Scavenging for Reusable Code in the Kubernetes Codebase

Kevin Lingerfelt

KubeCon + CloudNativeCon Europe 2019

LINKERD

~~Scavenging for Reusable Code~~

Roadside Picnic

by Arkady and Boris Strugatsky

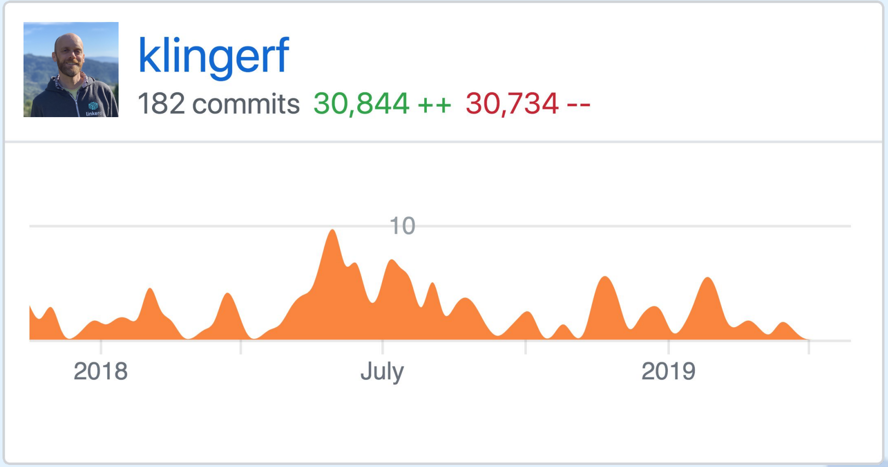
Kevin Lingerfelt

Software Engineer @ Buoyant

@klingerf

@klingerf

slack.linkerd.io: @kl



klingerf

182 commits 30,844 ++ 30,734 --

Scavenging Agenda

Linkerd: My portal to the K8s.io Zone

The K8s.io Zone: Does anybody have a map?

Mission 1: The case of the perplexing command line output

Mission 2: The thrill of the hunt for pods by their IP address

Finale: Lessons learned for future excursions

Scavenging Tools

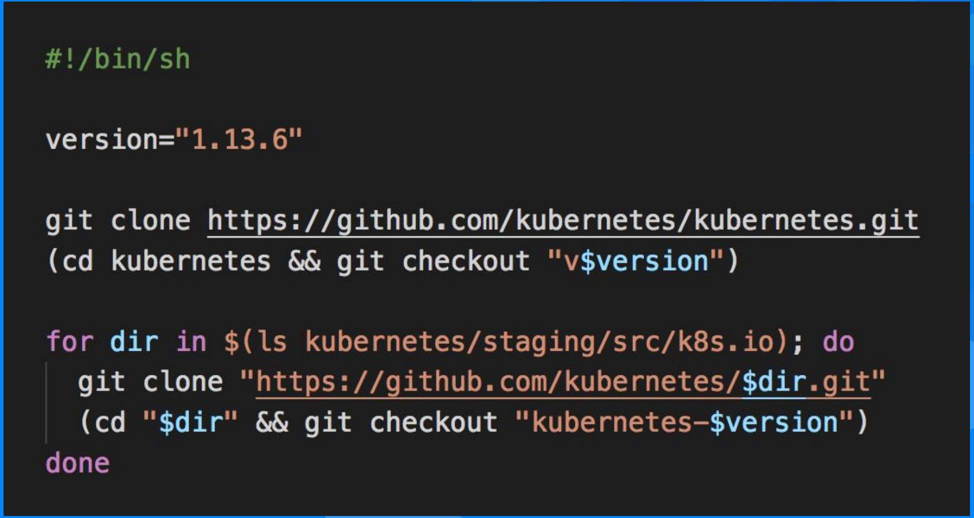
git

grep

google

godoc.org

go run test.go



```
#!/bin/sh

version="1.13.6"

git clone https://github.com/kubernetes/kubernetes.git
(cd kubernetes && git checkout "v$version")

for dir in $(ls kubernetes/staging/src/k8s.io); do
  git clone "https://github.com/kubernetes/$dir.git"
  (cd "$dir" && git checkout "kubernetes-$version")
done
```

