Code Europe

The largest programming conference in Poland

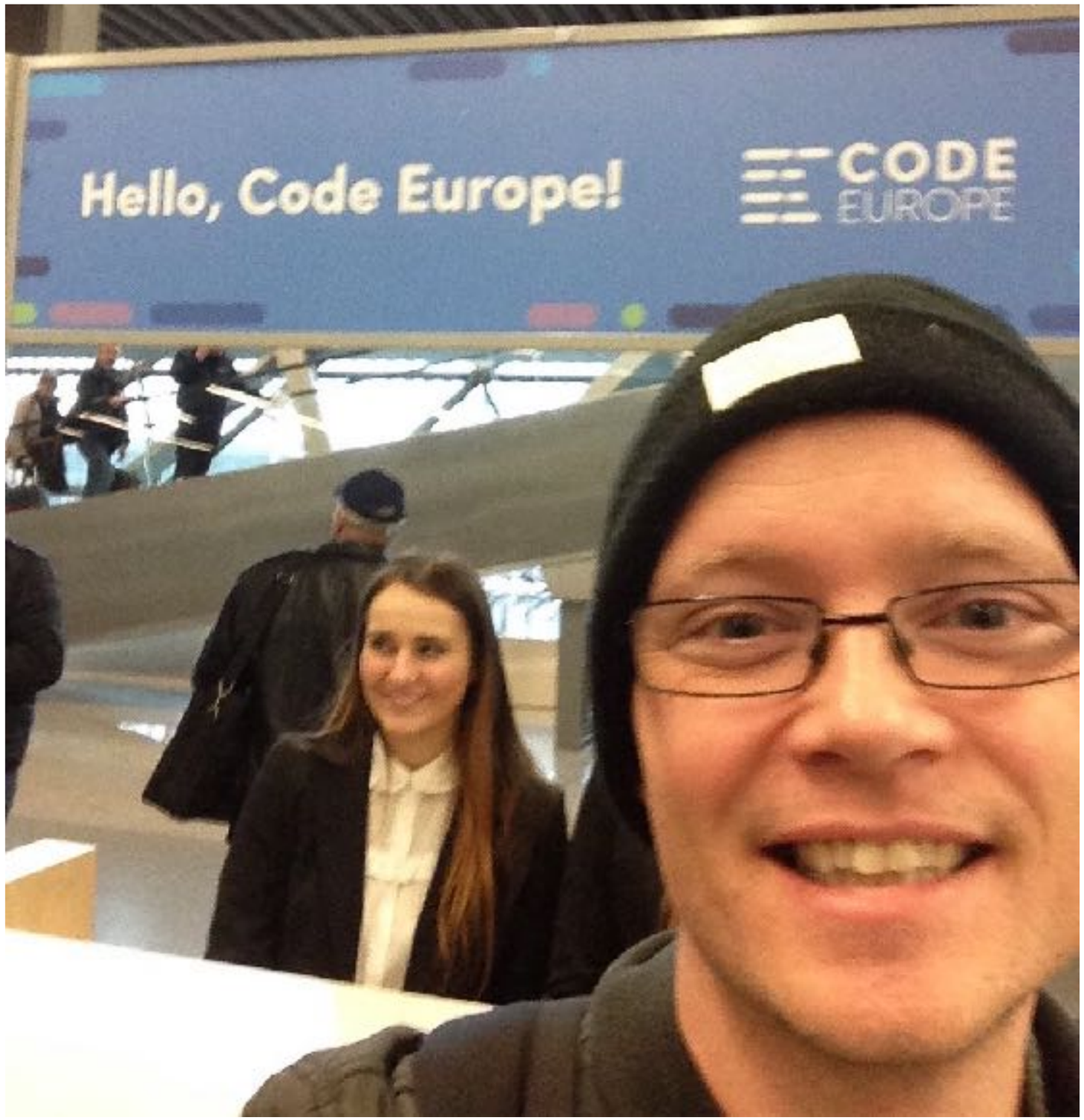
Agile integration with Apache Camel microservices in containers on Kubernetes

Claus Ibsen

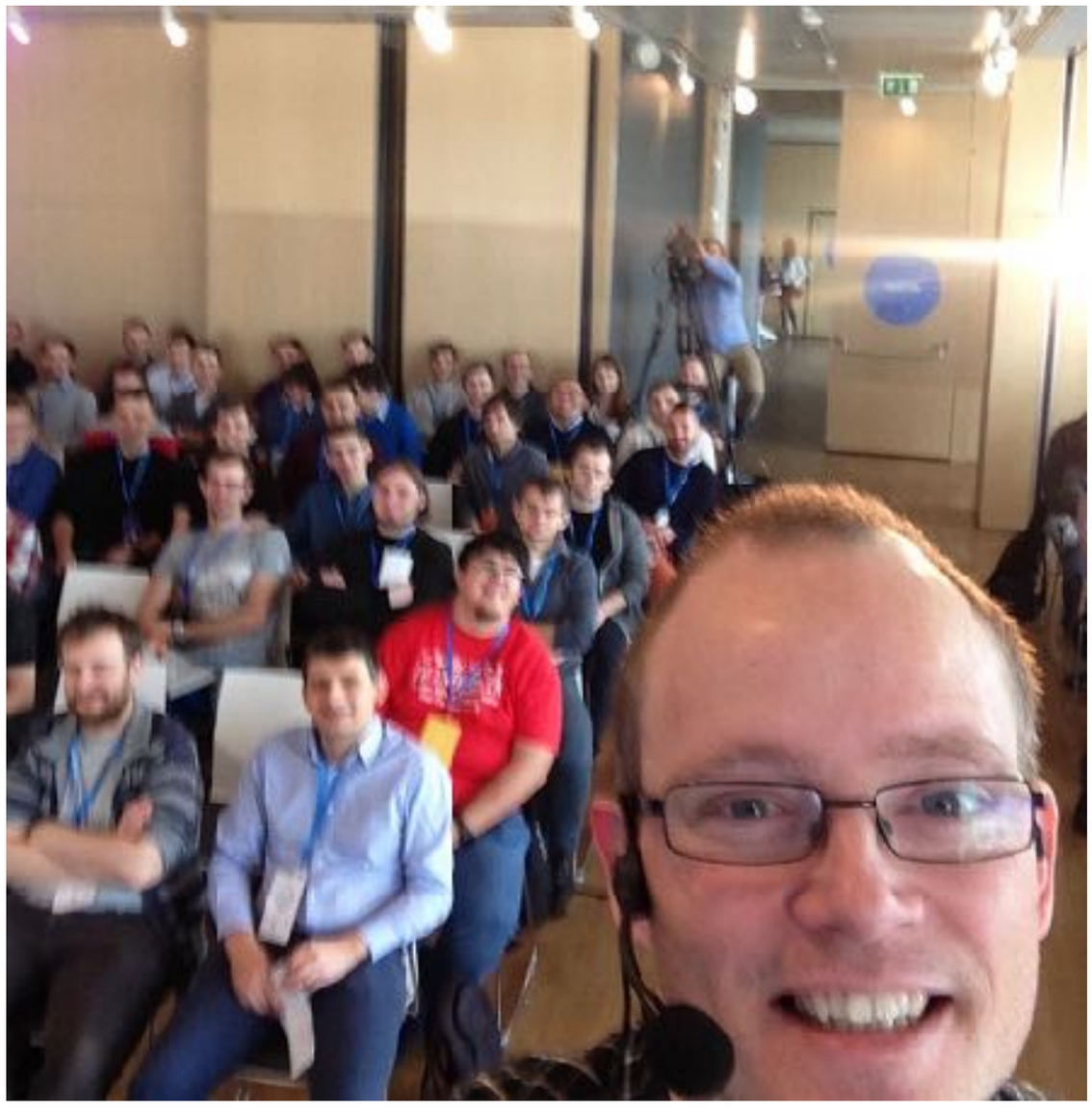
@davsclaus

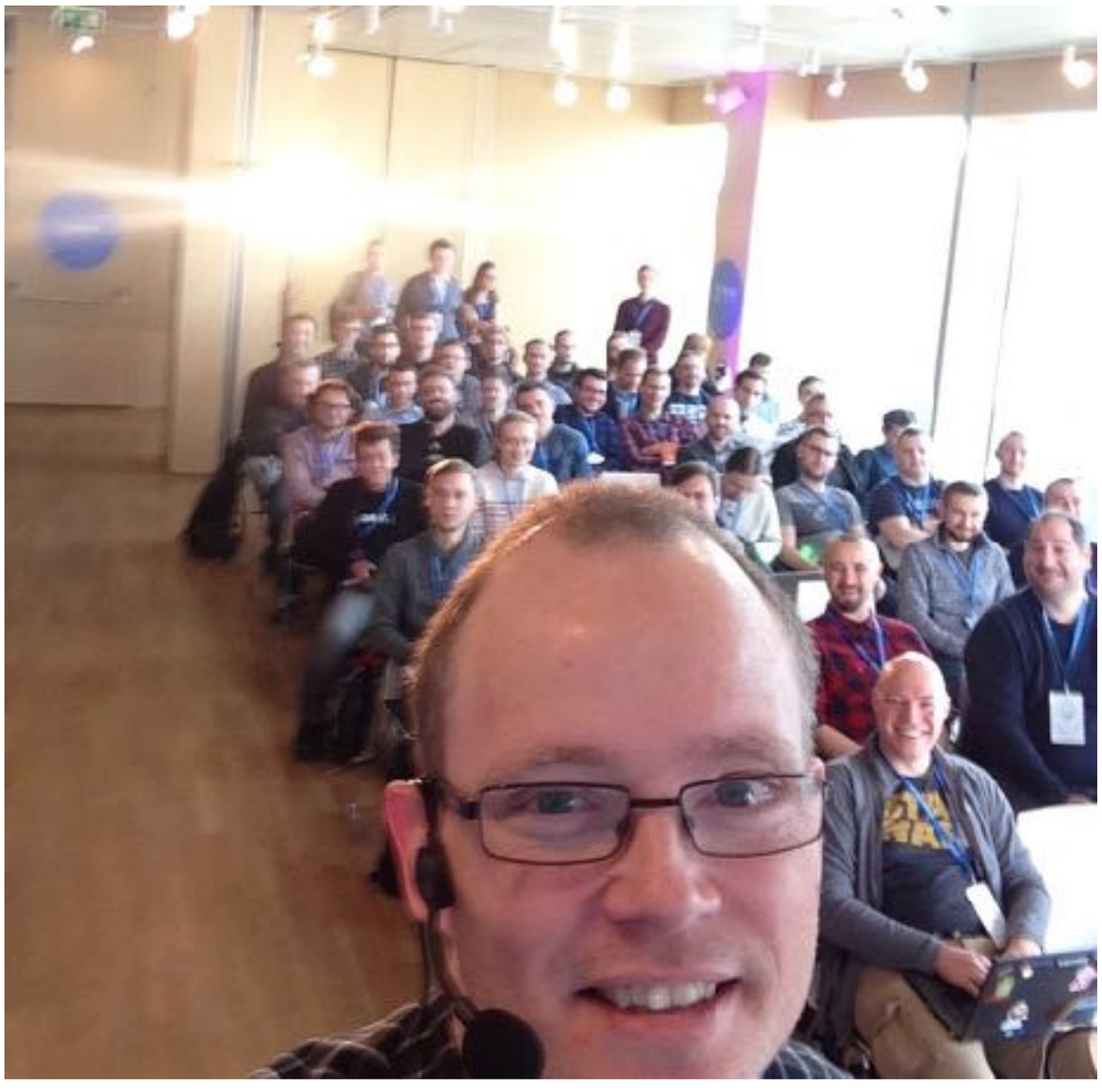
Poland
December 2017

My first selfie in Poland



My 2nd selfies in Poland





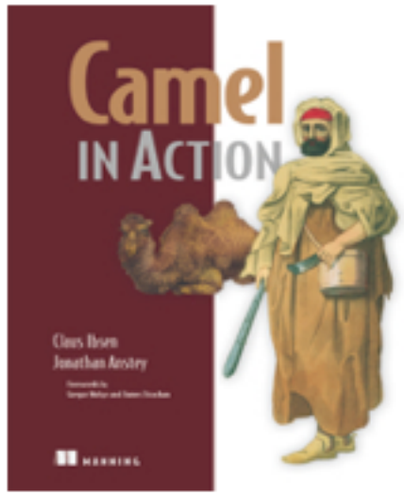
Claus Ibsen

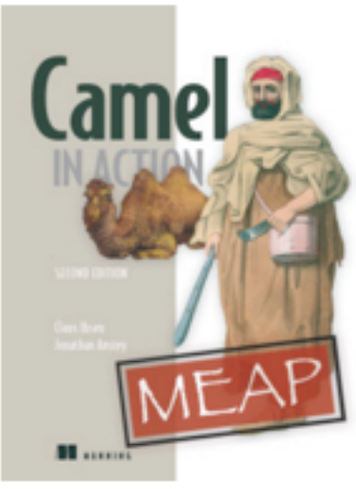
@davsclaus
davsclaus
www.davsclaus.com
cibsen@redhat.com

