Keep the Space Shuttle Flying

Writing Robust Operators

Illya Chekrygin
Founding Engineer, Upbound

ichekrygin

illya_chekrygin

What’s with the title?

```
// ==================================================================
// PLEASE DO NOT ATTEMPT TO SIMPLIFY THIS CODE.
// KEEP THE SPACE SHUTTLE FLYING.
// ==================================================================
//
// This controller is intentionally written in a very verbose style. You will
// notice:
//
// 1. Every 'if' statement has a matching 'else' (exception: simple error
//    checks for a client API call)
// 2. Things that may seem obvious are commented explicitly
//
// We call this style 'space shuttle style'.
…
```

Paul Morie

Circa July, 2016

https://github.com/kubernetes/kubernetes/blob/master/pkg/controller/volume/persistentvolume/pv_controller.go

Kubernetes

- Killer API
  - Declarative Style
  - Level-based
  - State separation: Desired (Spec) vs. Observed (Status)
  - Complete
  - Authoritative
  - Extensible

Extending Kubernetes

- Customization
  - Configuration
  - Extension
- Controller Pattern
- Custom Resources
  - API Aggregation
  - CRD
- Kubernetes Application, a.k.a Operator

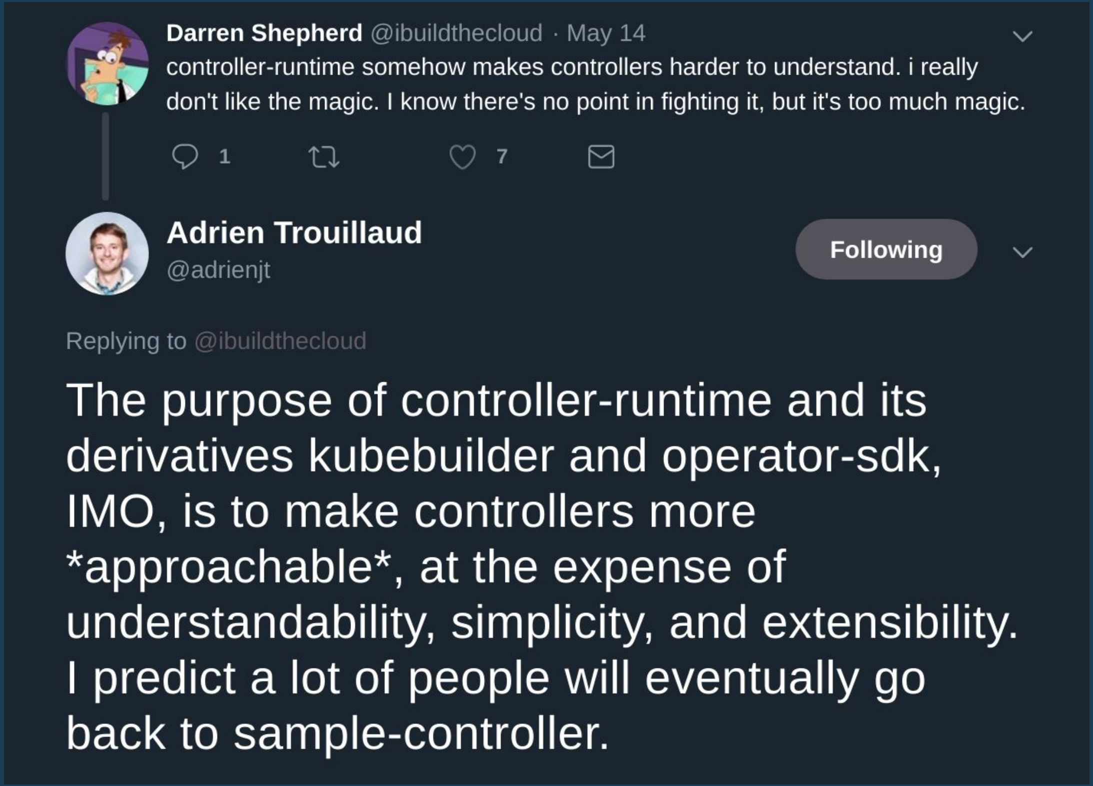
Frameworks

- controller-runtime
  - Kubebuilder
  - Operator-SDK
- Metacontroller
- Shell-operator
- Rook Operator-Kit
- client-go
  - sample-controller

FRESH OFF THE PRESS

Frameworks

- controller-runtime
    - Kubebuilder
    - Operator-SDK
- Metacontroller
- Rook Operator-Kit
- client-go
    - sample-controller



Darren Shepherd @ibuildthecloud · May 14

controller-runtime somehow makes controllers harder to understand. i really don't like the magic. I know there's no point in fighting it, but it's too much magic.

