Springwolf

1.it is use for async api doucmentation (same like swagger but for async api)

2.use for message driven platform (kafka,rabbitmq,apachemq)

Example:-

I have 2 microservice ,Loan services and credit services both are producer and consumer so how to integrate springwolf in microservices , sprinboot project?

(in kafka)

springwolf in LOAN SERVICE :-

1.add depedency

<!-- SpringWolf dependency for Kafka -->

```
<dependency>
    <groupId>io.github.springwolf</groupId>
    <artifactId>springwolf-kafka</artifactId>
    <version>1.12.0</version>
</dependency>

<!-- SpringWolf-UI dependency -->
<dependency>
    <groupId>io.github.springwolf</groupId>
    <artifactId>springwolf-ui</artifactId>
    <version>1.12.0</version>
</dependency>
```

2. application.yml

```
server:
 port: 9191


spring:
 application:
  name: loan-service


 kafka:
  bootstrap-servers: localhost:9092
  producer:
   key-serializer: org.apache.kafka.common.serialization.StringSerializer
   value-serializer: org.springframework.kafka.support.serializer.JsonSerializer
  consumer:
   group-id: loan-group
   auto-offset-reset: latest
   key-deserializer: org.apache.kafka.common.serialization.StringDeserializer
   value-deserializer: org.springframework.kafka.support.serializer.JsonDeserializer
   properties:
    spring.json.trusted.packages: "*"


loan:
 processing:
  topic-name: loan-process-topic6
```

```
credit:
  decision:
    topic-name: credit-decision-topic6

springwolf:
  enabled: true
  plugin:
    kafka:
      publishing:
        enabled: true
        producer:
          bootstrap-servers:
            - localhost:9092
          key-serializer: org.apache.kafka.common.serialization.StringSerializer
          value-serializer: org.springframework.kafka.support.serializer.JsonSerializer
      consumer:
        bootstrap-servers:
          - localhost:9092
        key-deserializer: org.apache.kafka.common.serialization.StringDeserializer
        value-deserializer: org.springframework.kafka.support.serializer.JsonDeserializer

  docket:
    base-package: com.javatechie
    info:
      title: ${spring.application.name}
```

```
  version: 1.0.0
  description: Loan service with producer and consumer
servers:
  kafka-server:
    protocol: kafka
    host: localhost:9092
```

step 3 :-

Publisher for LoanEvent

```
  @Autowired
  private KafkaTemplate<String, Object> kafkaTemplate;

  @Value("${loan.processing.topic-name}")
  private String topic;

  @AsyncPublisher(operation = @AsyncOperation(
      channelName = "loan-process-topic6",
      description = "Publish loan application events to Credit Service"
  ))
  @KafkaAsyncOperationBinding
  public void publishLoanSubmitKafkaEvent(LoanApplicationSubmitEvent event) {
    kafkaTemplate.send(topic, event);
    log.info("Published loan event to topic: {}", topic);
  }
}
```

step 4:-

Listener for CreditDecision

```
   @AsyncListener(operation = @AsyncOperation(
         channelName = "credit-decision-topic6",
         description = "Consume credit decision events from Credit Service"
   ))
   @KafkaAsyncOperationBinding
   @KafkaListener(topics = "credit-decision-topic6", groupId = "loan-group")
   public void consumeCreditDecision(CreditDecisionEvent event) {
      log.info("Received credit decision event: {}", event);
   }
}
```

springwolf in credit service:-

1.add depedency

```
<!-- SpringWolf dependency for Kafka -->
                    <dependency>
                        <groupId>io.github.springwolf</groupId>
                        <artifactId>springwolf-kafka</artifactId>
                        <version>1.12.0</version>
                    </dependency>

                    <!-- SpringWolf-UI dependency -->
```

```
<dependency>
    <groupId>io.github.springwolf</groupId>
    <artifactId>springwolf-ui</artifactId>
    <version>1.12.0</version>
</dependency>
```

