

Apache Kafka Installation

Version of Apache Hadoop 3.2.4

Version of Apache Kafka 2.8.0

Version of Scala 2.13

Download the Apache kafka and scala using the below mentioned link

https://archive.apache.org/dist/kafka/2.8.0/kafka_2.13-2.8.0.tgz<https://downloads.lightbend.com/scala/2.13.0/scala-2.13.0.tgz>

\$ tar -zxf kafka_2.13-2.8.0.tgz

\$ tar -zxf scala-2.13.0.tgz

\$ su

mv kafka_2.13-2.8.0 /usr/local/kafka

mv scala-2.13.0 /usr/local/scala

vim /etc/bash.bashrc

append the following lines to the end to the file

SCALA CONFIG

SCALA_HOME=/usr/local/scala

export PATH=\$PATH:\$SCALA_HOME/bin

<save and exit>

\$

```

hadoop@mainserver1:~$
hadoop@mainserver1:~$ sudo nano /etc/systemd/system/zookeeper.service
[sudo] password for hadoop:
hadoop@mainserver1:~$ sudo systemctl enable zookeeper.service
Created symlink /etc/systemd/system/multi-user.target.wants/zookeeper.service → /etc/systemd/system/zookeeper.service.
hadoop@mainserver1:~$ sudo systemctl start zookeeper.service
hadoop@mainserver1:~$ sudo systemctl status zookeeper.service
● zookeeper.service - Apache Zookeeper server
   Loaded: loaded (/etc/systemd/system/zookeeper.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2024-01-11 04:55:33 UTC; 10s ago
     Docs: http://zookeeper.apache.org
  Main PID: 2171 (java)
    Tasks: 25 (limit: 4595)
   Memory: 54.5M
    CGroup: /system.slice/zookeeper.service
            └─2171 java -Xmx512M -Xms512M -server -XX:+UseG1GC -XX:MaxGCPauseMillis=20 -XX:InitiatingHe>

Jan 11 04:55:38 mainserver1 zookeeper-server-start.sh[2171]: [2024-01-11 04:55:38,720] INFO maxSessionTi>
Jan 11 04:55:38 mainserver1 zookeeper-server-start.sh[2171]: [2024-01-11 04:55:38,722] INFO Created serv>
Jan 11 04:55:38 mainserver1 zookeeper-server-start.sh[2171]: [2024-01-11 04:55:38,748] INFO Using org.ap>
Jan 11 04:55:38 mainserver1 zookeeper-server-start.sh[2171]: [2024-01-11 04:55:38,759] INFO Configuring>
Jan 11 04:55:38 mainserver1 zookeeper-server-start.sh[2171]: [2024-01-11 04:55:38,787] INFO binding to p>
Jan 11 04:55:38 mainserver1 zookeeper-server-start.sh[2171]: [2024-01-11 04:55:38,856] INFO zookeeper.sn>
Jan 11 04:55:38 mainserver1 zookeeper-server-start.sh[2171]: [2024-01-11 04:55:38,867] INFO Snapshotting>
Jan 11 04:55:38 mainserver1 zookeeper-server-start.sh[2171]: [2024-01-11 04:55:38,876] INFO Snapshotting>
Jan 11 04:55:38 mainserver1 zookeeper-server-start.sh[2171]: [2024-01-11 04:55:38,955] INFO PrepRequestP>
Jan 11 04:55:38 mainserver1 zookeeper-server-start.sh[2171]: [2024-01-11 04:55:38,975] INFO Using checkI>
lines 1-20/20 (END)

```

\$ sudo nano /etc/systemd/system/zookeeper.service

[Unit]

Description=Apache Zookeeper server

Documentation=http://zookeeper.apache.org

Requires=network.target remote-fs.target

After=network.target remote-fs.target

[Service]

Type=simple

ExecStart=/usr/local/kafka/bin/zookeeper-server-start.sh /usr/local/kafka/config/zookeeper.properties

ExecStop=/usr/local/kafka/bin/zookeeper-server-stop.sh

Restart=on-abnormal

[Install]

WantedBy=multi-user.target

<save and exit>

```
$ sudo systemctl enable zookeeper.service
$ sudo systemctl start zookeeper.service
$ sudo systemctl status zookeeper.service
```

```
$ sudo nano /etc/systemd/system/kafka.service
```

