Themen-Tage
Deine Online-IT-Konferenz

Fabrizio Lazzaretti

# Einführung in Helm: Die bessere Art Kubernetes-Anwendungen zu managen

## Fabrizio Lazzaretti

- 6+ years of experience with Kubernetes
- 5+ years of experience with Helm
- 4+ years of experience with helmfile
- Cloud-, Integration- & Enterprise Architect

# Helm is part of the Cloud Native Computing Foundation (CNCF)

- CNCF is a foundation is a Linux Foundation project
- Enable open-source projects in the cloud & adds governance
- Founded in 2015 with the anouncment of Kubernetes 1.0 (donation from Google, before called Borg)
- The big projects: Kubernetes, Prometheus, Envoy, ArgoCD

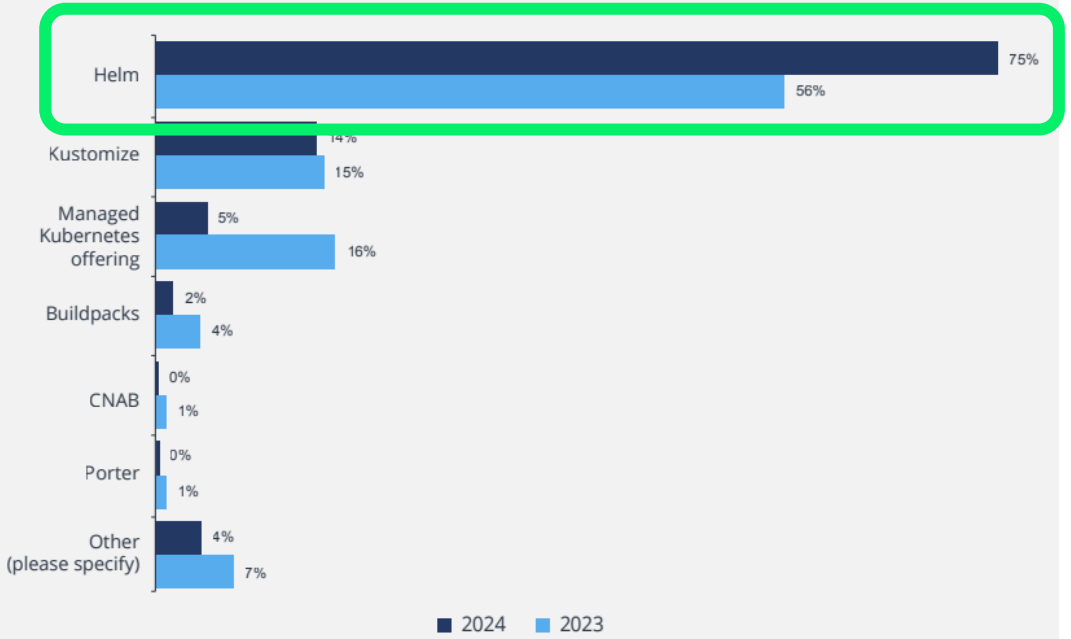
Sponsors:

# Helm: The most used method for packaging Kubernetes apps

FIGURE 12

## THE PREFERRED METHOD FOR PACKAGING KUBERNETES APPLICATIONS

**What is your preferred method for packaging Kubernetes applications? (select one)**




2024 CNCF Annual Survey, Q27, Sample Size = 320, shown to end-users, SIs or consultants based on Q9 and Q10, AND those who use or test containers in Q20, AND use or test Kubernetes in Q23, DKNS responses excluded from the analysis
2023 CNCF Annual Survey, Q26, Sample Size = 397, organizations whose primary revenue is not from offering cloud native technologies or services, DKNS responses excluded from the analysis
2022 CNCF Annual Survey, Q20, Sample Size = 1,476 (not shown but in the underlying data)

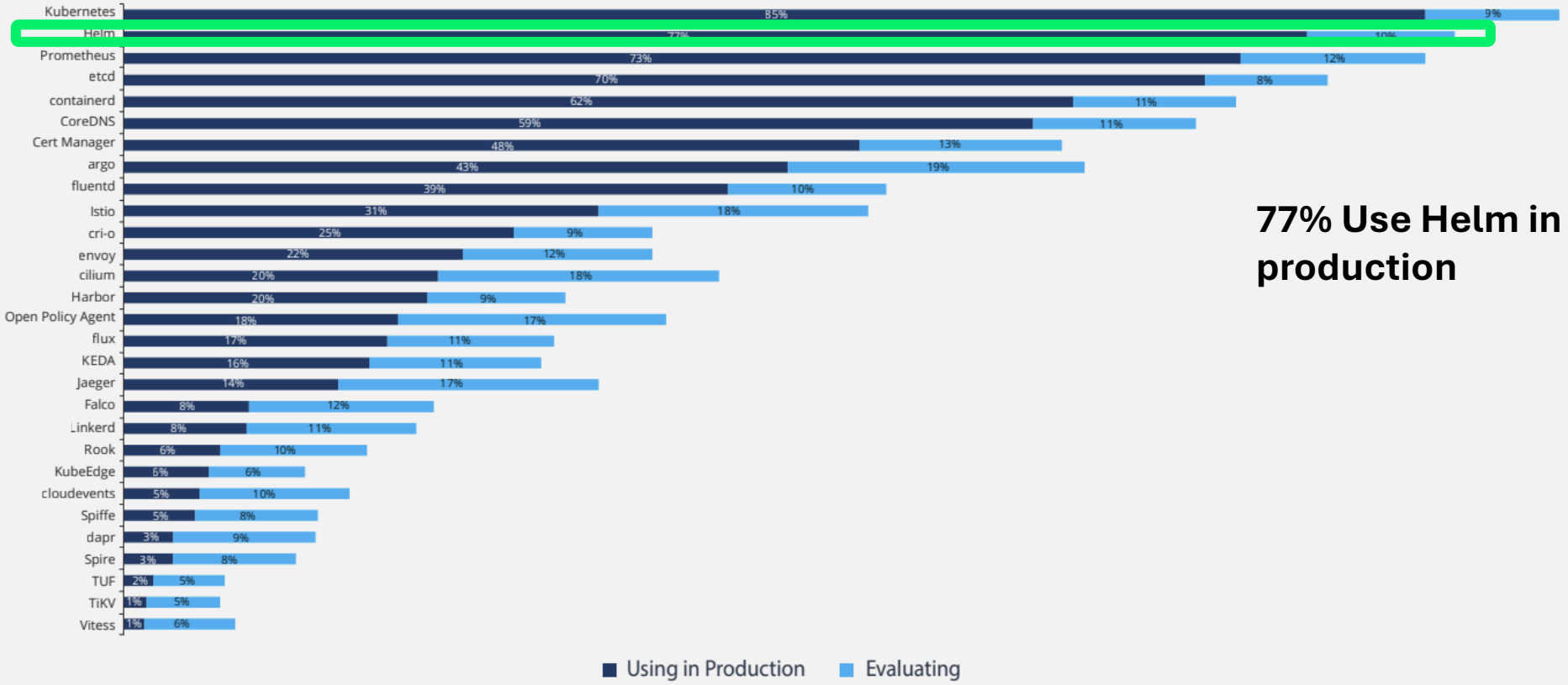
https://www.cncf.io/wp-content/uploads/2025/04/cncf_annual_survey24_031225a.pdf

