choose to code Spark in Scala. You can also code in Java.

```
1
2 $ brew install scala
3
```

5. Install wget

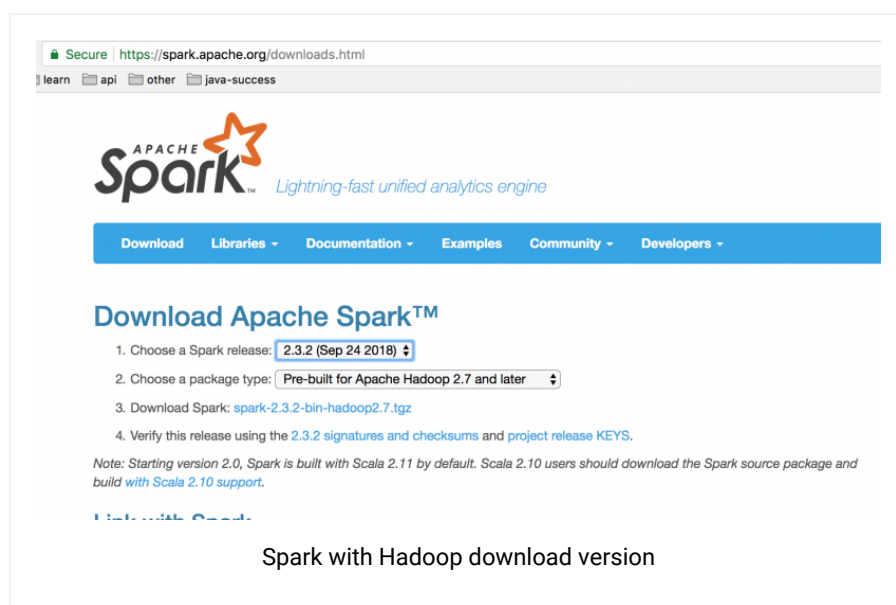
Wget is a handy tool to download files using http, https or ftp protocols from internet.

```
1
2 $ brew install wget
3
```

You can check in which folder brew has installed with

```
1
2 $ brew info wget
3
```

6. Download Hadoop with wget

The screenshot shows the Apache Spark download page. At the top, there's a navigation bar with links for 'learn', 'api', 'other', and 'java-success'. Below this is the Apache Spark logo and the tagline 'Lightning-fast unified analytics engine'. A blue navigation bar contains links for 'Download', 'Libraries', 'Documentation', 'Examples', 'Community', and 'Developers'. The main heading is 'Download Apache Spark™'. Below this, there are four steps: 1. Choose a Spark release: '2.3.2 (Sep 24 2018)'. 2. Choose a package type: 'Pre-built for Apache Hadoop 2.7 and later'. 3. Download Spark: 'spark-2.3.2-bin-hadoop2.7.tgz'. 4. Verify this release using the '2.3.2 signatures and checksums' and 'project release KEYS'. A note at the bottom states: 'Note: Starting version 2.0, Spark is built with Scala 2.11 by default. Scala 2.10 users should download the Spark source package and build with Scala 2.10 support.' The page is titled 'Spark with Hadoop download version'.

As we were using Spark 2.3.2 in the earlier tutorials, let's download hadoop 2.7.7 from <http://mirror.nohup.it/apache/hadoop/common>.

```
1
2 $ cd /usr/local
3 $ sudo wget http://mirror.nohup.it/apache/hadoop/common
4 $ sudo tar xvzf hadoop-2.7.7.tar.gz
5 $ sudo rm -f hadoop-2.7.7.tar.gz
6
```

Hadoop will be installed in “/usr/local/hadoop-2.7.7”, and this path will be reflected on all the steps outlined below.

Note: Alternatively, you can also use “brew install hadoop”. The Hadoop will be installed in the path “/usr/local/Cellar/hadoop”.

7. Make sure that the folders have the right permissions

```
1
2 $ sudo chown -R <user>:admin /usr/local/hadoop-2.7.7
3
```

8. Configure environment variables for Hadoop

Open ~/.bashrc or ~/.bash_profile file on your home directory and add following environment variables.

```
1 ## java variables
2 export JAVA_HOME=$(/usr/bin/java)
3
4 ## hadoop variables
5 export HADOOP_HOME=/usr/local/hadoop-2.7.7
6 export HADOOP_MAPRED_HOME=$HADOOP_HOME
7 export HADOOP_COMMON_HOME=$HADOOP_HOME
```

```
8 export HADOOP_HDFS_HOME=$HADOOP_HOME
9 export YARN_HOME=$HADOOP_HOME
10 export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
11 export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"
12 export PATH=$HADOOP_HOME/bin:$PATH
13 export PATH=$HADOOP_HOME/sbin:$PATH
14
```

