Ouch!

(what I learned from being hit with a Serverless Ruby boomerang)

Ewan Slater
@ewanslater

Cloud Architect

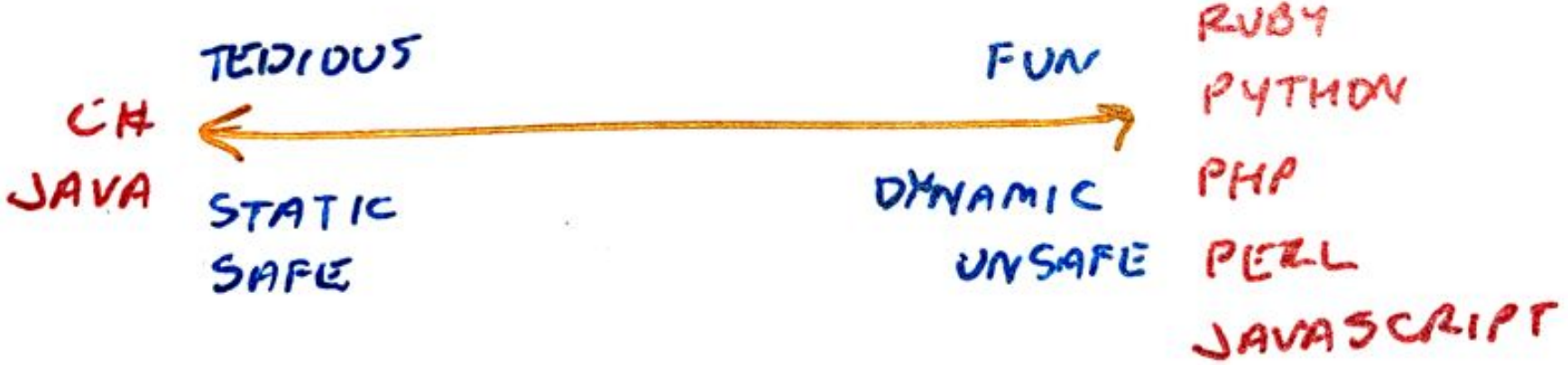
Accidental Rubyist

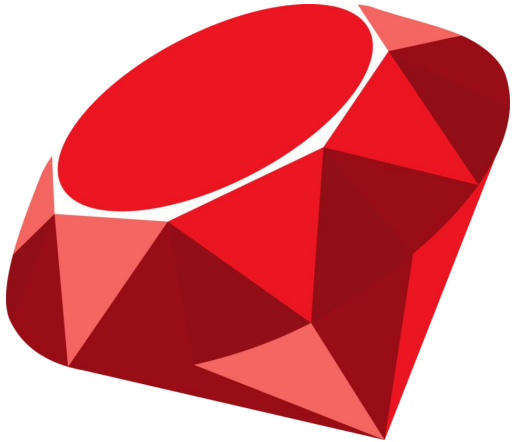

DANGER
UNSAFE
DO NOT USE


solaris

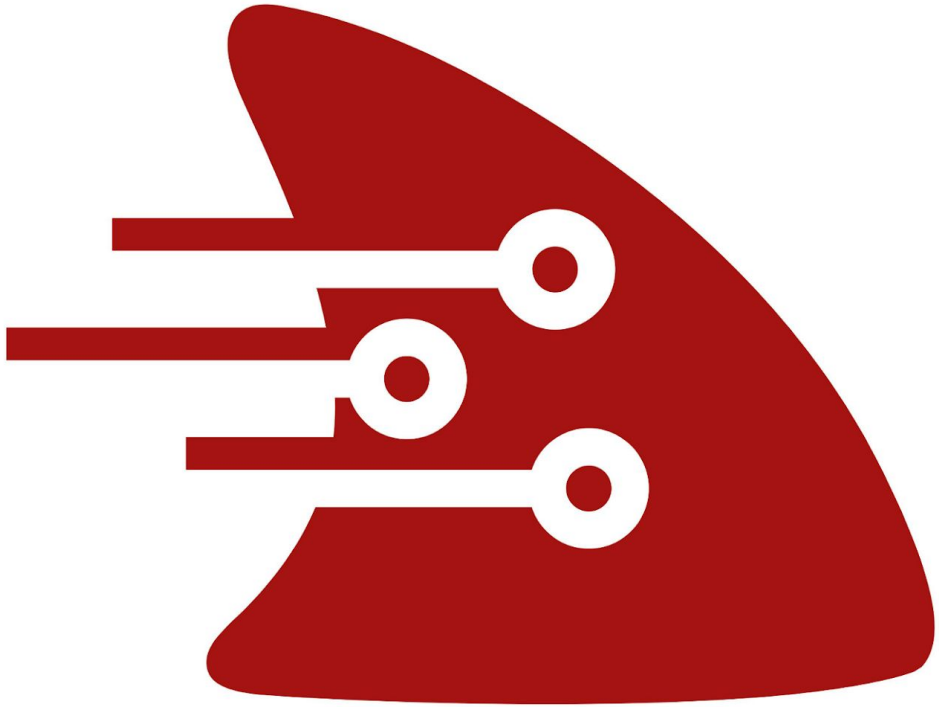
vs

Linux

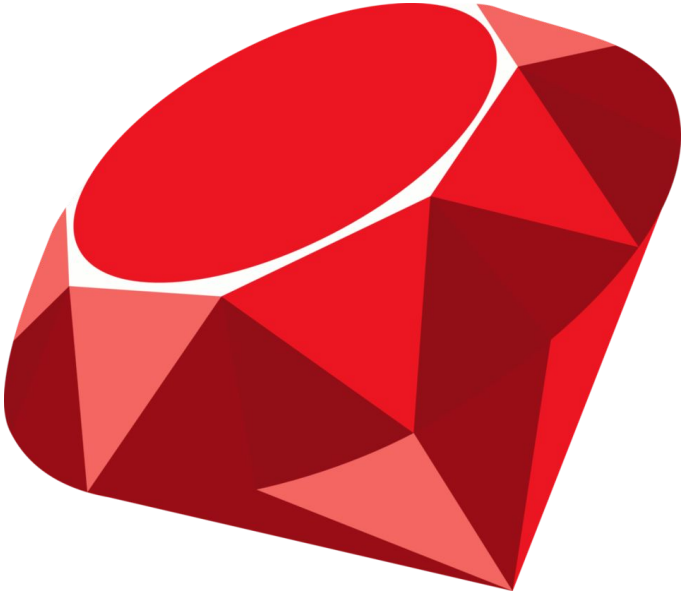

TEDIOUS
FUN
C
JAVA
STATIC
SAFE
DYNAMIC
UNSAFE
RUBY
PYTHON
PHP
PERL
JAVASCRIPT




FANATIC
SNOWBOARDS



fn



CNCF Definition

“**Serverless** computing refers to the concept of building and running **applications** that **do not require server management.**

It describes a finer-grained **deployment model** where **applications**, bundled as one or more **functions**, are uploaded to a **platform** and then **executed**, scaled, and billed in response to the **exact demand** needed **at the moment**.”

- CNCF Serverless Whitepaper

I Am Devloper
@iamdevloper

Serverless is server less in the same way a beef burger is vegetarian because you didn't personally see the cow being killed.

11:16 AM · May 1, 2019 · Twitter Web App

Abstraction

What is “Serverless”?

- **Serverless** is an abstraction of infrastructure and its operations including provisioning, scaling, patching, etc.
- **Serverless architecture** is when an app is built entirely on serverless components (compute, storage, networking)
- **Functions (or Functions as a Service)** is the compute component in a serverless architecture

FaaS

- Write small functions
- Do one thing well
- Easy to Understand
- Easy to Maintain
- Run on Serverless platform
  - Only consume resources at run time

Avoid


DID YOU FINISH WRITING THE SOFTWARE?
NO.
Dilbert.com
@ScottAdamsSays
I SPENT THE LAST THREE DAYS SETTING UP MY PROGRAMMING ENVIRONMENT.
1-2-17 © 2017 Scott Adams, Inc./Dist. by Andrews McMeel
SO... YOU'VE DONE... NOTHING?
NOTHING YOU'D UNDER-STAND.

Serverless Upsides

- Make development easier
- Improve developer productivity
- Increased agility (Dev & Business)
- Reduce costs (Dev & Operations)

