

Building a Custom K8S Controller

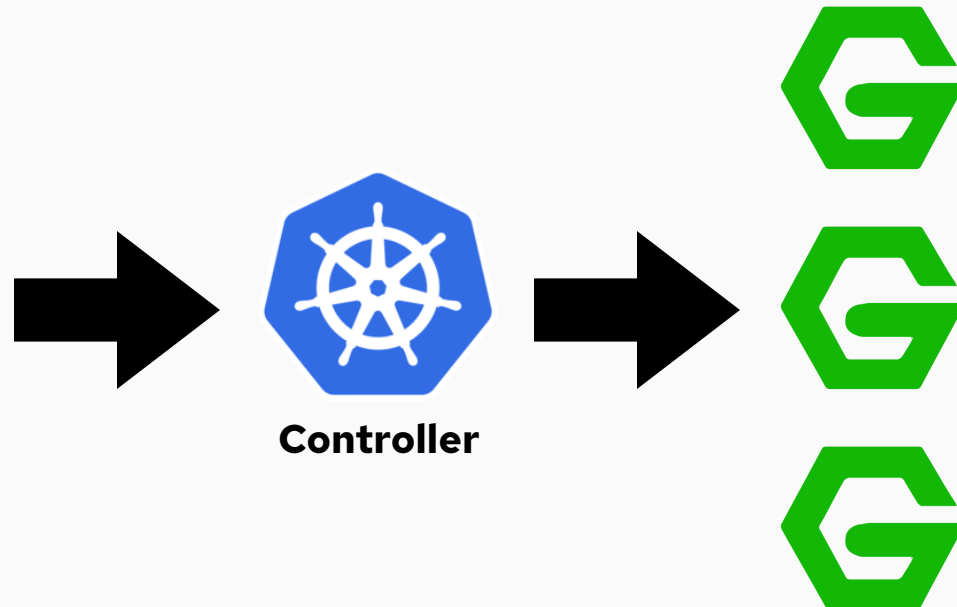


Hrishikesh
@hrishike8

Controller(s)

Maintains the **desired** state

```
1 spec:
2   replicas: 3
3   selector:
4     matchLabels:
5       app: nginx
6   template:
7     ....
8   spec:
9     containers:
10      - name: nginx
11        image: nginx:1.7.9
```



...Of Resource(s)

Pod Deployment Node

....

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: nginx-deployment
5    labels:
6      app: nginx
7  spec:
8    replicas: 3
9    .....
```

Custom Resource(s)

~= Database table or document

```
apiVersion: apiextensions.k8s.
kind: CustomResourceDefinition
metadata:
  name: podsets.demo.k8s.io
spec:
  group: demo.k8s.io
  version: v1alpha1
  names:
    kind: PodSet
```

Programming Entity

```
type PodSet struct {
    metav1.TypeMeta
    metav1.ObjectMeta
    Spec      PodSetSpec
    Status    PodSetStatus
}
```

- 1 apiVersion: **demo/v1**
- 2 kind: **PodSet**
- 3 metadata:
- 4 name: **two-podset**
- 5 spec:
- 6 replicas: 3

Custom Controller(s)

