# Big Data Engineer Bootcamp

Code 3

# Agenda

- **Dev Environment**
- Work with Docker
- Work with Redis
- Work with Spark

# Requirement

- Docker
- Virtualbox
- Docker-machine
  - 请一定要安装，不然会 导致环境不一致
- wget, tar
  - 不一定需要，可以用手工下 载代替
- Scala, SBT
- Python, Pip

# Requirement

- Start a docker-machine virtual machine called: bigdata
- You can get the ip address of that virtual machine using
  - docker-machine ip bigdata
  - Memorize this address
  - In the following commands, we will use 192.168.99.100 as an example

# Agenda

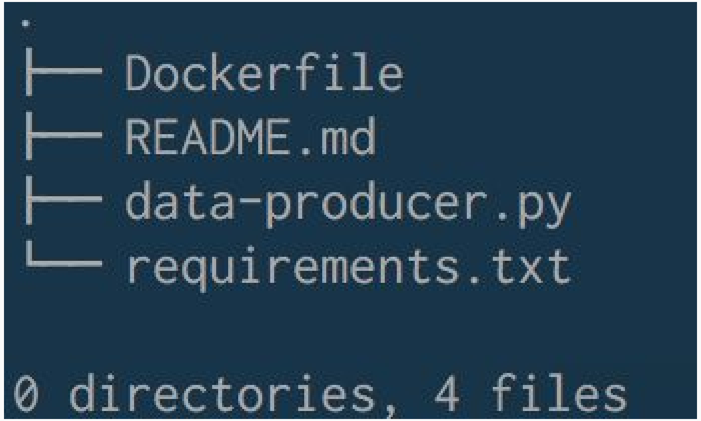
- Dev Environment
- **Work with Docker**
- Work with Redis
- Work with Spark

# Convert Your Program to Docker Container

- We have implemented data-producer.py script
  - `python data-producer.py AAPL stock-analyzer 192.168.99.100:9092`
- Your user might have different pip, python versions

# Create a Dockerfile

- Say you have your project file structure like this
- requirements.txt
  - Your python code dependencies
- data-producer.py
  - Your python code
- Dockerfile
  - You don’t have it yet, we will create one



```
.
├── Dockerfile
├── README.md
├── data-producer.py
└── requirements.txt

0 directories, 4 files
```

# Create a Dockerfile

```
FROM python:2.7.12-alpine  # based on python 2.7.12 image
ENV SYMBOL “AAPL”          # some environment variable settings, can be overridden
ENV TOPIC “stock-analyzer” # more setting
ENV KAFKA_LOCATION “192.168.99.100:9092”    # - even more settings
ADD . /code                # add your current building folder to /code folder in container
RUN pip install -r /code/requirements.txt   # run the following command in container
CMD python /code/data-producer.py ${SYMBOL} ${TOPIC} ${KAFKA_LOCATION}
```

- Make sure your terminal is connected to a docker-machine vm
- Dockerfile is a way to build a Docker image
- Dockerfile reference
  - https://docs.docker.com/engine/reference/builder/

# Build a Docker Image

- `docker build -t unclebarney/data-producer .`
    - Build a image called unclebarney/data-producer
- `docker images`

# Start the Docker Container

- docker run unclebarney/data-producer
  - This will start a container based on the image you built
- docker run -e SYMBOL=GOOGL unclebarney/data-producer
  - This will override your default SYMBOL environment variable