[Unit]

Description=Apache Kafka Server

Documentation=http://kafka.apache.org/documentation.html

Requires=zookeeper.service

[Service]

Type=simple

Environment="JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64"

ExecStart=/usr/local/kafka/bin/kafka-server-start.sh /usr/local/kafka/config/server.properties

ExecStop=/usr/local/kafka/bin/kafka-server-stop.sh

[Install]

WantedBy=multi-user.target

```
hadoop@mainserver1:~$ sudo nano /etc/systemd/system/kafka.service
hadoop@mainserver1:~$ ^C
hadoop@mainserver1:~$ sudo systemctl enable kafka.service
Created symlink /etc/systemd/system/multi-user.target.wants/kafka.service → /etc/systemd/system/kafka.service.
hadoop@mainserver1:~$ sudo systemctl start kafka.service
hadoop@mainserver1:~$ sudo systemctl status kafka.service
● kafka.service - Apache Kafka Server
   Loaded: loaded (/etc/systemd/system/kafka.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2024-01-11 05:01:42 UTC; 6s ago
     Docs: http://kafka.apache.org/documentation.html
    Main PID: 2598 (java)
      Tasks: 16 (limit: 4595)
     Memory: 32.3M
    CGroup: /system.slice/kafka.service
            └─2598 /usr/lib/jvm/java-11-openjdk-amd64/bin/java -Xmx1G -Xms1G -server -XX:+UseG1GC -XX:M
Jan 11 05:01:42 mainserver1 systemd[1]: Started Apache Kafka Server.
hadoop@mainserver1:~$
```

```
$ sudo systemctl enable kafka.service
$ sudo systemctl start kafka.service
$ sudo systemctl status kafka.service
$ ss -ant
```

kafka port

zookeeper port

```
hadoop@mainserver1:~$ ss -ant
State      Recv-Q    Send-Q    Local Address:Port    Peer Address:Port    Process
LISTEN     0          70        127.0.0.1:33060        0.0.0.0:*
LISTEN     0          151       127.0.0.1:3306        0.0.0.0:*
LISTEN     0          4096     127.0.0.53%lo:53      0.0.0.0:*
LISTEN     0          128        0.0.0.0:22           0.0.0.0:*
ESTAB      0          0    192.168.56.100:22    192.168.56.1:12309
ESTAB      0          0    192.168.56.100:22    192.168.56.1:13893
LISTEN     0          50          *:9092               *:
LISTEN     0          50          *:37861              *:
LISTEN     0          50          *:2181               *:
LISTEN     0          50          *:37301              *:
LISTEN     0          128        [::]:22              [::]:
ESTAB      0          0    [::ffff:127.0.0.1]:2181 [::ffff:127.0.0.1]:53066
ESTAB      0          0    [::ffff:127.0.0.1]:9092 [::ffff:127.0.0.1]:48822
ESTAB      0          0    [::ffff:127.0.0.1]:48822 [::ffff:127.0.1.1]:9092
ESTAB      0          0    [::ffff:127.0.0.1]:53066 [::ffff:127.0.0.1]:2181
hadoop@mainserver1:~$
```

- When all the components are installed, we need to create a topic and try to send a message.
- In kafka a topic is a fundamental unit used to organize messages
- Each topic should have a unique name across a cluster.
- Topics allows users to send and read data between kafka servers.
- We can create as many clusters as possible in kafka.
- Default port of zookeeper is 2181 and default port for kafka is 9092

Testing scala - This is used to check scala REPL (Read-Eval-Print-Loop)

```
curl -fL https://github.com/coursier/coursier/releases/latest/download/cs-x86_64-pc-linux.gz | gzip -d > cs && chmod +x cs
&& ./cs setup
```

Analysing CSV data using pyspark

We are using the titanic.csv file present in our home directory

```
$ Vim titanic_proj.py
from pyspark.sql import SparkSession
from pyspark.sql.functions import col

# Step 1: Create a Spark session
spark = SparkSession.builder.appName("TitanicAnalysis").getOrCreate()

# Step 2: Load the Titanic dataset into a PySpark DataFrame
titanic_df = spark.read.csv("/home/hadoop/titanic.csv", header=True, inferSchema=True)

# Step 3: Explore the data
print("Number of rows: ", titanic_df.count())
print("Schema: ")
titanic_df.printSchema()

# Display the data present in the file
titanic_df.show()

# Stop the spark session
spark.stop()

$ spark-submit ./titanic_proj.py
```

