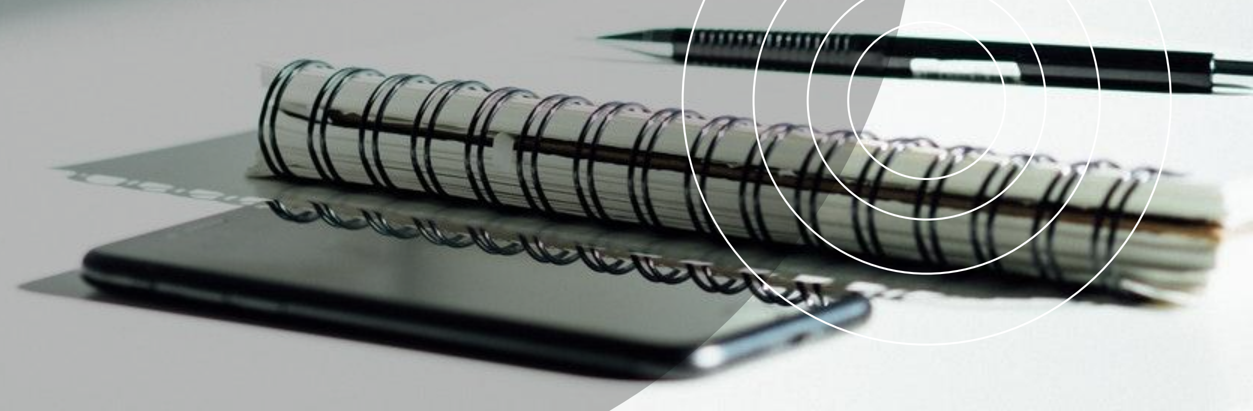
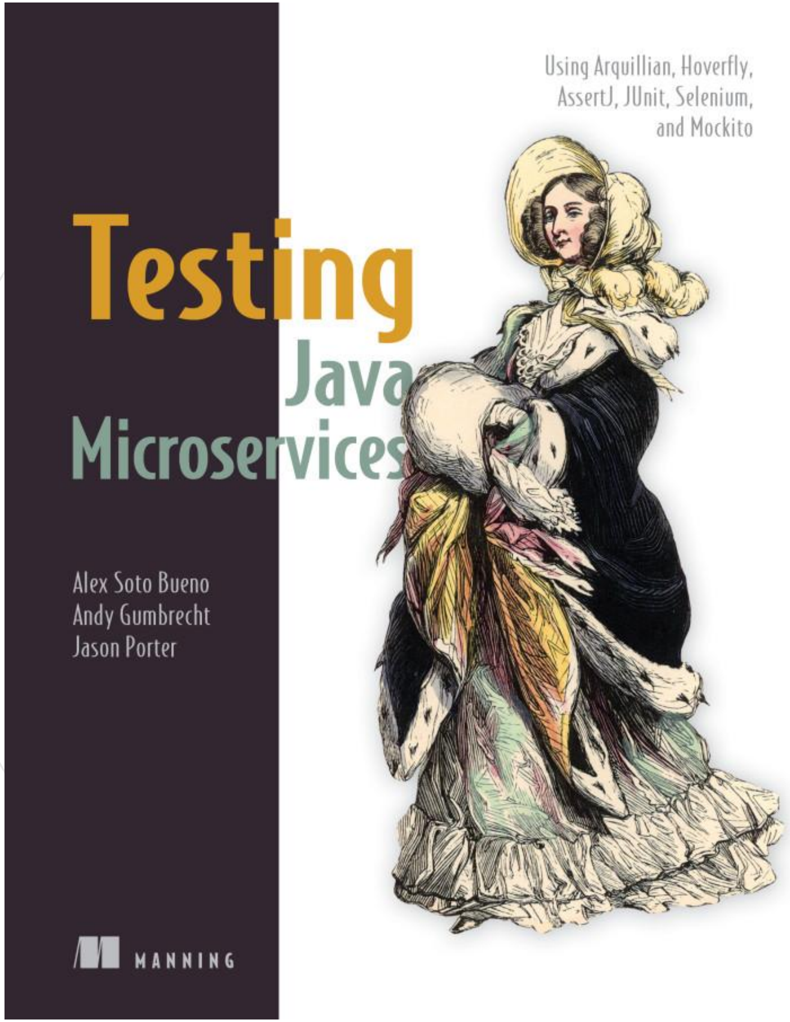
Testing Java Microservices



