

腾讯课堂零声教育: <https://ke.qq.com/course/420945?tuin=137bb271>

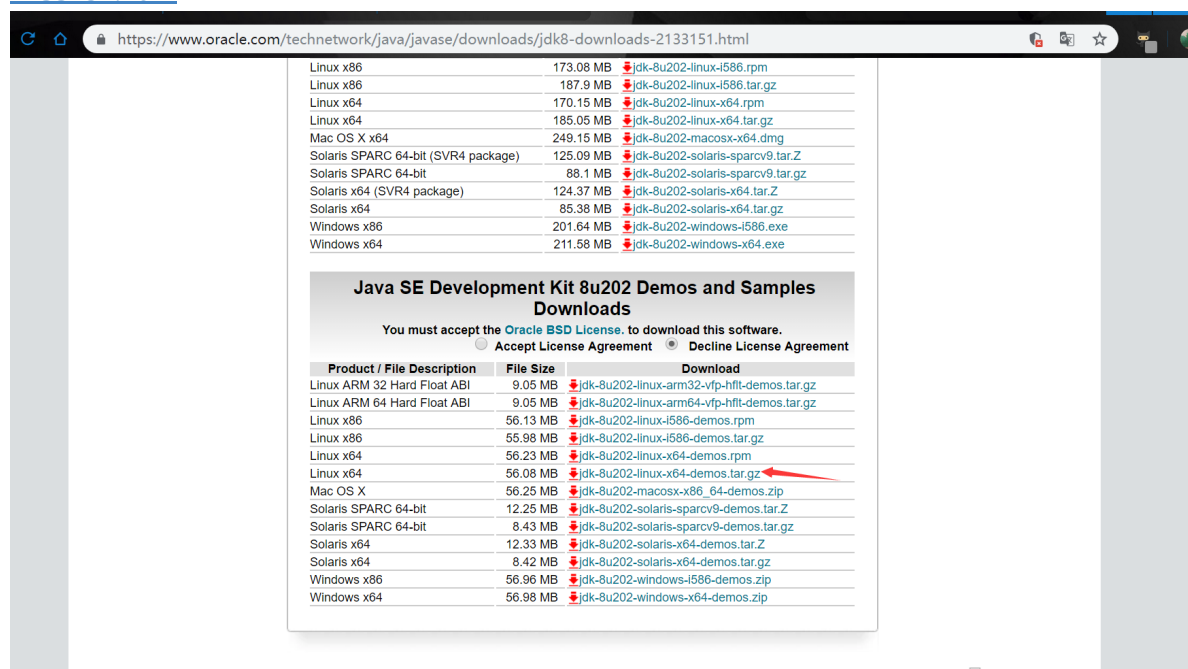
网页版本: <https://www.yuque.com/docs/share/fa589923-4368-4bcd-97ad-5e38d7207dee?#> 《1-1 kafka开发环境搭建》

1 kafka开发环境

1.1 安装Java环境

1.1.1 下载linux下的安装包

登陆网址<https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>



下载完成后, Linux默认下载位置在当前目录下的**Download**或**下载**文件夹下, 通过命令**cd ~/Downloads**或**cd ~/下载**即可查看到对应的文件。

1.1.2 解压安装包jdk-8u202-linux-x64.tar.gz

```
tar -zxvf jdk-8u202-linux-x64.tar.gz
```

解压后的文件夹为**jdk1.8.0_291**

进入文件夹和查看文件

```
cd jdk1.8.0_291
ls
```

可以看到bin目录

```
lqf@ubuntu:~/0voice/mq/kafka/jdk1.8.0_291$ ls
bin          jmc.txt    LICENSE    src.zip
COPYRIGHT   jre        man        THIRDPARTYLICENSEREADME-JAVAFX.txt
include     legal      README.html THIRDPARTYLICENSEREADME.txt
javafx-src.zip lib         release
```

1.1.3 将解压后的文件移到/usr/lib目录下

在/usr/bin目录下新建jdk目录

```
sudo mkdir /usr/lib/jdk
```

将解压的jdk文件移动到新建的/usr/lib/jdk目录下来

```
sudo mv jdk1.8.0_291 /usr/lib/jdk/
```

执行命令后可到 **usr/lib/jdk** 目录下查看是否移动成功。

1.1.4 配置java环境变量

这里是将环境变量配置在etc/profile，即为所有用户配置JDK环境。

使用命令打开/etc/profile文件

```
sudo vim /etc/profile
```

在末尾添加以下几行文字：

```
#set java env
export JAVA_HOME=/usr/lib/jdk/jdk1.8.0_291
export JRE_HOME=${JAVA_HOME}/jre
export CLASSPATH=.:${JAVA_HOME}/lib:${JRE_HOME}/lib
export PATH=${JAVA_HOME}/bin:$PATH
```

```
#set java env
export JAVA_HOME=/usr/lib/jdk/jdk1.8.0_291
export JRE_HOME=${JAVA_HOME}/jre
export CLASSPATH=.:${JAVA_HOME}/lib:${JRE_HOME}/lib
export PATH=${JAVA_HOME}/bin:$PATH
```

