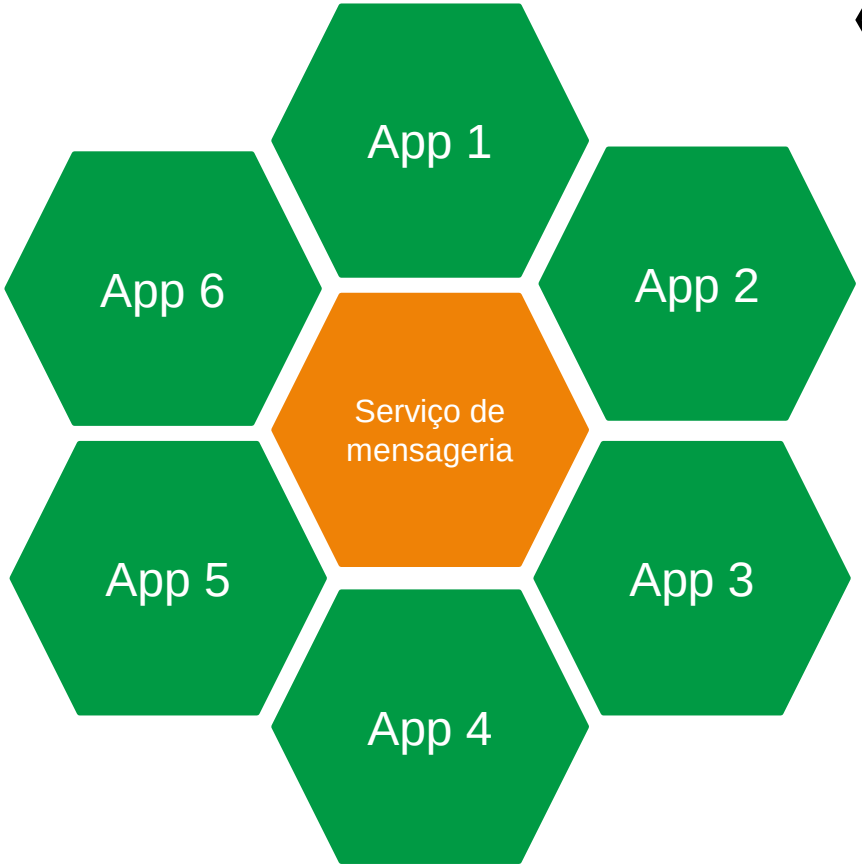
THE DEVELOPER'S CONFERENCE

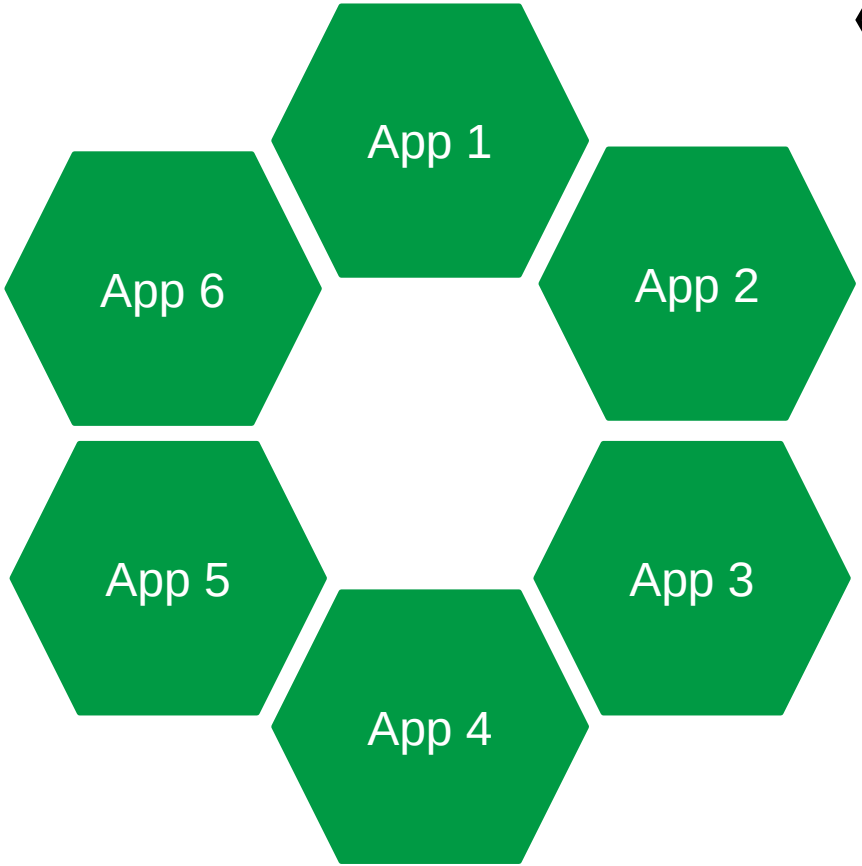
Trilha – Java

O que você precisa saber sobre serviços de mensageria em Java

Renan Roggia e Willian Alves
Software Developers @ SAP

Reduzido


App 1
App 2
App 3
App 4
App 5
App 6
Serviço de mensageria



Agenda

THE DEVELOPER'S CONFERENCE

- Serviços de mensageria
  - O que são?
  - Para que servem?
  - Vantagens
- Protocolos e APIs
- Taxonomia de um serviço de mensageria
- Demos e Exchanges
- Sugestões de leitura
- Perguntas

O que são?

THE DEVELOPER'S CONFERENCE

- Serviços independentes orientados a mensagens
  - *Message-Oriented **Middleware** (MOM)*
  - *Messaging System*
  - *Messaging Service*
- Exemplos: ActiveMQ, RabbitMQ, Google PubSub

Para que servem?

THE DEVELOPER'S CONFERENCE

- Troca de informações **entre** aplicações
  - Assíncrono
  - Através de mensagens

Vantagens

THE DEVELOPER'S CONFERENCE

- Assíncrono *(Fire and Forget)*
- Desacoplamento entre aplicações
- Resiliência
- *Load balancing* das aplicações

Vantagens

THE DEVELOPER'S CONFERENCE

- Assíncrono *(Fire and Forget)*
- Desacoplamento entre aplicações
- Resiliência
- *Load balancing* das aplicações

Vantagens

THE DEVELOPER'S CONFERENCE

- Assíncrono *(Fire and Forget)*
- Desacoplamento entre aplicações
- Resiliência
- *Load balancing* das aplicações

Vantagens

THE DEVELOPER'S CONFERENCE

- Assíncrono *(Fire and Forget)*
- Desacoplamento entre aplicações
- Resiliência
- *Load balancing* das aplicações

Vantagens

THE DEVELOPER'S CONFERENCE

- Assíncrono *(Fire and Forget)*
- Desacoplamento entre aplicações
- Resiliência
- *Load balancing* das aplicações