Linkerd

My portal to The K8s.io Zone

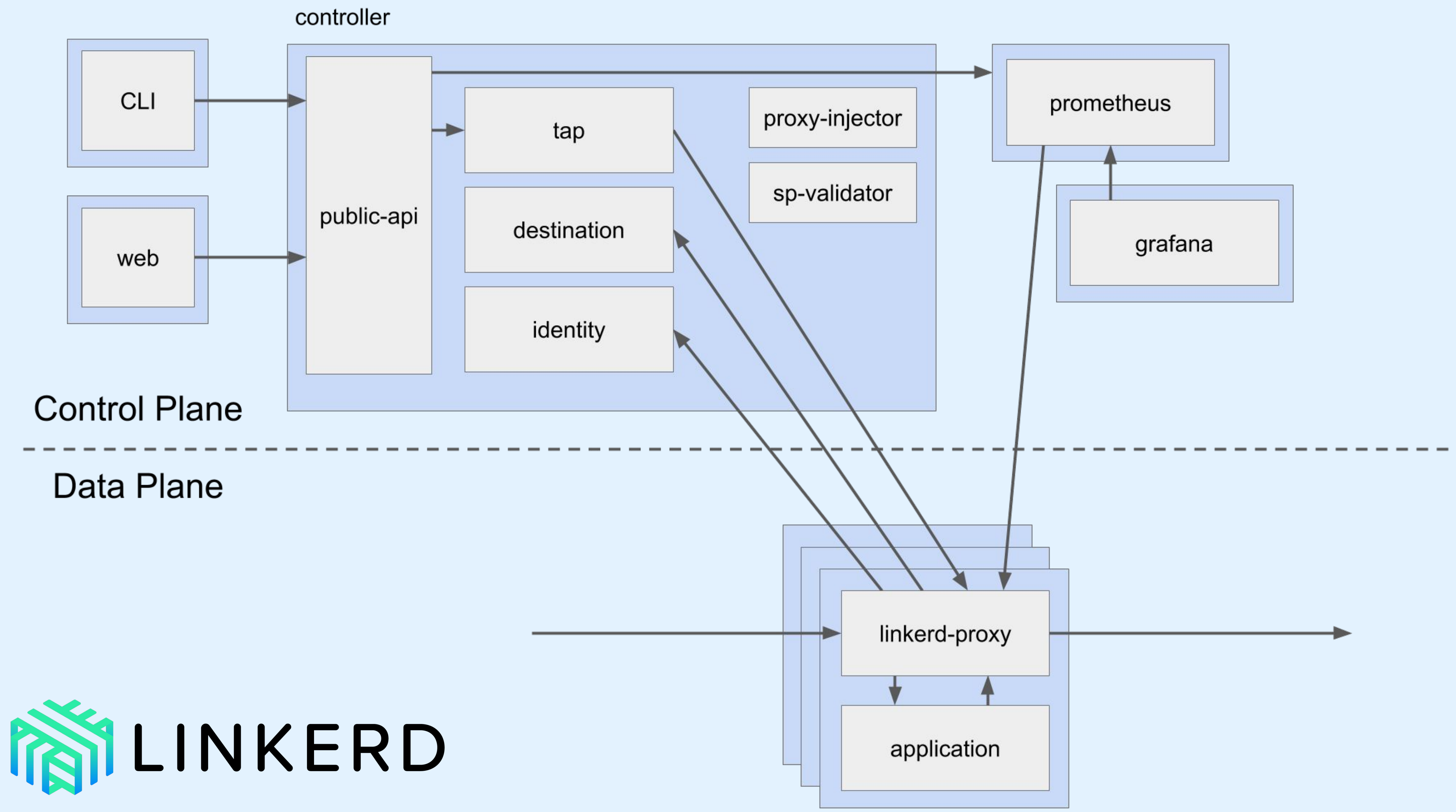
LINKERD

An open source *service mesh* and CNCF member project.

- 🔥 24+ months in production
- 🔥 3,000+ Slack channel members
- 🔥 10,000+ GitHub stars
- 🔥 100+ contributors

CLOUD NATIVE COMPUTING FOUNDATION

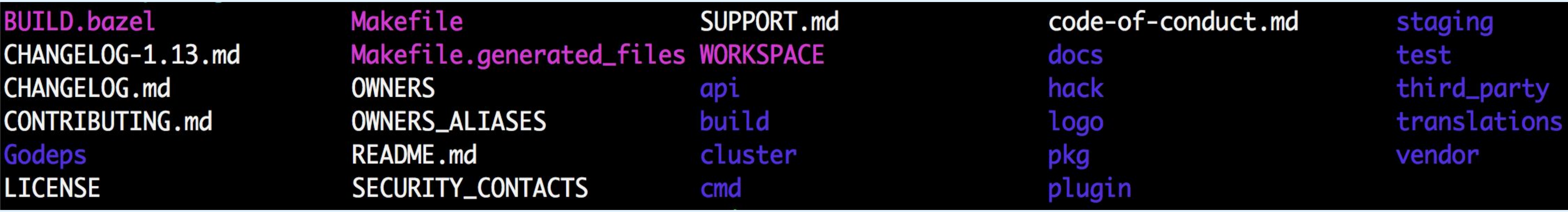
Ask CHASE
GoDaddy KAIROS
STRAVA BIGCOMMERCE
Expedia OfferUp
ebay Cisco webex
COMCAST planet.


controller
CLI
web
public-api
tap
destination
identity
proxy-injector
sp-validator
prometheus
grafana
Control Plane
Data Plane
linkerd-proxy
application
LINKERD

The K8s.io Zone

Does anybody have a map?

➡️ We enter the zone at **`k8s.io/kubernetes`**



```
BUILD.bazel           Makefile                        SUPPORT.md     code-of-conduct.md   staging
CHANGELOG-1.13.md     Makefile.generated_files        WORKSPACE      docs                 test
CHANGELOG.md          OWNERS                          api            hack                 third_party
CONTRIBUTING.md       OWNERS_ALIASES                  build          logo                 translations
Godeps                README.md                       cluster        pkg                  vendor
LICENSE               SECURITY_CONTACTS               cmd            plugin
```

`cmd`: all of the mains

`kube-controller-manager`, **`kubectl`**, **`kube-apiserver`**, etc.

`pkg`: all of the libs

`controller`, **`kubectl`**, **`kubeapiserver`**, etc.

`staging`: all of the projects

↘️ We start to explore the Projects (**`staging`**)

```
k8s.io/api
k8s.io/apiextensions-apiserver
k8s.io/apimachinery
k8s.io/apiserver
k8s.io/cli-runtime
k8s.io/client-go
k8s.io/code-generator
k8s.io/csi-api
k8s.io/kube-aggregator
k8s.io/kube-controller-manager
k8s.io/kube-proxy
k8s.io/kube-scheduler
k8s.io/kubelet
k8s.io/metrics
k8s.io/sample-apiserver
k8s.io/sample-cli-plugin
k8s.io/sample-controller
```

