

RK

# Hadoop K1 Group 10

Muhammad Najih Aflah - 2106653880

Mochammad Dyenta D - 2106731245

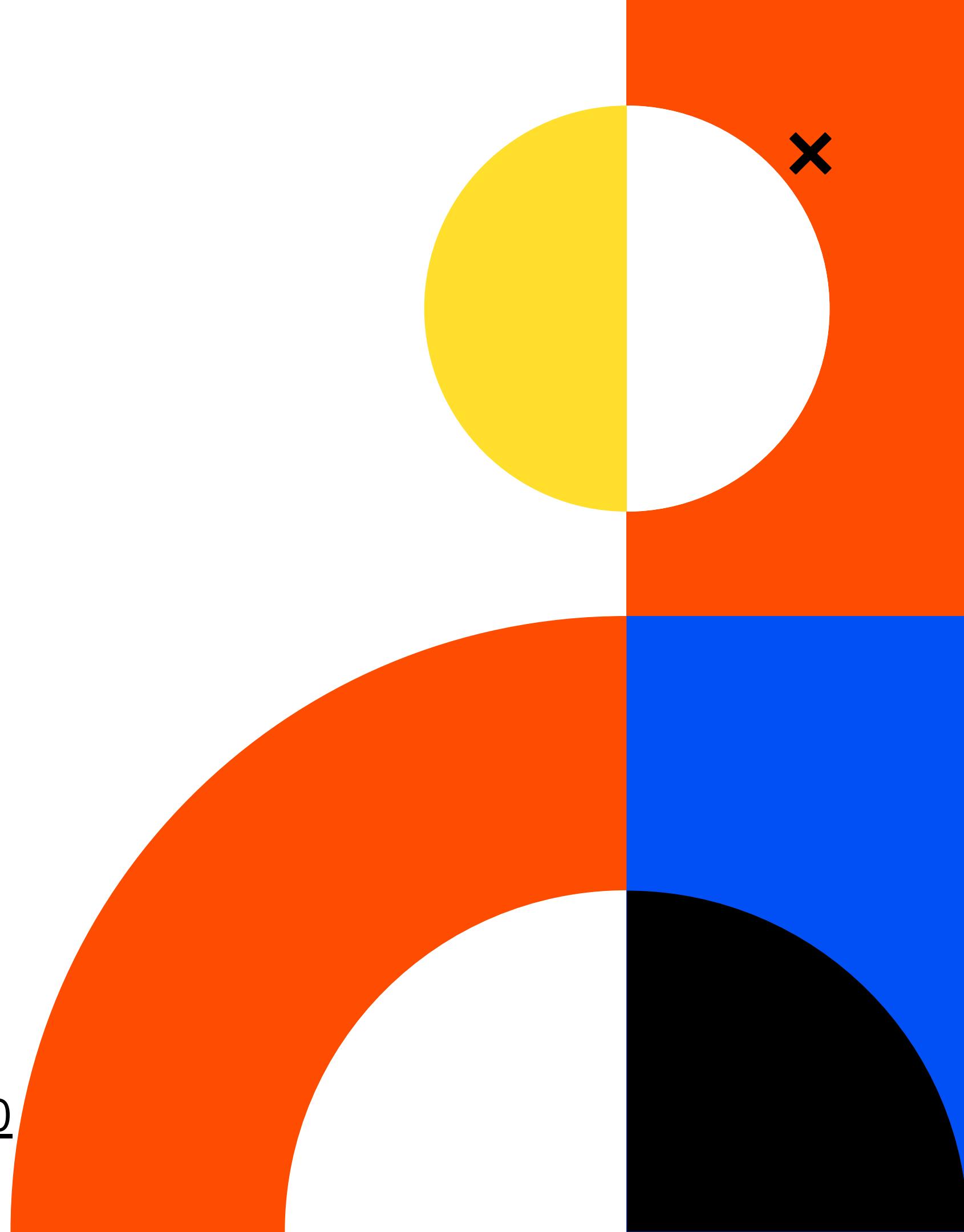
Rayhan Akbar Arrizky - 2106632655

Muhammad Fajri Alqomaril - 2106651635

Agung Firmansyah - 2006577454

Link Github:

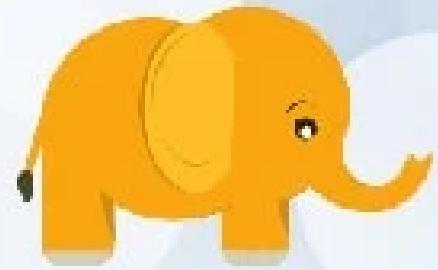
[github.com/rayhan-akbar/Hadoop\\_SBD-K1-Group10](https://github.com/rayhan-akbar/Hadoop_SBD-K1-Group10)





# Apa Itu Hadoop

Hadoop adalah kerangka kerja (framework) open-source yang dirancang untuk memproses, menyimpan, dan menganalisis jumlah data besar (big data) secara terdistribusi. Dikembangkan oleh Apache Software Foundation, Hadoop menyediakan solusi untuk mengatasi tantangan dalam pengelolaan dan analisis data yang sangat besar yang tidak dapat ditangani oleh sistem tradisional.



# Hadoop Ecosystem



**oozie**  
(Work flow)

**HCatalog**

Table & schema  
Management



**Pig**  
(Scripting)



**Hive**  
(Sql Query)



**Mahout**  
(Machine Learning)



**Drill**  
(Interactive Analysis)



**AVRO**  
(JSON)

**Thrift**

( Cross  
Language  
Service)



**HBASE**

(Columnar  
Store)



**Sqoop**  
(Data Collection)



**Zookeeper**  
(Coordination)



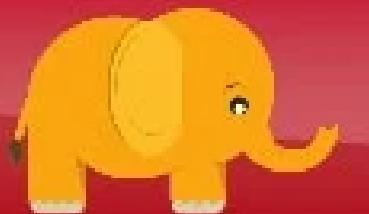
**Apache Ambari**  
(Management & Monitoring)

**Mapreduce**  
(Data Processing)



**Yarn**

(Cluster Resource Management)



**HDFS**

(Hadoop Distributed File system)



# Komponen utama Hadoop

RK

## HDFS (Hadoop Distributed File System)

HDFS adalah sistem file terdistribusi yang dirancang khusus untuk menyimpan data besar dalam cluster Hadoop.

## Apache MapReduce

Apache MapReduce membagi tugas pemrosesan menjadi langkah "map" dan "reduce" untuk pemrosesan paralel di dalam cluster Hadoop.

## YARN (Yet Another Resource Negotiator)

Apache YARN bertanggung jawab atas manajemen dan penjadwalan sumber daya di dalam cluster Hadoop, memungkinkan berbagai jenis aplikasi untuk berjalan secara bersamaan dan efisien.



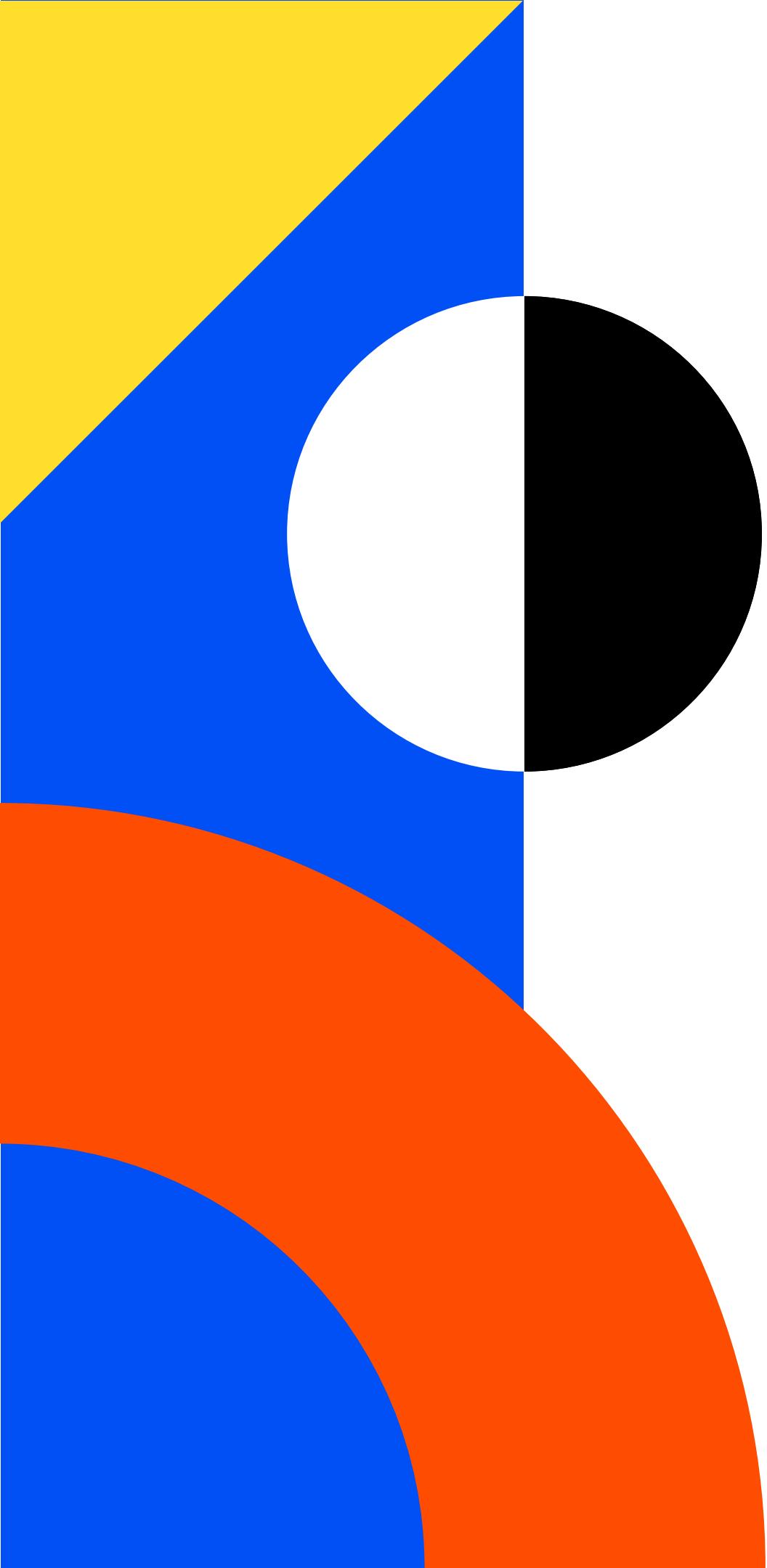
# Kelebihan dan Kekurangan ×

## Kelebihan

1. Skalabilitas: Dapat diukur secara horizontal untuk menangani pertumbuhan data yang besar.
2. Pemrosesan Terdistribusi: Memungkinkan pemrosesan data secara paralel di seluruh cluster.
3. Toleransi Kesalahan: Mampu mengatasi kegagalan perangkat keras atau perangkat lunak tanpa kehilangan data.
4. Ketersediaan Data: Menggunakan replikasi data untuk memastikan ketersediaan data yang tinggi.
5. Ekosistem yang Kaya: Menyediakan berbagai komponen tambahan untuk fungsionalitas dan fleksibilitas yang lebih luas.

## Kekurangan

1. Kompleksitas: Membutuhkan pemahaman teknis yang baik dan pengelolaan cluster yang rumit.
2. Latensi: Penundaan dalam pemrosesan data real-time atau interaktif.
3. Overhead: Tingginya overhead untuk pemrosesan data yang lebih kecil.
4. Keterbatasan Analisis Kompleks: Mungkin tidak efisien untuk analisis kompleks dan iteratif.

