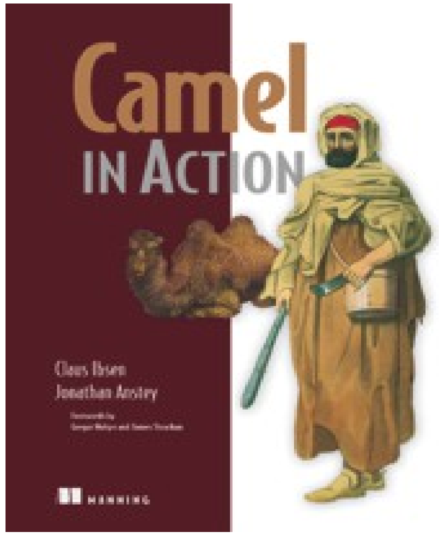
CamelOne 2013
June 10-11 2013
Boston, MA

Using Apache Camel Connectors for External Connectivity

Your Speaker

- Principal Software Engineer at Red Hat
- Apache Camel
  - 5 years working with Camel
- Author of Camel in Action book
- Contact
  - EMail: cibsen@redhat.com
  - Twitter: @davsclaus
  - Blog: http://davsclaus.com
  - Linkedin: http://www.linkedin.com/in/davsclaus



Agenda

- A little Example
- Understanding Components
- Essential Components
- Creating new Components
- Q and A

A Little Example

- File Copier Example

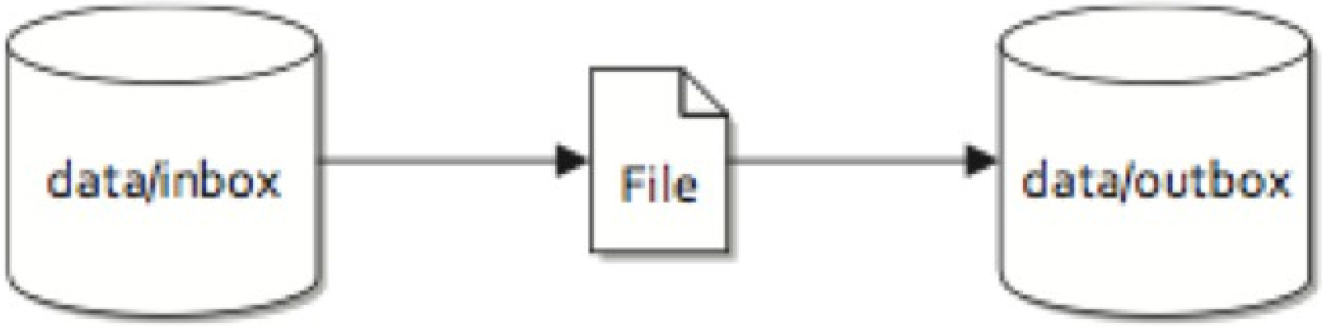
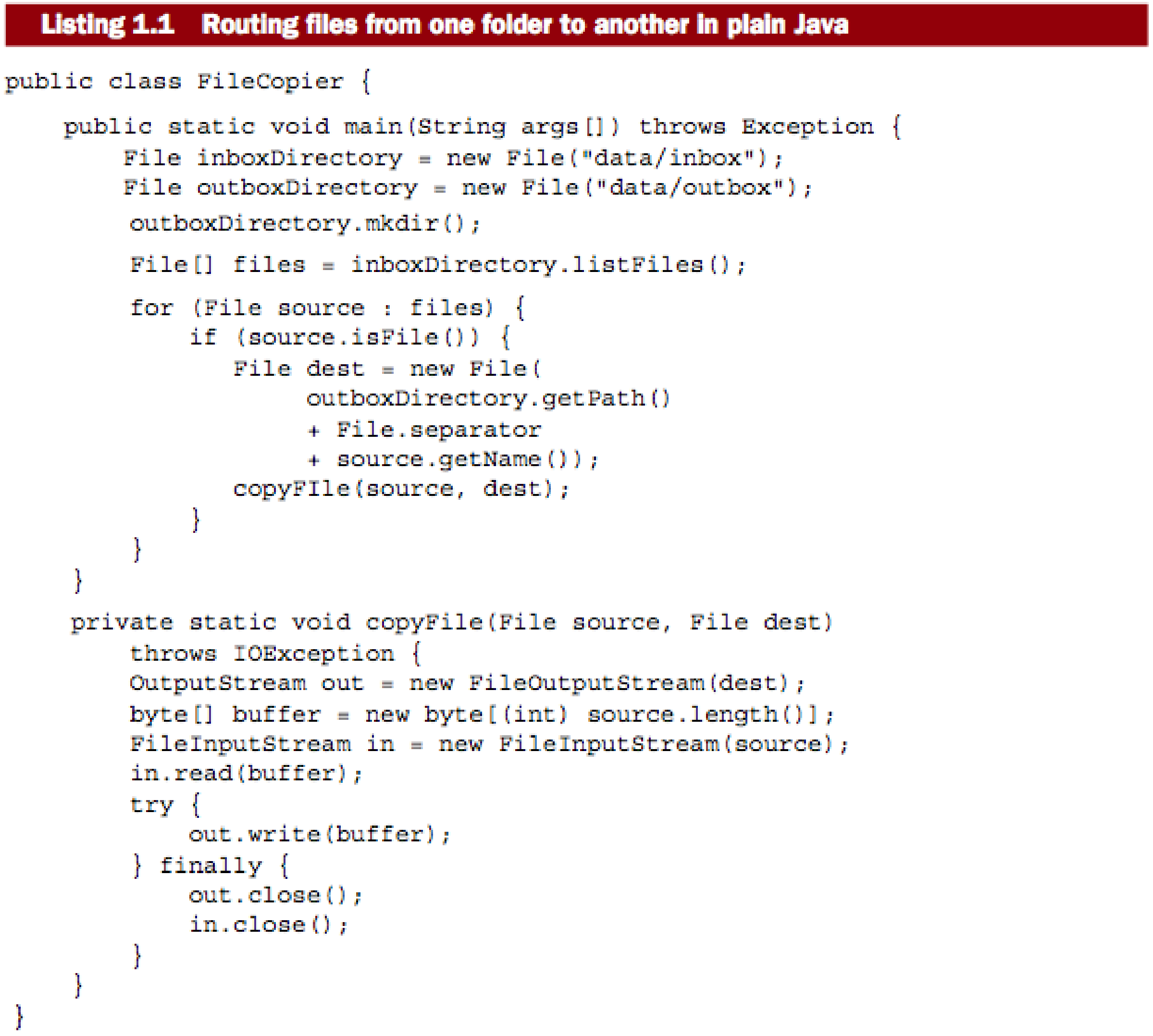


Figure 1.2 **Files are routed from the data/inbox directory to the data/outbox directory.**

A Little Example

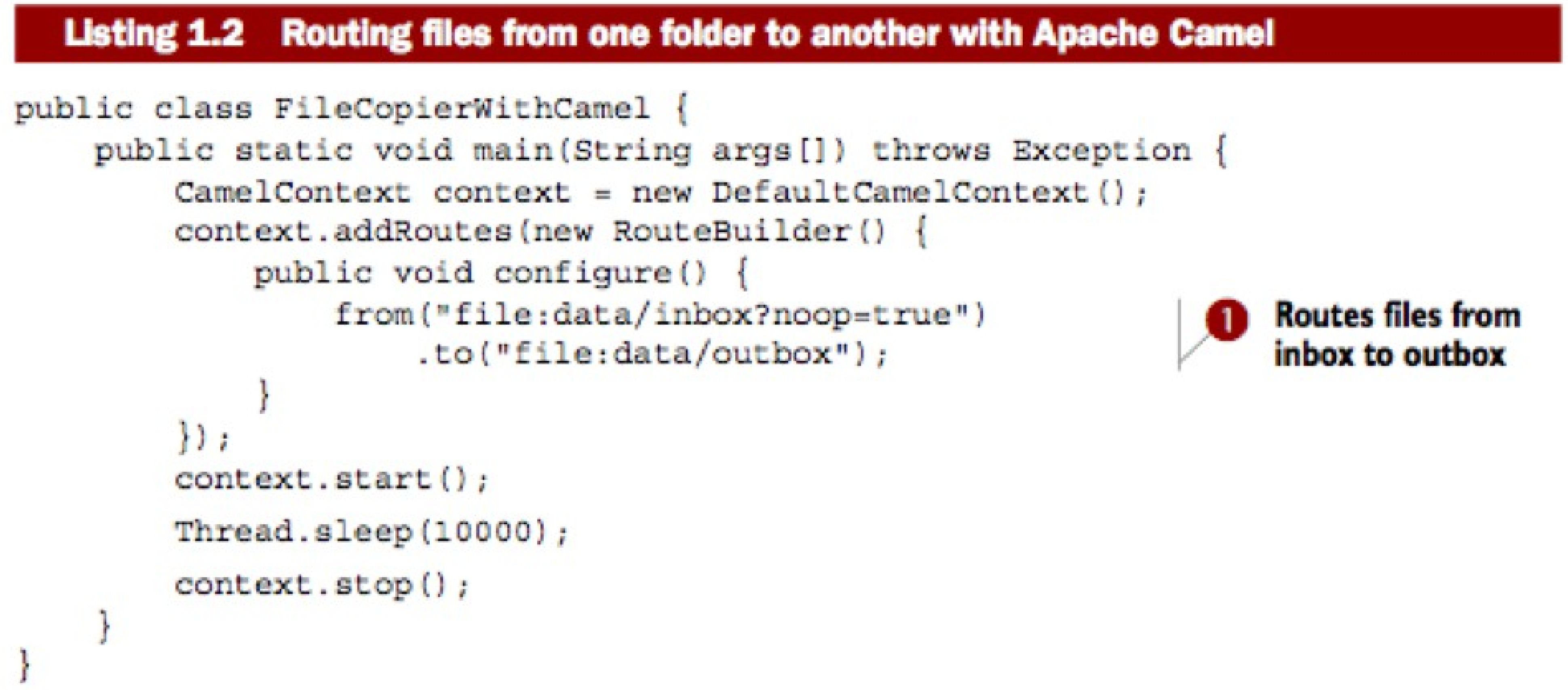


Listing 1.1 Routing files from one folder to another in plain Java

```java
public class FileCopier {

    public static void main(String args[]) throws Exception {
        File inboxDirectory = new File("data/inbox");
        File outboxDirectory = new File("data/outbox");
        outboxDirectory.mkdir();
        File[] files = inboxDirectory.listFiles();
        for (File source : files) {
            if (source.isFile()) {
               File dest = new File(
                    outboxDirectory.getPath()
                    + File.separator
                    + source.getName());
               copyFIle(source, dest);
            }
        }
    }
    private static void copyFile(File source, File dest)
        throws IOException {
        OutputStream out = new FileOutputStream(dest);
        byte[] buffer = new byte[(int) source.length()];
        FileInputStream in = new FileInputStream(source);
        in.read(buffer);
        try {
            out.write(buffer);
        } finally {
            out.close();
            in.close();
        }
    }
}
```

A Little Example

- File Copier Example



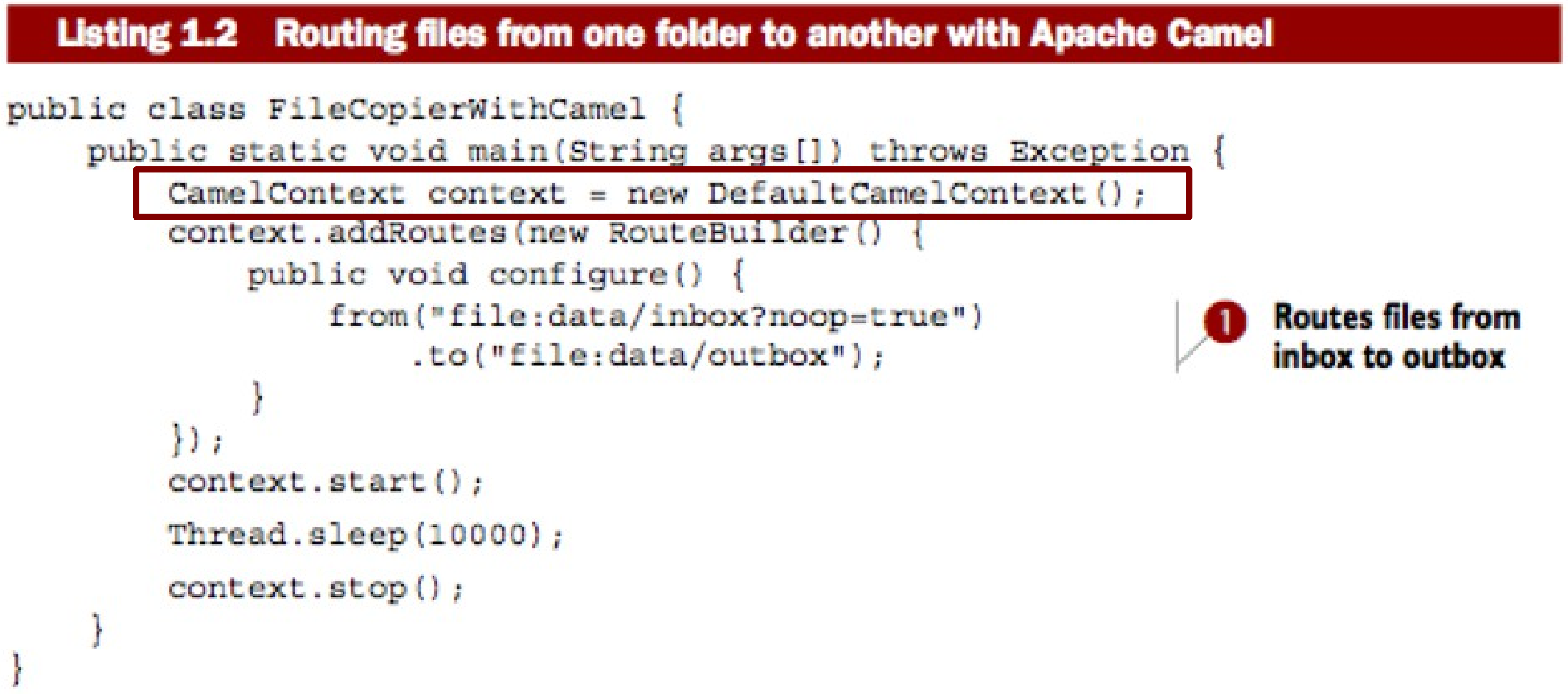
Listing 1.2 Routing files from one folder to another with Apache Camel

```java
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

A Little Example

- File Copier Example



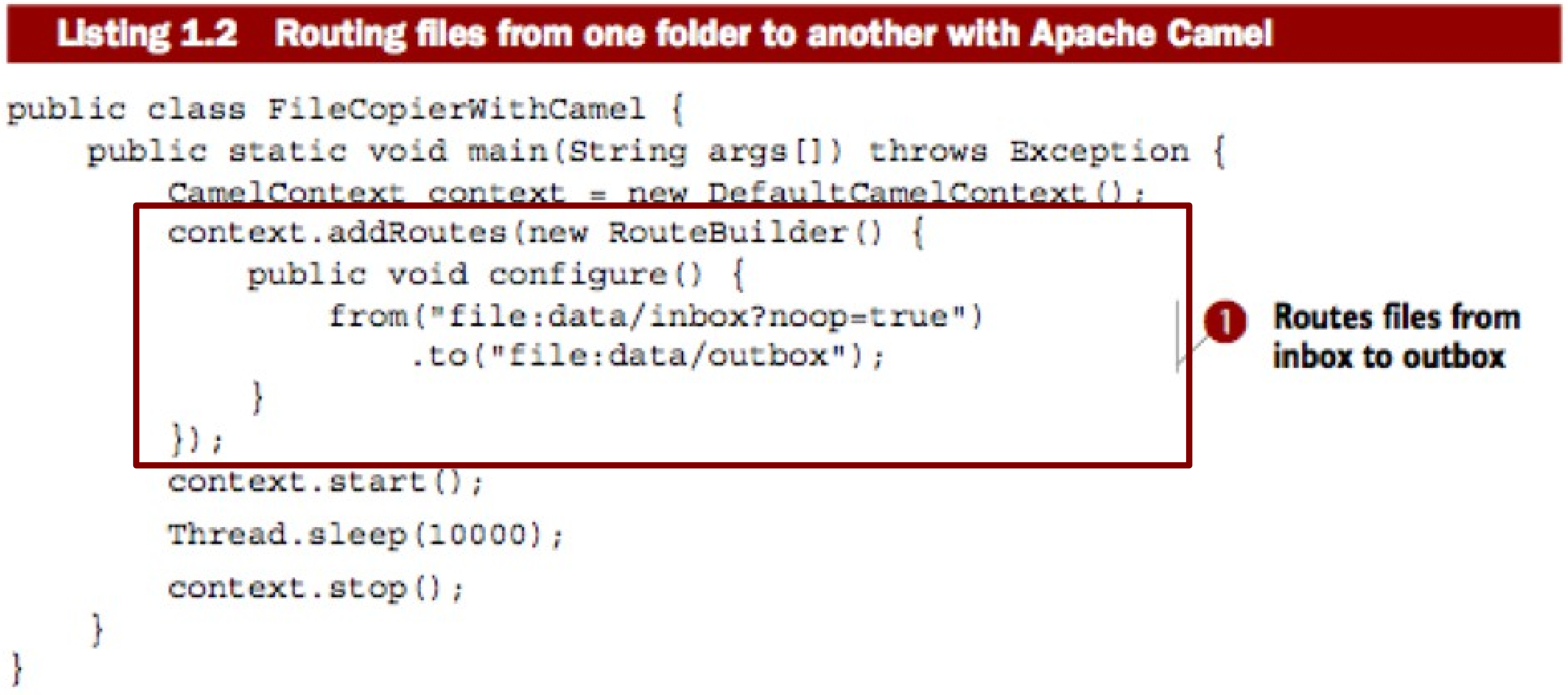
Listing 1.2 Routing files from one folder to another with Apache Camel