- Senior Principal Software Engineer at Red Hat

- Apache Camel
  9 years working with Camel

- Author of
  Camel in Action books

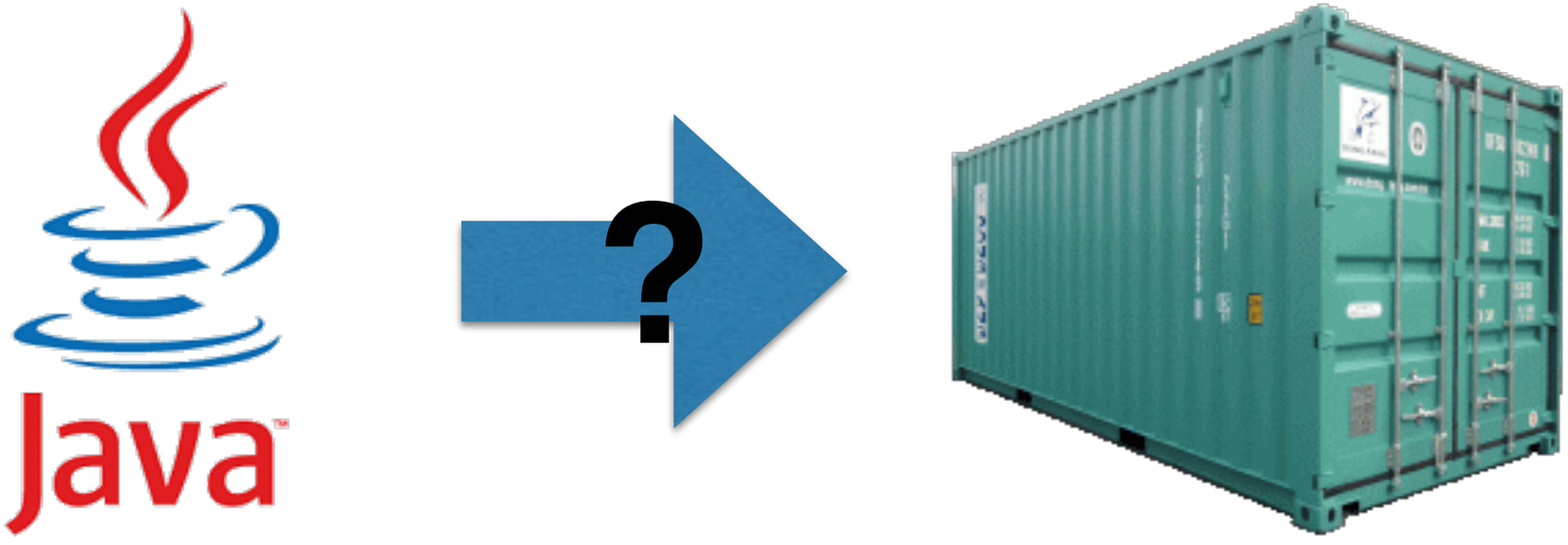




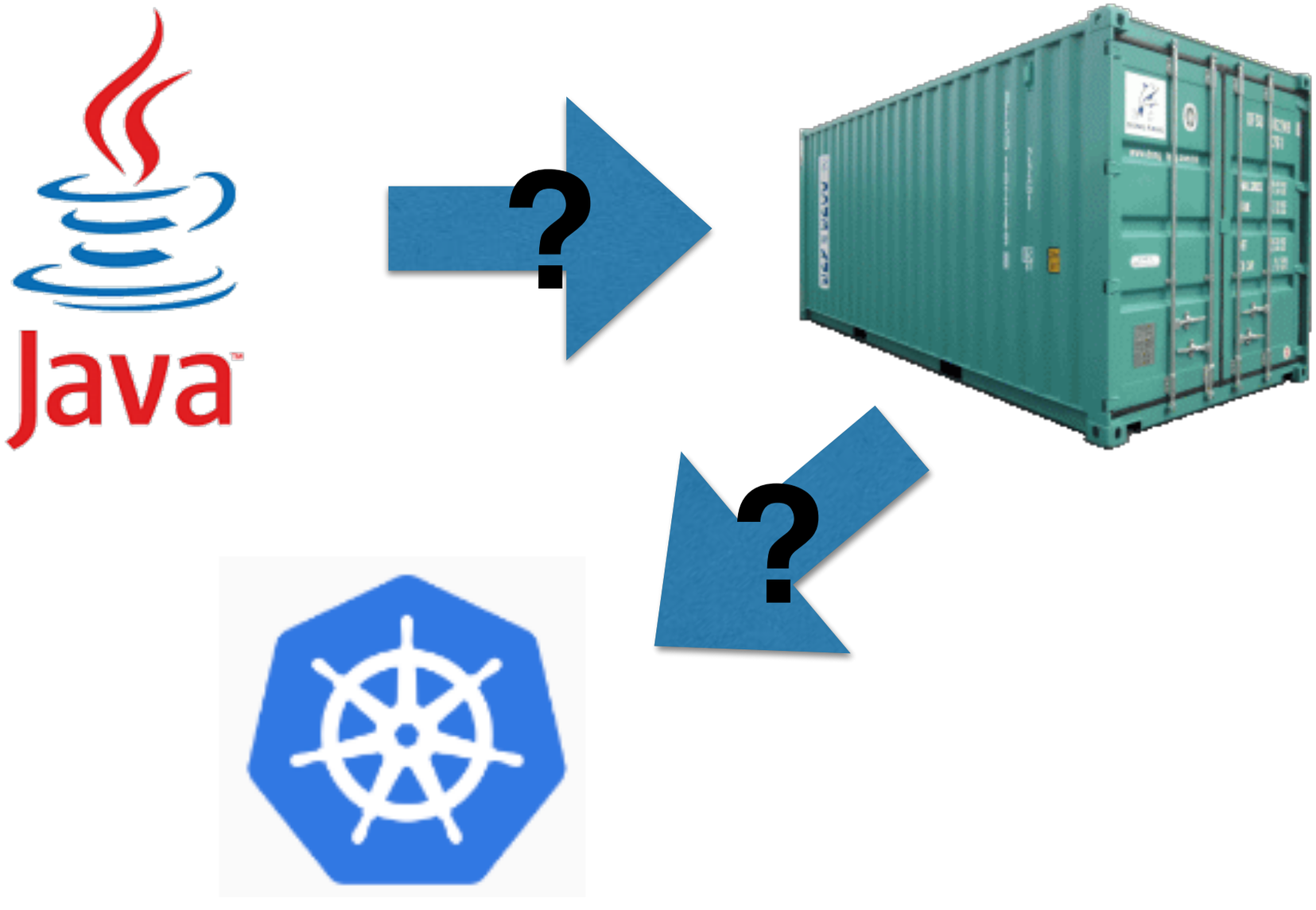
Key Message

Key Message

Key Message



Key Message



Tip of Iceberg

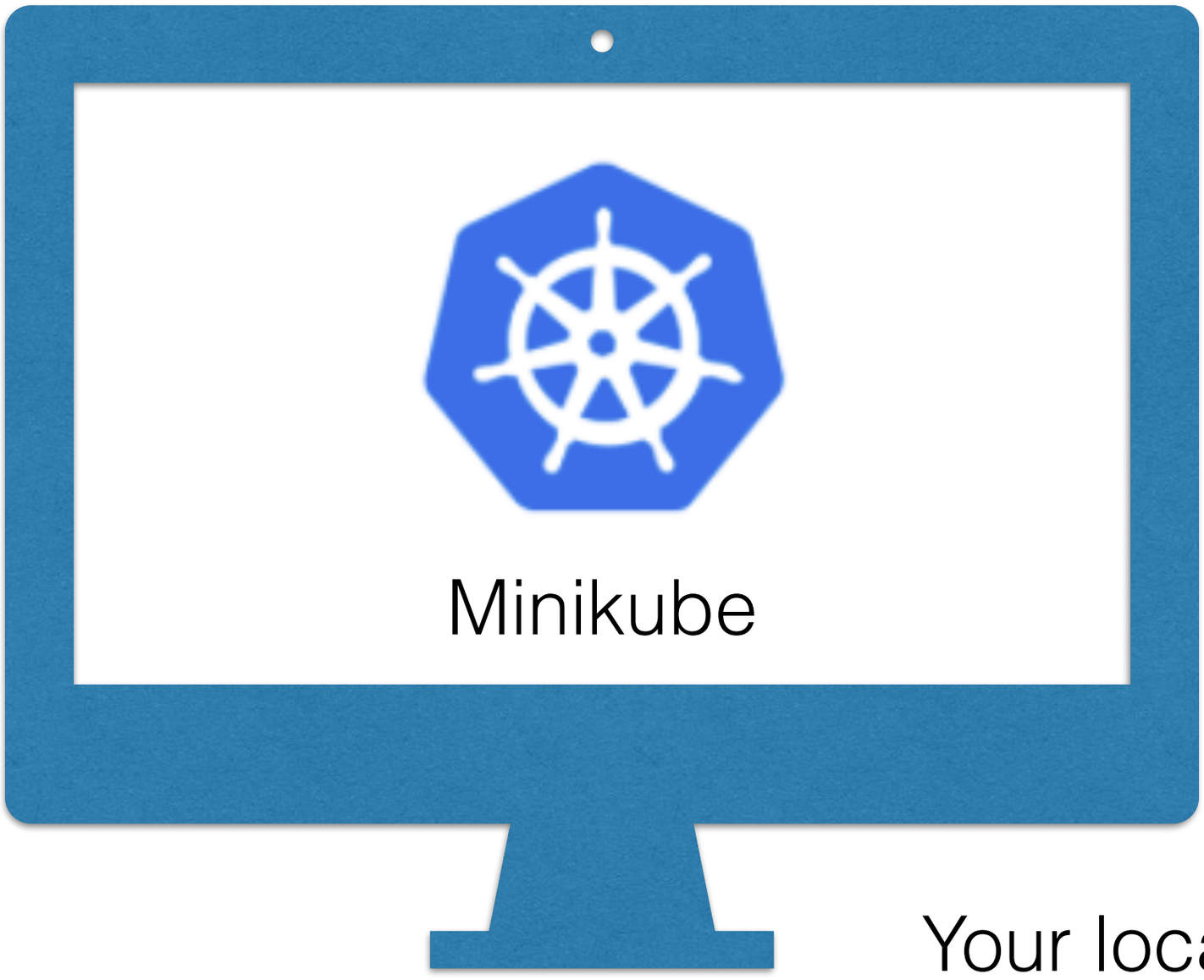
Running Kubernetes

Big "scary" cluster software

Running Local Kubernetes

Minikube

Running Local Kubernetes



Installing Minikube

1. Download Minikube

   https://github.com/kubernetes/minikube

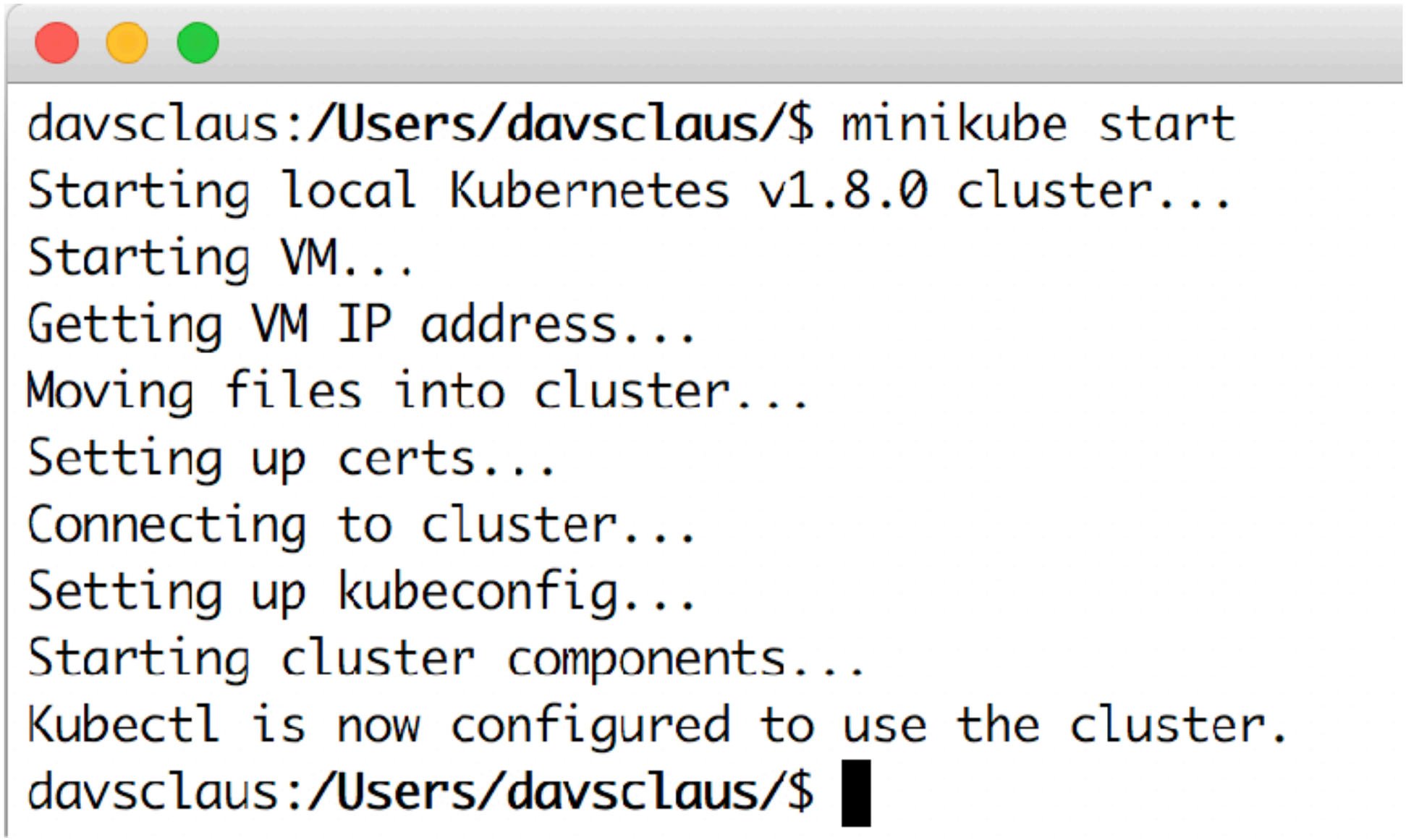
2. Install kubectl (command line tool)

3. Start Minikube

   ```
   $ minikube start
   ```

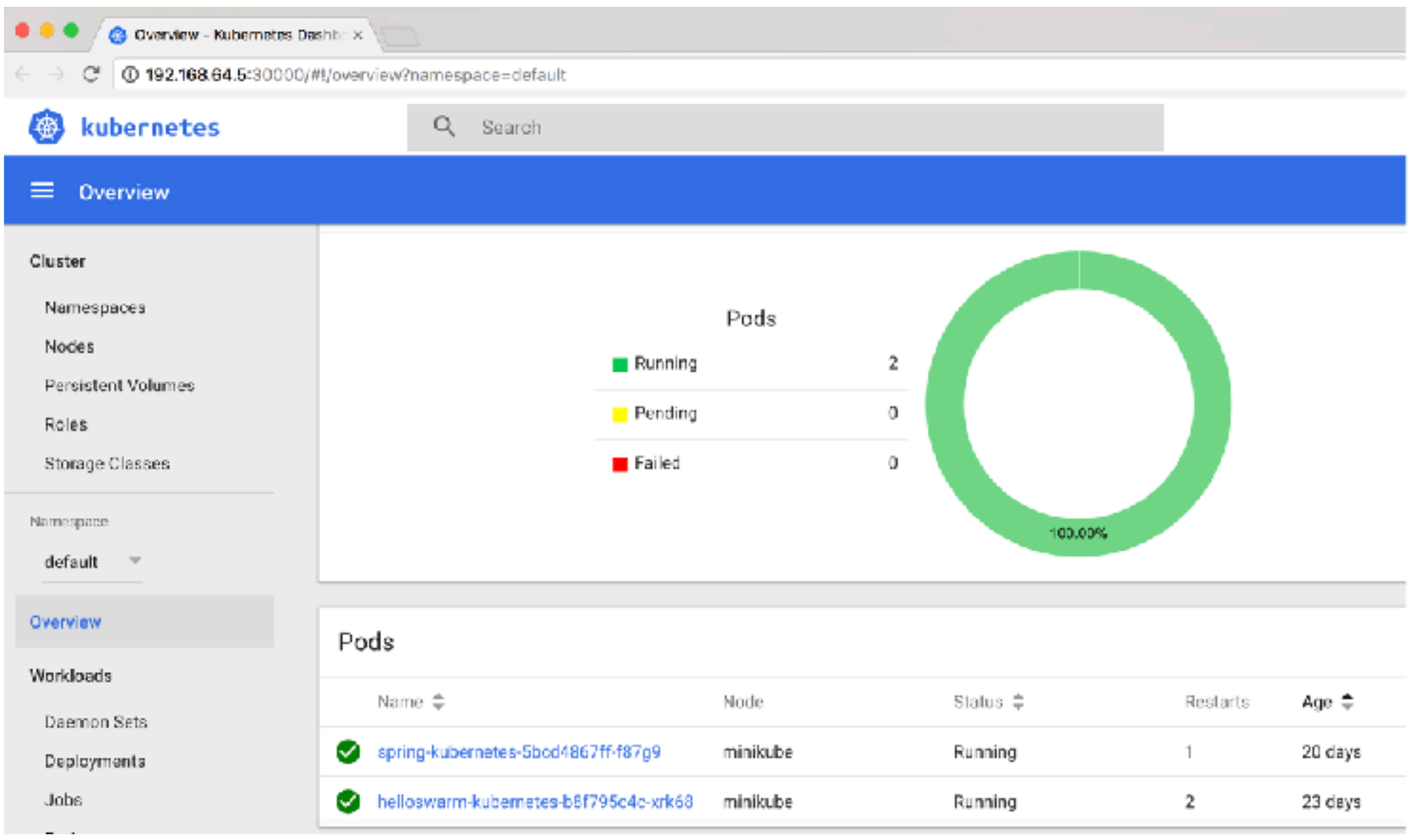
https://kubernetes.io/docs/tasks/tools/install-minikube/