Protocolos existentes

THE DEVELOPER'S CONFERENCE

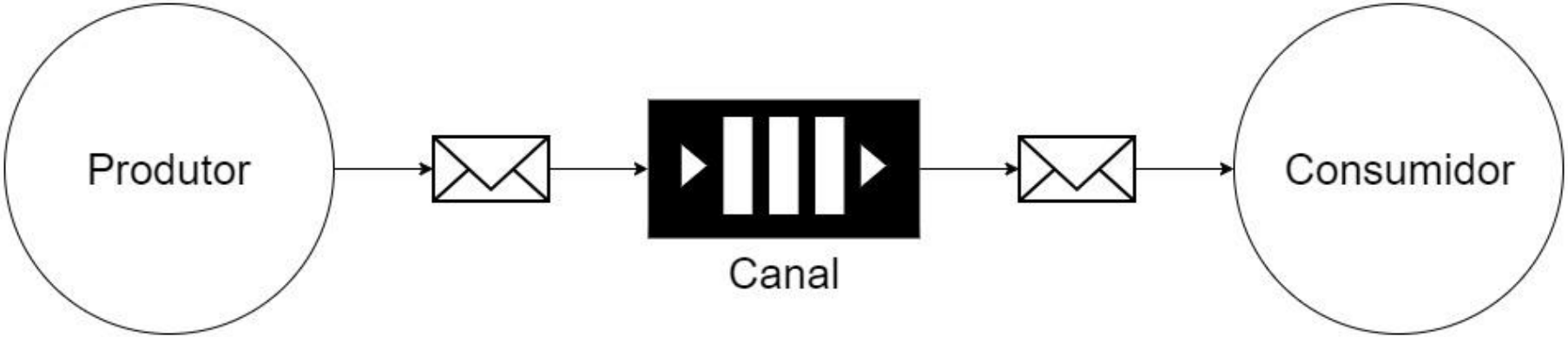
- AMQP (0.9.1 e 1.0)
- MQTT
- Stomp
- gRPC
- HTTP
- OpenWire

APIs

THE DEVELOPER'S CONFERENCE

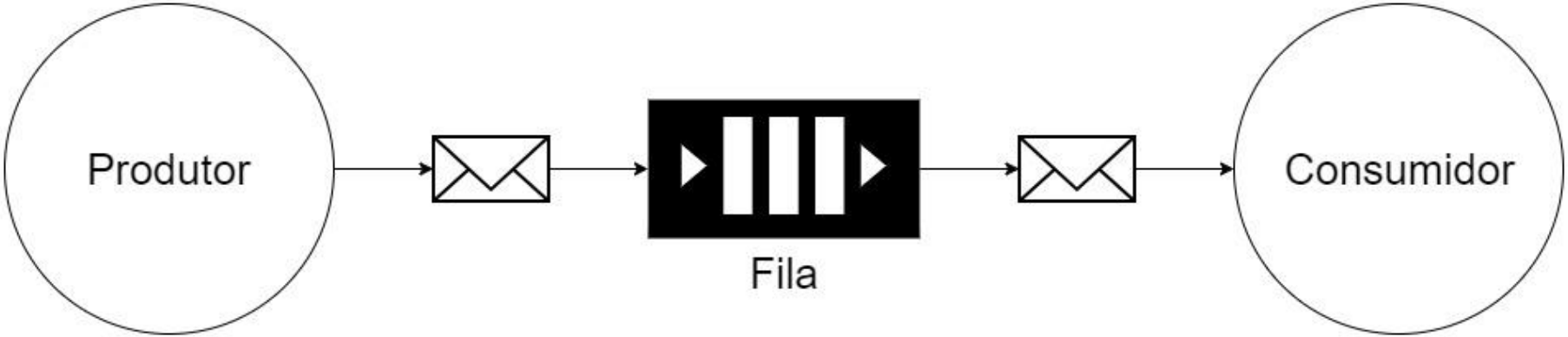
- JMS (1.0 e 2.0)
  - Qpid, active-client, rabbitmq-jms
- Spring AMQP
  - RabbitMQ
- Spring JMS
- Vendor specific
- cloudevents

