
DID YOU SAY KUBERNETES OPERATORS?
TELL ME MORE !!
makeameme.org

WHOAMI

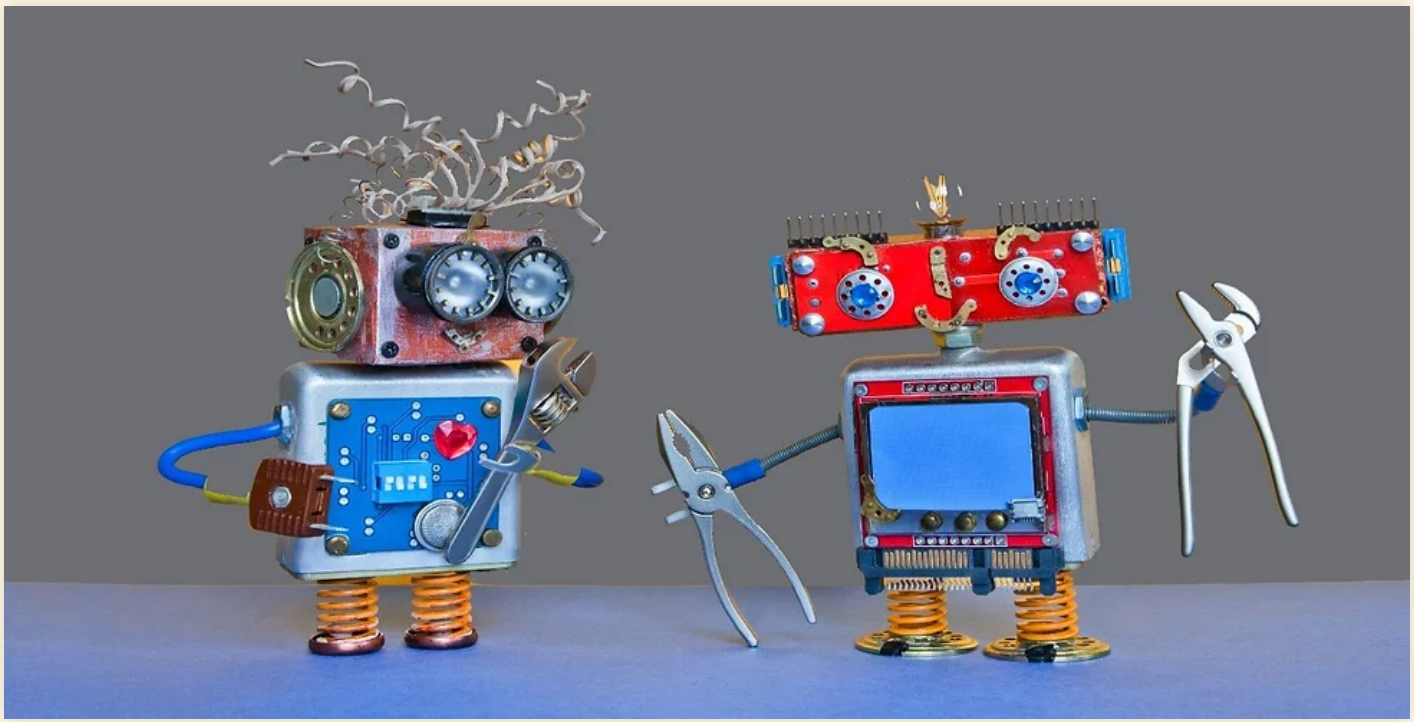
- Cloud Native Consultant
- You can find me as **mendrugory** on internet
- Enjoying a wonderful journey with **Clastix**
- Visit **www.adysof.com** to see professional stuffs

