<https://kafka.apache.org/quickstart>

<https://youtu.be/gE0sWA2kTfk>

<https://manishankarjaiswal.medium.com/installing-and-running-apache-kafka-on-windows-a-step-by-step-guide-part-1-7e957942e841>

**Step 1: Installing Apache Kafka on Windows**

**Prerequisites:**

1. **Java JDK**: Kafka requires Java to run. Make sure you have the Java Development Kit (JDK) installed on your system. You can download it from [here](https://www.oracle.com/java/technologies/javase-jdk11-downloads.html).
2. **Download Kafka**: You can download the latest version of Kafka from the [official Kafka website](https://kafka.apache.org/downloads).

**Installation:**

1. **Extract Kafka**: After downloading Kafka, extract the zip file to a directory of your choice (e.g., C:\kafka).
2. **Set Environment Variables**: Add the bin directory of your Kafka installation to the PATH environment variable. This will allow you to run Kafka commands from the command prompt.

* Right-click on This PC → Properties → Advanced system settings → Environment Variables.
* Under System Variables, find the Path variable, select it, and click Edit.
* Add the path to the Kafka bin directory (e.g., C:\kafka\bin) and click OK.

**Step 2: Starting Zookeeper**

Kafka relies on Zookeeper for managing and coordinating the Kafka brokers. Before starting Kafka, you need to start Zookeeper.

* Open a command prompt and navigate to the Kafka directory (e.g., C:\kafka).
* Run the following command to start Zookeeper:

**.\bin\windows\zookeeper-server-start.bat .\config\zookeeper.properties**

This command starts Zookeeper using the default configuration provided in the zookeeper.properties file. You should see logs indicating that Zookeeper has started successfully.

**Step 3: Starting the Kafka Server**

Once Zookeeper is up and running, you can start the Kafka server.

* Open another command prompt and navigate to the Kafka directory.
* Run the following command to start the Kafka server:

**.\bin\windows\kafka-server-start.bat .\config\server.properties**

This command starts the Kafka broker using the default configuration provided in the server.properties file. You should see logs indicating that the Kafka server is up and running.

**Step 4: Creating a Kafka Topic**

In Kafka, data is organized into topics. A topic is a logical channel to which producers send messages and consumers read them.

* Open another command prompt and navigate to the Kafka directory.
* Run the following command to create a new topic:

**.\bin\windows\kafka-topics.bat --create --topic my-topic --bootstrap-server localhost:9092 --partitions 1 --replication-factor 1**

* --topic my-topic: This creates a topic named "my-topic".
* --bootstrap-server localhost:9092: Specifies the Kafka broker's address.
* --partitions 1: Creates one partition for this topic.
* --replication-factor 1: Sets the replication factor to 1 (no replication).

You should see a message confirming that the topic has been created successfully.

**Step 5: Producing Messages to a Kafka Topic**

Now that you have a topic, you can start sending messages to it using a Kafka producer.

* Open another command prompt and navigate to the Kafka directory.
* Run the following command to start the Kafka producer:

**.\bin\windows\kafka-console-producer.bat --topic my-topic --bootstrap-server localhost:9092**

This will start a console where you can type messages. Each line you type will be sent as a message to the “my-topic” topic.

**Step 6: Consuming Messages from a Kafka Topic**

Finally, let’s consume the messages we just produced using a Kafka consumer.

* Open another command prompt and navigate to the Kafka directory.
* Run the following command to start the Kafka consumer:

**.\bin\windows\kafka-console-consumer.bat --topic my-topic --from-beginning --bootstrap-server localhost:9092**

--from-beginning: This tells the consumer to read messages from the beginning of the topic.

You should see the messages you produced earlier displayed in the console.

**Conclusion**

Congratulations! You’ve successfully installed and run Apache Kafka on Windows. You also learned how to start Zookeeper, run the Kafka server, create topics, produce messages, and consume messages. While this guide covers the basics, Kafka is a powerful tool with many advanced features, such as distributed processing, stream processing, and more.

As you continue to explore Kafka, you’ll discover its full potential in building real-time data pipelines and streaming applications. Whether you’re building a small project or a large-scale system, Kafka’s scalability, fault tolerance, and high throughput make it an excellent choice.

If you have any questions or run into issues, feel free to leave a comment below. Happy streaming with Kafka!