Taxonomia



Demo: Queues

THE DEVELOPER'S CONFERENCE



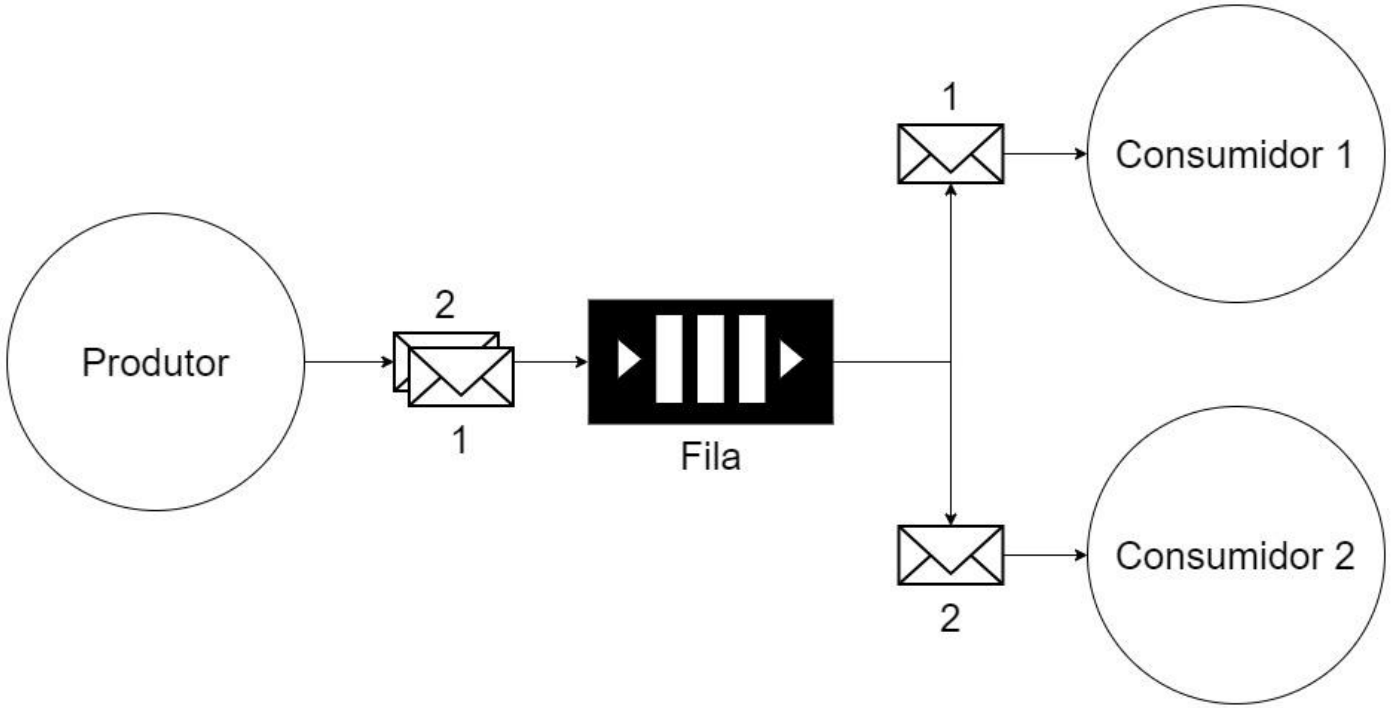
```java
@ApplicationScoped
public class Producer {
  @Inject
  ConnectionFactory connectionFactory;
  private final Random random = new Random();
  private final ScheduledExecutorService scheduler = Executors.newSingleThreadScheduledExecutor();
  private final Logger logger = LoggerFactory.getLogger(Producer.class);

  public void onStart(@Observes StartupEvent ev) {
    scheduler.scheduleWithFixedDelay(() -> {
      publishMessage();
    }, 0L, 2L, TimeUnit.SECONDS);
  }

  public void onStop(@Observes ShutdownEvent ev) {
    scheduler.shutdown();
  }

  private void publishMessage() {
    try (JMSContext context = connectionFactory.createContext()) {
      String value = Integer.toString(random.nextInt(100));
      logger.info("Publishing value: " + value);
      context.createProducer().send(context.createQueue("values-queue"), value);
    }
  }
}
```

```java
@ApplicationScoped
public class Consumer {
  @Inject
  ConnectionFactory connectionFactory;
  private final ExecutorService scheduler = Executors.newSingleThreadExecutor();
  private final Logger logger = LoggerFactory.getLogger(Consumer.class);

  public void onStart(@Observes StartupEvent ev) {
    scheduler.submit(() -> {
      consumeMessage();
    });
  }

  public void onStop(@Observes ShutdownEvent ev) {
    scheduler.shutdown();
  }

  private void consumeMessage() {
    try (JMSContext context = connectionFactory.createContext(Session.AUTO_ACKNOWLEDGE);
         JMSConsumer consumer = context.createConsumer(context.createQueue("values-queue"));) {
      while (true) {
        Message message = consumer.receive();
        logger.info("Value: " + message.getBody(String.class));
      }
    } catch (JMSException e) {
      throw new RuntimeException();
    }
  }
}
```

```
INFO  [tdc.Producer] (pool-5-thread-1) Publishing value: 70
INFO  [tdc.Consumer] (pool-6-thread-1) Value: 70
INFO  [tdc.Producer] (pool-5-thread-1) Publishing value: 58
INFO  [tdc.Consumer] (pool-6-thread-1) Value: 58
INFO  [tdc.Producer] (pool-5-thread-1) Publishing value: 2
INFO  [tdc.Consumer] (pool-6-thread-1) Value: 2
INFO  [tdc.Producer] (pool-5-thread-1) Publishing value: 5
INFO  [tdc.Consumer] (pool-6-thread-1) Value: 5
```