INTRODUCTION TO KUBERNETES OPERATORS

MANUAL MAINTENANCE

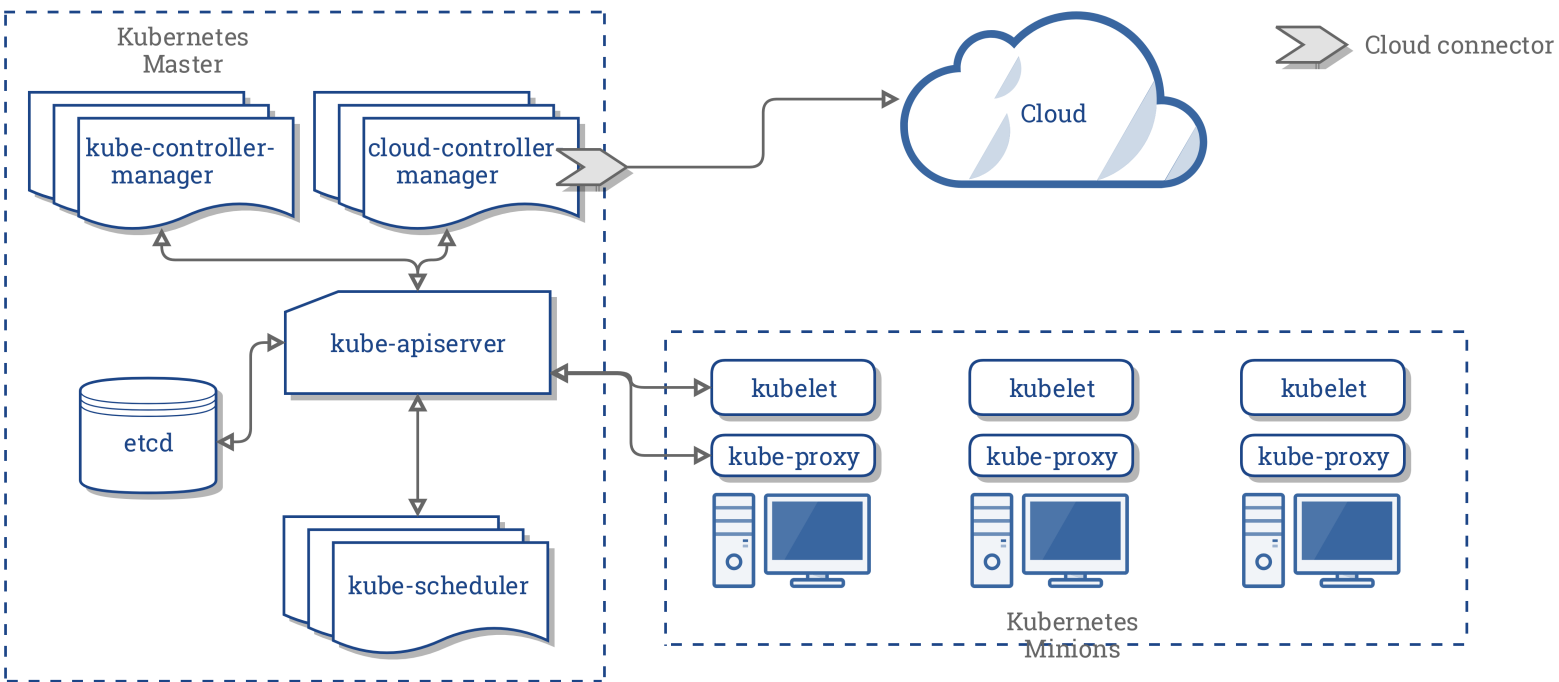

WHEN MAINTENANCE SAYS
ITS FIXED
38
makeameme.org

AUTOMATION



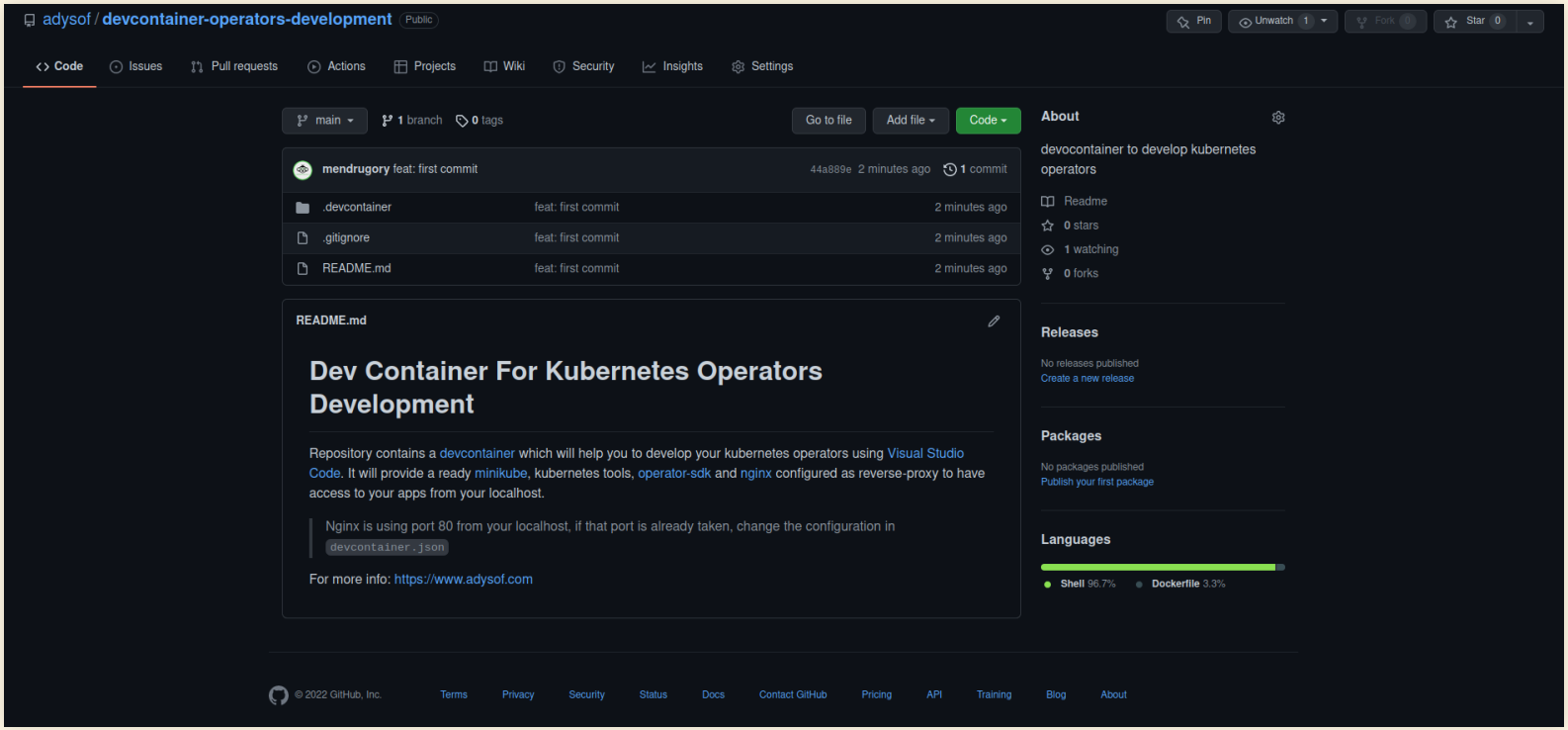

kubernetes

KUBERNETES ARCHITECTURE



LOCAL KUBERNETES

DEV CONTAINER

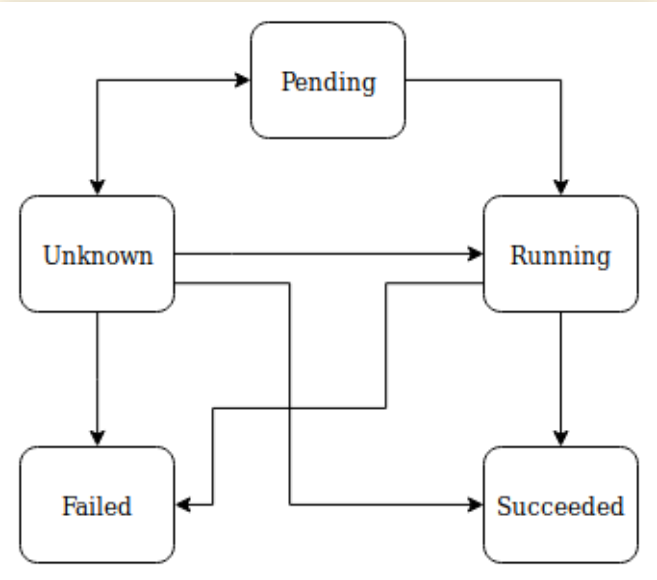


https://github.com/adysof/devcontainer-operators-development

KUBERNETES OPERATORS

