

APACHE CAMEL WORKSHOP



JFuture 2018
Claus Ibsen
(Original by Nicola Ferraro)

WORKSHOP 1ST TIME GIVEN IN SUMMER 2018



WELCOME, WHO AM I

Claus Ibsen

Software Engineer (Red Hat)

10 years Apache Camel



 @davsclaus

 davsclaus

www.davsclaus.com

WHO AM NICOLA

Nicola Ferraro

Software Engineer (Red Hat)

Less years Apache Camel :)

@ni_ferraro 

nicolaferraro 

www.nicolaferraro.me



AGENDA

- **Intro to Apache Camel** ←
- Overview of the Workshop
- Coding: Part 1
- Circuit Breakers and Saga
- Coding: Part 2
- Openshift (Optional)
- Summary
- Q/A

WHAT IS APACHE CAMEL ?



APACHE®

Camel

System Integration



Figure 1.1 Camel is the glue between disparate systems.

Integration Framework





APACHE®
Camel

PATTERN BASED INTEGRATION

Apache Camel, a powerful pattern-based integration engine with a comprehensive set of connectors and data formats to tackle any integration problem.



ENTERPRISE INTEGRATION PATTERNS

Build integrations using enterprise best practices.



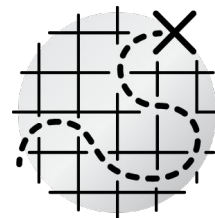
200+ COMPONENTS

Batch, messaging, web services, cloud, APIs, and more ...



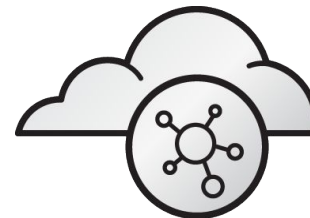
BUILT-IN DATA TRANSFORMATION

JSON, XML, HL7, YAML, SOAP, Java, CSV, and more ...



INTUITIVE ROUTING

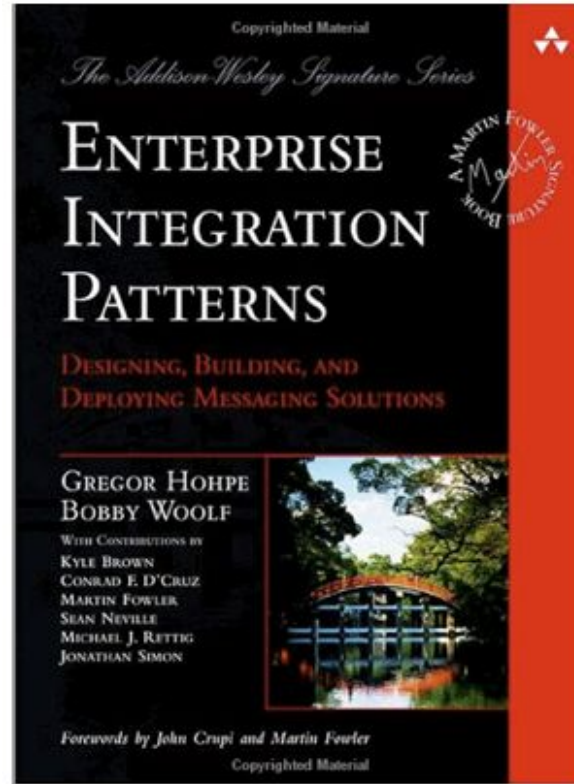
Develop integrations quickly in Java or XML.



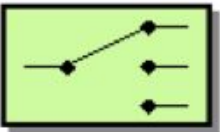
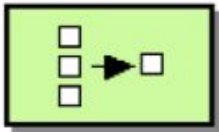

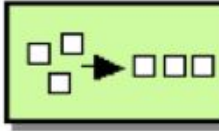
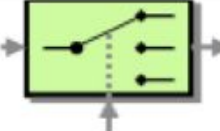
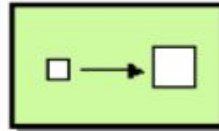
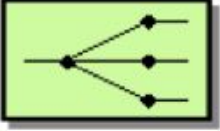
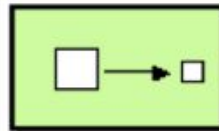
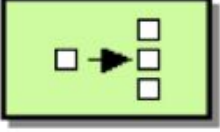
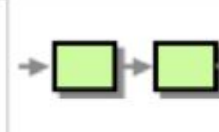
NATIVE REST SUPPORT

Create, connect, and compose APIs with ease.

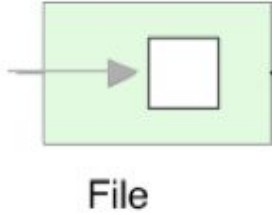
Enterprise Integration Patterns



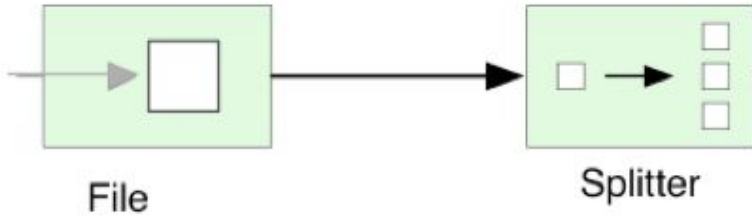
Enterprise Integration Patterns

| | | | |
|--|----------------------|--|-------------------|
|  | Content Based Router |  | Aggregator |
|  | Message Filter |  | Resequencer |
|  | Dynamic Router |  | Content Enricher |
|  | Recipient List |  | Content Filter |
|  | Splitter |  | Pipes and Filters |

