**Get starting learning Big Data**

#Disclaimer : this notes is taken from internet, surfing. Hence, using this notes or programs is not advisable to use in any production Environments.

**In total, training session has 7 chapters for now! Below are titles for each chapter**

**Main topics**

1. scope of Bigdata
2. Java basis
3. Scala

**Scope of Bigdata**

1. Introduction to Big Data
2. Hadoop Architecture
3. First Week Assignment
4. HDFS commands
5. Map Reduce concepts

**1.Introduction to Big Data**

Link TBD

**2.Hadoop Architecture**

Link TBD

**Hadoop node installation**

**Pre-Requests are:**

On test or Lab Environment, we need to get ready our laptop or machine. then after,

* Install virtual platform - Example : Vmware Workstation, virtalbox

https://www.virtualbox.org/wiki/Download\_Old\_Builds\_5\_2

* Create VM and install Guest Operating systems - Example : CentOS6,7 or Ubuntu. Download vigrant from below link.

https://www.vagrantup.com/downloads.html

* Install Java
* Install Hadoop 2.7.x

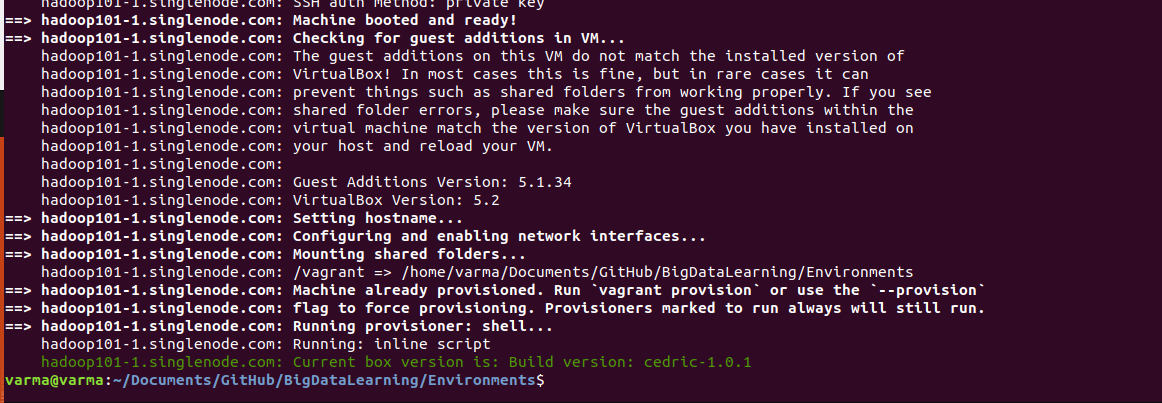
In our scenario, I am going to use vagrant on top of virtualbox to control VBs and this notes is not going to cover how to install virtual box and vagrant. It is pretty easy and follow YouTube.

**steps to standup Hadoop Single node are given below**

Once we finished installing virtualbox and vagrant, follow below steps

* Go Environments and copy VagrantFile to you local machine
* Open Terminal(In my case, I am using Mac OS, but, use cmd for Windows machine) and navigate to VagrantFile path
* run "vagrant up" command
* do "vagrant ssh"

This process will take time to download box, configure and make it online. At end, we will see screen similar to below

[](https://github.com/psanthoshkumar/BigDataLearning/blob/master/pictures/hadoopserver_up.png)

In ideal conditions, we should be able to login to ubuntu OS which has all packages what we need to continue with our training

**Note : vagrant box may change/remove/update from vagrant cloud storage. At this point of time, i have below features**

Following services are part of cluster Hadoop-2.7.2 Hive-1.2.2 on Postgresql Spark-1.6.0 Git Postgresql Anaconda2-5.0.1 Node-8.9.0 PhantomJS Monit (Manages Hadoop start/stop of hadoop services as per dependency) Redis-Server

Few nodeJs packages are added in the box and bumpped up the version.

Users: hadoop for managing cluster vagrant: Having access to Hadoop cluster

Use below Vagrant file for flawless deployment:-

Vagrantfile: https://github.com/psanthoshkumar/BigDataLearning/blob/master/Environments/Vagrantfile

**Web interfaces:**

Once VB is ready, we should be able to lauch any interfaces from below list. Here are some useful links to navigate to various UI's:

YARN resource manager: (http://localhost:8088)

HDFS: (http://localhost:50070/dfshealth.html)

Spark history server: (http://localhost:18080)

Spark context UI (if a Spark context is running): (http://localhost:4040) [Spark context server port is open from 4040 to 4044]

**Start/stop services manually from inside VB**

$ vagrant ssh

$ sudo su - hadoop

# Stop the services

$ jps | grep -v Jps | awk '{print $1}' | xargs kill -9

# Start the services

$ /bin/bash /opt/service-start-cluster.sh

**3. First Week Assignment**

1. JAVA knowledge
2. Install Hadoop Single node server

**Note: minimum JAVA knowledge is required. Hence, have to practice java and my notes for JAVA preparation at link**[**Java notes**](https://github.com/psanthoshkumar/BigDataLearning/blob/master/Docs/java/README.md)**, once you finished come back here!**

