《Apache Kafka实战剖析》第二讲——Kafka入门指引与重要操作

圣思园『知识星球』的各位会员朋友们，大家好。从本节课程开始，我们将会正式进入到《Apache Kafka实战剖析》课程的学习。在学习开始之前，我首先提出几点要求，为了确保每个人学习效果的最大化，请大家认真阅读并严格执行。我相信，但凡按照我的要求去做的人，一定会将课程的内容理解透彻并且学习到自己真正期望的技术内容。

1. 逐字逐句阅读文章中的每一句话，理解每一行代码：实际上，无论哪个技术，无论哪个框架，网上的资料都是非常多的，不过这些资料良莠不齐。要知道一点，网上的信息未见得都是对的，因此要有自己的甄别能力，这很重要。而我在『知识星球』中的文章则是将相关技术的精华以最为精炼，同时也是最为系统的方式呈现出来。文章中关于每一个知识点的介绍都是我精心思考的结果，也是我花费了巨大的精力的产物。
2. 动手极其重要。编程是一个讲求实践的领域，光看是没有任何意义的。草草看过后，也许当时觉得自己理解了，但过不了多久，你以为当时所理解的内容一定会忘的精光。只有自己真正动手操作了，并且配合我的讲解，同时加上自己的笔记才能将所讲解的技术点、知识点真正内化为自己的一部分，形成自己的知识体系的一部分。
3. 输出同样非常重要。在实践过、操作过文章中所讲解的内容后，需要通过适当的方式将所学内容输出出来，这个环节是一定不能省略的。输出的方式多种多样，比如说给别人讲、在自己的博客上、微信公众号上写文章，或是直接应用到项目开发中。无论哪种方式都是输出，都是我们在学习一项技术时的重要一环。这个环节从根本上决定了你的学习效果是不是能够达到自己当初的期望，也决定了你是否能够真正掌握一项技术。
4. 实际上，上面所说的3点不仅适用于我们接下来要讲解的Kafka这项技术，也同样适合于 任何技术的学习。通过这种手段，形成自己的学习方式闭环，从而建立起自己的知识体系与思维体系，在未来的学习道路上会令你越走越好，越走越快。