# Helm: second most used CNCF graduated project in production

FIGURE 13

## CNCF GRADUATED PROJECTS IN USE OR IN EVALUATION

**Which of these graduated CNCF projects is your organization using in production or evaluating? (select one response per row)**




**77% Use Helm in production**

2024 CNCF Annual Survey, Q32, sample size = 689

https://www.cncf.io/wp-content/uploads/2025/04/cncf_annual_survey24_031225a.pdf

# Kubernetes

- Open-source container orchestration system for automating software deployment
- Project is maintained by the Cloud Native Computing Foundation (CNCF)
- Uses the concept of “desired state”

- You normally use YAML files to deploy the resources
  - Problem
    - Hard to keep track on what is important in huge flies and what is different per use case
    - Low-level API
    - Copy-Paste Deployments (no parameters)
    - No simple rollback (delete is manual)

HELM solves these problems

# What is HELM?

“A *helmsman* or “helm” is a person who steers a ship, sailboat, submarine, other types of maritime vessel, or spacecraft.” — Wikipedia

# What is Helm? – apt get for Kubernetes

- Package manager for Kubernetes
- **Graduated CNCF project**
- A package is called “Chart”

**Helm does** `kubectl apply & kubectl delete` **for you**

- **Create**
- **Update**
- **Delete**

# The Chart Structure

```
mychart/
  Chart.yaml          # A YAML file containing information about the chart
  LICENSE             # OPTIONAL: A plain text file containing the license
                      # for the chart
  README.md           # OPTIONAL: A human-readable README file
  requirements.yaml   # OPTIONAL: A YAML file listing dependencies for the chart
  values.yaml         # The default configuration values for this chart
  charts/             # A directory containing any charts upon which this chart
                      # depends.
  templates/          # A directory of templates that, when combined with values,
                      # will generate valid Kubernetes manifest files.
  templates/NOTES.txt # OPTIONAL: A plain text file containing short usage notes
```

