

Getting started with Apache Camel

by Claus Ibsen

Principal Software Engineer, FuseSource

@davsclaus

FuseSource

integration everywhere

Agenda

- What is Apache Camel?
- Basic Camel Concepts
- Running Camel
- Whats included in the box?
- Creating new Camel Projects
- Extending Camel
- Roadmap
- Q and A

What is Apache Camel?

- "Swiss Army Knife" of Integration



System A



System B

What is Apache Camel?

- "Swiss Army Knife" of Integration with a lot of power



System A



System B

What is Apache Camel?

- Quote from the web site
 - <http://camel.apache.org>

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

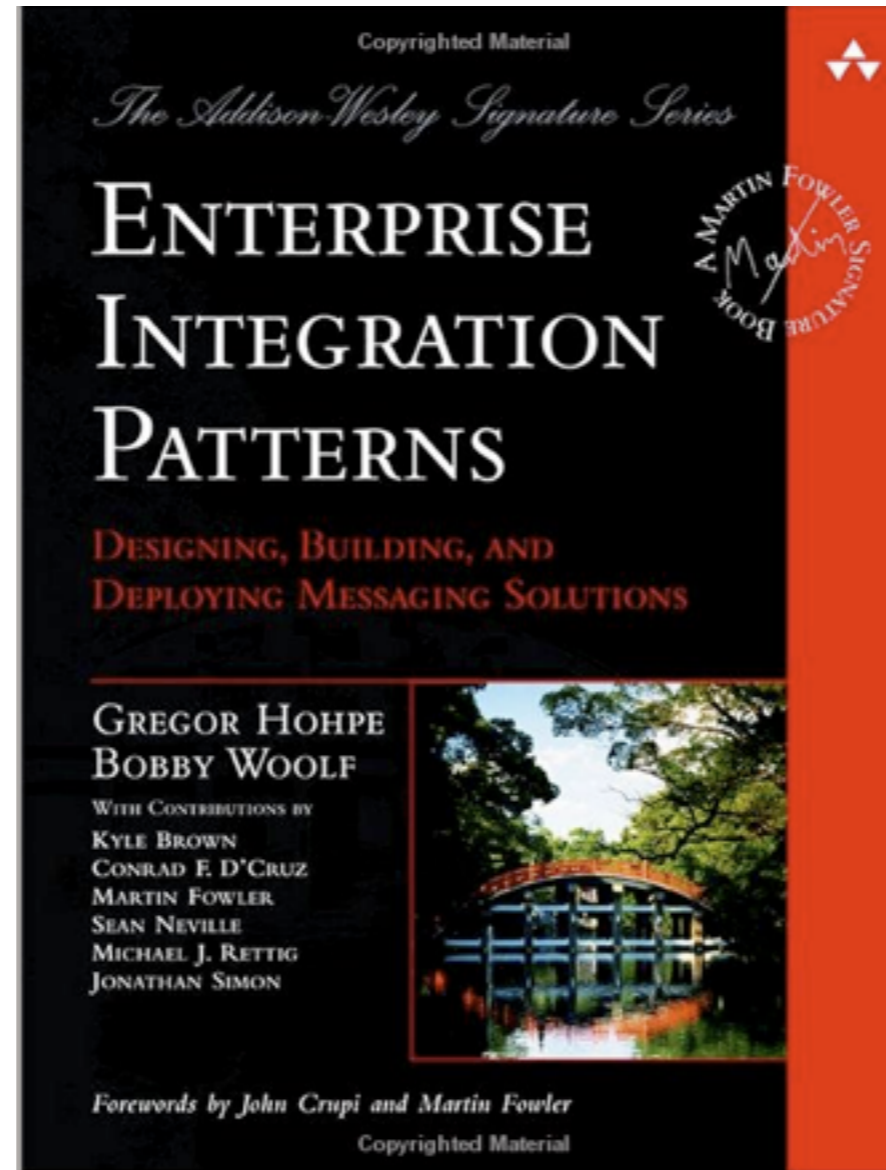
What is Apache Camel?

- Why do we need integration?
 - Your apps are build using different tech stacks
 - Critical for your business to integrate
- Why Integration Framework?
 - Framework do the heavy lifting
 - Focus on business problem
 - Not "reinventing the wheel"



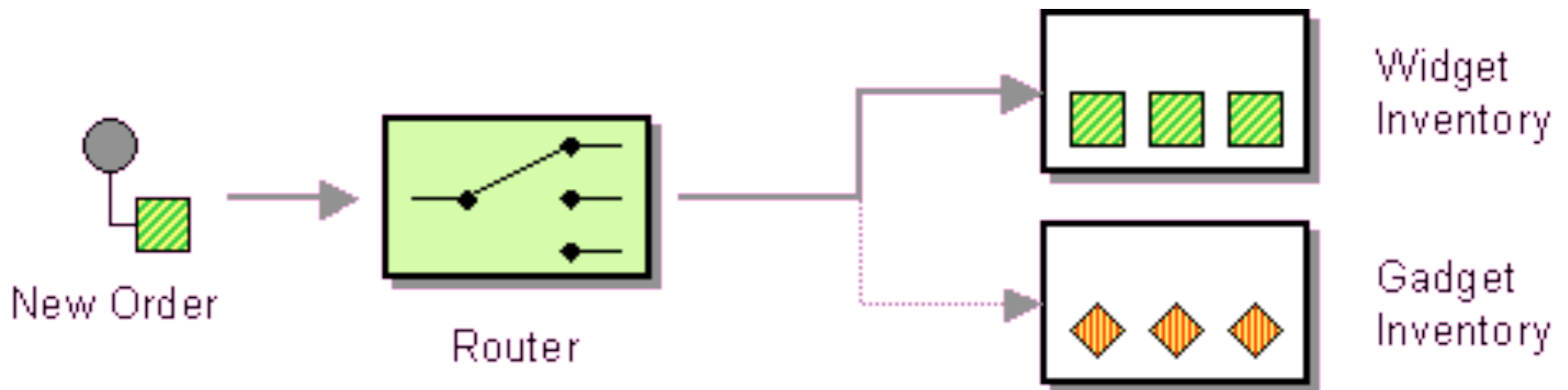
What is Apache Camel?

- It all starts with this book



What is Apache Camel?

- Enterprise Integration Pattern - Content Based Router



What is Apache Camel?

- Content Based Router - XML DSL

```
<camelContext>
  <route>
    <from uri="activemq:NewOrders" />
    <choice>
      <when>
        <xpath>/order/product = 'widget' </xpath>
        <to uri="activemq:Orders.Widgets" />
      </when>
      <otherwise>
        <to uri="activemq:Orders.Gadgets" />
      </otherwise>
    </choice>
  </route>
</camelContext>
```

What is Apache Camel?

- Content Based Router - Java DSL

```
from("activemq:NewOrders")  
  .choice()  
    .when().xpath("/order/product = 'widget'")  
      .to("activemq:Orders.Widget")  
    .otherwise()  
      .to("activemq:Orders.Gadget");
```

What is Apache Camel?

- Endpoints as URIs

use file instead

```
from("file:inbox/orders")  
  .choice()  
    .when().xpath("/order/product = 'widget'")  
      .to("activemq:Orders.Widget")  
    .otherwise()  
      .to("activemq:Orders.Gadget");
```

What is Apache Camel?

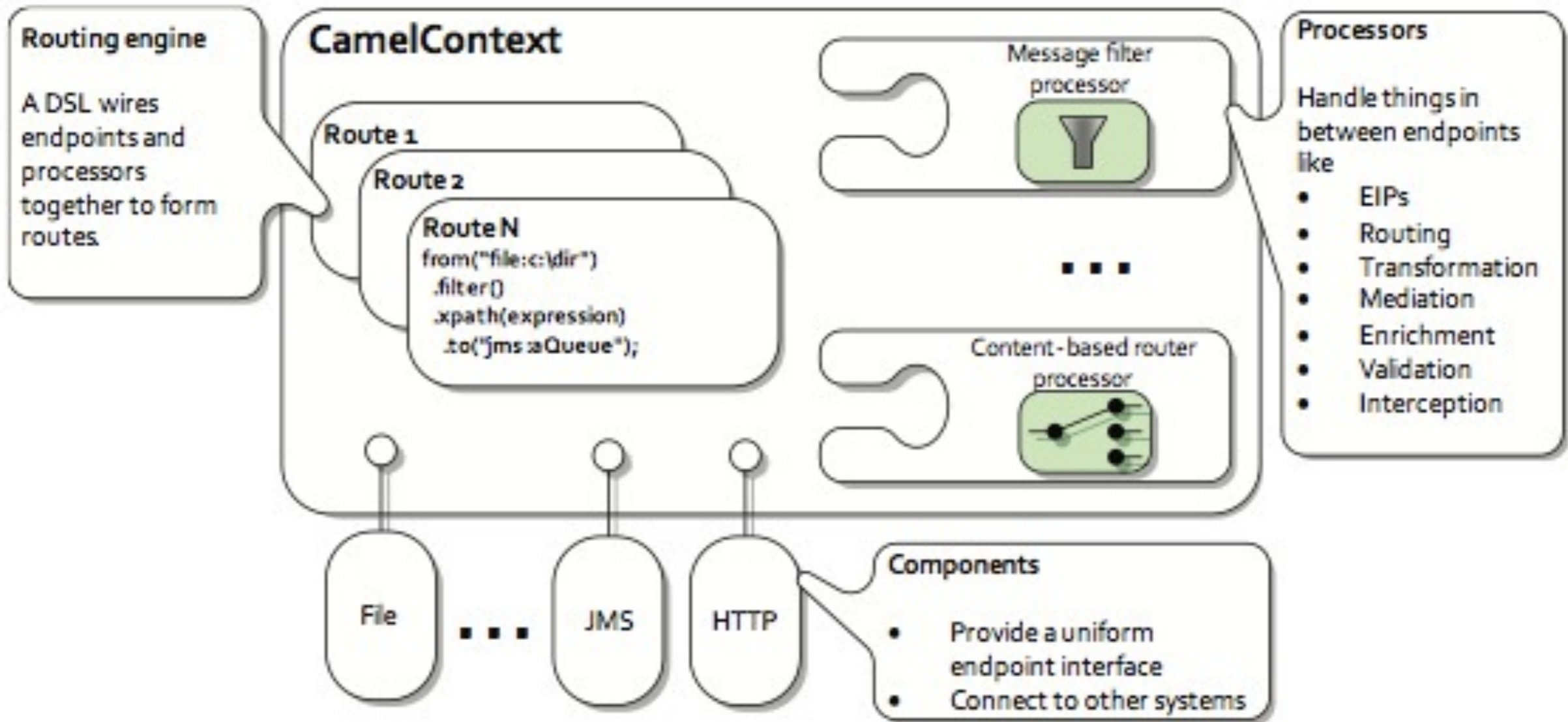
- Endpoints as URIs

parameters

```
from("file:inbox/orders?delete=true")  
  .choice()  
    .when().xpath("/order/product = 'widget'")  
      .to("activemq:Orders.Widget")  
    .otherwise()  
      .to("activemq:Orders.Gadget");
```

What is Apache Camel?