How I installed Minikube

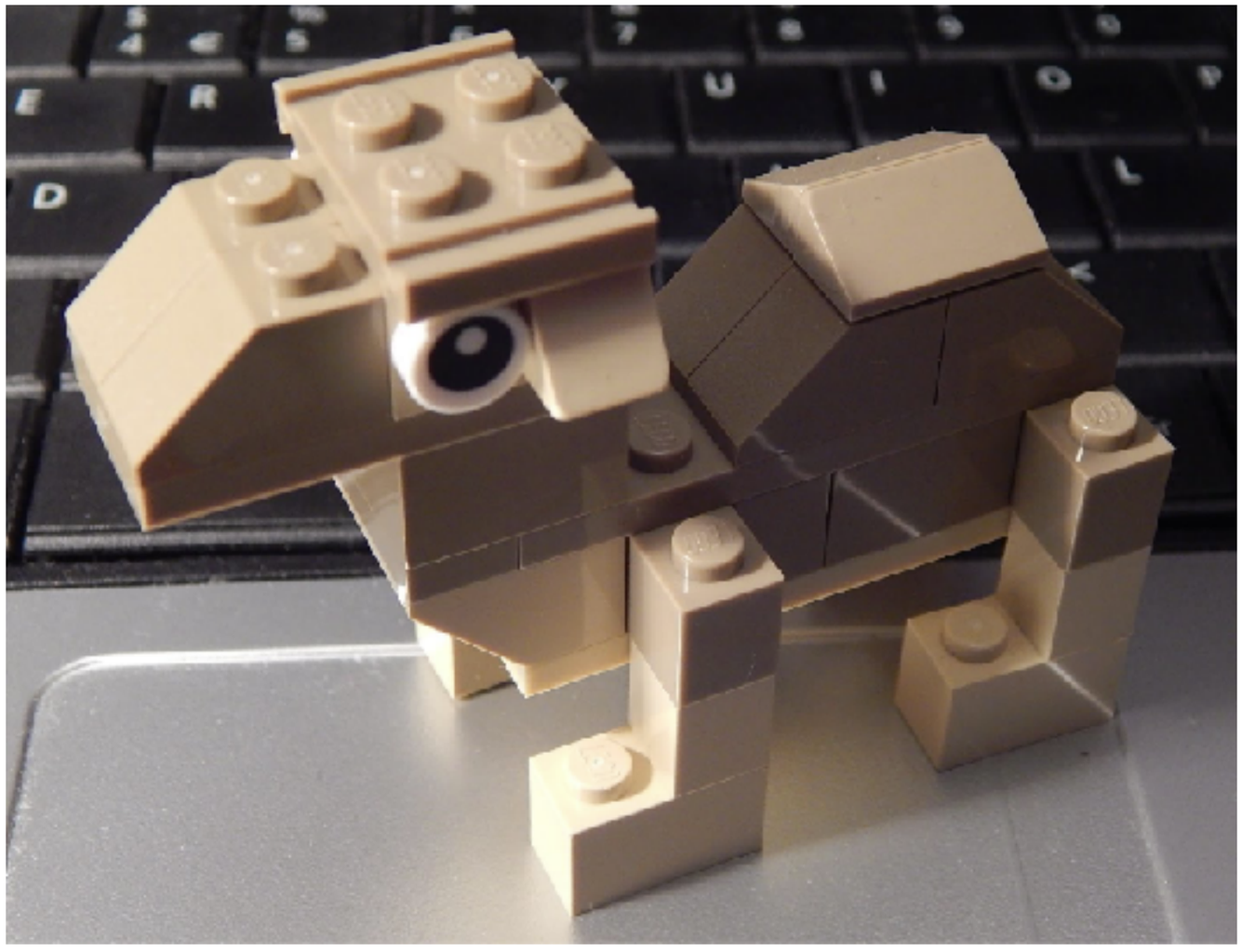


Web Console

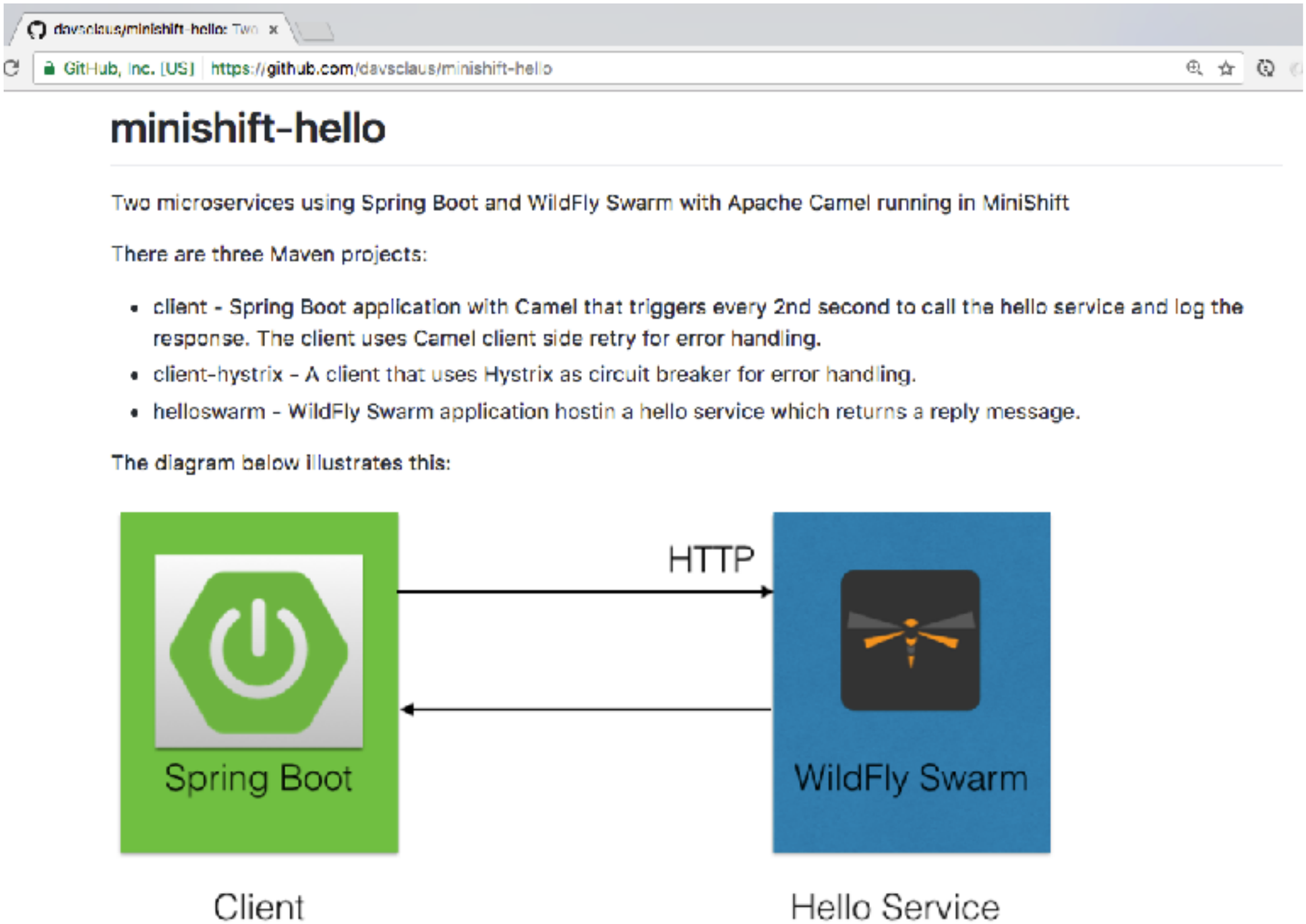
```
$ minikube dashboard
```



Build a Camel Demo Time



Source Code & Slides



https://github.com/davsclaus/minishift-hello

Hello Service

Client

Hello
Service

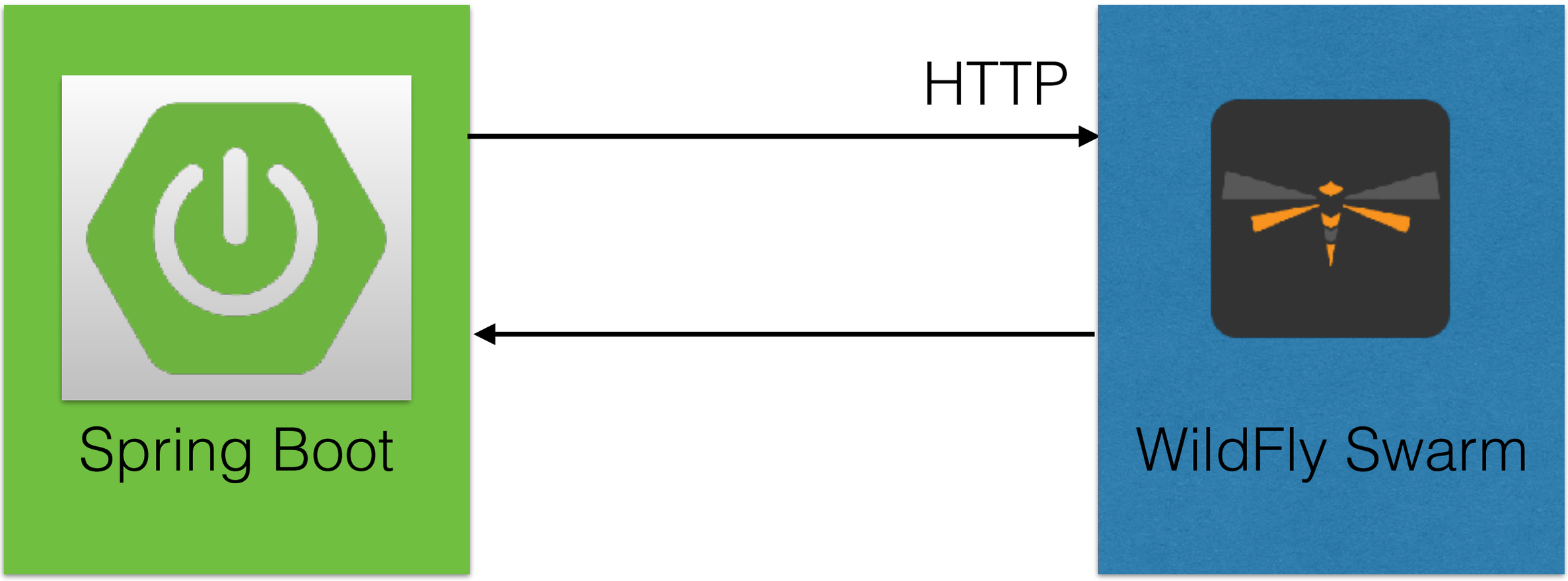
Hello Service

Swarm says hello from $HOSTNAME

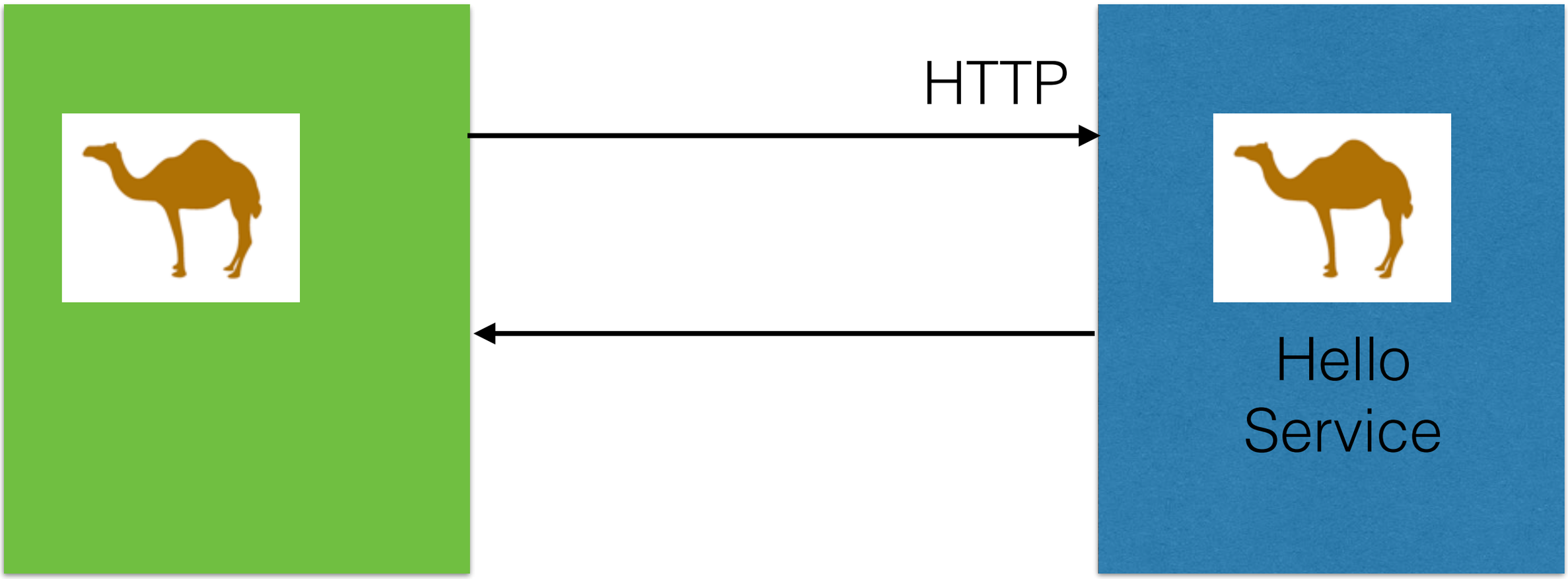
Client

Hello Service

Implementation



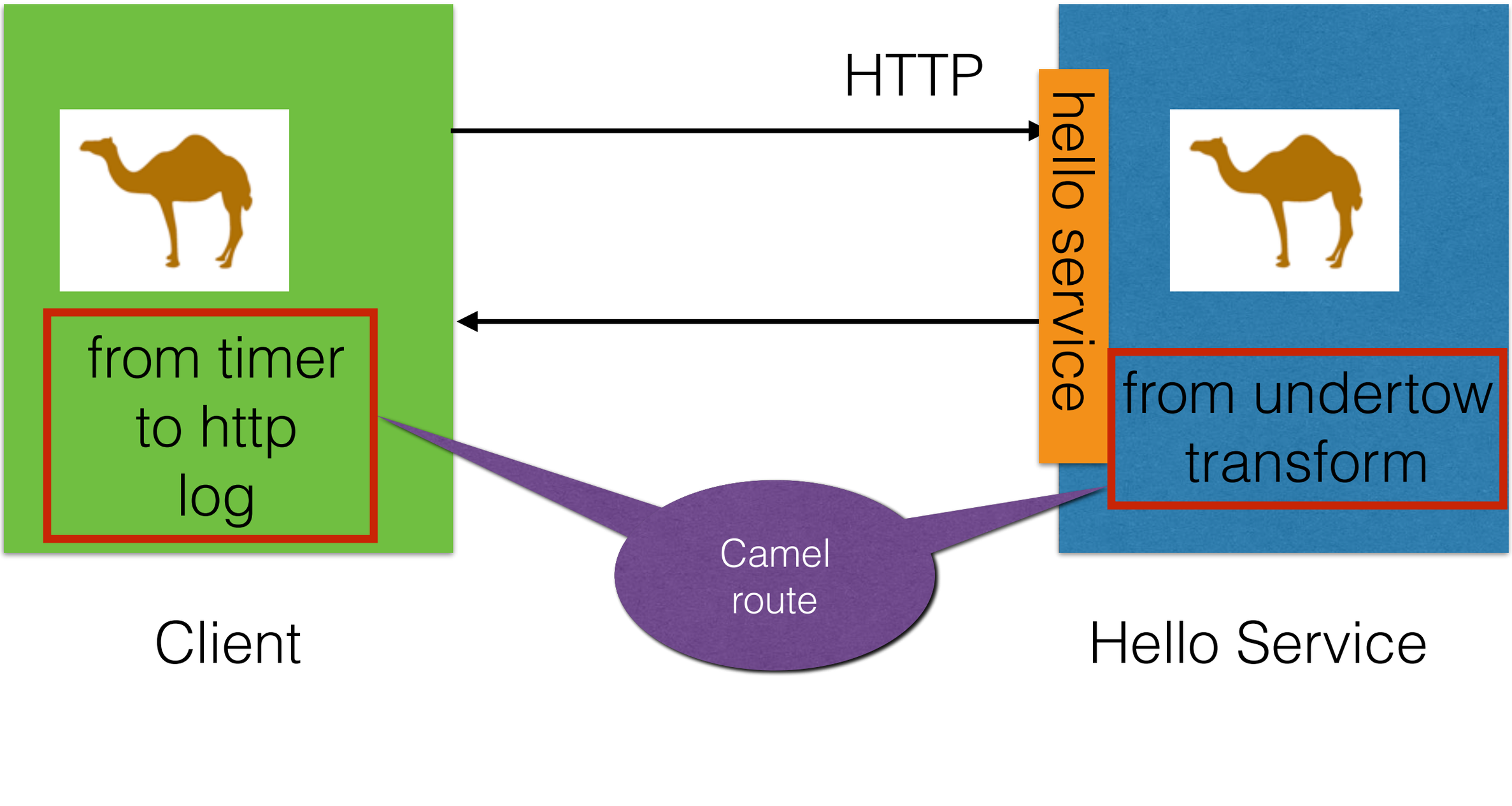
Implementation



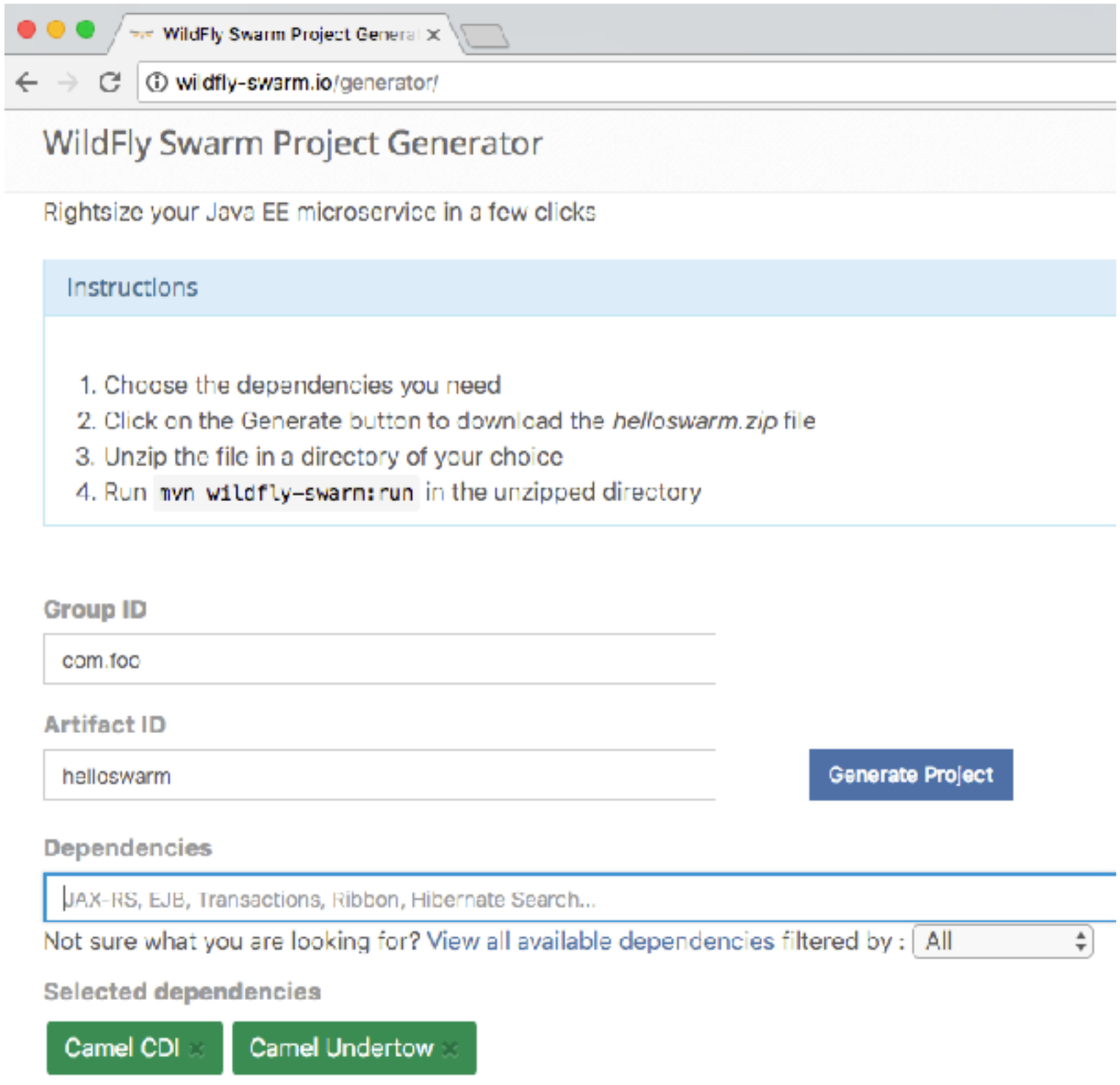
Client

Hello Service

Implementation



WildFly Swarm Generator



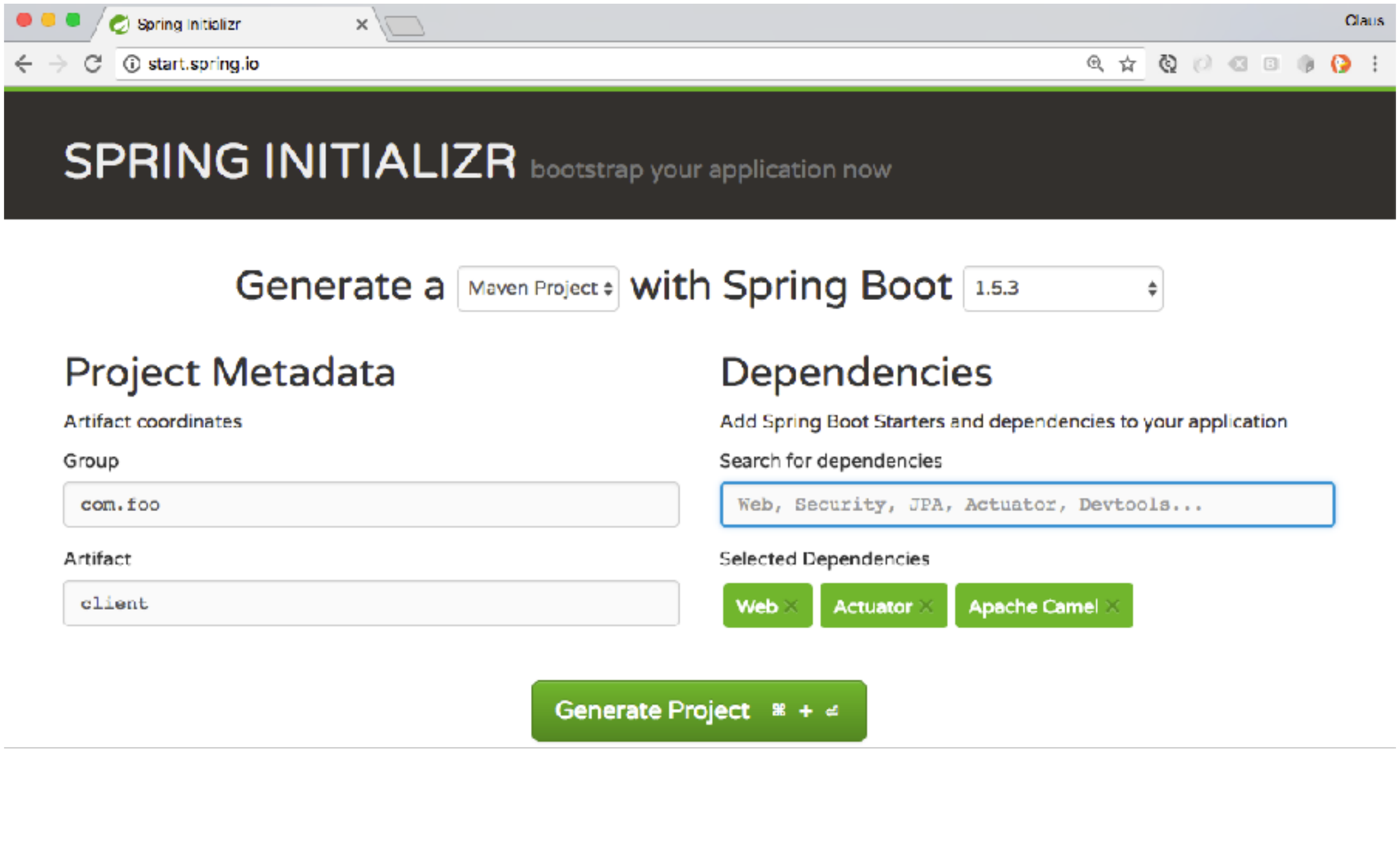
Hello Service

```
@Singleton
public class HelloRoute extends RouteBuilder {

    @Inject
    @Uri("undertow:http://0.0.0.0:8080/hello")
    private Endpoint undertow;

    @Inject
    private HelloBean hello;

    @Override
    public void configure() throws Exception {
        from(undertow).bean(hello);
    }
}
```

Hello Service

```
@Singleton
public class HelloBean {

    public String sayHello() throws Exception {
        String answer = "Swarm says hello from "
            + InetAddressUtil.getLocalHostName();
        return answer;
    }
}
```

Spring Boot Starter



Client

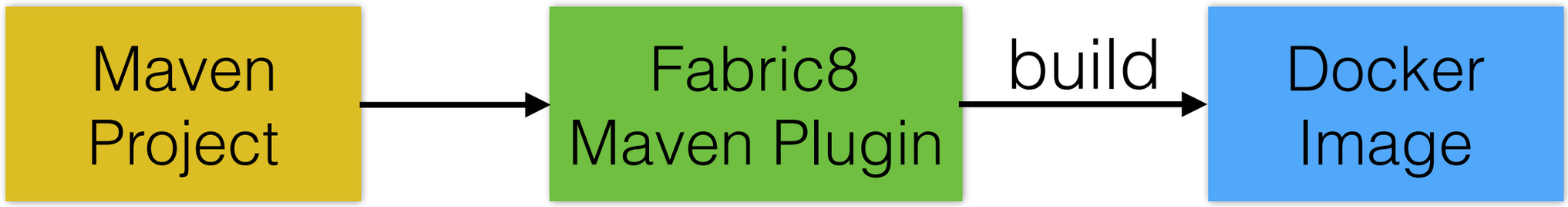
```
@Component
public class MyRoute extends RouteBuilder {

    @Override
    public void configure() throws Exception {
        from( uri: "timer:foo?period=2000")
                .to("http4:localhost:8080/hello")
                .log("${body}");
    }
}
```

How to build Docker Image?