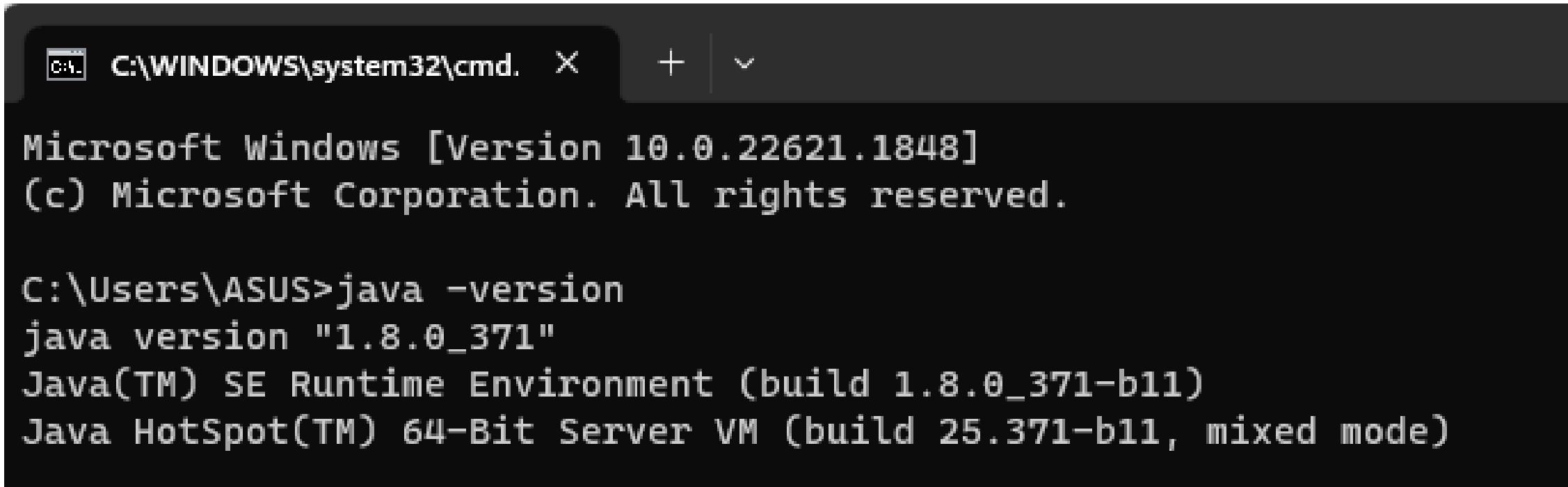


x

# Instalasi Hadoop



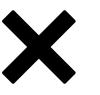
# Install Java 8



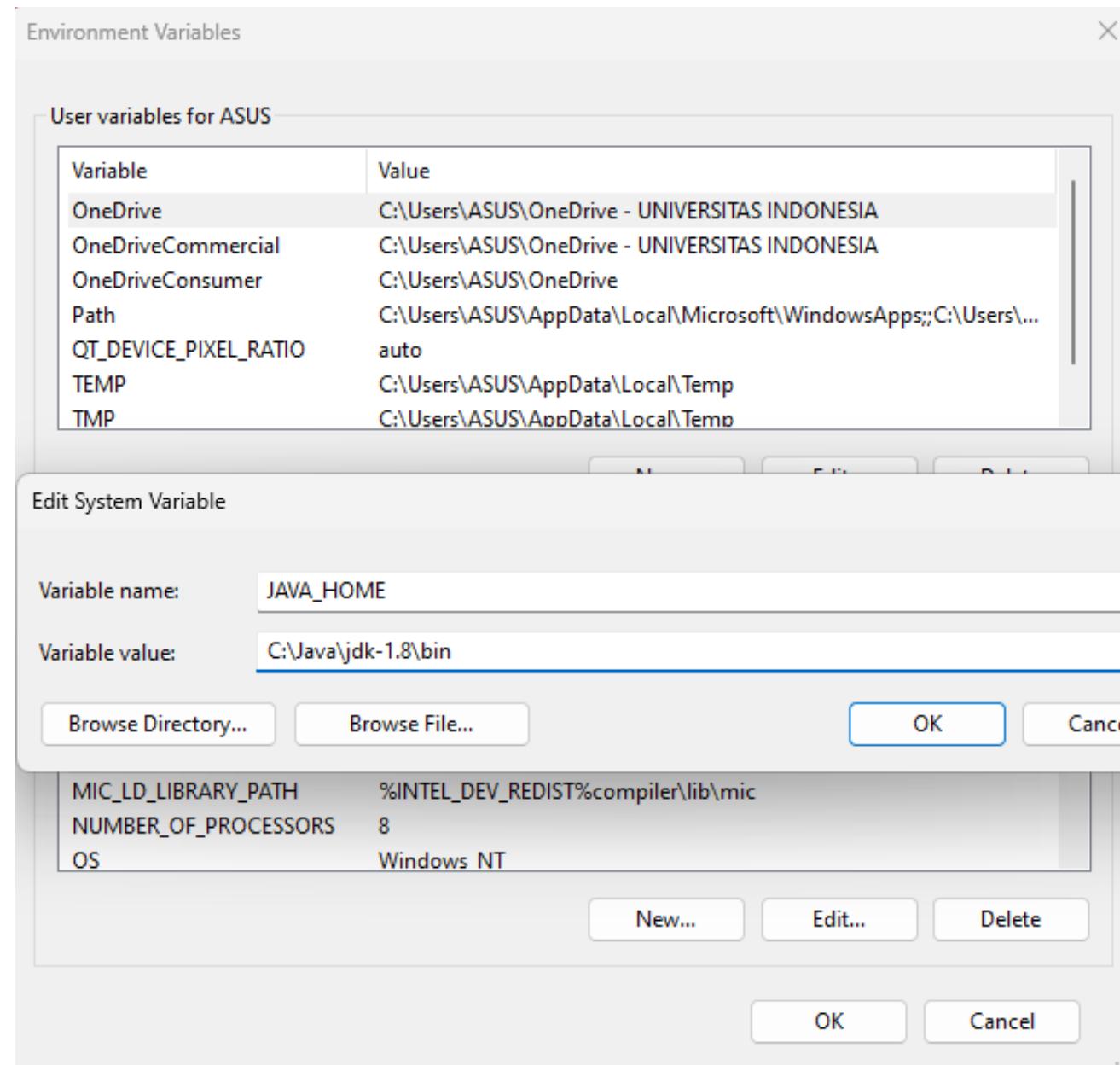
```
C:\WINDOWS\system32\cmd. + ▾
Microsoft Windows [Version 10.0.22621.1848]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ASUS>java -version
java version "1.8.0_371"
Java(TM) SE Runtime Environment (build 1.8.0_371-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.371-b11, mixed mode)
```

Sebelum melakukan instalasi hadoop, hal yang pertama dilakukan memastikan Java 8 telah terinstall pada perangkat/laptop



# Environment Java



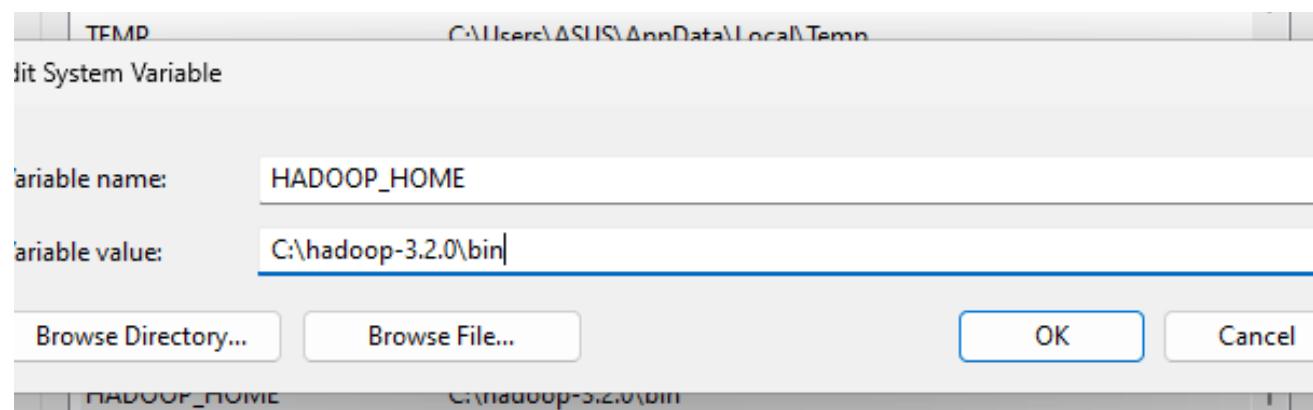
Membuat environment variable pada Java yang telah didownload/terinstall pada perangkat dengan nama variablenya adalah "JAVA\_HOME" dengan variable value berisikan path dari folder Java yang telah terinstall tersebut



# Hadoop

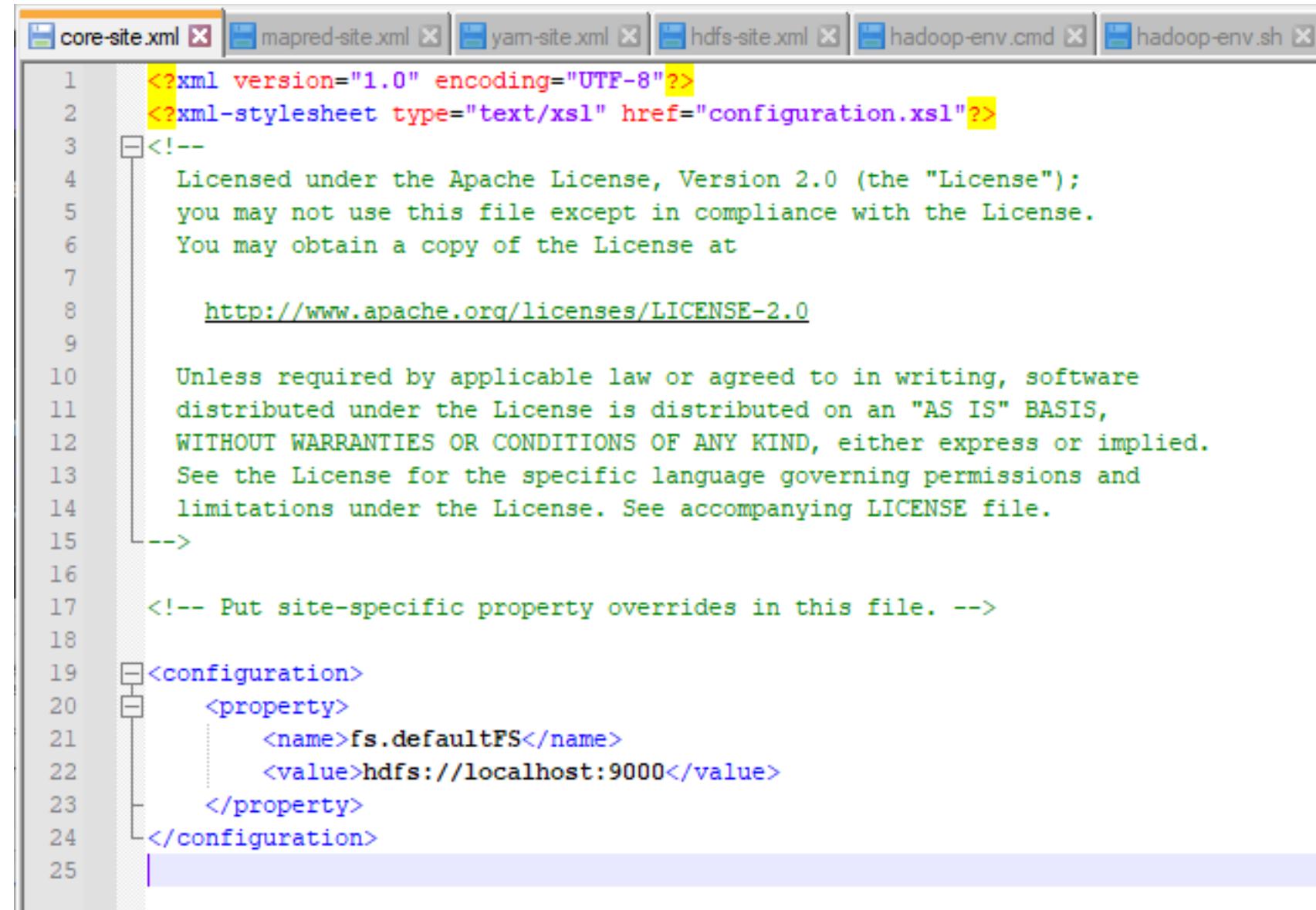
```
C:\Users\ASUS>hadoop version
Hadoop 3.2.0
Source code repository https://github.com/apache/hadoop.git -r e97acb3bd8f3befd27418996fa5d4b50bf2e17bf
Compiled by sunilg on 2019-01-08T06:08Z
Compiled with protoc 2.5.0
From source with checksum d3f0795ed0d9dc378e2c785d3668f39
This command was run using /C:/hadoop-3.2.0/share/hadoop/common/hadoop-common-3.2.0.jar
```

