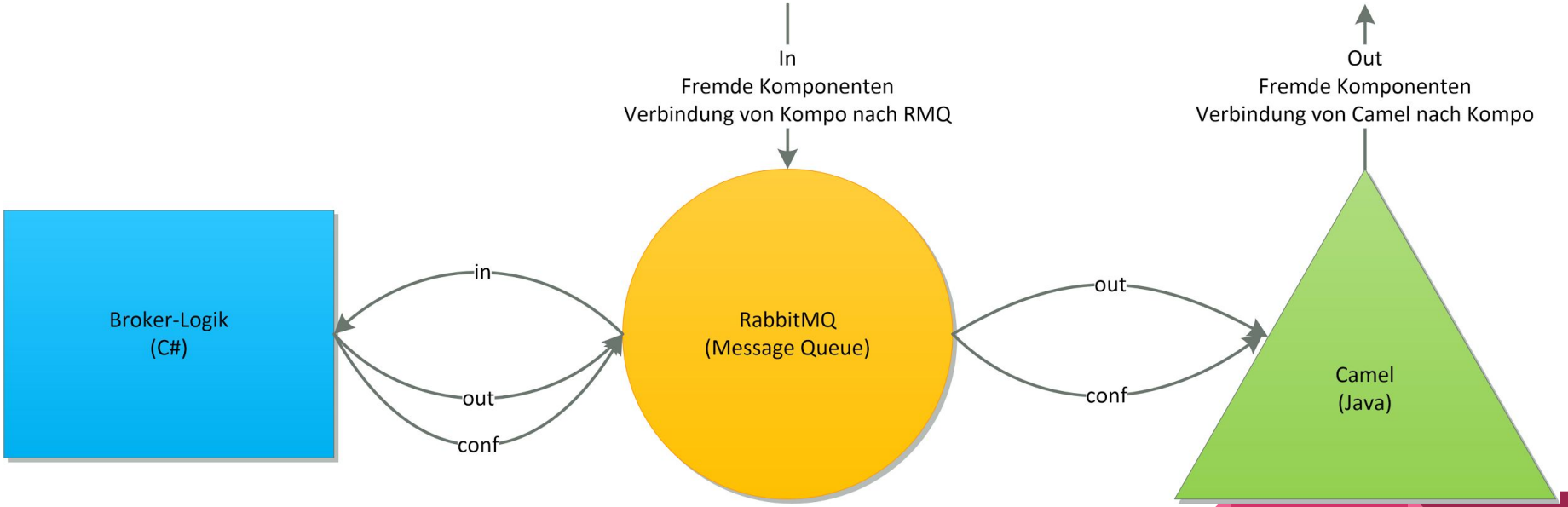
Broker (“Team Zoo”)

Hasen, Kamele, Wale und mehr...

UTR WS 2016/17, Kompotech: Fillipe Antunes Almeida, Jens Bruchertseifer, Alexander Landmesser, Nico Maas

Inhaltsverzeichnis

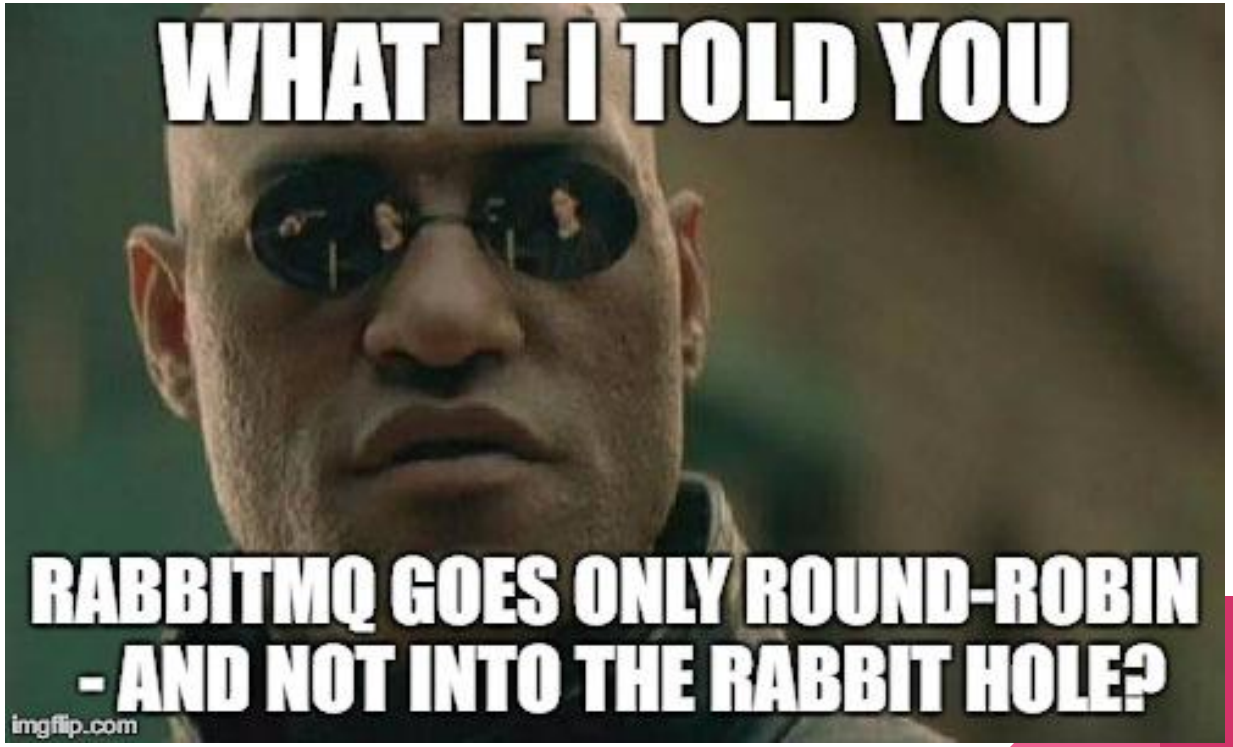
Einführung


In
Fremde Komponenten
Verbindung von Kompo nach RMQ
Out
Fremde Komponenten
Verbindung von Camel nach Kompo
Broker-Logik
(C#)
in
out
conf
RabbitMQ
(Message Queue)
out
conf
Camel
(Java)

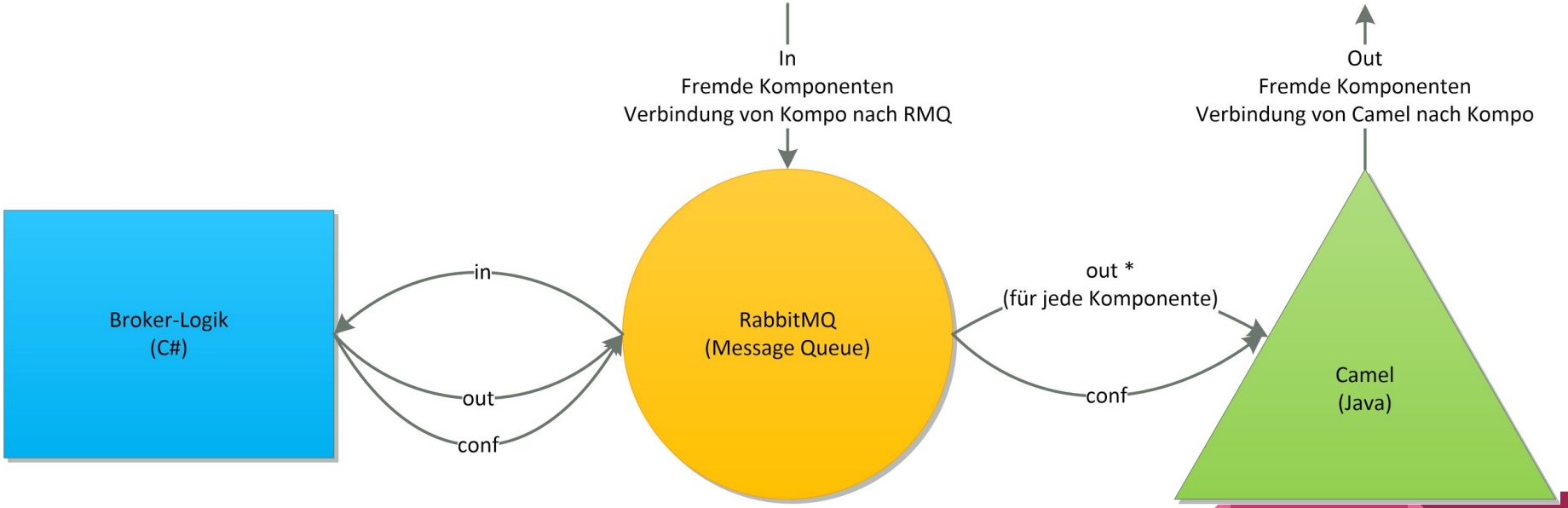
Einführung


RabbitMQ uses Round-Robin!
Camel is confused!
Camel attacks itself!

