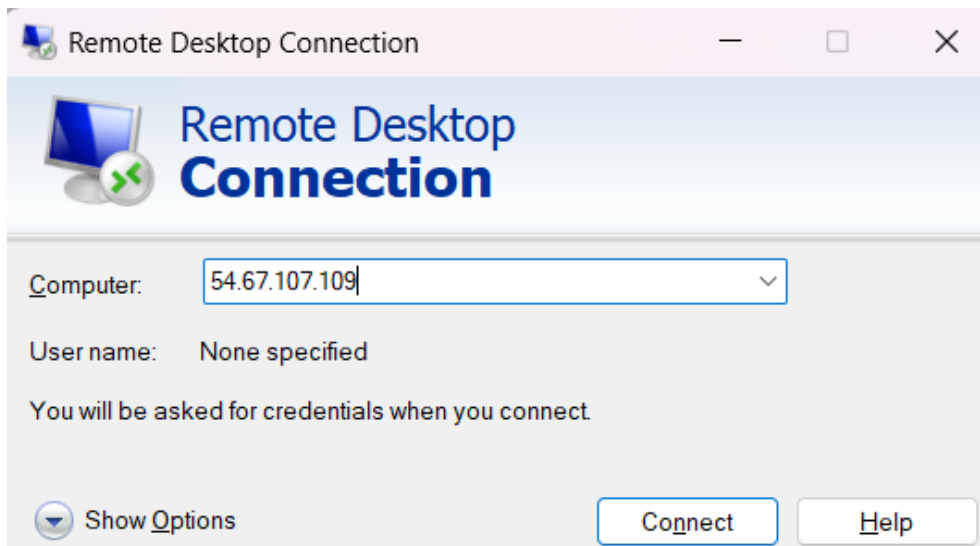


Apache Flink Getting Started

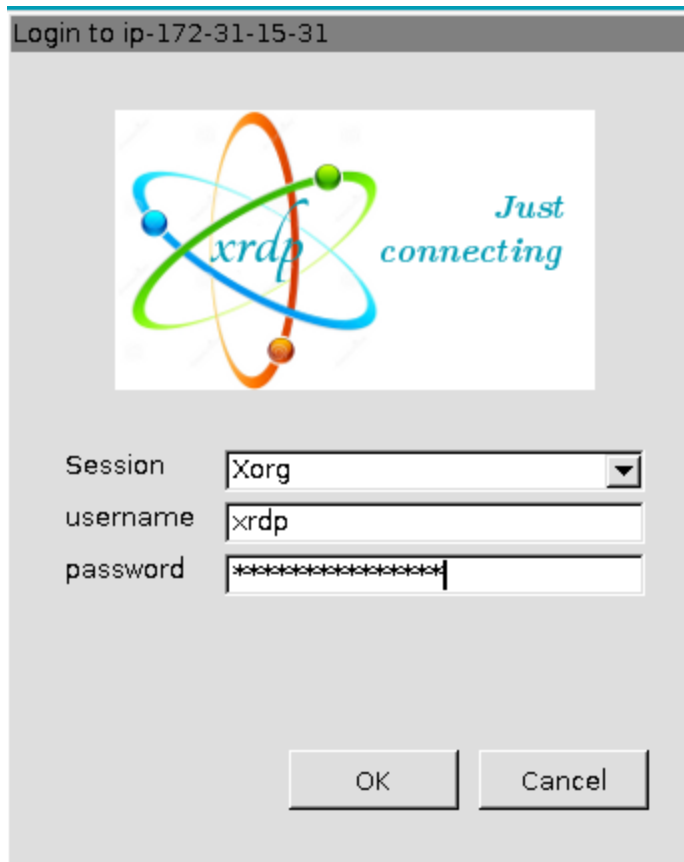
Experiment 5: Word Count

1.1 Steps to run your next Flink Program

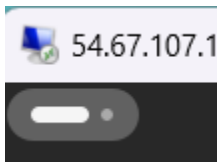
- 1.1.1 Browse to the GitHub repo that you cloned. This should be cloned to your Windows Jump Box and the Flink Development Server
<https://github.com/GeorgeNiece/flink-data-processing-2day>
- 1.1.2 From a command prompt on your jumpbox machine SSH to the Ubuntu server
ssh -o ServerAliveInterval=180 -o ServerAliveCountMax=2 -i ansible.pem ubuntu@ip_address_provided
- 1.1.4 Change to the flink folder, verify Flink isn't started, start the Flink dev cluster, and verify that it started
ps -ef | grep flink
cd ~/flink-2.0.0
./bin/start-cluster.sh
ps -ef | grep flink



