KubeCon | CloudNativeCon

North America 2018

SIG CLI Intro

Sean Sullivan
SIG CLI Co-Chair
Github: @seans3
Email: seans@google.com

December 12, 2018

Agenda

What are we going to talk about?

- SIG CLI sub-projects
- Recent major initiatives
- How to get involved
- Links and References
- Question and Answer

What is the SIG CLI ?

Who are we ?

From the SIG CLI charter: The Command Line Interface SIG (SIG CLI) is responsible for `kubectl` and related tools. This group focuses on general purpose command line tools and libraries to interface with Kubernetes API's.

Subprojects:

- `kubectl`
- `cli-sdk`
- `kustomize`

subproject: kubectl

`kubectl` is the standard command line client for running commands against Kubernetes clusters.

```
kubectl [command] [TYPE] [NAME] [flags]
```

Examples:

```
$ kubectl create -f pod.yaml
$ kubectl get pod pod1
$ kubectl apply -f deployment.yaml
```

`kubectl` code locations

- kubectl main
- kubectl libraries

subproject: CLI SDK

`cli-runtime`: Set of helpers for creating `kubectl` commands, as well as kubectl plugins.

`sample-cli-plugin`: This repository implements a single `kubectl` plugin for switching the namespace that the current KUBECONFIG context points to.

`cli-runtime` repository: https://github.com/kubernetes/cli-runtime

`sample cli plugin:` https://github.com/kubernetes/sample-cli-plugin

subproject: kustomize

`kustomize` lives in the space of resource configuration tools (e.g. helm, ksonnet, etc.)

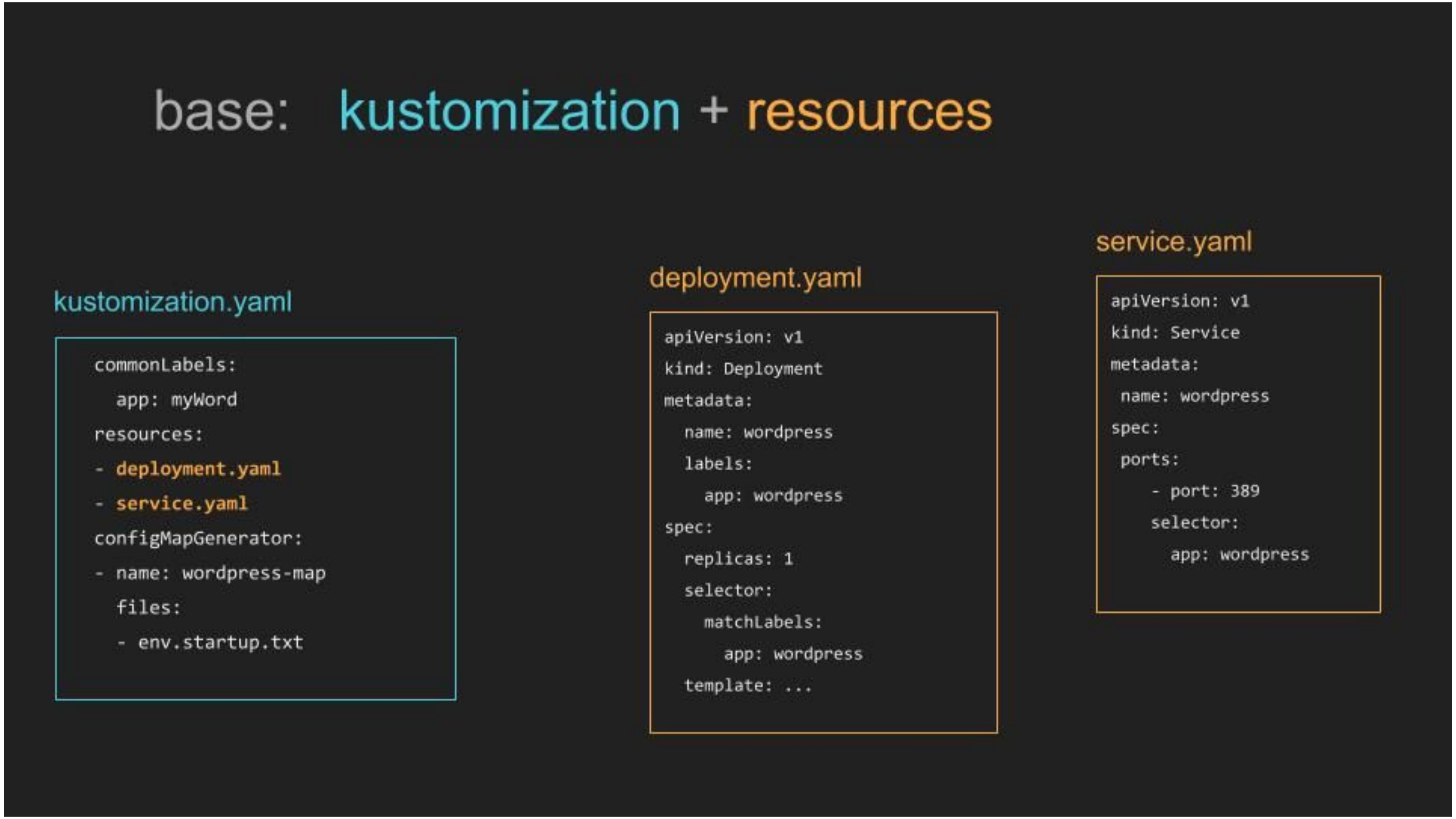
`kustomize` lets you customize raw, template-free YAML files for multiple purposes, leaving the original YAML untouched and usable as is.

