
KubeCon
CloudNativeCon
North America 2022
BUILDING FOR THE ROAD AHEAD
DETROIT 2022
TiKV
TiKV
HELM

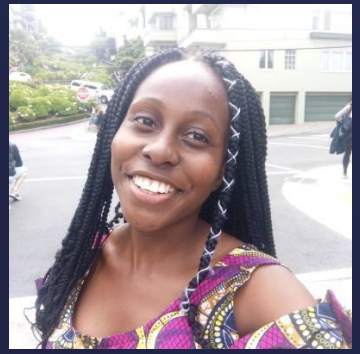
KubeCon CloudNativeCon North America 2022
BUILDING FOR THE ROAD AHEAD
DETROIT 2022

Tutorial: How To Write a Reconciler Using K8s Controller-Runtime!

Scott Rigby, Somtochi Onyekwere, Niki Manoledaki & Soulé Ba, Weaveworks; Amine Hilaly, Amazon Web Services

Tutorial: How To Write a Reconciler Using K8s Controller-Runtime!

Scott Rigby
Developer Experience Engineer
Weaveworks



Somtochi Onyekwere
Developer Experience Engineer
Weaveworks

Niki Manoledaki
Software Engineer
Weaveworks

KubeCon | CloudNativeCon
North America 2022

BUILDING FOR THE ROAD AHEAD

DETROIT 2022

October 24-28, 2021

Soulé Ba
Consulting Reliability Engineer
Weaveworks

Amine Hilaly
Software Development Engineer
Amazon Web Services

Agenda

- Demo - 5m
- Local dev setup - 10m
- Challenge - 5m
- Intro to concepts - 10m
- Step-by-step write the controller! - 40m
- Wrap up + Q&A - 10m

Challenge

The challenge

Goal

As a hypothetical KubeCon speaker
In order to have fun and learn things
I want to be able to manage my KubeCon proposal submissions declaratively

We have a CFP API

- Code and docs here
- It supports Speaker and Proposal objects, with a one to many relationship

Write a K8s controller that

- Defines CRDs for speaker and proposal
- Reconciles CFP API submissions with our declarative CRDs

Local dev setup

Repository

Please fork & clone this repository:

https://bit.ly/3TFnryi

github.com/scottrigby/
how-to-write-a-reconciler-using-k8s-controller-runtime

Prerequisites

Repository: https://bit.ly/3TFnryi

- Just watch
- GitPod (preferred)
  - Open the link in the repository

    Open in Gitpod
  - Please run it now

- Vagrantfile
  - Follow instructions in Readme

- DIY everything!
  - Requirements:
    - Go
    - Kind
    - Docker
    - Kustomize

Demo

Demo time!

Intro to concepts

- Reconciliation
- K8s controller-runtime
- Kubebuilder
- Conditions
- Status & observedGeneration

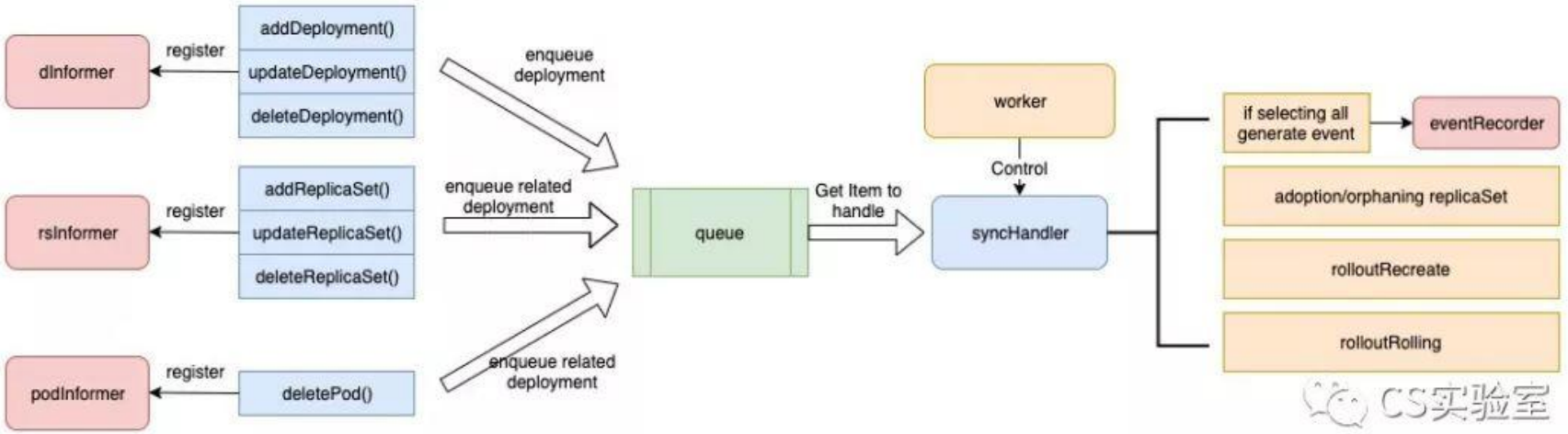
Reconciliation

```
for {
  desired := getDesiredState()
  current := getCurrentState()
  makeChanges(desired, current)
}
```

source: `kubernetes/community` repo
developer guide on "Writing Controllers"