Download hadoop versio yang sesuai dengan keinginan masing-masing. Setelah mendownload dan mengekstrak file hadoop yang telah didownload tersebut, selanjutnya membuat environment variable seperti yang dilakukan pada Java sebelumnya. Dimana variable name diberi nama "HADOOP\_HOME" dan variable value diisi dengan path file hadoop tersebut diekstrak.





# Hadoop Configuration



```
core-site.xml mapred-site.xml yam-site.xml hdfs-site.xml hadoop-env.cmd hadoop-env.sh

1 <?xml version="1.0" encoding="UTF-8"?>
2 <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
3 <!--
4 Licensed under the Apache License, Version 2.0 (the "License");
5 you may not use this file except in compliance with the License.
6 You may obtain a copy of the License at
7
8     http://www.apache.org/licenses/LICENSE-2.0
9
10 Unless required by applicable law or agreed to in writing, software
11 distributed under the License is distributed on an "AS IS" BASIS,
12 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
13 See the License for the specific language governing permissions and
14 limitations under the License. See accompanying LICENSE file.
15 -->
16
17 <!-- Put site-specific property overrides in this file. -->
18
19 <configuration>
20   <property>
21     <name>fs.defaultFS</name>
22     <value>hdfs://localhost:9000</value>
23   </property>
24 </configuration>
25
```

Setelah pembuatan variable hadoop, selanjutnya adalah melakukan konfigurasi pada beberapa file pada hadoop sebelum dijalankan. Konfigurasi dilakukan pada directory file hadoop yang berada pada "hadoop-3.2.0\etc\hadoop". Dimana pada file core-site.xml pada bagian configuration diubah menjadi seperti yang dibawah ini :

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



# Hadoop Configuration

```
core-site.xml x mapred-site.xml x yam-site.xml x hdfs-site.xml x hadoop-env.cmd x hadoop-env.sh x
1 <?xml version="1.0"?>
2 <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
3 <!--
4 Licensed under the Apache License, Version 2.0 (the "License");
5 you may not use this file except in compliance with the License.
6 You may obtain a copy of the License at
7
8     http://www.apache.org/licenses/LICENSE-2.0
9
10 Unless required by applicable law or agreed to in writing, software
11 distributed under the License is distributed on an "AS IS" BASIS,
12 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
13 See the License for the specific language governing permissions and
14 limitations under the License. See accompanying LICENSE file.
15 -->
16
17 <!-- Put site-specific property overrides in this file. -->
18
19 <configuration>
20   <property>
21     <name>mapreduce.framework.name</name>
22     <value>yarn</value>
23   </property>
24 </configuration>
25
```

Selanjutnya pada file mapred-site.xml dan pada bagian configuration diubah menjadi seperti dibawah ini :

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



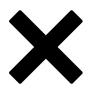
# Hadoop Configuration

```
1  <?xml version="1.0"?>
2  <!--
3      Licensed under the Apache License, Version 2.0 (the "License");
4      you may not use this file except in compliance with the License.
5      You may obtain a copy of the License at
6
7      http://www.apache.org/licenses/LICENSE-2.0
8
9      Unless required by applicable law or agreed to in writing, software
10     distributed under the License is distributed on an "AS IS" BASIS,
11     WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
12     See the License for the specific language governing permissions and
13     limitations under the License. See accompanying LICENSE file.
14 -->
15 <configuration>
16
17     <!-- Site specific YARN configuration properties -->
18
19     <property>
20         <name>yarn.nodemanager.aux-services</name>
21         <value>mapreduce_shuffle</value>
22     </property>
23
24     <property>
25         <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
26         <value>org.apache.hadoop.mapred.ShuffleHandler</value>
27     </property>
28
29 </configuration>
30 |
```

Selanjutnya pada file yarn-site.xml dan pada bagian configuration diubah menjadi seperti dibawah ini :

```
<configuration>
    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
    </property>

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



# Hadoop Configuration

```
core-site.xml mapred-site.xml yam-site.xml hdfs-site.xml hadoop-env.cmd hadoop-env.sh
1  <?xml version="1.0" encoding="UTF-8"?>
2  <?xmlstylesheet type="text/xsl" href="configuration.xsl"?>
3  <!--
4      Licensed under the Apache License, Version 2.0 (the "License");
5      you may not use this file except in compliance with the License.
6      You may obtain a copy of the License at
7
8          http://www.apache.org/licenses/LICENSE-2.0
9
10     Unless required by applicable law or agreed to in writing, software
11     distributed under the License is distributed on an "AS IS" BASIS,
12     WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
13     See the License for the specific language governing permissions and
14     limitations under the License. See accompanying LICENSE file.
15 -->
16
17  <!-- Put site-specific property overrides in this file. -->
18
19 <configuration>
20   <property>
21     <name>dfs.replication</name>
22     <value>1</value>
23   </property>
24
25   <property>
26     <name>dfs.namenode.name.dir</name>
27     <value>C:\hadoop-3.2.0\data\namenode</value>
28   </property>
29
30   <property>
31     <name>dfs.datanode.name.dir</name>
32     <value>C:\hadoop-3.2.0\data\datanode</value>
33   </property>
34 </configuration>
```

Selanjutnya pada file hdfs-site.xml dan pada bagian configuration diubah menjadi seperti dibawah ini :

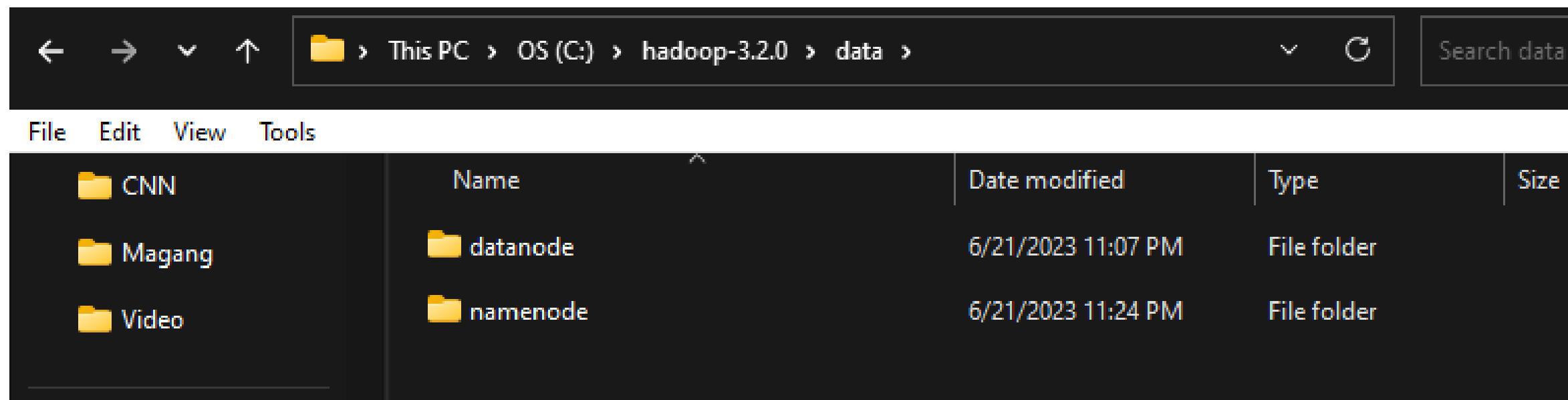
```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>

  <property>
    <name>dfs.namenode.name.dir</name>
    <value>C:\hadoop-3.2.0\data\namenode</value>
  </property>