Maven Project

Docker Image

Fabric8 Maven Plugin



https://maven.fabric8.io

Fabric8 Maven Plugin


```xml
<plugin>
    <groupId>io.fabric8</groupId>
    <artifactId>fabric8-maven-plugin</artifactId>
    <version>3.5.33</version>
</plugin>
```

https://maven.fabric8.io

Setup Shell

- Prepare command shell

```
$ minikube docker-env
eval $(minikube docker-env)
```

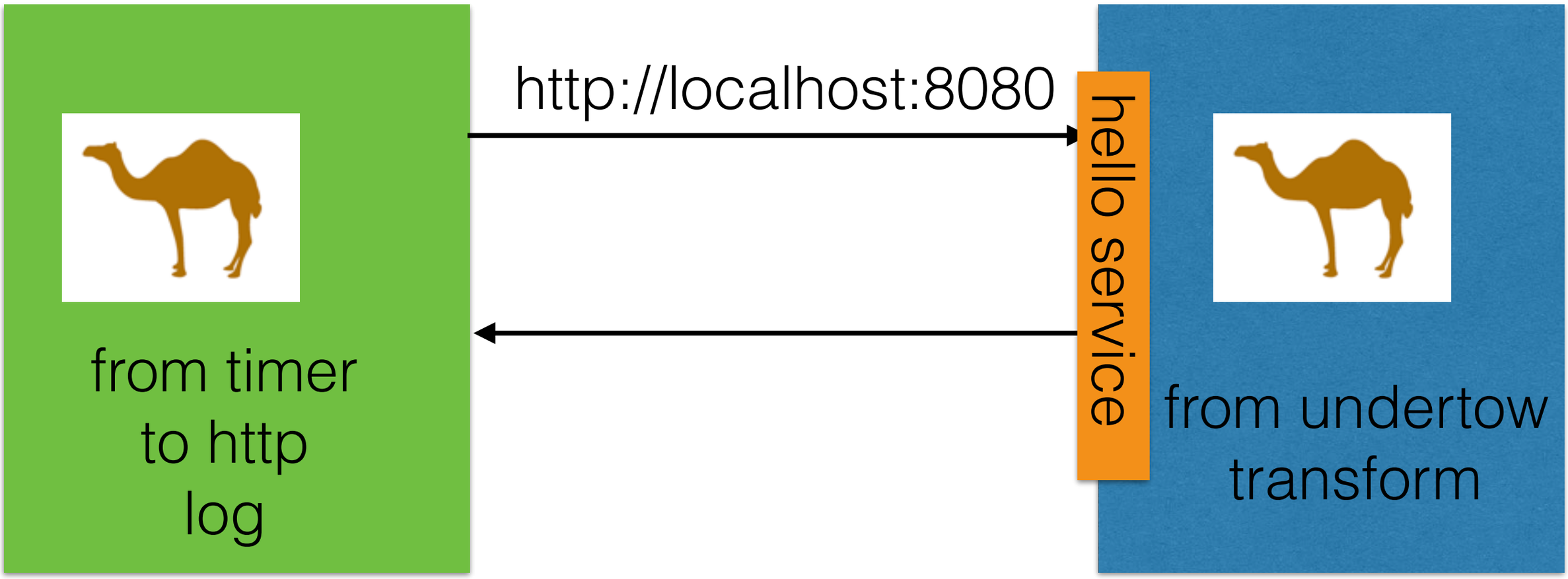
```
davsclaus:/Users/davsclaus/webinar/helloswarm (master)/$ minikube docker-env
export DOCKER_TLS_VERIFY="1"
export DOCKER_HOST="tcp://192.168.64.5:2376"
export DOCKER_CERT_PATH="/Users/davsclaus/.minikube/certs"
export DOCKER_API_VERSION="1.23"
# Run this command to configure your shell:
# eval $(minikube docker-env)
davsclaus:/Users/davsclaus/webinar/helloswarm (master)/$ eval $(minikube docker-env)
```

Build Docker Image

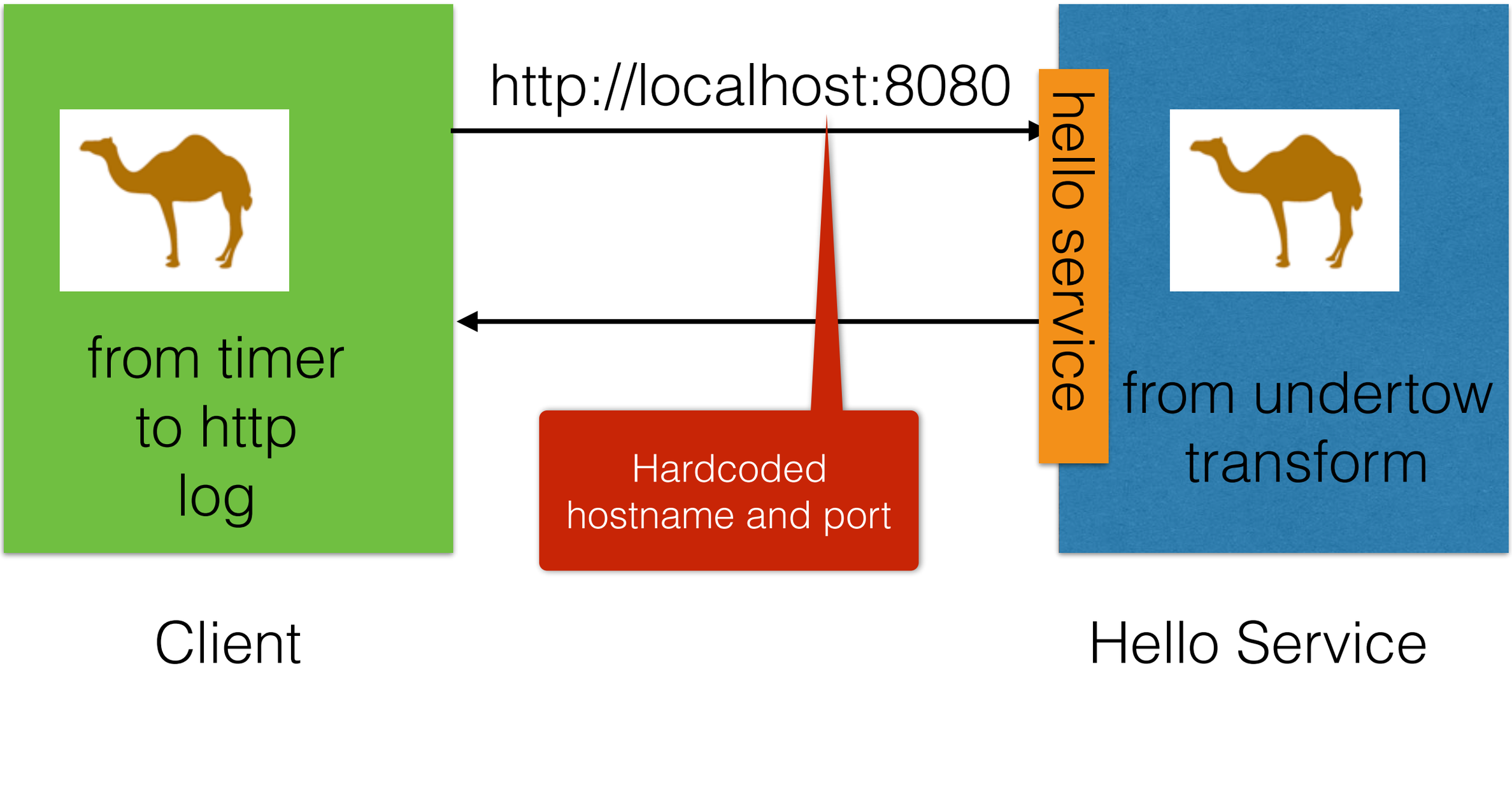
```
[INFO] --- fabric8-maven-plugin:3.3.5:resource (fmp) @ helloswarm ---
[INFO] F8: Running in Kubernetes mode
[INFO] F8: Running generator wildfly-swarm
[INFO] F8: wildfly-swarm: Using Docker image fabric8/java-jboss-openjdk8-jdk:1.2 as base / builder
[INFO] F8: fmp-controller: Adding a default Deployment
[WARNING] F8: fmp-service: Implicit service port mapping to port 80 has been disabled for the used
either use set the config port = 80 or use legacyPortMapping = true. See https://maven.fabric8.io/#
[INFO] F8: fmp-service: Adding a default service 'helloswarm' with ports [8080]
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ helloswarm ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 2 source files to /Users/davsclaus/Documents/workspace/minishift-hello/helloswarm/
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ helloswarm ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /Users/davsclaus/Documents/workspace/minishift-hello/hel
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ helloswarm ---
[INFO] No sources to compile
```

mvn fabric8:build

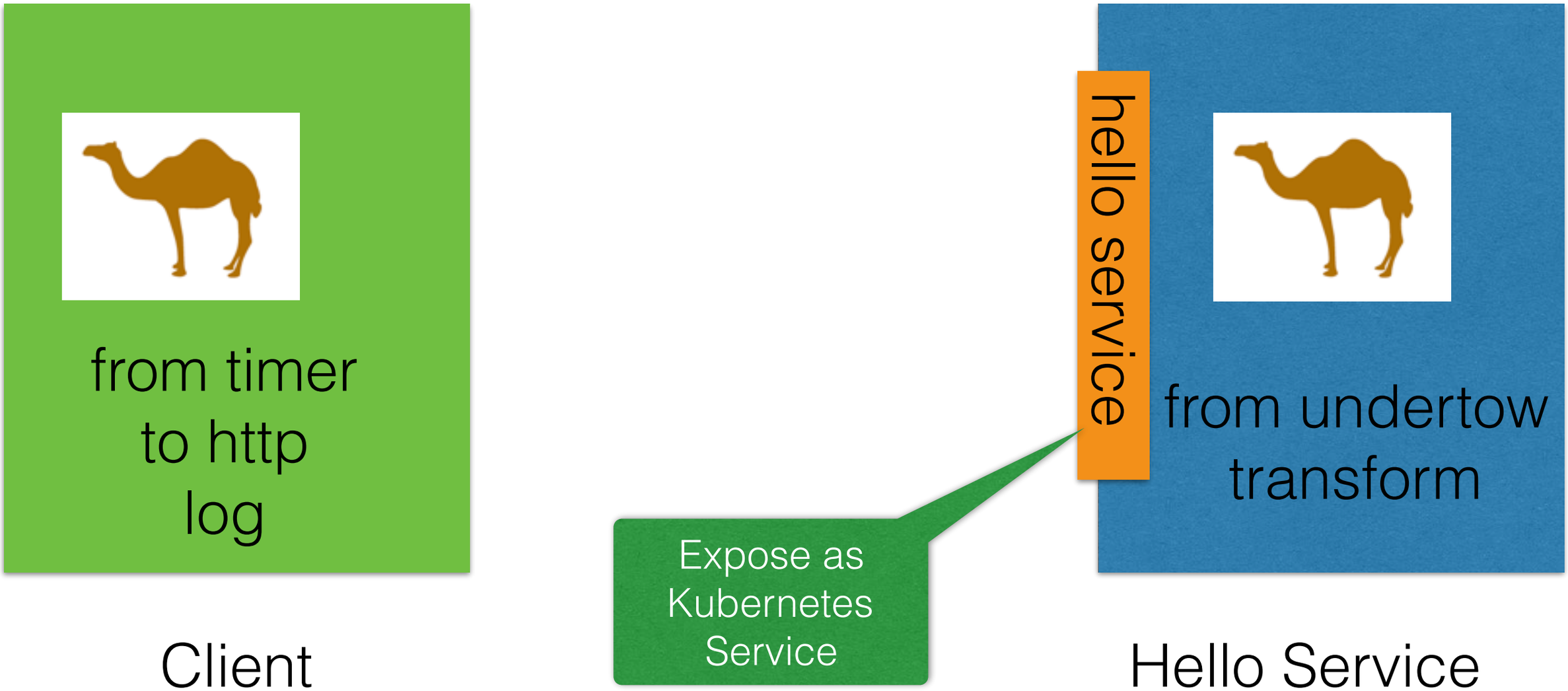
Our Demo


http://localhost:8080
hello service
from timer
to http
log
from undertow
transform
Client
Hello Service