1.1.5 执行命令使修改立即生效

```
source /etc/profile
```

1.1.6 测试安装是否成功

在终端输入，出现版本号说明安装成功。

```
java -version
```

结果如图：

```
lqf@ubuntu:~/0voice/mq/kafka$ java -version
java version "1.8.0_291"
Java(TM) SE Runtime Environment (build 1.8.0_291-b10)
Java HotSpot(TM) 64-Bit Server VM (build 25.291-b10, mixed mode)
```

1.2 Kafka的安装部署

1.2.1 下载kafka

```
wget https://archive.apache.org/dist/kafka/2.0.0/kafka_2.11-2.0.0.tgz
```

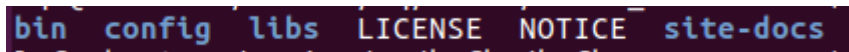
1.2.2 安装kafka

我们下载的kafka是已经编译好的程序，只需要解压即可得到执行程序。

```
tar -zxvf kafka_2.11-2.0.0.tgz
```

进入kafka目录，以及查看对应的文件和目录

```
cd kafka_2.11-2.0.0  
ls
```



```
bin  config  libs  LICENSE  NOTICE  site-docs
```

- bin: 为执行程序
- config: 为配置文件
- libs: 为库文件

1.2.3 配置和启动zookeeper

下载的kafka程序里自带了zookeeper，kafka自带的Zookeeper程序脚本与配置文件名与原生Zookeeper稍有不同。

kafka自带的Zookeeper程序使用bin/zookeeper-server-start.sh，以及bin/zookeeper-server-stop.sh来启动和停止Zookeeper。

kafka依赖于zookeeper来做master选举一起其他数据的维护。

- 启动zookeeper: zookeeper-server-start.sh
- 停止zookeeper: zookeeper-server-stop.sh

在config目录下，存在一些配置文件

```
zookeeper.properties  
server.properties
```

所以我们可以通过下面的脚本来启动zk服务，当然，也可以自己独立搭建zk的集群来实现。这里我们直接使用kafka自带的zookeeper。

启动zookeeper

```
前台运行: sh zookeeper-server-start.sh ../config/zookeeper.properties  
后台运行: sh zookeeper-server-start.sh -daemon ../config/zookeeper.properties
```

默认端口为: 2181，可以通过命令**ls -i:2181** 查看zookeeper是否启动成功。

1.2.4 启动和停止kafka

更多server.properties属性参考：《Kafka配置-server.properties详解.md》

- 修改server.properties（在config目录），增加zookeeper的配置，

这里只是本地的配置，如果是另一台机器运行zookeeper，要配置对应的ip地址。

```
zookeeper.connect=localhost:2181
```

- 启动kafka

```
$ sh kafka-server-start.sh -daemon ../config/server.properties
```

默认端口为：9092，可以通过命令**lsyf -i:9092**查看kafka是否启动成功。

- 停止kafka

```
$ sh kafka-server-stop.sh -daemon ../config/server.properties
```

1.3 kafka的基本操作

1.3.1 创建topic

```
$ sh kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1  
--partitions 1 --topic test
```

成功则显示：

```
Created topic "test".
```

参数说明：

- --create 是创建主题的动作指令
- --zookeeper 指定kafka所连接的zookeeper服务地址
- --replicator-factor 指定了副本因子（即副本数量）
- --partitions 指定分区个数
- --topic 指定所要创建主题的名称，比如test

replication-factor 表示该topic需要在不同的broker中保存几份，这里设置成1，表示在两个broker中保存两份Partitions分区数。

1.3.2 查看topic

```
$ sh kafka-topics.sh --list --zookeeper localhost:2181
```

显示：

test

1.3.3 查看topic属性

```
$ sh kafka-topics.sh --describe --zookeeper localhost:2181 --topic test
```

显示:

Topic:test PartitionCount:1 ReplicationFactor:1 Configs:

Topic: test Partition: 0 Leader: 0 Replicas: 0 Isr: 0

1.3.4 消费消息

```
$ sh kafka-console-consumer.sh --bootstrap-server 127.0.0.1:9092 --topic test  
--from-beginning
```

1.3.5 发送消息

再开启一个终端窗口:

```
$ sh kafka-console-producer.sh --broker-list 127.0.0.1:9092 --topic test
```

输入:

```
>darren  
>king  
>mark
```

消费端显示:

```
darren  
king  
mark
```

1.4 kafka-topics.sh 使用方式

围绕创建、修改、删除以及查看等功能。

1.4.1 查看帮助--help

/bin目录下的每一个脚本工具，都有着众多的参数选项，不可能所有命令都记得住，这些脚本都可以使用 --help 参数来打印列出其所需的参数信息。

```
$ sh kafka-topics.sh --help
```

Command must include exactly one action: `--list`, `--describe`, `--create`, `--alter` or `--delete`