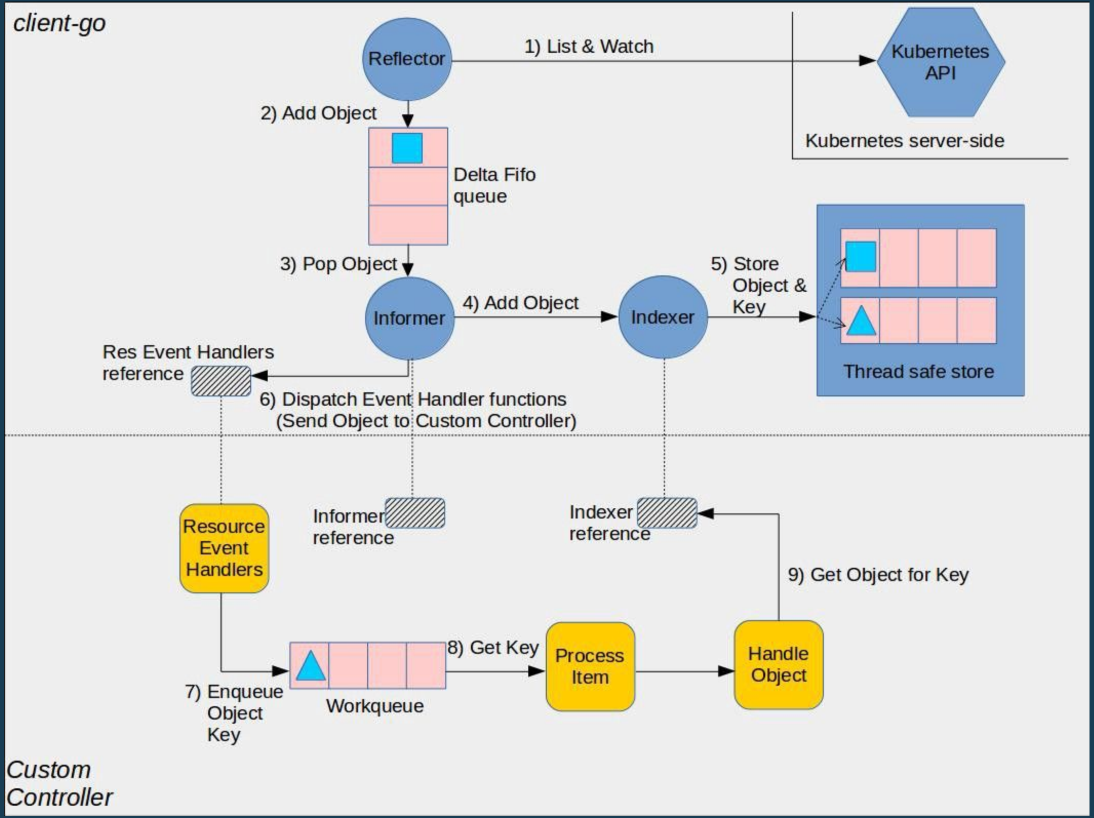
Adrien Trouillaud
@adrienjt

Replying to @ibuildthecloud

The purpose of controller-runtime and its derivatives kubebuilder and operator-sdk, IMO, is to make controllers more *approachable*, at the expense of understandability, simplicity, and extensibility. I predict a lot of people will eventually go back to sample-controller.

