CONFLUENT

APACHE KAFKA

# How to Run Apache Kafka on Windows

JIM GALASYN

DECEMBER 9, 2020

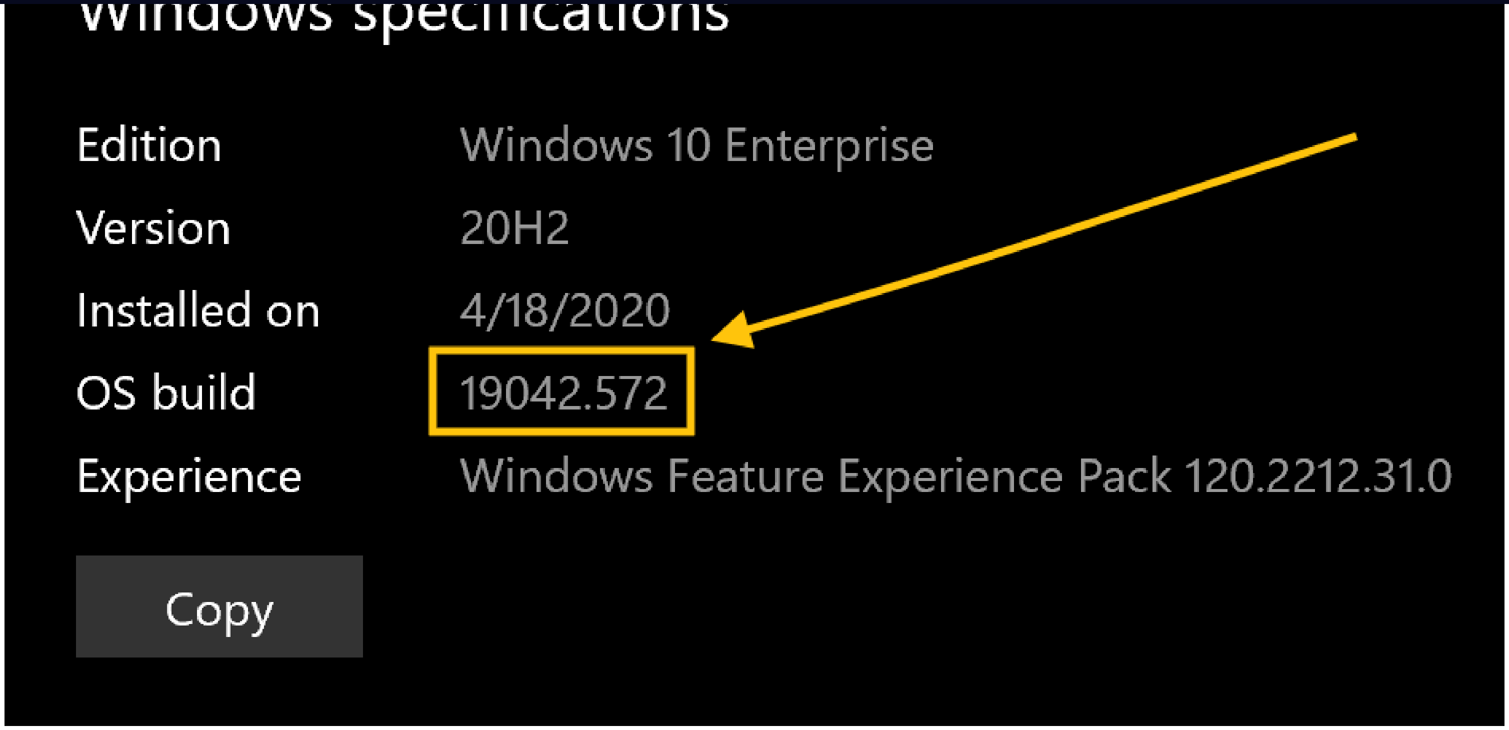
Is Windows your favorite development environment? Do you want to run Apache Kafka® on Windows? Thanks to the Windows Subsystem for Linux 2 (WSL 2), now you can, and with fewer tears than in the past. Windows still isn't the recommended platform for running Kafka with production workloads, but for trying out Kafka, it works just fine. Let's take a look at how it's done.

## Set up your environment

### Install WSL 2

The Windows Subsystem for Linux 2 makes it all possible. Microsoft describes WSL 2 as "a GNU/Linux environment—including most command line tools, utilities, and applications—directly on Windows, unmodified, without the overhead of a traditional virtual machine or dual boot setup."

Make sure you're running Windows 10, version 1903.18362 or higher. Click **Start** and navigate to **Settings > System > About**. In the "Windows specifications" section, find the "OS build."



Windows specifications

| | |
|---|---|
| Edition | Windows 10 Enterprise |
| Version | 20H2 |
| Installed on | 4/18/2020 |
| OS build | 19042.572 |
| Experience | Windows Feature Experience Pack 120.2212.31.0 |

If you're on the Windows Update train, you probably have the latest version and are good to go. If not, you need to update Windows 10.

When you're sure that Windows is up to date, follow these instructions below to install WSL 2.

### Enable the Windows Subsystem for Linux

Turn on the Windows Subsystem for Linux feature before installing a Linux distribution. Open PowerShell as an administrator, and run the following command:

```
dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart
```

This may take a minute or two. Your output should resemble the following:

```
Image Version: 10.0.18363.1139

Enabling feature(s)
[==========================100.0%==========================]
The operation completed successfully.
```

### Enable the Virtual Machine feature

WSL 2 requires the Virtual Machine Platform feature. In PowerShell, run the following command:

```
dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart
```

### Get the Linux kernel update

Download the Linux kernel update package, which is a regular Windows Installer (.msi) file.