K8s controller-runtime

Native controllers like Deployment and ReplicaSet were built using core components like Informers, Listers, work queues etc…



K8s controller-runtime

- Controller-runtime is a set of go libraries for building controllers.
- Product of years of experience building controllers.
- Hides the complexity of building the reconcilers
- Allows you to focus on the logic of the sync function
- Well tested and actively maintained

K8s controller-runtime

- Standardize logging `pkg/log`
- Leader election `pkg/leaderelection`
- Webhooks `pkg/webhooks`
- Ratelimiting `pkg/ratelimiter`
- Better and simpler testing `pkg/envtest`

Kubebuilder

Kubebuilder is framework for building operators following Kubernetes API best practices

- It builds on controller-runtime and controller-tools
- Also providers code -generation with markers

  Commands to scaffold

  ```
  kubebuilder init --domain my.domain --repo my.domain/guestbook
  kubebuilder create api --group webapp --version v1 --kind Guestbook
  ```

Other frameworks include operator-framework

Kubebuider book: https://book.kubebuilder.io/

Conditions

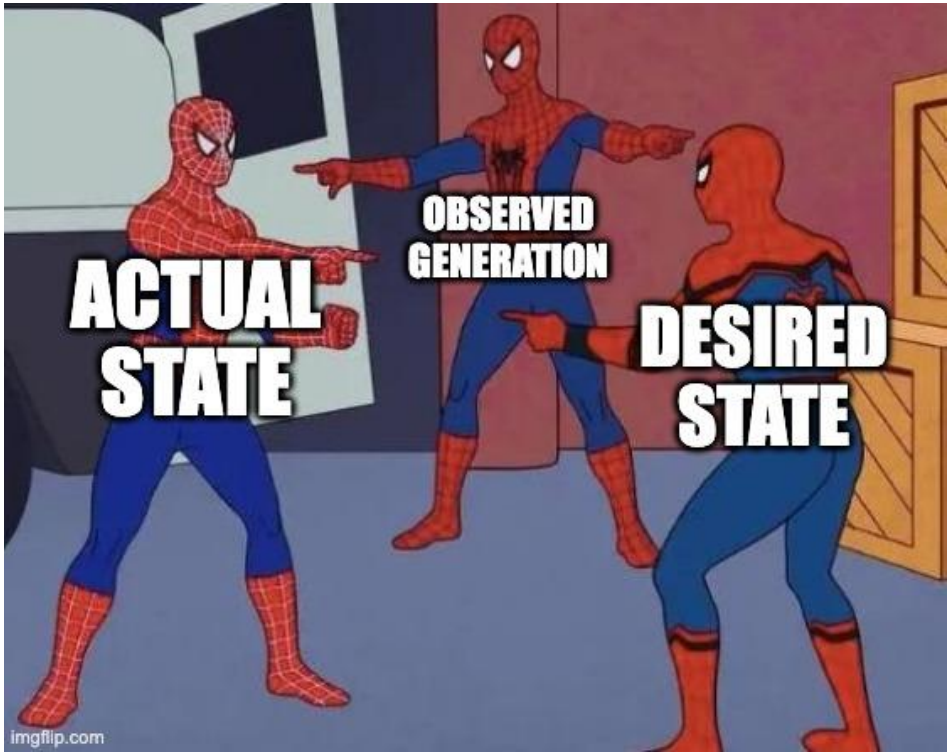
- The conditions is a set of types (Ready, PodScheduled…) with a status (True, False or Unknown) that make up the ‘computed state’ of a Resource.
- The set of conditions describes the ‘current’ state.
- Conditions are just a way of communicating changes of state between components, and this state can always be reconstructed by observing the system.
- Condition types should indicate state in the “abnormal-true” polarity.

https://github.com/kubernetes/community/blob/4c9ef2d/contributors/devel/sig-architecture/api-conventions.md
https://github.com/kubernetes-sigs/cli-utils/blob/master/pkg/kstatus/README.md

Status & observedGeneration

- A /status subresource MUST be provided to enable system components to update statuses of resources they manage.
- If the generation and the observedGeneration of a resource does not match, it means there are changes that the controller has not yet seen, and therefore not acted upon.

https://github.com/kubernetes/community/blob/4c9ef2d/contributors/devel/sig-architecture/api-conventions.md#spec-and-status



Step-by-step write the controller!

