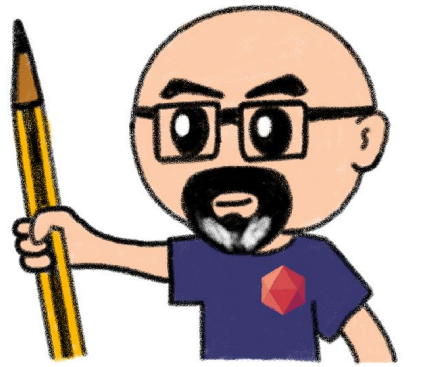
Let's dive into Kubernetes operator creation

Horacio Gonzalez

2024-04-06

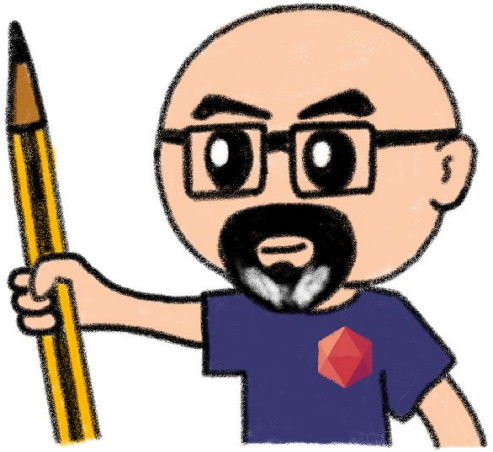
clever cloud



@LostInBrittany

Who are we?

Introducing myself and introducing Clever Cloud



Horacio Gonzalez

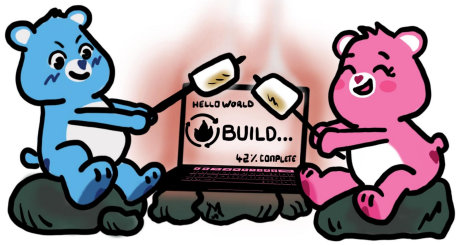
@LostInBrittany

Spaniard Lost in Brittany

Head of DevRel

clever cloud

Finist Devs



Clever Cloud

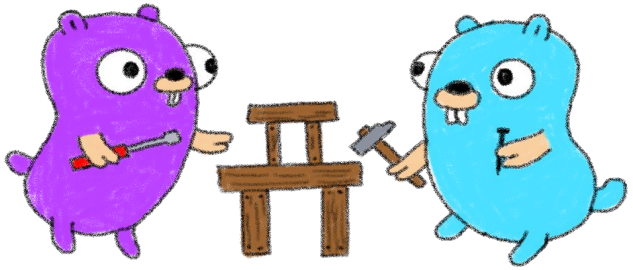
clever cloud

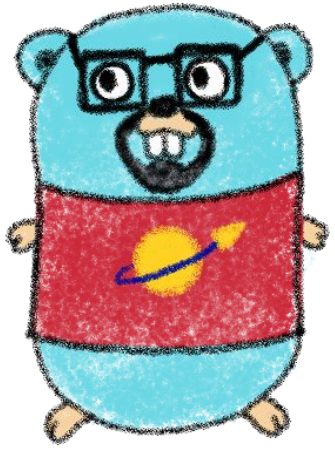
From Code to Product

Our mission: give more **speed** to your **teams**
and better **quality** to your **projects**

Warning

Gophers, gophers everywhere!

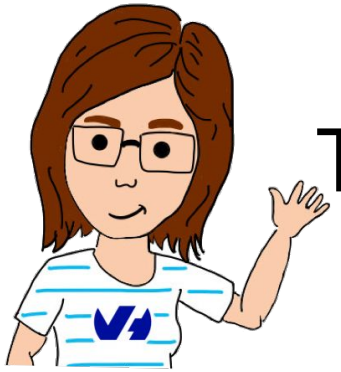




Last year in Config Management Camp

I proposed a sequel for this year

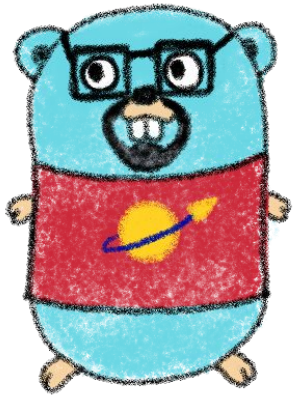
Let's dive into Kubernetes operator creation

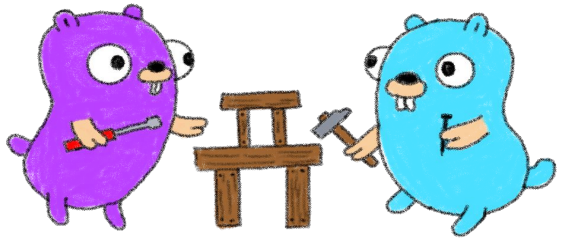


This time Aurélie can't do the talk with me 😢

So I must do it alone… Wish me luck!

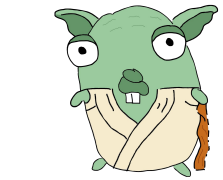
And why Gophers?

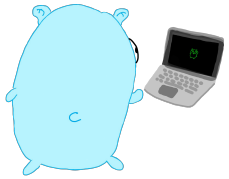




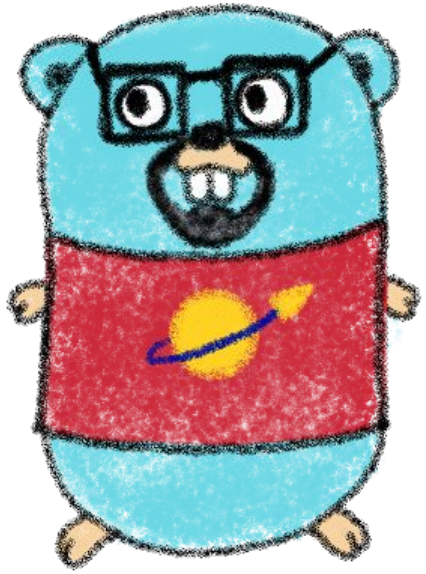
Because we love Gophers, of course!
And because Golang and Kubernetes are so linked…

Credit where it is due

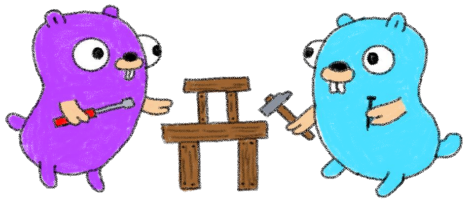


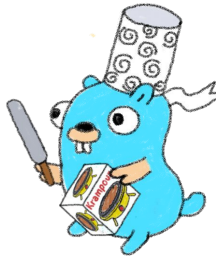


@AurelieVache



@LostInBrittany

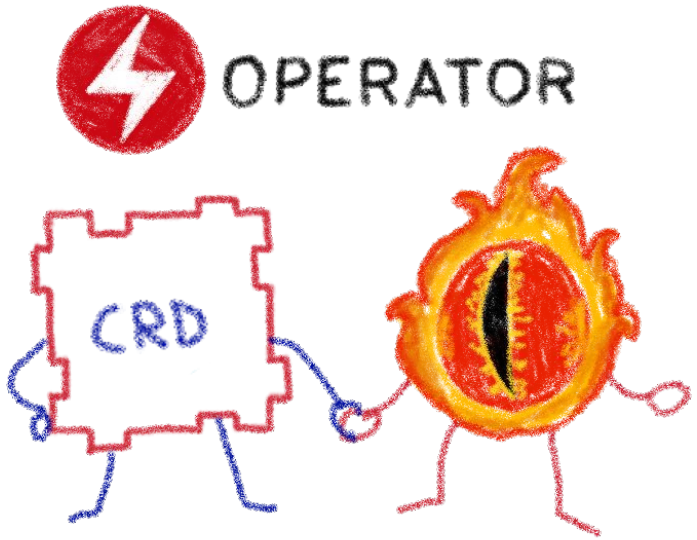




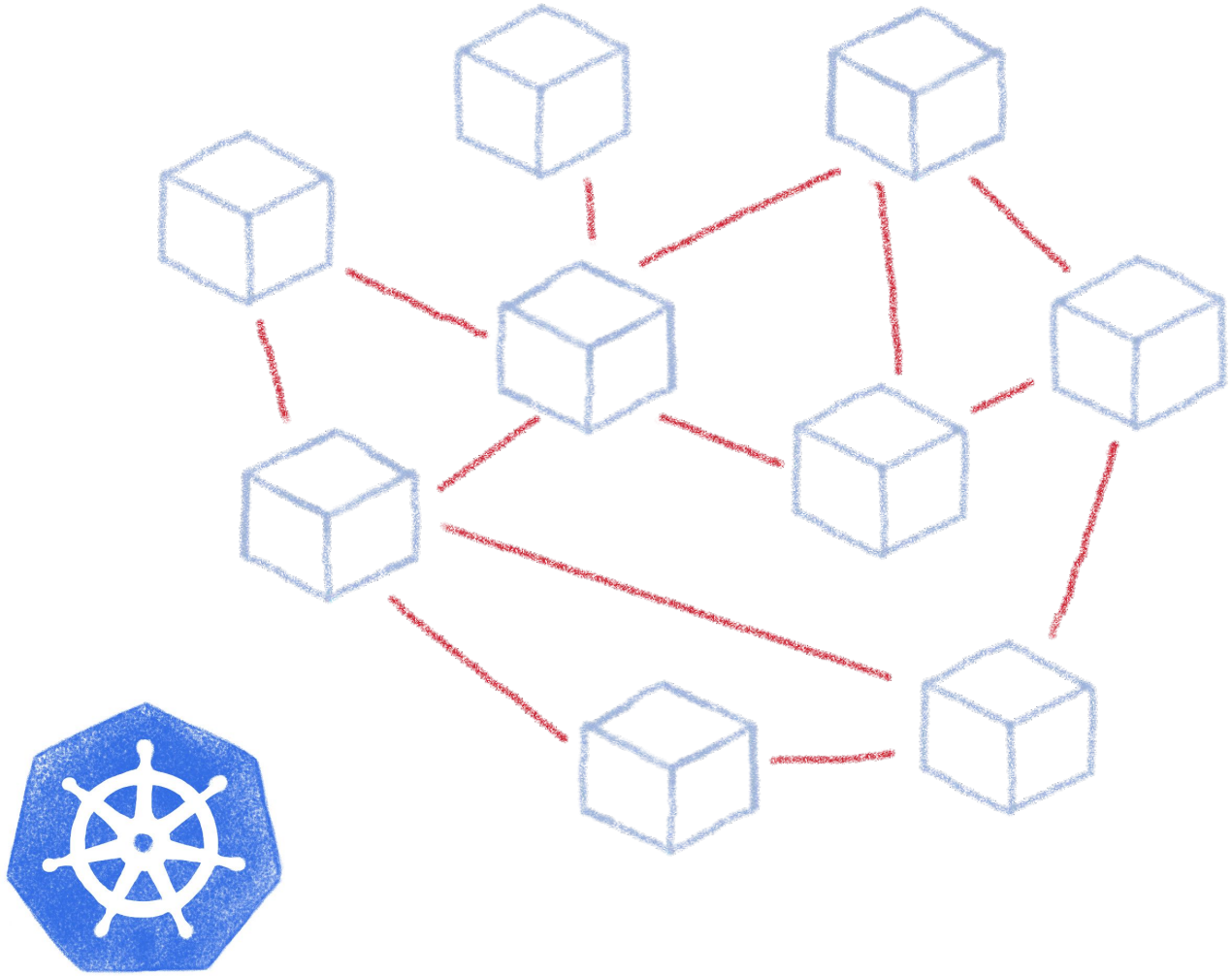
All the gophers you will see are drawn by Aurélie and Horacio, and are based on the Go mascot designed by Renee French which is licensed under CC BY 3.0.

