10

Desafio: tratamento com onException

O `errorHandler` permite definir um tratamento de error bastante detalhado, no entanto não podemos tratar exceções específicas. Imagine que queremos um tratamento específico para uma `IOException` e outro para uma `ParsingException`. Usando apenas o `errorHandler` é impossível.

É claro que o Camel possui uma alternativa para esse caso também. O método `onException` permite um tratamento parecido com o `errorHandler`, mas para uma exceção concreta:

```
//deve ser configurado antes de qualquer rota
onException(Exception.class).
    handled(true).
        maximumRedeliveries(3).
            redeliveryDelay(4000).
        onRedelivery(new Processor() {

            @Override
            public void process(Exchange exchange) throws Exception {
                    int counter = (int) exchange.getIn().getHeader(Exchange.REDELIVERY_COUNTER);
                    int max = (int) exchange.getIn().getHeader(Exchange.REDELIVERY_MAX_COUNTER);
                    System.out.println("Redelivery - " + counter + "/" + max );;
            }
    });
```

Repare que temos praticamente a mesma configuração do `errorHandler`. O método `onException` recebe o tipo de exceção que trataremos. Como usamos `Exception.class`, conseguimos tratar todas as exceções, mas poderia ser mais específico:

```
onException(SAXParseException.class).
```

Além disso, temos um método `handled(true)`. Faça o teste, use `onException` como foi apresentado, mas uma `SaxParseException`. Teste uma vez com e depois, sem `handled(true)`. Você percebe a diferença?