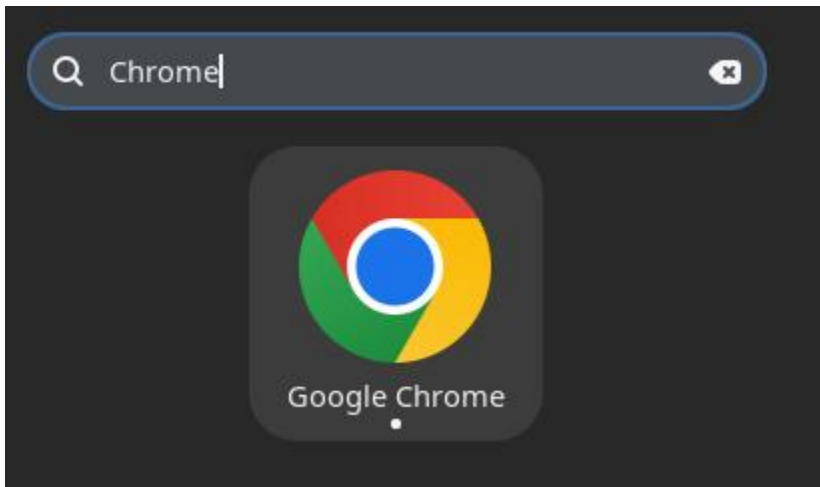
- 1.1.5 Login to the ubuntu dev sandbox using Windows RDP with the xrdp user and the password that you set in Step 1.14



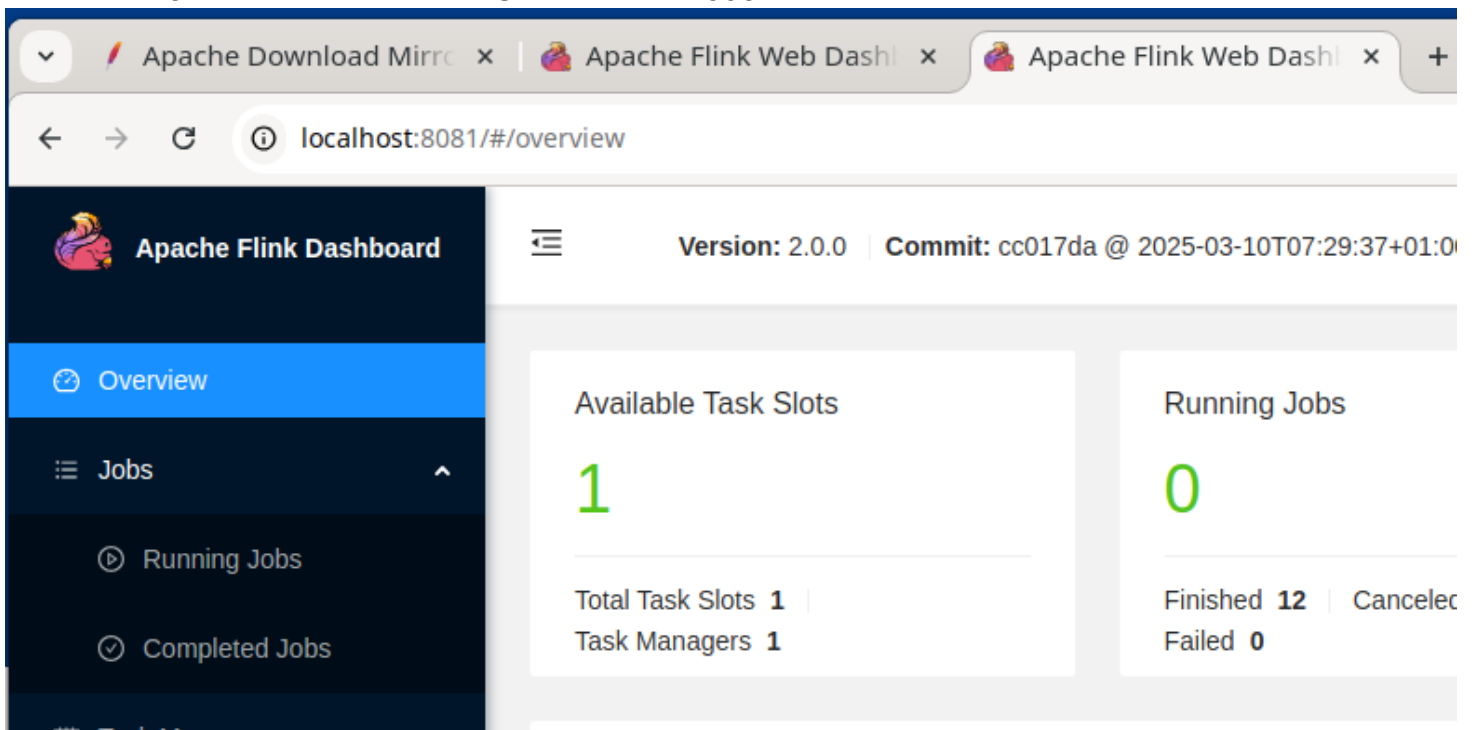
1.1.6 Click the Activities button in the top left corner of the Ubuntu Desktop



