

# 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](#) › 27:

Docker Tutorial: Apache Kafka with Java API on Cloudera quickstart

## 27: Docker Tutorial: Apache Kafka with Java API on Cloudera quickstart

 Posted on [June 22, 2019](#)

This requires [26: Docker Tutorial: Apache Kafka install, create topic & publish message on Cloudera quickstart](#). It is important that you continue from the previous tutorial.

### Add maven path to ~/.bash\_profile

**Step 1:** Maven is already in the quickstart/cloudera Docker image.

### 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 [kafka@quickstart ~]$ vi ~/.bash_profile
```

Modify so that “mvn” command can be run from any folder.

```
1 # .bash_profile
2
3 # Get the aliases and functions
4 if [ -f ~/.bashrc ]; then
5     . ~/.bashrc
6 fi
7
8 # User specific environment and startup programs
9
10 PATH=$PATH:$HOME/bin
11
12 export PATH
13
14 export MAVEN_OPTS=-Dhttps.protocols=TLSv1,TLSv1.1
15 PATH=$PATH:/usr/local/apache-maven/apache-maven-3
16
```

Activate:

```
1 [kafka@quickstart ~]$ source ~/.bash_profile
2
```

Now “mvn” command is available from any folder:

```
1 [kafka@quickstart ~]$ mvn --version
2 Apache Maven 3.0.4 (r1232337; 2012-01-17 08:44:56+0800)
3 Maven home: /usr/local/apache-maven/apache-maven-3.0.4
4 Java version: 1.7.0_67, vendor: Oracle Corporation
5 Java home: /usr/java/jdk1.7.0_67-cloudera/jre
6 Default locale: en_US, platform encoding: UTF-8
7 OS name: "linux", version: "4.9.125-linuxkit", arch: amd64
8 [kafka@quickstart ~]$
```

## Create a Java project structure

**Step 2:** Simple Java project in maven structure.

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

## 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



```
1 [kafka@quickstart /]$ sudo mkdir /projects
2 [kafka@quickstart /]$ cd !$
3
1 [kafka@quickstart /]$ sudo chown kafka:root /projec
```

Create the maven structure:

```
1 [kafka@quickstart /]$ mvn archetype:generate -Dgroovy
2 -DartifactId=my-app \
3 -DarchetypeArtifactId=maven-archetype-quickstart
4 -DinteractiveMode=false
5
```

It takes sometime as the dependencies & plugins are downloaded.

```
1 [kafka@quickstart projects]$ tree my-app
2 my-app
3 |__ pom.xml
4 |__ src
5 |   |__ main
6 |       |__ java
7 |           |__ com
8 |               |__ mycompany
9 |                   |__ app
10 |                       |__ App.java
11 |   |__ test
12 |       |__ java
13 |           |__ com
14 |               |__ mycompany
15 |                   |__ app
16 |                       |__ AppTest.java
17
18 11 directories, 3 files
19 [kafka@quickstart projects]$
20
```

## pom.xml

**Step 3:** Add kafka dependency to the pom.xml file.

```
1 [kafka@quickstart projects]$ cd my-app/  
2 [kafka@quickstart my-app]$ vi pom.xml  
3
```

Add “kafka-clients” dependency, and the plugin to build uber jars.

```
1 <project xmlns="http://maven.apache.org/POM/4.0.0"  
2   xsi:schemaLocation="http://maven.apache.org/POM  
3   <modelVersion>4.0.0</modelVersion>  
4   <groupId>com.mycompany.app</groupId>  
5   <artifactId>my-app</artifactId>  
6   <packaging>jar</packaging>  
7   <version>1.0-SNAPSHOT</version>  
8   <name>my-app</name>  
9   <url>http://maven.apache.org</url>  
10  
11   <dependencies>  
12     <dependency>  
13       <groupId>junit</groupId>  
14       <artifactId>junit</artifactId>  
15       <version>3.8.1</version>  
16       <scope>test</scope>  
17     </dependency>  
18  
19     <dependency>  
20       <groupId>org.apache.kafka</groupId>  
21       <artifactId>kafka-clients</artifactId>  
22       <version>0.11.0.0</version>  
23     </dependency>  
24  
25   </dependencies>  
26  
27   <!-- Build uber jar -->  
28   <build>  
29     <plugins>  
30       <plugin>  
31         <groupId>org.apache.maven.plugins</grou  
32         <artifactId>maven-shade-plugin</artifac  
33         <executions>  
34           <execution>  
35             <phase>package</phase>  
36             <goals>  
37               <goal>shade</goal>  
38             </goals>  
39           </execution>  
40         </executions>  
41       </plugin>  
42     </plugins>  
43   </build>  
44
```