`kustomize` targets kubernetes; it understands and can patch kubernetes style API objects. It's like make, in that what it does is declared in a file, and it's like sed, in that it emits edited text.

`kustomize` repository: https://github.com/kubernetes-sigs/kustomize

`kustomize` examples: https://github.com/kubernetes-sigs/kustomize/tree/master/examples

kustomize example: step 1



base: kustomization + resources

kustomization.yaml

```yaml
commonLabels:
  app: myWord
resources:
- deployment.yaml
- service.yaml
configMapGenerator:
- name: wordpress-map
  files:
  - env.startup.txt
```

deployment.yaml

```yaml
apiVersion: v1
kind: Deployment
metadata:
  name: wordpress
  labels:
    app: wordpress
spec:
  replicas: 1
  selector:
    matchLabels:
      app: wordpress
  template: ...
```

service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
 name: wordpress
spec:
 ports:
    - port: 389
    selector:
      app: wordpress
```

kustomize example: step 2

File structure:

```
~/someApp
├── deployment.yaml
├── kustomization.yaml
└── service.yaml
```

The resources in this directory could be a fork of someone else's configuration. If so, you can easily rebase from the source material to capture improvements, because you don't modify the resources directly.

Generate customized YAML with:

```
kustomize build ~/someApp
```

he YAML can be directly applied to a cluster:

```
kustomize build ~/someApp | kubectl apply -f -
```

Recent Initiatives

What have we been working on?

- `kubectl` plugins
  - Discussed in SIG CLI Deep Dive (tomorrow @10:50am)
- `kubectl` independence:
  - Moving the kubectl code base from the Kubernetes repository, into its own repository.
  - github umbrella issue
- `kubectl` server-side printing:
  - printing functionality happens on the server, returning a table of information to the client.
  - github umbrella issue
- `kubectl apply` moves to server side
- `kustomize` integration with `kubectl`
  - Currently we are in the process of integrating `kustomize` into `kubectl`.

kubectl plugins

Kubernetes Enhancement Proposal (KEP) for Plugins (jvallejo@redhat.com)

- Git-style plugins for `kubectl` (alpha in 1.12)
- PR to implement this plugin mechanism (jvallejo@redhat.com)

Generic CLI Options Library (CLI SDK)

- A library for plugin authors

KREW: A proposal for managing plugins

SIG CLI Deep Dive will discuss this in more depth (tomorrow @10:50am)

kubectl independence

We're moving the `kubectl` code base out of the Kubernetes core.

- New `kubectl` repository
  - https://github.com/kubernetes/kubectl
- `kubectl` codebase is currently in the core Kubernetes repository
  - https://github.com/kubernetes/kubernetes/tree/master/cmd/kubectl
  - https://github.com/kubernetes/kubernetes/tree/master/pkg/kubectl
- Challenges: removing core Kubernetes dependencies.
- Advantages
  - Increase development velocity
  - Simplify the `kubectl` code base
- Umbrella Issue: https://github.com/kubernetes/kubectl/issues/80

kubectl: Moving Logic into API Server

- **Move printing logic into API server** (Available since 1.11)
  - Moved knowledge of types and fields during printing to the API server. More information is here.
- **Move apply into API server** (in progress)
  - `kubectl apply` is a declarative create/update mechanism with patch/merging logic on the client. This logic is being moved to the API server.
  - This work is mostly directed by SIG API Machinery

kustomize integration into kubectl

- `kustomize` integration with `kubectl`
  - Currently we are in the process of integrating kustomize into kubectl.
  - Example: `kubectl apply -f <dir>`
    - If there is a `kustomization.yaml` in the specified directory, `kustomize` will generate the declarative resource YAML to pipe it to `kubectl apply`

How to get involved

- Regular SIG Meeting: Wednesdays at 09:00 PT (Pacific Time) (biweekly).
    - Meeting notes and Agenda (including zoom link).
    - Meeting recordings.
- Reach out on the SIG CLI Slack Channel
- Contact us through our email group
- Current Initiatives Release Tracking
- Prioritized bugs (look for label “help wanted”)
- Issues (look for "good first issue")

SIG CLI Leadership

Tech Leads:

Maciej Szulik (@soltysh): Red Hat

Phillip Wittrock (@pwittrock): Google

Chairs:

Maciej Szulik: (@soltysh): Red Hat

Sean Sullivan (@seans3): Google

Useful Links

- Release Tracking/Current Initiatives
- Prioritized Bugs
- Agenda Notes
- Slack Channel
- Email Group
- Testing Playbook
- Test Grid

Q&A