🔀 In the distance, we see even more Projects

`k8s.io/cloud-provider`

`k8s.io/cluster-bootstrap`

`k8s.io/component-base`

`k8s.io/cri-api`

`k8s.io/csi-translation-lib`

`k8s.io/gengo`

`k8s.io/helm`

`k8s.io/klog`

`k8s.io/kube-openapi`

`k8s.io/legacy-cloud-providers`

`k8s.io/node-api`

`k8s.io/repo-infra`

`k8s.io/test-infra`

`k8s.io/utils`

ℹ️ Frightened, we stick with the Projects we know

`k8s.io/cli-runtime`

helpers for creating **`kubectl`** and **`kubectl`**-like commands

`k8s.io/client-go`

code for talking to the Kubernetes API, both internally and externally

`k8s.io/api`

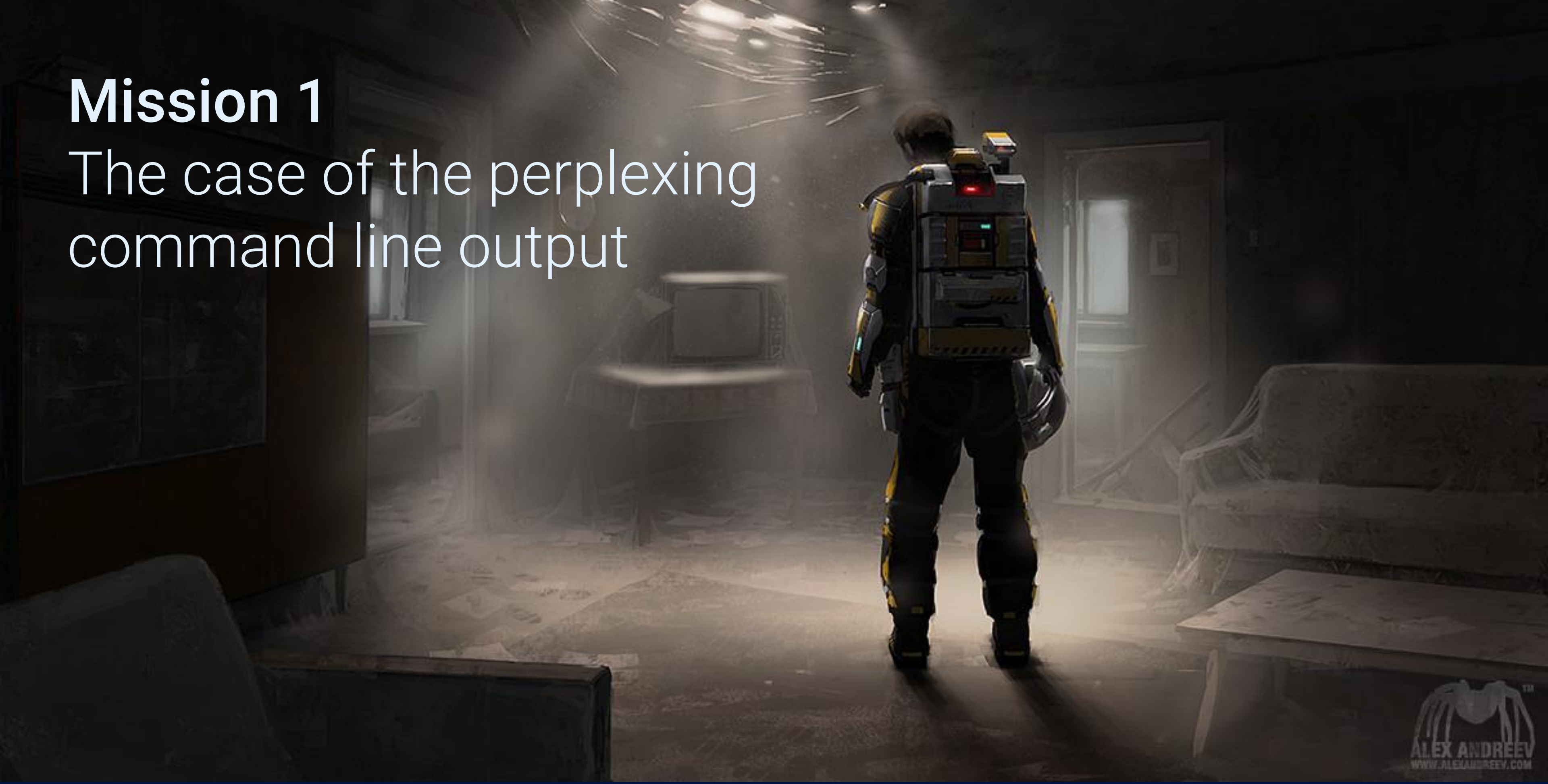
schema for the API itself; lotsa Protobuf

`k8s.io/apimachinery`

libs, interfaces and utilities for work with the API

`k8s.io/helm`, **`k8s.io/klog`**, **`k8s.io/apiextensions-apiserver`**

we use these too but I probably won't have time to talk about them



Mission 1
The case of the perplexing command line output

🙈 An egregious formatting error appears

```
$ linkerd inject hello-world.yml | kubectl apply -f -

deployment "hello" injected
service "hello" skipped
deployment "world" injected
service "world" skipped

deployment.extensions/hello configured
service/hello unchanged
deployment.extensions/world configured
service/world unchanged
```

Can we fix it?

Our mission begins at

➡️ `k8s.io`
➡️ `kubernetes`
➡️ `pkg`
➡️ `kubectl`
➡️ `cmd`

📦 k8s.io/kubernetes/pkg/kubectl/cmd
📄 cmd.go

```go
// NewKubectlCommand creates the `kubectl` command and its nested children.
func NewKubectlCommand(in io.Reader, out, err io.Writer) *cobra.Command {
  // Parent command to which all subcommands are added.
  cmds := &cobra.Command{
    Use:   "kubectl",
    Short: i18n.T("kubectl controls the Kubernetes cluster manager"),
    Long: templates.LongDesc(`
     kubectl controls the Kubernetes cluster manager.

     Find more information at:
           https://kubernetes.io/docs/reference/kubectl/overview/`),
    Run: runHelp,
    ...
```

📦 k8s.io/kubernetes/pkg/kubectl/cmd/apply
📄 apply.go

```go
func NewCmdApply(baseName string, f cmdutil.Factory, s gco.IOStreams) *cobra.Command {
  o := NewApplyOptions(ioStreams)

  cmd := &cobra.Command{
    Use:                   "apply -f FILENAME",
    DisableFlagsInUseLine: true,
    Short:                 i18n.T("Apply a configuration to a resource by filename"),
    Long:                  applyLong,
    Example:               applyExample,
    Run: func(cmd *cobra.Command, args []string) {
      cmdutil.CheckErr(o.Complete(f, cmd))
      cmdutil.CheckErr(validateArgs(cmd, args))
      cmdutil.CheckErr(validatePruneAll(o.Prune, o.All, o.Selector))
      cmdutil.CheckErr(o.Run())
    },
  }
```

📦 k8s.io/kubernetes/pkg/kubectl/cmd/util
📄 factory.go

```go
// Factory provides abstractions that allow the Kubectl command to be extended
// across multiple types of resources and different API sets.
// The rings are here for a reason. In order for composers to be able to provide
// alternative factory implementations they need to provide low level pieces of
// *certain* functions so that when the factory calls back into itself it uses
// the custom version of the function. Rather than try to enumerate everything
// that someone would want to override we split the factory into rings, where
// each ring can depend on methods in an earlier ring, but cannot depend upon
// peer methods in its own ring.
// TODO: make the functions interfaces
// TODO: pass the various interfaces on the factory directly into the command
// constructors (so the commands are decoupled from the factory).
type Factory interface {
  genericclioptions.RESTClientGetter
  ...
```

📦 k8s.io/kubernetes/pkg/kubectl/cmd/apply
📄 apply.go

```go
func NewCmdApply(baseName string, f cmdutil.Factory, s gco.IOStreams) *cobra.Command {
  o := NewApplyOptions(ioStreams)

  cmd := &cobra.Command{
    Use:                   "apply -f FILENAME",
    DisableFlagsInUseLine: true,
    Short:                 i18n.T("Apply a configuration to a resource by filename"),
    Long:                  applyLong,
    Example:               applyExample,
    Run: func(cmd *cobra.Command, args []string) {
      cmdutil.CheckErr(o.Complete(f, cmd))
      cmdutil.CheckErr(validateArgs(cmd, args))
      cmdutil.CheckErr(validatePruneAll(o.Prune, o.All, o.Selector))
      cmdutil.CheckErr(o.Run())
    },
  }
```