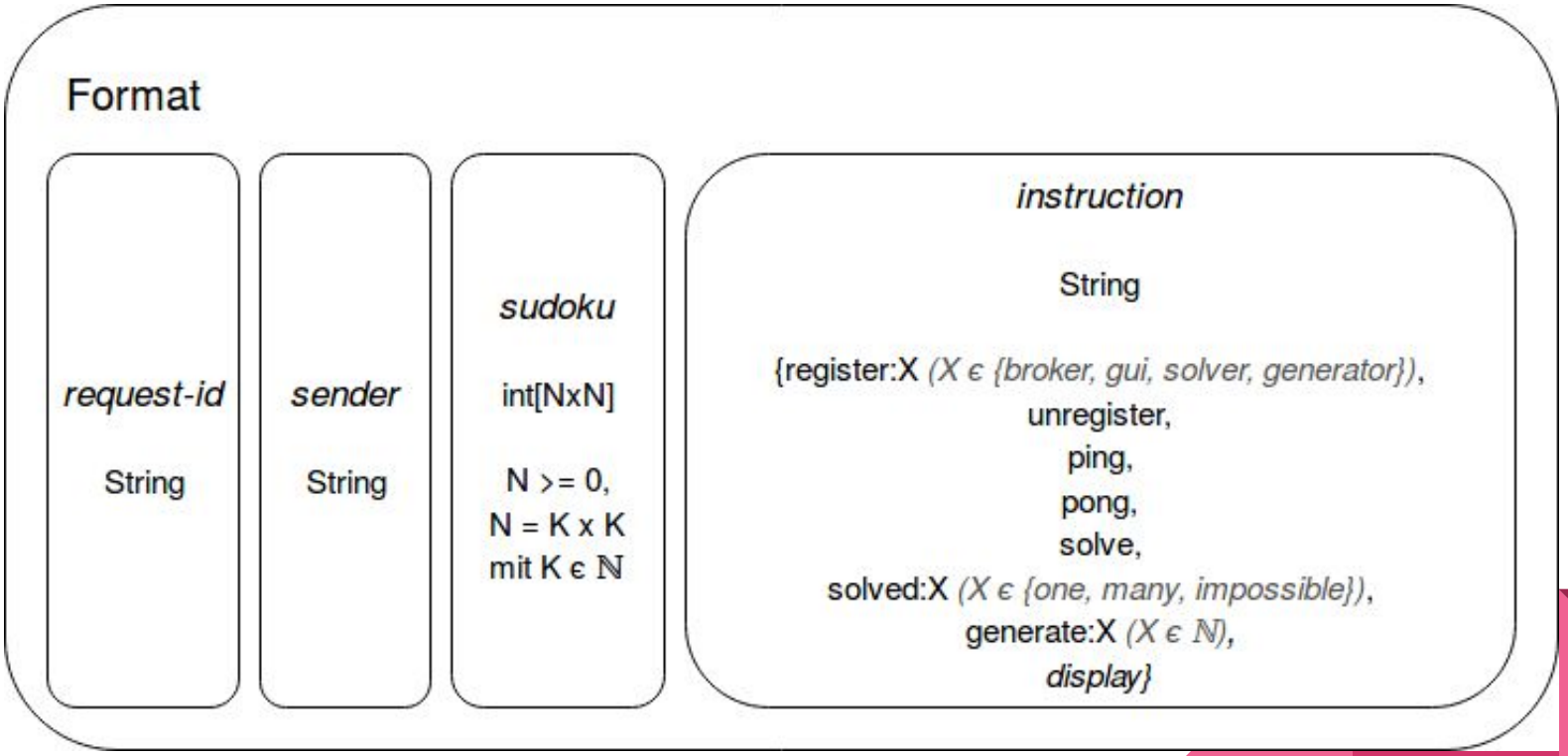
Einführung


WHAT IF I TOLD YOU
RABBITMQ GOES ONLY ROUND-ROBIN
- AND NOT INTO THE RABBIT HOLE?
imgflip.com

Einführung


In
Fremde Komponenten
Verbindung von Kompo nach RMQ
Out
Fremde Komponenten
Verbindung von Camel nach Kompo
Broker-Logik
(C#)
in
out
conf
RabbitMQ
(Message Queue)
out *
(für jede Komponente)
conf
Camel
(Java)

Nachrichtenformat

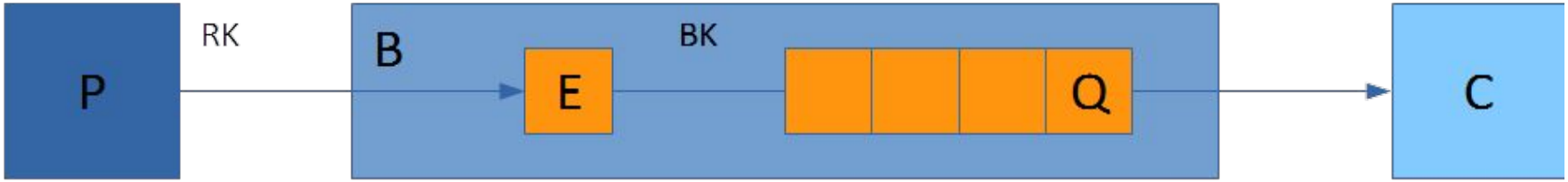


RabbitMQ

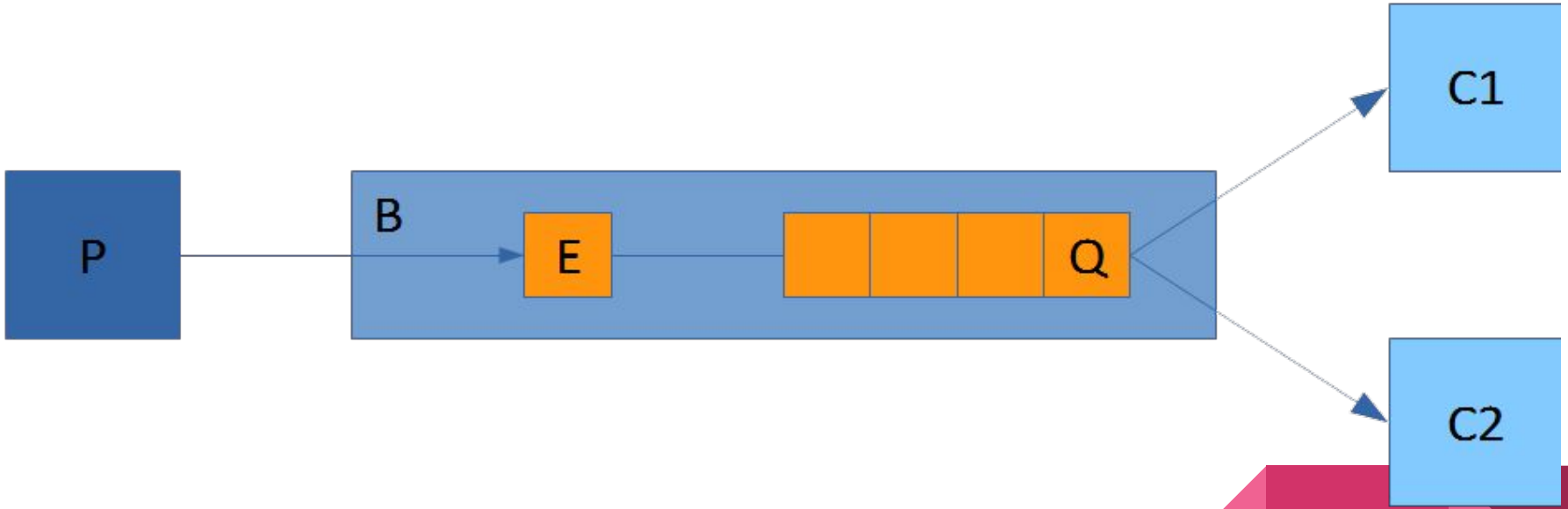
Was ist RabbitMQ?

