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.

LDPath 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

```
val riakKeyProcessor = (exchange : Exchange) => {
    exchange.getIn.setHeader(
        Exchange.HTTP_PATH,
        "/buckets/fcrepo/keys/" + URLEncoder.encode(exchange.getIn.getHeader("org.fcrepo.jms.identifier",
classOf[String]))
    )
}

"activemq:topic:fedora" => {
    choice() {
        when(_.in("org.fcrepo.jms.eventType") == "http://fedora.info/definitions/v4
/repository#NODE_REMOVED") {
            setHeader(Exchange.HTTP_METHOD, constant("DELETE"))
            process(riakKeyProcessor)
            to("http4:localhost:8098")
        }
        otherwise() {
            to("fcrepo:localhost:8080/fedora/rest")
            filter(xpathFilter) {
                to("fcrepo:localhost:8080/fedora/rest?accept=application/json&transform=mytransform")
                setHeader(Exchange.HTTP_METHOD, constant("PUT"))
                process(riakKeyProcessor)
                to("http4:localhost:8098")
            }
        }
    }
}
```

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

```
from("direct:delete")
   .process(new SparqlDescribeProcessor())
   .to("http4:localhost:3030/db/query")
   .process(new SparqlDeleteProcessor())
   .to("http4:localhost:3030/db/update");

from("direct:new")
  .to("fcrepo:localhost:8080/rest")
  .process(new SparqlInsertProcessor())
  .to("http4:localhost:3030/db/update");

from("direct:update")
   .to("fcrepo:localhost:8080/rest")
   .process(new SparqlUpdateProcessor())
   .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

```
< route id = "fcrepo-event-splitter" >
    < description >
        Retrieve messages from the fedora topic. Event types are comma-delimited, so split them into
separate messages before routing them.
    </ description >
    < from uri = "activemq:topic:fedora" />
    < setBody >
        < simple >${header.org.fcrepo.jms.eventType}</ simple >
    </ setBody >
    < split >
        < tokenize token = "," />
        < setHeader headerName = "org.fcrepo.jms.eventType" >
            < simple >${body}</ simple >
        </ setHeader >
        < setBody >
            < simple >null</ simple >
        </ setBody >
        < to uri = "seda:fcrepo-event-router" />
    </ split >
</ route >

< route id = "fcrepo-event-router" >
    < description >
        Route messages based on the eventType.
    </ description >
    < from uri = "seda:fcrepo-event-router" />
    < choice >
        < when >
            < simple >${header.org.fcrepo.jms.eventType} == "http://fedora.info/definitions/v4
/repository#NODE_REMOVED"</ simple >
            < to uri = "activemq:queue:fcrepo.delete" />
        </ when >
        < when >
            < simple >${header.org.fcrepo.jms.eventType} == "http://fedora.info/definitions/v4
/repository#NODE_ADDED"</ simple >
            < to uri = "activemq:queue:fcrepo.add" />
        </ when >
        < when >
            < simple >${header.org.fcrepo.jms.eventType} == "http://fedora.info/definitions/v4
/repository#PROPERTY_ADDED"</ simple >
            < to uri = "activemq:queue:fcrepo.update" />
        </ when >
        < when >
            < simple >${header.org.fcrepo.jms.eventType} == "http://fedora.info/definitions/v4
/repository#PROPERTY_CHANGED"</ simple >
            < to uri = "activemq:queue:fcrepo.update" />
        </ when >
        < when >
            < simple >${header.org.fcrepo.jms.eventType} == "http://fedora.info/definitions/v4
/repository#PROPERTY_REMOVED"</ simple >
            < to uri = "activemq:queue:fcrepo.update" />
        </ when >
        < otherwise >
            < log message = "No router for ${header.org.fcrepo.jms.eventType}" />
        </ otherwise >
    </ choice >
</ route >
```