- Notice how your program runs a bit slower

# Stop the Docker Container

- `docker stop [docker container id]`
- `docker kill [docker container id]`

# Agenda

- Dev Environment
- Work with Docker
- **Work with Redis**
- Work with Spark

# Start Redis Server

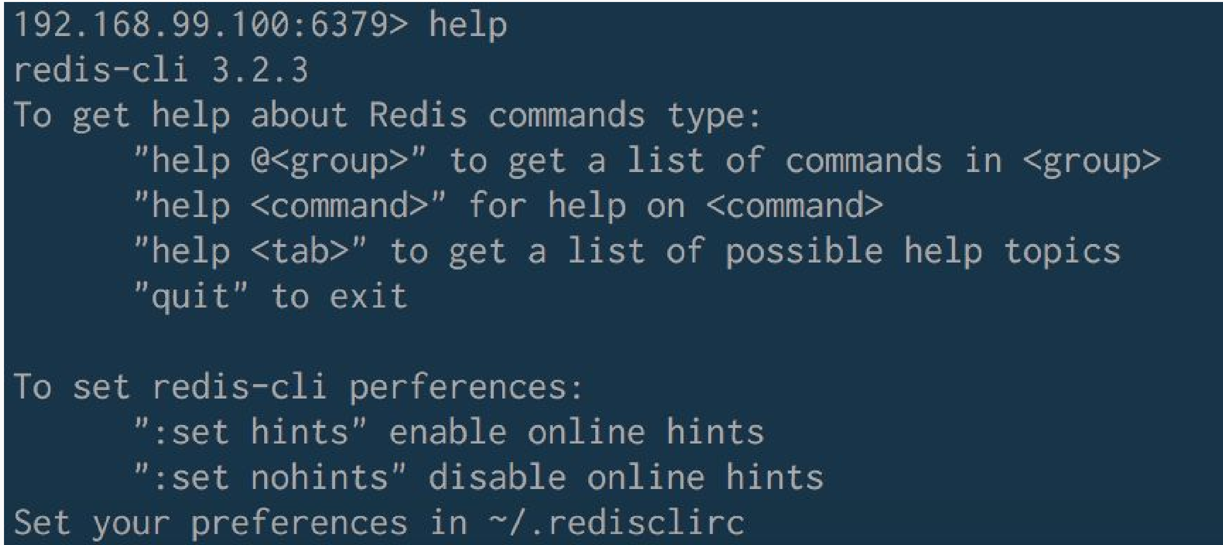
- `docker run -d -p 6379:6379 --name redis redis:alpine`
- `docker images`
  - You should see a new image being downloaded called redis:alpine
- `docker ps`
  - You should see a new redis container called redis

# Get Redis CLI

- MacOS, Unix, Linux (build from source)
  - wget http://download.redis.io/release/redis-3.2.3.tar.gz
  - tar xzf redis-3.2.3.tar.gz
  - mv redis-3.2.3 redis
  - cd redis-3.2.3
  - make
  - Rm redis-3.2.3.tar.gz
- MacOS using brew
  - brew install redis
- Windows
  - http://redis.io/download

# Connect to Redis Server

- Under your redis/src folder
- ./redis-cli -h 192.168.99.100
- help

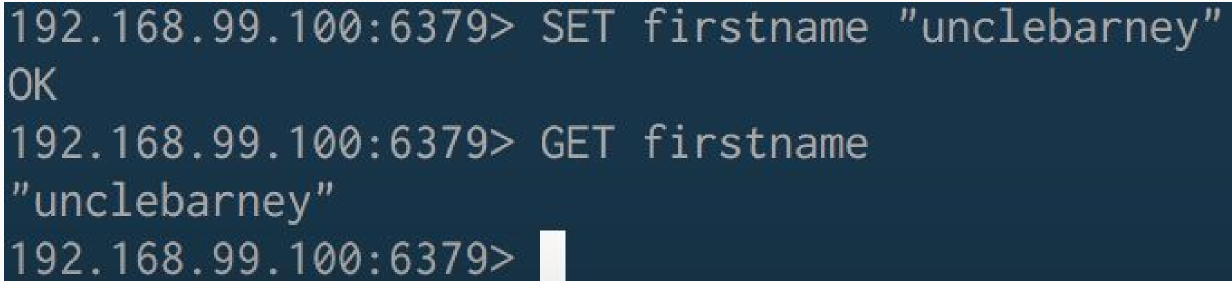


```
192.168.99.100:6379> help
redis-cli 3.2.3
To get help about Redis commands type:
      "help @<group>" to get a list of commands in <group>
      "help <command>" for help on <command>
      "help <tab>" to get a list of possible help topics
      "quit" to exit

To set redis-cli perferences:
      ":set hints" enable online hints
      ":set nohints" disable online hints
Set your preferences in ~/.redisclirc
```