- Open-Source Message Broker
- AMQP - Advanced Messaging and Queuing Protokoll
- In Ehrlang geschrieben
- Anwendungsbereich
  - Instagram Feeds, Daimler, Bosch, ...
- Studie mit Google: 1 Mio. Anfragen pro Sekunde
  - 86,4 Milliarden Nachrichten Pro Tag

Aufbau



Load Balancing



Reliability

- Verbindungsabbrüche
  - At-most-once Delivery
    - Nachrichten können verloren gehen
  - At-least-once Delivery
    - Verwendung von Acknowledgements (und Confirms)
    - Redelivery Flag
- System-Neustart
  - Durable, Persistent Flags
- Hardware-Ausfall
  - Cluster

Java Beispiel

Publisher:

```
ConnectionFactory factory = new ConnectionFactory();

factory.setHost(Host.IP_ADDRESS);
factory.setUsername(Host.USER_NAME);
factory.setPassword(Host.USER_PASSWORD);

Connection connection = factory.newConnection();
Channel channel = connection.createChannel();

channel.queueDeclare(QUEUE_NAME, false, false, false, null);
String message = "message";
channel.basicPublish("", QUEUE_NAME, null, message.getBytes());

channel.close();
connection.close();
```

Consumer:

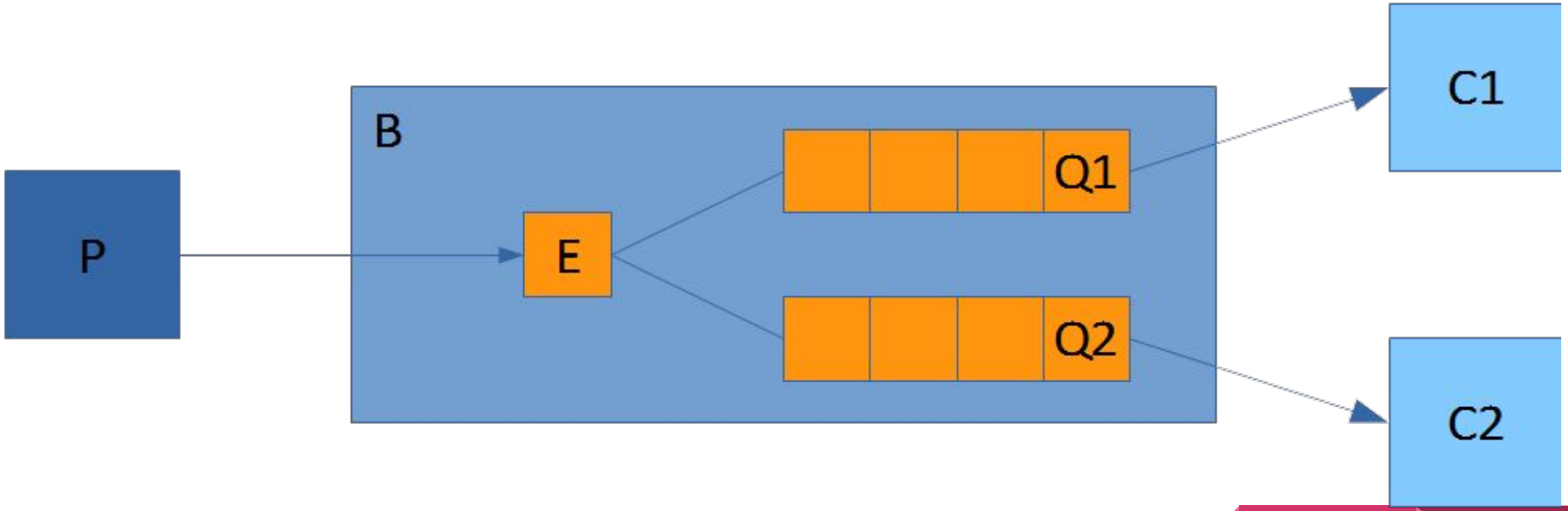
```
ConnectionFactory factory = new ConnectionFactory();

factory.setHost(Host.IP_ADDRESS);
factory.setUsername(Host.USER_NAME);
factory.setPassword(Host.USER_PASSWORD);

Connection connection = factory.newConnection();
Channel channel = connection.createChannel();

channel.queueDeclare(QUEUE_NAME, false, false, false, null);

Consumer consumer = new DefaultConsumer(channel);
channel.basicConsume(QUEUE_NAME, true, consumer);

channel.close();
connection.close();
```

Exchange-Typen

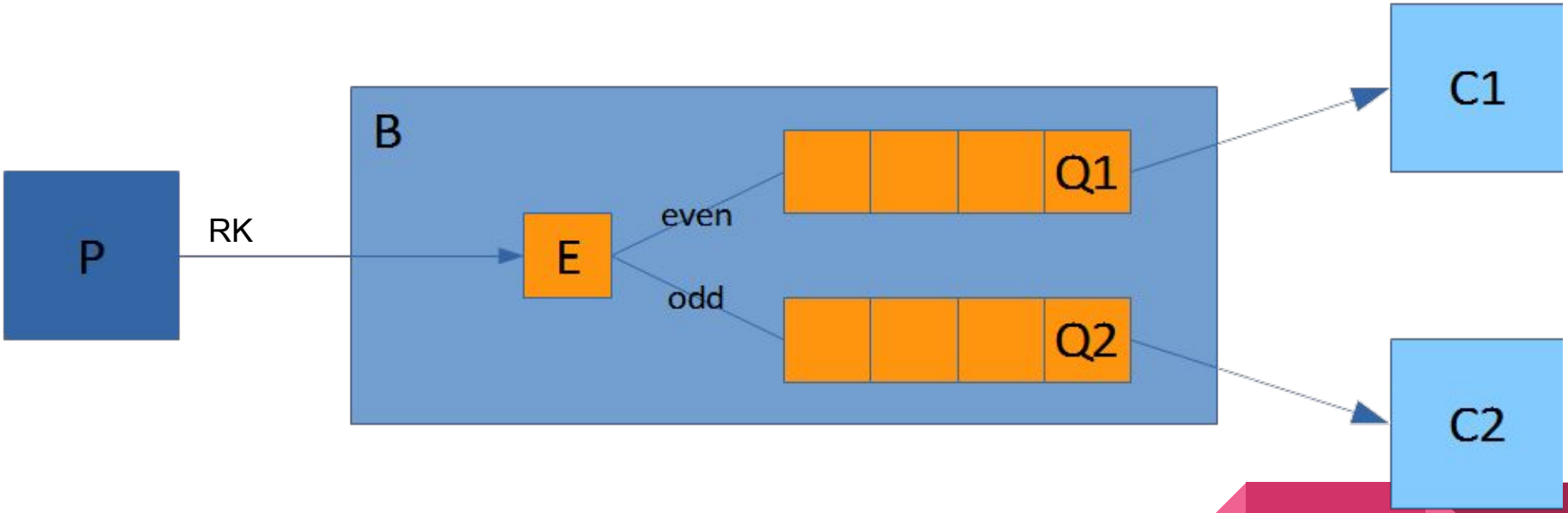
Exchange: Wohin mit den Nachrichten?