```
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

A Little Example

- File Copier Example



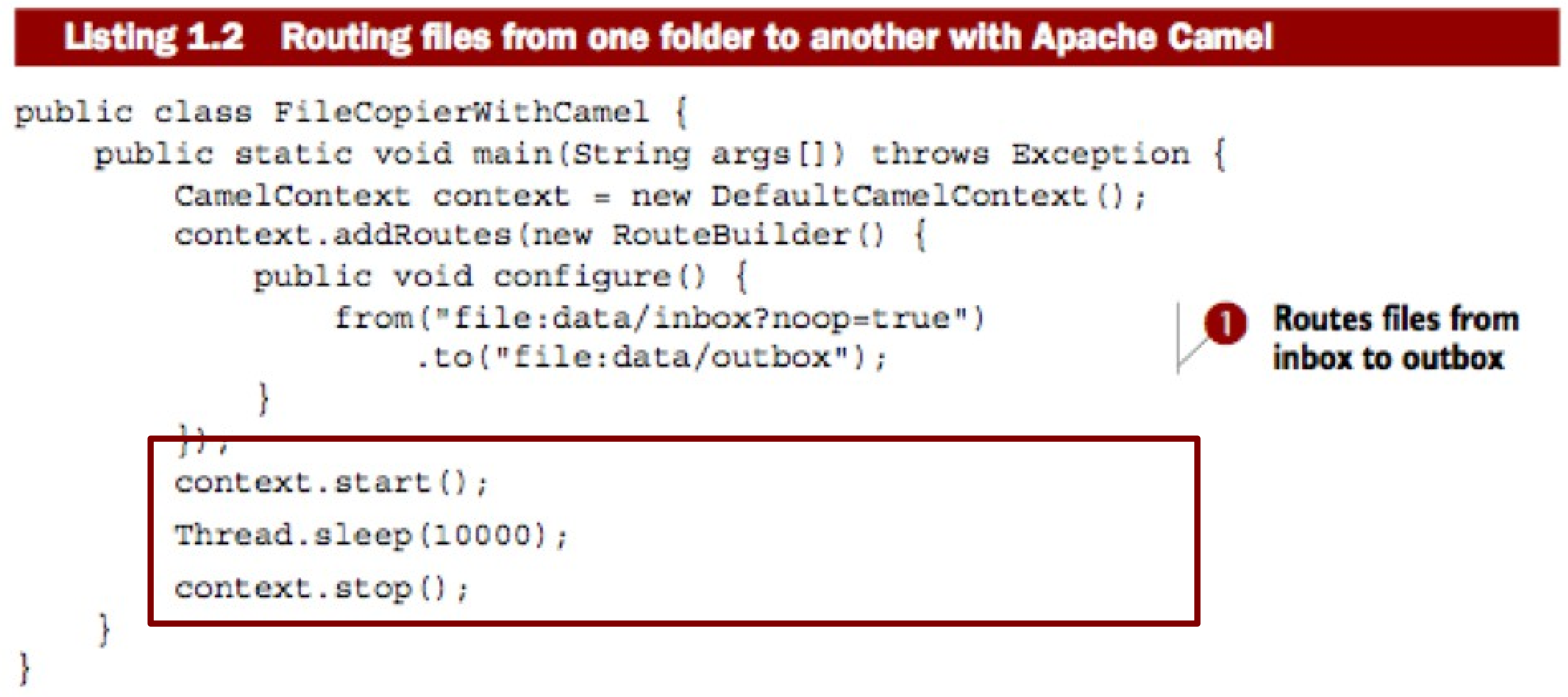
Listing 1.2 Routing files from one folder to another with Apache Camel

```
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

A Little Example

- File Copier Example



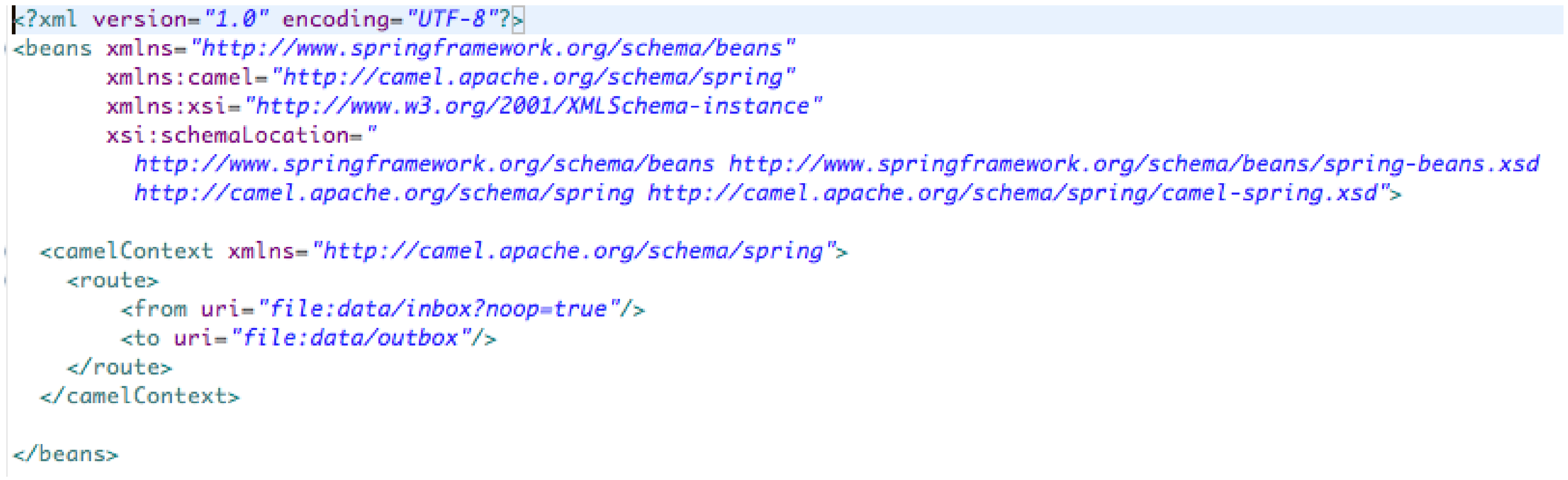
Listing 1.2 Routing files from one folder to another with Apache Camel

```
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

A Little Example

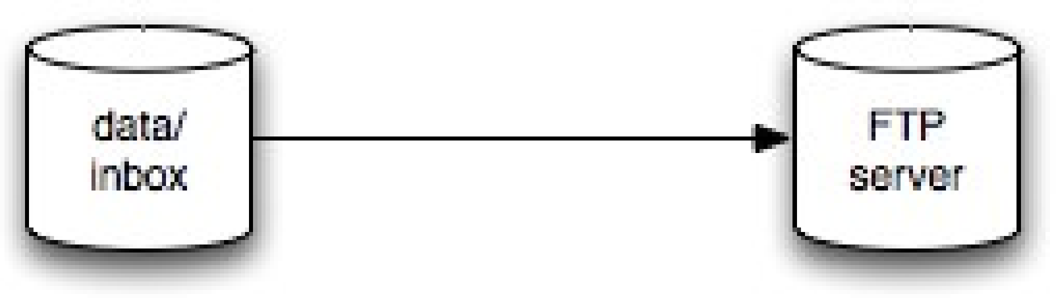
- File Copier Example (in XML)



```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:camel="http://camel.apache.org/schema/spring"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="
         http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
         http://camel.apache.org/schema/spring http://camel.apache.org/schema/spring/camel-spring.xsd">

  <camelContext xmlns="http://camel.apache.org/schema/spring">
    <route>
        <from uri="file:data/inbox?noop=true"/>
        <to uri="file:data/outbox"/>
    </route>
  </camelContext>

</beans>
```

A Little Example

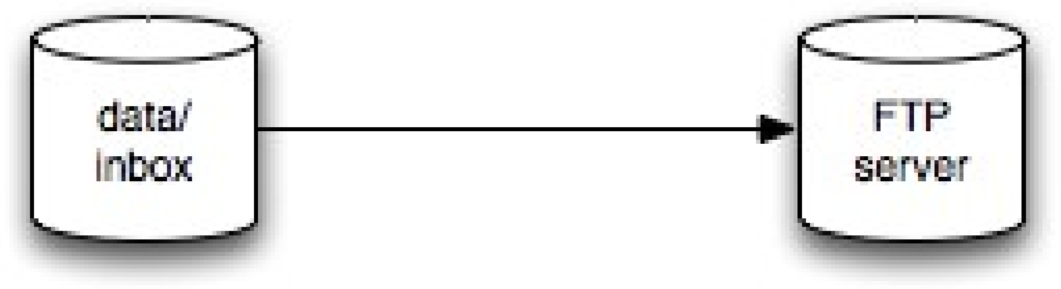
- File to FTP Example



Files is coped from data/inbox to a remote FTP server

A Little Example

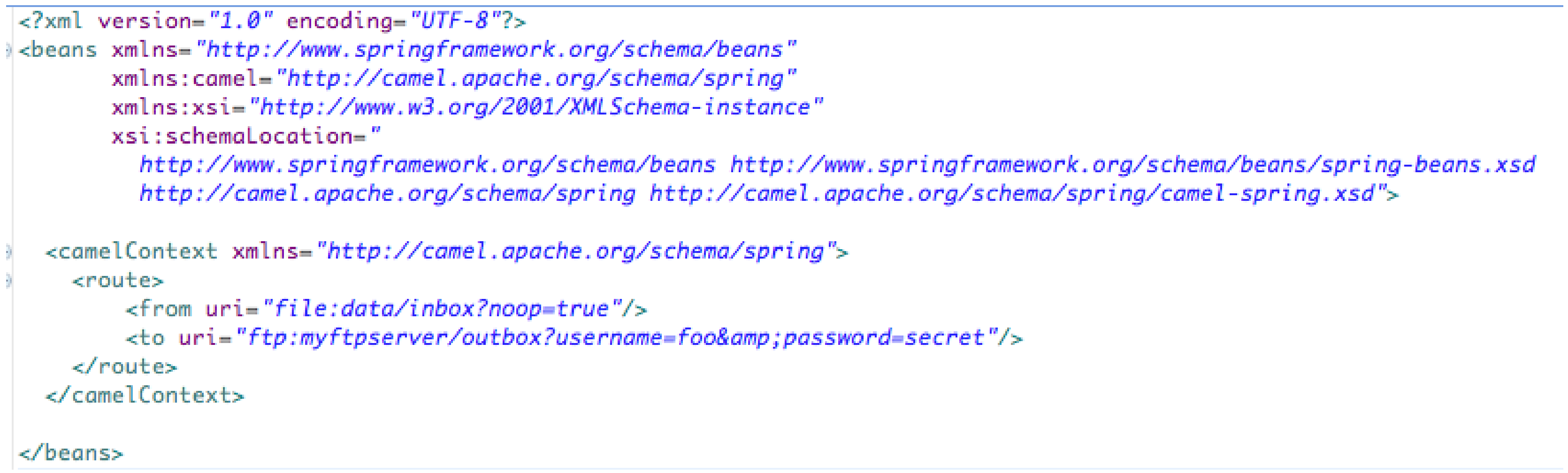
- File to FTP Example



How to write this in pure Java code ???

A Little Example

- File to FTP Example

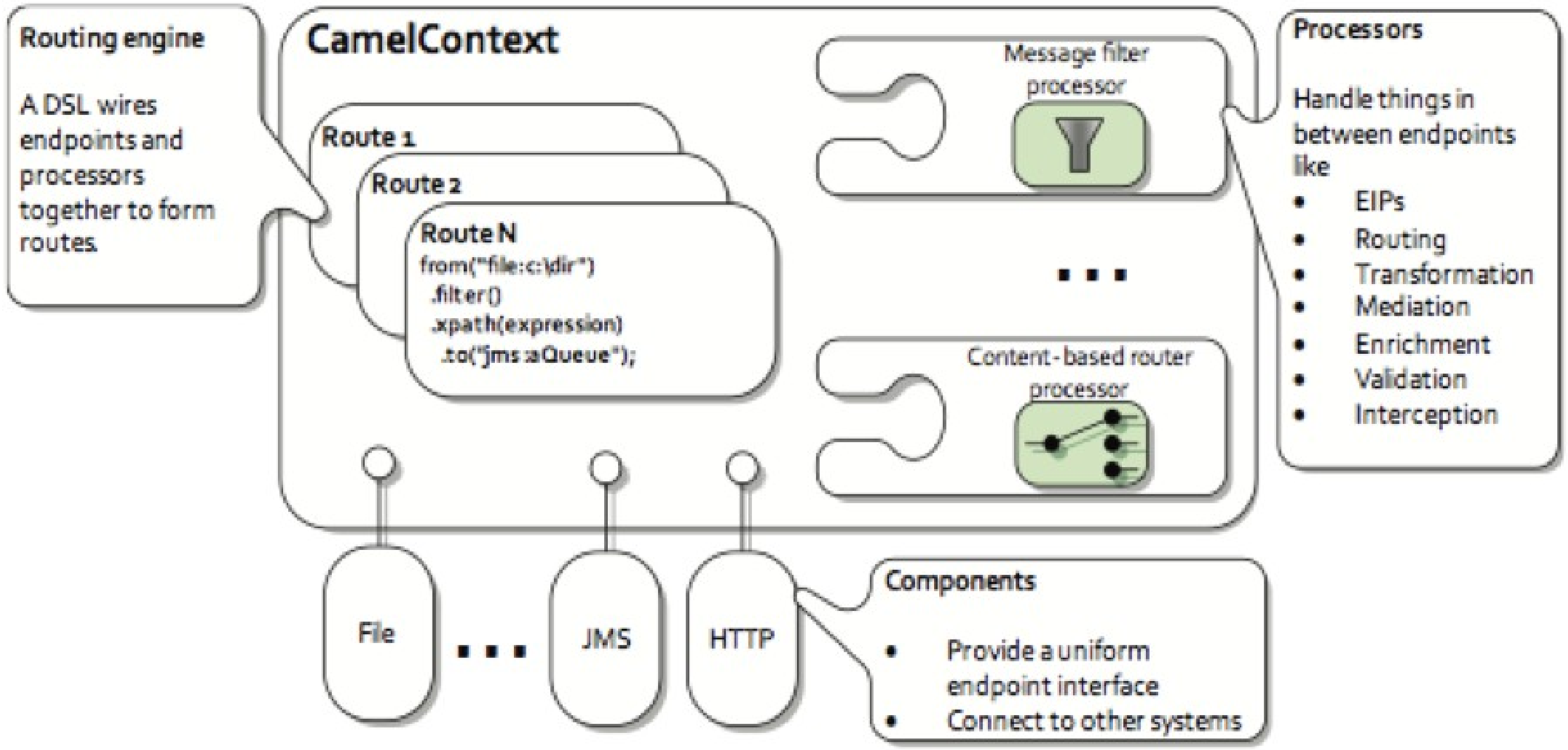


```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:camel="http://camel.apache.org/schema/spring"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="
         http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
         http://camel.apache.org/schema/spring http://camel.apache.org/schema/spring/camel-spring.xsd">

  <camelContext xmlns="http://camel.apache.org/schema/spring">
    <route>
        <from uri="file:data/inbox?noop=true"/>
        <to uri="ftp:myftpserver/outbox?username=foo&amp;password=secret"/>
    </route>
  </camelContext>

</beans>
```

Easy with Camel
Just use FTP component

A Little Example

- Camel's Architecture



A Little Example

120+ Components

activemq	cxf	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

A Little Example

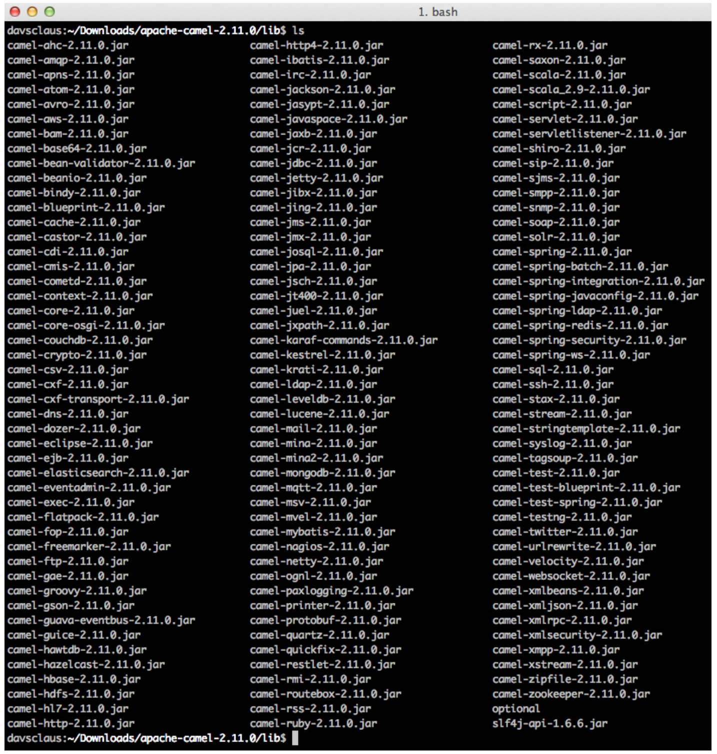
120+ Components

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

A Little Example

In fact we have 139 in latest release ...

```
davsclaus:~/Downloads/apache-camel-2.11.0/lib$ ls camel* | wc -l
    139
```