■ Camel Architecture



What is Apache Camel?

■ Summary

- Integration framework
- Enterprise Integration Patterns (EIP)
- Routing (using DSL)
- Easy configuration (endpoints as URIs)
- No heavy specification
- No container dependency
- Payload agnostic
- A lot of components

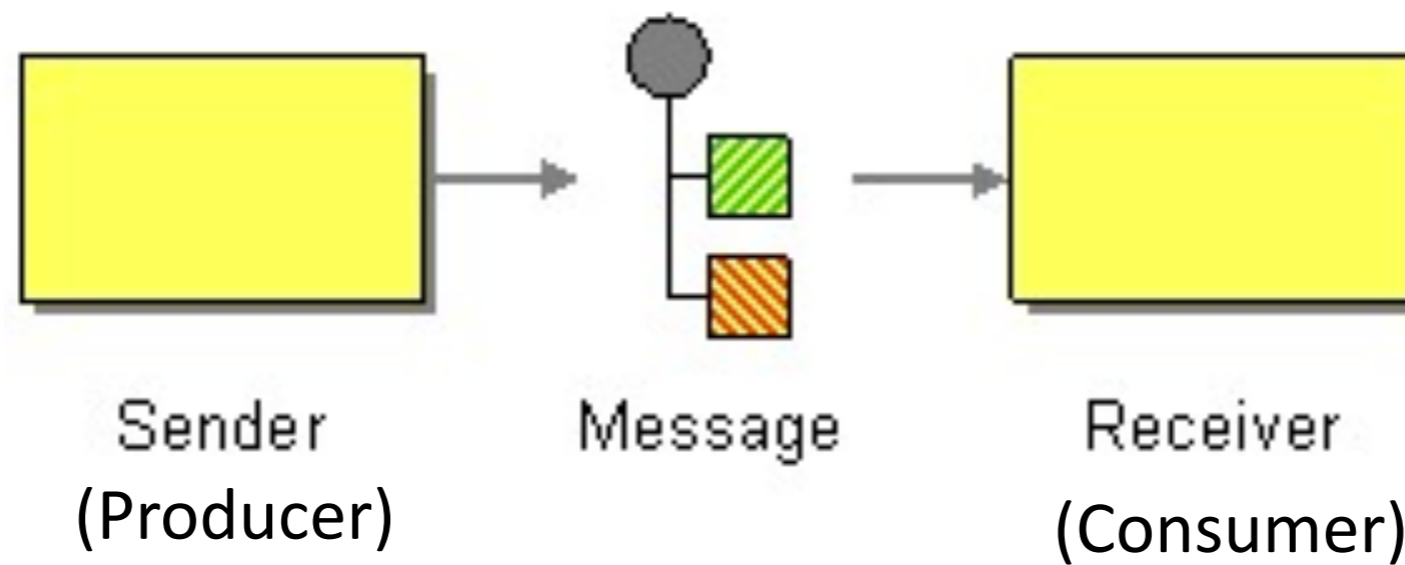


Agenda

- What is Apache Camel?
- **Basic Camel Concepts**
- Running Camel
- Whats included in the box?
- Creating new Camel Projects
- Extending Camel
- Roadmap
- Q and A

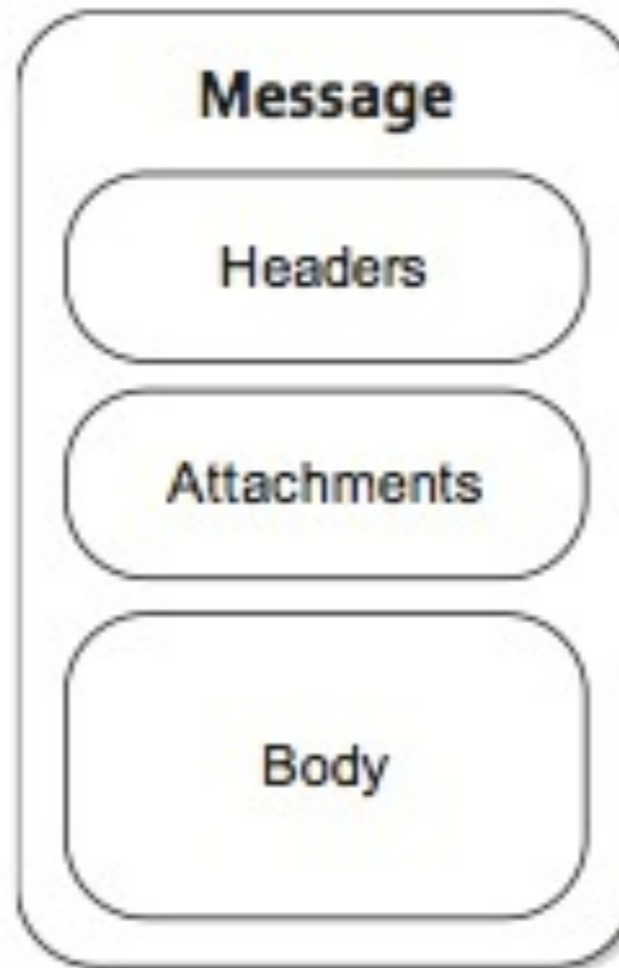
Basic Camel Concepts

- Messaging Concepts



Basic Camel Concepts

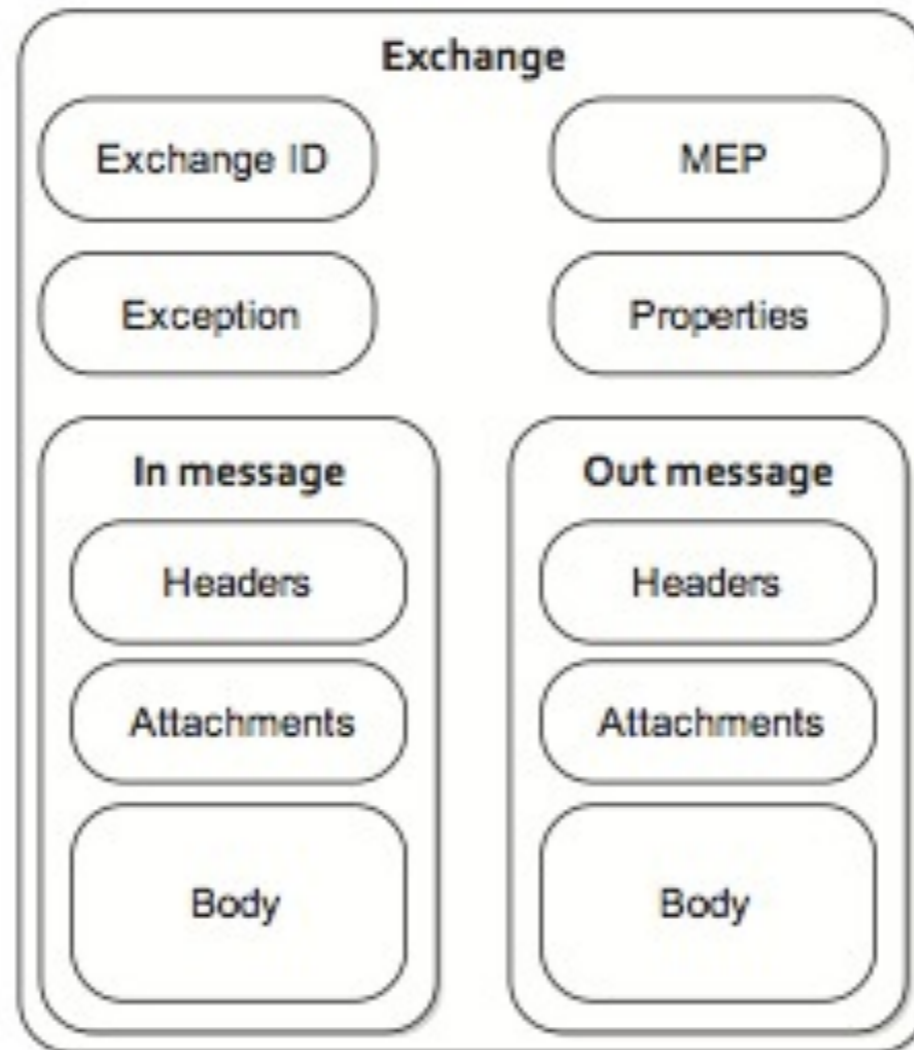
- What is a Message?



`org.apache.camel.Message`

Basic Camel Concepts

- ... And Message is contained in an Exchange



`org.apache.camel.Exchange`

Basic Camel Concepts

- File Copier Example



Figure 1.2 Files are routed from the `data/inbox` directory to the `data/outbox` directory.

