Tristan Mahinay

Technical Lead, Java and watsonx SME and Open Source Committer @ IBM
JUG PH Leader @ Java User Group Philippines

rjtmahinay/Tristan Mahinay

Java User Group
JUG
Philippines

Practical Spring for Apache Kafka

Kafka Applications via Dependency Injection and Inversion of Control Principle

Fundamentals

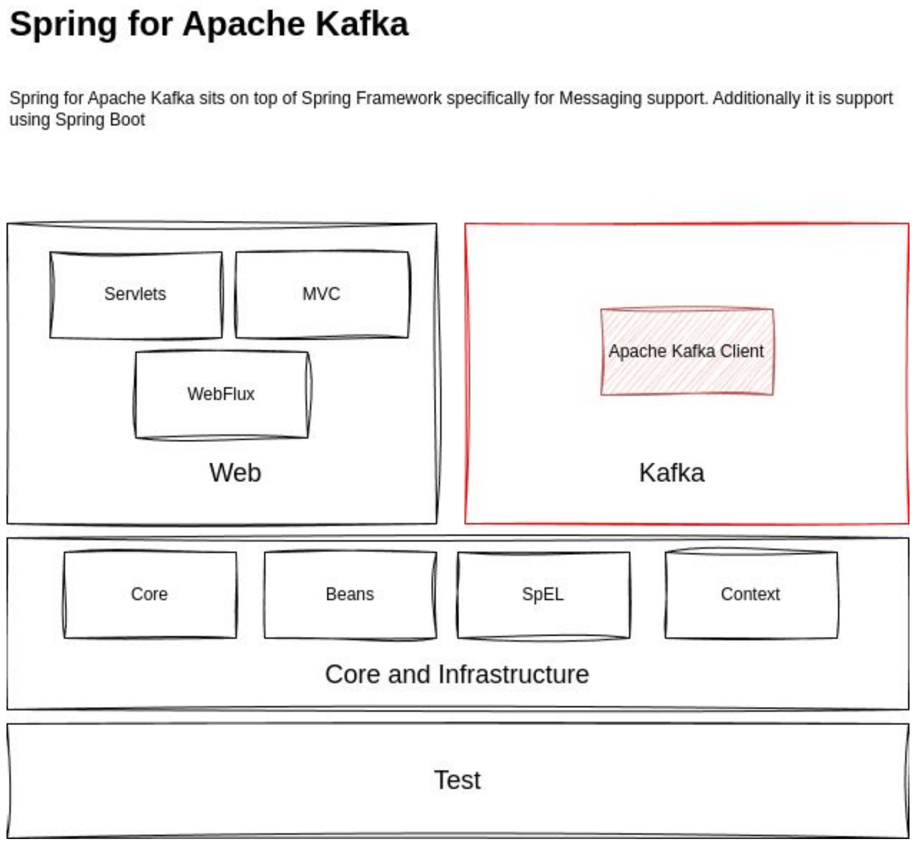
What is Spring for Apache Kafka?

- The Spring Framework abstraction of Apache Kafka to create Kafka-based application
- Used of Dependency Injection and Inversion of Control when creating Kafka-based application
- Managed Apache Kafka client by Spring

The Architecture

Spring for Apache Kafka is managed by Spring Framework. It works with the current Spring Components spanning from Core, Infrastructure, Web, and Test.

