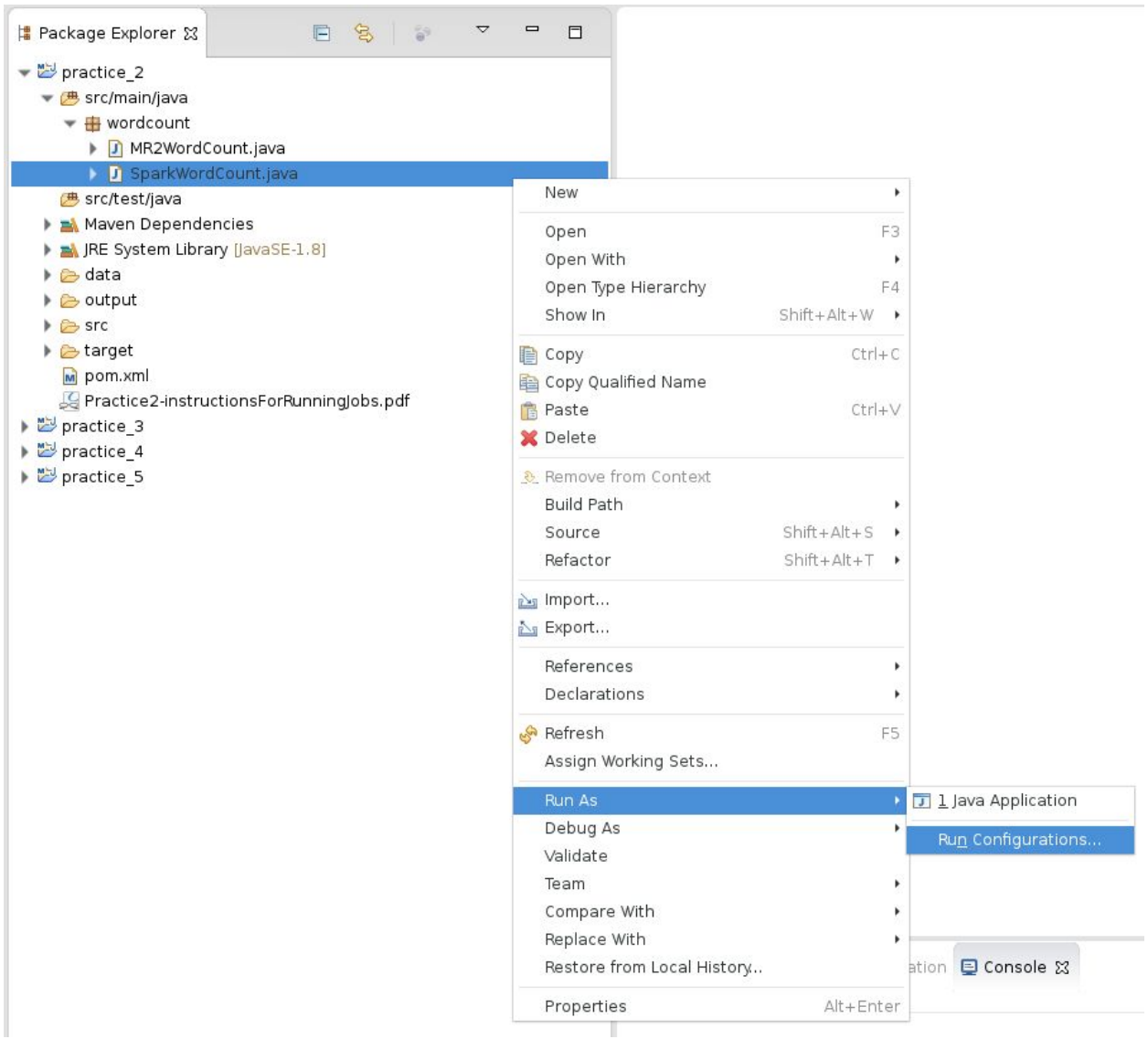
# Practice 2 - Learning to run a job in the VM