Basic Camel Concepts

- File Copier Example

Listing 1.2 Routing files from one folder to another with Apache Camel

```
public class FileCopierWithCamel {
    public static void main(String args[]) throws Exception {
        CamelContext context = new DefaultCamelContext();
        context.addRoutes(new RouteBuilder() {
            public void configure() {
                from("file:data/inbox?noop=true")
                    .to("file:data/outbox");
            }
        });
        context.start();
        Thread.sleep(10000);
        context.stop();
    }
}
```

1 Routes files from inbox to outbox

Basic Camel Concepts

- File Copier Example

Listing 1.2 Routing files from one folder to another with Apache Camel

```
public class FileCopierWithCamel {  
    public static void main(String args[]) throws Exception {  
        CamelContext context = new DefaultCamelContext();  
        context.addRoutes(new RouteBuilder() {  
            public void configure() {  
                from("file:data/inbox?noop=true")  
                    .to("file:data/outbox");  
            }  
        });  
        context.start();  
        Thread.sleep(10000);  
        context.stop();  
    }  
}
```

1 Routes files from
inbox to outbox

Basic Camel Concepts

- File Copier Example

Listing 1.2 Routing files from one folder to another with Apache Camel

```
public class FileCopierWithCamel {  
    public static void main(String args[]) throws Exception {  
        CamelContext context = new DefaultCamelContext();  
        context.addRoutes(new RouteBuilder() {  
            public void configure() {  
                from("file:data/inbox?noop=true")  
                    .to("file:data/outbox");  
            }  
        });  
        context.start();  
        Thread.sleep(10000);  
        context.stop();  
    }  
}
```

1 Routes files from
inbox to outbox

Basic Camel Concepts

- File Copier Example

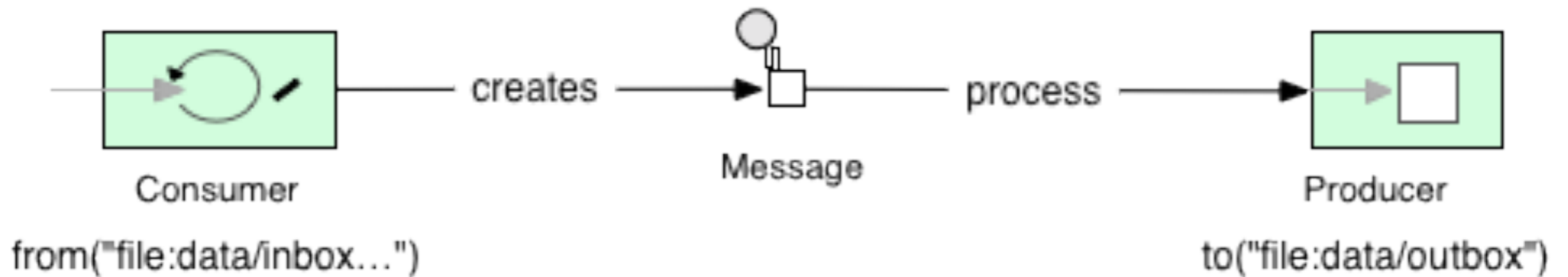
Listing 1.2 Routing files from one folder to another with Apache Camel

```
public class FileCopierWithCamel {  
    public static void main(String args[]) throws Exception {  
        CamelContext context = new DefaultCamelContext();  
        context.addRoutes(new RouteBuilder() {  
            public void configure() {  
                from("file:data/inbox?noop=true")  
                    .to("file:data/outbox");  
            }  
        });  
        context.start();  
        Thread.sleep(10000);  
        context.stop();  
    }  
}
```

1 Routes files from
inbox to outbox

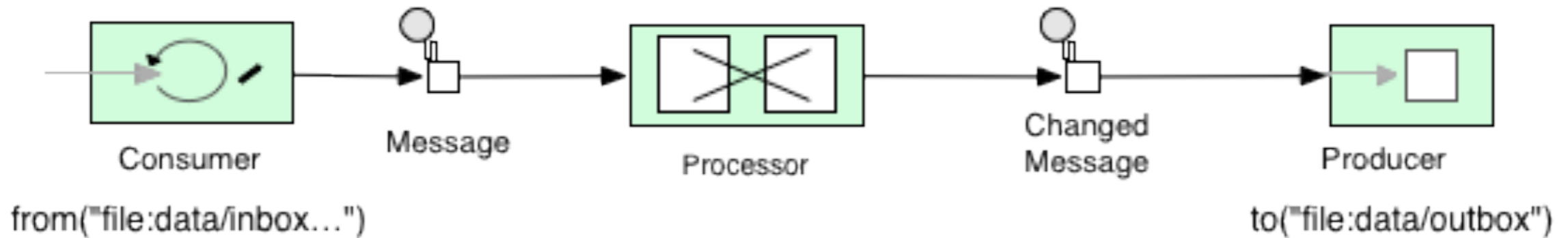
Basic Camel Concepts

- File Copier Example

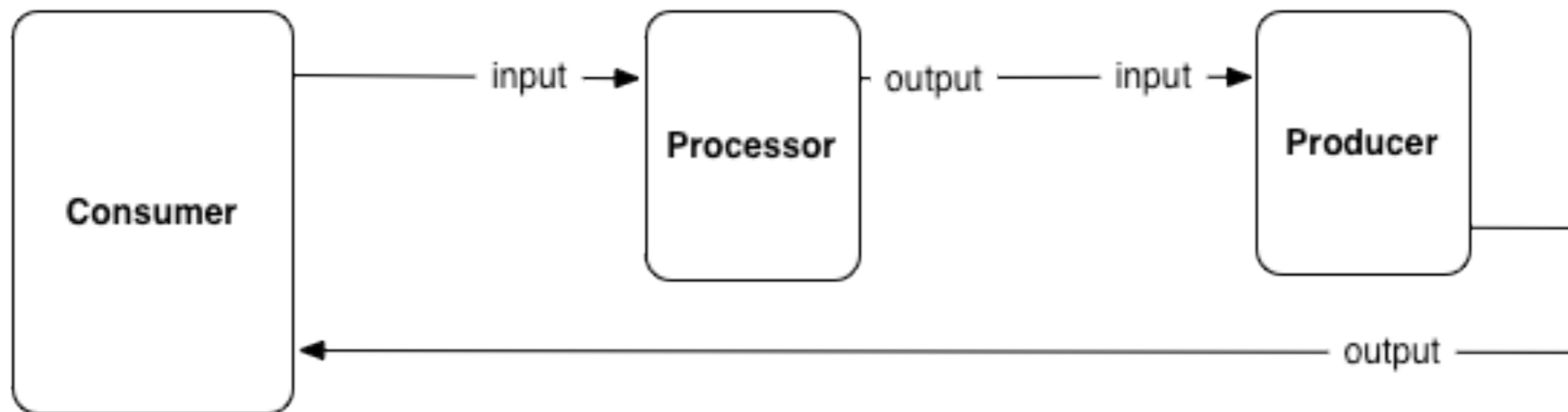


Basic Camel Concepts

- File Copier w/ Transformation Example



- Pipes and Filters EIP

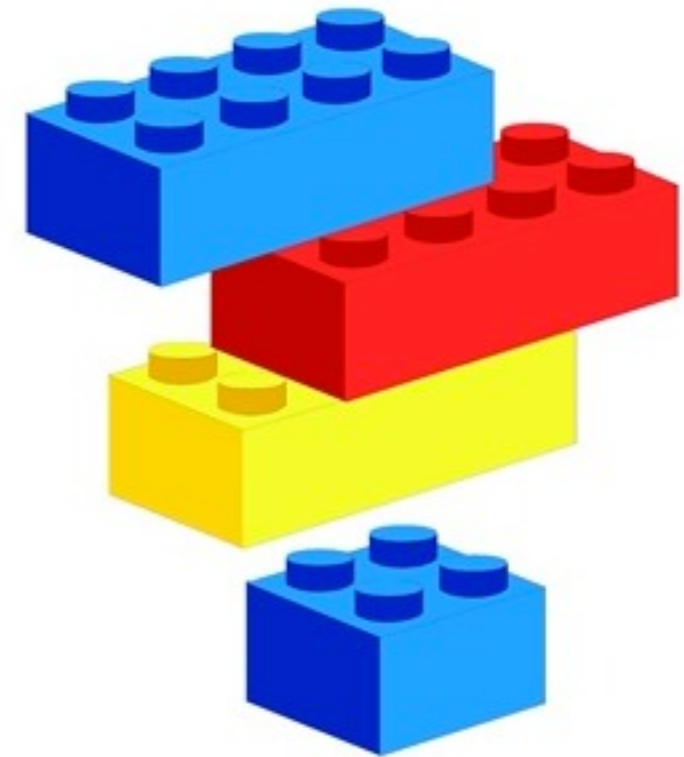


Dissecting the DSL

- Demystifying the "woodoo DSL"

Listing 1.2 Routing files from one folder to another with Apache Camel

```
public class FileCopierWithCamel {  
    public static void main(String args[]) throws Exception {  
        CamelContext context = new DefaultCamelContext();  
        context.addRoutes(new RouteBuilder() {  
            public void configure() {  
                from("file:data/inbox?noop=true")  
                    .to("file:data/outbox");  
            }  
        });  
        context.start();  
        Thread.sleep(10000);  
        context.stop();  
    }  
}
```