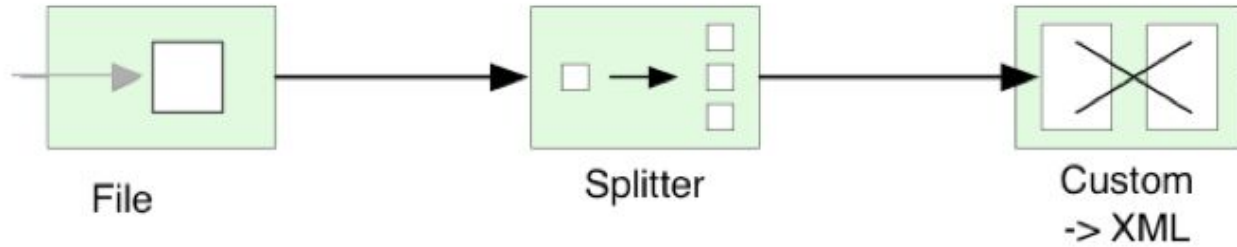
Camel Routes with Splitter



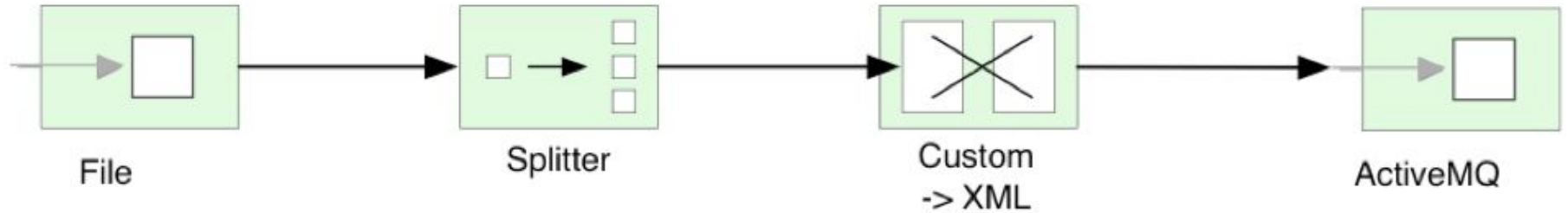
Camel Routes with Splitter



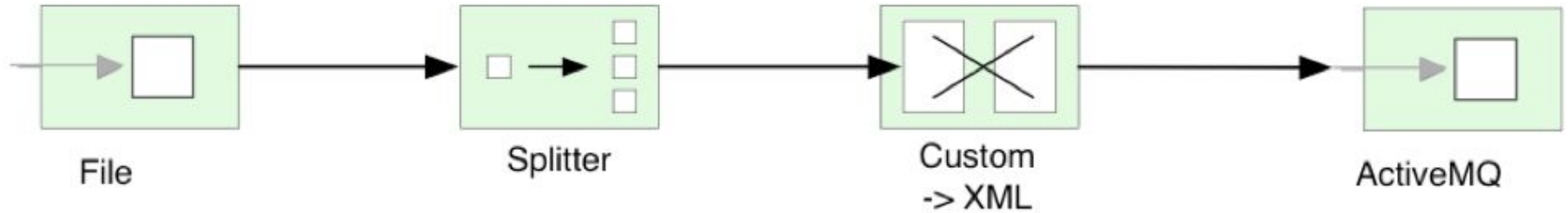
Camel Routes with Splitter



Camel Routes with Splitter

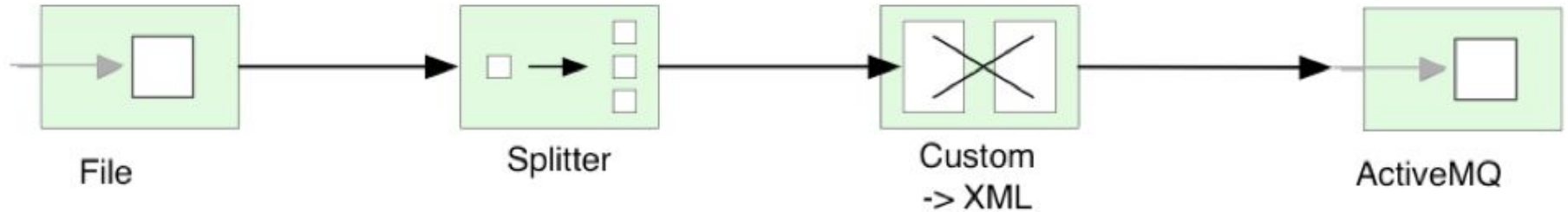


Camel Routes with Splitter



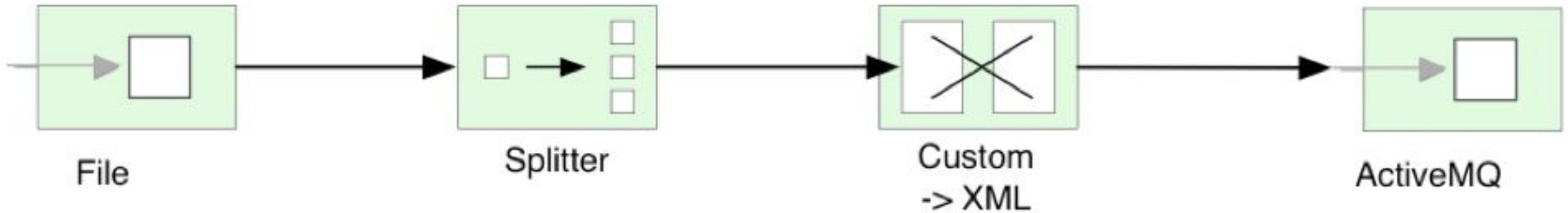
```
from("file:inbox")
```


Camel Routes with Splitter



```
from("file:inbox")  
    .split(body().tokenize("\n"))
```

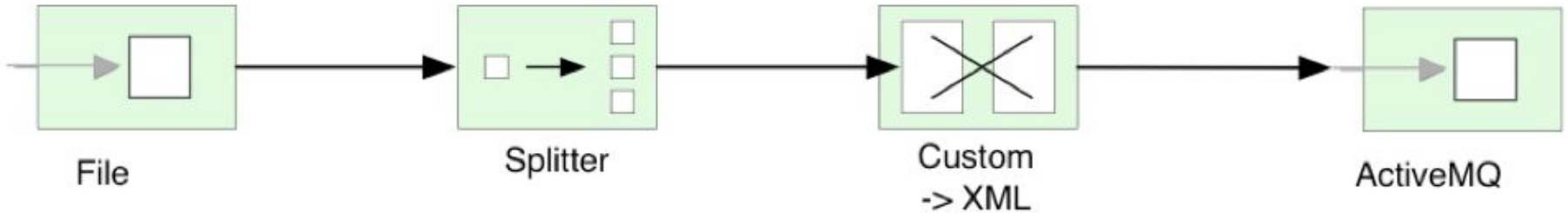
Camel Routes with Splitter



```
from("file:inbox")  
    .split(body().tokenize("\n"))  
    .marshal(customToXml)
```

Custom data
transformation

Camel Routes with Splitter



```
from("file:inbox")
    .split(body().tokenize("\n"))
    .marshal(customToXml)
    .to("activemq:line");
```

Custom data
transformation

Camel Routes with Splitter

```
from("file:inbox")  
    .split(body().tokenize("\n"))  
    .marshal(customToXml)  
    .to("activemq:line");
```

Camel Route in Java DSL

```
public void configure() throws Exception {  
  
    from("file:inbox")  
        .split(body().tokenize("\n"))  
        .marshal(customToXml)  
        .to("activemq:line");  
}
```

Camel Route in Java DSL

```
public class MyRoute extends RouteBuilder {  
    public void configure() throws Exception {  
        from("file:inbox")  
            .split(body().tokenize("\n"))  
            .marshal(customToXml)  
            .to("activemq:line");  
    }  
}
```

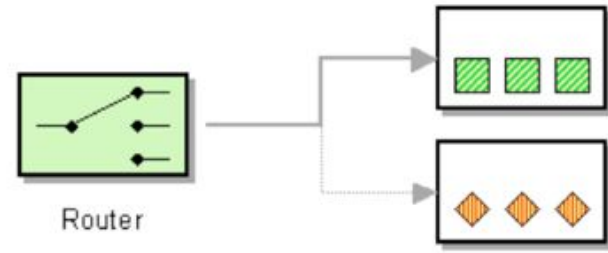
```
import org.apache.camel.builder.RouteBuilder;

public class MyRoute extends RouteBuilder {

    public void configure() throws Exception {

        from("file:inbox")
            .split(body().tokenize("\n"))
            .marshal(customToXml)
            .to("activemq:line");
    }
}
```