📦 k8s.io/kubernetes/pkg/kubectl/cmd/apply
📄 apply.go

```go
func NewApplyOptions(ioStreams genericclioptions.IOStreams) *ApplyOptions {
  return &ApplyOptions{
    RecordFlags: genericclioptions.NewRecordFlags(),
    DeleteFlags: delete.NewDeleteFlags("that contains the configuration to apply"),
    PrintFlags: genericclioptions.NewPrintFlags("created").
      WithTypeSetter(scheme.Scheme),

    Overwrite:    true,
    OpenApiPatch: true,

    Recorder: genericclioptions.NoopRecorder{},

    IOStreams: ioStreams,
  }
}
```

📦 k8s.io/kubernetes/pkg/kubectl/cmd/apply
📄 apply.go

```go
func (o *ApplyOptions) Run() error {
  ...

  err = r.Visit(func(info *resource.Info, err error) error {
    ...

    printer, err := o.ToPrinter("configured")
    if err != nil {
      return err
    }
    return printer.PrintObj(info.Object, o.Out)
  })
  ...

}
```

📦 k8s.io/kubernetes/pkg/kubectl/cmd/apply
📄 apply.go

```go
func (o *ApplyOptions) Complete(f cmdutil.Factory, cmd *cobra.Command) error {
  ...

  // allow for a success message operation to be specified at print time
  o.ToPrinter = func(operation string) (printers.ResourcePrinter, error) {
    o.PrintFlags.NamePrintFlags.Operation = operation
    if o.DryRun {
      o.PrintFlags.Complete("%s (dry run)")
    }
    if o.ServerDryRun {
      o.PrintFlags.Complete("%s (server dry run)")
    }
    return o.PrintFlags.ToPrinter()
  }
  ...

}
```

📦 k8s.io/kubernetes/pkg/kubectl/cmd/apply

📄 apply.go

```go
func NewApplyOptions(ioStreams genericclioptions.IOStreams) *ApplyOptions {
  return &ApplyOptions{
    RecordFlags: genericclioptions.NewRecordFlags(),
    DeleteFlags: delete.NewDeleteFlags("that contains the configuration to apply"),
    PrintFlags: genericclioptions.NewPrintFlags("created").
      WithTypeSetter(scheme.Scheme),

    Overwrite:    true,
    OpenApiPatch: true,

    Recorder: genericclioptions.NoopRecorder{},

    IOStreams: ioStreams,
  }
}
```

📦 k8s.io/cli-runtime/pkg/genericclioptions
📄 print_flags.go

```go
func NewPrintFlags(operation string) *PrintFlags {
  outputFormat := ""

  return &PrintFlags{
    OutputFormat: &outputFormat,

    JSONYamlPrintFlags:   NewJSONYamlPrintFlags(),
    NamePrintFlags:       NewNamePrintFlags(operation),
    TemplatePrinterFlags: NewKubeTemplatePrintFlags(),
  }
}
```

📦 k8s.io/cli-runtime/pkg/genericclioptions
📄 print_flags.go

```go
func (f *PrintFlags) ToPrinter() (printers.ResourcePrinter, error) {
  outputFormat := ""
  if f.OutputFormat != nil {
    outputFormat = *f.OutputFormat
  }
  ...

  if f.NamePrintFlags != nil {
    p, err := f.NamePrintFlags.ToPrinter(outputFormat)
    if !IsNoCompatiblePrinterError(err) {
      return f.TypeSetterPrinter.WrapToPrinter(p, err)
    }
  }
  ...

}
```

📦 k8s.io/cli-runtime/pkg/genericclioptions
📄 name_flags.go

```go
// ToPrinter receives an outputFmt and returns a printer capable of
// handling --output=name printing.
// Returns false if the specified outputFmt does not match a supported format.
// Supported format types can be found in pkg/printers/printers.go
func (f *NamePrintFlags) ToPrinter(outputFmt string) (printers.ResourcePrinter, error) {
  namePrinter := &printers.NamePrinter{
    Operation: f.Operation,
  }

  outputFmt = strings.ToLower(outputFmt)
  switch outputFmt {
  case "name":
    namePrinter.ShortOutput = true
    fallthrough
  case "":
    return namePrinter, nil
  ...
```

📦 k8s.io/cli-runtime/pkg/genericclioptions/printers
📄 name.go

```go
// NamePrinter is an implementation of ResourcePrinter which outputs
// "resource/name" pair of an object.
type NamePrinter struct {
  // ShortOutput indicates whether an operation should be
  // printed along side the "resource/name" pair for an object.
  ShortOutput bool
  // Operation describes the name of the action that
  // took place on an object, to be included in the
  // finalized "successful" message.
  Operation string
}
```

📦 k8s.io/cli-runtime/pkg/genericclioptions/printers

📄 name.go

```go
// PrintObj is an implementation of ResourcePrinter.PrintObj which decodes the
// object and print "resource/name" pair. If the object is a List, print all
// items in it.
func (p *NamePrinter) PrintObj(obj runtime.Object, w io.Writer) error {
  ...

  return printObj(w, name, p.Operation, p.ShortOutput, GetObjectGroupKind(obj))
}
```

📦 k8s.io/cli-runtime/pkg/genericclioptions/printers
📄 name.go

```go
func printObj(w io.Writer, name, op string, short bool, gk schema.GroupKind) error {
  ...

  if len(gk.Group) == 0 {
    fmt.Fprintf(w, "%s/%s%s\n", strings.ToLower(gk.Kind), name, op)
    return nil
  }

  fmt.Fprintf(w, "%s.%s/%s%s\n", strings.ToLower(gk.Kind), gk.Group, name, op)
  return nil
}
```

We found it!

🔧 And sure enough, we can fix it

```
$ go run cli/main.go inject hello-world.yml | kubectl apply -f -

deployment.extensions/hello injected
service/hello skipped
deployment.extensions/world injected
service/world skipped

deployment.extensions/hello configured
service/hello unchanged
deployment.extensions/world configured
service/world unchanged
```

💎 Bonus loot

📦 k8s.io/cli-runtime/pkg/genericclioptions
📄 config_flags.go

```go
// ToRESTConfig implements RESTClientGetter.
// Returns a REST client configuration based on a provided path
// to a .kubeconfig file, loading rules, and config flag overrides.
// Expects the AddFlags method to have been called.
func (f *ConfigFlags) ToRESTConfig() (*rest.Config, error) {
  return f.ToRawKubeConfigLoader().ClientConfig()
}

// AddFlags binds client configuration flags to a given flagset
func (f *ConfigFlags) AddFlags(flags *pflag.FlagSet) {
  if f.KubeConfig != nil {
    flags.StringVar(f.KubeConfig, "kubeconfig", *f.KubeConfig,
      "Path to the kubeconfig file to use for CLI requests.")
  }
  ...

}
```

Can we use cli-runtime to talk to ou cluster?

