Lab : Apache Spark WebUI

Pre-reqs:

- Google Chrome (Recommended)

Prerequisites

We need following packages to perform the lab exercise:

- Java Development Kit
- pyspark

JAVA

Verify the installation with: `java -version`

You'll see the following output:

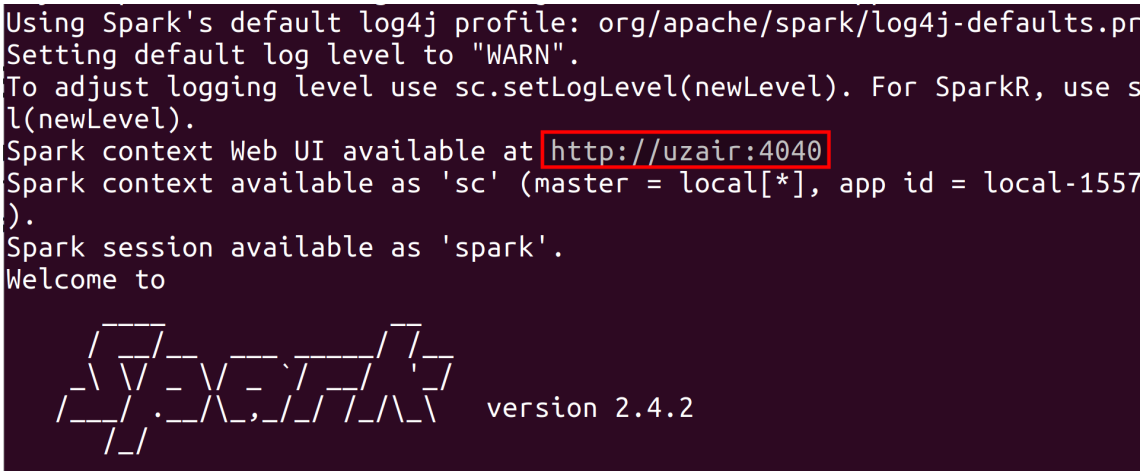
```
java version "1.8.0_201"
Java(TM) SE Runtime Environment (build 1.8.0_201-b09)
Java HotSpot(TM) 64-Bit Server VM (build 25.201-b09, mixed mode)
```

Task: Spark Web Interface

Step 1: Open a terminal and start the spark-shell by entering the following command.

`spark-shell`

The Spark shell should show you that the web interface is available `locally` at the following URL as shown below.



Your driverHostname might be different. If a port is being used by another application, Spark will increase the port by 1 until an open port is found. For example, if 4040 is already taken, it will increase the port number to 4041.

Spark Web Interface

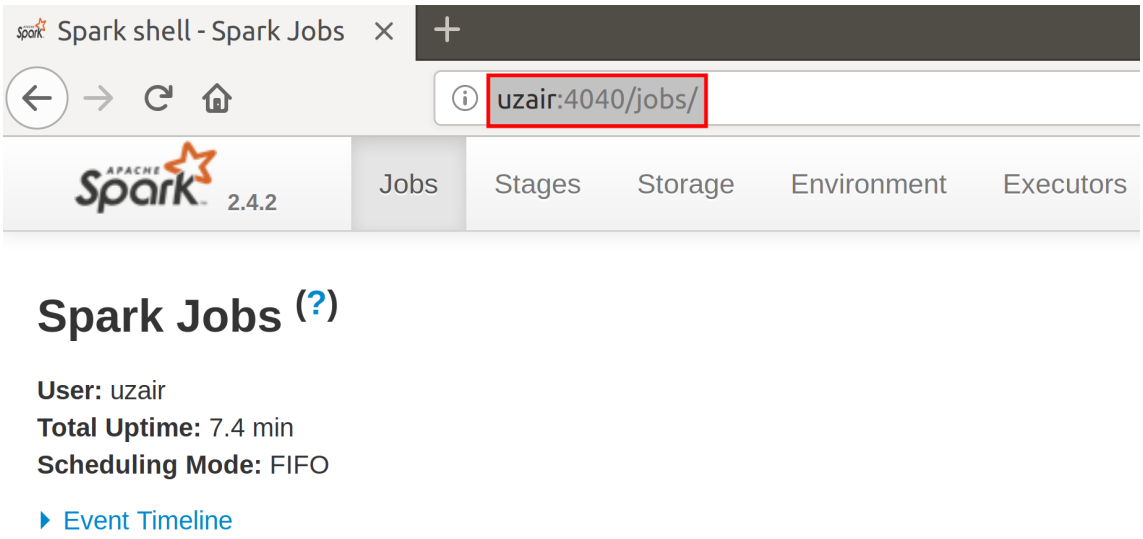
We can access the Spark web interface to monitor the execution of Spark applications through a web browser. The web interface can by accessed by navigating to the following URL. First, update host-ip with ip address of the host

machine where **jupyterLab** container is running:

Spark UI: `http://<host-ip>:4040`

The driverhostname is usually an IP address in the realtime environment and 4040 is the Spark's port by default.

Step 2: Once you navigate to the web interface URL. You should see the Spark web interface as shown in the screenshot below.



Since there is no job running, you won't be able to see any metrics.

Run Job

Step 3: Let us run a job. Create a List of few numbers and create an RDD from that list as shown below.

