# 23: Docker Tutorial: Apache Spark (spark-submit) in Python 3 with virtual env on Cloudera quickstart

Posted on June 15, 2019

Prerequisite: Docker is installed on your Windows or Mac, and you have a basic understanding of Docker. Docker tutorials step by step | Hadoop, Hive, Impala & Spark on Cloudera quickstart on Docker tutorials

Step 1: Pull the modified image “gdancik/cloudera” of cloudera/quickstart with python3.4 & vim installed.

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

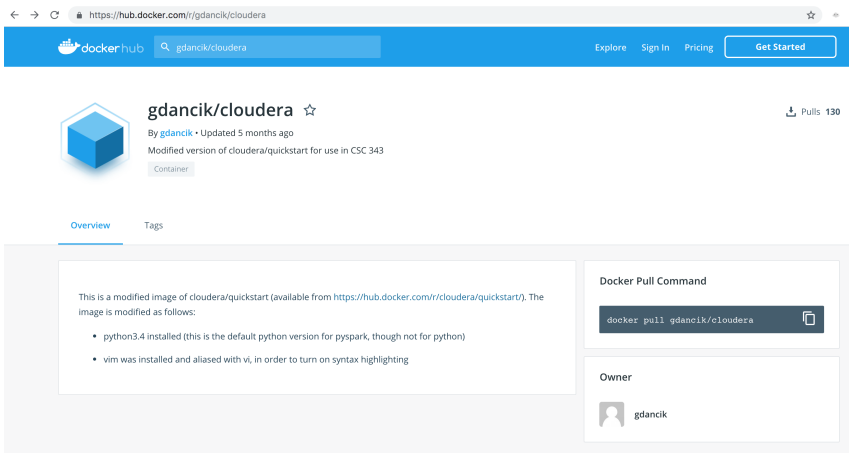
300+ Big Data FAQs 🔥

Tutorials - Big Data

TUT - ⏯ Starting Big Data

TUT - Starting Spark & Scala

vim is aliased with “vi”. This image is availbe via Docker hub – https://hub.docker.com/r/gdancik/cloudera.




Cloudera quickstart Docker with Python 3

```
$ docker pull gdancik/cloudera
```

Step 2: Run the container on a command line.

```
~/projects/docker-hadoop]$ docker run --hostname=qu
--privileged=true -t -i -v /Users/arulkumarankumar
--publish-all=true -p 8888:8888 -p 80:80 -p 7180:7
```

# Python3

Step 3: Configure python3.

Image “**gdancik/cloudera**” comes with python3.

```
[root@quickstart /]# python3 --version
Python 3.4.8
[root@quickstart /]#
```

To use Python3 for pyspark:

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

```
[root@quickstart ~]# export PYSPARK_PYTHON=python3

```

If you want python to point python3

```
[root@quickstart ~]# alias python=python3
[root@quickstart ~]# python --version
Python 3.4.8
[root@quickstart ~]#

```

# Install pip3

Step 4: Install pip3.

```
[root@quickstart /]# sudo yum install python34-setu
[root@quickstart /]# sudo easy_install-3.4 pip

```

```
[root@quickstart /]# pip3 list
DEPRECATION: Python 3.4 support has been deprecated
Package    Version
---------- -------
pip        19.1.1
setuptools 19.6.2
[root@quickstart /]#

```

```
[root@quickstart /]# pip3 freeze

```

# Virtual Environment

Step 5: Install virtualenv.

```
[root@quickstart ~]# sudo pip3 install virtualenv
```

```
[root@quickstart ~]# which virtualenv
/usr/bin/virtualenv

```

```
[root@quickstart ~]# pip3 freeze
```

```
virtualenv==16.6.0

```

```
[root@quickstart ~]# pip3 list
Package    Version
---------- -------
pip        19.1.1
setuptools 19.6.2
virtualenv 16.6.0

```

Step 6: Create “projects/my-app” directory

```
[root@quickstart projects]# mkdir -p /root/project
[root@quickstart projects]# cd !$
cd /root/projects/my-app
[root@quickstart my-app]#

```

Step 7: Create a virtual environment named “my-app-env”.

```
[root@quickstart my-app]# python3 -m venv my-app_e
[root@quickstart my-app]# ls -ltr
total 4
drwxr-xr-x 5 root root 4096 Jun 15 05:20 my-app_en
[root@quickstart my-app]#