Kubernetes operators

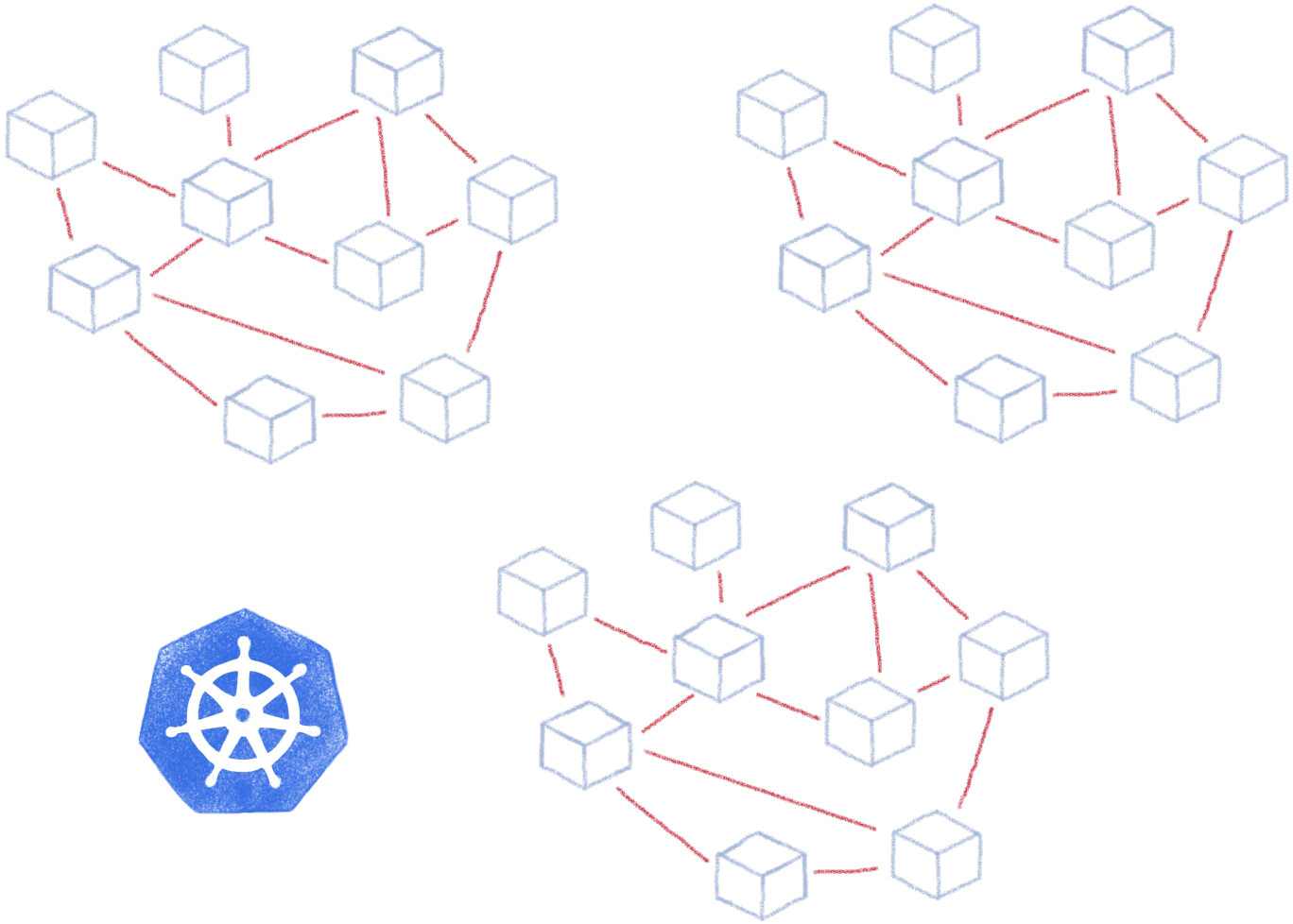
Helping to tame the complexity of K8s Ops



Taming microservices with Kubernetes



What about complex deployments



Ingress

Services

Deployments

Pods

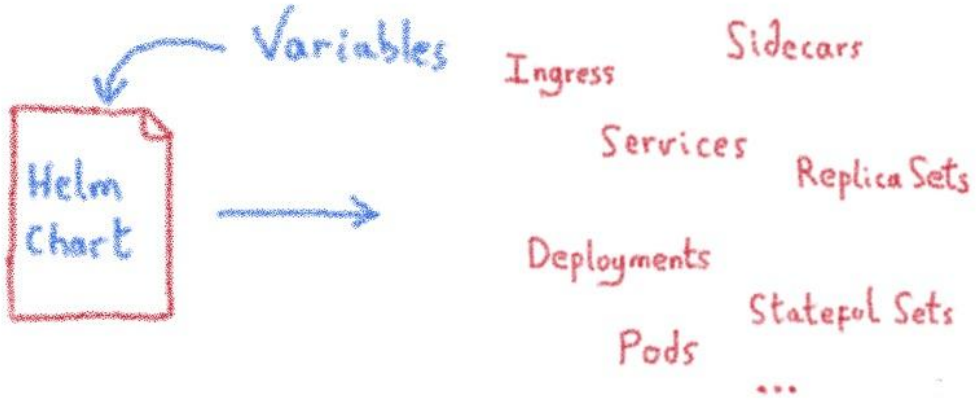
Sidecars

Replica Sets

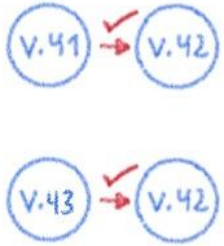
Stateful Sets

Tools like Helm helps with complexity

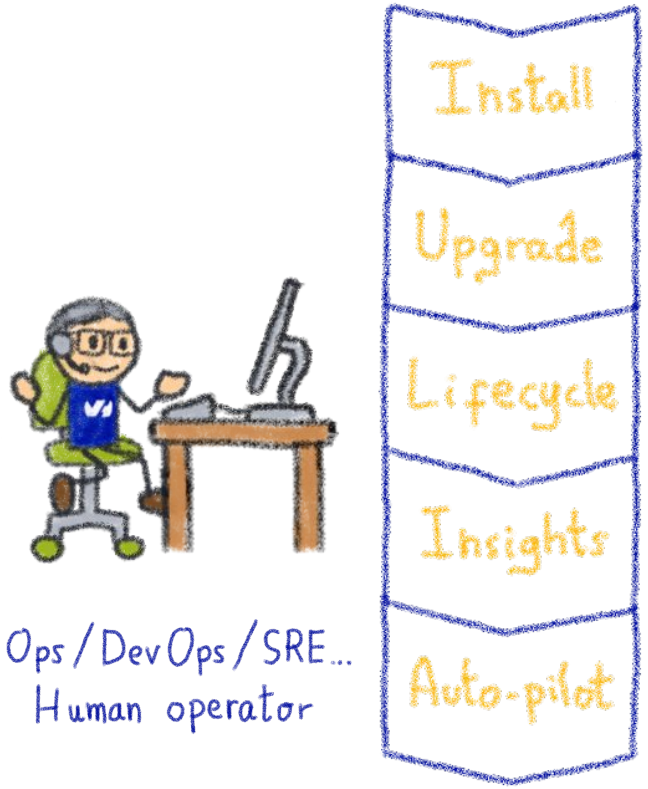
A package manager for Kubernetes



- Manage complexity
- Simple sharing
- Easy upgrades
- Easy rollbacks

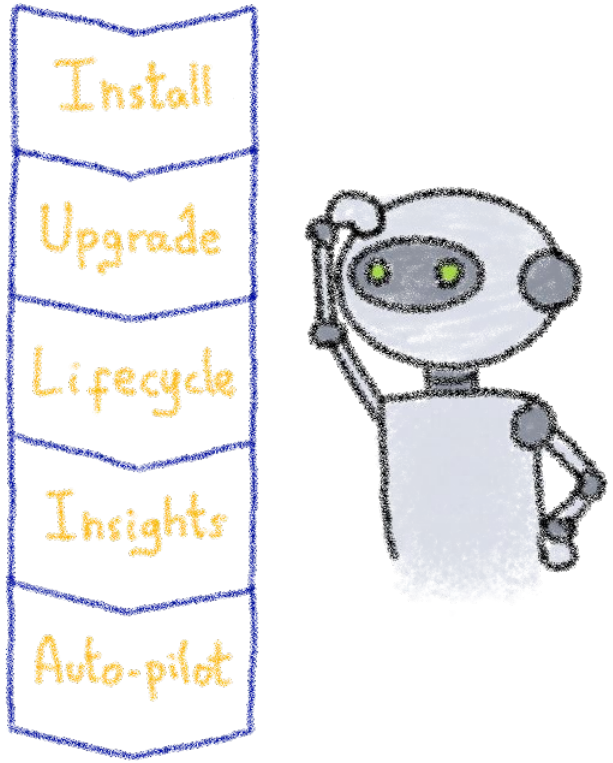


Helm Charts are configuration



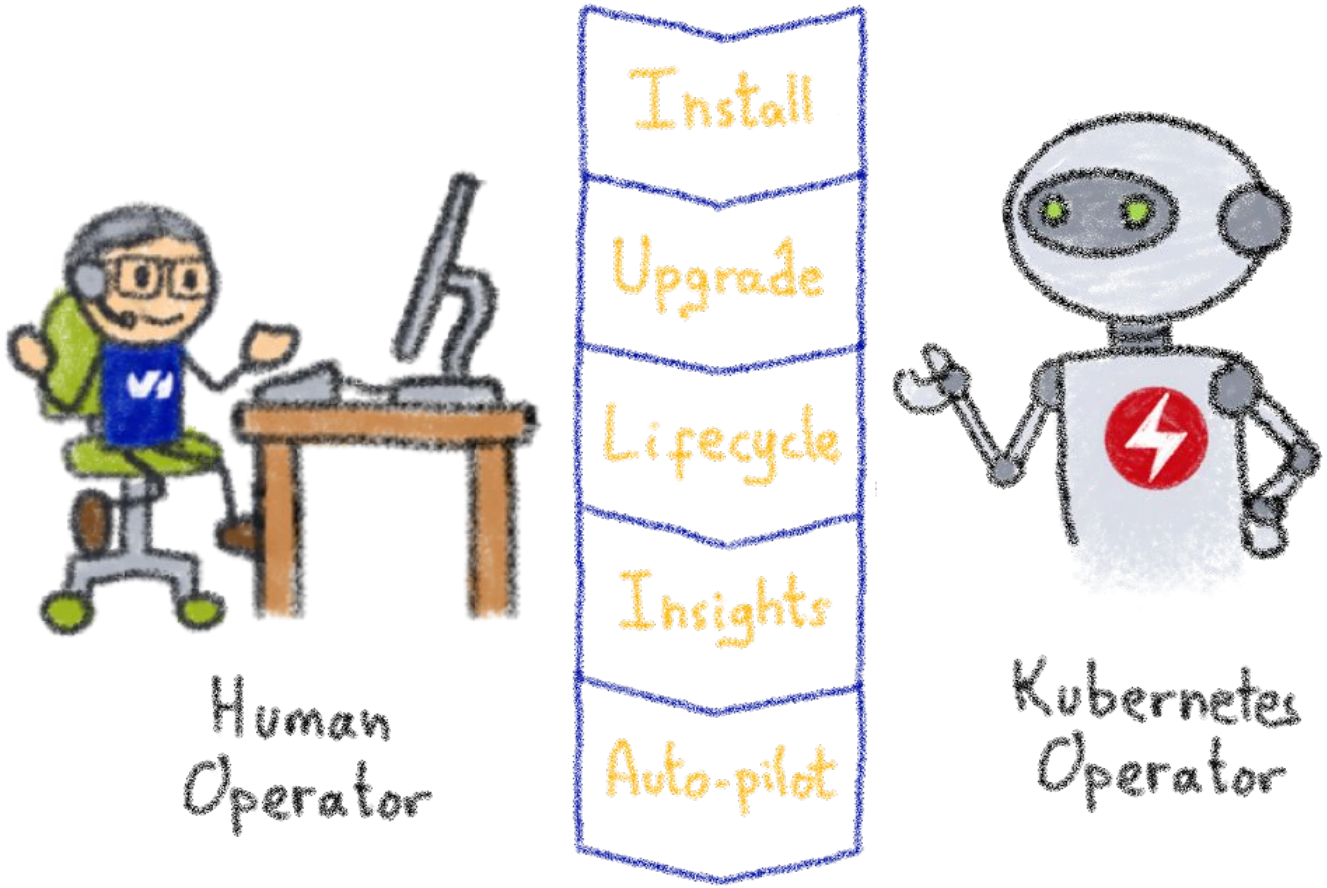
Operating is more than installs & upgrades

Kubernetes is about automation



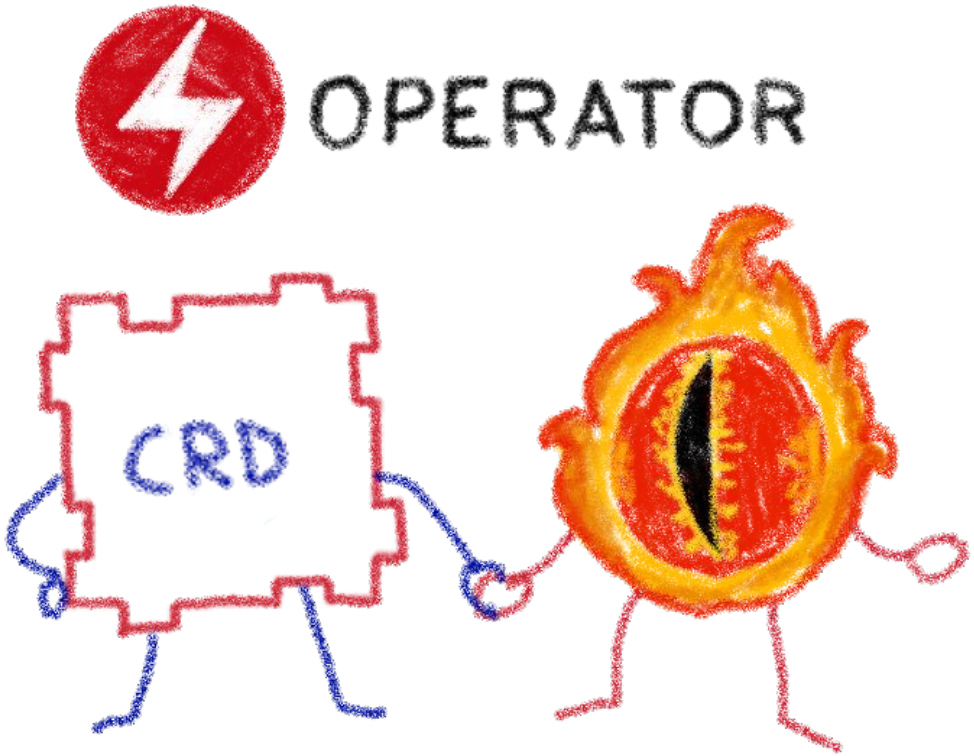
How about automating human operators?