@AndyGeeDe

Andy Gumbrecht

@AndyGeeDe



Using Arquillian, Hoverfly, AssertJ, JUnit, Selenium, and Mockito

Testing Java Microservices

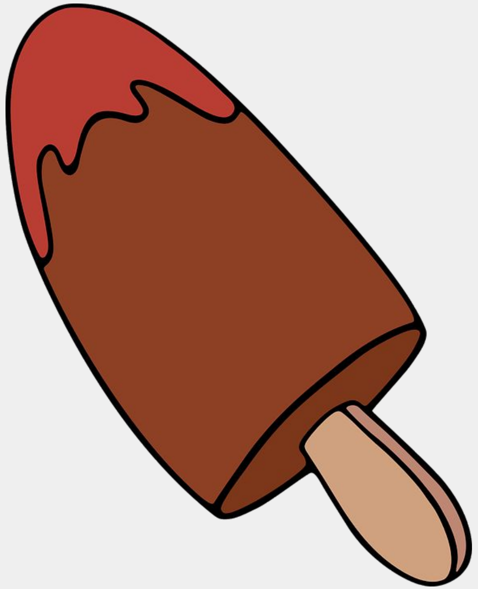
Alex Soto Bueno
Andy Gumbrecht
Jason Porter

MANNING

What is a Microservice?

What is a Microservice?

The surface area exposed by an endpoint



What is a Microservice?

The surface area exposed by an endpoint

@AndyGeeDe

What is a Microservice?

The surface area exposed by an endpoint

What is a Microservice?

The surface area exposed by an endpoint

Which can be as small as "you" want

Or as big as "you" want

What is a Microservice?

The surface area exposed by an endpoint

Which can be as small as “you” want

Or as big as “you” want

The idea is to make it scalable

It does not have to be an SCS

An SCS can expose multiple Microservices

What is a Microservice?

The surface area exposed by an endpoint
Which can be as small as "you" want
Or as big as "you" want
The idea is to make it scalable
It does not have to be an SCS
An SCS can expose multiple Microservices
It's your party, the goal is to test the beer

What is a Microservice?