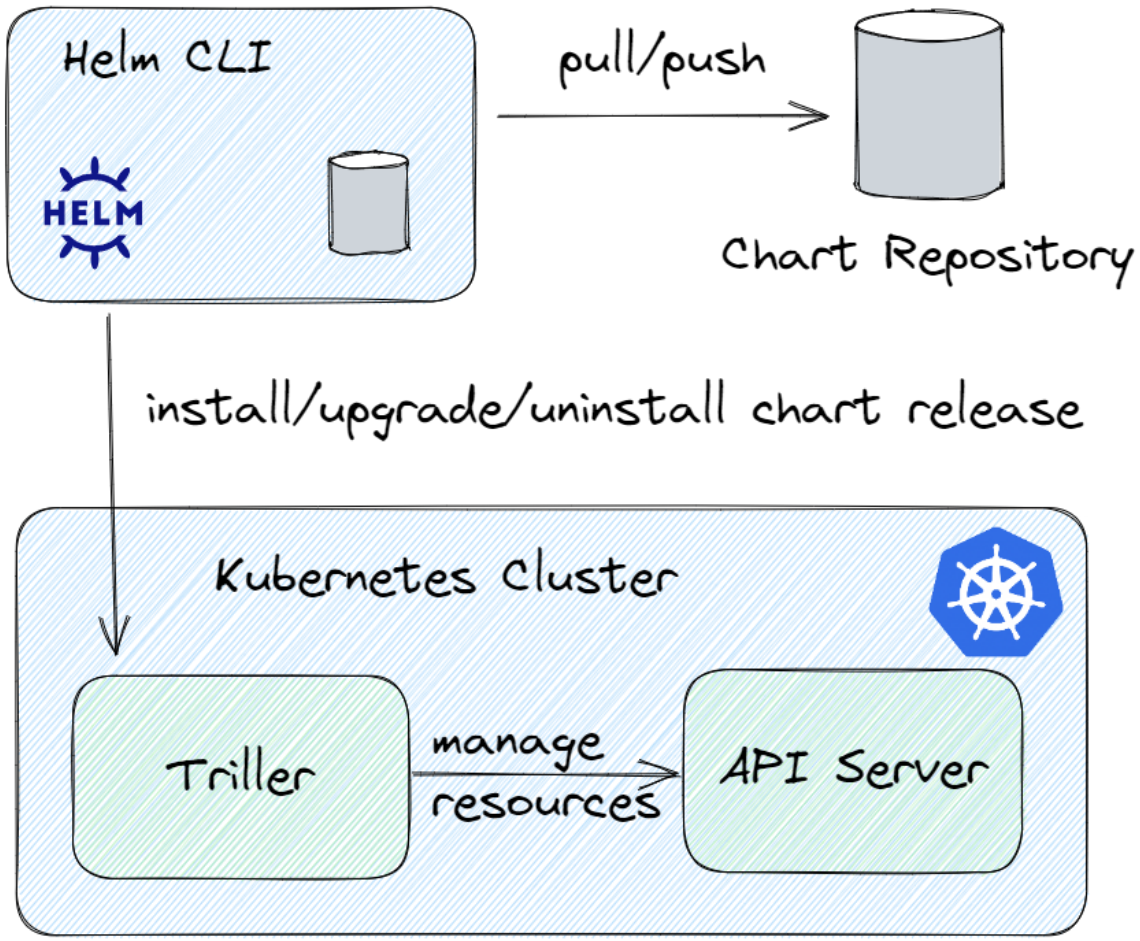
# Templates

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: {{ include "mychart.fullname" . }}-config-map
  labels:
    {{- include "mychart.labels" . | nindent 4 }}
data:
  config: {{ .Values.config }}
```

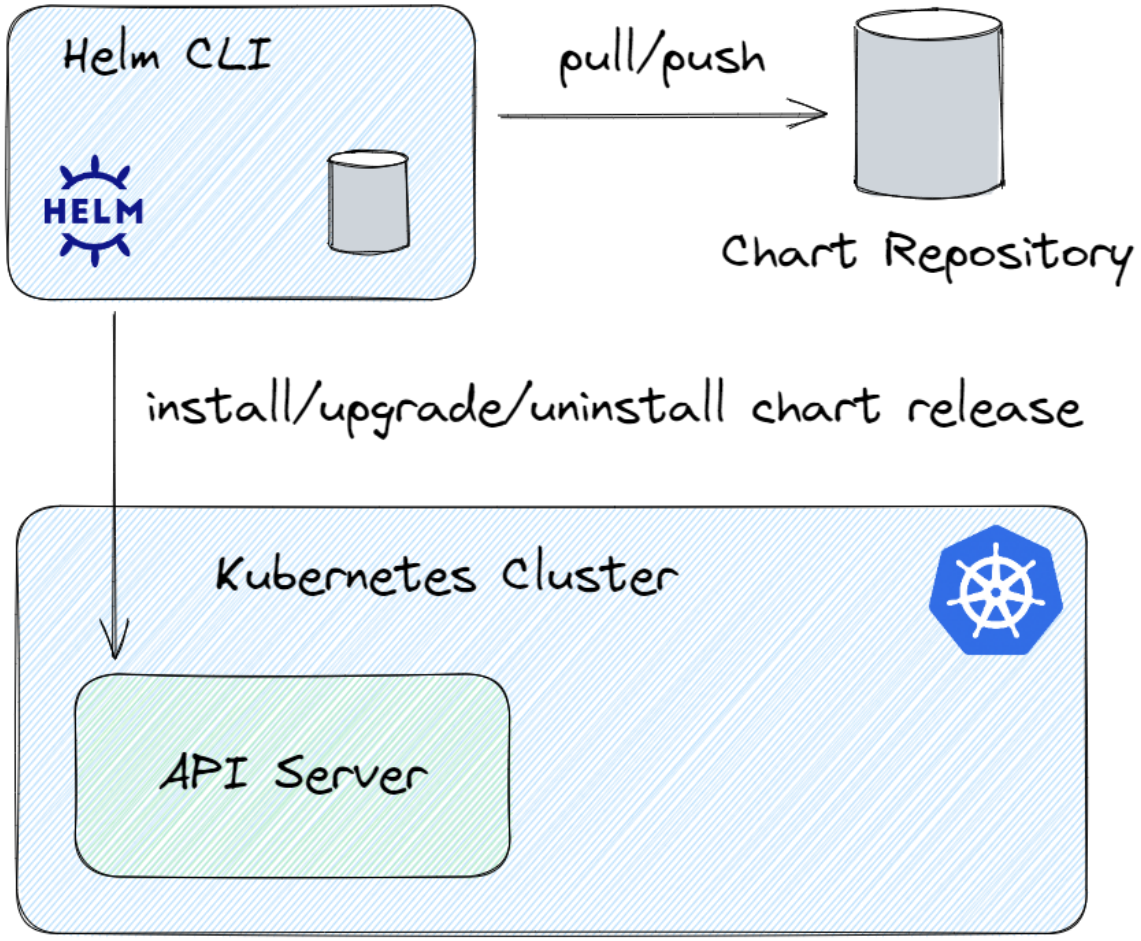
```yaml
some:
  - configA
  - configB
```

# Helm Moved with v3 to a Client-Only Implementation

## Helm 2




## Helm 3



# Helm in Action: Deploy Nginx

1. Find the Helm chart you want:

   nginx: https://artifacthub.io/packages/helm/bitnami/nginx

2. Install the chart:

   ```
   helm install hello-world
   oci://registry-1.docker.io
   /bitnamicharts/nginx
   ```

ArtifactHUB | Search packages | DOCS | STATS | SIGN UP | SIGN IN

Back to "**nginx**" results

# nginx

Helm chart | Networking

Bitnami | Bitnami

Star 79

NGINX Open Source is a web server that can be also used as a reverse proxy, load balancer, and HTTP cache. Recommended for high-demanding sites due to its ability to provide faster content.

SUBSCRIPTIONS: 23 | WEBHOOKS: 3 | PRODUCTION USERS: 2

## nginx

## Bitnami package for NGINX Open Source

NGINX Open Source is a web server that can be also used as a reverse proxy, load balancer, and HTTP cache. Recommended for high-demanding sites due to its ability to provide faster content.

Overview of NGINX Open Source

Trademarks: This software listing is packaged by Bitnami. The respective trademarks mentioned in the offering are owned by the respective companies, and use of them does not imply any affiliation or endorsement.

## TL;DR