Kubernetes Operators



A Kubernetes version of the human operator

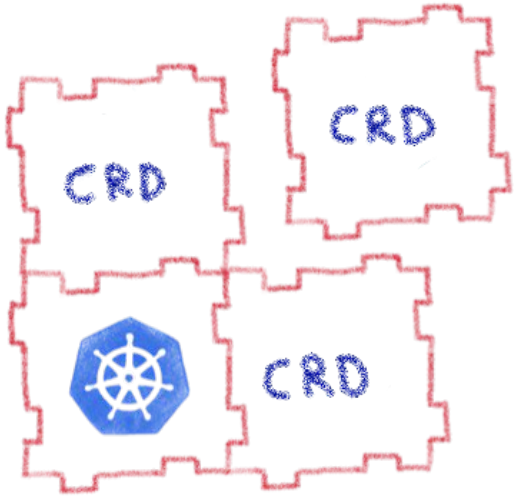
Building operators



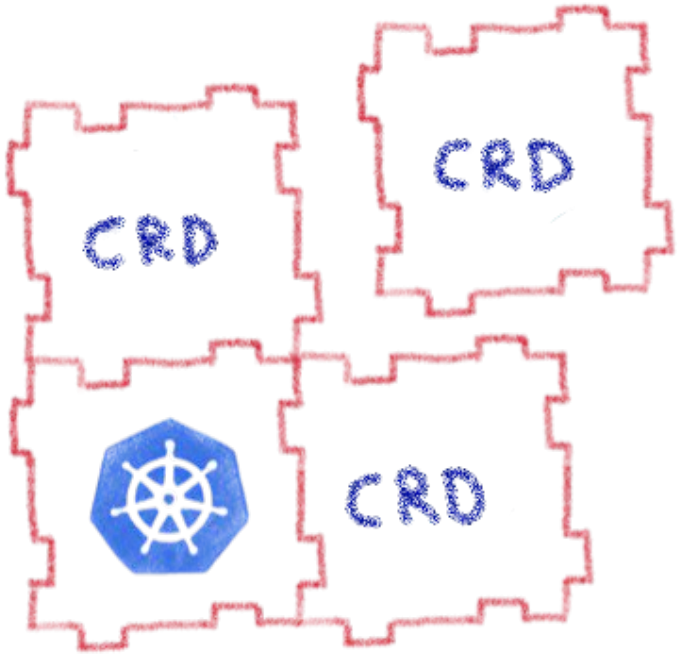
Basic K8s elements: Custom Resources & Controllers

Custom Resource Definitions

Extending Kubernetes API

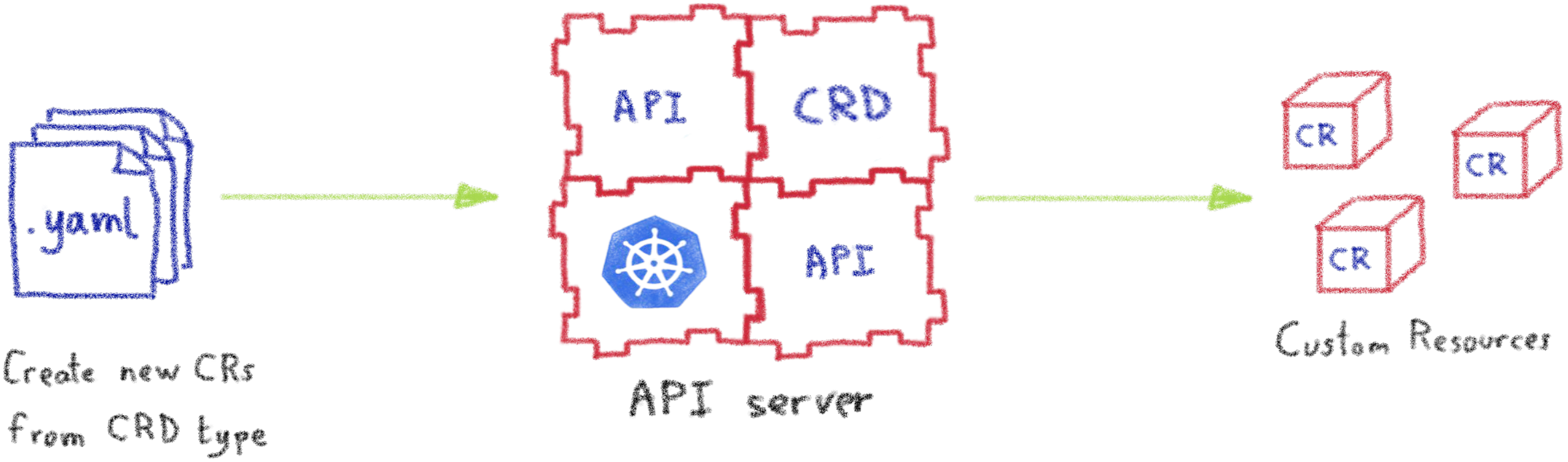


Extending Kubernetes API



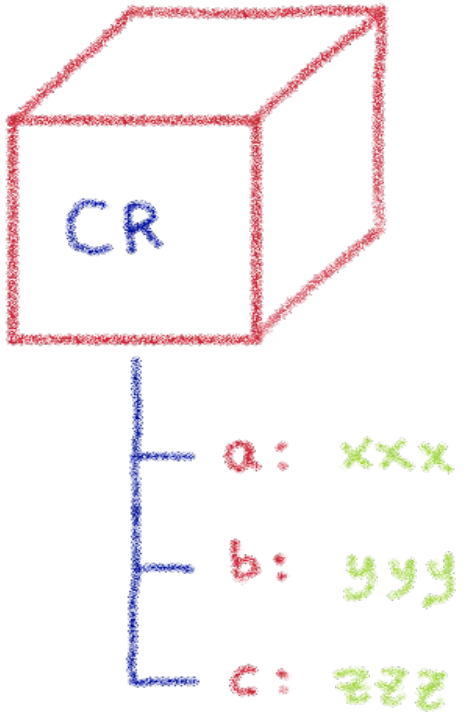
By defining new types of resources,
internal or external to the cluster

With a CRD you can create CR in the cluster



They are the blueprints of the Custom Resources

Custom Resources are simply data



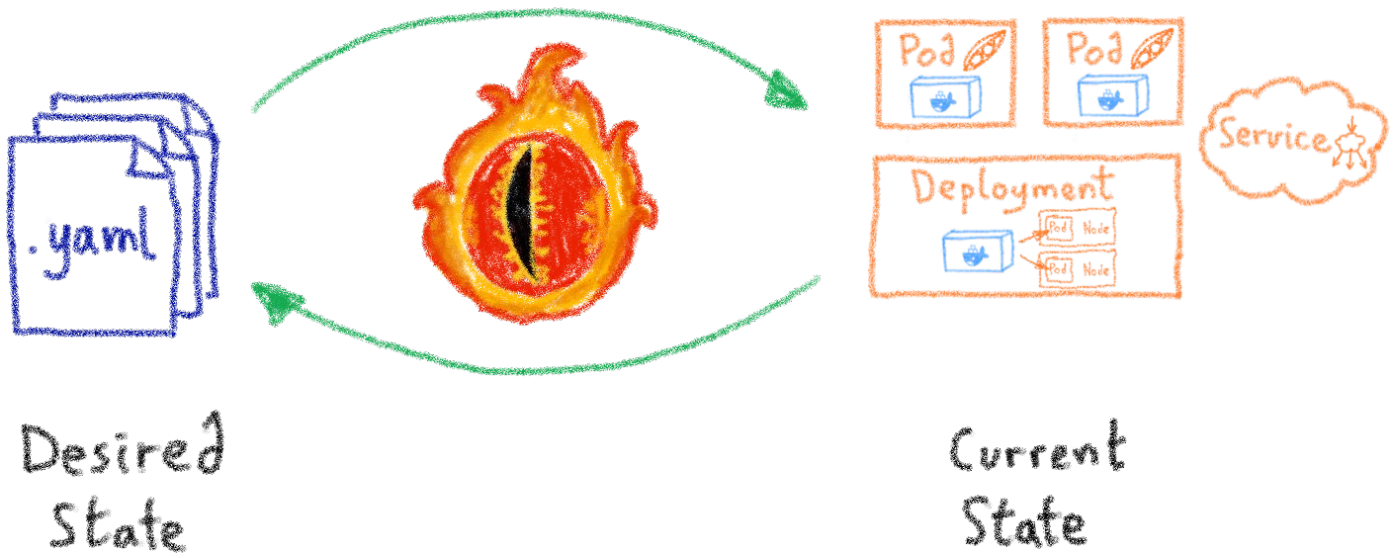
Only data, properties, no logic

All the logic must be in the Controller

Kubernetes Controllers

Keeping an eye on the resources

A reconcile loop

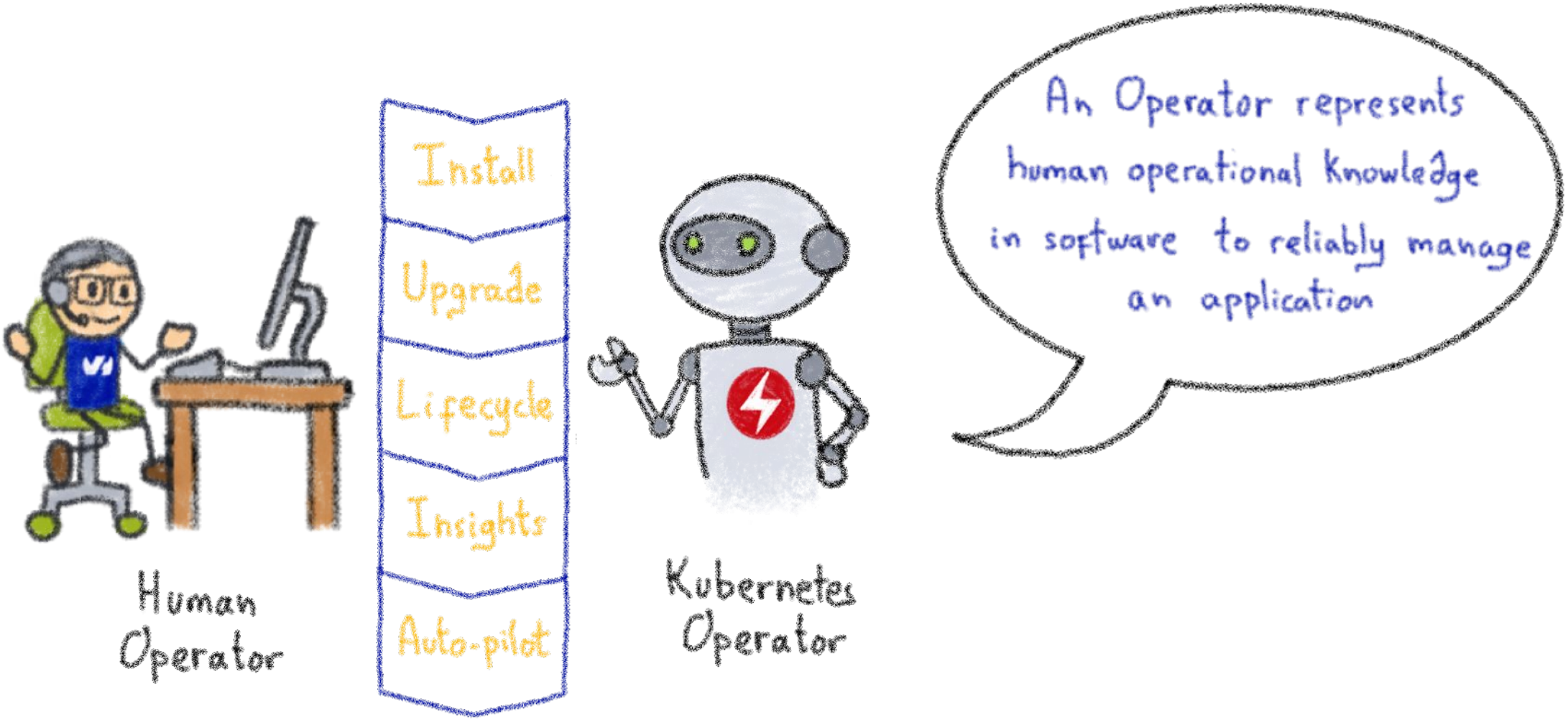


Controllers watch the state of the cluster,
and make or request changes where needed

