It is publisher subscribe model, where in messages are published by publisher and topics are consumed by the consumer to subscribed topics.

**1. Producer**

* **Definition**: A Kafka producer is a client that sends records (messages) to Kafka brokers.
* **Function**: Producers publish data to specific Kafka **topics**. They can send messages to a specific partition of a topic or allow Kafka to distribute them across partitions.
* **Key Feature**: Producers can operate in sync (wait for acknowledgement) or async (fire-and-forget mode) based on the level of reliability desired.

**2. Consumer**

* **Definition**: A consumer is a client that reads records from Kafka topics.
* **Function**: Consumers subscribe to one or more topics and fetch messages, processing them as they arrive.
* **Key Feature**: Kafka consumers are usually part of a consumer group. Each message in a partition is consumed by only one consumer within that group, providing scalability and parallelism.

**3. Topic**

* **Definition**: A topic in Kafka is a category or stream name to which records are sent by producers and from which records are read by consumers.
* **Function**: Topics are split into multiple partitions for scalability, and each partition can be consumed by only one consumer at a time within the same group.
* **Key Feature**: Topics allow multiple producers to publish and multiple consumers to subscribe, providing decoupling between producers and consumers.

**4. Partition**

* **Definition**: Partitions are a key unit of parallelism in Kafka, each topic consists of one or more partitions.
* **Function**: Each partition contains a sequence of messages and Kafka ensures that messages in a partition are stored in the order they were received. Messages within a partition are assigned offsets, which ensure unique and ordered identification.
* **Key Feature**: Partitions allow Kafka to scale horizontally across multiple brokers. They enable parallel processing by distributing the data across multiple consumers.

**5. Zookeeper**

* **Definition**: Zookeeper is a centralized service used by Kafka for managing configuration and coordination between brokers in the cluster.
* **Function**: It maintains metadata, such as the location of partitions, leader election for partition replicas, and state information.
* **Key Feature**: Zookeeper ensures that the Kafka cluster operates correctly, though newer versions of Kafka are moving towards removing Zookeeper dependency (via **KRaft**, Kafka’s own quorum-based controller).

**6. Offset**

* **Definition**: The offset is a unique identifier assigned to each message within a partition.
* **Function**: It keeps track of the position of a consumer within a partition. Consumers can resume from a specific offset in case of failure or restart.
* **Key Feature**: Offsets provide a mechanism for maintaining the order and state in which messages are consumed, and Kafka consumers can commit offsets to maintain a checkpoint of processed messages.
* Together, these components make Kafka a powerful, distributed event-streaming platform capable of handling large-scale data streams in real-time.
* **Each partition** of a topic is consumed by only **one consumer** within the same consumer group.

+-------------+

| Topic |

+-------------+

| Partition 0 | -- Consumer A (reads all messages in order from Partition 0)

| Partition 1 | -- Consumer B (reads all messages in order from Partition 1)

| Partition 2 | -- Consumer C (reads all messages in order from Partition 2)

* Zookeeper port: 2181 and Kafka Server port: 9092.

**Open Source Kafka Startup in local**

1. Start Zookeeper Server

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

1. Start Kafka Server / Broker

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

1. Create topic

bin/kafka-topics.sh --bootstrap-server localhost:9092 --create --topic NewTopic --partitions 3 --replication-factor 1

1. list out all topic names

bin/kafka-topics.sh --bootstrap-server localhost:9092 --list

1. Describe topics

bin/kafka-topics.sh --bootstrap-server localhost:9092 --describe --topic NewTopic

