Apache Flink Getting Started

Experiment 2: Getting Started Example

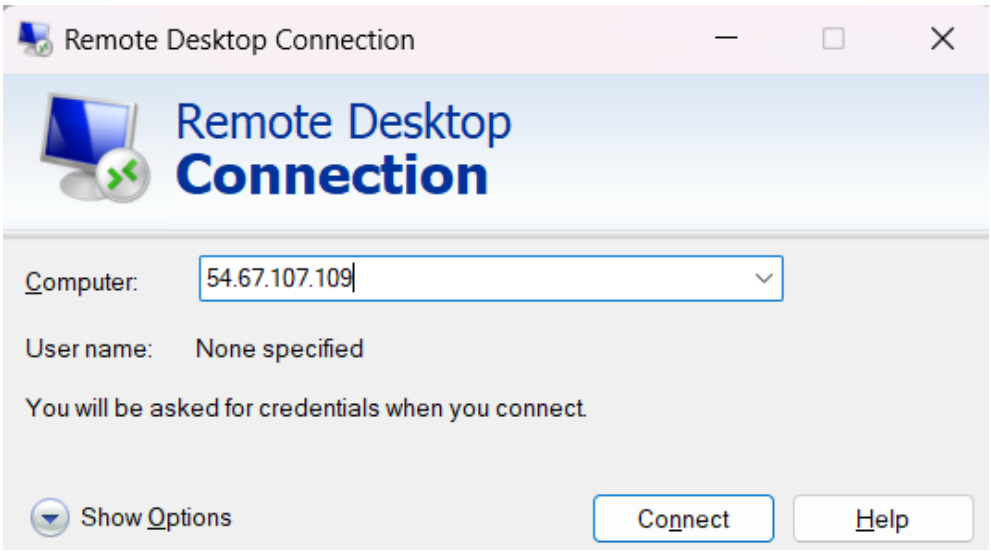
1.1 Steps to run your first Flink Program

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

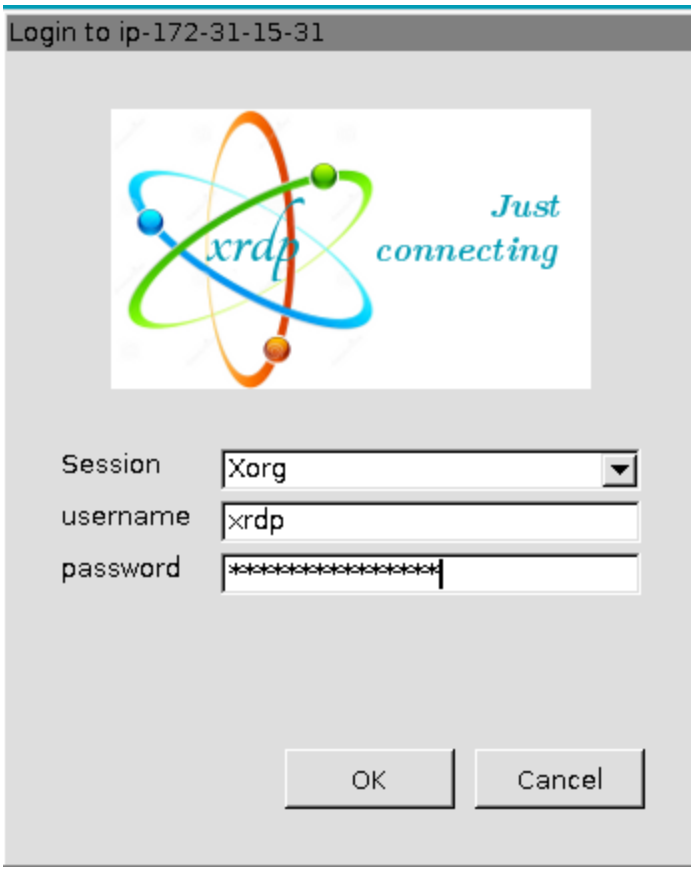
1.1.2 From a command prompt on your jumpbox machine SSH to the Unbuntu server
ssh -o ServerAliveInterval=180 -o ServerAliveCountMax=2 -i ansible.pem ubuntu@*ip_address_provided*