step 2: -

application.yml

```
server:
  port: 9292

spring:
  application:
    name: credit-service

  kafka:
    bootstrap-servers: localhost:9092
    producer:
      key-serializer: org.apache.kafka.common.serialization.StringSerializer
      value-serializer: org.springframework.kafka.support.serializer.JsonSerializer
    consumer:
      group-id: credit-group
      auto-offset-reset: latest
      key-deserializer: org.apache.kafka.common.serialization.StringDeserializer
      value-deserializer: org.springframework.kafka.support.serializer.JsonDeserializer
```

```
    properties:
      spring.json.trusted.packages: "*"

loan:
  processing:
    topic-name: loan-process-topic6

credit:
  decision:
    topic-name: credit-decision-topic6

springwolf:
  enabled: true
  plugin:
    kafka:
      publishing:
        enabled: true
        producer:
          bootstrap-servers:
            - localhost:9092
          key-serializer: org.apache.kafka.common.serialization.StringSerializer
          value-serializer: org.springframework.kafka.support.serializer.JsonSerializer
      consumer:
        bootstrap-servers:
          - localhost:9092
```

```
    key-deserializer: org.apache.kafka.common.serialization.StringDeserializer
    value-deserializer: org.springframework.kafka.support.serializer.JsonDeserializer

docket:
  base-package: com.javatechie
  info:
    title: ${spring.application.name}
    version: 1.0.0
    description: Credit service with producer and consumer
  servers:
    kafka-server:
      protocol: kafka
      host: localhost:9092
```

step3:-

CreditService – Listener for Loan Applications

```
@Component
@Slf4j
public class LoanApplicationEventListener {

  @AsyncListener(operation = @AsyncOperation(
        channelName = "loan-process-topic6",
        description = "Consume loan applications from Loan Service"
  ))
```

```
    @KafkaAsyncOperationBinding
    @KafkaListener(topics = "loan-process-topic6", groupId = "credit-group")
    public void consumeLoanEvent(LoanApplicationSubmitEvent event) {
        log.info("Received loan application: {}", event);
        // Simulate decision and publish
    }
}
```

step4:-

CreditService – Publisher for Credit Decision

```
@Component
@Slf4j
public class CreditDecisionPublisher {

    @Autowired
    private KafkaTemplate<String, Object> kafkaTemplate;

    @Value("${credit.decision.topic-name}")
    private String topic;

    @AsyncPublisher(operation = @AsyncOperation(
            channelName = "credit-decision-topic6",
            description = "Publish credit decisions to Loan Service"
    ))
```

```
    @KafkaAsyncOperationBinding
    public void publishCreditDecision(CreditDecisionEvent event) {
        kafkaTemplate.send(topic, event);
        log.info("Published credit decision: {}", event);
    }
}
```

✅ Springwolf URLs

Loan Service: http://localhost:9191/springwolf/docs.html or /sprinwolf/sync-api/uI-html

Credit Service: http://localhost:9292/springwolf/docs.html

(in Rabbitmq)

Dependency in both:-

```
<!-- Spring Boot RabbitMQ -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-amqp</artifactId>
</dependency>

<!-- Springwolf for RabbitMQ -->
<dependency>
```

```
  <groupId>io.github.springwolf</groupId>
  <artifactId>springwolf-amqp</artifactId>
  <version>1.12.0</version>
</dependency>

<!-- Springwolf UI (optional) -->
<dependency>
  <groupId>io.github.springwolf</groupId>
  <artifactId>springwolf-ui</artifactId>
  <version>1.12.0</version>
</dependency>
```

Common Constants

```
public class RabbitMQConstants {
   public static final String EXCHANGE = "loan-credit-exchange";
   public static final String LOAN_QUEUE = "loan-application-queue";
   public static final String CREDIT_QUEUE = "credit-decision-queue";
   public static final String LOAN_ROUTING_KEY = "loan.submit";
   public static final String CREDIT_ROUTING_KEY = "credit.decision";
}
```

