WIGM and Why You Want an Operator

Carson Anderson

Weave

@carsonoid

@carson_ops

What is an Operator?

- An "Operator" is any [illegible]es a Kubernetes system and seeks a desir[illegible]
- They are often u[illegible]e databas[illegible]ther software in Kubernetes
- They represent[illegible]manage[illegible]art of [illegible]naged service which runs insid[illegible]ubernete[illegible]
- They take all the [illegible]ional know[illegible]e managed Software and put it into code

A management pattern

Why Use One?

A "Practical" Example

Wasteful
Individual
GIF
Mirror

What WIGM Does

- Download a GIF from a given URL
- Store the GIF locally
- Generate a web page to host the GIF with a given title
- Serve the web page

Management Goals

- Exceedingly easy to...
  - Host many instances with many configurations
  - Update configurations
  - Report on configurations
- No dedicated container images

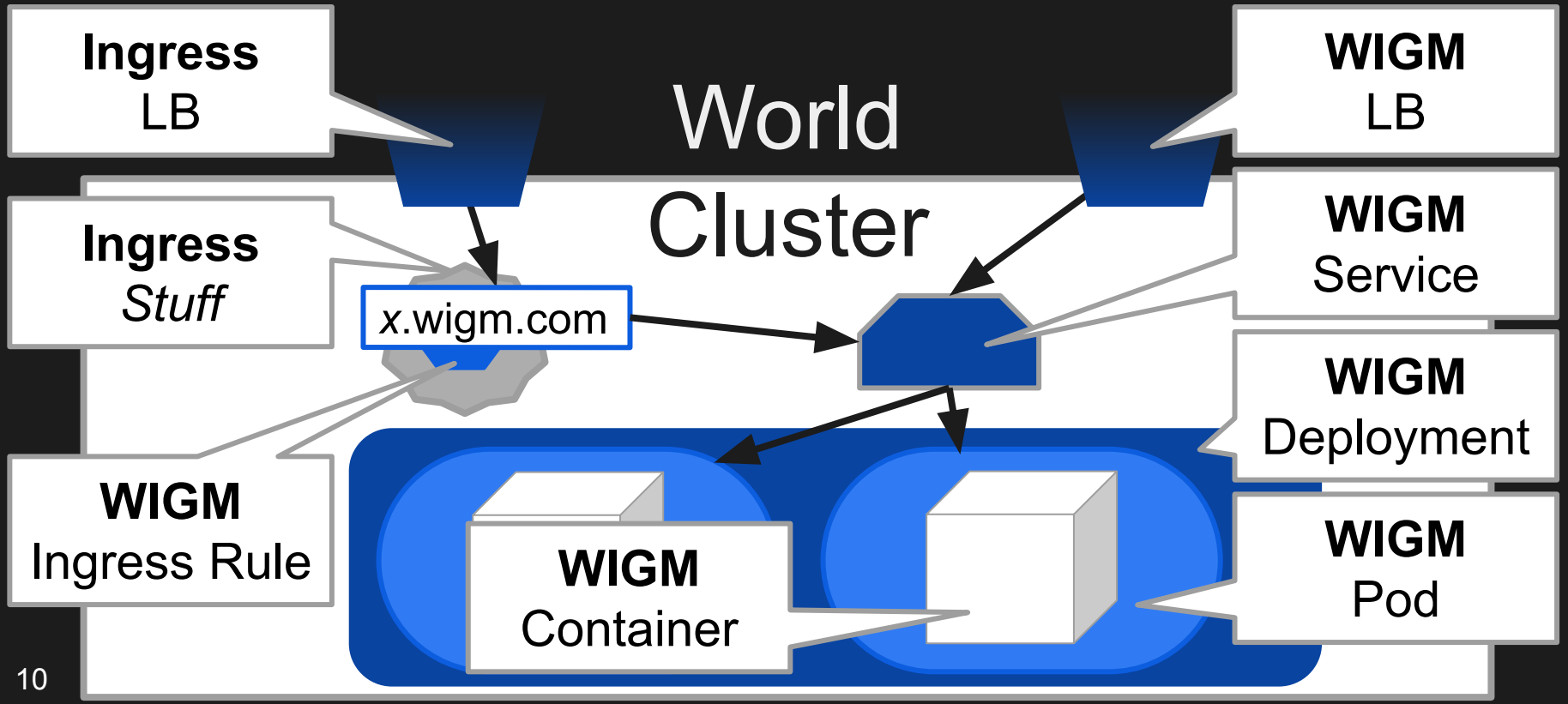
WIGM: Configuration

- **Name** - Kubernetes label value and resource name
- ENV variables:
  - **GIF_SOURCE_LINK** - Used to fetch original GIF
  - **GIF_TITLE** - Used as page title - HTML Friendly
- Access Decisions:
  - **Create** dedicated load balancer?
  - **Create** ingress rule?

WIGM: Outputs

- Kubernetes resources
  - Deployment
  - Service
  - Ingress rule (Optional)
- Dedicated Load balancer (Optional)

WIGM: Outputs Explained



WIGM Server

Pure Upstream NGINX

WIGM in Bash

Entrypoint for:
nginx:1.5-alpine

```
## Expects: GIF_SOURCE_LINK, GIF_NAME
apk update && apk add curl

# go to data dir
cd /usr/share/nginx/html

# fetch original
curl -Lo wigm.gif "$GIF_SOURCE_LINK"

# write page using heredoc
cat > index.html <<EOF
<head>
  <title>WIGM: $GIF_TITLE</title>
</head>
<body>
  <h1>WIGM: $GIF_TITLE</h1>
  <img src="wigm.gif" />
</body>
EOF

# start nginx
exec nginx -g "daemon off;"
```

Management Tools

1. YAML
2. Helm
3. Operators
   - Metacontroller (Webhooks)

     KOPF (Python)
   - Operator SDK (Golang)

Evaluation Tasks

1. Release an instance of WIGM
    a. Create a LoadBalancer
    b. Disable Ingress
2. Update the GIF title
3. Enable ingress
4. Report Configuration
5. Delete

Evaluation Task Tracker

```
----CREATE----
Yes LoadBalancer
No  Ingress
```

```
----UPDATE----
Change Title
```

```
----UPDATE----
Enable Ingress
```

```
----REPORT----
```

```
----DELETE----
```