DO YOU KNOW YOUR INFRA/APP?

LIFE CYCLE



Pod's life cycle example

RECONCILATION APPROACHES

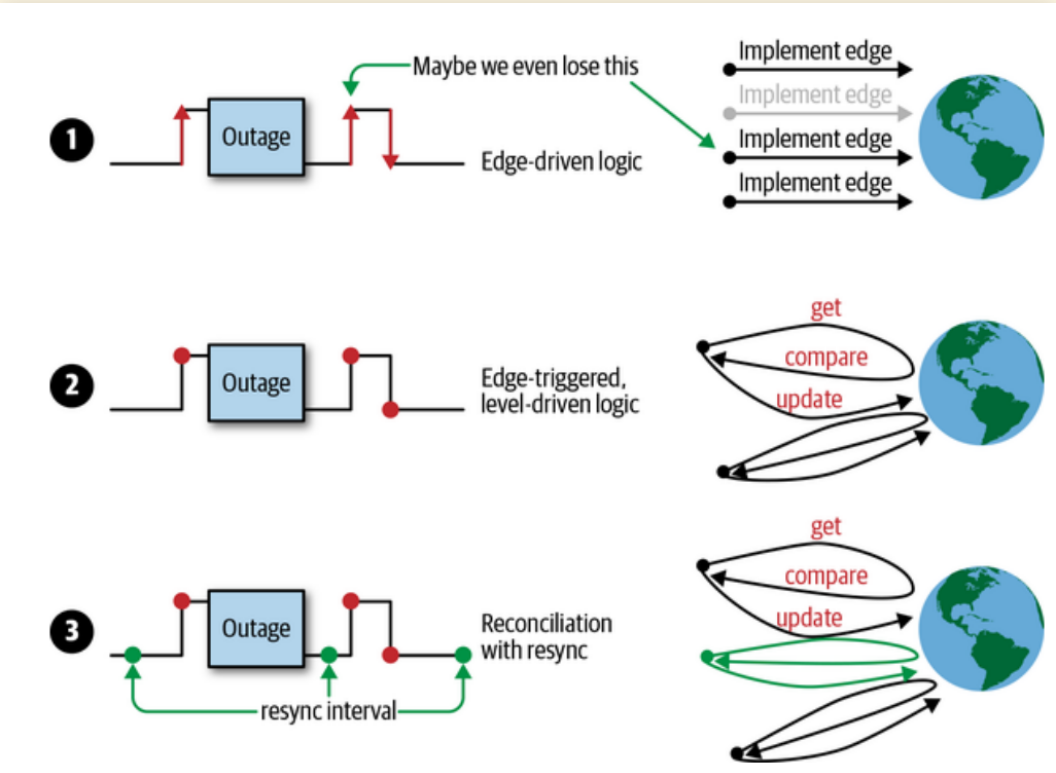


Image from Programming Kubernetes, O'Reilly Media

HOW IS MY INFRA/APP REPRESENTED?

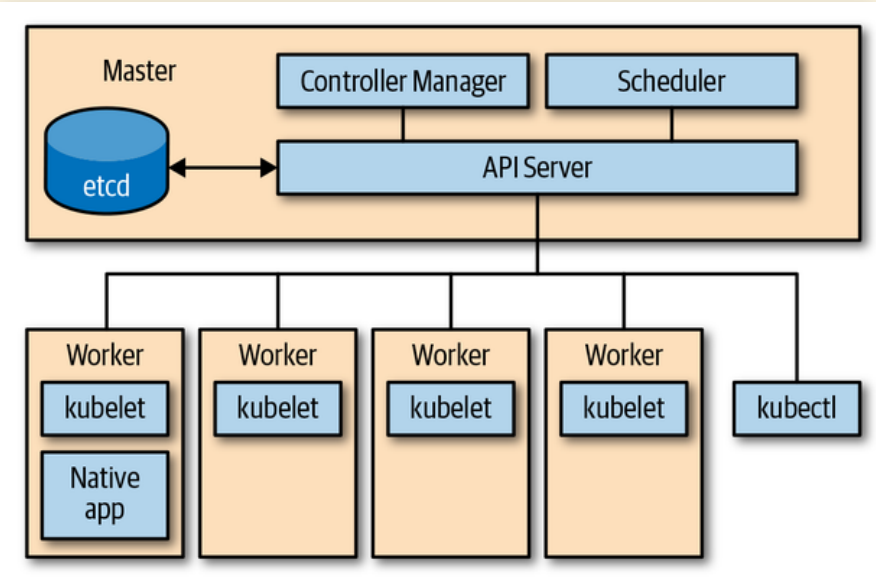
CRD

```yaml
apiVersion: operator.adysof.com/v1alpha1
kind: MyCRD
metadata:
    name: mycrd
    namespace: test
spec:
    replicas: 3
```