  <property>
    <name>dfs.datanode.name.dir</name>
    <value>C:\hadoop-3.2.0\data\datanode</value>
  </property>
</configuration>
```



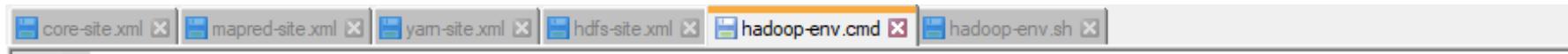
# Hadoop Configuration



Setelah melakukan konfigurasi pada file hdfs-site.xml, selanjutnya adalah membuat folder yang sesuai dengan konfigurasi yang dilakukan pada hdfs-site.xml, yaitu membuat folder data yang berisikan folder datanode dan namenode.



# Hadoop Configuration



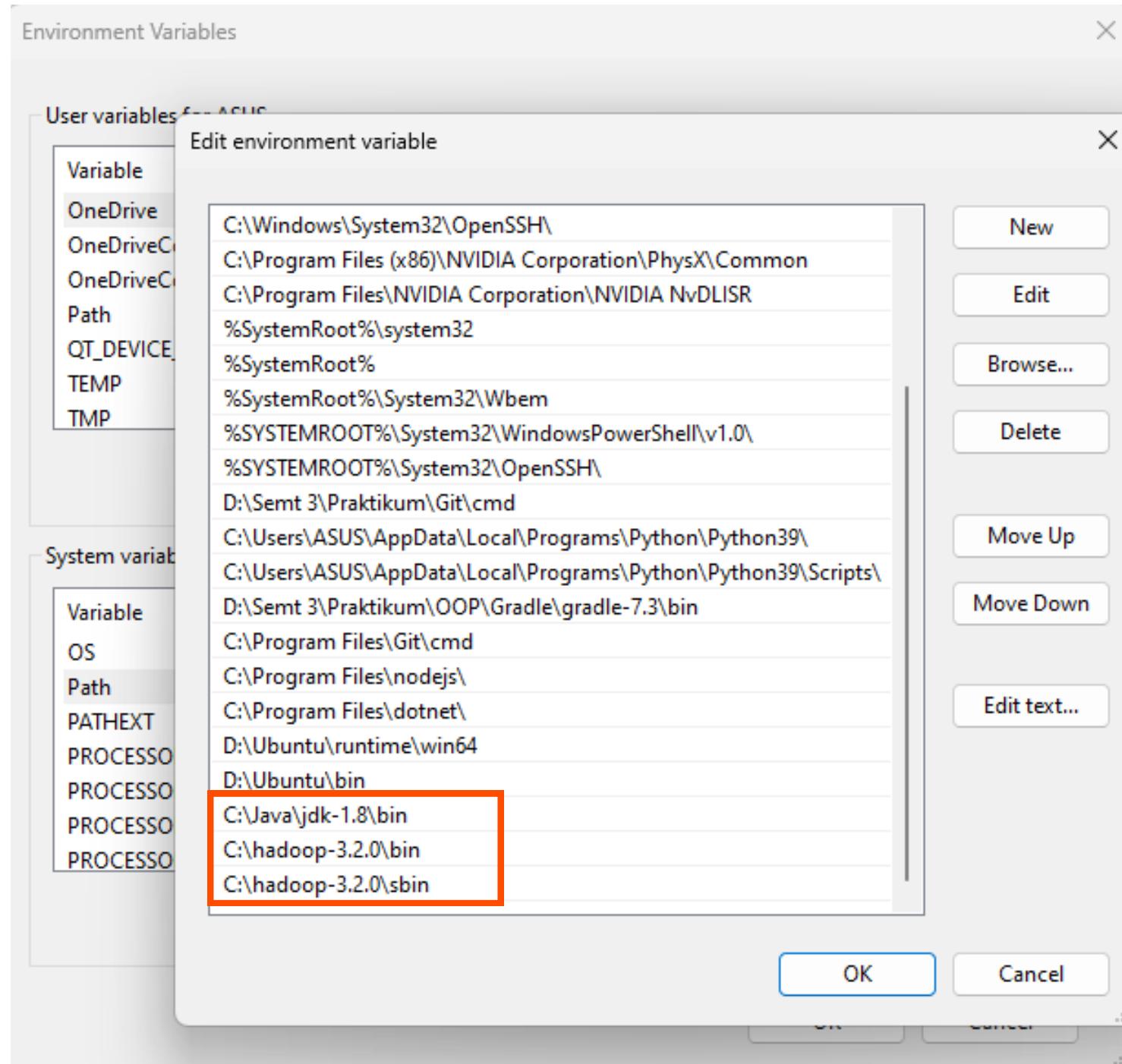
```
core-site.xml mapred-site.xml yarn-site.xml hdfs-site.xml hadoop-env.cmd hadoop-env.sh

1 echo off
2 @rem Licensed to the Apache Software Foundation (ASF) under one or more
3 @rem contributor license agreements. See the NOTICE file distributed with
4 @rem this work for additional information regarding copyright ownership.
5 @rem The ASF licenses this file to You under the Apache License, Version 2.0
6 @rem (the "License"); you may not use this file except in compliance with
7 @rem the License. You may obtain a copy of the License at
8 @rem
9 @rem     http://www.apache.org/licenses/LICENSE-2.0
10 @rem
11 @rem Unless required by applicable law or agreed to in writing, software
12 @rem distributed under the License is distributed on an "AS IS" BASIS,
13 @rem WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
14 @rem See the License for the specific language governing permissions and
15 @rem limitations under the License.
16
17 @rem Set Hadoop-specific environment variables here.
18 |
19 @rem The only required environment variable is JAVA_HOME. All others are
20 @rem optional. When running a distributed configuration it is best to
21 @rem set JAVA_HOME in this file, so that it is correctly defined on
22 @rem remote nodes.
23
24 @rem The java implementation to use. Required.
25 set JAVA_HOME=C:\Java\jdk-1.8"
26
27 @rem The jsvc implementation to use. Jsvc is required to run secure datanodes.
28 @rem set JSVC_HOME=%JSVC_HOME%
29
30 @rem set HADOOP_CONF_DIR=
31
32 @rem Extra Java CLASSPATH elements. Automatically insert capacity-scheduler.
33 if exist %HADOOP_HOME%\contrib\capacity-scheduler (
34   if not defined HADOOP_CLASSPATH (
35     set HADOOP_CLASSPATH=%HADOOP_HOME%\contrib\capacity-scheduler\*.jar
36   ) else (
37     set HADOOP_CLASSPATH=%HADOOP_CLASSPATH%;%HADOOP_HOME%\contrib\capacity-scheduler\*.jar
38   )
39 )
```

Dan konfigurasi terakhir adalah `hadoop-env.cmd`, dimana pada file tersebut varibale set `JAVA_HOME` diganti dengan path yang sesuai dan telah dibuat pada environment variable sebelumnya.



# Java dan Hadoop path



Setelah melakukan konfigurasi pada beberapa file hadoop, selanjutnya adalah menambahkan path pada environment variable yang terletak pada system variables seperti pada gambar disamping dengan kotak yang berwarna merah.



# Namenode format

```
C:\hadoop-3.2.0\sbin>hdfs namenode -format
2023-06-22 00:12:33,566 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = MyCompputer/192.168.135.1
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 3.2.0
STARTUP_MSG:   classpath = C:\hadoop-3.2.0\etc\hadoop;C:\hadoop-3.2.0\share\hadoop\common;C:\hadoop-3.2.0\share\hadoop\common\lib\accessors-smart-1.2.jar;C:\hadoop-3.2.0\share\hadoop\common\lib\asm-5.0.4.jar;C:\hadoop-3.2.0\share\hadoop\common\lib\audience-annotations-0.5.0.jar;C:\hadoop-3.2.0\share\hadoop\common\lib\avro-1.7.7.jar;C:\hadoop-3.2.0\share\hadoop\common\lib\commons-beanutils-1.9.3.jar;C:\hadoop-3.2.0\share\hadoop\common\lib\commons-cli-1.2.jar;C:\hadoop-3.2.0\sh
```

Setelah melakukan langkah-langkah sebelumnya, hal selanjutnya yang dilakukan adalah menjalankan "hdfs namenode -format". hal tersebut berguna untuk menghapus seluruh metadata pada HDFS (Hadoop Distributed File System) dan bukan untuk menjalankan hadoop, dengan kata lain "hdfs namenode -format" digunakan untuk mengatur ulang namenode.

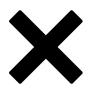


# Run Hadoop

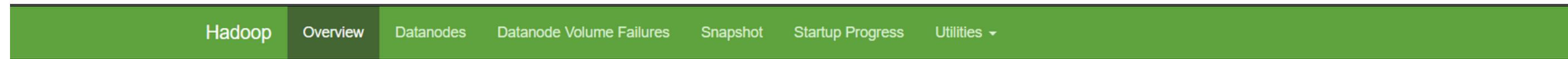
```
C:\Users\ASUS>cd /  
C:\>cd hadoop-3.2.0  
C:\hadoop-3.2.0>cd sbin  
C:\hadoop-3.2.0\sbin>start-all.cmd  
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd  
starting yarn daemons
```