Camel Routes



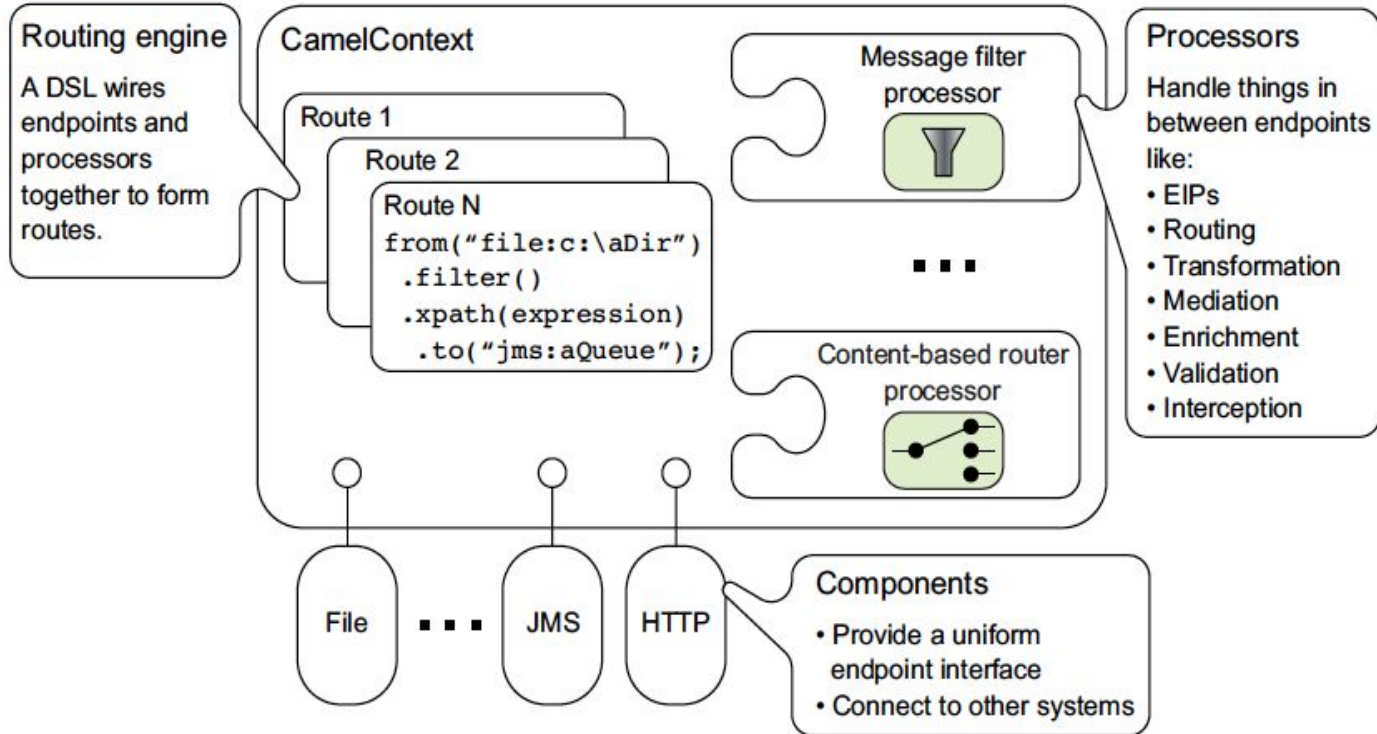
```
from("file:data/inbox")  
    .to("jms:queue:order");
```

Java DSL

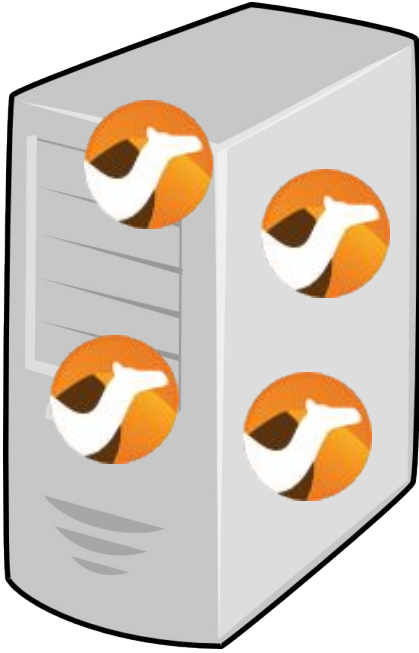
XML DSL

```
<route>  
    <from uri="file:data/inbox"/>  
    <to uri="jms:queue:order"/>  
</route>
```


Camel Architecture



Camel runs everywhere



Application
Servers

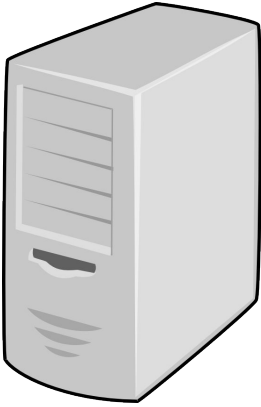


Standalone
Runtimes

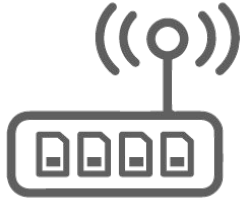


Linux
Containers

Camel connects everything



**Enterprise
Systems**



IoT

- File
- FTP
- JMS
- AMQP
- JDBC
- SQL
- TCP/UDP
- Mail
- HDFS
- JPA
- MongoDB
- Kafka
- ...

- CoAP
- MQTT
- PubNub

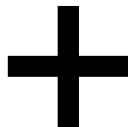
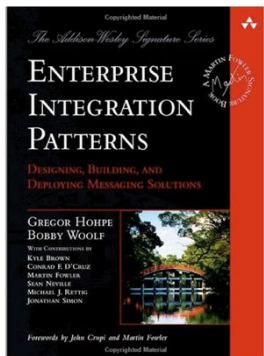


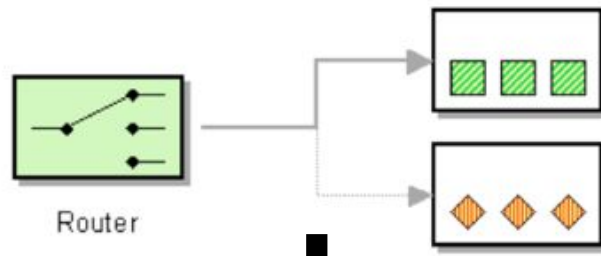
- AWS
 - S3
 - SQS
 - Kinesis
 - ...
- Google
 - BigQuery
 - PubSub
- Azure
 - Blob
 - Queue



