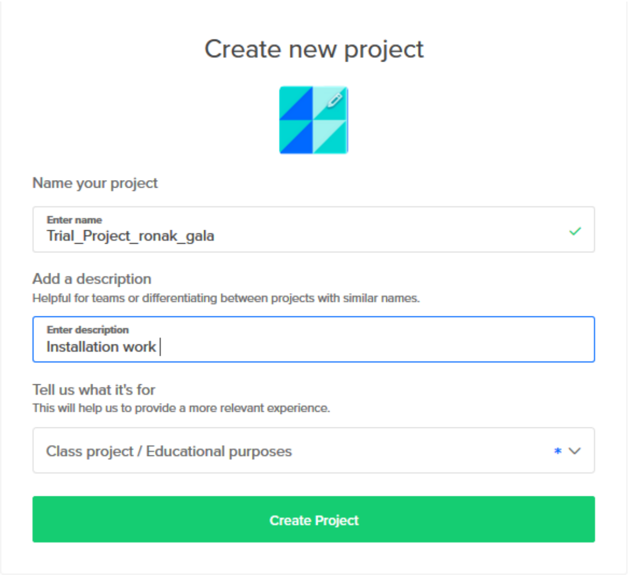
Practical 1 :

Aim : Install, configure and run Hadoop and HDFS ad explore HDFS.

Phase 1 : Creating Droplet

Step 1 : Sign up on https://www.digitalocean.com/

Step 2 : Create a new project.



Create new project

Name your project

Enter name: Trial_Project_ronak_gala

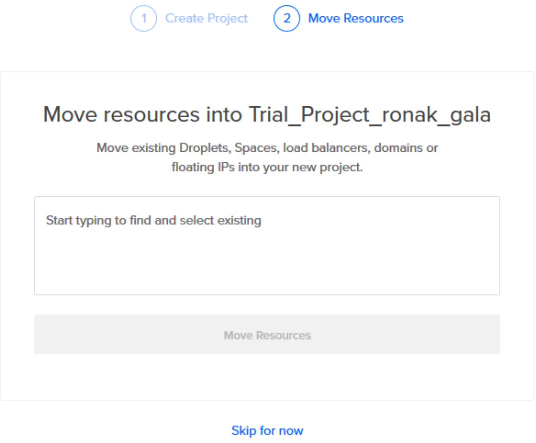
Add a description
Helpful for teams or differentiating between projects with similar names.

Enter description: Installation work

Tell us what it's for
This will help us to provide a more relevant experience.

Class project / Educational purposes

Step 3 : You can skip the move resources page.

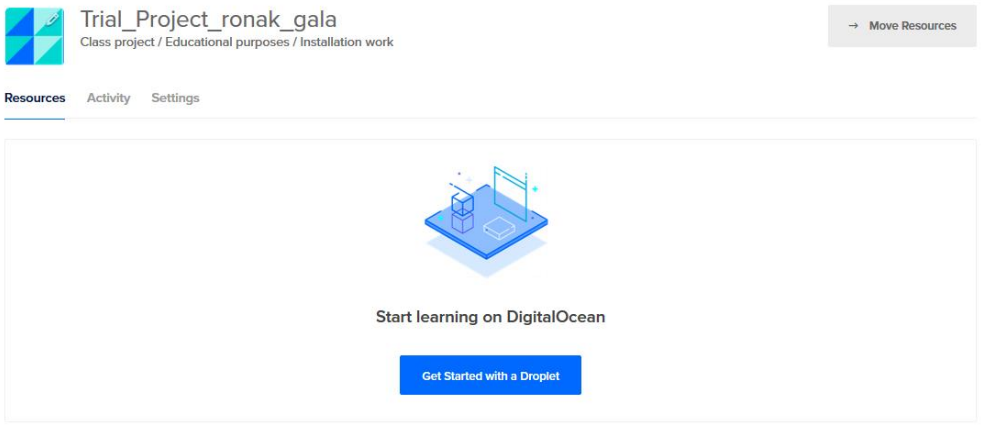


Move resources into Trial_Project_ronak_gala

Move existing Droplets, Spaces, load balancers, domains or floating IPs into your new project.

Start typing to find and select existing

Step 4 : Click ‘ Get Started with a Droplet’.