Double-click the .msi file to install the WSL 2 update. If you're prompted for elevated permissions, select **Yes** to approve the installation.

### Set the default WSL version

In PowerShell, run the following command to set WSL 2 as the default version for your Linux distributions:

```
wsl --set-default-version 2
```

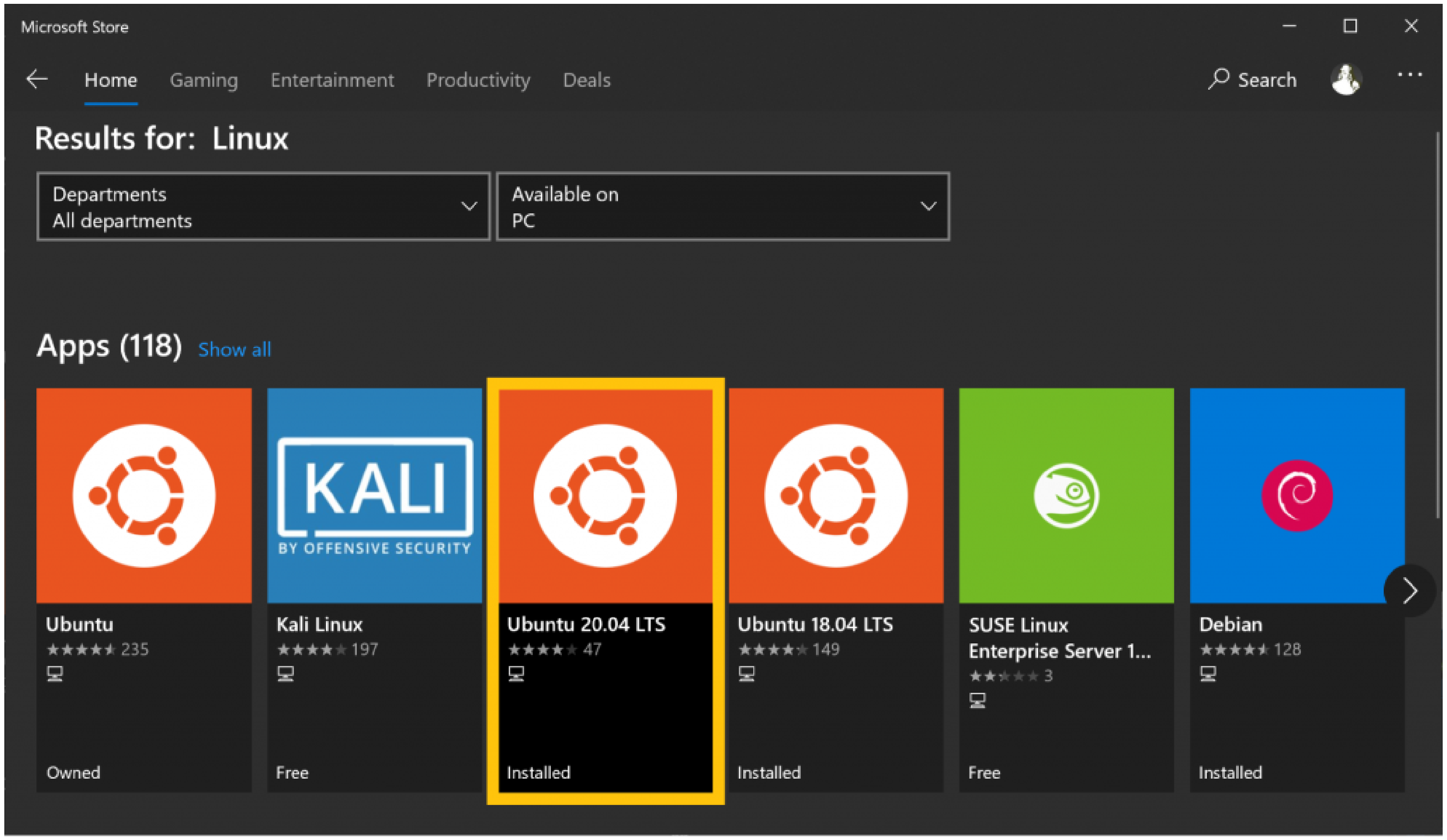
Your output should resemble the following:

```
For information on key differences with WSL 2 please visit https://aka.ms/wsl2
```

WSL 2 is ready to use. For more information on installing WSL 2, including troubleshooting, see Windows Subsystem for Linux Installation Guide for Windows 10.

Install Linux from the Microsoft Store, the same way you install other applications on Windows.