1.1.7 Wait for the Search Box at the top of the Ubuntu Desktop, and enter Chrome, click on the Launch Logo



1.1.8 Load the Flink Web UI at localhost:8081



1.1.9 Click the Job Manager in the left hand navigation

The screenshot displays the Apache Flink Dashboard interface. The sidebar on the left contains the following navigation items: Overview, Jobs (with a sub-menu arrow), Running Jobs, Completed Jobs, Task Managers, and Job Manager (highlighted in blue). The main content area is titled 'Version: 2.0.0 | Commit: cc017da @ 2025-03-10T07:29:37+01:00'. Below the title, there are tabs for Metrics, Configuration, Logs, Stdout, Log List, and Thread Dump. The 'Metrics' tab is active, showing a diagram of the Flink Memory Model and a table of Effective Configuration.

Flink Memory Model Diagram:

- Total Process Memory (outermost box)
- Total Flink Memory (inner box)
- JVM Heap (blue box inside Total Flink Memory)
- Off-Heap (yellow box inside Total Flink Memory)

Effective Configuration Table:

Configuration Item	Value
JVM Heap	1.00 GB
Off-Heap Memory	128 MB

1.1.10 Select Log List in the page navigation. These are the logs we'll monitor while we're running some of our experiments

Apache Flink Dashboard

Version: 2.0.0 | Commit: cc017da @ 2025-03-10T07:29:37+01:00 | Mess

Metrics Configuration Logs Stdout **Log List** Thread Dump Profi

Log Name	Last Modified Time	Size (KB)
flink-ubuntu-taskexecutor-0-ip-172-31-15-31.out	2025-04-13 22:08:34.730	0.36
flink-ubuntu-taskexecutor-0-ip-172-31-15-31.log	2025-04-14 00:05:49.716	164.46
flink-ubuntu-taskexecutor-1-ip-172-31-15-31.log.1	2025-04-13 16:07:54.038	49.7

- 1.1.11 The first `flink-ubuntu-taskexecutor*.out` file will be the one we spend the most time looking at. Select that so that we can view there. We could open two browser tabs and watch the Jobs -> Running Jobs page while we run our first experiment.
- 1.1.12 Navigate back to the SSH terminal to the flink distribution folder

cd ~/flink-2.0.0

./bin/flink run ~/flink-data-processing-2day/experiments/built/WordCount.jar

This should show us with the Job submission, Program execution finished, JobID and the Job Runtime

```
ubuntu@ip-172-31-78-140:~/flink-2.0.0$ ./bin/flink run ~/flink-data
Executing example with default input data.
Use --input to specify file input.
Printing result to stdout. Use --output to specify output path.
Job has been submitted with JobID 068084854350cf23790b23070d392cd4
Program execution finished
Job with JobID 068084854350cf23790b23070d392cd4 has finished.
Job Runtime: 183 ms
```

View the `flink-ubuntu-taskexecutor*.out`, this time we'll just tail the file which shows us the following. For my sandbox environment that would be

~/flink-2.0.0:\$ tail log/flink-ubuntu-taskexecutor-0-ip-172-31-78-140.out

Which shows us the following

```
[ubuntu@ip-172-31-78-140:~/flink-2.0.0$ tail log
(nymph,1)
(in,3)
(thy,1)
(orisons,1)
(be,4)
(all,2)
(my,1)
(sins,1)
(remember,1)
(d,4)
```