以上4点即是我对于大家的期望，希望大家能够按照要求进行。相信我，经过这样的一个完整过程，你会形成自己的学习方法论的，这会令你终生受益。

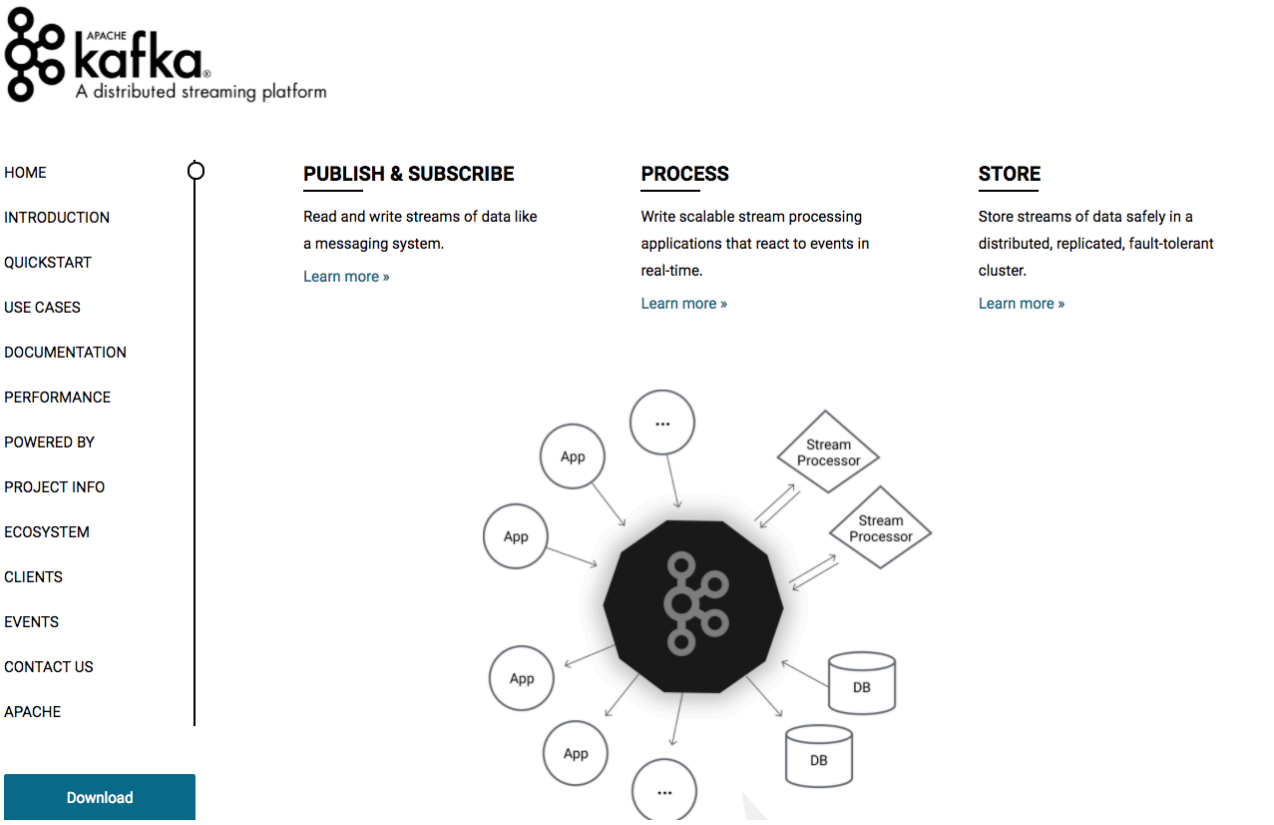
在明确了我对于大家的要求后，我们开始进入到Kafka的学习过程中。

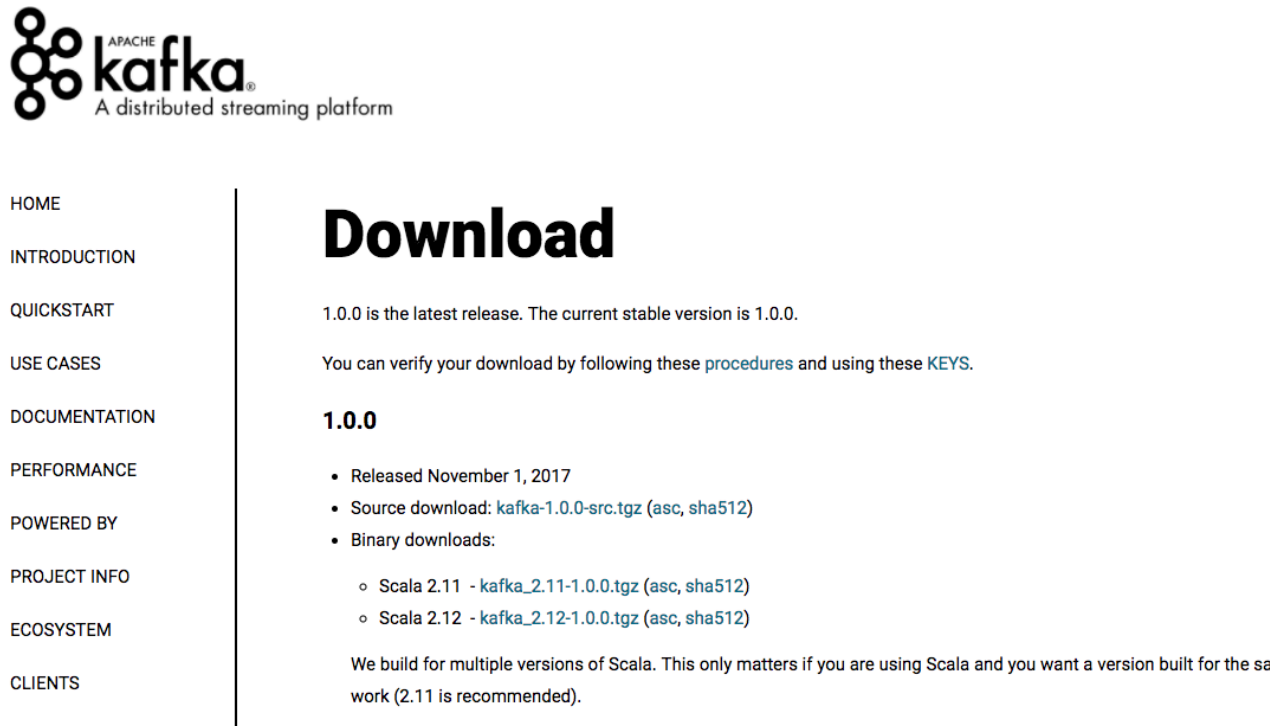
Kafka是一个典型的消息队列产品，它里面涉及到非常多的概念。我们的课程不会一开始就将这些概念全盘托出，因为对于从未接触过消息队列的人来说，