## Running SparkWordCount.java from Eclipse

### Create the run configuration

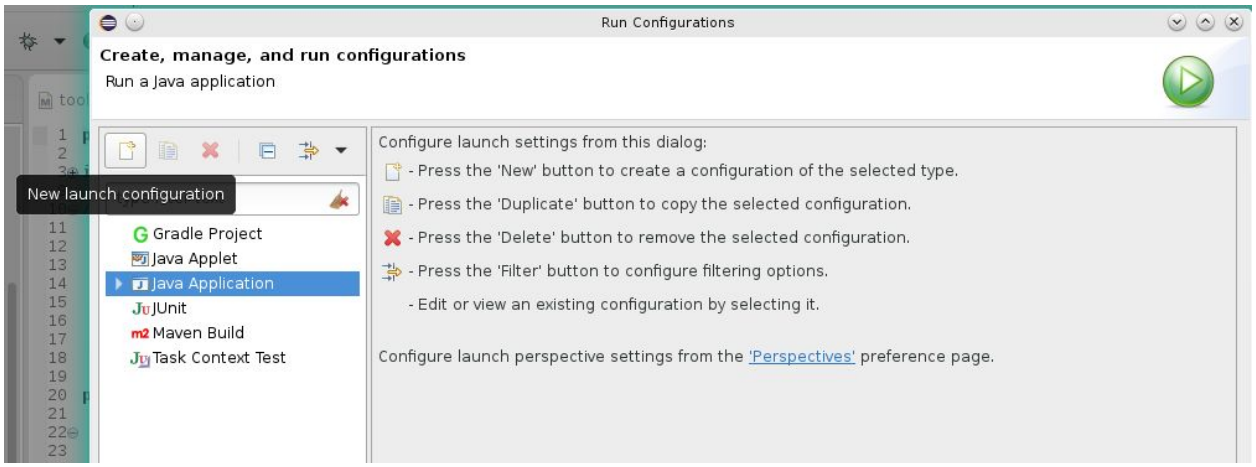
Start in Eclipse's package explorer:

- Select the project containing **`SparkWordCount.java`**
- Select the **`SparkWordCount.java`** file -> select "Run As" -> select "Run Configurations"
- An example is shown here:



## The Run Configuration dialog.

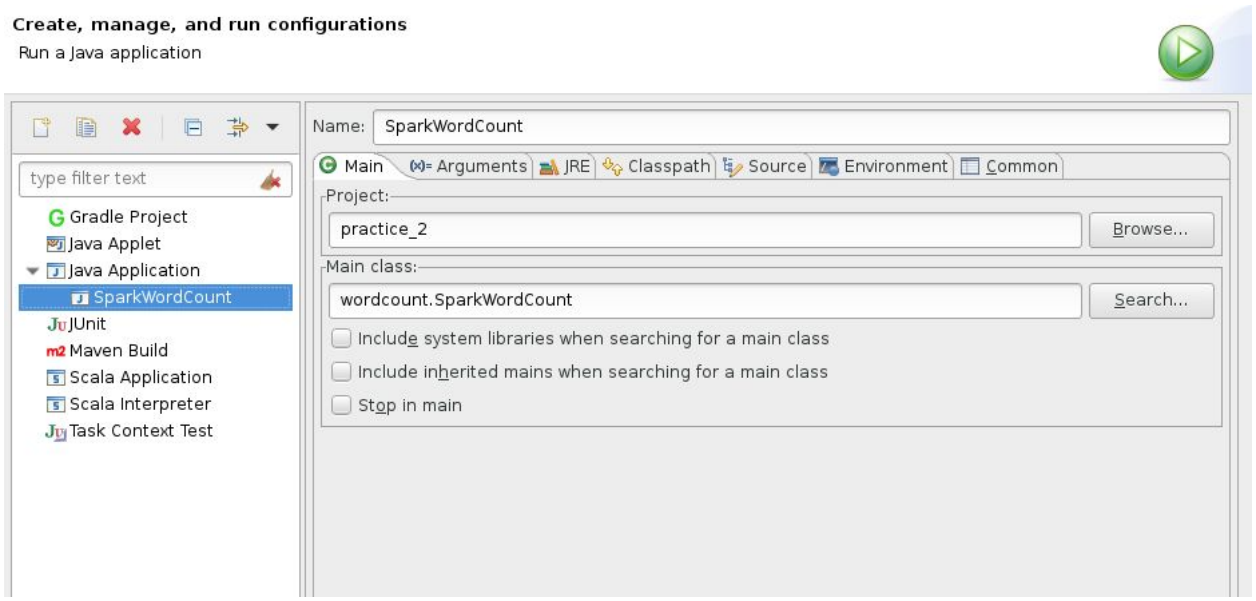
First, create a new launch configuration for a Java Application by selecting "Java Application" and then clicking on the 'New' button.




Now, you see a Java Application called SparkWordCount in the dialog for configuring the Application.