```
helm install my-release oci://registry-1.docker.io/bitnamicharts/nginx
```

Looking to use NGINX Open Source in production? Try VMware Tanzu Application Catalog, the commercial edition of the Bitnami catalog.

## Introduction

INSTALL

TEMPLATES

DEFAULT VALUES

VALUES SCHEMA

CHANGELOG


APPLICATION VERSION

1.27.2

CHART VERSIONS RSS

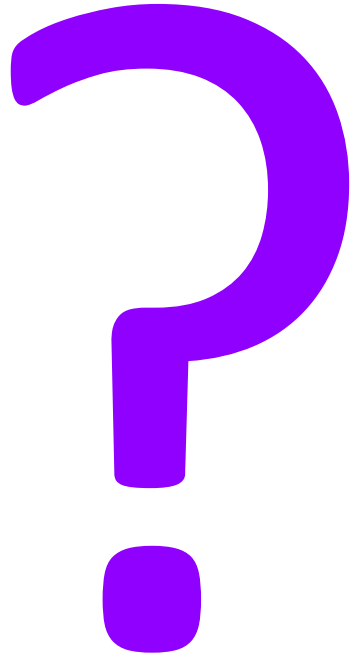
18.2.3 (14 Oct, 2024)

https://artifacthub.io/

# helm install hello-world oci://registry-1.docker.io/bitnamicharts/nginx

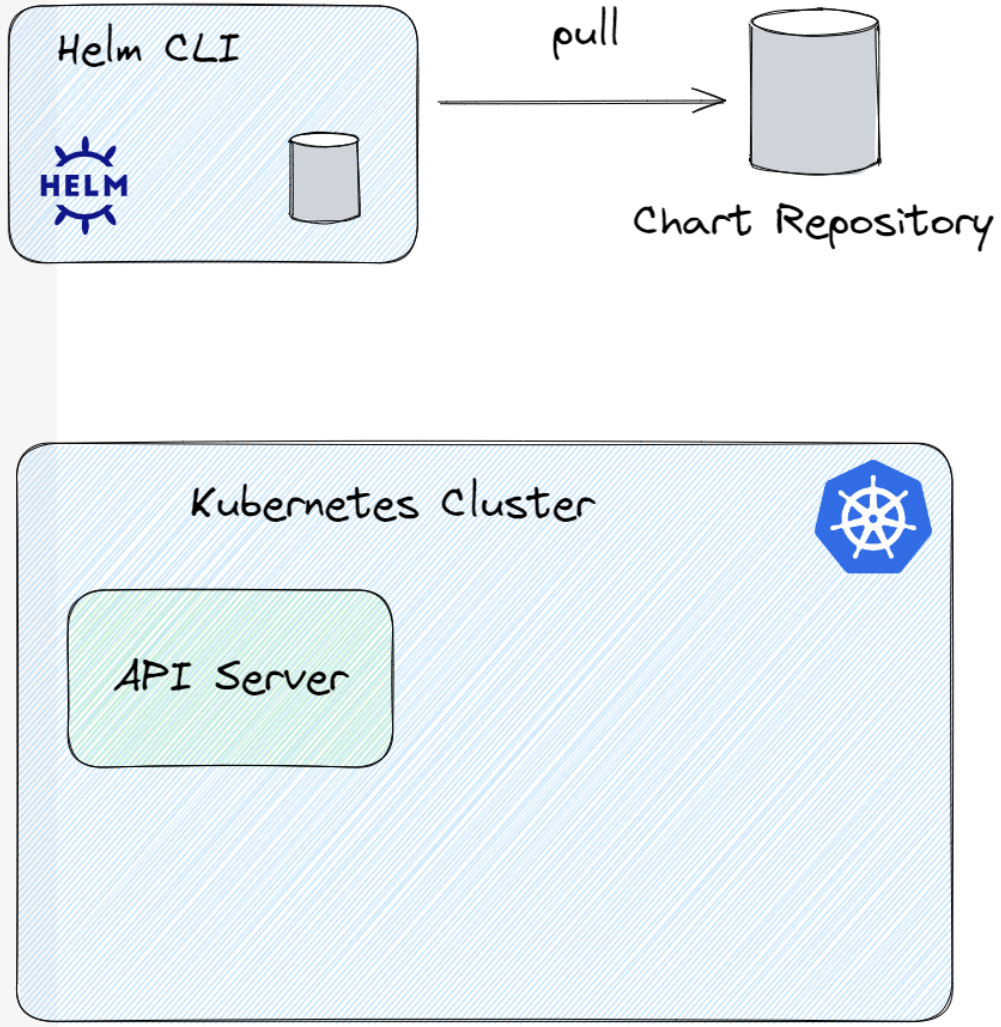
```
$
```

# What was that?



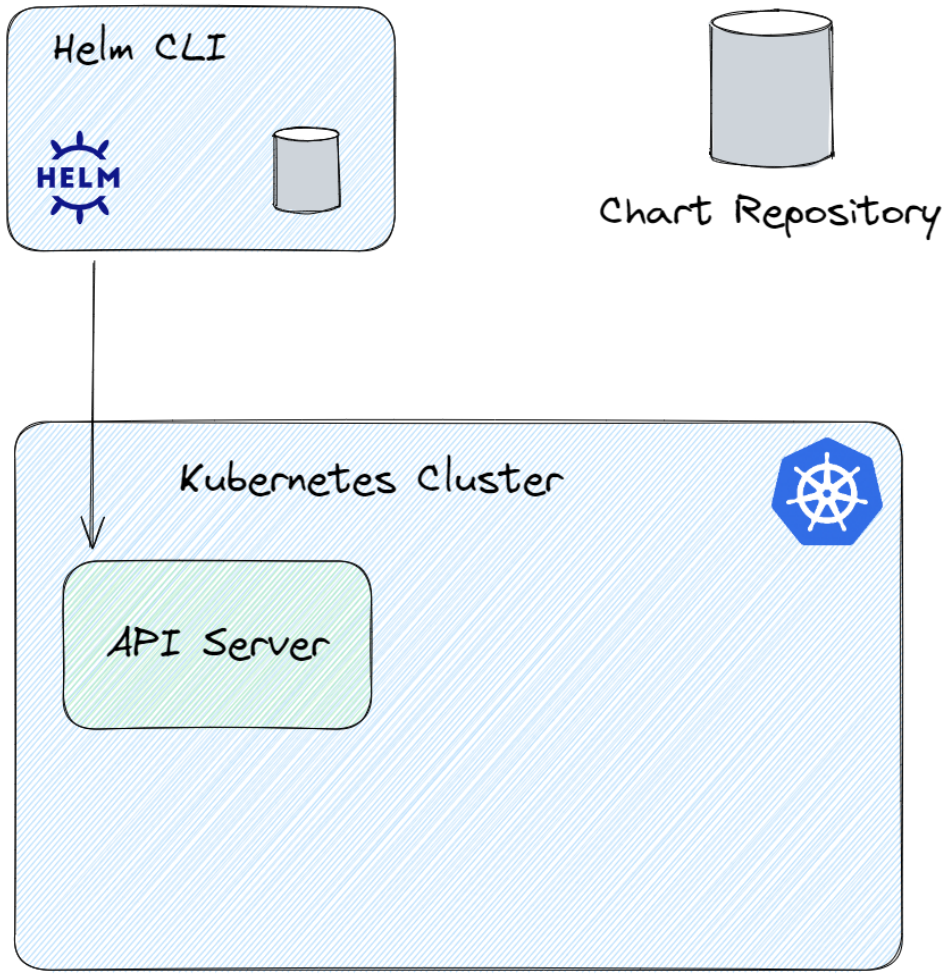
WAVESTONE

# Deploying a Chart: Pull



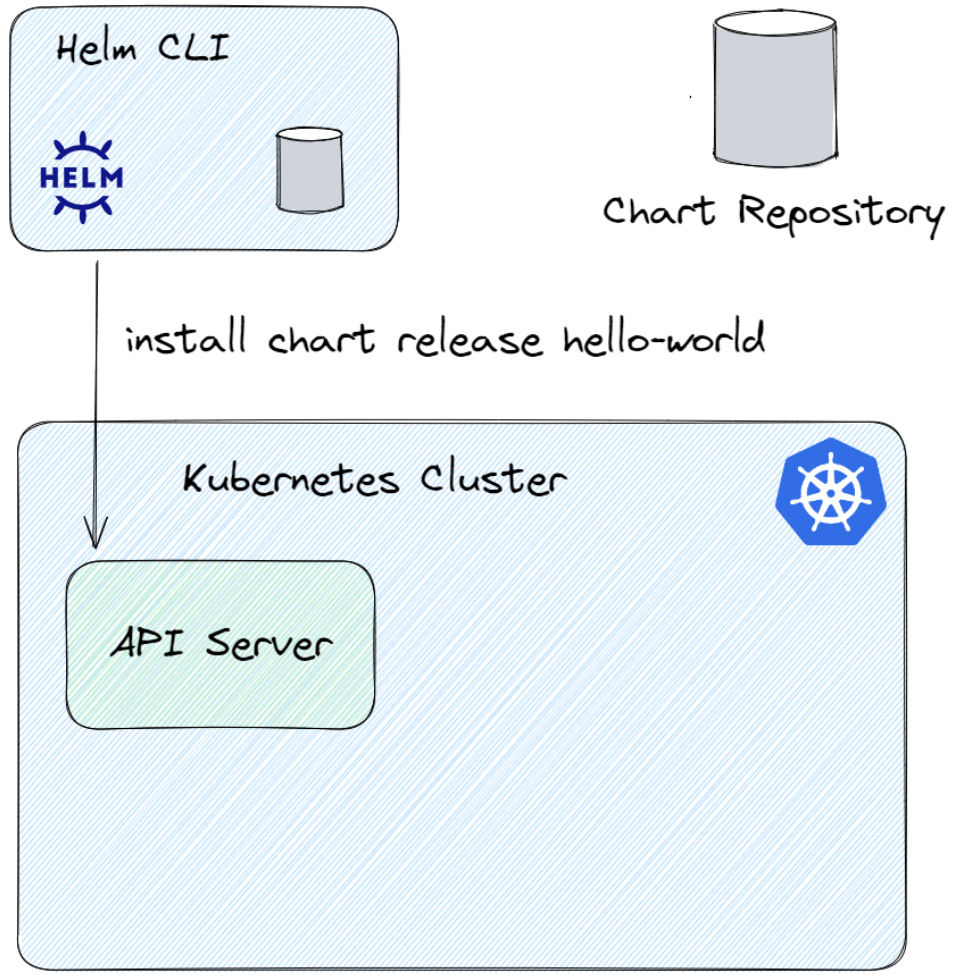
# Deploying a Chart: Template