AM I ALONE?

KUBERNETES API

API SERVER


Master
Controller Manager
Scheduler
etcd
API Server
Worker
kubelet
Native app
Worker
kubelet
Worker
kubelet
Worker
kubelet
kubectl

LET'S CHECK IT OUT

CHECKING API

```
$ kubectl get ns -v=6
I0209 16:26:13.738059  569624 loader.go:372] Config loaded from f
I0209 16:26:13.748526  569624 round_trippers.go:454] GET https://
NAME                 STATUS   AGE
default              Active   76m
ingress-nginx        Active   76m
kube-node-lease      Active   76m
kube-public          Active   76m
kube-system          Active   76m
local-path-storage   Active   76m
```

EXPOSING API

```
$ kubectl proxy --port=5555
Starting to serve on 127.0.0.1:5555
```

KUBERNETES API

```
$ curl http://127.0.0.1:5555/api/v1/namespaces
```

CHECKING API

```
$ kubectl create ns test-kubectl -v=10
```

KUBERNETES API

```
$ curl --version
curl 7.68.0 (x86_64-pc-linux-gnu) libcurl/7.68.0 OpenSSL/1.1.1f z

$ curl -XPOST \
  -H "Accept: application/json, */*" \
  -H "Content-Type: application/json" \
  -H "User-Agent: curl 7.68.0" \
  'http://127.0.0.1:5555/api/v1/namespaces?fieldManager=curl-crea
  -d '{"kind":"Namespace","apiVersion":"v1","metadata":{"name":"t
```

KUBERNETES API RESOURCES

```
$ kubectl api-resources
```

KUBERNETES API RESOURCES

```
$ curl http://127.0.0.1:5555/api/v1/nodes
```

```
$ curl http://127.0.0.1:5555/api/v1/nodes/operators-control-plane
```

KUBERNETES API RESOURCES

```
$ curl http://127.0.0.1:5555/api/v1/namespaces/default/services
```

```
$ curl http://127.0.0.1:5555/api/v1/namespaces/default/services/k
```

API ACCESS CONTROL

AUTHENTICATION

AUTHORIZATION

YOU ARE ADMIN, RIGHT?

CAN I LIST PODS?

```
$ kubectl auth can-i list pods
yes
```

CAN I CREATE DEPLOYMENTS?

```
$ kubectl auth can-i create deployments
yes
```

CAN I DELETE SERVICES?

```
$ kubectl auth can-i delete svc
yes
```

HOW TO CREATE A SERVICE ACCOUNT

```
$ kubectl create sa my-sa --dry-run=client -o yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  creationTimestamp: null
  name: my-sa
```

HOW TO CREATE A SERVICE ACCOUNT

```
$ kubectl create sa my-sa
```

CAN SA LIST PODS?

```
$ kubectl auth can-i list pods \
  --as system:serviceaccount:default:my-sa
no
```

CAN SA CREATE DEPLOYMENTS?

```
$ kubectl auth can-i create deployments \
  --as system:serviceaccount:default:my-sa
no
```

CAN SA DELETE SERVICES?

```
$ kubectl auth can-i delete svc \
--as system:serviceaccount:default:my-sa
no
```

CREATE A ROLE

```
$ kubectl create role my-sa --resource=pods --verb=list,get --dry
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  creationTimestamp: null
  name: my-sa
rules:
- apiGroups:
  - ""
  resources:
  - pods
  verbs:
  - list
  - get
```

CREATE A ROLE

```
$ kubectl create role my-sa --resource=pods --verb=list,get
role.rbac.authorization.k8s.io/my-sa created
```

CREATE A ROLEBINDING

```
$ kubectl create rolebinding my-sa --role=my-sa --serviceaccount=
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  creationTimestamp: null
  name: my-sa
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: my-sa
subjects:
- kind: ServiceAccount
  name: my-sa
  namespace: default
```

CREATE A ROLEBINDING

```
$ kubectl create rolebinding my-sa --role=my-sa --serviceaccount=
rolebinding.rbac.authorization.k8s.io/my-sa created
```

CAN SA GET PODS?

```
$ kubectl auth can-i get pods \
  --as system:serviceaccount:default:my-sa
yes
```

CAN SA DELETE PODS?

```
$ kubectl auth can-i delete pods \
  --as system:serviceaccount:default:my-sa
no
```

CHECK SA SECRET

```
$ kubectl get sa my-sa -o yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  creationTimestamp: "2022-02-11T15:28:43Z"
  name: my-sa
  namespace: default
  resourceVersion: "570"
  uid: 7af3e7c1-2f8b-4256-b5db-43b0699b931a
secrets:
- name: my-sa-token-jzntp
```