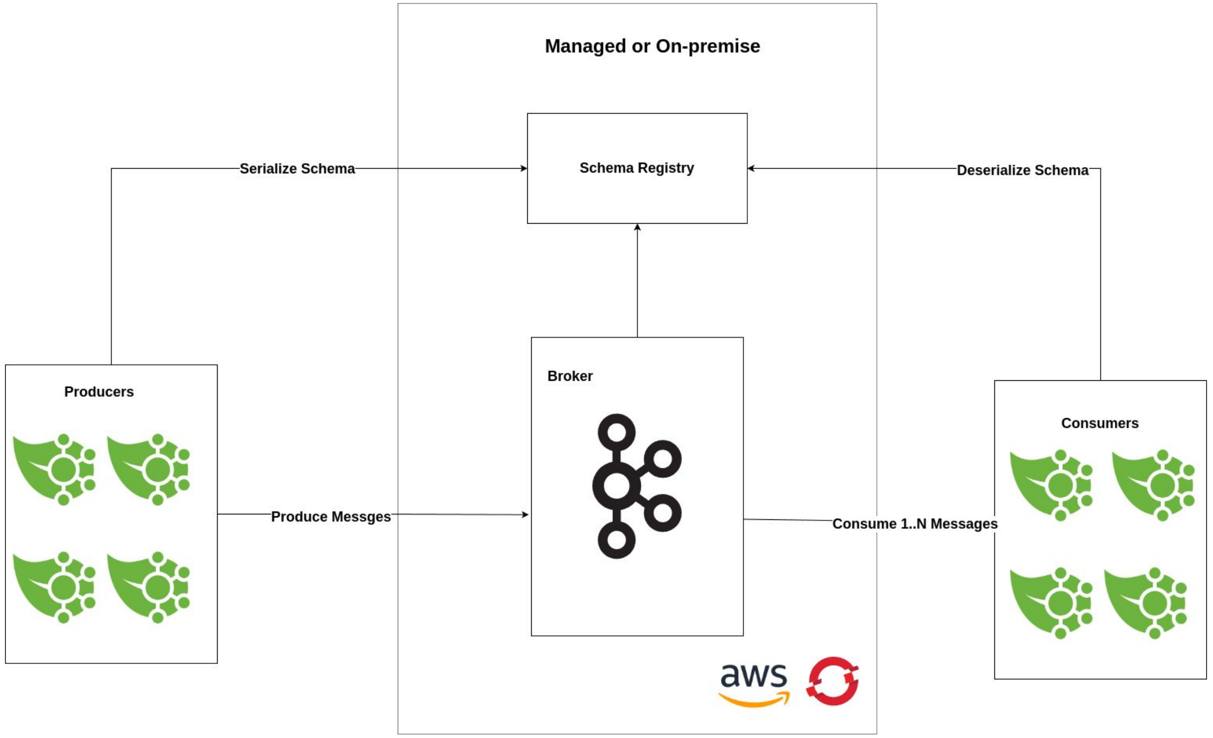
Spring for Apache Kafka today is used together with Spring Boot.


Spring for Apache Kafka
Spring for Apache Kafka sits on top of Spring Framework specifically for Messaging support. Additionally it is support using Spring Boot
Servlets
MVC
WebFlux
Web
Apache Kafka Client
Kafka
Core
Beans
SpEL
Context
Core and Infrastructure
Test

Kafka Architecture

The architecture consists of producers, consumers and a broker.

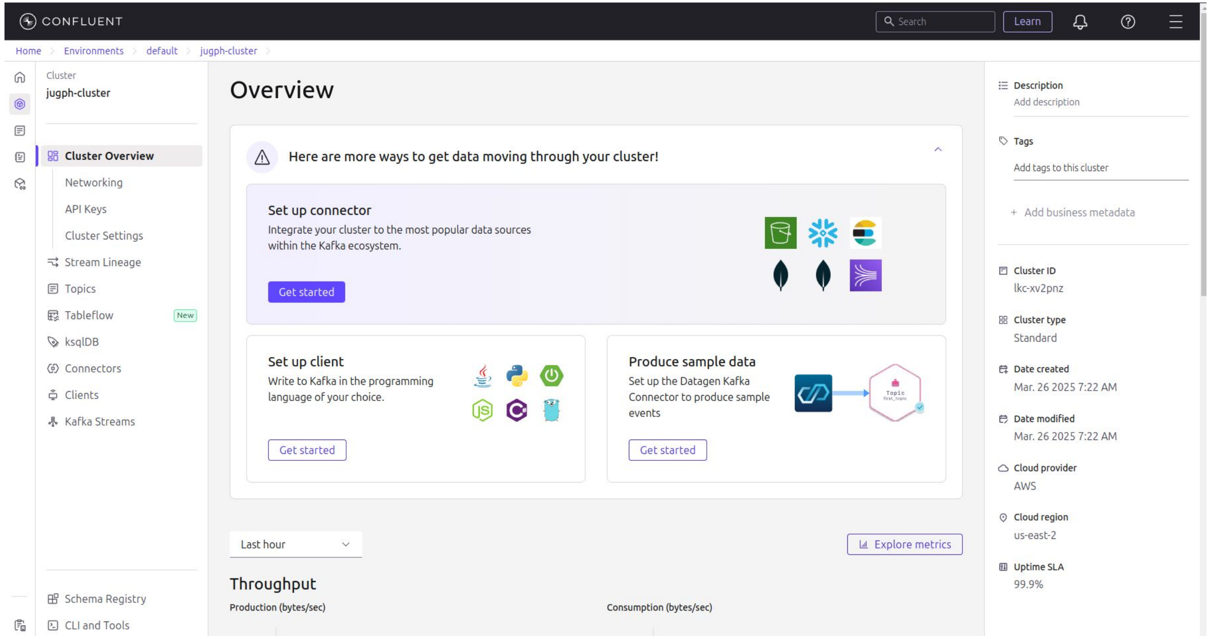
- A **producer** can produce 1..N messages in a given topic. It can produce to multiple topics
- A **consumer** can consume 1..N message from a given topic. Consumers is grouped via a consumer group.
- The **broker** is responsible for managing the topics, partitions, and consumer groups. Additionally, it manages transaction ids for transactional messages.
- The **schema registry** stores the schema being serialized and deserialized through the network. Apache Avro is the common schema used.