```
C:\hadoop-3.2.0\sbin>jps  
20560 DataNode  
27744 NodeManager  
17928 ResourceManager  
18716 Jps  
21388 NameNode
```

Untuk dapat menjalankan hadoop, hal yang pertama dilakukan adalah masuk ke directory sbin yang berada pada folder hadoop dan menjalankan perintah "start-all.cmd". Dan untuk memeriksa status proses Java yang terkait dengan komponen Hadoop yang sedang berjalan dapat menggunakan perintah "jps".

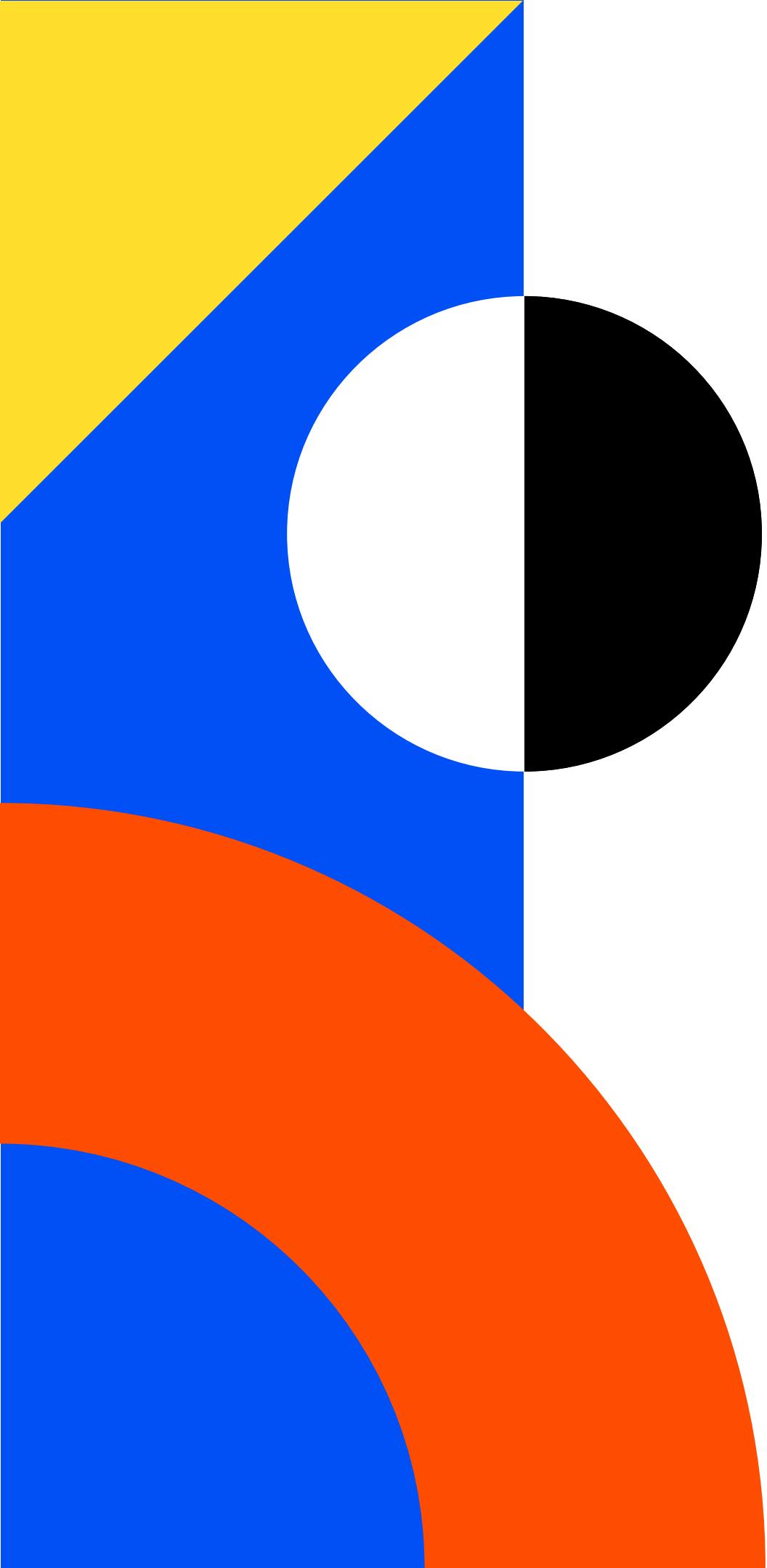


# Run Hadoop



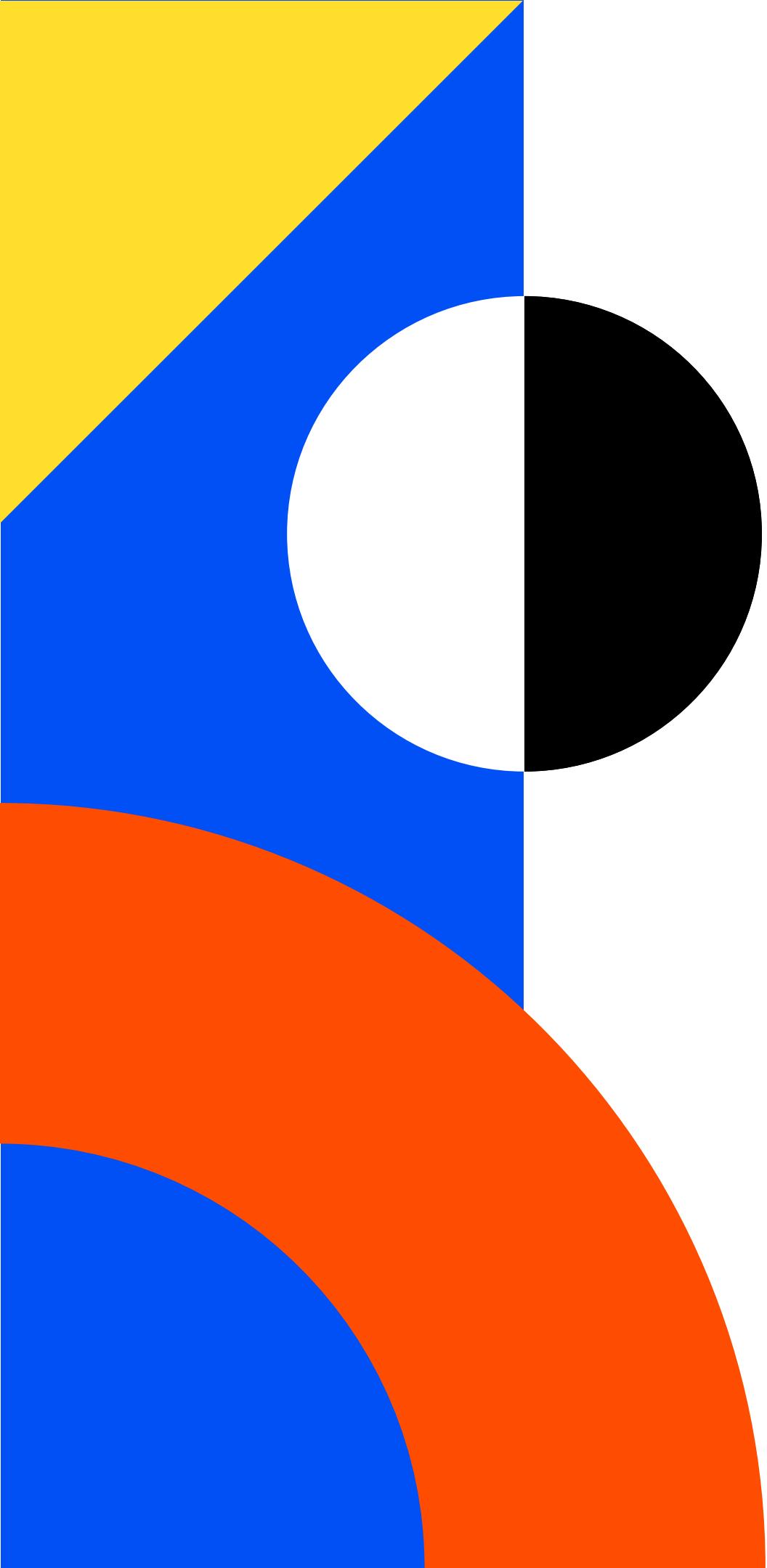
## Overview 'localhost:9000' (active)

Started:	Thu Jun 22 00:23:33 +0700 2023
Version:	3.2.0, re97
Compiled:	Tue Jan 08
Cluster ID:	CID-85266
Block Pool ID:	BP-156183



x

# **Langkah-langkah run Hadoop dengan Ubuntu**



×

# Konfigurasi Hadoop

# Konfigurasi Hadoop

Hadoop unggul ketika digunakan dalam mode terdistribusi penuh pada sekelompok besar server jaringan . Namun, jika Anda baru menggunakan Hadoop dan ingin menjelajahi perintah dasar atau menguji aplikasi, Anda dapat mengonfigurasi Hadoop di satu node.

Setup ini, juga disebut pseudo-distributed mode , memungkinkan setiap daemon Hadoop dijalankan sebagai satu proses Java. Hadoop environment dikonfigurasi dengan mengedit sekumpulan file konfigurasi yaitu :

- bashrc,
- hadoop-env.sh
- core-site.xml
- hdfs-site.xml
- mapred-site-xml
- yarn-site.xml



# File bashrc

```
GNU nano 4.8
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_HOME=/home/hadoop/hadoop
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export HDFS_NAMENODE_USER="hadoop"
export HDFS_DATANODE_USER="hadoop"
export HDFS_SECONDARYNAMENODE_USER="hadoop"
export YARN_RESOURCEMANAGER_USER="hadoop"
export YARN_NODEMANAGER_USER="hadoop"
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export PATH=${JAVA_HOME}/bin:${PATH}
export HADOOP_CLASSPATH=${JAVA_HOME}/lib/tools.jar
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"
# If not running interactively, don't do anything
case $- in
  *i*) ;;
  *) return;;
esac
```

File bashrc merupakan sebuah file yang berguna untuk mengkonfigurasi variabel lingkungan. Kita dapat mengedit file bashrc dengan menjalankan command berikut :

```
$ sudo nano ~/.bashrc
```

Lalu menambahkan konfigurasi pada gambar, setelah itu kita dapat mengaktifkan variabel lingkungan dengan command :

```
$ source ~/.bashrc
```



# File hadoop-env.sh

```
GNU nano 4.8
# Generic settings for HADOOP
###

# Technically, the only required environment variable is JAVA_HOME.
# All others are optional. However, the defaults are probably not
# preferred. Many sites configure these options outside of Hadoop,
# such as in /etc/profile.d

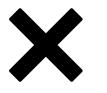
# The java implementation to use. By default, this environment
# variable is REQUIRED on ALL platforms except OS X!
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HDFS_NAMENODE_USER="hadoop"
export HDFS_DATANODE_USER="hadoop"
export HDFS_SECONDARYNAMENODE_USER="hadoop"
export YARN_RESOURCEMANAGER_USER="hadoop"
export YARN_NODEMANAGER_USER="hadoop"
# Location of Hadoop. By default, Hadoop will attempt to determine
# this location based upon its execution path.
# export HADOOP_HOME

# Location of Hadoop's configuration information. i.e., where this
# file is living. If this is not defined, Hadoop will attempt to
# locate it based upon its execution path.
#
# NOTE: It is recommend that this variable not be set here but in
# /etc/profile.d or equivalent. Some options (such as
# --config) may react strangely otherwise.
#
# export HADOOP_CONF_DIR=${HADOOP_HOME}/etc/hadoop
```

File hadoop-env.sh berfungsi sebagai file master untuk mengonfigurasi pengaturan proyek terkait YARN, HDFS , MapReduce , dan Hadoop. Kita dapat mengedit file hadoop-env.sh dengan menjalankan command berikut :

```
sudo nano $HADOOP_HOME/etc/hadoop/hadoop-
env.sh
```

Lalu menambahkan konfigurasi pada gambar.



# File core-site.xml

```
GNU nano 4.8
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
    Licensed under the Apache License, Version 2.0 (the "License");
    you may not use this file except in compliance with the License.
    You may obtain a copy of the License at

        http://www.apache.org/licenses/LICENSE-2.0

    Unless required by applicable law or agreed to in writing, software
    distributed under the License is distributed on an "AS IS" BASIS,
    WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
    See the License for the specific language governing permissions and
    limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

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

File core-site.xml mendefinisikan properti inti HDFS dan Hadoop. Kita dapat mengedit file core-site.xml dengan menjalankan command berikut :

```
sudo nano $HADOOP_HOME/etc/hadoop/core-site.xml
```

Lalu menambahkan konfigurasi pada gambar untuk mengganti nilai default untuk direktori sementara dan tambahkan URL HDFS Anda untuk mengganti pengaturan sistem file lokal default:



# File hdfs-site.xml

```
GNU nano 4.8
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
    <property>
        <name>dfs.replication</name>
        <value>1</value>
    </property>
<!--
    <property>
        <name>dfs.name.dir</name>
        <value>file:///home/hadoop/hadoopdata/hdfs/namenode</value>
    </property>
-->
    <property>
        <name>dfs.data.dir</name>
        <value>file:///home/hadoop/hadoopdata/hdfs/datanode</value>
    </property>
</configuration>
```

Properti dalam file hdfs-site.xml mengatur lokasi untuk menyimpan metadata node, file fsimage, dan mengedit file log. Konfigurasikan file dengan menentukan direktori penyimpanan NameNode dan DataNode . Dalam file “hdfs-site.xml” ini, kita akan mengubah jalur direktori “datanode” dan “namenode”. Kita dapat mengedit file hdfs-site.xml dengan menjalankan command berikut :

```
sudo nano $HADOOP_HOME/etc/hadoop/hdfs-site.xml
```

Lalu menambahkan konfigurasi pada gambar



# File mapred-site.xml

```
GNU nano 4.8
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
  <property>
    <name>mapreduce.application.classpath</name>
    <value>$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/*:$HADOOP_MAPRED_HOME/sha
  </property>
</configuration>
```

Gunakan perintah berikut untuk mengakses file mapred-site.xml dan menentukan nilai MapReduce :

```
sudo nano $HADOOP_HOME/etc/hadoop/mapred-
site.xml
```

Lalu ubah nilai nama framework MapReduce default menjadi yarn seperti konfigurasi pada gambar.

