OVHcloud intel

containerday

Let's dive into
Kubernetes operator creation

Horacio Gonzalez 2022-10-28

Who are we?

Introducing myself and introducing OVHcloud

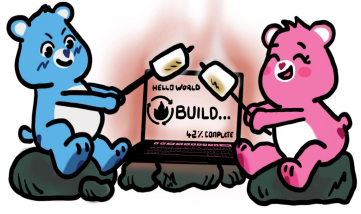
OVHcloud intel

@LostInBrittany

Horacio Gonzalez

@LostInBrittany

Spaniard Lost in Brittany



OVHcloud

DevRel Leader

DevFest du Bout du Monde

Google Developers Experts 2019

Web Technologies GDE

Flutter

OVHcloud: A global leader

- **Web Cloud & Telcom**
- **Private Cloud**
- **Public Cloud**
- **Storage**
- **Network & Security**

30 Data Centers
in 12 locations

34 Points of Presence
on a 20 TBPS Bandwidth Network

2200 Employees
worldwide

115K Private Cloud
VMS running

300K Public Cloud
instances running

380K Physical Servers
running in our data centers

1 Million+ Servers
produced since 1999

1.5 Million Customers
across 132 countries

3.8 Million Websites
hosting

1.5 Billion Euros Invested
since 2016

P.U.E. 1.09
Energy efficiency indicator

20+ Years in Business
Disrupting since 1999

OVHcloud intel

@LostInBrittany

High performance at affordable prices

Infra-4

Processore: 2x Intel Xeon Silver 4214 - 12 c / 24 t - 2.2 GHz / 3.2 GHz
Memoria: A partire da 96GB
Storage: NVMe, SATA disponibile

Banda passante pubblica: A partire da 1 Gbps
Banda passante privata: A partire da 2 Gbps

Disponibile in 7 datacenter	Consegna a partire da 120 s

OVHcloud

HGR-SDS-1

Processore: Intel Xeon Gold 6242R - 20 c / 40 t - 3.1 GHz / 4.1 GHz
Memoria: A partire da 96GB
Storage: NVMe, SAS disponibile

Banda passante pubblica: A partire da 1 Gbps
Banda passante privata: A partire da 10 Gbps

Disponibile in 5 datacenter	Consegna a partire da 120 s

HGR-HCI-2

Processore: 2x Intel Xeon Gold 6242R - 20 c / 40 t - 3.1 GHz / 4.1 GHz
Memoria: A partire da 384GB
Storage: NVMe, SAS disponibile

Banda passante pubblica: A partire da 1 Gbps
Banda passante privata: A partire da 10 Gbps