The surface area exposed by an endpoint

Which can be as small as "you" want

Or as big as "you" want

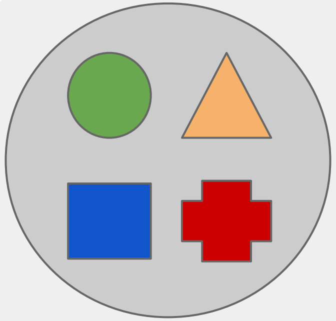
The idea is to make it scalable

It does not have to be an SCS

An SCS can expose multiple Microservices

It's your party, the goal is to test the ~~beer~~ wine

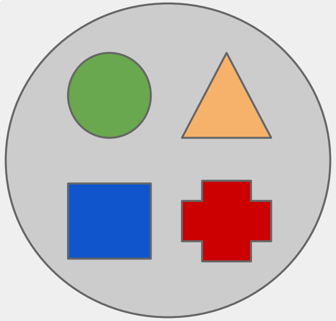
The Monolith



The Monolith

Is the entire application, packaged

Gets deployed to a node/container

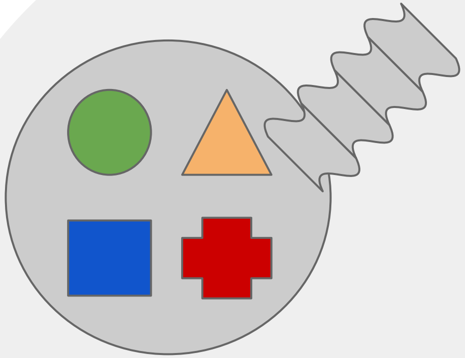


The Monolith

Is the entire application, packaged

Gets deployed to a node/container

Scaling means duplication



The Monolith

Is the entire application, packaged

Gets deployed to a node/container

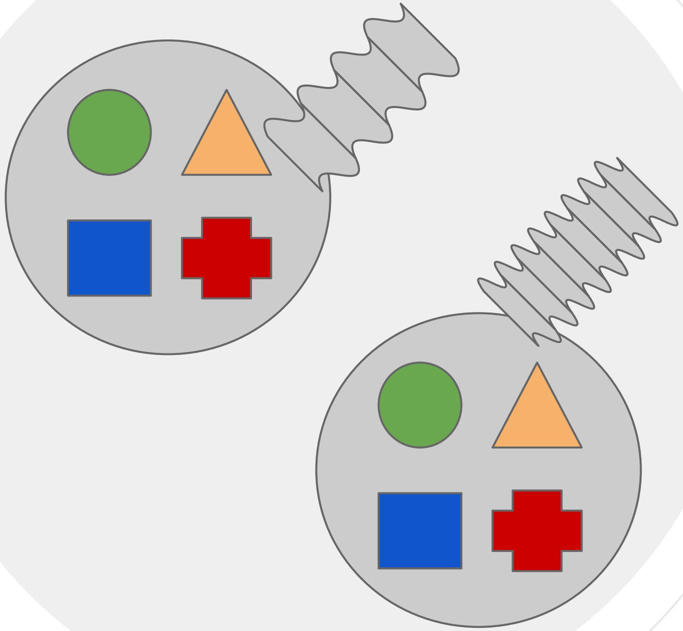
Scaling means duplication

Duplicating everything

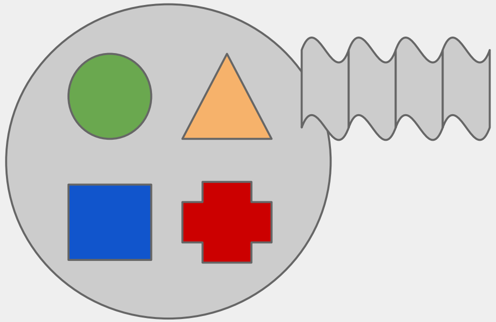
Small apps are fine

Even big apps can be fine

It’s the image that counts



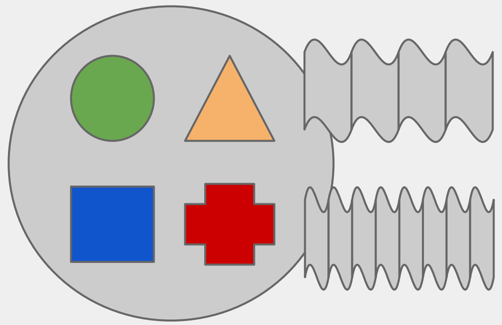
The Microservice



The Microservice

Identify the bottlenecks

Profiling requests

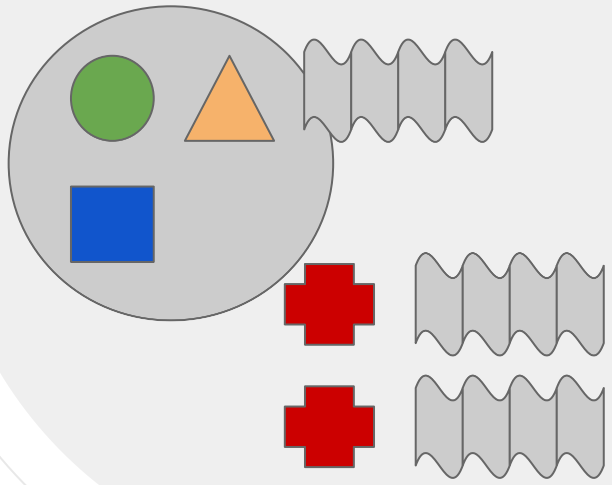


The Microservice

Extract the bottleneck

Make it elastic

Test it!



@AndyGeeDe

Test Pyramid

Test Pyramid

Unit

Smallest scope possible in order to verify functionality

No collaborators

Mocks & Co.

@AndyGeeDe

Test Pyramid

Unit

Smallest scope possible in order to verify functionality

No collaborators

Mocks & Co.

Integration

Using real collaborators to test the interaction of unit tested components

Mocks & Co.

Service Virtualization

Test Pyramid

Unit

Smallest scope possible in order to verify functionality

No collaborators

Mocks & Co.

Integration

Using real collaborators to test the interaction of unit tested components

Mocks & Co.

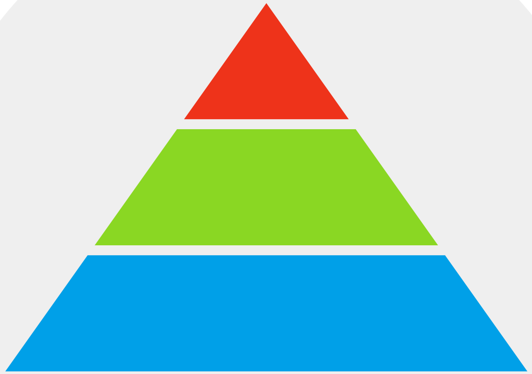
Service Virtualization

End-to-End (E2E)

Simulating the user interaction with the exposed application UI that uses all integration tested modules and collaborators

No Mocks & Co.*

UI Automation



REST Assured

Great for integration tests

```
@Test public void
lotto_resource() {

    when().
          ...
```

@AndyGeeDe

REST Assured

Great for integration tests

Understands http://localhost:8080/lotto/{id}

Easily evaluates REST responses

Find it here: **http://rest-assured.io/**

```
@Test public void
lotto_resource() {

    when().
            get("/lotto/{id}", 5).
    then().
            statusCode(200).
            body("lotto.lottoId",
             equalTo(5),
             "lotto.winners.winnerId",
             hasItems(23, 54));

  }
```

