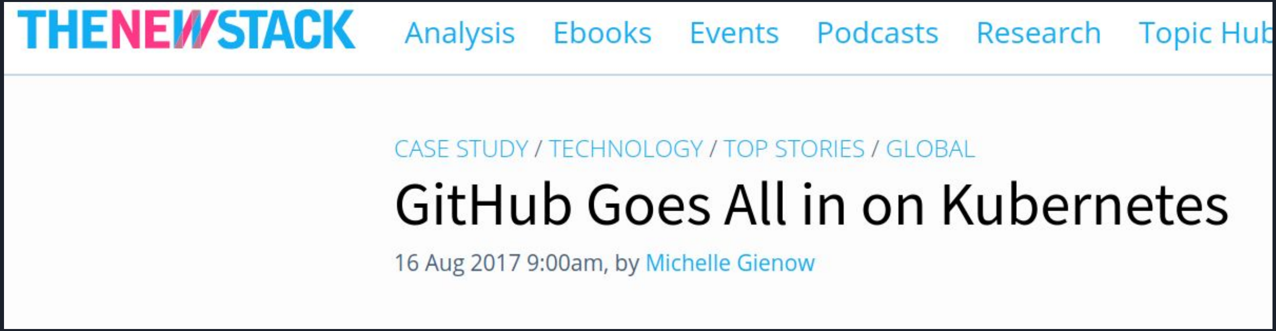
# The Self-Service Developer

In pursuit of awesome developer experiences

Laszlo Fogas
https://laszlo.cloud
@laszlocph

## Motivation

- Laszlo Fogas, DevOps Consultant
  - “elevate team cultures through automation and tooling”
  - What does it mean? Docker / Kubernetes work, CI/CD pipelines, Building testing infrastructure
- Deep in the rabbit hole, but then..
  - “GitHub Goes All in on Kubernetes”
  - Sitting on PromCon,  one talk in particular got me thinking
- I rediscovered the main reason why I jumped head first into the container ecosystem


THENEWSTACK Analysis Ebooks Events Podcasts Research Topic Hub

CASE STUDY / TECHNOLOGY / TOP STORIES / GLOBAL

GitHub Goes All in on Kubernetes

16 Aug 2017 9:00am, by Michelle Gienow

"New services could take days, weeks, or even months to deploy, depending on their complexity and our team's availability,"

"We needed to provide [..] a self-service platform [..] enables our engineers to try new things and experiment on their own when previously they would have needed to wait for the SRE team. They're no longer limited by the number of SRE's on staff."

"I can already see how the move to Kubernetes is creating an environment at GitHub for more rapid innovation — innovation that will benefit users and, ultimately, the software industry as a whole."

@laszlocph

## The self-service developer

- A developer who is able to perform all the common activities during the creation and maintenance of his/her service.
- Without asking permission..
- or support.
- Easily, through simple tooling
    - Not just for the Facebooks and Googles anymore..

## Activities like

Besides working on the codebase of course..

- Configuring the runtime environment
- Provisioning the computing resources
- Defining the CI  pipeline

## Everybody wins

- Developers can experiment and launch new ideas without coordinating with other teams / departments
- Does this mean snowflakes that ops has to fix in the middle of the night?
- On the contrary! This is built by ops. For devs.
- So Ops people can focus on building platforms, resiliency etc, instead of supporting developers
- Less constraints to navigate for managers

Recent innovation in the container and CI/CD space allowed this vision to be achievable.

## What are these innovations? 1/3

- Configuring the application runtime environment
  - Docker
    - Dockerfile is a simple text based file
    - Versioned with your application code

    -
      ```
      FROM anapsix/alpine-java
      ADD gotocon-1.0-SNAPSHOT.jar .
      EXPOSE 8080
      CMD java -jar gotocon-1.0-SNAPSHOT.jar
      ```

## What are these innovations? 2/3

- Provisioning the computing resources of a new service
  - Container orchestration, like Kubernetes
    - Service manifest file aka “yamls”, in version control
    - Define all aspects of the deployment topology: quantity, placement rules, resource needs
      The “scheduler” then puts on an appropriate node in the cluster

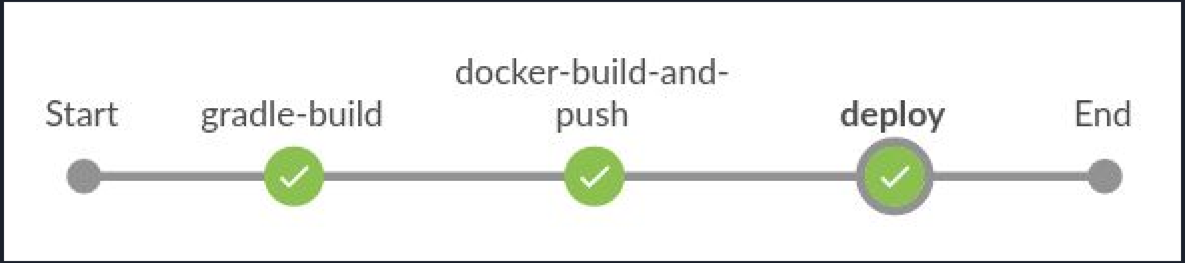
    -
      ```
      kind: Deployment
       spec:
         replicas: 1
         template:
           containers:
             - name: gotocon
               image: laszlocloud/gotocon
               resources:
                 limits:
                   memory: 1G
      ```

@laszlocph

## What are these innovations? 3/3

- Defining the CI pipeline for the app
    - Pipeline as code
    - Simple text based syntax again, versioned together with your application code
    - Now CI tools have been with us for some time Jenkins, Bamboo, etc
    - Minimum requirements in 2017
        - Pipeline as code
        - Independent, reproducible builds. Build steps run in Docker containers
    - Not a tall order: Drone.io, Gitlab CI, Jenkins BlueOcean, CircleCI

```
pipeline {
   agent none
   stages {
       stage('gradle-build') {
           agent { docker 'gradle:4.2.0-jdk8-alpine' }
           steps {
               sh 'gradle clean build'
               stash includes: 'build/libs/gotocon-1.0-SNAPSHOT.jar', name: 'gotocon-jar'
           }
       }
       stage('docker-build-and-push') {
           agent {
               docker {
                   image 'docker'
               }
           }
           steps {
               unstash 'gotocon-jar'
               sh 'cp build/libs/gotocon-1.0-SNAPSHOT.jar docker/'
               sh 'docker build -t laszlocloud/gotocon docker/'
               sh 'docker login -u $DOCKERHUB_USR -p $DOCKERHUB_PSW'
```



@laszlocph

## Recap: Elements of control

- Dockerfile
- Yaml service definitions for Kubernetes
    - Deployment
    - Service
    - Ingress
- Pipeline as code

⇒ Show me the code
⇒ Let’s deploy it on a K8s cluster..  and change a few things

## Recap

Configure the runtime environment (Dockerfile)

+

Provision the computing resources (Container orchestration)

+

Define the CI pipeline (Pipeline as Code)

+

In simple Yaml files

+

All in version control

=

You are in control

You know.. for speed. And profit!

@laszlocph