IPPON

Discovery to Delivery

Building a Microservices Application with JHipster 4 and Docker in 30 Minutes

3 May 2018

Introduction

Introduction

- Don't start the timer yet!
- First I'll give an overview

BUT...IF I CAN'T DELIVER

IN 30 MINUTES OR LESS...

JHipster is FREE for all attendees!!!

Introduction

- Who we are

Introduction

- Who you are

Overview

What is JHipster?

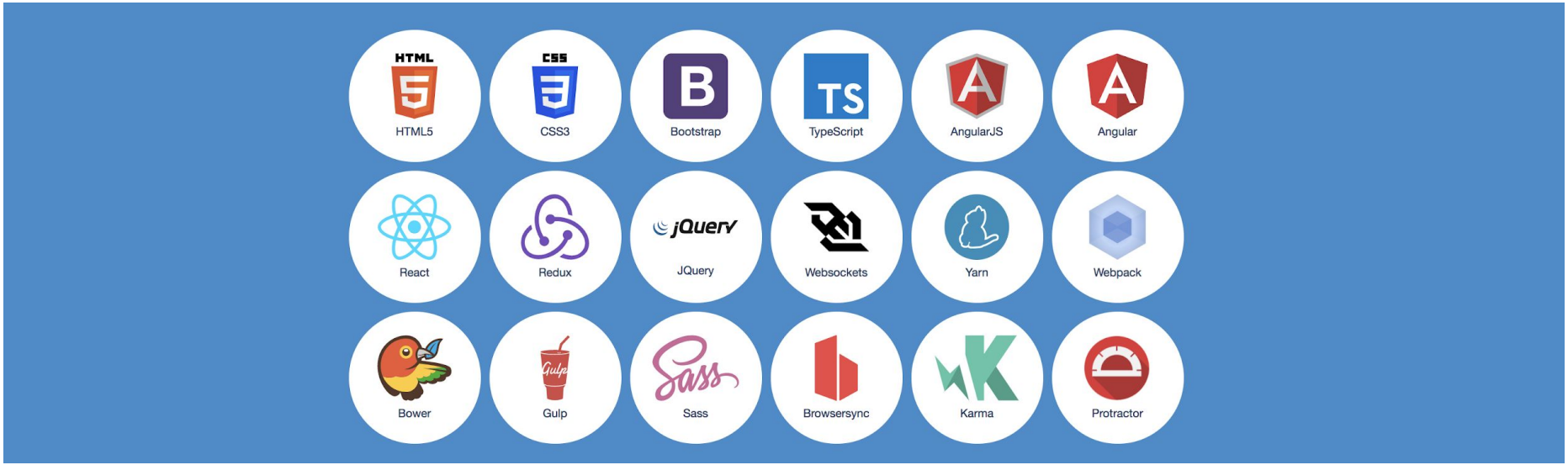
JHipster is a development platform to generate, develop and deploy Spring Boot + Angular/React Web applications and Spring microservices.

FAQ

- Is it mean.io?
  - ➔ No, it uses Java/Spring on the backend.
- Doesn't Spring Boot do that?
  - ➔ No, it uses Angular/React for front-end.
- Why does it not support Vue.js?
  - ➔ Someone always asks this. Don't be that guy. HINT: There is a Marketplace!

Server side options

Client side options



Deployment options

CI/CD options

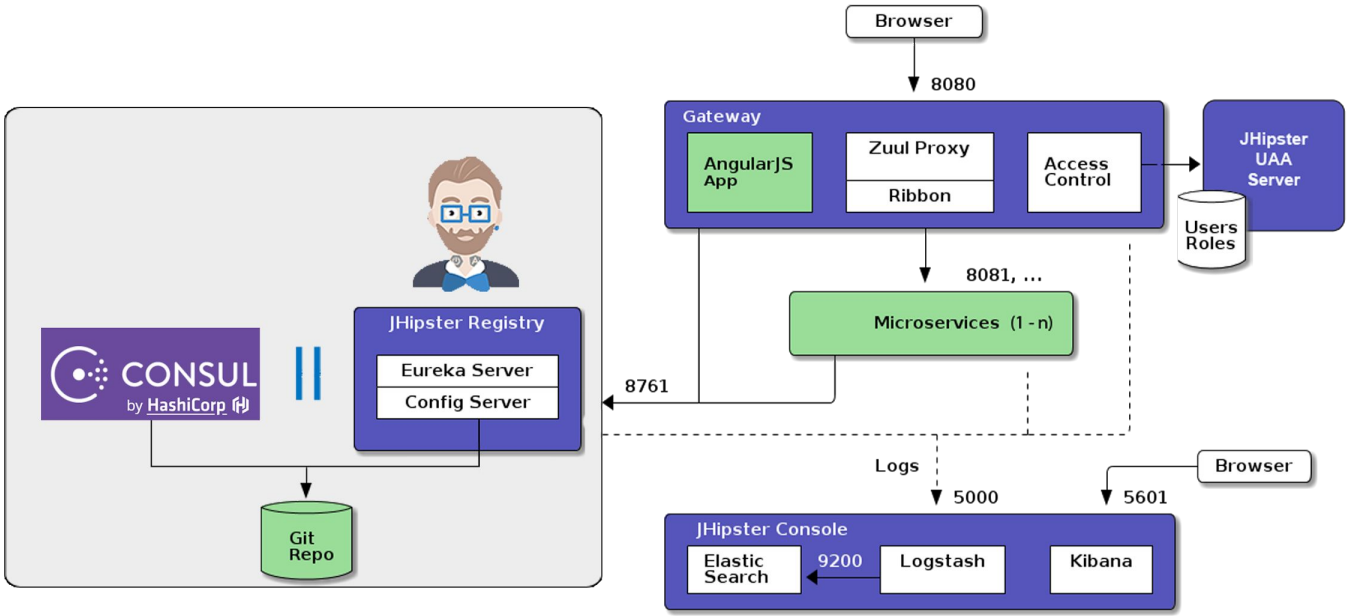
What are Microservices?

