Getting started with Apache Camel

by Claus Ibsen
Principal Software Engineer, FuseSource
@davsclaus
Agenda

- What is Apache Camel?
- Basic Camel Concepts
- Running Camel
- What's 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
What is Apache Camel?

- Quote from the web site
  - [http://camel.apache.org](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

```xml
<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

```java
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

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

use file instead
What is Apache Camel?

- Endpoints as URIs

```java
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

![Diagram of Camel Architecture]

- Routing engine: A DSL wires endpoints and processors together to form routes.
- CamelContext: Contains routes and processors.
- Route 1, Route 2, Route N: Define routes with various processors.
- Message filter processor: Handle things in between endpoints like EIPs, Routing, Transformation, Mediation, Enrichment, Validation, Interception.
- Content-based router processor: Provide a uniform endpoint interface.
- Components: File, JMS, HTTP, provide a uniform endpoint interface and connect to other systems.
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

Sender (Producer)  Message  Receiver (Consumer)
Basic Camel Concepts

- What is a Message?

```java
org.apache.camel.Message
```

![Message Diagram](image)
Basic Camel Concepts

- ... And Message is contained in an Exchange

![Diagram of 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

```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();
    }
}
```

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

1. Routes files from inbox to outbox
Basic Camel Concepts

- File Copier Example

```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();
  }
}
```
Basic Camel Concepts

- 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();
    }
}
```
Basic Camel Concepts

- File Copier Example

```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();
    }
}
```
Basic Camel Concepts

- File Copier Example

```
Consumer
from("file:data/inbox...")
creates
Message
process
Producer
to("file:data/outbox")
```
Basic Camel Concepts

- File Copier w/ Transformation Example

  ![Diagram of File Copier w/ Transformation Example]

  ```
  Consumer
  from("file:data/inbox...")
  Message
  Processor
  Changed Message
  Producer
  to("file:data/outbox")
  ```

- Pipes and Filters EIP

  ![Diagram of Pipes and Filters EIP]

  ```
  Consumer
  input
  Processor
  output
  input
  Producer
  ```
Dissecting the DSL

- Demystifying the "woodoo DSL"

```
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"

```java
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"

```java
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
Agenda

- What is Apache Camel?
- Basic Camel Concepts
- Running Camel
- What's 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

```java
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)

```java
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
- *What's 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
What's included in the box?

- 50 Enterprise Integration Patterns

http://camel.apache.org/eip
Whats included in the box?

- Splitter EIP

```java
from("file:inbox")
  .split(body().tokenize("\n"))
  .marshal(customToXml)
  .to("activemq:line");
```
Whats included in the box?

- **100 Components**

<table>
<thead>
<tr>
<th>activemq</th>
<th>cxf</th>
<th>flatpack</th>
<th>jasypt</th>
</tr>
</thead>
<tbody>
<tr>
<td>activemq-journal</td>
<td>cxfrs</td>
<td>freemarker</td>
<td>javaspace</td>
</tr>
<tr>
<td>amqp</td>
<td>dataset</td>
<td>ftp/ftps/sftp</td>
<td>jbi</td>
</tr>
<tr>
<td>atom</td>
<td>db4o</td>
<td>gae</td>
<td>jcr</td>
</tr>
<tr>
<td>bean</td>
<td>direct</td>
<td>hdfs</td>
<td>jjdbc</td>
</tr>
<tr>
<td>bean validation</td>
<td>ejb</td>
<td>hibernate</td>
<td>jetty</td>
</tr>
<tr>
<td>browse</td>
<td>esper</td>
<td>hl7</td>
<td>jms</td>
</tr>
<tr>
<td>cache</td>
<td>event</td>
<td>http</td>
<td>jmx</td>
</tr>
<tr>
<td>cometd</td>
<td>exec</td>
<td>ibatis</td>
<td>jpa</td>
</tr>
<tr>
<td>crypto</td>
<td>file</td>
<td>irc</td>
<td>jt/400</td>
</tr>
</tbody>
</table>

http://camel.apache.org/components.html
**Whats included in the box?**

- **100 Components**

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

[http://camel.apache.org/components.html](http://camel.apache.org/components.html)
Whats included in the box?

- 22 Data Formats

<table>
<thead>
<tr>
<th>Data Format</th>
<th>Data Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>bindy</td>
<td>json</td>
</tr>
<tr>
<td>beanio</td>
<td>protobuf</td>
</tr>
<tr>
<td>castor</td>
<td>rss</td>
</tr>
<tr>
<td>csv</td>
<td>serialization</td>
</tr>
<tr>
<td>crypto</td>
<td>soap</td>
</tr>
<tr>
<td>dozer</td>
<td>syslog</td>
</tr>
<tr>
<td>flatpack</td>
<td>tidy markup</td>
</tr>
<tr>
<td>gzip</td>
<td>xml beans</td>
</tr>
<tr>
<td>hl7</td>
<td>xml security</td>
</tr>
<tr>
<td>jaxb</td>
<td>xstream</td>
</tr>
<tr>
<td>jibx</td>
<td>zip</td>
</tr>
</tbody>
</table>

http://camel.apache.org/data-format.html
Whats included in the box?

- 16 Expression Languages

| 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
Whatis included in the box?

- DSL in multiple flavors

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

```java
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

```java
errorHandler(
    deadLetterChannel("activemq:error")
);

from("activemq:incoming")
  .marshal().jaxb()
  .to("mq:QueueWithXmlMessages");
```
What's included in the box?

- Error Handling
  - Dead Letter Channel EIP w/ Redelivery

```java
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)

```java
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

  \texttt{mvn archetype:generate}

- Or from Fuse IDE and/or Eclipse
### 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**

```xml
<Blueprint xmlns="http://www.osgi/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/xmlns/blueprint/v1.0.0 http://www.osgi/xmlns/blueprint/v1.0.0/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"/>
      </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

```java
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
    Bundle ID: 214
    FuseESB:karaf@root> osgi:start 214

- Using Fuse IDE

![Fuse IDE screenshot showing deploy options]

Copyright © 2012 FuseSource Corp. All rights reserved.
Agenda

- What is Apache Camel?
- Basic Camel Concepts
- Running Camel
- What's 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

```java
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

```java
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

```java
public class HelloWorldConsumer extends ScheduledPollConsumer {
    private final HelloWorldEndpoint endpoint;

    public HelloWorldConsumer(HelloWorldEndpoint endpoint, Processor processor) {
        super(endpoint, processor);
        this.endpoint = endpoint;
    }

    @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
    }
}
```

scheduled consumer
logic
Extending Camel

- **HelloWorldProducer**

```java
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);
    }
}
```
Extending Camel

- **HelloWorldComponentTest**

```java
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
• 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

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/mailing-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)

http://manning.com/ibsen
Q and A

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