- fanout
- direct
- topic

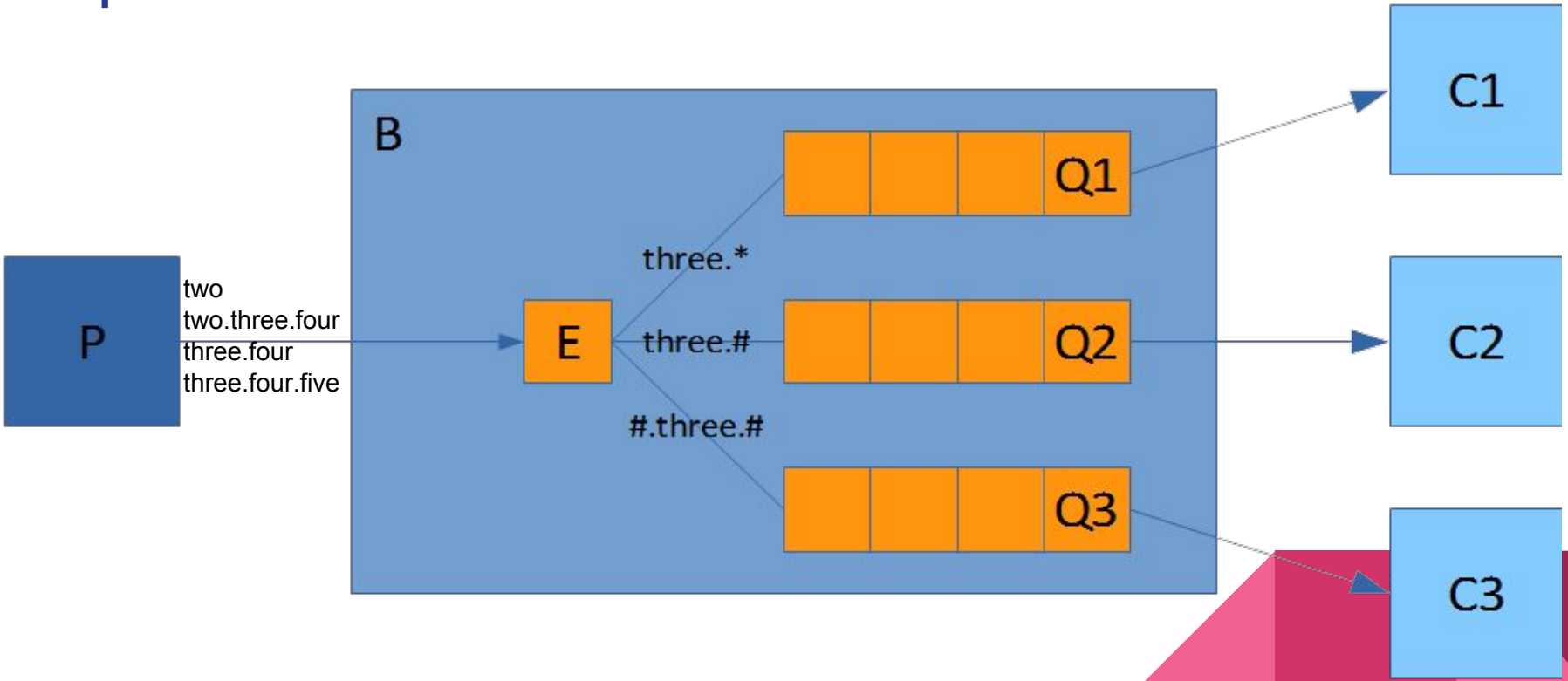
fanout



direct



topic


B
Q1
Q2
Q3
three.*
three.#
#.three.#
E
P
two
two.three.four
three.four
three.four.five
C1
C2
C3

Broker-Logik

- Welche Komponenten gibt es?
- Wer bekommt welche Nachrichten?
- Entfernen alter Komponenten
- Verhindern von Duplikaten

Nachrichten

- Registrierung
- Abmeldung
- Solve
- Solved
- Ping

Umwandlung JSON <-> Objekt

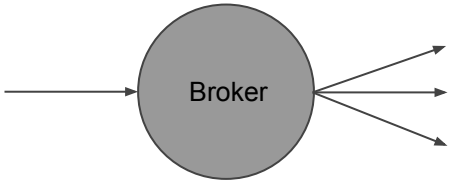
```
m = JsonConvert.DeserializeObject<Message>(message);
```

```
String json = JsonConvert.SerializeObject(m);
```

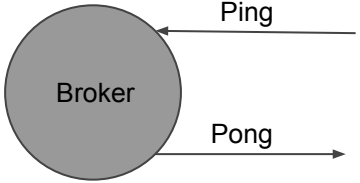
Ein einfacher Broadcast

- Liste angemeldeter Komponenten

id
Typ
URI



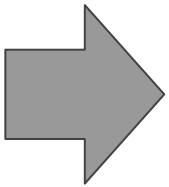
Ausnahme:



Schlauere Features

Nachrichtenverteilung

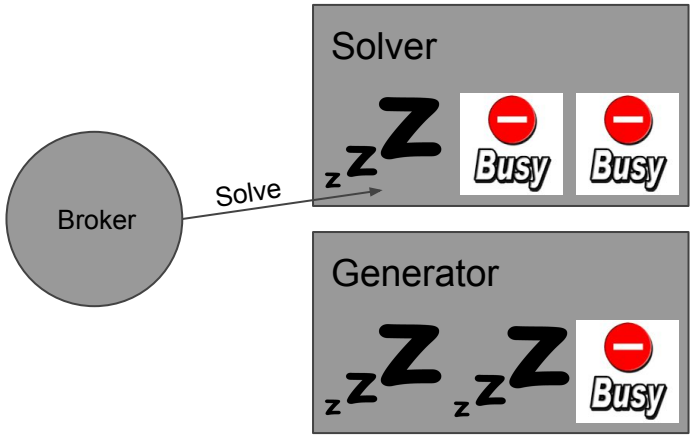
Broadcast



Unicast

Ungelöste	-> Solver
Gelöste	-> Generator
Alle	-> GUI

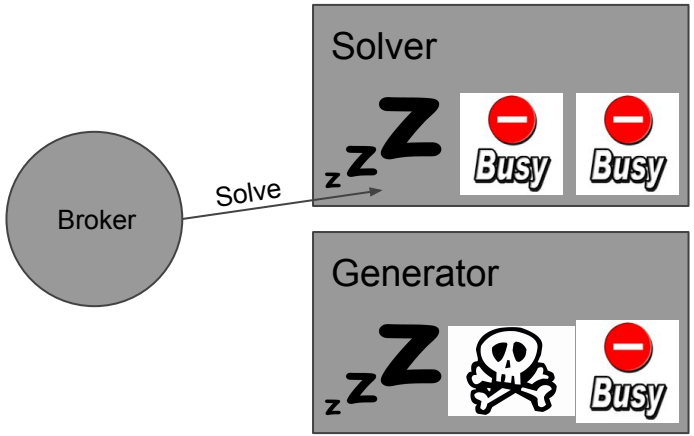
Performance



- Bevorzuge freie Komponenten
- Verwalte Zustand der Komponenten

Busy: Wenn eine Anfrage geschickt wurde und noch keine Antwort kam

Altlasten



Lösung:

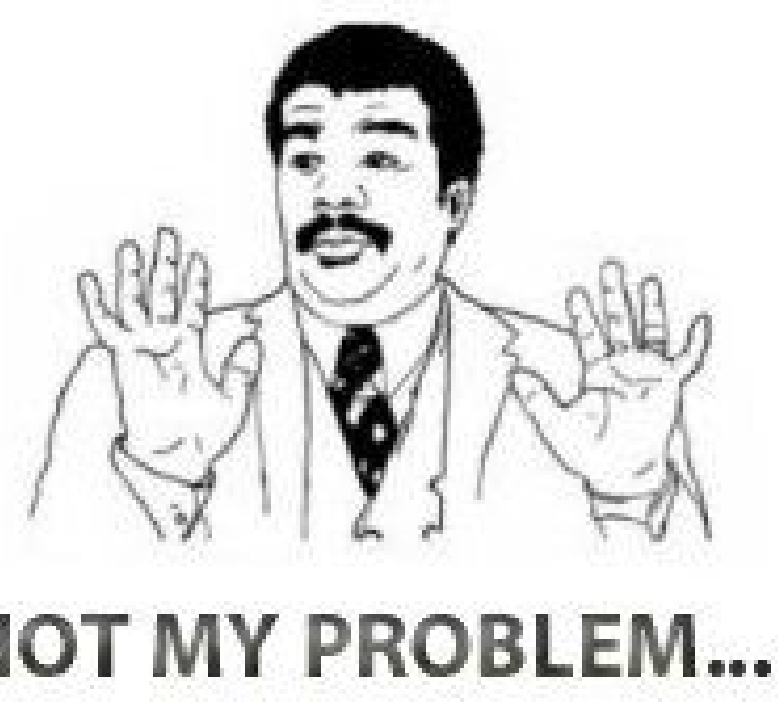
- Verwalte eine Liste mit dem letzten Kontakt
- Entferne Komponenten die sich 5 Minuten nicht gemeldet haben