```
1 apiVersion: demo/v1
2 kind: PodSet
3 metadata:
4   name: two-podset
5 spec:
6   replicas: 3
```

CR



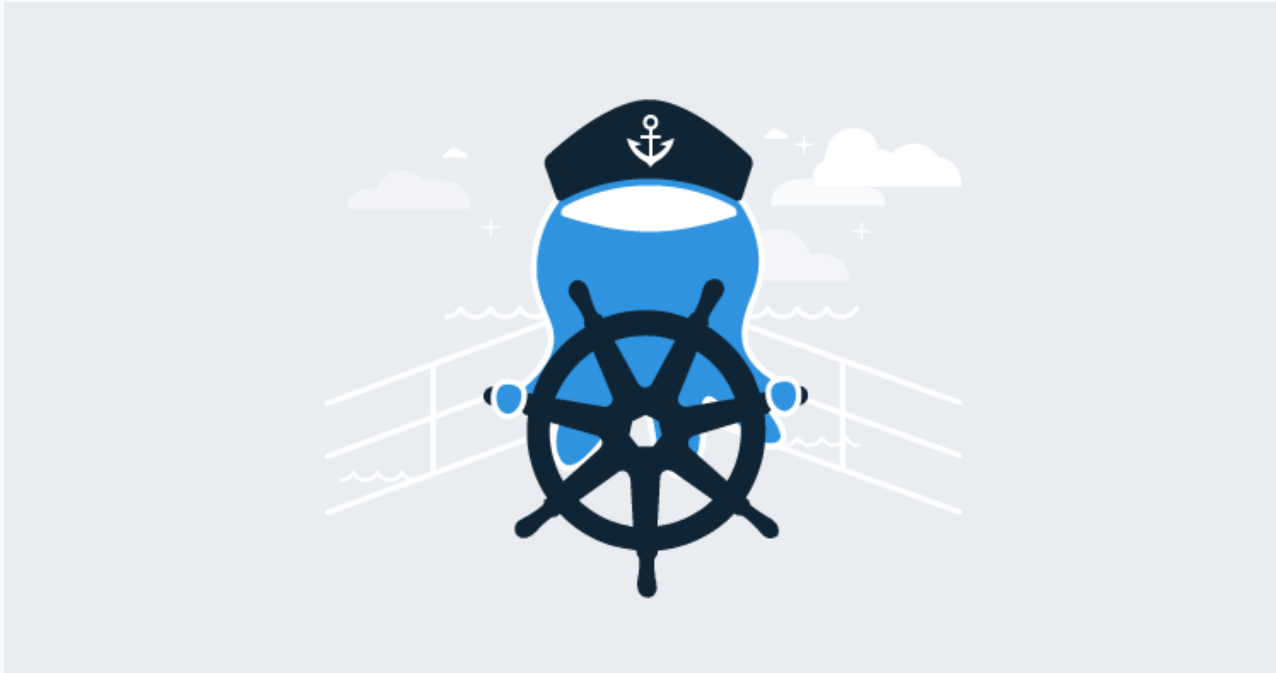
**Controller to
manage the
CR's state**

Use of Custom Controllers?



Extends kubernetes platform

Use of Custom Controllers?



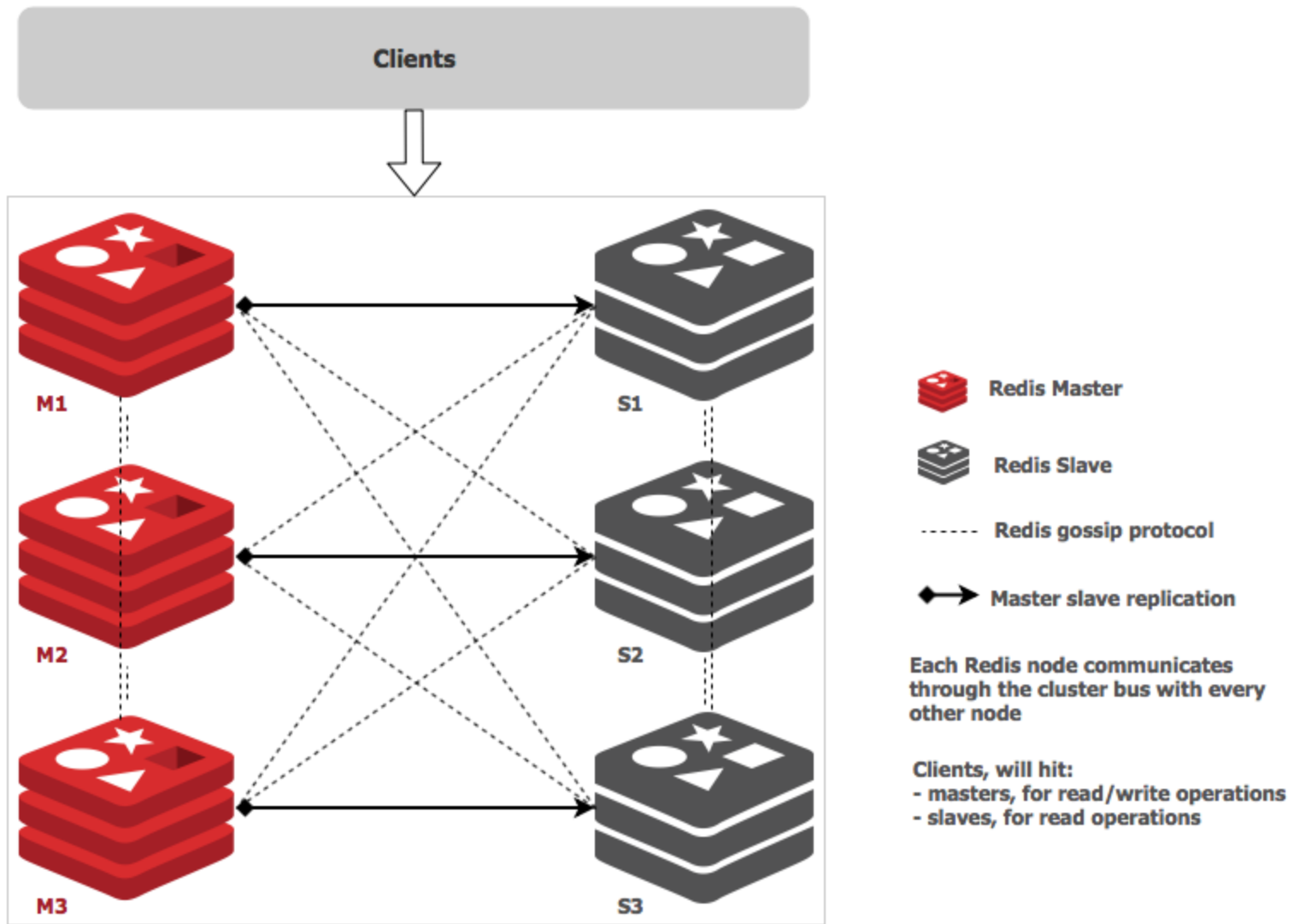
Operators

Operators

Its an autopilot for an application where it extends Kubernetes API to install, configure, scaling or recovery of application instances.

It bakes application operational knowledge into application specific controller