Note: You might need to get back to prompt by pressing `Enter`.

```
19/10/05 16:25:21 WARN HttpParser: bad HTTP parsed: 400 Illegal character 0x16 for
HttpChannelOverHttp@3006dd4d{r=0,c=false,a=IDLE,uri=null}

scala>
```

```
val num = List(1, 2, 3, 4)
```

```
val numRDD = sc.parallelize(num)
```

Now let us write a map function which takes the numRDD and gives a squaredRDD as shown below.

```
val squaredRDD = numRDD.map(x => x * x)
```

```
squaredRDD.foreach(println)
```

After you see the output in the console, navigate back to the browser and refresh the Spark web interface. You should see a completed job as shown in the screenshot below.

Spark Jobs (?)

User: uzair
Total Uptime: 22 min
Scheduling Mode: FIFO
Completed Jobs: 1

Event Timeline

Completed Jobs (1)

Job Id ▼	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
0	collect at <console>:26 collect at <console>:26	2019/05/11 11:13:41	0.9 s	1/1	1/1

DAG Visualization

Step 4: You can click on the collect link below the Description column and you will be taken to stages. Click on the collect link again to check more information as shown in the screenshot below.

Details for Stage 0 (Attempt 0)

Total Time Across All Tasks: 65 ms
Locality Level Summary: Process local: 1

DAG Visualization
Show Additional Metrics
Event Timeline

Summary Metrics for 1 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	65 ms	65 ms	65 ms	65 ms	65 ms
GC Time	0 ms	0 ms	0 ms	0 ms	0 ms

Aggregated Metrics by Executor

Executor ID ▲	Address	Task Time	Total Tasks	Failed Tasks	Killed Tasks	Succeeded Tasks	Blacklisted
driver	uzair:33167	0.3 s	1	0	0	1	false

Tasks (1)

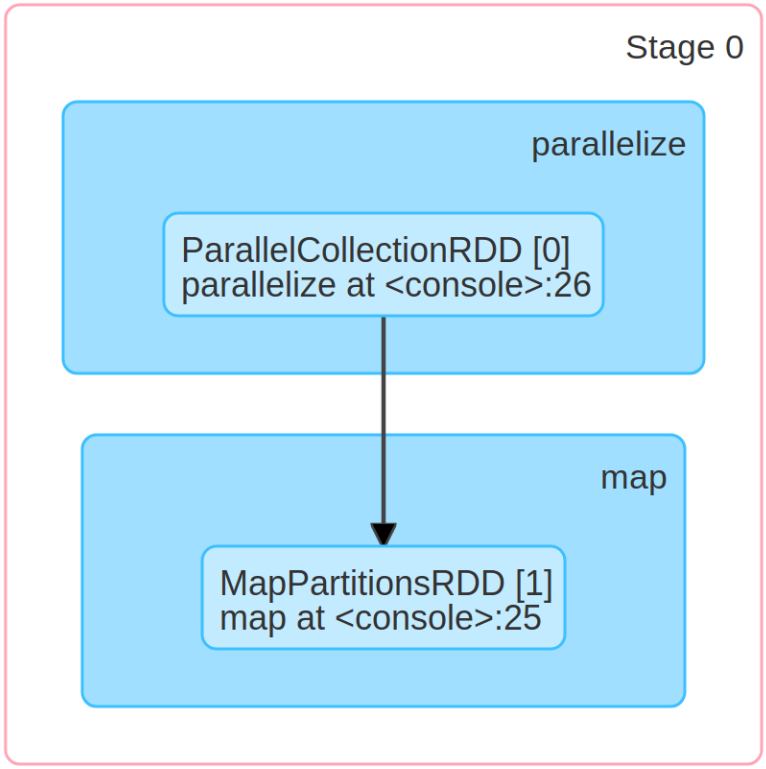
Index ▲	ID	Attempt	Status	Locality Level	Executor ID	Host	Launch Time	Duration	GC Time	Errors
0	0	0	SUCCESS	PROCESS_LOCAL	driver	localhost	2019/05/11 11:13:42	65 ms		

Step 5: Click on the DAG Visualization link to view the DAG.

Details for Stage 0 (Attempt 0)

Total Time Across All Tasks: 65 ms
Locality Level Summary: Process local: 1

▼ DAG Visualization



Executors

Click on the `Executors` link in the navigation bar to monitor the executors.

Executors

Summary

	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Blacklisted
Active(1)	0	1.8 KB / 434 MB	0.0 B	1	0	0	1	1	0.3 s (0 ms)	0.0 B	0.0 B	0.0 B	0
Dead(0)	0	0.0 B / 0.0 B	0.0 B	0	0	0	0	0	0 ms (0 ms)	0.0 B	0.0 B	0.0 B	0
Total(1)	0	1.8 KB / 434 MB	0.0 B	1	0	0	1	1	0.3 s (0 ms)	0.0 B	0.0 B	0.0 B	0

Executors

Show 20 entries Search:

Executor ID	Address	Status	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Thread Dump
driver	uzair:33167	Active	0	1.8 KB / 434 MB	0.0 B	1	0	0	1	1	0.3 s (0 ms)	0.0 B	0.0 B	0.0 B	Thread Dump

Task is complete. We have seen the Spark architecture in detail by discussing the Lineage Graph and DAG.