Demo: Competing consumers

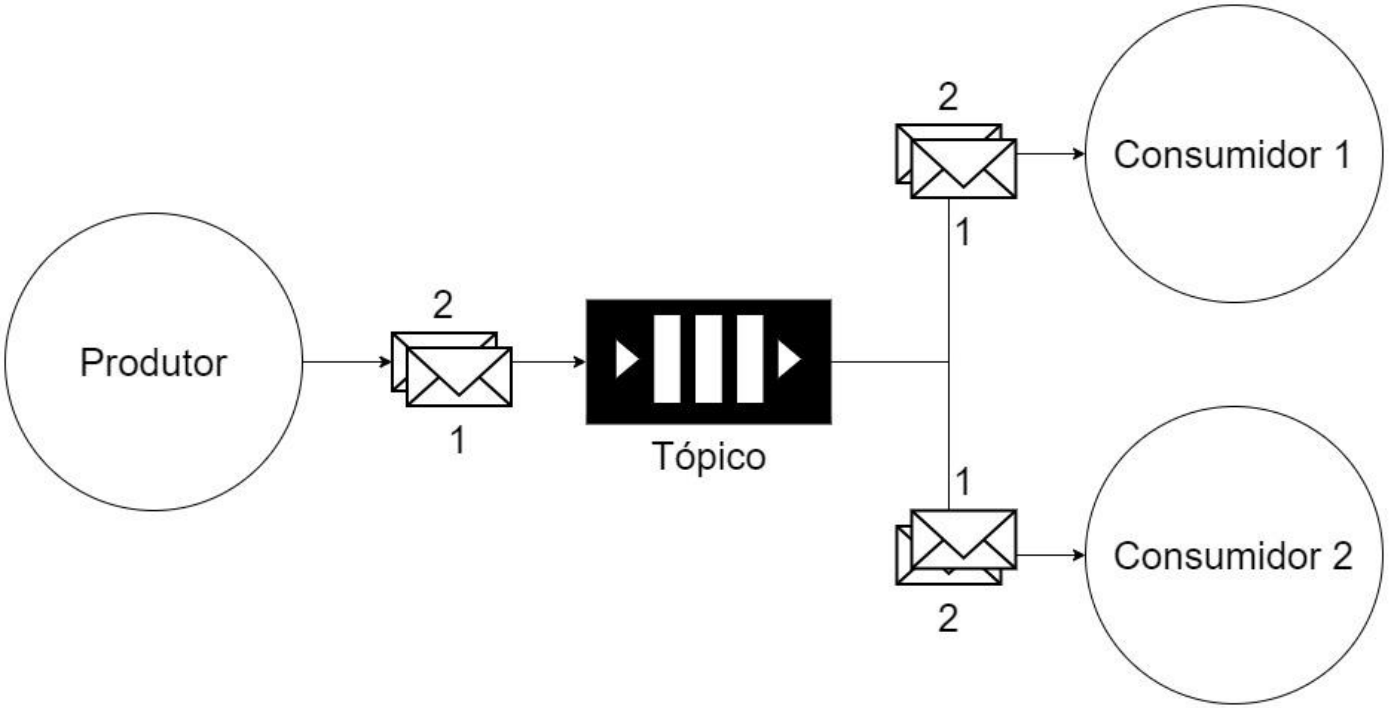
THE DEVELOPER'S CONFERENCE



```
INFO  [tdc.Producer] (pool-8-thread-1) Publishing message: 3
INFO  [tdc.FirstConsumer] (pool-7-thread-1) Value: 3
INFO  [tdc.Producer] (pool-8-thread-1) Publishing message: 34
INFO  [tdc.SecondConsumer] (pool-6-thread-1) Value: 34
INFO  [tdc.Producer] (pool-8-thread-1) Publishing message: 43
INFO  [tdc.FirstConsumer] (pool-7-thread-1) Value: 43
INFO  [tdc.Producer] (pool-8-thread-1) Publishing message: 64
INFO  [tdc.SecondConsumer] (pool-6-thread-1) Value: 64
```

