

 免费电子书

学习

apache-camel

Free unaffiliated eBook created from
Stack Overflow contributors.

#apache-
camel

.....	1
1: apache-camel	2
.....	2
Examples.....	2
.....	2
Maven.....	2
.....	2
.....	2
.....	2
2: Pub / SubCamel + Redis	4
.....	4
Examples.....	4
RedisPublisher.....	4
RedisSubscriber.....	4
.....	5
.....	5
ManagedCamel.....	5
3: Apache-CamelSpringDBUnit	8
.....	8
.....	8
.....	8
Examples.....	8
.....	8
Camel.....	9
Camel Integration.....	10
.....	13

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [apache-camel](#)

It is an unofficial and free apache-camel ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official apache-camel.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

1: apache-camel

Apache Camel。 。 。 。

Apache CamelEIP。 JavaWildFlyTomcat。

1. Maven
- 2.
- 3.
4. Plain Old JAR。

Examples

Camel。

Maven

Apache CamelMaven。 MavenCamel。

```
<dependency>
  <groupId>org.apache.camel</groupId>
  <artifactId>camel-core</artifactId>
  <version>2.17.3</version>
</dependency>
```

Apache CamelGradle。 GradleCamel。

```
// https://mvnrepository.com/artifact/org.apache.camel/camel-core
compile group: 'org.apache.camel', name: 'camel-core', version: '2.17.3'
```

Camel 2.15 Apache CamelSpring Boot。 CamelCamel。

```
<dependency>
  <groupId>org.apache.camel</groupId>
  <artifactId>camel-spring-boot</artifactId>
  <version>${camel.version}</version> <!-- use the same version as your Camel core version -
->
</dependency>
```

CamelDSL CamelIntegration。 DSLXMLDSL。

CamelDSLJavaScalaGroovyXML。

- Java DSL

```
from("file:data/in").to("file:data/out");
```

- / Spring DSLXML

```
<route>  
  <from uri="file:data/inbox"/>  
  <to uri="file:data/out"/>  
</route>
```

- Scala DSL

```
from "file:data/inbox" -> "file:data/out"
```

apache-camel <https://riptutorial.com/zh-CN/apache-camel/topic/3511/apache-camel>

2: Pub / SubCamel + Redis

```
producerTemplate.asyncSendBody("direct:myprocedure", messageBody);
```

ManagedCamel“createProducer”producerTemplate。

Examples

RedisPublisher

```
public class RedisPublisher extends RouteBuilder {

    public static final String CAMEL_REDIS_CHANNEL = "CamelRedis.Channel";
    public static final String CAMEL_REDIS_MESSAGE = "CamelRedis.Message";

    @Value("${redis.host}")
    private String redisHost;
    @Value("${redis.port}")
    private int redisPort;
    @Value("${redis.channel.mychannel}")
    private String redisChannel;

    private String producerName;

    @Required
    public void setProducerName(String producerName) {
        this.producerName = producerName;
    }

    @Override
    public void configure() throws Exception {
        from(producerName)
            .log(String.format("Publishing with redis in channel: %s, message body: %s", redisChannel), redisChannel)
            .setHeader(CAMEL_REDIS_CHANNEL, constant(redisChannel))
            .setHeader(CAMEL_REDIS_MESSAGE, body())
            .to(String.format("spring-redis://%s:%s?command=PUBLISH&redisTemplate=#%s",
redisHost, redisPort, ManagedCamel.REDIS_TEMPLATE));
    }
}
```

RedisSubscriber

```
public class RedisSubscriber extends RouteBuilder {

    @Value("${redis.host}")
    private String redisHost;
    @Value("${redis.port}")
    private int redisPort;
    @Value("${redis.channel.mychannel}")
    private String redisChannel;

    private Object bean;
```

```

private String method;

@Required
public void setBean(Object bean) {
    this.bean = bean;
}

@Required
public void setMethod(String method) {
    this.method = method;
}

@Override
public void configure() throws Exception {
    from(String.format("spring-
redis://%s:%s?command=SUBSCRIBE&channels=%s&serializer=#%s", redisHost, redisPort,
redisChannel, ManagedCamel.REDIS_SERIALIZER))
        .log(String.format("Consuming with redis in channel: %s, message body:
${body}", redisChannel))
        .process(exchange -> {
            }).bean(bean, String.format("%s(${body})", method));
}
}