一下子遇到这么多概念的最直接的后果就是想要放弃。因此，我们会在后面对Kafka所涉及到的每一个重要概念进行有针对性的讲解。

一开始，我们要做的就是输出『Hello World』，即先将程序运行起来，用最简单的方式得到我们想要看到的最直观的结果，为后续学习开一个好头。

Kafka的官网是http://kafka.apache.org，首先访问该网址进入到Kafka的官方网站。

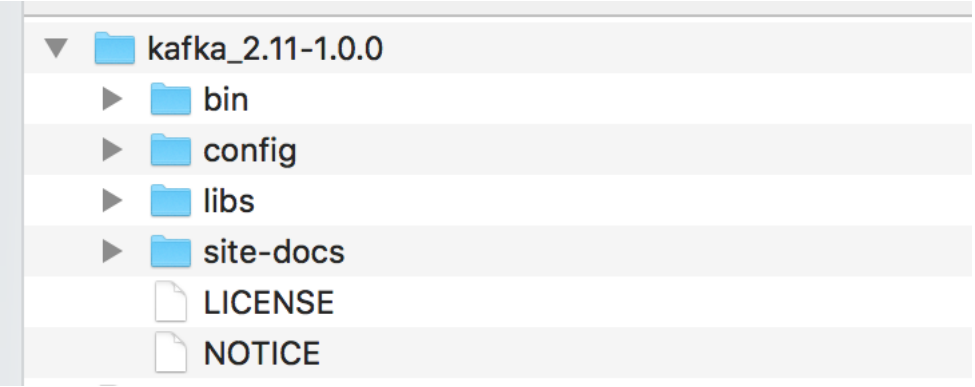




主页简要列出了Kafka的各项特点，我们先不管它，首先请单击页面左下角的下载链接下载Kafka，如上图所示。

从图中我们可以看出，目前Kafka的最新版本是1.0，它是在2017-11-01发布的，这也是我们课程中所要使用的Kafka版本。这里给出了两个下载链接，实际上对于我们的学习来说，哪个链接的下载都是可以的。他们是根据不同的Scala版本来划分的，因为Kafka内核是由Scala语言来编写的。当然，Kafka提供了针对不同语言的客户端。这里我们根据网站的建议，下载Scala 2.11 - kafka_2.11-1.0.0.tgz这一版本。

下载好后，将文件解压缩，我们会得到一系列文件。



其中，bin下面是Kafka所提供的一系列脚本文件，用于帮助我们更方便地使用Kafka，如启动、停止、创建主题、向Kafka发送消息，从Kafka接收消息等。里面的windows目录则提供了windows下的批处理脚本。
config目录则包含了Kafka所需要的各种配置文件。
libs目录包含了Kafka所需的各种jar包。
site-docs则是各种文档。