Run HA Redis



DIY way

Deployments

PVC

Images

Services

**Expert devops
knowledge of
Redis HA**

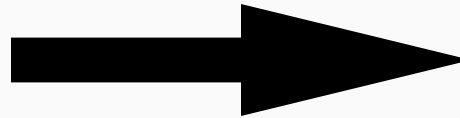
.....

Operator Way

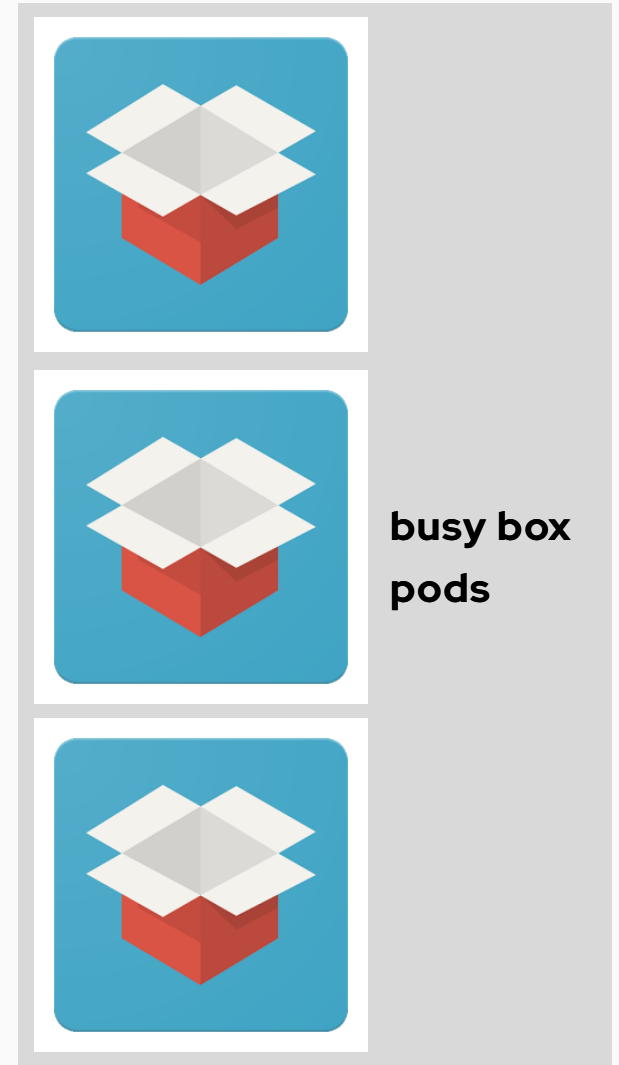
```
1  apiVersion: databases.spotahome.com/v1
2  kind: RedisFailover
3  metadata:
4    name: redisfailover-persistent-keep
5  spec:
6    sentinel:
7      replicas: 3
8    image: redis:4.0-alpine
9    redis:
10     replicas: 3
11     image: redis:4.0-alpine
```

PodSet Example