Fowler: An approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API

- Scale independently
- Teams and architecture organized around business capabilities (instead of by technical discipline)
- Smart endpoints and dumb pipes
- Teams typically develop, test, deploy and support their microservices

The big picture

NETFLIX OSS + + docker



elastic + logstash + kibana

Generate Demo

Build Dockerfile (Microservice)

```
mvn -Pprod package dockerfile:build
```

- Compiles and tests the production profile (maven and Spring)
  - ➔ Includes running all the JUnit tests
  - ➔ Creates the Dockerfile and supporting files

JHipster Microservices

- **Microservice**
  - ➔ No GUI
  - ➔ No user management code
- **Gateway**
  - ➔ A “router” to microservices
  - ➔ Load balancing and circuit breaking
  - ➔ Security and user management
  - ➔ Rate limiting
  - ➔ Generated UI based on microservices

Build Dockerfile (Gateway)

```
mvn -Pprod package dockerfile:build
```

- Compiles and tests the production profile (maven and Spring)
  - ➔ Includes running all the JUnit tests
- Compiles and tests the typescript
  - ➔ Includes running all the front-end Karma unit tests
- Creates the Dockerfile and supporting files

Gateway

https://jhipster.github.io/api-gateway/

- Routing with Netflix Zuul
- Load balancing with Netflix Ribbon
- Circuit breaker with Netflix Hystrix
- Security using JWT or OAuth2
- Documentation generation with Swagger
- Rate limiting with Bucket4j and Hazelcast

The JHipster Registry

- Runtime component provided by JHipster
- Fully Open Source (Apache 2 license)
- Service Registry based on Spring Cloud Eureka
  - ➔ All services register themselves on the JHipster Registry
  - ➔ Allows load balancing on the gateways
  - ➔ Allows microservice scalability and cluster configuration
- Configuration server based on Spring Cloud Config
  - ➔ Sends configuration data to all services
  - ➔ Useful to version, tag, rollback configurations
  - ➔ Allows to store “sensitive” information like database passwords

JHipster Console

- Monitoring console based on the ELK stack
  - ➔ Elasticsearch, Logstash, Kibana
  - ➔ Aggregates logs from microservices and gateways
  - ➔ Provides pre-defined dashboards
- Logs are sent by each JHipster application
  - ➔ Log messages sent by using the logback API: “log.debug()”
  - ➔ Dropwizard Metrics data dumped regularly to the logs, with detailed information from the JVM, Spring Beans, etc.
- Alerting is also available
  - ➔ Using Elastalert from Yelp

Demo

Scaling with Docker

- Scale the “catalog” microservice with Docker
  - ➔ Run “docker-compose scale catalog-app=2”
- A second instance of “catalog” is running
  - ➔ As it uses HazelCast, a distributed cache will be automatically configured between both instances
  - ➔ This second instance will be available in the JHipster Registry and in the gateway’s admin screen
  - ➔ It will also be automatically monitored by the JHipster Console
- You can launch new instances, and kill existing ones, to see how the architecture handles failure, circuit breaking and load balancing
- When you have finished, just run “docker-compose down” to destroy

Conclusion

- This stuff is pretty cool
- Even if you can’t use JHipster, use it to learn
- Microservices can be scaled independently and easily with Docker
- https://github.com/dsharpe/vox

PARIS
BORDEAUX
NANTES
LYON
MARRAKECH
WASHINGTON DC
NEW YORK
RICHMOND
MELBOURNE

contact@ippon.fr
www.ippon.fr - www.ippon-hosting.com - www.ippon-digital.fr
@ippontech
-
01 46 12 48 48