Helm and Kubernetes

How to use and write Helm Charts

By Aaron Johnson

What is Helm?

Helm claims to be a package manager for Kubernetes but what does that mean?

Helm's primary function is templatizing (uses go templates) and parameterizing yaml and then submitting the completed yaml to kubeapi.

Graduated CNCF project.

Give users the ability to "reuse" kubernetes yaml manifests for their deployments in a more efficient way than without Helm.

REQUIREMENTS

- a Kubernetes cluster
- ~~Tiller~~
- strike that, Tiller is no longer required with Helm v3+

WHAT WAS TILLER?

- Tiller was t ernetes API to man in-cluster service.
- The remova ace.
- Without Till rom the Kubernetes nd stores a record of the installation in Kubernetes.

SO ALL YOU NEED TO USE HELM IS KUBERNETES? GREAT HOW DO I USE IT?

- Download the binary from https://github.com/helm/helm/releases
- Run it

SET UP DEMO

- Install k3d https://github.com/rancher/k3d/releases (requires docker)

```
# k3d cluster create mycluster
```

- or export the kubeconfig for an existing cluster

```
# k3d kubeconfig get mycluster > ~/.kube/config
```

- Install ingress-nginx
- Also `--debug --dry-run` is your friend

NGINX-INGRESS (INGRESS-NGINX?)

Note: this is a non-production grade installation since it is not HA and listens on non-default ports

```
# helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx
# helm repo update
# helm upgrade ingress-nginx \
  --install \
  --set defaultBackend.replicaCount=1 \
  --set defaultBackend.minAvailable=1 \
  --set controller.replicaCount=1 \
  --set controller.metrics.enabled=true \
  --set controller.extraArgs.v=2 \
  --set-string controller.metrics.service.annotations."prometheus\.io/scrape"="true" \
  --set-string controller.metrics.service.annotations."prometheus\.io/port"="10254" \
  --set-string controller.metrics.service.annotations."prometheus\.io/scheme"="http" \
  --set controller.config.ssl-redirect=true \
  --set controller.config.ssl-protocols="TLSv1.2 TLSv1.3" \
  --set controller.config.ssl-ecdh-curve="X25519" \
  --set-string controller.config.ssl-
ciphers='TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:ECDHE-ECDSA-AES128-GCM-
SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-
CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:!aNULL:!
eNULL:!EXPORT:!DES:!RC4:!3DES:!MD5:!PSK' \
  --set controller.config.proxy-body-size="100m" \
  --set controller.config.use-gzip=true \
  --set controller.config.gzip-level=6 \
  --set controller.updateStrategy.type=Recreate \
  --set controller.service.ports.https=4433 \
  --set controller.service.ports.http=8081 \
  ingress-nginx/ingress-nginx
```

TEST YOUR INGRESS

- This is just an example, use any manifest or helm chart to test that your ingress is working

```
# kubectl create namespace grafana
# helm -n grafana upgrade grafana \
  --install \
  --set ingress.enabled=true \
  --set ingress.annotations."kubernetes\.io/ingress\.class"="nginx" \
  --set ingress.hosts[0]="grafana.example" \
  grafana/grafana
# kubectl -n grafana get services
# curl -H "Host: grafana.example" 172.18.0.2:8081/login
```

WRITING YOUR FIRST HELM CHART

- First how does it work?
  - If you've written any Ansible (Jinja2) or Puppet (ERB) templates you should feel right at home
  - Except that Helm uses Go templates which are slightly different
  - For example, you may find yourself using the quote function to ensure a value is returned as a string.
  - Since Go is a statically typed language it will generally be a bit more picky about typing even in the go templating language

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: {{ .Release.Name }}-configmap
data:
  myvalue: "Hello World"
  drink: {{ quote .Values.favorite.drink }}
  food: {{ quote .Values.favorite.food }}
```

CHART STRUCTURE

