800+ Q&As | Logout | Contact

Java-Success.com

Prepare to fast-track, choose & go places with 800+ Java & Big Data Q&As with lots of code & diagrams.

search here … Go

Home Why? 300+ Java FAQs 300+ Big Data FAQs Courses

Membership Your Career

Home › bigdata-success.com › Tutorials - Big Data › TUT - Cloudera on Docker › 30: Docker Tutorial: Apache Spark streaming in Python 3 with Apache Kafka on Cloudera quickstart

30: Docker Tutorial: Apache Spark streaming in Python 3 with Apache Kafka on Cloudera quickstart

Posted on July 6, 2019

This extends Docker Tutorial: Apache Kafka with Python 3 on Cloudera quickstart

Step 1: Create the pyspark streaming code in python.

driver.py

300+ Java Interview FAQs

300+ Java FAQs 🔥

16+ Java Key Areas Q&As

150+ Java Architect FAQs

80+ Java Code Quality Q&As

150+ Java Coding Q&As

300+ Big Data Interview FAQs

300+ Big Data FAQs 🔥

Tutorials - Big Data

TUT - ⏯ Starting Big Data

TUT - Starting Spark & Scala

```
(my-app_env) [root@quickstart my-app]# vi driver.py
```

```
from pyspark import SparkConf, SparkContext
from pyspark.streaming import StreamingContext
from mypackage import simple

if __name__ == "__main__":
    conf = SparkConf().setAppName("Simple App")
    conf = conf.setMaster("local[*]")
    sc = SparkContext(conf=conf)
    ssc = StreamingContext(sc, 10) # every 10 seco
    simple.SimpleSpark().myfunc(ssc)

    ssc.start();
    ssc.awaitTermination();

```

simple.py – receive messages from a topic

```
(my-app_env) [root@quickstart my-app]# vi mypackage
```

```
from pyspark.streaming import StreamingContext
from pyspark.streaming.kafka import KafkaUtils

class SimpleSpark:

    def myfunc(self, ssc):
        # 2181 is the zookeeper port & 9092 is the
        kafkaStream = KafkaUtils.createStream(ssc
        lines = kafkaStream.map(lambda x : x[1])
        lines.pprint()

```

Build the .egg file

Step 2: Create setup.py to build the .egg file, which is a zip file.

```
(my-app_env) [root@quickstart my-app]# vi setup.py
```

```
from setuptools import setup

setup(
    name = 'simple-spark',
```

800+ Java Interview Q&As

```
    author = 'java-success',
    packages=['mypackage'],
    # Whatever arguments you need/want
)

```

Step 3: Build and install the .egg file package.

```
(my-app_env) [root@quickstart my-app]# python setu

```

```
(my-app_env) [root@quickstart my-app]# pip freeze
kafka-python==1.4.6
simple-producer==0.0.0
simple-spark==0.0.0

```

```
(my-app_env) [root@quickstart my-app]# pip list
DEPRECATION: The default format will switch to colu
kafka-python (1.4.6)
pip (9.0.1)
setuptools (28.8.0)
simple-producer (0.0.0)
simple-spark (0.0.0)

```

Run spark-submit

Step 4: Run the Spark streaming job.This will consume messages from the Kafka topic “MyTestTopic”

```
(my-app_env) [root@quickstart my-app]# spark-submi
--packages org.apache.spark:spark-streaming-kafka_
--py-files dist/simple_spark-0.0.0-py3.4.egg \
driver.py

```

Produce messages

Step 5: Open a new terminal, and type the following docker command to find the container id.

```
~/projects/docker-hadoop]$ docker ps

CONTAINER ID        IMAGE               COMMAND
86def26a4fd4        gdancik/cloudera    "/usr/bin/
```

Step 6: Login to the container with the following command.

```
~/projects/docker-hadoop]$ docker exec -it 86def26

[root@quickstart /]#
```

Step 7: Produce messages.

```
[root@quickstart /]# ./home/kafka/kafka/bin/kafka-
--broker-list localhost:9092 \
--topic MyTestTopic
```

Start sending messages:

```
>
>Sending a message to the topic
>to see if the spark-streaming picks it up
>
```

Check consumption of the messages

In the terminal where spark is streaming every 10 seconds.

```
-------------------------------------------
Time: 2019-07-06 11:33:10
-------------------------------------------
```

```
Sending a message to the topic

.....
-------------------------------------------
Time: 2019-07-06 11:33:30
-------------------------------------------
to see if the spark-streaming picks it up

```

simple.py – count the words

Here is a varied example that counts the number of words:

```
(my-app_env) [root@quickstart my-app]# vi mypackag

```

```
from pyspark.streaming import StreamingContext
from pyspark.streaming.kafka import KafkaUtils

class SimpleSpark:

    def myfunc(self, ssc):
        # 2181 is the zookeeper port & 9092 is th
        kafkaStream = KafkaUtils.createStream(ssc
        lines = kafkaStream.map(lambda x : x[1])

        counts = lines.flatMap(lambda line: line.
                  .map(lambda word: (word, 1)) \
                  .reduceByKey(lambda a, b: a+b)
        counts.pprint()

```

```
(my-app_env) [root@quickstart my-app]# python setu
```

```
(my-app_env) [root@quickstart my-app]# spark-submi
--verbose \
--packages org.apache.spark:spark-streaming-kafka_
--py-files dist/simple_spark-0.0.0-py3.4.egg \
driver.py

```

Produce a message

from a different terminal

```
[root@quickstart /]# /home/kafka/kafka/bin/kafka-c

```

```
>big brown fox jumped over a brown fence

```

Check the pyspark streaming console:

```
-------------------------------------------
Time: 2019-07-06 11:45:40
-------------------------------------------
('fox', 1)
('jumped', 1)
('brown', 2)
('fence', 1)
('a', 1)
('over', 1)
('big', 1)

```

‹ 29: Docker Tutorial: Apache Kafka with Python 3 on Cloudera quickstart

31: Docker Tutorial: Apache Spark streaming in Scala with Apache Kafka on Cloudera quickstart ›