1.1.4 Change to the flink folder, verify Flink isn't started, start the Flink dev cluster, and verify that it started

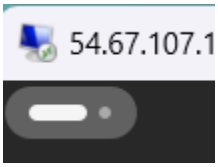
ps -ef | grep flink
cd ~/flink-2.0.0
./bin/start-cluster.sh
ps -ef | grep flink



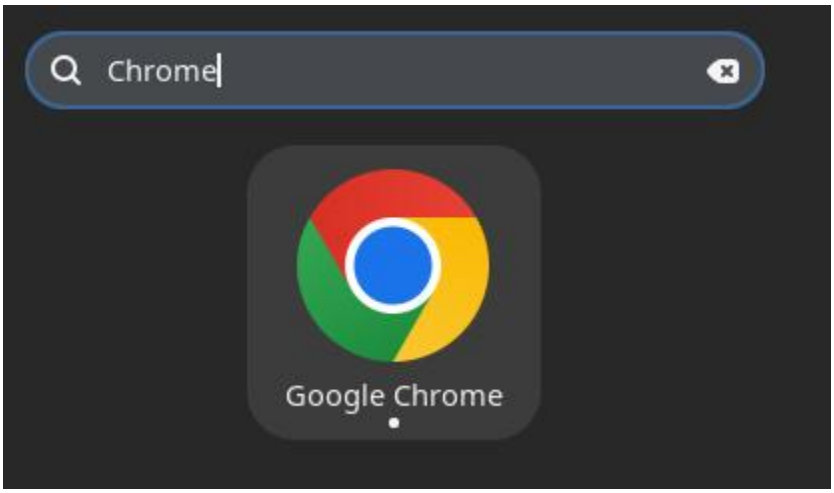
1.1.5 Login to the ubuntu dev sandbox using Windows RDP with the xrdp user and the password that you set in Step 1.14



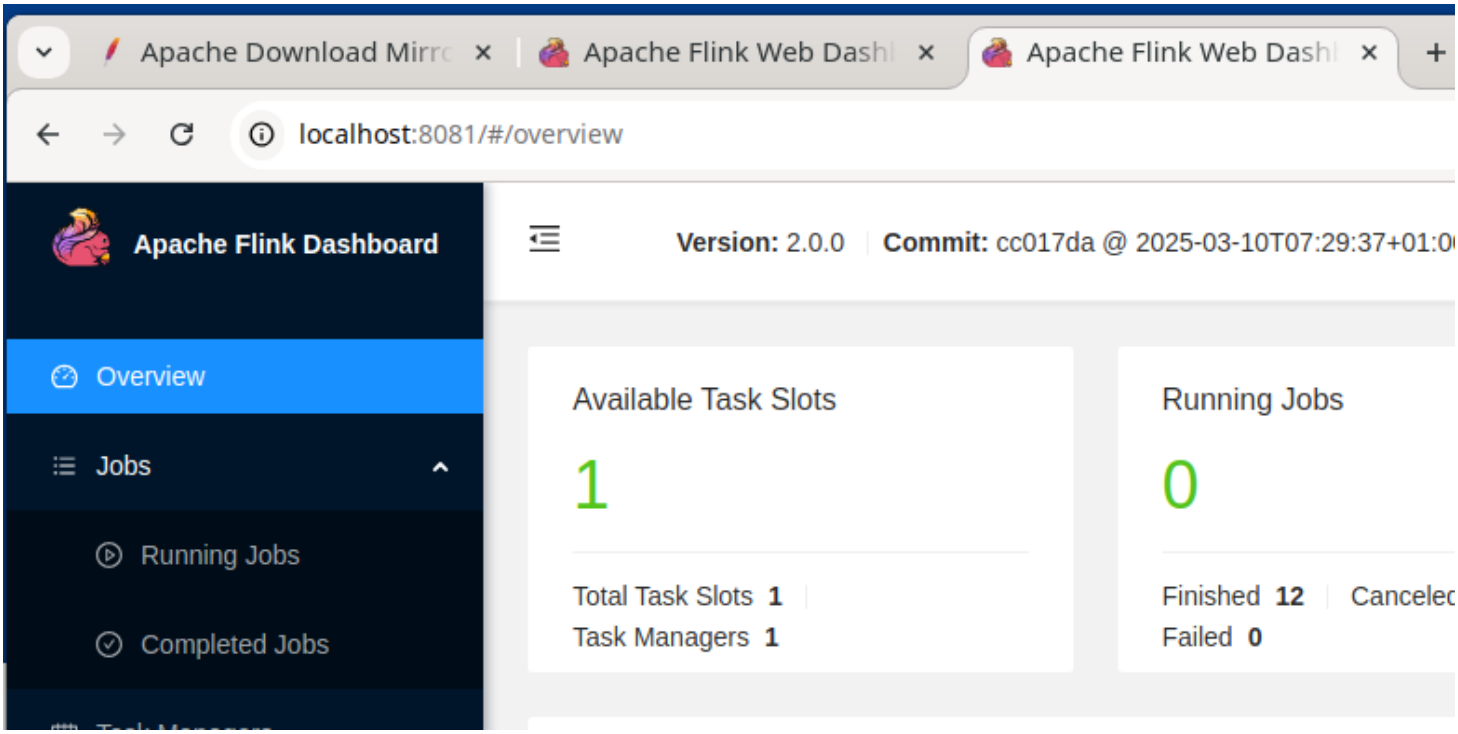
1.1.6 Click the Activities button in the top left corner of the Ubuntu Desktop