WIGM Via YAML

YAML Code Structure

```
./wigm/yaml/
├── resources.yaml
└── releases
    ├── yaml1.yaml
    └── yaml2.yaml
```

YAML

resources.yaml Deployment Section

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wigm-host-NAME
  labels:
    app: wigm
    gif: NAME
spec:
  replicas: 2
  selector:
    matchLabels:
      app: wigm
      gif: NAME
  template:
    metadata:
      labels:
        app: wigm
        gif: NAME
    spec:
      [...]
```

YAML

resources.yaml
Deployment
Section
(Continued)

```
apiVersion: apps/v1
kind: Deployment
metadata:
  [...]
spec:
  [...]
  template:
    [...]
    spec:
      containers:
      - name: gifhost
        image: nginx:1.15-alpine
        ports:
        - containerPort: 80
        env:
        - name: GIF_TITLE
          value: TITLE
        - name: GIF_SOURCE_LINK
          value: LINK
        command:
          [...]
```

YAML

resources.yaml
Deployment
Section
(Continued)


```
[...]
    spec:
      containers:
      - name: gifhost
      [...]
        command:
        - sh
        - -exc
        - |

          # go to data dir
          [...]

          # fetch original
          [...]

          # write page using heredoc
          [...]

          # start nginx
          exec nginx -g "daemon off;"
```

YAML

resources.yaml
Service Section

```
---
kind: Service
apiVersion: v1
metadata:
  name: wigm-host-NAME
  labels:
    app: wigm
    gif: NAME
spec:
  selector:
    app: wigm
    gif: NAME
  # Uncomment line below to request
  # a cloud LB for the service
  # type: LoadBalancer
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
```

YAML

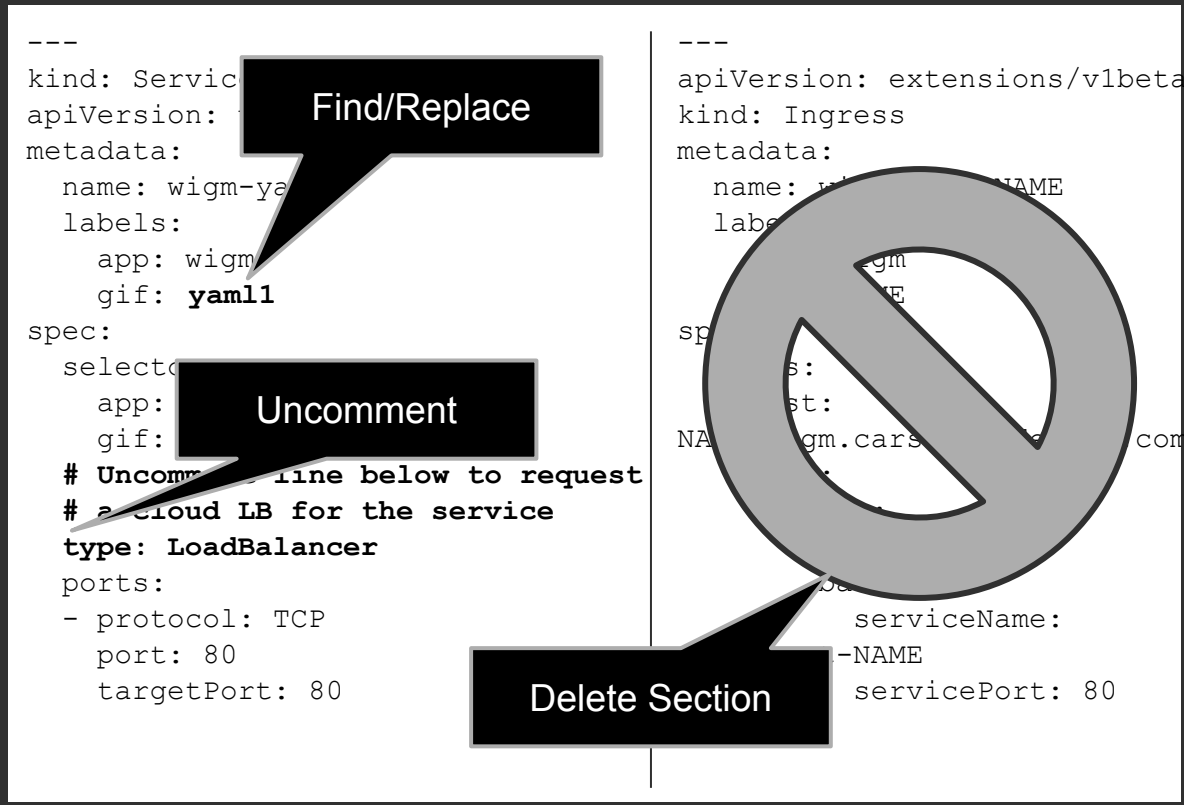
resources.yaml
Ingress Section

```
# Delete or comment to disable ingress
---
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: wigm-host-NAME
  labels:
    app: wigm
    gif: NAME
spec:
  rules:
  - host: NAME.wigm.carson-anderson.com
    http:
      paths:
      - path: /
        backend:
          serviceName: wigm-host-NAME
          servicePort: 80
```

YAML Evaluation

----CREATE----	----UPDATE----	----UPDATE----	----REPORT----	----DELETE----
Yes LoadBalancer No Ingress	Change Title	Enable Ingress		

- Copy resources.yml
- Paste to release folder
- Find/Replace
  - NAME -> yaml1
  - TITLE -> yaml1
  - LINK -> https://…
- Uncomment LB Line
- Delete Ingress YAML
- "kubectl create -f …"



- Edit YAML
- "kubectl apply"

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wigm-host-yamlrelease
  labels:
    app: wigm
    gif: yamlrelease
spec:
  [...]
    spec:
     [...]
       env:
        - name: GIF_TITLE
          value: "This is YAML!"
        - name: GIF_SOURCE_LINK
          value: https://media.giphy.com/media/JloRBVCgTP6RW/giphy.gif
```