@AndyGeeDe

Service Virtualization

Captures the response of a service

Service Virtualization

Captures the response of a service

```
@ClassRule
    public static HoverflyRule hoverflyRule
=
```

@AndyGeeDe

Service Virtualization

Captures the response of a service

Stores the response locally

Replays the canned responses

Is a stubbed remote call

Can be spied upon

Can be evaluated

https://hoverfly.readthedocs.io

Hoverfly

```
@ClassRule
   public static HoverflyRule hoverflyRule
= HoverflyRule.inSimulationMode(dsl(
      service("www.my-test.com")
        .get("/api/bookings/1")
        .willReturn(
        success("{\"bookingId\":\"1\"}",
        "application/json"))
   ));
```

@AndyGeeDe

Contract Testing

@AndyGeeDe

Contract Testing

Like Service Virtualization

PACT FOUNDATION

Contract Testing

Like Service Virtualization

PACT FOUNDATION

Pact (noun):

A formal agreement between individuals or parties.
"The country negotiated a trade pact with the UK"

Synonyms: agreement, protocol, deal, contract
~ Oxford Dictionaries

@AndyGeeDe PHŒNIX CONTACT

Contract Testing

Like Service Virtualization

Record service conversation

To provide canned responses

In the form of a contract

Stored and versioned centrally

PACT FOUNDATION

Pact (noun):

A formal agreement between individuals or parties.
"The country negotiated a trade pact with the UK"

Synonyms: agreement, protocol, deal, contract
~ Oxford Dictionaries

@AndyGeeDe PHOENIX CONTACT

Contract Testing

Like Service Virtualization
Record service conversation
To provide canned responses
In the form of a contract
Stored and versioned centrally
Is found here: https://docs.pact.io/

https://github.com/DiUS/pact-workshop-jvm

PACT FOUNDATION

Pact (noun):

A formal agreement between individuals or parties.
"The country negotiated a trade pact with the UK"

Synonyms: agreement, protocol, deal, contract
~ Oxford Dictionaries

E2E Testing

E2E Testing

Probably the most important

E2E Testing

Probably the most important

Also the most difficult to write

Hardest to set up

The most time consuming

Really frickin annoying

E2E Testing

Probably the most important

Also the most difficult to write

Hardest to set up

The most time consuming

Really frickin annoying

= Fun!

docker-compose

@AndyGeeDe

docker-compose

https://docs.docker.com/compose/

Orchestration of docker containers

```
version: '3'
services:
 e2e-zookeeper:
   image: confluentinc/cp-zookeeper:5.0.0
   environment:
     - ZOOKEEPER_CLIENT_PORT=32181
     - ZOOKEEPER_TICK_TIME=2000
   ports:
     - 32181:32181
```

docker-compose

https://docs.docker.com/compose/

Orchestration of docker containers

Easy way to start multiple containers

```
version: '3'
services:
 e2e-zookeeper:
  image: confluentinc/cp-zookeeper:5.0.0
  environment:
    - ZOOKEEPER_CLIENT_PORT=32181
    - ZOOKEEPER_TICK_TIME=2000
  ports:
    - 32181:32181
```

docker-compose

https://docs.docker.com/compose/

Orchestration of docker containers

Easy way to start multiple containers

And to stop them

```
version: '3'
services:
 e2e-zookeeper:
  image: confluentinc/cp-zookeeper:5.0.0
  environment:
   - ZOOKEEPER_CLIENT_PORT=32181
   - ZOOKEEPER_TICK_TIME=2000
  ports:
   - 32181:32181
```

Testing Framework

Testing Framework

```
@ClassRule
 public static final ArquillianTestClass
        arquillianTestClass
        = new ArquillianTestClass();
```

Testing Framework

Arquillian, here to squash bugs

Java Testing Framework

Not just for E2E

Not just for EE

RedHat Project

Supports all containers.

Is found here: **http://arquillian.org/**

```
@ClassRule
 public static final ArquillianTestClass
        arquillianTestClass
        = new ArquillianTestClass();
```

The Build Pipeline

Continuous Integration (CI)

Continuous Delivery (CD)

The Build Pipeline

Continuous Integration (CI)
Continuous Delivery (CD)

Have a play with the docker container!

```
sudo docker run --detach \
          --hostname gitlab.example.com \
          --publish 8243:443 --publish 8280:80 --publish 8222:22 \
          --name gitlab \
          --restart always \
          --volume /srv/gitlab/config:/etc/gitlab \
          --volume /srv/gitlab/logs:/var/log/gitlab \
          --volume /srv/gitlab/data:/var/opt/gitlab \
          gitlab/gitlab-ce:latest
```

Thank you for your time

Happy testing!

@AndyGeeDe