Trial_Project_ronak_gala

Class project / Educational purposes / Installation work

Start learning on DigitalOcean

Step 5 : Select the following configurations :-

Choose an Image – Ubuntu

Choose datacentre region – Bangalore

Authentication – OTP (you can also select SSH but you might need to set it up)

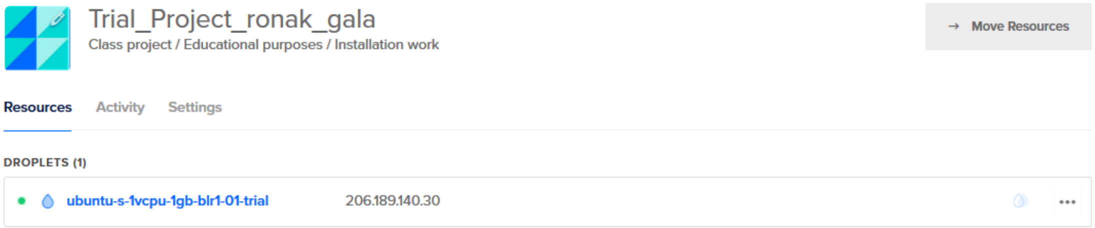
How many droplets – 1

Choose a new hostname or go with the provided one

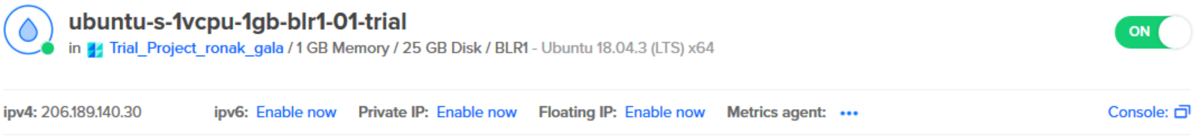
Then select Create Droplet.

Phase 2: Installing JDK

Step 6 : Click on the droplet after it has been created.



Step 7 : Click on console.


ubuntu-s-1vcpu-1gb-blr1-01-trial
in Trial_Project_ronak_gala / 1 GB Memory / 25 GB Disk / BLR1 - Ubuntu 18.04.3 (LTS) x64
ON
ipv4: 206.189.140.30
ipv6: Enable now
Private IP: Enable now
Floating IP: Enable now
Metrics agent: •••
Console:

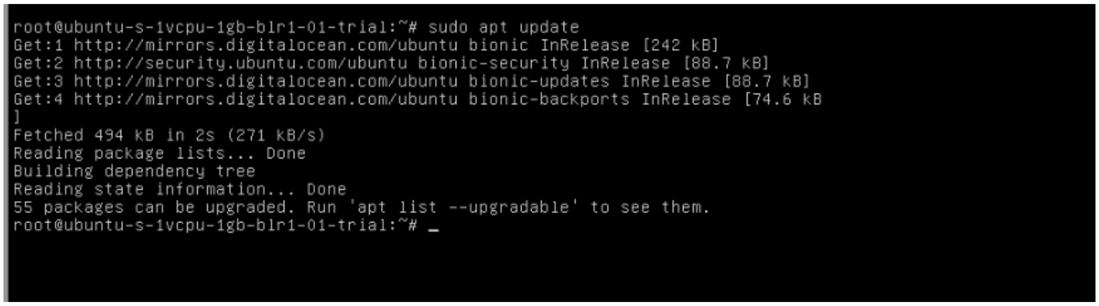
A new window will open with the console.

Step 8 : Login in to your registered email id and look for a mail with username and password of your new droplet. Insert the credentials and change the password as per your requirement.

Note : For password the cursor doesn’t move, i.e. it won’t show the password length just like ubuntu and other Linux flavours.

Step 9 : Check for updates

Command : sudo apt update



```
root@ubuntu-s-1vcpu-1gb-blr1-01-trial:~# sudo apt update
Get:1 http://mirrors.digitalocean.com/ubuntu bionic InRelease [242 kB]
Get:2 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:3 http://mirrors.digitalocean.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:4 http://mirrors.digitalocean.com/ubuntu bionic-backports InRelease [74.6 kB
]
Fetched 494 kB in 2s (271 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
55 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ubuntu-s-1vcpu-1gb-blr1-01-trial:~# _
```