- Box
- Dropbox
- Facebook
- LinkedIn
- Salesforce
- SAP
- ServiceNow

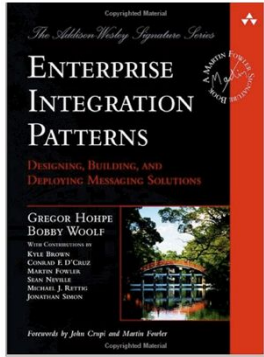


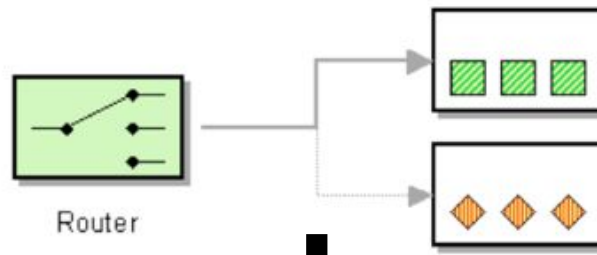




+

+



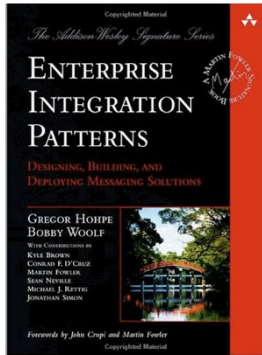
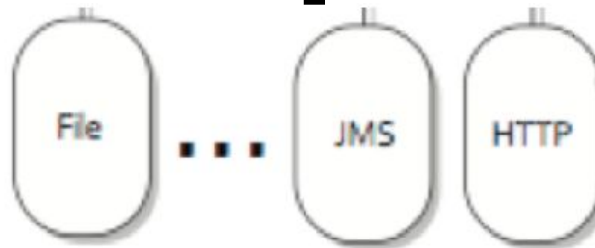


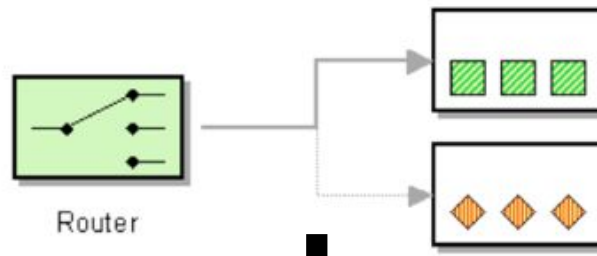
+

+



+



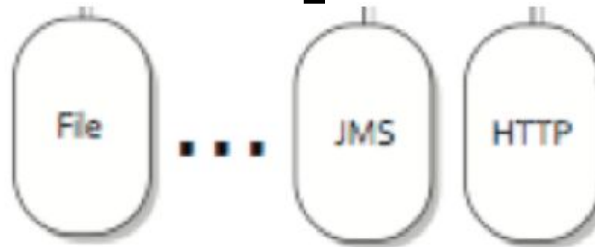


+

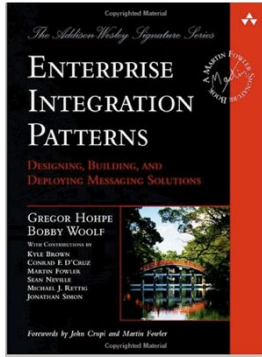
+



+



=



AGENDA

- Intro to Apache Camel
- **Overview of the Workshop** ←
- Coding: Part 1
- Circuit Breakers and Saga
- Coding: Part 2
- Openshift (Optional)
- Summary
- Q/A

OVERVIEW OF THE WORKSHOP



Apache Camel Workshop

Plush Toys Shop



Itchy & Scratchy

\$1.00

10 items in stock

- 0 +



The Joker

\$1.00

10 items in stock

- 0 +



Mr. Camel

\$1.00

10 items in stock

- 0 +



eDonkey

\$1.00

10 items in stock

- 0 +



Lupin

\$1.00

10 items in stock

- 0 +



Dali

\$1.00

10 items in stock

- 0 +



Darth Vader

\$1.00

10 items in stock

- 0 +



Secco

\$1.00

10 items in stock

- 0 +

0 Items Selected

Reset

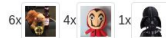
Checkout

Done! Order submitted successfully.



Purchases (11)

Purchase #1528069682033-1000171



Payments (\$11.00)

Payment #1528069682033-1000171

\$11.00

OVERVIEW OF THE WORKSHOP




Apache Camel Workshop

Plush Toys Shop


Inventory Service

- Items
 - Amount
 - Price
- Item purchases




Itchy & Scratchy
\$1.00
10 Items in stock

- 0 +




The Joker
\$1.00
10 Items in stock

- 0 +




Mr. Camel
\$1.00
10 Items in stock

- 0 +




eDonkey
\$1.00
10 Items in stock

- 0 +




Lupin
\$1.00
10 Items in stock

- 0 +




Dali
\$1.00
10 Items in stock

- 0 +



Darth Vader
\$1.00
10 Items in stock

- 0 +



Secco
\$1.00
10 Items in stock

- 0 +

0 Items Selected

Reset

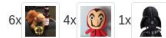
Checkout

Done! Order submitted successfully.



Purchases (11)