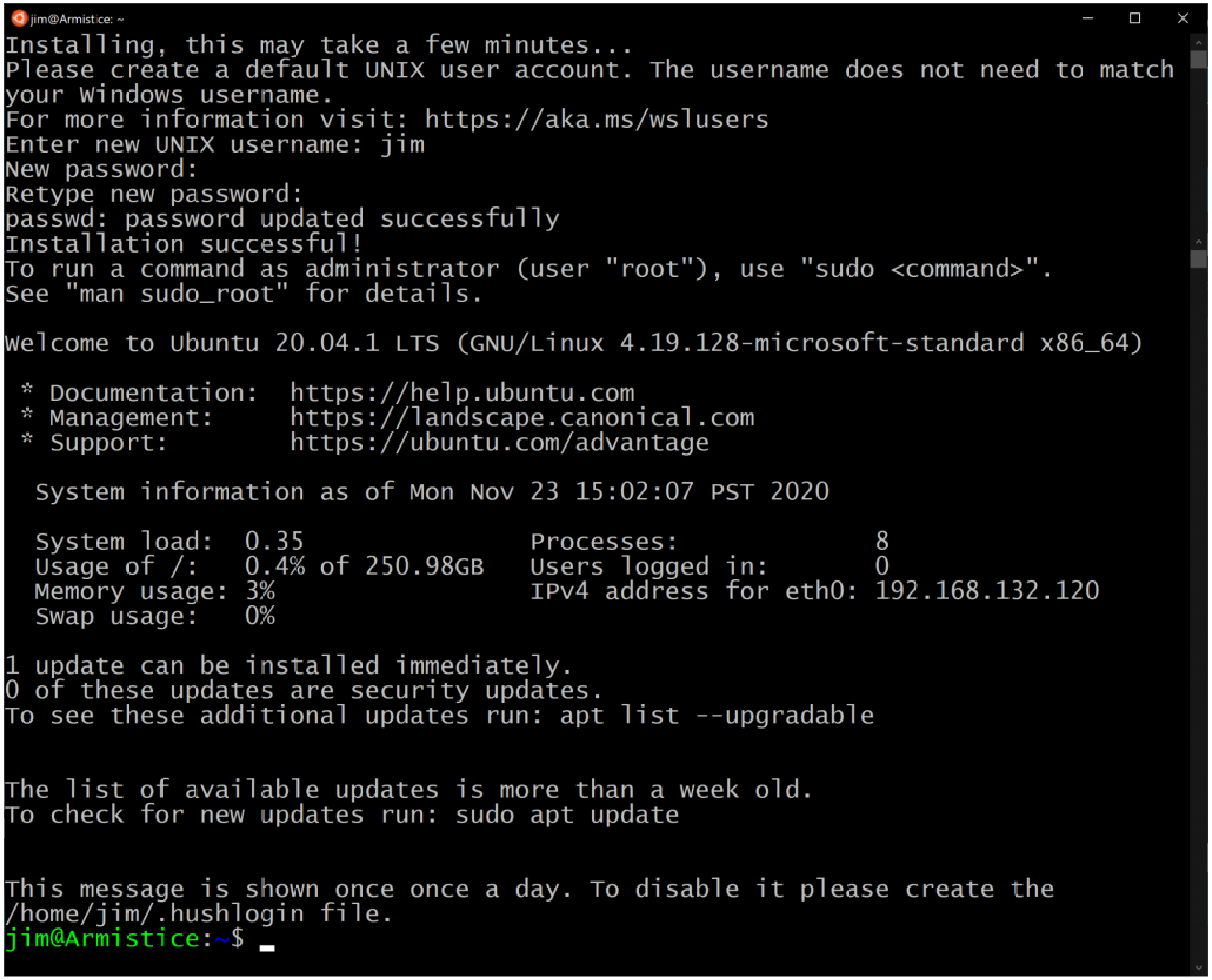
Open the Microsoft Store app and search for "Linux."



This blog post uses Ubuntu 20.04. Select **Ubuntu 20.04 LTS** and click **Install**.

When the installation is complete, click **Launch**. The shell opens and displays the following message:

```
Installing, this may take a few minutes...
Please create a default UNIX user account. The username does not need to match your Windows username.
For more information visit: https://aka.ms/wslusers
Enter new UNIX username:
```



jim@Armistice: ~

```
Installing, this may take a few minutes...
Please create a default UNIX user account. The username does not need to match
your Windows username.
For more information visit: https://aka.ms/wslusers
Enter new UNIX username: jim
New password:
Retype new password:
passwd: password updated successfully
Installation successful!
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 4.19.128-microsoft-standard x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Mon Nov 23 15:02:07 PST 2020

  System load:  0.35               Processes:             8
  Usage of /:   0.4% of 250.98GB   Users logged in:       0
  Memory usage: 3%                 IPv4 address for eth0: 192.168.132.120
  Swap usage:   0%

1 update can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


This message is shown once once a day. To disable it please create the
/home/jim/.hushlogin file.
jim@Armistice:~$
```

## Install Java

Run the package manager to get the latest updates. In the Ubuntu shell window that opened above, run the following commands:

ℹ Tip: Right-click pastes text into the terminal window.

Kafka requires the Java runtime to be version 8 or higher. Check the Java version in your Linux installation:

```
java -version
```

Your output should resemble this:

```
openjdk version "1.8.0_265"
OpenJDK Runtime Environment (build 1.8.0_265-8u265-b01-0ubuntu2~20.04-b01)
OpenJDK 64-Bit Server VM (build 25.265-b01, mixed mode)
```

If Java isn't installed (likely) or it's not the right version, install it by using your distribution's package manager. There are a lot of ways to install Java. On Ubuntu, this is one of the simplest:

```
sudo apt install openjdk-8-jdk -y
```

## Download Kafka

You can install Kafka by using a package manager, or you can download the tarball and extract it to your local machine directly.