# File yarn-site.xml



```
GNU nano 4.8
yarn-site.xml
<?xml version="1.0"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

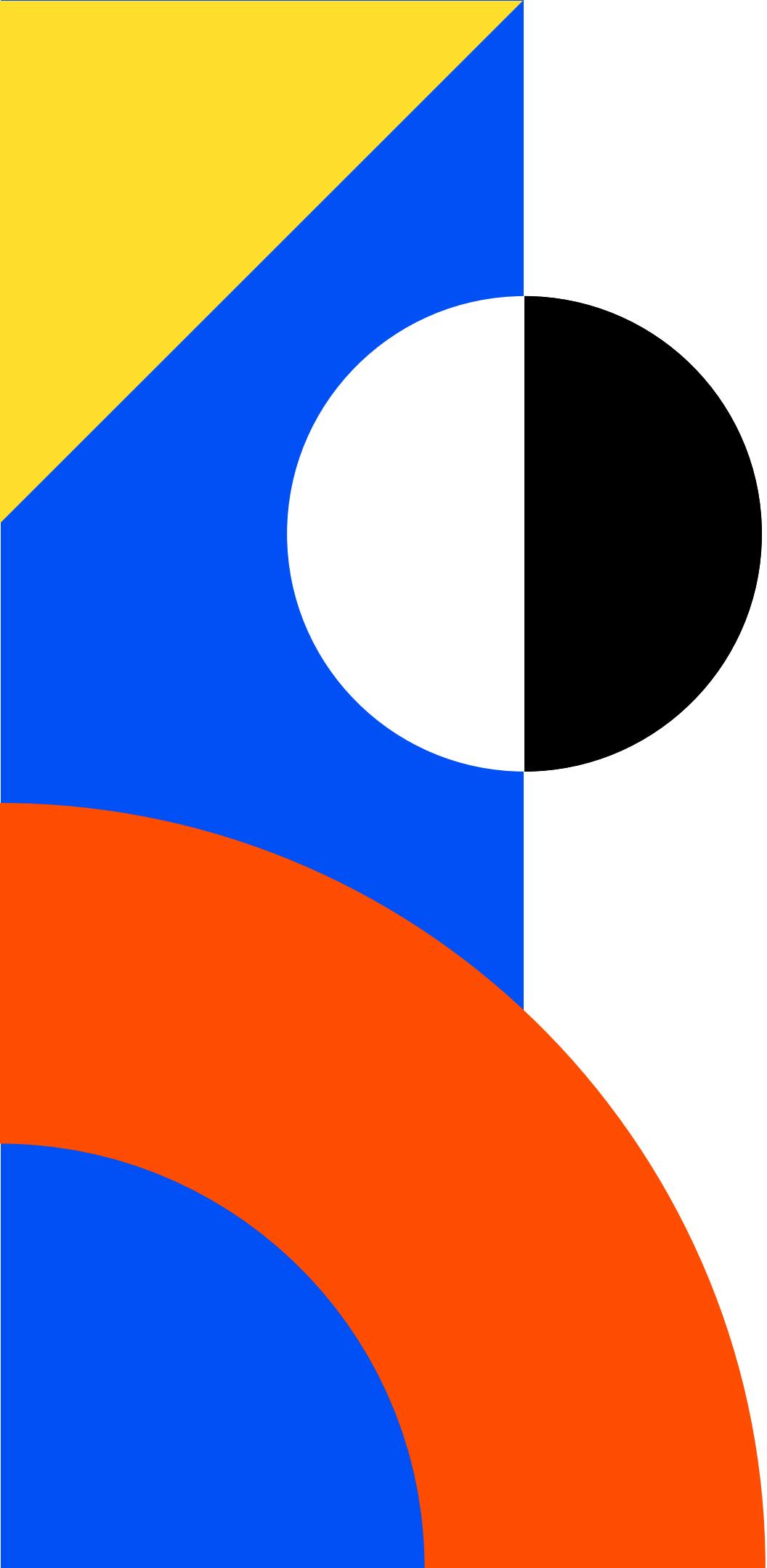
<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.nodemanager.env-whitelist</name>
    <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PREPEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_HOME,PATH,LANG,TZ,HADOOP_MAPRED_HOME</value>
  </property>
  <property>
    <name>yarn.nodemanager.disk-health-checker.max-disk-utilization-per-disk-percentage</name>
    <value>98.5</value>
  </property>
  <property>
    <name>yarn.nodemanager.resource.memory-mb</name>
    <value>32000</value>
  </property>
</configuration>
```

File yarn-site.xml digunakan untuk menentukan pengaturan yang relevan dengan YARN . Ini berisi konfigurasi untuk Node Manager, Resource Manager, Containers, dan Application Master. Kita dapat mengedit file yarn-site.xml dengan menjalankan command berikut :

```
sudo nano $HADOOP_HOME/etc/hadoop/yarn-site.xml
```

Lalu menambahkan konfigurasi pada gambar.

x



# Start Hadoop

# Format HDFS NameNode

```
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop/hadoop$ hdfs namenode -format
2023-06-21 12:31:48,528 INFO namenode.NameNode: STARTUP_MSG:
*****STARTUP_MSG: Starting NameNode
STARTUP_MSG: host = rayhanakbar-VirtualBox/127.0.1.1
STARTUP_MSG: args = [-format]
STARTUP_MSG: version = 3.3.2
STARTUP_MSG: classpath = /home/hadoop/hadoop/etc/hadoop:/home/hadoop/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.30.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jackson-annotations-2.13.0.jar:/home/hadoop/hadoop/share/hadoop/common/lib/httpclient-4.5.13.jar:/home/hadoop/hadoop/share/hadoop/common/lib/accessors-smart-2.4.7.jar:/home/hadoop/hadoop/share/hadoop/common/lib/kerb-admin-1.0.1.jar:/home/hadoop/hadoop/share/hadoop/common/lib/zookeeper-3.5.6.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jetty-server-9.4.43.v20210629.jar:/home/hadoop/hadoop/share/hadoop/common/lib/snappy-java-1.1.8.2.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jackson-jaxrs-1.9.13.jar:/home/hadoop/hadoop/share/hadoop/common/lib/commons-daemon-1.0.13.jar:/home/hadoop/hadoop/share/hadoop/common/lib/kerb-crypto-1.0.1.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jul-to-slf4j-1.7.30.jar:/home/hadoop/hadoop/share/hadoop/common/lib/token-provider-1.0.1.jar:/home/hadoop/hadoop/share/hadoop/common/lib/failureaccess-1.0.jar:/home/hadoop/hadoop/share/hadoop/common/lib/kerb-core-1.0.1.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jetty-servlet-9.4.43.v20210629.jar:/home/hadoop/hadoop/share/hadoop/common/lib/hecker-qual-2.5.2.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jsr305-3.0.2.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jersey-servlet-1.19.jar:/home/hadoop/hadoop/share/hadoop/common/lib/jackson-databind-2.13.0.jar:/home/hadoop/hadoop/share/hadoop/common/lib/commons-text
```

Penting untuk memformat NameNode sebelum memulai layanan Hadoop untuk pertama kalinya. Langkah ini penting untuk menginisialisasi metadata dan struktur data yang diperlukan oleh Hadoop Distributed File System (HDFS). Lakukan dengan command :

hdfs namenode -format

# Mulai Hadoop Cluster

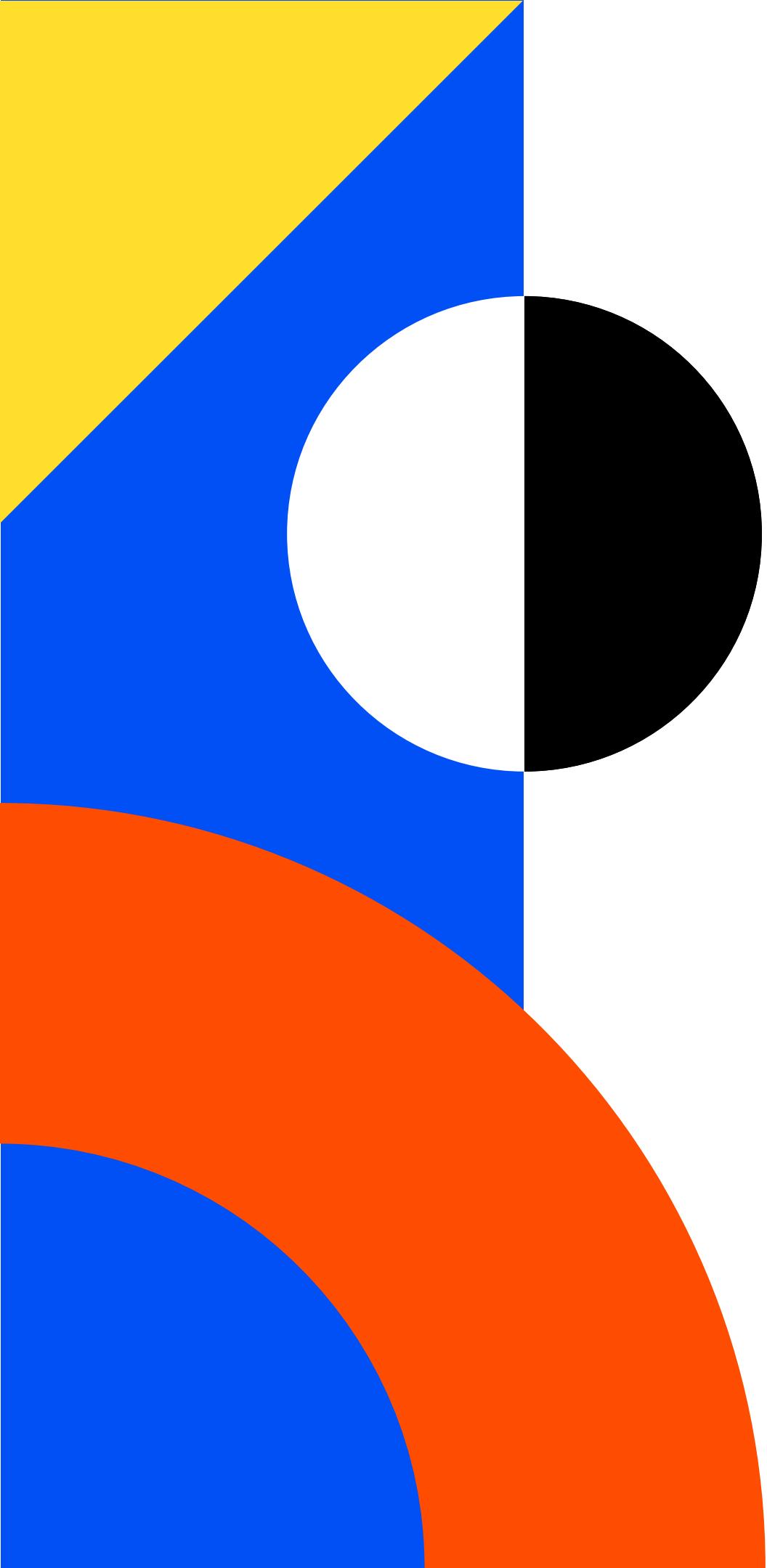
```
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop/hadoop$ start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [rayhanakbar-VirtualBox]
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop/hadoop$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop/hadoop$ jps
10482 SecondaryNameNode
10691 ResourceManager
10083 NameNode
10266 DataNode
11195 Jps
10845 NodeManager
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop/hadoop$
```

Mulai Hadoop cluster dengan menjalankan kedua command berikut :

```
$ start-dfs.sh
$ start-yarn.sh
```

Kita dapat memverifikasi semua komponen yang berjalan dengan command :

```
$jps
```