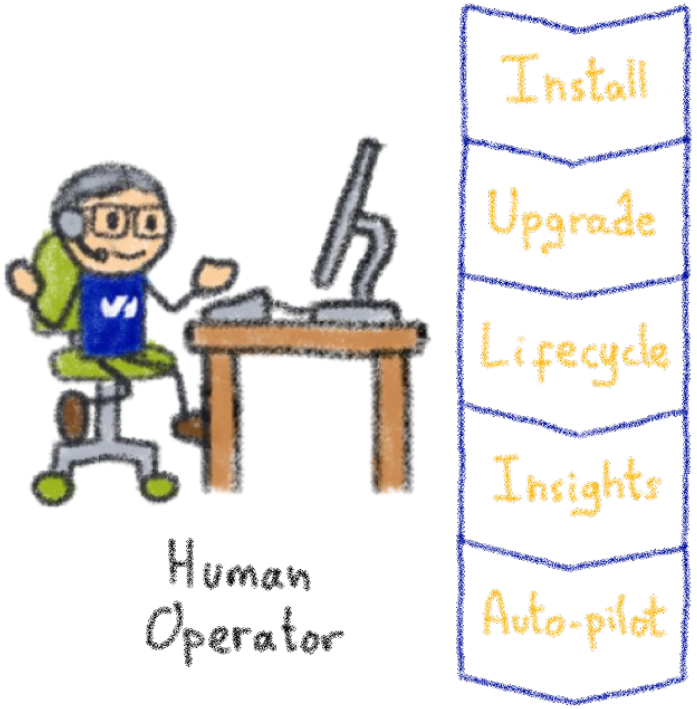
Kubernetes Operator

Automating operations

What's a Kubernetes Operator?



Example: databases



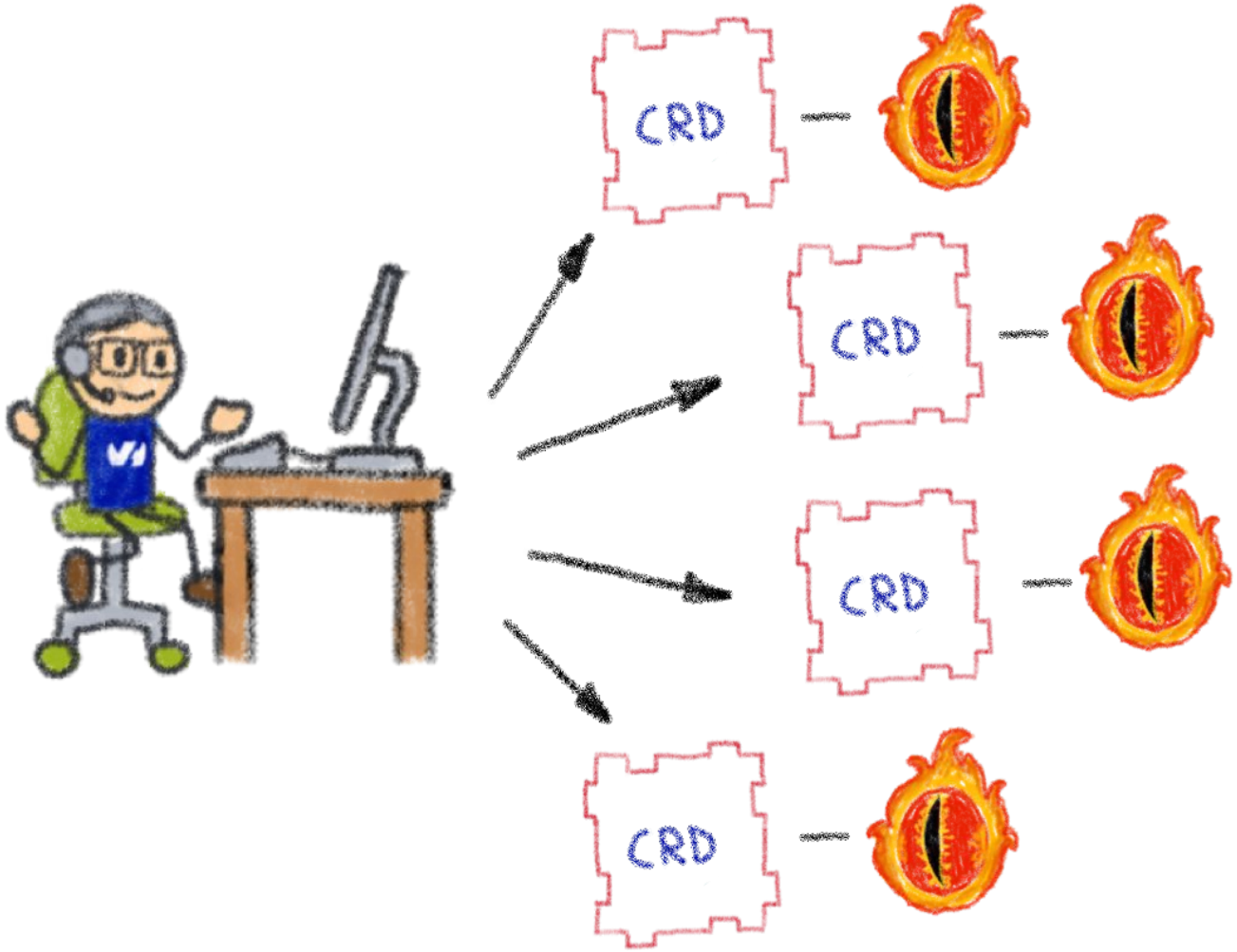
Install
Upgrade
Lifecycle
Insights
Auto-pilot

PostgreSQL

redis

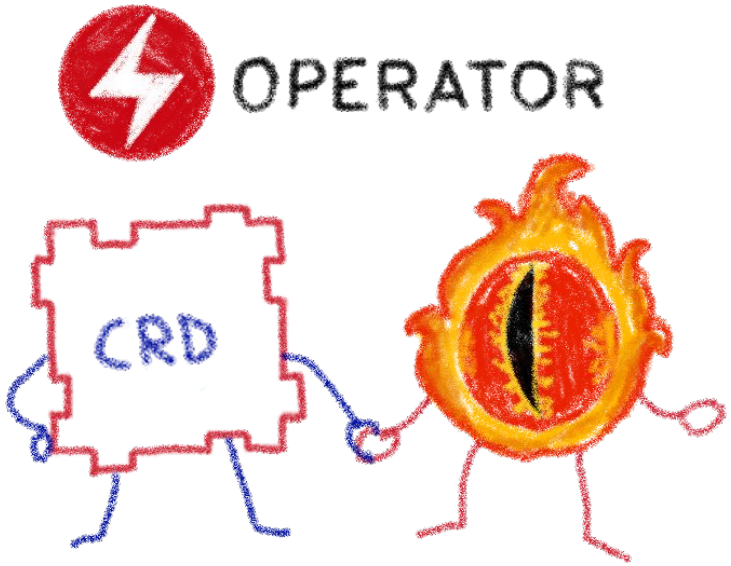
Things like adding an instance to a pool,
doing a backup, sharding...

Knowledge encoded in CRDs and Controllers



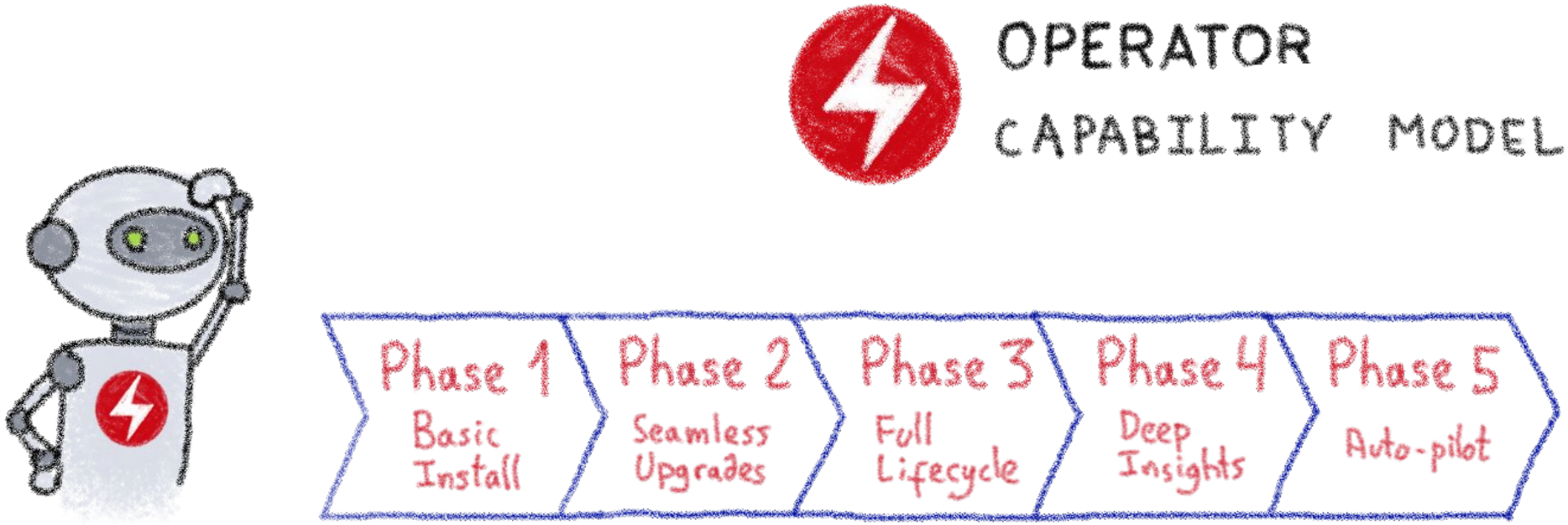
Encapsulating business logic in CRDs & Controllers

Custom Controllers for Custom Resources



Operators implement and manage Custom Resources using custom reconciliation logic

Operator Capability Model



Gauging the operator maturity

A real, open-source example

The Clever Operator

clever cloud

Available on Operator Hub & GitHub

OperatorHub.io

Search OperatorHub...

Contribute

clever-operator

A kubernetes operator that expose clever cloud's resources through custom resource definition

Home > clever-operator

clever-operator

A kubernetes operator that expose clever cloud's resources through custom resource definition

Install

Custom Resource Definitions

PostgreSql

Clever Cloud's managed postgresql database

View YAML Example

Redis

Clever Cloud's managed redis database

View YAML Example

MySql

Clever Cloud's managed mysql database

View YAML Example

MongoDb

Pulsar

ConfigProvider

CHANNEL
alpha

VERSION
0.5.5 (Current)