----CREATE----
Yes LoadBalancer
No Ingress

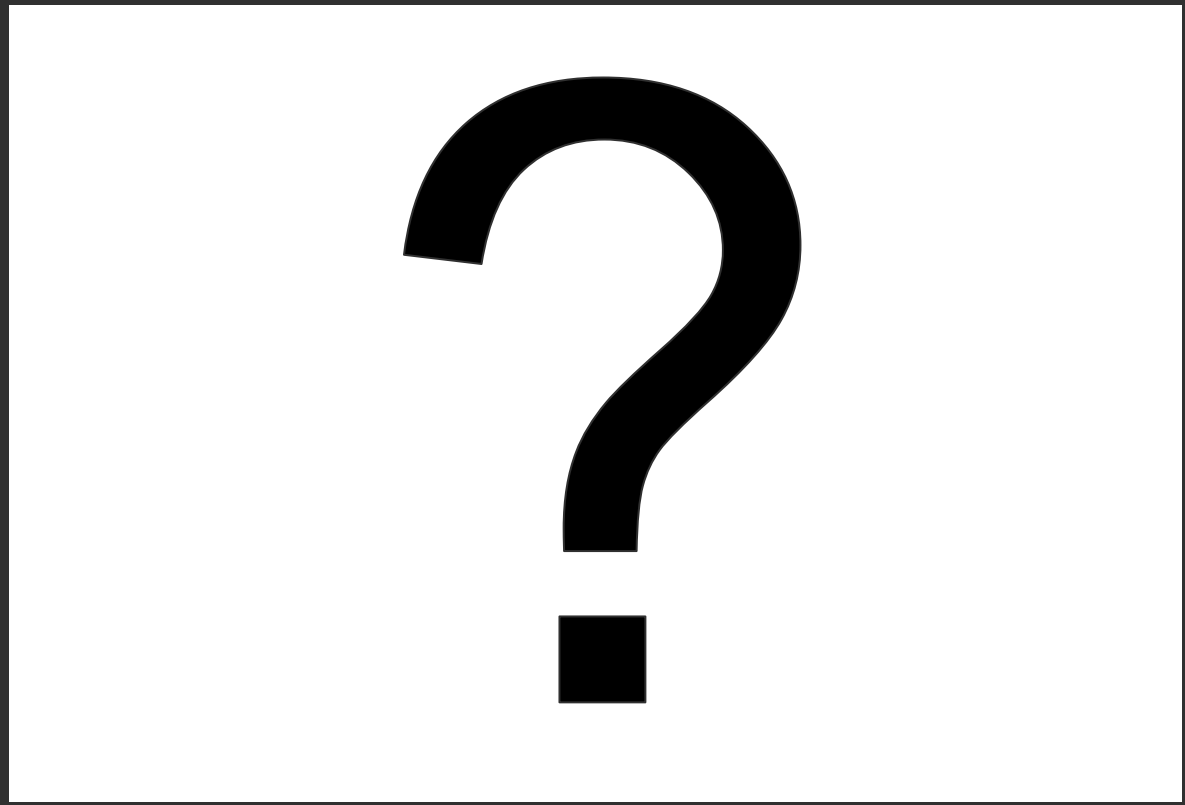
----UPDATE----
Change Title

----UPDATE----
Enable Ingress

----REPORT----

----DELETE----

- Uhh...
- Copy...
- That thing...



$ kubectl get -f releases/yaml1.yaml

```
NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/wigm-host-yaml1     2/2     2            0           7s

NAME                          TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
service/wigm-host-yaml1       ClusterIP   10.43.140.191   <none>        80/TCP    7s

NAME                                    HOSTS                                ADDRESS   PORTS   AGE
ingress.extensions/wigm-host-yaml1      yaml1.wigm.carson-anderson.com               80      7s
```

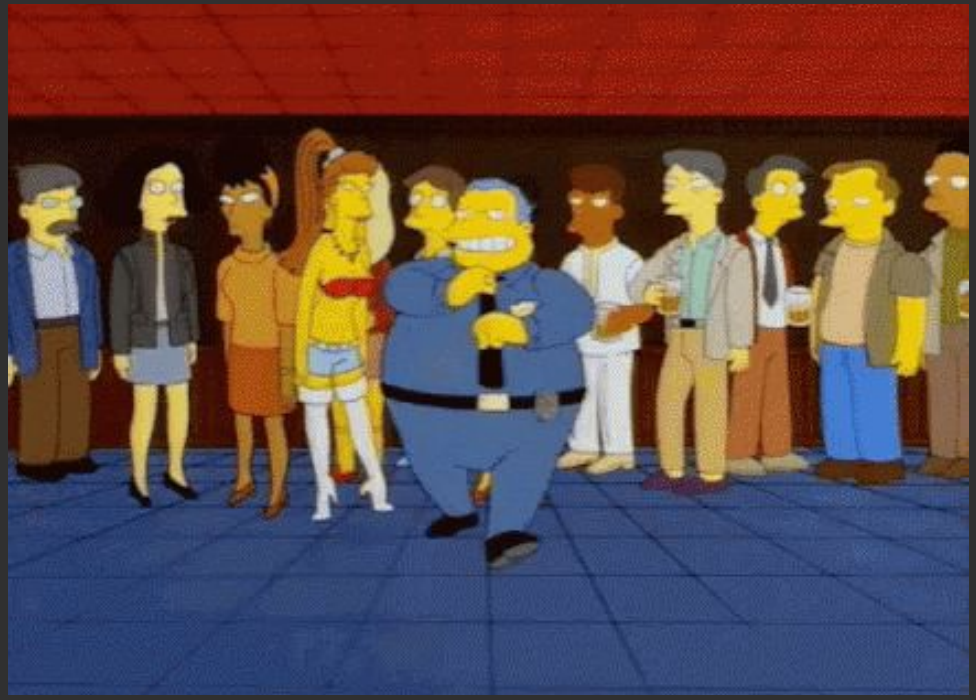
```
$ kubectl delete -f releases/yaml1.yaml

deployment.apps "wigm-host-yaml1" deleted
service "wigm-host-yaml1" deleted
ingress.extensions "wigm-host-yaml1" deleted
```

Summary

WIGM Via YAML

WIGM Via

Helm



Helm Code Structure

```
./wigm/helm/
├── Chart.yaml
├── values.yaml
├── templates
│   ├── deployment.yaml
│   ├── ingress.yaml
│   └── service.yaml
└── releases
    ├── helm1.yaml
    └── helm1.yaml
```

Helm

values.yaml

```
# Default values for wigm.
# This is a YAML-formatted file.
# Declare variables to be passed into
your templates.

gif:
  title: "" # Defaults to helm stack name
  link: "" # Required!

service:
  create_cloud_lb: false

ingress:
  enabled: true
```