```

beanMethod".

```

<bean id="managedCamel" class="com.pubsub.example.ManagedCamel" >
    <constructor-arg name="routes">
        <list>
            <ref bean="redisSubscriber"/>
        </list>
    </constructor-arg>
</bean>

<bean id="redisSubscriber" class="com.pubSub.example.RedisSubscriber" >
    <property name="bean" ref="myBean"/>
    <property name="method" value="process"/>
</bean>

```

```

<bean id="managedCamel" class="com.pubSub.example.ManagedCamel" >
    <constructor-arg name="routes">
        <list>
            <ref bean="redisPublisher"/>
        </list>
    </constructor-arg>
</bean>

<bean id="redisPublisher" class="com.pubSub.example.RedisPublisher" >
    <property name="producerName" value="direct:myprocedure"/>
</bean>

```

ManagedCamel

```

public class ManagedCamel implements Managed {

    public static final String REDIS_TEMPLATE = "redisTemplate";
    public static final String LISTENER_CONTAINER = "listenerContainer";
}

```

```

public static final String REDIS_SERIALIZER = "redisSerializer";
private DefaultCamelContext camelContext;

private List<RouteBuilder> routes;
@Value("${redis.host}")
private String redisHost;
@Value("${redis.port}")
private int redisPort;
@Value("${redis.password}")
private String redisPassword;

public ManagedCamel(List<RouteBuilder> routes) throws Exception {
    this.routes = routes;
}

@PostConstruct
private void postInit() throws Exception {
    JndiRegistry registry = new JndiRegistry();
    final StringRedisSerializer serializer = new StringRedisSerializer();
    RedisTemplate<String, Object> redisTemplate = getRedisTemplate(serializer);
    registry.bind(REDIS_TEMPLATE, redisTemplate);
    RedisMessageListenerContainer messageListenerContainer = new
RedisMessageListenerContainer();
    registry.bind(LISTENER_CONTAINER, messageListenerContainer);
    registry.bind(REDIS_SERIALIZER, serializer);

    camelContext = new DefaultCamelContext(registry);
    for (RouteBuilder routeBuilder : routes) {
        camelContext.addRoutes(routeBuilder);
    }
    start();
}

private RedisTemplate<String, Object> getRedisTemplate(StringRedisSerializer serializer) {
    RedisTemplate<String, Object> redisTemplate = new RedisTemplate<String, Object>();
    redisTemplate.setConnectionFactory(redisConnectionFactory());
    redisTemplate.setKeySerializer(new StringRedisSerializer());
    redisTemplate.setValueSerializer(serializer);
    redisTemplate.setEnableDefaultSerializer(false);
    redisTemplate.afterPropertiesSet();
    return redisTemplate;
}

private RedisConnectionFactory redisConnectionFactory() {
    final JedisConnectionFactory jedisConnectionFactory = new JedisConnectionFactory();
    jedisConnectionFactory.setHostName(redisHost);
    jedisConnectionFactory.setPort(redisPort);
    jedisConnectionFactory.setPassword(redisPassword);
    jedisConnectionFactory.afterPropertiesSet();
    return jedisConnectionFactory;
}

public void start() throws Exception {
    camelContext.start();
}

public void stop() throws Exception {
    camelContext.stop();
}

public ProducerTemplate createProducer() {

```



```
        return camelContext.createProducerTemplate();  
    }  
}
```

Pub / SubCamel + Redis <https://riptutorial.com/zh-CN/apache-camel/topic/7105/pub---subcamel-plus-redis>

3: Apache-CamelSpringDBUnit

wikiApache Camel。

。

Spring with xmlDBUnit。 。

/	
CamelContext	。
ProducerTemplate	/
AdviceWith	
WeaveById	<i>weaveByToString</i>
MockEndpoint	mockendpoint。 weaveByIdmockEndpoint。

。

- *AdviceWithweaveByIdApache*
- *ProducerTemplate*
- Camel [Enterprise Integration Patterns](#)

。 。

Examples

- **ImportDocumentProcess**
- **ImportDocumentTraitement** ImportDocumentProcess

```
@Component
public class TestExampleRoute extends SpringRouteBuilder {

    public static final String ENDPOINT_EXAMPLE = "direct:testExampleEndpoint";

    @Override
    public void configure() throws Exception {
        from(ENDPOINT_EXAMPLE).routeId("testExample")
            .bean(TestExampleProcessor.class,
"getImportDocumentProcess").id("getImportDocumentProcess")
            .bean(TestExampleProcessor.class,
"createImportDocumentTraitement").id("createImportDocumentTraitement")
            .to("com.pack.camel.routeshowAll=true&multiline=true");
    }
}
```

```
}  
  
}
```

idbean. *id*.

