

Install Hadoop single node cluster and run simple applications like word count.

Aim:

To Install Hadoop single node cluster and run simple Applications like word count.

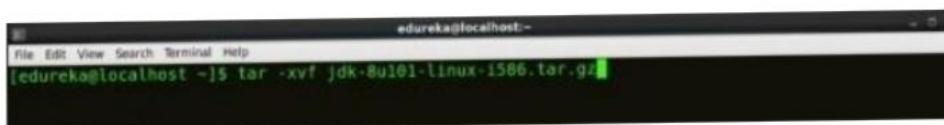
Steps:

Install Hadoop

Step1: [Click here](#) to download the Java 8 Package. Save this file in your home directory.

Step2: Extract the JavaTarFile.

Command: tar -xvf jdk-8u101-linux-i586.tar.gz

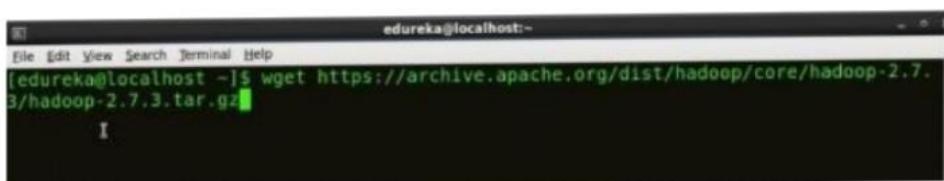


```
edureka@localhost:~$ tar -xvf jdk-8u101-linux-i586.tar.gz
```

Fig:Hadoop Installation—Extracting Java Files

Step3: Download the Hadoop 2.7.3 Package.

Command: wget <https://archive.apache.org/dist/hadoop/core/hadoop-2.7.3/hadoop-2.7.3.tar.gz>



```
edureka@localhost:~$ wget https://archive.apache.org/dist/hadoop/core/hadoop-2.7.3/hadoop-2.7.3.tar.gz
```

Fig: Hadoop Installation— Downloading Hadoop

Step4: Extract the Hadoop tar File.

Command: tar -xvf hadoop-2.7.3.tar.gz



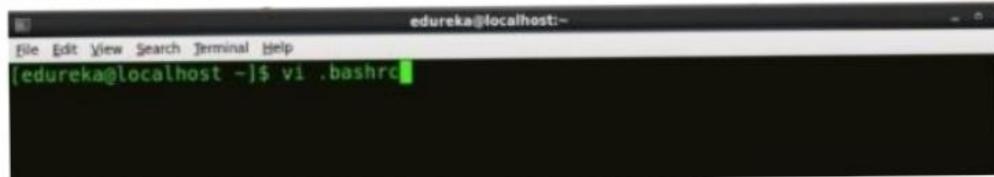
```
edureka@localhost:~$ tar -xvf hadoop-2.7.3.tar.gz
```

Fig: Hadoop Installation – Extracting Hadoop Files Step 5:

Add the Hadoop and Java paths in the bash file (.bashrc). Open **.bashrc** file.

Now, add Hadoop and Java Path as shown below.

Command: vi.bashrc



```
[edureka@localhost ~]$ vi .bashrc
```

```
# User specific aliases and functions

export HADOOP_HOME=$HOME/hadoop-2.7.3
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export HADOOP_MAPRED_HOME=$HADOOP_HOME/hadoop-2.7.3
export HADOOP_COMMON_HOME=$HADOOP_HOME/hadoop-2.7.3
export HADOOP_HDFS_HOME=$HADOOP_HOME/hadoop-2.7.3
export YARN_HOME=$HADOOP_HOME/hadoop-2.7.3
export PATH=$PATH:$HADOOP_HOME/hadoop-2.7.3/bin

# Set JAVA_HOME

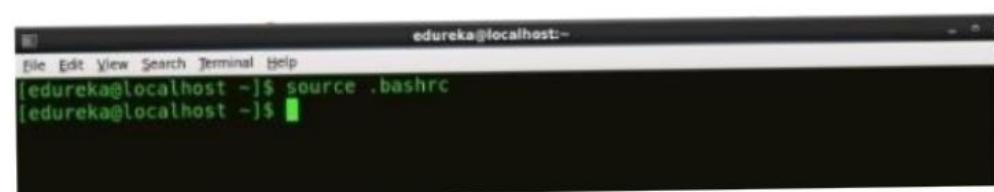
export JAVA_HOME=/home/edureka/jdk1.8.0_101
export PATH=/home/edureka/jdk1.8.0_101/bin:$PATH
```

Fig:Hadoop Installation–Setting Environment Variable

Then, save the bash file and close it.

