

Atelier 8

```
hadoop@maram-VirtualBox:~$ export HADOOP_CLASSPATH=$(($HADOOP_HOME/bin/hadoop classpath))
hadoop@maram-VirtualBox:~$ echo $HADOOP_CLASSPATH
/usr/local/hadoop/etc/hadoop:/usr/local/hadoop/share/hadoop/common/lib/*:/usr/local/hadoop/share/hadoop/common/*:/usr/local/hadoop/share/hadoop/hdfs:/usr/local/hadoop/share/hadoop/hdfs/lib/*:/usr/local/hadoop/share/hadoop/hdfs/*:/usr/local/hadoop/share/hadoop/mapreduce/*:/usr/local/hadoop/share/hadoop/yarn:/usr/local/hadoop/share/hadoop/yarn/lib/*:/usr/local/hadoop/share/hadoop/yarn/*:/usr/local/hadoop/lib/javax.activation-api-1.2.0.jar
```

compilateur javac de savoir où trouver les classes et packages Hadoop
Et ensuite faire l'affiche le contenu de la variable HADOOP_CLASSPATH pour s'assurer qu'elle est bien initialisée.

```
hadoop@maram-VirtualBox:~$ javac -classpath $HADOOP_CLASSPATH -d WCClasses WordCount.java
hadoop@maram-VirtualBox:~$ jar -cvf WordCount.jar -C WCClasses/ .
added manifest
adding: WordCount$IntSumReducer.class(in = 1793) (out= 777)(deflated 56%)
adding: WordCount$TokenizerMapper.class(in = 1790) (out= 792)(deflated 55%)
adding: WordCount.class(in = 1511) (out= 832)(deflated 44%)
```

après avoir écrit le programme *WordCount.java*, j'ai procédé à sa compilation et à la création du fichier exécutable JAR.

javac -classpath \$HADOOP_CLASSPATH -d WCClasses WordCount.java

m'a permis de compiler le code source Java en utilisant les bibliothèques Hadoop spécifiées dans la variable d'environnement \$HADOOP_CLASSPATH. Les fichiers .class générés ont été placés dans le dossier WCClasses, assurant une bonne organisation du projet.

Ensuite, la commande:

jar -cvf WordCount.jar -C WCClasses/ .

a servi à regrouper les fichiers compilés dans une archive JAR nommée *WordCount.jar*. Ce fichier représente le programme MapReduce complet, prêt à être exécuté sur le cluster Hadoop.

```
hadoop@maram-VirtualBox:/usr/local/hadoop/logs$ hadoop jar /home/hadoop/WordCount.jar WordCount /At_WC/Input/test.txt /At_WC/Output
2025-11-12 10:51:15,718 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2025-11-12 10:51:15,815 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2025-11-12 10:51:15,815 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2025-11-12 10:51:15,914 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2025-11-12 10:51:16,000 INFO input.FileInputFormat: Total input files to process : 1
2025-11-12 10:51:16,144 INFO mapreduce.JobSubmitter: number of splits:1
2025-11-12 10:51:16,267 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1147009910_0001
2025-11-12 10:51:16,267 INFO mapreduce.JobSubmitter: Executing with tokens: []
2025-11-12 10:51:16,368 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2025-11-12 10:51:16,369 INFO mapreduce.Job: Running job: job_local1147009910_0001
2025-11-12 10:51:16,374 INFO mapred.LocalJobRunner: OutputCommitter set in config null
2025-11-12 10:51:16,386 INFO output.PathOutputCommitterFactory: No output commit
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
```

Ici, le fichier WordCount.jar contient le code compilé du programme, et WordCount est le nom de la classe principale à exécuter. Le chemin /At_WC/Input/test.txt désigne le fichier d'entrée stocké dans le système de fichiers distribué HDFS, tandis que /At_WC/Output correspond au répertoire de sortie où les résultats du traitement seront enregistrés.

```
hadoop@maram-VirtualBox:/usr/local/hadoop/logs$ hadoop fs -cat /At_WC/Output/*
hello    1
world    1
```

affichage de contenu des fichiers de sortie générés par le job MapReduce dans le système de fichiers HDFS.