1. Produce message

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

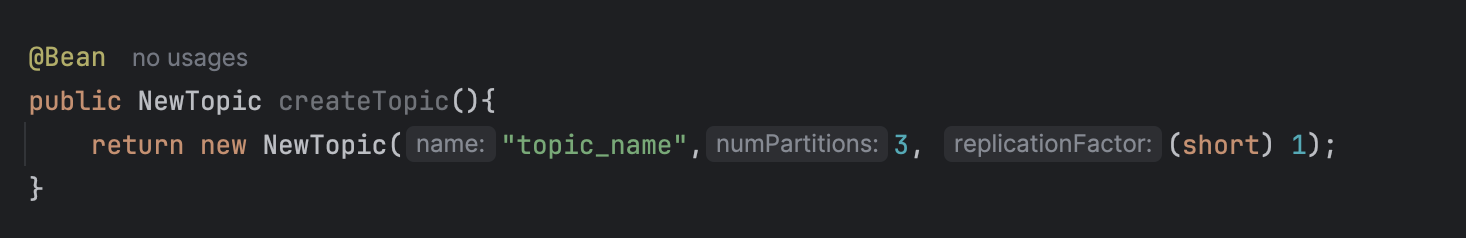
1. consume message

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

## Kafka Using Spring:

* Spring Dependency that needs to be added Spring for kafka

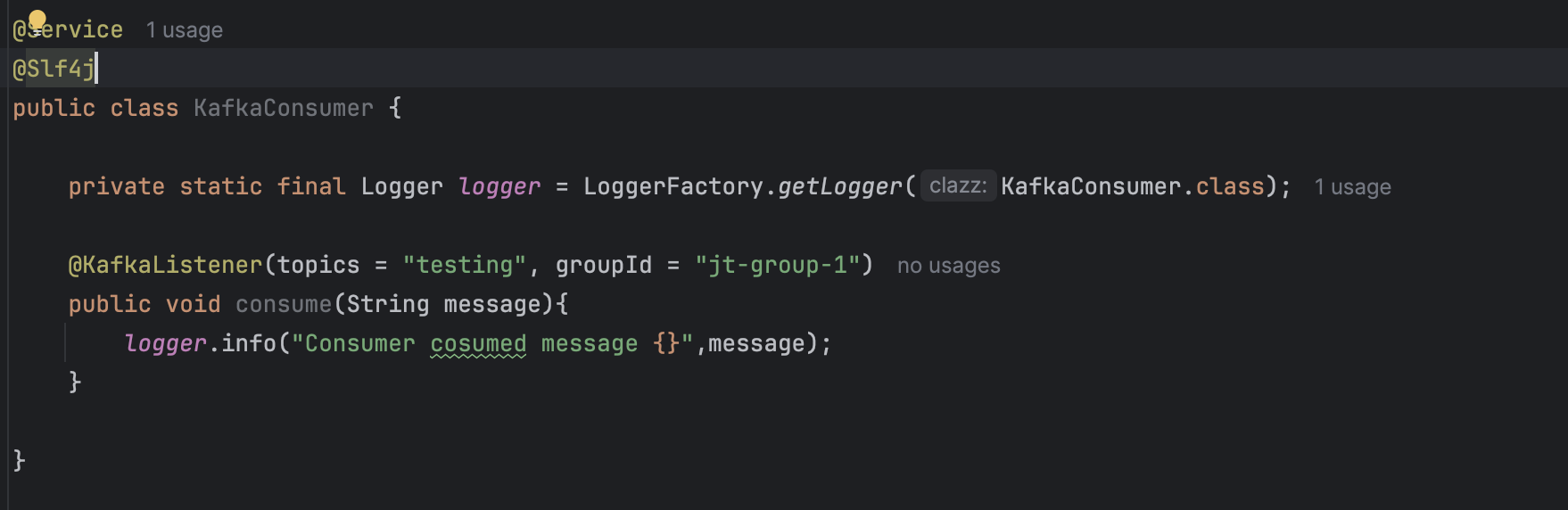
### Creation of topic in spring



### Kafka Publisher creation



### Kafka Listener (1 Consumer to Many Partitioners)



* Following changes are to be made in application properties and group-id should be the same id as mentioned in .properties file.

spring.kafka.consumer.builder()  
 .bootstrap-servers = localhost:9092  
 .group-id = jt-group-1  
 .build()