For apply in gall the sechan gesto the current Terminal, execute the source command.

Command: source.bashrc



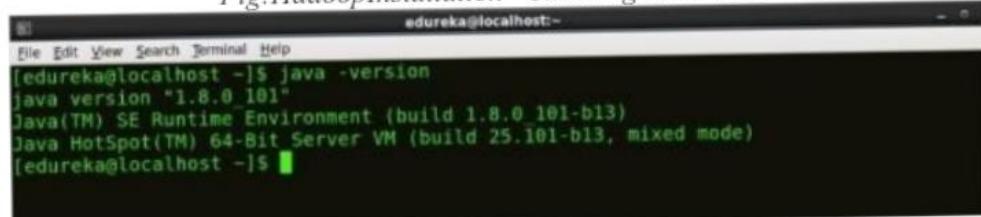
```
[edureka@localhost ~]$ source .bashrc
[edureka@localhost ~]$
```

Fig:Hadoop Installation–Refreshing environment variables

To make sure that Java and Hadoop have been properly installed on your system and can be accessed through the Terminal, execute the java -version and hadoop version commands.

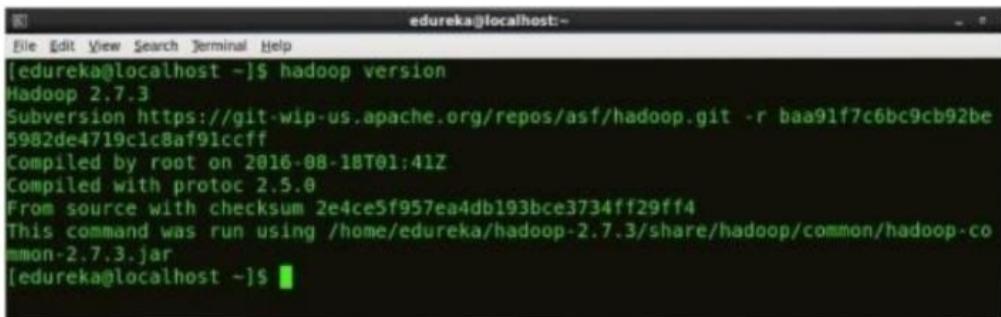
Command: java-version

Fig:Hadoop Installation– Checking Java Version



```
[edureka@localhost ~]$ java -version
java version "1.8.0_101"
Java(TM) SE Runtime Environment (build 1.8.0_101-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.101-b13, mixed mode)
[edureka@localhost ~]$
```

Command: hadoop version



```
[edureka@localhost ~]$ hadoop version
Hadoop 2.7.3
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r baa91f7c6bc9cb92be
5982de4719c1c8af91ccff
Compiled by root on 2016-08-18T01:41Z
Compiled with protoc 2.5.0
From source with checksum 2e4ce5f957ea4db193bce3734ff29ff4
This command was run using /home/edureka/hadoop-2.7.3/share/hadoop/common/hadoop-common-2.7.3.jar
[edureka@localhost ~]$
```

Fig:Hadoop Installation– Checking Hadoop Version

Step6:Edit the **Hadoop Configuration files**.

Command: cdhadoop-2.7.3/etc/hadoop/



Command:ls

All the Hadoop configuration files are located in **hadoop-2.7.3/etc/hadoop** directory as you can see in the snapshot below:



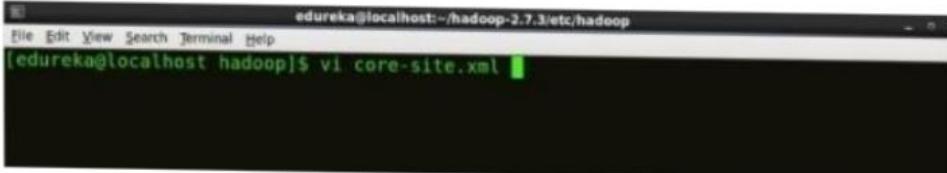
```
[edureka@localhost ~]$ cd hadoop-2.7.3/etc/hadoop/
[edureka@localhost hadoop]$ ls
capacity-scheduler.xml      httpfs-env.sh          mapred-env.sh
configuration.xml           httpfs-log4j.properties   mapred-queues.xml.template
container-executor.cfg       httpfs-signature.secret  mapred-site.xml.template
core-site.xml                httpfs-site.xml        slaves
hadoop-env.cmd              kms-acls.xml          ssl-client.xml.example
hadoop-env.sh               kms-env.sh            ssl-server.xml.example
hadoop-metrics2.properties  kms-log4j.properties   yarn-env.cmd
hadoop-metrics.properties   kms-site.xml          yarn-env.sh
hadoop-policy.xml           log4j.properties     yarn-site.xml
hdfs-site.xml                mapred-env.cmd
```

Fig:Hadoop Installation– Hadoop Configuration Files

Step 7: Open *core-site.xml* and edit the property mentioned below inside configuration tag:

core-site.xml informs Hadoop daemon where Name Node runs in the cluster. It contains configuration settings of Hadoop core such as I/O settings that are common to HDFS & Map Reduce.

Command: vi*core-site.xml*



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

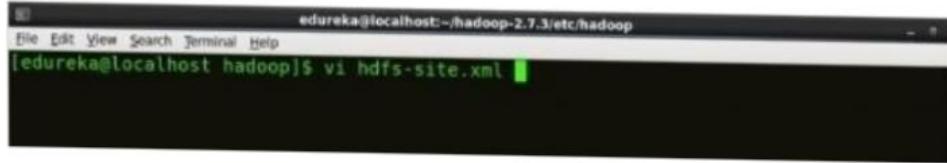
Fig:Hadoop Installation– Configuring core-site.xml

```
1           <?xmlversion="1.0"encoding="UTF-8"?>
2             <?xmlstylesheet type="text/xsl" href="configuration.xsl"?>
3               <configuration>
4                 <property>
5                   <name>fs.default.name</name>
6                     <value>hdfs://localhost:9000</value>
7                   </property>
8               </configuration>
```

Step 8: Edit *hdfs-site.xml* and edit the property mentioned below inside configuration tag:

hdfs-site.xml contains configuration setting so fHDFS daemons(i.e.NameNode,DataNode, Secondary Name Node). It also includes the replication factor and block size of HDFS.

Command: vi*hdfs-site.xml*



```
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>dfs.permission</name>
<value>false</value>
</property>
```

Fig:Hadoop Installation– Configuring hdfs-site.xml

```
1      <?xmlversion="1.0"encoding="UTF-8"?>
2  <?xmlstylesheet type="text/xsl" href="configuration.xsl"?>
3      <configuration>
4          <property>
5              <name>dfs.replication</name>
6              <value>1</value>
7          </property>
8          <property>
9              <name>dfs.permission</name>
10             <value>false</value>
11         </property>
12     </configuration>
```

Step9:Edit the *mapred-site.xml* file and edit the property mentioned below

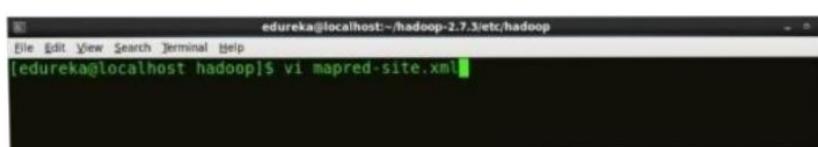
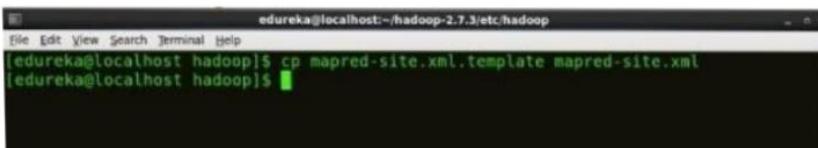
Inside configuration tag:

mapred-site.xml contains configuration settings of Map Reduce application like number of JVM that can run in parallel, the size of the mapper and the reducer process, CPU cores available for a process, etc.

In some cases, *mapred-site.xml* file is not available. So, we have to create the map red- site.xml file using *mapred-site.xml* template.

Command: cp*mapred-site.xml.template* *mapred-site.xml*

Command:vim*mapred-site.xml*.



