

Future business owner?

Check out these 7 easy steps to start the business you've been dreaming of [legalzoom.com](#)

How to start Apache Kafka

Apache Kafka is an open-source for distributed streaming system. We can use it as a **Messaging System**, a **Storage System** or **Stream Processing**. So in the tutorial, **JavaSampleApproach** will show you the first step to quick start with **Apache Kafka**.

Related Articles:

- [How to start Spring Kafka Application with Spring Boot](#)
- [How to start Spring Apache Kafka Application with SpringBoot Auto-Configuration](#)

Contents [\[hide\]](#)

- [1. Download Apache Kafka](#)
- [2. Start a Kafka server](#)
- [3. Create a Kafka Topic](#)
- [4. Use Kafka Producer to send messages](#)
- [5. Start a Kafka consumer](#)
- [6. Create a multi-broker cluster](#)
 - [6.1 Create a config file for new brokers](#)
 - [6.2 Start new nodes](#)
 - [6.3 Create a new topic for replication](#)
 - [6.4 Create a Producer and Consumer to replicated-topic](#)
 - [6.5 Fault-Tolerance](#)

1. Download Apache Kafka

Go to the [download](#) page, download file **Scala 2.12 – kafka_2.12-0.10.2.1.tgz (asc, md5)**

Download

0.10.2.1 is the latest release. The current stable version is 0.10.2.1.

You can verify your download by following these [procedures](#) and using these [KEYS](#).

0.10.2.1

- [Release Notes](#)
- Source download: [kafka-0.10.2.1-src.tgz](#) (asc, md5)
- Binary downloads:
 - [Scala 2.10 - kafka_2.10-0.10.2.1.tgz](#) (asc, md5)
 - [Scala 2.11 - kafka_2.11-0.10.2.1.tgz](#) (asc, md5)
 - [Scala 2.12 - kafka_2.12-0.10.2.1.tgz](#) (asc, md5)

We add 2.12 to the supported Scala version. These different versions only matter if you use a different version you use. Otherwise any version should work (2.12 is recommended).



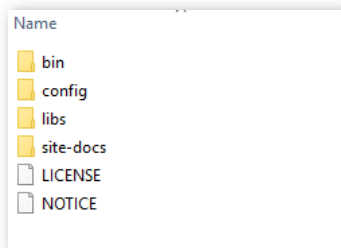
Search Product Developer Jobs On LinkedIn. Find Your Dream Job To

Ad Search Product Developer Jobs On LinkedIn. Find Your Dream Job Today.

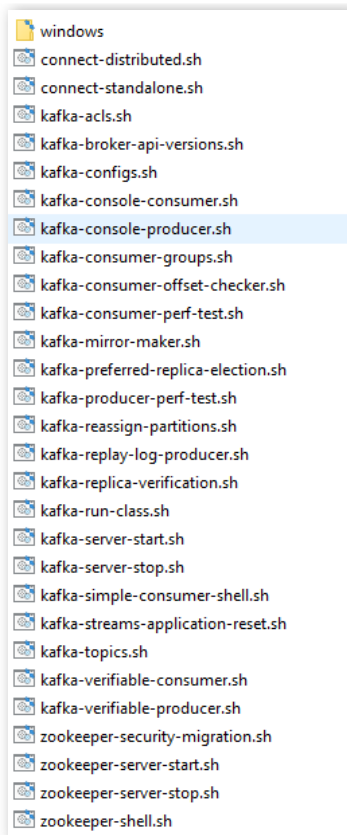
LinkedIn Careers

[Learn more](#)

We will get a **.tar** file, then un-tar it, we have folder **kafka_2.12-0.10.2.1**:

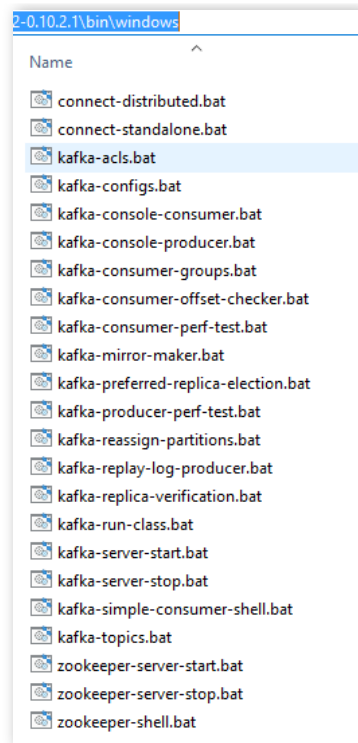


cd to `./kafka_2.12-0.10.2.1/bin`:



All files **.sh** under folder `.\kafka_2.12-0.10.2.1\bin` is used to setup Apache Kafka in **Unix-based** environment.

cd to `.\kafka_2.12-0.10.2.1\bin\windows`:



All files **.bat** under folder `.\kafka_2.12-0.10.2.1\bin\windows` is used to setup Apache Kafka in **Windows** platforms.

