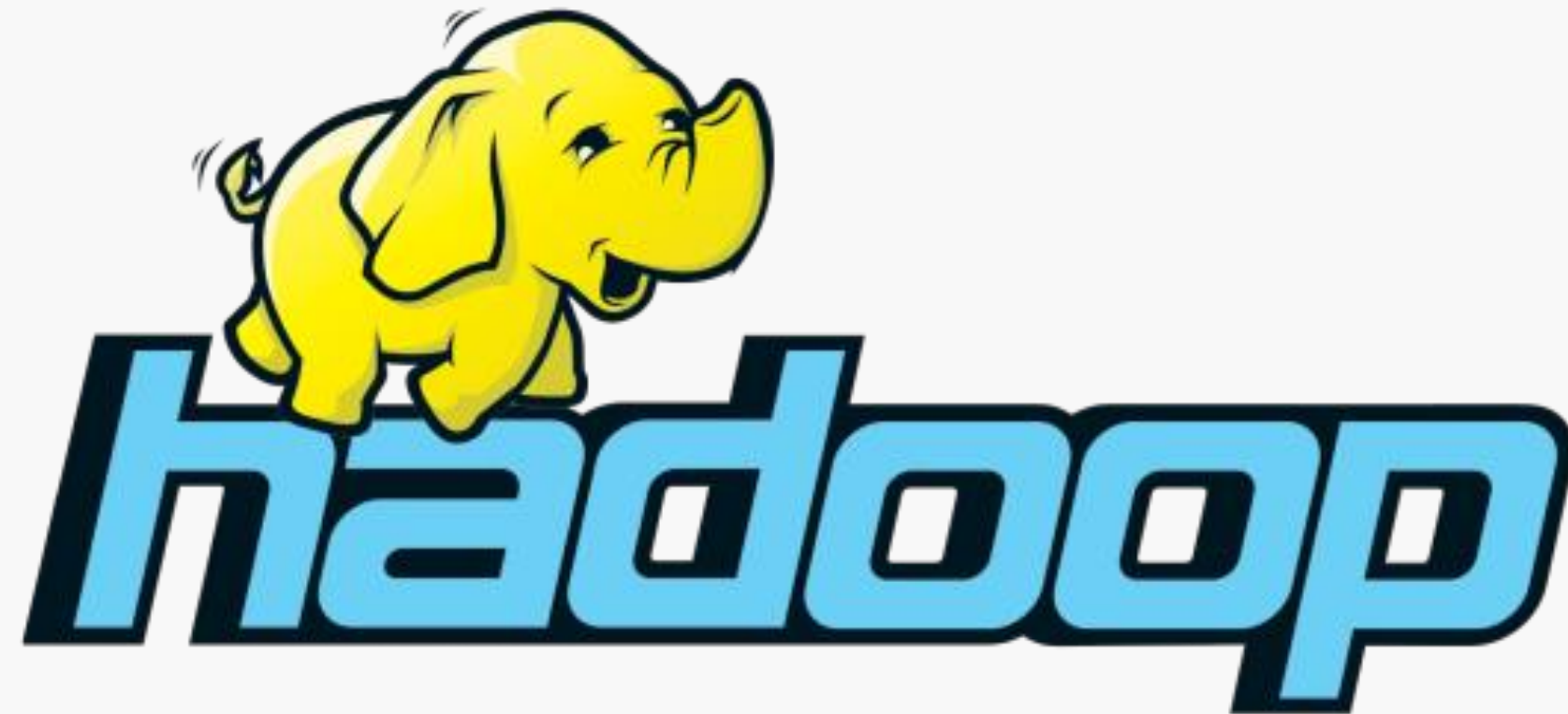




DỮ LIỆU LỚN VÀ ỨNG DỤNG

# HDFS - VÍ DỤ ĐỌC GHI FILE SỬ DỤNG JAVA API





## Giới thiệu

Hướng dẫn này cung cấp kiến thức về cách tạo, đọc, ghi tệp trong HDFS (Hệ thống tệp phân tán Hadoop) bằng cách sử dụng API Java của Apache Hadoop.