Serverless Downsides

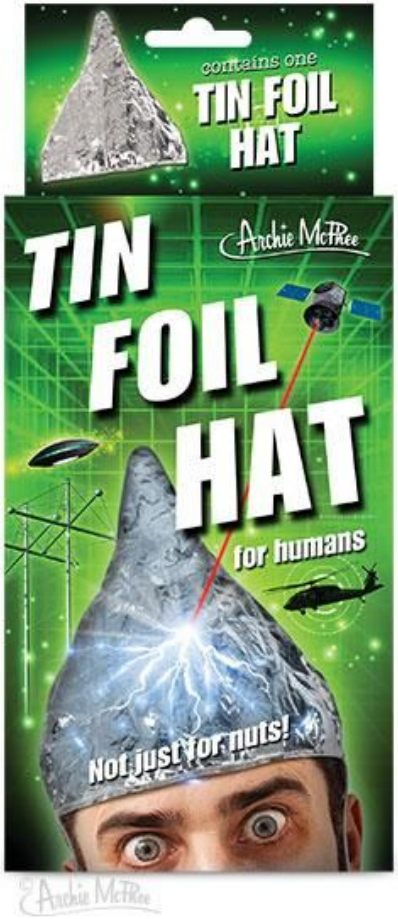
- Shiny
- ball_of_mud++
- Lock - in
- Restricted choice
  - Language
  - Run - time environment
- Prescriptive / too opinionated

“Lock - in” = “Switching Costs”

Risk

“Lock - in” = “Switching Costs” * Risk


Mother Should
I Trust
The Government?


contains one
TIN FOIL HAT
Archie McPhee
TIN FOIL HAT
for humans
Not just for nuts!
© Archie McPhee

Freedom

- Language(s)
- Runtime
- Vendor independence
- Portability (multi - cloud / on premises)
- Decentralisation
- Privacy


OFF

Open Source Serverless

Language

WTF!? No Ruby?

(or Rust, or Erlang, or FORTRAN)

The Fn Project (fnproject.io)

- Open-source serverless compute platform
- Can be deployed to any cloud or on - premises
- Containers are primitives
- Language agnostic
- Active w/ large core team, 3500+ commits, 75+ contributors
- Native CloudEvents support
- Independently governed with representation at CNCF
- Promises based Orchestration (Flow)

Functions as Containers

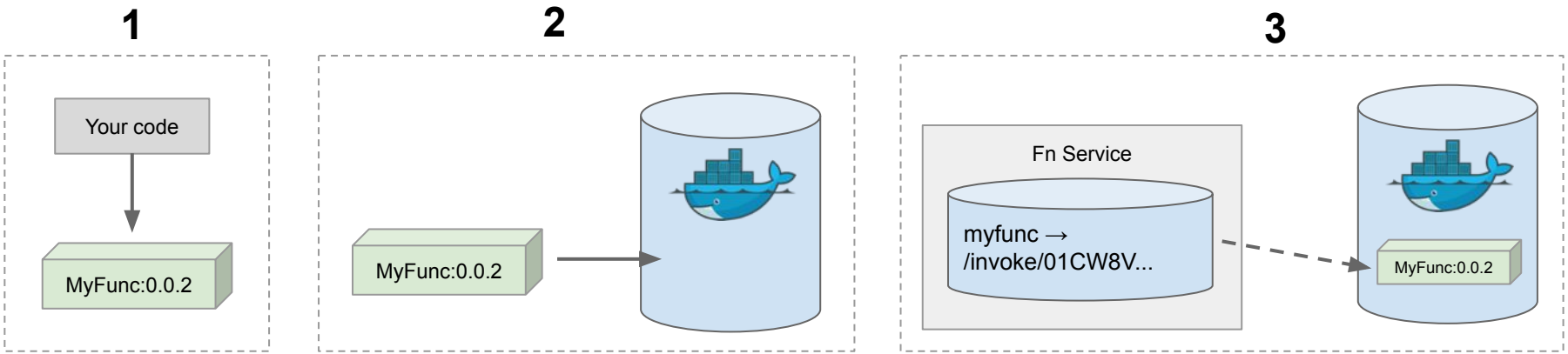
- Function + dependencies
- Single - purpose
- Self - Contained
- Stateless
- Ephemeral
- Run on Demand

An Fn Function

- Small chunk of code wrapped into a container image
- A descriptor (func.yaml)
- Gets input via http-stream and environment
- Produces output to http-stream
- Logs to syslog

Fn deploy details

1. Build container (multi-stage) + bump version
2. Push container to registry
3. Create/update function endpoint(s) (servers lazy load images)



Endpoints

- Fn deploy creates default endpoint:
  - http://localhost:8080/invoke/01CW8VDK3BNG8G00GZJ000008S
  - Used by CLI
- A bit unfriendly
- Only one per function deployment