Mission 2

The thrill of the hunt for pods by their IP address

Our mission begins at

➡️ **`k8s.io`**
➡️ **`client-go`**
➡️ **`examples`**
➡️ **`out-of-cluster-client-configuration`**

📦 k8s.io/client-go/examples/out-of-cluster-client-config
📄 main.go

```go
func main() {
  ...
  // use the current context in kubeconfig
  config, err := clientcmd.BuildConfigFromFlags("", *kubeconfig)
  checkErr(err)

  // create the clientset
  clientset, err := kubernetes.NewForConfig(config)
  checkErr(err)

  for {
    pods, err := clientset.CoreV1().Pods("").List(metav1.ListOptions{})
    checkErr(err)

    fmt.Printf("There are %d pods in the cluster\n", len(pods.Items))
    ...
  }
```

1️⃣ The list all pods approach

1️⃣ The list all pods approach

```go
func main() {
  ...
  clientset, err := kubernetes.NewForConfig(config)
  checkErr(err)

  pods, err := clientset.CoreV1().Pods("").List(metav1.ListOptions{})
  checkErr(err)

  for _, pod := range pods.Items {
    if pod.Status.PodIP == ip {
      fmt.Printf("%s\t%s\n", pod.Namespace, pod.Name)
      return
    }
  }
  fmt.Println("pod not found")
}
```

2️⃣ The watch all pods approach

2️⃣ The watch all pods approach

```go
type podIndex struct {
  index map[string]*corev1.Pod
  sync.RWMutex
}

func (i *podIndex) set(k string, v *corev1.Pod) {
  i.Lock()
  defer i.Unlock()
  i.index[k] = v
}

func (i *podIndex) get(k string) (*corev1.Pod, bool) {
  i.RLock()
  defer i.RUnlock()
  v, ok := i.index[k]
  return v, ok
}
```

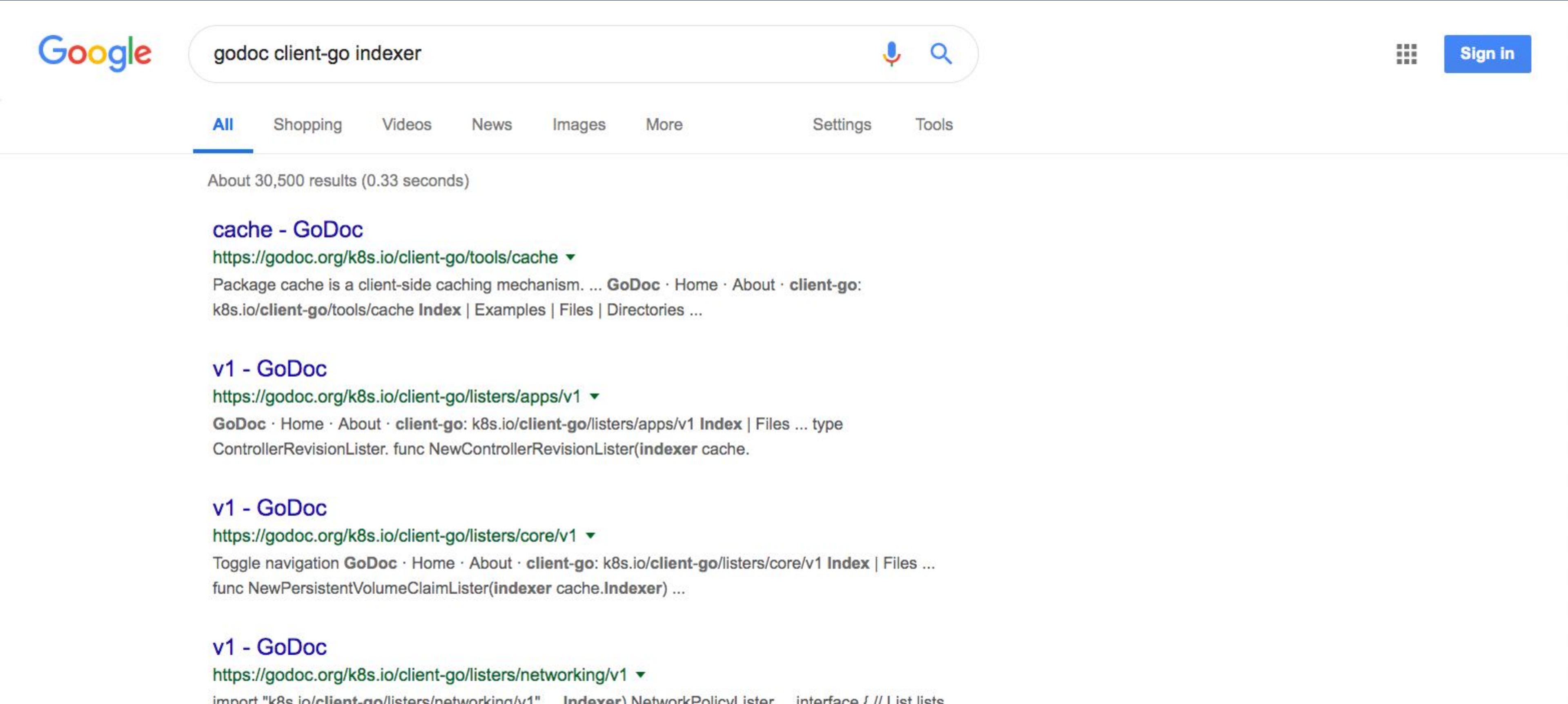
2️⃣ The watch all pods approach

```go
podsByIP := podIndex{index: map[string]*corev1.Pod{}}
watch, err := clientset.CoreV1().Pods("").Watch(metav1.ListOptions{})
checkErr(err)

go func() {
  for event := range watch.ResultChan() {
    pod := event.Object.(*corev1.Pod)
    podsByIP.set(pod.Status.PodIP, pod)
  }
}()
time.Sleep(time.Second)

if pod, ok := podsByIP.get(ip); ok {
  fmt.Printf("%s\t%s\n", pod.Namespace, pod.Name)
  return
}
fmt.Println("pod not found")
```

⌛ Meanwhile...



godoc client-go indexer

Sign in

All Shopping Videos News Images More Settings Tools

About 30,500 results (0.33 seconds)