Camel

◦ Java Bean ◦ *Processor*

```
@Component("testExampleProcessor")  
public class TestExampleProcessor {  
  
    private static final Logger LOGGER = LogManager.getLogger(TestExampleProcessor.class);  
  
    @Autowired  
    public ImportDocumentTraitementServiceImpl importDocumentTraitementService;  
  
    @Autowired  
    public ImportDocumentProcessDAOImpl importDocumentProcessDAO;  
  
    @Autowired  
    public ImportDocumentTraitementDAOImpl importDocumentTraitementDAO;  
  
    // ---- Constants to name camel headers and bodies  
    public static final String HEADER_ENTREPRISE = "entreprise";  
  
    public static final String HEADER_UTILISATEUR = "utilisateur";  
  
    public static final String HEADER_IMPORTDOCPROCESS = "importDocumentProcess";  
  
    public void getImportDocumentProcess(@Header(HEADER_ENTREPRISE) Entreprise entreprise,  
Exchange exchange) {  
        LOGGER.info("Entering TestExampleProcessor method : getImportDocumentProcess");  
  
        Utilisateur utilisateur = SessionUtils.getUtilisateur();  
        ImportDocumentProcess importDocumentProcess =  
importDocumentProcessDAO.getImportDocumentProcessByEntreprise(  
            entreprise);  
  
        exchange.getIn().setHeader(HEADER_UTILISATEUR, utilisateur);  
        exchange.getIn().setHeader(HEADER_IMPORTDOCPROCESS, importDocumentProcess);  
    }  
  
    public void createImportDocumentTraitement(@Header(HEADER_ENTREPRISE) Entreprise  
entreprise,  
        @Header(HEADER_UTILISATEUR) Utilisateur utilisateur,  
        @Header(HEADER_IMPORTDOCPROCESS) ImportDocumentProcess importDocumentProcess,  
Exchange exchange) {  
        LOGGER.info("Entering TestExampleProcessor method : createImportDocumentTraitement");  
  
        long nbImportTraitementBefore =  
this.importDocumentTraitementDAO.countNumberOfImportDocumentTraitement();  
        ImportDocumentTraitement importDocumentTraitement =  
this.importDocumentTraitementService.createImportDocumentTraitement(  
            entreprise, utilisateur, importDocumentProcess, "md5_fichier_example_test",  
"fichier_example_test.xml");  
        long nbImportTraitementAfter =
```

```

this.importDocumentTraitementDAO.countNumberOfImportDocumentTraitement ();

        exchange.getIn().setHeader("nbImportTraitementBefore",
Long.valueOf(nbImportTraitementBefore));
        exchange.getIn().setHeader("nbImportTraitementAfter",
Long.valueOf(nbImportTraitementAfter));
        exchange.getIn().setHeader("importDocumentTraitement", importDocumentTraitement);
    }
// Rest of the code contains getters and setters for imported dependencies
}

```

◦ ◦

Camel Integration

camelspring camel◦ maven

```

<dependency>
  <groupId>org.apache.camel</groupId>
  <artifactId>camel-test</artifactId>
  <version>${camel.version}</version>
  <scope>test</scope>
</dependency>
<dependency>
  <groupId>org.apache.camel</groupId>
  <artifactId>camel-test-spring</artifactId>
  <version>${camel.version}</version>
  <scope>test</scope>
</dependency>

```

◦ **DBUnit**◦

Camel Integration Test

```

@RunWith (CamelSpringRunner.class)
@BootstrapWith (CamelTestContextBootstrapper.class)
@ContextConfiguration(locations = { "classpath:/test-beans.xml" })
@DbUnitConfiguration(dataSetLoader = ReplacementDataSetLoader.class)
@TestExecutionListeners({ DependencyInjectionTestExecutionListener.class,
  DirtiesContextTestExecutionListener.class,
    DbUnitTestExecutionListener.class })
@DirtiesContext (classMode = ClassMode.AFTER_EACH_TEST_METHOD)
public abstract class AbstractCamelTI {

}