Triggers

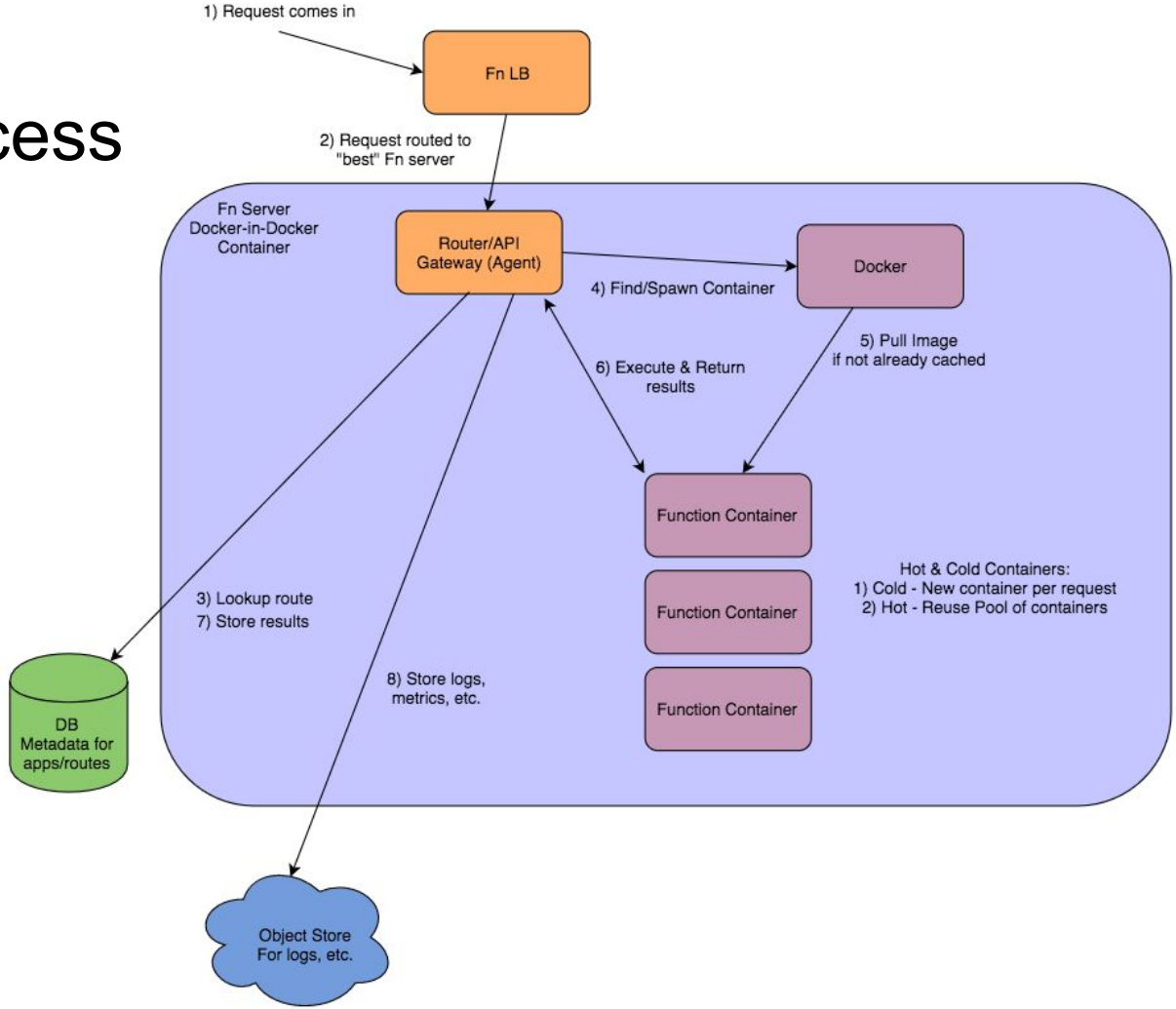
- Meaningful URLs
  - http://localhost:8080/t/kubecon/speaker-trigger
  - http://localhost:8080/t/kubecon/delegate-trigger
- Multiple endpoints
  - e.g internal / external

- Creation:
  - CLI
  - func.yaml

Fn Server

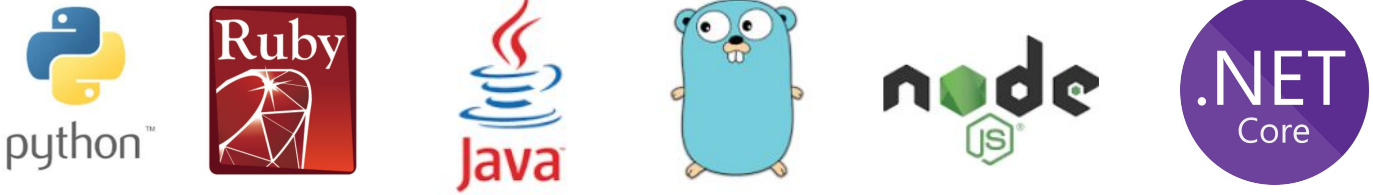
- API Gateway
- Manages **apps** and **functions**
  - app ⇒ group of functions (namespace)
- Handles function invocations
- Runs as a Container
- Hosts function containers
  - (Docker in Docker)