```
$ h
```



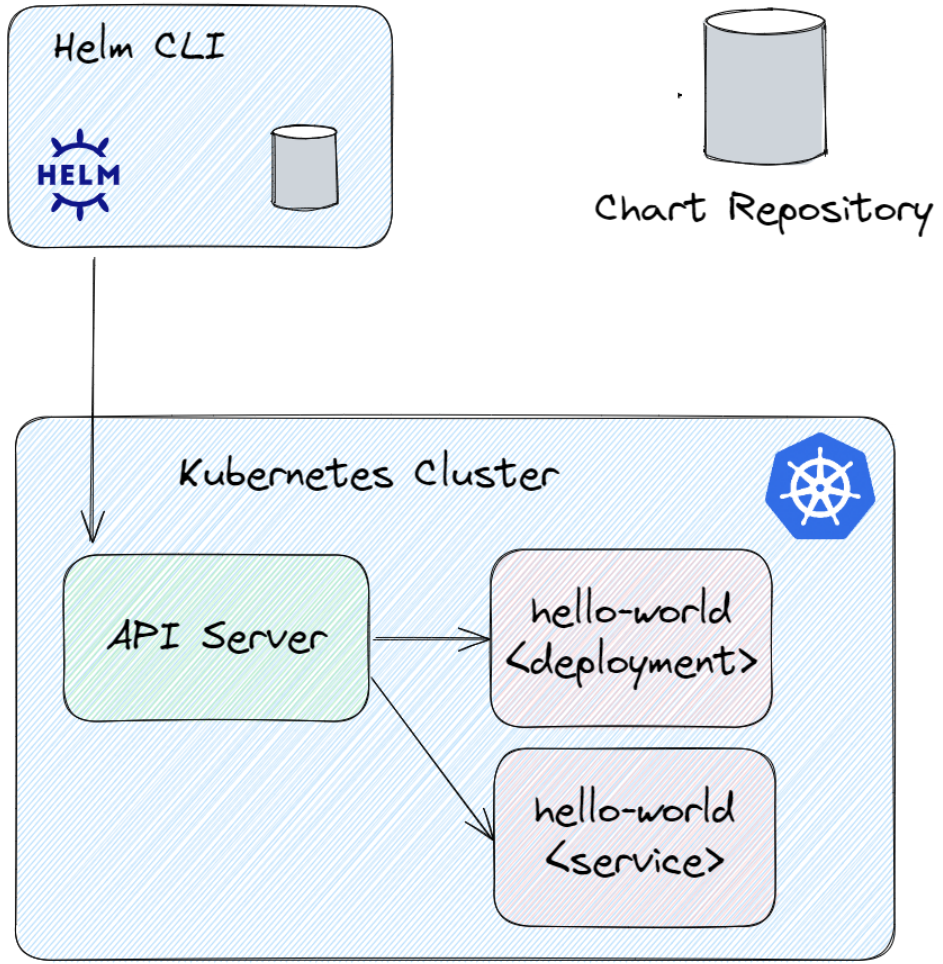
# Deploying a Chart: Install