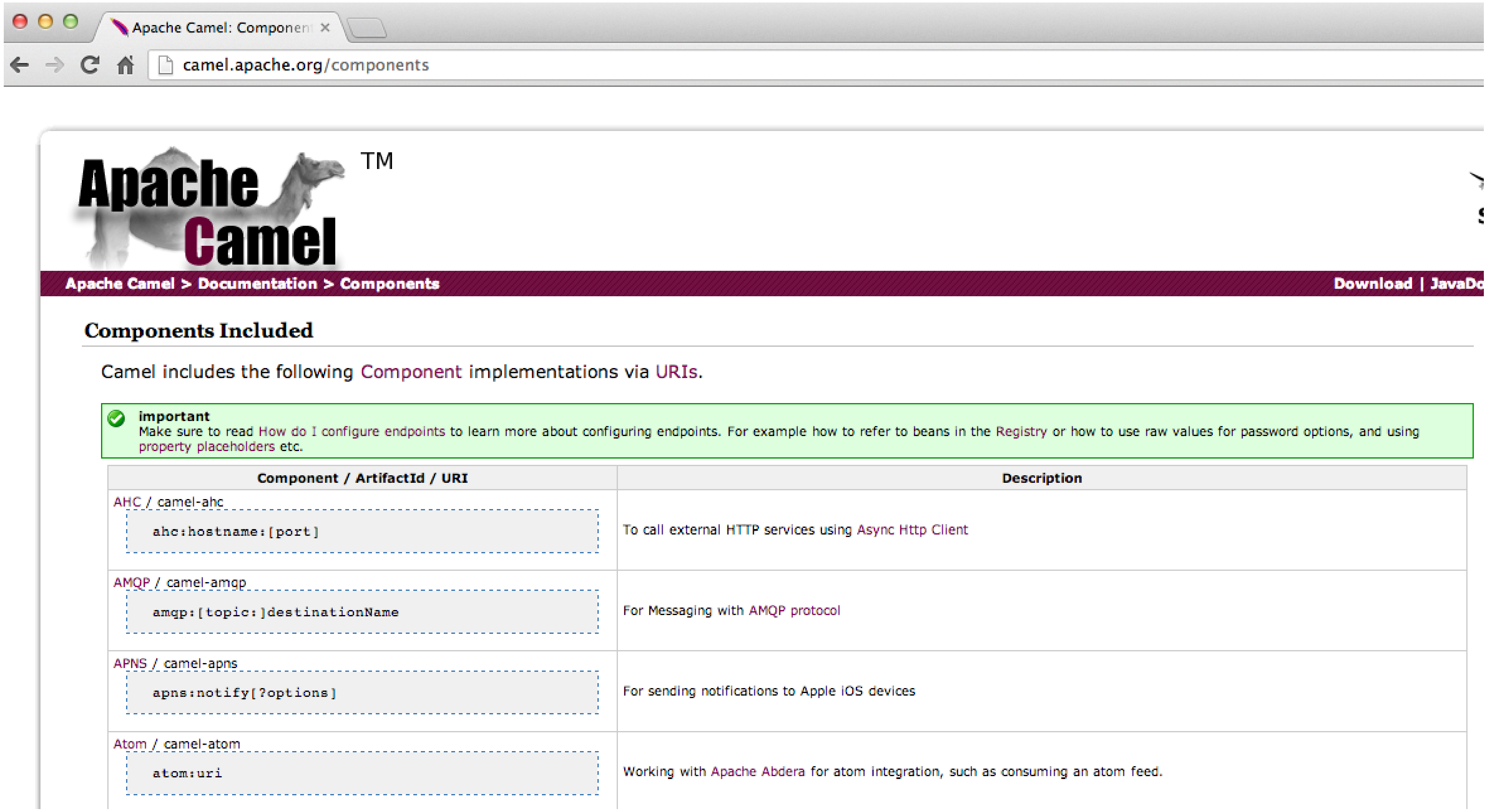


1. bash

```
davsclaus:~/Downloads/apache-camel-2.11.0/lib$ ls
camel-ahc-2.11.0.jar                    camel-http4-2.11.0.jar                  camel-rx-2.11.0.jar
camel-amqp-2.11.0.jar                   camel-ibatis-2.11.0.jar                 camel-saxon-2.11.0.jar
camel-apns-2.11.0.jar                   camel-irc-2.11.0.jar                    camel-scala-2.11.0.jar
camel-atom-2.11.0.jar                   camel-jackson-2.11.0.jar                camel-scala_2.9-2.11.0.jar
camel-avro-2.11.0.jar                   camel-jasypt-2.11.0.jar                 camel-script-2.11.0.jar
camel-aws-2.11.0.jar                    camel-javaspace-2.11.0.jar              camel-servlet-2.11.0.jar
camel-bam-2.11.0.jar                    camel-jaxb-2.11.0.jar                   camel-servletlistener-2.11.0.jar
camel-base64-2.11.0.jar                 camel-jcr-2.11.0.jar                    camel-shiro-2.11.0.jar
camel-bean-validator-2.11.0.jar         camel-jdbc-2.11.0.jar                   camel-sip-2.11.0.jar
camel-beanio-2.11.0.jar                 camel-jetty-2.11.0.jar                  camel-sjms-2.11.0.jar
camel-bindy-2.11.0.jar                  camel-jibx-2.11.0.jar                   camel-smpp-2.11.0.jar
camel-blueprint-2.11.0.jar              camel-jing-2.11.0.jar                   camel-snmp-2.11.0.jar
camel-cache-2.11.0.jar                  camel-jms-2.11.0.jar                    camel-soap-2.11.0.jar
camel-castor-2.11.0.jar                 camel-jmx-2.11.0.jar                    camel-solr-2.11.0.jar
camel-cdi-2.11.0.jar                    camel-josql-2.11.0.jar                  camel-spring-2.11.0.jar
camel-cmis-2.11.0.jar                   camel-jpa-2.11.0.jar                    camel-spring-batch-2.11.0.jar
camel-cometd-2.11.0.jar                 camel-jsch-2.11.0.jar                   camel-spring-integration-2.11.0.jar
camel-context-2.11.0.jar                camel-jt400-2.11.0.jar                  camel-spring-javaconfig-2.11.0.jar
camel-core-2.11.0.jar                   camel-juel-2.11.0.jar                   camel-spring-ldap-2.11.0.jar
camel-core-osgi-2.11.0.jar              camel-jxpath-2.11.0.jar                 camel-spring-redis-2.11.0.jar
camel-couchdb-2.11.0.jar                camel-karaf-commands-2.11.0.jar         camel-spring-security-2.11.0.jar
camel-crypto-2.11.0.jar                 camel-kestrel-2.11.0.jar                camel-spring-ws-2.11.0.jar
camel-csv-2.11.0.jar                    camel-krati-2.11.0.jar                  camel-sql-2.11.0.jar
camel-cxf-2.11.0.jar                    camel-ldap-2.11.0.jar                   camel-ssh-2.11.0.jar
camel-cxf-transport-2.11.0.jar          camel-leveldb-2.11.0.jar                camel-stax-2.11.0.jar
camel-dns-2.11.0.jar                    camel-lucene-2.11.0.jar                 camel-stream-2.11.0.jar
camel-dozer-2.11.0.jar                  camel-mail-2.11.0.jar                   camel-stringtemplate-2.11.0.jar
camel-eclipse-2.11.0.jar                camel-mina-2.11.0.jar                   camel-syslog-2.11.0.jar
camel-ejb-2.11.0.jar                    camel-mina2-2.11.0.jar                  camel-tagsoup-2.11.0.jar
camel-elasticsearch-2.11.0.jar          camel-mongodb-2.11.0.jar                camel-test-2.11.0.jar
camel-eventadmin-2.11.0.jar             camel-mqtt-2.11.0.jar                   camel-test-blueprint-2.11.0.jar
camel-exec-2.11.0.jar                   camel-msv-2.11.0.jar                    camel-test-spring-2.11.0.jar
camel-flatpack-2.11.0.jar               camel-mvel-2.11.0.jar                   camel-testng-2.11.0.jar
camel-fop-2.11.0.jar                    camel-mybatis-2.11.0.jar                camel-twitter-2.11.0.jar
camel-freemarker-2.11.0.jar             camel-nagios-2.11.0.jar                 camel-urlrewrite-2.11.0.jar
camel-ftp-2.11.0.jar                    camel-netty-2.11.0.jar                  camel-velocity-2.11.0.jar
camel-gae-2.11.0.jar                    camel-ognl-2.11.0.jar                   camel-websocket-2.11.0.jar
camel-groovy-2.11.0.jar                 camel-paxlogging-2.11.0.jar             camel-xmlbeans-2.11.0.jar
camel-gson-2.11.0.jar                   camel-printer-2.11.0.jar                camel-xmljson-2.11.0.jar
camel-guava-eventbus-2.11.0.jar         camel-protobuf-2.11.0.jar               camel-xmlrpc-2.11.0.jar
camel-guice-2.11.0.jar                  camel-quartz-2.11.0.jar                 camel-xmlsecurity-2.11.0.jar
camel-hawtdb-2.11.0.jar                 camel-quickfix-2.11.0.jar               camel-xmpp-2.11.0.jar
camel-hazelcast-2.11.0.jar              camel-restlet-2.11.0.jar                camel-xstream-2.11.0.jar
camel-hbase-2.11.0.jar                  camel-rmi-2.11.0.jar                    camel-zipfile-2.11.0.jar
camel-hdfs-2.11.0.jar                   camel-routebox-2.11.0.jar               camel-zookeeper-2.11.0.jar
camel-hl7-2.11.0.jar                    camel-rss-2.11.0.jar                    optional
camel-http-2.11.0.jar                   camel-ruby-2.11.0.jar                   slf4j-api-1.6.6.jar
davsclaus:~/Downloads/apache-camel-2.11.0/lib$
```

A Little Example

… All components on website



Apache Camel: Componen

camel.apache.org/components

Apache Camel > Documentation > Components

Download | JavaDo

Components Included

Camel includes the following Component implementations via URIs.

important
Make sure to read How do I configure endpoints to learn more about configuring endpoints. For example how to refer to beans in the Registry or how to use raw values for password options, and using property placeholders etc.

Component / ArtifactId / URI	Description
AHC / camel-ahc `ahc:hostname:[port]`	To call external HTTP services using Async Http Client
AMQP / camel-amqp `amqp:[topic:]destinationName`	For Messaging with AMQP protocol
APNS / camel-apns `apns:notify[?options]`	For sending notifications to Apple iOS devices
Atom / camel-atom `atom:uri`	Working with Apache Abdera for atom integration, such as consuming an atom feed.

A Little Example

- Summary
  - Components for connectivity
  - Camel routes with components and EIPs
  - Components easy to configure
  - A lot of components
  - Very composeable
  - Learn Once – Can use 'em All

Agenda

- A little Example
- **Understanding Components**
- Essential Components
- Creating new Components
- Q and A

Understanding Components

- Facilitate messaging for connectivity

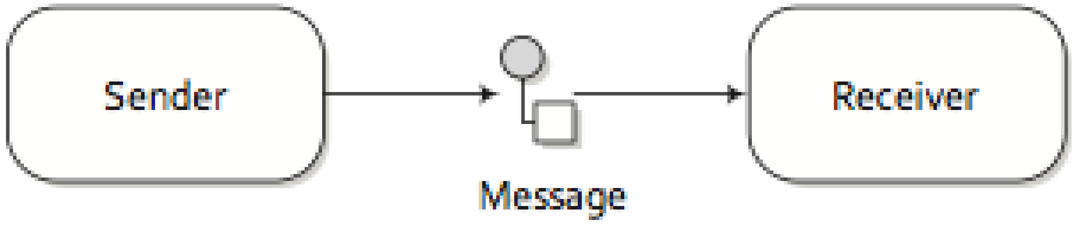


Figure 1.3 **Messages are entities used to send data from one system to another.**

Understanding Components

- Facilitate messaging for connectivity

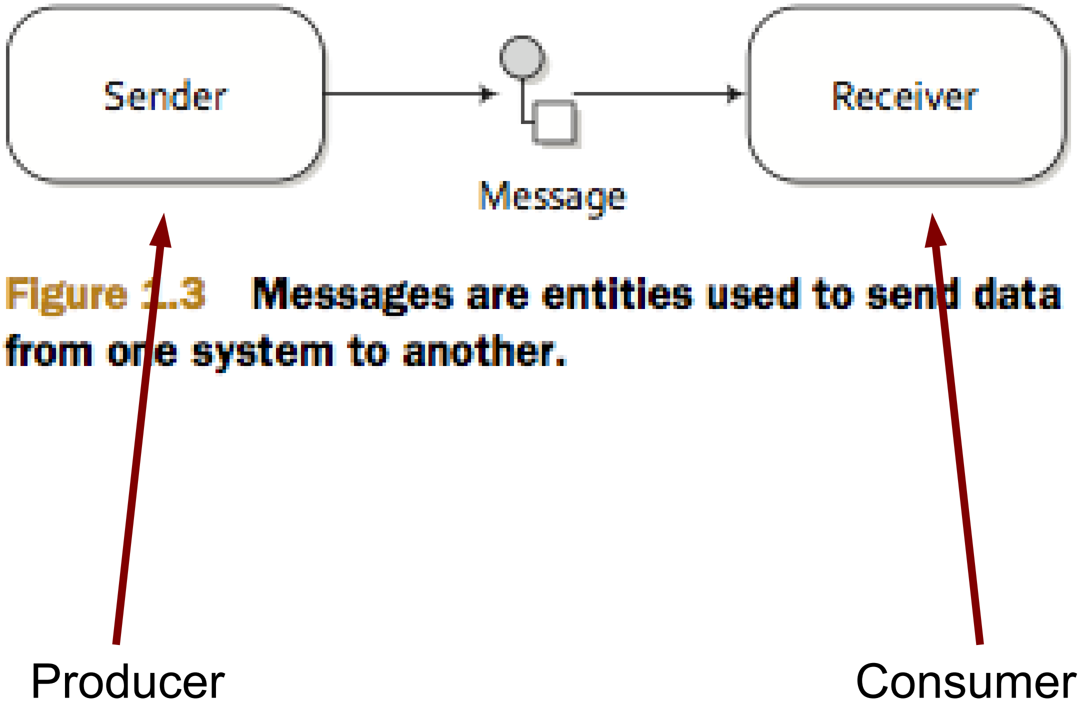


Figure 1.3 Messages are entities used to send data from one system to another.

Understanding Components

- … using endpoints via message channels

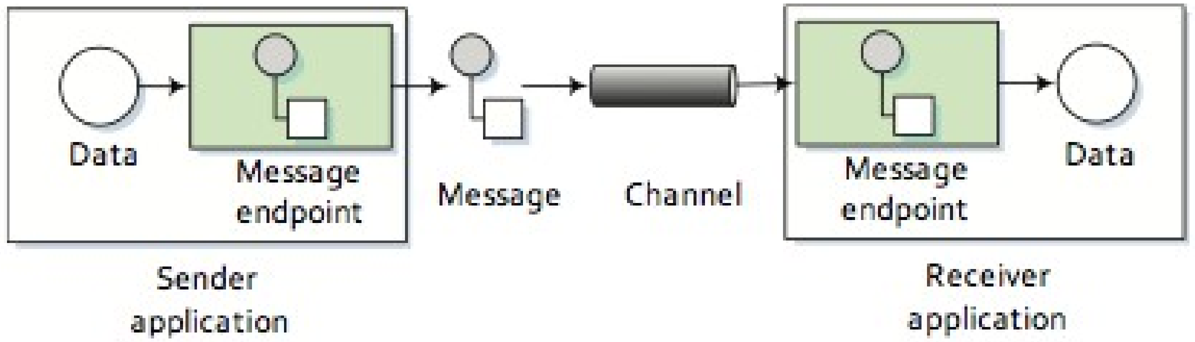


Figure 1.8
An endpoint acts as a neutral interface allowing systems to integrate.

Understanding Components

- Component is a factory for creating endpoints

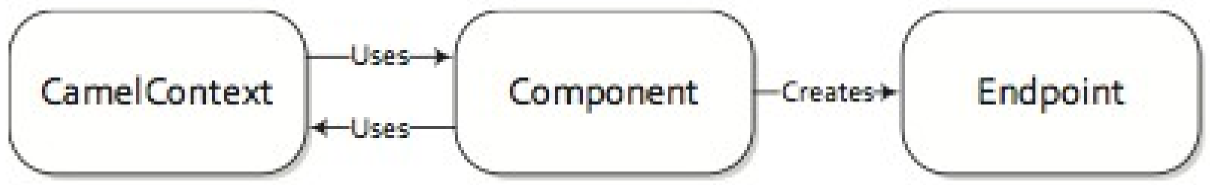
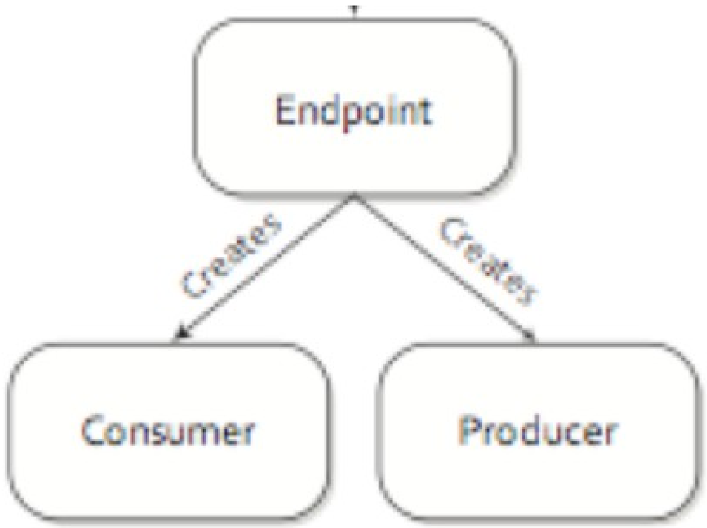


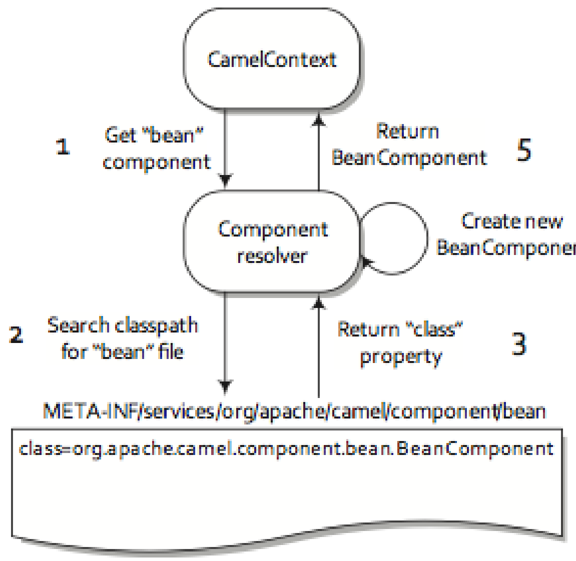
Figure 7.1 A component creates endpoints and may use the `CamelContext`'s facilities to accomplish this.

- … and endpoint factory for creating producer and/or consumers



Understanding Components

- Component auto discovered



CamelContext
1
Get "bean" component
Return BeanComponent
5
Component resolver
Create new BeanComponent
4
2
Search classpath for "bean" file
Return "class" property
3
META-INF/services/org/apache/camel/component/bean
class=org.apache.camel.component.bean.BeanComponent

Figure 7.2 To autodiscover a component named "bean", the component resolver searches for a file named "bean" in a specific directory on the classpath. This file specifies that the component class that will be created is `BeanComponent`.

- ... or manually added to CamelContext

```
CamelContext context = new DefaultCamelContext();
context.addComponent("jms",
    JmsComponent.jmsComponentAutoAcknowledge(connectionFactory));
```

Understanding Components

- Endpoints can be configured using URIs

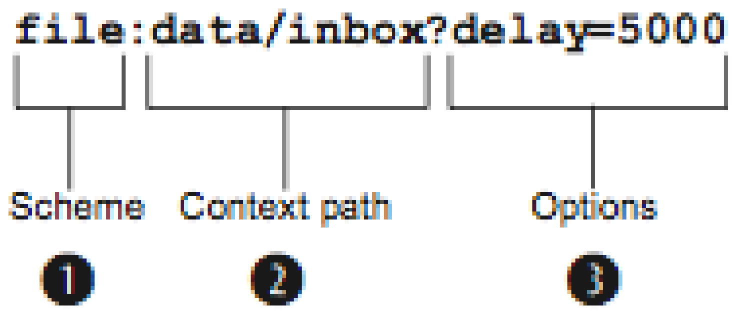
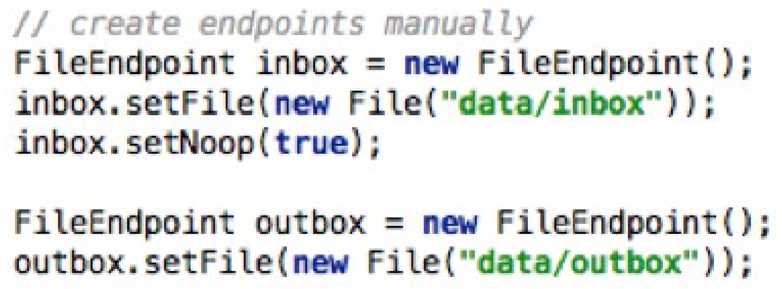


Figure 1.9 Endpoint URIs are divided into three parts: a scheme, a context path, and options.

- … or for example using Java code



```
// 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"));
```

Understanding Components

- Revisit File Copier Example

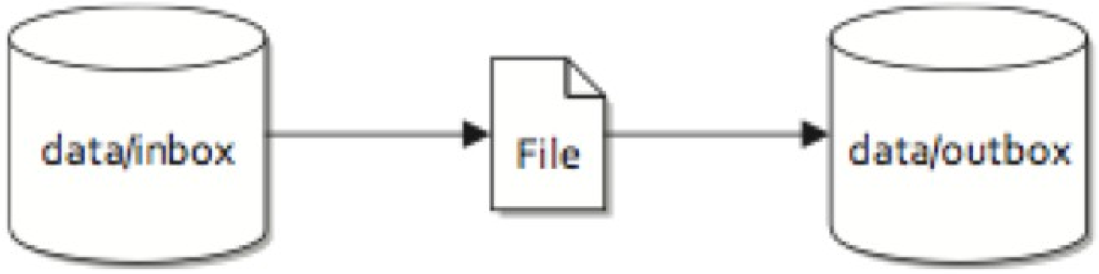
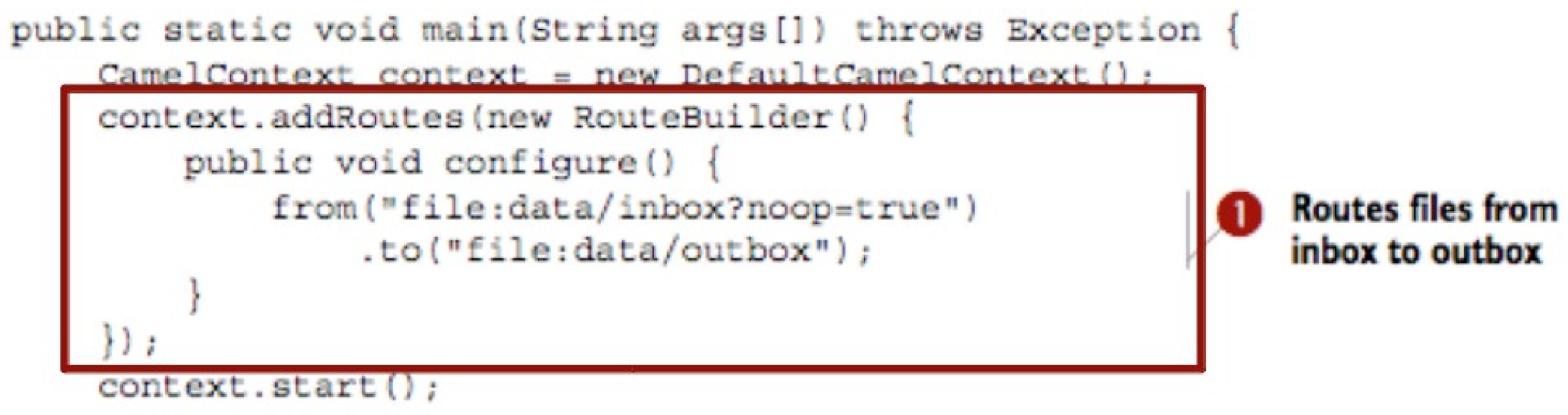


Figure 1.2 Files are routed from the data/inbox directory to the data/outbox directory.