Confluent Cloud

A fully managed, cloud-native data streaming platform built around Apache Kafka.

Essentially, it takes the power of Kafka and delivers it as a service, removing the complexities of infrastructure management



Cluster
jugph-cluster

Overview

Here are more ways to get data moving through your cluster!

Set up connector
Integrate your cluster to the most popular data sources within the Kafka ecosystem.

Set up client
Write to Kafka in the programming language of your choice.

Produce sample data
Set up the Datagen Kafka Connector to produce sample events

Throughput
Production (bytes/sec)
Consumption (bytes/sec)

Description
Add description

Tags
Add tags to this cluster
+ Add business metadata

Cluster ID
lkc-xv2pnz

Cluster type
Standard

Date created
Mar. 26 2025 7:22 AM

Date modified
Mar. 26 2025 7:22 AM

Cloud provider
AWS

Cloud region
us-east-2

Uptime SLA
99.9%

Advantage of a Managed Kafka

- Reduced Operational Overhead
- Scalability and Elasticity
- Cost Optimization
- Increased Reliability and Availability
- Faster Time to Market
- Focus on Core Business

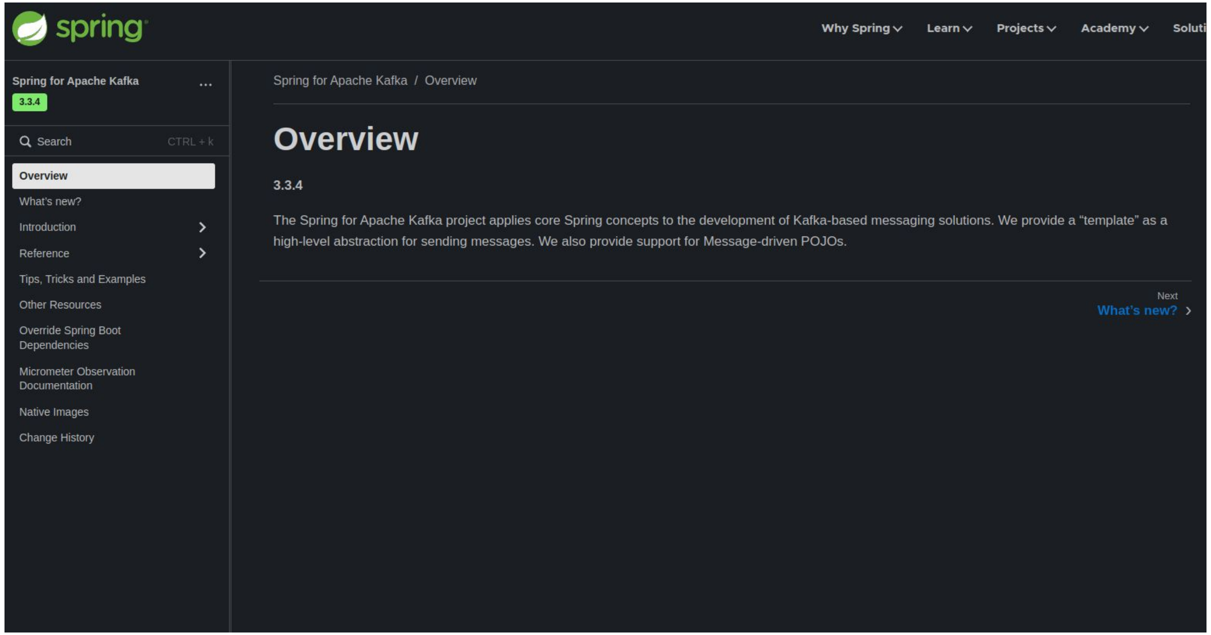
How to develop?