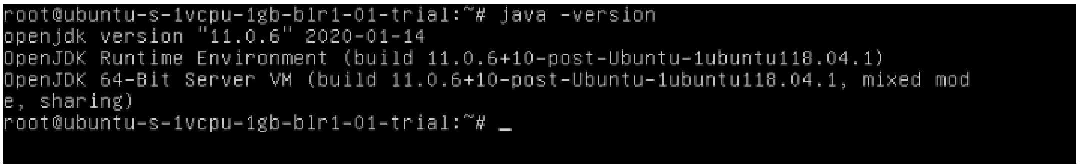
Step 10 : Install OpenJDK

Command : sudo apt install default-jdk

It will ask for permission as it will take up 727mb of disk, so allow it accordingly.

Step 11 : Check for the java version.

Command : java -version

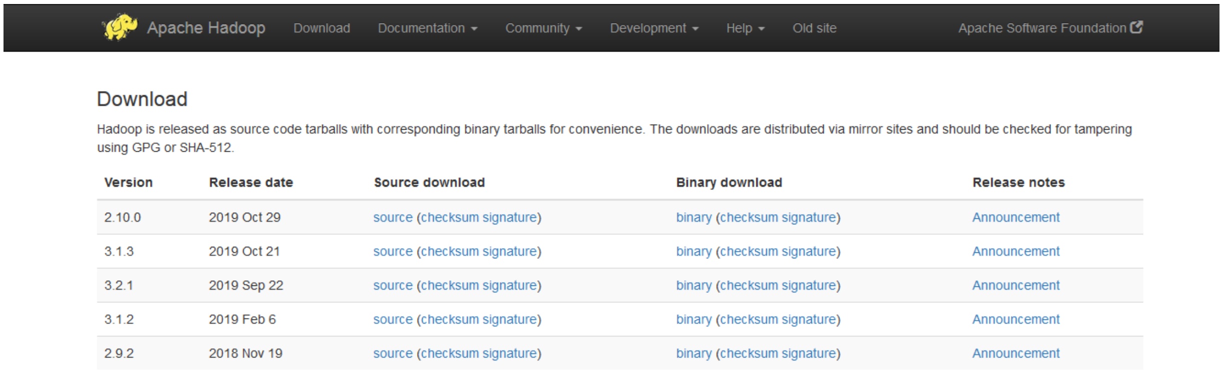


```
root@ubuntu-s-1vcpu-1gb-blr1-01-trial:~# java -version
openjdk version "11.0.6" 2020-01-14
OpenJDK Runtime Environment (build 11.0.6+10-post-Ubuntu-1ubuntu118.04.1)
OpenJDK 64-Bit Server VM (build 11.0.6+10-post-Ubuntu-1ubuntu118.04.1, mixed mod
e, sharing)
root@ubuntu-s-1vcpu-1gb-blr1-01-trial:~# _
```

Phase 3: Installing Hadoop

Step 12 : Go to https://hadoop.apache.org/releases.html to get the stable release.

Step 13 : Navigate to the binary you wish to install. In this guide we will be installing 3.1.3



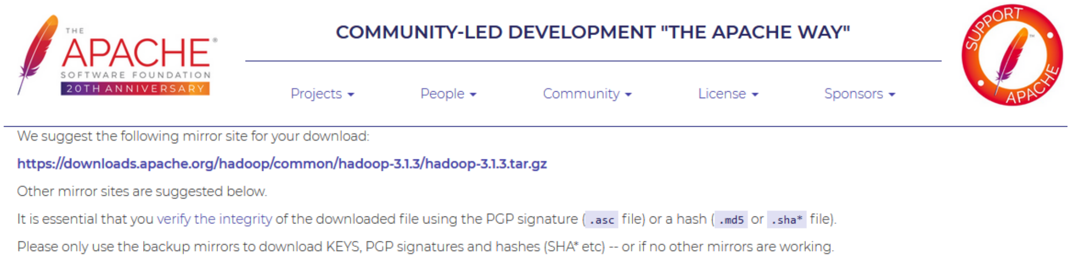
Download

Hadoop is released as source code tarballs with corresponding binary tarballs for convenience. The downloads are distributed via mirror sites and should be checked for tampering using GPG or SHA-512.