```
45 </project>
46
```

## Write the producer Java code

```
1 [kafka@quickstart my-app]$ vi src/main/java/com/myc
2
package com.mycompany.app;
import org.apache.kafka.clients.producer.*;
import org.apache.kafka.common.serialization.LongS
import org.apache.kafka.common.serialization.Strin
import java.util.Properties;

public class KafkaProducerExample {
    private final static String TOPIC = "MyTestTop
    private final static String BOOTSTRAP_SERVERS
        "quickstart.cloudera:9092,quickstart.c

    public static void main(String[] args) {
        Properties props = new Properties();
        props.put(ProducerConfig.BOOTSTRAP_SERVERS_C
            BOOTSTRA
        props.put(ProducerConfig.CLIENT_ID_CONFIG, '
        props.put(ProducerConfig.KEY_SERIALIZER_CLAS
            LongSerializ
        props.put(ProducerConfig.VALUE_SERIALIZER_CI
            StringSerializ

        final Producer<Long, String> producer = new

        try {
            final ProducerRecord record =
                new ProducerRecord(TOPIC, "He

            producer.send(record);
        } finally {
            producer.flush();
            producer.close();
        }

        System.out.println("A message has been sent

    }
}
}
```

## Package with mvn

```
1 [kafka@quickstart my-app]$ mvn package
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17 9 directories, 4 files
18 [kafka@quickstart my-app]$
19
```

```
1 [kafka@quickstart my-app]$ tree -L 2
2 .
3 ├── dependency-reduced-pom.xml
4 ├── pom.xml
5 ├── src
6 │   ├── main
7 │   └── test
8 └── target
    ├── classes
    ├── maven-archiver
    ├── my-app-1.0-SNAPSHOT.jar
    ├── original-my-app-1.0-SNAPSHOT.jar
    ├── surefire
    ├── surefire-reports
    └── test-classes
```

## Run the producer via Java code

```
1 [kafka@quickstart my-app]$ java -cp target/my-app-1.0-SNAPSHOT.jar:target/dependency:org.slf4j:slf4j-api:1.7.30:jar:slf4j-log4j12:1.7.30:jar:log4j:1.2.17:jar: MyTestProducer
2 SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder"
3 SLF4J: Defaulting to no-operation (NOP) logger implementation
4 SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
5 A message has been sent to the topic: MyTestTopic
6 [kafka@quickstart my-app]$
7
```

## Run the consumer

```
1 [kafka@quickstart my-app]$ /home/kafka/kafka/bin/kafka-console-consumer --zookeeper localhost:2181 --topic MyTestTopic --from-beginning
2 Hello, World
3 Hello, my topic
4 Hello From Java
5 Hello From Java
6
```

## Write the consumer via Java code

```
1 [kafka@quickstart my-app]$ vi src/main/java/com/myc
2
package com.mycompany.app;
import org.apache.kafka.clients.consumer.*;
import org.apache.kafka.common.serialization.Long
import org.apache.kafka.common.serialization.Stri
import java.util.Properties;
import java.util.Collections;
public class KafkaConsumerExample {
    private final static String TOPIC = "MyTestTop
    private final static String BOOTSTRAP_SERVERS
        "quickstart.cloudera:9092,quickstart.c
    public static void main(String[] args) {
        Properties props = new Properties();
        props.put(ConsumerConfig.BOOTSTRAP_SERVERS_C
            BOOTSTRAP_SERVERS_CONFIG, "quickstart.c
        props.put(ConsumerConfig.GROUP_ID_CONFIG, "I
        props.put(ConsumerConfig.KEY_DESERIALIZER_C
        props.put(ConsumerConfig.VALUE_DESERIALIZER
        final Consumer consumer = new KafkaConsumer
        consumer.subscribe(Collections.singletonList
        int noMessageFound = 0;
        while (true) {
            ConsumerRecords consumerRecords = consur
            System.out.println("Msg count:" + consur
            if (consumerRecords.count() == 0) {
                noMessageFound++;
                if (noMessageFound > 10)
                    break;
                else
                    continue;
            }
            for (Object o : consumerRecords) {
                ConsumerRecord record = (ConsumerRe
                System.out.println("Record Key " + r
                System.out.println("Record value " +
                System.out.println("Record partition
                System.out.println("Record offset "
            }
            consumer.commitAsync();
```

```
51     }
52
53     consumer.close();
54
55 }
56
57 }
58
```

## Run the producer & consumer via Java code

### Package:

```
1 [kafka@quickstart my-app]$ mvn package
2
```

```
1 [kafka@quickstart my-app]$ tree -L 7
2 .
3 ├── dependency-reduced-pom.xml
4 ├── pom.xml
5 ├── src
6 │   ├── main
7 │   │   ├── java
8 │   │   │   ├── com
9 │   │   │   │   ├── mycompany
10 │   │   │   │   │   ├── app
11 │   │   │   │   │   │   ├── App.java
12 │   │   │   │   │   │   ├── KafkaConsumerExample.java
13 │   │   │   │   │   │   └── KafkaProducerExample.java
14 │   └── test
15 │       ├── java
16 │       │   ├── com
17 │       │   │   ├── mycompany
18 │       │   │   │   ├── app
19 │       │   │   │   │   └── AppTest.java
20 └── target
21     ├── classes
22     │   ├── com
23     │   │   ├── mycompany
24     │   │   │   ├── app
25     │   │   │   │   ├── App.class
26     │   │   │   │   ├── KafkaConsumerExample.class
27     │   │   │   │   └── KafkaProducerExample.class
28     ├── maven-archiver
29     │   └── pom.properties
30     ├── my-app-1.0-SNAPSHOT.jar
31     ├── original-my-app-1.0-SNAPSHOT.jar
32     ├── surefire
33     └── surefire-reports
```