Yêu cầu:

- Đã cài đặt Hadoop
- Đã cài đặt Eclipse IDE

# 1. Khởi động máy ảo

1. Kiểm tra trạng thái của Hadoop trên trình duyệt qua địa chỉ:

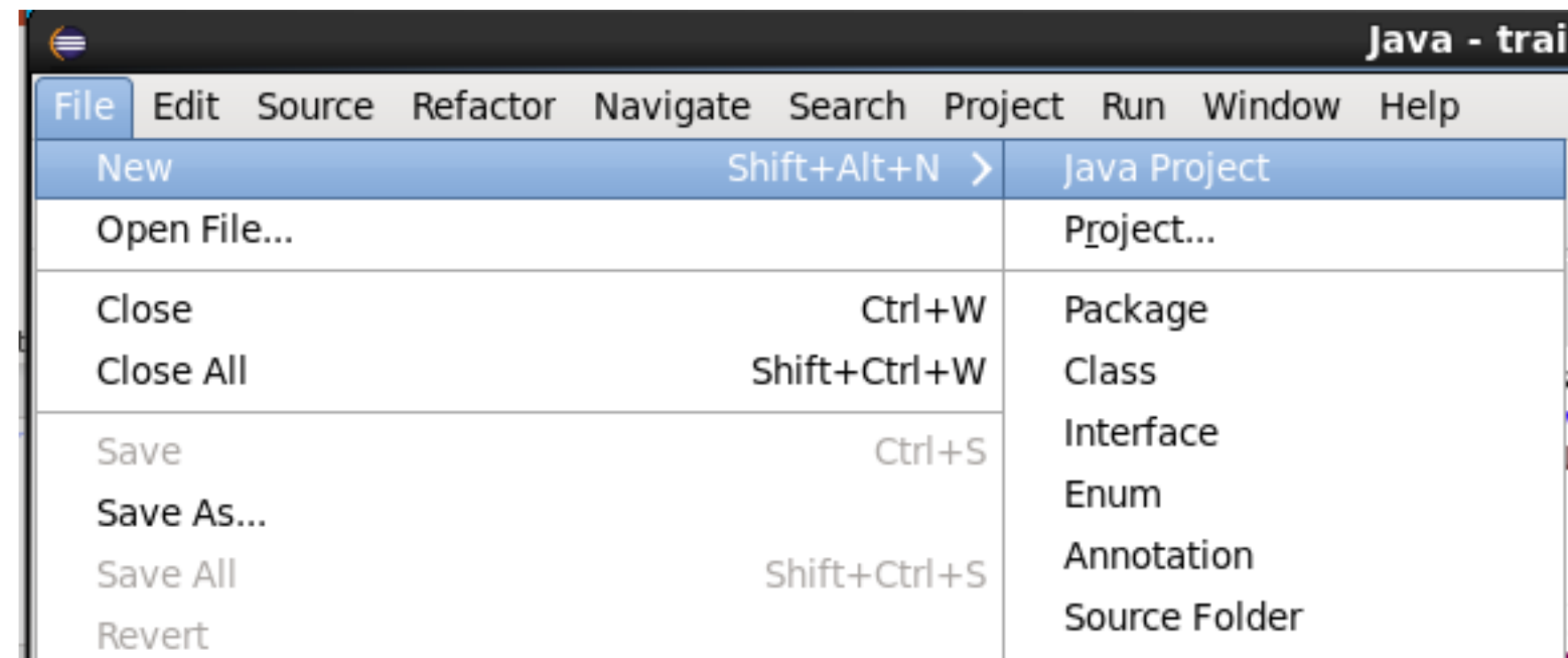
<http://quickstart.cloudera:50070/dfshealth.html#tab-overview>