CHECK SA SECRET

```
$ kubectl get secret my-sa-token-jzntp -o yaml
```

EXPORT TOKEN

```
$ TOKEN=$(kubectl get secret my-sa-token-jzntp -o json \
  | jq -r ".data.token" \
  | base64 -d)
```

REST API SA

REST API SA

```
$ curl -k \
  -H "Authorization: Bearer $TOKEN" \
  https://127.0.0.1:33257/api/v1/namespaces/default/pods

$ curl -k \
  -H "Authorization: Bearer $TOKEN" \
  https://127.0.0.1:33257/apis/apps/v1/namespaces/default/deploym
```

OPERATOR FRAMEWORK SDK

DOWNLOAD FROM GITHUB

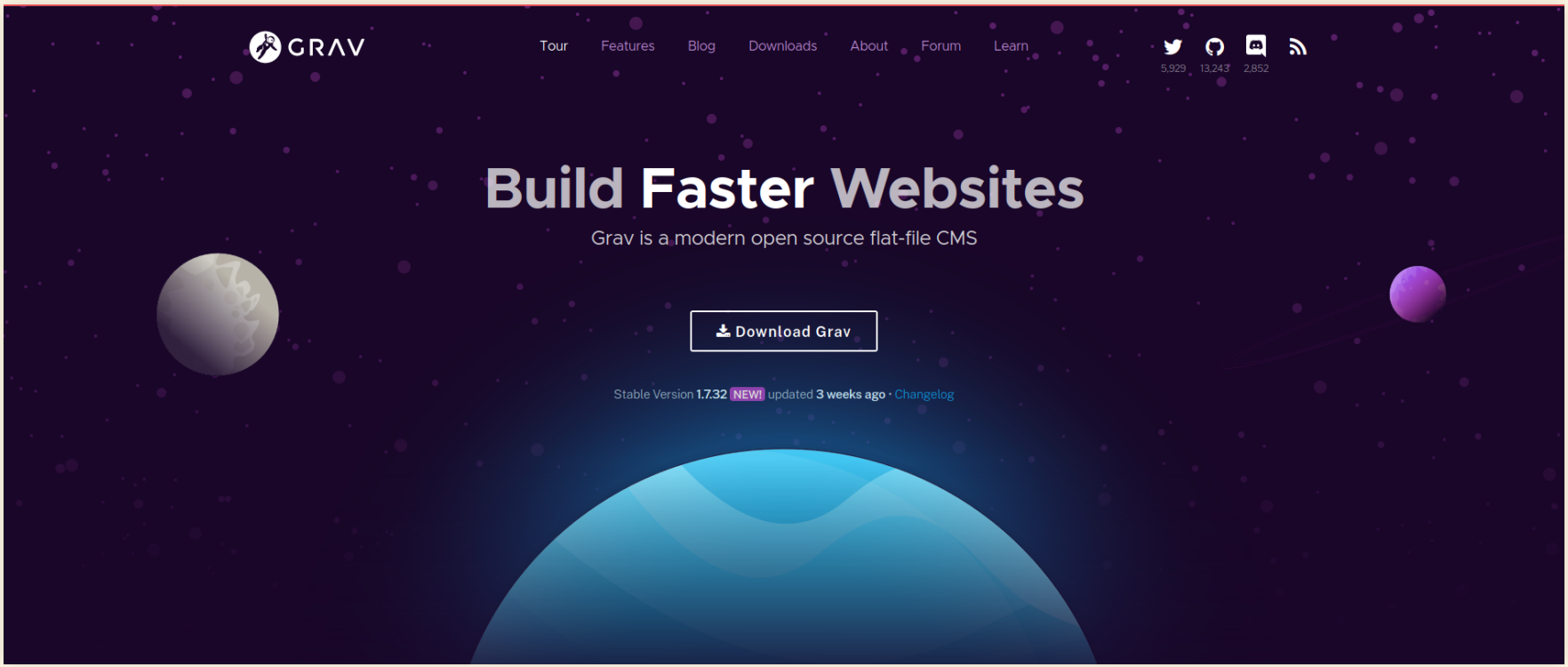
```
## Linux by default
$ export ARCH=$(case $(uname -m) in x86_64) echo -n amd64 ;; aarc
$ export OS=$(uname | awk '{print tolower($0)}')
$ export VERSION=v1.17.0
$ export OPERATOR_SDK_DL_URL=https://github.com/operator-framewor
$ curl -LO ${OPERATOR_SDK_DL_URL}/operator-sdk_${OS}_${ARCH}
$ chmod +x operator-sdk_${OS}_${ARCH}
$ sudo mv operator-sdk_${OS}_${ARCH} /usr/local/bin/operator-sdk
```