Purchase #1528069682033-1000171

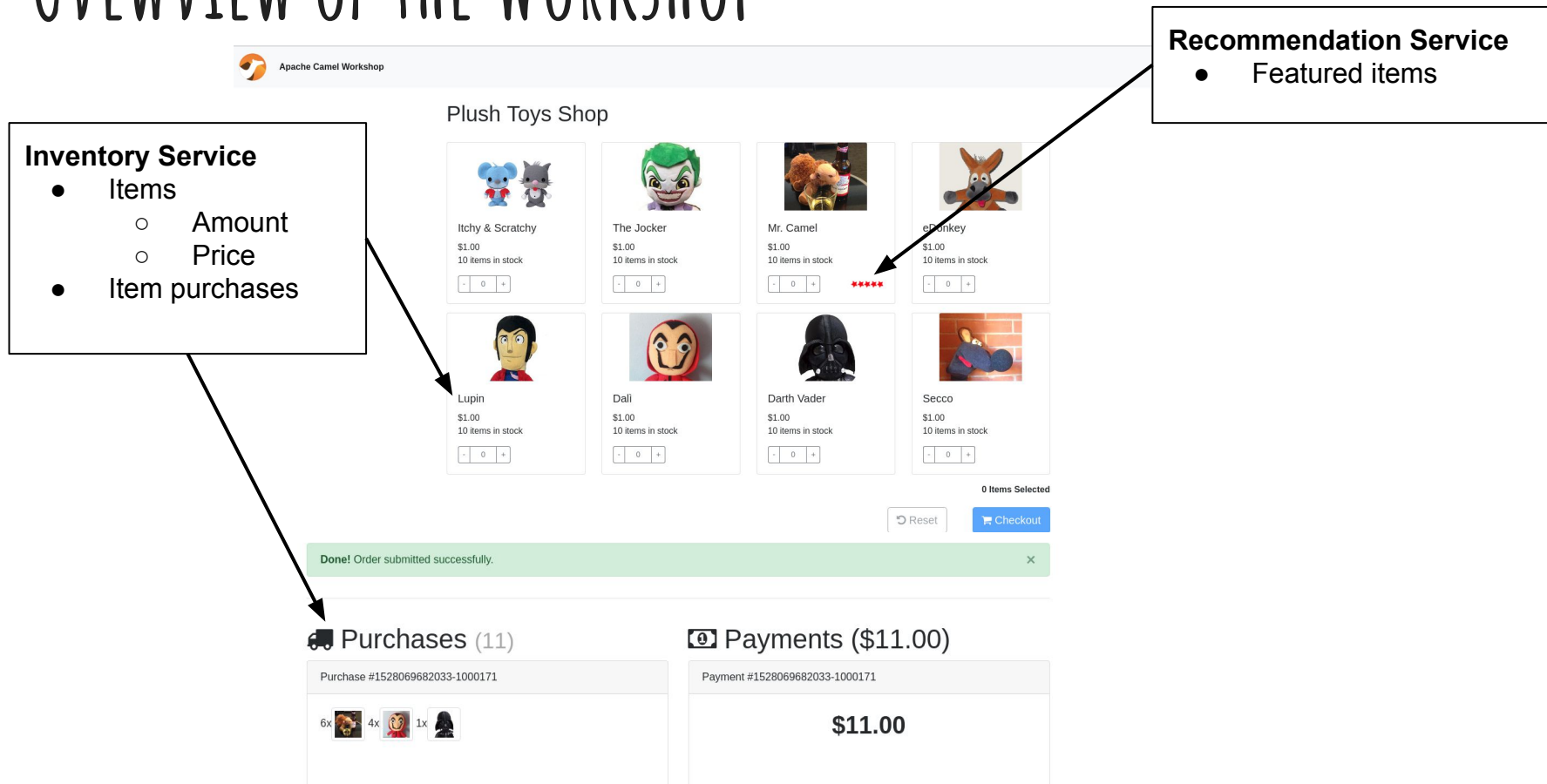


Payments (\$11.00)

