

# Message Dispatcher

## Message Dispatcher

Camel supports the [Message Dispatcher](#) from the [EIP patterns](#) using various approaches.

[blocked URL](#)

You can use a component like [JMS](#) with selectors to implement a [Selective Consumer](#) as the Message Dispatcher implementation. Or you can use an [Endpoint](#) as the Message Dispatcher itself and then use a [Content Based Router](#) as the Message Dispatcher.

## Example

The following example demonstrates [Message Dispatcher](#) pattern using the [Competing Consumers](#) functionality of the [JMS](#) component to offload messages to a [Content Based Router](#) and custom [Processors](#) registered in the Camel [Registry](#) running in separate threads from originating consumer.

### Using the [Fluent Builders](#)

```
from( "jms:queue:foo?concurrentConsumers=5" )
    .threads(5)
    .choice()
        .when( header( "type" ).isEqualTo( "A" ) )
            .processRef( "messageDispatchProcessorA" )
        .when( header( "type" ).isEqualTo( "B" ) )
            .processRef( "messageDispatchProcessorB" )
        .when( header( "type" ).isEqualTo( "C" ) )
            .processRef( "messageDispatchProcessorC" )
        .otherwise()
            .to( "jms:queue:invalidMessageType" );
```

### Using the [Spring XML Extensions](#)

```
<route>
  <from uri="jms:queue:foo?concurrentConsumers=5"/>
  <threads poolSize="5">
    <choice>
      <when>
        <simple>${in.header.type} == 'A'</simple>
        <to ref="messageDispatchProcessorA"/>
      </when>
      <when>
        <simple>${in.header.type} == 'B'</simple>
        <to ref="messageDispatchProcessorB"/>
      </when>
      <when>
        <simple>${in.header.type} == 'C'</simple>
        <to ref="messageDispatchProcessorC"/>
      </when>
      <otherwise>
        <to uri="jms:queue:invalidMessageType"/>
      </otherwise>
    </choice>
  </threads>
</route>
```

## See Also

- [JMS](#)
- [Selective Consumer](#)
- [Content Based Router](#)
- [Endpoint](#)

## Using This Pattern

If you would like to use this EIP Pattern then please read the [Getting Started](#), you may also find the [Architecture](#) useful particularly the description of [Endpoint](#) and [URIs](#). Then you could try out some of the [Examples](#) first before trying this pattern out.