×

# Menjalankan Wordcount

# Kode Wordcount

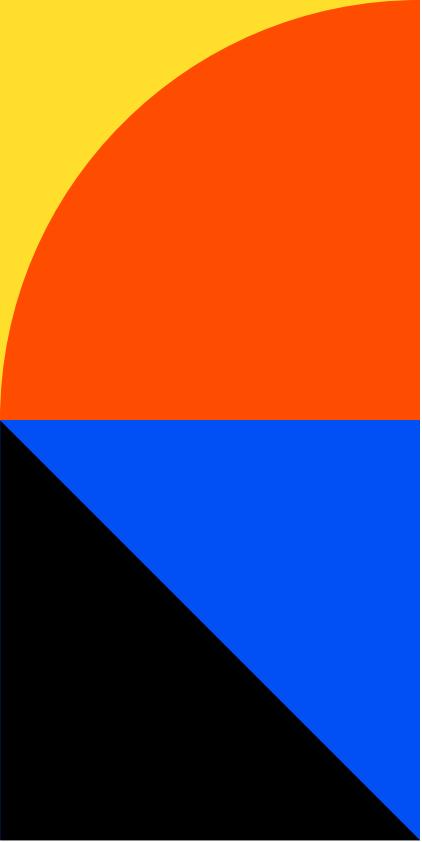
```
J WordCount.java /home/.../bin X J WordCount.java ./  
home > hadoop > hadoop > bin > J WordCount.java  
14  public class WordCount {  
15  
16    public static class TokenizerMapper  
17      extends Mapper<Object, Text, Text, IntWritable>{  
18  
19      private final static IntWritable one = new IntWritable(1);  
20      private Text word = new Text();  
21  
22      public void map(Object key, Text value, Context context  
23                      ) throws IOException, InterruptedException {  
24          StringTokenizer itr = new StringTokenizer(value.toString());  
25          while (itr.hasMoreTokens()) {  
26              word.set(itr.nextToken());  
27              context.write(word, one);  
28          }  
29      }  
30  }  
31  
32  public static class IntSumReducer  
33    extends Reducer<Text,IntWritable,Text,IntWritable> {  
34      private IntWritable result = new IntWritable();  
35  
36      public void reduce(Text key, Iterable<IntWritable> values,  
37                          Context context  
38                          ) throws IOException, InterruptedException {  
39          int sum = 0;  
40          for (IntWritable val : values) {  
41              sum += val.get();  
42          }  
43          result.set(sum);  
44          context.write(key, result);  
45      }  
46  }  
47  
48  public static void main(String[] args) throws Exception {  
49      Configuration conf = new Configuration();  
50      Job job = Job.getInstance(conf, "word count");  
51      job.setJarByClass(WordCount.class);  
52      job.setMapperClass(TokenizerMapper.class);  
53      job.setCombinerClass(IntSumReducer.class);  
54      job.setReducerClass(IntSumReducer.class);  
55      job.setOutputKeyClass(Text.class);  
56      job.setOutputValueClass(IntWritable.class);  
57      FileInputFormat.addInputPath(job, new Path(args[0]));  
58      FileOutputFormat.setOutputPath(job, new Path(args[1]));  
59      System.exit(job.waitForCompletion(true) ? 0 : 1);  
60  }  
61 }
```

# Menjalankan Wordcount dengan Hadoop

```
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop/hadoop$ hdfs dfs -mkdir /WordCountInput
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop/hadoop$ hdfs dfs -put /home/rayhanakbar/Downloads/Dataset/text10
text100MB.txt text10MB.txt
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop/hadoop$ hdfs dfs -put /home/rayhanakbar/Downloads/Dataset/text10MB.txt /WordCountInput
```

```
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop/hadoop/bin$ hadoop jar wc.jar WordCount /WordCountInput/text10MB.txt /text10MB_output
2023-06-21 12:47:00,300 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2023-06-21 12:47:00,535 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application in a Mapper or Reducer class.
2023-06-21 12:47:00,554 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/rayhanakbar/.staging/job_1687325899166_0001
2023-06-21 12:47:00,725 INFO input.FileInputFormat: Total input files to process : 1
2023-06-21 12:47:01,189 INFO mapreduce.JobSubmitter: number of splits:1
2023-06-21 12:47:01,297 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1687325899166_0001
2023-06-21 12:47:01,297 INFO mapreduce.JobSubmitter: Executing with tokens: []
2023-06-21 12:47:01,464 INFO conf.Configuration: resource-types.xml not found
2023-06-21 12:47:01,464 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2023-06-21 12:47:01,687 INFO impl.YarnClientImpl: Submitted application application_1687325899166_0001
2023-06-21 12:47:01,728 INFO mapreduce.Job: The url to track the job: http://rayhanakbar-VirtualBox:8088/proxy/application_1687325899166_0001/
2023-06-21 12:47:01,729 INFO mapreduce.Job: Running job: job_1687325899166_0001
2023-06-21 12:47:07,798 INFO mapreduce.Job: Job job_1687325899166_0001 running in uber mode : false
2023-06-21 12:47:07,799 INFO mapreduce.Job: map 0% reduce 0%
2023-06-21 12:47:12,866 INFO mapreduce.Job: map 100% reduce 0%
2023-06-21 12:47:16,888 INFO mapreduce.Job: map 100% reduce 100%
2023-06-21 12:47:17,902 INFO mapreduce.Job: Job job_1687325899166_0001 completed successfully
2023-06-21 12:47:17,963 INFO mapreduce.Job: Counters: 54
    File System Counters
        FILE: Number of bytes read=983264
        FILE: Number of bytes written=2516911
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=10000114
        HDFS: Number of bytes written=732767
        HDFS: Number of read operations=8
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=2
        HDFS: Number of bytes read erasure-coded=0
    Job Counters
        Launched map tasks=1
        Launched reduce tasks=1
        Data-local map tasks=1
        Total time spent by all maps in occupied slots (ms)=3381
        Total time spent by all reduces in occupied slots (ms)=1453
        Total time spent by all map tasks (ms)=3381
        Total time spent by all reduce tasks (ms)=1453
        Total vcore-milliseconds taken by all map tasks=3381
        Total vcore-milliseconds taken by all reduce tasks=1453
        Total megabyte-milliseconds taken by all map tasks=3462144
        Total megabyte-milliseconds taken by all reduce tasks=1487872
```

# Hasil Wordcount



```
Open ▾  Save  TestHadoop.txt  ~/Documents  -  ×
468 99      2
469 A       4260
470 A-horseback,   2
471 AARON    22
472 AARON,   6
473 AARON.  116
474 ABBESS.   32
475 ABBOT    2
476 ABBOT.   4
477 ABERGAVENNY,  2
478 ABERGAVENNY. 10
479 ABHORSON   8
480 ABHORSON,  2
481 ABHORSON. 26
482 ABOUT     4
483 ABRAM,   2
484 ABRAM.  10
485 ACHILLES  2
486 ACHILLES. 156
487 ACT      644
488 ADAM,   2
489 ADAM.  20
490 ADO      4
491 ADONIS   2
492 ADRIAN,   2
493 ADRIAN.  18
494 ADRIANA,  2
495 ADRIANA. 158
496 ADRIANO   6
497 AEDILE.  18
498 AEDILES. 2
499 AEMILIUS  6
500 AEMILIUS, 4
```

Plain Text ▾ Tab Width: 8 ▾ Ln 1, Col 1 ▾ INS

# WORD COUNT

-With or without Hadoop comparison

---

# System Specification

- ◆ **OS**

- Ubuntu 20.04 (Virtual Machine)

- ◆ **CPU allocated**

- 6

- ◆ **Base Memory allocated**

- 8 GB

# Program with hadoop

```
J WordCount.java /home/.../bin X J WordCount.java ./  
home > hadoop > hadoop > bin > J WordCount.java  
14  public class WordCount {  
15      public static class TokenizerMapper  
16          extends Mapper<Object, Text, Text, IntWritable>{  
17              private final static IntWritable one = new IntWritable(1);  
18              private Text word = new Text();  
19              public void map(Object key, Text value, Context context  
20                  throws IOException, InterruptedException {  
21                  StringTokenizer itr = new StringTokenizer(value.toString());  
22                  while (itr.hasMoreTokens()) {  
23                      word.set(itr.nextToken());  
24                      context.write(word, one);  
25                  }  
26              }  
27          }  
28      }  
29      public static class IntSumReducer  
30          extends Reducer<Text,IntWritable,Text,IntWritable> {  
31          private IntWritable result = new IntWritable();  
32          public void reduce(Text key, Iterable<IntWritable> values,  
33              Context context  
34              throws IOException, InterruptedException {  
35              int sum = 0;  
36              for (IntWritable val : values) {  
37                  sum += val.get();  
38              }  
39              result.set(sum);  
40              context.write(key, result);  
41          }  
42      }  
43      public static void main(String[] args) throws Exception {  
44          Configuration conf = new Configuration();  
45          Job job = Job.getInstance(conf, "word count");  
46          job.setJarByClass(WordCount.class);  
47          job.setMapperClass(TokenizerMapper.class);  
48          job.setCombinerClass(IntSumReducer.class);  
49          job.setReducerClass(IntSumReducer.class);  
50          job.setOutputKeyClass(Text.class);  
51          job.setOutputValueClass(IntWritable.class);  
52          FileInputFormat.addInputPath(job, new Path(args[0]));  
53          FileOutputFormat.setOutputPath(job, new Path(args[1]));  
54          System.exit(job.waitForCompletion(true) ? 0 : 1);  
55      }  
56  }
```

# Program without hadoop

```
J WordCount.java  
6  public class WordCount {  
7      public static void main(String[] args) {  
8          String inputFile = "text500MB.txt"; // Path to the input file  
9          try {  
10              Map<String, Integer> wordCountMap = countWords(inputFile);  
11              // Print the word count results  
12              for (Map.Entry<String, Integer> entry : wordCountMap.entrySet()) {  
13                  System.out.println(entry.getKey() + ": " + entry.getValue());  
14              }  
15          } catch (IOException e) {  
16              System.err.println("Error reading the input file: " + e.getMessage());  
17          }  
18      }  
19  }  
20  public static Map<String, Integer> countWords(String inputFile) throws IOException {  
21      Map<String, Integer> wordCountMap = new HashMap<>();  
22      BufferedReader reader = new BufferedReader(new FileReader(inputFile));  
23      String line;  
24      // Read each line from the input file  
25      while ((line = reader.readLine()) != null) {  
26          // Split the line into words using whitespace as the delimiter  
27          String[] words = line.split("\\s+");  
28          // Count the occurrences of each word  
29          for (String word : words) {  
30              if (wordCountMap.containsKey(word)) {  
31                  // Increment the count if the word is already present  
32                  wordCountMap.put(word, wordCountMap.get(word) + 1);  
33              } else {  
34                  // Initialize the count to 1 for new words  
35                  wordCountMap.put(word, 1);  
36              }  
37          }  
38      }  
39      reader.close();  
40      return wordCountMap;  
41  }  
42 }  
43 }
```

# Test Sample

Downloads	Dataset ▾	Search	Grid View	Sort	Filter
Name				Size	Actions
	 <a href="#">text1MB.txt</a>			1,3 MB	
	 <a href="#">text10MB.txt</a>			10,0 MB	
	 <a href="#">text100MB.txt</a>			104,9 MB	
	 <a href="#">text200MB.txt</a>			209,7 MB	
	 <a href="#">text500MB.txt</a>			524,3 MB	
	 <a href="#">text1GB.txt</a>			1,1 GB	

# Hadoop Setup

```
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop$ hdfs dfs -mkdir /WordCountInput
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop$ hdfs dfs -put /home/rayhanakbar/Downloads/Dataset/text1MB.txt /WordCountInput
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop$ hdfs dfs -put /home/rayhanakbar/Downloads/Dataset/text10MB.txt /WordCountInput
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop$ hdfs dfs -put /home/rayhanakbar/Downloads/Dataset/text100MB.txt /WordCountInput
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop$ hdfs dfs -put /home/rayhanakbar/Downloads/Dataset/text200MB.txt /WordCountInput
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop$ hdfs dfs -put /home/rayhanakbar/Downloads/Dataset/text500MB.txt /WordCountInput
rayhanakbar@rayhanakbar-VirtualBox:/home/hadoop$ hdfs dfs -put /home/rayhanakbar/Downloads/Dataset/text1GB.txt /WordCountInput
```

