gepardec

JMS with Quarkus

gepardec

JMS Component with Quarkus (ActiveMQ Artemis)

1. Configure a ConnectionFactory for Artemis in Quarkus
2. Configure a Camel JmsComponent with the ConnectionFactory
3. Register the JmsComponent in the Quarkus Registry with a name (e.g. `jms`)
4. Use the component name in the for URIs in your routes

```
from("direct:writeToQueue”)
  .to("jms://queue:eggs");
```

component-name

queue-name

gepardec

Configure Artemis ConnectionFactory

- Add Artemis libraries

```
<dependency>
    <groupId>io.quarkiverse.artemis</groupId>
    <artifactId>quarkus-artemis-jms</artifactId>
</dependency>
```

- Configure the connection in `resources/application.properties`

```
quarkus.artemis.mybroker.url=tcp://localhost:61616
quarkus.artemis.mybroker.username=camel
quarkus.artemis.mybroker.password=camel
```

optional identifier

gepardec

Configure Camel JMS Component

- Add component libraries

```
<dependency>
    <groupId>org.apache.camel.quarkus</groupId>
    <artifactId>camel-quarkus-jms</artifactId>
</dependency>
```

- Configure and register the JMS component

component-name

optional identifier

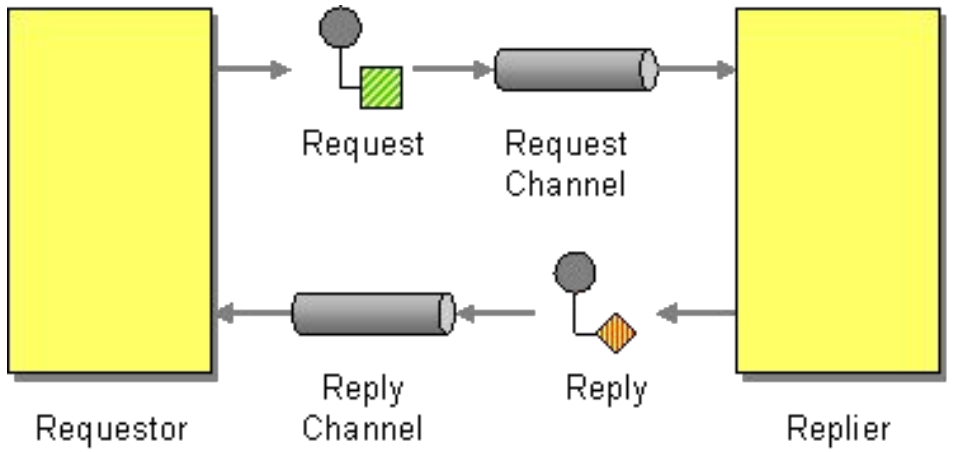
```
import io.smallrye.common.annotation.Identifier;
import jakarta.enterprise.context.ApplicationScoped;
import jakarta.inject.Named;
import jakarta.jms.ConnectionFactory;
@ApplicationScoped
public class MyConfiguration {

    @Named("jms")
    public JmsComponent jms(@Identifier("mybroker") ConnectionFactory connectionFactory) {
        return JmsComponent.jmsComponentClientAcknowledge(connectionFactory);
    }
}
```

gepardec

Using JMS Component

- Usually "Fire and Forget" (ExchangePattern.***InOnly***) is used with JMS
  - set with: .setExchangePattern(ExchangePattern.***InOnly***)
- You may use Request-Reply pattern:




```
.setExchangePattern(ExchangePattern.InOut)
.to("jms:queue:foo?replyTo=bar&receiveTimeout=250")
```

gepardec

Example best_10_jms_artemis