Disponibile in 5 datacenter	Consegna a partire da 10 g

From bare-metal servers to public or private cloud

OVHcloud intel

@LostInBrittany

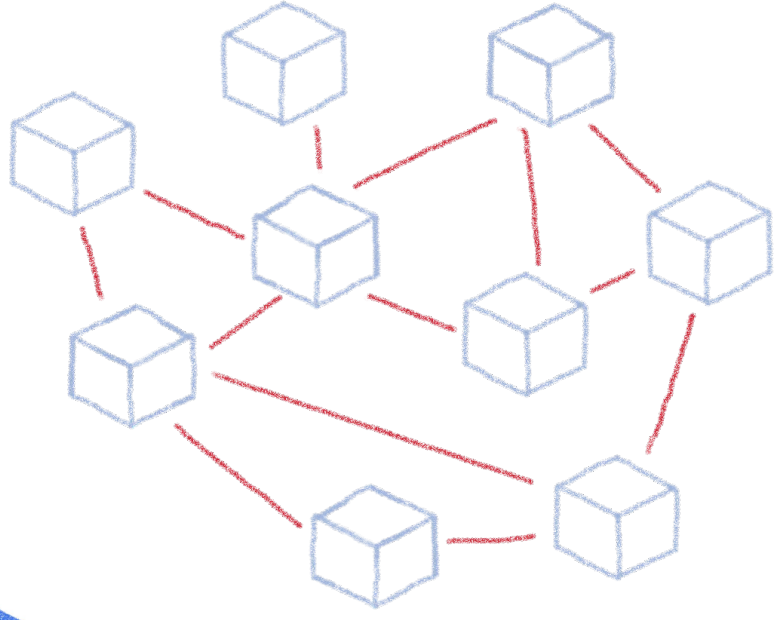
Kubernetes Operators

Helping to tame the complexity of K8s Ops

OVHcloud intel

@LostInBrittany

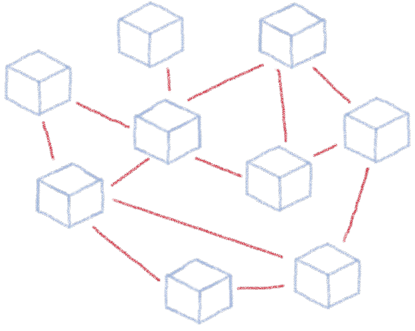
Taming microservices with Kubernetes

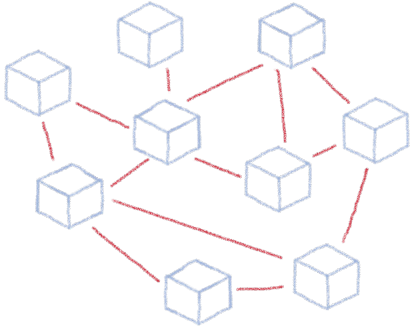


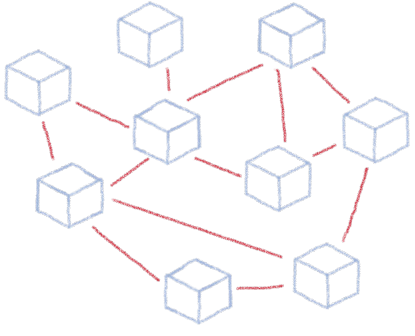
OVHcloud intel

@LostInBrittany

What about complex deployments







Ingress

Services

Deployments

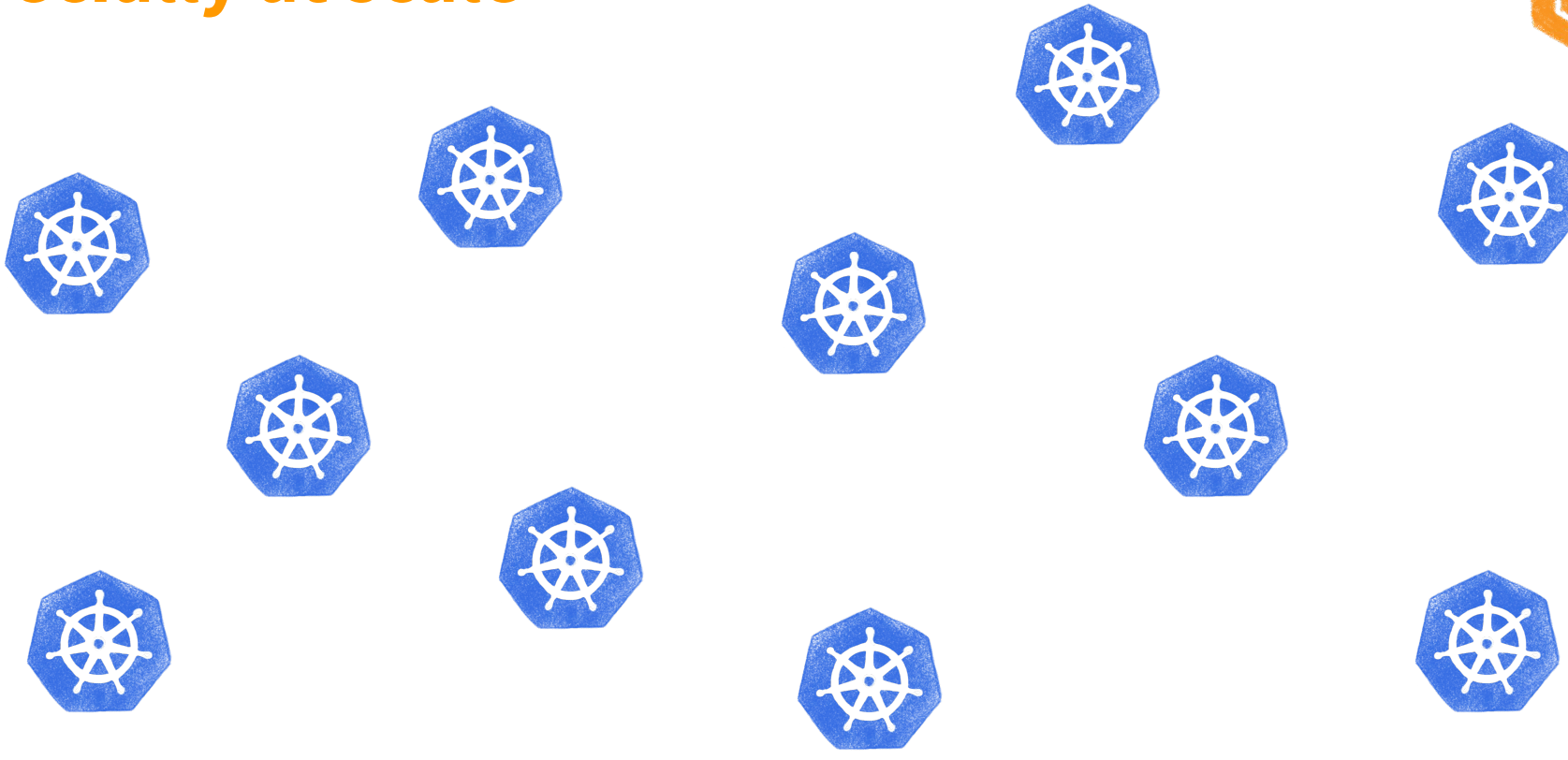
Pods

Sidecars

Replica Sets

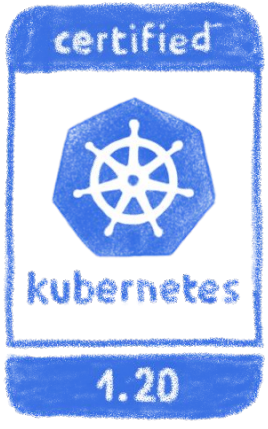
Statepul Sets

Specially at scale



Lots of clusters with lots and lots of deployments

That's just our case

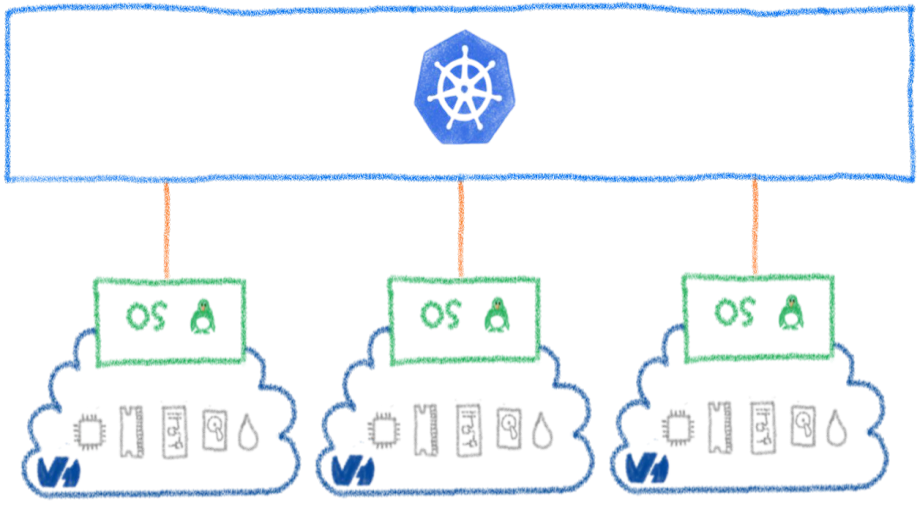


We both use Kubernetes and
operate a Managed Kubernetes platform

OVHcloud intel

@LostInBrittany

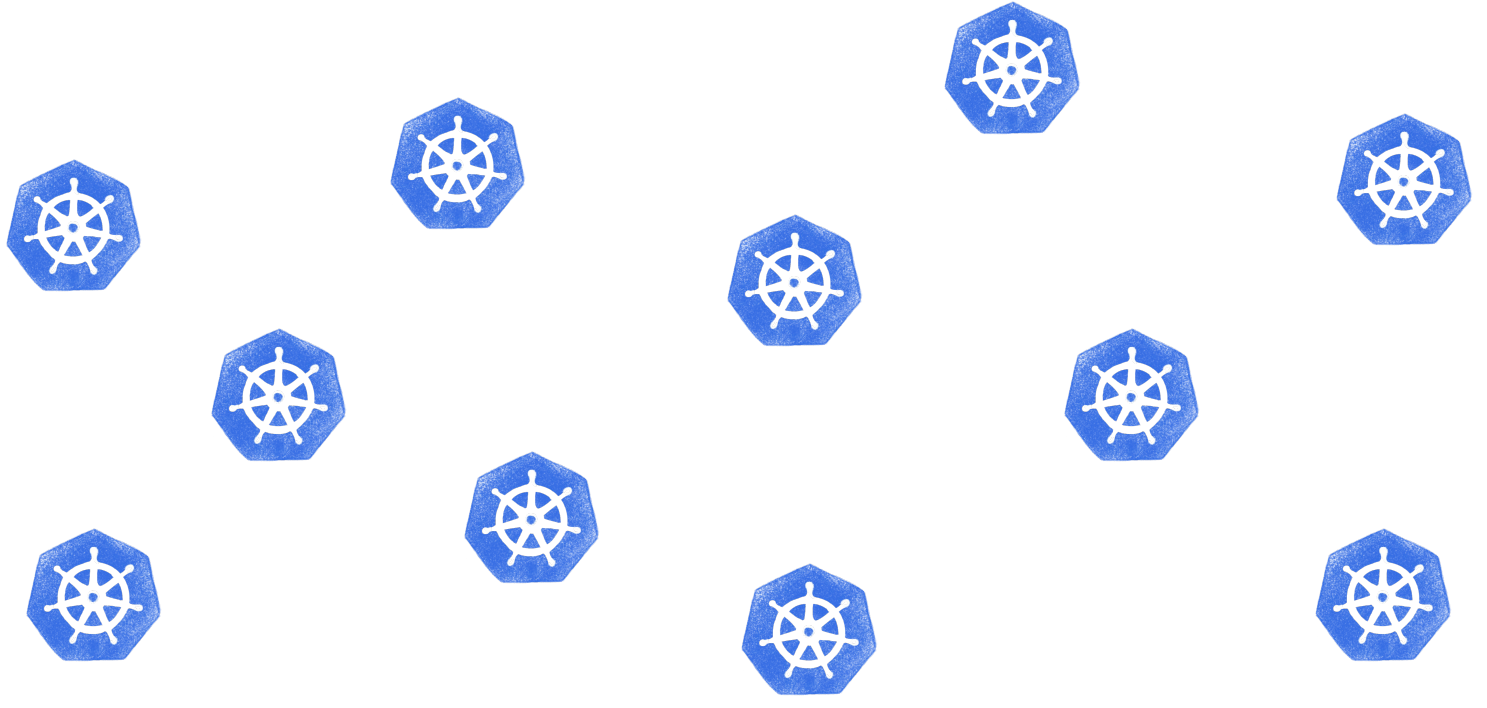
Built over our Openstack based Public Cloud



OVHcloud intel

@LostInBrittany

We need to tame the complexity

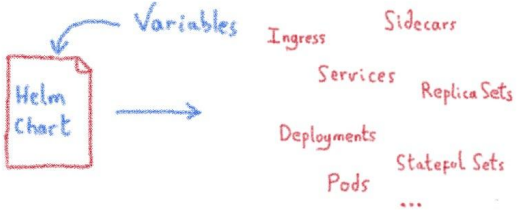


OVHcloud intel

@LostInBrittany

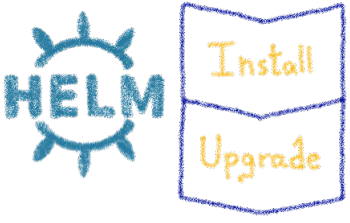
Taming the complexity

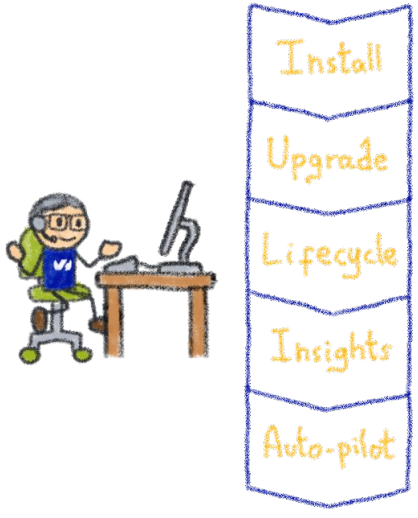
A package manager for Kubernetes



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

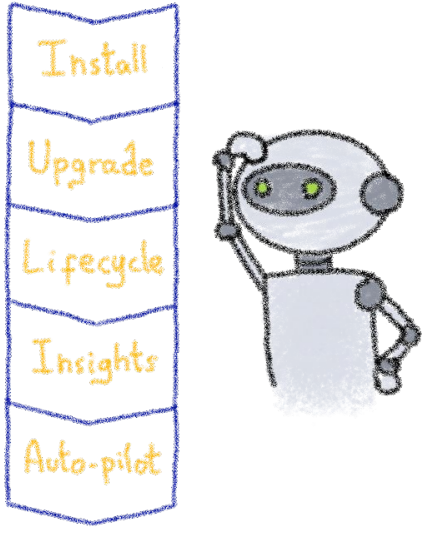
Helm Charts are configuration





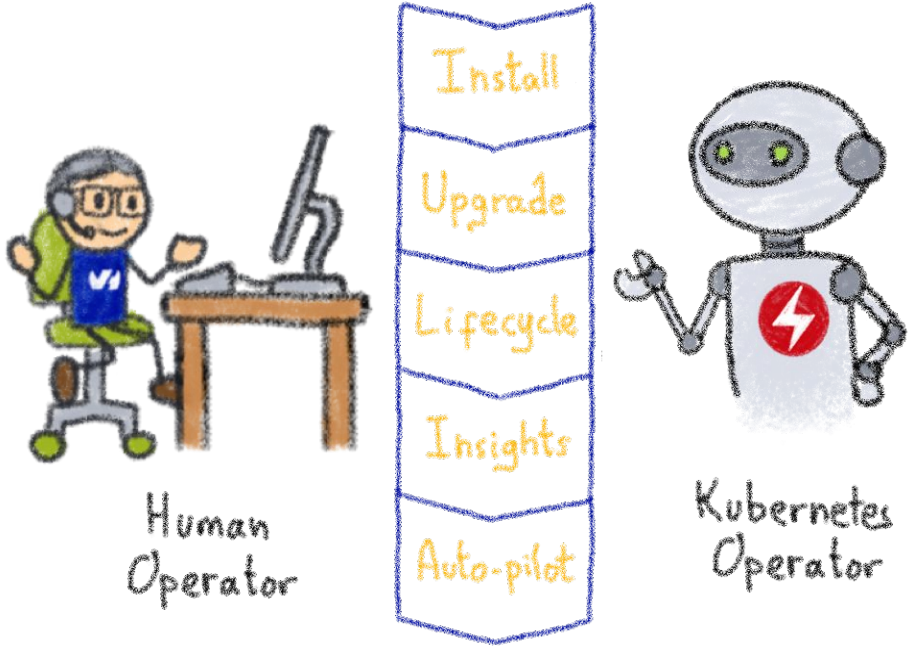
Operating is more than installs & upgrades

Kubernetes is about automation



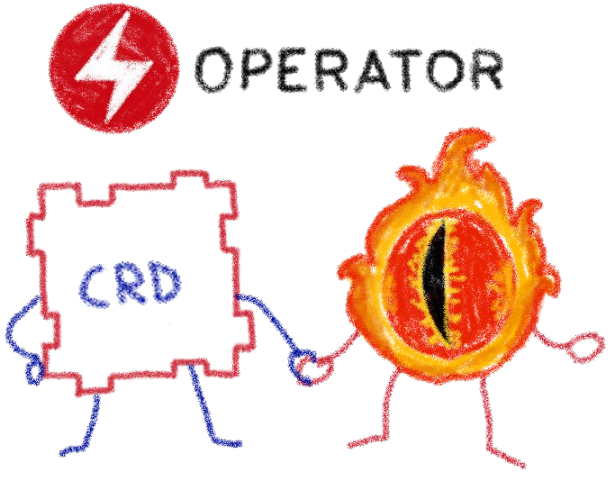
How about automating human operators?