Reload the configuration into memory so all new environment variables are effective

```
1 source ~/.bash_profile
2
```

Test it with:

```
1 $ echo $HADOOP_HOME
```

Output:

```
1 /usr/local/hadoop-2.7.7
```

9. Check hadoop-env.sh

```
1 $ cd $HADOOP_HOME/etc/hadoop
2 $ vi hadoop-env.sh
3
```

and set JAVA_HOME as shown below:

```
1
2 # The java implementation to use.
3 export JAVA_HOME=$(/usr/libexec/java_home)
4
```

“~ /usr/libexec/java_home” is an OS X utility that allows you to easily generate a path to a JDK

10. Hadoop config core-site.xml

```
1 $ cd $HADOOP_HOME/etc/hadoop
2 $ sudo vi core-site.xml
3
```

The “core-site.xml” should look like

```
1 <configuration>
2
3   <property>
4       <name>hadoop.tmp.dir</name>
5       <value>/usr/local/hadoop-2.7.7/hdfs/tmp</value>
6       <description>base for other temp directorie
7   </property>
8
9   <property>
10      <name>fs.default.name</name>
11      <value>hdfs://localhost:9000</value>
12  </property>
13 </configuration>
14
```

11. Hadoop config hdfs-site.xml

```
1 $ cd $HADOOP_HOME/etc/hadoop
2 $ sudo vi hdfs-site.xml
3
```

The “hdfs-site.xml” set the replication factor to 1, which means don’t keep multiple copies of data blocks in our single node cluster. This is not recommended for production environments, where it must be set to least 3 to prevent data loss in case of node failures.

```
1 <configuration>
2   <property>
3       <name>dfs.replication</name>
4       <value>1</value>
5   </property>
```

```
6 </configuration>
7
```

12. Hadoop resource manager config yarn-site.xml

```
1 $ cd $HADOOP_HOME/etc/hadoop
2 $ sudo vi yarn-site.xml
3
1 <configuration>
2
3   <property>
4     <name>yarn.nodemanager.aux-services</name>
5     <value>mapreduce_shuffle</value>
6   </property>
7
8   <property>
9     <name>yarn.nodemanager.aux-services.mapred
10    <value>org.apache.hadoop.mapred.ShuffleHand
11  </property>
12
13 </configuration>
14
```

13. Hadoop mapred-site.xml

```
1 $ cd $HADOOP_HOME/etc/hadoop
2 $ sudo vi mapred-site.xml
3
```

YARN will take care of resource management on our single node cluster.

```
1 <configuration>
2
3   <property>
4     <name>mapred.job.tracker</name>
5     <value>yarn</value>
6   </property>
7
8   <property>
9     <name>mapreduce.framework.name</name>
10    <value>yarn</value>
11  </property>
```

```
12 |  
13 | </configuration>  
14 |
```

14. Format the name node

Before starting the daemons, you must format the NameNode so it starts fresh.

```
1 |  
2 | $ hdfs namenode -format  
3 |
```

15. SSH to localhost

In order to connect to Hadoop cluster, you need to setup a secure connection. Open **System Preferences** -> **Sharing** and **enable Remote Login** option.

If you have not already done it, you need to generate public/private key pairs and copy the public key to the `authorized_keys` as shown below. The keys will be generated in the “`~/.ssh`” folder. “`~`” means your Unix home directory.

```
1 |  
2 | $ ssh-keygen -t rsa  
3 | $ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys  
4 |
```

Now see if you can ssh to the localhost.

```
1 |  
2 | $ ssh localhost  
3 |
```

If you can ssh, you are good to move to the next step.

16. Start the namenode, datanode and the secondary namenode

```
1  
2 $ start-dfs.sh  
3
```

17. jps command to list the Java processes

```
1  
2 $ jps -lm  
3
```

Output:

```
1  
2 3585 org.apache.hadoop.hdfs.server.namenode.NameNode  
3 3683 org.apache.hadoop.hdfs.server.datanode.DataNode  
4 4377 sun.tools.jps.Jps -lm  
5 3805 org.apache.hadoop.hdfs.server.namenode.SecondaryNameNode  
6
```

18. start-yarn.sh to start NodeManager & ResourceManager

```
1  
2 $ start-yarn.sh  
3
```

```
1  
2 $ jps -lm  
3
```

Output:

```
1  
2 3585 org.apache.hadoop.hdfs.server.namenode.NameNode  
3 3938 org.apache.hadoop.yarn.server.resourcemanager.ResourceManager
```

```
4 3683 org.apache.hadoop.hdfs.server.datanode.DataNode
5 4377 sun.tools.jps.Jps -lm
6 4041 org.apache.hadoop.yarn.server.nodemanager.NodeManager
7 3805 org.apache.hadoop.hdfs.server.namenode.SecondaryNameNode
8
```

You can also install telnet with

```
1 brew install telnet
2
```

and test if you can connect to the name node with:

```
1 telnet localhost 9000
```

The host & port are configured with the value
“hdfs://localhost:9000” in the “[core-site.xml](#)” file.