```
$ h
```



# Deploying a Chart: The Deployment

```
$ kubectl get service
NAME                TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)
hello-world-nginx   LoadBalancer   10.102.60.28    <pending>     80:32320/
kubernetes          ClusterIP      10.96.0.1       <none>        443/TCP
$ kubectl get deployments.app
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
hello-world-nginx   1/1     1            1           13m
$ kubectl get pod
NAME                                 READY   STATUS    RESTARTS   AGE
hello-world-nginx-8579d6c9c8-7574h   1/1     Running   0          13m
$ helm ls
NAME            NAMESPACE       REVISION        UPDATED
hello-world     default         1               2023-11-20 10:54:45.37137
```



## Summary of the Steps

1. Download and unpack the Helm chart

```
helm pull oci://registry-1.docker.io/bitnamicharts/nginx --untar --untardir out/
```

2. See how it looks after it is rendered

```
helm template --output-dir './out' hello-world oci://registry-1.docker.io/bitnamicharts/nginx
```

3. Release

```
helm install hello-world oci://registry-1.docker.io/bitnamicharts/nginx
```

4. View the release

```
helm ls
```

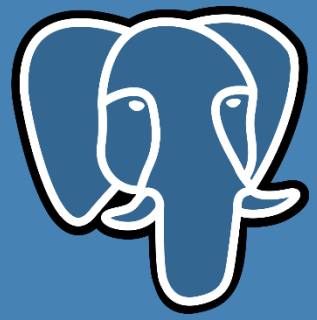
# Some of the Most Popular Charts on the Artifact Hub

kube-prometheus-stack

redis


cert-manager

harbor



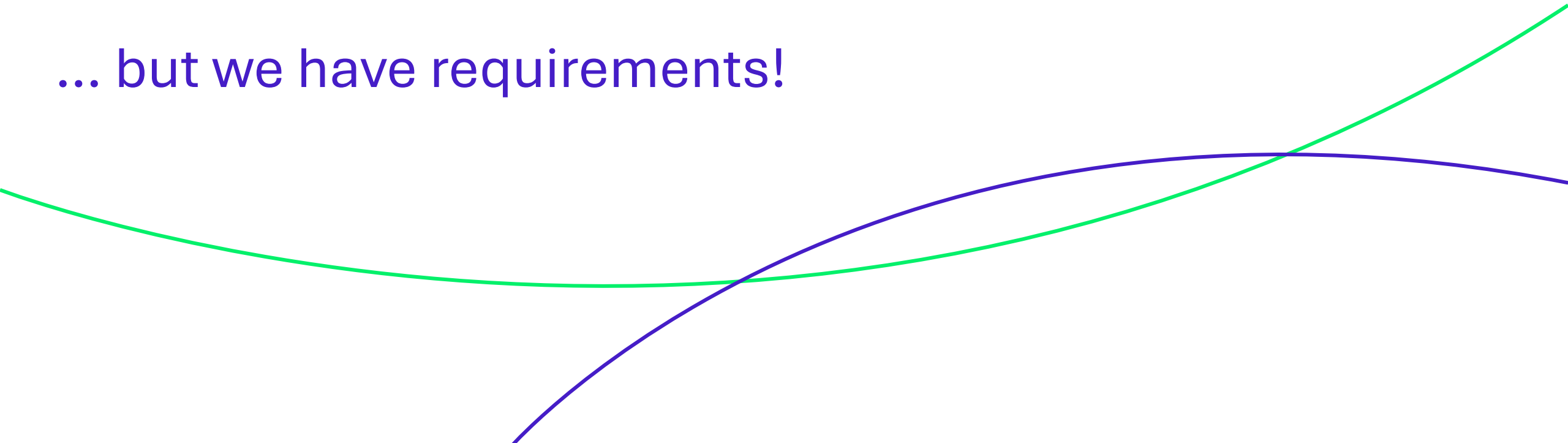
postgresql


nginx

argo-cd

Great!

... but we have requirements!



# Customize the Chart with Parameters

- Add cli parameter

```
helm upgrade hello-world oci://registry-1.docker.io/bitnamicharts/nginx --set replicaCount=2
```

- Add a YAML file

```
helm upgrade hello-world oci://registry-1.docker.io/bitnamicharts/nginx -f hello-world-values.yaml
```

# Helm Keeps the Previews Released Versions

1. Diff

   helm diff upgrade hello-world oci://registry-1.docker.io/bitnamicharts/nginx -f hello-world-values.yaml

2. Upgrade

   helm upgrade

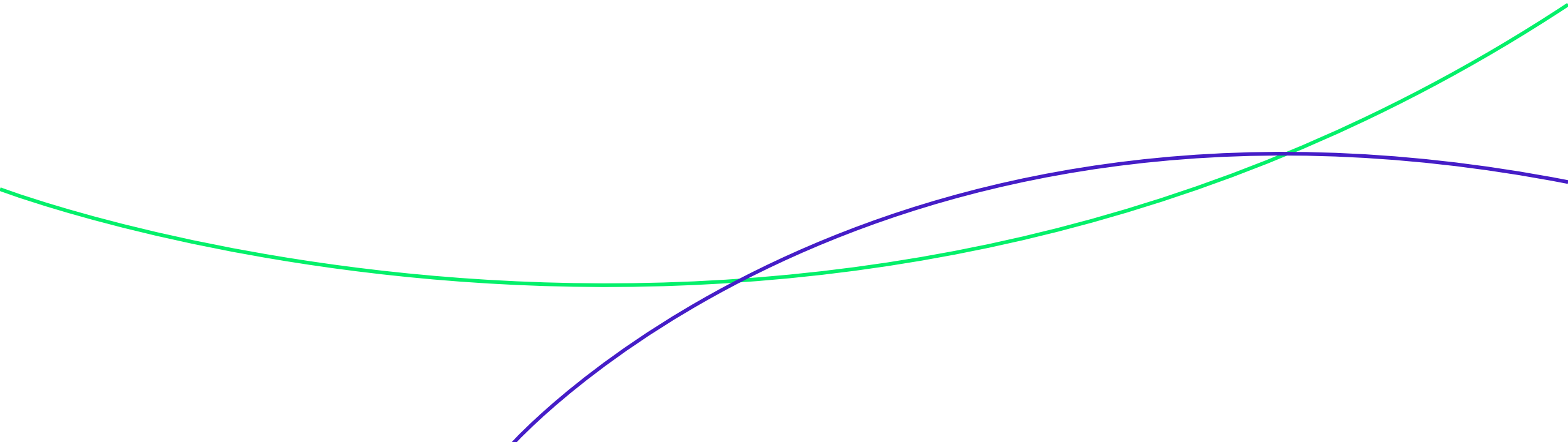
3. Show versions & status