Kubernetes Operators



A Kubernetes version of the human operator

Building operators



Basic K8s elements: Controllers and Custom Resources

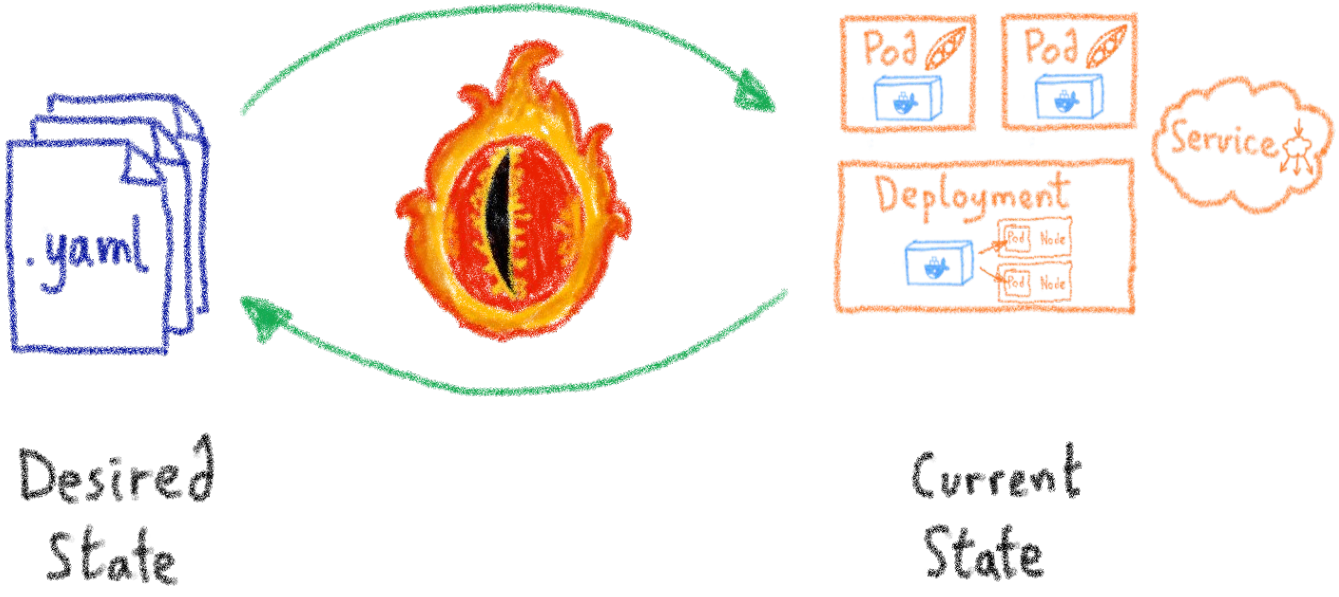
Kubernetes Controllers

Keeping an eye on the resources

OVHcloud intel

@LostInBrittany

A control loop



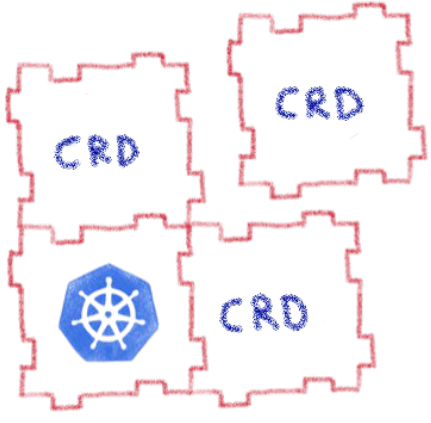
They watch the state of the cluster,
and make or request changes where needed

A reconcile loop

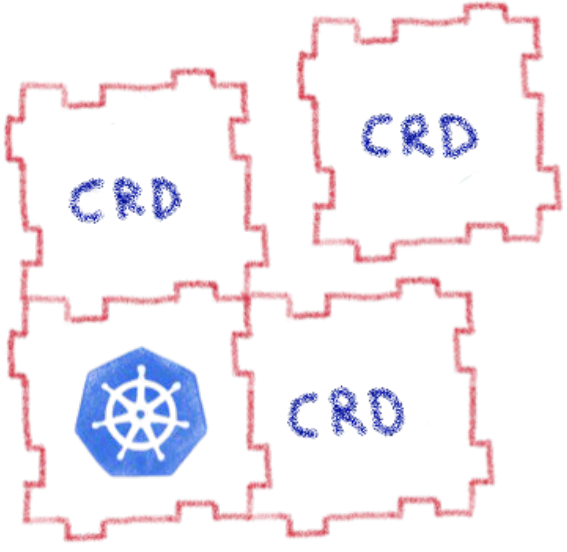
Strives to reconcile current state and desired state