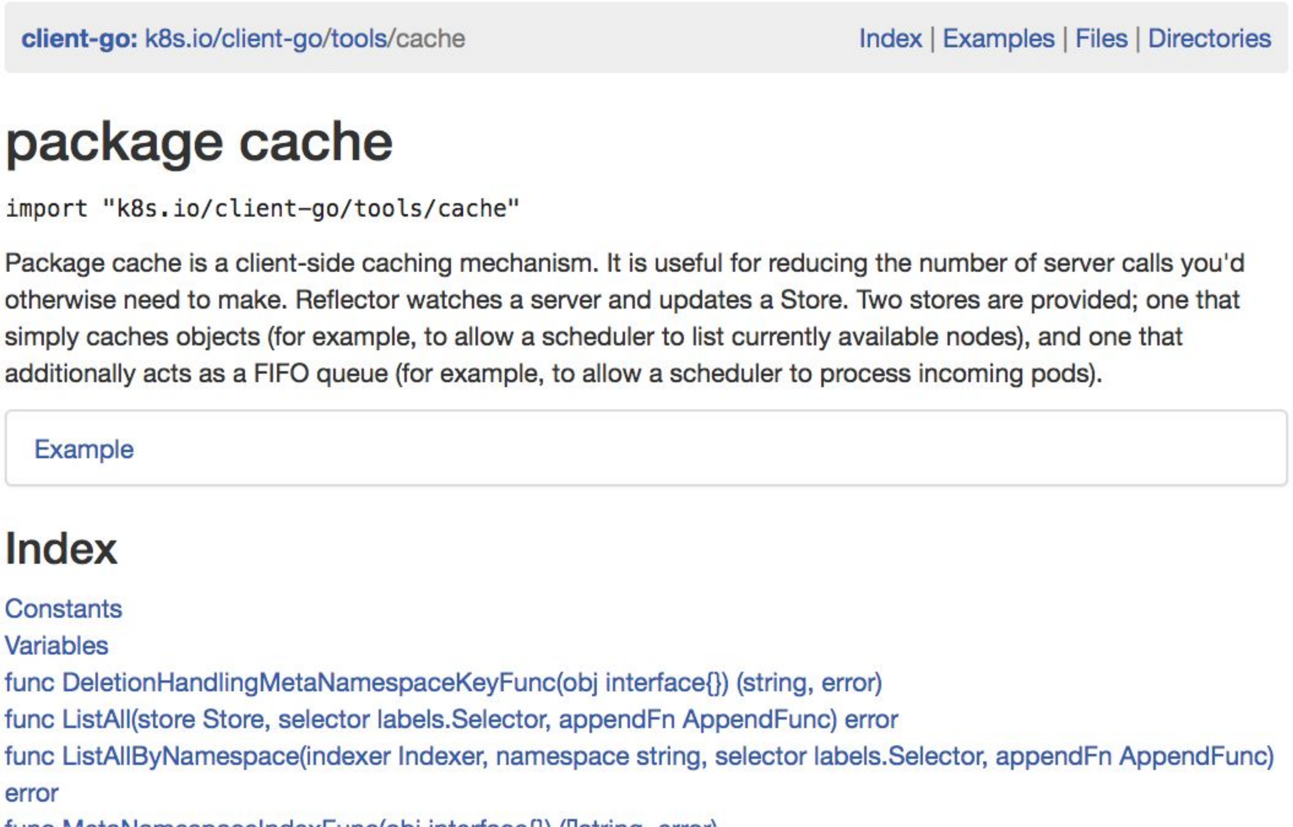
cache - GoDoc
https://godoc.org/k8s.io/client-go/tools/cache ▾
Package cache is a client-side caching mechanism. ... **GoDoc** · Home · About · **client-go**: k8s.io/**client-go**/tools/cache **Index** | Examples | Files | Directories ...

v1 - GoDoc
https://godoc.org/k8s.io/client-go/listers/apps/v1 ▾
GoDoc · Home · About · **client-go**: k8s.io/**client-go**/listers/apps/v1 **Index** | Files ... type ControllerRevisionLister. func NewControllerRevisionLister(**indexer** cache.

v1 - GoDoc
https://godoc.org/k8s.io/client-go/listers/core/v1 ▾
Toggle navigation **GoDoc** · Home · About · **client-go**: k8s.io/**client-go**/listers/core/v1 **Index** | Files ... func NewPersistentVolumeClaimLister(**indexer** cache.**Indexer**) ...

v1 - GoDoc
https://godoc.org/k8s.io/client-go/listers/networking/v1 ▾
import "k8s.io/**client-go**/listers/networking/v1" ... **Indexer**) NetworkPolicyLister ... interface { // List lists

📁 godoc.org/k8s.io/client-go/tools/cache



client-go: k8s.io/client-go/tools/cache   Index | Examples | Files | Directories

package cache

```
import "k8s.io/client-go/tools/cache"
```

Package cache is a client-side caching mechanism. It is useful for reducing the number of server calls you'd otherwise need to make. Reflector watches a server and updates a Store. Two stores are provided; one that simply caches objects (for example, to allow a scheduler to list currently available nodes), and one that additionally acts as a FIFO queue (for example, to allow a scheduler to process incoming pods).

Example

Index

Constants
Variables
func DeletionHandlingMetaNamespaceKeyFunc(obj interface{}) (string, error)
func ListAll(store Store, selector labels.Selector, appendFn AppendFunc) error
func ListAllByNamespace(indexer Indexer, namespace string, selector labels.Selector, appendFn AppendFunc) error
func MetaNamespaceIndexFunc(obj interface{}) ([]string, error)

📁 godoc.org/k8s.io/client-go/tools/cache

Package **cache** is a client-side caching mechanism.

Store is a generic object storage interface.

Queue is exactly like a **Store**, but has a Pop() method too.

Heap is a thread-safe producer/consumer queue that implements a heap data structure.

Reflector watches a specified resource and causes all changes to be reflected in a **Store**.

Config contains all the settings for a **Controller**.

Controller [has no documentation]

Indexer is a storage interface that lets you list objects using multiple indexing functions.

NewIndexer returns an **Indexer** implemented simply with a map and a lock.

🔍 grep -rl 'cache\.NewIndexer('

```
$ grep -rl 'cache\.NewIndexer(' . --exclude '*_test.go'
./kubernetes/pkg/controller/volume/persistentvolume/index.go
./kubernetes/pkg/controller/volume/persistentvolume/scheduler_assume_cache.go
./kubernetes/pkg/kubelet/kubelet.go
```

📦 k8s.io/kubernetes/pkg/kubelet
📄 kubelet.go

```go
serviceIndexer := cache.NewIndexer(
  cache.MetaNamespaceKeyFunc,
  cache.Indexers{cache.NamespaceIndex: cache.MetaNamespaceIndexFunc},
)

if kubeDeps.KubeClient != nil {
  serviceLW := cache.NewListWatchFromClient(
    kubeDeps.KubeClient.CoreV1().RESTClient(),
    "services",
    metav1.NamespaceAll,
    fields.Everything(),
  )
  r := cache.NewReflector(serviceLW, &v1.Service{}, serviceIndexer, 0)
  go r.Run(wait.NeverStop)
}
```

