Compiling Hadoop Code

These instructions will get you to the point that you can compile a JAR on a Linux machine at the command line. You can then run it on your machine or on our [Cluster](https://coursys.sfu.ca/2018fa-cmpt-732-g1/pages/Cluster).

Setup

Start by [downloading the Hadoop release](http://hadoop.apache.org/releases.html). Get the “binary” version in the version you want.

Unpack the Hadoop release somewhere that we'll refer to as HADOOP\_HOME (adjust the code below for the place you put the files). Hadoop also likes to have JAVA\_HOME set:

export HADOOP\_HOME=/home/me/hadoop-3.1.1

export JAVA\_HOME=/usr/lib/jvm/java-8-openjdk-amd64

Note that you **must compile with Java 8** (or earlier) if you're going to run on the cluster: you need class-file compatibility with the installation there.

CSIL Linux

On the lab computers, the right values are:

export JAVA\_HOME=/usr/lib/jvm/java-8-oracle/

export HADOOP\_HOME=/usr/shared/CMPT/big-data/hadoop-2.6.0

On the Gateway

If you need to compile code on gateway.sfucloud.ca, you don't need to set those environment variables, but see the alternate versions of the commands below.

Compiling

Now you can compile your Java code (adding more javac calls as necessary) and create a JAR like this:

${JAVA\_HOME}/bin/javac -classpath `${HADOOP\_HOME}/bin/hadoop classpath` WordCount.java

${JAVA\_HOME}/bin/jar cf wordcount.jar WordCount\*.class

On the Gateway

It will be easier to compile code on your local machine, but it's possible to do it on gateway.sfucloud.ca with commands like:

javac -classpath `hadoop classpath` WordCount.java

jar cf wordcount.jar WordCount\*.class

Running Locally

You should be able to run the job on your computer (or one of the computers in the lab) with a command like:

${HADOOP\_HOME}/bin/yarn jar wordcount.jar WordCount \

wordcount-1 output-1

less output-1/part-\*

Running on the Cluster

You can transfer this JAR file to the cluster like this (but see the [Cluster](https://coursys.sfu.ca/2018fa-cmpt-732-g1/pages/Cluster) instructions for more details):

scp wordcount.jar <USERID>@gateway.sfucloud.ca:

And on the cluster run it:

yarn jar wordcount.jar WordCount \

/courses/732/wordcount-1 output-1

hdfs dfs -cat output-1/part-\* | less

Adding JARs

If you have additional dependencies in .jar files, you can tell the Hadoop tools about them:

export HADOOP\_CLASSPATH=/path/to/jar1.jar:/path/to/jar2.jar

And then compile as above. (The hadoop classpath command checks that variable and adds them to your compilation.)

When you run your code on the cluster, you need to tell YARN that the extra .jar file needs to be distributed to the nodes during execution. That is done with the -libjars argument like this:

yarn jar jarfile.jar ClassName -libjars /path/to/jar1.jar,/path/to/jar2.jar arg0 arg1