2. Start a Kafka server


Apache Kafka uses **ZooKeeper** as a centralized service, so we need to start a **ZooKeeper**.

-> Open a **cmd**, **cd** to `.\kafka_2.12-0.10.2.1`:


– Unix-based

```
> bin/zookeeper-server-start.sh config/zookeeper.properties
```

– Windows



Finding a job doesn't have to be a full-time job



Search Product Developer Jobs On LinkedIn. Find Your Dream Job Today.

Ad Search Product Developer Jobs On LinkedIn. Find Your Dream Job Today.

LinkedIn Careers

[Learn more](#)

```
C:\kafka_2.12-0.10.2.1>. \bin\windows\zookeeper-server-start.bat .\config\zookeeper.properties
```

```
-> Logs:
[2017-06-06 17:59:52,636] INFO Reading configuration from: .\config\zookeeper.properties
...
```

Start the **Apache Kafka server**.

-> Open a new **cmd**, **cd** to `.\kafka_2.12-0.10.2.1:`

– Unix-based

```
> bin/kafka-server-start.sh config/server.properties
```

– Windows

```
C:\kafka_2.12-0.10.2.1>. \bin\windows\kafka-server-start.bat .\config\server.properties

-> Logs:
[2017-06-06 18:06:53,365] INFO starting (kafka.server.KafkaServer)
[2017-06-06 18:06:53,367] INFO Connecting to zookeeper on localhost:2181 (kafka.server.KafkaServer)
[2017-06-06 18:06:53,379] INFO Starting ZkClient event thread. (org.I0Itec.zkclient.ZkEventThread)
...
```

3. Create a Kafka Topic

Setup a topic with name `jsa-test`, that has only **one partition** & **one replica**.

-> Open a new **cmd** and **cd** to `.\kafka_2.12-0.10.2.1:`

– Unix-based

```
> bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic jsa-test
```

– Windows

```
C:\kafka_2.12-0.10.2.1>.bin\windows\kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic jsa-test  
-> Logs:  
Created topic "jsa-test".
```

List out the topics:

– Unix-based

```
> bin/kafka-topics.sh --list --zookeeper localhost:2181  
jsa-test
```

– Windows

```
C:\kafka_2.12-0.10.2.1>.bin\windows\kafka-topics.bat --list --zookeeper localhost:2181  
jsa-test
```

4. Use Kafka Producer to send messages

Use Kafka command line client to send messages to Kafka topic.

– Unix-based

```
> bin/kafka-console-producer.sh --broker-list localhost:9092 --topic jsa-test  
Hello World!<Enter>  
This is a message to jsa-test topic!<Enter>
```

– Windows



Simplify cloud complexity

Ad Think all-in-one. Think Dynatrace.

Dynatrace

Sign Up

```
C:\kafka_2.12-0.10.2.1>.bin\windows\kafka-console-producer.bat --broker-list localhost:9092 --topic jsa-test  
Hello World!<Enter>  
This is a message to jsa-test topic!<Enter>
```

5. Start a Kafka consumer

Use Kafka command line consumer to print out messages on standard output.

– Unix-based

```
> bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic jsa-test --from-beginning
```

– Windows

```
C:\kafka_2.12-0.10.2.1>.bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic jsa-test --from-beginning
```

-> Results:

```
Hello World!  
This is a message to jsa-test topic!
```

6. Create a multi-broker cluster

Up to now, we had setup the **Kafka cluster** with single node. Now we go to next step: setup **2 new nodes** for the **Kafka cluster**.