Custom Resource Definitions

Extending Kubernetes API

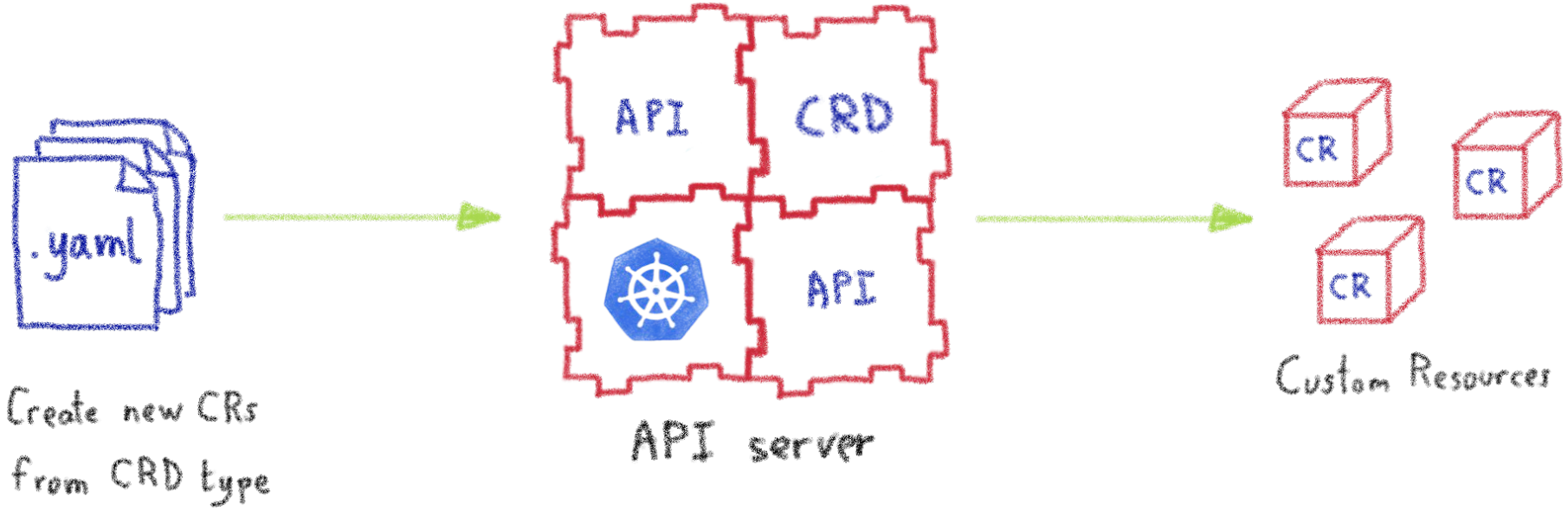


Extending Kubernetes API



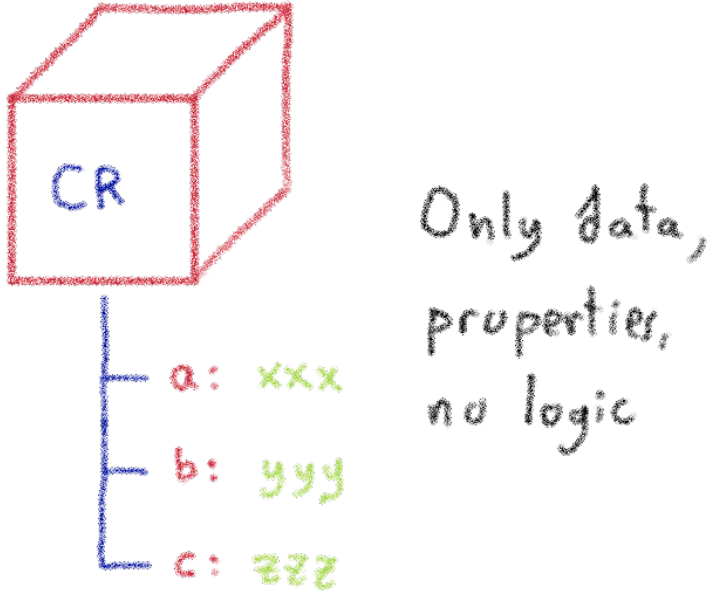
By defining new types of resources

With a CRD you can create CR in the cluster



They are the blueprints of the Custom Resources

Custom Resources are simply data



All the logic must be in the Controller

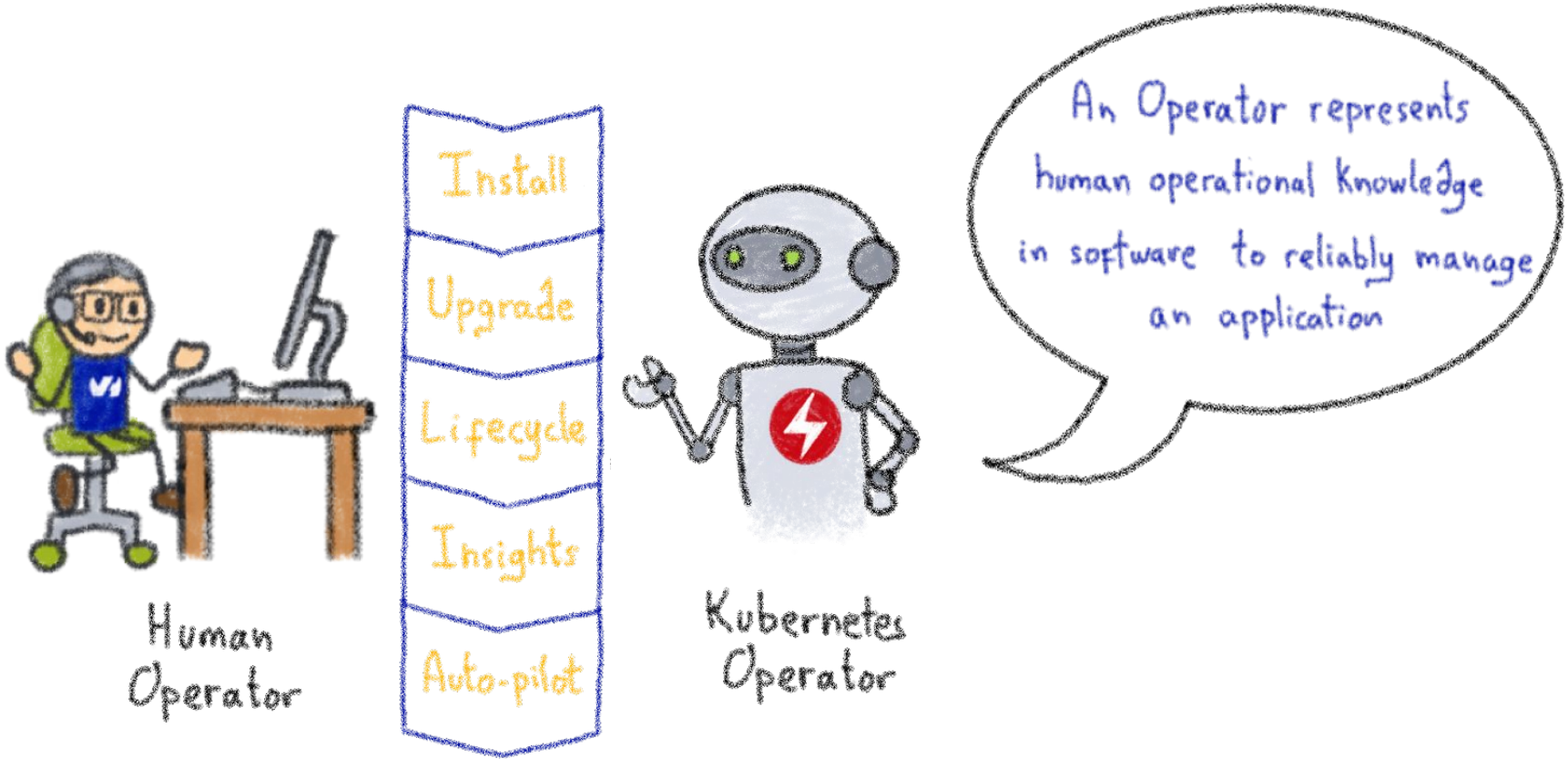
Kubernetes Operator

Automating operations

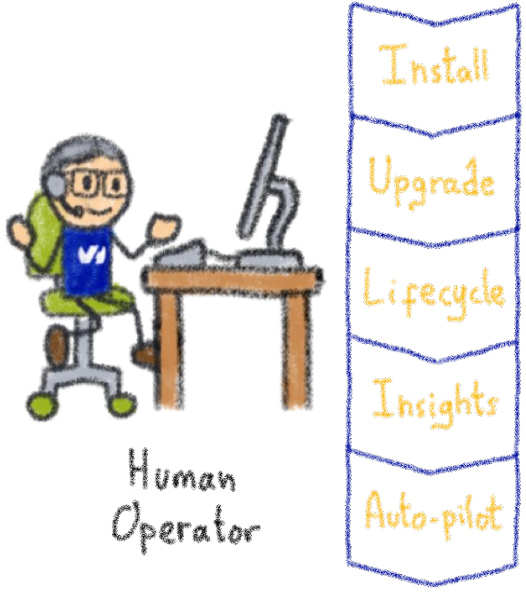
OVHcloud intel

@LostInBrittany

What's a Kubernetes Operator?



Example: databases

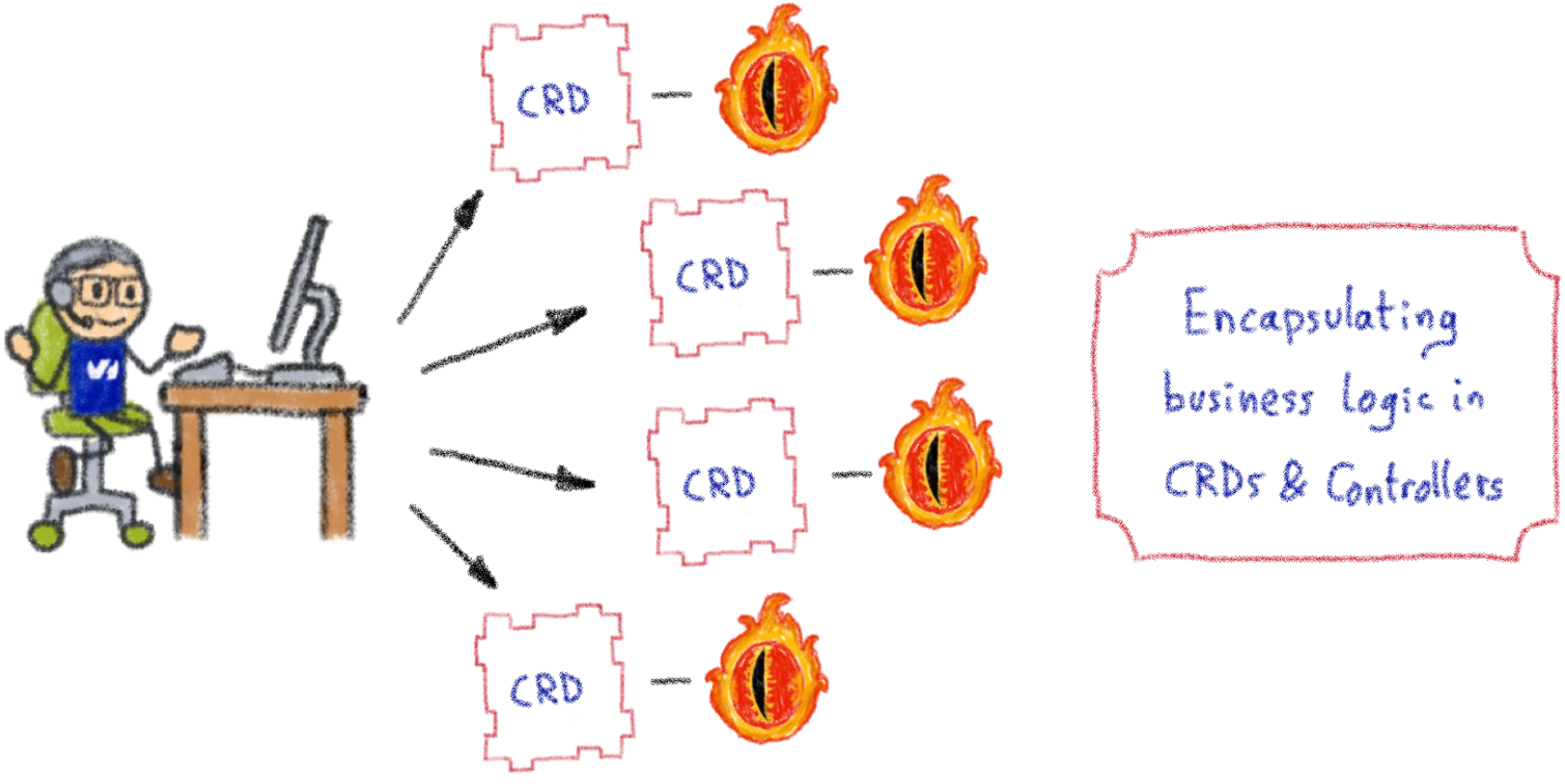


PostgreSQL

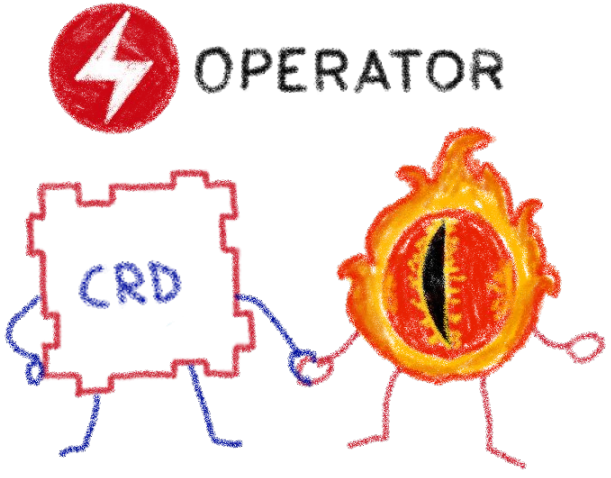
redis

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

Knowledge encoded in CRDs and Controllers

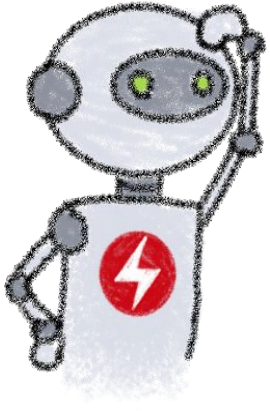


Custom Controllers for Custom Resources



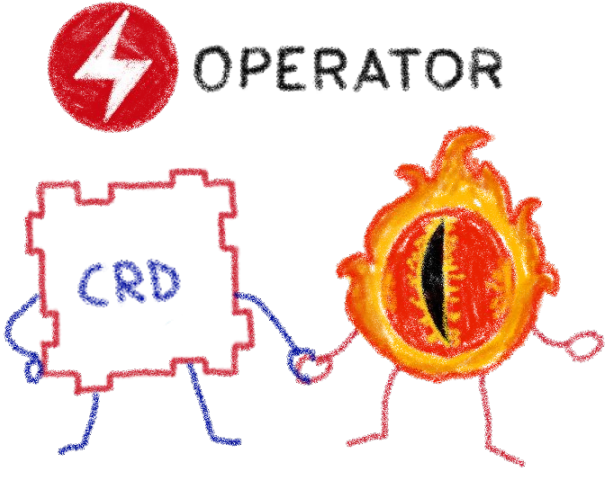
Operators implement and manage Custom Resources
using custom reconciliation logic

Operator Capability Model



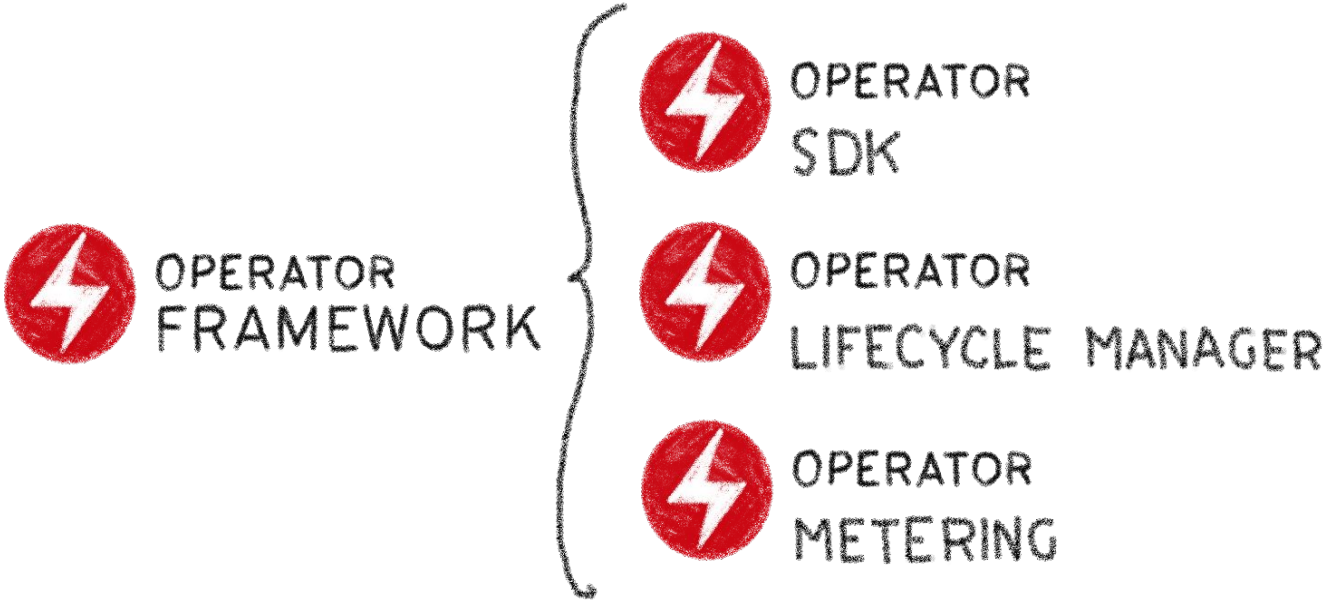
Gauging the operator maturity

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

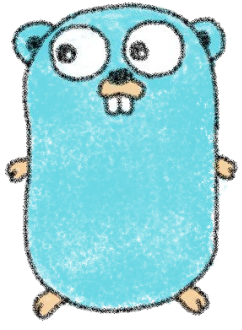


Open source framework to accelerate
the development of an Operator

Using Operator SDK with Go

Let's see how we code an operator!

