

Writing Custom Camel Routes

A number of production-ready Camel Routes and projects in [fcrepo-camel-toolbox](#). Below are some additional Camel Routes to get you started.

LDAPPath Transformations

Using the [fcrepo-ldpath](#) Camel component, you can generate a JSON representation of a [container](#) and send it to a low-latency, highly available document store, such as [Riak](#). The following route determines if a container has been removed or simply added/updated. It then routes the message appropriately to a load-balancer sitting in front of the Riak HTTP endpoint.

Camel route to populate a Riak store, using the Scala DSL

```
1 val riakKeyProcessor = (exchange : Exchange) => {
2     exchange.getIn.setHeader(
3         Exchange.HTTP_PATH,
4         "/buckets/fcrepo/keys/" + URLEncoder.encode(exchange.getIn.getHeader( "org.fcrepo.jms.identifier" ,
5         classOf[String]))
6     )
7 }
8
9 "activemq:topic:fedora" ==> {
10     choice() {
11         when( _ .in( "org.fcrepo.jms.eventType" ) == "http://fedora.info/definitions/v4
12         /repository#NODE_REMOVED" ) {
13             setHeader(Exchange.HTTP_METHOD, constant( "DELETE" ))
14             process(riakKeyProcessor)
15             to( "http4:localhost:8098" )
16         }
17         otherwise() {
18             to( "fcrepo:localhost:8080/fedora/rest" )
19             filter(xpathFilter) {
20                 to( "fcrepo:localhost:8080/fedora/rest?accept=application/json&transform=mytransform" )
21                 setHeader(Exchange.HTTP_METHOD, constant( "PUT" ))
22                 process(riakKeyProcessor)
23                 to( "http4:localhost:8098" )
24             }
25         }
26     }
27 }
```

External Triplestore

Some additional processing must be done to transform an `application/n-triples` response into a valid `application/sparql-update` payload before sending to an external triplestore such as Fuseki or Sesame. The `fcrepo` component contains some processors in `org.fcrepo.camel.processor` to handle this case. The examples below assume that messages have already been routed based on `eventType` (see below) and passed to the appropriate queue.

Populate an external triplestore

```
1 from( "direct:delete" )
2     .process( new SparqlDescribeProcessor() )
3     .to( "http4:localhost:3030/db/query" )
4     .process( new SparqlDeleteProcessor() )
5     .to( "http4:localhost:3030/db/update" );
6
7 from( "direct:new" )
8     .to( "fcrepo:localhost:8080/rest" )
9     .process( new SparqlInsertProcessor() )
10    .to( "http4:localhost:3030/db/update" );
11
12 from( "direct:update" )
13    .to( "fcrepo:localhost:8080/rest" )
14    .process( new SparqlUpdateProcessor() )
15    .to( "http4:localhost:3030/db/update" );
```

Event-based Routing

It is often helpful to route messages to different queues based on the `eventType` value. This example splits messages on `eventType` values and routes the messages to appropriate queues. Following this example, it would be prudent to aggregate the messages based on `org.fcrepo.jms.identifier` value after retrieving the messages from the downstream queues.

Content-based Routing

```
1 <route id = "fcrepo-event-splitter" >
2   <description >
3     Retrieve messages from the fedora topic. Event types are comma-delimited, so split them into
4   separate messages before routing them.
5   </description >
6   <from uri = "activemq:topic:fedora" />
7   <setBody >
8     <simple>${header.org.fcrepo.jms.eventType}</simple >
9   </setBody >
10  <split >
11    <tokenize token = "," />
12    <setHeader headerName = "org.fcrepo.jms.eventType" >
13      <simple>${body}</simple >
14    </setHeader >
15    <setBody >
16      <simple>null</simple >
17    </setBody >
18    <to uri = "seda:fcrepo-event-router" />
19  </split >
20</route >
21
22<route id = "fcrepo-event-router" >
23  <description >
24    Route messages based on the eventType.
25  </description >
26  <from uri = "seda:fcrepo-event-router" />
27  <choice >
28    <when >
29      <simple>${header.org.fcrepo.jms.eventType} == "http://fedora.info/definitions/v4
30/repository#NODE_REMOVED"</simple >
31      <to uri = "activemq:queue:fcrepo.delete" />
32    </when >
33    <when >
34      <simple>${header.org.fcrepo.jms.eventType} == "http://fedora.info/definitions/v4
35/repository#NODE_ADDED"</simple >
36      <to uri = "activemq:queue:fcrepo.add" />
37    </when >
38    <when >
39      <simple>${header.org.fcrepo.jms.eventType} == "http://fedora.info/definitions/v4
40/repository#PROPERTY_ADDED"</simple >
41      <to uri = "activemq:queue:fcrepo.update" />
42    </when >
43    <when >
44      <simple>${header.org.fcrepo.jms.eventType} == "http://fedora.info/definitions/v4
45/repository#PROPERTY_CHANGED"</simple >
46      <to uri = "activemq:queue:fcrepo.update" />
47    </when >
48    <when >
49      <simple>${header.org.fcrepo.jms.eventType} == "http://fedora.info/definitions/v4
50/repository#PROPERTY_REMOVED"</simple >
51      <to uri = "activemq:queue:fcrepo.update" />
52    </when >
53    <otherwise >
54      <log message = "No router for ${header.org.fcrepo.jms.eventType}" />
55    </otherwise >
56  </choice >
57</route >
```