# Create and Retrieve Record

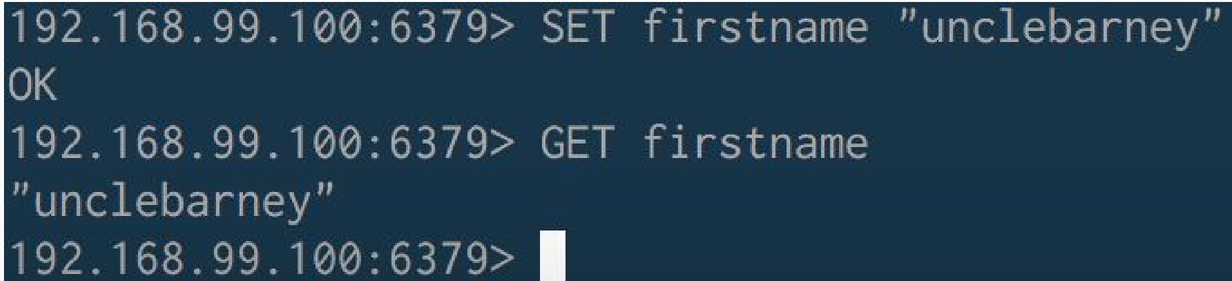
- Inside redis-cli
- SET firstname "unclebarney"
- GET firstname



```
192.168.99.100:6379> SET firstname "unclebarney"
OK
192.168.99.100:6379> GET firstname
"unclebarney"
192.168.99.100:6379>
```

# Basic Operations

- Inside redis-cli
- SET firstname "unclebarney"
- GET firstname
- DEL firstname



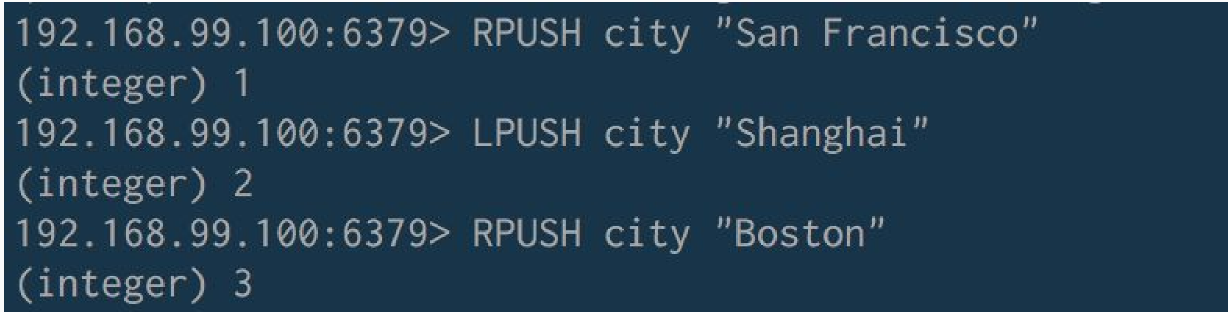
```
192.168.99.100:6379> SET firstname "unclebarney"
OK
192.168.99.100:6379> GET firstname
"unclebarney"
192.168.99.100:6379>
```

# Time To Live

- Inside redis-cli
- SET firstname "unclebarney"
- EXPIRE firstname 10
  - This record will be gone after 10 seconds

