HADOOP Cheat Sheet

Working with HDFS from the command line

`hadoop dfs <CMD>`

Inspect files

- `-ls <path>`: list all files in `<path>`
- `-cat <src>`: print `<src>` on `stdout`
- `-tail [-f] <file>`: output the last part of the `<file>`
- `-du <path>`: show `<path>` space utilization

Create/remove files

- `-mkdir <path>`: create a directory
- `-mv <src> <dst>`: move (rename) files
- `-cp <src> <dst>`: copy files
- `-rmr <path>`: remove files

Copy/Put files from a remote machine into the HADOOP cluster

- `-copyFromLocal <localsrc> <dst>`: copy a local file to the HDFS
- `-copyToLocal <src> <localdst>`: copy a file on the HDFS to the local disk

HELP

- `-help [cmd]`: hopefully this is self-describing

Examples: `hadoop dfs -ls /`

```
hadoop dfs -copyFromLocal myfile remotefile
```

Launching Hadoop Jobs - Command line

- Copy the jar file of your job to the client machine (let's call it `machine_name`)

```
scp localJarFile studentXX@machine_name:~/
```

- SSH to `machine_name`:

```
ssh studentXX@machine_name
```

- Launch the job:

```
hadoop jar jarFile.jar ClassNameWithPackage [job args]
```

Note that if the output directory exists (and you don't want it) you need to remove it:

```
hadoop dfs -rmr output
```

Example: `hadoop jar fr.eurecom.dsg.WordCount /user/hadoop/wikismall.xml output 2`

Reading (Textual) Input Data in the Mapper

This is the class you're looking for: `org.apache.hadoop.mapreduce.lib.input.TextInputFormat<K,V>`

Precisely, this is the class hierarchy:

`java.lang.Object`

`org.apache.hadoop.mapreduce.InputFormat<K,V>`

`org.apache.hadoop.mapreduce.lib.input.FileInputFormat<LongWritable,Text>`

`org.apache.hadoop.mapreduce.lib.input.TextInputFormat`

Basically, this is an `InputFormat` specifically designed for plain text files. Files are broken into lines. Either linefeed or carriage-return are used to signal end of line. Keys are the position in the file, and values are the line of text. You need to take care of the following:

Key Type: `LongWritable`

Value Type: `Text`

Writing (Textual) Output Data in the Reducer

This is the class you're looking for: `org.apache.hadoop.mapreduce.lib.output.TextOutputFormat<K,V>`

Precisely, this is the class hierarchy:

`java.lang.Object`

`org.apache.hadoop.mapreduce.OutputFormat<K,V>`

`org.apache.hadoop.mapreduce.lib.output.FileOutputFormat<K,V>`

`org.apache.hadoop.mapreduce.lib.output.TextOutputFormat<K,V>`

Essentially, this `OutputFormat` writes plain text files. `TextOutputFormat` calls `toString()` for each key and value pair in output, so any (`Writable`) type can be used.