Non Heap Memory used 41.69 MB of 42.25 MB Committed Non Heap Memory. Max Non Heap Memory is 130 MB.	
Configured Capacity:	54.51 GB
DFS Used:	832.22 MB (1.49%)
Non DFS Used:	8.04 GB
DFS Remaining:	42.63 GB (78.21%)
Block Pool Used:	832.22 MB (1.49%)
DataNodes usages% (Min/Median/Max/stdDev):	1.49% / 1.49% / 1.49% / 0.00%
Live Nodes	1 (Decommissioned: 0, In Maintenance: 0)
Dead Nodes	0 (Decommissioned: 0, In Maintenance: 0)
Decommissioning Nodes	0
Entering Maintenance Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	2
Number of Blocks Pending Deletion	0
Block Deletion Start Time	Sun Apr 16 06:32:30 -0700 2023



## 2. Tạo project trên Eclipse

1. Trên Eclipse tạo một Project bằng cách File -> New -> Java Project

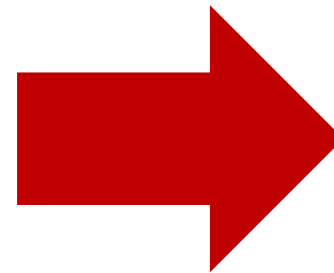




## 2. Tạo project trên Eclipse

2. Tại bước tiếp theo, nhập tên cho Project sau đó nhấn Finish để hoàn thành việc tạo Project

**Yêu cầu đặt tên project: HDFS\_MSV (VD: HDFS\_1921050102)**



**New Java Project**

Create a Java project in the workspace or in an external location.

Project name:

☒ Use default location

Location:  [Browse...](#)

JRE

☒ Use an execution environment JRE:

☐ Use a project specific JRE:

☐ Use default JRE (currently 'jdk1.7.0\_67-cloudera') [Configure JREs...](#)

Project layout

☐ Use project folder as root for sources and class files

☒ Create separate folders for sources and class files [Configure default...](#)

Working sets

☐ Add project to working sets

Working sets:  [Select...](#)

[?](#) [< Back](#) [Next >](#) [Cancel](#) [Finish](#)



## 3. Đọc ghi file

### 3.1 Tạo thư mục mới trong HDFS

Lớp Hadoop FileSystem cung cấp tất cả các chức năng liên quan đến các thao tác với tệp hoặc thư mục như tạo hay xóa tệp, v.v. Phương thức mkdirs được sử dụng để tạo thư mục trong HDFS:

```
public static void createDirectory() throws IOException {  
    Configuration configuration = new Configuration();  
    configuration.set("fs.defaultFS", "hdfs://10.0.2.15");  
    FileSystem fileSystem = FileSystem.get(configuration);  
    String directoryName = "BigdataHUMG/HDFSexample";  
    Path path = new Path(directoryName);  
    fileSystem.mkdirs(path);  
}
```

**Yêu cầu: thay đổi tên thư mục bằng MSV (VD: BigdataHUMG/HDFS\_1921050102)**



## 3. Đọc ghi file

### 3.1 Tạo thư mục mới trong HDFS

Trong phương thức: `configuration.set("fs.defaultFS", "hdfs://10.0.2.15")` tCó tham số 10.0.2.15 là địa chỉ IP của máy ảo. Để lấy địa chỉ máy ảo chúng ta mở cửa sổ **terminal** và gõ câu lệnh **ifconfig**:

```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
[cloudera@quickstart ~]$ ipconfig  
bash: ipconfig: command not found  
[cloudera@quickstart ~]$ ifconfig  
eth0      Link encap:Ethernet  HWaddr 08:00:27:B2:38:58  
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0  
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1  
          RX packets:11024 errors:0 dropped:0 overruns:0 frame:0  
          TX packets:6506 errors:0 dropped:0 overruns:0 carrier:0  
          collisions:0 txqueuelen:1000  
          RX bytes:12056844 (11.4 MiB)  TX bytes:611072 (596.7 KiB)  
  
lo        Link encap:Local Loopback  
          inet addr:127.0.0.1  Mask:255.0.0.0  
          UP LOOPBACK RUNNING  MTU:65536  Metric:1  
          RX packets:64733065 errors:0 dropped:0 overruns:0 frame:0  
          TX packets:64733065 errors:0 dropped:0 overruns:0 carrier:0  
          collisions:0 txqueuelen:0  
          RX bytes:6168364650 (5.7 GiB)  TX bytes:6168364650 (5.7 GiB)
```