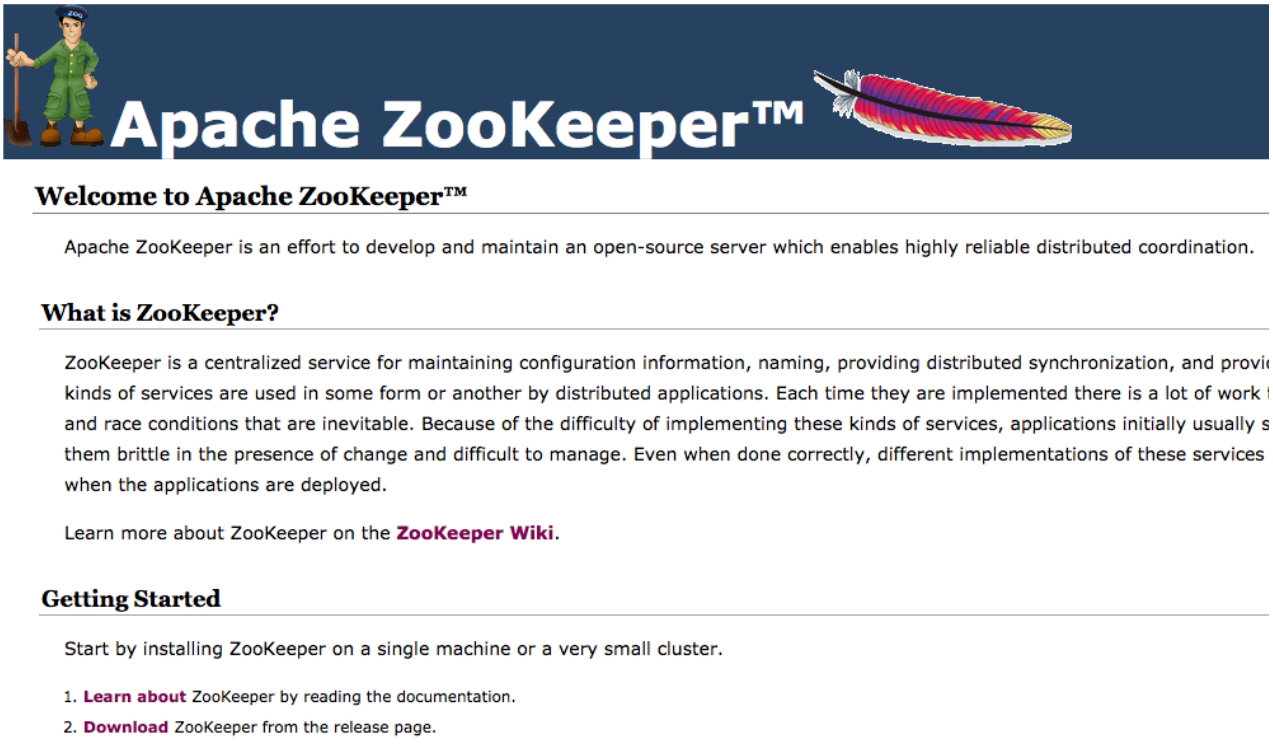
以上这些内容都是比较容易理解的。

下面，我们来通过Kafka创建第一个实例：生产者向Kafka发送一条消息，消费者从Kafka接收一条消息，以此作为我们Kafka课程的第一个示例。

Kafka是严重依赖于ZooKeeper的，通过ZooKeeper来管理各种数据与元数据，因此原则上还需要下载ZooKeeper，不过Kafka二进制包中也加入了

ZooKeeper的依赖，因此也可以直接使用Kafka自带的ZooKeeper。以上两种方式都可以，我们的课程将会使用单独下载的ZooKeepr。因此，还需要下载ZooKeeper。

ZooKeeper的网址是：http://zookeeper.apache.org。



首先，请下载ZooKeeper，这里我们下载的是ZooKeeper最新版：3.4.11。

将下载后的ZooKeeper解压缩。

这里，我们需要将config目录下的zoo_sample.cfg文件备份，然后重命名为zoo.cfg。它是ZooKeeper默认寻找的配置文件名。

接下来，打开zoo.cfg文件。

找到文件的第12行，这是ZooKeeper存放数据的目录位置（dataDir），我们可以将其修改为自己系统上的一个已知未知，其他内容则无需修改。值得注意的是，第14行表示ZooKeeper启动时的端口号，默认值为2181，我们就使用默认值即可。

