

Java-Success.com

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

[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

```

1 (my-app_env) [root@quickstart my-app]# vi driver.py
2
3
4
5 from pyspark import SparkConf, SparkContext
6 from pyspark.streaming import StreamingContext
7 from mypackage import simple
8
9 if __name__ == "__main__":
10     conf = SparkConf().setAppName("Simple App")
11     conf = conf.setMaster("local[*]")
12     sc = SparkContext(conf=conf)
13     ssc = StreamingContext(sc, 10) # every 10 seconds
14     simple.SimpleSpark().myfunc(ssc)
15
16     ssc.start();
17     ssc.awaitTermination();
18

```

TUT - Starting with Python

TUT - Kafka

TUT - Pig

TUT - Apache Storm

TUT - Spark Scala on Zeppelin

TUT - Cloudera

TUT - Cloudera on Docker

TUT - File Formats

TUT - Spark on Docker

TUT - Flume

TUT - Hadoop (HDFS)

TUT - HBase (NoSQL)

TUT - Hive (SQL)

TUT - Hadoop & Spark

TUT - MapReduce

TUT - Spark and Scala

TUT - Spark & Java

TUT - PySpark on Databricks

TUT - Zookeeper

simple.py – receive messages from a topic

```

1 (my-app_env) [root@quickstart my-app]# vi mypackage.py
2
3
4
5 from pyspark.streaming import StreamingContext
6 from pyspark.streaming.kafka import KafkaUtils
7
8 class SimpleSpark:
9
10     def myfunc(self, ssc):
11         # 2181 is the zookeeper port & 9092 is the kafka port
12         kafkaStream = KafkaUtils.createStream(ssc, "localhost:2181", "group1", ["topic1"])
13         lines = kafkaStream.map(lambda x : x[1])
14         lines.pprint()
15

```

Build the .egg file

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

```

1 (my-app_env) [root@quickstart my-app]# vi setup.py
2
3
4 from setuptools import setup
5
6 setup(
7     name = 'simple-spark',
8

```

800+ Java Interview Q&As

300+ Core Java Q&As



300+ Enterprise Java Q&As



150+ Java Frameworks Q&As



120+ Companion Tech Q&As



Tutorials - Enterprise Java



```
5     author = 'java-success',
6     packages=['mypackage'],
7     # Whatever arguments you need/want
8 )
9
```

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

```
1 (my-app_env) [root@quickstart my-app]# python setup.py sdist
2
```

```
1 (my-app_env) [root@quickstart my-app]# pip freeze
2 kafka-python==1.4.6
3 simple-producer==0.0.0
4 simple-spark==0.0.0
5
```

```
1 (my-app_env) [root@quickstart my-app]# pip list
2 DEPRECATION: The default format will switch to columns in the future.
3 kafka-python (1.4.6)
4 pip (9.0.1)
5 setuptools (28.8.0)
6 simple-producer (0.0.0)
7 simple-spark (0.0.0)
8
```

Run spark-submit

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

```
1 (my-app_env) [root@quickstart my-app]# spark-submit \
2 --packages org.apache.spark:spark-streaming-kafka_2.12:2.4.0 \
3 --py-files dist/simple_spark-0.0.0-py3.4.egg \
4 driver.py
5
```

Produce messages

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

```
1 ~/projects/docker-hadoop]$ docker ps
2
3 CONTAINER ID        IMAGE               COMMAND
4 86def26a4fd4        gdancik/cloudera   "/usr/bin/c
5
```

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

```
1 ~/projects/docker-hadoop]$ docker exec -it 86def26a4fd4 /bin/bash
2
3 [root@quickstart /]#
4
```

Step 7: Produce messages.

```
1 [root@quickstart /]# ./home/kafka/kafka/bin/kafka-avro-console-producer
2 --broker-list localhost:9092 \
3 --topic MyTestTopic
4
```

Start sending messages:

```
1 >
2 >Sending a message to the topic
3 >to see if the spark-streaming picks it up
4 >
5
6
```

Check consumption of the messages

In the terminal where spark is streaming every 10 seconds.

```
1 -----
2 Time: 2019-07-06 11:33:10
3 -----
4
```

```
5 Sending a message to the topic
6
7 .....
8 -----
9 Time: 2019-07-06 11:33:30
10 -----
11 to see if the spark-streaming picks it up
12
```

simple.py – count the words

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

```
1 (my-app_env) [root@quickstart my-app]# vi mypackage
2
```

```
1 from pyspark.streaming import StreamingContext
2 from pyspark.streaming.kafka import KafkaUtils
3
4 class SimpleSpark:
5
6     def myfunc(self, ssc):
7         # 2181 is the zookeeper port & 9092 is the kafka port
8         kafkaStream = KafkaUtils.createStream(ssc, zkQuorum, group, topics)
9         lines = kafkaStream.map(lambda x : x[1])
10
11         counts = lines.flatMap(lambda line: line.split(" ")) \
12                        .map(lambda word: (word, 1)) \
13                        .reduceByKey(lambda a, b: a+b)
14         counts.pprint()
15
```

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

```
1 (my-app_env) [root@quickstart my-app]# spark-submit \
2 --verbose \
3 --packages org.apache.spark:spark-streaming-kafka_2.10:2.4.0 \
4 --py-files dist/simple_spark-0.0.0-py3.4.egg \
5 driver.py
6
```

Produce a message

from a different terminal

```
1 [root@quickstart /]# /home/kafka/kafka/bin/kafka-c
2
1 >big brown fox jumped over a brown fence
2
```

Check the pyspark streaming console:

```
1 -----
2 Time: 2019-07-06 11:45:40
3 -----
4 ('fox', 1)
5 ('jumped', 1)
6 ('brown', 2)
7 ('fence', 1)
8 ('a', 1)
9 ('over', 1)
10 ('big', 1)
11
```

◀ 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 ▶

Disclaimer

The contents in this Java-Success are copyrighted and from EmpoweringTech pty ltd. The EmpoweringTech pty ltd has the right to correct or enhance the current content without any prior notice. These are general advice only, and one needs to take his/her own circumstances into consideration. The EmpoweringTech pty ltd will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. Links to external sites do not imply endorsement of the linked-to sites. [Privacy Policy](#)