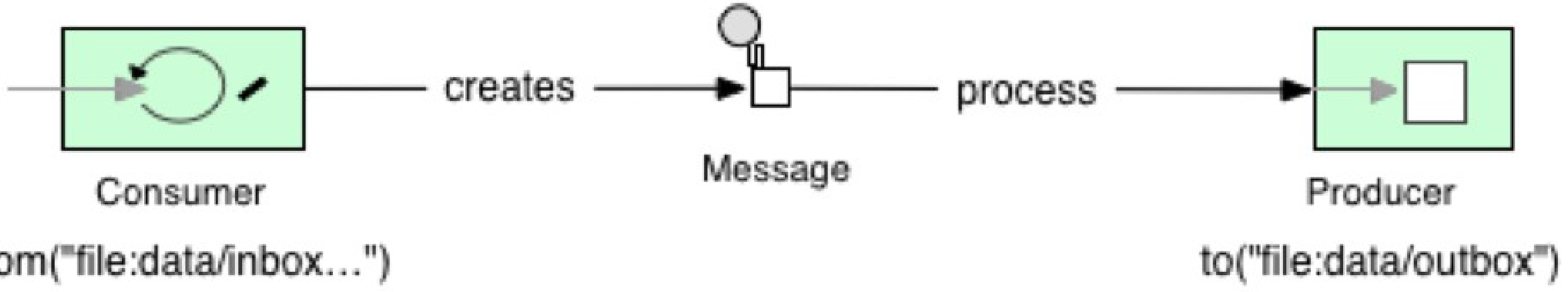
Understanding Components

- Revisit File Copier Example



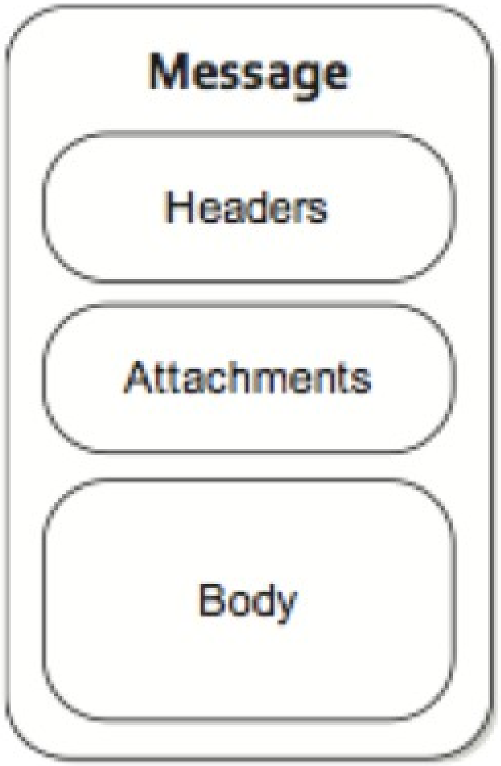
```
public static void main(String args[]) throws Exception {
    CamelContext context = new DefaultCamelContext();
    context.addRoutes(new RouteBuilder() {
        public void configure() {
            from("file:data/inbox?noop=true")
                .to("file:data/outbox");
        }
    });
  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

1 Routes files from inbox to outbox



Understanding Components

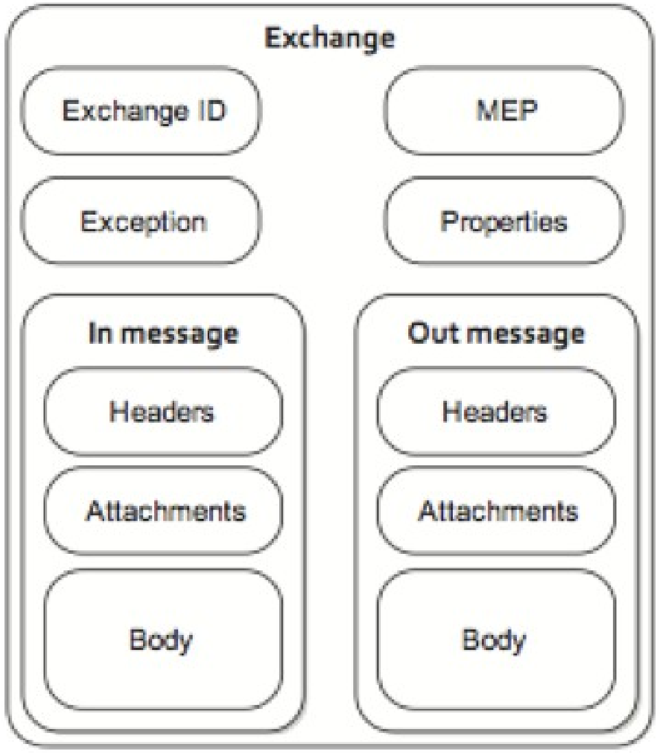
- What is a Message in Camel?



org.apache.camel.Message

Understanding Components

- … and contained in an Exchange during routing



org.apache.camel.Exchange

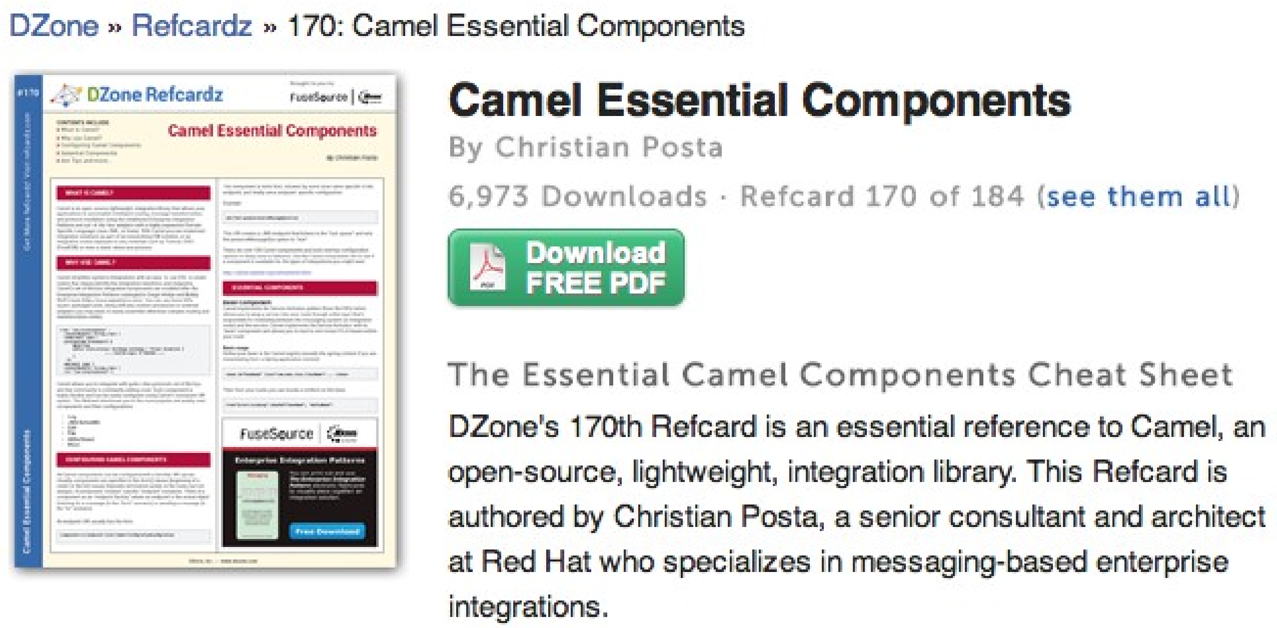
http://camel.apache.org/using-getin-or-getout-methods-on-exchange.html

Agenda

- A little Example
- Understanding Components
- **Essential Components**
- Creating new Components
- Q and A

Essential Components

- Camel Essential Components Reference Card



DZone » Refcardz » 170: Camel Essential Components

Camel Essential Components

By Christian Posta

6,973 Downloads · Refcard 170 of 184 (see them all)

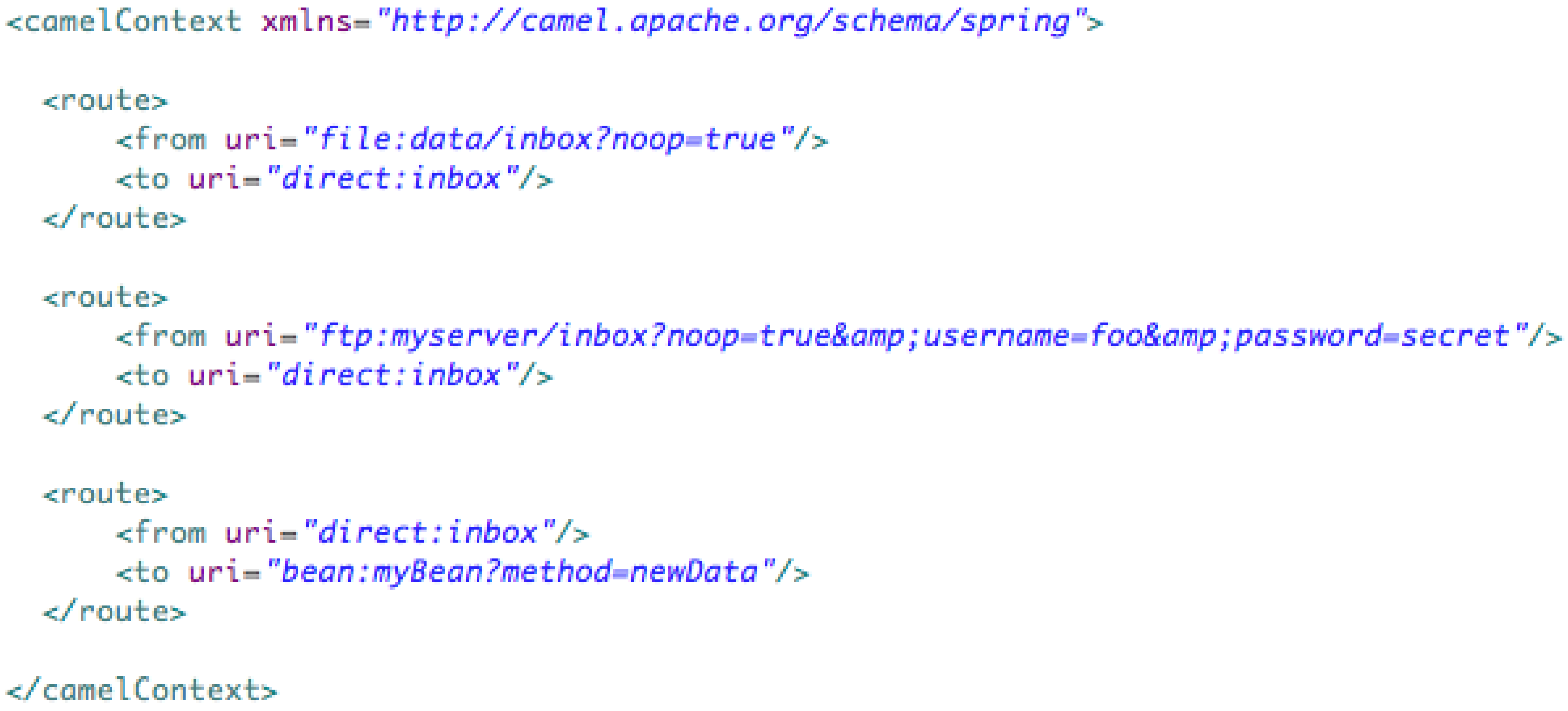
The Essential Camel Components Cheat Sheet

DZone's 170th Refcard is an essential reference to Camel, an open-source, lightweight, integration library. This Refcard is authored by Christian Posta, a senior consultant and architect at Red Hat who specializes in messaging-based enterprise integrations.

http://refcardz.dzone.com/refcardz/essential-camel-components

Essential Components

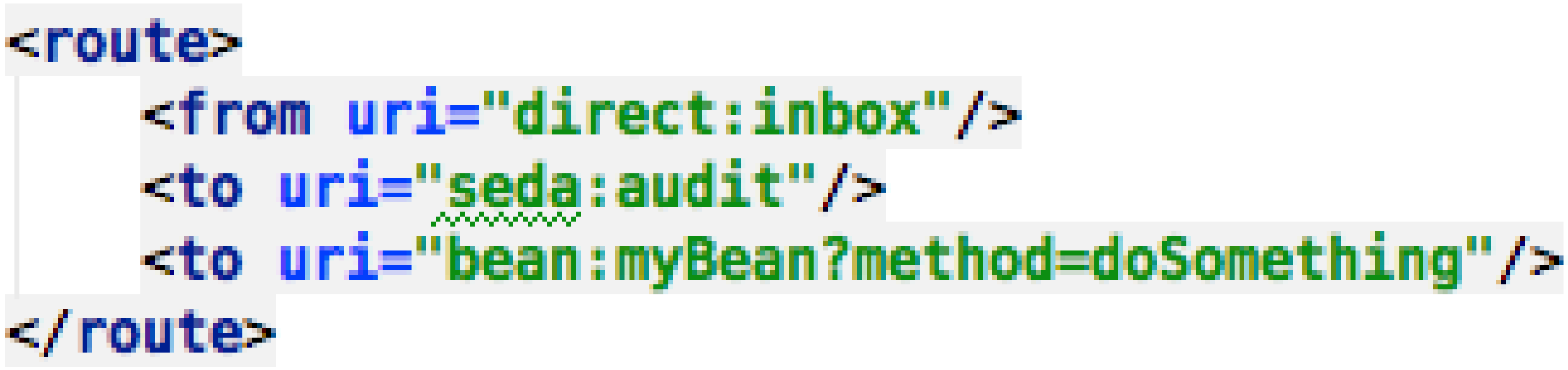
- Direct Component



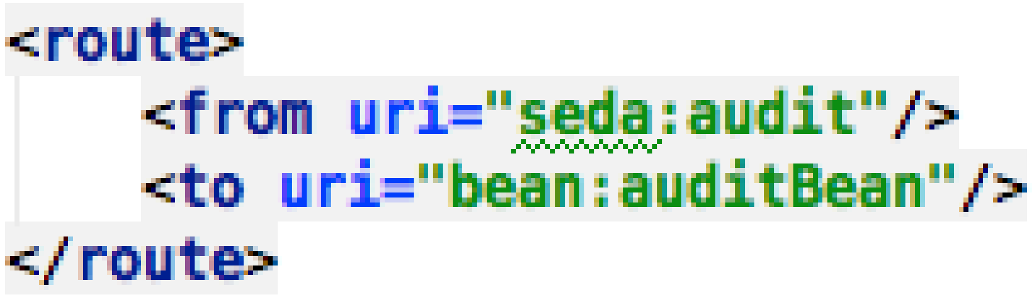
```
<camelContext xmlns="http://camel.apache.org/schema/spring">

  <route>
      <from uri="file:data/inbox?noop=true"/>
      <to uri="direct:inbox"/>
  </route>

  <route>
      <from uri="ftp:myserver/inbox?noop=true&amp;username=foo&amp;password=secret"/>
      <to uri="direct:inbox"/>
  </route>

  <route>
      <from uri="direct:inbox"/>
      <to uri="bean:myBean?method=newData"/>
  </route>

</camelContext>
```

Essential Components

- SEDA Component



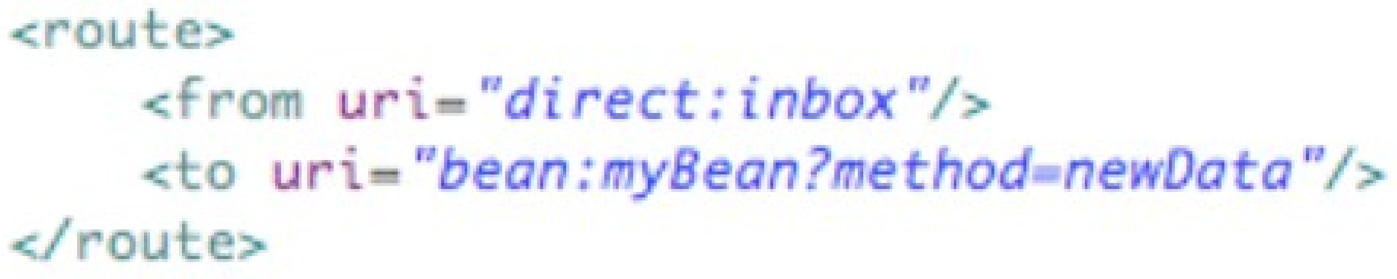
```
<route>
    <from uri="direct:inbox"/>
    <to uri="seda:audit"/>
    <to uri="bean:myBean?method=doSomething"/>
</route>
```



```
<route>
    <from uri="seda:audit"/>
    <to uri="bean:auditBean"/>
</route>
```

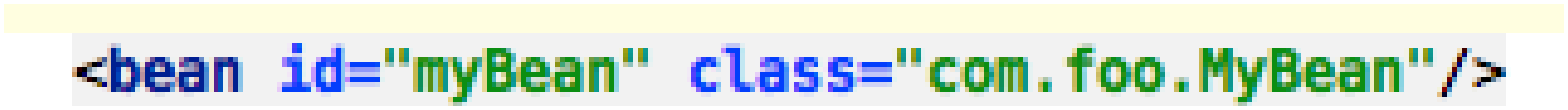
Essential Components

- Bean Component



```
<route>
    <from uri="direct:inbox"/>
    <to uri="bean:myBean?method=newData"/>
</route>
```

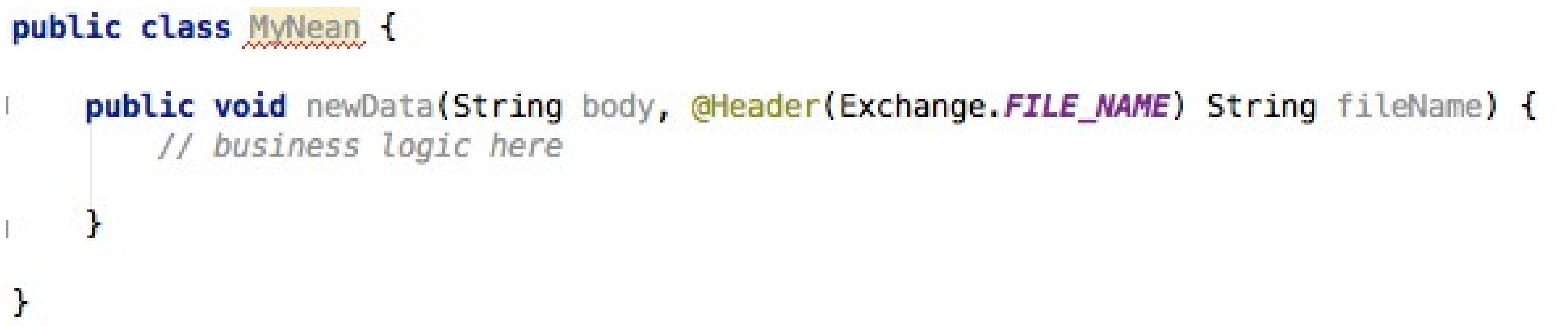
- … and use <bean> to declare the bean (standard Spring)