It already has

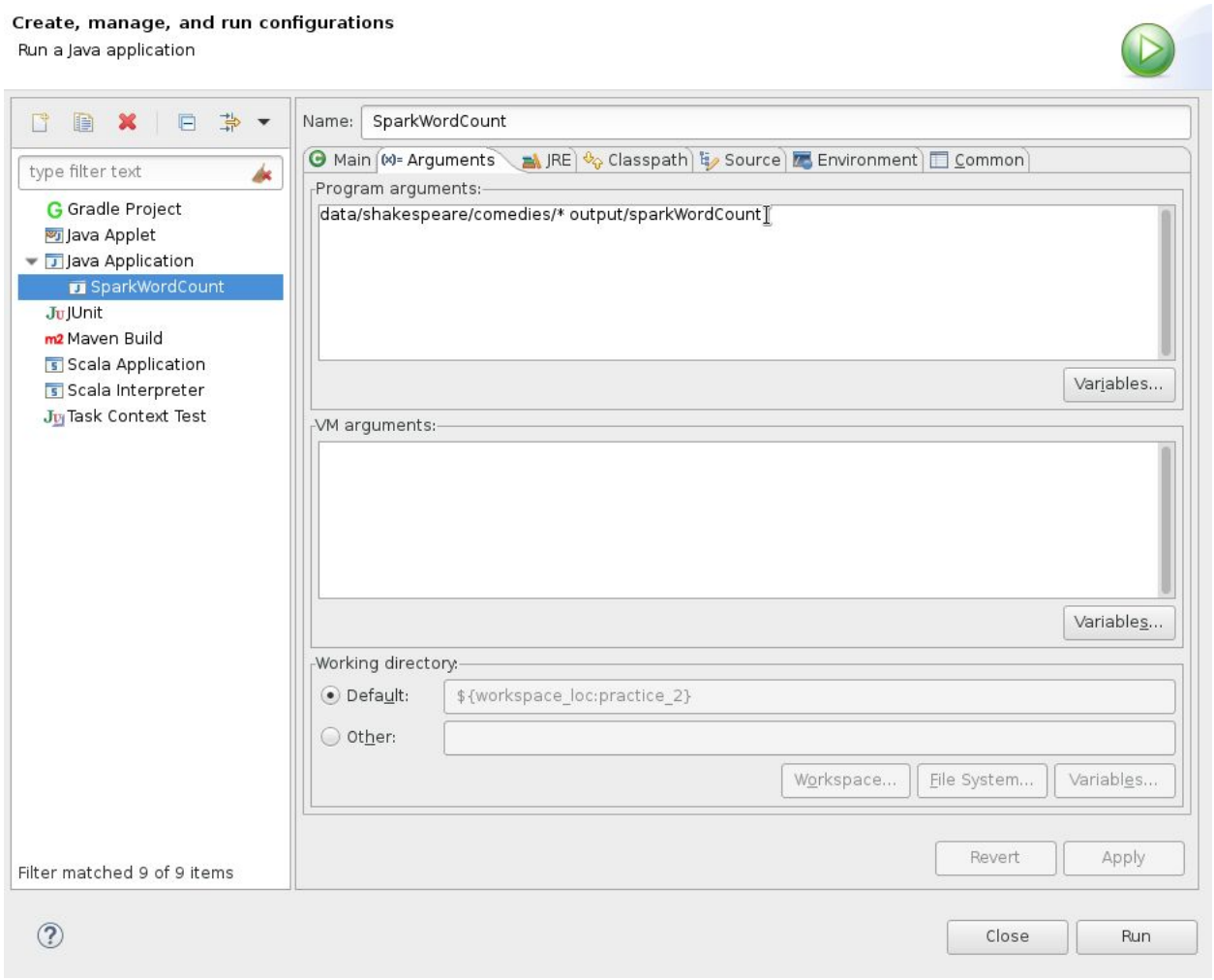
- Name: SparkWordCount (you can change this)
- Project: practice_2
- Main class: wordcount.SparkWordCount



## Click on the arguments tab and fill in the arguments.

- The first argument is the location of the input data.
    - Note: example shows a *relative path* (does not start with a slash "/")
    - Because `shakespeare/comedies` is in a folder in our Eclipse project, we can use a relative path.
    - To refer to data NOT to your project, you need to use the full path.

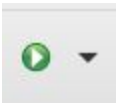
- The second argument is the location of your output directory.
    - Again, this uses a relative path to the "output" folder in your project.