OPERATOR
SDK



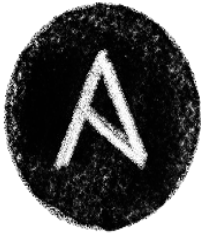
OVHcloud intel

@LostInBrittany

Operator SDK

OPERATOR
SDK

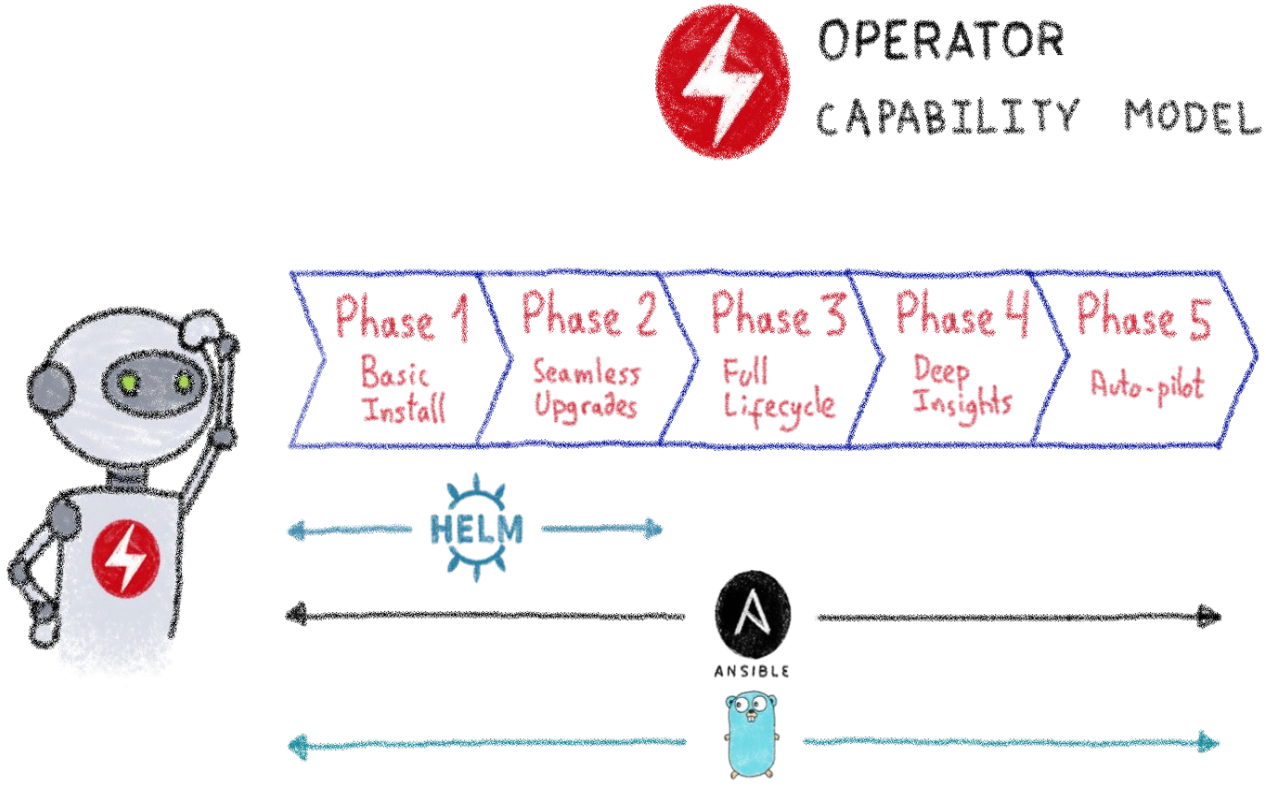
BUILD
TEST
ITERATE



ANSIBLE

Three different ways to build an Operator

Operator SDK and Capability Model



Installing the Operator SDK CLI

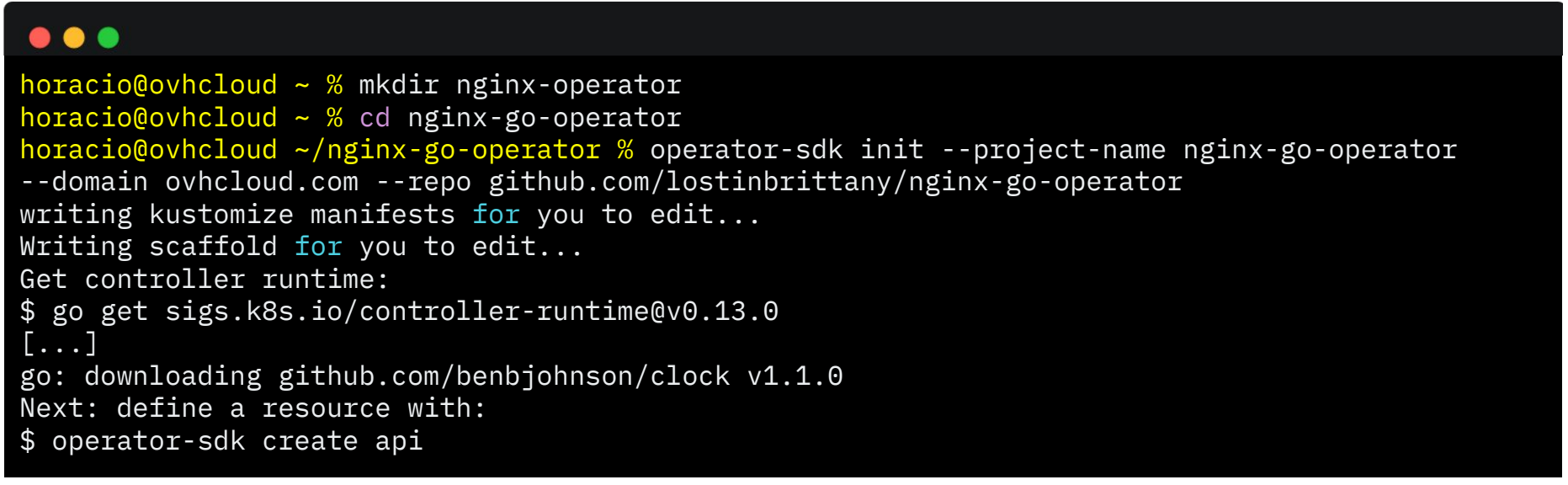
```
horacio@ovhcloud ~ % export ARCH=$(case $(uname -m) in x86_64) echo -n amd64 ;; aarch64) echo
-n arm64 ;; *) echo -n $(uname -m) ;; esac)
export OS=$(uname | awk '{print tolower($0)}')
horacio@ovhcloud ~ % echo $ARCH $OS
arm64 darwin
horacio@ovhcloud ~ % export
OPERATOR_SDK_DL_URL=https://github.com/operator-framework/operator-sdk/releases/download/v1.25
.0
curl -LO ${OPERATOR_SDK_DL_URL}/operator-sdk_${OS}_${ARCH}
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 80.6M  100 80.6M    0     0  1845k      0  0:00:44  0:00:44 --:--:-- 1947k
horacio@ovhcloud ~ % chmod +x operator-sdk_${OS}_${ARCH}
horacio@ovhcloud ~ % mv operator-sdk_${OS}_${ARCH} /usr/local/bin/operator-sdk
horacio@ovhcloud ~ % operator-sdk
CLI tool for building Kubernetes extensions and tools.
[...]
```

https://sdk.operatorframework.io/docs/installation/

OVHcloud intel

@LostInBrittany

An example operator in Go: Nginx



```
horacio@ovhcloud ~ % mkdir nginx-operator
horacio@ovhcloud ~ % cd nginx-go-operator
horacio@ovhcloud ~/nginx-go-operator % operator-sdk init --project-name nginx-go-operator
--domain ovhcloud.com --repo github.com/lostinbrittany/nginx-go-operator
writing kustomize manifests for you to edit...
Writing scaffold for you to edit...
Get controller runtime:
$ go get sigs.k8s.io/controller-runtime@v0.13.0
[...]
go: downloading github.com/benbjohnson/clock v1.1.0
Next: define a resource with:
$ operator-sdk create api
```