```
<bean id="myBean" class="com.foo.MyBean"/>
```

Essential Components

- … Camel adapts to bean method signature

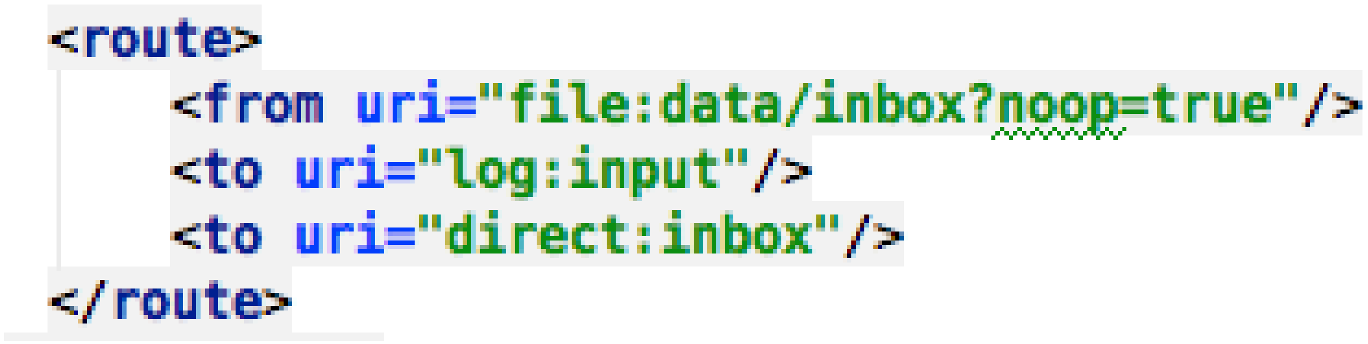

```
public class MyNean {

    public void newData(String body, @Header(Exchange.FILE_NAME) String fileName) {
        // business logic here

    }

}
```

… using bean parameter binding

- http://camel.apache.org/bean-binding.html
- http://camel.apache.org/parameter-binding-annotations.html

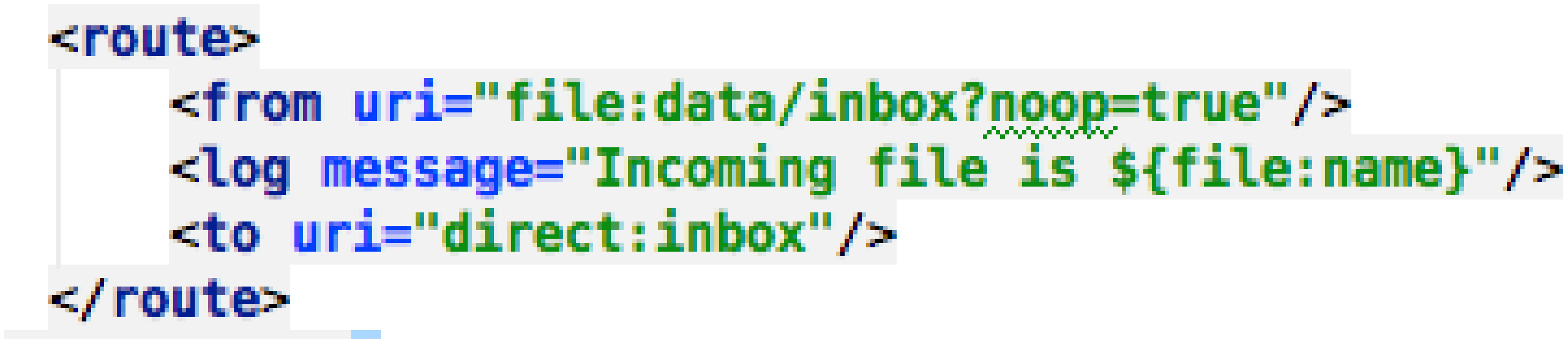
Essential Components

- Log Component



```
<route>
    <from uri="file:data/inbox?noop=true"/>
    <to uri="log:input"/>
    <to uri="direct:inbox"/>
</route>
```

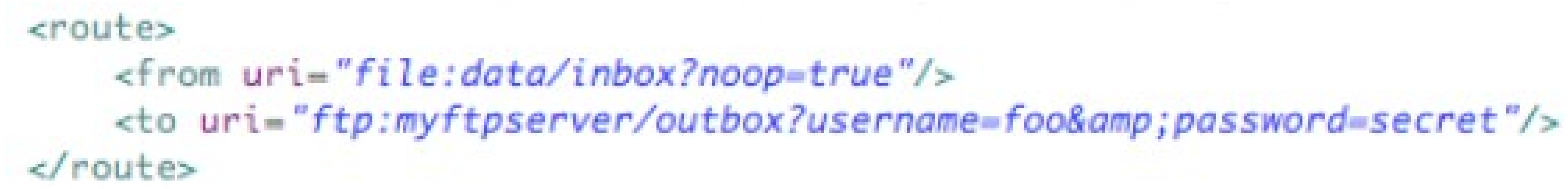
- Log EIP (human readable message)



```
<route>
    <from uri="file:data/inbox?noop=true"/>
    <log message="Incoming file is ${file:name}"/>
    <to uri="direct:inbox"/>
</route>
```

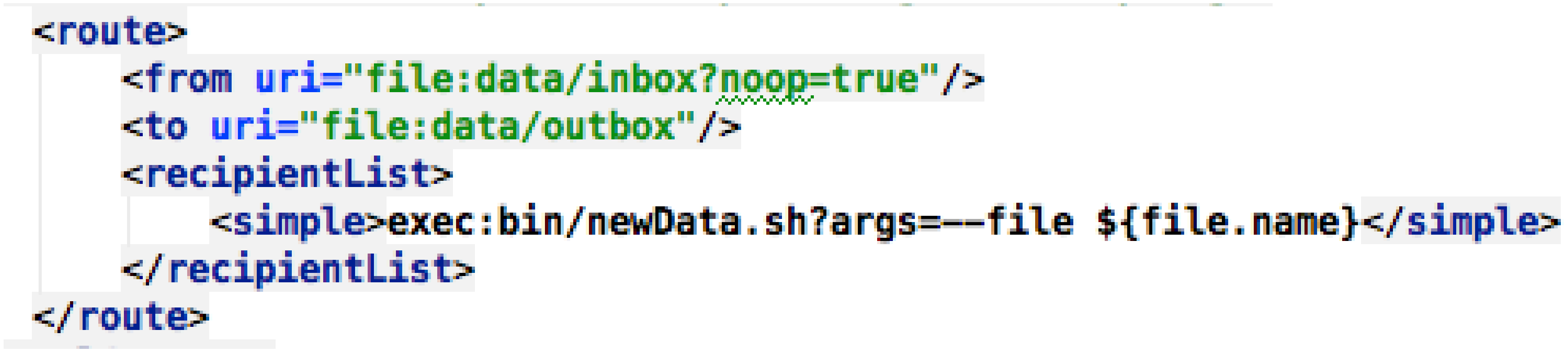
Essential Components

- File and FTP Components



```
<route>
    <from uri="file:data/inbox?noop=true"/>
    <to uri="ftp:myftpserver/outbox?username=foo&amp;password=secret"/>
</route>
```

- Exec Component

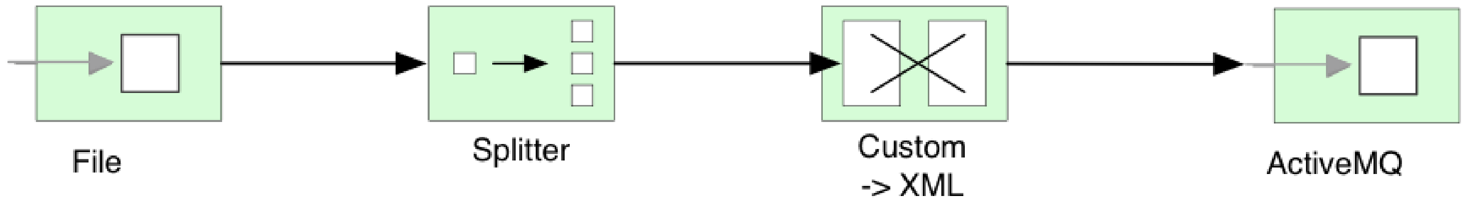


```
<route>
    <from uri="file:data/inbox?noop=true"/>
    <to uri="file:data/outbox"/>
    <recipientList>
        <simple>exec:bin/newData.sh?args=--file ${file.name}</simple>
    </recipientList>
</route>
```

http://camel.apache.org/how-do-i-use-dynamic-uri-in-to.html

Essential Components

- ActiveMQ / JMS



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

Take time to read about JMS at:
http://camel.apache.org/jms

Essential Components

- ActiveMQ / JMS (cont.)

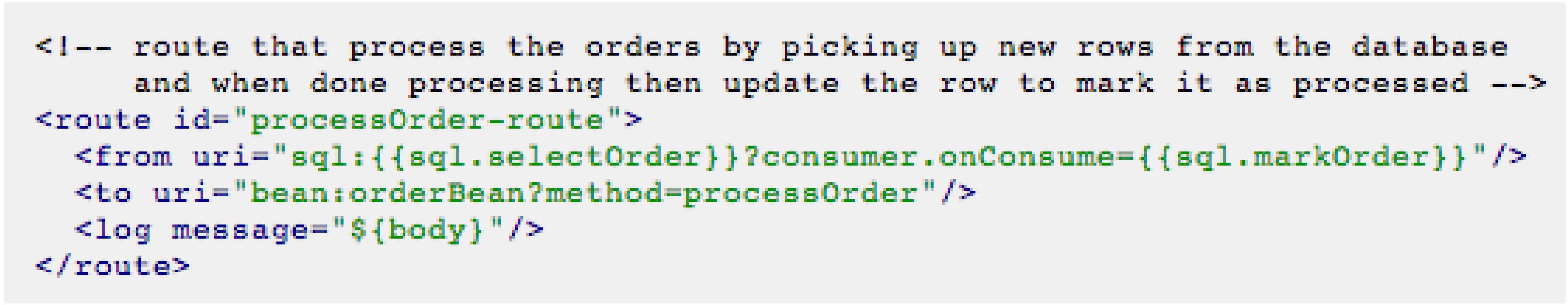
```
<bean id="jmsConnectionFactory"
   class="org.apache.activemq.ActiveMQConnectionFactory">
   <property name="brokerURL" value="tcp://localhost:61616" />
</bean>

<bean id="pooledConnectionFactory"
   class="org.apache.activemq.pool.PooledConnectionFactory" init-method="start" destroy-method="stop">
   <property name="maxConnections" value="8" />
   <property name="connectionFactory" ref="jmsConnectionFactory" />
</bean>

<bean id="jmsConfig"
   class="org.apache.camel.component.jms.JmsConfiguration">
   <property name="connectionFactory" ref="pooledConnectionFactory"/>
   <property name="concurrentConsumers" value="10"/>
</bean>

<bean id="activemq"
    class="org.apache.activemq.camel.component.ActiveMQComponent">
    <property name="configuration" ref="jmsConfig"/>
</bean>
```

http://camel.apache.org/activemq

If using transactions with JMS make sure to read about cache levels at: http://camel.apache.org/jms

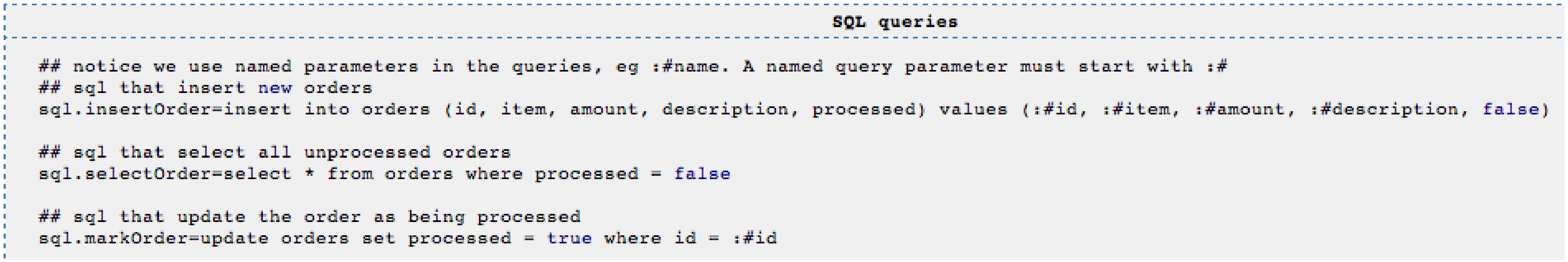
Essential Components

- SQL



```
<!-- route that process the orders by picking up new rows from the database
     and when done processing then update the row to mark it as processed -->
<route id="processOrder-route">
  <from uri="sql:{{sql.selectOrder}}?consumer.onConsume={{sql.markOrder}}"/>
  <to uri="bean:orderBean?method=processOrder"/>
  <log message="${body}"/>
</route>
```

- … uri is SQL, and body is SQL parameters.
- Externalize queries in .properties file

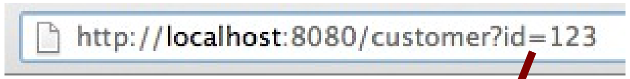


SQL queries

```
## notice we use named parameters in the queries, eg :#name. A named query parameter must start with :#
## sql that insert new orders
sql.insertOrder=insert into orders (id, item, amount, description, processed) values (:#id, :#item, :#amount, :#description, false)

## sql that select all unprocessed orders
sql.selectOrder=select * from orders where processed = false

## sql that update the order as being processed
sql.markOrder=update orders set processed = true where id = :#id
```

http://camel.apache.org/sql-example.html

Essential Components


http://localhost:8080/customer?id=123

- JDBC

```
<route>
    <from uri="jetty:http://0.0.0.0:8080/customer"/>
    <setBody>
        <simple>select cust_name as name from customer where cust_id = ${header.id}</simple>
    </setBody>
    <to uri="jdbc:customerDB"/>
    <setBody>
        <simple>The customer is: ${body[0]['name']}</simple>
    </setBody>
</route>
```

- … body is SQL and result is List<Map> (eg like ResultSet)

  Improvement on the way: https://issues.apache.org/jira/browse/CAMEL-6367

http://camel.apache.org/sql-example.html

Essential Components

- Other Database Components
  - JPA
  - Hibernate
  - MyBatis

http://camel.apache.org/sql-example.html

Essential Components

- HTTP Server Components
  - Jetty / Servlet
- HTTP Client Components
  - HTTP / HTTP4 / Jetty / AHC
- Web Service Components
  - CXF / Spring-WS
- REST Components
  - CXF-RS / Restlet

http://camel.apache.org/sql-example.html

Essential Components

- TCP/UDP Components
  - Mina / Mina2
  - Netty

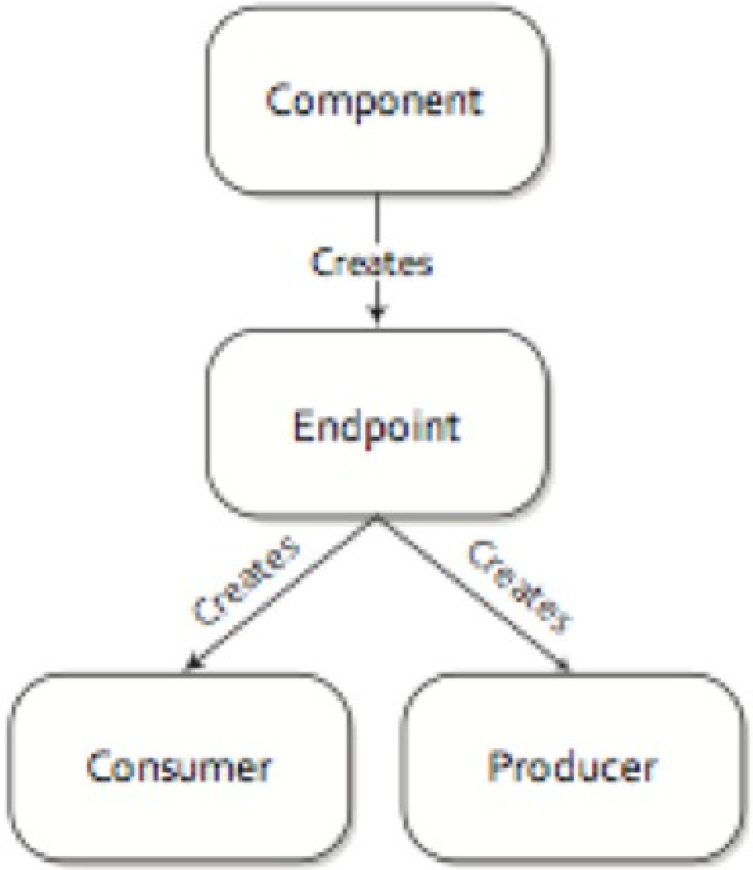
http://camel.apache.org/sql-example.html

Agenda

- A little Example
- Understanding Components
- Essential Components
- **Creating new Components**
- Q and A

Creating new Camel Components

- The big picture

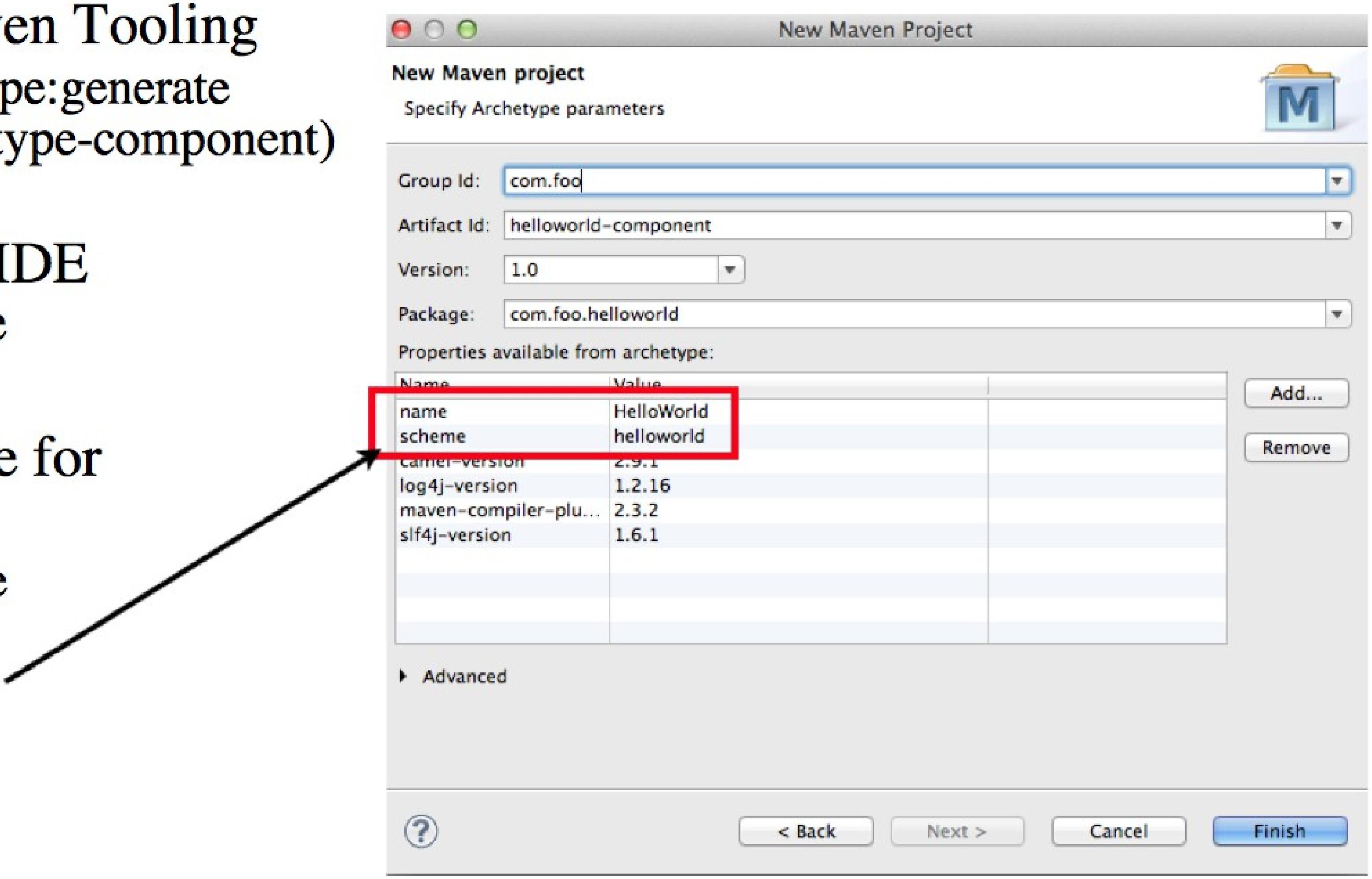


Creating new Camel Components

- 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



New Maven project

Specify Archetype parameters

Group Id: com.foo

Artifact Id: helloworld-component

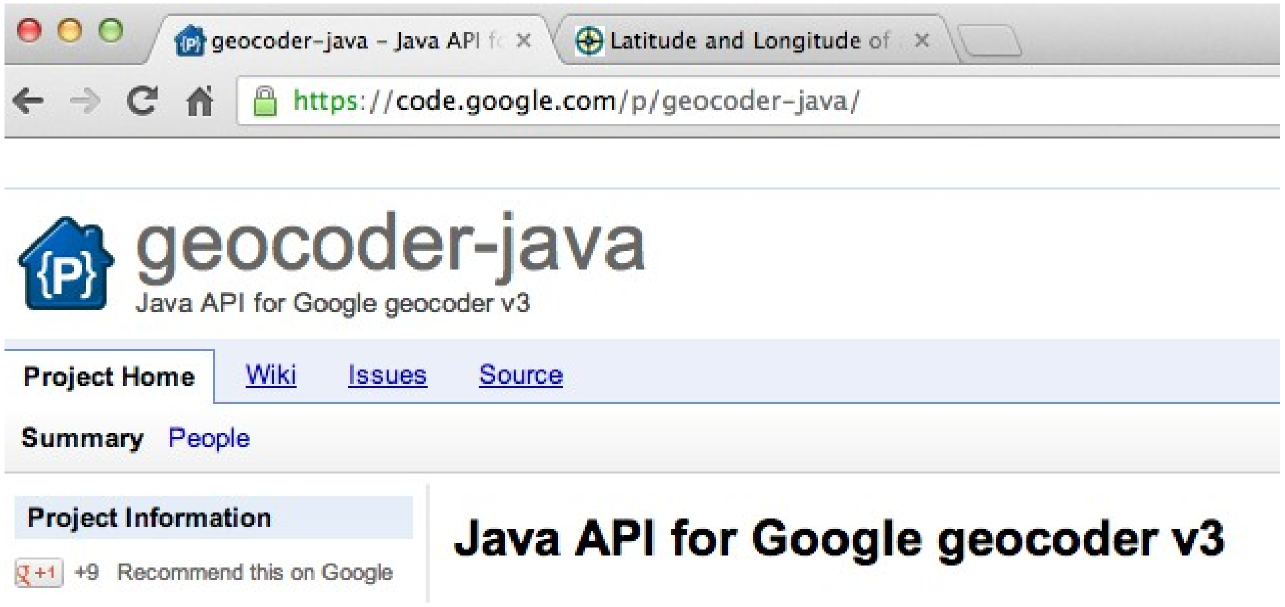
Version: 1.0

Package: com.foo.helloworld

Properties available from archetype:

Name	Value
name	HelloWorld
scheme	helloworld
camel-version	2.9.1
log4j-version	1.2.16
maven-compiler-plu...	2.3.2
slf4j-version	1.6.1

Creating new Camel Components


geocoder-java - Java API f
Latitude and Longitude of
https://code.google.com/p/geocoder-java/

geocoder-java
Java API for Google geocoder v3

Project Home | Wiki | Issues | Source
Summary | People

Project Information
+1 +9 Recommend this on Google

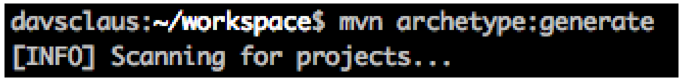
Java API for Google geocoder v3

- Using Command Shell