## Run

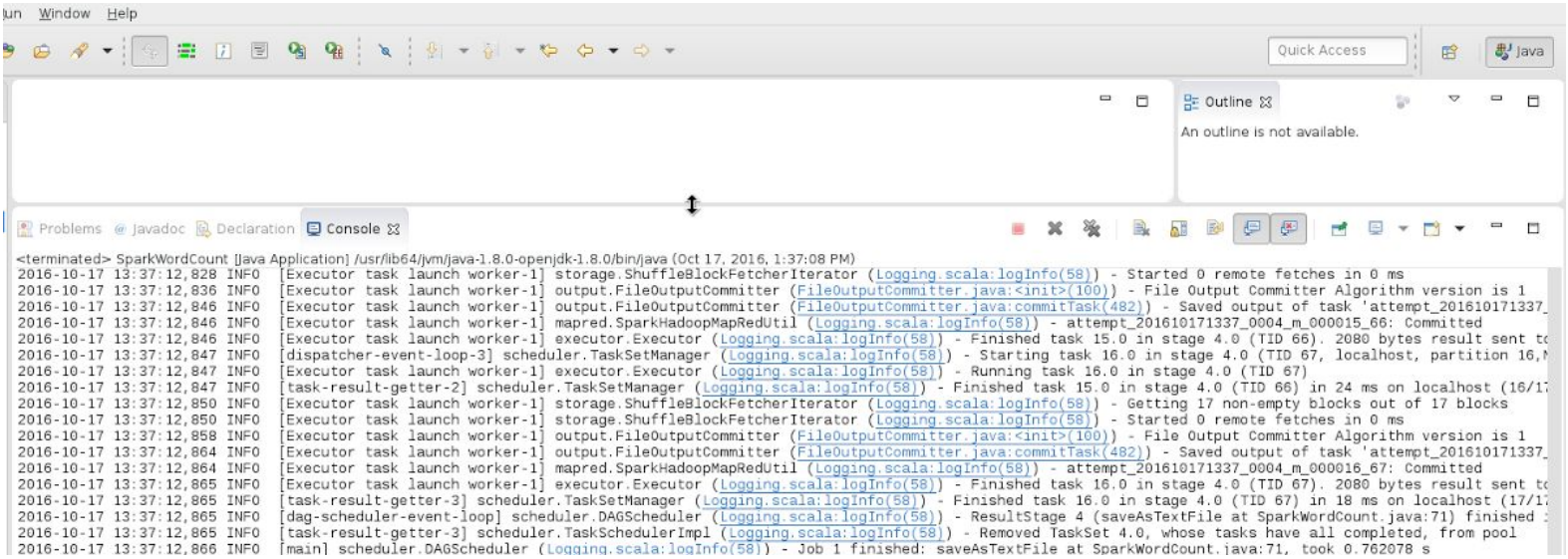
At this point you can run the program by clicking "Run".

You can always rerun this program by clicking the "Run" arrow. It will show you which run configuration you are using when you mouse-over it.



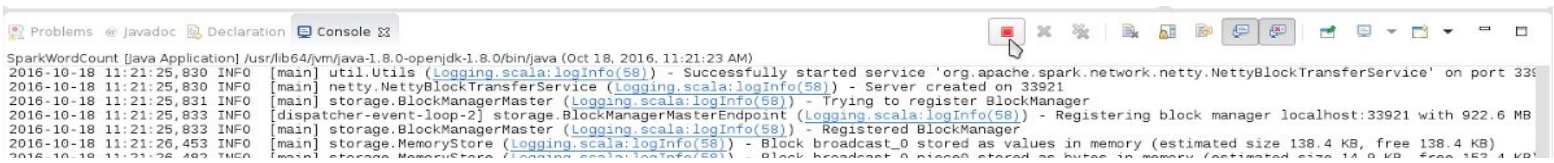
## Processing is shown in the console

After you select run, the logged messages for the run will scroll onto your console. You can resize the console to see the results more clearly. Here are the results of running SparkWordCount:



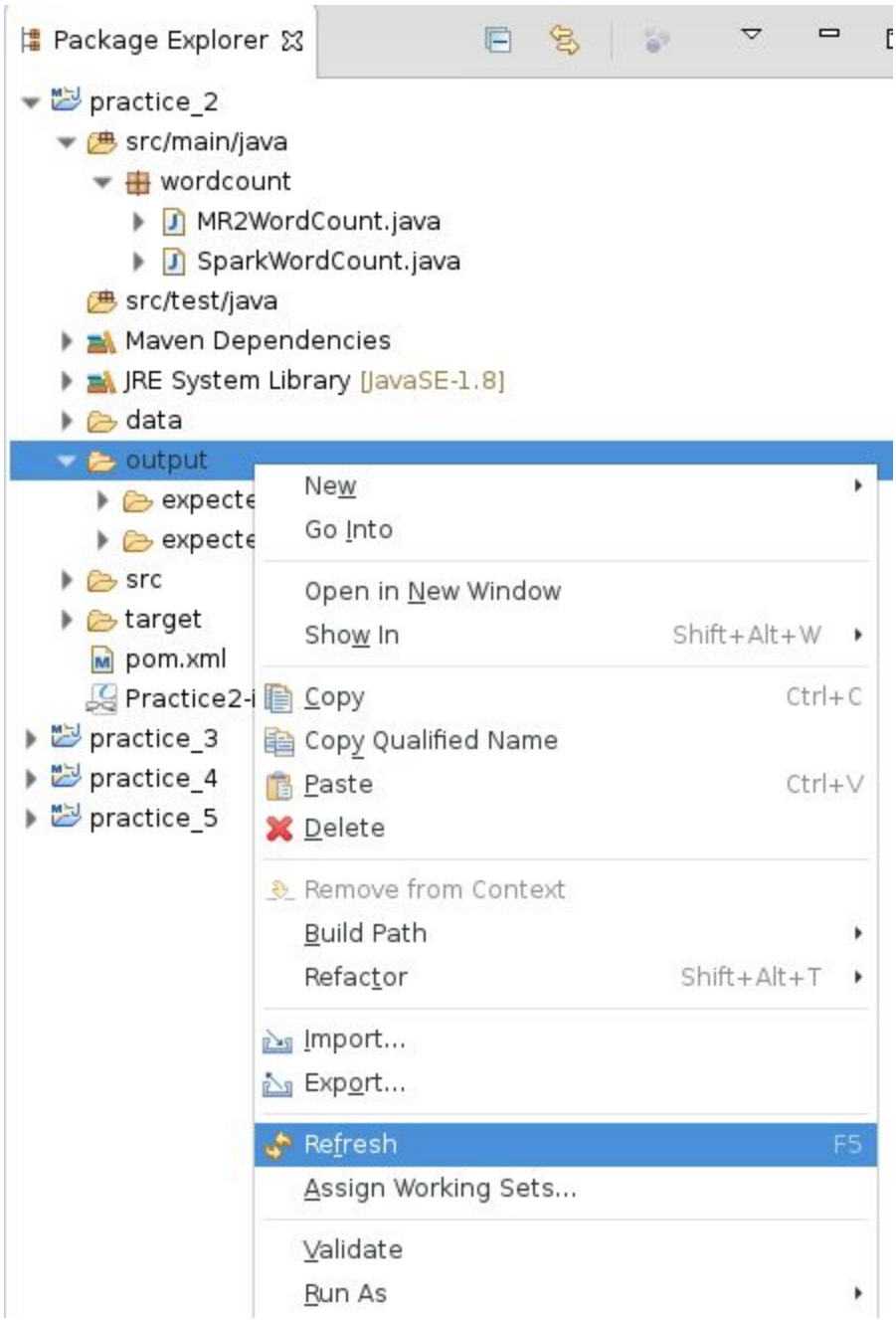

## Stopping a job

Simple - just click on the little red box shown on the console



## Output

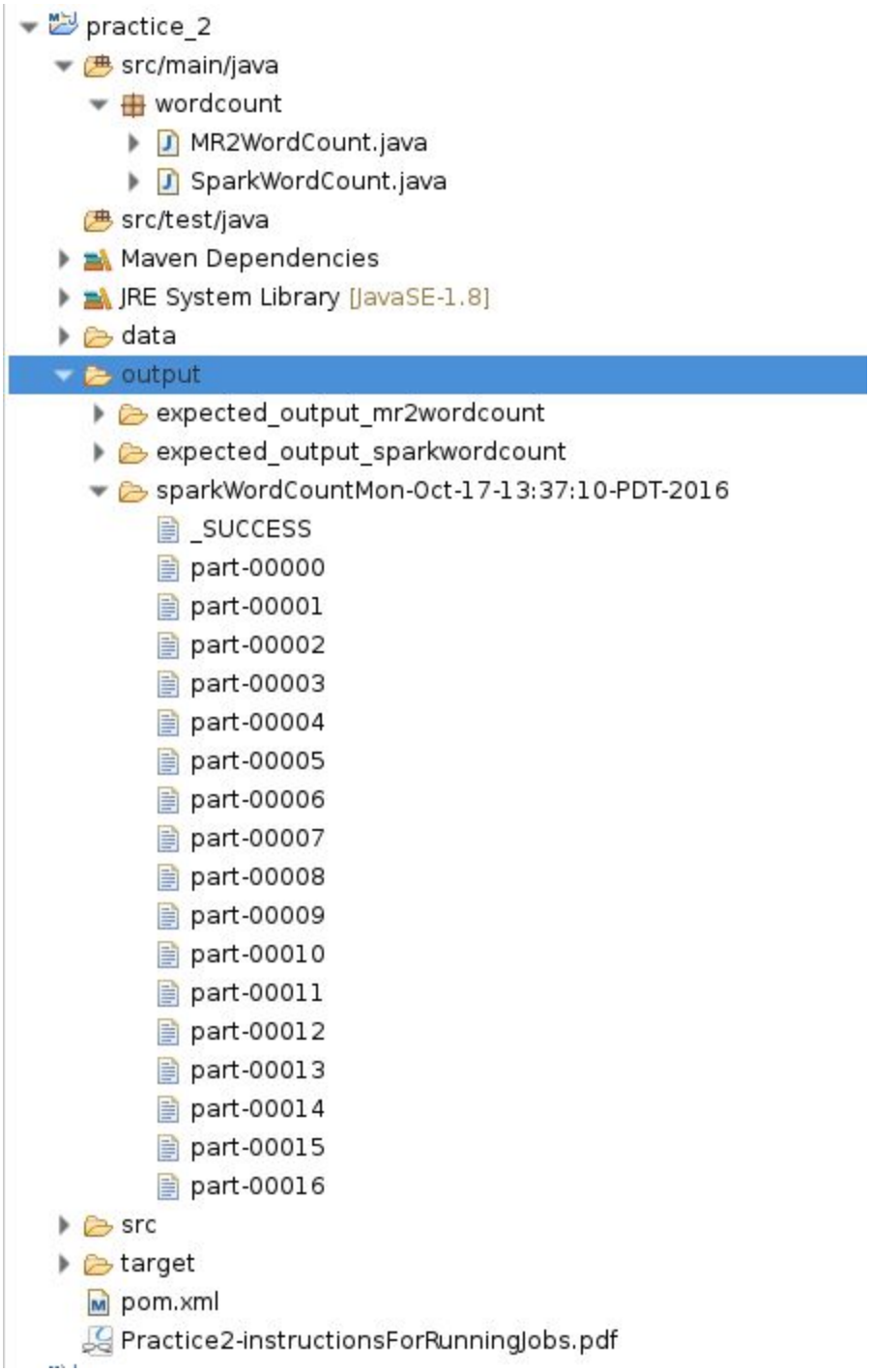
To see the output, right-click on the output folder and refresh:



**Finally, you will see the output.** Click on the SparkWordCount<date> folder to expand.

The _SUCCESS file indicates that the job completed successfully. You can ignore it.

The 'part-*' files contain the output. Each part file is the output from one task. In this case, there are 17 reduce tasks running, so there are 17 output files.

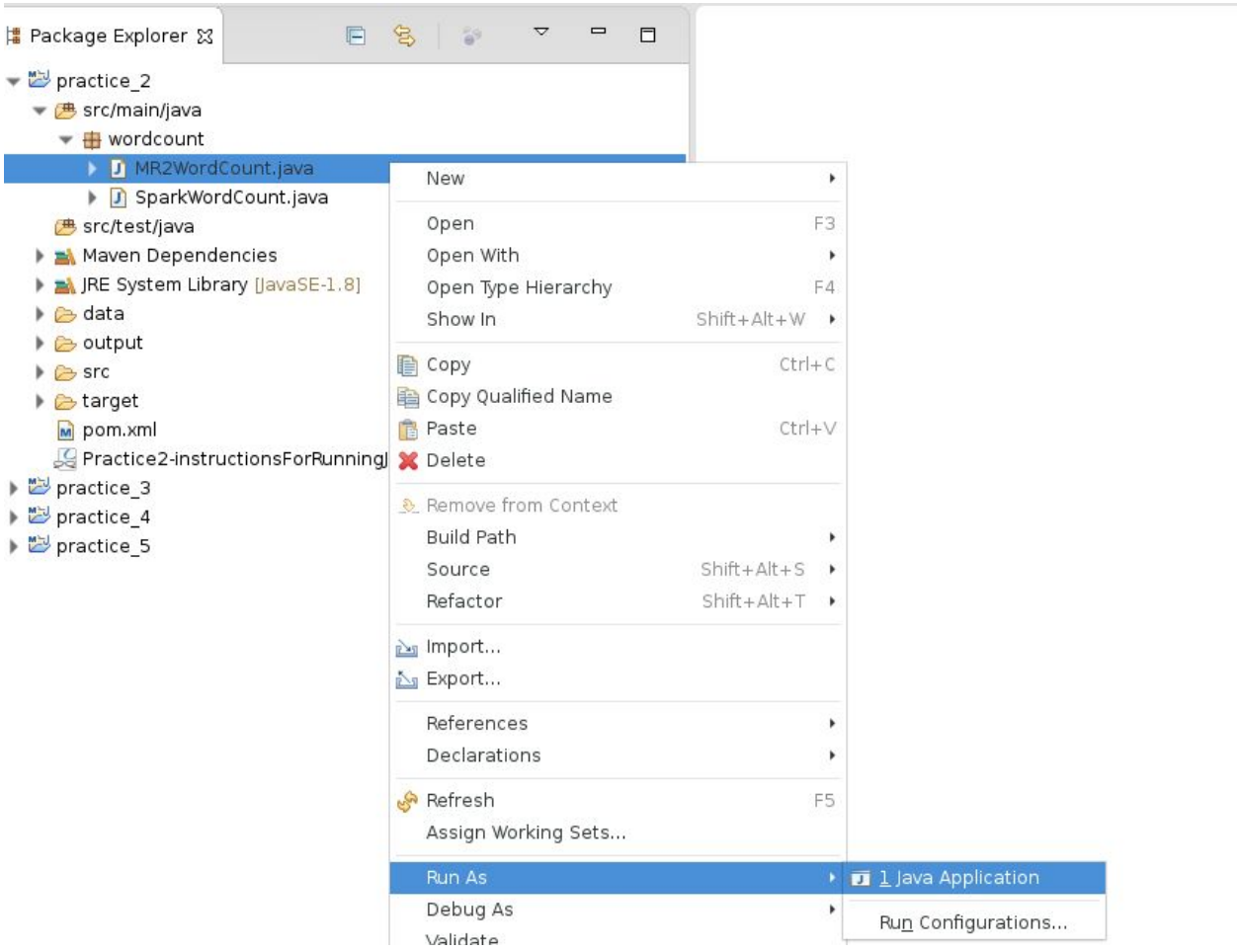


# Running MR2WordCount.java from Eclipse.

You may notice that there is an MR2WordCount.java program in Practice 2. The run configuration for MR2WordCount.java is already setup, so you can just run it.

Once a Run Configuration exists, you can use it (shown below):

- right-click on the app you want to run in the Package Explorer
- selecting “RunAs”
- select “Java Application”



# Running from the commandline

## 1) Create the jar file for the project:

- In the **Package Explorer, right-click on the project (practice_2)**
- Select **Run-As** and then select **Maven Install.**
  - This will create the jar file
  - You will see the process in the Eclipse console
  - Notice, you can cleanup old jar files by running **Maven clean** *before* you run Maven install.
- In the **Package Explorer,** open **target** folder