Static vs Dynamic Platform



Dynamic Platform



Client

Hello Service

Dynamic Platform

Client

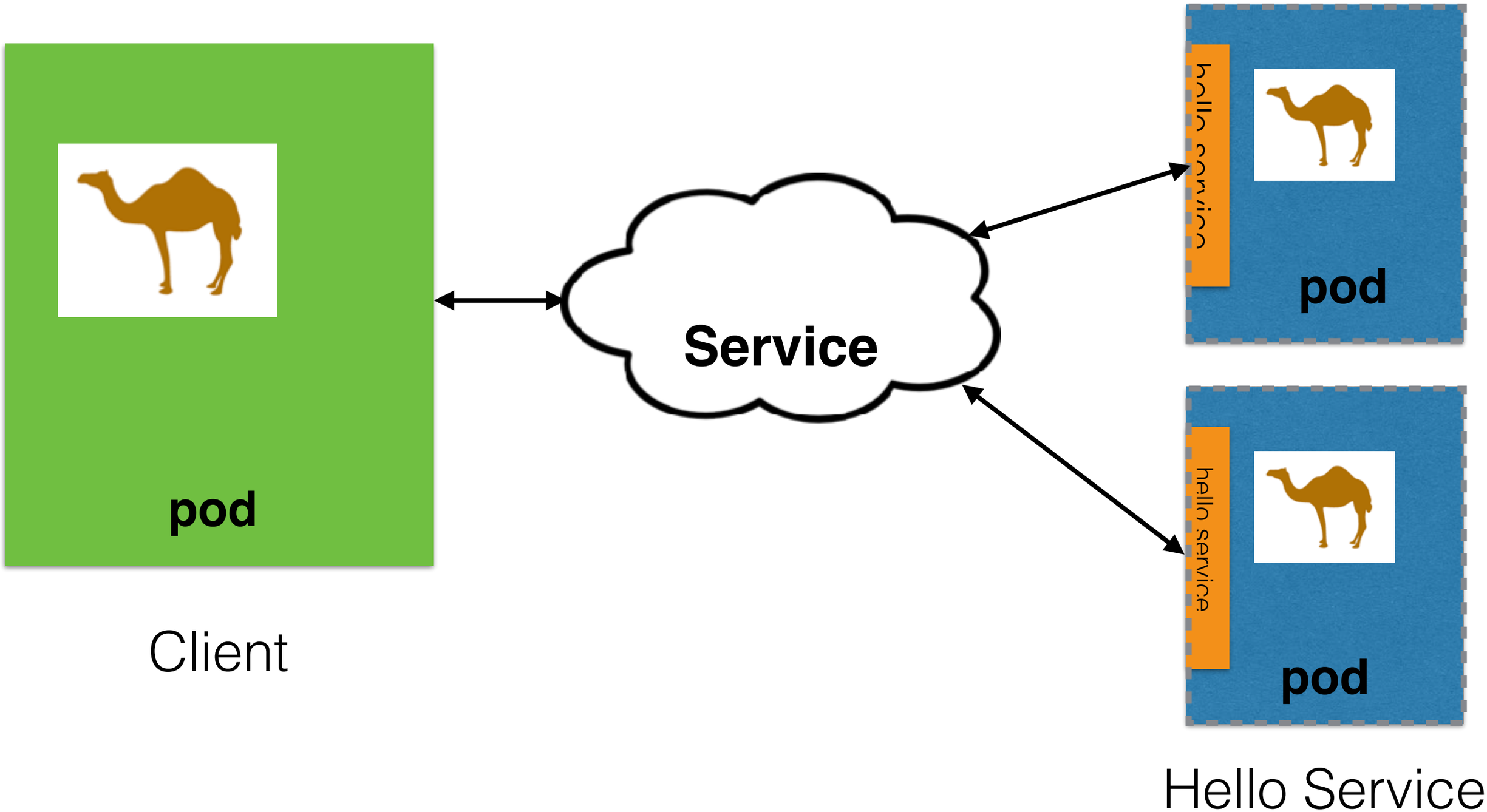
Hello Service

Dynamic Platform

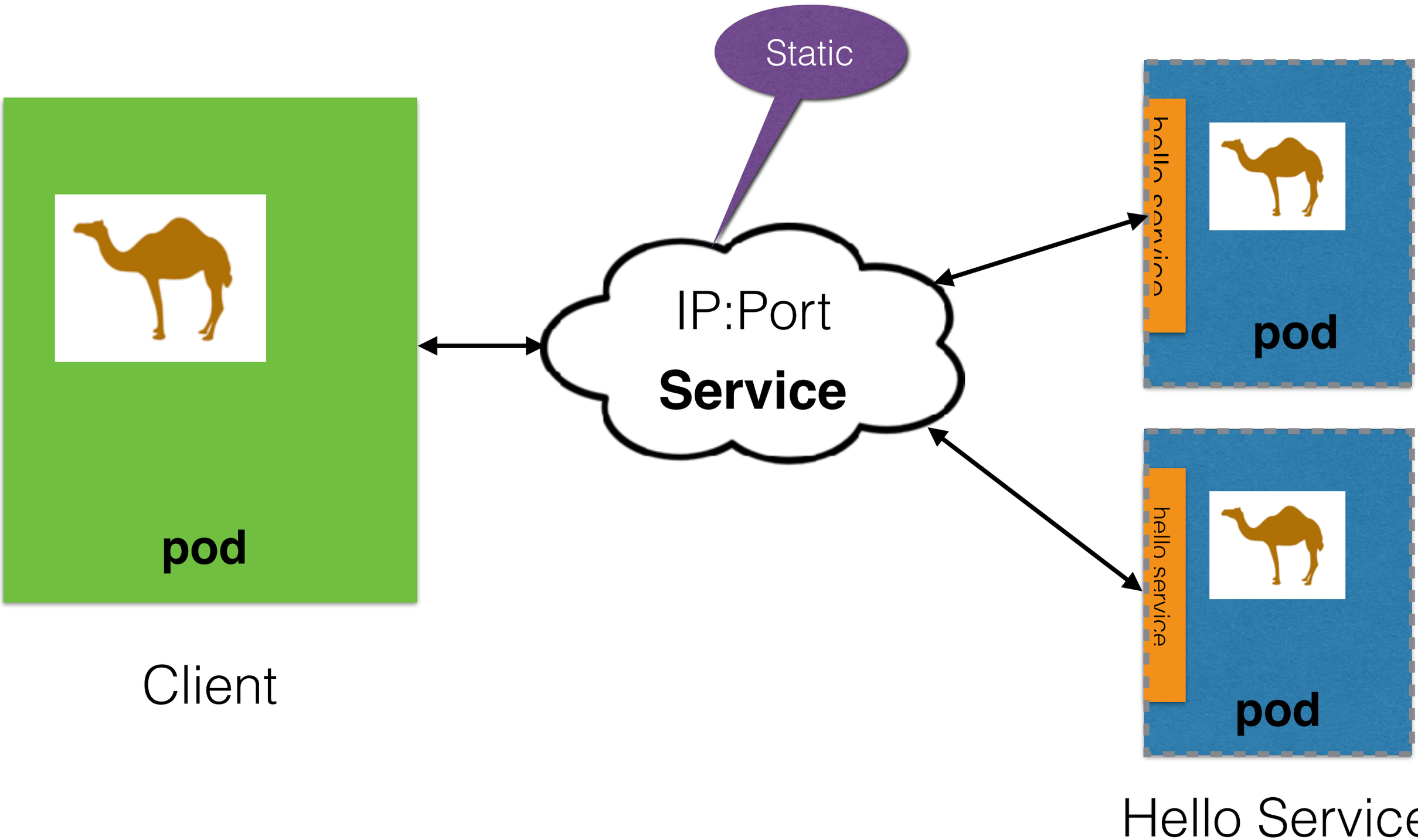
Client

Hello Service

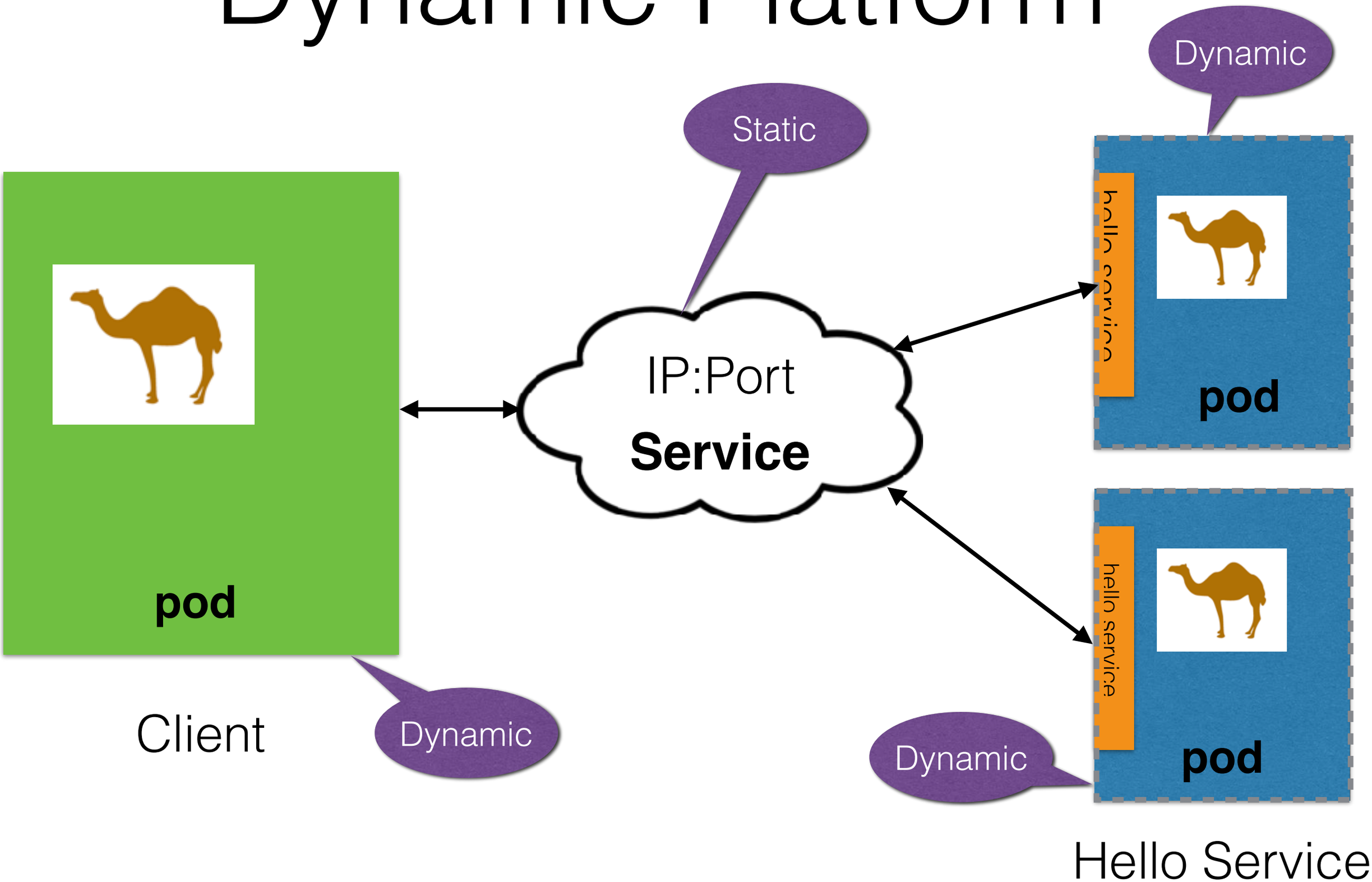
Dynamic Platform



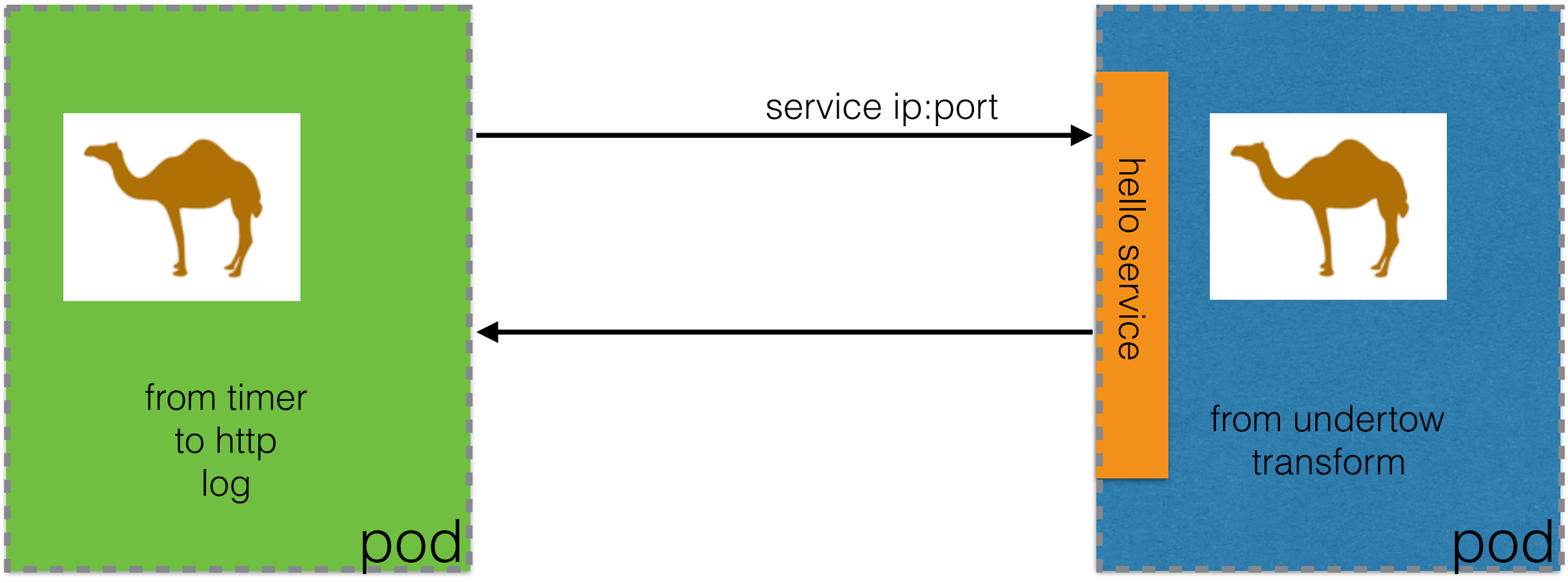
Dynamic Platform



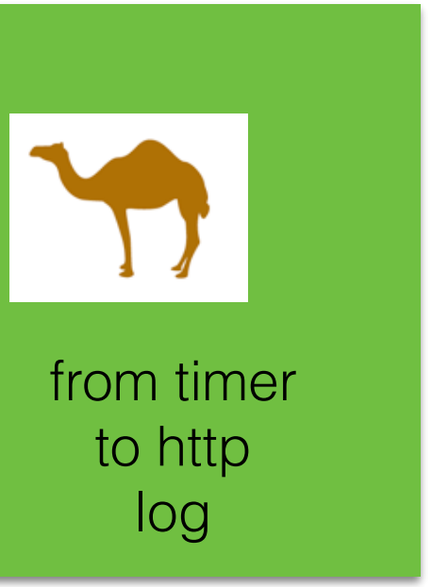
Dynamic Platform



Kubernetes Service from user point of view



Using Kubernetes Service



Client

We want to use hello service

How do we do that?

Using Kubernetes Service

- Environment Variables
  - Hostname
  - Port

```
export HELLOSWARM_SERVICE_HOST="172.30.237.105"
export HELLOSWARM_SERVICE_PORT="8080"
```

Service Discovery using DNS is also available

Service using ENV

- {{service:name}}

Client

```
@Component
public class MyRoute extends RouteBuilder {

    @Override
    public void configure() throws Exception {
        from( uri: "timer:too?period=2000")
                .to("http4:{{service:helloswarm}}/hello")
                .log("${body}");
    }
}
```

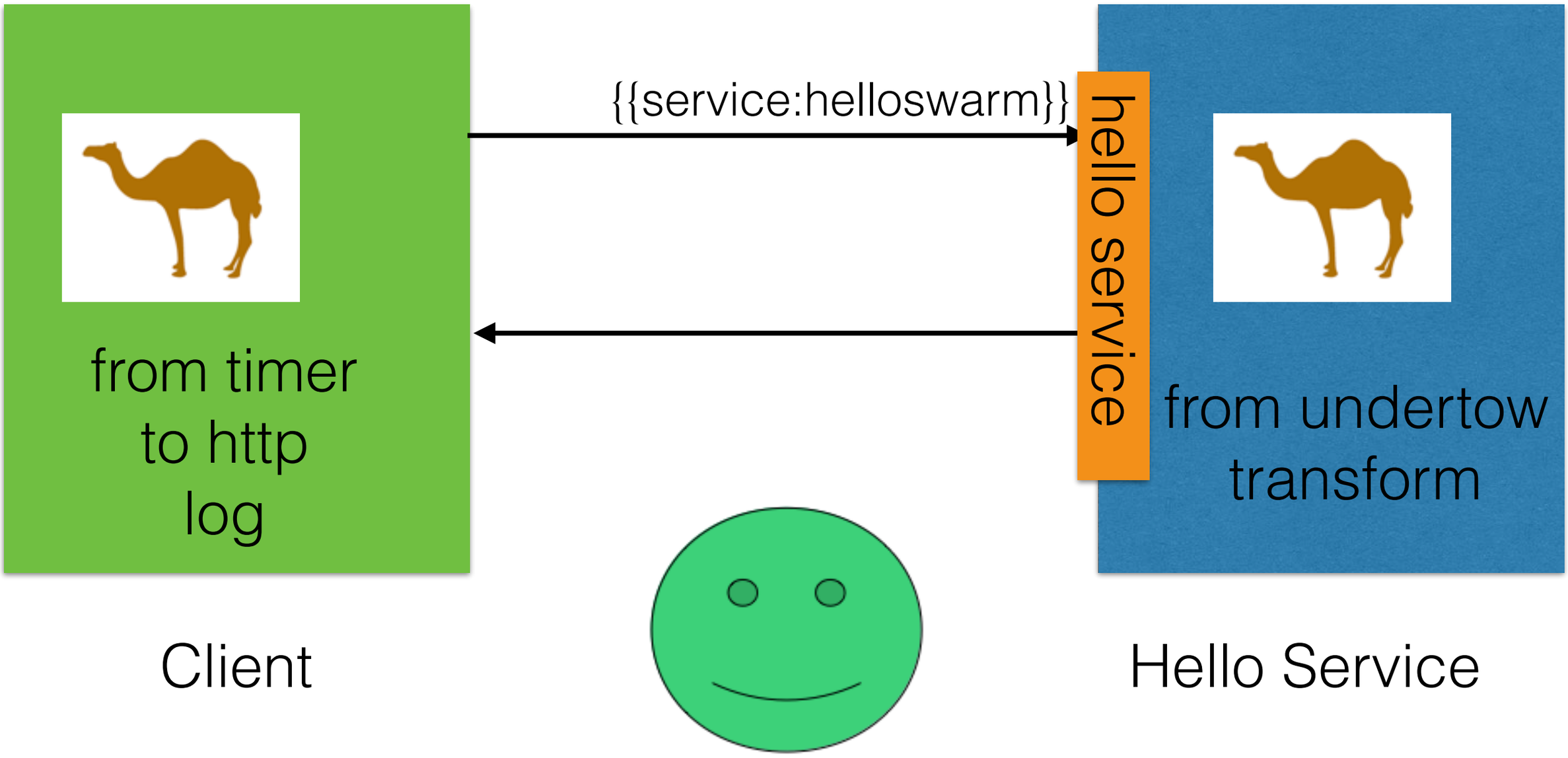
Service using DNS

Client

- servicename:port

```
@Component
public class MyRoute extends RouteBuilder {

    @Override
    public void configure() throws Exception {
        from( uri: "timer:too?period=2000")
                .to("http4:helloswarm:8080/hello")
                .log("${body}");
    }
}
```

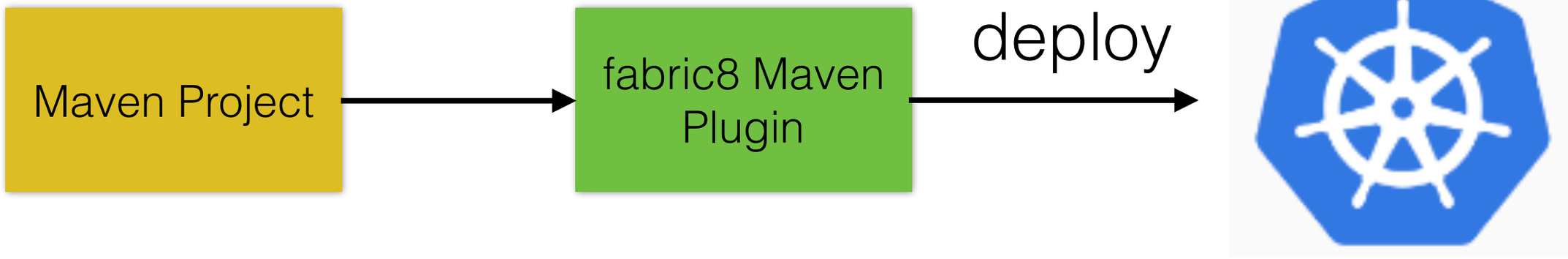
Ready to run in Kubernetes



How to deploy to Kubernetes?

Maven Project

How to deploy to Kubernetes?