```

Step 8: Activate the virtual environment.

```
[root@quickstart my-app]# source my-app_env/bin/ac
(my-app_env) [root@quickstart my-app]#

```

(my-app_env) means we are in “my-app_env” virtual environment. So if you install a package like say “pytest” it will installed in the virtual environment site packages and not in the global site packages.

```
(my-app_env) [root@quickstart my-app]# pip3 instal
....
(my-app_env) [root@quickstart my-app]# pip3 freeze
atomicwrites==1.3.0
attrs==19.1.0
importlib-metadata==0.18
more-itertools==7.0.0
packaging==19.0
pathlib2==2.3.3
pluggy==0.12.0
py==1.8.0
pyparsing==2.4.0
pytest==4.6.3
scandir==1.10.0
six==1.12.0
wcwidth==0.1.7
zipp==0.5.1

```

```
(my-app_env) [root@quickstart my-app]# ls -ltr my-
```

# switch to global

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

```

```
[root@quickstart my-app]# pip3 freeze
virtualenv==16.6.0
[root@quickstart my-app]#

```

# Switch back to virtual env

“history” command to re-run the “source” command with “!”

```
[root@quickstart my-app]# history | grep source
   35  source my-app_env/bin/activate
   51  history | grep source
[root@quickstart my-app]# !35
source my-app_env/bin/activate
(my-app_env) [root@quickstart my-app]#

```

# Create a Python project structure

Step 9: Create the project structure and the relevant python files.

```
(my-app_env) [root@quickstart my-app]# mkdir -p /r
```

## simple.py

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

```

```
from pyspark import SparkConf, SparkContext

class SimpleSpark:

    def myfunc(self, sc):
        rdd = sc.parallelize(["John", "Peter", "Sa
        print(rdd.collect())

```

## driver.py

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

```

```
from pyspark import SparkConf, SparkContext
from mypackage import simple

if __name__ == "__main__":
    conf = SparkConf().setAppName("Simple App")
    conf = conf.setMaster("local[*]")
    sc = SparkContext(conf=conf)
    simple.SimpleSpark().myfunc(sc)


```

## setup.py

setup.py to build .egg (i.e. zip) files containing all the modules. setup.py is a python file, which usually tells you that the module/package you are about to install has been packaged and distributed with Distutils,

which is the standard for distributing Python Modules.

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

```

```
from setuptools import setup

setup(
    name = 'simple-spark',
    author = 'java-success',
    packages=['mypackage'],
    # Whatever arguments you need/want
)

```

## tree -L 4

```
(my-app_env) [root@quickstart my-app]# tree -L 5
.
├── driver.py
├── my-app_env
│   ├── bin
│   │   ├── activate
│   │   ├── activate.csh
│   │   ├── activate.fish
│   │   ├── easy_install
│   │   ├── easy_install-3.4
│   │   ├── pip
│   │   ├── pip3
│   │   ├── pip3.4
│   │   ├── pytest
│   │   ├── py.test
│   │   ├── python -> python3
│   │   └── python3 -> /usr/bin/python3
│   ├── include
│   ├── lib
│   │   └── python3.4
│   │       └── site-packages
│   │           ├── atomicwrites
│   │           ├── atomicwrites-1.3.0.dist-info
│   │           ├── attr
│   │           ├── attrs-19.1.0.dist-info
│   │           ├── easy_install.py
│   │           ├── importlib_metadata
│   │           ├── importlib_metadata-0.18.dist-
│   │           ├── more_itertools
│   │           ├── more_itertools-7.0.0.dist-info
│   │           ├── packaging
│   │           ├── packaging-19.0.dist-info
```

```
│   │           ├── pathlib2
│   │           ├── pathlib2-2.3.3.dist-info
│   │           ├── pip
│   │           ├── pip-9.0.1.dist-info
│   │           ├── pkg_resources
│   │           ├── pluggy
│   │           ├── pluggy-0.12.0.dist-info
│   │           ├── py
│   │           ├── py-1.8.0.dist-info
│   │           ├── __pycache__
│   │           ├── pyparsing-2.4.0.dist-info
│   │           ├── pyparsing.py
│   │           ├── _pytest
│   │           ├── pytest-4.6.3.dist-info
│   │           ├── pytest.py
│   │           ├── scandir-1.10.0-py3.4.egg-info
│   │           ├── scandir.py
│   │           ├── setuptools
│   │           ├── setuptools-28.8.0.dist-info
│   │           ├── six-1.12.0.dist-info
│   │           ├── six.py
│   │           ├── wcwidth
│   │           ├── wcwidth-0.1.7.dist-info
│   │           ├── zipp-0.5.1.dist-info
│   │           └── zipp.py
│   ├── lib64 -> lib
│   ├── pip-selfcheck.json
│   └── pyvenv.cfg
├── mypackage
│   └── simple.py
└── setup.py

38 directories, 23 files
(my-app_env) [root@quickstart my-app]#
```

# Build an .egg file

Step 10: Let's build an .egg file with setup.py in a virtual environment.