**ipconfig**



## 3. Đọc ghi file

### 3.1 Tạo thư mục mới trong HDFS

Tiếp theo khởi tạo hàm **main()** và gọi phương thức **createDirectory()** vừa khởi tạo và chạy chương trình:

```
public static void main(String[] args) {  
    try {  
        createDirectory();  
    } catch (IOException e) {  
        // TODO Auto-generated catch block  
        e.printStackTrace();  
    }  
}
```

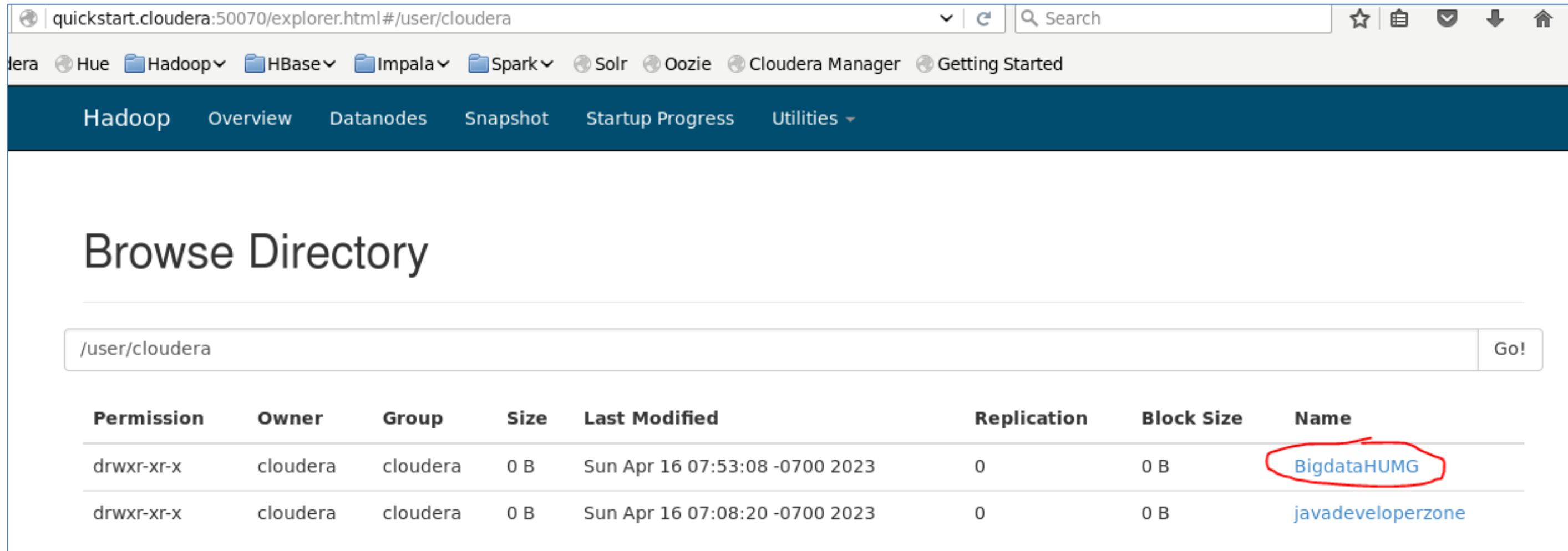




## 3. Đọc ghi file

### 3.1 Tạo thư mục mới trong HDFS

Cuối cùng kiểm tra kết quả thư mục được tạo trên HDFS:



quickstart.cloudera:50070/explorer.html#/user/cloudera

Hadoop Overview Datanodes Snapshot Startup Progress Utilities

### Browse Directory

/user/cloudera Go!