OPERATOR SDK

```
$ operator-sdk --help
```

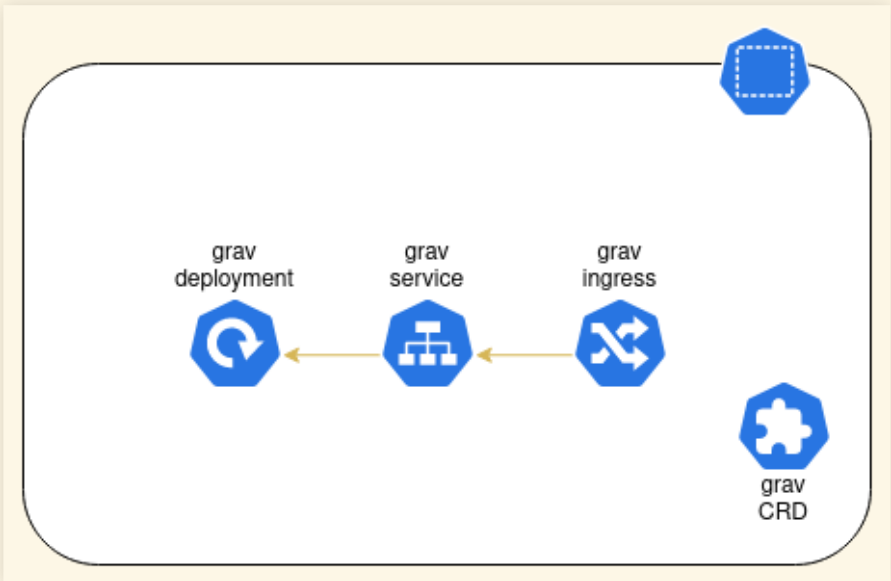
GRAV AS A SERVICE

GRAV


GRAV
Tour
Features
Blog
Downloads
About
Forum
Learn
5,929
13,243
2,852
Build Faster Websites
Grav is a modern open source flat-file CMS
Download Grav
Stable Version 1.7.32 NEW! updated 3 weeks ago · Changelog

HOW TO ARCHITECT A VIRTUAL GRAV INFRASTRUCTURE IN KUBERNETES?

KUBERNETES ARCHITECTURE



LET'S CODE

OPERATOR SDK

```
$ operator-sdk init \
  --domain=adysof.com \
  --repo=github.com/adysof/course-operator
Writing kustomize manifests for you to edit...
Writing scaffold for you to edit...
Get controller runtime:
$ go get sigs.k8s.io/controller-runtime@v0.11.0
go: downloading golang.org/x/net v0.0.0-20210825183410-e898025ed9
go: downloading golang.org/x/oauth2 v0.0.0-20210819190943-2bc19b1
go: downloading golang.org/x/sys v0.0.0-20211029165221-6e7872819d
Update dependencies:
$ go mod tidy
go: downloading cloud.google.com/go v0.81.0
go: downloading golang.org/x/crypto v0.0.0-20210817164053-32db794
Next: define a resource with:
$ operator-sdk create api
```

OPERATOR SDK

```
$ operator-sdk create api \
  --group operator \
  --version v1alpha1 \
  --kind Grav \
  --resource=true \
  --controller=true
```

OPERATOR SDK

```
$ make generate
```

OPERATOR SDK

```
$ make manifests
```

API

```go
package v1alpha1

import (
    appsv1 "k8s.io/api/apps/v1"
    corev1 "k8s.io/api/core/v1"
    networkingv1 "k8s.io/api/networking/v1"
    metav1 "k8s.io/apimachinery/pkg/apis/meta/v1"
)

// GravSpec defines the desired state of Grav
type GravSpec struct {
    Domain string `json:"domain"`
}

// GravStatus defines the observed state of Grav
type GravStatus struct {
```

INSTALL

```
$ make generate && make manifests && make install
```

CHECK NEW CRD

```
$ kubectl api-resources | grep grav
gravs    operator.adysof.com/v1alpha1    true    Grav
```

RESOURCE

./config/samples/operator_v1alpha1_grav.yaml

```
apiVersion: operator.adysof.com/v1alpha1
kind: Grav
metadata:
  name: mygrav
spec:
  domain: www.mygrav.com
```

RESOURCE

```
$ kubectl apply -f ./config/samples/operator_v1alpha1_grav.yaml
grav.operator.adysof.com/mygrav created
```

RESOURCE

```
$ kubectl get gravs.operator.adysof.com
NAME     AGE
mygrav   24s
```

That's just an API REST saving data into ETCD

RBAC

```
//+kubebuilder:rbac:groups=operator.adysof.com,resources=gravs,ve
//+kubebuilder:rbac:groups=operator.adysof.com,resources=gravs/st
//+kubebuilder:rbac:groups=operator.adysof.com,resources=gravs/fi
//+kubebuilder:rbac:groups=apps,resources=deployments,verbs=get;l
//+kubebuilder:rbac:groups=core,resources=services,verbs=get;list
//+kubebuilder:rbac:groups=*,resources=ingresses,verbs=get;list;w
```

RBAC

```
$ make manifests
```

Check ./config/rbac/role.yaml

OPERATOR

```
func (r *WordpressReconciler) Reconcile(ctx context.Context, req
  _ = log.FromContext(ctx)

  grav := &operatorv1alpha1.Grav{}
  if err := r.Client.Get(ctx, req.NamespacedName, grav); err != n
    if k8serrors.IsNotFound(err) {
      return ctrl.Result{}, nil
    }
    return ctrl.Result{}, err
  }

  return ctrl.Result{}, nil
}
```

DEBUG OPERATOR

```
{
    "version": "0.2.0",
    "configurations": [
        {
            "name": "Debug",
            "type": "go",
            "request": "launch",
            "mode": "auto",
            "program": "${workspaceFolder}",
            "args": [
                "--metrics-bind-address=:7085",
                "--health-probe-bind-address=:7086",
            ],
            "env": {
            }
        }
```

WATCHING RESOURCES

```
// SetupWithManager sets up the controller with the Manager.
func (r *GravReconciler) SetupWithManager(mgr ctrl.Manager) error
    return ctrl.NewControllerManagedBy(mgr).
        For(&operatorv1alpha1.Grav{}).
        Owns(&appsv1.Deployment{}).
        Owns(&corev1.Service{}).
        Owns(&networkingv1.Ingress{}).
        Complete(r)
}
```