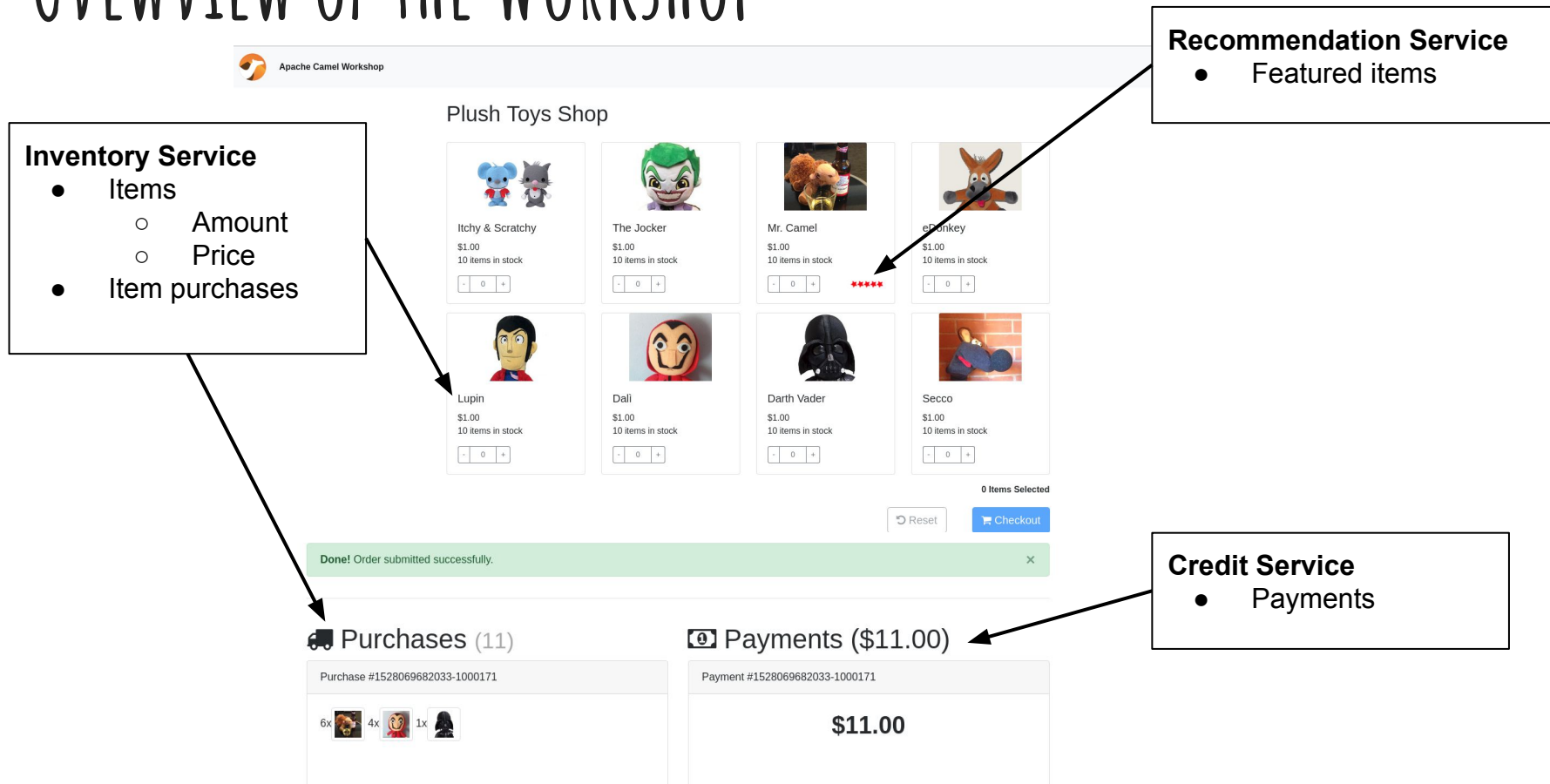
Payment #1528069682033-1000171

\$11.00

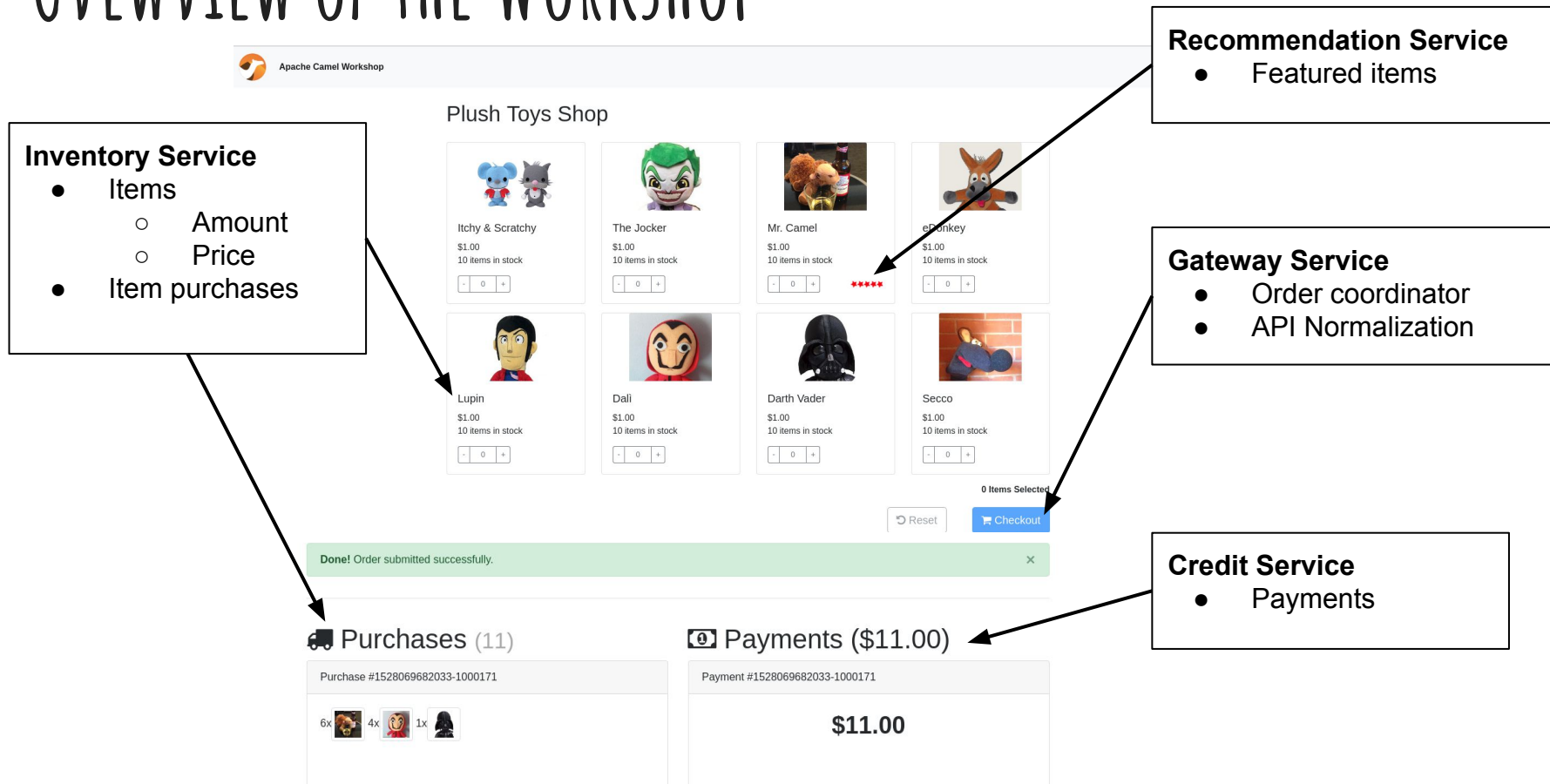
OVERVIEW OF THE WORKSHOP



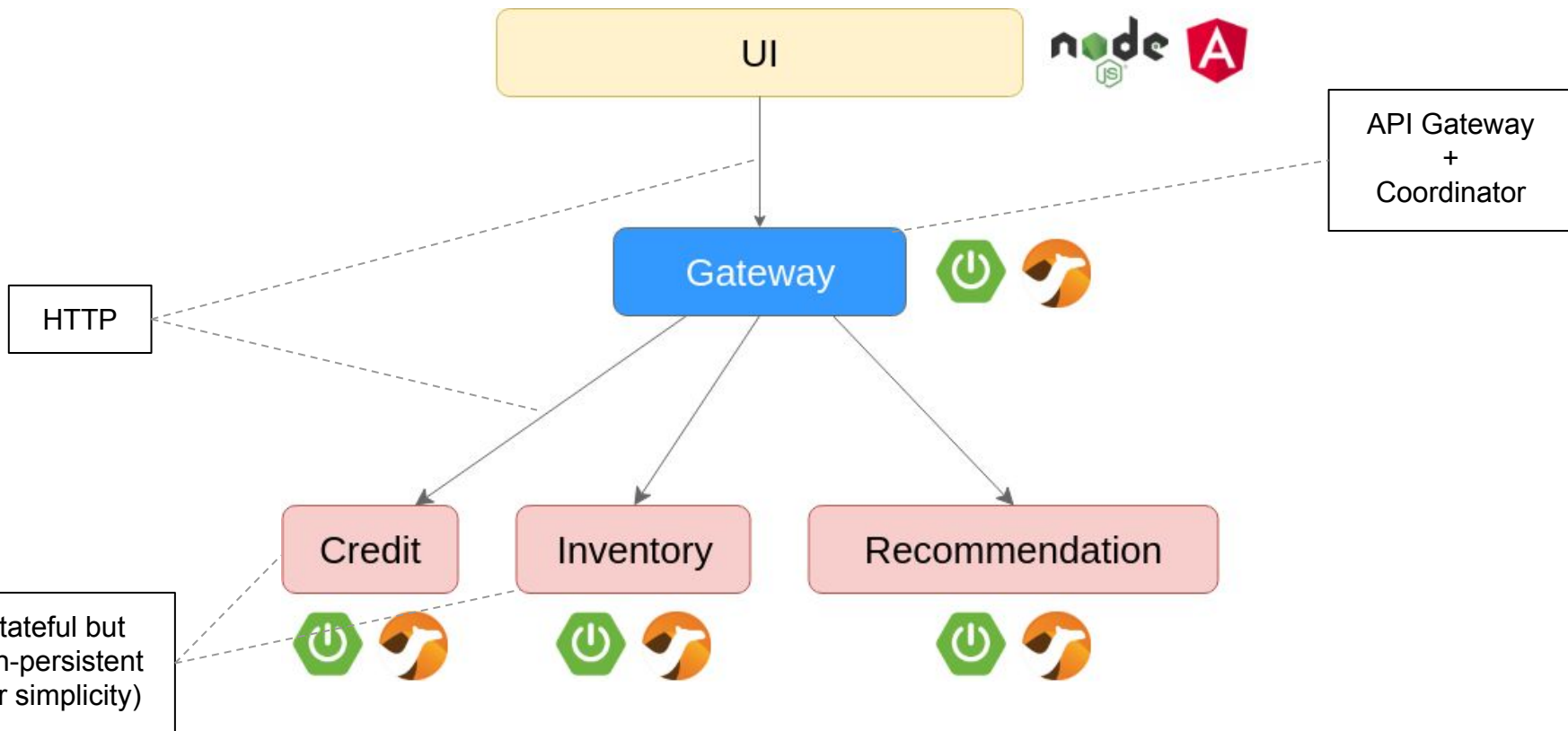
OVERVIEW OF THE WORKSHOP



OVERVIEW OF THE WORKSHOP



ARCHITECTURE



AGENDA

- Intro to Apache Camel
- Overview of the Workshop
- **Coding: Part 1** ←
- Circuit Breakers and Saga
- Coding: Part 2
- Openshift (Optional)
- Summary
- Q/A

CODING: PART 1

We'll see in this coding session:

- **REST DSL** (camel-servlet to bridge Spring Web)
- **Transformation** (camel-jackson)
- **Validation** (camel-bean-validator)
- **REST Service Documentation** (camel-swagger-java)
- **HTTP Forwarding** (camel-undertow)
- **Enterprise Integration Patterns (EIP)**: service-call, multicast, split, choice, bean call
- **Misc**: property placeholders, simple language