Frameworks

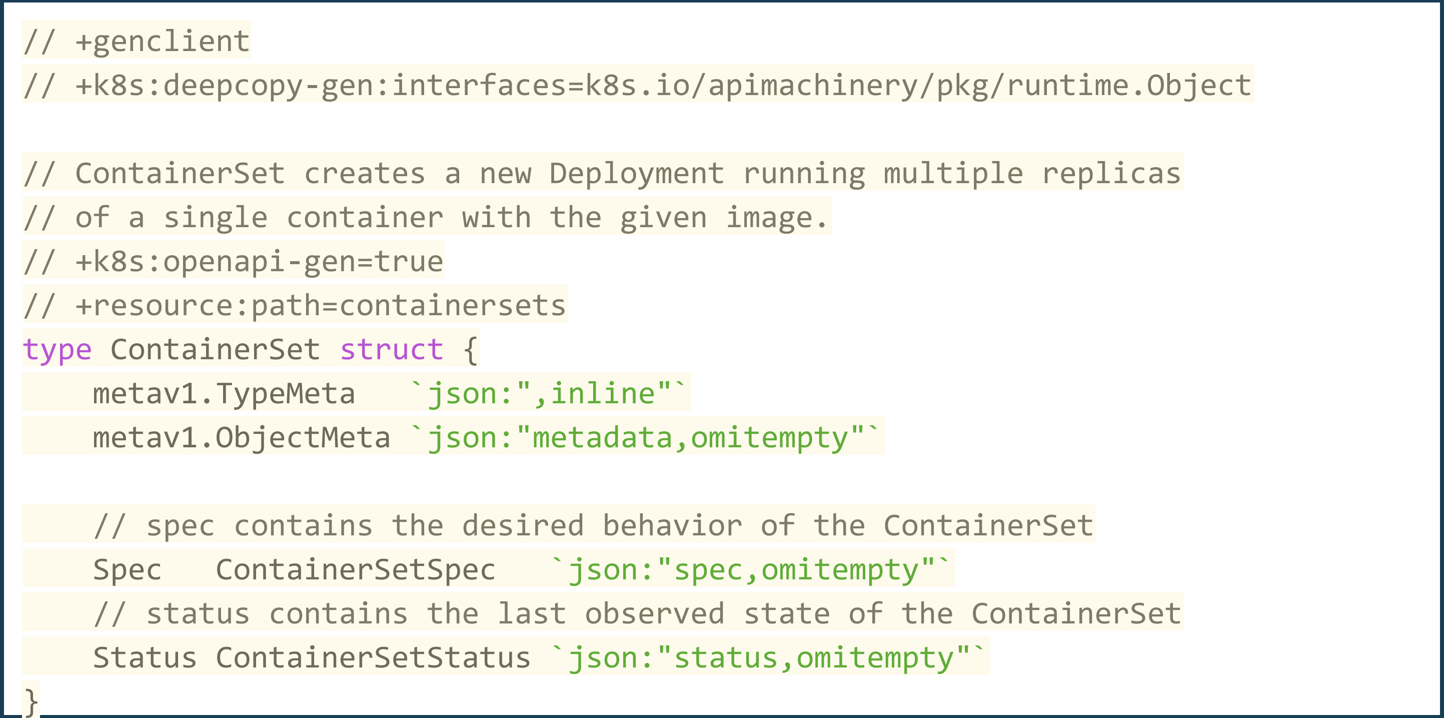
- controller-runtime
  - Kubebuilder
  - Operator-SDK
- Metacontroller
- Rook Operator-Kit
- client-go
  - sample-controller



Operator Anatomy

- Type
- Controller
- Webhook(s)

Type

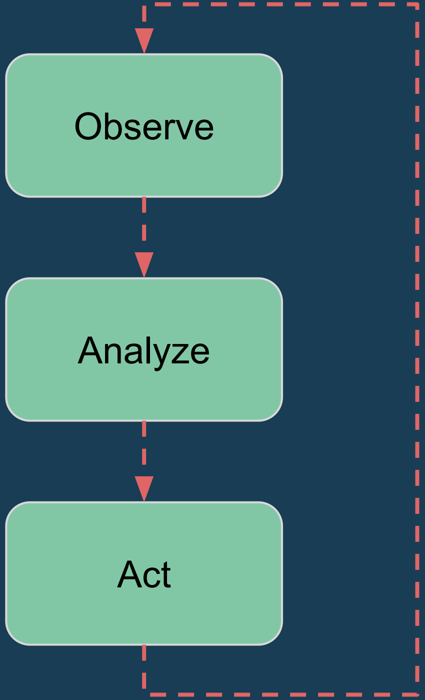


```go
// +genclient
// +k8s:deepcopy-gen:interfaces=k8s.io/apimachinery/pkg/runtime.Object

// ContainerSet creates a new Deployment running multiple replicas
// of a single container with the given image.
// +k8s:openapi-gen=true
// +resource:path=containersets
type ContainerSet struct {
    metav1.TypeMeta   `json:",inline"`
    metav1.ObjectMeta `json:"metadata,omitempty"`

    // spec contains the desired behavior of the ContainerSet
    Spec   ContainerSetSpec   `json:"spec,omitempty"`
    // status contains the last observed state of the ContainerSet
    Status ContainerSetStatus `json:"status,omitempty"`
}
```