6.1 Create a config file for new brokers

– Unix-based

```
> cp config/server.properties config/server-1.properties  
> cp config/server.properties config/server-2.properties
```

Edit new files as below:

```
config/server-1.properties:  
  broker.id=1  
  listeners=PLAINTEXT://:9093  
  log.dirs=/tmp/kafka-logs-1  
  
config/server-2.properties:  
  broker.id=2  
  listeners=PLAINTEXT://:9094  
  log.dirs=/tmp/kafka-logs-2
```

– Windows

```
C:\kafka_2.12-0.10.2.1>copy .\config\server.properties .\config\server-1.properties  
1 file(s) copied.  
  
C:\kafka_2.12-0.10.2.1>copy .\config\server.properties .\config\server-2.properties  
1 file(s) copied.
```

Edit new files {*server-1.properties*, *server-2.properties*} as below:

```
config/server-1.properties:
  broker.id=1
  listeners=PLAINTEXT://:9093
  log.dirs=kafka-logs-1

config/server-2.properties:
  broker.id=2
  listeners=PLAINTEXT://:9094
  log.dirs=kafka-logs-2
```

6.2 Start new nodes

– Unix-based

```
> bin/kafka-server-start.sh config/server-1.properties &
...
> bin/kafka-server-start.sh config/server-2.properties &
...
```

– Windows

```
C:\kafka_2.12-0.10.2.1>. \bin\windows\kafka-server-start.bat .\config\server-1.properties &
...

C:\kafka_2.12-0.10.2.1>. \bin\windows\kafka-server-start.bat .\config\server-2.properties &
...
```

6.3 Create a new topic for replication

– Unix-based

```
> bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 3 --partitions 1 --topic jsa-replicated-topic
```

– Windows

```
C:\kafka_2.12-0.10.2.1>. \bin\windows\kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 3 --partitions 1 --topic jsa-replicated-topic
Created topic "jsa-replicated-topic".
```

Check the replicated-topic by describe topics command:

– Unix-based

```
> bin/kafka-topics.sh --describe --zookeeper localhost:2181 --topic jsa-replicated-topic
```

– Windows

```
C:\kafka_2.12-0.10.2.1>. \bin\windows\kafka-topics.bat --describe --zookeeper localhost:2181 --topic jsa-replicated-topic
```

-> Results:


```
Topic:jsa-replicated-topic    PartitionCount:1    ReplicationFactor:3    Configs:
      Topic: jsa-replicated-topic    Partition: 0    Leader: 2    Replicas: 2,0,1 Isr: 2,0,1
```

The first line gives a summary of all the partitions. Each additional line gives information about one partition.

- **leader** is the node responsible for all reads and writes for the given partition.
- **replicas** is the list of nodes that replicate the log for this partition regardless of whether they are the leader or even if they are currently alive.
- **isr** is the set of "in-sync" replicas. This is the subset of the replicas list that is currently alive and caught-up to the leader.

6.4 Create a Producer and Consumer to replicated-topic

- Unix-based

```
> bin/kafka-console-producer.sh --broker-list localhost:9092 --topic jsa-replicated-topic
Message 1<Enter>
Message 2<Enter>
```

```
> bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --from-beginning --topic jsa-replicated-topic
Message 1
Message 2
```

- Window

```
C:\kafka_2.12-0.10.2.1>.bin\windows\kafka-console-producer.bat --broker-list localhost:9092 --topic jsa-replicated-topic
Message 1<Enter>
Message 2<Enter>
```

```
C:\kafka_2.12-0.10.2.1>.bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --from-beginning --topic jsa-replicated-topic
Message 1
Message 2
```

6.5 Fault-Tolerance

See again the topic description:

```
Topic:jsa-replicated-topic    PartitionCount:1    ReplicationFactor:3    Configs:
      Topic: jsa-replicated-topic    Partition: 0    Leader: 2    Replicas: 2,0,1 Isr: 2,0,1
```

Broker 2 is the **leader**. -> We will kill it by commandline:

- Unix-based

```
> ps aux | grep server-1.properties
435 ttys002    0:15.91 /System/Library/Frameworks/JavaVM.framework/Versions/1.8/Home/bin/java...
> kill -9 435
```

- Windows

```
> wmic process get processid,caption,commandline | find "java.exe" | find "server-2.properties"
```

```
java.exe ... config\server-1.properties 422
> taskkill /pid 422 /f
```

Or We can use **Ctrl + C** on cmd of Broker 2 to kill it.

