Reactive messaging

JoeDayz | 2026

Evento

```
package com.redhat.training;

public class UserWasCreated {
    public Long id;
    public String username;

    public UserWasCreated() {}

    public UserWasCreated(Long id, String username) {
        this.id = id;
```

Eclipse Microfile Reactive Messaging Spec 2.0

- Quarkus usa SmallRye Reactive Messaging que soporta Kafka, AMQP o MQTT.

- Conceptos:

Messages (org.eclipse.microprofile.reactive.messaging.Message)

Channels (Incoming channels [consumo], outgoing channels [producción])

Connectors (Connector Kafka, RabbitMQ, Pulsar, etc)

```
@Incoming("user-creation-channel") (1)
public CompletionStage<Void> consumeUserWasCreatedEvents(
        Message<UserWasCreated> message
) { (2)
    UserWasCreated event = message.getPayload(); (3)

    if (event.profile == "business") {
        notifyMarketingAboutNewBusiness(event.id);
    }

    return message.ack(); (4)
}
```

```
@Incoming("user-creation-channel") (2)
@ActivateRequestContext (3)
public Uni<Void> processEvent(UserWasCreated event) { (4)
    String assignedAccountRegion = calculateAccountRegion(event.country);

    return session.withTransaction( (5)
        t -> User.<User>findById(event.id)
            .onItem()
            .ifNotNull()
            .invoke( (6)
                entity -> entity.region = assignedAccountRegion
            ).replaceWithVoid() (7)
    ).onTermination().call(() -> session.close()); (8)
}
```

Consumo

De Mensajes

- Payload

```
@Outgoing("incidents-ratio-channel") 1
public Message<Double> notifyRatioOfIncidents() { 2
    Uni<Long> high = Speaker.count("status", "HIGH"); 3
    Uni<Long> total = Speaker.count(); 4

    return Message.of( 5
        high.await().atMost(Duration.ofSeconds(1).doubleValue() 6
        /
        total.await().atMost(Duration.ofSeconds(1).doubleValue()
    );
}
```

```
@Outgoing("incidents-ratio-channel") 1
public Double notifyRatioOfIncidents() { 2
    Uni<Long> high = Speaker.count("status", "HIGH");
    Uni<Long> total = Speaker.count();

    return high.await().atMost(Duration.ofSeconds(1).doubleValue() 3
        /
        total.await().atMost(Duration.ofSeconds(1).doubleValue();
}
```

Producir

Mensajes

Código imperativo

Emitter

```
package com.redhat.training;

import org.eclipse.microprofile.reactive.messaging.Channel;
import org.eclipse.microprofile.reactive.messaging.Emitter;

@ApplicationScoped
public class TemperatureResource {

    @Channel("temperatures-channel") (1)
    Emitter<Double> emitter; (2)

    // Some business logic

    public void sendTemperature(double measurement) {
        emitter.send(measurement); (3)
    }
}
```

```
package com.redhat.training;

import org.eclipse.microprofile.reactive.messaging.Channel;
import org.eclipse.microprofile.reactive.messaging.Emitter;

@ApplicationScoped
public class SomeImperativeLogic {

    @Channel("temperatures-channel") (1)
    Emitter<Double> emitter; (2)

    // Some business logic

    public void sendTemperature(double measurement) {
        emitter.send(Message.of(measurement)); (3)
    }
}
```

Kafka

quarkus-messaging-kafka

```
<dependency>
    <groupId>io.quarkus</groupId>
    <artifactId>quarkus-messaging-kafka</artifactId>
</dependency>
```