如下是我修改了zoo.cfg文件后的样子：

```
# The number of milliseconds of each tick
tickTime=2000
# The number of ticks that the initial
# synchronization phase can take
initLimit=10
# The number of ticks that can pass between
# sending a request and getting an acknowledgement
syncLimit=5
# the directory where the snapshot is stored.
# do not use /tmp for storage, /tmp here is just
# example sakes.
dataDir=/Users/zhanglong/software/kafka/zookeeper-3.4.11/data
# the port at which the clients will connect
clientPort=2181
# the maximum number of client connections.
# increase this if you need to handle more clients
#maxClientCnxns=60
#
# Be sure to read the maintenance section of the
# administrator guide before turning on autopurge.
#
# http://zookeeper.apache.org/doc/current/zookeeperAdmin.html#sc_maintenance
#
# The number of snapshots to retain in dataDir
#autopurge.snapRetainCount=3
# Purge task interval in hours
# Set to "0" to disable auto purge feature
#autopurge.purgeInterval=1
```

从图中可以看到，我将dataDir值修改为了/Users/zhanglong/software/kafka/zookeeper-3.4.11/data，该目录是我提前创建好的。

接下来，关闭zoo.cfg文件。现在就可以启动ZooKeeper了。进入到bin目录，然后运行如下命令：

./zkServer.sh start-foreground

如果提示权限不足，则先赋予运行该脚本的权限，执行如下命令：

chmod 777 zkServer.sh

然后再来运行启动ZooKeeper的命令：

./zkServer.sh start-foreground

如果出现提示：operation not permitted: ./zkServer.sh，那么还需要在命令行执行如下命令：

xattr -d com.apple.quarantine zkServer.sh

接下来，再执行如下命令：

./zkServer.sh start-foreground

这表示以前台方式启动ZooKeeper，启动成功的样子如下所示：

如果出现上述界面，则表示ZooKeeper（后文简称为zk）启动成功。

接下来回到Kafka。

进入到Kafka解压缩后的目录，执行如下命令：

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

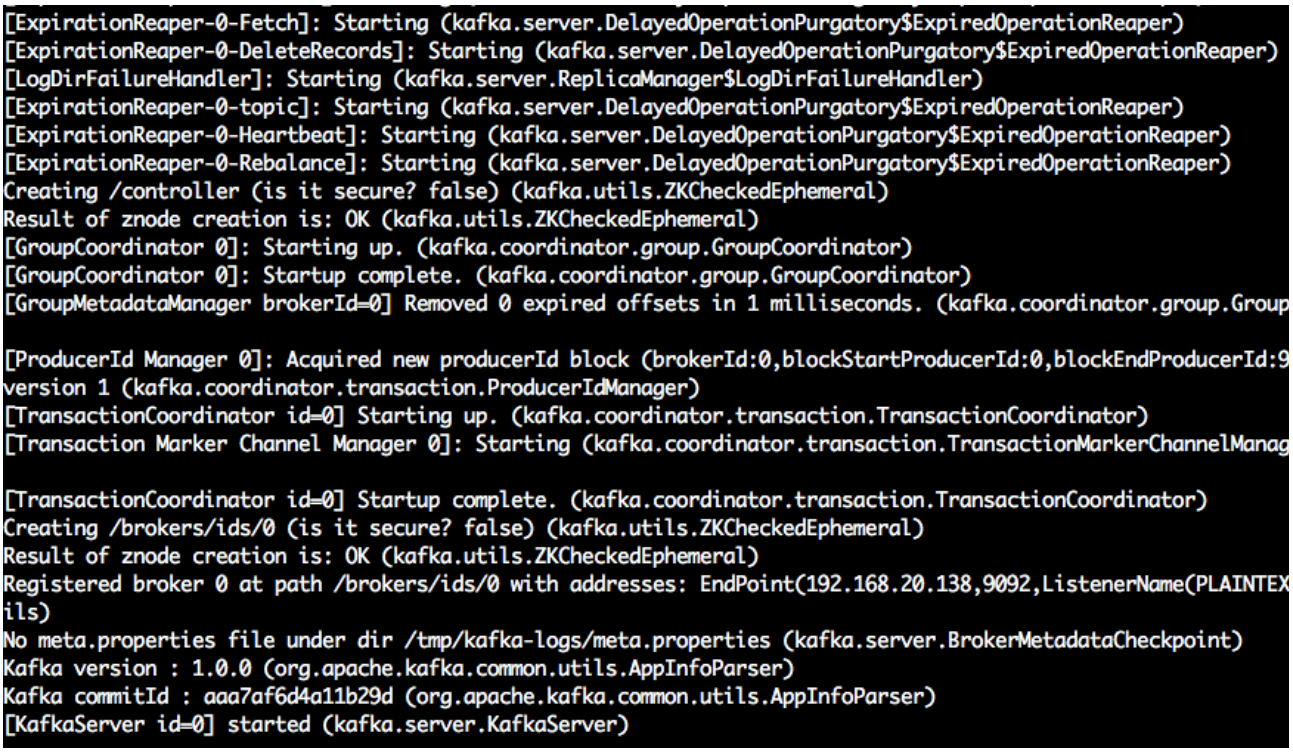
如果还是出现permission denied:，则进入到bin目录并执行如下命令：

