Enterprise Integration Patterns using Apache Camel

Shailendra Kumar Singh
SME-Red Hat
1. What is Apache Camel?
2. What is EIP?
3. Architecture
4. EIP with Apache Camel.
5. Example.
6. Demo
What is Apache Camel?

Apache Camel is a powerful Open Source Integration Framework based on Enterprise Integration Patterns

http://camel.apache.org/
Why do we need integration?
○ Critical for your business to integrate

Why Integration Framework?
○ Framework do the heavy lifting
○ You can focus on business problem
What is Enterprise Integration Patterns?

- Technology-independent **design guidance**
- Describe and develop robust integration solutions.
- **65** Integration patterns

https://www.enterpriseintegrationpatterns.com/
Enterprise Integration Patterns

<table>
<thead>
<tr>
<th>Content Based Router</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message Filter</td>
</tr>
<tr>
<td>Dynamic Router</td>
</tr>
<tr>
<td>Recipient List</td>
</tr>
<tr>
<td>Splitter</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Aggregator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resequencer</td>
</tr>
<tr>
<td>Content Enricher</td>
</tr>
<tr>
<td>Content Filter</td>
</tr>
<tr>
<td>Pipes and Filters</td>
</tr>
</tbody>
</table>
Content Based Router

New Order → Router → Widget Inventory → Gadget Inventory
Content Based Router

from newOrder

ENDPOINT newOrder = endpoint("activemq:queue:newOrder");

Predicate isWidget = xpath("/order/product = 'widget'");

ENDPOINT widget = endpoint("activemq:queue:widget");

ENDPOINT gadget = endpoint("activemq:queue:gadget");

from newOrder.choice().when(isWidget).to(widget).otherwise().to(gadget);
from newOrder
choice

Content Based Router
from newOrder
choice
when isWidget  to widget
from newOrder
choice
when isWidget to widget
otherwise to gadget
from(newOrder)
.choice()
.when(isWidget).to(widget)
.otherwise().to(gadget);
Endpoint newOrder = endpoint("activemq:queue:newOrder");

from(newOrder)
.choice()
.when(isWidget).to(widget)
.otherwise().to(gadget);
Endpoint `newOrder` = endpoint("activemq:queue:newOrder");
Predicate `isWidget` = xpath("/order/product = 'widget'");

from(newOrder)
.choice()
.when(isWidget).to(widget)
.otherwise().to(gadget);
Endpoint `newOrder` = endpoint("activemq:queue:newOrder");
Predicate `isWidget` = xpath("/order/product = 'widget'");
Endpoint `widget` = endpoint("activemq:queue:widget");
Endpoint `gadget` = endpoint("activemq:queue:gadget");

from(`newOrder`)
.choice()
.when(`isWidget`).to(`widget`)
.otherwise().to(`gadget`);
public void configure() throws Exception {
    Endpoint newOrder = endpoint("activemq:queue:newOrder");
    Predicate isWidget = xpath("/order/product = 'widget'");
    Endpoint widget = endpoint("activemq:queue:widget");
    Endpoint gadget = endpoint("activemq:queue:gadget");

    from(newOrder)
        .choice()
        .when(isWidget).to(widget)
        .otherwise().to(gadget);
}
import org.apache.camel.Endpoint;
import org.apache.camel.Predicate;
import org.apache.camel.builder.RouteBuilder;

public class MyRoute extends RouteBuilder {
    public void configure() throws Exception {
        Endpoint newOrder = endpoint("activemq:queue:newOrder");
        Predicate isWidget = xpath("/order/product = 'widget'");
        Endpoint widget = endpoint("activemq:queue:widget");
        Endpoint gadget = endpoint("activemq:queue:gadget");

        from(newOrder)
            .choice()
            .when(isWidget).to(widget)
            .otherwise().to(gadget);
    }
}
import org.apache.camel.builder.RouteBuilder;

public class MyRoute extends RouteBuilder {

    public void configure() throws Exception {

        from("activemq:queue:newOrder")
            .choice()
            .when(xpath("/order/product = 'widget'"))
            .to("activemq:queue:widget")
            .otherwise()
            .to("activemq:queue:gadget");
    }
}
<route>
  <from uri="activemq:queue:newOrder"/>
  <choice>
    <when>
      <xpath>/order/product = 'widget'</xpath>
      <to uri="activemq:queue:widget"/>
    </when>
    <otherwise>
      <to uri="activemq:queue:gadget"/>
    </otherwise>
  </choice>
</route>
<route>
   <from uri="file:inbox/orders"/>
   <choice>
      <when>
         <xpath>/order/product = 'widget'</xpath>
         <to uri="activemq:queue:widget"/>
      </when>
      <otherwise>
         <to uri="activemq:queue:gadget"/>
      </otherwise>
   </choice>
</route>
Architecture

Camel
Integration Engine And Router

Camel Endpoints
* Camel can send messages to them
* Or receive Messages from them

Camel Components
* Provide a uniform Endpoint interface
* Act as connectors to all other systems

Camel Processors
* Are used to wire Endpoints together
* Routing
* Transformation
* Mediation
* Interception
* Enrichment
* Validation
* Tracking
* Logging

JMS Component
JMS API
JMS Provider
ActiveMQ | IBM | Tibco | Sonic...