```
1 #cr.yaml
2
3 apiVersion: demo/v1
4 kind: PodSet
5 metadata:
6   name: two-podset
7 spec:
8   replicas: 3
```



```
1 $ kubectl apply -f cr.yaml
```



Go Lang Dependencies

client-go : k8s.io/client-go

apimachinery : k8s.io/apimachinery

api : k8s.io/api

code-generator : k8s.io/code-generator

Setup

```
git clone git@github.com:hrishin/podset-operator.git \  
$GOPATH /src/github.com/hrishin/podset-operator
```

Define API Resource

```
1 type PodSet struct {  
2     metav1.TypeMeta `json:",inline"`  
3     metav1.ObjectMeta `json:"metadata,omitempty"`  
4  
5     Spec PodSetSpec `json:"spec"`  
6     Status PodSetStatus `json:"status"`  
7 }
```

Golang API's ...

..to get, create, delete, watch
PodSet resource

```
watcher, err := client.DemoV1alpha1().  
    PodSets("pods").  
    Watch(metav1.ListOptions{})
```

```
1 ~/go/src/k8s.io/code-generator/generate-groups.sh \  
2 "client" \  
3 github.com/hrishin/podset-operator/pkg/client \  
4 github.com/hrishin/podset-operator/pkg/apis \  
5 demo:v1alpha1
```


Hit the wall

```
x-255 ~/go/src/github.com/hrishin/podset-operator [master L|+ 5...4]
19:44 $ ~/go/src/k8s.io/code-generator/generate-groups.sh "deepcopy,client" github.com/hrishin/podset-operator/pkg/apis demo:v1alpha1
Generating deepcopy funcs
F0524 19:44:54.564606      1761 main.go:82] Error: Failed making a parser: unable to add directory "github.com/hrishin/podset-operator/pkg/apis/demo/v1alpha1": go/build: importGo github.com/hrishin/podset-operator/pkg/apis/demo/v1alpha1: cannot find module providing package github.com/hrishin/podset-operator/pkg/apis/demo/v1alpha1"
it status 1
go: finding github.com/hrishin/podset-operator/pkg/apis/demo/v1alpha1 latest
go: finding github.com/hrishin/podset-operator/pkg/apis/demo latest
go: finding github.com/hrishin/podset-operator/pkg/apis latest
go: finding github.com/hrishin/podset-operator/pkg latest
can't load package: package github.com/hrishin/podset-operator/pkg/apis/demo/v1alpha1: unknown import path "github.com/hrishin/podset-operator/pkg/apis/demo/v1alpha1": cannot find module providing package github.com/hrishin/podset-operator/pkg/apis/demo/v1alpha1
```

Boilerplate code

Define custom types(CRD)


▲ PODSET-OPERATOR

▲ pkg


▲ apis

▲ demo

▲ v1alpha1

 doc.go

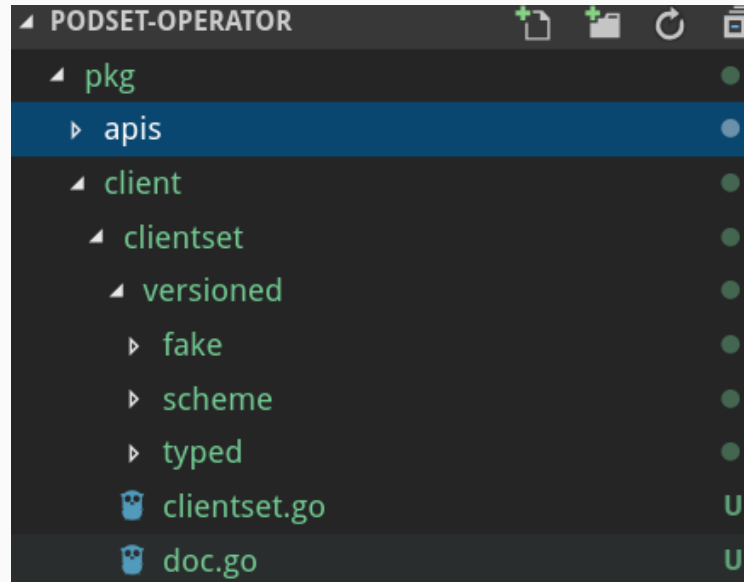
 register.go

 types.go

 register.go

CRD and Clients