Deduplikation

- Listen:
  - aktuell offene Anfragen
  - gelöste Sudokus
- MD5 Hash des angefragten Sudokus
- Sende Lösung bekannter Sudokus ohne Solver zu fragen

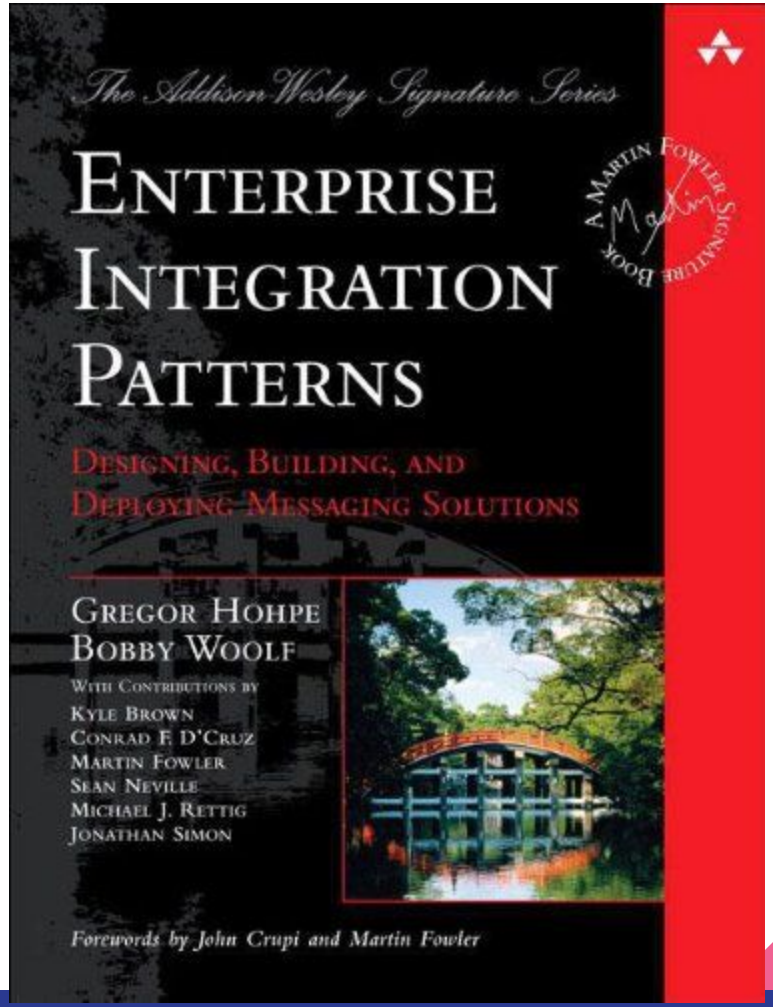
Sinnvolle Ergänzungen

Sinnvolle Ergänzungen




Apache
Camel

Camel???


The Addison Wesley Signature Series
A Martin Fowler Signature Book
Enterprise Integration Patterns
Designing, Building, and Deploying Messaging Solutions
Gregor Hohpe
Bobby Woolf
With Contributions by
Kyle Brown
Conrad F. D'Cruz
Martin Fowler
Sean Neville
Michael J. Rettig
Jonathan Simon
Forewords by John Crupi and Martin Fowler

```
from("file:data/in/requests")
.to("jms:queue:tickets")
```

```
<route>
<from uri="file:data/in/requests"/>
<to uri="jms:queue:tickets"/>
</route>
```

Warum Camel?

- Weil wir mussten ...

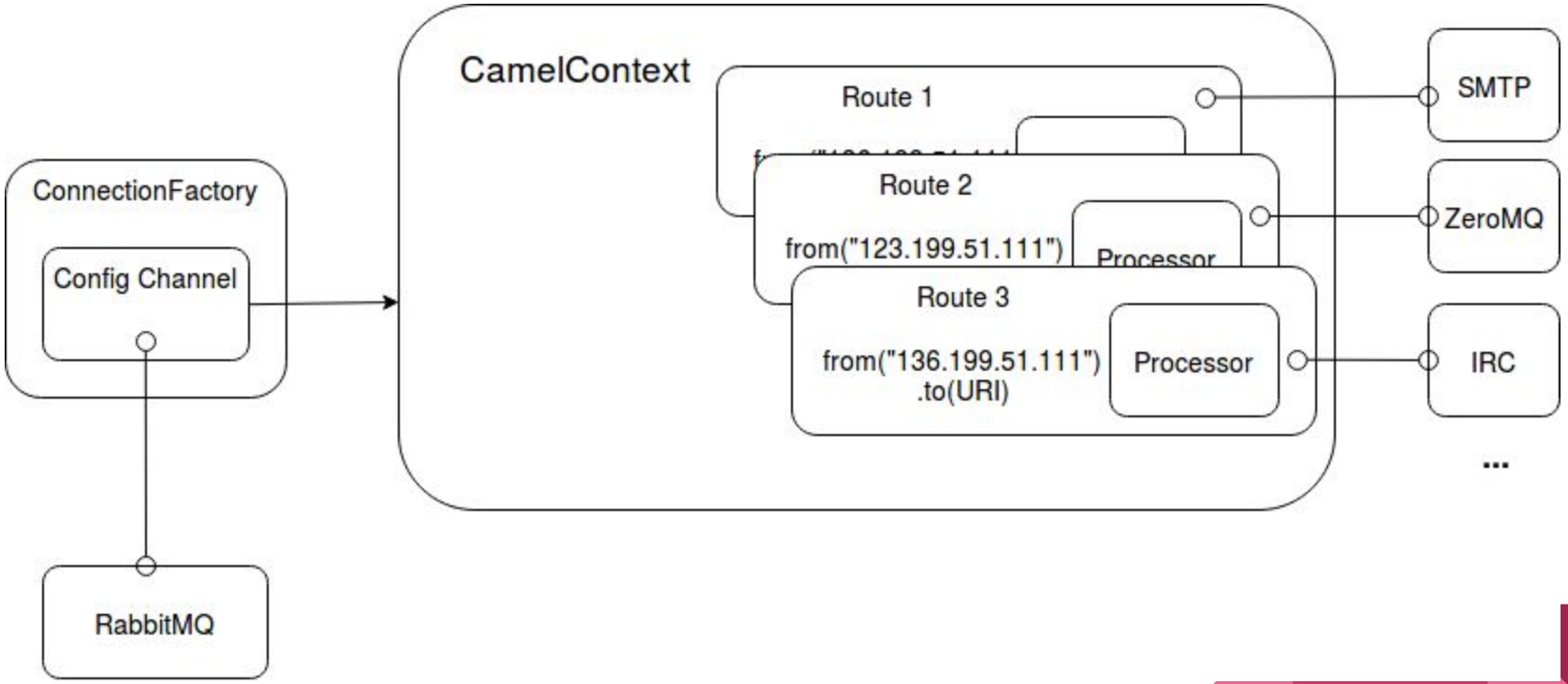
Warum Camel?

- Routing- und Vermittlungsengine
- Keine Annahmen über den Datentyp
- → kein einheitliches Datenformat notwendig
- → Higher-Level Abstraction
- Unterstützung von ca. 200 Protokollen und Datentypen
- Leichtgewichtiger Kern, modulare Architektur

Grenzen?