```
mvn archetype:generate
```

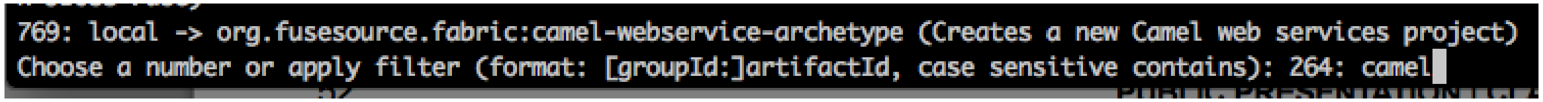
Creating new Camel Components

1. execute this maven command



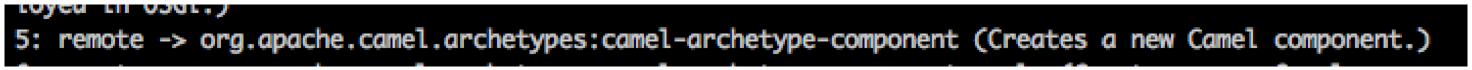
```
davsclaus:~/workspace$ mvn archetype:generate
[INFO] Scanning for projects...
```

2. type camel to filter only Camel archetypes



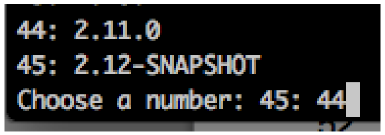
```
769: local -> org.fusesource.fabric:camel-webservice-archetype (Creates a new Camel web services project)
Choose a number or apply filter (format: [groupId:]artifactId, case sensitive contains): 264: camel
```

3. type number to select "camel-archetype-component" (in this ex its 5)

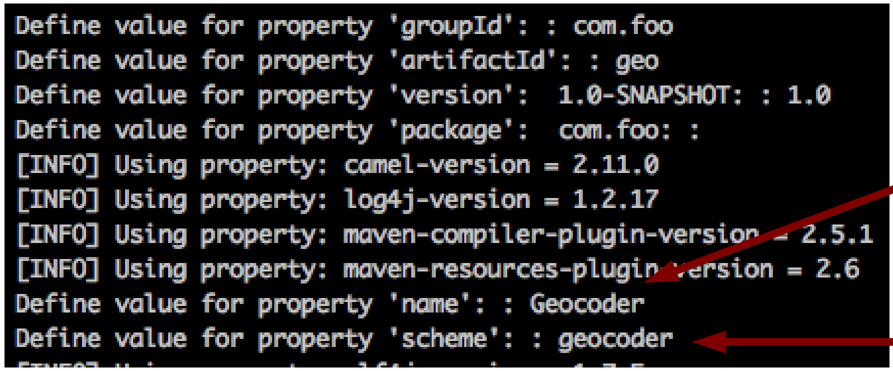


```
5: remote -> org.apache.camel.archetypes:camel-archetype-component (Creates a new Camel component.)
```

4. select the Camel version to use



```
44: 2.11.0
45: 2.12-SNAPSHOT
Choose a number: 45: 44
```



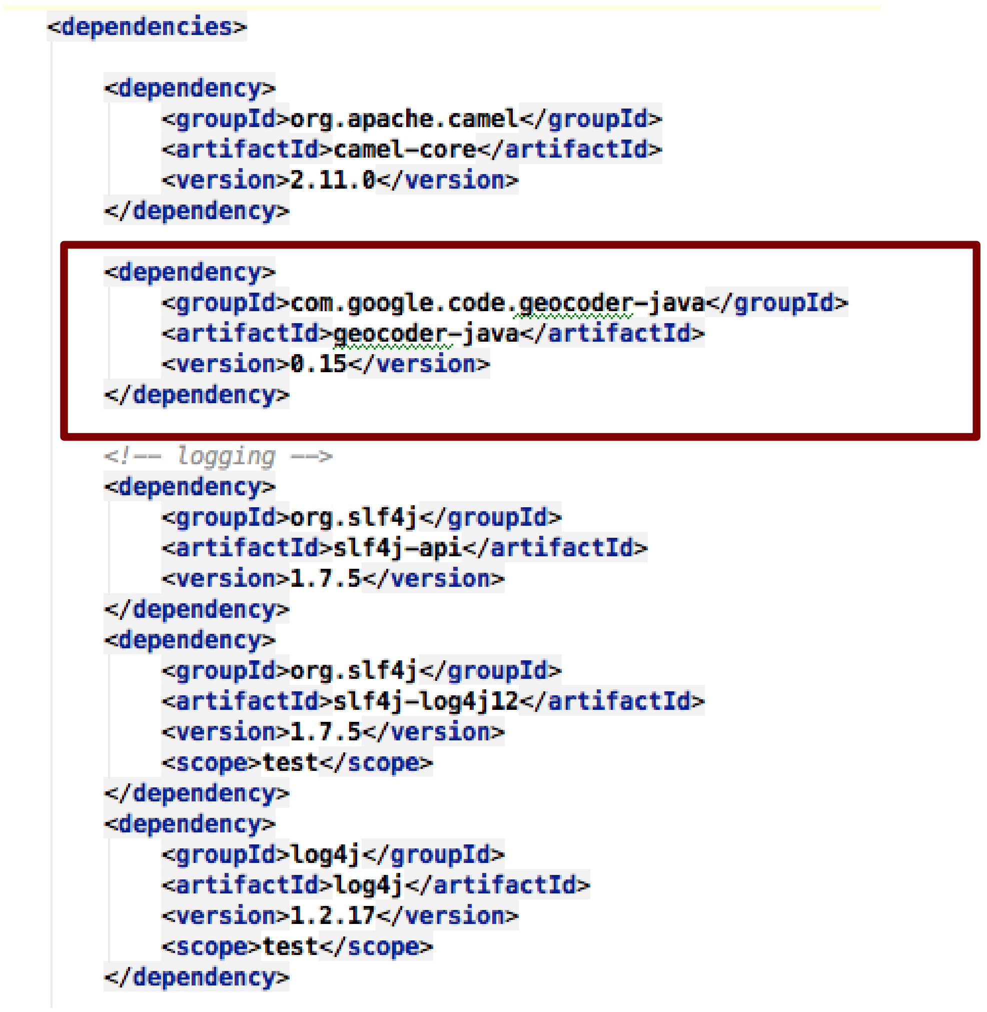
```
Define value for property 'groupId': : com.foo
Define value for property 'artifactId': : geo
Define value for property 'version':  1.0-SNAPSHOT: : 1.0
Define value for property 'package':  com.foo: :
[INFO] Using property: camel-version = 2.11.0
[INFO] Using property: log4j-version = 1.2.17
[INFO] Using property: maven-compiler-plugin-version = 2.5.1
[INFO] Using property: maven-resources-plugin-version = 2.6
Define value for property 'name': : Geocoder
Define value for property 'scheme': : geocoder
```

Geocoder = Java component name
(must be first letter in upper case)

geocoder = Camel component name
(must be lower-case)

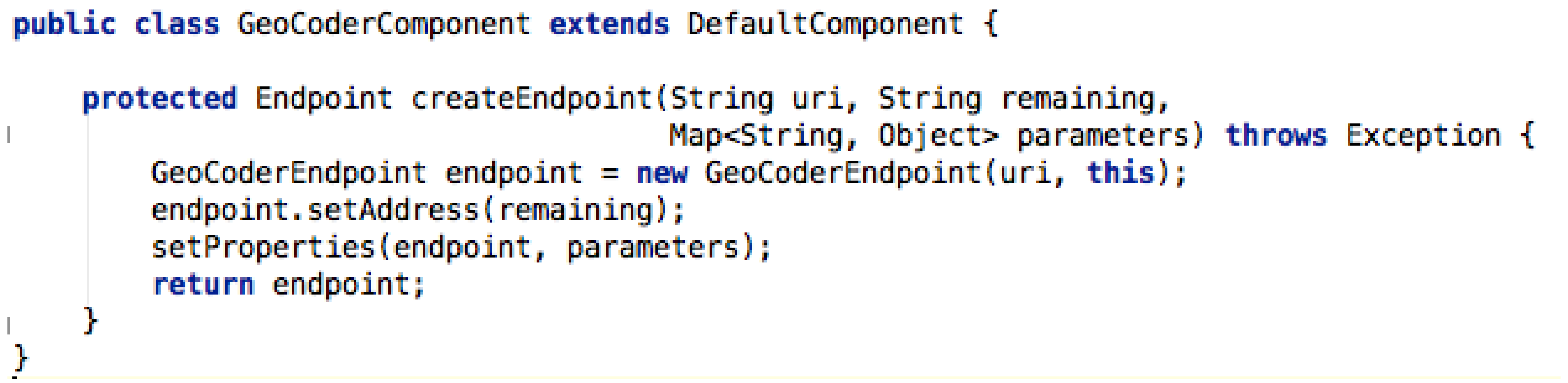
Creating new Camel Components

- Add 3rd party library to pom.xml file



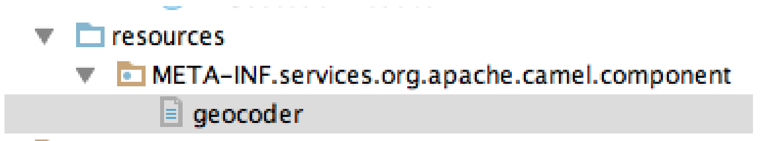
```
<dependencies>

    <dependency>
        <groupId>org.apache.camel</groupId>
        <artifactId>camel-core</artifactId>
        <version>2.11.0</version>
    </dependency>

    <dependency>
        <groupId>com.google.code.geocoder-java</groupId>
        <artifactId>geocoder-java</artifactId>
        <version>0.15</version>
    </dependency>

    <!-- logging -->
    <dependency>
        <groupId>org.slf4j</groupId>
        <artifactId>slf4j-api</artifactId>
        <version>1.7.5</version>
    </dependency>
    <dependency>
        <groupId>org.slf4j</groupId>
        <artifactId>slf4j-log4j12</artifactId>
        <version>1.7.5</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>log4j</groupId>
        <artifactId>log4j</artifactId>
        <version>1.2.17</version>
        <scope>test</scope>
    </dependency>
```

Creating new Camel Components



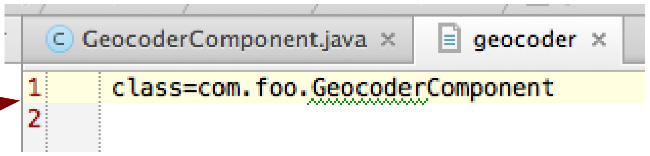
```
public class GeoCoderComponent extends DefaultComponent {

    protected Endpoint createEndpoint(String uri, String remaining,
                                      Map<String, Object> parameters) throws Exception {
        GeoCoderEndpoint endpoint = new GeoCoderEndpoint(uri, this);
        endpoint.setAddress(remaining);
        setProperties(endpoint, parameters);
        return endpoint;
    }
}
```

- Auto discover component



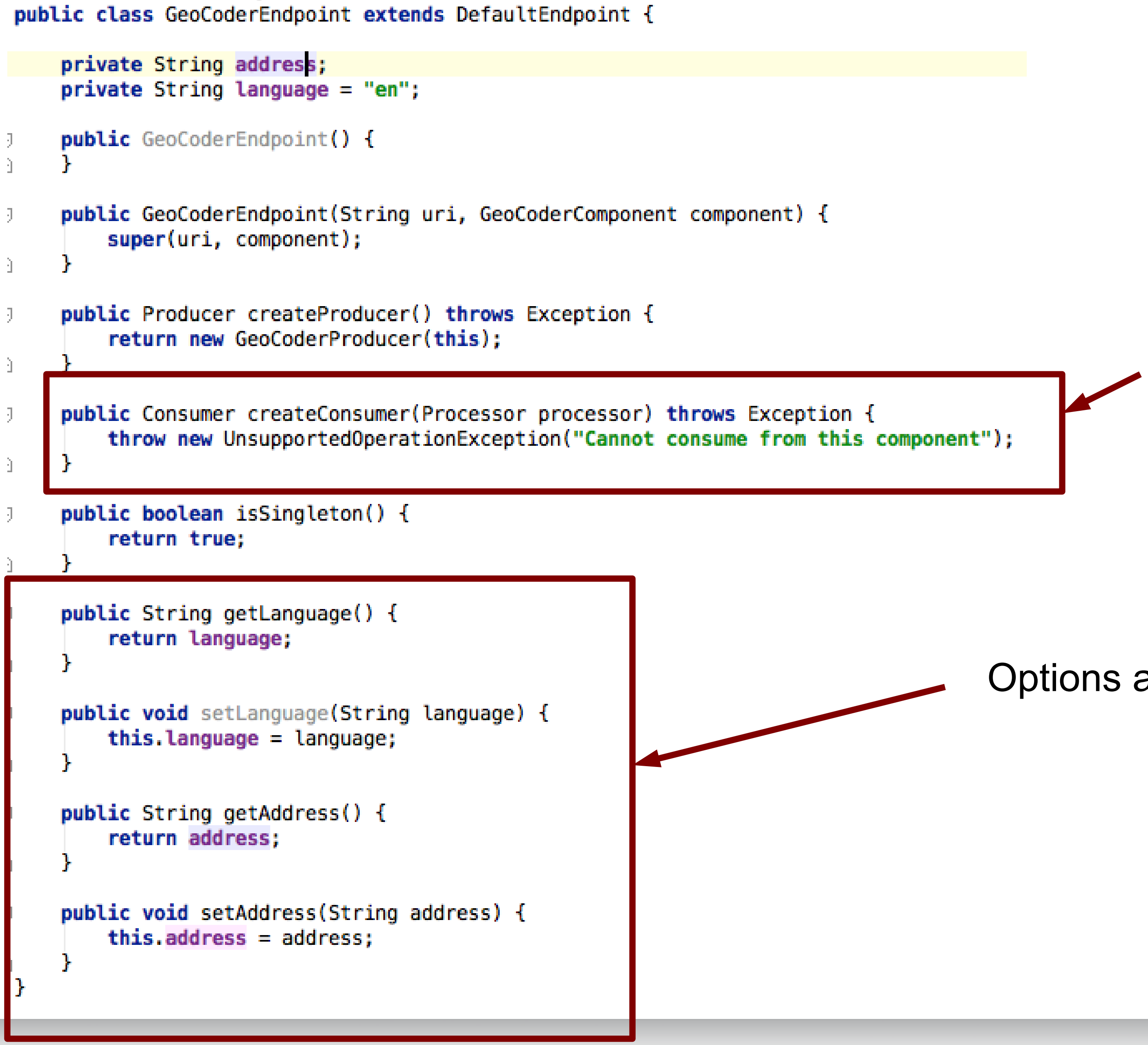
```
▼ resources
  ▼ META-INF.services.org.apache.camel.component
      geocoder
```

file in META-INF classpath