```
$ helm ls
NAME            NAMESPACE       REVISION        UPDATED                                 STATUS
hello-world     default         2               2023-11-20 12:18:44.022768466 +0000 UTC deployed
```

4. Rollback

   helm rollback hello-world

# We need an own app!



## Building an Own Chart

1. Create a chart

```
helm create mychart
```

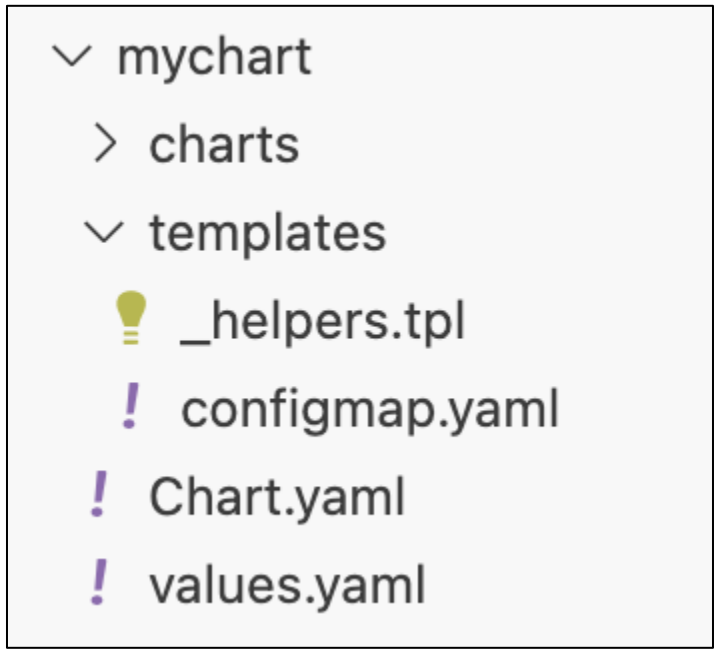
2. Configure as needed

3. Deploy it

```
helm install myapp ./mychart
```

4. Publish (with OCI repository)

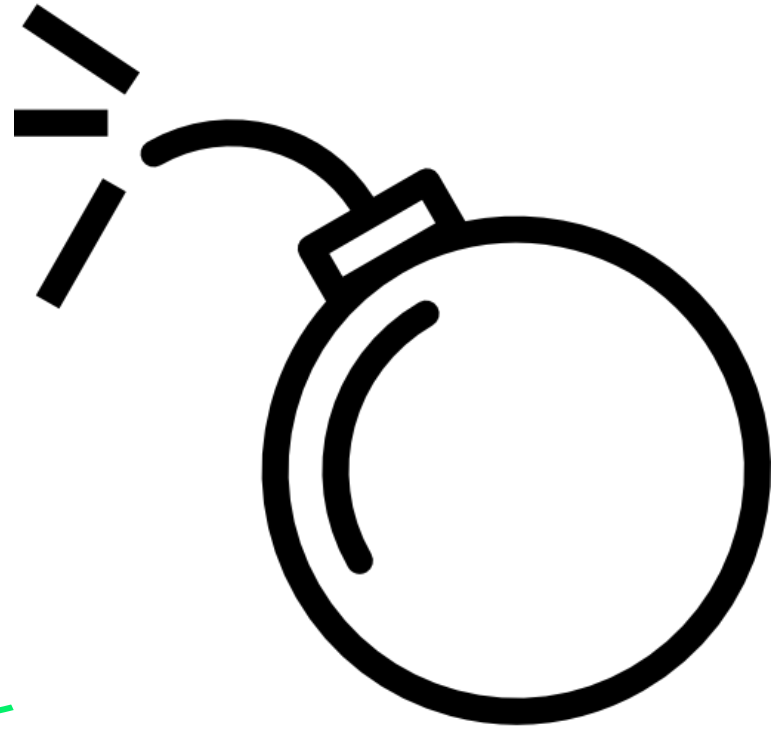
```
helm package ./mychart
helm push mychart-0.1.0.tgz oci://my.registry/project/mychart
```



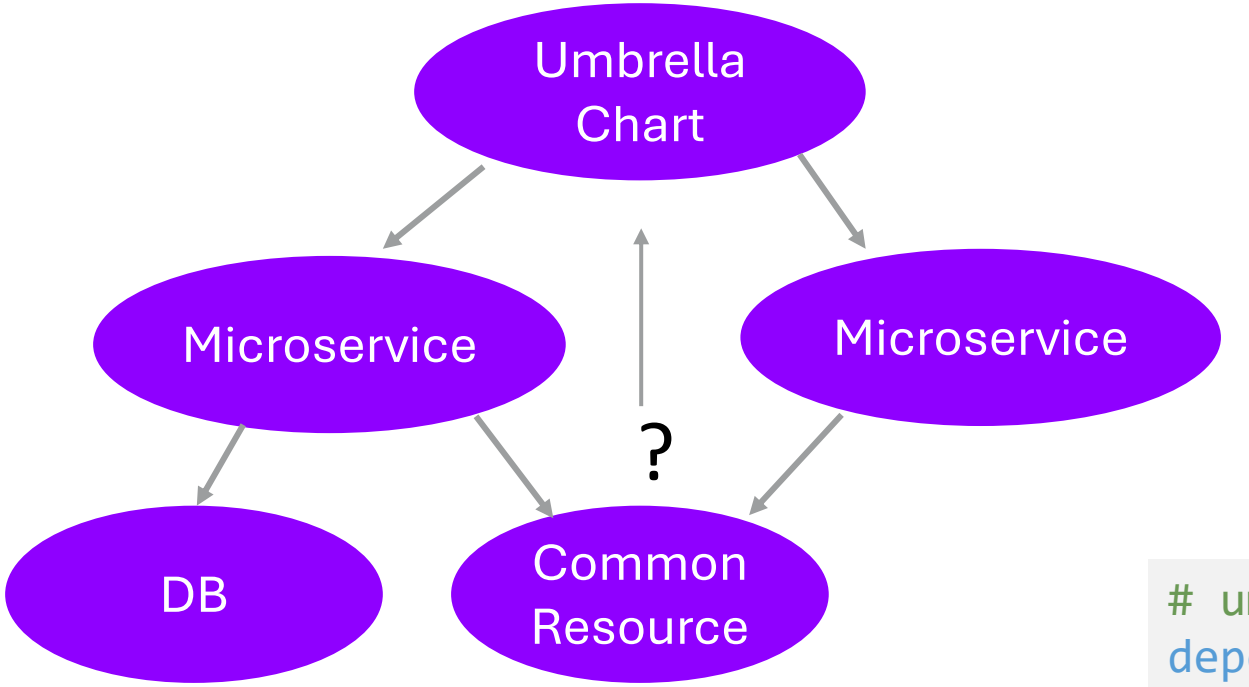
Many apps

many charts

Do we have a mess now?



# Umbrella Charts




- 1 chart for everything
  - => 1 version, atomic chart

Problems:

- No isolation for names
- Minor inconvenience: install subcharts, …