Helm

templates/
deployment.yaml

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wigm-host-{{ .Release.Name }}
  labels:
    app: wigm
    gif: {{ .Release.Name }}
spec:
    [...]
    spec:
      containers:
      - name: gifhost
        [...]
        env:
       - name: GIF_TITLE
          {{- if .Values.gif.title}}
          value: {{ .Values.gif.title| quote  }}
          {{- else }}
          value: {{ .Release.Name }}
          {{- end }}
        - name: GIF_SOURCE_LINK
          value: {{ .Values.gif.link }}
```

Helm

templates/
service.yaml

```
---
kind: Service
apiVersion: v1
metadata:
  name: wigm-host-{{ .Release.Name }}
  labels:
    app: wigm
    gif: {{ .Release.Name }}
spec:
  selector:
    app: wigm
    gif: {{ .Release.Name }}
  {{ if .Values.service.create_cloud_lb }}
  type: LoadBalancer
  {{ end }}
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
```

Helm

templates/ ingress.yaml

```
{{ if .Values.ingress.enabled }}
---
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: wigm-host-{{ .Release.Name }}
  labels:
    app: wigm
    gif: {{ .Release.Name }}
spec:
  rules:
  - host: {{ .Release.Name }}.wigm.carson-anderson.com
    http:
      paths:
      - path: /
        backend:
          serviceName: wigm-host-{{ .Release.Name }}
          servicePort: 80
{{ end }}
```

Helm Evaluation

----CREATE----	----UPDATE----	----UPDATE----	----REPORT----	----DELETE----
Yes LoadBalancer No Ingress	Change Title	Enable Ingress		

- Create Values File

```
cat > releases/helm1.yaml <<EOF
gif:
  title: "helm1"
  link:  "https://media.giphy.com/media/wa6hNG157vUA25ncAu/giphy.gif"

service:
  create_cloud_lb: true

ingress:
  enabled: false
EOF
```

- Run Helm Install

```
helm install \
  -n helm1 \
  -f releases/helm1.yaml .
```

- Edit Values File

```
cat > releases/helm1.yaml <<EOF
gif:
  title: "Helm is better!"
  link:  "https://media.giphy.com/media/wa6hNG157vUA25ncAu/giphy.gif"

service:
  create_cloud_lb: true

ingress:
  enabled: false
EOF
```

- Run Helm Upgrade

```
helm upgrade \
  helm1 \
  -f releases/helm1.yaml .
```

- Edit Values File

```
cat > releases/helm1.yaml <<EOF
gif:
  title: "Helm is better!"
  link:  "https://media.giphy.com/media/wa6hNG157vUA25ncAu/giphy.gif"

service:
  create_cloud_lb: true

ingress:
  enabled: true
EOF
```

- Run Helm Upgrade

```
helm upgrade \
  helm1 \
  -f releases/helm1.yaml .
```

$ helm status helm1

```
LAST DEPLOYED: Thu Oct 31 01:09:59 2019
NAMESPACE: default
STATUS: DEPLOYED

RESOURCES:
==> v1/Deployment
NAME              READY  UP-TO-DATE  AVAILABLE  AGE
wigm-host-helm1   2/2    2           2          23s

==> v1/Pod(related)
NAME                               READY  STATUS    RESTARTS  AGE
wigm-host-helm1-56b595bc57-mwgj5   1/1    Running   0         23s
wigm-host-helm1-56b595bc57-wvf6f   1/1    Running   0         23s

==> v1/Service
NAME              TYPE          CLUSTER-IP    EXTERNAL-IP  PORT(S)       AGE
wigm-host-helm1   LoadBalancer  10.43.7.140   <pending>    80:32043/TCP  23s
```

```
$ helm delete --purge helm1

release "helm1" deleted
```

Summary

WIGM Via Helm

WIGM Via Any Operator

Operators

Custom Resources

```
apiVersion: wigm.carson-anderson.com/v1
kind: WigmGif
metadata:
  name: operators
spec:
  gif:
    title: "Deploying With Operators"
    link: https://m.../giphy.gif

  service:
    create_cloud_lb: true

  ingress:
    enabled: true
```

Operators

Custom Resource Definition

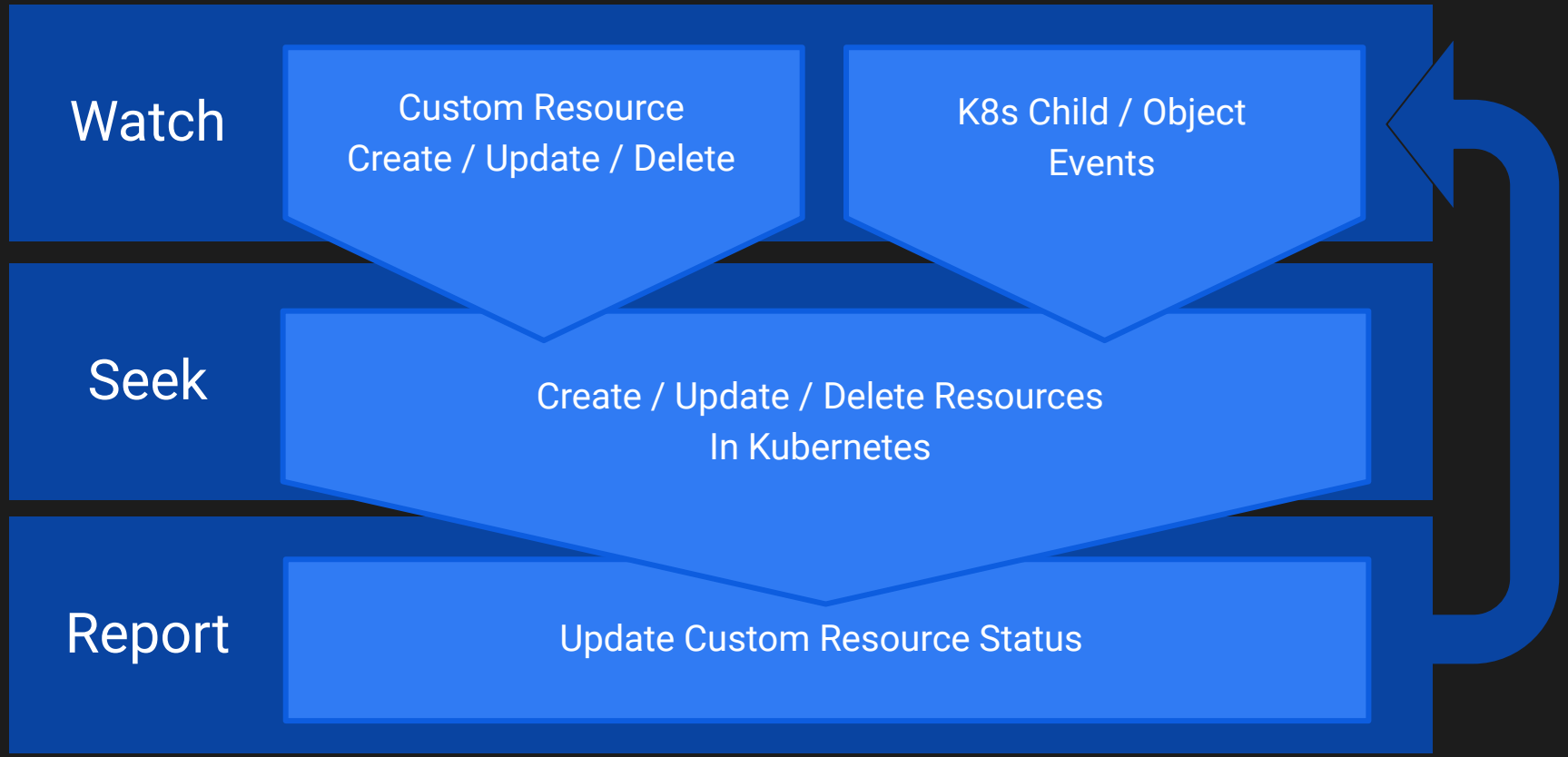
```
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
metadata:
  name: wigmgifs.wigm.carson-anderson.com
spec:
  group: wigm.carson-anderson.com
  names:
    kind: WigmGif
    listKind: WigmGifList
    plural: wigmgifs
    singular: wigmgif
  scope: Namespaced
  version: v1
  subresources:
    status: {}
  additionalPrinterColumns:
  [...]
```

Operators

CRD Continued