DEPLOYMENT

DEPLOYMENT

```
+    deployment := &appsv1.Deployment{
+        ObjectMeta: v1.ObjectMeta{
+            Name:        grav.GetName(),
+            Namespace:    grav.GetNamespace(),
+        },
+    }
```

MUTATE DEPLOYMENT

```
+func mutateDeployment(ctx context.Context, deployment *appsv1.De
+    return func() error {
+        deployment.Spec.Template.ObjectMeta = metav1.ObjectMeta{
+            Labels: map[string]string{
+                "app": grav.GetName(),
+            },
+        }
+        deployment.Spec.Selector = &metav1.LabelSelector{
+            MatchLabels: map[string]string{
+                "app": grav.GetName(),
+            },
+        }
+        deployment.Spec.Template.Spec.Containers = []corev1.Cont
+            {
+                Name:                    "grav",
+                Image:                   "advsof/grav"
```

DEPLOYMENT

```
+    result, err := controllerutil.CreateOrUpdate(ctx, r.Client,
+}
```

UPDATE STATUS

```
+func (r *GravReconciler) updateStatus(ctx context.Context, names
+    grav := &operatorv1alpha1.Grav{}
+
+    if err := r.Client.Get(ctx, namespacedName, grav); err != ni
+        return err
+    }
+
+    update(grav)
+
+    if err := r.Status().Update(ctx, grav); err != nil {
+        return fmt.Errorf("error updating grav status: %s", err)
+    }
+
+    return nil
+}
```

DEPLOYMENT

```
+    if err != nil {
+        return ctrl.Result{}, err
+    }
+    if result != controllerutil.OperationResultNone {
+        if err := r.updateStatus(ctx, req.NamespacedName, func(g
+            grav.Status.Deployment.Name = deployment.GetName()
+            grav.Status.Deployment.DeploymentStatus = deployment
+        }); err != nil {
+            return ctrl.Result{}, err
+        }
+
+        logger.Info(fmt.Sprintf("%s deployment has been configur
+        return ctrl.Result{}, nil
+    }
+}
```

MUTATE SERVICE

```
+func mutateService(ctx context.Context, service *corev1.Service,
+    return func() error {
+        service.ObjectMeta.Labels = map[string]string{
+            "app": grav.GetName(),
+        }
+        service.Spec.Selector = map[string]string{
+            "app": grav.GetName(),
+        }
+
+        service.Spec.Ports = []corev1.ServicePort{
+            {
+                Protocol:   corev1.ProtocolTCP,
+                Port:       80,
+                TargetPort: intstr.FromInt(80),
+            },
+        }
```

SERVICE

```
+    // Service
+    service := &corev1.Service{
+        ObjectMeta: v1.ObjectMeta{
+            Name:      grav.GetName(),
+            Namespace: grav.GetNamespace(),
+        },
+    }
+
+    result, err = controllerutil.CreateOrUpdate(ctx, r.Client, s
+    if err != nil {
+        return ctrl.Result{}, err
+    }
+    if result != controllerutil.OperationResultNone {
+        if err := r.updateStatus(ctx, req.NamespacedName, func(g
+            grav.Status.Service.Name = service.GetName()
+            grav.Status.Service.ServiceStatus = service.Status
```

MUTATE INGRESS

```
+func mutateIngress(ctx context.Context, ingress *networkingv1.In
+    return func() error {
+        ingress.ObjectMeta.Labels = map[string]string{
+            "app": grav.GetName(),
+        }
+
+        pathType := networkingv1.PathTypePrefix
+
+        ingress.Spec.Rules = []networkingv1.IngressRule{
+            {
+                Host: grav.Spec.Domain,
+                IngressRuleValue: networkingv1.IngressRuleValue{
+                    HTTP: &networkingv1.HTTPIngressRuleValue{
+                        Paths: []networkingv1.HTTPIngressPath{
+                            {
+                                Path:     "/"
```