Controller

- Reconcile (key)
  - API Get Object (key)
  - Process Object
    - Actual State -> Desired State
  - CRUD children resources
  - API Update:
    - Object (as whole) or
    - Object Status (sub-resource)

```
spec:
...
  scope: Namespaced
  subresources:
    status: {}
```

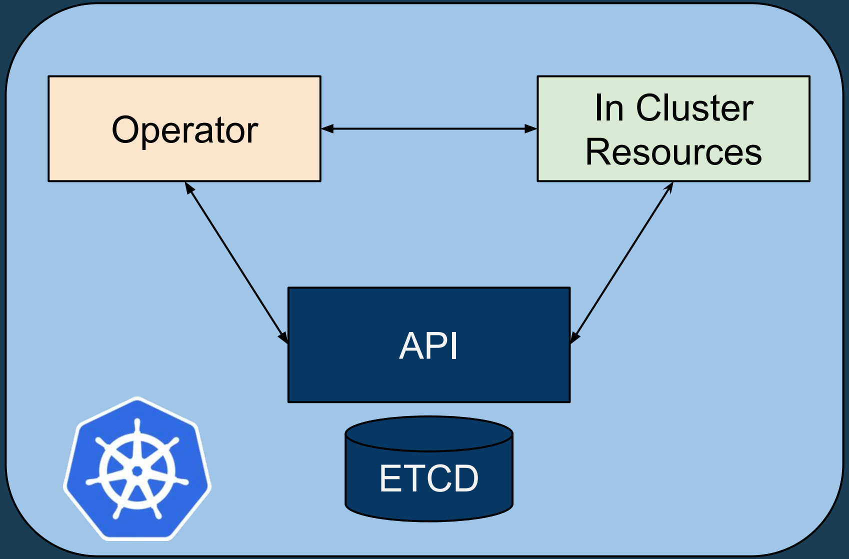


Observe

Analyze

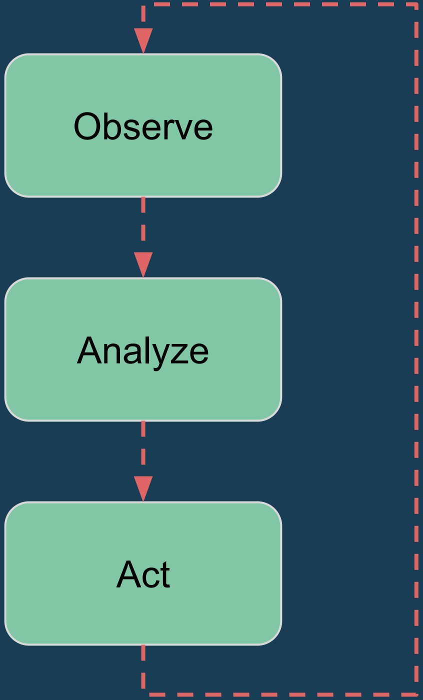
Act

In Cluster Resources

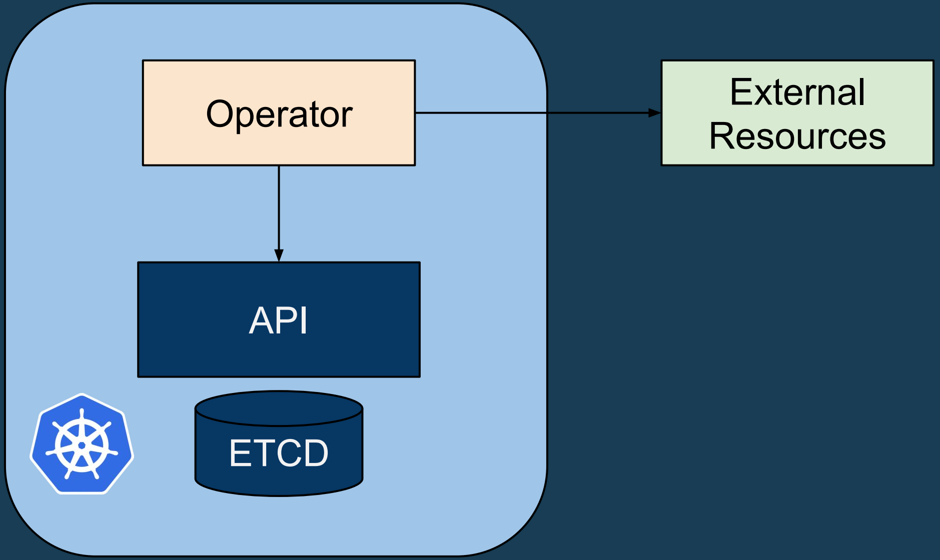


External Resource Controller

- Reconcile (key)
  - API Get Object (key)
  - Process Object
    - Actual State -> Desired State
  - CRUD children resources
  - API Update:
    - Object (as whole) or
    - Object Status (sub-resource)