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	cloudera	cloudera	0 B	Sun Apr 16 07:53:08 -0700 2023	0	0 B	BigdataHUMG
drwxr-xr-x	cloudera	cloudera	0 B	Sun Apr 16 07:08:20 -0700 2023	0	0 B	javadeveloperzone



## 3. Đọc ghi file

### 3.2 Ghi file trong HDFS

Tương tự, đầu tiên chúng ta sẽ tạo một phương thức mới đặt tên **writeFileToHDFS()** để thực hiện ghi file trên HDFS, phương thức được triển khai như sau:

```
public static void writeFileToHDFS() throws IOException {
    Configuration configuration = new Configuration();
    configuration.set("fs.defaultFS", "hdfs://10.0.2.15");
    FileSystem fileSystem = FileSystem.get(configuration);

    String fileName = "read_write_hdfs_example.txt";
    Path hdfsWritePath = new Path("/user/BigdataHUMG/HDFSexample/" + fileName);
    FSDataOutputStream fsDataOutputStream = fileSystem.create(hdfsWritePath, true);
    BufferedWriter bufferedWriter = new BufferedWriter(new OutputStreamWriter(fsDataOutputStream, StandardCharsets.UTF_8));
    bufferedWriter.write("Java API to write data in HDFS"); ➡ Y/câu: Thay đổi giá trị bằng MSV + Họ tên (VD: 1921050102 Nguyễn văn A)
    bufferedWriter.newLine();
    bufferedWriter.close();
    fileSystem.close();
}
```



## 3. Đọc ghi file

nguyenvanthang@humg

### 3.2 Ghi file trong HDFS

**Giải thích:** Lớp FSDataOutputStream dùng để ghi dữ liệu vào tệp HDFS. Nó cung cấp các phương thức khác nhau như writeUTF, writeInt, WriteChar, v.v. Ở đây chúng ta sử dụng lớp BufferedWrite để wrapped lớp FSDataOutputStream.

- Dòng 2,3,4 thực hiện khởi tạo kết nối đến HDFS
- Dòng 6,7,8,9 khởi tạo đường dẫn và file
- Dòng 10,11 thực hiện ghi nội dung "Java API to write data in HDFS" vào file
- Dòng 12,13 đóng luồng đã khởi tạo



## 3. Đọc ghi file

### 3.2 Ghi file trong HDFS

Bước tiếp theo, tại hàm main() chúng ta gọi phương thức vừa tạo và chạy chương trình

```
public static void main(String[] args) {  
    try {  
        writeToFileToHDFS();  
    } catch (IOException e) {  
        // TODO Auto-generated catch block  
        e.printStackTrace();  
    }  
}
```

Sau đó truy cập trình duyệt để kiểm tra kết quả:

Browse Directory							
<input type="text" value="/user/BigdataHUMG/HDFSexample"/>							<input type="button" value="Go!"/>
Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	cloudera	supergroup	31 B	Sun Apr 16 08:01:33 -0700 2023	3	128 MB	<a href="#">read_write_hdfs_example.txt</a>



## 3. Đọc ghi file

### 3.2 Ghi file trong HDFS

**Bổ sung:** chúng ta có thể sử dụng phương thức **append()** của lớp `FileSystem` để nối dữ liệu vào một tệp hiện có. Cách triển khai sẽ gần giống với phương thức **writeFileToHDFS()** chỉ có dòng số 8 chúng ta sẽ sử dụng phương thức **append()** thay vì sử dụng phương thức **create()**.



## 3. Đọc ghi file

### 3.3 Đọc file trong HDFS

Lớp `FSDataInputStream` cung cấp các phương thức để đọc tệp từ HDFS. Chúng ta sẽ tạo một phương thức để đọc file như sau:

```
public static void readFileFromHDFS() throws IOException {
    Configuration configuration = new Configuration();
    configuration.set("fs.defaultFS", "hdfs://10.0.2.15");
    FileSystem fileSystem = FileSystem.get(configuration);
    //Create a path
    String fileName = "read_write_hdfs_example.txt";
    Path hdfsReadPath = new Path("/user/BigdataHUMG/HDFSexample/" + fileName);

    FSDataInputStream inputStream = fileSystem.open(hdfsReadPath);

    String out= IOUtils.toString(inputStream, "UTF-8");
    System.out.println(out);

    inputStream.close();
    fileSystem.close();
}
```



## 3. Đọc ghi file

### 3.3 Đọc file trong HDFS

Giải thích:

- Dòng 2,3,4 thực hiện khởi tạo kết nối đến HDFS
- Dòng 5,6 khởi tạo đường dẫn và file
- Dòng 7 khởi tạo luồng đọc
- Dòng 8 lấy nội dung của file và gán vào biến **out**
- Dòng 9 in kết quả đọc được ra màn hình console.
- Dòng 10,11 đóng luồng đọc file.