```
✓ ~/go/src/github.com/hrishin/podset-operator [master L|+ 2...15]
20:30 $ ~/go/src/k8s.io/code-generator/generate-groups.sh "deepcopy,client" \
> github.com/hrishin/podset-operator/pkg/client \
> github.com/hrishin/podset-operator/pkg/apis \
> demo:v1alpha1
Generating deepcopy funcs
Generating clientset for demo:v1alpha1 at github.com/hrishin/podset-operator/pkg/client/clientset
```



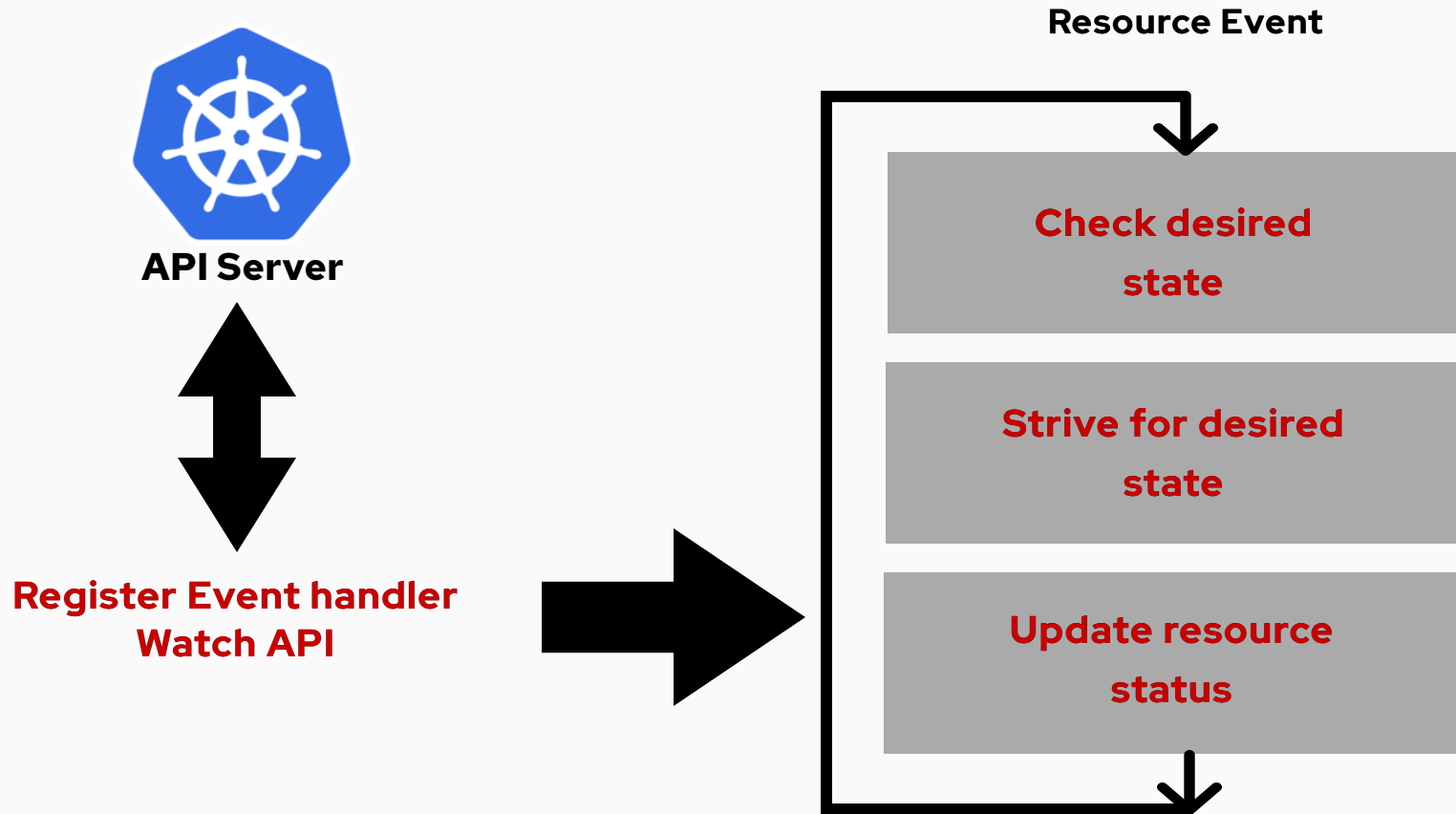
in place now, yay!

Hello "Watch" Demo

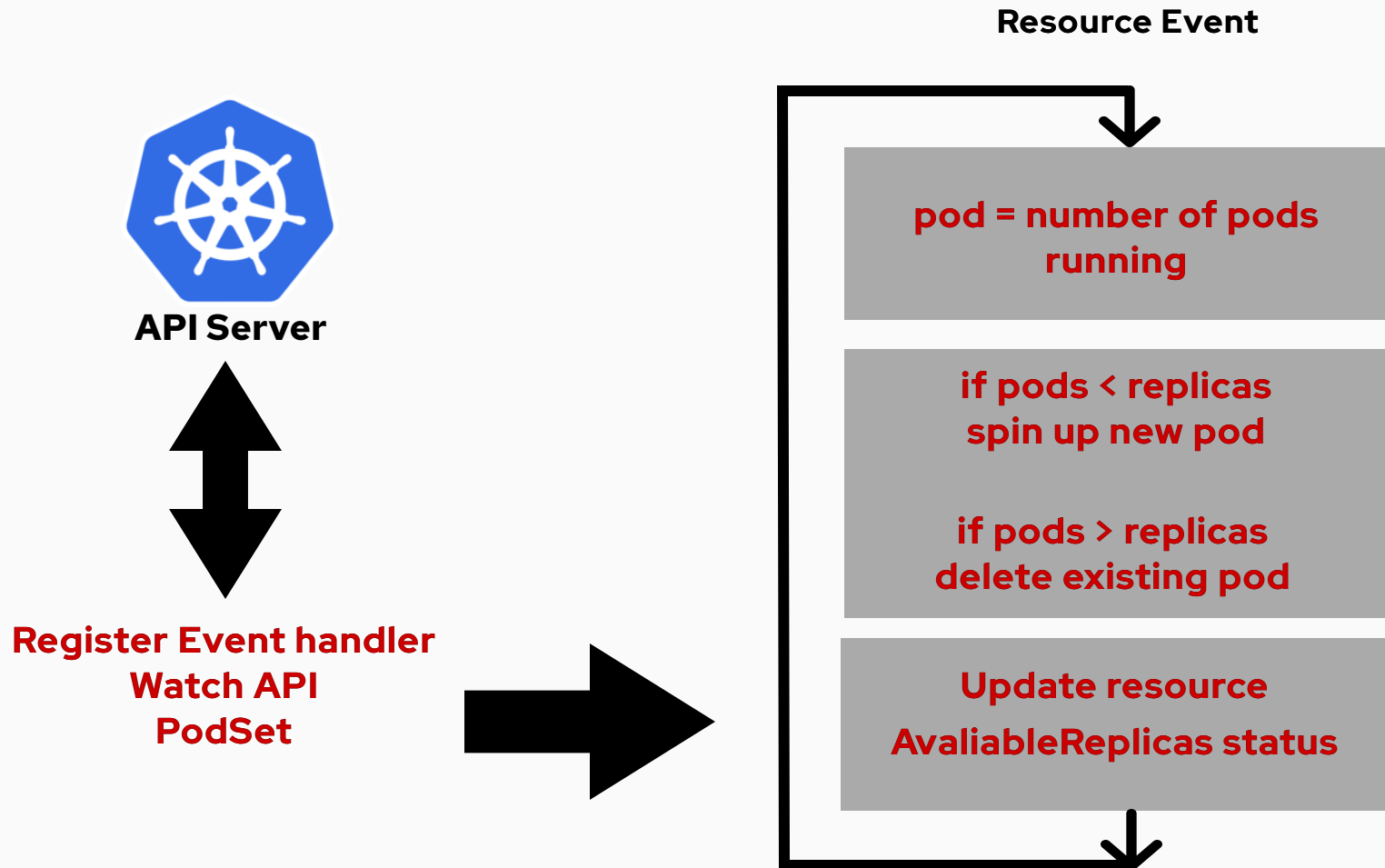


git checkout step-2

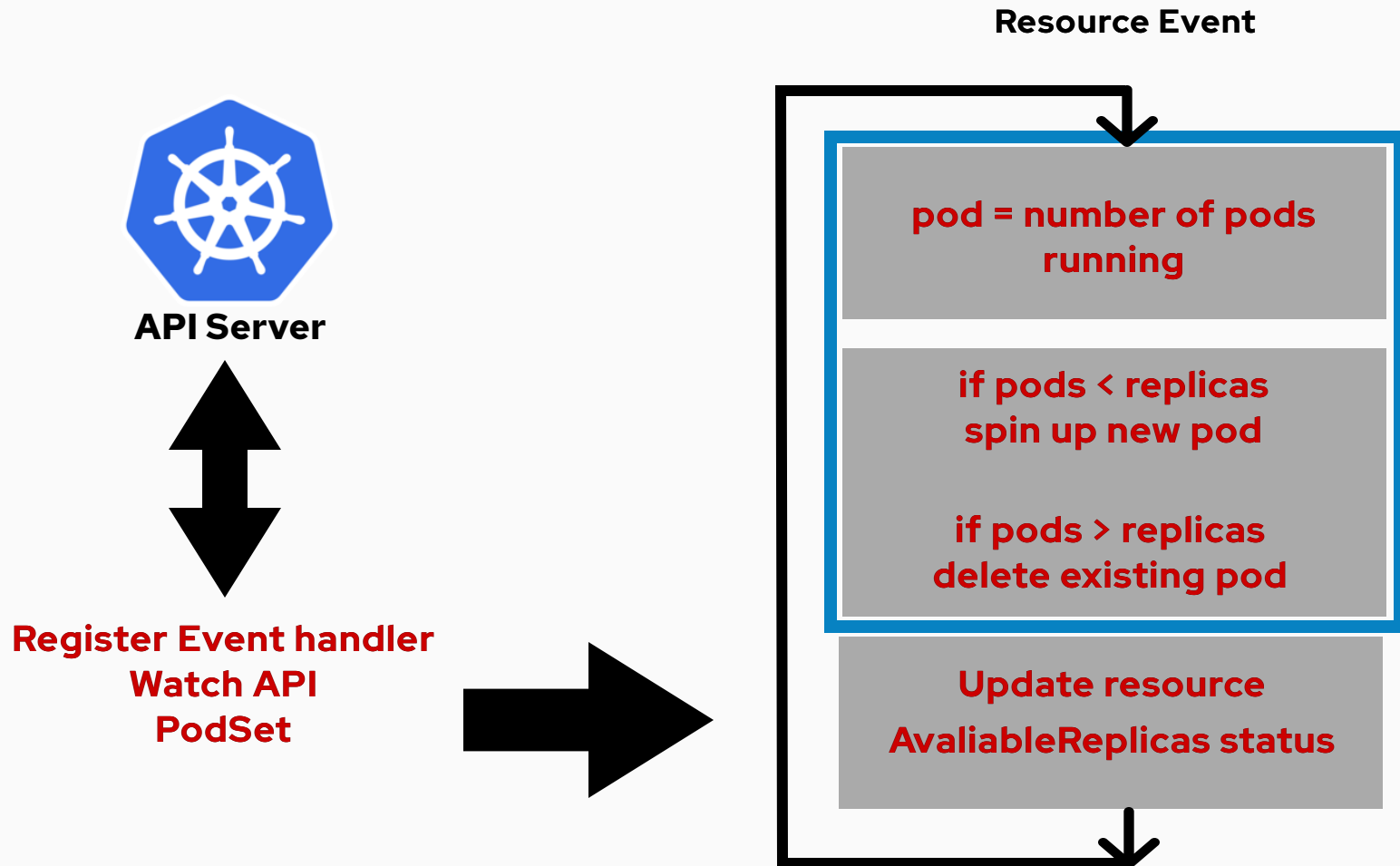
Controller Flow



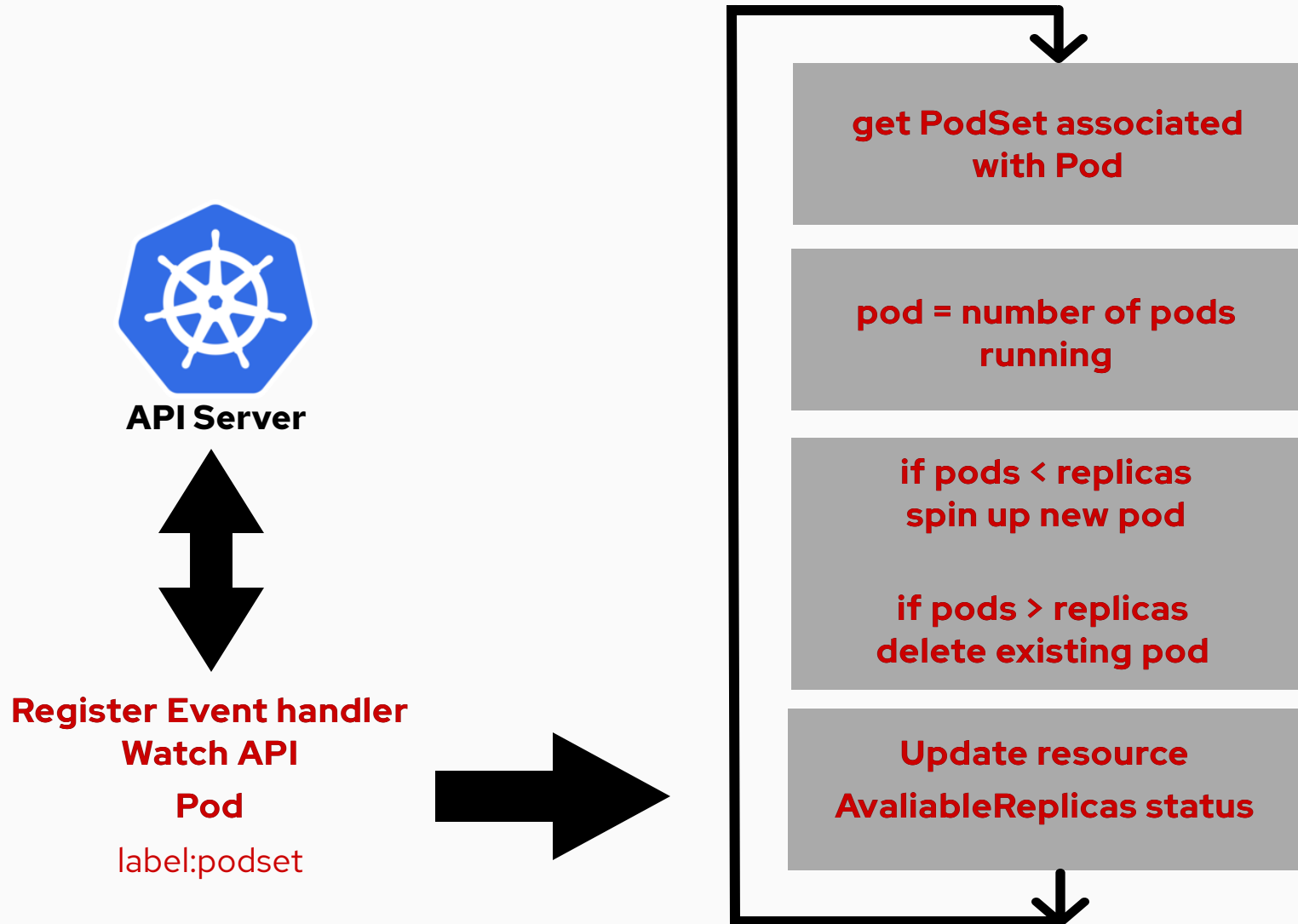
PodSet Controller Flow



Anything Missing?



Lets Watch Pods



Functional "PodSet Controller" Demo



git checkout step-3