```
34 |         |   | com.mycompany.app.AppTest.txt
35 |         |   | TEST-com.mycompany.app.AppTest.xml
36 |         |   | test-classes
37 |         |   | com
38 |         |   |   | mycompany
39 |         |   |   |   | app
40 |         |   |   |   |   | AppTest.class
41 |         |   |   |   |   |
```

## Producer:

```
1 [kafka@quickstart my-app]$ java -cp target/my-app-1
2 SLF4J: Failed to load class "org.slf4j.impl.StaticLog
3 SLF4J: Defaulting to no-operation (NOP) logger impl
4 SLF4J: See http://www.slf4j.org/codes.html#StaticLog
5 A message has been sent to the topic: MyTestTopic
6
```

## Consumer:

```
1 [kafka@quickstart my-app]$ java -cp target/my-app-1
2 SLF4J: Failed to load class "org.slf4j.impl.StaticLog
3 SLF4J: Defaulting to no-operation (NOP) logger impl
4 SLF4J: See http://www.slf4j.org/codes.html#StaticLog
5 Msg count:1
6 Record Key null
7 Record value Hello From Java
8 Record partition 0
9 Record offset 7
10 Msg count:0
11 Msg count:0
12 Msg count:0
13 Msg count:0
14 Msg count:0
15 Msg count:0
16 Msg count:0
17 Msg count:0
18 Msg count:0
19 Msg count:0
20 Msg count:0
21 [kafka@quickstart my-app]$
22
```

## Getting all the messages from beginning

```
1 package com.mycompany.app;
2
3 import org.apache.kafka.clients.consumer.*;
4 import org.apache.kafka.common.serialization.LongSer
5 import org.apache.kafka.common.serialization.String
6
7 import java.util.Properties;
8 import java.util.Collections;
9 import org.apache.kafka.common.TopicPartition;
10 import java.util.Arrays;
11 import java.util.List;
12
13 public class KafkaConsumerExample {
14     private final static String TOPIC = "MyTestTop
15     private final static String BOOTSTRAP_SERVERS
16         "quickstart.cloudera:9092,quickstart.c
17
18
19     public static void main(String[] args) {
20         Properties props = new Properties();
21         props.put(ConsumerConfig.BOOTSTRAP_SERVERS_C
22             BOOTSTRAP_SERVERS_CONFIG, "quickstart.cloudera:9092,quickstart.c
23         props.put(ConsumerConfig.GROUP_ID_CONFIG, "I
24         props.put(ConsumerConfig.KEY_DESERIALIZER_CL
25         props.put(ConsumerConfig.VALUE_DESERIALIZER
26         props.put("auto.offset.reset", "earliest");
27         props.put("enable.auto.commit", false);
28
29         final Consumer consumer = new KafkaConsumer(
30
31         TopicPartition topicPartition = new TopicPar
32         List<TopicPartition> partitions = Arrays.asl
33         consumer.assign(partitions);
34         consumer.seekToBeginning(partitions);
35
36         //consumer.subscribe(Collections.singletonListL
37
38         int noMessageFound = 0;
39
40         while (true) {
41             ConsumerRecords consumerRecords = consum
42             System.out.println("Msg count:" + consum
43             if (consumerRecords.count() == 0) {
44                 noMessageFound++;
45                 if (noMessageFound > 10)
46                     break;
47                 else
48                     continue;
49             }
50
51             for (Object o : consumerRecords) {
52                 ConsumerRecord record = (ConsumerRec
53                 System.out.println("Record Key " + r
54                 System.out.println("Record value " -
```

```
55         System.out.println("Record partition " + record.partition());
56         System.out.println("Record offset " + record.offset());
57     }
58
59     consumer.commitAsync();
60 }
61
62 consumer.close();
63
64 }
65
66 }
67
```

## Compile & package it:

```
1 [kafka@quickstart my-app]$ mvn package
2
```

## Run:

```
1 [kafka@quickstart my-app]$ java -cp target/my-app-1.0-SNAPSHOT-jar-with-dependencies.jar
2
```

## Output:

```
1 Msg count:8
2 Record Key null
3 Record value Hello, World
4 Record partition 0
5 Record offset 0
6 Record Key null
7 Record value Hello, my topic
8 Record partition 0
9 Record offset 1
10 Record Key null
11 Record value Hello From Java
12 Record partition 0
13 .....
14
```

◀ 26: Docker Tutorial: Apache Kafka install, create topic & publish message on Cloudera quickstart

28: Docker Tutorial: Apache Spark streaming with Kafka in Java 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](#)

© 2022 [java-success.com](https://www.java-success.com)