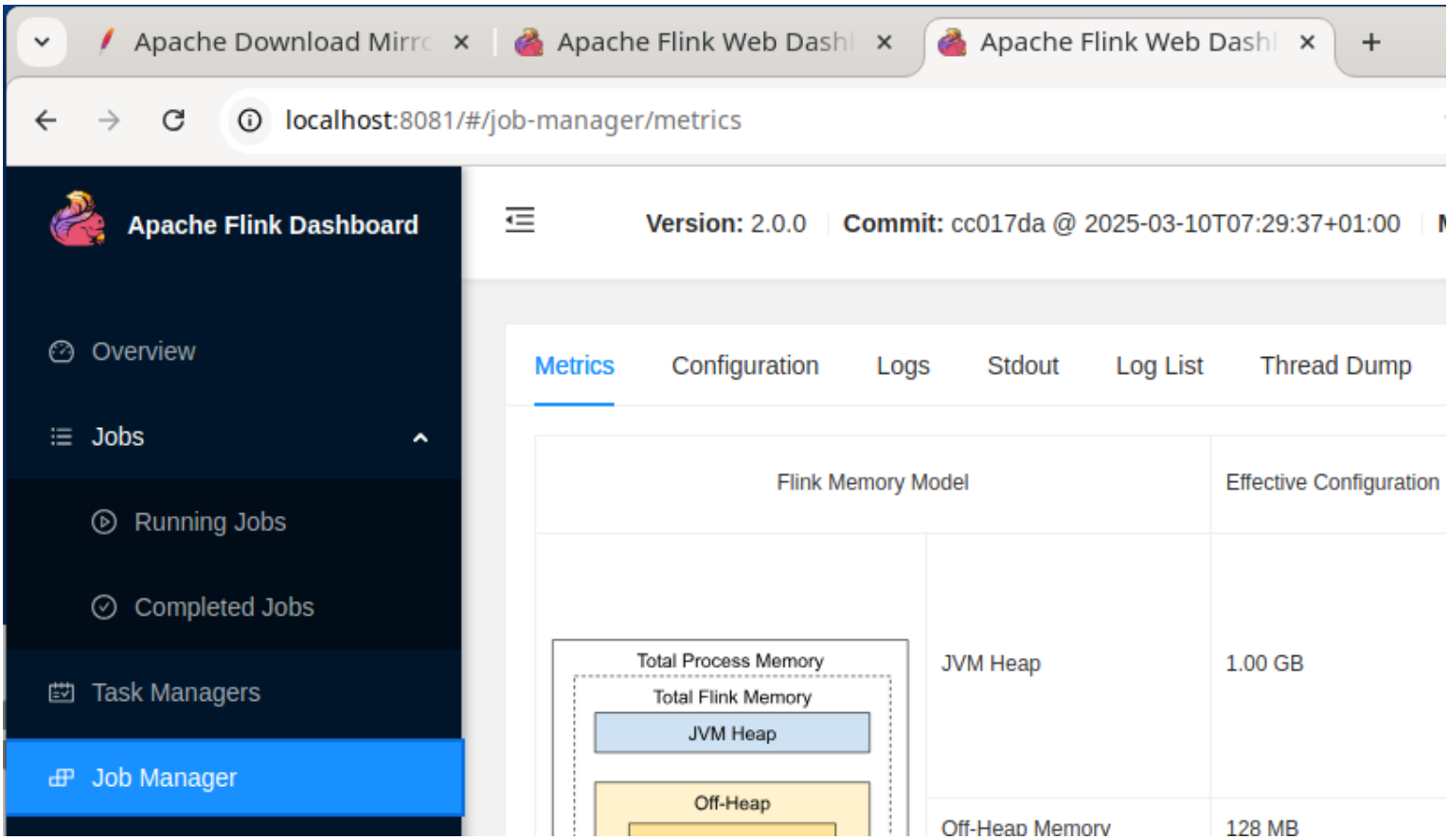
1.1.7 Wait for the Search Box at the top of the Ubuntu Desktop, and enter Chrome, click on the Launch Logo