Option	Description
-----	-----
<code>--alter</code>	Alter the number of partitions, replica assignment, and/or configuration <code>for</code> the topic.
<code>--config</code> <String: <code>name=value</code> >	A topic configuration override <code>for</code> the topic being created or altered. The following is a list of valid configurations: cleanup.policy compression.type delete.retention.ms file.delete.delay.ms flush.messages flush.ms follower.replication.throttled.replicas

....省略

下面我们挑选其中使用最为频繁且重要的参数进行说明，以及其中一些坑进行标明。

1.4.2 副本数量不能大于broker的数量

kafka 创建主题的时候其副本数量不能大于broker的数量，否则创建主题 topic 失败。

```
$ sh kafka-topics.sh --create --zookeeper localhost:2181 -replication-factor 2 -partitions 1 --topic test1
```

详细报错信息见下图

其中红色框注明了其副本数量已经超过了可使用的 kafka broker 数量。

```
Error while executing topic command : Replication factor: 2 larger than available brokers: 1.[2021-07-12 17:16:32,476] ERROR org.apache.kafka.common.errors.InvalidReplicationFactorException: Replication factor: 2 larger than available brokers: 1. (kafka.admin.TopicCommand$)
```

注意：副本数量和分区数量的区别。

1.4.3 创建主题--create

创建主题时候，有3个参数是必填的，分别是 --partitions（分区数量）、--topic（主题名）、--replication-factor（复制系数），同时还需使用 --create 参数表明本次操作是想要创建一个主题操作。

```
$ sh kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test1
```

返回：

Created topic "test1".

此时主题 test1 就已经创建了。另外在创建主题的时候，还可以附加以下两个选项：--if-not-exists 和 --if-exists。第一个参数表明仅当该主题不存在时候，创建；第二个参数表明当修改或删除这个主题时候，仅在该主题存在的时候去执行操作。

1.4.4 查看broker上所有的主题 --list

```
$ sh kafka-topics.sh --list --zookeeper localhost:2181
```

返回：test1

其中test1便为我们创建的主题。

1.4.5 查看指定主题 topic 的详细信息 --describe

该参数会将该主题的所有信息——列出打印出来，比如分区数量、副本系数、领导者等待。

```
$ sh kafka-topics.sh --describe --zookeeper localhost:2181 --topic test1
```

返回：

Topic:test1 PartitionCount:1 ReplicationFactor:1 Configs:

Topic: test1 Partition: 0 Leader: 0 Replicas: 0 Isr: 0

1.4.6 修改主题信息 --alter（增加主题分区数量）

```
$ sh kafka-topics.sh --zookeeper localhost:2181 --topic test1 --alter --partitions 2
```

```
WARNING: If partitions are increased for a topic that has a key, the partition
logic or ordering of the messages will be affected
Adding partitions succeeded!
```

可以看到已经成功的将主题的分区数量从1修改为了2。

如果去修改一个不存在的topic信息会怎么样？比如修改主题 test2，当前这主题是不存在的。

```
$ sh kafka-topics.sh --zookeeper localhost:2181 --topic test2 --alter --
partitions 2

Error while executing topic command : Topic test2 does not exist on ZK path
localhost:2181
[2021-07-12 17:28:59,253] ERROR java.lang.IllegalArgumentException: Topic test2
does not exist on ZK path localhost:2181
    at kafka.admin.TopicCommand$.alterTopic(TopicCommand.scala:123)
    at kafka.admin.TopicCommand$.main(TopicCommand.scala:65)
    at kafka.admin.TopicCommand.main(TopicCommand.scala)
(kafka.admin.TopicCommand$)
```

注意：不要使用 `--alter` 去尝试减少分区数量，如果非要减少分区数量，只能删除整个主题 topic，然后重新创建

1.4.7 删除主题 topic `--delete`

```
$ sh kafka-topics.sh --zookeeper localhost:2181 --delete --topic test1

Topic test1 is marked for deletion.
Note: This will have no impact if delete.topic.enable is not set to true.
```

日志信息提示，主题 test1 已经被标记删除状态，但是若 `delete.topic.enable` 没有设置为 `true`，则将不会有任何作用。

启动生产者：sh kafka-console-producer.sh --broker-list 127.0.0.1:9092 --topic test1

启动消费者：sh kafka-console-consumer.sh --bootstrap-server 127.0.0.1:9092 --topic test1 --from-beginning

发现此时还是可以发送消息和接收消息。

如果要支持能够删除主题的操作，则需要到 `/bin` 的同级目录 `/config` 目录下的文件 `server.properties` 中，修改配置 `delete.topic.enable=true`（如果置为 `false`，则 kafka broker 是不允许删除主题的）。

```
# topic setting
delete.topic.enable=true
```

需要 `server.properties` 中设置 `delete.topic.enable=true` 否则只是标记删除或者直接重启。

重启 kafka

停止：sh kafka-server-stop.sh -daemon ../config/server.properties

启动：sh kafka-server-start.sh -daemon ../config/server.properties

再次删除

```
$ sh kafka-topics.sh --zookeeper localhost:2181 --delete --topic test1
```

2 进一步扩展

更多 `server.properties` 属性参考：《Kafka 配置-server.properties 详解.md》