Select a mirror from the Kafka download site and download the tarball. The following command downloads Apache Kafka version 2.6:

```
wget https://ftp.wayne.edu/apache/kafka/2.6.0/kafka_2.13-2.6.0.tgz
```

```
tar -xzf kafka_2.13-2.6.0.tgz
cd kafka_2.13-2.6.0
```

Run the `ls -al` command to list the contents of the **kafka** directory:

```
total 64
drwxr-xr-x  7 jim jim  4096 Oct 14 12:27 ./
drwxr-xr-x 25 jim jim  4096 Nov 20 12:52 ../
-rw-r--r--  1 jim jim 29975 Jul 28 11:16 LICENSE
-rw-r--r--  1 jim jim   337 Jul 28 11:16 NOTICE
drwxr-xr-x  3 jim jim  4096 Jul 28 11:23 bin/
drwxr-xr-x  2 jim jim  4096 Jul 28 11:23 config/
drwxr-xr-x  2 jim jim  4096 Oct 14 12:26 libs/
drwxr-xr-x  2 jim jim  4096 Oct 14 12:28 logs/
drwxr-xr-x  2 jim jim  4096 Jul 28 11:23 site-docs/
```

## Start the Kafka cluster

Run the following command to start ZooKeeper:

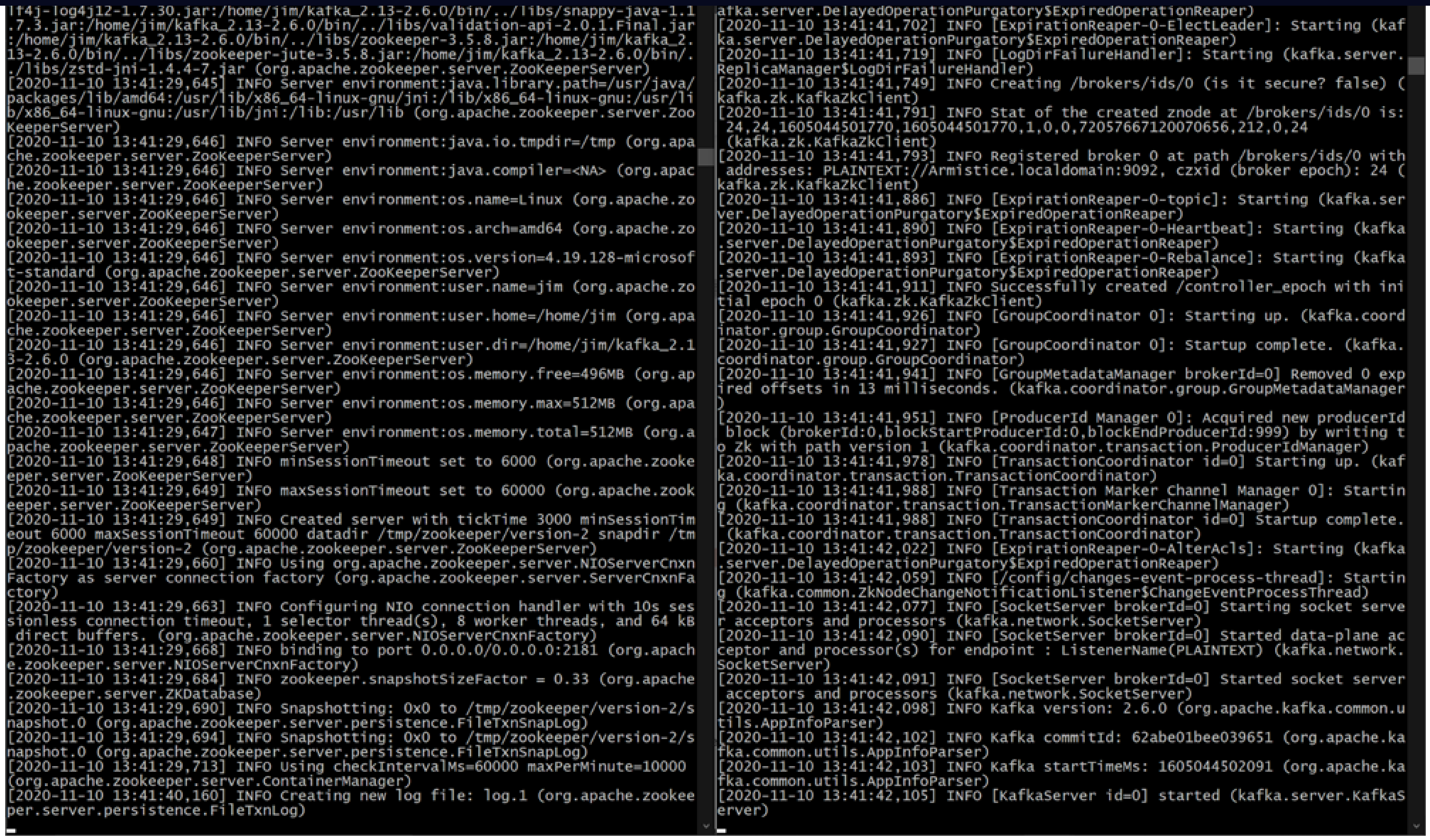
```
bin/zookeeper-server-start.sh config/zookeeper.properties
```

There will be a lot of output, and ZooKeeper will be ready in a short time, typically around a second or two.