We can view the logs to see the last written with the unix command `ls -alrt`

1.2 Steps to build your next Flink Program

- 1.2.1 We have the source for the Flink program in both the flink-project structure and the executable jar. **WordCount** uses the DataStream API to create a source data in the default WordCountData class that loads a paragraph of a Shakespeare sonnet. Then it is parsed to count all the words. Interestingly enough the last entry in our output is d due to that being a contraction ending for a number of words used. It keeps a running total of each word that is viewed and examining the larger log shows us the flow from start to finish of the program.
- 1.2.2 To compile from command line with Java you would need to reference the flink distribution jar files in your classpath, either directly as noted here or in a build tool like Maven or Gradle. A sample POM file is included in our course GitHub repo in the flink-project
- 1.2.3 To compile from command line with Java you would need to reference the flink distribution jar files in your classpath, either directly as noted here or in a build tool like Maven or Gradle. A sample POM file is included in our course GitHub repo in the flink-project

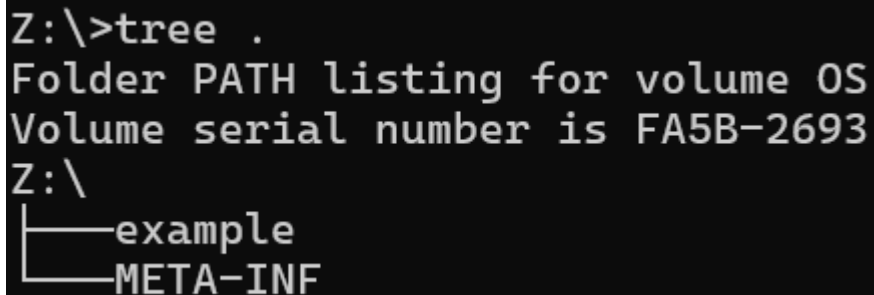
```
javac -classpath C:\lib-2.0\flink-cep-2.0.0.jar;C:\lib-2.0\flink-connector-files-2.0.0.jar;C:\lib-2.0\flink-csv-2.0.0.jar;C:\lib-2.0\flink-dist-2.0.0.jar;C:\lib-2.0\flink-json-2.0.0.jar;C:\lib-2.0\flink-scala_2.12-2.0.0.jar;C:\lib-2.0\flink-table-api-java-uber-2.0.0.jar;C:\lib-2.0\flink-table-planner-loader-2.0.0.jar;C:\lib-2.0\flink-table-runtime-2.0.0.jar;C:\lib-2.0\log4j-1.2-api-2.24.1.jar;C:\lib-2.0\log4j-api-2.24.1.jar;C:\lib-2.0\log4j-core-2.24.1.jar;C:\lib-2.0\log4j-slf4j-impl-2.24.1.jar;C:\lib-2.0\flink-streaming-java-
```

```
1.20.1.jar;C:\lib-2.0\flink-runtime-2.0.0.jar;  
example/WordCount.java
```

- 1.2.4 To package the executable jar for the Flink program we create a MANIFEST.MF that we'll use in the packaging, notice that the

```
Manifest-Version: 1.0  
Implementation-Title: Flink : Examples : Simple Stream  
Implementation-Version: 2.0.0  
Archiver-Version: Plexus Archiver  
Built-By: geoniece  
Specification-Vendor: Innovation in Software  
Specification-Title: Flink : Examples : Simple Stream  
Implementation-Vendor-Id: com.innovationinsoftware  
program-class: example.WordCount  
Implementation-Vendor: Innovation in Software  
Created-By: Apache Maven 3.8.6  
Build-Jdk: 1.11.0_312  
Specification-Version: 2.0.0
```

- 1.2.5 We have a folder structure with our package

A terminal window showing the output of the 'tree .' command. The output lists the current directory structure: 'example' and 'META-INF'.

```
Z:\>tree .  
Folder PATH listing for volume OS  
Volume serial number is FA5B-2693  
Z:\  
├── example  
└── META-INF
```

- 1.2.6 To package the executable jar for our Flink program we do the following

```
Ubuntu-@ip-172.15.50.23:~$ jar --manifest=META-INF/MANIFEST.MF --create -  
-file c:\users\Geo\WordCount.jar example/*
```

- 1.2.7 **Congratulations, time to celebrate** you ran another Flink program in our session