Sample output :

```
24/01/11 07:00:13 INFO TaskSchedulerImpl: Removed TaskSet 4.0, whose tasks have all completed, from pool
24/01/11 07:00:13 INFO DAGScheduler: ResultStage 4 (count at NativeMethodAccessorImpl.java:0) finished in
0.399 s
24/01/11 07:00:13 INFO DAGScheduler: Job 3 is finished. Cancelling potential speculative or zombie tasks
for this job
24/01/11 07:00:13 INFO TaskSchedulerImpl: Killing all running tasks in stage 4: Stage finished
24/01/11 07:00:13 INFO DAGScheduler: Job 3 finished: count at NativeMethodAccessorImpl.java:0, took 0.443
568 s
Number of rows: 891
Schema:
root
 |-- PassengerId: integer (nullable = true)
 |-- Survived: integer (nullable = true)
 |-- Pclass: integer (nullable = true)
 |-- Name: string (nullable = true)
 |-- Sex: string (nullable = true)
 |-- Age: double (nullable = true)
 |-- SibSp: integer (nullable = true)
 |-- Parch: integer (nullable = true)
 |-- Ticket: string (nullable = true)
 |-- Fare: double (nullable = true)
 |-- Cabin: string (nullable = true)
 |-- Embarked: string (nullable = true)

24/01/11 07:00:14 INFO FileSourceStrategy: Pushed Filters:
24/01/11 07:00:14 INFO FileSourceStrategy: Post-Scan Filters:
24/01/11 07:00:14 INFO FileSourceStrategy: Output Data Schema: struct<PassengerId: int, Survived: int, Pc
lass: int, Name: string, Sex: string ... 10 more fields>
```

```
192.168.56.100 - Putty
|      16|      1|      2|Hewlett, Mrs. (Ma...|female|55.0|      0|      0|      248706|      16.0| null|
|      S|
|      17|      0|      3|Rice, Master. Eugene| male| 2.0|      4|      1|      382652|      29.125| null|
|      Q|
|      18|      1|      2|Williams, Mr. Cha...| male|null|      0|      0|      244373|      13.0| null|
|      S|
|      19|      0|      3|Vander Planke, Mr...|female|31.0|      1|      0|      345763|      18.0| null|
|      S|
|      20|      1|      3|Masselmani, Mrs. ...|female|null|      0|      0|      2649|      7.225| null|
|      C|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
-----+
only showing top 20 rows

24/01/11 07:00:15 INFO SparkUI: Stopped Spark web UI at http://10.0.2.15:4040
24/01/11 07:00:15 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
24/01/11 07:00:15 INFO MemoryStore: MemoryStore cleared
24/01/11 07:00:15 INFO BlockManager: BlockManager stopped
24/01/11 07:00:15 INFO BlockManagerMaster: BlockManagerMaster stopped
24/01/11 07:00:15 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator s
topped!
24/01/11 07:00:15 INFO SparkContext: Successfully stopped SparkContext
24/01/11 07:00:15 INFO ShutdownHookManager: Shutdown hook called
24/01/11 07:00:15 INFO ShutdownHookManager: Deleting directory /tmp/spark-3fa91d31-7bed-48d6-8801-2d04f4a
5ef1e
24/01/11 07:00:15 INFO ShutdownHookManager: Deleting directory /tmp/spark-3fa91d31-7bed-48d6-8801-2d04f4a
5ef1e/pyspark-e05eb3f7-d2cd-4b41-a31f-244400a77cb2
24/01/11 07:00:15 INFO ShutdownHookManager: Deleting directory /tmp/spark-a1dab337-ae42-4645-bd9f-17db82a
2a820
```

Alter the file titanic_proj.py, run the script and view the output

```
# Step 4: if you need to display only few rows or eg : 5 rows only we can alter it as
titanic_df.show(5)

# Step 5: Let us perform the data cleaning and preprocessing
```

```
# drop irrelevant columns and handle missing values
titanic_df = titanic_df.drop("Cabin") # Drop the cabin column
titanic_df = titanic_df.dropna() # Drop the rows with missing null values
```