```
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
metadata:
  name: wigmgifs.wigm.carson-anderson.com
spec:
  [...]
  additionalPrinterColumns:
  - name: Title
    type: string
    description: The title of the gif
    JSONPath: .spec.gif.title
  - name: DeploymentRDY
    type: boolean
    description: Is the deployment ready
    JSONPath: .status.deployment.ready
  [...]
```

Operator Principles



Operator code structure

```
./wigm/operators/
└── releases/
    ├── operator1.yaml
    └── operator2.yaml
```

Operator Evaluation

----CREATE----	----UPDATE----	----UPDATE----	----REPORT----	----DELETE----
Yes LoadBalancer No Ingress	Change Title	Enable Ingress		

- Create Custom Resource file
- kubectl apply

OR

- API POST

```
cat > releases/operator1.yaml <<EOF
apiVersion: wigm.carson-anderson.com/v1
kind: WigmGif
metadata:
  name: operator1
spec:
  gif:
    title: operators
    link: https://media.giphy.com/[...]/giphy.gif

service:
  create_cloud_lb: true

ingress:
  enabled: false
EOF

kubectl apply -f \
  releases/operator1.yaml
```

----CREATE----	----UPDATE----	----UPDATE----	----REPORT----	----DELETE----
Yes LoadBalancer No Ingress	Change Title	Enable Ingress		

- Update resource file
- kubectl apply …

OR

- Kubectl edit …

OR

- kubectl patch …

OR

- API PATCH

```
cat > releases/operator1.yaml <<EOF
apiVersion: wigm.carson-anderson.com/v1
kind: WigmGif
metadata:
  name: operators
spec:
  gif:
    title: "Operators are awesome!"
    link: https://media.giphy.com/[...]/giphy.gif

service:
  create_cloud_lb: true

ingress:
  enabled: false
EOF

kubectl apply -f \
  releases/operator1.yaml
```

----CREATE----	----UPDATE----	----UPDATE----	----REPORT----	----DELETE----
Yes LoadBalancer No Ingress	Change Title	Enable Ingress		

- Update resource file
- kubectl apply …

OR

- Kubectl edit …

OR

- kubectl patch …

OR

- API PATCH

```
cat > releases/operator1.yaml <<EOF
apiVersion: wigm.carson-anderson.com/v1
kind: WigmGif
metadata:
  name: operators
spec:
  gif:
    title: "Operators are awesome!"
    link: https://media.giphy.com/[...]/giphy.gif

service:
  create_cloud_lb: true

ingress:
  enabled: true
EOF

kubectl apply -f \
  releases/operator1.yaml
```

```
$ kubectl get wigmgif operator1

NAME        TITLE                   DEPLOYMENTRDY   SERVICECRTD   SERVICETYPE    INGRESSCRTD
operator1   Operators are great!    true            true          LoadBalancer   true

$ kubectl get wigmgifs -ocustom-columns=
  NAME:.metadata.name,\
  DEPLOYMENT:.status.deployment.ready,\
  SERVICE:.status.service.created,\
  SVCTYPE:.status.service.type,\
  INGRESS:.status.ingress.created