Deploy - Hello Service

- `$ mvn fabric8:deploy`

Hello Service

```
[INFO] --- fabric8-maven-plugin:3.3.5:resource (fmp) @ helloswarm ---
[INFO] F8: Running in Kubernetes mode
[INFO] F8: Running generator wildfly-swarm
[INFO] F8: wildfly-swarm: Using Docker image fabric8/java-jboss-openjdk8-jdk:1.2 as base / builder
[INFO] F8: fmp-controller: Adding a default Deployment
[WARNING] F8: fmp-service: Implicit service port mapping to port 80 has been disabled for the used
either use set the config port = 80 or use legacyPortMapping = true. See https://maven.fabric8.io/#
[INFO] F8: fmp-service: Adding a default service 'helloswarm' with ports [8080]
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ helloswarm ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 2 source files to /Users/davsclaus/Documents/workspace/minishift-hello/helloswarm/
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ helloswarm ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /Users/davsclaus/Documents/workspace/minishift-hello/hel
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ helloswarm ---
[INFO] No sources to compile
```

Deploy - Client

from timer
to http
log

- `$ mvn fabric8:deploy`

Client

```
[INFO] >>> fabric8-maven-plugin:3.5.33:deploy (default-cli) > install @ client >>>
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ client ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 1 resource
[INFO] Copying 0 resource
[INFO]
[INFO] --- fabric8-maven-plugin:3.5.33:resource (fmp) @ client ---
[INFO] F8: Running in Kubernetes mode
[INFO] F8: Running generator spring-boot
[INFO] F8: spring-boot: Using Docker image fabric8/java-jboss-openjdk8-jdk:1.2 as base / builder
[INFO] F8: fmp-controller: Adding a default Deployment
[INFO] F8: fmp-service: Adding a default service 'client' with ports [8080]
[INFO] F8: spring-boot-health-check: Adding readiness probe on port 8080, path='/health', scheme='
[INFO] F8: spring-boot-health-check: Adding liveness probe on port 8080, path='/health', scheme='H
[INFO] F8: fmp-revision-history: Adding revision history limit to 2
[INFO] F8: f8-icon: Adding icon for deployment
[INFO] F8: f8-icon: Adding icon for service
[INFO] F8: validating /Users/davsclaus/Documents/workspace/minishift-hello/client/target/classes/M
tconfig.yml resource
```

Run - Client

from timer
to http
log

Runs in foreground, tail log, undeploys when ctrl + c

- `$ mvn fabric8:run`

Client

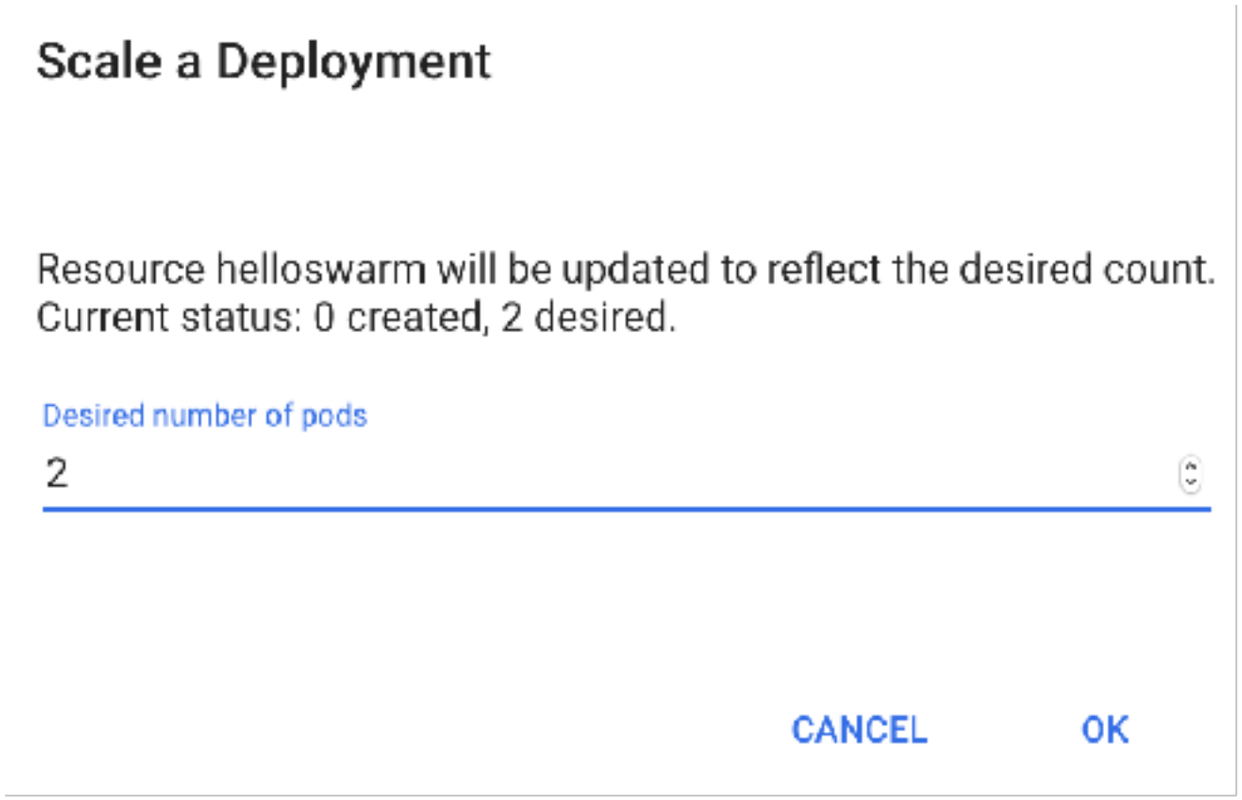
```
[INFO] F8: Scaling Deployment default/client to replicas: 1
[INFO] F8: Watching pods with selector LabelSelector(matchExpressions=[], matchLabels={app=client, provider=fab
ric8, group=com.foo}, additionalProperties={}) waiting for a running pod...
[INFO] F8:[NEW] client-76cb7c865b-fh6tv status: Running
[INFO] F8:[NEW] Tailing log of pod: client-76cb7c865b-fh6tv
[INFO] F8:[NEW] Press Ctrl-C to scale down the app and stop tailing the log
[INFO] F8:[NEW]
[INFO] F8: exec java -javaagent:/opt/agent-bond/agent-bond.jar=jolokia{{host=0.0.0.0}},jmx_exporter{{9779:/opt/
agent-bond/jmx_exporter_config.yml}} -cp . -jar /deployments/client-1.0-SNAPSHOT.jar
[INFO] F8: I> No access restrictor found, access to any MBean is allowed
[INFO] F8: Jolokia: Agent started with URL http://172.17.0.4:8778/jolokia/
[INFO] F8: 2017-12-04 15:29:52.870:INFO:ifasjipjsoejs.Server:jetty-8.y.z-SNAPSHOT
[INFO] F8: 2017-12-04 15:29:52.909:INFO:ifasjipjsoejs.AbstractConnector:Started SelectChannelConnector@0.0.0.0:
9779
[INFO] F8:
[INFO] F8:   .   ____          _            __ _ _
[INFO] F8:  /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
[INFO] F8: ( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
[INFO] F8:  \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
[INFO] F8:   '  |____| .__|_| |_|_| |_\__, | / / / /
[INFO] F8:  =========|_|==============|___/=/_/_/_/
[INFO] F8:  :: Spring Boot ::        (v1.5.8.RELEASE)
[INFO] F8:
[INFO] F8: 2017-12-04 15:29:53.880  INFO 1 --- [           main] com.foo.DemoApplication                  : Sta
```

Ready to run in Kubernetes

Scaling

- Change deployment

```
$ kubectl scale deployment helloswarm
--replicas=2
```



Scaling using the web console

Scaling

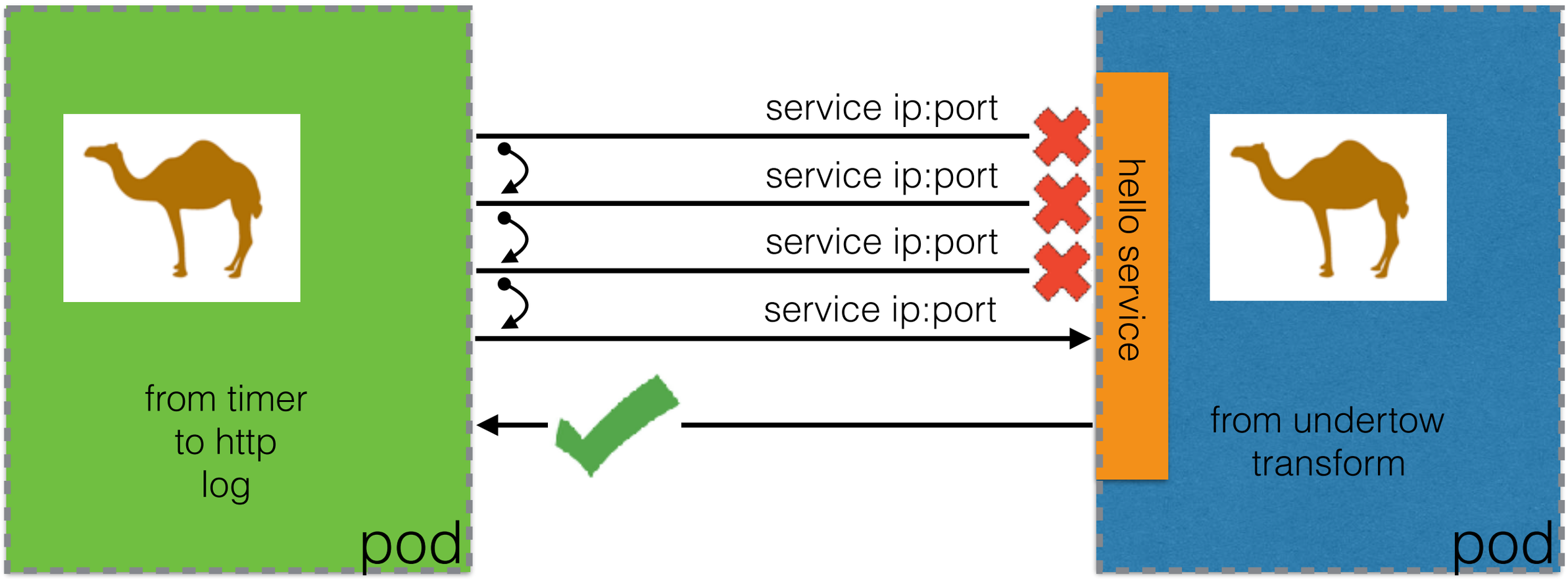
- Service load balancing

Load balancing is random

```
: Swarm says hello from helloswarm-1-bjdbj
: Swarm says hello from helloswarm-1-bjdbj
: Swarm says hello from helloswarm-1-bjdbj
: Swarm says hello from helloswarm-1-bjdbj
: Swarm says hello from helloswarm-1-s5rfh
: Swarm says hello from helloswarm-1-s5rfh
: Swarm says hello from helloswarm-1-bjdbj
: Swarm says hello from helloswarm-1-bjdbj
: Swarm says hello from helloswarm-1-bjdbj
: Swarm says hello from helloswarm-1-bjdbj
: Swarm says hello from helloswarm-1-s5rfh
: Swarm says hello from helloswarm-1-s5rfh
: Swarm says hello from helloswarm-1-s5rfh
: Swarm says hello from helloswarm-1-bjdbj
: Swarm says hello from helloswarm-1-s5rfh
: Swarm says hello from helloswarm-1-bjdbj
```

Error Handling

- Client Side Retry



Error Handling

- Client Side Retry

```
@Component
public class MyRoute extends RouteBuilder {

    @Override
    public void configure() throws Exception {
        // try to call the service again
        onException(Exception.class)
                .maximumRedeliveries(10)
                .redeliveryDelay(1000);

        from( uri: "timer:foo?period=2000")
                .to("http4:{{service:helloswarm}}/hello")
                .log("${body}");
    }
}
```

Error Handling

- Client Side Retry

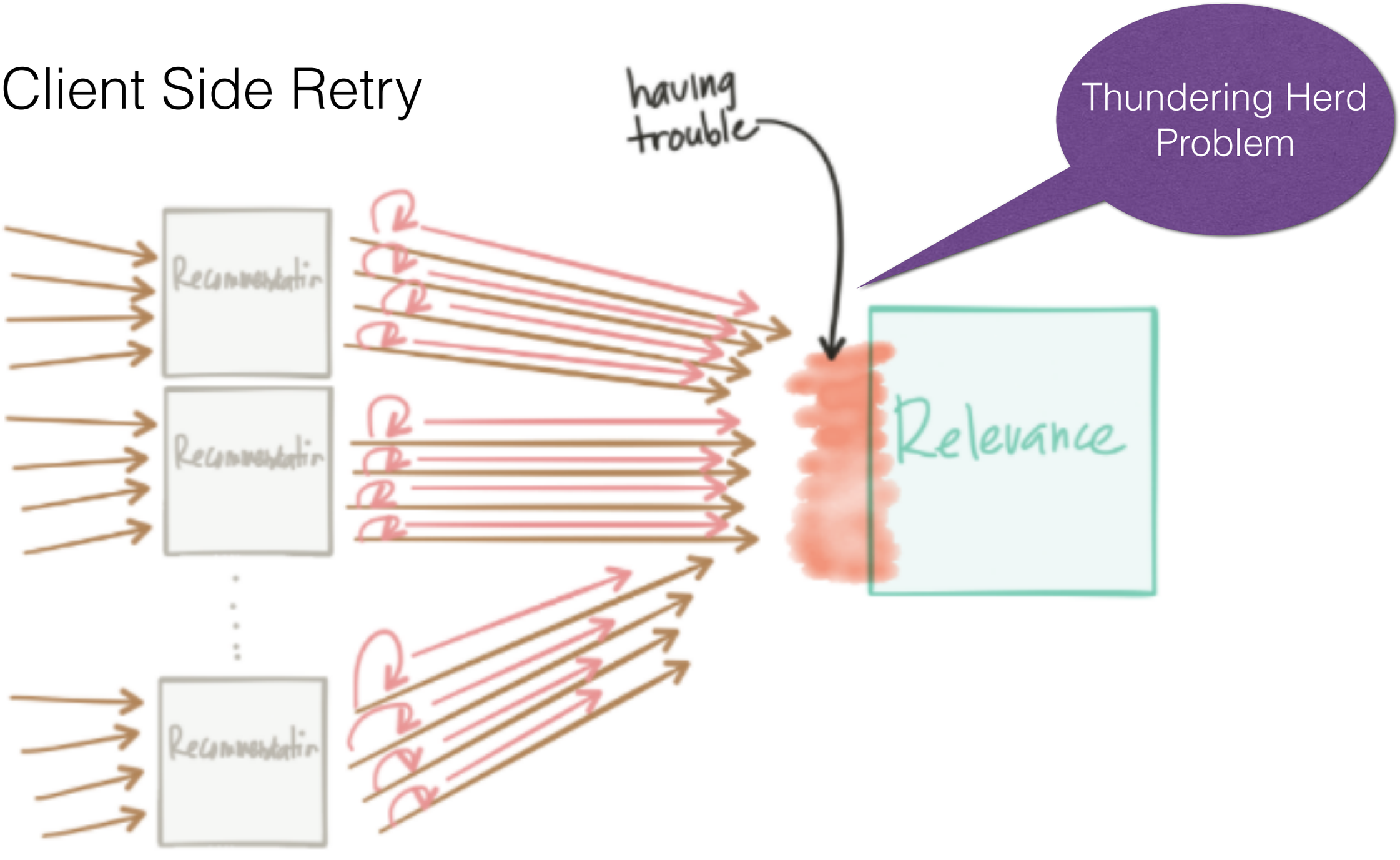
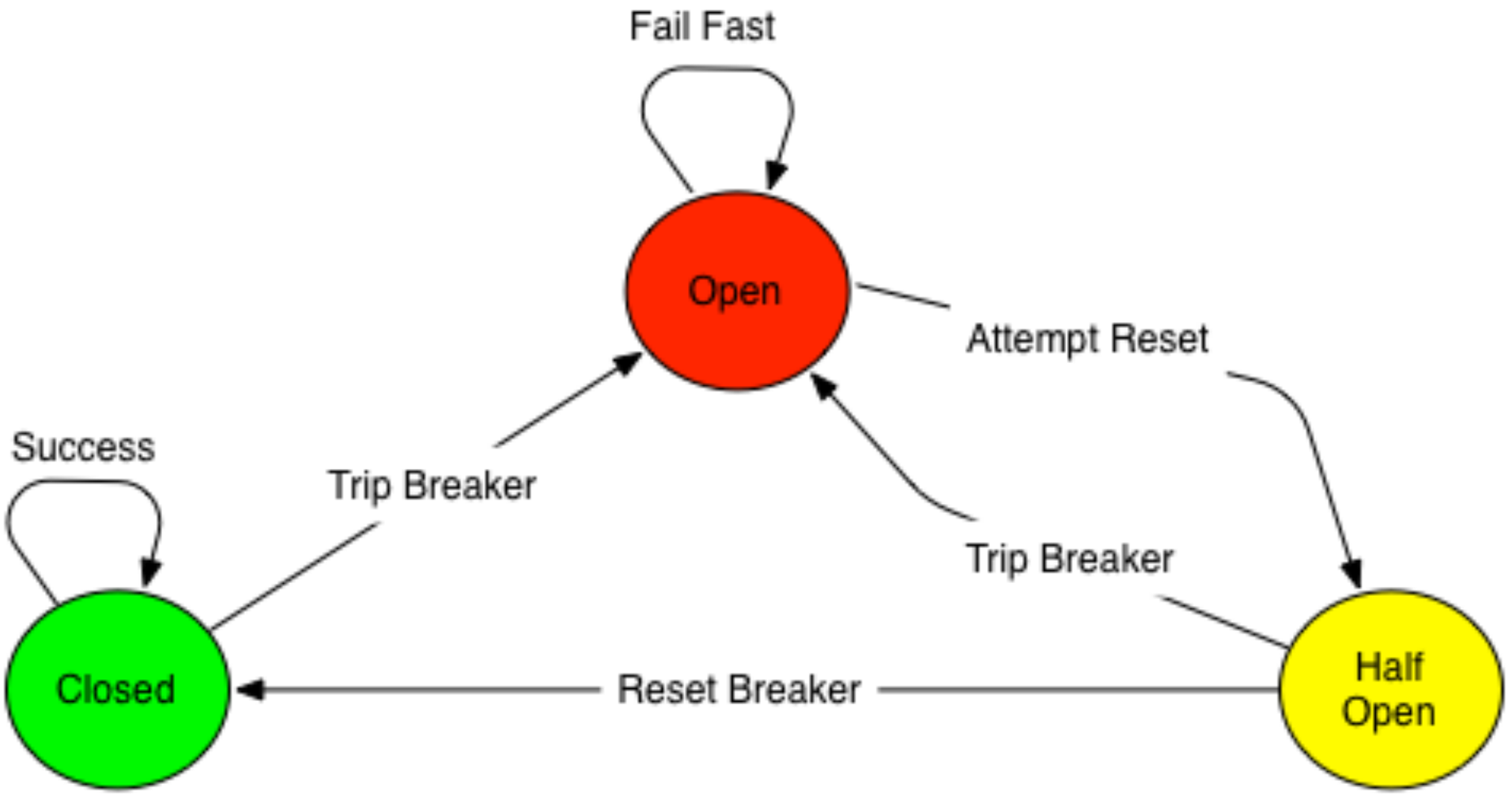