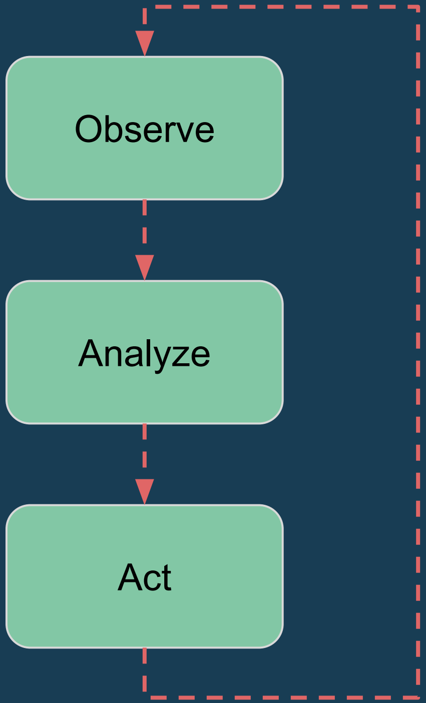


External Resource Controller



External vs. Internal CRUD

- Internal Resources:
  - Kubernetes API + client-go
- External API Resources:
  - AWS API + aws-sdk-go
  - Azure API + azure-sdk-for-go
  - GCP API + google.golang.org/api
- Other External Resources

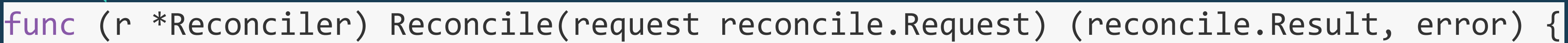


Reconciler Patterns

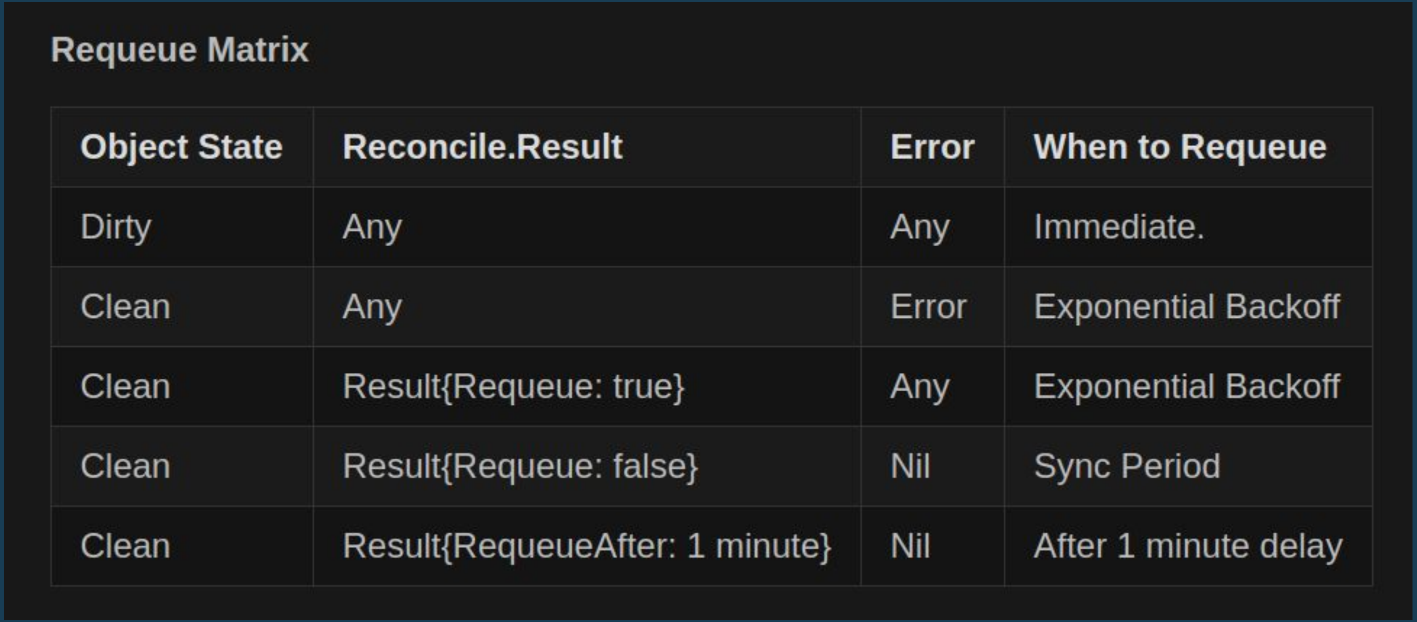
```
func (r *Reconciler) Reconcile(request reconcile.Request) (reconcile.Result, error)
```