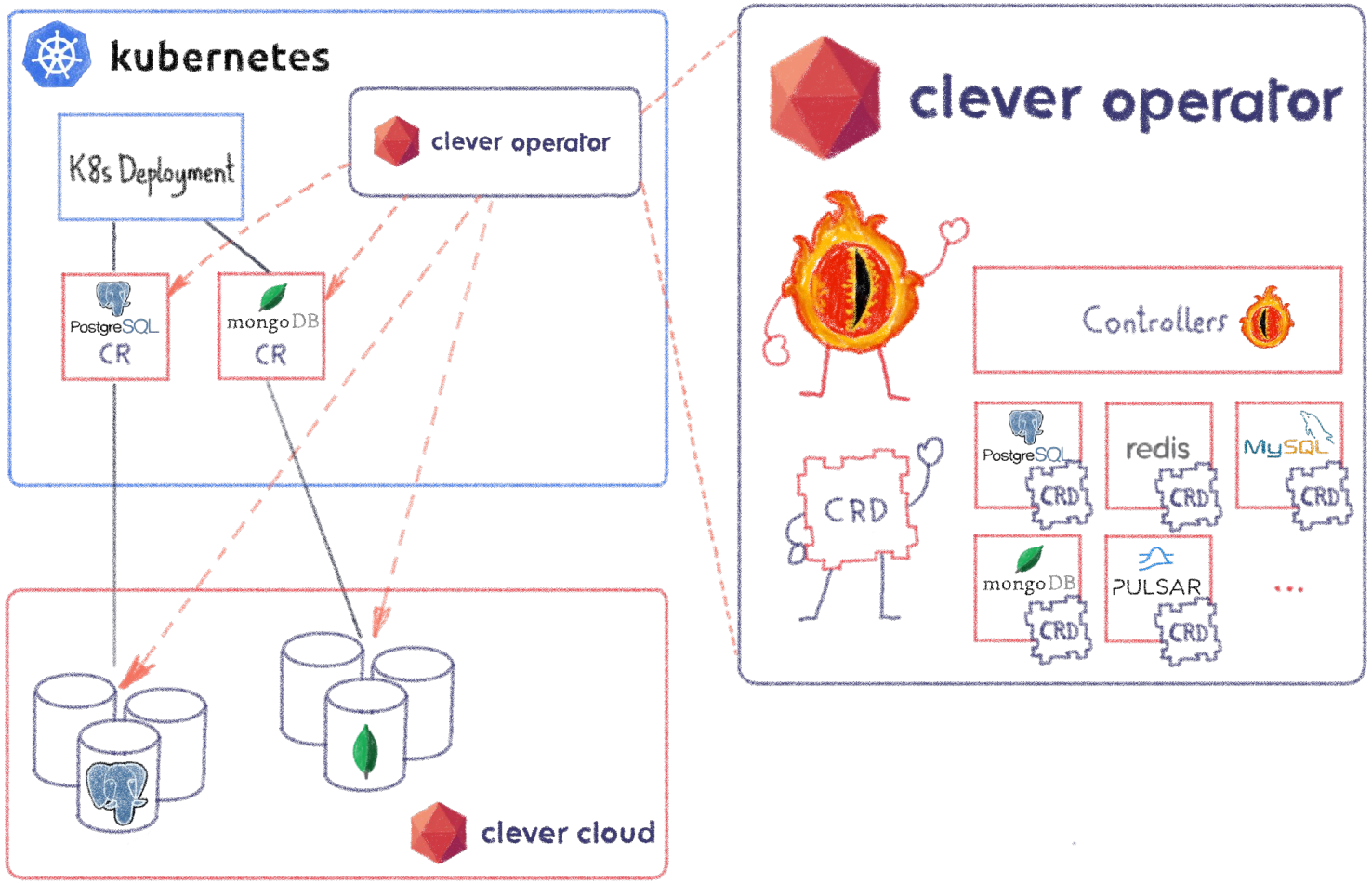
MIN K8S VERSION
v1.24.0

CAPABILITY LEVEL
- Basic Install
- Seamless Upgrades
- Full Lifecycle
- Deep Insights

https://operatorhub.io/operator/clever-operator

https://github.com/CleverCloud/clever-operator

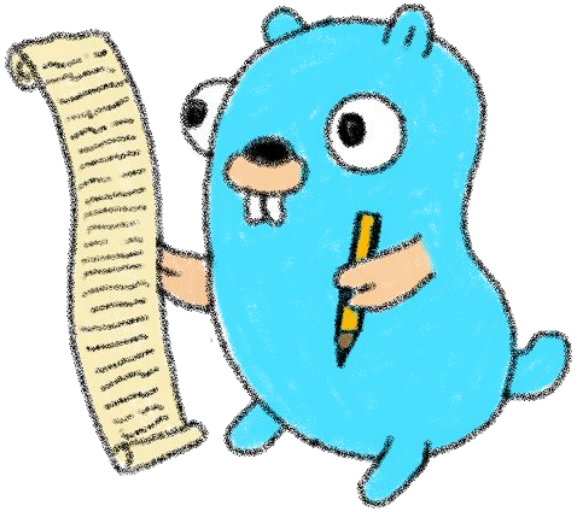
Exposing Clever Cloud resources as CRD



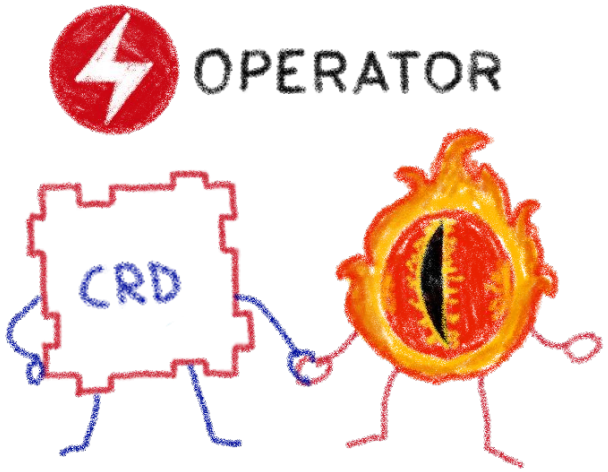
Allowing your apps to use our DBs as if they were in K8s

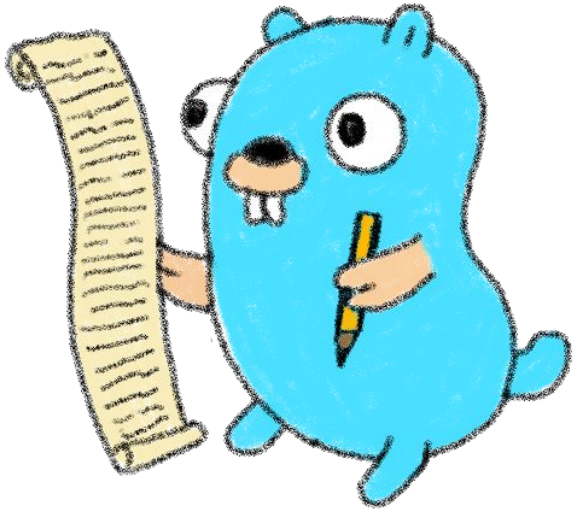
How can we write Operators?

Which language? Any framework?



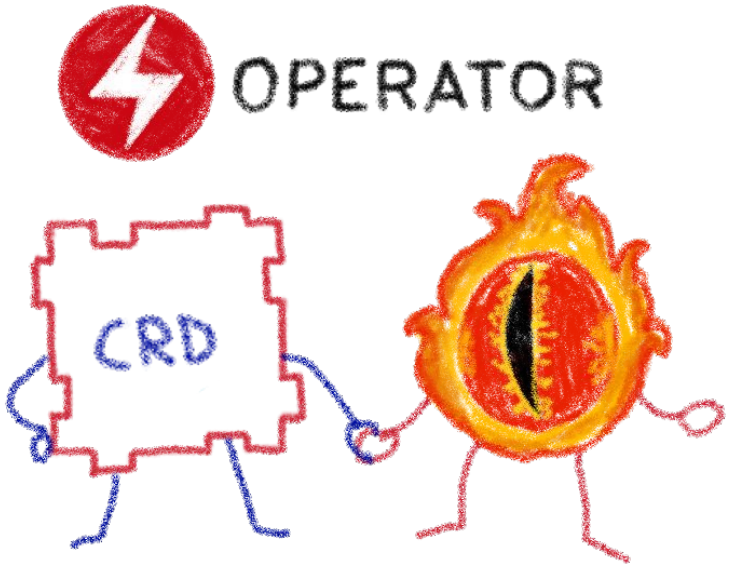
They are simply pods and manifests





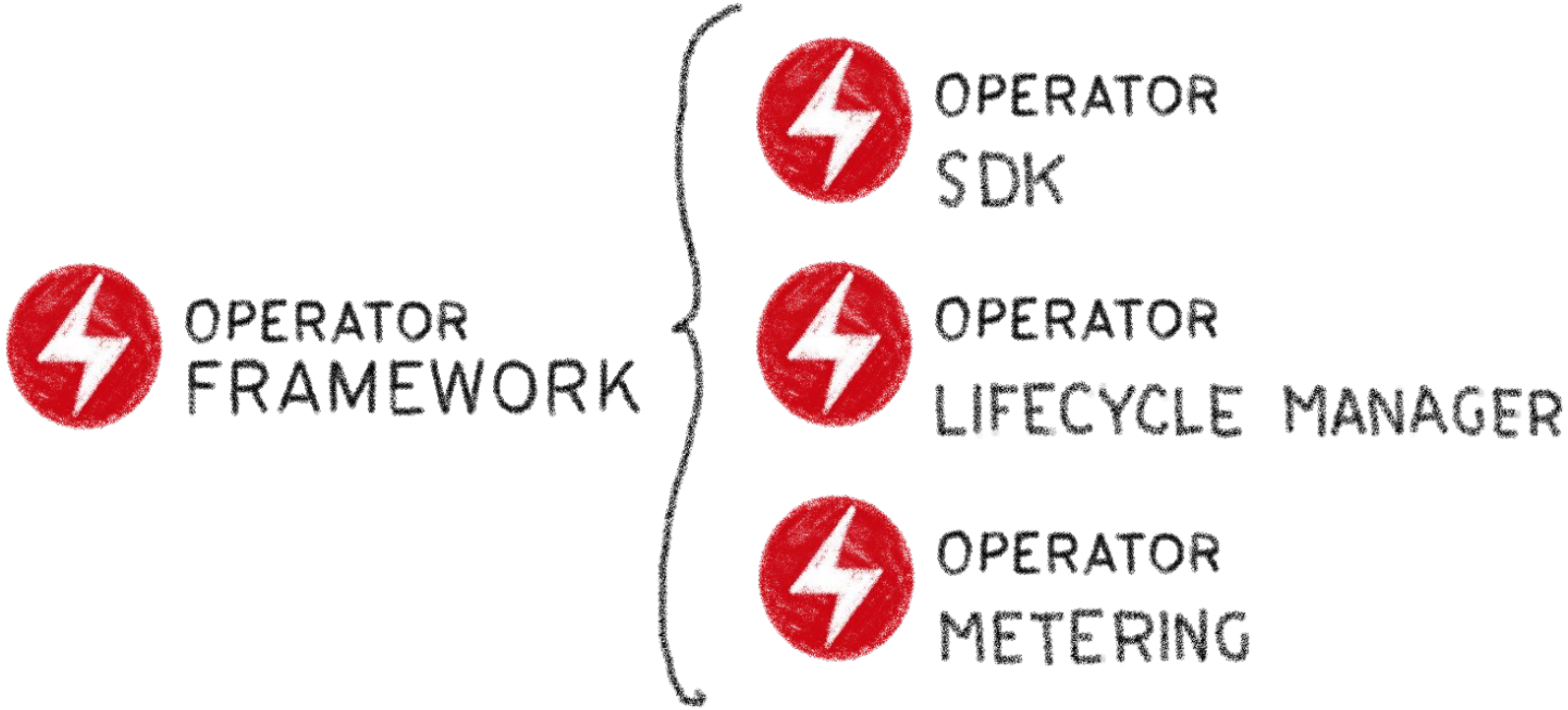
You can simply call Kubernetes APIs
or use a compatible client

How to write an Operator



1- Create a new project
2- Write the CRDs to define new resource APIs
3- Specify resources to watch
4- Define the reconciliation logic in the Controllers
5- Build the Operator

The Operator Framework



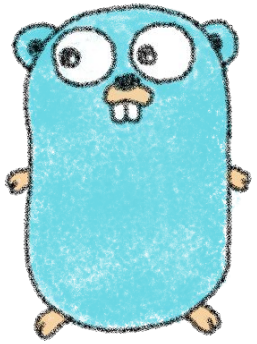
Open source framework to accelerate the development of an Operator