3️⃣ The cache indexer approach

3️⃣ The cache indexer approach

```go
func podIPIndexFunc(obj interface{}) ([]string, error) {
  pod := obj.(*corev1.Pod)
  return []string{pod.Status.PodIP}, nil
}
```

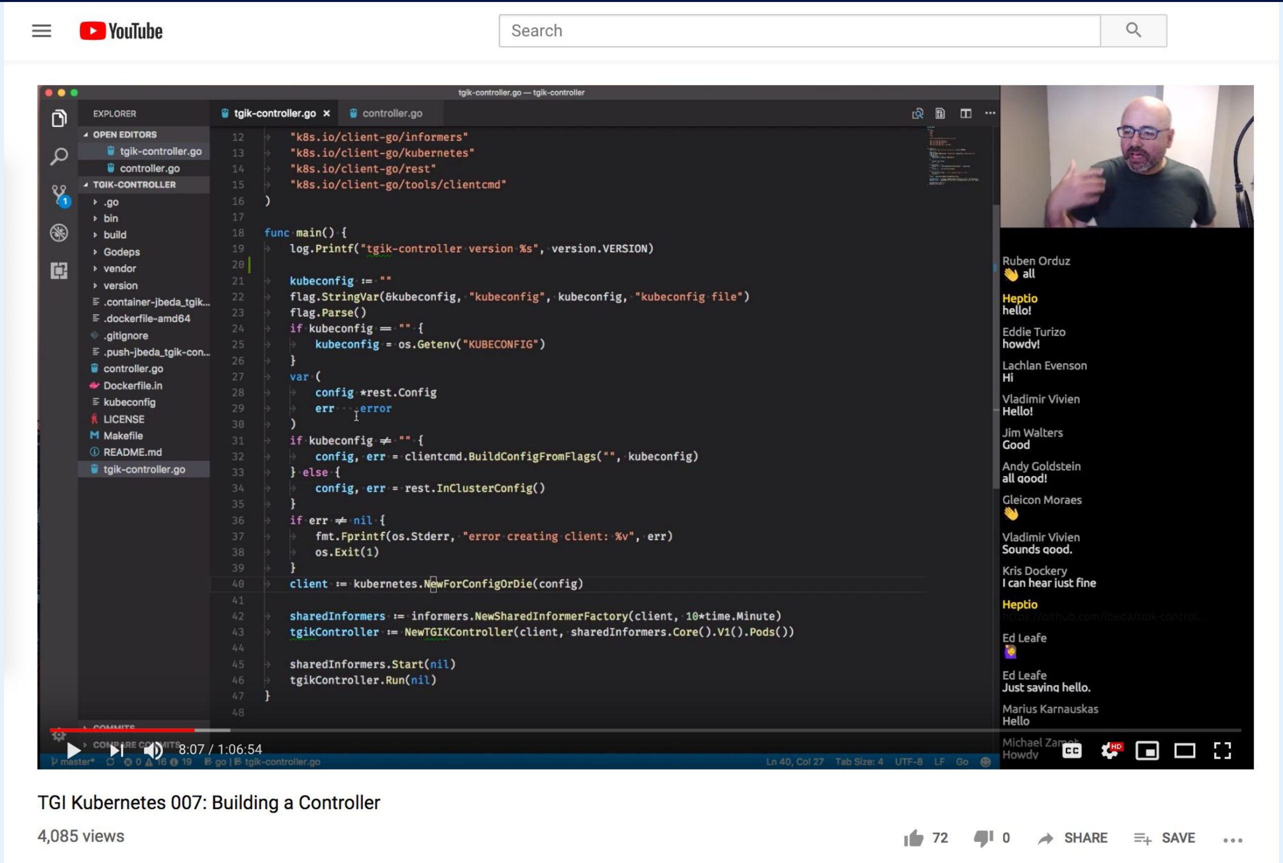
3️⃣ The cache indexer approach

```go
indexer := cache.NewIndexer(cache.MetaNamespaceKeyFunc,
  cache.Indexers{"ip": podIPIndexFunc})
lw := cache.NewListWatchFromClient(clientset.CoreV1().RESTClient(),
  "pods", metav1.NamespaceAll, fields.Everything())
reflector := cache.NewReflector(lw, &corev1.Pod{}, indexer, 10*time.Minute)

go reflector.Run(wait.NeverStop)
for range time.Tick(100 * time.Millisecond) {
  if reflector.LastSyncResourceVersion() != "" {
    break
  }
}

if items, err := indexer.ByIndex("ip", ip); err == nil {
  for _, item := range items {
    pod := item.(*corev1.Pod)
    fmt.Printf("%s\t%s\n", pod.Namespace, pod.Name)
```

⌛ Meanwhile...



```go
	"k8s.io/client-go/informers"
	"k8s.io/client-go/kubernetes"
	"k8s.io/client-go/rest"
	"k8s.io/client-go/tools/clientcmd"
)

func main() {
	log.Printf("tgik-controller version %s", version.VERSION)

	kubeconfig := ""
	flag.StringVar(&kubeconfig, "kubeconfig", kubeconfig, "kubeconfig file")
	flag.Parse()
	if kubeconfig == "" {
		kubeconfig = os.Getenv("KUBECONFIG")
	}
	var (
		config *rest.Config
		err    error
	)
	if kubeconfig != "" {
		config, err = clientcmd.BuildConfigFromFlags("", kubeconfig)
	} else {
		config, err = rest.InClusterConfig()
	}
	if err != nil {
		fmt.Fprintf(os.Stderr, "error creating client: %v", err)
		os.Exit(1)
	}
	client := kubernetes.NewForConfigOrDie(config)

	sharedInformers := informers.NewSharedInformerFactory(client, 10*time.Minute)
	tgikController := NewTGIKController(client, sharedInformers.Core().V1().Pods())

	sharedInformers.Start(nil)
	tgikController.Run(nil)
}
```

Ruben Orduz
👋 all

Heptio
hello!

Eddie Turizo
howdy!

Lachlan Evenson
Hi

Vladimir Vivien
Hello!

Jim Walters
Good

Andy Goldstein
all good!

Gleicon Moraes
👋

Vladimir Vivien
Sounds good.

Kris Dockery
I can hear just fine

Heptio
https://github.com/jbeda/tgik-control...

Ed Leafe
🙋

Ed Leafe
Just saying hello.