Dissecting the DSL

- Demystifying the "woodoo DSL"

```
public class FileCopierWithCamel {  
  
    public static void main(String[] args) throws Exception {  
        ModelCamelContext context = new DefaultCamelContext();  
  
        // create endpoints manually  
        FileEndpoint inbox = new FileEndpoint();  
        inbox.setFile(new File("data/inbox"));  
        inbox.setNoop(true);  
  
        FileEndpoint outbox = new FileEndpoint();  
        outbox.setFile(new File("data/outbox"));  
  
        // demystify DSL by creating the route manually in hand  
        FromDefinition from = new FromDefinition(inbox);  
        ToDefinition to = new ToDefinition(outbox);  
  
        RouteDefinition route = new RouteDefinition();  
        route.getInputs().add(from);  
        route.getOutputs().add(to);  
  
        context.addRouteDefinition(route);  
  
        context.start();  
        Thread.sleep(10000);  
        context.stop();  
    }  
}
```

Dissecting the DSL

- Demystifying the "woodoo DSL"

```
public static void main(String[] args) throws Exception {
    ModelCamelContext context = new DefaultCamelContext();

    // demystify DSL by creating the route manually in hand
    FromDefinition from = new FromDefinition("file:data/inbox/?noop=true");
    ToDefinition to = new ToDefinition("file:data/outbox");

    RouteDefinition route = new RouteDefinition();
    route.getInputs().add(from);
    route.getOutputs().add(to);

    context.addRouteDefinition(route);

    context.start();
    Thread.sleep(10000);
    context.stop();
}
```

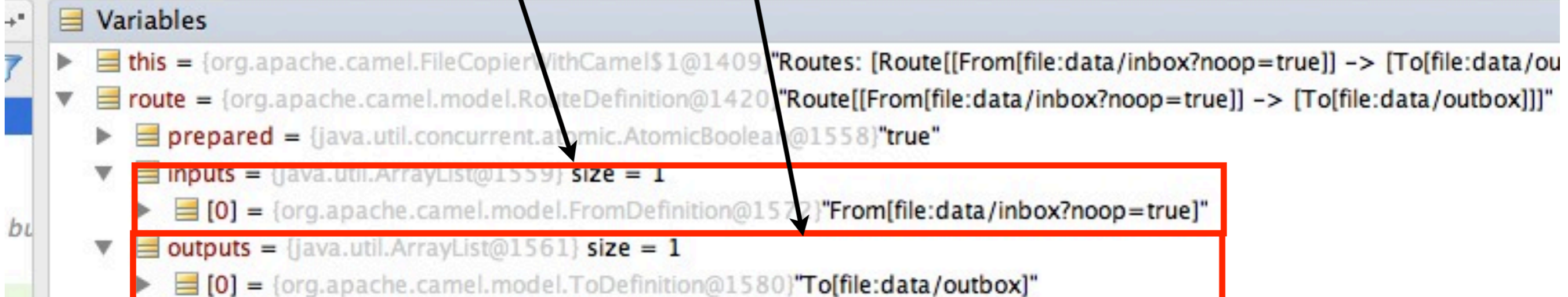
Dissecting the DSL

```
26 public class FileCopierWithCamel {
27
28     public static void main(String[] args) throws Exception {
29         CamelContext context = new DefaultCamelContext();
30         context.addRoutes(new RouteBuilder() {
31             @Override
32             public void configure() throws Exception {
33                 Object route =
34                     from("file:data/inbox?noop=true")
35                     .to("file:data/outbox");
36
37                 log.info("The route: {}", route);
38             }
39         });

```

Variables

- ▶ this = {org.apache.camel.FileCopierWithCamel\$1@1409 "Routes: [Route[[From[file:data/inbox?noop=true]] -> [To[file:data/ou
- ▼ route = {org.apache.camel.model.RouteDefinition@1420 "Route[[From[file:data/inbox?noop=true]] -> [To[file:data/outbox]]"
- ▶ prepared = {java.util.concurrent.atomic.AtomicBoolean@1558}"true"
- ▼ inputs = {java.util.ArrayList@1559} size = 1
 - ▶ [0] = {org.apache.camel.model.FromDefinition@1572}"From[file:data/inbox?noop=true]"
- ▼ outputs = {java.util.ArrayList@1561} size = 1
 - ▶ [0] = {org.apache.camel.model.ToDefinition@1580}"To[file:data/outbox]"

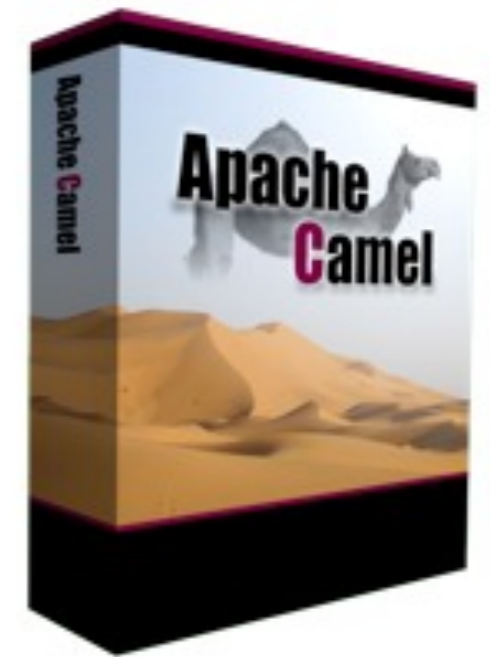
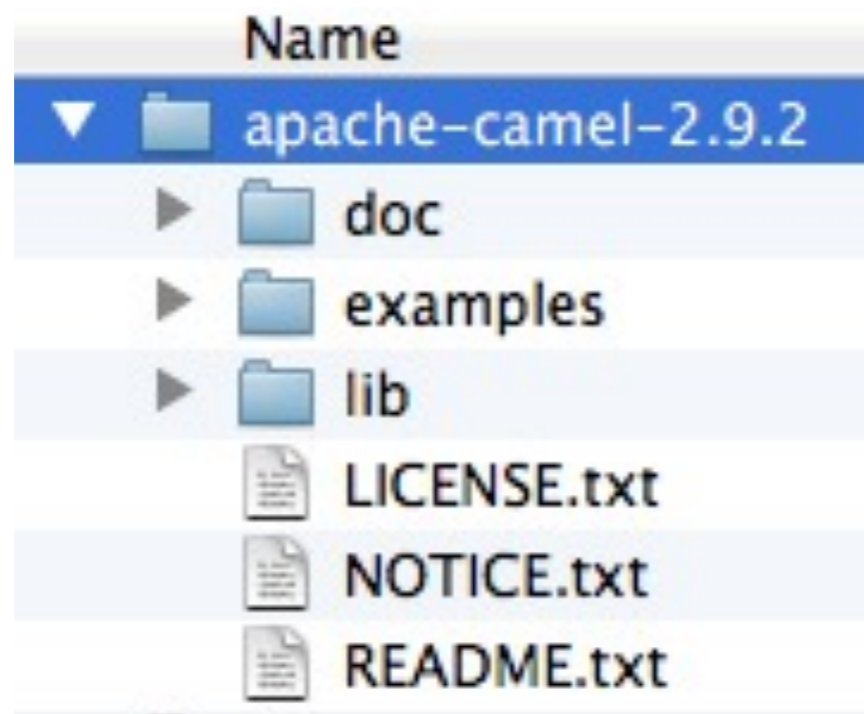


Agenda

- What is Apache Camel?
- Basic Camel Concepts
- **Running Camel**
- Whats included in the box?
- Creating new Camel Projects
- Extending Camel
- Roadmap
- Q and A

Downloading Camel

- Download from Apache Camel website
 - zip/tarball (approx 13mb)



Running your first example

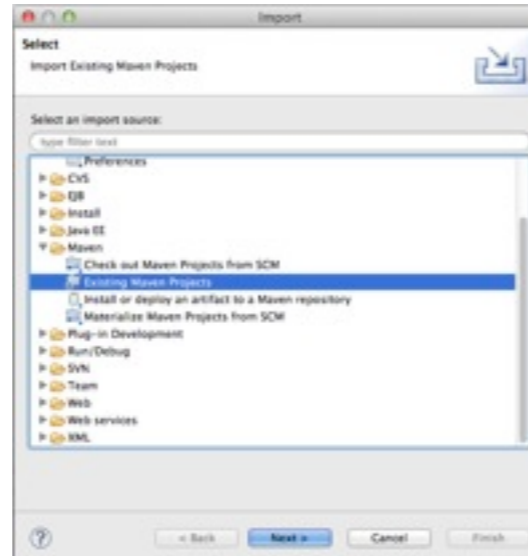
■ Trying your first example



- unzip/untar the Apache Camel distribution
- cd examples/camel-example-spring
- cat README.txt
- mvn compile
- mvn camel:run
 - ctrl + c to stop

Running your first example

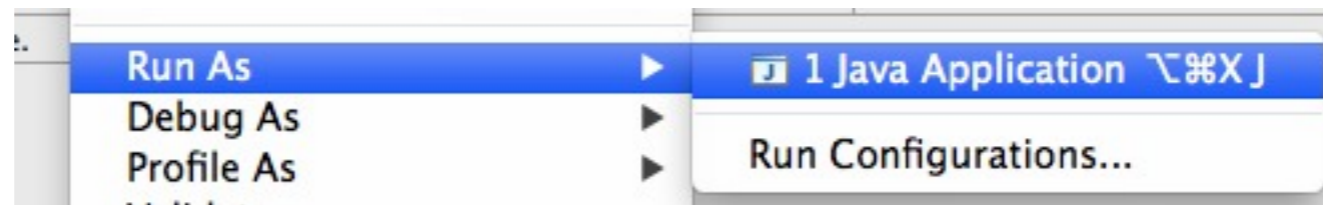
- Import into Eclipse



- Edit Camel route

```
public void configure() {  
    // populate the message queue with some messages  
    from("file:src/data?noop=true").  
        to("jms:test.MyQueue");  
  
    from("jms:test.MyQueue").  
        to("file://target/test?noop=true");  
  
    // set up a listener on the file component  
    from("file://target/test?noop=true").  
        bean(new SomeBean());  
}
```

- Run Application



Deployment Options

- Deployment Strategy
 - No container dependency
 - Lightweight
 - Embedable
- Deployment Options
 - Standalone
 - WAR
 - Spring
 - JEE
 - OSGi
 - Cloud