# Basic Operations with list

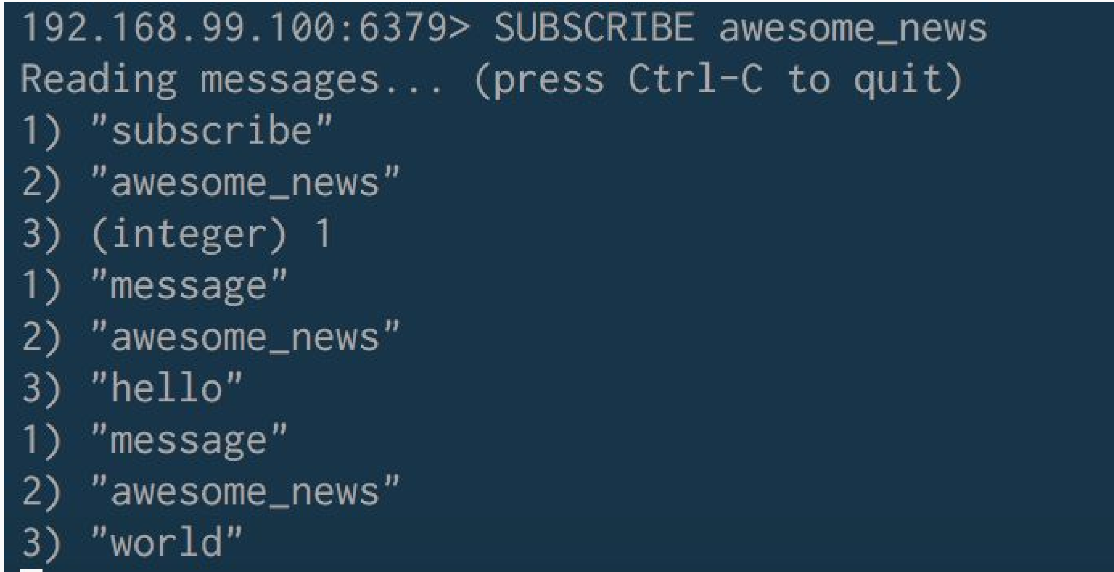
- Inside redis-cli
- RPUSH city "San Francisco"
- LPUSH city "Shanghai"
- RPUSH city "Boston"
- LRANGE city 0 -1
    - Use this command to print out what’s in the list



```
192.168.99.100:6379> RPUSH city "San Francisco"
(integer) 1
192.168.99.100:6379> LPUSH city "Shanghai"
(integer) 2
192.168.99.100:6379> RPUSH city "Boston"
(integer) 3
```

# Publish and Subscribe

- Open two redis-cli terminal
- In one terminal
  - SUBSCRIBE awesome_news
- In another terminal
  - PUBLISH awesome_news hello
  - PUBLISH awesome_news world



```
192.168.99.100:6379> SUBSCRIBE awesome_news
Reading messages... (press Ctrl-C to quit)
1) "subscribe"
2) "awesome_news"
3) (integer) 1
1) "message"
2) "awesome_news"
3) "hello"
1) "message"
2) "awesome_news"
3) "world"
```

# Further Reading

- Redis Data Types: http://redis.io/topics/data-types-intro
- The Little Redis Book: http://openmymind.net/redis.pdf

# Agenda

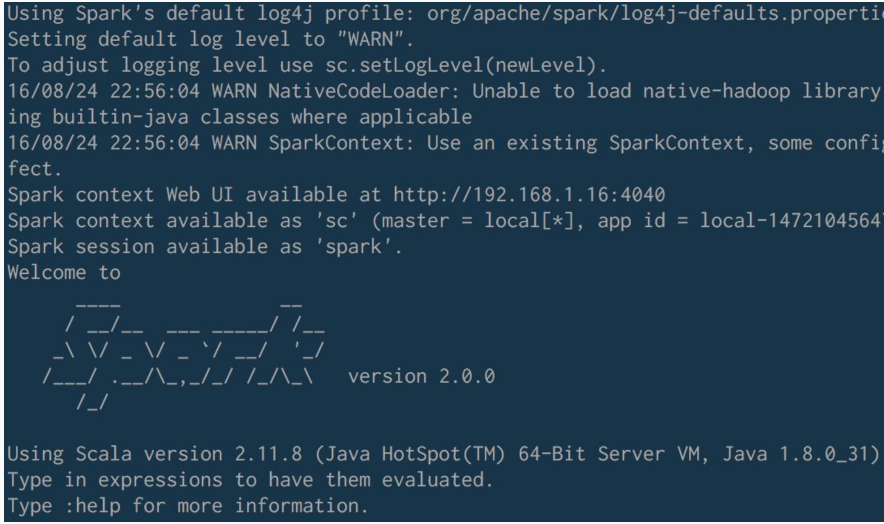
- Dev Environment
- Work with Docker
- Work with Redis
- Work with Spark