## 3. Đọc ghi file

### 3.3 Đọc file trong HDFS

Tiếp theo, trong hàm Main chúng ta gọi đến phương thức đọc và chạy chương trình

```
public static void main(String[] args) {  
    try {  
        readFileFromHDFS();  
    } catch (IOException e) {  
        // TODO Auto-generated catch block  
        e.printStackTrace();  
    }  
}
```

Kết quả hiển thị trên eclipse như sau:

The screenshot shows the Eclipse IDE's console window. At the top, there are tabs for 'Problems', 'Javadoc', 'Declaration', and 'Console'. The 'Console' tab is active. The output text in the console is as follows:

```
<terminated> Main [Java Application] /usr/java/jdk1.7.0_67-cloudera/bin/java (Apr 16, 2023, 8:12:58 AM)  
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".  
SLF4J: Defaulting to no-operation (NOP) logger implementation  
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.  
23/04/16 08:13:01 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform...  
Java API to write data in HDFS
```

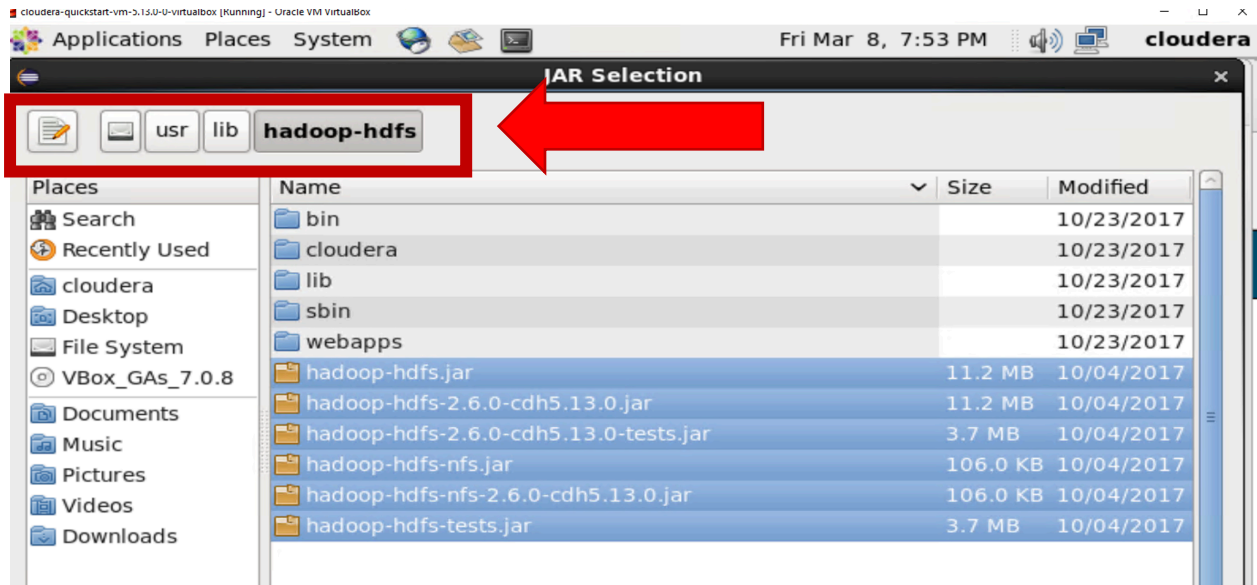


## Import các packages cần thiết sau:

Các bước:

- Chuột phải vào project trên Eclipse và chọn Build PATH
- Chọn Add External Archives

Import lần 1 tại đường dẫn: usr/lib/hadoop-hdfs (Chọn tất cả các file .jar như hình bên dưới)



Import lần 2,3: thực hiện tương tự tại các đường dẫn:

- usr/lib/hadoop
- usr/lib/Hadoop/lib