# OR API GET
```

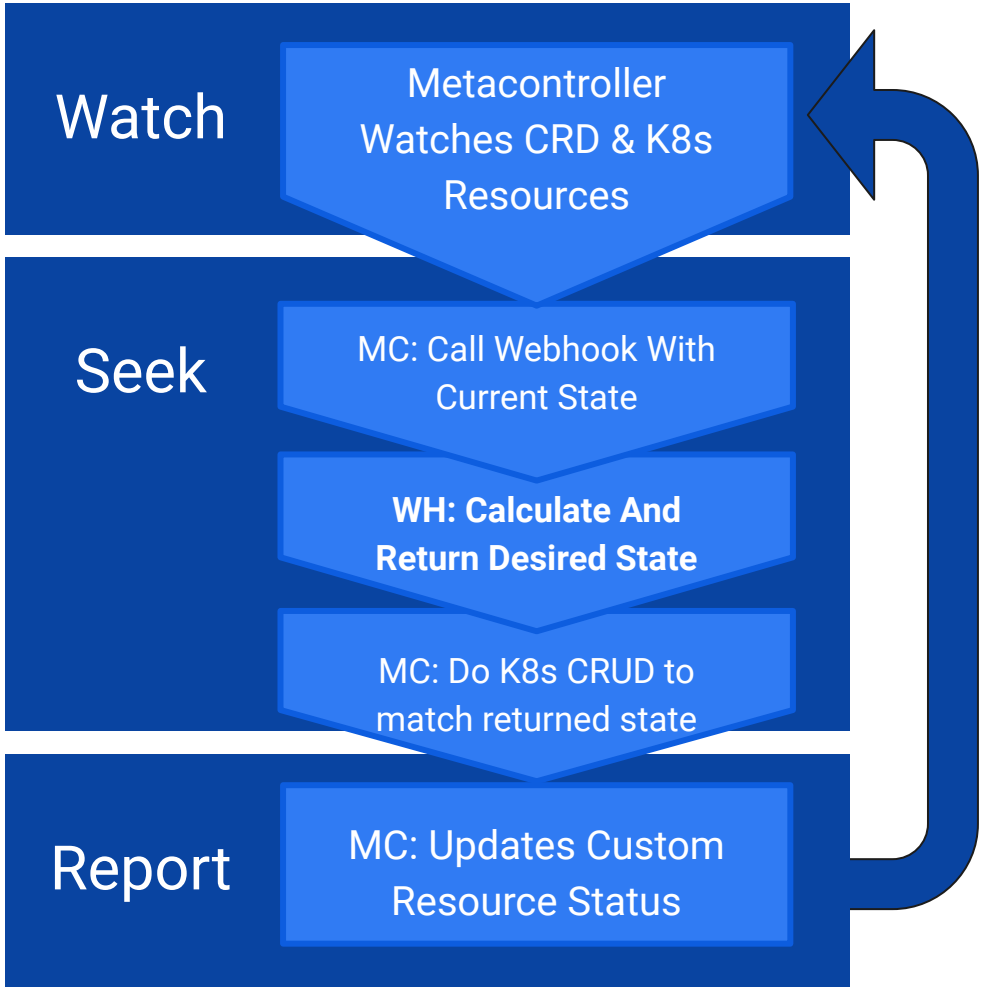
```
$ kubectl delete wigmgif operator1

wigmgif.wigm.carson-anderson.com "operator1" deleted

# OR API DELETE
```

Operator Demo

WIGM Via Metacontroller

Operators

Metacontroller



Metacontroller Setup

- Write the webhook sync function
- Install Metacontroller into cluster
- Run a webhook server in cluster
- Register webhook with metacontroller
  - CompositeController custom resource

Metacontroller Code Structure

```
./wigm/operators/with-metacontroller/
├── controller-metaconfig.yaml
├── controller-deploy.yaml
└── controller.py
```

Operators

Metacontroller Configuration: CompositeController

```
apiVersion: metacontroller.k8s.io/v1alpha1
kind: CompositeController
metadata:
  name: wigm-controller
spec:
  generateSelector: true
  parentResource:
    apiVersion: wigm.carson-anderson.com/v1
    resource: wigmgifs
  childResources:
  - apiVersion: apps/v1
    resource: deployments
    updateStrategy:
      method: InPlace
  - apiVersion: v1
    resource: services
    updateStrategy:
      method: InPlace
  - apiVersion: extensions/v1beta1
    resource: ingresses
    updateStrategy:
      method: InPlace
  hooks:
    sync:
      webhook:
        url: http://wigm-controller.wigm/sync
```

Operators

Metacontroller
controller.py
sync

```
[...]
def sync(self, parent, children):
  name = parent["metadata"]["name"]

  # default status
  status = {
  [...]
  }

  # Generate the desired child objects
  children = [
    self.get_deployment(parent, name),
    self.get_service(parent, name),
  ]

  [...]

  return {"status": status, "children": children}
```

Take in current state

Calculate desired state

Return progress and desired state

Operators

Metacontroller
controller.py
get_deployment

```
def get_deployment(self, parent, name):
    giftitle = parent["spec"]["gif"].get("title", name)
    giflink = parent["spec"]["gif"]["link"]

    deployment = {
      "apiVersion": "apps/v1",
      "kind": "Deployment",
      "metadata": {
        "name": "wigm-host-%s"%(name),
        "labels": { "app": "wigm",
                    "gif": name }
      },
      "spec": {
        [...]
          "spec": {
            "containers": [
              {
               [...]
                "env": [
                  { "name": "GIF_TITLE",
                    "value":  giftitle },
                  { "name": "GIF_SOURCE_LINK",
                    "value":  giflink
                  }
                [...]
    return deployment
```

Operators

Metacontroller Code sample

```
def get_service(self, parent, name):
    service = {
      "kind": "Service",
      "apiVersion": "v1",
      "metadata": {
        "name": "wigm-host-%s"%(name),
        "labels": {
          "app": "wigm",
          "gif": name
        }
      },

    # conditionally set service ki
    if parent["spec"].get(
         "service", {}).get("create_cloud_lb"):
      service["spec"]["type"] = "LoadBalancer"

    return service
```

String building

Business decisions in code

Operators

Metacontroller
controller.py
get_ingress

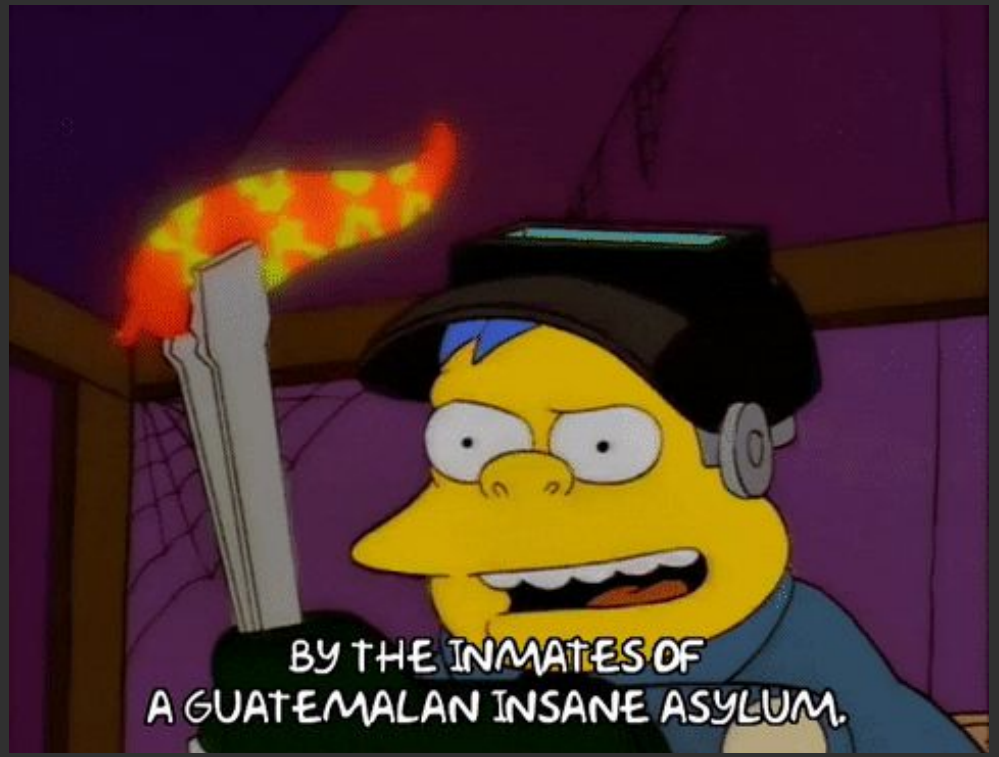
```
# if parent["spec"].get("ingress", {}).get("enabled",):
#   children.append(
#    self.get_ingress(parent, name)
#   )