LoanService

application.yml

```
server:
  port: 9191
```

```
spring:
  application:
    name: loan-service

  rabbitmq:
    host: localhost
    port: 5672
    username: guest
    password: guest

springwolf:
  enabled: true
  plugin:
    amqp:
      publishing:
        enabled: true
      consuming:
        enabled: true
  docket:
    base-package: com.loan
    info:
      title: Loan Service
      version: 1.0
```

description: Loan service using RabbitMQ with Exchange

RabbitMQConfig.java

```
@Configuration
public class RabbitMQConfig {

  @Bean
  public DirectExchange exchange() {
    return new DirectExchange(RabbitMQConstants.EXCHANGE);
  }

  @Bean
  public Queue loanQueue() {
    return new Queue(RabbitMQConstants.LOAN_QUEUE);
  }

  @Bean
  public Queue creditQueue() {
    return new Queue(RabbitMQConstants.CREDIT_QUEUE);
  }

  @Bean
  public Binding loanBinding() {
    return BindingBuilder
        .bind(loanQueue())
```

```
            .to(exchange())
            .with(RabbitMQConstants.LOAN_ROUTING_KEY);
    }

    @Bean
    public Binding creditBinding() {
        return BindingBuilder
            .bind(creditQueue())
            .to(exchange())
            .with(RabbitMQConstants.CREDIT_ROUTING_KEY);
    }
}
```

LoanSubmitEventPublisher.java

```
@Component
@Slf4j
public class CreditDecisionListener {

    @AsyncListener(operation = @AsyncOperation(
            channelName = RabbitMQConstants.EXCHANGE,
            description = "Receive credit decision from Credit Service"
    ))
    @AmqpAsyncOperationBinding
    @RabbitListener(queues = RabbitMQConstants.CREDIT_QUEUE)
    public void receiveCreditDecision(CreditDecision decision) {
        log.info("Received Credit Decision: {}", decision);
    }
}
```

CreditDecisionListener.java

```
@Component@Slf4jpublic class CreditDecisionListener {

   @AsyncListener(operation = @AsyncOperation(
         channelName = RabbitMQConstants.EXCHANGE,
         description = "Receive credit decision from Credit Service"
   ))
   @AmqpAsyncOperationBinding
   @RabbitListener(queues = RabbitMQConstants.CREDIT_QUEUE)
   public void receiveCreditDecision(CreditDecision decision) {
      log.info("Received Credit Decision: {}", decision);
   }
}
```

CreditService

application.yml

yaml
CopyEdit

```
server:
  port: 9292
spring:
  application:
    name: credit-service

  rabbitmq:
    host: localhost
    port: 5672
    username: guest
    password: guest
springwolf:
  enabled: true
  plugin:
    amqp:
      publishing:
        enabled: true
      consuming:
        enabled: true
  docket:
    base-package: com.credit
    info:
      title: Credit Service
      version: 1.0
```

```
      description: Credit service using RabbitMQ with Exchange
```

CreditDecisionPublisher.java

java
CopyEdit

```
@Component@Slf4jpublic class CreditDecisionPublisher {

    @Autowired
    private AmqpTemplate rabbitTemplate;

    @AsyncPublisher(operation = @AsyncOperation(
            channelName = RabbitMQConstants.EXCHANGE,
            description = "Send credit decision to Loan Service"
    ))
```

```
    @AmqpAsyncOperationBinding
    public void sendCreditDecision(CreditDecision decision) {
        rabbitTemplate.convertAndSend(
            RabbitMQConstants.EXCHANGE,
            RabbitMQConstants.CREDIT_ROUTING_KEY,
            decision
        );
        log.info("Credit Decision sent using routing key {}",
RabbitMQConstants.CREDIT_ROUTING_KEY);
    }
```

```
}
```

LoanApplicationListener.java

```
java
CopyEdit
@Component@Slf4jpublic class LoanApplicationListener {

    @AsyncListener(operation = @AsyncOperation(
            channelName = RabbitMQConstants.EXCHANGE,
            description = "Receive loan application from Loan Service"
    ))
    @AmqpAsyncOperationBinding
    @RabbitListener(queues = RabbitMQConstants.LOAN_QUEUE)
    public void receiveLoanApplication(LoanApplication application) {
        log.info("Received Loan Application: {}", application);
    }
}
```