in the belly of the
Brought to you by...

Henryk Konsek
@hekonsek

Engineer at Red Hat (JBoss). Open source hacker (JBoss Fabric8, Apache Camel, JBoss Hawt.io). Evangelist of the open source software that rocks.

Poland

about.me/hekonsek
What is... Server for running and managing Linux containers.
What are Linux containers?

Para-virtualized Linux instances.
Why not regular virtualization?

- sloooooooow
- gigantic images
- aggressive resource allocation
- bad API
Key concepts

- image (immutable, no state)
- container (has state)

*Container* is the running *image*. 
Docker awesomeness #1

Commands:
Docker awesomeness #2

<table>
<thead>
<tr>
<th>Layers</th>
</tr>
</thead>
<tbody>
<tr>
<td>WAR</td>
</tr>
<tr>
<td>Tomcat</td>
</tr>
<tr>
<td>JRE</td>
</tr>
<tr>
<td>Ubuntu base</td>
</tr>
</tbody>
</table>
Docker awesomeness #3

Registries.
Docker awesomeness #4

Build once. Deploy everywhere!
Dockerfiles

FROM ubuntu
EXPOSE 8080
RUN apt-get install java
RUN mkdir /jars
ADD target/app.jar /jars/
CMD ["java", "-jar", "/jars/app.jar"]

“Recipes” for the new images.
Create new image in the local repo

$ docker build -t com.me/app:1.0
$ docker run -t com.me/app:1.0
Apache Camel

Framework that routes messages like crazy.
How to deploy Camel?

- Karaf (JBoss Fuse, ServiceMix, Talend ESB)
- Tomcat
- WildFly (JBoss EAP)
- standalone/embedded
- Akka plugin
- Grails plugin
- Spring Boot
Camel and Docker

How to split it? What should I dockerize?
Messaging architecture in a nutshell

1. In endpoint (HTTP, REST, SOAP, FTP, ...)
2. JMS broker (AMQ, HornetQ, ...)
3. Message store (LevelDB, KahaDB, JDBC, ...)
4. Processor (database, FTP, HTTP, ...)
5. Out endpoint (database, FTP, HTTP, ...)

---

1. In endpoint (HTTP, REST, SOAP, FTP, ...)
2. JMS broker (AMQ, HornetQ, ...)
3. Message store (LevelDB, KahaDB, JDBC, ...)
4. Processor (database, FTP, HTTP, ...)
5. Out endpoint (database, FTP, HTTP, ...)
Concrete messaging architecture

Netty HTTP server (submit invoice) → ActiveMQ broker → Business logic (calculate gross value) → MongoDB (persist enriched invoice)

LevelDB

DOCKERIZE'EM!
I DON'T DOCKERIZE OFTEN

BUT WHEN I DO, I DOCKERIZE EVERYTHING
In-endpoint route

```java
from("netty-http:http://0.0.0.0:18080").
    setBody(randomUUID()).
    inOnly("jms:invoices");

...

ew ActiveMQConnectionFactory("tcp://amqbroker:6162")
```
Processing route

from("jms:invoices").
    setBody().
    groovy("new Invoice(request.body,currentTimeMillis())")
    to("mongodb:mongo?option=insert");

...

new ActiveMQConnectionFactory("tcp://amqbroker:6162");
...
new MongoClient("mongodb");
How can I get database images?

docker run -d -p 27017:27017
--name mongodb dockerfile/mongodb

Provided by database community/vendor.
How can I put fresh jar into image?

Docker Maven plugin by Roland ‘Jolokia’ Huß
How to bootstrap Camel?

- no Karaf bundle activators
- no server (Tomcat, etc.)
- how can we start CamelContext?
Custom class with the main method
Spring Boot for Camel

http://projects.spring.io/spring-boot
Camel + Spring Boot: step #1

Take a Spring Boot fat jar.
Camel + Spring Boot: step #2

Add camel-spring-boot jar to your classpath.
Camel + Spring Boot: step #3

Add Camel route to your classpath.
Camel + Spring Boot: step #4

Dockerize your fat jar and run it!
ENV-centric runtime configuration

# override endpoint definition via ENV variable
docker run -e FROM=jms:queue -it my-springboot-camel-app

# run with the given Spring profile
docker run -e spring.profiles.active=production -it my-springboot-camel-app
Expoze JMX via REST with the Jolokia base image.
Kubernetes

- orchestration of many Docker containers
- ...and many Docker servers!
- logical container groups (pods)
- auto-scaling
- wiring your Docker stuff together
Thank you!

THIS IS DEMO!!!