- Update: External or Internal
- Result
  - Requeue: `true` ← *don't use while waiting*
  - RequeueAfter: `duration` ← *maybe a better choice*
  - Requeue: `false` ← *fuhgeddaboudit*
- Errors
- Controller Manager: `SyncPeriod` ← *everybody dance now*

Reconcile Patterns - Requeue



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



Requeue Matrix

Object State	Reconcile.Result	Error	When to Requeue
Dirty	Any	Any	Immediate.
Clean	Any	Error	Exponential Backoff
Clean	Result{Requeue: true}	Any	Exponential Backoff
Clean	Result{Requeue: false}	Nil	Sync Period
Clean	Result{RequeueAfter: 1 minute}	Nil	After 1 minute delay

Reconcile Patterns - Status

- Phase - deprecated
- Conditions
    - Multiple Active
    - Oscillating
    - Monotonic

Reconciler Patterns - Conditions

- Type: Pending, Ready, Failed, etc
- Status: True/False/Unknown
- Reason
- Message
- LastTransitionTime ← watch out
- LastHeartbeatTime ← new to me

Reconcile Patterns - Concurrency

- Typically Single Goroutine per Controller
  - Multiple - do your own bookkeeping
- Long running operations
  - When is it too long?
- Don’ts
  - Don’t Block
  - Don’t Branch
  - Don’t listen to any of these

Reconcile Patterns - Status

- Conditions
- Additional Fields:
  - References
  - Resource Properties
  - External Running Operations

```
additionalPrinterColumns:
- JSONPath: .status.bindingPhase
  name: STATUS
  type: string
- JSONPath: .status.state
  name: STATE
  type: string
- JSONPath: .status.clusterName
  name: CLUSTER-NAME
  type: string
- JSONPath: .status.endpoint
  name: ENDPOINT
  type: string
```

Reconcile Patterns - Garbage Collection

- Finalizer
- Owner Reference
- DeleteOptions.propagationPolicy:
  - Background
  - Foreground
  - Orphan

Reconcile Patterns - Issues

- Project Structure
  - pkg/apis/group/version/type.go ←
  - pkg/apis/group/sub-group/version/type.go ←
- Tests - integration like
  - Good for API and/or small projects
  - Maybe not so good for controllers and/or large projects
- Stale Reads
  - `Operation cannot be fulfilled on mytype \"foo\": the object has been modified; please apply your changes to the latest version and try again`

Takeaways

- Keep It Simple!
  - At least try

References

- kubernetes/community/api-conventions
- https://coreos.com/operators/
- https://blog.couchbase.com/kubernetes-operators-game-changer/
- https://kubernetes.io/docs/concepts/workloads/controllers/garbage-collection/
- https://kubernetes.io/docs/concepts/extend-kubernetes/extend-cluster/
- https://kubernetes.io/docs/concepts/extend-kubernetes/api-extension/custom-resources/
- https://book.kubebuilder.io/basics/simple_controller.html
- https://github.com/crossplaneio/crossplane/blob/master/design/reconciler-patterns.md
- https://github.com/operator-framework/operator-sdk
- https://github.com/kubernetes-sigs/kubebuilder
- https://github.com/GoogleCloudPlatform/metacontroller
- https://github.com/kubernetes/kubernetes/issues/59850 [Propagation Policy: Foreground]
- https://github.com/kubernetes-sigs/controller-runtime/issues/403 [Stale Read]

Thank you

Q&A

Illya Chekrygin

Crossplane