```
wordpress/
├── charts/                    # A directory containing charts upon which this chart depends
├── Chart.yaml                 # A YAML file containing information (metadata) about the chart
├── templates/                 # A directory of templates that, when combined with values,
                               # will generate valid Kubernetes manifest files.
│   ├── deployment.yaml
│   ├── _helpers.tpl           # The default location for template partials
│   ├── hpa.yaml
│   ├── ingress.yaml
│   ├── NOTES.txt              # A template of short usage notes displayed after chart installation
│   ├── serviceaccount.yaml
│   ├── service.yaml
│   └── tests/                 # A test is a pod manifest with a given success/fail command to run
│       └── test-connection.yaml
├── requirements.yaml          # A YAML file listing dependencies for the chart
└── values.yaml                # The default configuration values for this chart
```

HELM CREATE

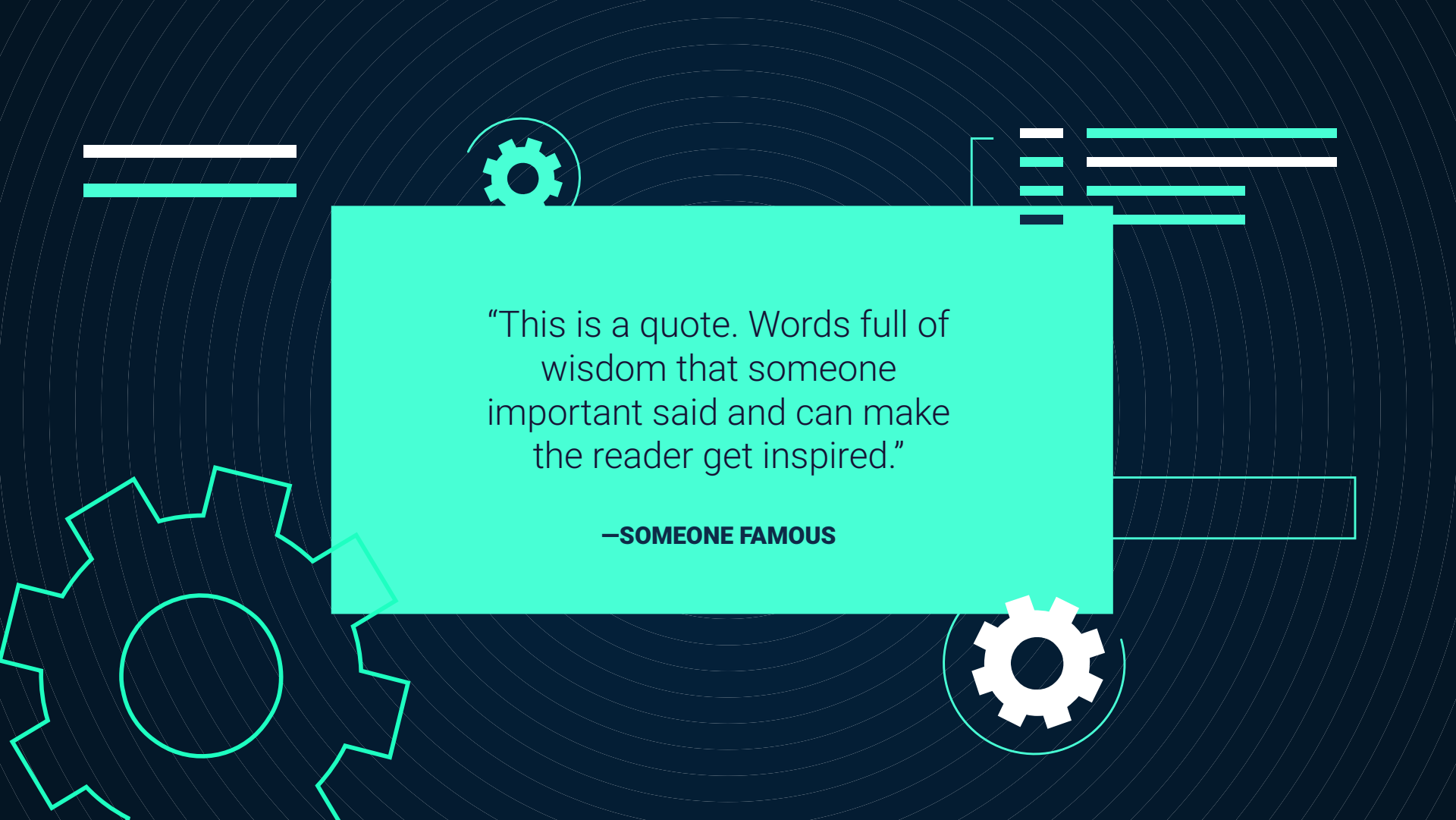
- The helm create command will make a boiler plate starting point for you