## Browse Directory

/WordCountInput										Go!				
Show 25 entries										Search:				
	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name						
<input type="checkbox"/>	-rw-r--r--	rayhanakbar	supergroup	100 MB	Jun 09 16:38	1	128 MB	text100MB.txt						
<input type="checkbox"/>	-rw-r--r--	rayhanakbar	supergroup	9.54 MB	Jun 09 16:37	1	128 MB	text10MB.txt						
<input type="checkbox"/>	-rw-r--r--	rayhanakbar	supergroup	1 GB	Jun 09 16:38	1	128 MB	text1GB.txt						
<input type="checkbox"/>	-rw-r--r--	rayhanakbar	supergroup	1.2 MB	Jun 09 16:37	1	128 MB	text1MB.txt						
<input type="checkbox"/>	-rw-r--r--	rayhanakbar	supergroup	200 MB	Jun 09 16:38	1	128 MB	text200MB.txt						
<input type="checkbox"/>	-rw-r--r--	rayhanakbar	supergroup	500 MB	Jun 09 16:38	1	128 MB	text500MB.txt						
Showing 1 to 6 of 6 entries										Previous	1	Next		

# File 1MB

## With Hadoop

Command:

```
hadoop jar wc.jar WordCount  
/WordCountInput/text1MB.txt /text1MB_output
```

User:	<u>rayhanakbar</u>
Name:	word count
Application Type:	MAPREDUCE
Application Tags:	
Application Priority:	0 (Higher Integer value indicates higher priority)
YarnApplicationState:	FINISHED
Queue:	<u>default</u>
FinalStatus Reported by AM:	SUCCEEDED
Started:	Fri Jun 09 16:41:56 +0700 2023
Launched:	Fri Jun 09 16:41:57 +0700 2023
Finished:	Fri Jun 09 16:42:08 +0700 2023
Elapsed:	11sec

## Java (Without Hadoop)

Command:

```
time java WordCount
```

```
rayhanakbar@rayhanakbar-VirtualBox:~/Downloads/Dataset$ time java WordCount  
: 3438  
spilling: 2  
passionateness: 1  
shrink?: 1  
animosity: 1  
pretend: 7  
Gros: 1  
huzza!: 1  
him,: 1  
him,": 3  
mankind-devilish: 1  
blast=: 1  
capacity.: 2  
coals: 2  
halyards: 1  
  
real    0m0,334s  
user    0m0,422s  
sys     0m0,197s
```

# File 10MB

## With Hadoop

Command:

```
hadoop jar wc.jar WordCount  
/WordCountInput/text10MB.txt  
/text10MB_output
```

User:	rayhanakbar
Name:	word count
Application Type:	MAPREDUCE
Application Tags:	
Application Priority:	0 (Higher Integer value indicates higher priority)
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Fri Jun 09 17:43:25 +0700 2023
Launched:	Fri Jun 09 17:43:26 +0700 2023
Finished:	Fri Jun 09 17:43:37 +0700 2023
Elapsed:	12sec

## Java (Without Hadoop)

Command:

```
time java WordCount
```

```
rayhanakbar@rayhanakbar-VirtualBox:~/Downloads/Dataset$ time java WordCount  
: 110633  
Reserve: 6  
frowning: 24  
favour!: 4  
spilling: 2  
DUKE.: 548  
Tisick,: 2  
DUKE,: 14  
abrupt: 2  
favour-: 2  
favour.: 36  
TAURUS.: 2  
clapper;: 2  
TAURUS,: 2  
favour,: 52  
sithence,: 2  
Grow: 12  
real 0m0,836s  
user 0m0,996s  
sys 0m0,308s
```

# File 100MB

## With Hadoop

Command:

```
hadoop jar wc.jar WordCount  
/WordCountInput/text100MB.txt  
/text100MB_output
```

```
User: rayhanakbar  
Name: word count  
Application Type: MAPREDUCE  
Application Tags:  
Application Priority: 0 (Higher Integer value indicates higher priority)  
YarnApplicationState: FINISHED  
Queue: default  
FinalStatus Reported by AM: SUCCEEDED  
Started: Fri Jun 09 17:44:58 +0700 2023  
Launched: Fri Jun 09 17:44:59 +0700 2023  
Finished: Fri Jun 09 17:45:24 +0700 2023  
Elapsed: 25sec
```

## Java (Without Hadoop)

Command:

```
time java WordCount
```

```
rayhanakbar@rayhanakbar-VirtualBox:~/Downloads/Dataset$ javac WordCount.java  
rayhanakbar@rayhanakbar-VirtualBox:~/Downloads/Dataset$ time java WordCount
```

```
real    0m6,116s  
user    0m5,194s  
sys     0m1,674s
```

# File 200MB

## With Hadoop

Command:

```
hadoop jar wc.jar WordCount  
/WordCountInput/text200MB.txt  
/text200MB_output
```

```
User: rayhanakbar  
Name: word count  
Application Type: MAPREDUCE  
Application Tags:  
Application Priority: 0 (Higher Integer value indicates higher priority)  
YarnApplicationState: FINISHED  
Queue: default  
FinalStatus Reported by AM: SUCCEEDED  
Started: Fri Jun 09 17:47:16 +0700 2023  
Launched: Fri Jun 09 17:47:17 +0700 2023  
Finished: Fri Jun 09 17:47:46 +0700 2023  
Elapsed: 30sec
```

## Java (Without Hadoop)

Command:

```
time java WordCount
```

```
rayhanakbar@rayhanakbar-VirtualBox:~/Downloads/Dataset$ javac WordCount.java  
rayhanakbar@rayhanakbar-VirtualBox:~/Downloads/Dataset$ time java WordCount
```

```
real    0m10,416s  
user    0m8,669s  
sys     0m2,257s
```

# File 500MB

## With Hadoop

Command:

```
time hadoop jar wc.jar WordCount  
/WordCountInput/text500MB.txt  
/text500MB_output
```

User:	rayhanakbar
Name:	word count
Application Type:	MAPREDUCE
Application Tags:	
Application Priority:	0 (Higher Integer value indicates higher priority)
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Fri Jun 09 19:05:44 +0700 2023
Launched:	Fri Jun 09 19:05:45 +0700 2023
Finished:	Fri Jun 09 19:06:19 +0700 2023
Elapsed:	34sec

## Java (Without Hadoop)

Command:

```
time java WordCount
```

```
rayhanakbar@rayhanakbar-VirtualBox:~/Downloads/Dataset$ javac WordCount.java  
rayhanakbar@rayhanakbar-VirtualBox:~/Downloads/Dataset$ time java WordCount
```

```
real    0m22,642s  
user    0m21,818s  
sys     0m2,092s
```

# File 1GB

## With Hadoop

Command:

```
hadoop jar wc.jar WordCount  
/WordCountInput/text1GB.txt  
/text1GB_output
```

User:	rayhanakbar
Name:	word count
Application Type:	MAPREDUCE
Application Tags:	
Application Priority:	0 (Higher Integer value indicates higher priority)
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Fri Jun 09 19:27:33 +0700 2023
Launched:	Fri Jun 09 19:27:34 +0700 2023
Finished:	Fri Jun 09 19:28:27 +0700 2023
Elapsed:	53sec

## Java (Without Hadoop)

Command:

```
time java WordCount
```

```
rayhanakbar@rayhanakbar-VirtualBox:~/Downloads/Dataset$ javac WordCount.java  
rayhanakbar@rayhanakbar-VirtualBox:~/Downloads/Dataset$ time java WordCount
```

```
real      0m57,493s  
user      0m3,146s  
sys       0m0,267s
```

# Graph

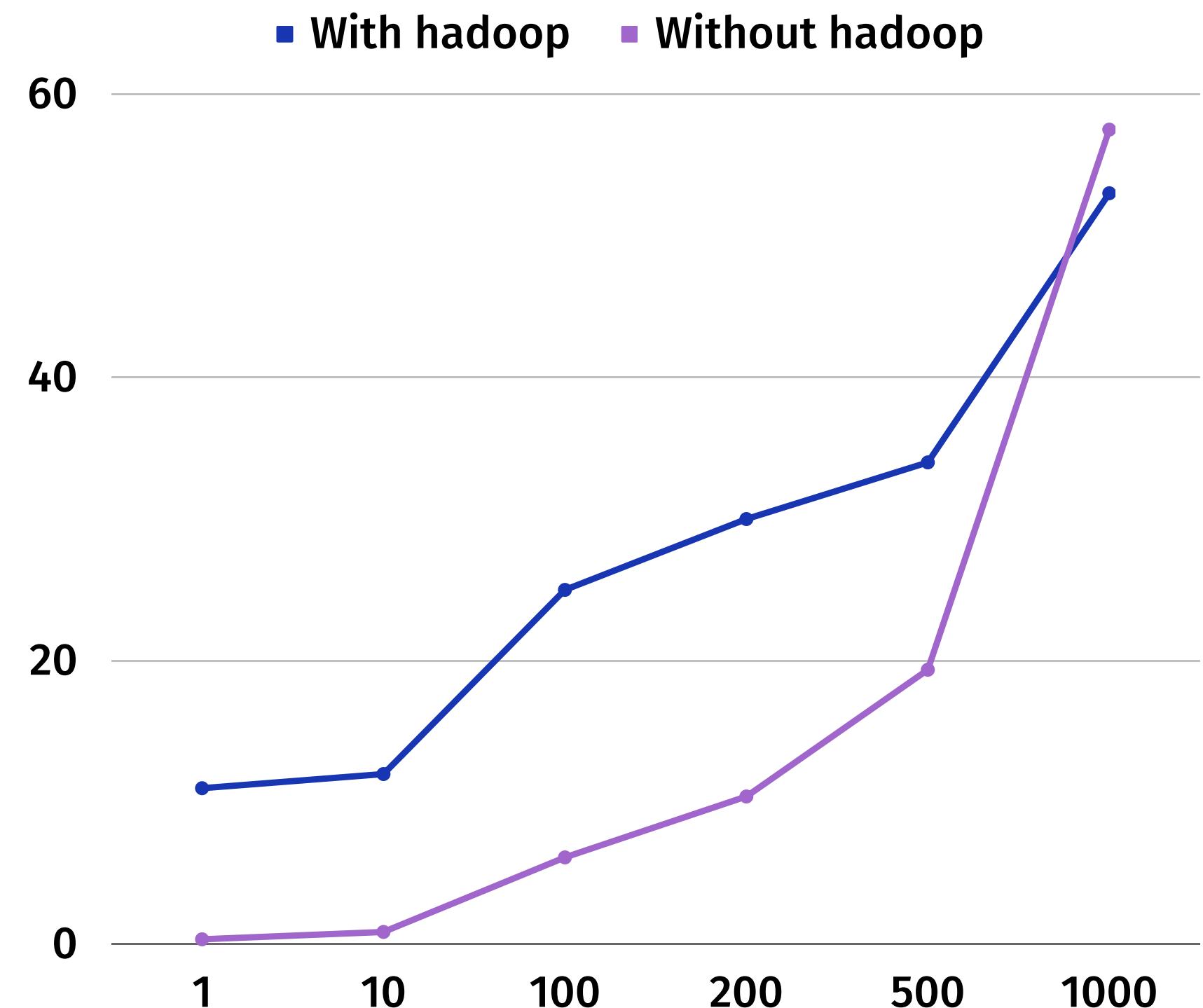
Analisis:

Pada ukuran file yang masih kecil mulai dari 1-500MB, Word Count tanpa hadoop terlihat lebih cepat dibanding word count dengan hadoop.

Namun berdasarkan grafik yang dihasilkan, dapat dilihat bahwa Word Count tanpa hadoop memiliki peningkatan waktu yang lebih signifikan dibandingkan Word Count dengan hadoop.

Kesimpulan:

Pada ukuran file yang besar, word count dengan hadoop lebih unggul dibandingkan word count tanpa hadoop



RK

# Thank you!