We are now ready to perform Hadoop command-line commands with either “hadoop fs -xxxx” or “[hdfs dfs -xxxx](#)” commands. Refer to “[Hadoop Commands Guide](#)” for the full list of possible commands and options.

19. Create a new folder on HDFS

```
1
2 $ hdfs dfs -mkdir -p /user/<user>
3 $ hdfs dfs -ls /
4
```

20. Add a file from local file system to HDFS

```
1
2 $ touch test.csv
3 $ vi test.csv
4
```

The “test.csv” file data looks like

```
1 Maths, 85
2 English, 94
3 Science, 75

1
2 $ hdfs dfs -mkdir -p /data/marks
3 $ hdfs dfs -put ./test.csv /data/marks
4
```

21. The Web UIs

Hadoop Web interface <http://localhost:50070/>

localhost:50070/dfshealth.html#tab-overview

learnapiotherjava-success

HadoopOverviewDatanodesDataNode Volume FailuresSnapshotStartup ProgressUtilities

Overview'localhost:9000' (active)

Started:Tue Nov 13 10:21:01 AEDT 2018

Version:2.7.7, rc1aa84bd270d79c3d7a7d58d202d0c3ee1ed3ac

Compiled:2018-07-18T22:47Z by stevel from branch-2.7.7

Cluster ID:CID-c548da05-11f7-417a-e43a-611a55aee130

Block Pool ID:BP-1071144237-192.168.0.18-154236460809

Summary

Security is off.
SafeMode is off.
4 files and directories, 1 blocks = 5 total filesystem object(s).
Heap Memory used 98.59 MB of 221.5 MB Heap Memory. Max Heap Memory is 889 MB.
Non Heap Memory used 43.04 MB of 43.88 MB Committed Non Heap Memory. Max Non Heap Memory is -1 B.

Configured Capacity:

DFS Used:

Non DFS Used:

DFS Remaining:

Block Pool Used:

DataNodes usage% (Min/Median/Max/StdDev):

Live Nodes

Dead Nodes

Decommissioning Nodes

Total DataNode Volume Failures

Number of Under-Replicated Blocks

Number of Blocks Pending Deletion

Hadoop Web Interface

JobTracker <http://localhost:8088>

localhost:8088/cluster

Appslearnapiotherjava-success

hadoop

Cluster Metrics

Scheduler Metrics

Tools

All Applications

Cluster Metrics

Scheduler Metrics

Tools

Cluster Metrics

Scheduler Metrics

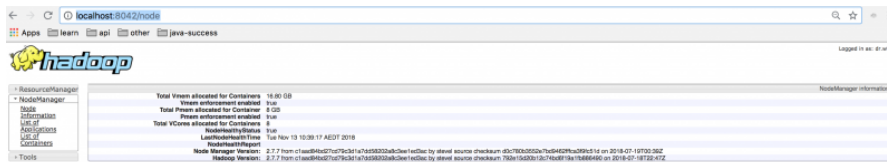
Tools

JobTracker Web UI

Single Node <http://localhost:8042/node>

<https://www.java-success.com/08-setting-getting-started-hadoop-mac/>

11/13



22. stop-xxx.sh to stop the servers

```
1  
2 $ stop-yarn.sh  
3 $ stop-dfs.sh  
4
```

23. start-all.sh & stop-all.sh

You can use the following commands to start or stop both hdfs & yarn services

```
1  
2 $ stop-all.sh  
3 $ start-all.sh  
4
```

In the previous tutorials I have covered how to install Scala, Scala IDE for Eclipse, Sbt (i.e. Scala Build Tool), and Spark.

In the next tutorial will use

Spark to read a file from the HDFS

(i.e. Hadoop Distributed File System). [Getting started with Spark & Hadoop – client & cluster modes.](#)

Hive to process data in HDFS

[Setting up & getting started with Hive](#)

Learn more about Hadoop Q&As style

Hadoop overview Interview FAQs

HDFS Interview FAQs

Hadoop MapReduce Interview FAQs

◀ 07: Getting started with Spark-submit

09. Getting started with Spark & Hadoop – client mode on local & cluster
mode on yarn ▶

Disclaimer

The contents in this Java-Success are copyrighted and from EmpoweringTech pty ltd. The EmpoweringTech pty ltd has the right to correct or enhance the current content without any prior notice. These are general advice only, and one needs to take his/her own circumstances into consideration. The EmpoweringTech pty ltd will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. Links to external sites do not imply endorsement of the linked-to sites. [Privacy Policy](#).