- However, starting with boiler plate code BEFORE having a working kubernetes yaml manifest for your deployment is an exercise in futility in my experience…
- If this is your first time writing a Helm chart I highly suggest you write the manifests first, test them, and then templatize them otherwise troubleshooting Helm errors is exhausting
- That said, YMMV

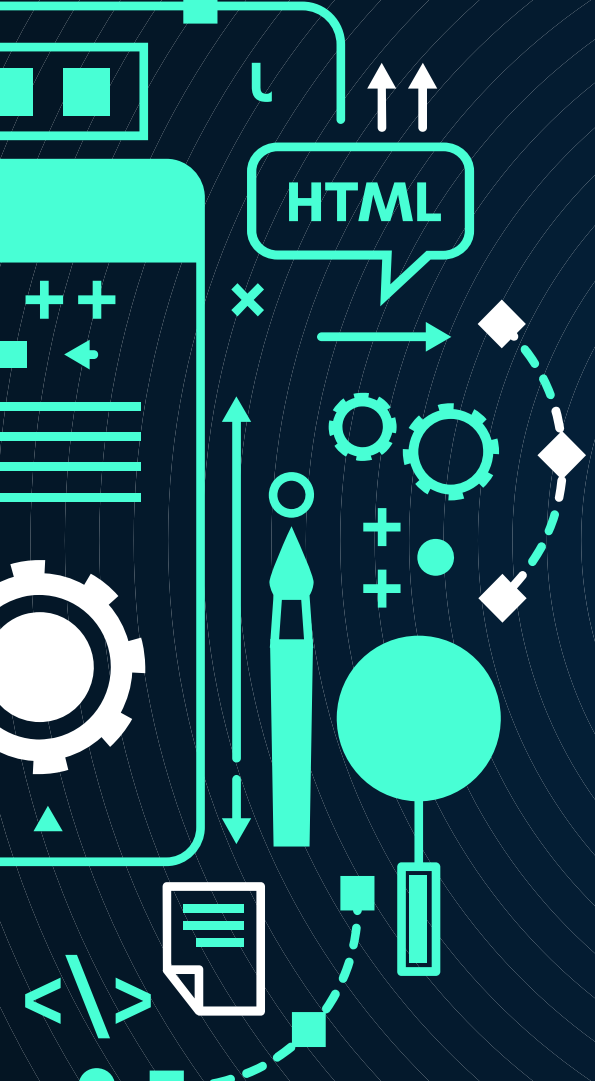
CHARTS REPOSITORY

```
# mkdir my-helm-charts
# cd my-helm-charts/
# git init .
# mkdir charts
# cd charts/
# helm create myapp
# git add myapp
# git commit -am "Initial commit"
```

LEARN GO TEMPLATES

- Finally once you have your chart structure started you can begin writing your Helm charts which is basically just writing Go template files wrapped in YAML
- If you are looking for inspiration I suggest these resources:
- Official Helm Documentation
- Read and learn from official helm charts on artifacthub.io
- Learn to use best practices patterns such as Library Charts


“This is a quote. Words full of wisdom that someone important said and can make the reader get inspired.”
—SOMEONE FAMOUS



THANKS!

Does anyone have any question?

+91 620 421 838
yourcompany.com