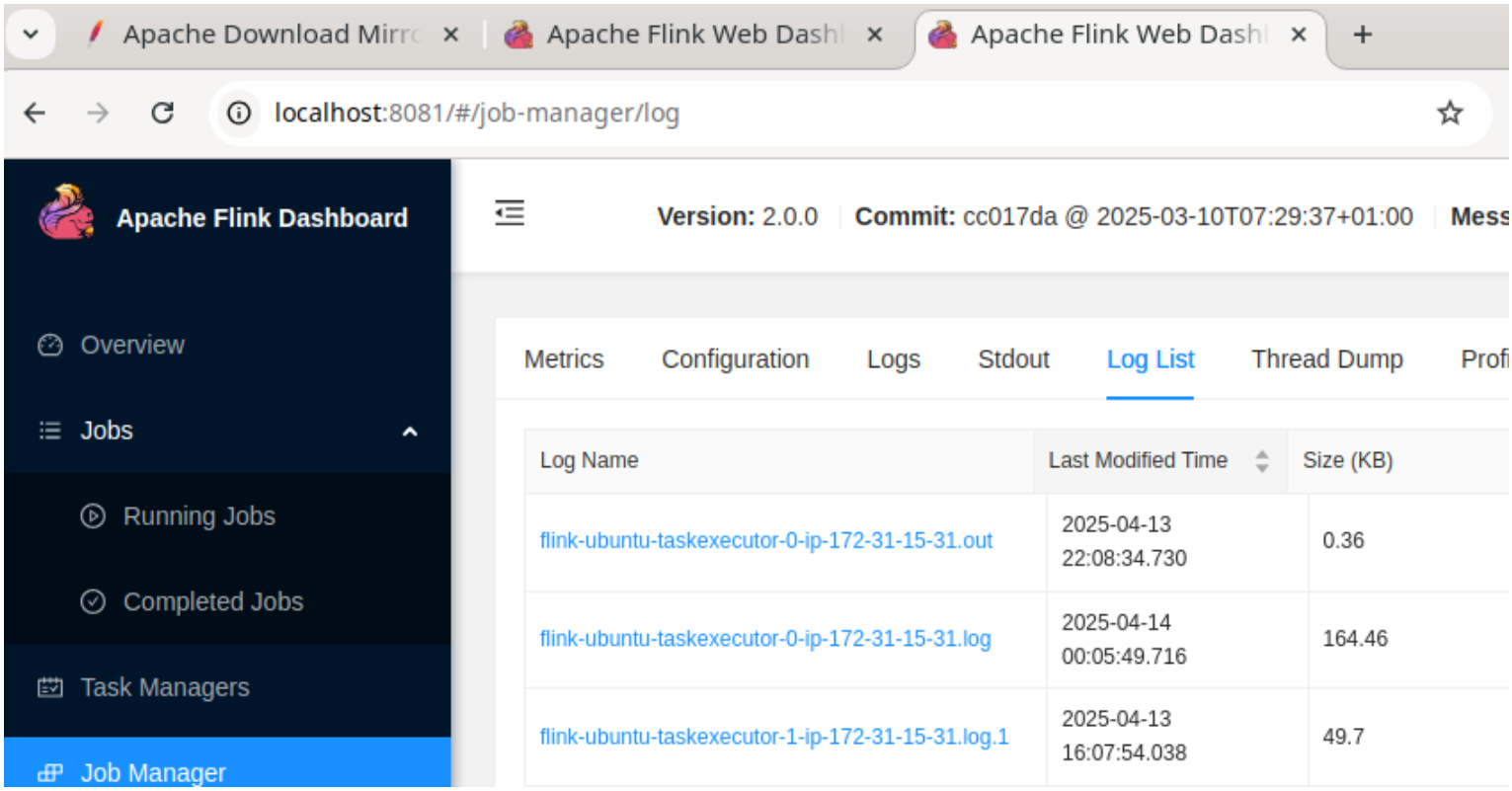
1.1.8 Load the Flink Web UI at localhost:8081