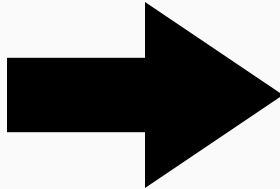
Not enough?



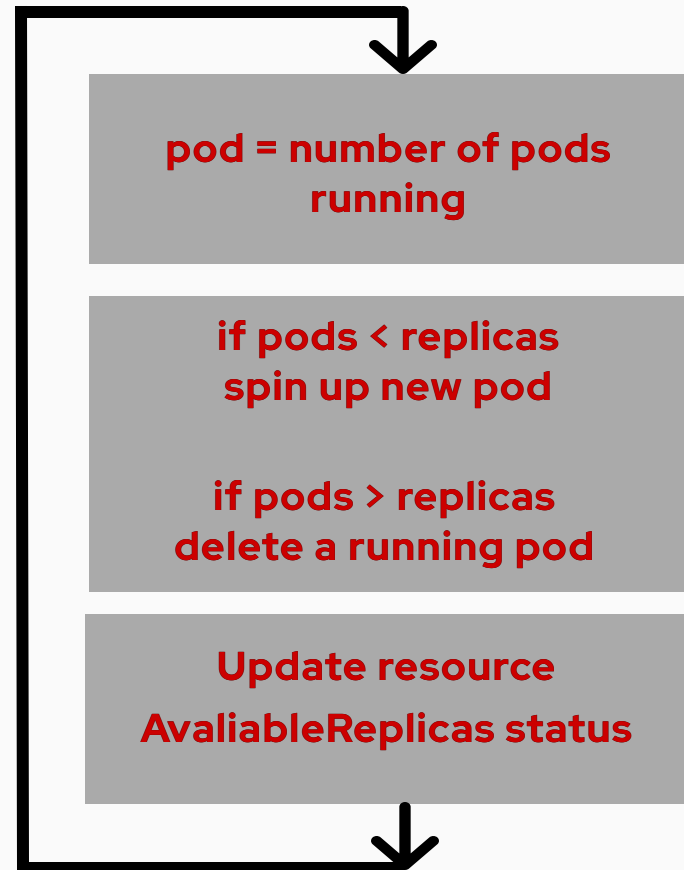
API Server



Register Event handler
Watch API
Pod
PodSet

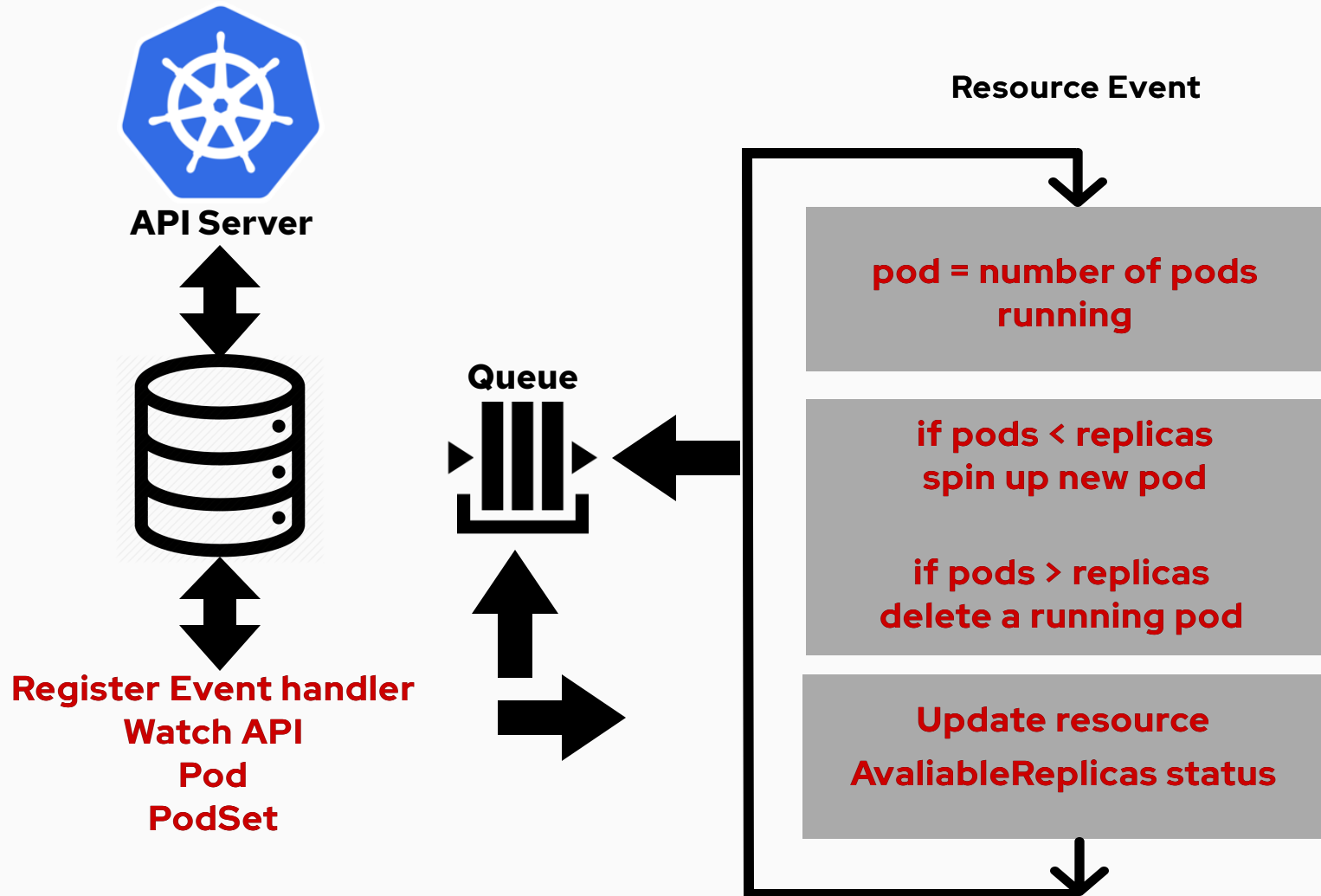


Resource Event

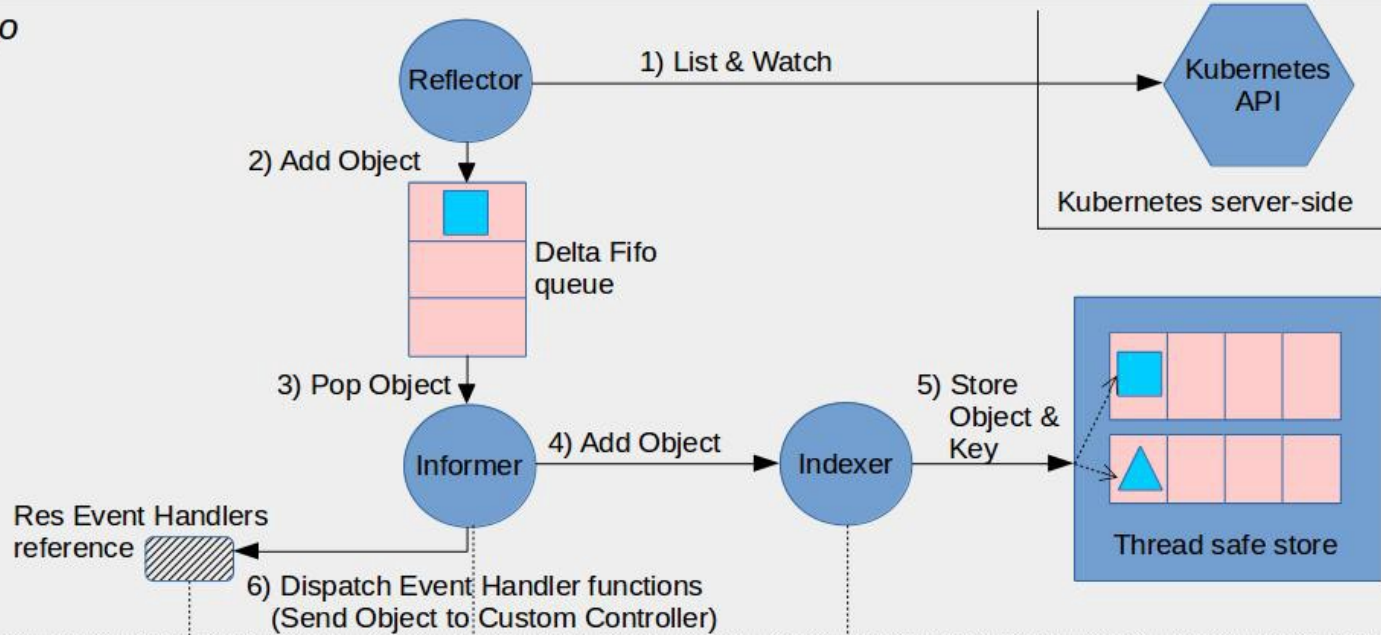




Workqueue



client-go



Custom
Controller

Credit: Devdutta Kulkarni

Generate informers and listers

```
1 $ ~/go/src/k8s.io/code-generator/generate-groups.sh \  
2 "deepcopy,client,informer,lister" \  
3 github.com/hrishin/podset-operator/pkg/client \  
4 github.com/hrishin/podset-operator/pkg/apis \  
5 demo:v1alpha1
```

Final Controller!



git checkout step-4

Still issues?



Controller: the Unicorn way!

Write the same controller using

kubebuilder

or

operator-sdk

Thank you!!