WORKSHOP INSTRUCTIONS

Requirements for Part 1

- Java 8
- Maven 3.5.x
- Your favourite IDE
- HTTPie (to test endpoints, instructions to install in the readme)

Follow the readme:

<https://github.com/nicolaferraro/camel-workshop>

START.SPRING.IO

Activities Google Chrome Wed 11:44

Camel Workshop x nicolaferarro/camel x Spring Initializr x

start.spring.io

SPRING INITIALIZR bootstrap your application now

Generate a with and Spring Boot

Project Metadata

Artifact coordinates

Group

Artifact

Dependencies

Add Spring Boot Starters and dependencies to your application

Search for dependencies

Selected Dependencies

Don't know what to look for? Want more options? [Switch to the full version.](#)

AGENDA

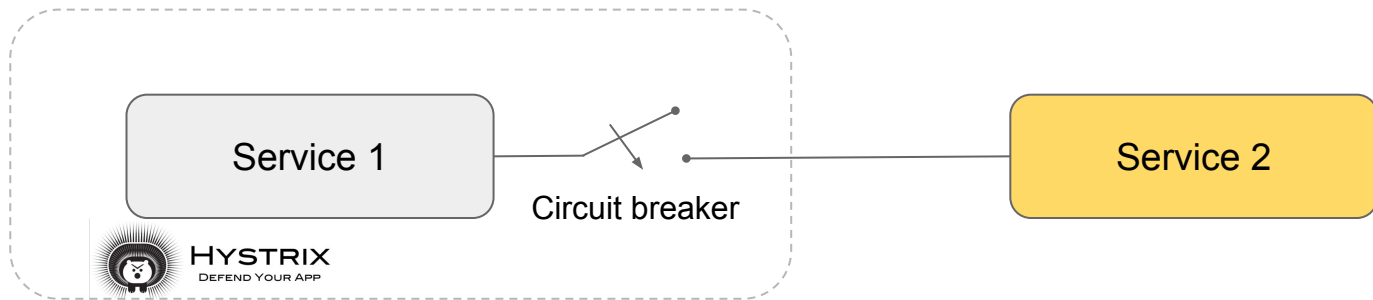
- Intro to Apache Camel
- Overview of the Workshop
- Coding: Part 1
- **Circuit Breakers and Saga** ←
- Coding: Part 2
- Openshift (Optional)
- Summary
- Q/A

CIRCUIT BREAKERS: OPEN / CLOSED

Circuit breakers are like 2-state light switches:

- Closed: service 1 tries to contact service 2
- Open: service 1 does not even try contact service 2

Why? **To prevent cascading failures, fault tolerance, resilience, fail fast adding a fallback strategy**



The hystrix library is embedded in the calling service

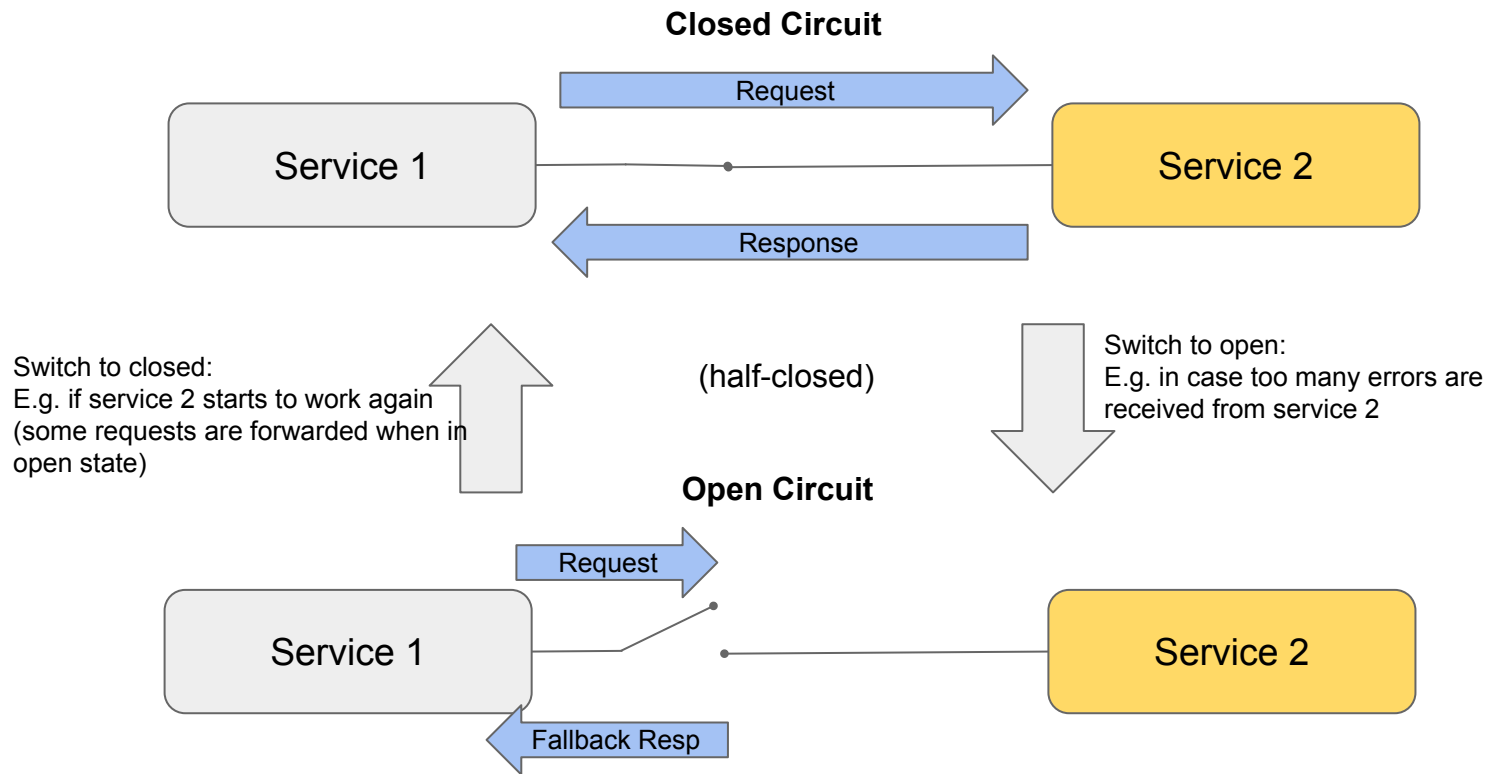
CIRCUIT BREAKERS

Camel includes a `.hystrix()` EIP for circuit breaking.

```
from("direct:route")
  .hystrix()
    .to("direct:main-action") // ← action(s) we want to make
  .onFallback()
    .setBody(constant("fallback value")) // ← fallback value in case of error
  .end()
```

It's different from a **try/catch**!