Known Containers

Apache ActiveMQ
Apache ServiceMix
Apache Tomcat
Fuse ESB / Fuse MQ
Glassfish
IBM WebSphere
JBoss AS
Jetty
Oracle OC4j
Oracle WebLogic
Amazon EC2
Google App Engine
Rack space
... others

Running Camel as Client

- Java Client Application (no routes)

```
CamelContext context = new DefaultCamelContext();
ProducerTemplate template = context.createProducerTemplate();

String data = ...
String fileName = ...

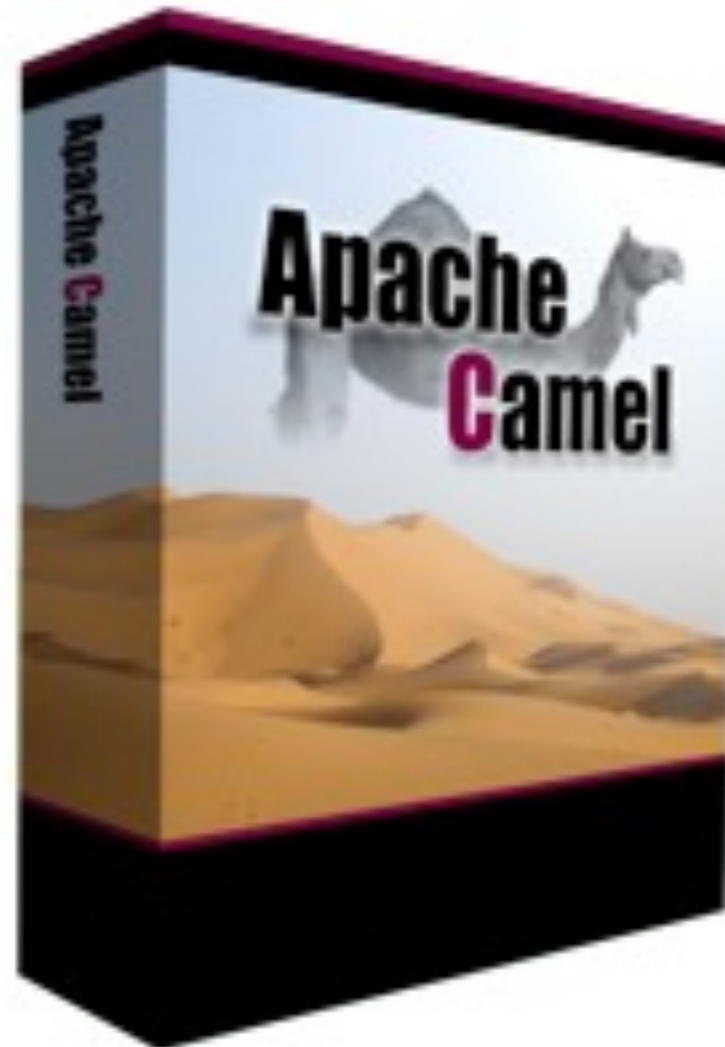
String uri = "ftp://myserver?username=foo&password=secret";
template.sendBodyAndHeader(uri, data, Exchange.FILE_NAME, fileName);
```

Agenda

- What is Apache Camel?
- Basic Camel Concepts
- Running Camel
- **Whats included in the box?**
- Creating new Camel Projects
- Extending Camel
- Roadmap
- Q and A

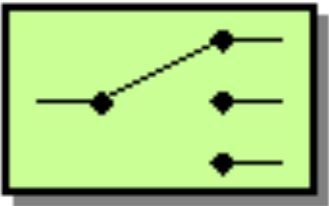
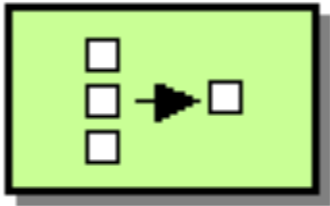
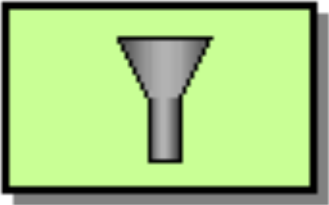
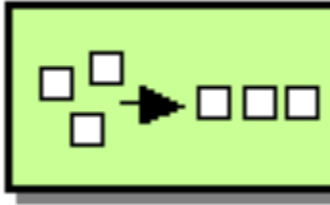
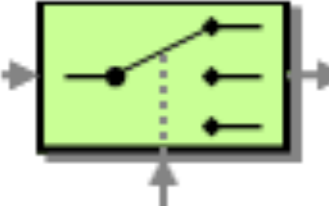
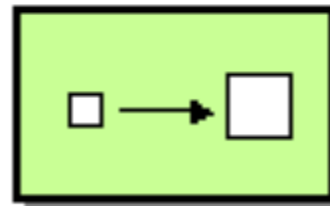
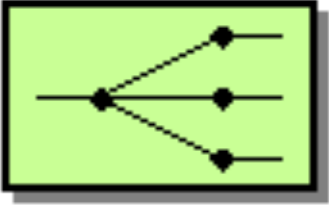
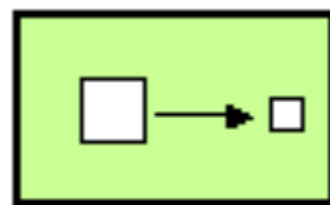
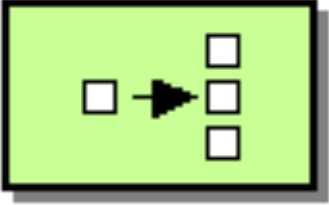
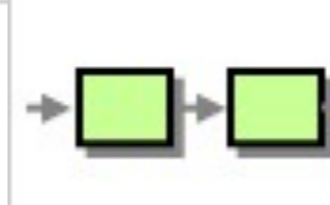
Whats included in the box?

- Highlights of some of the features



Whats included in the box?

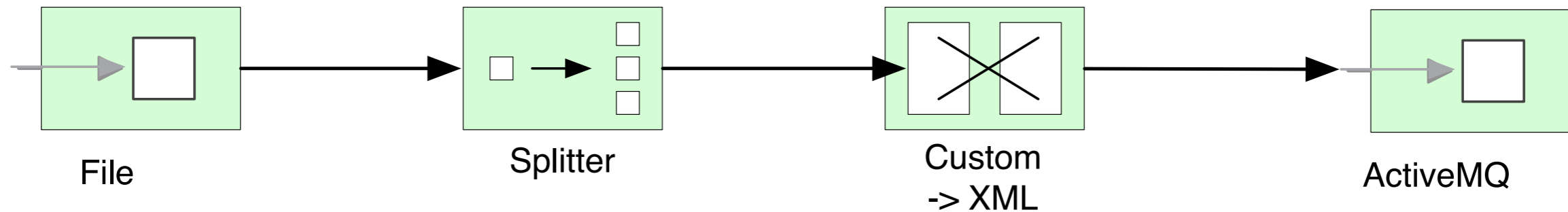
- 50 Enterprise Integration Patterns

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

<http://camel.apache.org/eip>

Whats included in the box?

- Splitter EIP



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

Whats included in the box?

- 100 Components

activemq	cxfrs	flatpack	jasypt
activemq-journal	cxfrs	freemarker	javaspace
amqp	dataset	ftp/ftps/sftp	jbi
atom	db4o	gae	jcr
bean	direct	hdfs	jdbc
bean validation	ejb	hibernate	jetty
browse	esper	hl7	jms
cache	event	http	jmx
cometd	exec	ibatis	jpa
crypto	file	irc	jt/400

<http://camel.apache.org/components.html>

Whats included in the box?

- 100 Components

language	properties	seda	stream
ldap	quartz	servlet	string-template
mail/imap/pop3	quickfix	sip	test
mina	ref	smooks	timer
mock	restlet	smpp	validation
msv	rmi	snmp	velocity
nagios	rnc	spring-integration	vm
netty	rng	spring-security	xmpp
nmr	rss	spring-ws	xquery
printer	scalate	sql	xslt

<http://camel.apache.org/components.html>

Whats included in the box?

- 22 Data Formats

bindy	json
beanio	protobuf
castor	rss
csv	serialization
crypto	soap
dozer	syslog
flatpack	tidy markup
gzip	xml beans
hl7	xml security
jaxb	xstream
jibx	zip

<http://camel.apache.org/data-format.html>

Whats included in the box?

- 16 Expression Languages

BeanShell	PHP
EL	Python
Groovy	Ruby
JavaScript	Simple
JoSQL	SpEL
JSR 223	SQL
OGNL	XPath
MVEL	XQuery

<http://camel.apache.org/languages.html>

Whats included in the box?

- DSL in multiple flavors

```
<route>
  <from ref="A" />
  <filter>
    <xpath>/quote/product = 'widget' </xpath>
    <to ref="B" />
  </filter>
</route>
```

Java

```
from(A).filter(isWidget).to(B);
```

Scala

```
from(A) filter(isWidget) --> B
```

Whats included in the box?

- Error Handling
 - Errors happen



Whats included in the box?

- Error Handling
 - Try ... Catch style

```
from("activemq:incoming")
  .doTry()
    .marshal().jaxb()
    .to("mq:QueueWithXmlMessages")
  .doCatch(Exception.class)
    .to("activemq:error")
  .end();
```

Whats included in the box?

- Error Handling
 - Dead Letter Channel EIP

```
errorHandler(  
    deadLetterChannel("activemq:error")  
);  
  
from("activemq:incoming")  
    .marshal().jaxb()  
    .to("mq:QueueWithXmlMessages");
```

Whats included in the box?

- Error Handling
 - Dead Letter Channel EIP w/ Redelivery

```
errorHandler(  
    deadLetterChannel("activemq:error")  
        .maximumRedeliveries(5)  
        .redeliveryDelay(5000)  
);  
  
from("activemq:incoming")  
    .marshal().jaxb()  
    .to("mq:QueueWithXmlMessages");
```


Whats included in the box?