**if you follow above steps, your single node server will be ready for next chapter.**

**4. HDFS commands**

This is big chapter to cover lot of stuff.. Better, i should mention in another page for this chapter to make our lives easier. Follow this [link](https://github.com/psanthoshkumar/BigDataLearning/blob/master/Docs/hadoop/hdfs_commands/README.md) for HDFS commands

**5. Map-reduce concepts**

Map-reduce framework will work on key, value concept. There are 3 different parts or logics in each Map-reduce program

**1. Mapper**

**2. Reducer**

**3. Driver**

**1. Mapper:**

there are 3 different stages in Mapper part. They are :

Shuffle

Sort

Group

Then output goes to Reducer logic and reduce will take care of giving output

**Example: Find word count from given file**

**Take the file1.txt with below data**

Apple Apple Ball Ball Ball

Apple Ball Sam

Sam Apple

Sam Happy

In general, this data will be saved to HDFS with finite block size. Here, Mapper(the Java class) execute on each block. If data saved into 2 blocks, then 2 Mapper objects will be created to process the data.

In this example, First of all.. data will be stored into key and value formate

key ----> value

0 ----> Apple Apple Ball Ball Ball

26 ----> Apple Ball Sam

26+14=40 ----> Sam Apple

40+9=49 ----> Sam Happy

Here, each line is segmented as key and value which is going to scan by mapper code first of all.

Now Mapper class will begin processing data.. As said there are 3 stages in Mapper.

**First stage is Shuffle**

Firstly, Mapper logic will take line 1 where value 0 and value "Apple Apple Ball Ball Ball". Here value will be stored in Array and separator should be either "space" or "comma" depends on how's data stored in input file. In this case, "space" is the separator as we have spaces between words.

This logic will take a word, example "Apple" and assign value as 1 to it. This logic will be looped/executed until array reaches its max length.

Output of first line from example is:

Apple 1

Apple 1

Ball 1

Ball 1

Ball 1

similarly, All lines will be executed by mapper class then after each "word taken as key" and assign value as "1".

**Second stage is sort,**

Once shuffle is done, we will have each word(key) and values(in this case 1 is value as it is word count program). In this stage, all words will be sorted like below

<Key value>

Apple 1

Apple 1

Apple 1

Apple 1

Ball 1

Ball 1

Ball 1

Ball 1

Sam 1

Sam 1

Sam 1

Happy 1

Note: of course, Data will be sorted in alphabetical order, Ignore if you find wrong order.

**third stage is grouping,**

In this stage, above data will be grouped by Hadoop framework as below. Key should be word and value will be stored as array with values

Apple [1,1,1,1]

Ball [1,1,1,1]

Sam [1,1,1]

Happy [1]

**2. Reducer**

In this logic, array will be looped to add all the values present in array. So final output is:

Apple 4

Ball 4

Sam 3

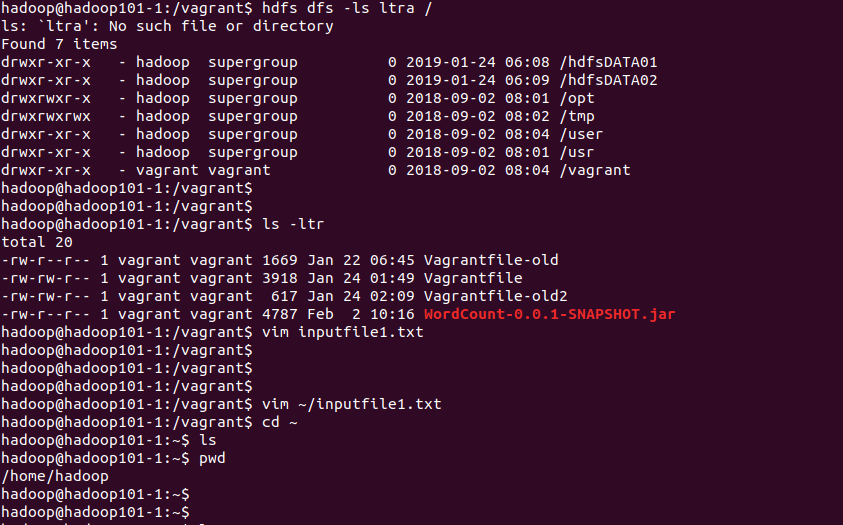
Happy 1

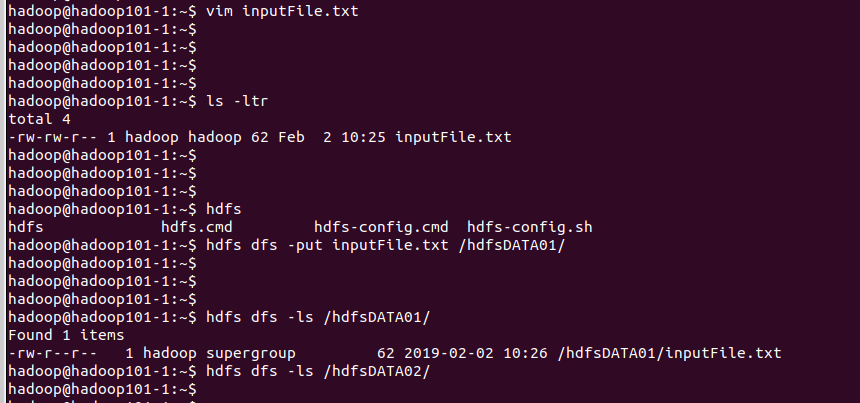
**3. Driver**

This is the main program to run Mapper and Reducer

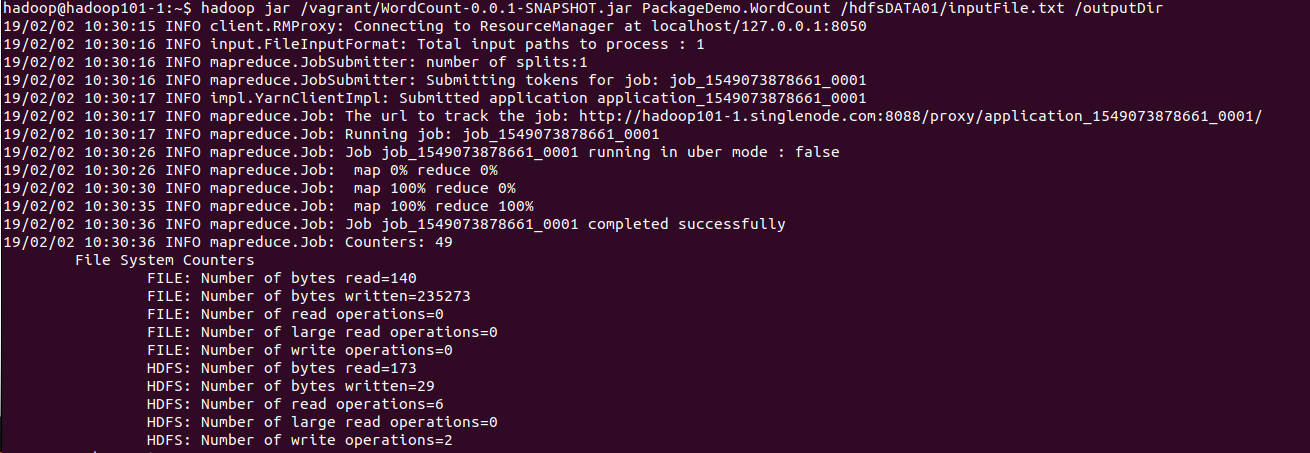
Code is located under Java learning path.. Once code(jar file) is ready to test, copy file to Environments Directory so that it will be available from Guest OS

Once code is copied, create a text file and upload it to HDFS file system using following commands

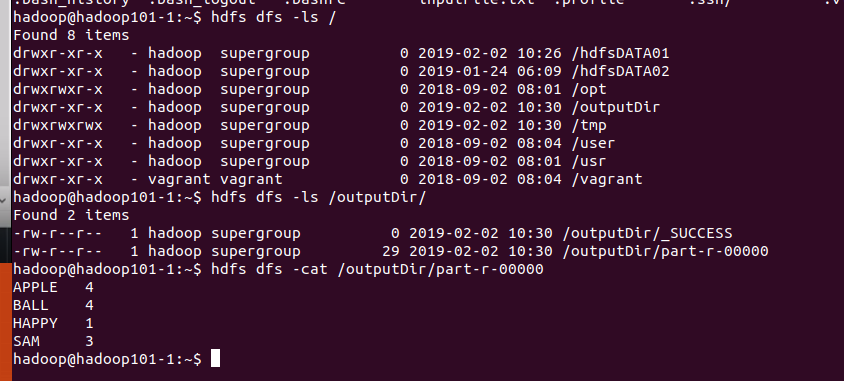
[](https://github.com/psanthoshkumar/BigDataLearning/blob/master/pictures/filesListed_and_WordCount_jarFile.png)

[](https://github.com/psanthoshkumar/BigDataLearning/blob/master/pictures/put_inputFile_to_HDFS.png)

once file has been loaded, we are ready for pilot test.. Run the code as given below

[](https://github.com/psanthoshkumar/BigDataLearning/blob/master/pictures/runProgram.png)

if there are no errors, we will see output like below. complete log is copied to log directory in Source

[](https://github.com/psanthoshkumar/BigDataLearning/blob/master/pictures/output.png)