See shutdown logs:

```
...

[2017-06-06 19:16:40,807] INFO Truncating log jsa-replicated-topic-0 to offset 2. (kafka.log.Log)
[2017-06-06 19:16:40,836] INFO [Kafka Server 2], Controlled shutdown succeeded (kafka.server.KafkaServer)
[2017-06-06 19:16:40,838] INFO [Socket Server on Broker 2], Shutting down (kafka.network.SocketServer)
[2017-06-06 19:16:40,844] INFO [Socket Server on Broker 2], Shutdown completed (kafka.network.SocketServer)
[2017-06-06 19:16:40,847] INFO [Kafka Request Handler on Broker 2], shutting down (kafka.server.KafkaRequestHandlerPool)
[2017-06-06 19:16:40,873] INFO [Kafka Request Handler on Broker 2], shut down completely (kafka.server.KafkaRequestHandlerPool)
[2017-06-06 19:16:40,877] INFO [ThrottledRequestReaper-Fetch], Shutting down (kafka.server.ClientQuotaManager$ThrottledRequestReaper)
[2017-06-06 19:16:41,864] INFO [ThrottledRequestReaper-Fetch], Stopped (kafka.server.ClientQuotaManager$ThrottledRequestReaper)
[2017-06-06 19:16:41,864] INFO [ThrottledRequestReaper-Fetch], Shutdown completed (kafka.server.ClientQuotaManager$ThrottledRequestReaper)
[2017-06-06 19:16:41,866] INFO [ThrottledRequestReaper-Produce], Shutting down (kafka.server.ClientQuotaManager$ThrottledRequestReaper)
[2017-06-06 19:16:41,896] INFO [ThrottledRequestReaper-Produce], Stopped (kafka.server.ClientQuotaManager$ThrottledRequestReaper)
[2017-06-06 19:16:41,896] INFO [ThrottledRequestReaper-Produce], Shutdown completed (kafka.server.ClientQuotaManager$ThrottledRequestReaper)
[2017-06-06 19:16:41,898] INFO [KafkaApi-2] Shutdown complete. (kafka.server.KafkaApis)

...
```

Again check description of **replicated-topic**:

```
C:\kafka_2.12-0.10.2.1>.bin\windows\kafka-topics.bat --describe --zookeeper localhost:2181 --topic jsa-replicated-topic
Topic:jsa-replicated-topic      PartitionCount:1      ReplicationFactor:3      Configs:
      Topic: jsa-replicated-topic      Partition: 0      Leader: 0      Replicas: 2,0,1 Isr: 0,1
```

-> Now the **Leader** is **Broker 0**. **Broker 2** had been shutdown so **in-sync** list, so We just have 2 brokers **{0, 1}**.

Make a consumer for checking the available messages:

```
C:\kafka_2.12-0.10.2.1>.bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --from-beginning --topic jsa-replicated-topic
Message 1
Message 2
```

-> **Kafka Fault-Tolerance** works right! Now you can start development with **Apache Kafka**!

By [grokonez](#) | June 6, 2017.



Related Posts

- [How to use Spring Kafka JsonSerializer \(JsonDeserializer\) to produce/consume Java Object messages](#)
- [How to start Spring Kafka Application with Spring Boot](#)
- [How to start Spring Apache Kafka Application with SpringBoot Auto-Configuration](#)
- [Angular 6 WebSocket example with Spring Boot WebSocket Server | SockJS + STOMP](#)
- [Spring Boot WebSocket with Angular 5 Client | SockJS + STOMP](#)
- [SpringBoot RabbitMq Exchange to Exchange Topology](#)
- [SpringBoot RabbitMQ Topic Exchange](#)
- [Couchbase – How to create Spring Cache Couchbase application with SpringBoot](#)
- [Spring JMS ActiveMq – How to implement a runtime SpringBoot ActiveMQ JmsResponse application](#)
- [Amazon S3 – Upload/download large files to S3 with SpringBoot Amazon S3 MultipartFile application](#)

Post Tags

[Apache Kafka](#)[integration](#)[java integration](#)[messaging system](#)

grokonez

[Home](#) | [Privacy Policy](#) | [Contact Us](#) | [Our Team](#)

© 2018–2019 grokonez. All rights reserved

DMCA  PROTECTED

FOLLOW US



ABOUT US

We are passionate engineers in software development by Java Technology & Spring Framework. We believe that creating little good thing with specific orientation everyday can make great influence on the world someday.