Ruby to JRuby and beyond

We'll talk about:

- Ruby, concurrency & the JVM
- process of innovation in an agile environment
- introducing docker for applications & services
- future outlook for a specific docker use case

We won't talk about:

- The Silver bullet for docker migrations
- How awesome microservices are

Me, myself & Elmar

Hi, I'm Boyan!

- A software developer **ba dum tss**
- Working with Docker since 2013
- Clojure, Scala, PHP, Javascript…
- With *Elmar* for about 7 months now

Hi, we're Elmar Reizen!

- Online Travel Agency
- 20+ year history
- ***Agile***
- Scala, Clojure, Ruby…

ELMAR
de Vakantie Discounter

The story so far:

- Clojurescript frontend
- Scala Spark App

- Scala “Play” API
- Polyglot tools/utls

RoR business logic customer facing application

The story so far…continued

- Hardware? Baremetal (< 50 machines)
- Provisioning & configuration
- Devops attitude ready
- Different teams, different needs

Where is our value

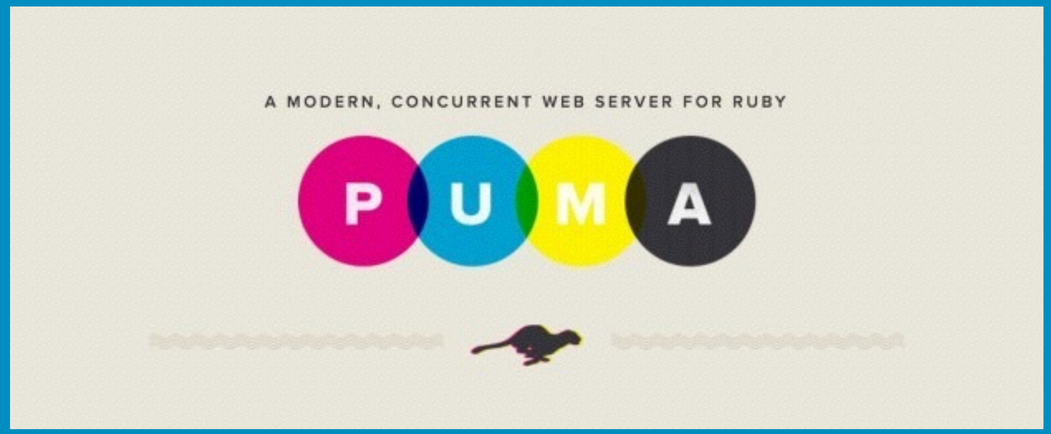
- Ruby on the way out, hello concurrency
- Server side rendering made easy (j2v8)
- Unifying frontend/backend stack
- Faster iteration for tech adoption
- Service application environment separation
- Per project/team infrastructure responsibility
- Phase out Puppet & Capistrano
- We <3 the JVM

The Plan

- MigrateRuby to JRuby
- Bundle to Jar, run with Docker
- Enter Clojure
- Move to computational resources management framework

The Migration

- Ruby -> JRuby
- Can we package our app in Docker?
- Can we run a docker container?
- Do we need to run our own Docker registry?
- What is our rollback strategy?



The Strategy

- **make <target>**
  - Log to build machine
  - (send notifications about deployment)
  - Checkout revision
  - Build docker image and push to registry.
  - Loop through production cluster machines, pull & retag previous/current
  - Stop & remove container
  - Run new container, wait till up or signal error
  - Send out deployment summary notification.

The Pains

- Don’t skip on knowledge sharing
- Saved by —build-args
- Logs & statistics matter
- Waiting on automatic Service Discovery

The Gains

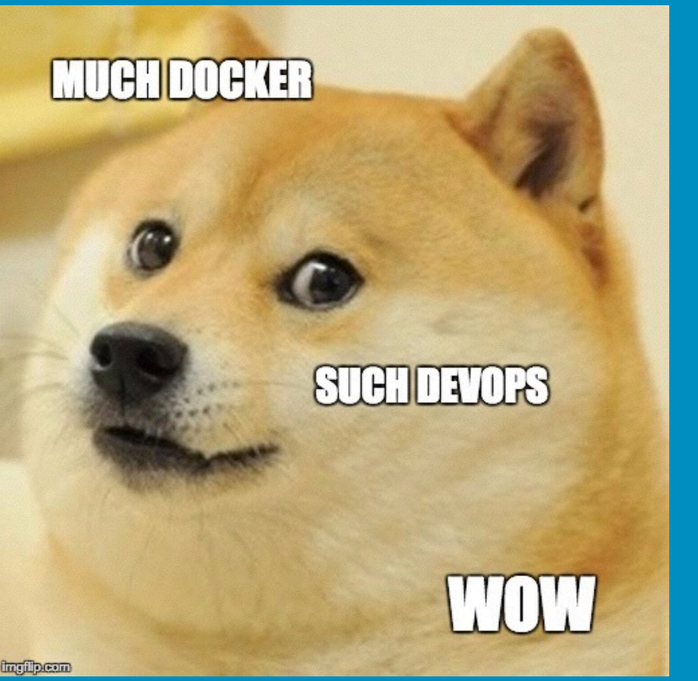
- Fast rollback
- Consistent version distribution
- No environment interaction
- Dockerize all the things

ELMAR

de Vakantie Discounter

The Verdict

- Environment control
- Performance
- Reproducibility
- Feature versioning
- New service deployment is easier
- 10/10 - will dock again



ELMAR
de Vakantie Discounter

What next?

- Explore Kubernetes & Mesos
- Move Scala API to Docker (sbt plugin)
- Move out non-persistent services to docker (redis)
- Run persistent services within Docker - still a maybe

ELMAR

de Vakantie Discounter

Questions?