<Save and exit>

Alter the file again with the below parameters

```
# Step 5: Basic Analysis
# Count the number of survivors by gender
survivors_by_gender =
titanic_df.groupBy("Sex").agg({"Survived": "sum"}).withColumnRenamed("sum(Survived)", "SurvivorCount")
```

```
survivors_by_gender.show()
```

Sample output:

```
24/01/11 09:10:49 INFO TaskSchedulerImpl: Removed TaskSet 7.0, whose tasks have all completed, from pool
24/01/11 09:10:49 INFO DAGScheduler: ResultStage 7 (showString at NativeMethodAccessorImpl.java:0) finished in 0.590 s
24/01/11 09:10:49 INFO DAGScheduler: Job 5 is finished. Cancelling potential speculative or zombie tasks for this job
24/01/11 09:10:49 INFO TaskSchedulerImpl: Killing all running tasks in stage 7: Stage finished
24/01/11 09:10:49 INFO DAGScheduler: Job 5 finished: showString at NativeMethodAccessorImpl.java:0, took 0.688359 s
24/01/11 09:10:49 INFO CodeGenerator: Code generated in 84.442611 ms
+-----+-----+
| Sex|SurvivorCount|
+-----+-----+
|female|      195|
| male|       93|
+-----+-----+

24/01/11 09:10:50 INFO FileSourceStrategy: Pushed Filters:
24/01/11 09:10:50 INFO FileSourceStrategy: Post-Scan Filters: atleastnonnulls(11, PassengerId#16, Survived#17, Pclass#18, Name#19, Sex#20, Age#21, SibSp#22, Parch#23, Ticket#24, Fare#25, Embarked#27)
24/01/11 09:10:50 INFO FileSourceStrategy: Output Data Schema: struct<PassengerId: int, Survived: int, Pclass: int, Name: string, Sex: string ... 9 more fields>
24/01/11 09:10:50 INFO CodeGenerator: Code generated in 266.508376 ms
```

STEP 6: Calculating the average age of a passenger

```
average_age = titanic_df.select("Pclass", "Age").groupBy("Pclass").agg({"Age": "avg"}).withColumnRenamed("avg(Age)",
"AverageAge")
average_age.show()
```

Sample output:

```
24/01/11 09:23:29 INFO DAGScheduler: ResultStage 10 (showString at NativeMethodAccessorImpl.java:0) finished in 0.148 s
24/01/11 09:23:29 INFO DAGScheduler: Job 7 is finished. Cancelling potential speculative or zombie tasks for this job
24/01/11 09:23:29 INFO TaskSchedulerImpl: Killing all running tasks in stage 10: Stage finished
24/01/11 09:23:29 INFO DAGScheduler: Job 7 finished: showString at NativeMethodAccessorImpl.java:0, took 0.188337 s
+-----+-----+
|Pclass|      AverageAge|
+-----+-----+
|      1|38.10554347826087|
|      3|25.14061971830986|
|      2|29.87763005780347|
+-----+-----+

24/01/11 09:23:30 INFO FileSourceStrategy: Pushed Filters:
24/01/11 09:23:30 INFO FileSourceStrategy: Post-Scan Filters: atleastnonnulls(11, PassengerId#16, Survived#17, Pclass#18, Name#19, Sex#20, Age#21, SibSp#22, Parch#23, Ticket#24, Fare#25, Embarked#27)
24/01/11 09:23:30 INFO FileSourceStrategy: Output Data Schema: struct<PassengerId: int, Survived: int, Pclass: int, Name: string, Sex: string ... 9 more fields>
24/01/11 09:23:30 INFO CodeGenerator: Code generated in 56.384357 ms
```

Creating an Airflow DAG - to schedule and execute the script as a Directed Acyclic Graph and we shall save the results to another file.