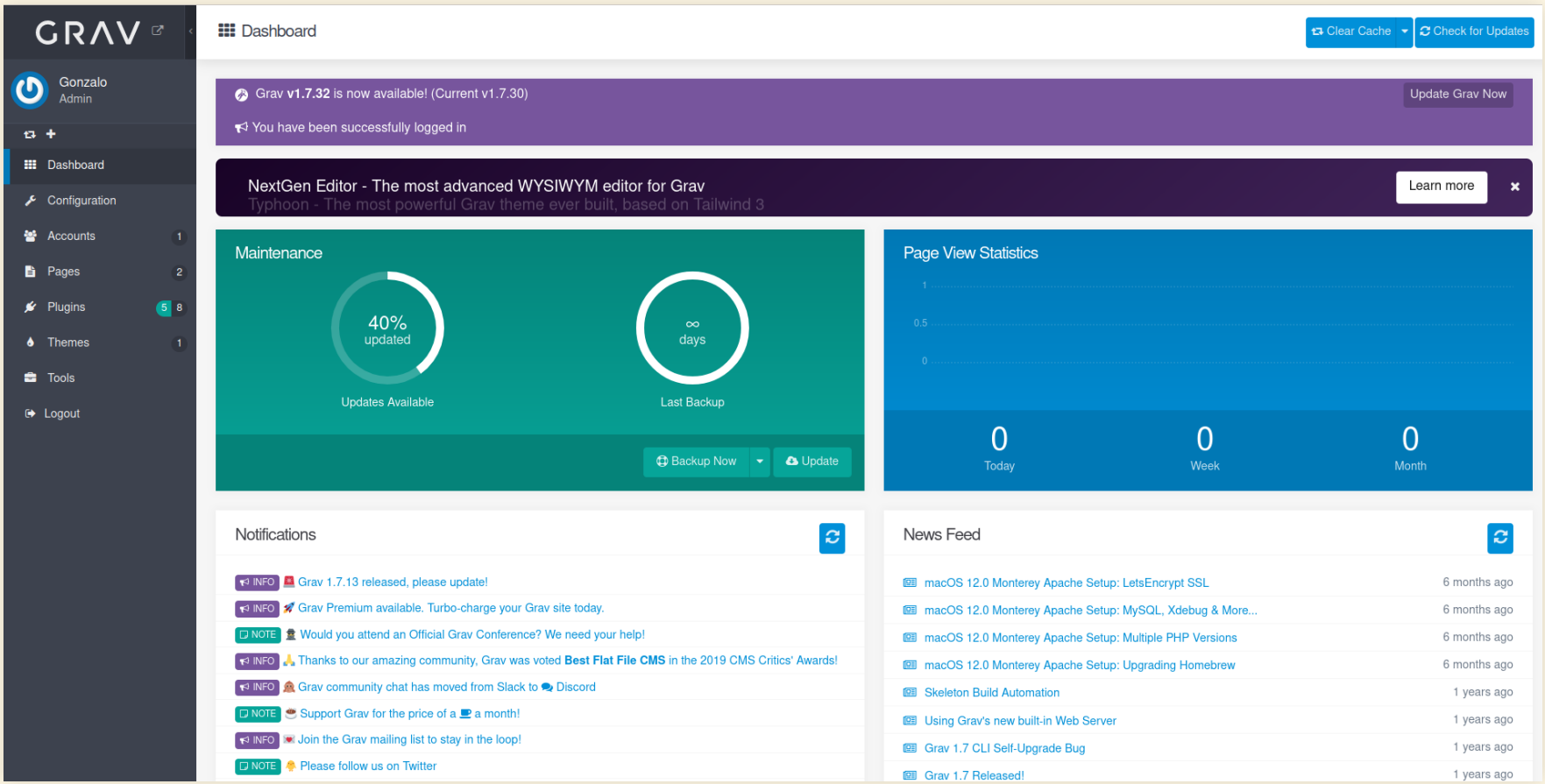
INGRESS

```
+    // Ingress
+    ingress := &networkingv1.Ingress{
+        ObjectMeta: v1.ObjectMeta{
+            Name:      grav.GetName(),
+            Namespace: grav.GetNamespace(),
+        },
+    }
+
+    result, err = controllerutil.CreateOrUpdate(ctx, r.Client, i
+    if err != nil {
+        return ctrl.Result{}, err
+    }
+    if result != controllerutil.OperationResultNone {
+        if err := r.updateStatus(ctx, req.NamespacedName, func(g
+            grav.Status.Ingress.Name = ingress.GetName()
+            grav.Status.Ingress.IngressStatus = ingress.Status
```

REGISTERING DOMAIN

```
$ echo "127.0.0.1    www.mygrav.com" >> /etc/hosts
```

GRAV


GRAV
Dashboard
Clear Cache
Check for Updates
Gonzalo
Admin
Grav v1.7.32 is now available! (Current v1.7.30)
Update Grav Now
You have been successfully logged in
Dashboard
Configuration
Accounts 1
Pages 2
Plugins 5 8
Themes 1
Tools
Logout
NextGen Editor - The most advanced WYSIWYM editor for Grav
Typhoon - The most powerful Grav theme ever built, based on Tailwind 3
Learn more
Maintenance
40%
updated
Updates Available
∞
days
Last Backup
Backup Now
Update
Page View Statistics
1
0.5
0
0
Today
0
Week
0
Month
Notifications
INFO Grav 1.7.13 released, please update!
INFO Grav Premium available. Turbo-charge your Grav site today.
NOTE Would you attend an Official Grav Conference? We need your help!
INFO Thanks to our amazing community, Grav was voted Best Flat File CMS in the 2019 CMS Critics' Awards!
INFO Grav community chat has moved from Slack to Discord
NOTE Support Grav for the price of a a month!
INFO Join the Grav mailing list to stay in the loop!
NOTE Please follow us on Twitter
News Feed
macOS 12.0 Monterey Apache Setup: LetsEncrypt SSL 6 months ago
macOS 12.0 Monterey Apache Setup: MySQL, Xdebug & More... 6 months ago
macOS 12.0 Monterey Apache Setup: Multiple PHP Versions 6 months ago
macOS 12.0 Monterey Apache Setup: Upgrading Homebrew 6 months ago
Skeleton Build Automation 1 years ago
Using Grav's new built-in Web Server 1 years ago
Grav 1.7 CLI Self-Upgrade Bug 1 years ago
Grav 1.7 Released! 1 years ago

RBAC

```
$ kubectl create role gravadmin \
  --resource=gravs \
  --verb=list,get,create,delete,update,patch

$ kubectl create rolebinding my-sa-gravadmin \
 --role=gravadmin \
 --serviceaccount=default:my-sa
```

```
$ curl -XPOST -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" -k \
  https://127.0.0.1:46221/apis/operator.adysof.com/v1alpha1/names
  -d '{"apiVersion":"operator.adysof.com/v1alpha1","kind":"Grav",
```

NEXT STEPS

- SW Engineering
- Storage
- HA
- DNS Record Registration
- ...
- Whatever you can imaging

Q&A