Demo: Topics

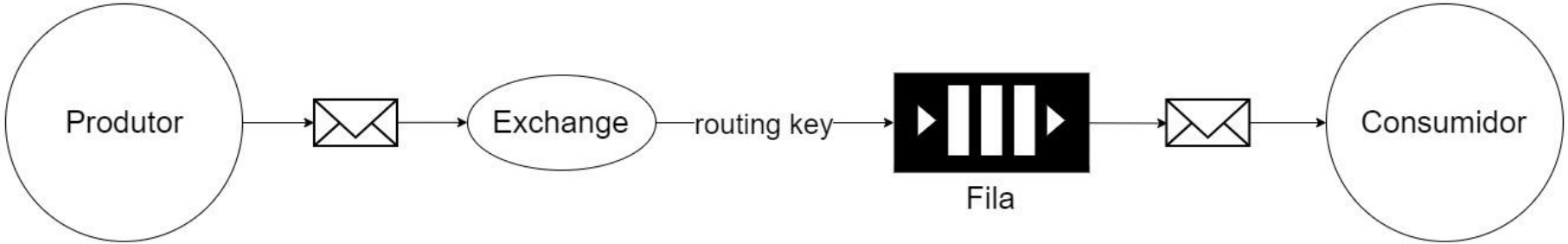
THE DEVELOPER'S CONFERENCE



```
INFO  [tdc.Producer] (pool-6-thread-1) Publishing value: 70
INFO  [tdc.FirstConsumer] (pool-8-thread-1) Value: 70
INFO  [tdc.SecondConsumer] (pool-7-thread-1) Value: 70
INFO  [tdc.Producer] (pool-6-thread-1) Publishing value: 18
INFO  [tdc.SecondConsumer] (pool-7-thread-1) Value: 18
INFO  [tdc.FirstConsumer] (pool-8-thread-1) Value: 18
INFO  [tdc.Producer] (pool-6-thread-1) Publishing value: 29
INFO  [tdc.FirstConsumer] (pool-8-thread-1) Value: 29
INFO  [tdc.SecondConsumer] (pool-7-thread-1) Value: 29
```