- Kein Enterprise Service Bus
- … aber manche ESBs beinhalten auch Camel (z. B. Apache Service Mix)
- Weniger geeignet für kleine Projekte mit sehr wenigen Technologien

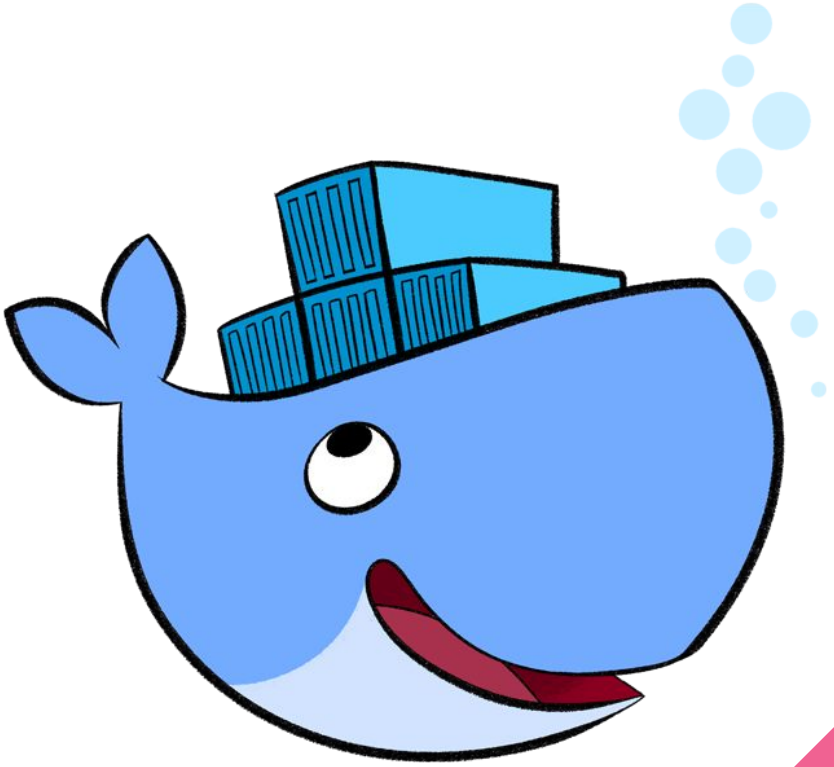
```
import org.apache.camel.CamelContext;
import org.apache.camel.builder.RouteBuilder;
import org.apache.camel.impl.DefaultCamelContext;
public class CamelExample {
	public static void main(String args[]) throws Exception {
		CamelContext context = new DefaultCamelContext();
		context.addRoutes(new RouteBuilder() {
			public void configure() {
				from("file:data/in/requests?noop=true")
				.id("1234")
				.process(new Processor() {
					@Override
					public void process(Exchange exchange)
							throws Exception {
						// Process Message here
					}
				}).to("jms:queue:tickets");
			}
		});
		context.start();
		Thread.sleep(10000);
		context.stop();
	}
}
```


CamelContext
Route 1
Route 2
from("123.199.51.111")
Processor
Route 3
from("136.199.51.111")
.to(URI)
Processor
ConnectionFactory
Config Channel
RabbitMQ
SMTP
ZeroMQ
IRC
...