Open another terminal session. Change the directory to the **kafka** directory, and start the Kafka broker:

```
cd kafka_2.13-2.6.0
bin/kafka-server-start.sh config/server.properties
```

If you arrange the windows to be side by side, your output should resemble the following screenshot:



```
lf4j-log4j12-1.7.30.jar:/home/jim/kafka_2.13-2.6.0/bin/../libs/snappy-java-1.1
.7.3.jar:/home/jim/kafka_2.13-2.6.0/bin/../libs/validation-api-2.0.1.Final.jar
:/home/jim/kafka_2.13-2.6.0/bin/../libs/zookeeper-3.5.8.jar:/home/jim/kafka_2.
13-2.6.0/bin/../libs/zookeeper-jute-3.5.8.jar:/home/jim/kafka_2.13-2.6.0/bin/.
./libs/zstd-jni-1.4.4-7.jar (org.apache.zookeeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,645] INFO Server environment:java.library.path=/usr/java/
packages/lib/amd64:/usr/lib/x86_64-linux-gnu/jni:/lib/x86_64-linux-gnu:/usr/li
b/x86_64-linux-gnu:/usr/lib/jni:/lib:/usr/lib (org.apache.zookeeper.server.Zoo
KeeperServer)
[2020-11-10 13:41:29,646] INFO Server environment:java.io.tmpdir=/tmp (org.apa
che.zookeeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,646] INFO Server environment:java.compiler=<NA> (org.apac
he.zookeeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,646] INFO Server environment:os.name=Linux (org.apache.zo
okeeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,646] INFO Server environment:os.arch=amd64 (org.apache.zo
okeeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,646] INFO Server environment:os.version=4.19.128-microsof
t-standard (org.apache.zookeeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,646] INFO Server environment:user.name=jim (org.apache.zo
okeeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,646] INFO Server environment:user.home=/home/jim (org.apa
che.zookeeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,646] INFO Server environment:user.dir=/home/jim/kafka_2.1
3-2.6.0 (org.apache.zookeeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,646] INFO Server environment:os.memory.free=496MB (org.ap
ache.zookeeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,646] INFO Server environment:os.memory.max=512MB (org.apa
che.zookeeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,647] INFO Server environment:os.memory.total=512MB (org.a
pache.zookeeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,648] INFO minSessionTimeout set to 6000 (org.apache.zooke
eper.server.ZooKeeperServer)
[2020-11-10 13:41:29,649] INFO maxSessionTimeout set to 60000 (org.apache.zook
eeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,649] INFO Created server with tickTime 3000 minSessionTim
eout 6000 maxSessionTimeout 60000 datadir /tmp/zookeeper/version-2 snapdir /tm
p/zookeeper/version-2 (org.apache.zookeeper.server.ZooKeeperServer)
[2020-11-10 13:41:29,660] INFO Using org.apache.zookeeper.server.NIOServerCnxn
Factory as server connection factory (org.apache.zookeeper.server.ServerCnxnFa
ctory)
[2020-11-10 13:41:29,663] INFO Configuring NIO connection handler with 10s ses
sionless connection timeout, 1 selector thread(s), 8 worker threads, and 64 kB
 direct buffers. (org.apache.zookeeper.server.NIOServerCnxnFactory)
[2020-11-10 13:41:29,668] INFO binding to port 0.0.0.0/0.0.0.0:2181 (org.apach
e.zookeeper.server.NIOServerCnxnFactory)
[2020-11-10 13:41:29,684] INFO zookeeper.snapshotSizeFactor = 0.33 (org.apache
.zookeeper.server.ZKDatabase)
[2020-11-10 13:41:29,690] INFO Snapshotting: 0x0 to /tmp/zookeeper/version-2/s
napshot.0 (org.apache.zookeeper.server.persistence.FileTxnSnapLog)
[2020-11-10 13:41:29,694] INFO Snapshotting: 0x0 to /tmp/zookeeper/version-2/s
napshot.0 (org.apache.zookeeper.server.persistence.FileTxnSnapLog)
[2020-11-10 13:41:29,713] INFO Using checkIntervalMs=60000 maxPerMinute=10000
(org.apache.zookeeper.server.ContainerManager)
[2020-11-10 13:41:40,160] INFO Creating new log file: log.1 (org.apache.zookee
per.server.persistence.FileTxnLog)
```

```
afka.server.DelayedOperationPurgatory$ExpiredOperationReaper)
[2020-11-10 13:41:41,702] INFO [ExpirationReaper-0-ElectLeader]: Starting (kaf
ka.server.DelayedOperationPurgatory$ExpiredOperationReaper)
[2020-11-10 13:41:41,719] INFO [LogDirFailureHandler]: Starting (kafka.server.
ReplicaManager$LogDirFailureHandler)
[2020-11-10 13:41:41,749] INFO Creating /brokers/ids/0 (is it secure? false) (
kafka.zk.KafkaZkClient)
[2020-11-10 13:41:41,791] INFO Stat of the created znode at /brokers/ids/0 is:
 24,24,1605044501770,1605044501770,1,0,0,72057667120070656,212,0,24
 (kafka.zk.KafkaZkClient)
[2020-11-10 13:41:41,793] INFO Registered broker 0 at path /brokers/ids/0 with
 addresses: PLAINTEXT://Armistice.localdomain:9092, czxid (broker epoch): 24 (
kafka.zk.KafkaZkClient)
[2020-11-10 13:41:41,886] INFO [ExpirationReaper-0-topic]: Starting (kafka.ser
ver.DelayedOperationPurgatory$ExpiredOperationReaper)
[2020-11-10 13:41:41,890] INFO [ExpirationReaper-0-Heartbeat]: Starting (kafka
.server.DelayedOperationPurgatory$ExpiredOperationReaper)
[2020-11-10 13:41:41,893] INFO [ExpirationReaper-0-Rebalance]: Starting (kafka
.server.DelayedOperationPurgatory$ExpiredOperationReaper)
[2020-11-10 13:41:41,911] INFO Successfully created /controller_epoch with ini
tial epoch 0 (kafka.zk.KafkaZkClient)
[2020-11-10 13:41:41,926] INFO [GroupCoordinator 0]: Starting up. (kafka.coord
inator.group.GroupCoordinator)
[2020-11-10 13:41:41,927] INFO [GroupCoordinator 0]: Startup complete. (kafka.
coordinator.group.GroupCoordinator)
[2020-11-10 13:41:41,941] INFO [GroupMetadataManager brokerId=0] Removed 0 exp
ired offsets in 13 milliseconds. (kafka.coordinator.group.GroupMetadataManager
)
[2020-11-10 13:41:41,951] INFO [ProducerId Manager 0]: Acquired new producerId
 block (brokerId:0,blockStartProducerId:0,blockEndProducerId:999) by writing t
o Zk with path version 1 (kafka.coordinator.transaction.ProducerIdManager)
[2020-11-10 13:41:41,978] INFO [TransactionCoordinator id=0] Starting up. (kaf
ka.coordinator.transaction.TransactionCoordinator)
[2020-11-10 13:41:41,988] INFO [Transaction Marker Channel Manager 0]: Startin
g (kafka.coordinator.transaction.TransactionMarkerChannelManager)
[2020-11-10 13:41:41,988] INFO [TransactionCoordinator id=0] Startup complete.
 (kafka.coordinator.transaction.TransactionCoordinator)
[2020-11-10 13:41:42,022] INFO [ExpirationReaper-0-AlterAcls]: Starting (kafka
.server.DelayedOperationPurgatory$ExpiredOperationReaper)
[2020-11-10 13:41:42,059] INFO [/config/changes-event-process-thread]: Startin
g (kafka.common.ZkNodeChangeNotificationListener$ChangeEventProcessThread)
[2020-11-10 13:41:42,077] INFO [SocketServer brokerId=0] Starting socket serve
r acceptors and processors (kafka.network.SocketServer)
[2020-11-10 13:41:42,090] INFO [SocketServer brokerId=0] Started data-plane ac
ceptor and processor(s) for endpoint : ListenerName(PLAINTEXT) (kafka.network.
SocketServer)
[2020-11-10 13:41:42,091] INFO [SocketServer brokerId=0] Started socket server
 acceptors and processors (kafka.network.SocketServer)
[2020-11-10 13:41:42,098] INFO Kafka version: 2.6.0 (org.apache.kafka.common.u
tils.AppInfoParser)
[2020-11-10 13:41:42,102] INFO Kafka commitId: 62abe01bee039651 (org.apache.ka
fka.common.utils.AppInfoParser)
[2020-11-10 13:41:42,103] INFO Kafka startTimeMs: 1605044502091 (org.apache.ka
fka.common.utils.AppInfoParser)
[2020-11-10 13:41:42,105] INFO [KafkaServer id=0] started (kafka.server.KafkaS
erver)
```