Step 1

Run

```
$ git checkout tags/s1 -b s1-branch
```

run `make test` to run the test cases

Step 2

Run

```
$ git checkout tags/s2 -b s2-branch
```

run `make test` to run the test cases

Step 3

Run

```
$ git checkout tags/s3 -b s3-branch
```

run `make test` to run the test cases

Step 4

Run

```
$ git checkout tags/s4 -b s4-branch
```

run `make test` to run the test cases

Step 5

Run

```
$ git checkout tags/s5 -b s5-branch
```

run `make test` to run the test cases

Step 6

Run

```
$ git checkout tags/s6 -b s6-branch
```

run `make test` to run the test cases

Step 7

Run

```
$ git checkout tags/s7 -b s7-branch
```

run `make test` to run the test cases

Wrap up + Q&A

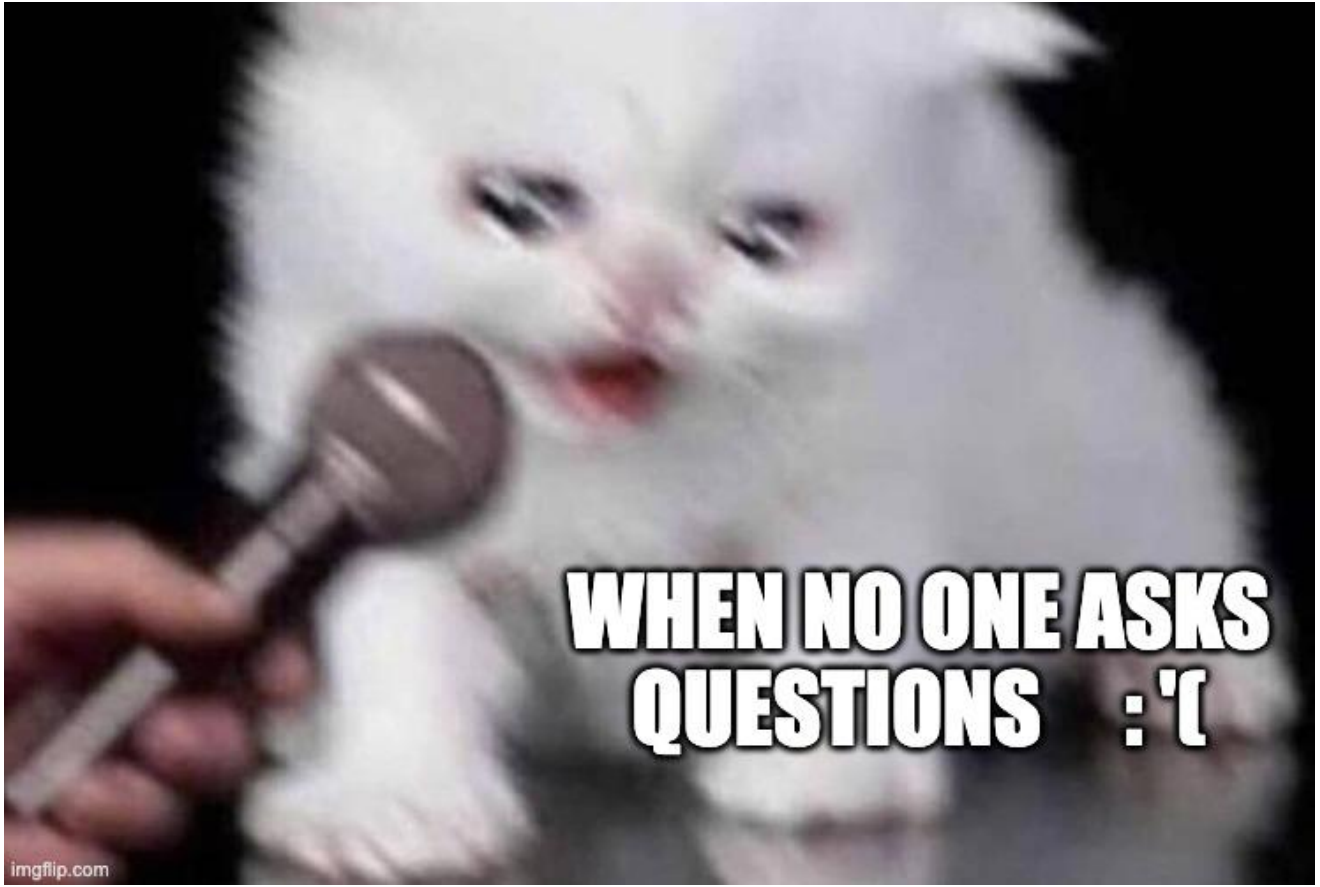
What we learned

TO-DO: Add the most salient points here to remind folks they actually learned things, since their brain is probably full by now

Resources

- Cloud Native Glossary: glossary.cncf.io
- Kubernetes community developer guide: https://github.com/kubernetes/community/blob/master/contributors/devel/sig-api-machinery/controllers.md
- Kubebuilder book: sigs.k8s.io/kubebuilder
- Kubernetes controller-runtime Project: github.com/kubernetes-sigs/controller-runtime
- Flux pkg/runtime: pkg.go.dev/github.com/fluxcd/pkg/runtime
- Finalizers to Control Deletion: kubernetes.io/blog/2021/05/14/using-finalizers-to-control-deletion
- Implementing observedGeneration: alenkacz.medium.com/kubernetes-operator-best-practices-implementing-observedgeneration-250728868792

Q&A



Q&A

Session QR Codes will be sent via email before the event

Please scan the QR Code above to leave feedback on this session