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

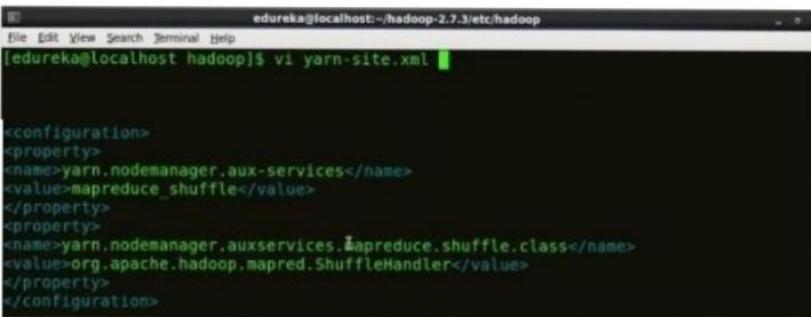
Fig:Hadoop Installation– Configuring mapred-site.xml

```
1           <?xmlversion="1.0"encoding="UTF-8"?>
2             <xmlestsheettype="text/xsl" href="configuration.xsl"?>
3               <configuration>
4                 <property>
5                   <name>mapreduce.framework.name</name>
6                     <value>yarn</value>
7                   </property>
8               </configuration>
```

Step 10: Edit *yarn-site.xml* and edit the property mentioned below inside configuration tag:

yarn-site.xml contains configuration settings of Resource Manager and Node Manager like application memory management size, the operation needed on program & algorithm, etc.

Command: vi *yarn-site.xml*



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

Fig:HadoopInstallation– Configuring *yarn-site.xml*

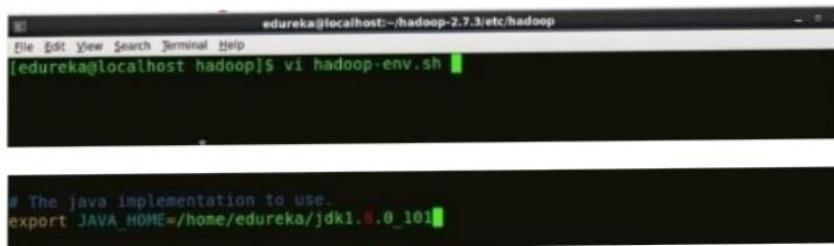
Step11: Edit *hadoop-env.sh* and add the JavaPath as mentioned below:

```
1
2           <?xmlversion="1.0">
3             <configuration>
4               <property>
5                 <name>yarn.nodemanager.aux-services</name>
6                   <value>mapreduce_shuffle</value>
7                 </property>
8               <name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</
9                 name>
10                 <value>org.apache.hadoop.mapred.ShuffleHandler</value>
11               </property>
12
13
```

hadoop-env.sh contains the environment variables that are used in the script to run Hadoop

like Java home path, etc.

Command: vihadoop-env.sh



```
# The java implementation to use.  
export JAVA_HOME=/home/edureka/jdk1.8.0_101
```

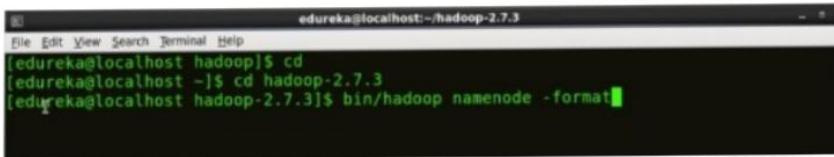
Fig:HadoopInstallation–Configuringhadoop-env.sh**Step 12:**

Go to Hadoop home directory and format the Name Node.

Command: cd

Command: cd hadoop-2.7.3

Command: bin/hadoop namenode -format



```
[edureka@localhost hadoop]$ cd  
[edureka@localhost ~]$ cd hadoop-2.7.3  
[edureka@localhost hadoop-2.7.3]$ bin/hadoop namenode -format
```

Fig:HadoopInstallation– Formatting NameNode

This formats the HDFS via Name Node. This command is only executed for the first time. Formatting the file system means initializing the directory specified by the dfs.name.dir variable.

Never format, up and running Hadoop file system. You will lose all your data stored in the HDFS.

Step13:Once the Name Node is formatted, goto hadoop-2.7.3/s bin directory and start all the daemons.

Command: cdhadoop-2.7.3/sbin

Either you can start all daemons with a single command or do it individually.