HTTP Component
Servlet API
HTTP Client

File Component
File System
Local File System
<table>
<thead>
<tr>
<th>ahc</th>
<th>bindy</th>
<th>coap</th>
<th>docker</th>
</tr>
</thead>
<tbody>
<tr>
<td>ahc-ws</td>
<td>blueprint</td>
<td>cometd</td>
<td>dozer</td>
</tr>
<tr>
<td>apns</td>
<td>boon</td>
<td>context</td>
<td>dropbox</td>
</tr>
<tr>
<td>atmosphere</td>
<td>box</td>
<td>couchdb</td>
<td>eclipse</td>
</tr>
<tr>
<td>atom</td>
<td>cache</td>
<td>crypto</td>
<td>ejb</td>
</tr>
<tr>
<td>aws</td>
<td>cassandraql</td>
<td>csv</td>
<td>elasticsearch</td>
</tr>
<tr>
<td>bam</td>
<td>castor</td>
<td>cfx</td>
<td>elsqql</td>
</tr>
<tr>
<td>bean-validator</td>
<td>cdi</td>
<td>cxf-transport</td>
<td>eventadmin</td>
</tr>
<tr>
<td>beanio</td>
<td>chunk</td>
<td>disruptor</td>
<td>exec</td>
</tr>
<tr>
<td>beanstalk</td>
<td>cmis</td>
<td>dns</td>
<td>facebook</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------</td>
<td>----------------------</td>
<td></td>
</tr>
<tr>
<td>Standalone</td>
<td>Web Application</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Camel Spring XML</td>
<td>JEE Application</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Camel Spring Boot</td>
<td>Apache Karaf (OSGi)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Camel CDI</td>
<td>Wildfly (wildfly-camel)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Camel Guice</td>
<td>vert.x (vertx-camel)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
EIP with Camel
Recipient List

How to route messages based on a **static** or **dynamic list** of destinations

```java
from("jms:queue:inbox")
    .multicast().to("file://backup","seda:inbox");

from("seda:confirmMails").beanRef(processMails)
    .recipientList("destinations")
```
Camel has support for **Splitter** using the **split** node

```java
from("file://inbox")
   .split(body().tokenize("\n"))
   .to("seda:orderLines");
```
Camel has support for **Aggregator** using the aggregator node

```java
from("jms:topic:stock:quote")
.aggregate()
.xpath("/quote/@symbol")
=batchTimeout(5 * 60 *1000)
.to("seda:quotes");
```
Pipes and Filters EIP

```
from("jms:queue:order:in")
  .pipeline()
  .to("bean:decrypt")
  .to("bean:authenticate");
  .to("bean:deDup");

from("jms:queue:order:in")
  .to("bean:decrypt")
  .to("bean:authenticate");
  .to("bean:deDup");
```

`pipeline()` is default mode in camel.
Camel supports the message translator using the processor, bean or transform nodes.
Camel has support for **Message Filter** using the **filter node**.

```java
from("jms:queue:inbox")
    .filter(header("test").isNotEqualTo("true"))
    .to("jms:queue:order");
```
Camel has support for **Resequencer** using the `resequence` node.

```camel
from("jms:topic:stock:quote")
.resequence().xpath("/quote/@symbol")
.timeout(60 * 1000)
.to("seda:quotes");
```
from("jms:queue:order")
    .wireTap("direct:order")
    .to("bean:processOrder")
DEMO
<orders>
  <order id="1">
    <customer id="A0001">
      <message>A0001-Message1</message>
    </customer>
  </order>
  <order id="2">
    <customer id="B0001">
      <message>B0001-Message1</message>
    </customer>
  </order>
  <order id="3">
    <customer id="C0001">
      <message>C0001-Message1</message>
    </customer>
  </order>
  <order id="4">
    <customer id="A0001">
      <message>A0001-Message2</message>
    </customer>
  </order>
</orders>
References
❖ http://camel.apache.org/
❖ https://github.com/apache/camel
❖ https://github.com/apache/camel/tree/master/examples
❖ https://www.enterpriseintegrationpatterns.com/
❖ https://github.com/camelinaction/camelinaction
❖ https://dzone.com/articles/enterprise-integration