How to get started?

Spring Framework documentation consist of rich information about different Spring components showing its reference and API documentation.

You can learn Spring for Apache Kafka by visiting

https://docs.spring.io/spring-kafka/reference/index.html

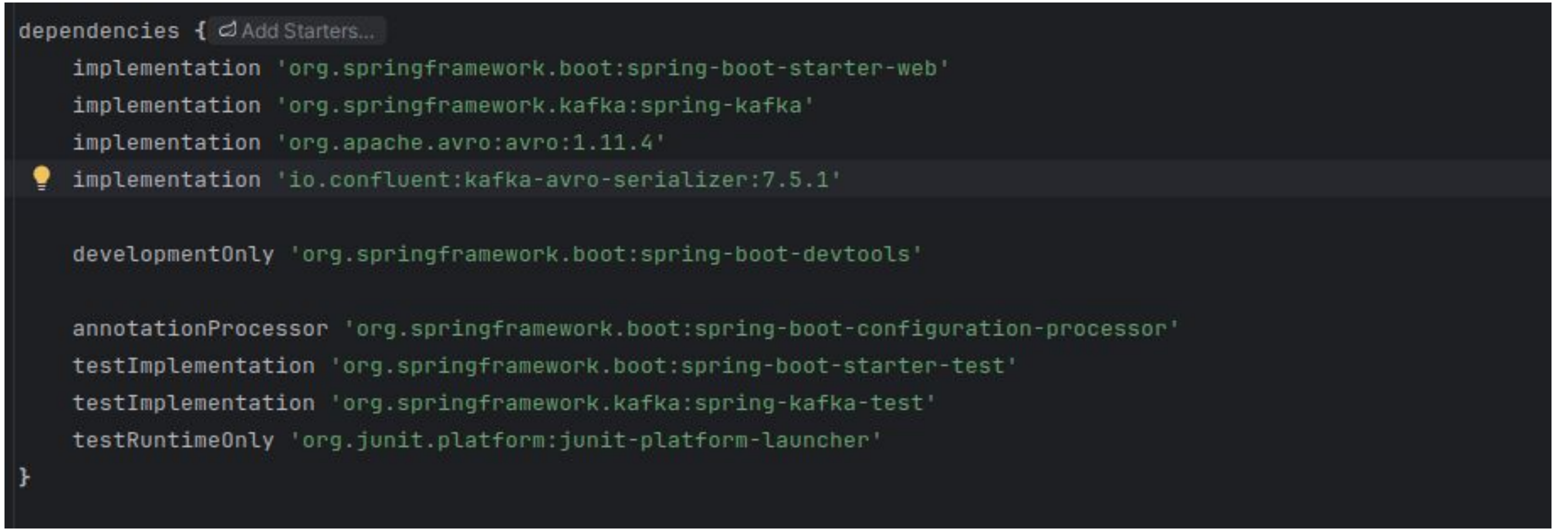


Overview

3.3.4

The Spring for Apache Kafka project applies core Spring concepts to the development of Kafka-based messaging solutions. We provide a “template” as a high-level abstraction for sending messages. We also provide support for Message-driven POJOs.

Dependencies



```
dependencies {
    implementation 'org.springframework.boot:spring-boot-starter-web'
    implementation 'org.springframework.kafka:spring-kafka'
    implementation 'org.apache.avro:avro:1.11.4'
    implementation 'io.confluent:kafka-avro-serializer:7.5.1'

    developmentOnly 'org.springframework.boot:spring-boot-devtools'

    annotationProcessor 'org.springframework.boot:spring-boot-configuration-processor'
    testImplementation 'org.springframework.boot:spring-boot-starter-test'
    testImplementation 'org.springframework.kafka:spring-kafka-test'
    testRuntimeOnly 'org.junit.platform:junit-platform-launcher'
}
```

Annotations and Classes

In this session we will discuss how to quickly implement a Kafka application using the following annotations to push and consume a message to a Kafka broker.

- ProducerFactory<K, V>
- ConsumerFactory<K, V>
- KafkaTemplate<K, V>
- ConsumerRecord<K, V>
- Kafka
- @PayLoad
- NewTopic
- KafkaTransactionManager

DEMO

Kafka Producer

Kafka Producer Service Repository

Kafka Consumer

Kafka Consumer Service Repository

Connect with me!