```
class=com.foo.GeocoderComponent
```

Creating new Camel Components



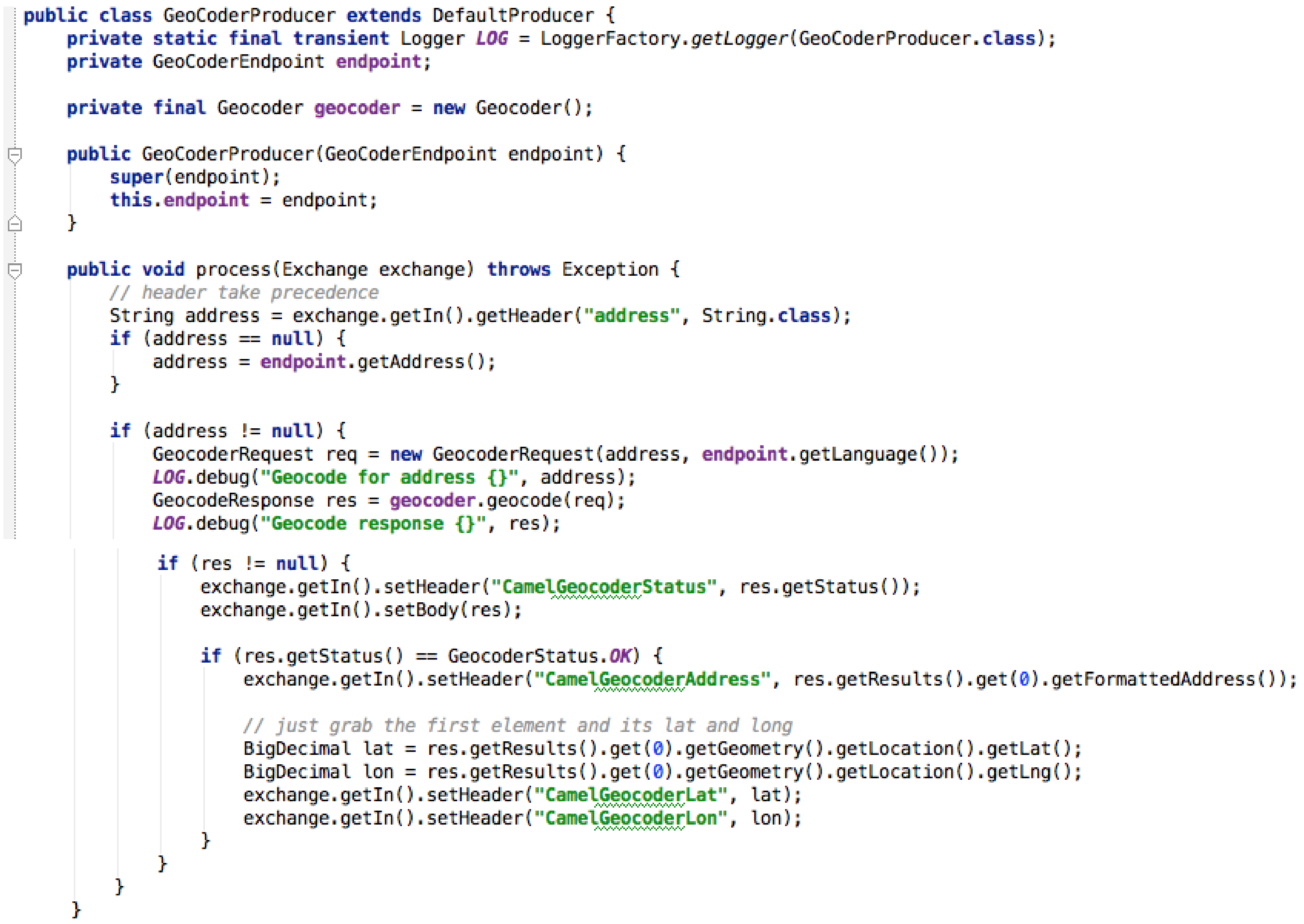
```java
public class GeoCoderEndpoint extends DefaultEndpoint {

    private String address;
    private String language = "en";

    public GeoCoderEndpoint() {
    }

    public GeoCoderEndpoint(String uri, GeoCoderComponent component) {
        super(uri, component);
    }

    public Producer createProducer() throws Exception {
        return new GeoCoderProducer(this);
    }

    public Consumer createConsumer(Processor processor) throws Exception {
        throw new UnsupportedOperationException("Cannot consume from this component");
    }

    public boolean isSingleton() {
        return true;
    }

    public String getLanguage() {
        return language;
    }

    public void setLanguage(String language) {
        this.language = language;
    }

    public String getAddress() {
        return address;
    }

    public void setAddress(String address) {
        this.address = address;
    }
}
```

Consumer is not supported

Options as getter/setter

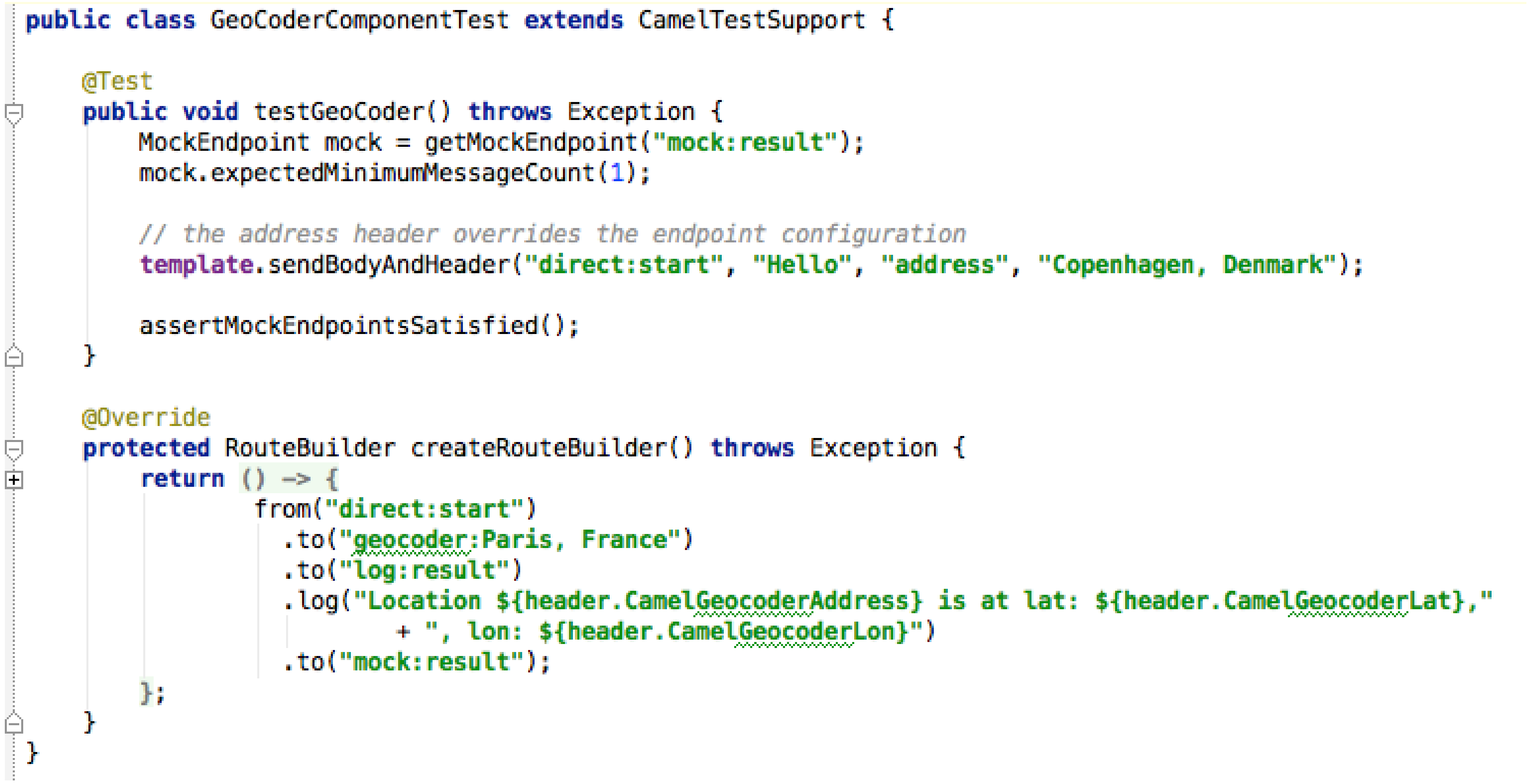
redhat.

Creating new Camel Components



```java
public class GeoCoderProducer extends DefaultProducer {
    private static final transient Logger LOG = LoggerFactory.getLogger(GeoCoderProducer.class);
    private GeoCoderEndpoint endpoint;

    private final Geocoder geocoder = new Geocoder();

    public GeoCoderProducer(GeoCoderEndpoint endpoint) {
        super(endpoint);
        this.endpoint = endpoint;
    }

    public void process(Exchange exchange) throws Exception {
        // header take precedence
        String address = exchange.getIn().getHeader("address", String.class);
        if (address == null) {
            address = endpoint.getAddress();
        }

        if (address != null) {
            GeocoderRequest req = new GeocoderRequest(address, endpoint.getLanguage());
            LOG.debug("Geocode for address {}", address);
            GeocodeResponse res = geocoder.geocode(req);
            LOG.debug("Geocode response {}", res);

            if (res != null) {
                exchange.getIn().setHeader("CamelGeocoderStatus", res.getStatus());
                exchange.getIn().setBody(res);

                if (res.getStatus() == GeocoderStatus.OK) {
                    exchange.getIn().setHeader("CamelGeocoderAddress", res.getResults().get(0).getFormattedAddress());

                    // just grab the first element and its lat and long
                    BigDecimal lat = res.getResults().get(0).getGeometry().getLocation().getLat();
                    BigDecimal lon = res.getResults().get(0).getGeometry().getLocation().getLng();
                    exchange.getIn().setHeader("CamelGeocoderLat", lat);
                    exchange.getIn().setHeader("CamelGeocoderLon", lon);
                }
            }
        }
    }
```

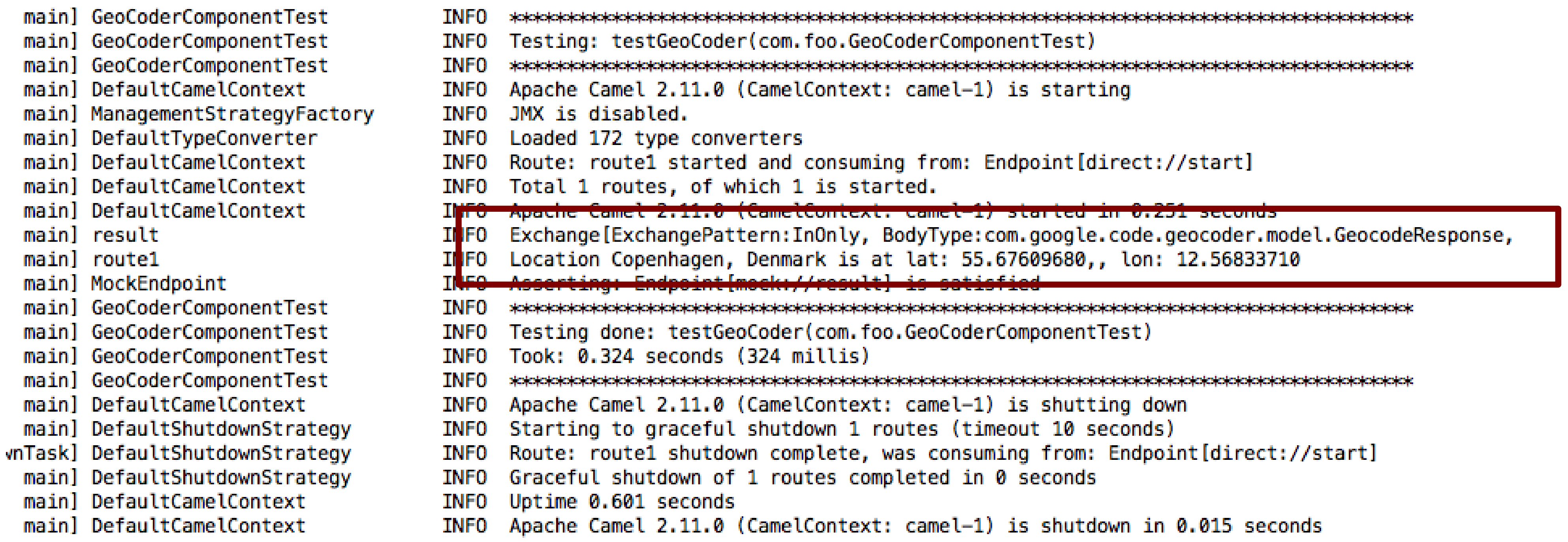
Creating new Camel Components



```
public class GeoCoderComponentTest extends CamelTestSupport {

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

        // the address header overrides the endpoint configuration
        template.sendBodyAndHeader("direct:start", "Hello", "address", "Copenhagen, Denmark");

        assertMockEndpointsSatisfied();
    }

    @Override
    protected RouteBuilder createRouteBuilder() throws Exception {
        return () -> {
                from("direct:start")
                    .to("geocoder:Paris, France")
                    .to("log:result")
                    .log("Location ${header.CamelGeocoderAddress} is at lat: ${header.CamelGeocoderLat},"
                            + ", lon: ${header.CamelGeocoderLon}")
                    .to("mock:result");
        };
    }
}
```

Creating new Camel Components

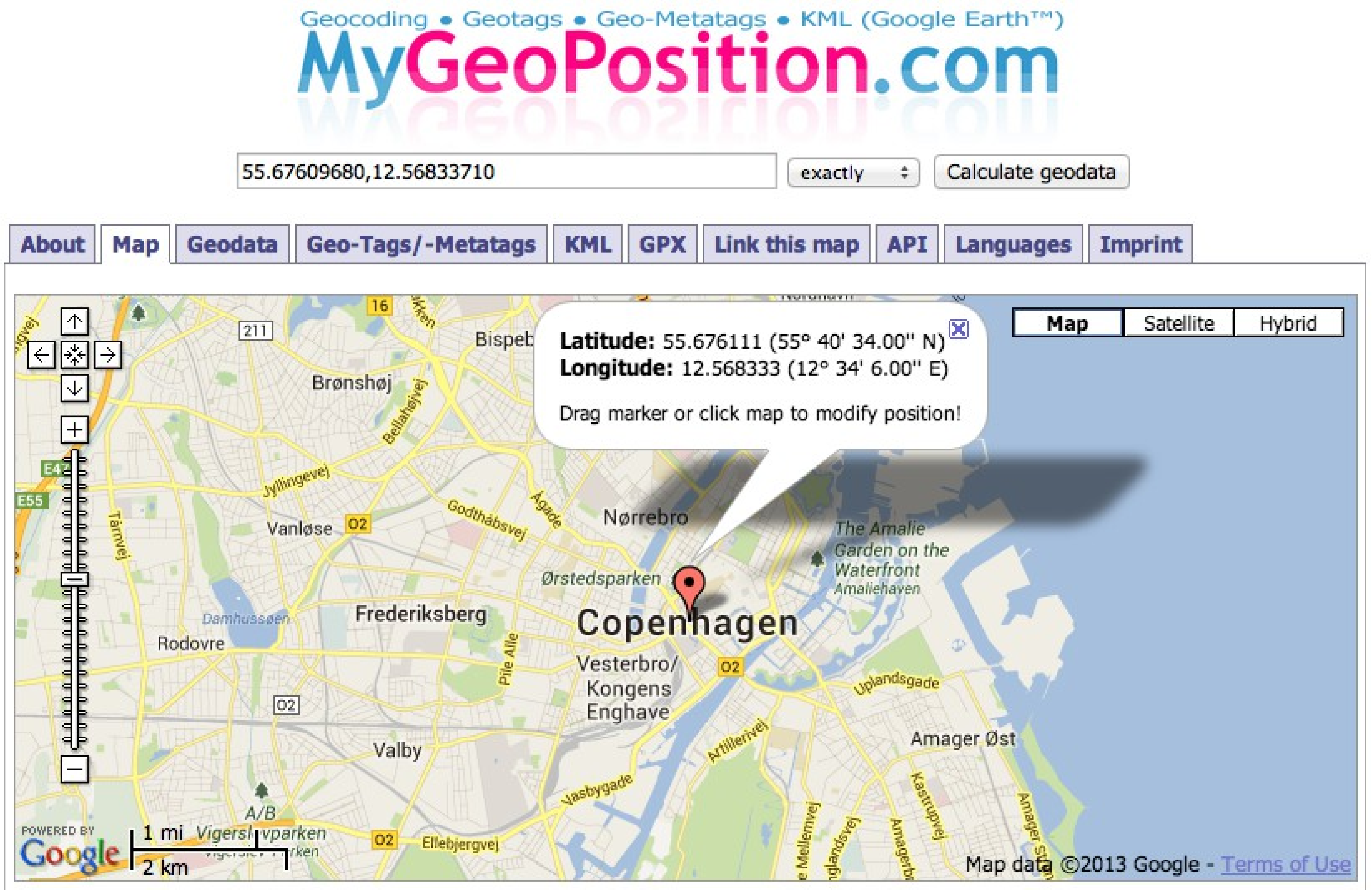
- Running unit test ...