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

```

```
(my-app_env) [root@quickstart my-app]# tree -L 2
.
├── build
│   ├── bdist.linux-x86_64
│   └── lib
├── dist
```

```
│   └── simple_spark-0.0.0-py3.4.egg
├── driver.py
├── my-app_env
│   ├── bin
│   ├── include
│   ├── lib
│   ├── lib64 -> lib
│   ├── pip-selfcheck.json
│   └── pyvenv.cfg
├── mypackage
│   └── simple.py
├── setup.py
└── simple_spark.egg-info
    ├── dependency_links.txt
    ├── PKG-INFO
    ├── SOURCES.txt
    └── top_level.txt

11 directories, 10 files
(my-app_env) [root@quickstart my-app]#

```

# View an .egg file

```
(my-app_env) [root@quickstart my-app]# unzip -l d
Archive:  dist/simple_spark-0.0.0-py3.4.egg
  Length      Date    Time    Name
---------  ---------- -----   ----
       10  06-15-2019 07:53   EGG-INFO/top_level.
        1  06-15-2019 07:53   EGG-INFO/zip-safe
      172  06-15-2019 07:53   EGG-INFO/SOURCES.tx
        1  06-15-2019 07:53   EGG-INFO/dependency_
      191  06-15-2019 07:53   EGG-INFO/PKG-INFO
      180  06-15-2019 07:41   mypackage/simple.py
      579  06-15-2019 07:53   mypackage/__pycache_
---------                     -------
     1134                     7 files
(my-app_env) [root@quickstart my-app]#

```

# spark-submit

```
(my-app_env) [root@quickstart my-app]#spark-submit
--verbose \
--py-files dist/simple_spark-0.0.0-py3.4.egg \
driver.py

```

### Outputs:

```
....
['John', 'Peter', 'Samuel']
.....

```

## Create a requirements.txt file

```
(my-app_env) [root@quickstart my-app]# pip3 freeze
```

```
(my-app_env) [root@quickstart my-app]# tree -L 1
.
├── build
├── dist
├── driver.py
├── my-app_env
├── mypackage
├── requirements.txt
├── setup.py
└── simple_spark.egg-info

5 directories, 3 files
(my-app_env) [root@quickstart my-app]#

```

```
(my-app_env) [root@quickstart my-app]# cat requir
atomicwrites==1.3.0
attrs==19.1.0
importlib-metadata==0.18
more-itertools==7.0.0
packaging==19.0
pathlib2==2.3.3
pluggy==0.12.0
py==1.8.0
pyparsing==2.4.0
pytest==4.6.3
scandir==1.10.0
simple-spark==0.0.0
six==1.12.0
wcwidth==0.1.7
zipp==0.5.1

```

## Switch to global environment

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

```

```
 [root@quickstart my-app]# pip3 freeze
virtualenv==16.6.0

```

Let's install the packages from "requirements.txt" file. Remove the line "simple-spark==0.0.0".

```
[root@quickstart my-app]# pip3 install -r requirem

```

Now the global environment will have the following package dependencies.

```
[root@quickstart my-app]# pip3 freeze
atomicwrites==1.3.0
attrs==19.1.0
importlib-metadata==0.18
more-itertools==7.0.0
packaging==19.0
pathlib2==2.3.3
pluggy==0.12.0
py==1.8.0
pyparsing==2.4.0
pytest==4.6.3
scandir==1.10.0
six==1.12.0
virtualenv==16.6.0
wcwidth==0.1.7
zipp==0.5.1
[root@quickstart my-app]#

```

‹ 22: Docker Tutorial: Apache Spark (spark-submit) in Python 2.6 on Cloudera quickstart

24: Docker Tutorial: HBase (i.e. NoSQL DB) Shell on Cloudera quickstart ›

# Disclaimer