1.1.9 Click the Job Manager in the left hand navigation



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



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/GettingStartedExample.jar

This should show us with the Job submission, Program execution finished, JobID and the Job Runtime, as well as, our Print sink output which from the Table API with Tableau setting will look like this

```
ubuntu@ip-172-31-78-140:~/flink-2.0.0$ ./bin/flink run ~/fli
Job has been submitted with JobID f42aa2a5784d027d13ad0681f6
+-------------------------------+
|                      greeting |
+-------------------------------+
|                   Hello World |
|                      Hi World |
|                   Howdy World |
+-------------------------------+
3 rows in set
```

View the first **flink-ubuntu-taskexecutor*.out** and notice that for this program using the Table API it didn't output to the stdout in our log folder, but rather back to the invocation.

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. **GettingStartedExample** uses the Table API to create a simple Hello World set of text selects from our greeting temporary table.

1.2.2 Our built jar files have the source as noted previously which we can see by examining the source jar.

```
Z:\>jar -tvf c:\users\geo\gettingstartedexample.jar
     0 Tue Apr 15 05:47:48 CDT 2025 META-INF/
   479 Tue Apr 15 05:47:48 CDT 2025 META-INF/MANIFEST.MF
  2154 Tue Apr 15 05:45:10 CDT 2025 example/GettingStartedExample.class
  2826 Sun Apr 13 13:06:06 CDT 2025 example/GettingStartedExample.java
```

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

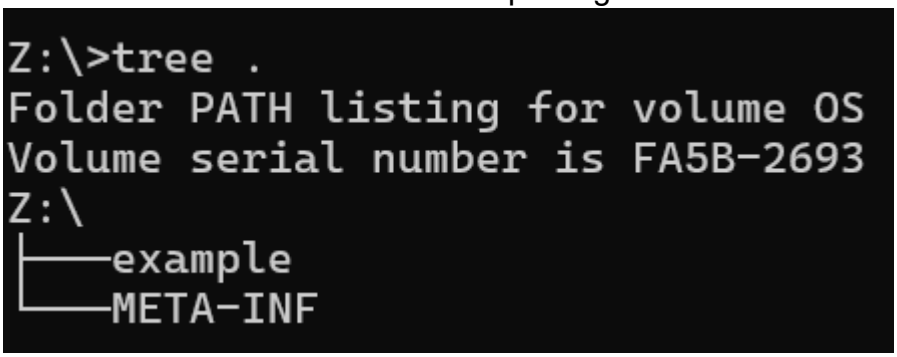
```
javac -classpath C:\lib-2.0\flink-cep-2.0.0.jar;C:\lib-2.0\flink-
connector-files-2.0.0.jar;C:\lib-2.0\flink-csv-2.0.0.jar;C:\lib-
2.0\flink-dist-2.0.0.jar;C:\lib-2.0\flink-json-2.0.0.jar;C:\lib-
2.0\flink-scala_2.12-2.0.0.jar;C:\lib-2.0\flink-table-api-java-
uber-2.0.0.jar;C:\lib-2.0\flink-table-planner-loader-
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/GettingStartedExample.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 Table
Implementation-Version: 2.0.0
Archiver-Version: Plexus Archiver
Built-By: geoniece
Specification-Vendor: Innovation in Software
Specification-Title: Flink : Examples : Simple Table
Implementation-Vendor-Id: com.innovationinsoftware
program-class: example.GettingStartedExample
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



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\GettingStartedExample.jar example/***

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