Executing Hadoop Jar Files in opt/hadoop Docker Environment

This guide provides step-by-step instructions on how to compile, package, and execute Hadoop jobs inside the **opt/hadoop Docker environment**.

---

1. Setting Up the Environment

Ensure that your **Hadoop environment** is properly configured inside Docker. If not, start the Hadoop cluster inside Docker.

Start Hadoop inside the Docker container (if not already running)

```
docker exec -it hadoop-container-name /bin/bash
```

Inside the container, navigate to your project directory:

```
cd /opt/hadoop/proj1/
```

2. Compiling the Java File

The Hadoop job must be compiled before packaging it into a JAR file.

Steps to compile

Create a directory for compiled Java class files:

```
mkdir -p /home/ubuntu/Proj1/outPart3
```

Compile the Java file using Hadoop's classpath:

```
javac -classpath "$(hadoop classpath)" -d /home/ubuntu/Proj1/outPart3
/home/ubuntu/Proj1/Part3Seven.java
```

(replace the name of java file with the file name we want to create the classes for JAR files)

3. Creating the Manifest File

A **manifest file** is needed to specify the main class inside the JAR file.

Steps to Create the Manifest File

Create a new file named `manifestPart3.txt`:

```
touch /home/ubuntu/Proj1/manifestPart3.txt
```

Edit the file and add the following content:

```
Main-Class: Part3Seven
```

(Replace `Part3Seven` with the actual main class name if different.)

4. Creating the JAR File

Now, we need to package the compiled Java files into a **Hadoop JAR**.

Command to Create the JAR

```
jar cvfm part3ten.jar /home/ubuntu/Proj1/manifestPart3.txt -C /home/ubuntu/Proj1/outPart3/ .
```

This will create `part3ten.jar` inside the `/home/ubuntu/Proj1/` directory.

5. Running the Hadoop Job

Once the JAR file is created, you can execute it using Hadoop.

Command to Run the JAR

```
hadoop jar /opt/hadoop/proj1/jars/part3ten.jar Part3Ten /accesslogInput/access_log
/output3/oten
```

(Just change the {jar filename} and the {class name} and the {output folder }to generate a new result output)

Another example:

```
hadoop jar /opt/hadoop/proj1/jars/part3one.jar Part3One /opt/hadoop/proj1/access_log
/opt/hadoop/proj1/output3/outputPart3one
```

6. Viewing the Output

To check if the output was generated successfully, list the HDFS output directory:

```
hdfs dfs -ls /outputPart3Two
```

To view the actual content of the output file:

```
hdfs dfs -cat /outputPart3Nine/part-r-00000
```

(Replace `/outputPart3Nine/`with the actual output directory and filename as provided in the screenshot in the report file.)

7. Creating Files & Folders in HDFS

If you need to manually add input files to HDFS before running the Hadoop job:

Create an Input Folder in HDFS

```
hadoop fs -mkdir /input
```

Upload a Local File to HDFS

```
hadoop fs -put /path/to/local/inputfile.txt /input
```

(Replace `/path/to/local/inputfile.txt` with the actual path to your input file.)