Figure by Christian Posta

Error Handling

- Client Side Circuit Breaker with Hystrix

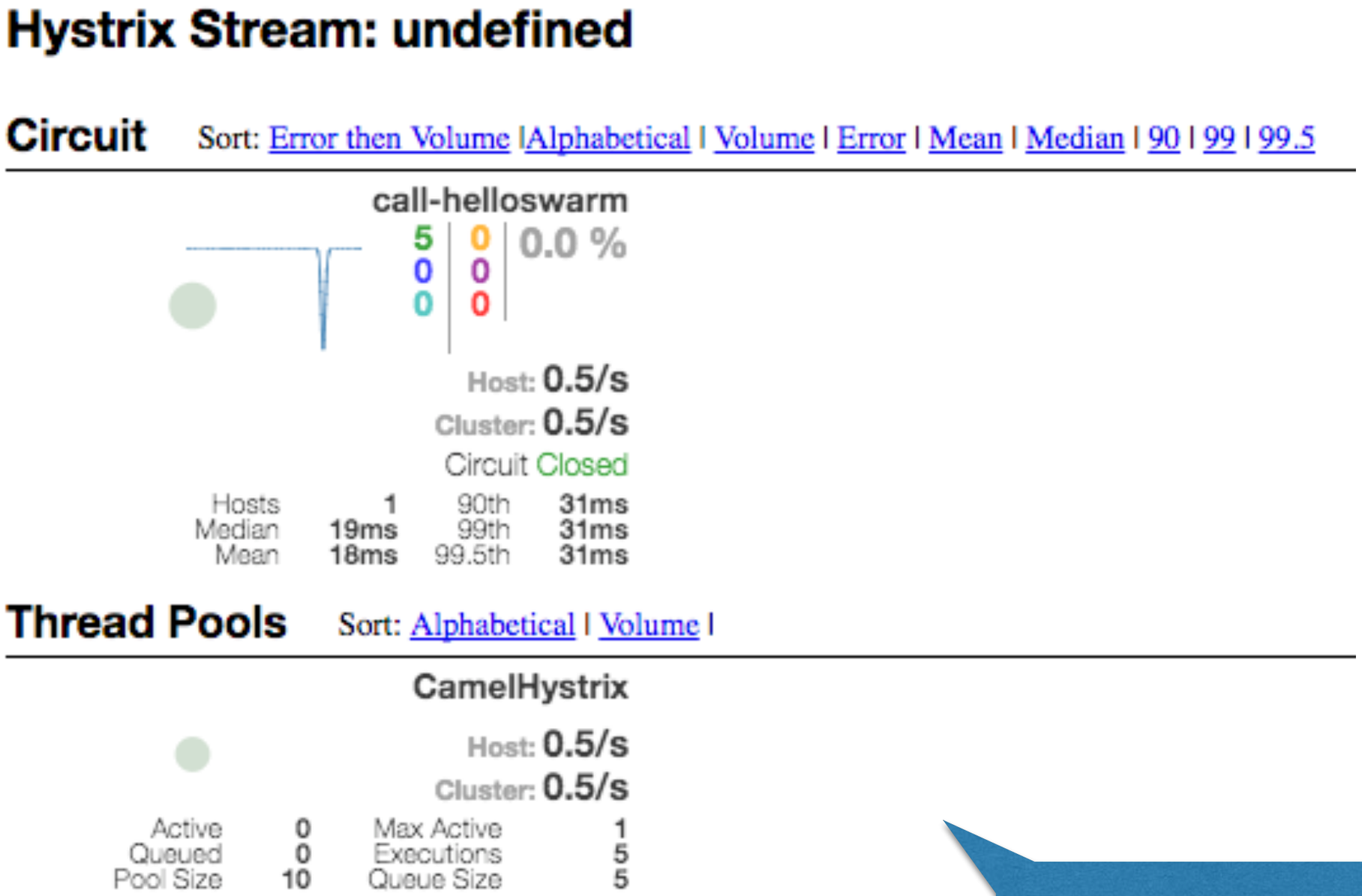


Error Handling

- Client Side Circuit Breaker with Hystrix

```
@Component
public class MyRoute extends RouteBuilder {

    @Override
    public void configure() throws Exception {
        from( uri: "timer:foo?period=2000")
            .hystrix()
                .to("http4:{{service:helloswarm}}/hello")
            .onFallback()
                .setBody().constant( value: "Nobody want to talk to me")
            .end()
                .log("${body}");
    }
}
```

Hystrix Dashboard



Instructions in readme how to install and use

Hystrix Demo

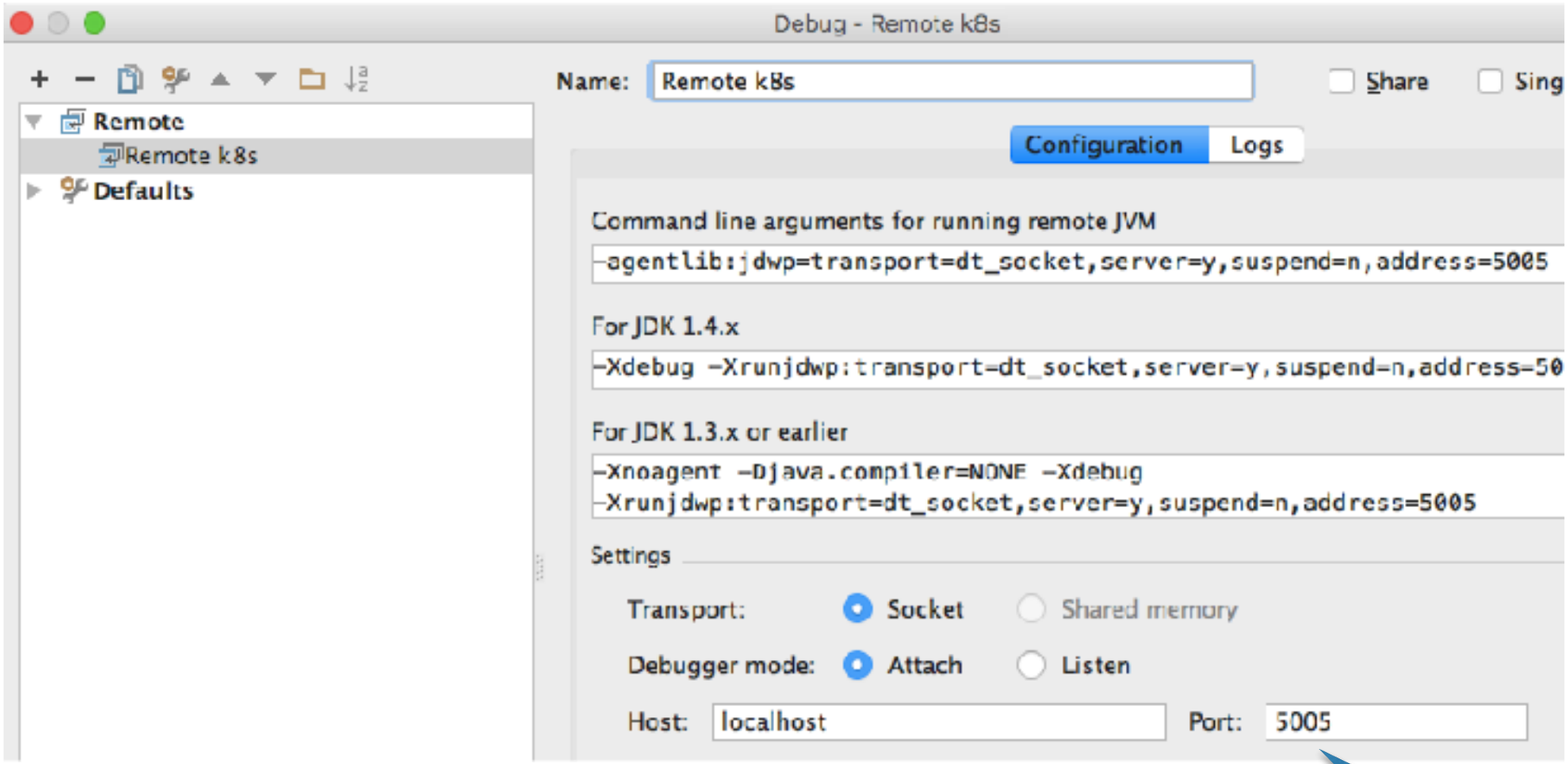
Debugging

- Debugging Pods

```
$ mvn fabric8:debug
```

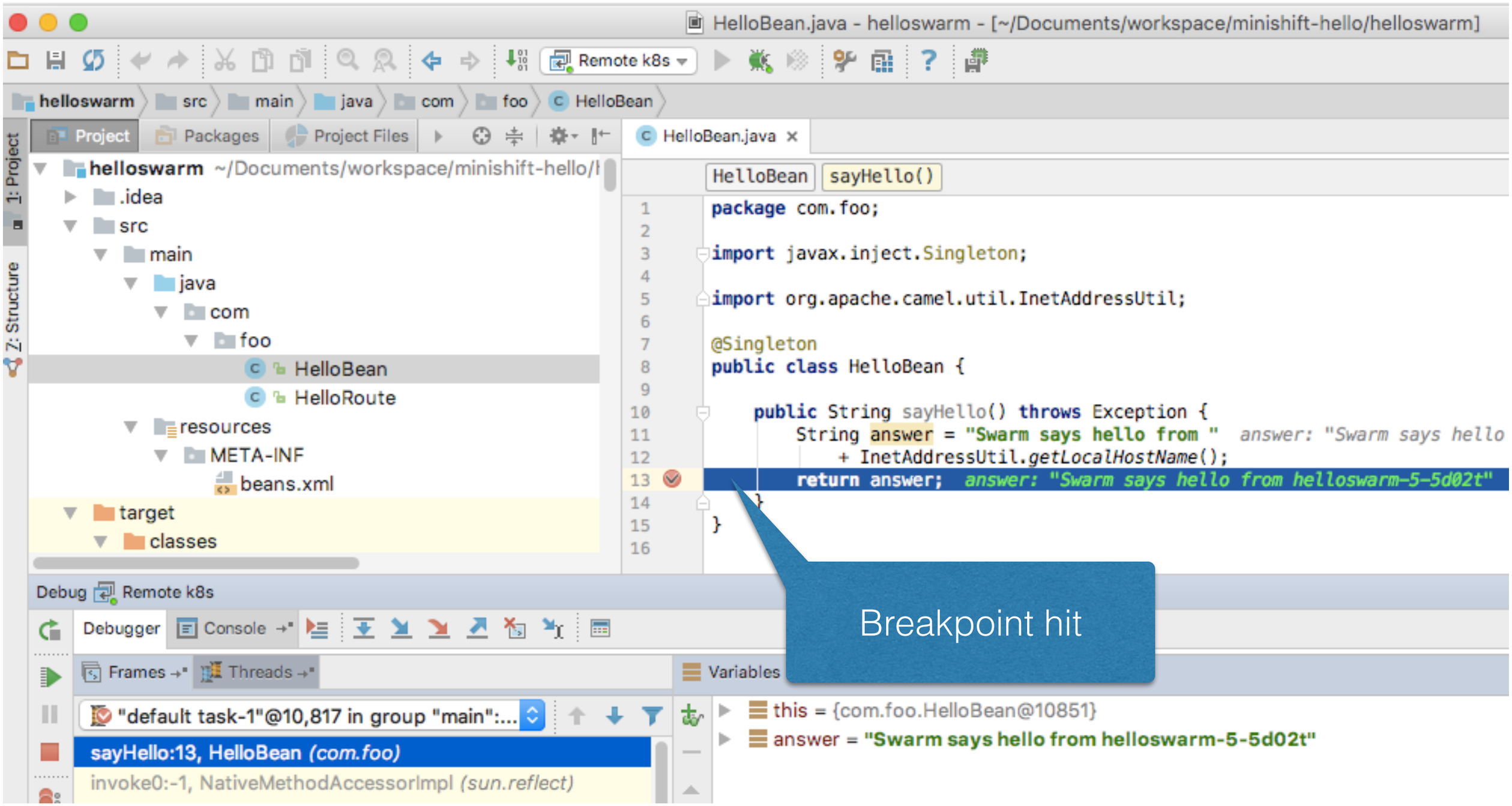
```
[INFO] F8> Port forwarding to port 5005 on pod helloswarm-1942392107-13p2p using command: kubectl
[INFO] F8> Executing command: kubectl port-forward helloswarm-1942392107-13p2p 5005:5005
[INFO] F8>
[INFO] F8> Now you can start a Remote debug execution in your IDE by using localhost and the debug port 5005
[INFO] F8>
[INFO] kubectl> Forwarding from 127.0.0.1:5005 -> 5005
[INFO] kubectl> Forwarding from [::1]:5005 -> 5005
[INFO] kubectl> Handling connection for 5005
```