Version	Release date	Source download	Binary download	Release notes
2.10.0	2019 Oct 29	source (checksum signature)	binary (checksum signature)	Announcement
3.1.3	2019 Oct 21	source (checksum signature)	binary (checksum signature)	Announcement
3.2.1	2019 Sep 22	source (checksum signature)	binary (checksum signature)	Announcement
3.1.2	2019 Feb 6	source (checksum signature)	binary (checksum signature)	Announcement
2.9.2	2018 Nov 19	source (checksum signature)	binary (checksum signature)	Announcement

Step 14 : Click on binary, from your desired version’s row.

Step 15 : Copy the http url.


THE APACHE SOFTWARE FOUNDATION 20TH ANNIVERSARY
COMMUNITY-LED DEVELOPMENT "THE APACHE WAY"
Projects ▾
People ▾
Community ▾
License ▾
Sponsors ▾
SUPPORT APACHE

We suggest the following mirror site for your download:

https://downloads.apache.org/hadoop/common/hadoop-3.1.3/hadoop-3.1.3.tar.gz

Other mirror sites are suggested below.

It is essential that you verify the integrity of the downloaded file using the PGP signature (`.asc` file) or a hash (`.md5` or `.sha*` file).

Please only use the backup mirrors to download KEYS, PGP signatures and hashes (SHA* etc) -- or if no other mirrors are working.

Step 16 : On the droplet’s console we will use ‘wget’ to fetch it.

Command : wget https://downloads.apache.org/hadoop/common/hadoop-3.1.3/hadoop-3.1.3.tar.gz

Step 17 : We’ll use the tar command with the -x flag to extract, -z to uncompress, -v for verbose output, and -f to specify that we’re extracting from a file.

Command : tar -xzvf hadoop–3.1.3.tar.gz

Step 18 : We’ll move the extracted files into /usr/local

Command : sudo mv hadoop-3.1.3 /usr/local/Hadoop

Phase 4: Configuring Hadoop’s Java Home

Step 19 : We’ll find java’s default path

Command : readlink -f /usr/bin/java | sed "s:bin/java::"

Output : /usr/lib/jvm/java-11-openjdk-amd64/

Step 20 : You can copy this output to set Hadoop’s Java home to this specific version, which ensures that if the default Java changes, this value will not. Alternatively, you can use the readlink command dynamically in the file so that Hadoop will automatically use whatever Java version is set as the system default.

To begin, open hadoop-env.sh:

Command : sudo nano /usr/local/hadoop/etc/hadoop/hadoop-env.sh

Step 21 : Move the cursor till the last line and then to set the value to Java_Home write this command:

Option 1 : Set a Static Value

Command : #export JAVA_HOME=${JAVA_HOME} (this line is a comment)

export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64/

Option 2 :Use Readlink to Set the Value Dynamically

Command : #export JAVA_HOME=${JAVA_HOME} (this line is a comment)

export JAVA_HOME=$(readlink -f /usr/bin/java | sed "s:bin/java::")

Press ctrl + X to exit file

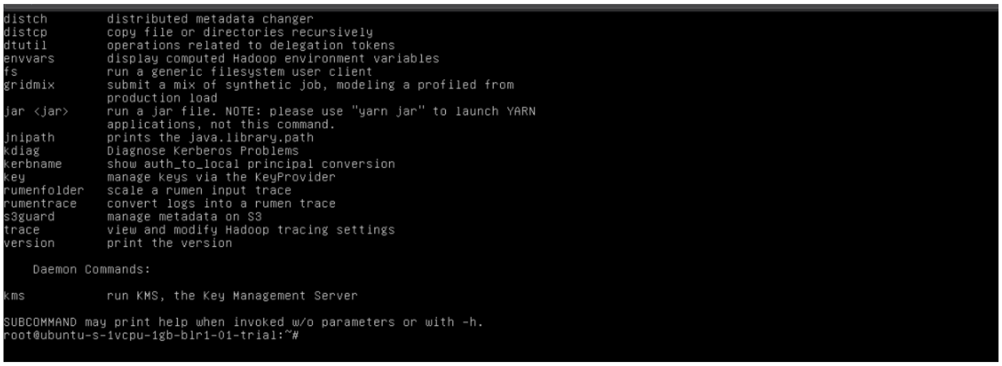
It will ask whether you want to save the modified buffer, answer with YES