*ZooKeeper (left) and a Kafka broker (right) on Ubuntu 20.04 running on Windows and WSL 2*

## Produce and consume some messages

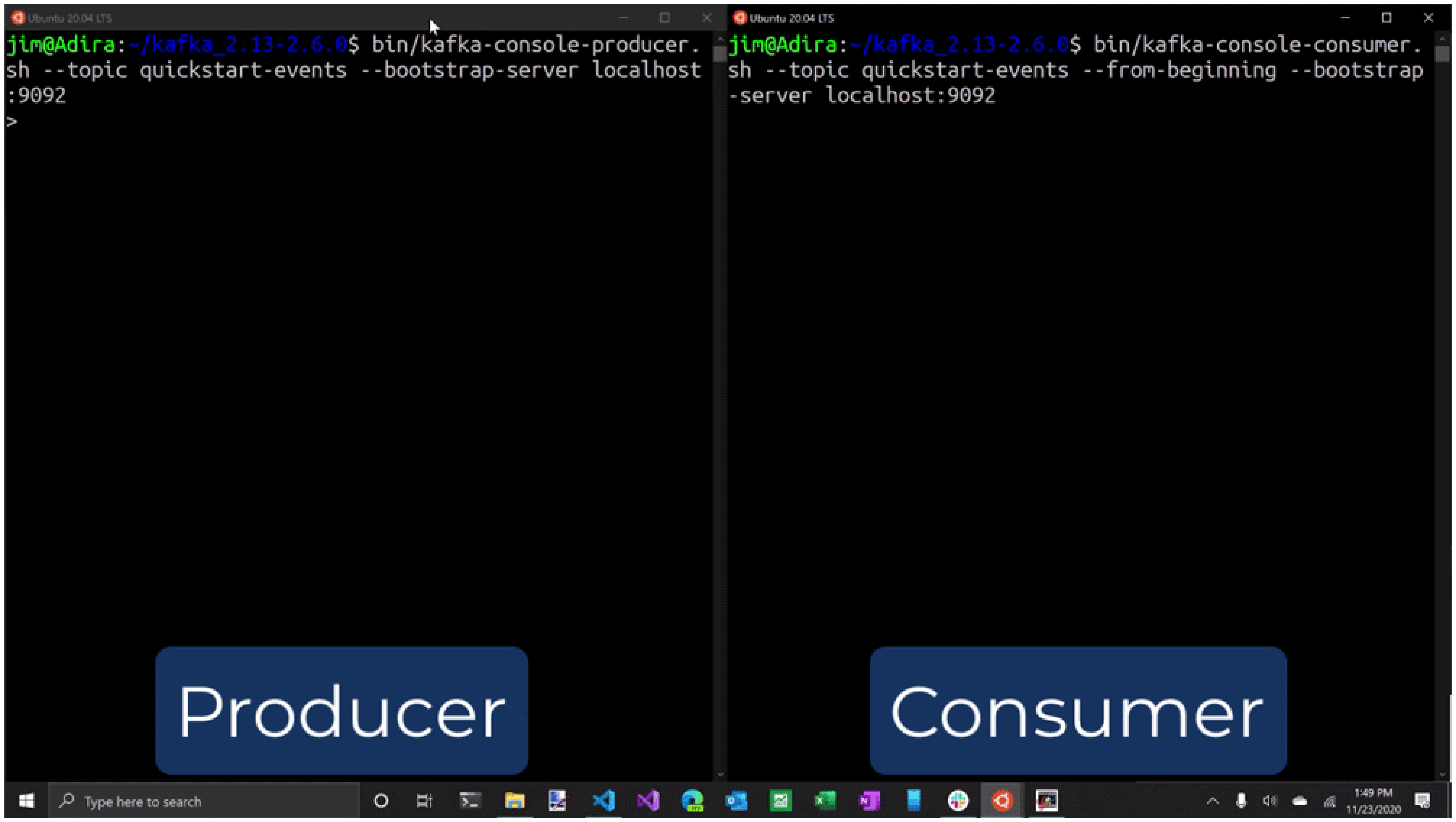
Open another terminal session and run the `kafka-topics` command to create a Kafka topic named `quickstart-events`:

```
cd kafka_2.13-2.6.0
bin/kafka-topics.sh --create --topic quickstart-events --bootstrap-server localhost:9092
```