Remote Debug



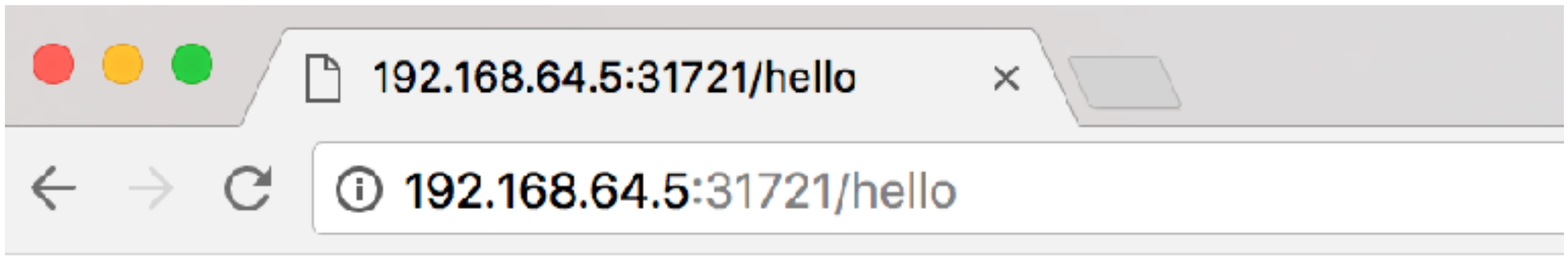
Port 5005

Remote Debug



Access Service from your computer

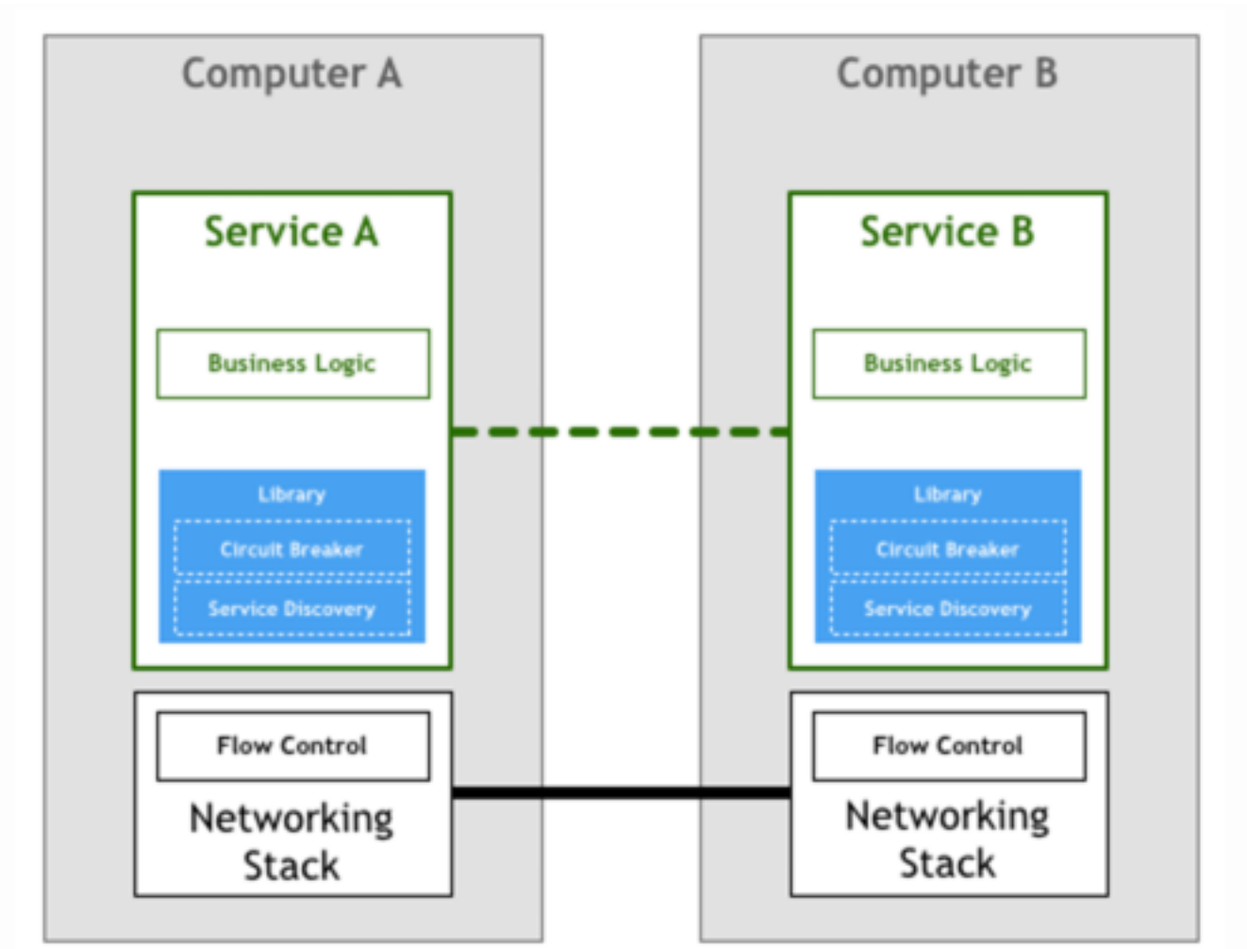
- `$ minikube service helloswarm`



Tip of Iceberg

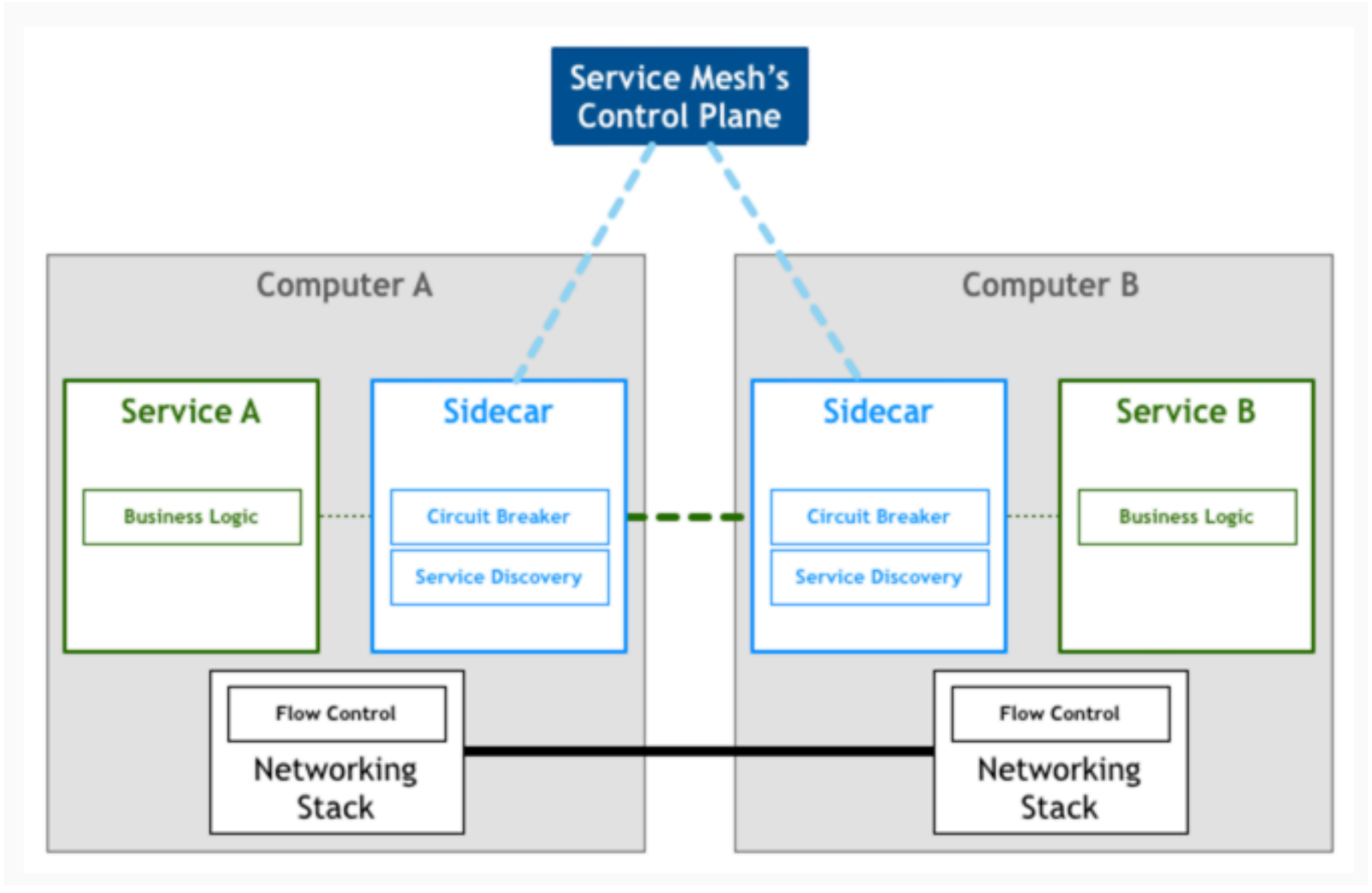
Figure by Bilgin Ibryam

Service Mesh



http://philcalcado.com/2017/08/03/pattern_service_mesh.html

Service Mesh

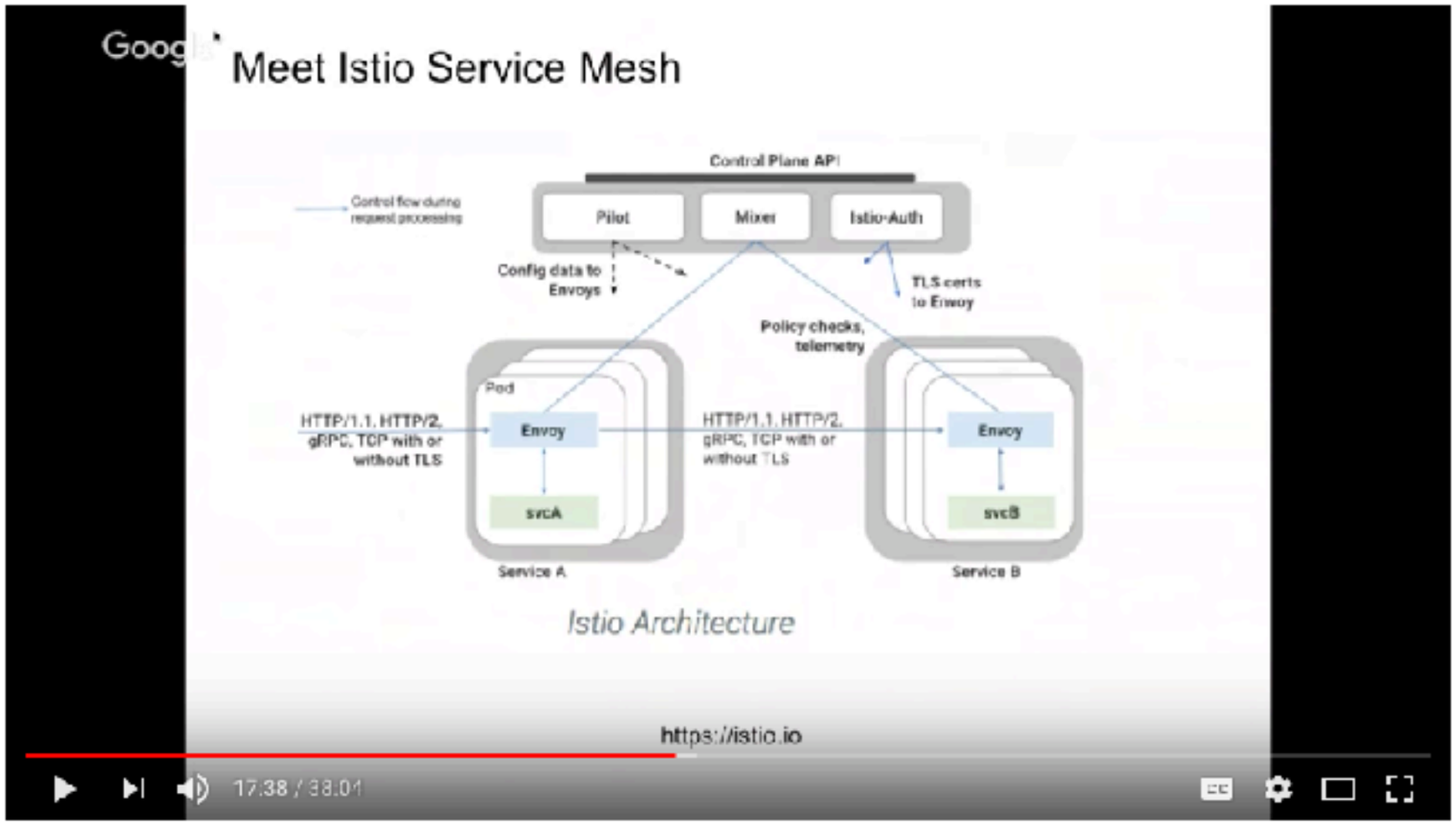


http://philcalcado.com/2017/08/03/pattern_service_mesh.html

Service Mesh Projects

Istio

Sidecars and a Service Mesh - Video



Sidecars and a Microservices Mesh

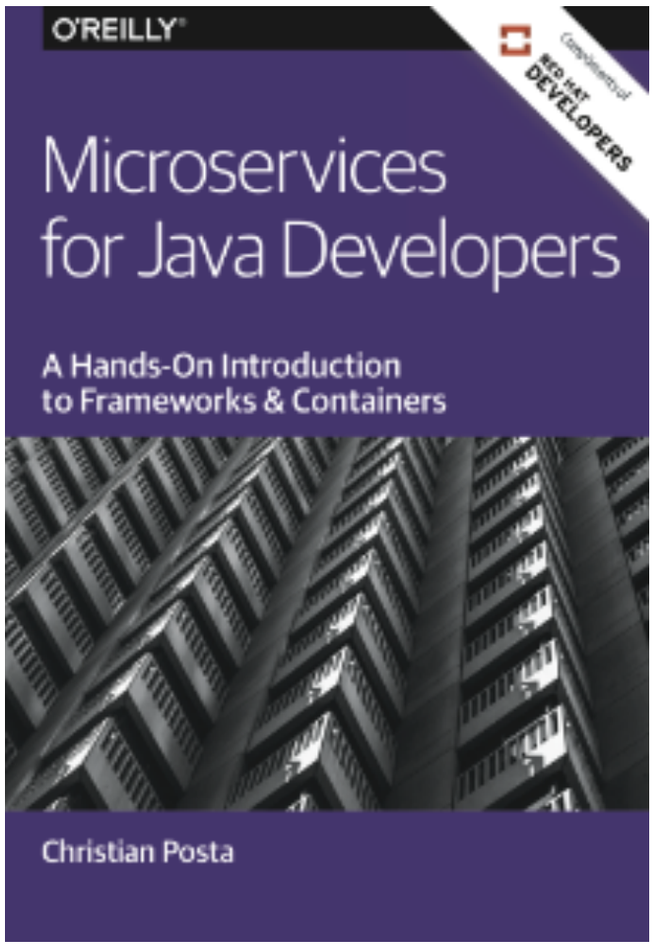
Christian Posta

@christianposta FOLLOWS YOU

Chief Architect @RedHatCloud, Autho
Developers (O'Reilly), opensource ent
Camel/ActiveMQ, @fabric8io et al

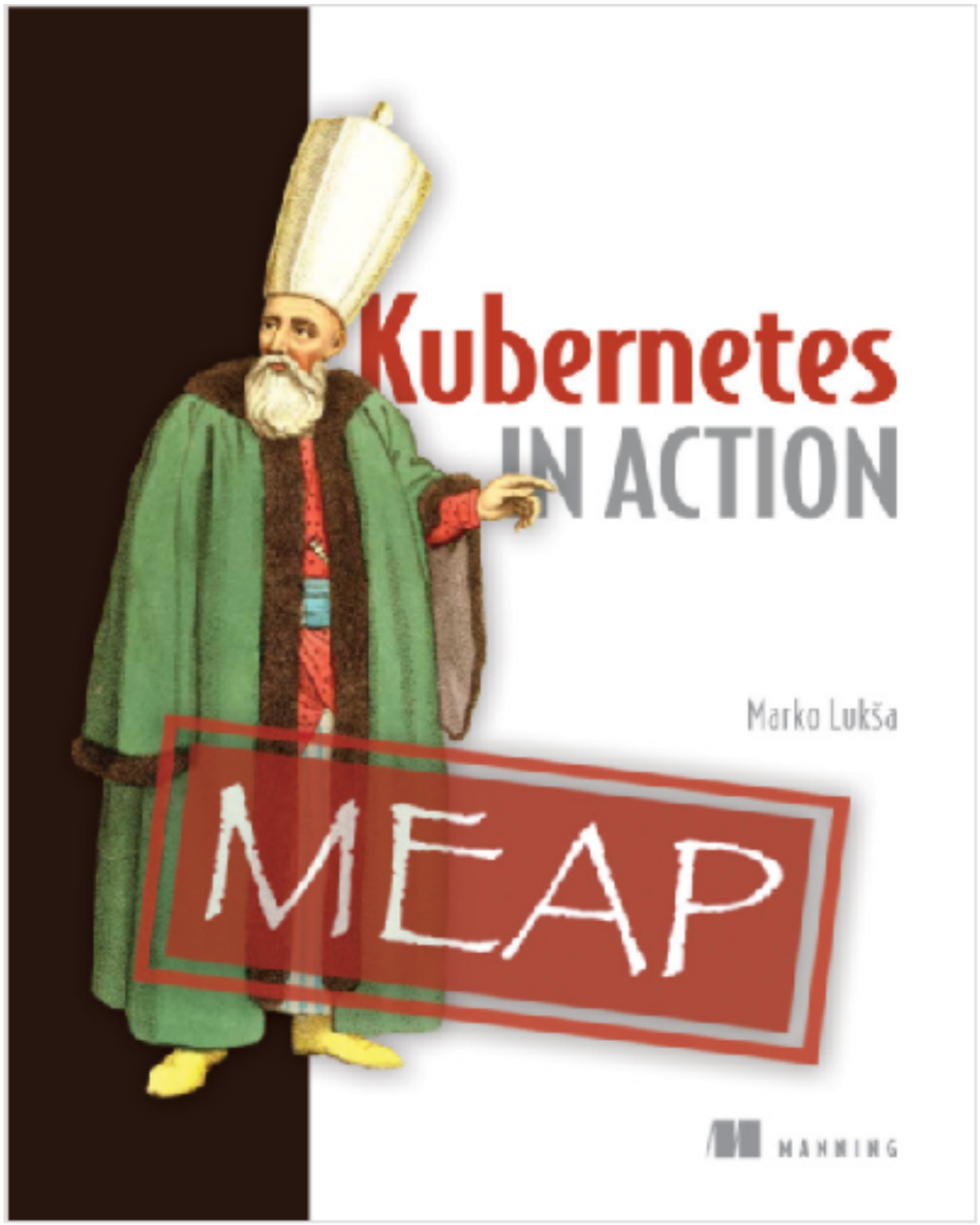
https://www.youtube.com/watch?v=ZR2GGoT_GTg

Free Book



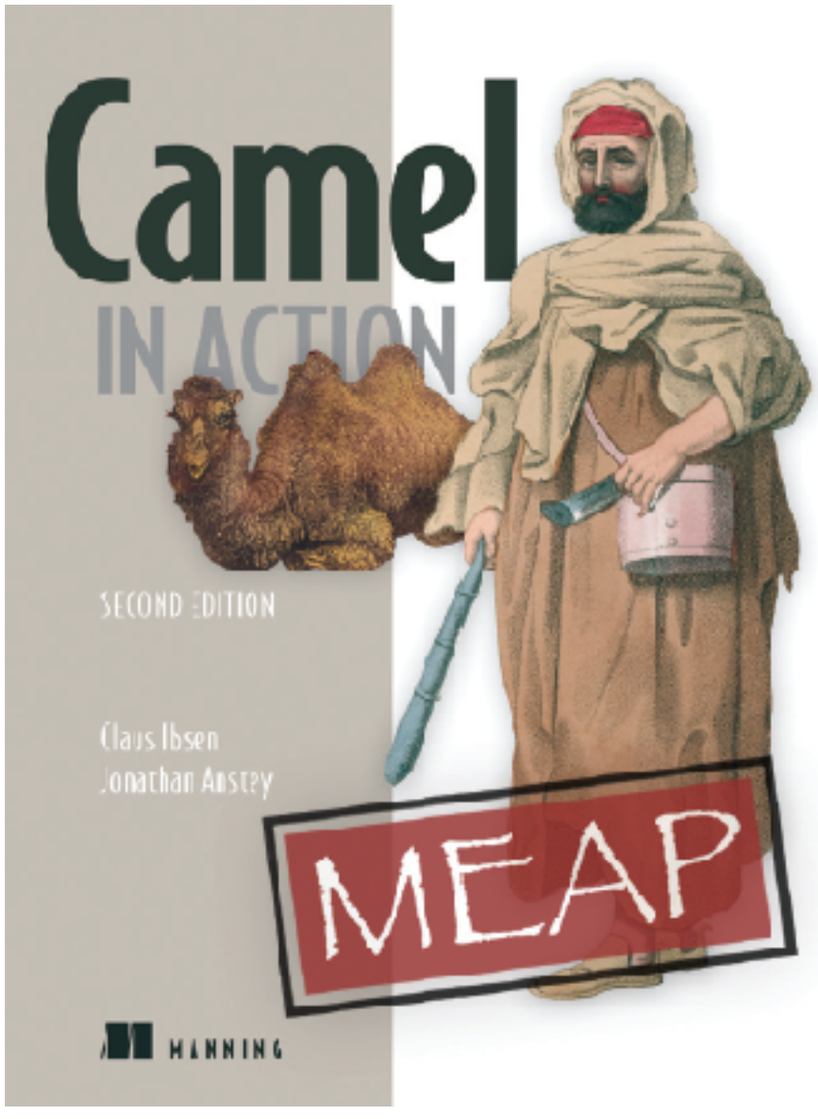
http://developers.redhat.com/promotions/microservices-for-java-developers/

Not so free Book



https://www.manning.com/books/kubernetes-in-action

Not so free Book



Claus Ibsen
@davsclaus

@ManningBooks is pushing us for a deadline on Dec 15th for Camel in Action 2nd edition 📘 just in time for an awesome x-mas present 🎁🎄

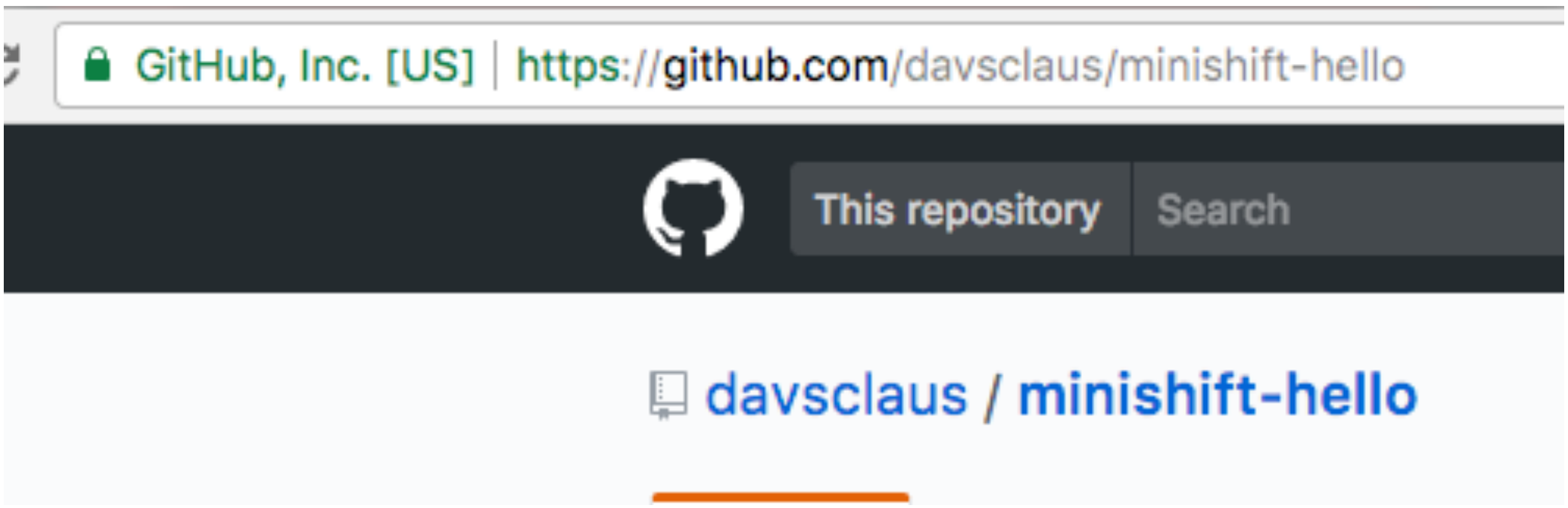
6 Dec 16.21

coupon code:
camel39

https://www.manning.com/books/ibsen2

Slides & Source Code

@davsclaus
davsclaus
www.davsclaus.com
cibsen@redhat.com



https://github.com/davsclaus/minishift-hello