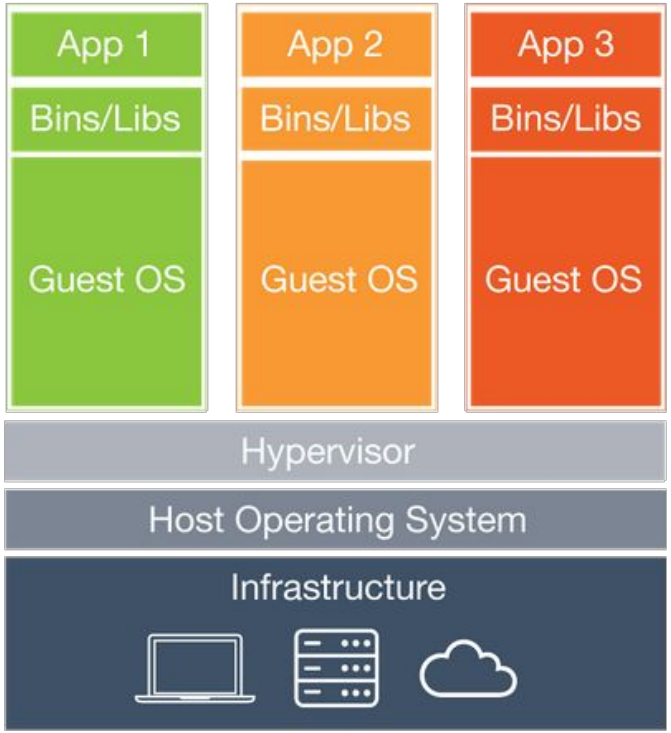
Docker

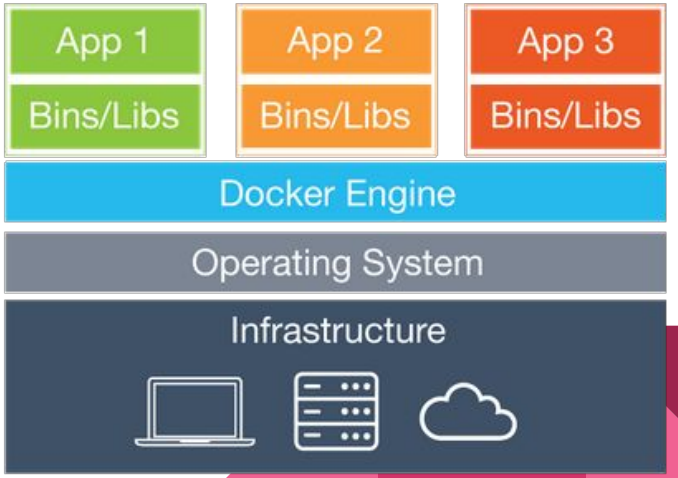

BRACE YOURSELF
THE DOCKERS BAND WAGON
IS COMING
quickmeme.com

Docker



Docker: VM vs Container

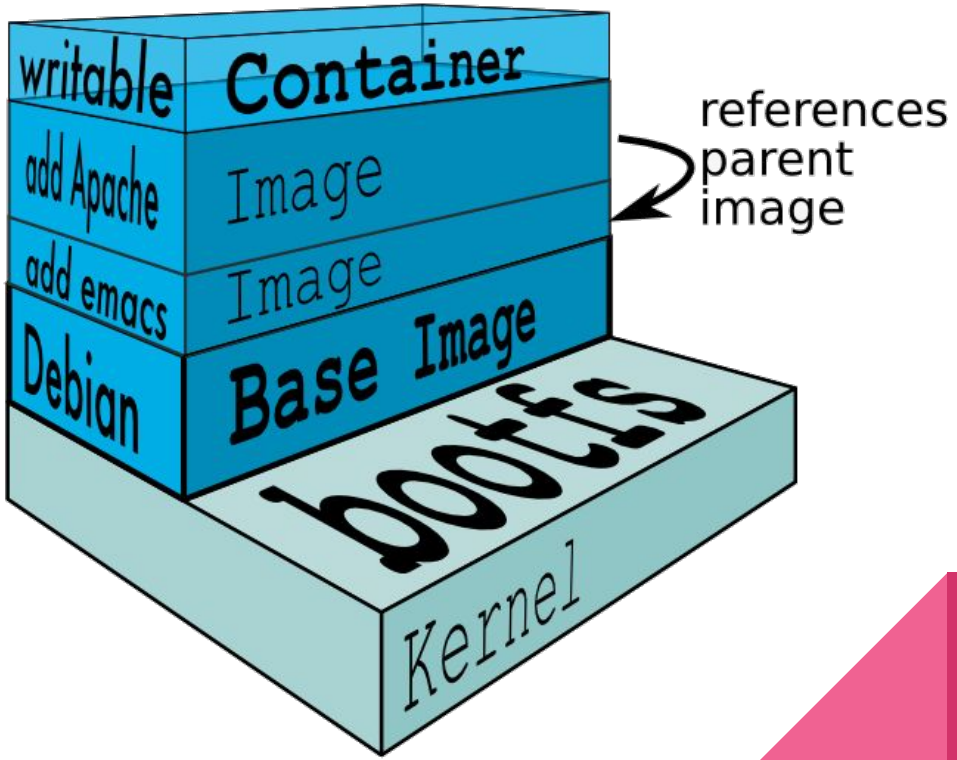




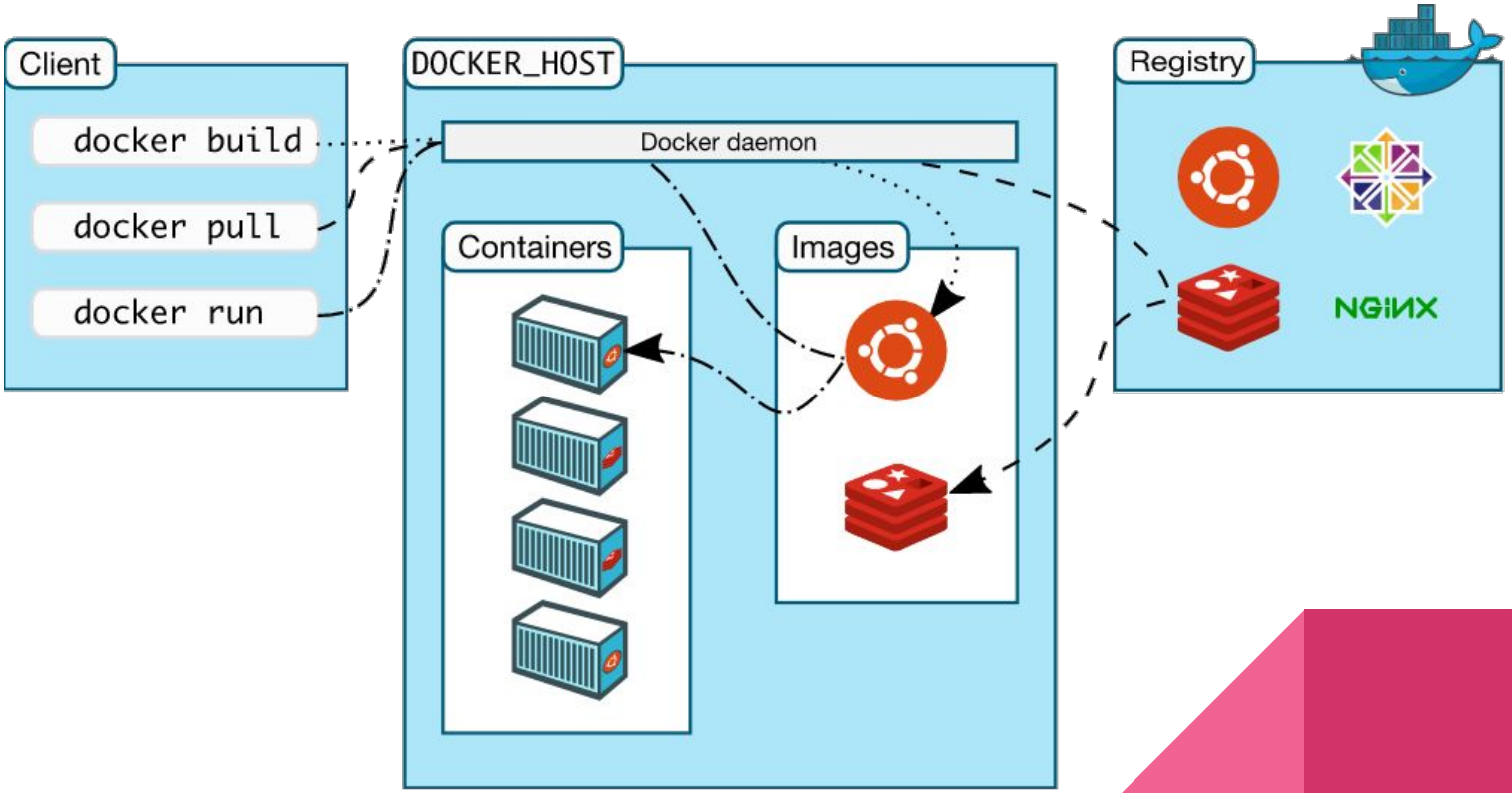
Docker: Matrix of Hell

Docker: Matrix of Hell

Docker: Image, Container und Layer



Docker: Architektur



Docker

- Docker Container werden wie immutable Objects verwendet
  - Alle States werden außerhalb gespeichert
  - Container werden nicht geupdated, sondern weggeworfen und neu gebaut
  - -> onbuild Images sind Gold wert!
- Dockerfile

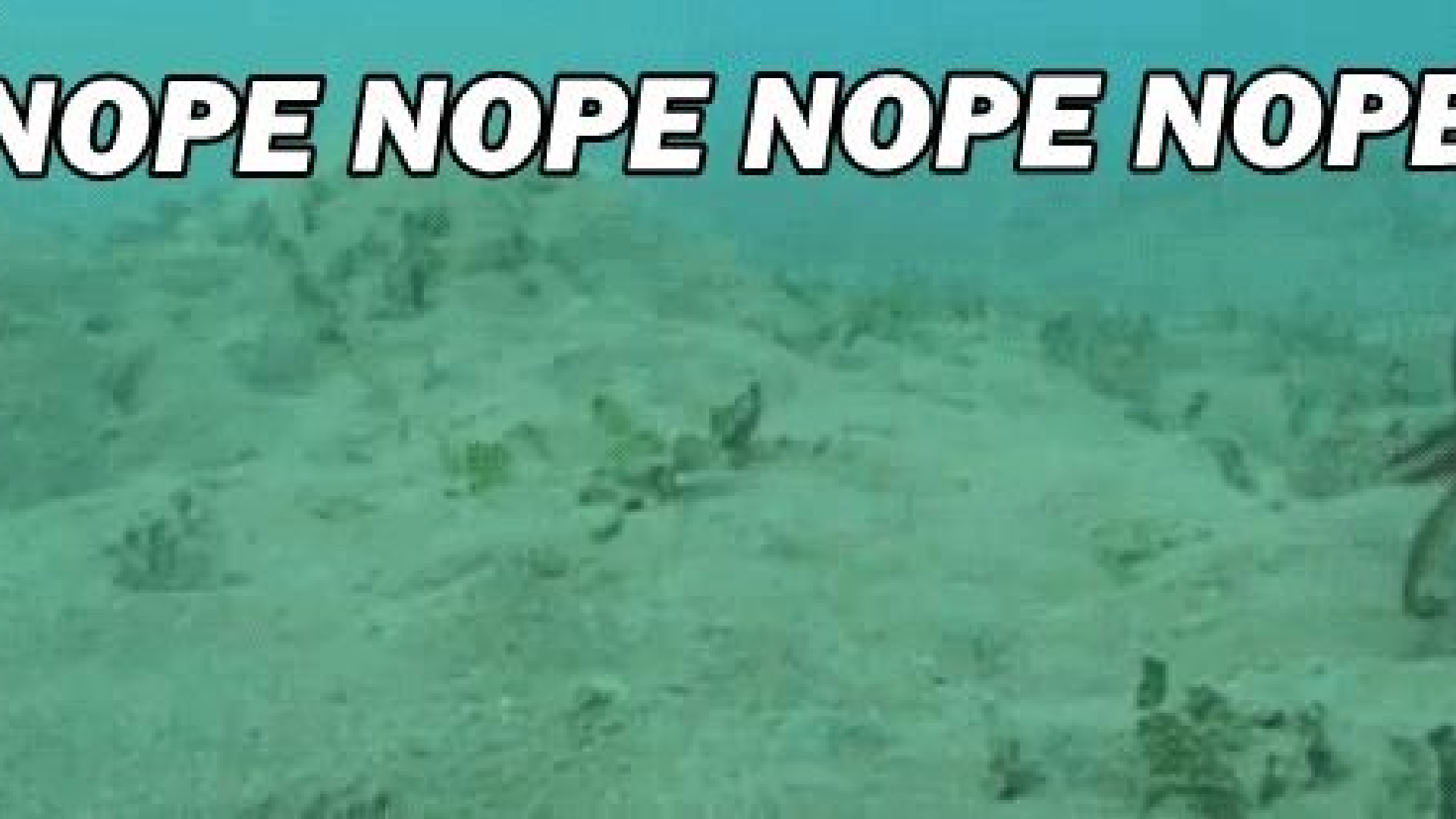
```
1 FROM maven:3.3-jdk-8-onbuild-alpine
2 CMD [ "java", "-jar", "target/CamelInstance-0.0.1-SNAPSHOT.jar" ]
```

- docker-compose

```
1 java:
2   container_name: broker_java
3   restart: unless-stopped
4   build: .
```

- Also total easy?!

JUST DO IT!


NOPE NOPE NOPE NOPE


JUST
DO IT!

Docker

- Dockern ohne zu wissen was man da eigentlich gerade docker-t?