Request Process



Function Development Kits (FDKs)

- Makes it a lot easier to write functions
- Developer includes FDK package / library / gem
- Developer writes function to the FDK's interface
- FDK
  - Provides input data to function
  - Writes output & errors



Anatomy of a Function

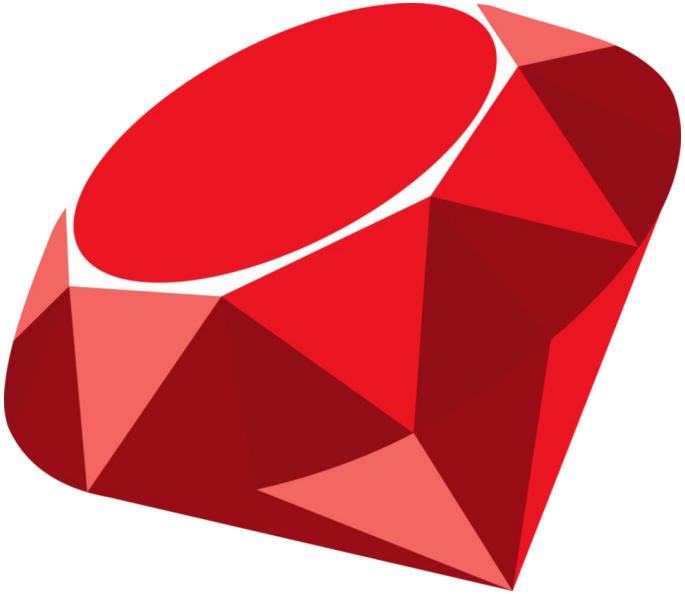
```ruby
require 'fdk'

def myfunction(context:, input:)
  input_value = input.respond_to?(:fetch) ? input.fetch('name') : input
  name = input_value.to_s.strip.empty? ? 'World' : input_value
  { message: "Hello #{name}!" }
end

FDK.handle(target: :myfunction)
~
~
```

Ruby FDK

- Opens a socket in function container
  - (Fn Server connects to socket)
- Parses input from http-stream
- Executes function
  - Input
  - Context
- Sends back output on http-stream



Starting Out

- How it Works!
- Project Structure
- Contribution Guidelines
- Communication Channels
- People
- Processes / practices

What I’ve done

- First issue: fix JSON format
- Support for additional formats
- Coding Standards
- Modern syntax
- Documentation
- Tutorials

Summer Holiday

And then...

- API changed
  - http-stream
  - Triggers

- My day job ⇒ Someone else's rush job
  - "Ruby as Go"
  - It worked! ⇒ bought me time
  - Refactored ⇒ "Ruby as Ruby"

What I want to do

- Improved tests
- Flow support
- TruffleRuby runtime
- Rust FDK

What I’ve learned

- Be Polite
- Learning Curve
  - Always better to ask than assume
- People are incredibly helpful
  - Try to be as helpful yourself
- Don’t be possessive
- Focus on what you can bring to the project
- Respect other projects
  - Fellow travellers


MONDAINE
SBB CFF FFS
OFFICIAL RAILWAYS CLOCK DESIGN

What I’ve learned

- Manage your time
  - Family
  - Work
  - You
- Be responsive, but honest
- Set realistic expectations
  - For the team
  - For yourself
- Align with day job (if possible)

Getting others involved

- Meetups
- Conferences
- Slack as the “gateway drug”
  - User
  - “Pusher”
    - Git
    - Promoter

Why get involved?

- Fun
- Sense of Achievement
- Learning
- Moral Debt

Get Involved

- Learn more: fnproject.io

- Get in touch
  - Slack: fnproject.slack.com
  - Twitter: @fnproject

- Contribute: github.com/fnproject

Take Aways #1

- Serverless is an Abstraction
  - Productivity
  - Agility
  - Scalability
  - Economics

- Fn ⇒ Open Source Serverless Platform
  - Docker based
  - Portable
  - Language agnostic
  - Multiple FDKs

Take Aways #2

- Open Source matters
- Open Source Serverless matters
- If you care…
  - Act like it
  - Contribute
  - Help others

Thank You

@ewanslater