CIRCUIT BREAKERS: OPEN / CLOSED

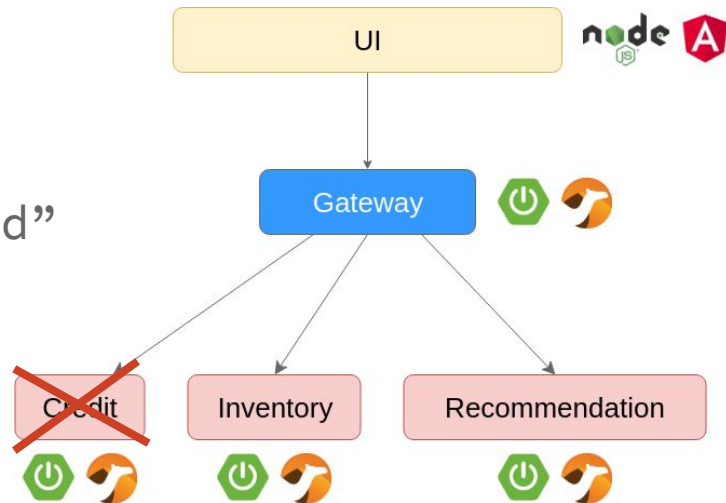


SAGA

Imagine the following scenario:

- Credit service unavailable
- User makes a order
- Items are marked as “to be shipped” in Inventory
- But credit cannot be decreased!
- **Result = items given for free!**

The solution: a saga!

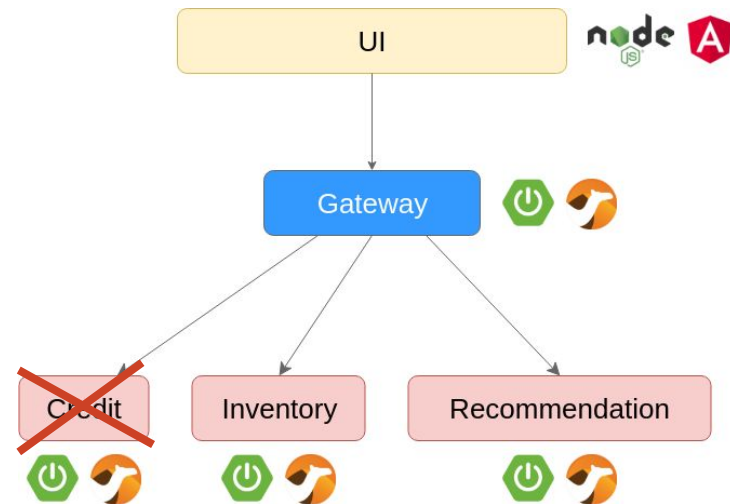


SAGA

A pattern for coordinating actions in remote services in order to obtain a consistent outcome.

A different workflow:

- ...
- Credit cannot be decreased!
- **Compensation: re-add the items in the inventory!**
- **Result: consistency is reestablished!**



SAGA: IN CAMEL

Camel includes a `.saga()` EIP for declaring sagas.

```
from("direct:route")  
  .saga().compensation("direct:cancel-action")  
    .to("direct:main-action") // ← action(s) we want to make
```

```
from("direct:cancel-action")  
  .to("...") // ← compensation
```

Rule: if something fails, compensation **must** be done!

Otherwise consistency is not guaranteed.

SAGA: IN CAMEL

How consistency is guaranteed:

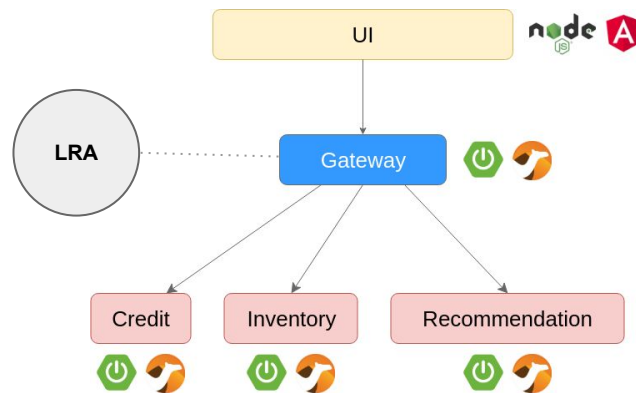
- Compensation is retried until successful
- Compensating action must be idempotent
- Compensating action must be commutative wrt main action
- Status of Sagas can be written to a persistent log

LRA Coordinator for persistence:

<https://github.com/eclipse/microprofile-lra>

camel-lra supports it already!

Implementation: <https://github.com/jbosstm/narayana>



AGENDA

- Intro to Apache Camel
- Overview of the Workshop
- Coding: Part 1
- Circuit Breakers and Saga
- **Coding: Part 2** ←
- Openshift (Optional)
- Summary
- Q/A

CODING: PART 2

We'll see in this coding session:

- **Circuit Breaker** (camel-hystrix)
- **Cache** (camel-caffeine)
- **Saga** (camel-core)

Follow **part 2** on:

<https://github.com/nicolaferraro/camel-workshop>

AGENDA

- Intro to Apache Camel
- Overview of the Workshop
- Coding: Part 1
- Circuit Breakers and Saga
- Coding: Part 2
- **Openshift (Optional)** ←
- Summary
- Q/A

AGENDA

- Intro to Apache Camel
- Overview of the Workshop
- Coding: Part 1
- Circuit Breakers and Saga
- Coding: Part 2
- Openshift (Optional)
- **Summary** ←
- Q/A

SUMMARY




APACHE™

Camel

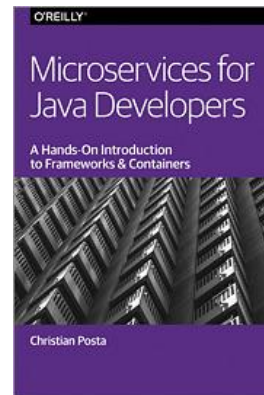
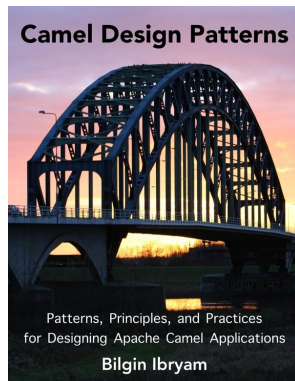
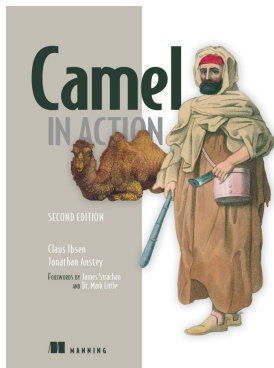
- **Camel is Awesome!**
- Camel loves Spring Boot and Spring Boot loves Camel
<https://spring.io/blog/2018/05/23/spring-tips-apache-camel>
- Camel can do microservices
- Camel can integrate with everything

THANK YOU!

 @davsclaus
 @ni_ferraro

Source code and slides:

<https://github.com/nicolaferraro/camel-workshop>



<https://www.manning.com/books/camel-in-action-second-edition>

<https://leanpub.com/camel-design-patterns>

<https://developers.redhat.com/promotions/microservices-for-java-developers/>