  - it should now contain a jar file called: **practice-2.1.0.0-SNAPSHOT.jar**
  - this is the jar file you will use to run the jobs below

## 2) Run a Spark job on the cluster

When you run locally, Spark's master does not get involved and there is no webUI for viewing the job and its execution. To get the master involved, you can run on the cluster. The syntax is:

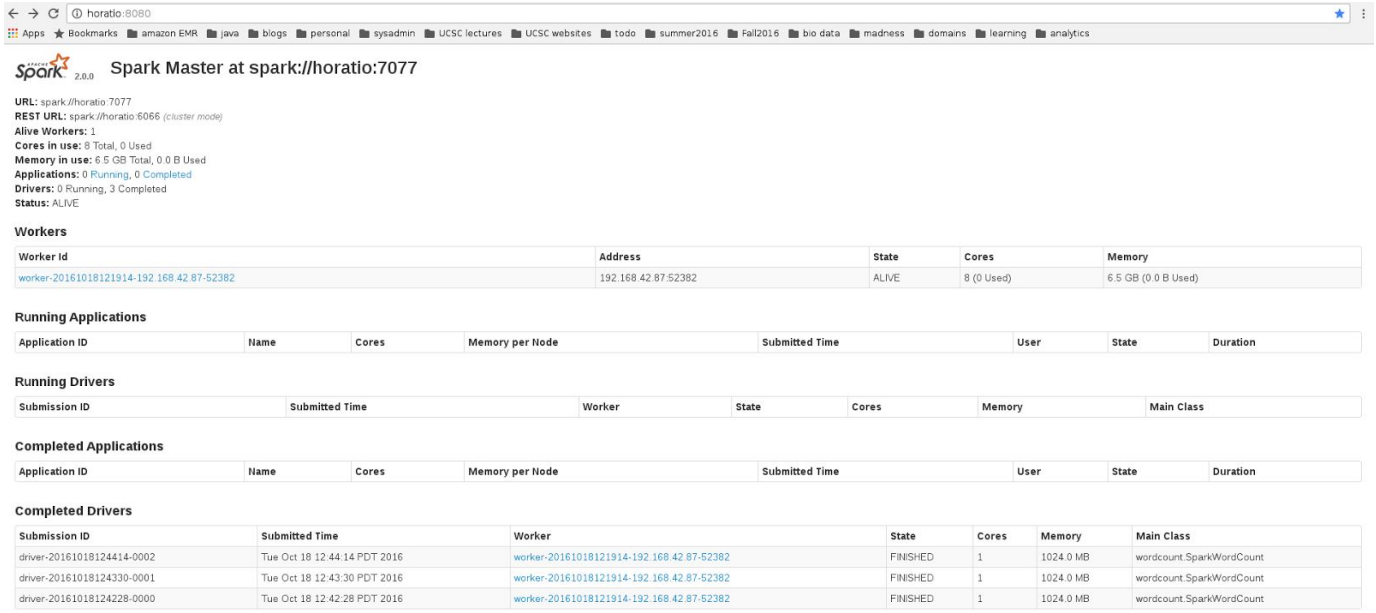
```
spark-submit \
--master <REST URL> \
--deploy-mode cluster \
--class <MainClassName> \
<myJar>.jar <input> <output>
```

- make sure you are in the target directory containing your jar file.
  - Note, you can type `pwd` to check your present working directory.
  - if necessary, `cd /home/cloudera/workspace/practice_2/target`
- run the SparkWordCount job *on the cluster* via the REST server on port 6066:

```
$ spark-submit \
--master spark://quickstart.cloudera:6066  \
--deploy-mode cluster \
--class wordcount.SparkWordCount \
practice_2-1.0.0-SNAPSHOT.jar \
/home/cloudera/shakespeare/* /home/cloudera/practice2_sparkout
```

## View the job - Use the Spark Master UI

- Open a browser and go to the master's UI at **quickstart.cloudera:8080.**
- On my machine, it looks like this. On your VM, you will see a different master url.