- Error Handling
 - Exception Clauses (per exception handling)

```
onException(IOException.class)
    .maximumRedeliveries(5)
    .redeliveryDelay(5000)
);

from("activemq:incoming")
    .marshal().jaxb()
    .to("ftp:someserver?username=...");
```

Whats included in the box?

- Additional features
 - Transactions
 - JMX management
 - Add/remove routes at runtime
 - Non-blocking asynchronous routing engine
 - Interceptors
 - Configurable Threading Model
 - Graceful shutdown
 - Powerful Test Kit
 - ... and more

Agenda

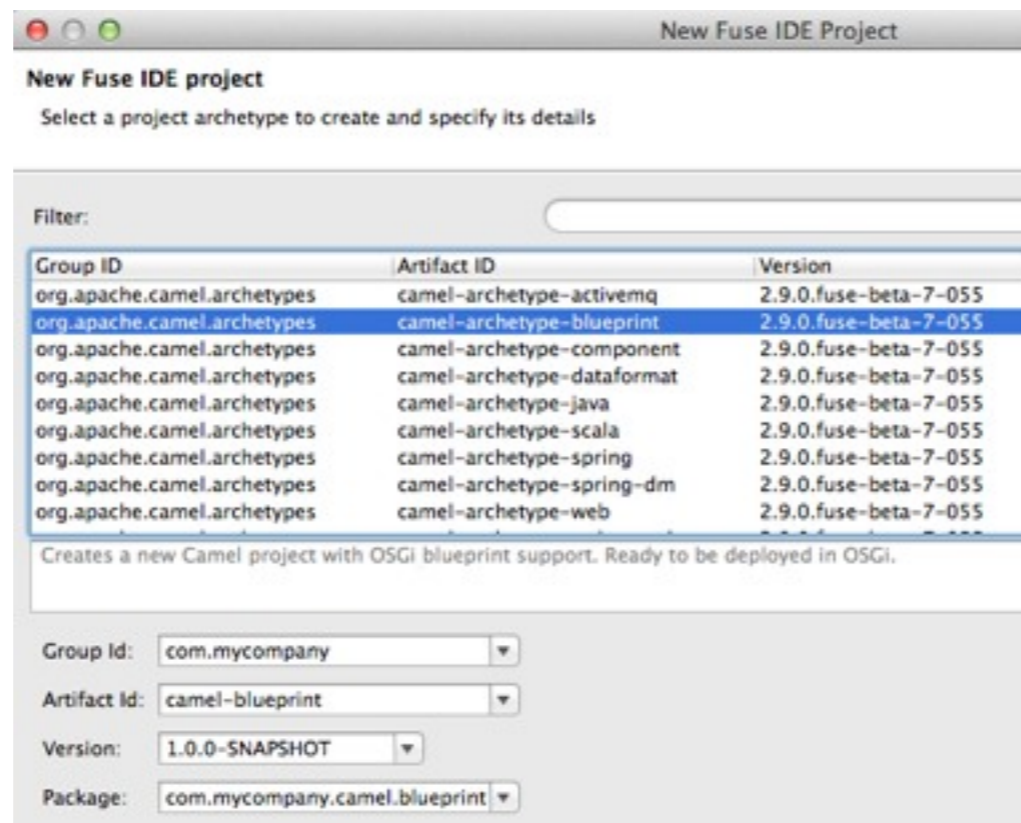
- What is Apache Camel?
- Basic Camel Concepts
- Running Camel
- Whats included in the box?
- **Creating new Camel Projects**
- Extending Camel
- Roadmap
- Q and A

Create New Project using Maven

- Create new Project using Maven

```
mvn archetype:generate
```

- Or from Fuse IDE and/or Eclipse



Create New Project using Maven

■ Maven Archetypes

camel-archetype-activemq
camel-archetype-blueprint
camel-archetype-component
camel-archetype-dataformat
camel-archetype-java
camel-archetype-scala
camel-archetype-spring
camel-archetype-spring-dm
camel-archetype-web
camel-archetype-webconsole

Create New Project using Maven

■ camel-archetype-blueprint

```
<blueprint xmlns="http://www.osgi.org/xmlns/blueprint/v1.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:camel="http://camel.apache.org/schema/blueprint"
  xsi:schemaLocation="
    http://www.osgi.org/xmlns/blueprint/v1.0.0 http://www.osgi.org/xmlns/blueprint/v1.0.0/blueprint.xsd
    http://camel.apache.org/schema/blueprint http://camel.apache.org/schema/blueprint/camel-blueprint.xsd">

  <bean id="helloBean" class="com.foo.HelloBean">
    <property name="say" value="Hi from Camel"/>
  </bean>

  <camelContext trace="false" id="blueprintContext" xmlns="http://camel.apache.org/schema/blueprint">
    <route id="timerToLog">
      <from uri="timer:foo?period=5000"/>
      <setBody>
        <method method="hello" ref="helloBean"></method>
      </setBody>
      <log message="The message contains ${body}"/>
      <to uri="mock:result"/>
    </route>
  </camelContext>

</blueprint>
```

```
mvn install
mvn camel:run
```

Create New Project using Maven

■ Testing with Camel

```
public class RouteTest extends CamelBlueprintTestSupport {  
  
    @Override  
    protected String getBlueprintDescriptor() {  
        return "/OSGI-INF/blueprint/blueprint.xml";  
    }  
  
    @Test  
    public void testRoute() throws Exception {  
        // the route is timer based, so every 5th second a message is send  
        // we should then expect at least one message  
        getMockEndpoint("mock:result").expectedMinimumMessageCount(1);  
  
        // assert expectations  
        assertMockEndpointsSatisfied();  
    }  
}
```

■ ... from inside Eclipse



Create New Project using Maven

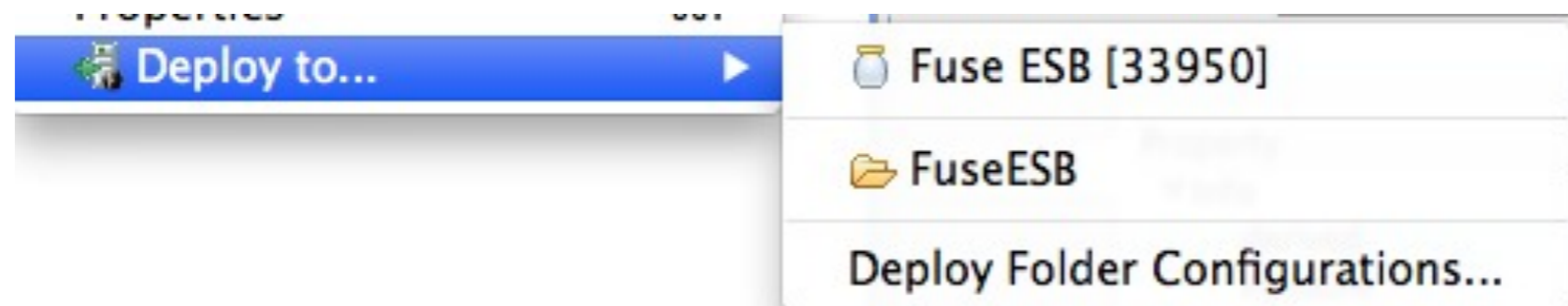
- Deploying to container
 - using Karaf Shell

```
FuseESB:karaf@root> osgi:install mvn:com.foo/blue
```

```
BundleID: 214
```

```
FuseESB:karaf@root> osgi:start 214
```