---

def get_ingress(self, parent, name):
    ingress = {
      "apiVersion": "extensions/v1beta1",
      "kind": "Ingress",
      "metadata": {
        "name": "wigm-host-%s"%(name),
       [...]
      },
      "spec": {
        "rules": [
          {
           "host": "%s.wigm.carson-anderson.com"%(name),
           [...]
    }

    return ingress
```

Summary

WIGM Via Operator (Metacontroller)

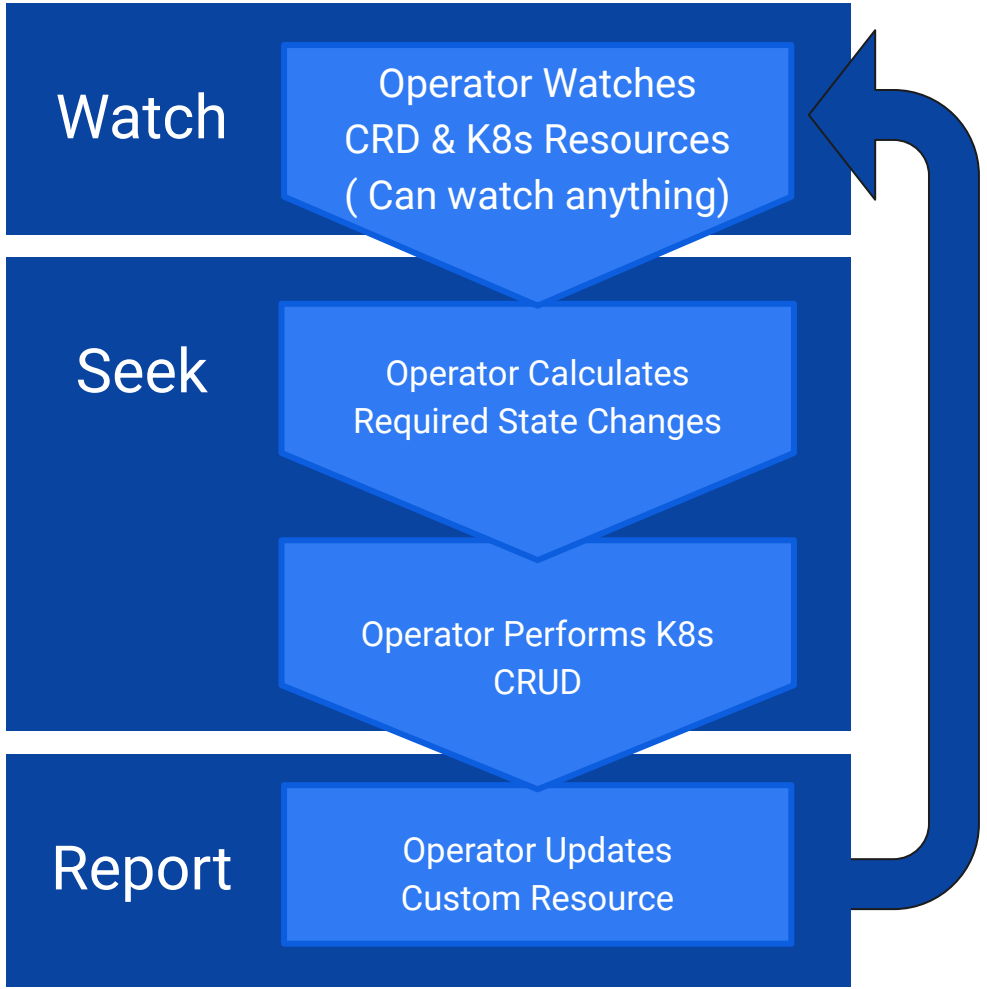
WIGM Via

Operator-SDK (Golang)



Operators

Operator SDK



Operator-SDK Setup

- Generate boilerplate
- Generate and update custom types
  - WigmGif
- Generate and update reconciler
- Build and run controller binary

Operators

Operator SDK Generate Boilerplate

```
# Generate boilerplate controller
operator-sdk new wigm

# Go to controller root
cd wigm
```

Operator-SDK Deploy Files

```
./wigm/operators/with-operator-sdk/
└── deploy
    ├── operator.yaml
    ├── role_binding.yaml
    ├── role.yaml
    └── service_account.yaml
```

Operators

Operator SDK Generate Types

```
# Generate boilerplate CRD and API Spec
operator-sdk add api \
--api-version=wigm.carson-anderson.com/v1 \
--kind=WigmGif
```

Operator-SDK Deploy Files

```
./wigm/operators/with-operator-sdk/
└── deploy
    ├── crds
    │   └── wigm_v1_wigmgif_crd.yaml
    ├── operator.yaml
    ├── role_binding.yaml
    ├── role.yaml
    └── service_account.yaml
```

Operator-SDK Types Files

```
./wigm/operators/with-operator-sdk/
└── pkg
    └── apis
        └── wigm
            └── v1
                └── wigmgif_types.go
```

Operators

Operator SDK Edit Types