Note If your local environment is Apple Silicon (`darwin/arm64`) use the `go/v4-alpha` plugin which provides support for this platform by adding to the init subCommand the flag `--plugins=go/v4-alpha`

https://docs.ovh.com/gb/en/kubernetes/deploying-go-operator/

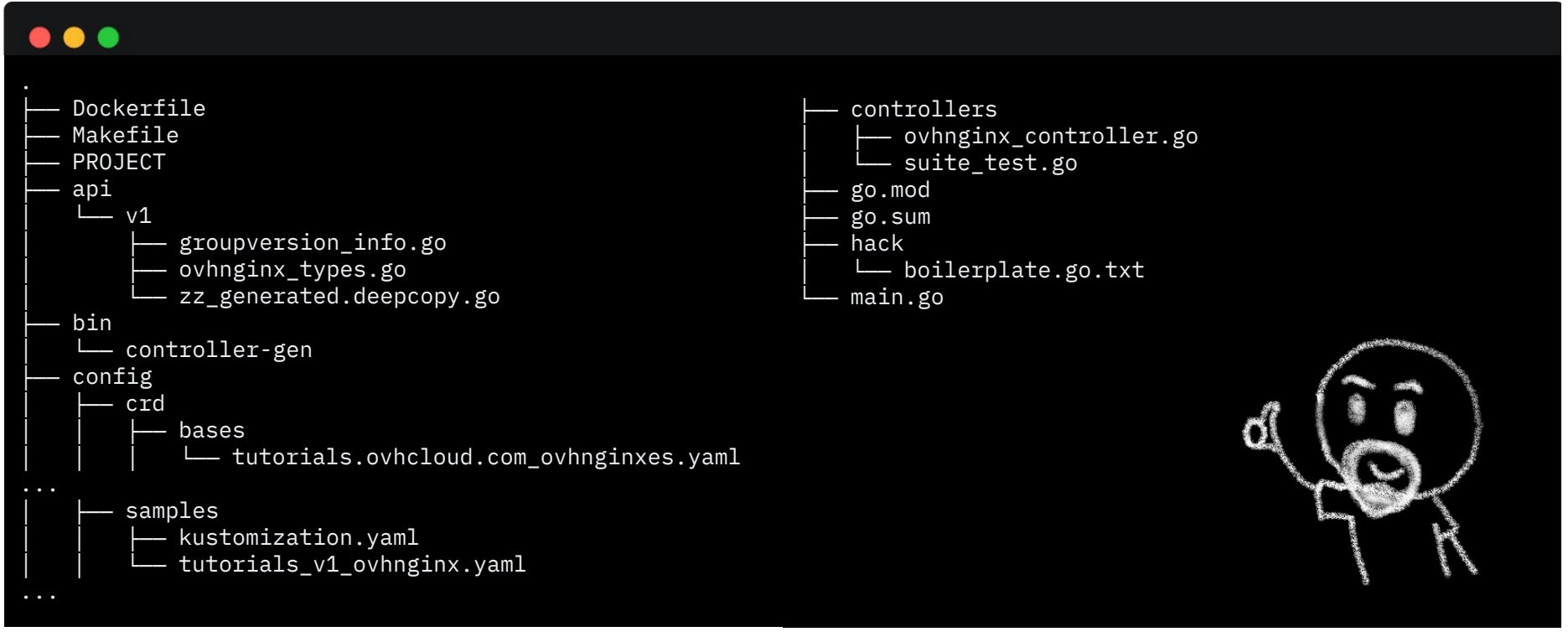
A scaffold has been generated

```
.
├── Dockerfile
├── Makefile
├── PROJECT
├── config
│   ├── default
│   ├── manager
│   ├── manifests
│   ├── prometheus
│   ├── rbac
│   └── scorecard
│       ├── bases
│       └── patches
├── go.mod
├── go.sum
├── hack
│   └── boilerplate.go.txt
└── main.go
```

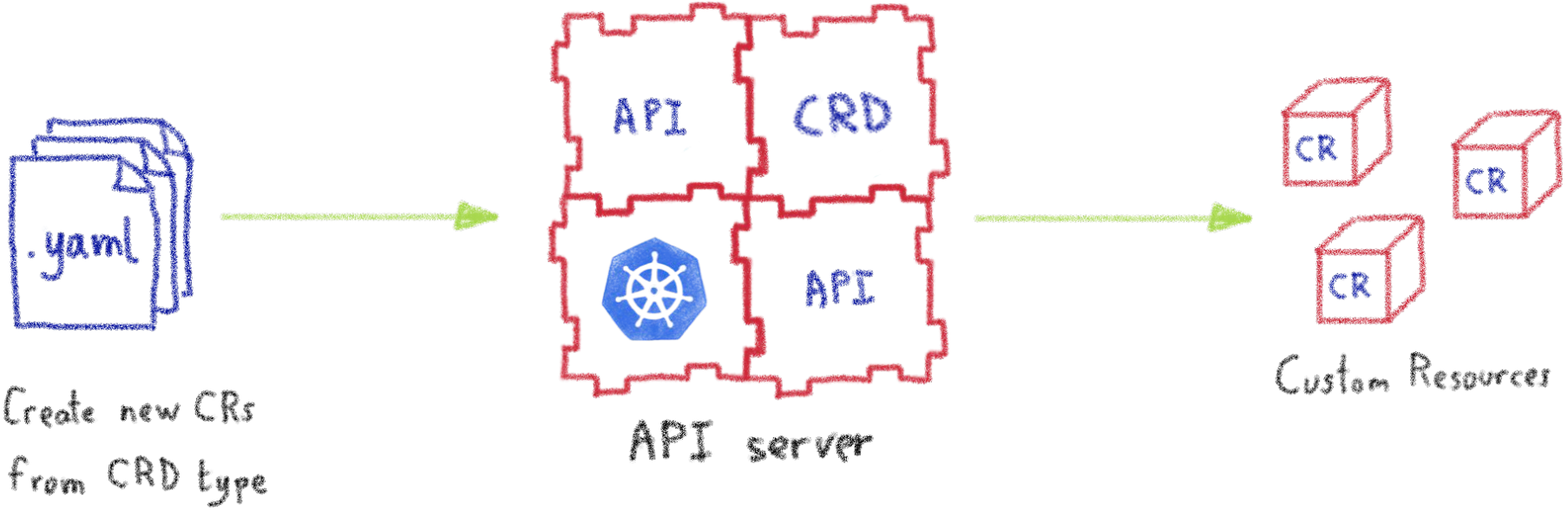
Custom resources definition and controller

```
horacio@ovhcloud ~/nginx-go-operator % operator-sdk create api --group tutorials --version v1
--kind OvhNginx --resource --controller
Writing kustomize manifests for you to edit...
Writing scaffold for you to edit...
api/v1/ovhnginx_types.go
controllers/ovhnginx_controller.go
Update dependencies:
$ go mod tidy
Running make:
$ make generate
mkdir -p /workspace/k8s-and-golang-gitpod/nginx-go-operator/bin[...]
/workspace/k8s-and-golang-gitpod/nginx-go-operator/bin/controller-gen
object:headerFile="hack/boilerplate.go.txt" paths="./..."
Next: implement your new API and generate the manifests (e.g. CRDs,CRs) with:
$ make manifests

horacio@ovhcloud ~/nginx-go-operator % make manifests
/workspace/k8s-and-golang-gitpod/nginx-go-operator/bin/controller-gen
rbac:roleName=manager-role crd webhook paths="./..."
output:crd:artifacts:config=config/crd/bases
```