We will be using the existing titanic.csv dataset and the previously written script in this exercise path where the titanic dataset is present /home/hadoop/titanic.csv

```
$ nohup airflow scheduler > scheduler.log 2>&1 &
```

```
$ nohup airflow webserver -p 8080 > webserver.log 2>&1 &
```

```
$ mkdir AirflowDAG
$ cd AirflowDAG
```

```
$ vim titanic_analysis.py
```

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col
```

```
# Step 1: Create a Spark session
```

```
spark = SparkSession.builder.appName("TitanicAnalysis").getOrCreate()
```

```
# Step 2: Load the Titanic dataset into a PySpark DataFrame
```

```
titanic_df = spark.read.csv("/home/hadoop/spark/AirflowDAG/titanic.csv", header=True, inferSchema=True)
```

```

# Step 3: Explore the data
print("Number of rows: ", titanic_df.count())
print("Schema: ")
titanic_df.printSchema()

# Step 4: Let us perform the data cleaning and preprocessing
# drop irrelevant columns and handle missing values
# Drop the cabin column
titanic_df = titanic_df.drop("Cabin")

# Drop the rows with missing null values
titanic_df = titanic_df.dropna()

# Step 5: Basic Analysis
# Count the number of survivors by gender
survivors_by_gender =
titanic_df.groupBy("Sex").agg({"Survived": "sum"}).withColumnRenamed("sum(Survived)", "SurvivorCount")
survivors_by_gender.show()

# Save the results to another file (in csv format itself)
survivors_by_gender.write.csv("/home/hadoop/spark/AirflowDAG/survivors_by_gender.csv", header=True, mode="overwrite")

# STEP 6: Calculating the average age of a passenger
average_age = titanic_df.select("Pclass", "Age").groupBy("Pclass").agg({"Age": "avg"}).withColumnRenamed("avg(Age)",
"AverageAge")
average_age.show()

# Save the results to another file (in csv format itself)
average_age.write.csv("/home/hadoop/spark/AirflowDAG/average_age.csv", header=True, mode="overwrite")

# STEP 6: Display the data present in the file
titanic_df.show()
# -- this will display by default 20 rows

# Step 6-1: if you need to display only few rows or eg : 5 rows only we can alter it as
# titanic_df.show(5)

# Step 7: Stop the spark session
spark.stop()

<save and exit>

# vim titanic_analysis_dag.py

from datetime import datetime, timedelta
from airflow import DAG
from airflow.operators.bash_operator import BashOperator

# Default arguments for the DAG
default_args = {
    'owner': 'airflow',
    'depends_on_past': False,
    'start_date': datetime(2024, 1, 11),
    'email_on_failure': False,
    'email_on_retry': False,
    'retries': 1,
    'retry_delay': timedelta(minutes=5),
}

# Create a DAG
dag = DAG(
    'titanic_analysis_dag',
    default_args=default_args,
    description='Perform Titanic data analysis and save results',
    schedule_interval=timedelta(days=1), # Set your desired schedule
)

# Task to run the PySpark script
run_spark_task = BashOperator(
    task_id='run_spark_task',
    bash_command='spark-submit /home/hadoop/spark/AirflowDAG/titanic_analysis.py',
    dag=dag,
)