```
# File ./pkg/apis/wigm/v1/wigmgif_types.go
[...]
// WigmGifSpec defines the desired state of
WigmGif
type WigmGifSpec struct {
 Gif      GifProperties [...]
 Service *ServiceProperties [...]
 Ingess  *IngressProperties [...]
}

type GifProperties struct {
 Title string `json:"title,omitempty"`
 Link string `json:"link"`
}

type ServiceProperties struct {
 CreateCloudLB bool `json:"create_cloud_lb"`
}

type IngressProperties struct {
 Enabled bool `json:"enabled"`
}
```

Operators

Ex: Power of Embedded Types

```
# File ./pkg/apis/wigm/v1/wigmgif_types.go
import (
  corev1 "k8s.io/api/core/v1"
)
[...]
// WigmGifSpec defines the desired state of
WigmGif
type WigmGifSpec struct {
 Gif GifProperties [...]
 Service *ServiceProperties [...]
 Resources *corev1.ResourceRequirements
}

type GifProperties struct {
 Title string `json:"title,omitempty"`
 Link string `json:"link"`
}

type ServiceProperties struct {
 CreateCloudLB bool `json:"create_cloud_lb"`
}
```

Operators

Generate Reconciler

```
# Generate boilerplate controller
operator-sdk add controller \
--api-version=wigm.carson-anderson.com/v1 \
--kind=WigmGif
```

Operator-SDK Controller Files

```
./wigm/operators/with-operator-sdk/
└── pkg
    └── controller
        └── wigmgif
            └── wigmgif_controller.go
```

Operators

WigmGIF Controller Psudocode

```
controller = NewController()

controller.Watch(WigmGifs)
controller.Watch(Child Deployments)
controller.Watch(Child Services)
controller.Watch(Child Ingresses)

for wigmggif in controller.Changes() {
  controller.SyncDeployment(wigmggif)
  if error {
    handleError()
  }
  controller.SyncService(wigmggif)
  if error {
    handleError()
  }
  controller.SyncIngress(wigmggif)
  if error {
    handleError()
  }
}
```

Graceful error handling

Graceful error handling

Graceful error handling

Operators

WigmGIF Controller

```
# /pkg/controller/wigmgif/wigmgif_controller.go

func add(mgr manager.Manager, r reconcile.Reconciler)
error {
      // Create a new controller
      c, err := controller.New("wigmgif-controller", mgr,
controller.Options{Reconciler: r})
      if err != nil {
            return err
      }

      // Watch for changes to primary resource WigmGif
      err = c.Watch(&source.Kind{Type: &wigmv1.WigmGif{}},
&handler.EnqueueRequestForObject{})
      if err != nil {
            return err
      }

      // Watch for changes to child deployments and
requeue the owner WigmGif
      err = c.Watch(&source.Kind{Type:
&appsv1.Deployment{}}, &handler.EnqueueRequestForOwner{
            IsController: true,
            OwnerType:    &wigmv1.WigmGif{},
      })
      [...]
```

Operators

WigmGIF Controller Syncs

```
func (r *ReconcileWigmGif) Reconcile(request reconcile.Request)
(reconcile.Result, error) {

[...]

// Sync the deployment, updating status in the passed instance
      if r, err := r.syncDeployment(instance); err != nil ||
r.Requeue == true {
            return r, err
      }

      // Sync the service, updating status in the passed instance
      if r, err := r.syncService(instance); err != nil ||
r.Requeue == true {
            return r, err
      }

      // Sync the ingress, updating status in the passed instance
      if r, err := r.syncIngress(instance); err != nil ||
r.Requeue == true {
            return r, err
      }

      // update the status
      if err := r.client.Status().Update(context.TODO(),
instance); err != nil
[...]
```

Operators

Golang Operator Reconciler Code Sample

```
func newDeploymentForWG(wg *wigmv1.WigmGif)
*appsv1.Deployment {
      name := wg.GetName()

      giflink := wg.Spec.Gif.Link
      giftitle := name
      if wg.Spec.Gif.Title != "" {
            giftitle = wg.Spec.Gif.Name
      }

      labels := map[string]string{
            "app": "wigm",
            "gif": name,
      }

      deployment := &appsv1.Deployment{
            ObjectMeta: metav1.ObjectMeta{
                  Name:      "wigm-host-" + name,
                  Namespace: wg.GetNamespace(),
                  Labels:    labels,
            },
            Spec: appsv1.DeploymentSpec{
                  [...]
                  Containers: []corev1.Container{{
                        Name:  "gifhost",
                        Image: "nginx:1.15-alpine",
                        Env: []corev1.EnvVar{{
                              Name:  "GIF_NAME",
                              Value: gifname,},
            [...]
```

Struct fields

Strictly Typed

String building

Side Note: Structify Demo!

structify.carsonoid.net

Operators

WigmGIF Controller Get Service

```
func newServiceForWG(wg *wigmv1.WigmGif) *corev1.Service
{
      [...]

    if (
        wg.Spec.Service != nil &&
        wg.Spec.Service.CreateCloudLB
        ) {
        service.Spec.Type = corev1.ServiceTypeLoadBalancer
    }

      [...]
}
```

Operators

Ex: Golang Fake Client Testing

```
func TestWigmGifBasicCreate(t *testing.T) {

// Create a fake client to mock API calls.
cl := fake.NewFakeClient(objs...)

// Run Reconciler
res, err := r.Reconcile(req)

// Get Result
err = cl.Get(context.TODO(),
  types.NamespacedName{
  Name: expectedDeployment.GetName(),
  Namespace: expectedDeployment.GetNamespace()},
  foundDeployment)

// Validate
if err != nil {
  t.Errorf(
    "Error getting expected deployment: %s", err
  )
}

}
```

Same Reconcile Function as the Controller

Summary

WIGM Via Operator (Operator-SDK)

The Evolution WIGM

YAML

HELM

Operators

Management Pattern Comparison

	Develop Speed	Portability	Release Creation	Release Updates	Testability	Automatic State Seek	Performance
YAML			Manual	Manual	Manual		
Helm		Req. local copy of the templates and values		Manually Applied			
Operator: Metacontroller	Webhook Redeploy	Supports any K8s API tool or Code				Batched	
Operator: operator-sdk	Golang Learning Curve	Supports any K8s API tool or Code					

More of Me

github.com/carsonoid/talk-wigm-and-operators

K3s (demos): github.com/rancher/k3s

@carsonoid

@carson_ops

kube-decon.carson-anderson.com

dynamic-kubernetes.carson-anderson.com

salt-decon.carson-anderson.com

structify.carsonoid.net