It will ask for a file name to write press ENTER

Phase 5: Running Hadoop

Step 22: We will run Hadoop

Command : /usr/local/hadoop/bin/hadoop



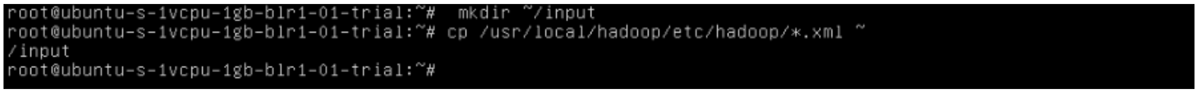
```
distch        distributed metadata changer
distcp        copy file or directories recursively
dtutil        operations related to delegation tokens
envvars       display computed Hadoop environment variables
fs            run a generic filesystem user client
gridmix       submit a mix of synthetic job, modeling a profiled from
              production load
jar <jar>     run a jar file. NOTE: please use "yarn jar" to launch YARN
              applications, not this command.
jnipath       prints the java.library.path
kdiag         Diagnose Kerberos Problems
kerbname      show auth_to_local principal conversion
key           manage keys via the KeyProvider
rumenfolder   scale a rumen input trace
rumentrace    convert logs into a rumen trace
s3guard       manage metadata on S3
trace         view and modify Hadoop tracing settings
version       print the version

    Daemon Commands:

kms           run KMS, the Key Management Server

SUBCOMMAND may print help when invoked w/o parameters or with -h.
root@ubuntu-s-1vcpu-1gb-blr1-01-trial:~#
```

The help means we’ve successfully configured Hadoop to run in stand-alone mode.

Step 23 : We'll ensure that it is functioning properly by running the example MapReduce program it ships with. To do so, create a directory called input in our home directory and copy Hadoop's configuration files into it to use those files as our data.

Command 1: mkdir ~/input

Command 2: cp /usr/local/hadoop/etc/hadoop/*.xml ~/input



```
root@ubuntu-s-1vcpu-1gb-blr1-01-trial:~#  mkdir ~/input
root@ubuntu-s-1vcpu-1gb-blr1-01-trial:~# cp /usr/local/hadoop/etc/hadoop/*.xml ~
/input
root@ubuntu-s-1vcpu-1gb-blr1-01-trial:~#
```

Step 24 : We can use the following command to run the MapReduce hadoop-mapreduce-examples program, a Java archive with several options. We'll invoke its grep program, one of the many examples included in hadoop-mapreduce-examples, followed by the input directory, input and the output directory grep_example. The MapReduce grep program will count the matches of a literal word or regular expression. Finally, we'll supply the regular expression allowed[.]* to find occurrences of the word allowed within or at the end of a declarative sentence. The expression is case-sensitive, so we wouldn't find the word if it were capitalized at the beginning of a sentence:

Command : /usr/local/hadoop/bin/hadoop jar /usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.1.3.jar grep ~/input ~/grep_example 'allowed[.]*'

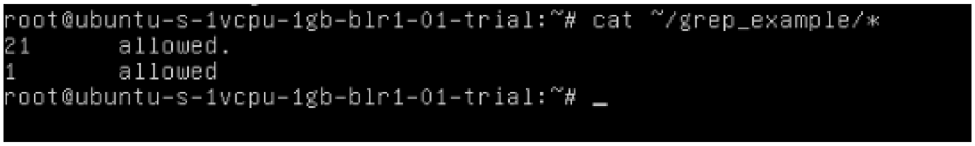
Note: If the output directory already exists, the program will fail, and rather than seeing the summary, the output will look something like:

Output

```
 . . .
    at java.base/java.lang.reflect.Method.invoke(Method.java:564)
    at org.apache.hadoop.util.RunJar.run(RunJar.java:244)
    at org.apache.hadoop.util.RunJar.main(RunJar.java:158)
```

Step 25 : Results are stored in the output directory and can be checked by running cat on the output directory:

Command : cat ~/grep_example/*



```
root@ubuntu-s-1vcpu-1gb-blr1-01-trial:~# cat ~/grep_example/*
21      allowed.
1       allowed
root@ubuntu-s-1vcpu-1gb-blr1-01-trial:~# _
```

The MapReduce task found 21 occurrences of the word allowed followed by a period and one occurrence where it was not. Running the example program has verified that our stand-alone installation is working properly and that non-privileged users on the system can run Hadoop for exploration or debugging.