# Define task dependencies
run_spark_task

<save and exit>

Final output

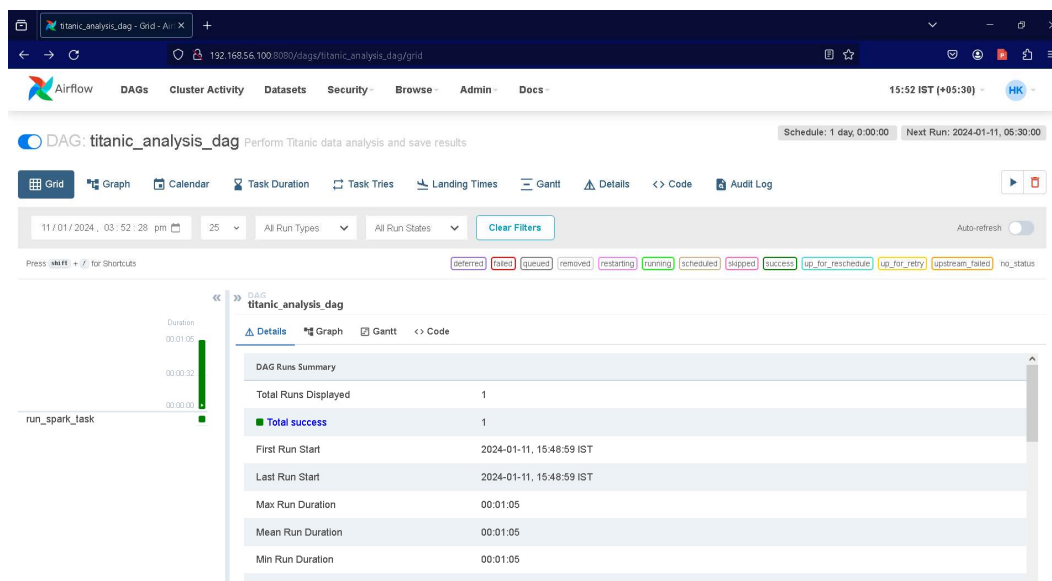
```

```

root@mainserver1: /home/hadoop/spark/AirflowDAG# tree
.
├── average_age.csv
│   ├── part-00000-b347010b-7c01-4d52-b3b1-ac6d593b647c-c000.csv
│   └── SUCCESS
├── scheduler.log
├── survivors_by_gender.csv
│   ├── part-00000-972df388-eabf-49b9-abe2-7cf690a362cc-c000.csv
│   └── SUCCESS
├── titanic_analysis_dag.py
├── titanic_analysis.py
├── titanic.csv
└── webserver.log

2 directories, 9 files
root@mainserver1: /home/hadoop/spark/AirflowDAG# cat average_age.csv/part-00000-b347010b-7c01-4d52-b3b1-ac6d593b647c-c000.csv
Polass,AverageAge
1,38.10554347826087
3,25.14061971830986
2,29.87763005780347
root@mainserver1: /home/hadoop/spark/AirflowDAG# cat survivors_by_gender.csv/part-00000-972df388-eabf-49b9-abe2-7cf690a362cc-c000.csv
Sex,SurvivorCount
female,195
male,93
root@mainserver1: /home/hadoop/spark/AirflowDAG#

```



Creating RDDs:

- Parallelize: Creating an RDD from a Python list
from pyspark import SparkContext

```
# Create a SparkContext
sc = SparkContext("local", "RDDPersistenceExample")
```

```
# Create an RDD from a list of numbers
data = [1, 2, 3, 4, 5]
rdd = sc.parallelize(data)
```

- Text File: Create an RDD by reading data from a text file
from pyspark import SparkContext

```
sc = SparkContext("local", "RDDExample")
rdd = sc.textFile("path/to/textfile.txt")
```

Transformation Operations : Map, Filter, FlatMap
Action Operations: Collect, Count, Reduce, Take

Transformation and actions -- combined together

Wordcount example which we used earlier

```
words = rdd_flatmap.map(lambda x: (x, 1)) ----> Map, FlatMap
word_counts = words.reduceByKey(add) ----> Reduce By
result_word_counts = word_counts.collect() ----> Collect
```

```

hadoop@bigadmin:~$ sudo curl -fL https://github.com/coursier/coursier/releases/latest/download/cs-x86_64-
pc-linux.gz | gzip -d > cs && chmod +x cs && ./cs setup
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left     Speed
  0      0    0     0    0     0      0      0  --:--:-- --:--:-- --:--:--    0
  0      0    0     0    0     0      0      0  --:--:-- --:--:-- --:--:--    0
100 19.8M 100 19.8M    0     0 2370k      0  0:00:08  0:00:08 --:--:-- 4106k
Checking if a JVM is installed
https://github.com/coursier/jvm-index/raw/master/index.json
100.0% [#####] 1.4 MiB (417.8 KiB / s)
No JVM found, should we try to install one? [Y/n] Y
Should we update ~/.profile? [Y/n] Y
Some shell configuration files were updated. It is recommended to close this terminal once the setup comm
and is done, and open a new one for the changes to be taken into account.

Checking if ~/.local/share/coursier/bin is in PATH
Should we add ~/.local/share/coursier/bin to your PATH via ~/.profile? [Y/n] Y

Checking if the standard Scala applications are installed
Installed ammonite
Installed cs
Installed coursier
Installed scala
Installed scalac
Installed scala-cli
Installed sbt
Installed sbt-n
Installed scalafmt

```