# Agenda

- Dev Environment
- Work with Docker
- Work with Redis
- **Work with Spark**

# Get Spark CLI

- Spark is hard to install, be patient
- Verify that Scala and SBT are installed
  - scala -version
  - sbt --version
- Download spark-2.0.0-bin-hadoop2.7
  - [http://spark.apache.org/downloads.html](http://spark.apache.org/downloads.html)
- tar xvf spark-2.0.0-bin-hadoop2.7
- mv spark-2.0.0-bin-hadoop2.7 spark
- rm spark-2.0.0-bin-hadoop2.7.tgz
- Under your spark/bin folder
  - Spark-shell (interactive shell in scala)
  - pyspark (interactive shell in python)
- Open up browser to checkout Spark UI
  - http://localhost:4040

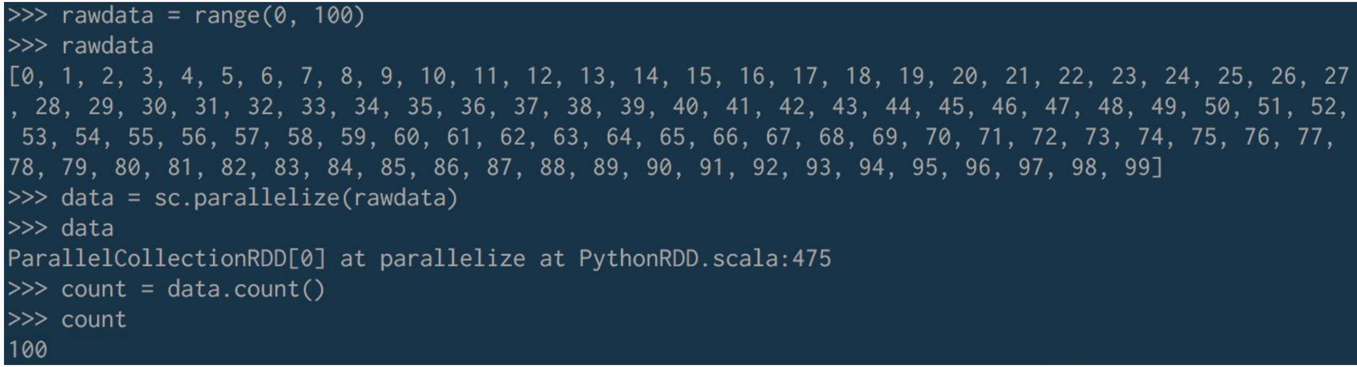


```
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properti
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel).
16/08/24 22:56:04 WARN NativeCodeLoader: Unable to load native-hadoop library
ing builtin-java classes where applicable
16/08/24 22:56:04 WARN SparkContext: Use an existing SparkContext, some confi
fect.
Spark context Web UI available at http://192.168.1.16:4040
Spark context available as 'sc' (master = local[*], app id = local-1472104564
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.0.0
      /_/

Using Scala version 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_31)
Type in expressions to have them evaluated.
Type :help for more information.
```

# Simple Example

- rawdata = range(0, 100)
  - Use python range to generate 100 numbers
- data = sc.parallelize(rawdata)
  - Convert rawdata into spark RDD
- count = data.count()
  - Spark will count the number of records



```
>>> rawdata = range(0, 100)
>>> rawdata
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27
, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52,
 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77,
78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99]
>>> data = sc.parallelize(rawdata)
>>> data
ParallelCollectionRDD[0] at parallelize at PythonRDD.scala:475
>>> count = data.count()
>>> count
100
```

# Further Reading

- Spark Quick Start: http://spark.apache.org/docs/latest/quick-start.html