Your output should resemble this:

```
third event
```

Arrange the producer and consumer terminal windows to be side by side. In the producer terminal, type a few more messages, and watch as they appear in the consumer terminal.



```
jim@Adira:~/kafka_2.13-2.6.0$ bin/kafka-console-producer.
sh --topic quickstart-events --bootstrap-server localhost
:9092
>
```

```
jim@Adira:~/kafka_2.13-2.6.0$ bin/kafka-console-consumer.
sh --topic quickstart-events --from-beginning --bootstrap
-server localhost:9092
```

## Stop Kafka

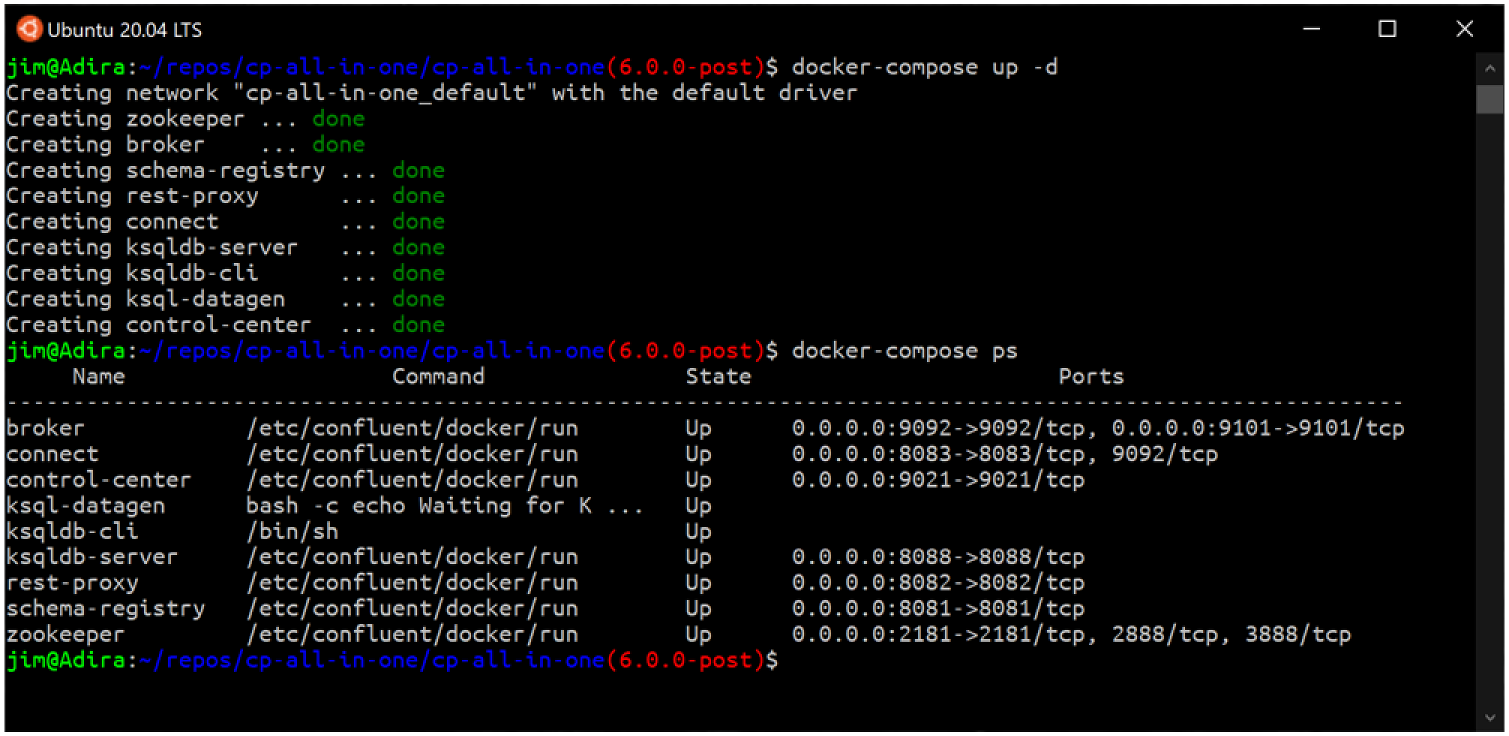
When you're done experimenting with Kafka, follow these steps to exit the Kafka environment:

1. Stop the consumer and producer clients with Ctrl+C
2. Stop the Kafka broker with Ctrl+C
3. Stop the ZooKeeper server with Ctrl+C
4. Run the following command to clean up:

   ```
   rm -rf /tmp/kafka-logs /tmp/zookeeper
   ```

There are lots of Kafka-on-Windows tutorials, but most make the mistake of running Kafka directly on the JVM on Windows. Superficially, this appears to work, but there are limitations: Kafka uses specific features of POSIX to achieve high performance, so emulations—which happen on WSL 1—are insufficient. For example, the broker will crash when it rolls a segment file. Always run Kafka on Windows in a Linux environment backed by WSL 2.

Another approach that works well is to run Kafka in Docker containers. Docker Desktop for Windows has been updated to use the WSL 2 back end, so Docker works exactly as it does on native Linux, without needing to spin up an entire VM. If you want to give this approach a go, try it out using the Confluent Platform demo.



Ubuntu 20.04 LTS

```
jim@Adira:~/repos/cp-all-in-one/cp-all-in-one(6.0.0-post)$ docker-compose up -d
Creating network "cp-all-in-one_default" with the default driver
Creating zookeeper ... done
Creating broker    ... done
Creating schema-registry ... done
Creating rest-proxy      ... done
Creating connect         ... done
Creating ksqldb-server   ... done
Creating ksqldb-cli      ... done
Creating ksql-datagen    ... done
Creating control-center  ... done
jim@Adira:~/repos/cp-all-in-one/cp-all-in-one(6.0.0-post)$ docker-compose ps
     Name                   Command               State                     Ports
---------------------------------------------------------------------------------------------------------
broker            /etc/confluent/docker/run       Up      0.0.0.0:9092->9092/tcp, 0.0.0.0:9101->9101/tcp
connect           /etc/confluent/docker/run       Up      0.0.0.0:8083->8083/tcp, 9092/tcp
control-center    /etc/confluent/docker/run       Up      0.0.0.0:9021->9021/tcp
ksql-datagen      bash -c echo Waiting for K ...  Up
ksqldb-cli        /bin/sh                         Up
ksqldb-server     /etc/confluent/docker/run       Up      0.0.0.0:8088->8088/tcp
rest-proxy        /etc/confluent/docker/run       Up      0.0.0.0:8082->8082/tcp
schema-registry   /etc/confluent/docker/run       Up      0.0.0.0:8081->8081/tcp
zookeeper         /etc/confluent/docker/run       Up      0.0.0.0:2181->2181/tcp, 2888/tcp, 3888/tcp
jim@Adira:~/repos/cp-all-in-one/cp-all-in-one(6.0.0-post)$
```

*Apache Kafka and Confluent Platform running in Docker containers on Ubuntu 20.04 on Windows and WSL 2*

## You're just getting started!

Although Kafka provides an event streaming platform to build your applications on, you'll want to take advantage of the broader ecosystem of components—like ksqlDB, Confluent Schema Registry, and Confluent Control Center—all provided as part of Confluent Platform.

Kafka Streams and Kafka Connect.

LEARN MORE

## *Kafka on Windows? What made this possible?*

*You may recall a time when Linux was anathema to Microsoft. Back in 2001, Microsoft CEO Steve Ballmer famously called Linux a "malignant cancer," but he has since come around to "loving" it. Microsoft's current CEO Satya Nadella seems intent on making it a first-class citizen in the Microsoft ecosystem, which means that a new era has arrived for software developers on the Windows platform.*

*When the Windows Subsystem for Linux (WSL 1) was released in 2016, it became possible to run a real Linux dev environment in a Linux shell, while retaining the familiar Windows UX around the shell. Even File Explorer was integrated nicely with the Linux file system.*

*The big drawbacks are that WSL 1 emulates a Linux kernel, and it runs in a full VM. The first means processes that require a native kernel, like Docker, can't run. The second means that WSL 1 consumes a lot of resources. WSL 1 was not sufficient to run Kafka reliably.*

*But Microsoft delivered WSL 2 in 2019, and it's a whole new world. They fixed the two biggest limitations, so WSL 2 runs a real Linux kernel, and the kernel runs on a subset of Hyper-V features, not in a full VM. For details, see Comparing WSL 1 and WSL 2. Now the path is clear for devs to build Kafka and ksqlDB apps on Windows.*

Jim Galasyn is a technical writer at Confluent, working on Kafka Streams and ksqlDB. He came to Confluent after a stint at Docker, and before that, 14 years at Microsoft writing developer documentation. Even after four years of working in Silicon Valley companies, he still prefers Windows.

Your Apache Kafka® journey begins here