Command: ./start-all.sh

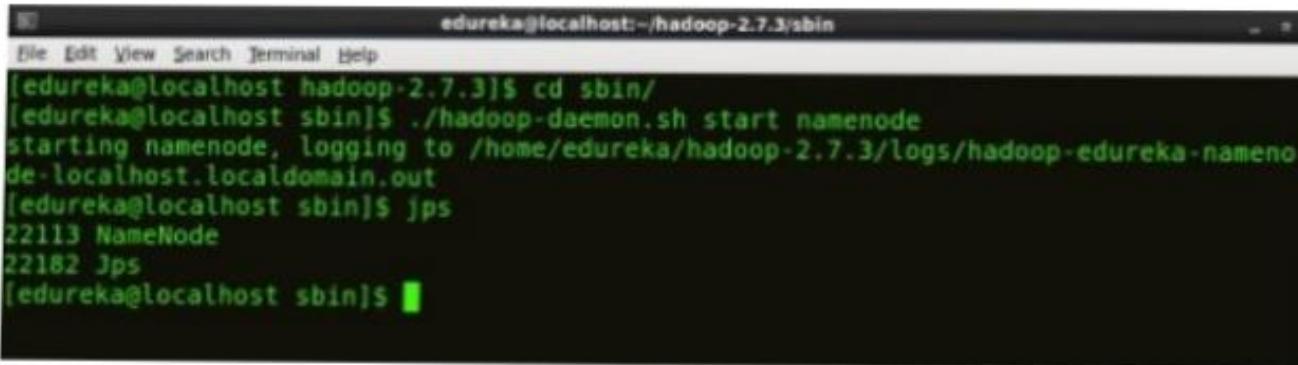
The above command is a combination of *start-dfs.sh*, *start-yarn.sh* & *mr-jobhistory-daemon.sh*

Or you can run all the services individually as below:

Start Name Node:

The Name Node is the center piece of an HDFS file system. It keeps the directory tree of all files stored in the HDFS and tracks all the file stored across the cluster.

Command:./hadoop-daemon.sh start name node



A screenshot of a terminal window titled "edureka@localhost:~/hadoop-2.7.3/sbin". The window shows the following command sequence:

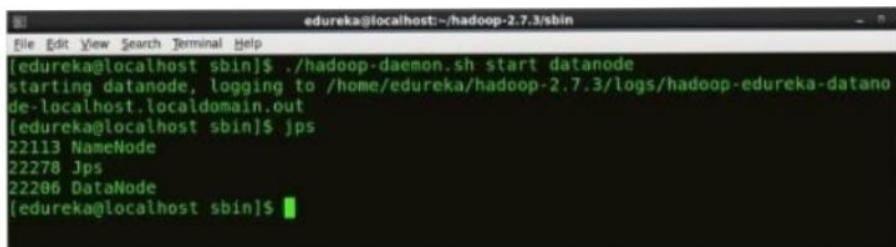
```
[edureka@localhost hadoop-2.7.3]$ cd sbin/  
[edureka@localhost sbin]$ ./hadoop-daemon.sh start namenode  
starting namenode, logging to /home/edureka/hadoop-2.7.3/logs/hadoop-edureka-namenode-localhost.localdomain.out  
[edureka@localhost sbin]$ jps  
22113 NameNode  
22182 Jps  
[edureka@localhost sbin]$ █
```

Fig: Hadoop Installation–Starting Name Node

Start Data Node:

On startup, a Data Node connects to the Name node and it responds to the requests from the Name node for different operations.

Command: ./hadoop-daemon.sh start data node



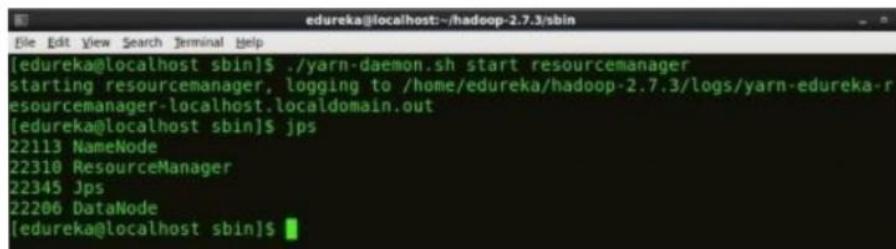
```
edureka@localhost sbin$ ./hadoop-daemon.sh start datanode
starting datanode, logging to /home/edureka/hadoop-2.7.3/logs/hadoop-edureka-datanode-localhost.localdomain.out
[edureka@localhost sbin]$ jps
22113 NameNode
22278 Jps
22206 DataNode
[edureka@localhost sbin]$
```

Fig: Hadoop Installation–StartingDataNode

Start Resource Manager:

Resource Manager is the master that arbitrates all the available cluster resources and thus helps in managing the distributed applications running on the YARN system. Its work is to manage each Node Managers and the each application's Application Master.