Serializers: io.quarkus.kafka.client.serialization.ObjectMapperSerializer

Deserializers: io.quarkus.kafka.client.serialization.ObjectMapperDeserializer

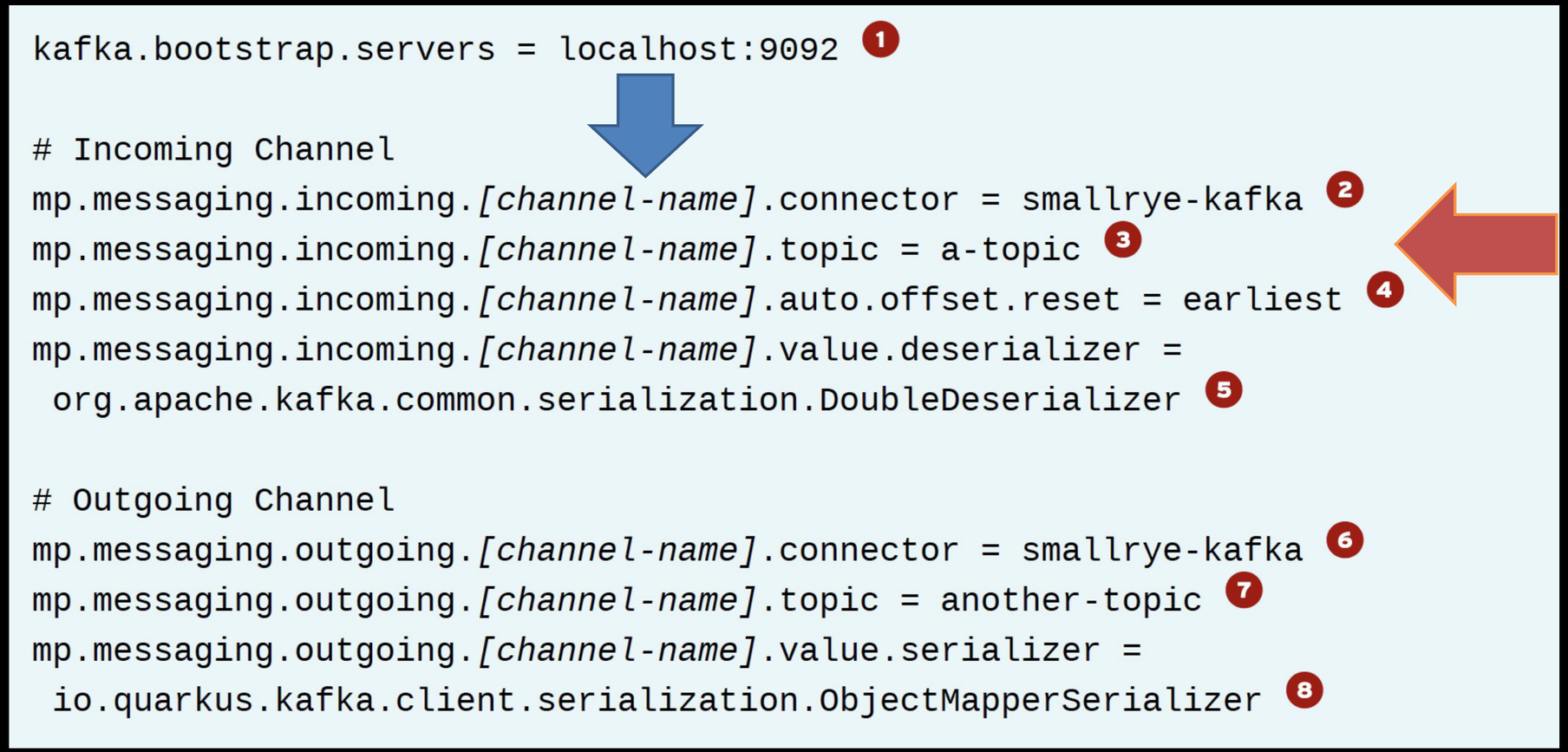
```
package com.redhat.training;

import io.quarkus.kafka.client.serialization.ObjectMapperDeserializer;

public class UserWasCreatedDeserializer extends
 ObjectMapperDeserializer<UserWasCreated> {
    public UserWasCreatedDeserializer() {
        super(UserWasCreated.class);
    }
}
```

Kafka

Configuration



Recursos

https://smallrye.io/smallrye-reactive-messaging/3.21.0/

https://quarkus.io/version/2.13/guides/kafka-reactive-getting-started

https://quarkus.io/version/2.13/guides/amqp

https://kafka.apache.org