Operator SDK

OPERATOR
SDK

BUILD
TEST
ITERATE

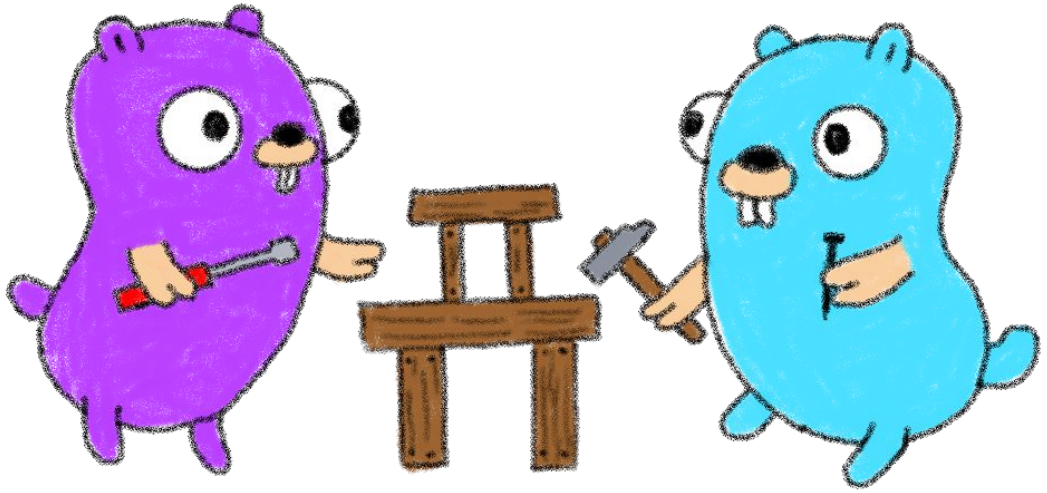
ANSIBLE



Three different ways to build an Operator

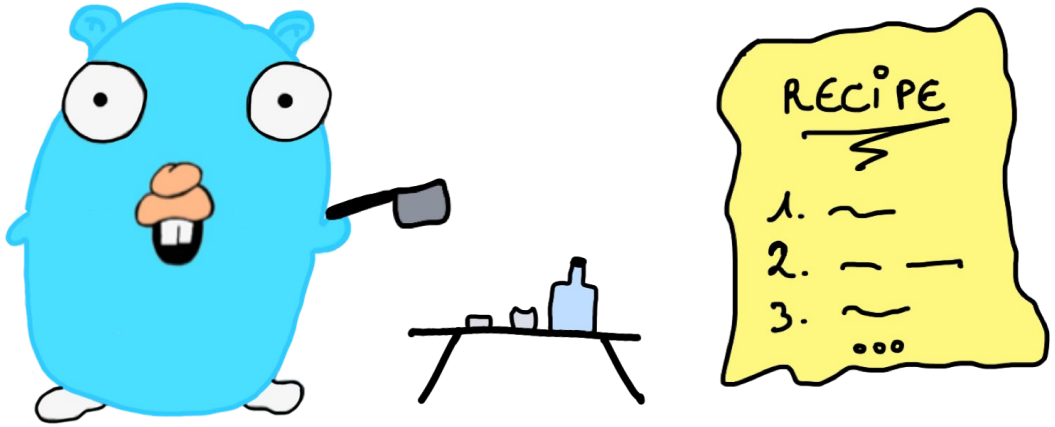
Our objective

Why? Because we can!



What do we want?

- In a simple and easy Kubernetes operator
- Handle cute Gophers
- In Javascript, because it's very expressive and easy to understand... and I like it 😁



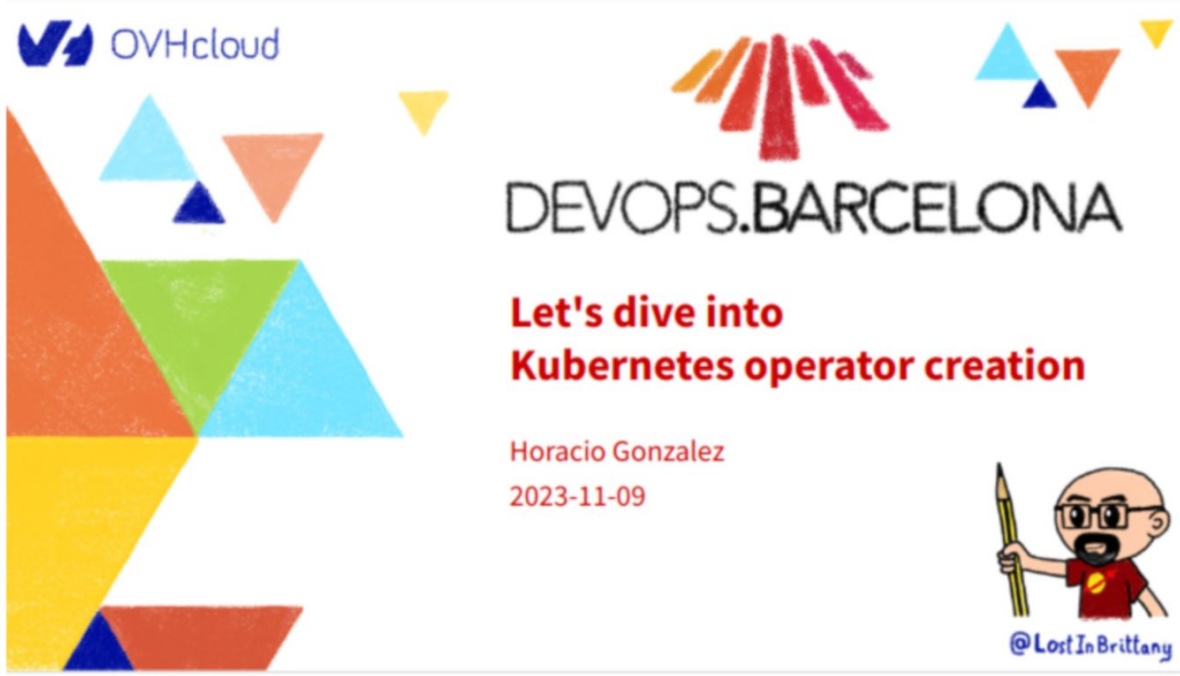
All the code is available

README.md

Let's dive into Kubernetes operator creation

This repository stores all the code for my talk *Let's dive into Kubernetes operator creation*, that I have given at:

- 2023-11-09 - DevOps Barcelona - Slides



Packages

No packages published
Publish your first package

Languages

TypeScript 80.9% JavaScript 15.8%
HTML 2.1% Other 1.2%

Suggested Workflows
Based on your tech stack

Datadog Synthetics Configure
Run Datadog Synthetic tests within your GitHub Actions workflow

Gulp Configure
Build a NodeJS project with npm and gulp.

Deno Configure
Test your Deno project

More workflows Dismiss suggestions

https://github.com/LostInBrittany/lets-dive-into-kubernetes-operator-creation

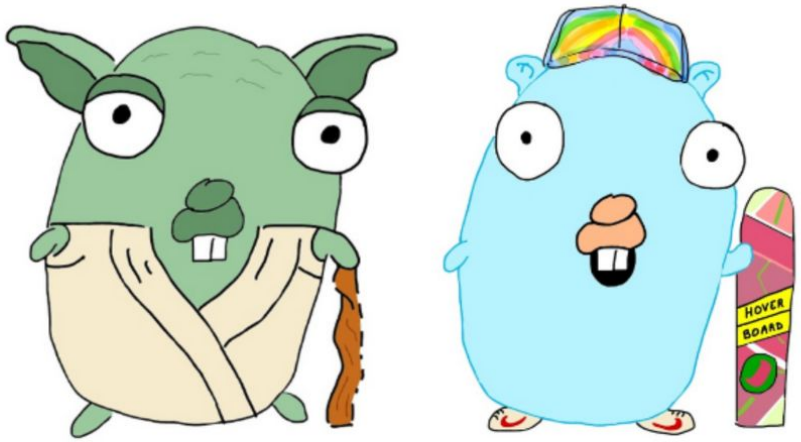
Aurélie's Gopher repository

github.com/scraly/gophers

README.md

Gopher artwork (Golang mascot)

This repository contains gopher artwork, based on the original artwork by Renee French.



Why

As I love Gophers, I started to draw more and more Gophers and I decided to share them in this GitHub repository.

https://github.com/scraly/gophers

random-gopher container

hub.docker.com/r/lostinbrittany/random-gopher/tags

docker hub | lostinbrittany/random-gopher | Explore | Pricing | Sign In | Sign up

Explore / lostinbrittany/random-gopher

lostinbrittany/random-gopher ☆

Pulls 16

By lostinbrittany • Updated 3 hours ago

Image

Overview | Tags

Sort by Newest | Filter Tags

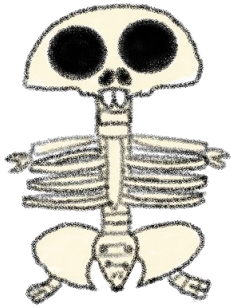
TAG			
0.0.4 Last pushed 3 hours ago by lostinbrittany		docker pull lostinbrittany/rando…	
DIGEST	OS/ARCH		COMPRESSED SIZE
098b143a73f5	linux/amd64		438.51 MB

TAG			
0.0.3 Last pushed 7 days ago by lostinbrittany		docker pull lostinbrittany/rando…	
DIGEST	OS/ARCH		COMPRESSED SIZE
6ee889948db8	linux/amd64		438.51 MB

https://hub.docker.com/r/lostinbrittany/random-gopher

random-gopher container

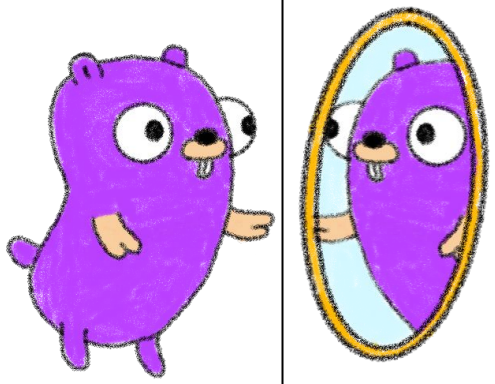
```
import express from 'express';

import { readdir } from 'node:fs/promises';
import path  from 'node:path';

let app = express();
let chosenGopher;

async function initFiles() {
   try {
       const files = await readdir('gophers');
       const gophers = files.filter(
           (item) => item.endsWith('png') || item.endsWith('jpg')
       );
       const randomIndex = Math.floor((Math.random()*gophers.length));
       chosenGopher = gophers[randomIndex];
       console.log(chosenGopher);
     } catch (err) {
       console.error(err);
     }
}
```



At startup it chooses and exposes a random gopher

random-gopher-deployment

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
 name: random-gopher
spec:
 selector:
   matchLabels:
     run: random-gopher
 replicas: 10
 template:
   metadata:
     labels:
       run: random-gopher
   spec:
     containers:
     - name: random-gopher
       image: lostinbrittany/random-gopher:0.0.4
       ports:
       - containerPort: 8080
```



Deploying lots of random-gophers in the cluster

Applying it to the cluster

Deploying `random-gopher-deployment`

Deploying the manifest

```
kubectl apply -f manifests random-gopher-deployment.yaml
```

Getting pods' address

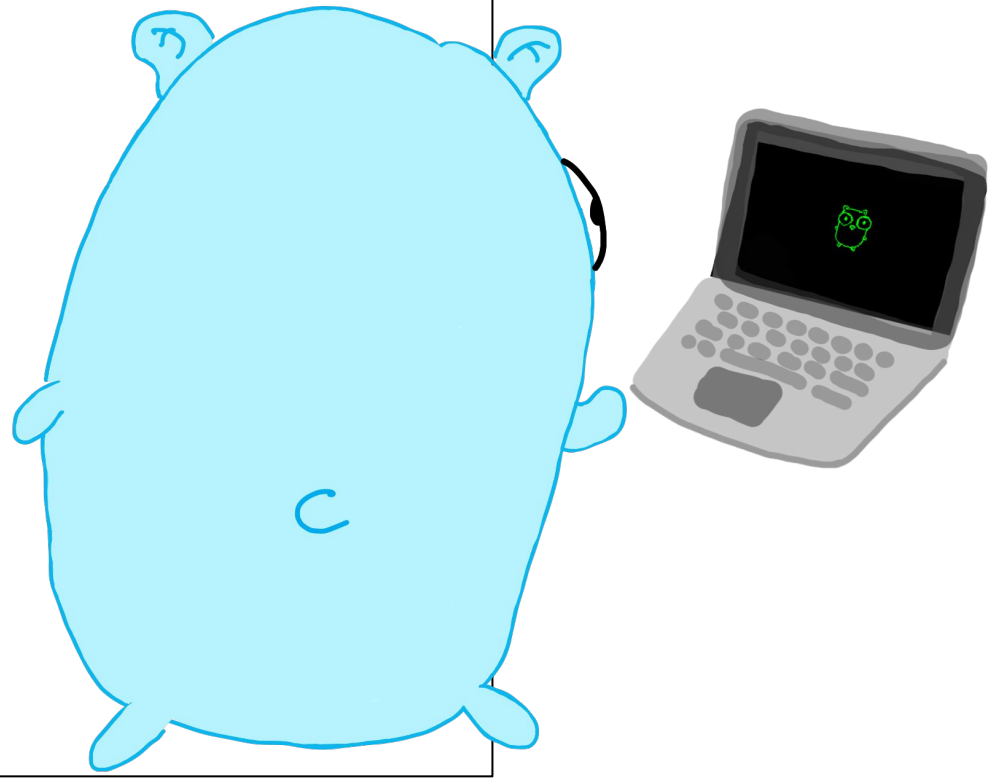
```
kubectl get pods -o wide
```

Create a busybox

```
kubectl run -i --tty --rm debug --image=busybox --restart=Never -- sh
```

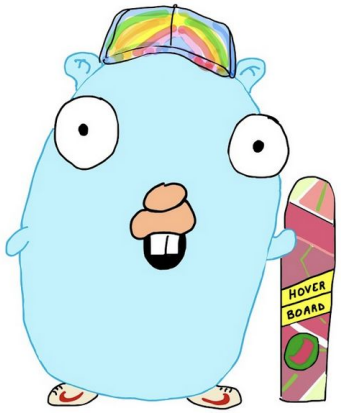
Asking for a Gopher name

