Testing Camel K integrations with Cloud Native BDD
Testing Camel K integrations

with Cloud Native BDD
Speaker information

Christoph Deppisch

- Open source enthusiast
- Middleware Integration
- Test Automation
- Founder of https://citrusframework.org
- Senior Software Engineer @RedHat
- Twitter: @freaky_styley
Agenda

- Apache Camel K
- Behavior Driven Development
- Why Cloud Native BDD?
- YAKS
- Demo
- Status & future work
Testing Camel K integrations with Cloud Native BDD

Apache Camel K
What is Camel K!?  

Camel K  
A lightweight integration platform, born on Kubernetes, with serverless superpowers  

https://github.com/apache/camel-k
What is Camel K!? 

- Subproject of Apache Camel ([https://camel.apache.org/](https://camel.apache.org/))
  - Swiss knife of integration
  - 340+ components
  - >10 years of development - still one of the most active Apache projects
- Camel K
  - Started on August 31st, 2018
  - Reached version 1.0.0 on June 9, 2020
  - A platform for directly running integrations on Openshift and Kubernetes
  - Based on (ex Core OS) operator-sdk
Why integration?

My app/service
Why integration?

App A

My app/service

App C

App B
Why Camel?

![Diagram showing integration between App A, App B, and My app/service using Camel K, with Pulsar and Kubernetes icons.]

Testing Camel K integrations with Cloud Native BDD
Testing Camel K integrations with Cloud Native BDD

Why Camel?

from("pulsar://company/nsx/topic1")
  .unmarshal().json()
  .transform().simple("${body[data]}")
  .to("knative:event/activity")

from("knative:event/produced")
  .pollEnrich()
  .simple("aws2-s3://bucket/fs/${header.Ce-File}"
  .to("kafka:ext-topic")
Camel K in action

- Write an integration file (Java, Groovy, Kotlin, JS, XML...)

  from("knative:channel/xxxx")
  .transform()...
  .to("kafka:topic")

- Execute (CLI)

  $ kamel run integration.java

- Running serverless on Kubernetes
How can I test Camel K?
Behavior Driven Development
Communication
Explaining the behavior
Concrete Examples
Gherkin

Given a certain context
When some event happens
Then an outcome should occur
Why BDD?

- Write a feature file

  Given a certain context
  When some event happens
  Then an outcome should occur

- Execute (CLI)

  $ yaks test integration.feature

- Running test on Kubernetes
Why Cloud Native testing?
Why Cloud Native!?

- Tests running within the container management platform
  - Tests as integrated part of the cloud infrastructure
  - Tests able to access internal services
  - Tests can simulate service providers
  - Tests are self contained
Why Cloud Native?

Testing tools

Kubernetes / OpenShift
YAKS
What is YAKS?

YAKS

Integration test framework to enable Cloud Native BDD testing on Kubernetes

- [https://github.com/citrusframework/yaks](https://github.com/citrusframework/yaks)
- Born out of Camel K
- Current version 0.1.0 on September 29, 2020
Testing Camel K integrations with Cloud Native BDD

Framework to run Gherkin BDD feature files as Java unit tests

1st class tool support in IDE

Open Source ACL 2.0

Test framework with focus on messaging integration

Powerful validation for message content such as Json, XML, plaintext, ...

Open Source ACL 2.0
Testing Camel K integrations with Cloud Native BDD

**Feature:** Todo service

**Background:**
- **Given** URL: http://todo.service

**Scenario:** Health check
- **Given** wait for URL http://todo.service to return 200 OK
- **Then** path /health is healthy

**Scenario:** Create task
- **Given** variable id is "citrus:randomNumber(5)"
- **Given** HTTP request body:{"id": "${id}" , "task": "New task", "completed": 0}
- **When** send POST /todo/${id}
- **Then** receive HTTP 201 CREATED

**Scenario:** GET
- **When** send GET /todo/${id}
- **Then** verify HTTP response body: {"id": "${id}" , "task": "New task", "completed": 0}
- **And** receive HTTP 200 OK
$ yaks test integration.feature
YAKS architecture

Development

YAKS CLI
- generates
- install
- uses

System Under Test

Operator
- compiles
- uses

YAKS Operator
- defines
- generates
- watches
- reconciles
- watches/manages

Maven
- runtime
- invokes/verifies

Kubernetes / OpenShift

Maven registry
- references

Runtime
- http
- jms
- kafka
- knative
- openapi

OLM
- uses
- defines
- generates
Demo time!
Testing Camel K integrations with Cloud Native BDD

```
greeting-service

Http GET /en

greeting.feature
```

en: “Hello ApacheCon!”
de: “Hallo ApacheCon!”
fr: “Bonjour ApacheCon!”
it: “Ciao ApacheCon!”
esp: “Hola ApacheCon!”
Testing Camel K integrations with Cloud Native BDD

ApacheCon @Home 2020
Testing Camel K integrations with Cloud Native BDD

ApacheCon @Home 2020

Http POST /en

greeting-feature

greeting-service

{"message": "Hello ApacheCon!"}

greeting-splitter

topic: greetings

ApacheCon @Home 2020
Testing Camel K integrations with Cloud Native BDD

ApacheCon @Home 2020
Future work

- Attract community by helping to test
  - Camel K
  - Knative
  - Kafka
- Integrate test approach with CI pipelines (Tekton)
- Autodiscover infrastructure components (message brokers, APIs, DBs, ...)
- Contract driven testing (Open API)
- Visual tooling

Get involved! [https://github.com/citrusframework/yaks/issues](https://github.com/citrusframework/yaks/issues)
Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

linkedin.com/company/red-hat
youtube.com/user/RedHatVideos
facebook.com/redhatinc
twitter.com/RedHat