```

DAO◦ DBUnit◦

@DirtiesContext (classMode = ClassMode.AFTER_EACH_TEST_METHOD) ◦ *camel* ◦ ◦ *remove* ◦ ◦ ◦

```

@DatabaseSetup(value = { "/db_data/dao/common.xml",
"/db_data/dao/importDocumentDAOCommonTest.xml" })
public class TestExampleProcessorTest extends AbstractCamelTI {

```

```

@Autowired
protected CamelContext camelContext;

@EndpointInject(uri = "mock:catchTestEndpoint")
protected MockEndpoint mockEndpoint;

@Produce(uri = TestExampleRoute.ENDPOINT_EXAMPLE)
protected ProducerTemplate template;

@Autowired
ImportDocumentTraitementDAO importDocumentTraitementDAO;

// -- Variables for tests
ImportDocumentProcess importDocumentProcess;

@Override
@Before
public void setUp() throws Exception {
    super.setUp();

    importDocumentProcess = new ImportDocumentProcess();
    //specific implementation of your choice
}
}

```

mockEndpoint *ImportDocumentProcess*

```

@Test
public void processCorrectlyObtained_getImportDocumentProcess() throws Exception {
    camelContext.getRouteDefinitions().get(0).adviceWith(camelContext, new
AdviceWithRouteBuilder() {

        @Override
        public void configure() throws Exception {
            weaveById("getImportDocumentProcess").after().to(mockEndpoint);
        }
    });

    // -- Launching the route
    camelContext.start();
    template.sendBodyAndHeader(null, "entreprise", company);

    mockEndpoint.expectedMessageCount(1);
    mockEndpoint.expectedHeaderReceived(TestExampleProcessor.HEADER_UTILISATEUR, null);
    mockEndpoint.expectedHeaderReceived(TestExampleProcessor.HEADER_IMPORTDOCPROCESS,
importDocumentProcess);
    mockEndpoint.assertIsSatisfied();

    camelContext.stop();
}

```

```

@Test
public void traitementCorrectlyCreated_createImportDocumentTraitement() throws Exception {
    camelContext.getRouteDefinitions().get(0).adviceWith(camelContext, new
AdviceWithRouteBuilder() {

        @Override
        public void configure() throws Exception {
            weaveById("createImportDocumentTraitement").after().to(mockEndpoint);
        }
    });
}

```

```

    }
});

// -- Launching the route
camelContext.start();

Exchange exchange = new DefaultExchange(camelContext);
exchange.getIn().setHeader(TestExampleProcessor.HEADER_ENTREPRISE, company);
exchange.getIn().setHeader(TestExampleProcessor.HEADER_UTILISATEUR, null); // No user in
this case
exchange.getIn().setHeader(TestExampleProcessor.HEADER_IMPORTDOCPROCESS,
importDocumentProcess);

long numberOfTraitementBefore =
this.importDocumentTraitementDAO.countNumberOfImportDocumentTraitement();

template.send(exchange);

mockEndpoint.expectedMessageCount(1);
mockEndpoint.assertIsSatisfied();

camelContext.stop();

long numberOfTraitementAfter =
this.importDocumentTraitementDAO.countNumberOfImportDocumentTraitement();
assertEquals(numberOfTraitementBefore + 1L, numberOfTraitementAfter);
}

```

◦ mockEndpoint ◦ ◦

adviceWith

```

camelContext.getRouteDefinitions().get(0).adviceWith(camelContext, new
AdviceWithRouteBuilder() { [...] });

```

ID

```

camelContext.getRouteDefinition("routeId").adviceWith(camelContext, new
AdviceWithRouteBuilder() { [...] });

```

Apache-CamelSpringDBUnit <https://riptutorial.com/zh-CN/apache-camel/topic/10630/apache-camelspring-dbunit->

S. No		Contributors
1	apache-camel	Community , Michael Hoffman , Namphibian
2	Pub / SubCamel + Redis	Lior
3	Apache-Camel SpringDBUnit	DamienB , Flanfl , matthieusb