```
wget -qO - [pod_ip]:8080/gopher/name
```



Let's switch to the terminal…

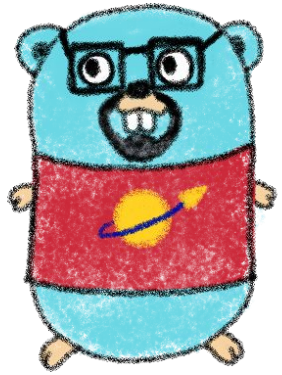
We also have an API for Gophers



gophers-api

This simple API handle a list of Gophers. It alllows to:

- list the existing Gophers
- display the information about a Gopher
- create a new Gopher
- delete a Gopher
- update the path and the URL of a Gopher



https://github.com/LostInBrittany/lets-dive-into-kubernetes-operator-creation/tree/main/gopher-api-and-ui

We also have an API for Gophers

Swagger
Supported by SMARTBEAR

Explore

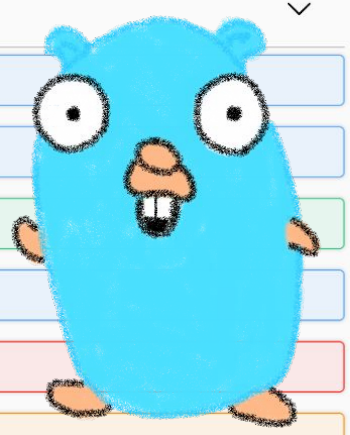
gophers-api 0.1.0

HTTP server that handle cute Gophers.

Schemes
HTTP

default

GET	/healthz
GET	/gophers
POST	/gopher Add a new Gopher
GET	/gopher
DELETE	/gopher
PUT	/gopher

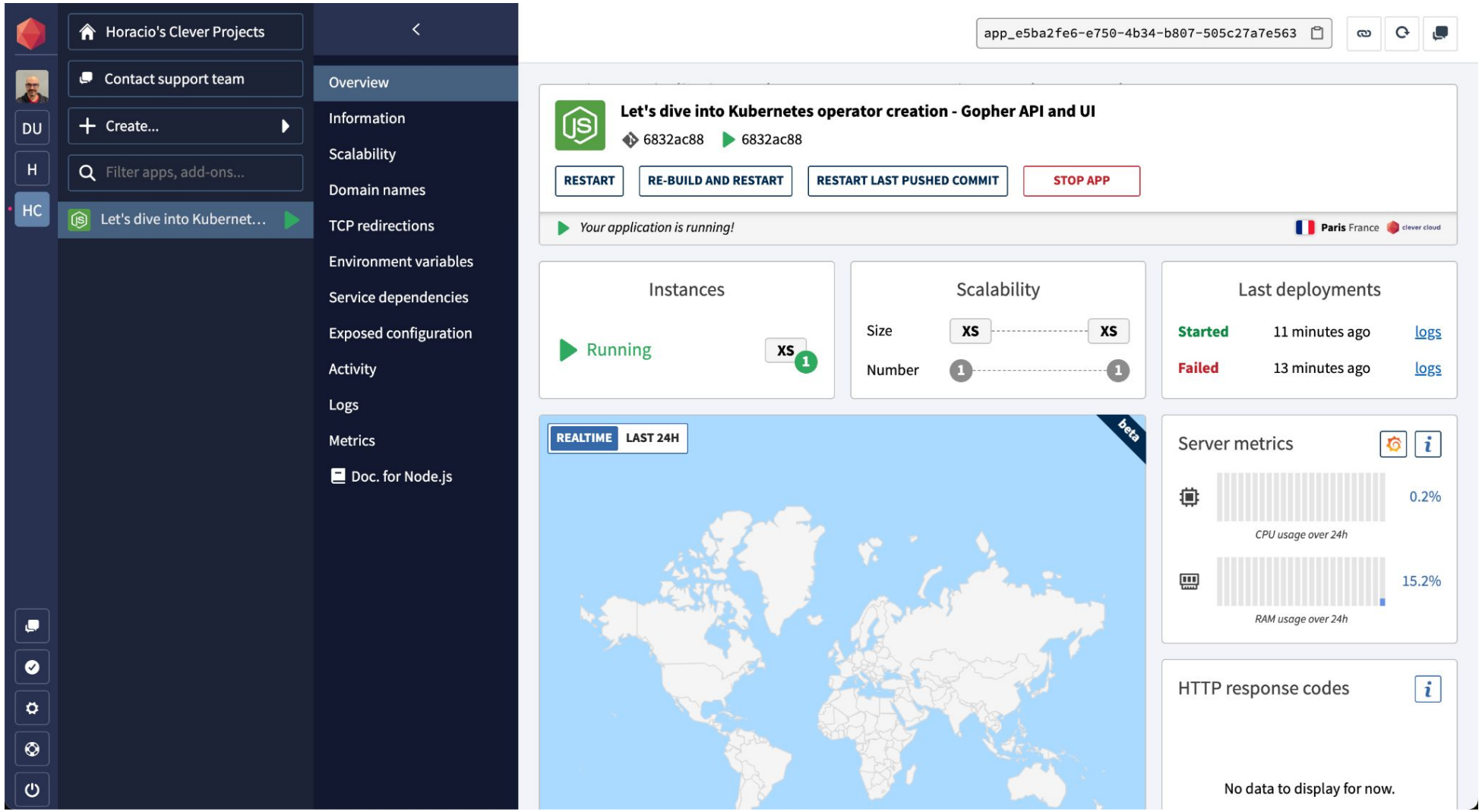


Create
Read
Update
Delete

And an UI to see the Gophers in the API

https://github.com/LostInBrittany/gophers-api-watcher

To make it easy we deploy on Clever Cloud



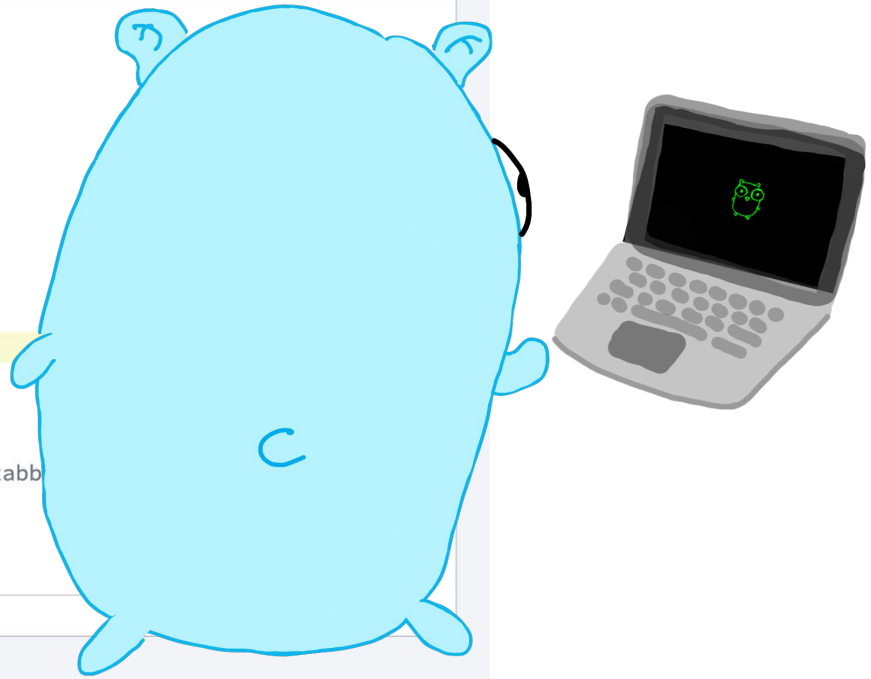
In the logs we get the API key

app_e5ba2fe6-e750-4b34-b807-505c27a7e563

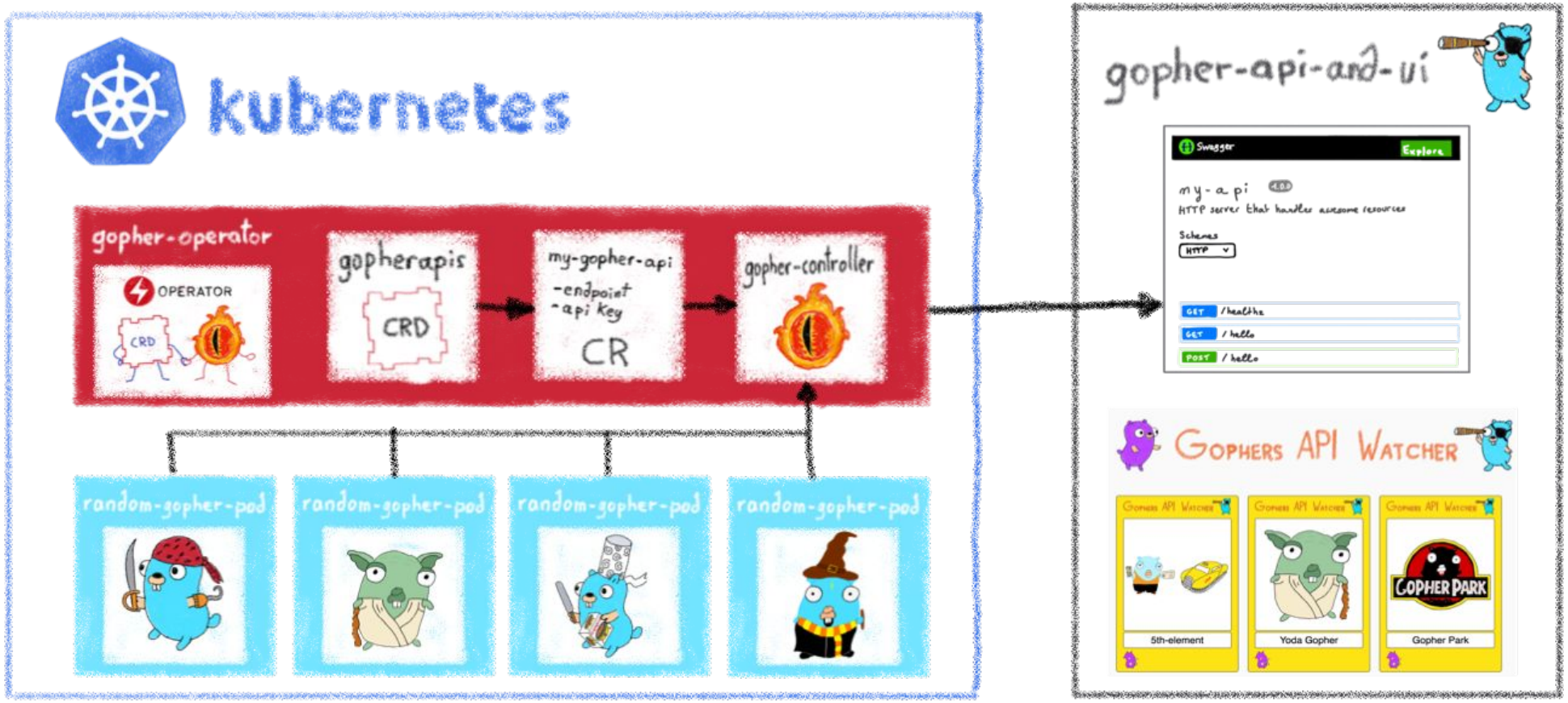
Logs - Let's dive into Kubernetes operator creation - Gopher API and UI

2024-02-05 20:04 OK | All instances | CLEAR LOGS | DOWNLOAD LOGS