**Refresh the page.** You should see at least one "Running Drivers" or "Completed Drivers". You can keep refreshing until your job is listed under "Completed Drivers". It should take less than a minute.

Under **Completed Drivers** click on the Worker for the job you just ran.

- The **Worker UI** will show your job and also logs for the job.
- Click on *stderr* to see the information logged for the job.

## Killing a Spark job[1]

When you do need to kill a Spark application, you can do so by: using spark-class:

```
$ spark-class org.apache.spark.deploy.Client kill <master url> <submission ID>
```

You can find the `master url` and the `submission ID` on the master's web UI, **quickstart.cloudera:8080.**

[1] Don't try this now, you will drive yourself crazy trying to kill it before it completes.

# 3) Run an MR2 job on the cluster

The simplest way to submit job to Hadoop is to use the following syntax:

```
hadoop jar <myJar>.jar <MainClassName> <input> <output>
```

For this practice:

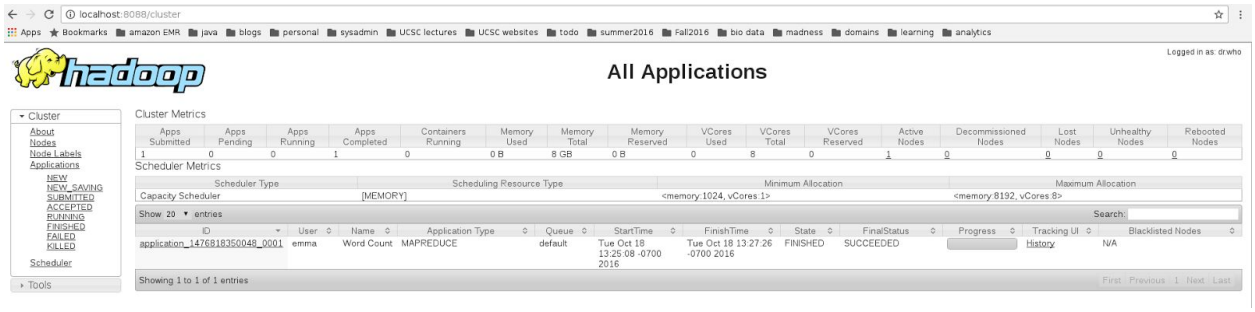
- if you haven't already done so, open a terminal window and cd to the target dir:

  ```
  $ cd /home/cloudera/workspace/practice_2/target
  ```

- run the MR2WordCount job by typing:

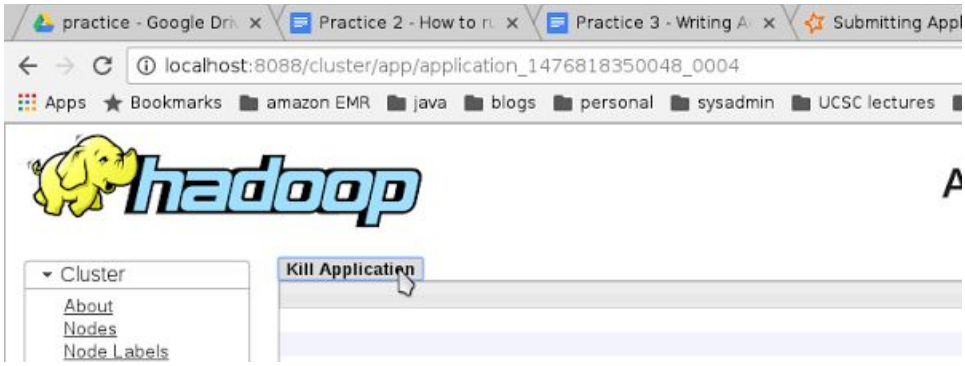
  ```
  $ hadoop jar practice_2-1.0.0-SNAPSHOT.jar \
  wordcount.MR2WordCount \
  /home/cloudera/shakespeare/* /home/cloudera/practice2_mr2out
  ```

- open the UI for Hadoop jobs at **quickstart.cloudera:8088**




## Killing an MR2 job

The simplest way to kill a Hadoop job is to use the UI. From the same UI, drill-down on the link for the running application, and select kill.



# End of Practice 2