Marius Karnauskas
Hello

Michael Zamot
Howdy

TGI Kubernetes 007: Building a Controller

4,085 views

📦 github.com/jbeda/tgik-controller
📄 tgik-controller.go

```go
func main() {
  ...
  client := kubernetes.NewForConfigOrDie(config)

  sharedInformers := informers.NewSharedInformerFactory(client, 10*time.Minute)
  tgikController := NewTGIKController(
    client,
    sharedInformers.Core().V1().Secrets(),
    sharedInformers.Core().V1().Namespaces(),
  )

  sharedInformers.Start(nil)
  tgikController.Run(nil)
}
```

4️⃣ The shared informer approach

4️⃣ The shared informer approach

```go
clientset := kubernetes.NewForConfigOrDie(config)

sharedInformers := informers.NewSharedInformerFactory(clientset, 10*time.Minute)
podInformer := sharedInformers.Core().V1().Pods().Informer()
podInformer.AddIndexers(cache.Indexers{"ip": podIPIndexFunc})

sharedInformers.Start(wait.NeverStop)
cache.WaitForCacheSync(wait.NeverStop, podInformer.HasSynced)

if items, err := podInformer.GetIndexer().ByIndex("ip", ip); err == nil {
  for _, item := range items {
    pod := item.(*corev1.Pod)
    fmt.Printf("%s\t%s\n", pod.Namespace, pod.Name)
    return
  }
}
fmt.Println("pod not found")
```

Winner!

💎 Bonus loot

Restricted permission mode

```diff
  clientset := kubernetes.NewForConfigOrDie(config)

- sharedInformers := informers.NewSharedInformerFactory(clientset, 10*time.Minute)
+ sharedInformers := informers.NewSharedInformerFactoryWithOptions(
+   clientset,
+   10*time.Minute,
+   informers.WithNamespace("linkerd"),
+ )
  podInformer := sharedInformers.Core().V1().Pods().Informer()
  podInformer.AddIndexers(cache.Indexers{"ip": podIPIndexFunc})
```

Straightforward test fixtures

```
func main() {
- var ip string
- if len(os.Args) > 1 {
-   ip = os.Args[1]
- }
-
- configFile := filepath.Join(os.Getenv("HOME"), ".kube", "config")
- config, err := clientcmd.BuildConfigFromFlags("", configFile)
- checkErr(err)
+ ip := "10.1.16.65"
+ pod := &corev1.Pod{Status: corev1.PodStatus{PodIP: ip},
+   ObjectMeta: metav1.ObjectMeta{Name: "my-pod", Namespace: "default"}}

- clientset := kubernetes.NewForConfigOrDie(config)
+ clientset := fake.NewSimpleClientset(pod)

  sharedInformers := informers.NewSharedInformerFactory(clientset, 10*time.Minute)
```

Finale
Lessons learned for future excursions

Scavenging Summary

- Gathered intelligence on the Linkerd codebase
- Reconnoitered the K8s.io Zone, found our most likely entry points
- Completed a successful mission to cli-runtime, located a name printer
- Executed a flawed but ultimately successful retrieval of shared informers
- Used some tools of the trade: git, grep, google, godoc.org, “go run test.go”

For future missions...

Research in advance

Check blogs and talks before diving in

Reusable examples

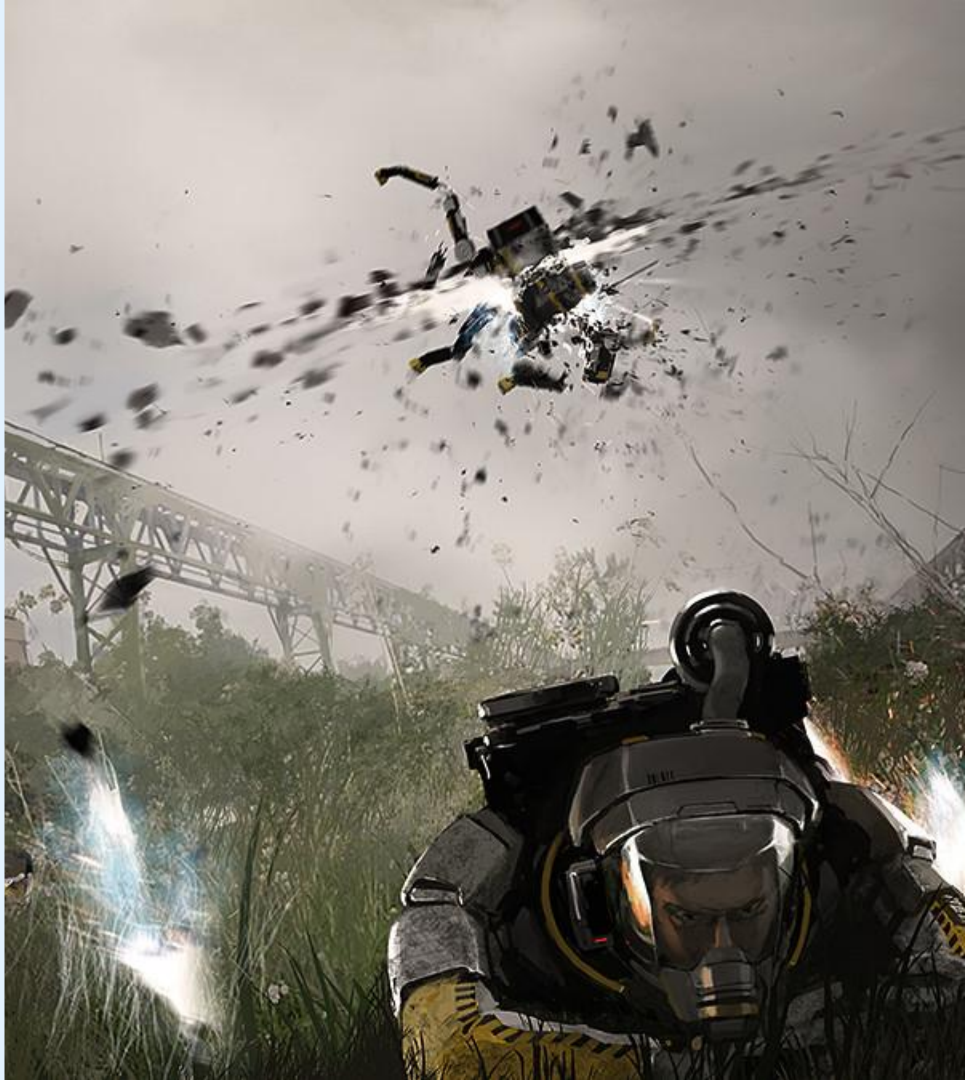
All staging projects could ship with a standalone directory of examples

Higher-level documentation

Godoc can't quite capture how all of the individual pieces fit together

Smaller packages

Break up some of the really big ones into more manageable chunks



LINKERD

Happy scavenging, brave developers.

github.com/linkerd slack.linkerd.io @linkerd

FROM YOUR FRIENDS AT

BUOYANT