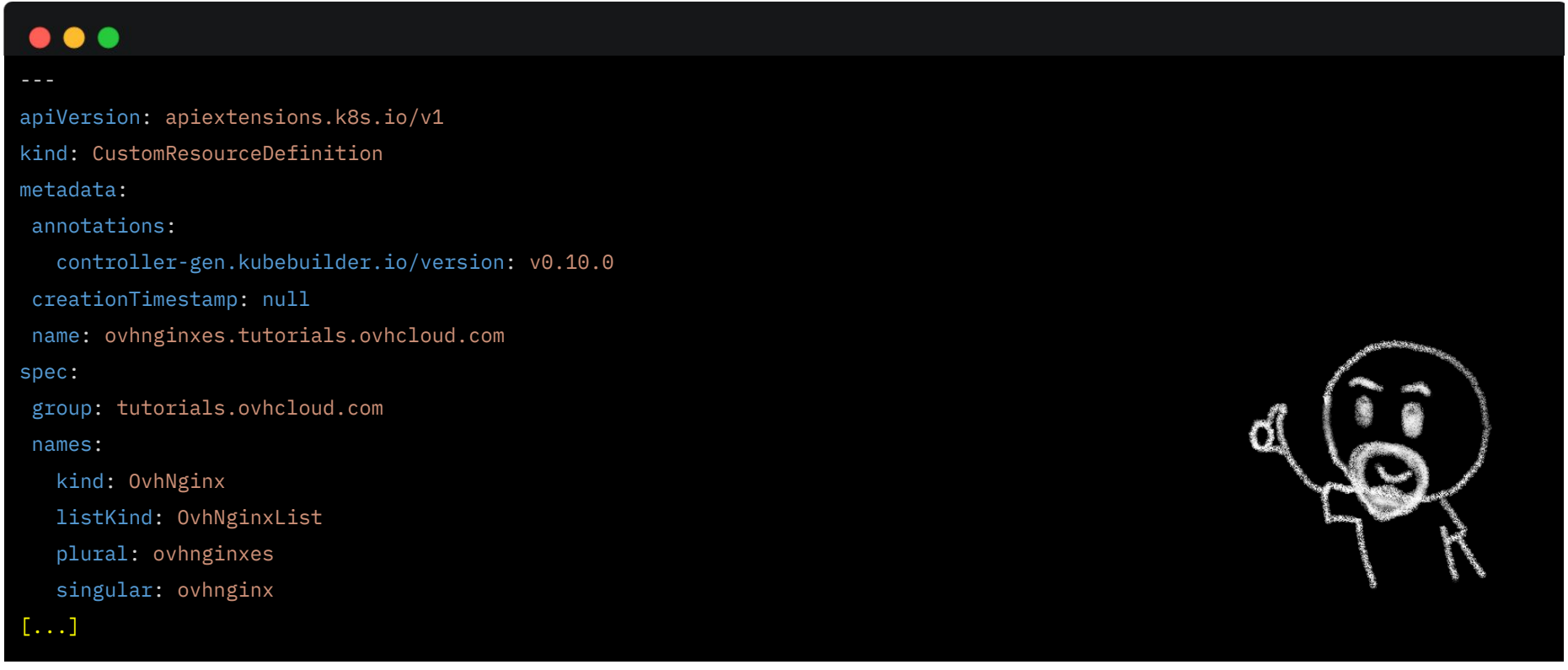

CRD and Controller are scaffolded



Let's look at the CRD and its CRs



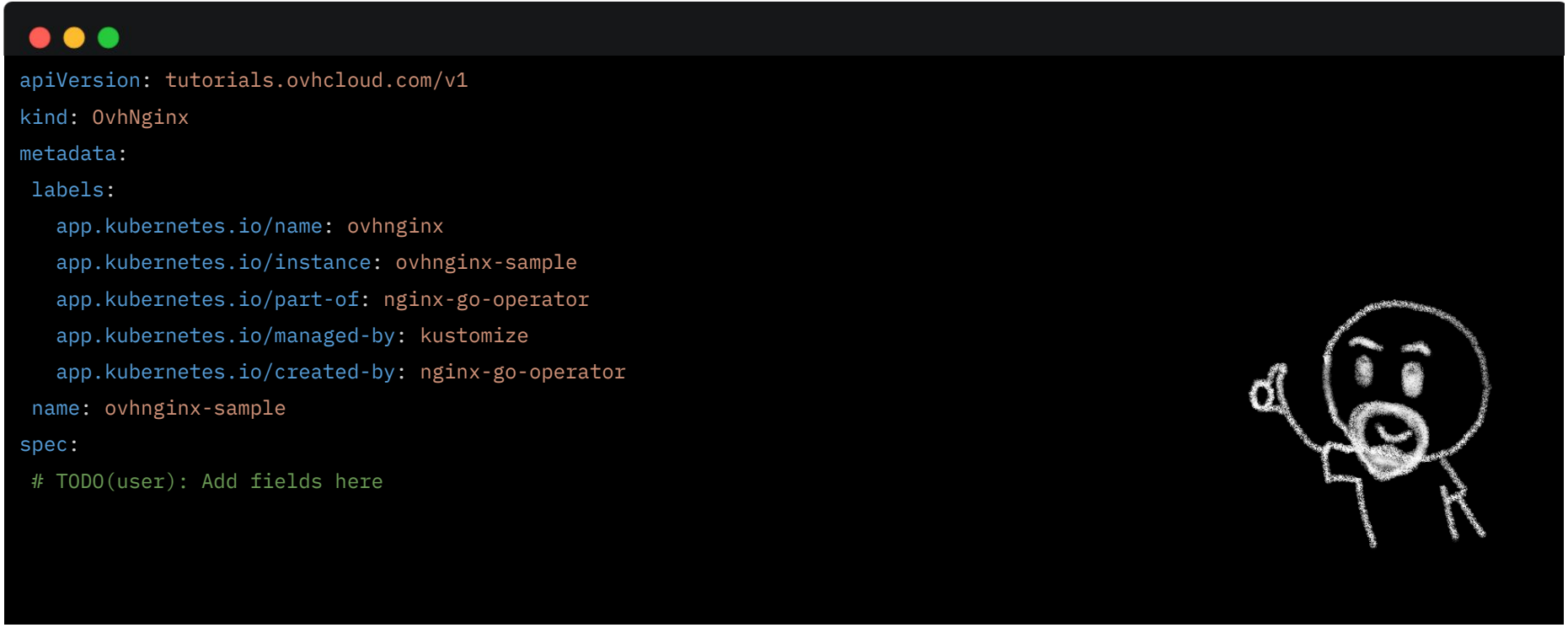
CRD: tutorials.ovhcloud.com_ovhnginxes.yaml



```yaml
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
 annotations:
   controller-gen.kubebuilder.io/version: v0.10.0
 creationTimestamp: null
 name: ovhnginxes.tutorials.ovhcloud.com
spec:
 group: tutorials.ovhcloud.com
 names:
   kind: OvhNginx
   listKind: OvhNginxList
   plural: ovhnginxes
   singular: ovhnginx
[...]
```


config/crd/based/tutorials.ovhcloud.com_ovhnginxes.yaml

Sample CR: `tutorials_v1_ovhnginx.yaml`



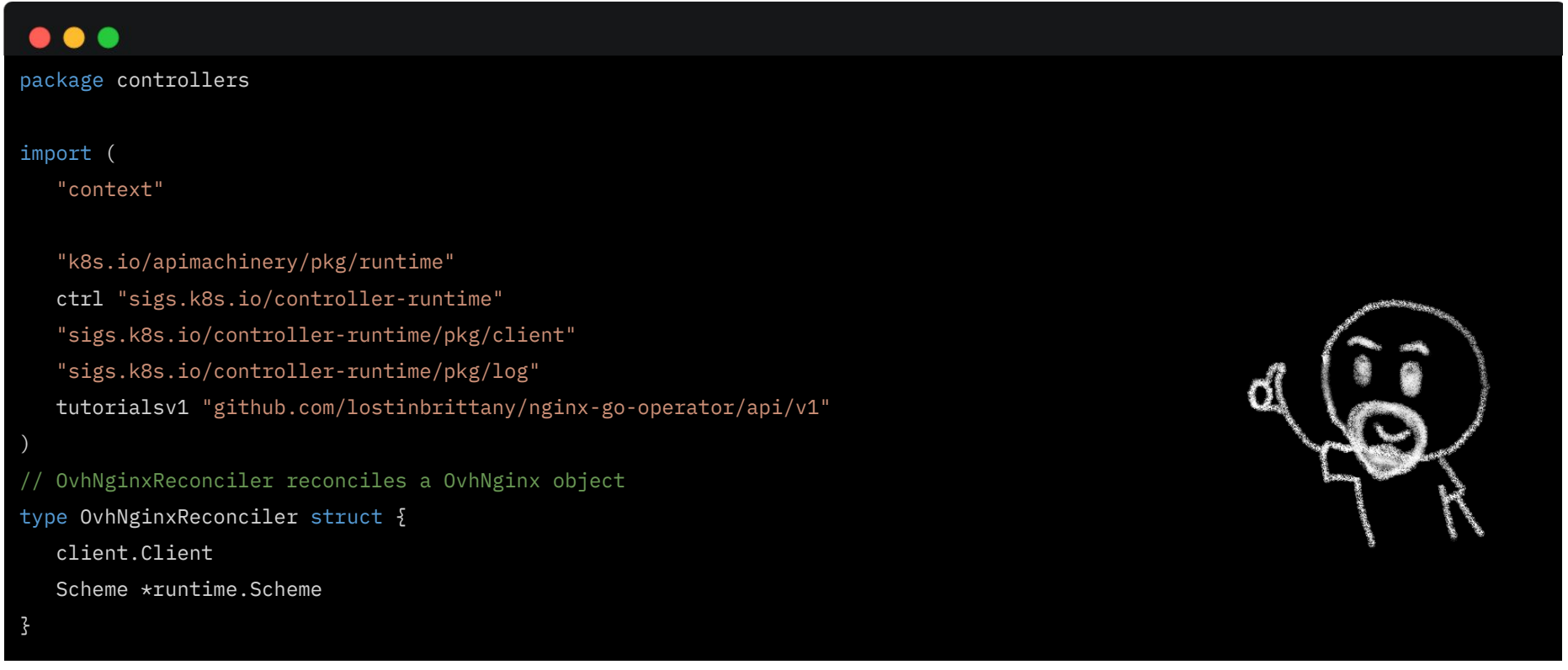
```yaml
apiVersion: tutorials.ovhcloud.com/v1
kind: OvhNginx
metadata:
 labels:
   app.kubernetes.io/name: ovhnginx
   app.kubernetes.io/instance: ovhnginx-sample
   app.kubernetes.io/part-of: nginx-go-operator
   app.kubernetes.io/managed-by: kustomize
   app.kubernetes.io/created-by: nginx-go-operator
 name: ovhnginx-sample
spec:
 # TODO(user): Add fields here
```


`config/samples/tutorials_v1_ovhnginx.yaml`

OVHcloud intel

@LostInBrittany

Go controller: ovhnginx_controller.go



```go
package controllers

import (
    "context"

    "k8s.io/apimachinery/pkg/runtime"
    ctrl "sigs.k8s.io/controller-runtime"
    "sigs.k8s.io/controller-runtime/pkg/client"
    "sigs.k8s.io/controller-runtime/pkg/log"
    tutorialsv1 "github.com/lostinbrittany/nginx-go-operator/api/v1"
)
// OvhNginxReconciler reconciles a OvhNginx object
type OvhNginxReconciler struct {
    client.Client
    Scheme *runtime.Scheme
}
```


controllers/ovhnginx_controller.go

OVHcloud intel

@LostInBrittany

Let's make it do something...

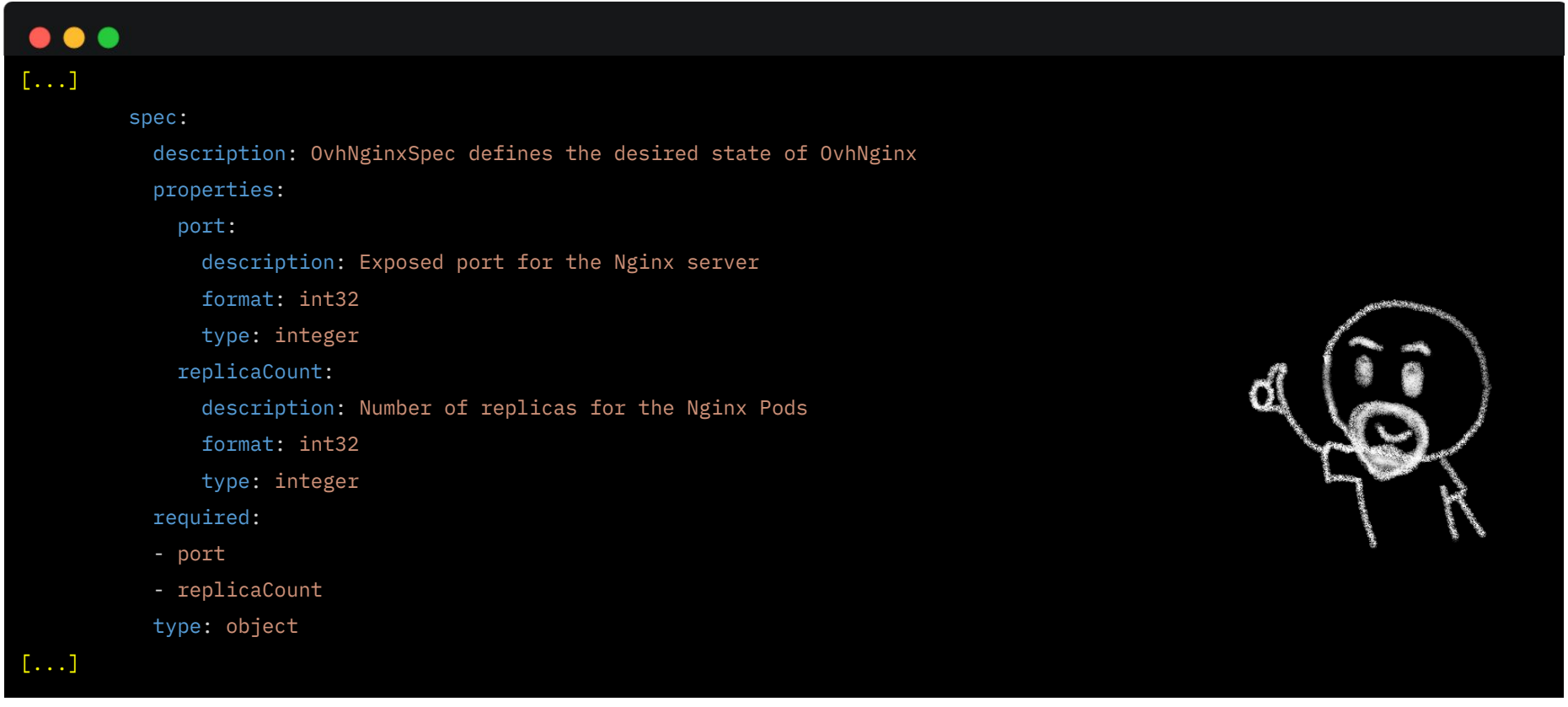
```go
import (
    metav1 "k8s.io/apimachinery/pkg/apis/meta/v1"
)

// OvhNginxSpec defines the desired state of OvhNginx
type OvhNginxSpec struct {
    // INSERT ADDITIONAL SPEC FIELDS - desired state of cluster
    // Important: Run "make" to regenerate code after modifying this file

    // Number of replicas for the Nginx Pods
    ReplicaCount int32 `json:"replicaCount"`
    // Exposed port for the Nginx server
    Port int32 `json:"port"`
}
```