```
2024-02-05T20:04:38+01:00 11 packages are looking for funding
2024-02-05T20:04:38+01:00 npm notice
2024-02-05T20:04:38+01:00 found 0 vulnerabilities
2024-02-05T20:04:38+01:00 run `npm fund` for details
2024-02-05T20:04:38+01:00 added 67 packages, and audited 68 packages in 1s
2024-02-05T20:04:38+01:00 npm notice
2024-02-05T20:04:38+01:00 Running as unit: bas-deploy.service
2024-02-05T20:04:38+01:00 npm notice Run `npm install -g npm@10.4.0` to update!
2024-02-05T20:04:38+01:00 build cache archive successfully created
2024-02-05T20:04:40+01:00 start script is node index.js
2024-02-05T20:04:40+01:00 Starting the application…
2024-02-05T20:04:40+01:00 API key o5hy28cu9sd1unn9rnsc1kg131zm08
2024-02-05T20:04:40+01:00 > node index.js
2024-02-05T20:04:40+01:00 > @lostinbrittany/gopher-api-and-ui@1.0.0 start
2024-02-05T20:04:41+01:00 Listening at http://:::8080
2024-02-05T20:04:43+01:00 Created symlink /etc/systemd/system/multi-user.target.wants/zabb
/usr/x86_64-pc-linux-gnu/lib/systemd/system/zabbix-agentd.service.
2024-02-05T20:04:43+01:00 No cron to setup
2024-02-05T20:04:43+01:00 Application start successful
```

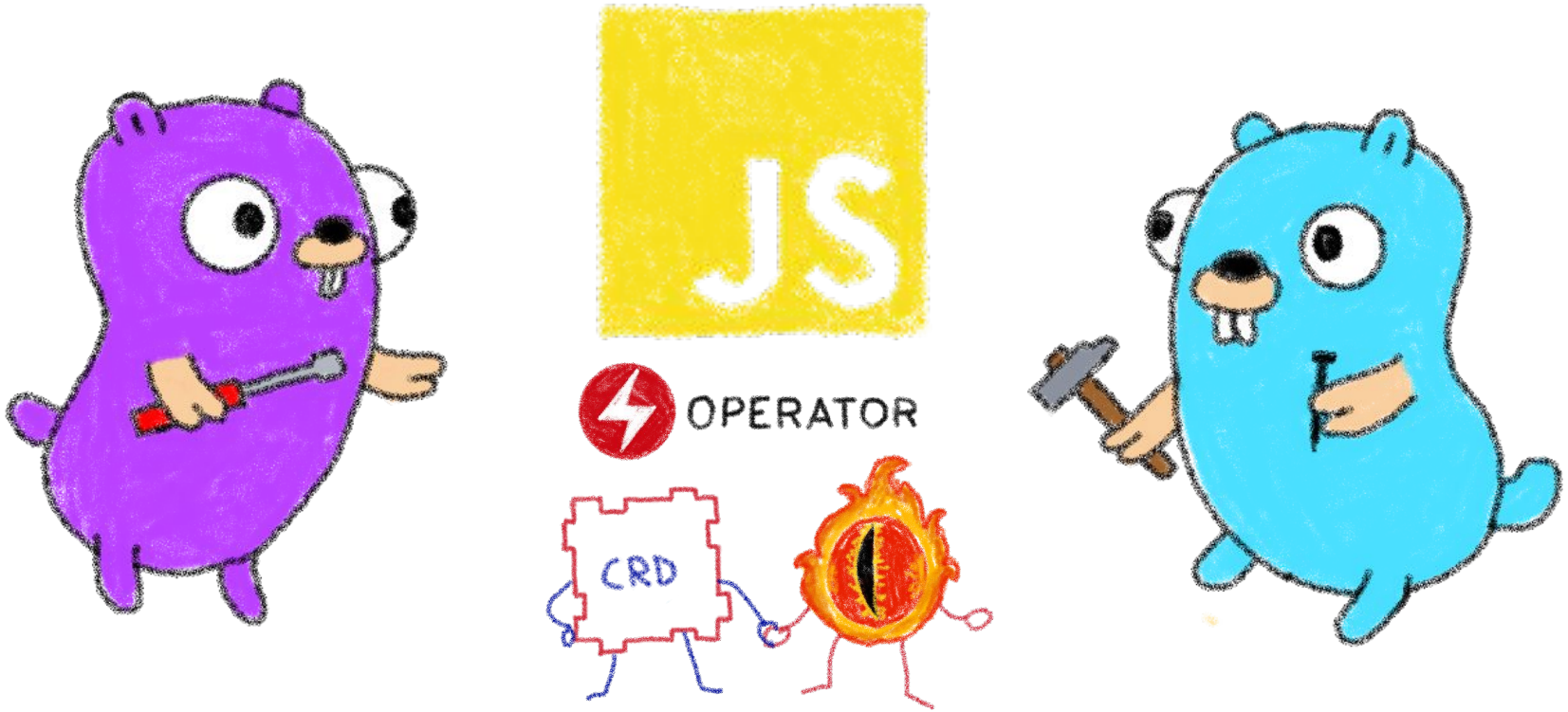


What we want



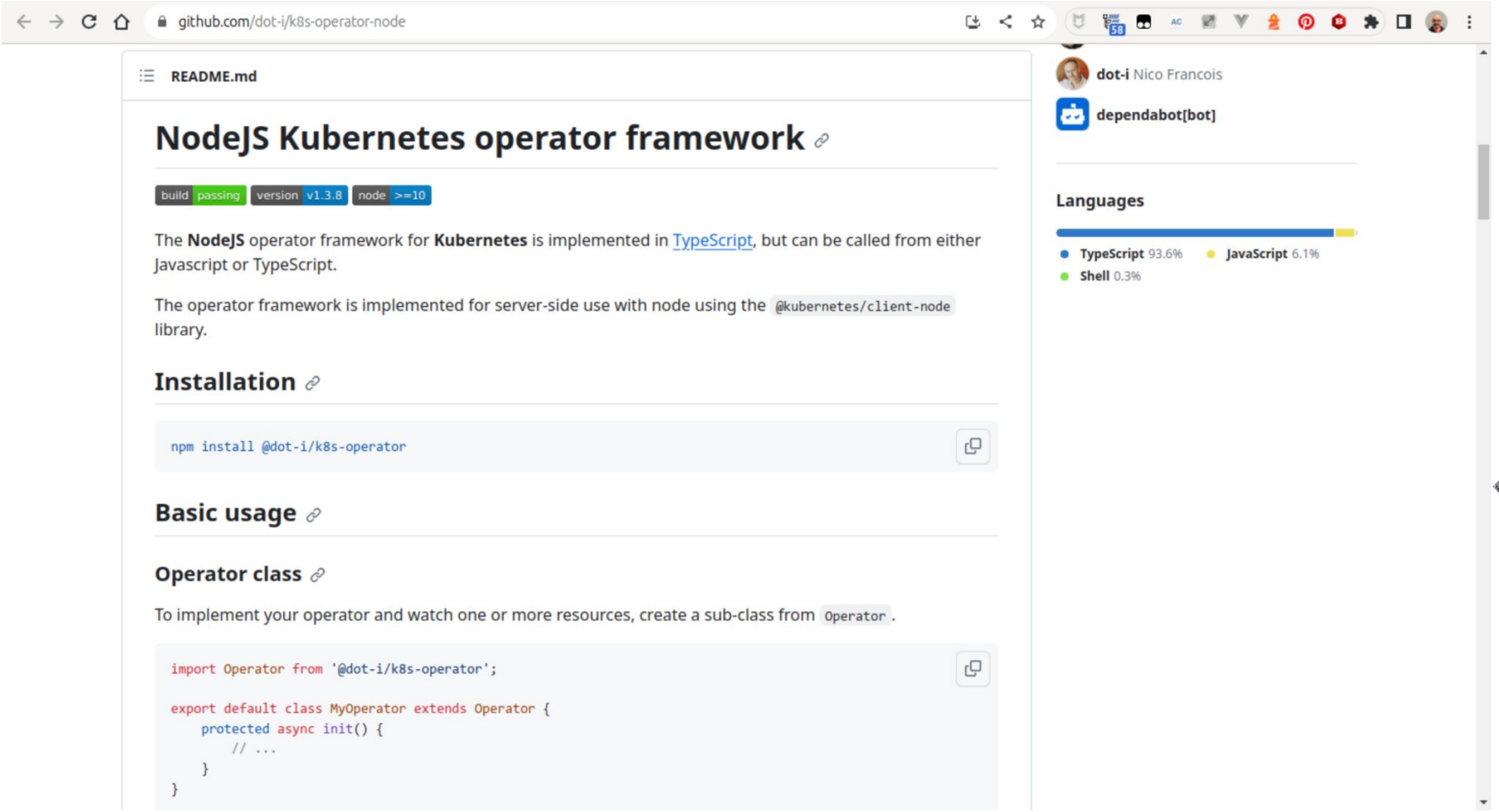
An operator to feed the API with the deployed pods info

And we are doing it in the simplest way



In JavaScript, yeah!

Taking as base k8s-operator-node



github.com/dot-i/k8s-operator-node

README.md

NodeJS Kubernetes operator framework

build passing | version v1.3.8 | node >=10

The **NodeJS** operator framework for **Kubernetes** is implemented in TypeScript, but can be called from either Javascript or TypeScript.

The operator framework is implemented for server-side use with node using the `@kubernetes/client-node` library.

Installation

```
npm install @dot-i/k8s-operator
```

Basic usage

Operator class

To implement your operator and watch one or more resources, create a sub-class from `Operator`.

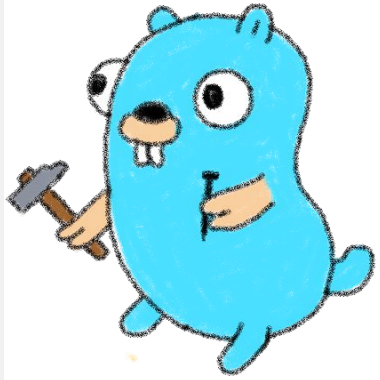
```
import Operator from '@dot-i/k8s-operator';

export default class MyOperator extends Operator {
    protected async init() {
        // ...
    }
}
```

dot-i Nico Francois

dependabot[bot]

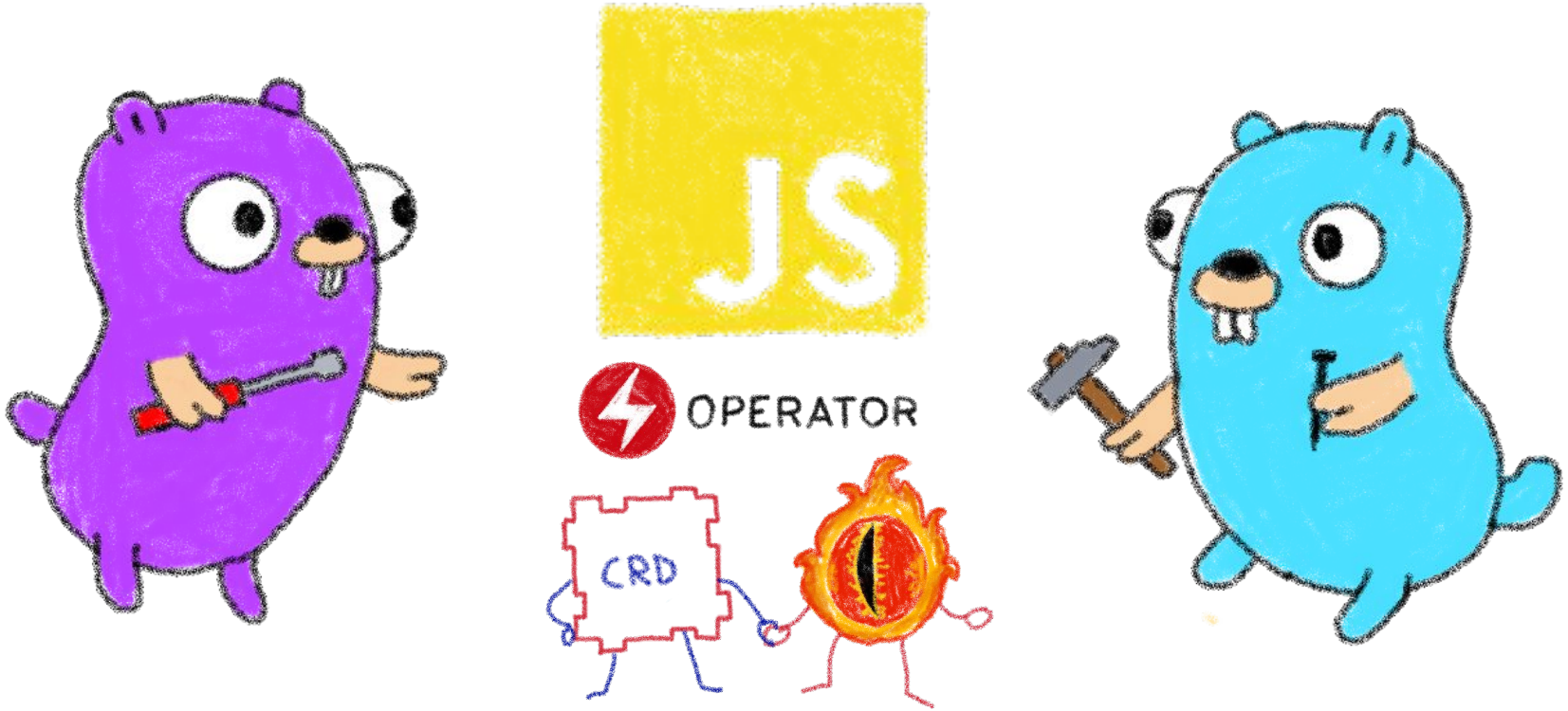
Languages

TypeScript 93.6% JavaScript 6.1% Shell 0.3%



https://github.com/dot-i/k8s-operator-node

Building the gopher operator



Let's switch to VS Code…

That's all, folks!

Thank you all!