Docker und RabbitMQ

- RabbitMQ ist vergesslicher als ein Goldfisch!
- docker-compose.yml

```
rabbitmq:
  container_name: broker_rabbitmq
  restart: unless-stopped
  image: rabbitmq:3.6-management
  volumes:
    - ./rabbitmq_config:/etc/rabbitmq
  ports:
    - 8080:15672
    - 5672:5672
```

Docker und RabbitMQ

- enabled_plugins

```
1 [rabbitmq_management].
```

- rabbitmq.config

```
1  [
2    { rabbit,
3      [
4        { loopback_users, [ ] }
5      ]
6    },
7    { rabbitmq_management,
8      [
9       {load_definitions, "/etc/rabbitmq/definitions.json"}
10     ]
11   }
12 ].
```

Docker und RabbitMQ

- definitions.json


```
1 {"rabbit_version":"3.6.6","users":[{"name":"admin","password_hash":"
rTmvLdNL1VdudpRbJLS6SOYZyF0IuYiPeA","hashing_algorithm":"
rabbit_password_hashing_sha256","tags":"administrator"},{"name":"kom
password_hash":"SVy3RcvgzXQe/BDzDv8mCYUJqNTP6Zup7rfxb","hashing_algo
rabbit_password_hashing_sha256","tags":""}],"vhosts":[{"name":"/"}],
permissions":[{"user":"admin","vhost":"/","configure":".*","write":"
.*"},{"user":"kompo","vhost":"/","configure":".*","write":".*","read
parameters":[],"policies":[],"queues":[{"name":"debug","vhost":"/","
true,"auto_delete":false,"arguments":{}},{"name":"out","vhost":"/","
true,"auto_delete":false,"arguments":{}},{"name":"config","vhost":"/
true,"auto_delete":false,"arguments":{}},{"name":"in","vhost":"/","d
true,"auto_delete":false,"arguments":{}}],"exchanges":[{"name":"inEx
vhost":"/","type":"fanout","durable":true,"auto_delete":false,"inter
arguments":{}},{"name":"debug","vhost":"/","type":"fanout","durable"
auto_delete":false,"internal":false,"arguments":{}},{"name":"outExch
vhost":"/","type":"direct","durable":true,"auto_delete":false,"inter
arguments":{}},{"name":"config","vhost":"/","type":"fanout","durable
auto_delete":false,"internal":false,"arguments":{}},{"name":"out","v
type":"direct","durable":true,"auto_delete":true,"internal":false,"a
}],"bindings":[{"source":"config","vhost":"/","destination":"config"
destination_type":"queue","routing_key":"","arguments":{}},{"source"
vhost":"/","destination":"debug","destination_type":"queue","routing
arguments":{}},{"source":"inExchange","vhost":"/","destination":"in"
destination_type":"queue","routing_key":"","arguments":{}}]}
```


Docker und RabbitMQ

- Dann passts ja schon? :)


JUSCHDEN
MIR HAN
HUDDEL

Docker und RabbitMQ

- … er vergisst trotzdem immer die Rechte vom kompo User -.-’
- Trotzdem, immer noch besser als von Hand zu basteln ;)!


DOCKER
RESISTANCE IS FUTILE
made on imgur

It’s not a bug, it’s a feature

HTTP Header…

Content-Type: Application/Json

Content-Length: X

Oder:

Content-Type:Application/x-www-form-urlencoded

It’s not a bug, it’s a … Camel

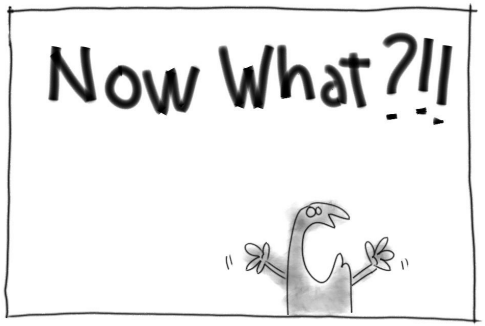
```
from(“src”).id(3).to(“dest”)                    -> id = 3
from(“src”).processor(..).id(3).to(“dest”) -> “route3”
from(“src”).id(3).processor(..).to(“dest”) -> id = 3
from(“src”).to(“dest”).id(3)                    -> no id
```

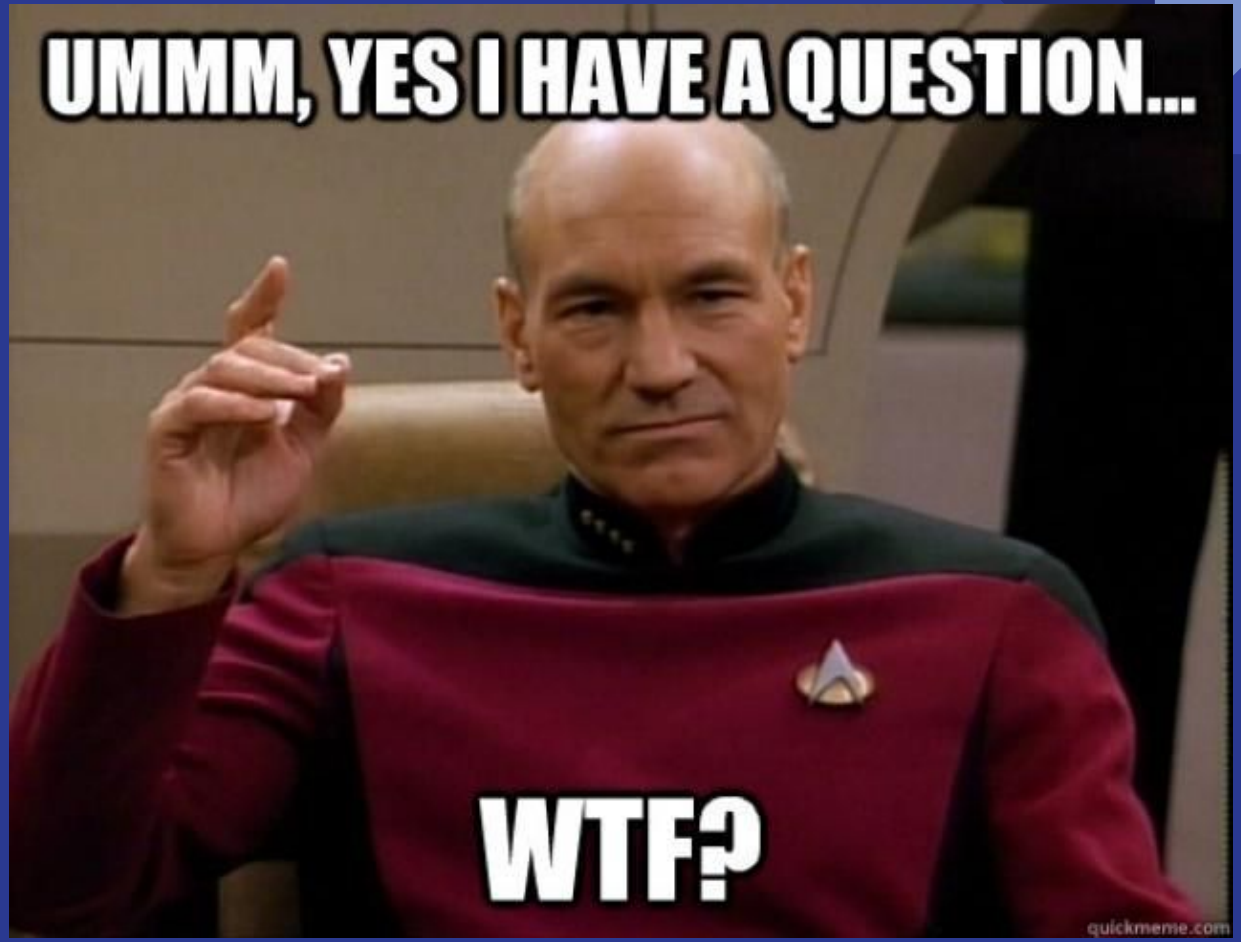
Stuck in rabbit holes

Messages are routed to the queue with the name specified by routing_key, if it exists.

....

We need to supply a routing_key when sending, but its value is ignored for fanout exchanges.




UMMM, YES I HAVE A QUESTION...
WTF?
quickmeme.com

Quellenverzeichnis

- RabbitMQ (Message Queue)
  - https://www.rabbitmq.com
  - https://content.pivotal.io/blog/rabbitmq-hits-one-million-messages-per-second-on-google-compute-engine
  -
- Camel (Java)
  - Enterprise Integration Patterns. Gregory Hohpe, Bobby Woolf. Addison Wesley
  - Camel in Action. Claus Ibsen, Jonathan Anstey. Manning Pubn
  - https://camel.apache.org/documentation.html
- Docker
  - www.docker.com