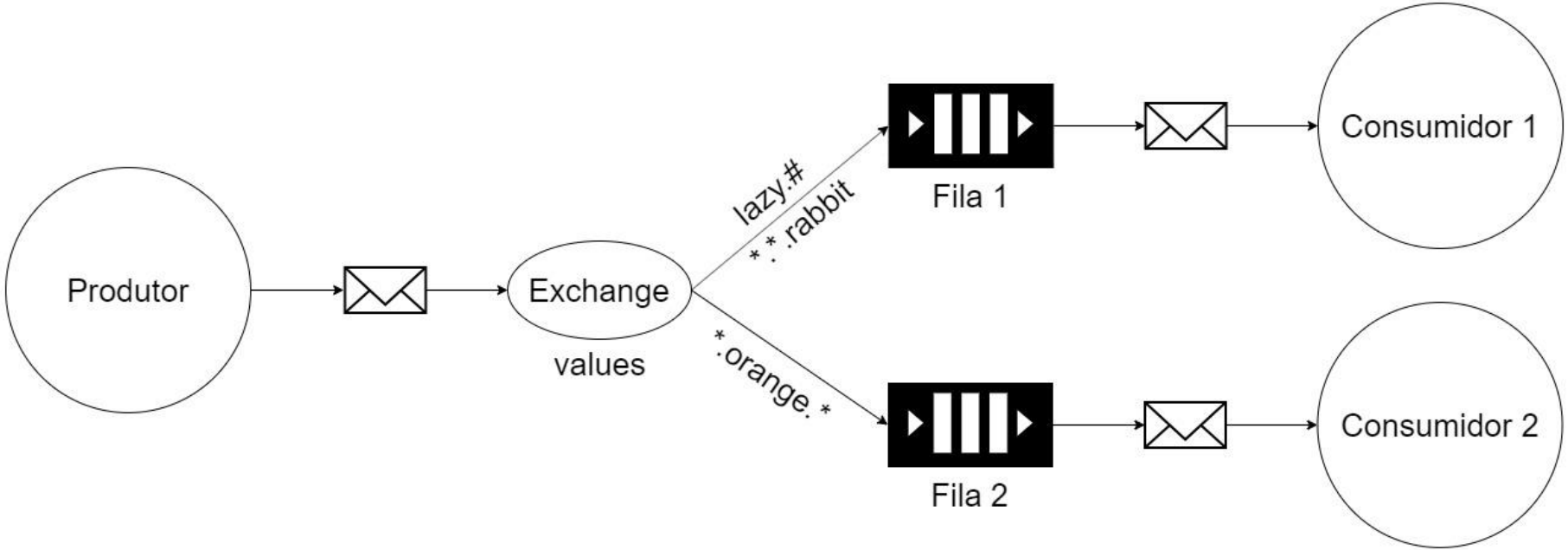
Exchanges

THE DEVELOPER'S CONFERENCE



Demo: Topic exchange

THE DEVELOPER'S CONFERENCE



INFO --- [scheduling-1] tdc.Publisher: Publishing value 28 to Routing Key **lazy.orange.elephant**
INFO --- [ntContainer#0-1] tdc.FirstConsumer: Value: 28
INFO --- [ntContainer#1-1] tdc.SecondConsumer: Value: 28
INFO --- [scheduling-1] tdc.Publisher: Publishing value 61 to Routing Key **quick.orange.fox**
INFO --- [ntContainer#1-1] tdc.SecondConsumer: Value: 61
INFO --- [scheduling-1] tdc.Publisher: Publishing value 39 to Routing Key **lazy.brown.fox**
INFO --- [ntContainer#0-1] tdc.FirstConsumer: Value: 39
INFO --- [scheduling-1] tdc.Publisher: Publishing value 52 to Routing Key **lazy.pink.rabbit**
INFO --- [ntContainer#0-1] tdc.FirstConsumer: Value: 52
INFO --- [scheduling-1] tdc.Publisher: Publishing value 65 to Routing Key **quick.brown.fox**
INFO --- [scheduling-1] tdc.Publisher: Publishing value 43 to Routing Key **quick.orange.rabbit**
INFO --- [ntContainer#0-1] tdc.FirstConsumer: Value: 43
INFO --- [ntContainer#1-1] tdc.SecondConsumer: Value: 43

Sugestões de leitura

THE DEVELOPER'S CONFERENCE

- Enterprise Integration Patterns
  - Gregor Hohpe; Bobby Woolf
- Documentação RabbitMQ
- Especificação AMQP 0.9.1
- Especificação JMS 2.0
- Spring/Quarkus docs

THE DEVELOPER'S CONFERENCE

PERGUNTAS

THE DEVELOPER'S CONFERENCE