Command: ./yarn-daemon.sh start resourcemanager



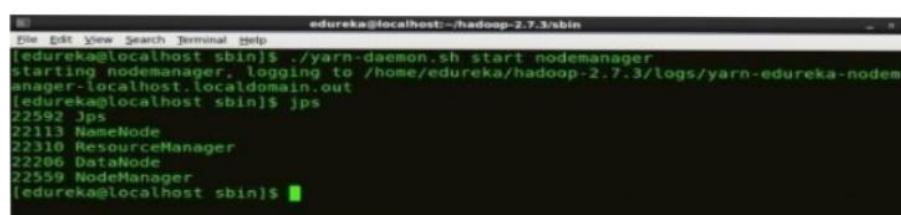
```
edureka@localhost sbin$ ./yarn-daemon.sh start resourcemanager
starting resourcemanager, logging to /home/edureka/hadoop-2.7.3/logs/yarn-edureka-resourcemanager-localhost.localdomain.out
[edureka@localhost sbin]$ jps
22113 NameNode
22310 ResourceManager
22345 Jps
22206 DataNode
[edureka@localhost sbin]$
```

Fig:Hadoop Installation–StartingResourceManager

Start Node Manager:

The Node Manager in each machine framework is the agent which is responsible for managing containers, monitoring their resource usage and reporting the same to the Resource Manager.

Command: ./yarn-daemon.sh start node manager



```
edureka@localhost sbin$ ./yarn-daemon.sh start nodemanager
starting nodemanager, logging to /home/edureka/hadoop-2.7.3/logs/yarn-edureka-nodemanager-localhost.localdomain.out
[edureka@localhost sbin]$ jps
22592 Jps
22113 NameNode
22310 ResourceManager
22206 DataNode
22559 NodeManager
[edureka@localhost sbin]$
```



[See BatchDetails](#)

Fig:Hadoop Installation–Starting Node Manager

Start Job History Server:

Job History Server is responsible for servicing all job history related requests from client.

Command:./mr-jobhistory-daemon.sh start history server

Step14:To check that all the Hadoop services are up and running, run the below command.

Command:jps

```
File Edit View Search Terminal Help edureka@localhost:~/hadoop-2.7.3/sbin
[edureka@localhost sbin]$ ./mr-jobhistory-daemon.sh start historyserver
starting historyserver, logging to /home/edureka/hadoop-2.7.3/logs/mapred-edureka-h
istoryserver-localhost.localdomain.out
[edureka@localhost sbin]$ jps
22113 NameNode
22310 ResourceManager
22694 JobHistoryServer
22727 Jps
22296 DataNode
22559 NodeManager
[edureka@localhost sbin]$
```

Fig:Hadoop Installation– Checking Daemons

Step15: Now open the Mozilla browser and go
To **localhost:50070/dfshealth.html** to check the NameNode interface.

The screenshot shows a Mozilla Firefox window titled "Namenode Information - Mozilla Firefox". The address bar displays the URL "http://localhost:50070/dfshealth.html#tab-overview". The main content area is titled "Overview 'localhost:9000' (active)". Below this, there is a table with the following data:

Started:	Wed Nov 02 08:32:45 CET 2016
Version:	2.7.3, rbaa91f7c6bc9cb92be5982de4719c1c8af91ccff
Compiled:	2016-08-18T01:41Z by root from branch-2.7.3
Cluster ID:	CID-617e6b4f-a7e8-45ee-abae-e59744b38d66
Block Pool ID:	BP-1874109370-127.0.0.1-1477077288629

Fig: Hadoop Installation–Starting WebUI

Congratulations, you have successfully installed a single node Hadoop cluster

Result:

Thus the Hadoop one cluster was installed and simple applications executed successfully.