```
 main] GeoCoderComponentTest        INFO  ********************************************************************************
 main] GeoCoderComponentTest        INFO  Testing: testGeoCoder(com.foo.GeoCoderComponentTest)
 main] GeoCoderComponentTest        INFO  ********************************************************************************
 main] DefaultCamelContext          INFO  Apache Camel 2.11.0 (CamelContext: camel-1) is starting
 main] ManagementStrategyFactory    INFO  JMX is disabled.
 main] DefaultTypeConverter         INFO  Loaded 172 type converters
 main] DefaultCamelContext          INFO  Route: route1 started and consuming from: Endpoint[direct://start]
 main] DefaultCamelContext          INFO  Total 1 routes, of which 1 is started.
 main] DefaultCamelContext          INFO  Apache Camel 2.11.0 (CamelContext: camel-1) started in 0.251 seconds
 main] result                       INFO  Exchange[ExchangePattern:InOnly, BodyType:com.google.code.geocoder.model.GeocodeResponse,
 main] route1                       INFO  Location Copenhagen, Denmark is at lat: 55.67609680,, lon: 12.56833710
 main] MockEndpoint                 INFO  Asserting: Endpoint[mock://result] is satisfied
 main] GeoCoderComponentTest        INFO  ********************************************************************************
 main] GeoCoderComponentTest        INFO  Testing done: testGeoCoder(com.foo.GeoCoderComponentTest)
 main] GeoCoderComponentTest        INFO  Took: 0.324 seconds (324 millis)
 main] GeoCoderComponentTest        INFO  ********************************************************************************
 main] DefaultCamelContext          INFO  Apache Camel 2.11.0 (CamelContext: camel-1) is shutting down
 main] DefaultShutdownStrategy      INFO  Starting to graceful shutdown 1 routes (timeout 10 seconds)
wnTask] DefaultShutdownStrategy     INFO  Route: route1 shutdown complete, was consuming from: Endpoint[direct://start]
 main] DefaultShutdownStrategy      INFO  Graceful shutdown of 1 routes completed in 0 seconds
 main] DefaultCamelContext          INFO  Uptime 0.601 seconds
 main] DefaultCamelContext          INFO  Apache Camel 2.11.0 (CamelContext: camel-1) is shutdown in 0.015 seconds
```

Creating new Camel Components

- Map of location



Geocoding • Geotags • Geo-Metatags • KML (Google Earth™)

MyGeoPosition.com

55.67609680,12.56833710 | exactly | Calculate geodata

About | Map | Geodata | Geo-Tags/-Metatags | KML | GPX | Link this map | API | Languages | Imprint

Latitude: 55.676111 (55° 40' 34.00" N)
Longitude: 12.568333 (12° 34' 6.00" E)

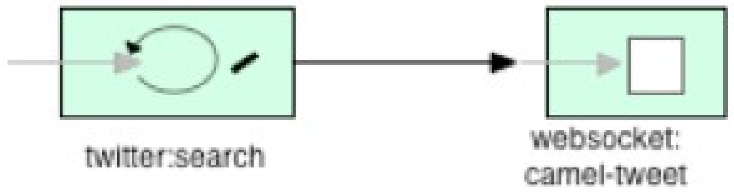
Drag marker or click map to modify position!

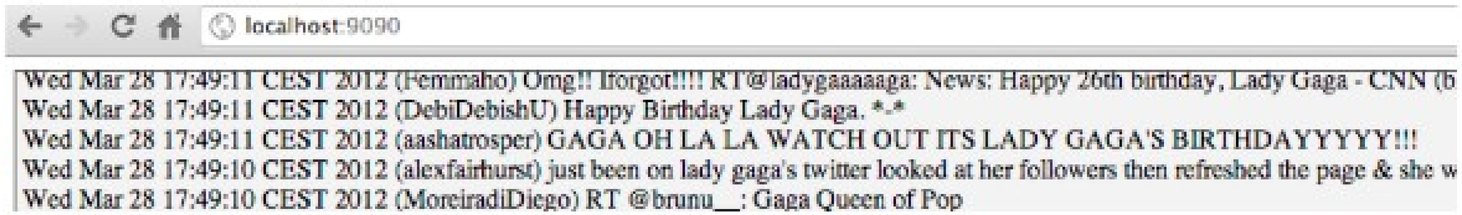
Map data ©2013 Google - Terms of Use

Fun with our new component

- Extending twitter example with geo and weather data

  - Twitter Example



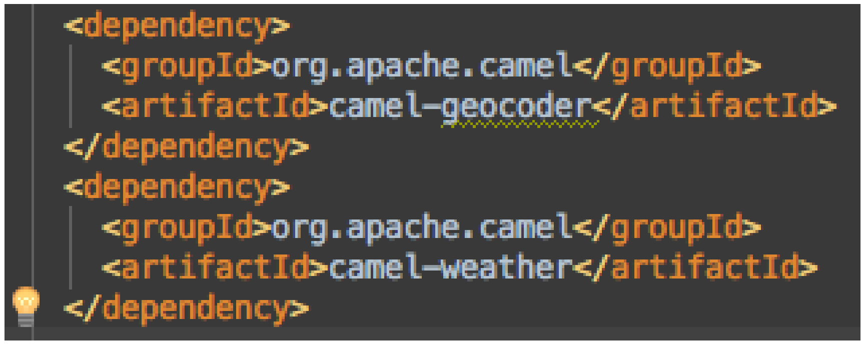


```
Wed Mar 28 17:49:11 CEST 2012 (Femmaho) Omg!! Iforgot!!!! RT@ladygaaaaaga: News: Happy 26th birthday, Lady Gaga - CNN (b
Wed Mar 28 17:49:11 CEST 2012 (DebiDebishU) Happy Birthday Lady Gaga. *-*
Wed Mar 28 17:49:11 CEST 2012 (aashatrosper) GAGA OH LA LA WATCH OUT ITS LADY GAGA'S BIRTHDAYYYYY!!!
Wed Mar 28 17:49:10 CEST 2012 (alexfairhurst) just been on lady gaga's twitter looked at her followers then refreshed the page & she w
Wed Mar 28 17:49:10 CEST 2012 (MoreiradiDiego) RT @brunu__: Gaga Queen of Pop
```

  - cd examples/camel-example-twitter-websocket
  - mvn compile exec:java

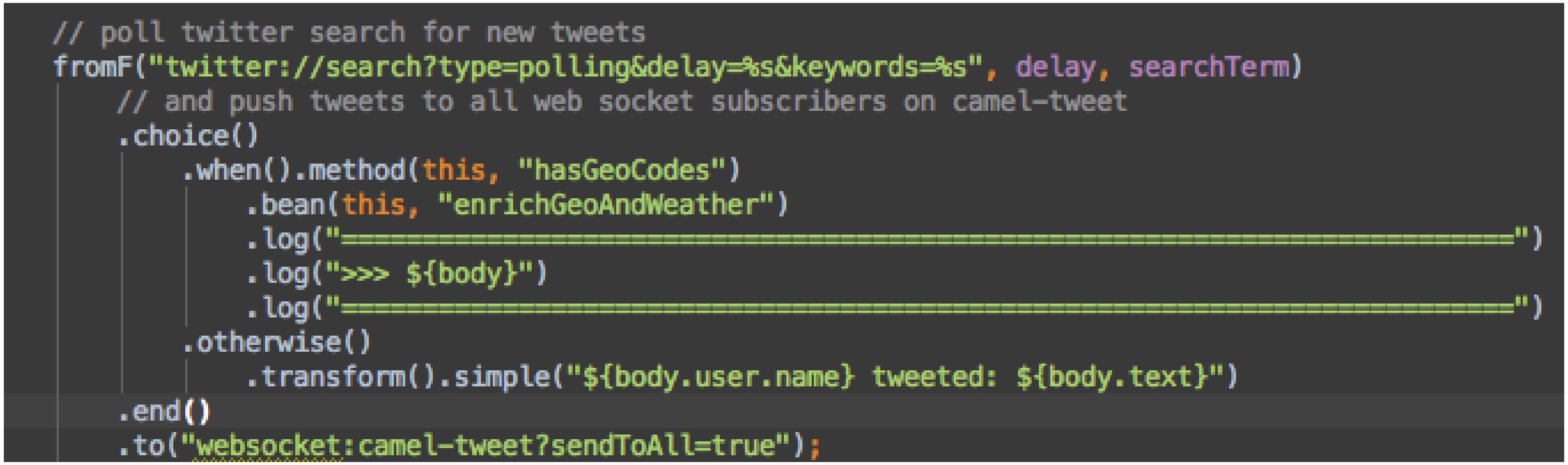
Fun with our new component

- Adding new dependencies



```
<dependency>
  <groupId>org.apache.camel</groupId>
  <artifactId>camel-geocoder</artifactId>
</dependency>
<dependency>
  <groupId>org.apache.camel</groupId>
  <artifactId>camel-weather</artifactId>
</dependency>
```

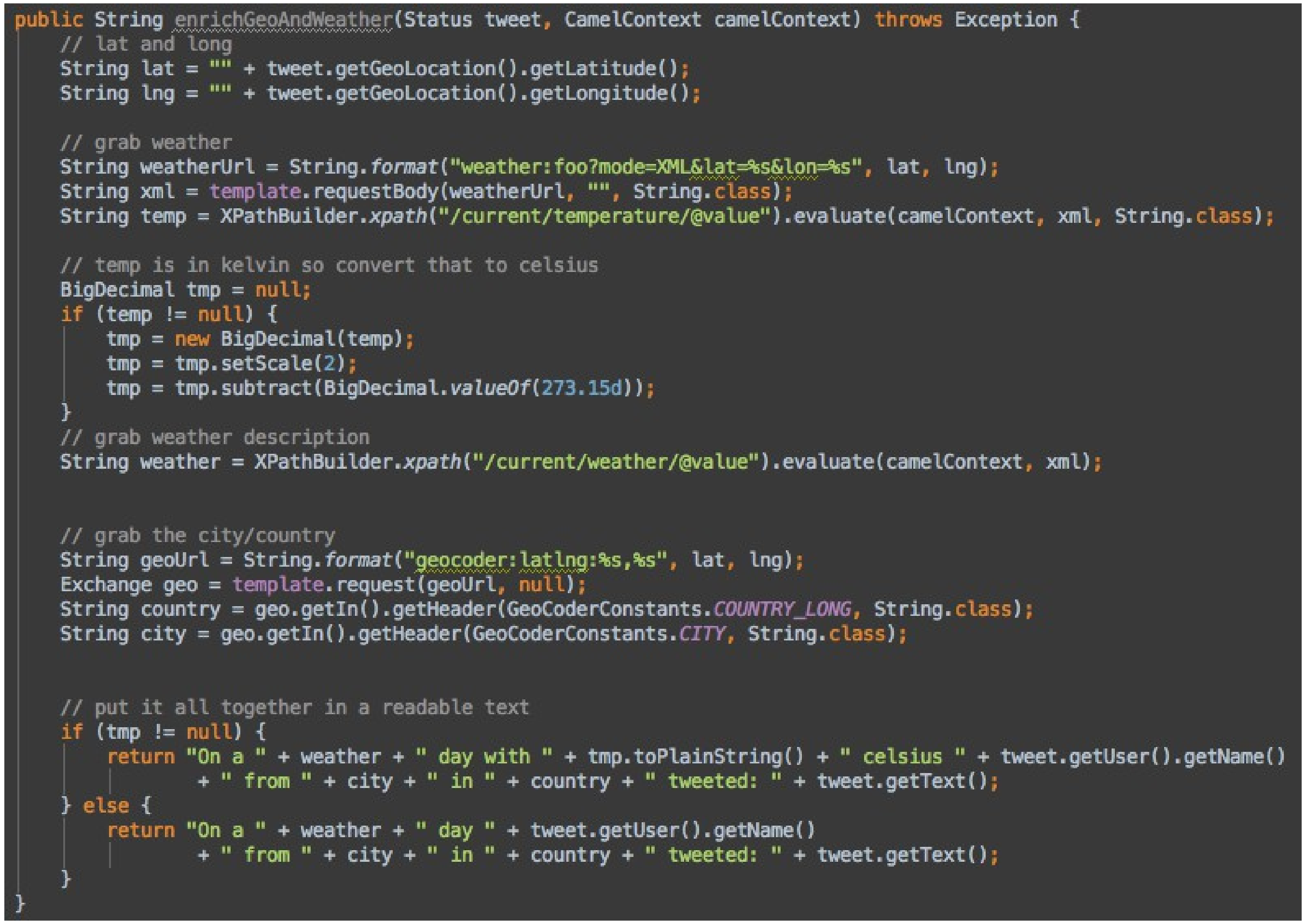
- The Camel route ...



```
// poll twitter search for new tweets
fromF("twitter://search?type=polling&delay=%s&keywords=%s", delay, searchTerm)
    // and push tweets to all web socket subscribers on camel-tweet
    .choice()
        .when().method(this, "hasGeoCodes")
            .bean(this, "enrichGeoAndWeather")
            .log("=============================================================================")
            .log(">>> ${body}")
            .log("=============================================================================")
        .otherwise()
            .transform().simple("${body.user.name} tweeted: ${body.text}")
    .end()
    .to("websocket:camel-tweet?sendToAll=true");
```

Fun with our new component

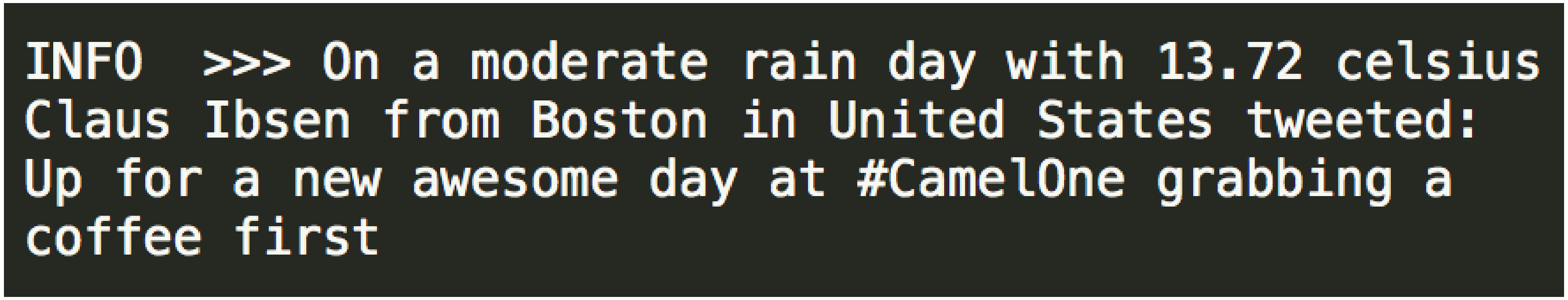
- Grabbing the data



```java
public String enrichGeoAndWeather(Status tweet, CamelContext camelContext) throws Exception {
    // lat and long
    String lat = "" + tweet.getGeoLocation().getLatitude();
    String lng = "" + tweet.getGeoLocation().getLongitude();

    // grab weather
    String weatherUrl = String.format("weather:foo?mode=XML&lat=%s&lon=%s", lat, lng);
    String xml = template.requestBody(weatherUrl, "", String.class);
    String temp = XPathBuilder.xpath("/current/temperature/@value").evaluate(camelContext, xml, String.class);

    // temp is in kelvin so convert that to celsius
    BigDecimal tmp = null;
    if (temp != null) {
        tmp = new BigDecimal(temp);
        tmp = tmp.setScale(2);
        tmp = tmp.subtract(BigDecimal.valueOf(273.15d));
    }
    // grab weather description
    String weather = XPathBuilder.xpath("/current/weather/@value").evaluate(camelContext, xml);


    // grab the city/country
    String geoUrl = String.format("geocoder:latlng:%s,%s", lat, lng);
    Exchange geo = template.request(geoUrl, null);
    String country = geo.getIn().getHeader(GeoCoderConstants.COUNTRY_LONG, String.class);
    String city = geo.getIn().getHeader(GeoCoderConstants.CITY, String.class);


    // put it all together in a readable text
    if (tmp != null) {
        return "On a " + weather + " day with " + tmp.toPlainString() + " celsius " + tweet.getUser().getName()
                + " from " + city + " in " + country + " tweeted: " + tweet.getText();
    } else {
        return "On a " + weather + " day " + tweet.getUser().getName()
                + " from " + city + " in " + country + " tweeted: " + tweet.getText();
    }
}
```

Fun with our new component

- And an example tweet



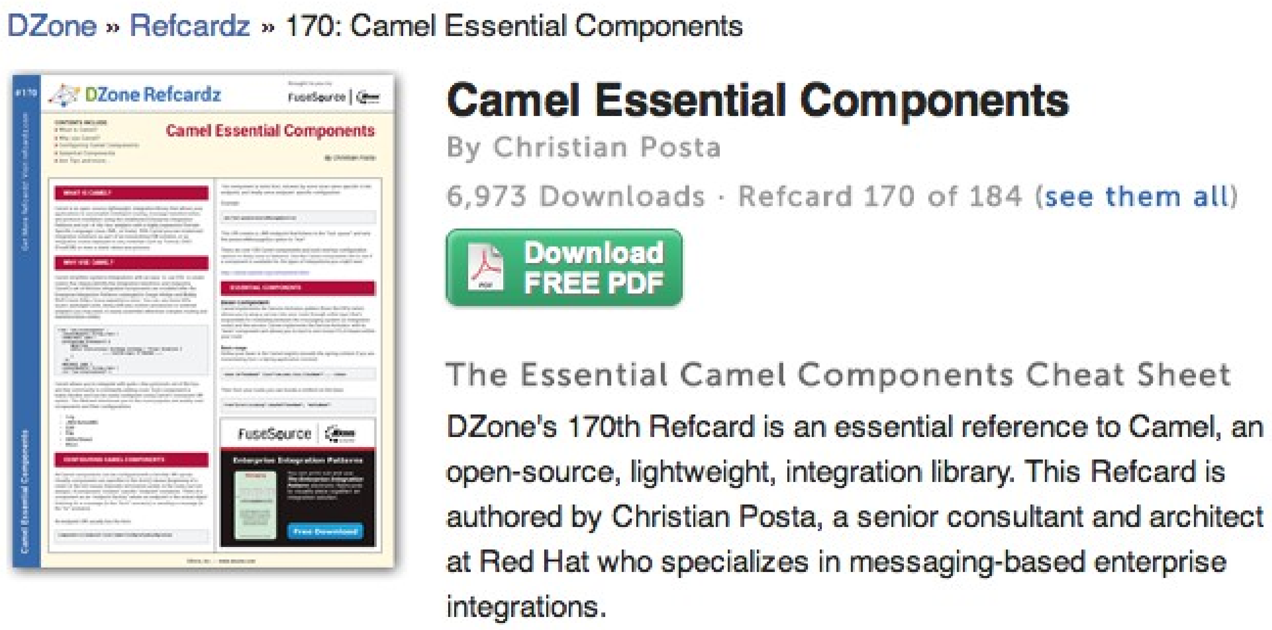
```
INFO  >>> On a moderate rain day with 13.72 celsius
Claus Ibsen from Boston in United States tweeted:
Up for a new awesome day at #CamelOne grabbing a
coffee first
```

Agenda

- A little Example
- Understanding Components
- Essential Components
- Creating new Components
- **Q and A**

Where do I get more information?

- Camel Essential Components Reference Card



DZone » Refcardz » 170: Camel Essential Components

Camel Essential Components

By Christian Posta

6,973 Downloads · Refcard 170 of 184 (see them all)

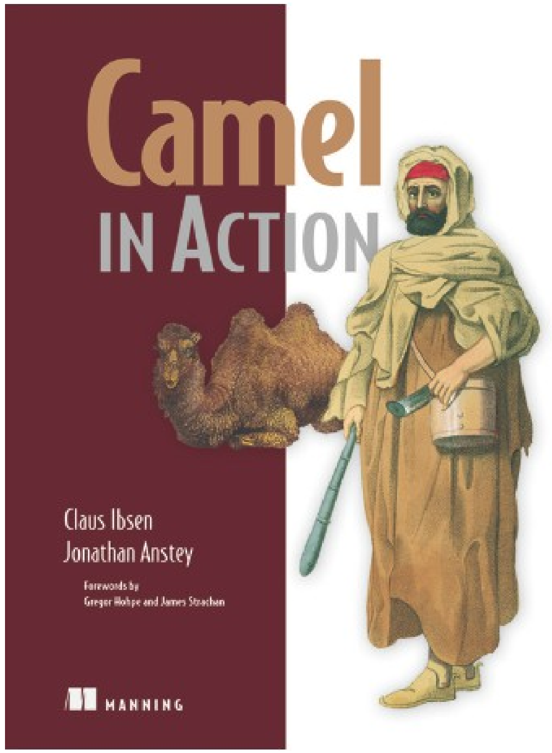
The Essential Camel Components Cheat Sheet

DZone's 170th Refcard is an essential reference to Camel, an open-source, lightweight, integration library. This Refcard is authored by Christian Posta, a senior consultant and architect at Red Hat who specializes in messaging-based enterprise integrations.

http://refcardz.dzone.com/refcardz/essential-camel-components

Where do I get more information?

- Buy the Camel in Action book



http://manning.com/ibsen/

Use code ...
camel40
… for 40% discount

Any Questions ?

- Contact
  - EMail: cibsen@redhat.com
  - Twitter: @davsclaus
  - Blog: http://davsclaus.com
  - Linkedin: http://www.linkedin.com/in/davsclaus