Adding fields to manage the Nginx server
by updating `api/v1/ovhnginx_types.go`

OVHcloud intel

@LostInBrittany

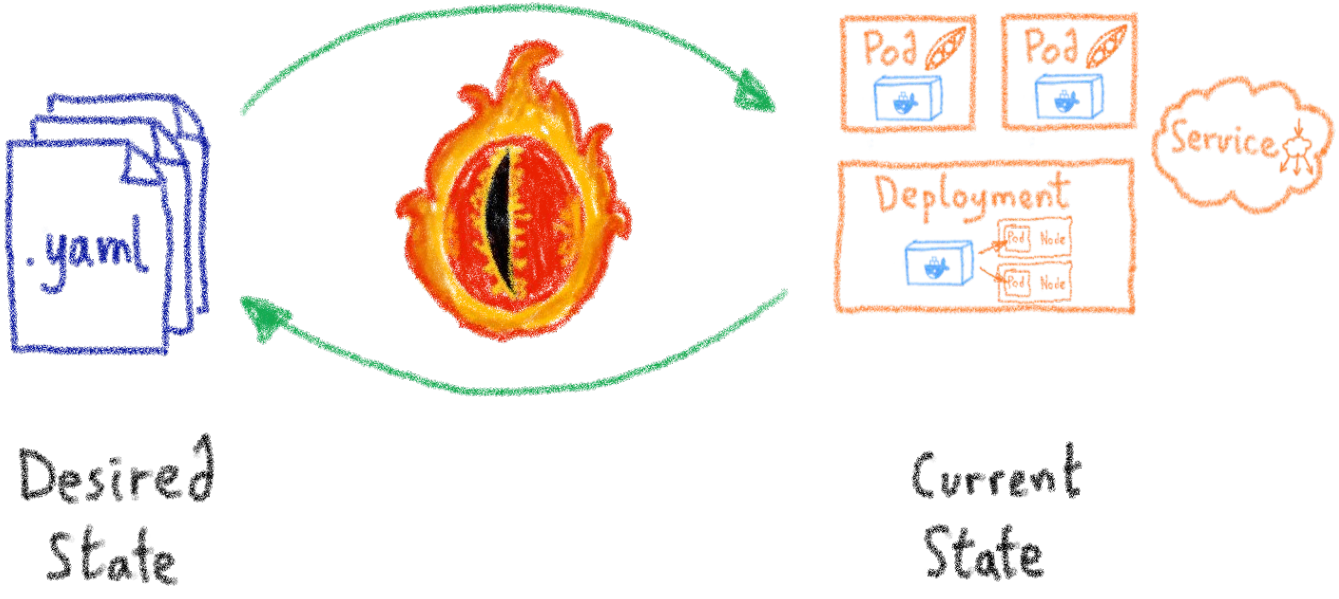
After a `make manifests`



```
[...]
        spec:
          description: OvhNginxSpec defines the desired state of OvhNginx
          properties:
            port:
              description: Exposed port for the Nginx server
              format: int32
              type: integer
            replicaCount:
              description: Number of replicas for the Nginx Pods
              format: int32
              type: integer
          required:
          - port
          - replicaCount
          type: object
[...]
```


config/crd/based/tutorials.ovhcloud.com_ovhnginxes.yaml

What about the Controller?



We need to work on the reconciler fonction

The reconciler in `ovhnginx_controller.go`

Several steps:

```
if CR doesn't exist delete Deployment and/or Service if they exist
else
    if Deployment doesn't exist create it
    else update it if needed

    if Service doesn't exist create it
    else update it if needed
```


The reconciler in `ovhnginx_controller.go`

Several steps:

```
if CR doesn't exist delete Deployment and/or Service if they exist
else
    if Deployment doesn't exist create it
    else update it if needed

    if Service doesn't exist create it
    else update it if needed
```


Test the operator in “dev mode”

```
horacio@ovhcloud ~/nginx-go-operator % make install run
test -s /workspace/k8s-and-golang-gitpod/nginx-go-operator/bin/controller-gen ||
GOBIN=/workspace/k8s-and-golang-gitpod/nginx-go-operator/bin go install
sigs.k8s.io/controller-tools/cmd/controller-gen@v0.10.0
/workspace/k8s-and-golang-gitpod/nginx-go-operator/bin/controller-gen
rbac:roleName=manager-role crd webhook paths="./..."
output:crd:artifacts:config=config/crd/bases
/workspace/k8s-and-golang-gitpod/nginx-go-operator/bin/kustomize build config/crd | kubectl
apply -f -
customresourcedefinition.apiextensions.k8s.io/ovhnginxes.tutorials.ovhcloud.com created
/workspace/k8s-and-golang-gitpod/nginx-go-operator/bin/controller-gen
object:headerFile="hack/boilerplate.go.txt" paths="./..."
go fmt ./...
go vet ./...
go run ./main.go
1.666943595602441e+09   INFO    controller-runtime.metrics      Metrics server is starting to
listen    {"addr": ":8080"}
[...]
```

OVHcloud intel

@LostInBrittany

Create a CR from the CRD

```yaml
apiVersion: tutorials.ovhcloud.com/v1
kind: OvhNginx
metadata:
 name: ovhnginx-sample
spec:
 port: 80
 replicaCount: 1
```

config/samples/tutorials_v1_ovhnginx.yaml

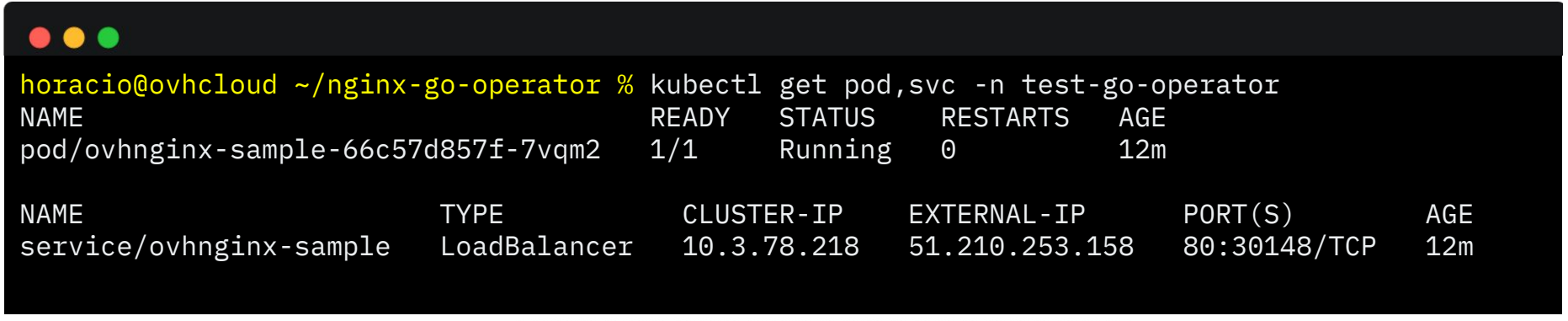
Apply it to the cluster

```
horacio@ovhcloud ~/nginx-go-operator % kubectl apply -f
./config/samples/tutorials_v1_ovhnginx.yaml -n test-go-operator
ovhnginx.tutorials.ovhcloud.com/ovhnginx-sample created
```

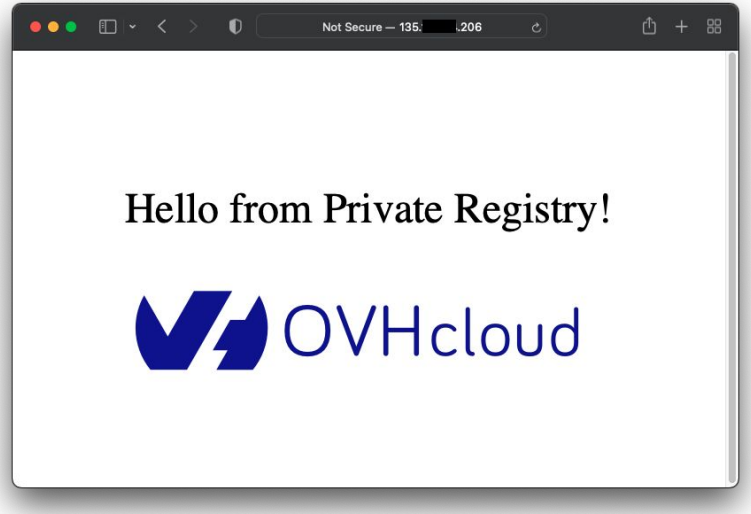
And the operator creates deployment and service

```
1.6669437496311843e+09  INFO    ✨ Creating a new Deployment    {"controller": "ovhnginx", "controllerGroup":
"tutorials.ovhcloud.com", "controllerKind": "OvhNginx", "OvhNginx": {"name":"ovhnginx-sample","namespace":"test-go-operator"},
"namespace": "test-go-operator", "name": "ovhnginx-sample", "reconcileID": "eb7fc1fa-b2c3-4762-847e-4895dc07c9a2",
"Deployment.Namespace": "test-go-operator", "Deployment.Name": "ovhnginx-sample"}
1.6669437496554422e+09  INFO    ✨ Creating a new Service       {"controller": "ovhnginx", "controllerGroup":
"tutorials.ovhcloud.com", "controllerKind": "OvhNginx", "OvhNginx": {"name":"ovhnginx-sample","namespace":"test-go-operator"},
"namespace": "test-go-operator", "name": "ovhnginx-sample", "reconcileID": "eb7fc1fa-b2c3-4762-847e-4895dc07c9a2",
"Service.Namespace": "test-go-operator", "Service.Name": "ovhnginx-sample"}
```


Let's test it!



```
horacio@ovhcloud ~/nginx-go-operator % kubectl get pod,svc -n test-go-operator
NAME                                   READY   STATUS    RESTARTS   AGE
pod/ovhnginx-sample-66c57d857f-7vqm2   1/1     Running   0          12m

NAME                       TYPE           CLUSTER-IP    EXTERNAL-IP      PORT(S)        AGE
service/ovhnginx-sample    LoadBalancer   10.3.78.218   51.210.253.158   80:30148/TCP   12m
```



That's all, folks!

Thank you all!

OVHcloud intel

@LostInBrittany