- Using Fuse IDE

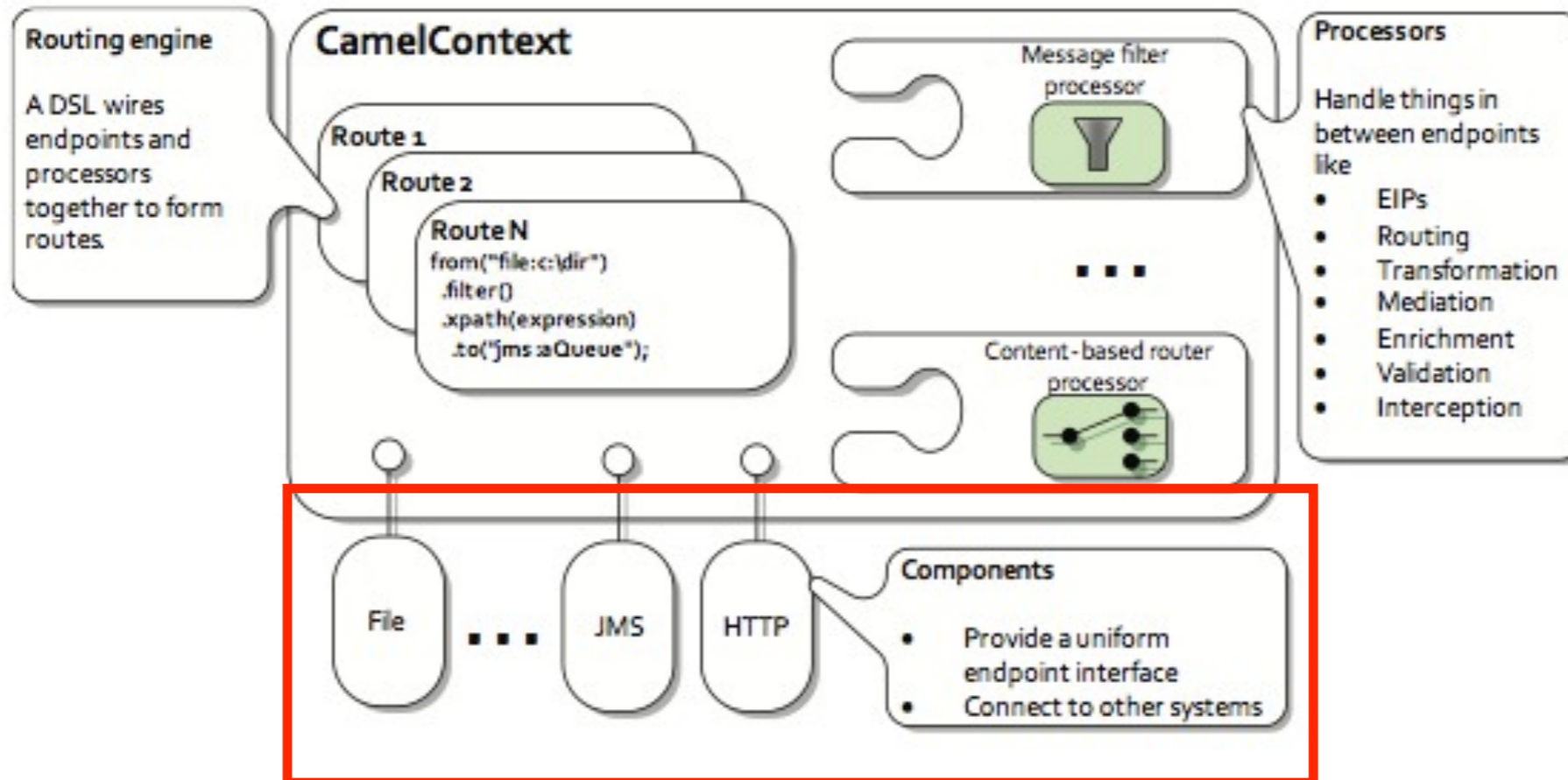


Agenda

- What is Apache Camel?
- Basic Camel Concepts
- Running Camel
- Whats included in the box?
- Creating new Camel Projects
- **Extending Camel**
- Roadmap
- Q and A

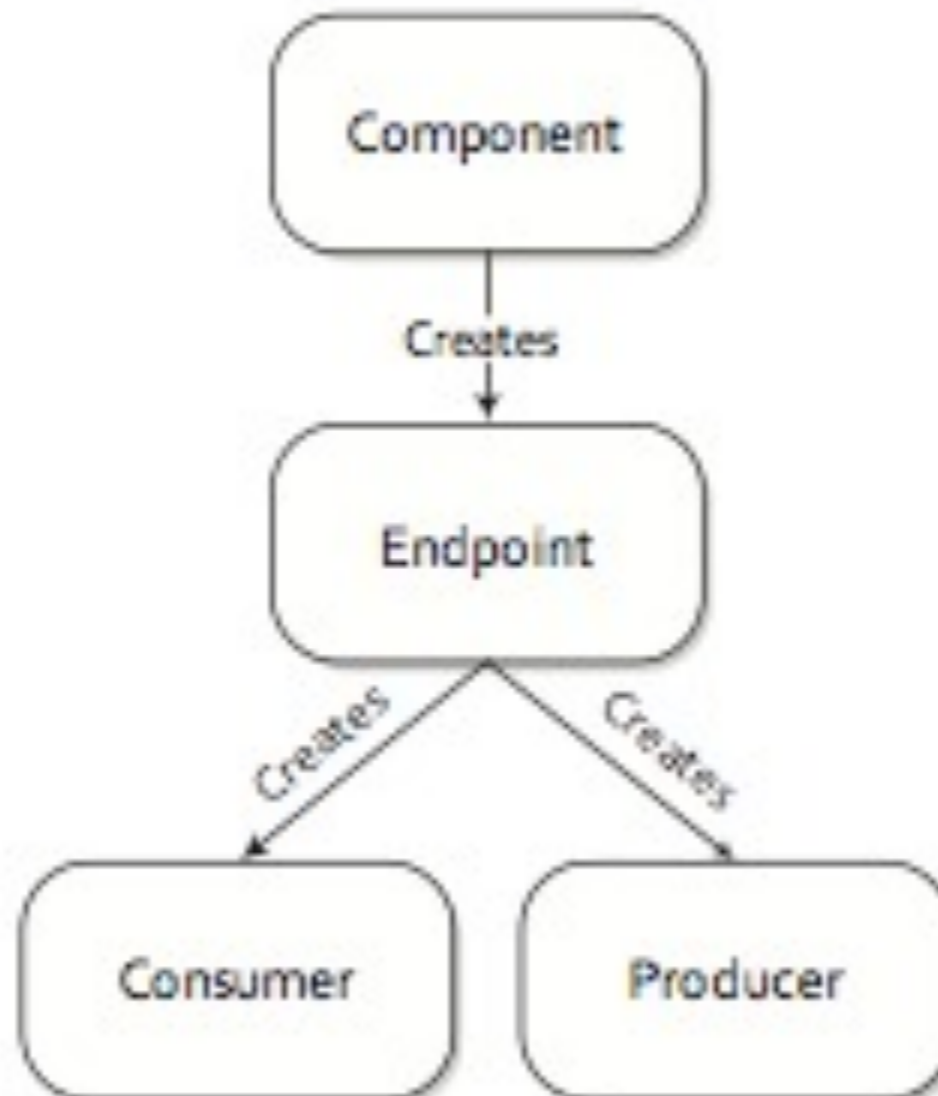
Extending Camel

■ Using Components



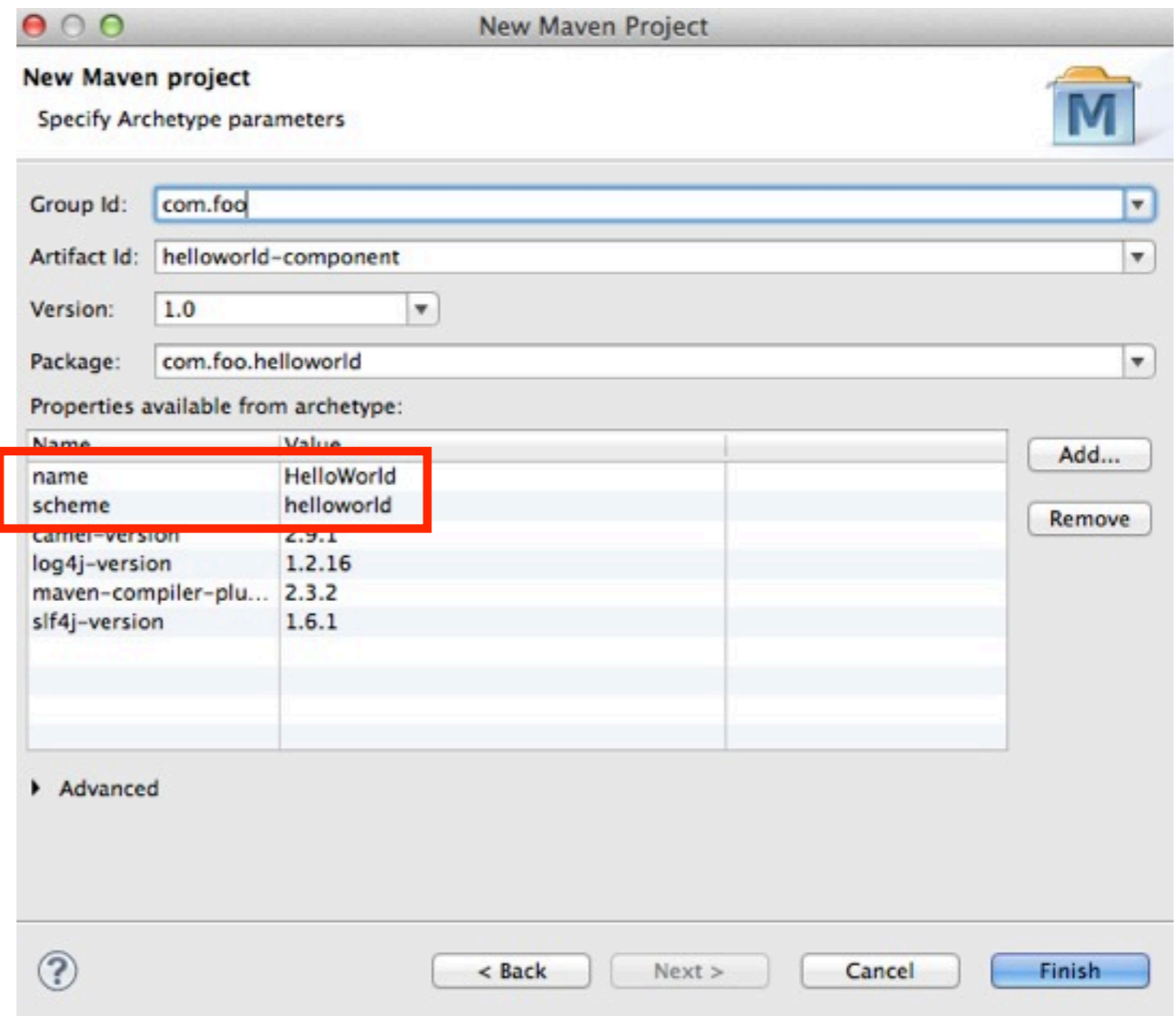
Extending Camel

- Dissecting a Component



Extending Camel

- Creating a new Component
- ... using Maven Tooling
 - mvn archetype:generate (camel-archetype-component)
- Or use Fuse IDE and/or Eclipse
- Specify name for
 - Component
 - URI Scheme