chmod 777 *.sh

表示赋予bin目录下所有文件的可执行权限，接下来回到bin的上层目录，执行如下命令：

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

启动成功的界面如下所示：



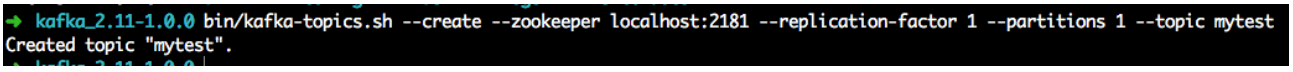
至此为止，zk与Kafka全部启动成功。

现在，我们通过Kafka的脚本，通过一个生产者向Kafka发送消息，接下来通过一个消费者来接收该消息。

执行如下命令：

bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic mytest

该命令表示创建一个名字为mytest的主题（该命令涉及到诸多参数，现在可以完全不管），创建成功的界面如下所示：

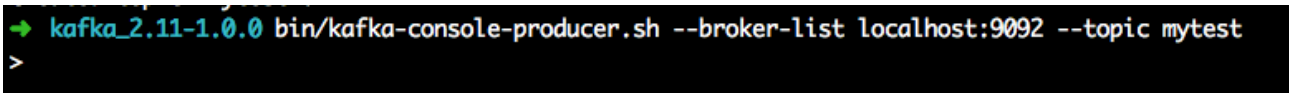


上图表示mytest这个主题创建成功，接下来需要向Kafka发送消息。

我们首先启动生产者，执行如下命令：

bin/kafka-console-producer.sh --broker-list localhost:9092 --topic mytest

命令成功执行后的结果如下所示：



表示等待我们的输入。

接下来，启动消费者，输入如下命令：

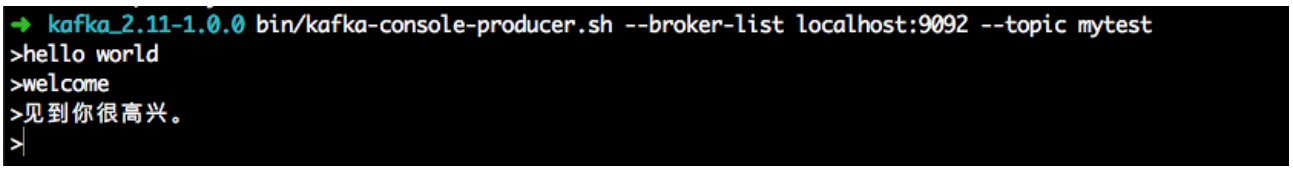
bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic mytest --from-beginning

命令成功执行后的结果如下所示：

表示等待接收消息。

现在，回到生产者窗口，在界面中随意输入一些字符，观察消费者窗口的输出内容。

成功执行后的界面如下所示：



以上是在生产者窗口中输入的内容。

以上是在消费者窗口中的输出内容。

从图中可以清晰地看到，生产者向Kafka发送的全部消息都会在消费者窗口中显示出来，这也证明了我们的整体环境搭建是正确的，且可以顺利工作。

通过这个例子，我们初步运行了Kafka，zk，并体会到了Kafka的基本工作模式，现在涉及到的一些参数可以不必去管，后续课程会进行介绍。

至此，我们的Kafka第一个示例就完整运行出来了，下一节课再见。