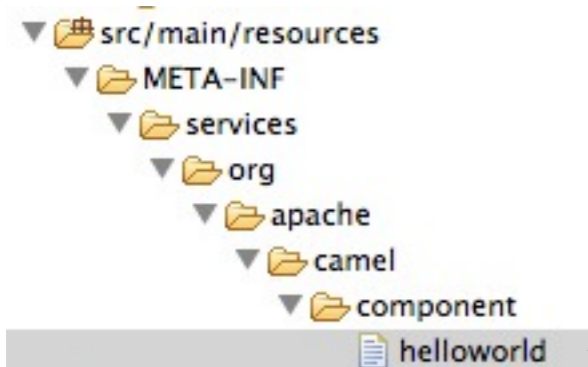


Extending Camel

- HelloWorldComponent

```
public class HelloWorldComponent extends DefaultComponent {  
    protected Endpoint createEndpoint(String uri, String remaining,  
                                     Map<String, Object> parameters) throws Exception {  
        Endpoint endpoint = new HelloWorldEndpoint(uri, this);  
        setProperties(endpoint, parameters);  
        return endpoint;  
    }  
}
```

- Auto discover component name



class=com.mycompany.camel.component.HelloWorldComponent

Extending Camel

■ HelloWorldEndpoint

```
public class HelloWorldEndpoint extends DefaultEndpoint {  
  
    private String greeting;  
    private int age;  
  
    public HelloWorldEndpoint(String uri, HelloWorldComponent component) {  
        super(uri, component);  
    }  
  
    public Producer createProducer() throws Exception {  
        return new HelloWorldProducer(this);  
    }  
  
    public Consumer createConsumer(Processor processor) throws Exception {  
        return new HelloWorldConsumer(this, processor);  
    }  
  
    public boolean isSingleton() {  
        return true;  
    }  
  
    public String getGreeting() {  
        return greeting;  
    }  
    public void setGreeting(String greeting) {  
        this.greeting = greeting;  
    }  
    public int getAge() {  
        return age;  
    }  
    public void setAge(int age) {  
        this.age = age;  
    }  
}
```

options as getter/setters



Extending Camel

- HelloWorldConsumer

```
public class HelloWorldConsumer extends ScheduledPollConsumer {  
    private final HelloWorldEndpoint endpoint;  
  
    public HelloWorldConsumer(HelloWorldEndpoint endpoint, Processor processor) {  
        super(endpoint, processor);  
        this.endpoint = endpoint;  
    }  
}
```

← scheduled consumer

```
@Override  
protected int poll() throws Exception {  
    // create new exchange  
    Exchange exchange = endpoint.createExchange();  
  
    // and set a fixed message body  
    exchange.getIn().setBody("Camel");  
  
    // send message to next processor in the route  
    getProcessor().process(exchange);  
  
    return 1; // number of messages polled  
}
```

← logic

Extending Camel

■ HelloWorldProducer

```
public class HelloWorldProducer extends DefaultProducer {  
    private static final transient Logger LOG = LoggerFactory.getLogger(HelloWorldProducer.class);  
    private HelloWorldEndpoint endpoint;  
  
    public HelloWorldProducer(HelloWorldEndpoint endpoint) {  
        super(endpoint);  
        this.endpoint = endpoint;  
    }  
}
```

```
public void process(Exchange exchange) throws Exception {  
    String greeting = endpoint.getGreeting();  
    String body = exchange.getIn().getBody(String.class);  
    int age = endpoint.getAge();  
  
    String text = String.format("%s %s are you %s years?", greeting, body, age);  
    System.out.println(text);  
}
```

↑
logic

Extending Camel

- HelloWorldComponentTest

```
public class HelloWorldComponentTest extends CamelTestSupport {

    @Test
    public void testHelloWorld() throws Exception {
        MockEndpoint mock = getMockEndpoint("mock:result");
        mock.expectedMinimumMessageCount(1);

        assertMockEndpointsSatisfied();
    }

    @Override
    protected RouteBuilder createRouteBuilder() throws Exception {
        return new RouteBuilder() {
            public void configure() {
                from("helloworld://foo")
                    .to("helloworld://bar?greeting=Hello&age=5")
                    .to("mock:result");
            }
        };
    }
}
```

Agenda

- What is Apache Camel?
- Basic Camel Concepts
- Running Camel
- Whats included in the box?
- Creating new Camel Projects
- Extending Camel
- **Roadmap**
- Q and A

Camel Roadmap

- Apache Camel 2.10 (ETA june 2012)
 - Java 7 supported
 - Spring 3.1 supported
 - OSGi Blueprint test kit
 - JMX management name can be locked down
 - Maven archetypes overall improved



Camel Roadmap

- Apache Camel 2.10 (ETA june 2012) cont.
 - CDI Dependency Injection supported
 - File/ftp consumer can bridge exceptions with Camel error handler
 - Splitter EIP can split in groups (chunks)
 - 12 new components, among:
 - dynamoDB for AWS
 - MongoDB
 - SSH
 - Twitter
 - WebSocket



Camel Roadmap

- Apache Camel 3.0
(ETA end of 2012/start of 2013)
 - API for component writers
 - Tighten up DSL
 - Optimize routing engine
 - Enriched runtime routing model
 - Hot add/remove interceptors, et all. to running routes



Roadmap subject for changes!!!

Camel Roadmap

- Apache Camel 3.0
(ETA end of 2012/start of 2013) cont.
 - Overhauled Message History EIP
 - Tooling support for validating endpoints
 - Simpler JMS component (just JMS API, no Spring API)
 - REST API for management
 - Improved Groovy DSL



Roadmap subject for changes!!!

Agenda

- What is Apache Camel?
- Basic Camel Concepts
- Running Camel
- Whats included in the box?
- Creating new Camel Projects
- Extending Camel
- Roadmap
- **Q and A**

More Information

- Where do I get more information?
 - Apache Camel Website
<http://camel.apache.org>

- **Overview**
- Home
- Download
- Getting Started
- FAQ

Getting Started


First you need to **Download** the Camel distribution; or you could grab the **Source** and try **Building** it yourself.



Great introduction article

The best introduction to Camel is in fact an article posted at dzone. We strongly suggest you go read the article: [Apache Camel: Integration Nirvana](#) great article to learn what Camel is and has a good use case example.

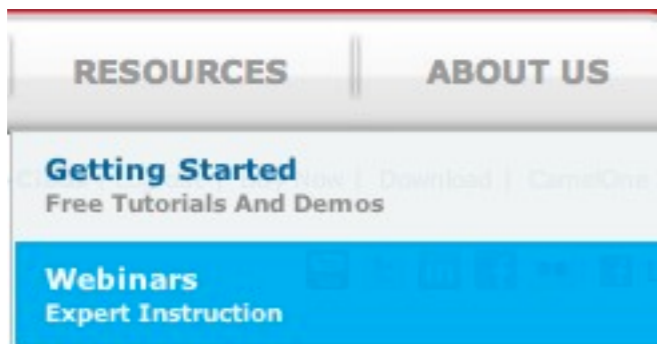
More Information

- Where do I get more information?
 - Join the Mailing list
<http://camel.apache.org/mailling-lists.html>
 - Try the IRC-room
<http://camel.apache.org/irc-room.html>
 - Use the Search Box

 - Checkout the Camel Articles
<http://camel.apache.org/articles.html>
 - Run Camel Examples
<http://camel.apache.org/examples.html>

More Information

- Where do I get more information?
 - FuseSource - One Stop for all docs
<http://fusesource.com/documentation/>
 - Specific Guides
 - Mirror Apache Documentation
 - Versioned Documentation
 - Java Doc
 - XML Schema

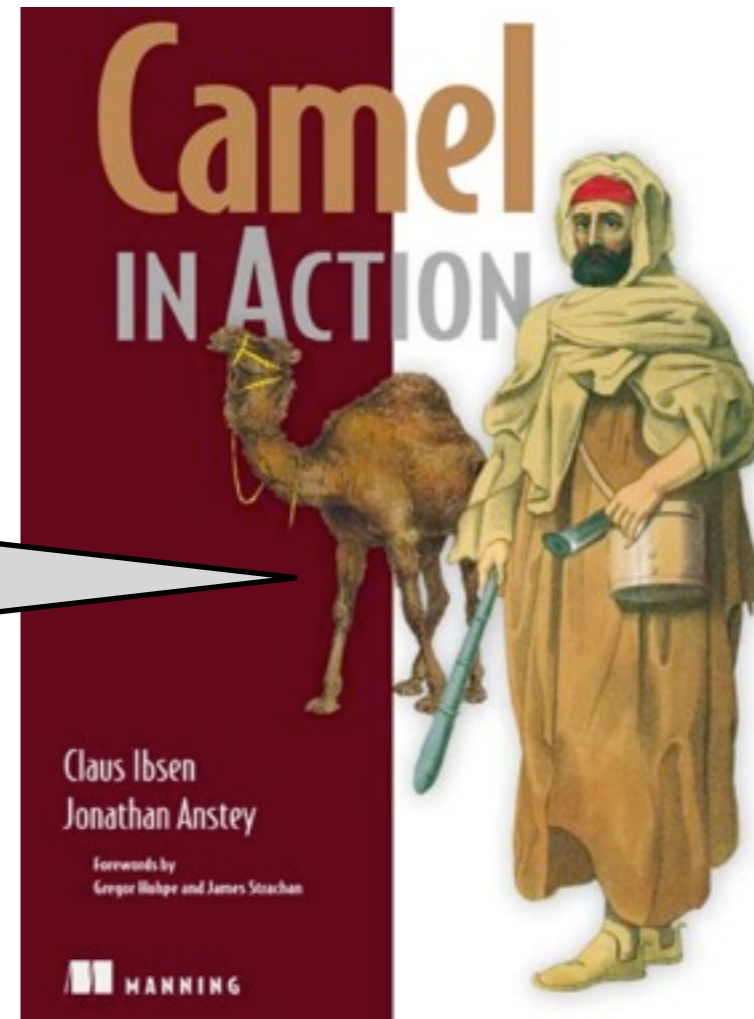
- FuseSource - Webinars



More Information

- Where do I get more information?
 - Read the book
(Chapter 1 is free)

*Collectors Edition
with standing Camel*



<http://manning.com/ibsen>



- Twitter: @davsclaus
- Blog: <http://davsclaus.blogspot.com>
- Email: cibsen@fusesource.com