```yaml
# umbrella-chart/Chart.yaml
dependencies:
  - name: svc1
    repository: "https://charts.ex.com/repo"
    version: 1.2.3
  - name: svc2
    repository: "https://charts.x.com/repo"
    version: 2.3.4
```

## Multiple Environments

- `helm install mychart . -f values.yaml -f values-env.yaml`
- Helm will merge them right-to-left (right taking precedence).

```
mychart/
values-dev.yaml
values-prod.yaml
values.yaml
```

# Simplify with helmfile

# Manage Helm with a Single helmfile

```yaml
# helmfile.yaml
releases:
- name: hello-world
  namespace: default
  chart: oci://registry-1.docker...
  version: 15.4.2
  set:
  - name: replicaCount
    value: 2
```

```
$ helmfile apply
```

## Manage Helm with a Single helmfile

- Keep separate Helm releases
  - => separate upgrade & rollback
  - With dependency awareness

```
# helmfile.yaml
releases:
- name: hello-world
  namespace: default
  chart: oci://registry-1.docker...
  version: 15.4.2
  set:
  - name: replicaCount
    value: 2
```

```
$ helmfile apply
```

## Manage Helm with a Single helmfile

- Keep separate Helm releases
- Install dependencies into multiple namespaces

```
# helmfile.yaml
releases:
- name: hello-world
  namespace: default
  chart: oci://registry-1.docker...
  version: 15.4.2
  set:
  - name: replicaCount
    value: 2
```

```
$ helmfile apply
```

## Manage Helm with a Single helmfile

- Keep separate Helm releases
- Install dependencies into multiple namespaces
- Environment support out of the box

```
environments:
  production:
    values:
    - production.yaml
  development:
---
releases:
- name: hello-world
  namespace: default
  chart: oci://registry-
1.docker.io/bitnamicharts/nginx
  version: 15.4.2
  set:
  - name: replicaCount
    value: 2
```

```
$ helmfile --environment production apply
```

## Manage Helm with a Single helmfile

- Keep separate Helm releases
- Install dependencies into multiple namespaces
- Environment support out of the box
- Go templating for values and more
  - Make deployment simpler

```
releases:
- name: chart-{{ .Values.myrelease.name }}
  chart: mychart
  version: {{ .Values.myrelease.version }}
  values:
  - values.yaml.gotmpl
```

```
$ helmfile apply
```

## Manage Helm with a Single helmfile

- Keep separate Helm releases
- Install dependencies into multiple namespaces
- Environment support out of the box
- Go templating for values and more
- No difference between install/update
  - No `helm repo add`
  - No `helm install & helm upgrade`
  - => just `helmfile apply`

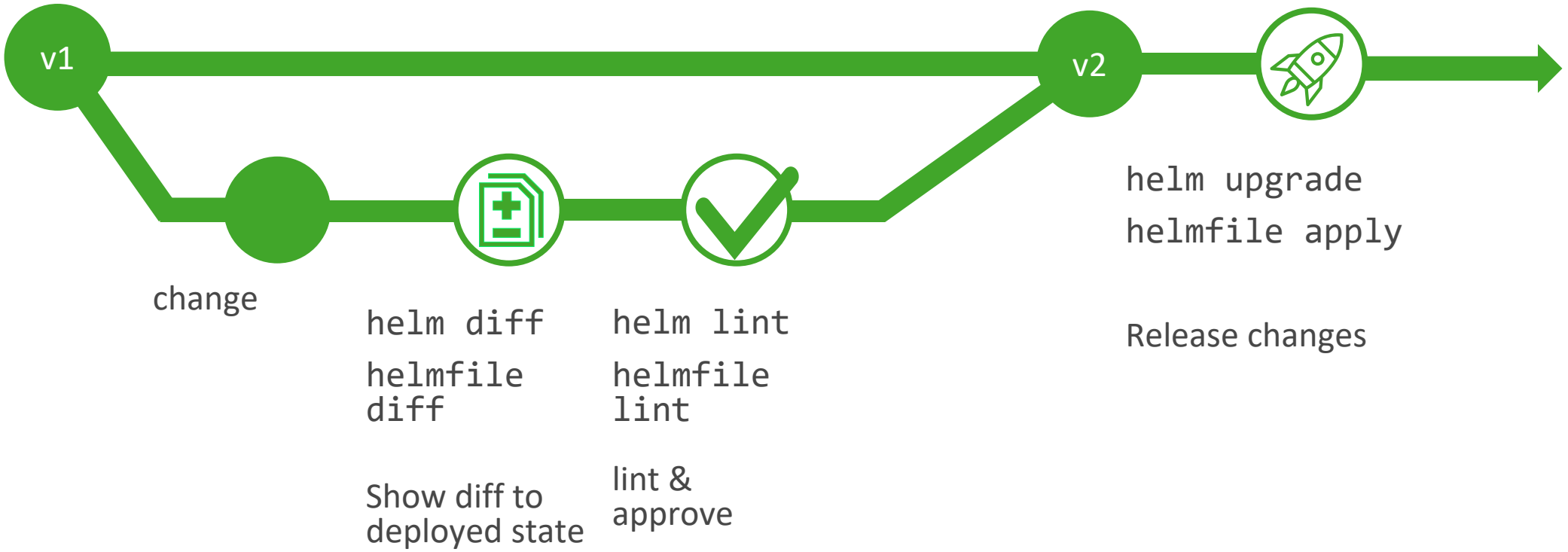
```yaml
# helmfile.yaml
repositories:
  - name: prometheus-community
    url: https://prometheus-
community.gi...

releases:
- name: prom-norbac-ubuntu
  namespace: prometheus
  chart: prometheus-community/prometheus
```

```
$ helmfile apply
```

# CI/CD

# GitOps Works Perfectly with Helm (& Helmfile)

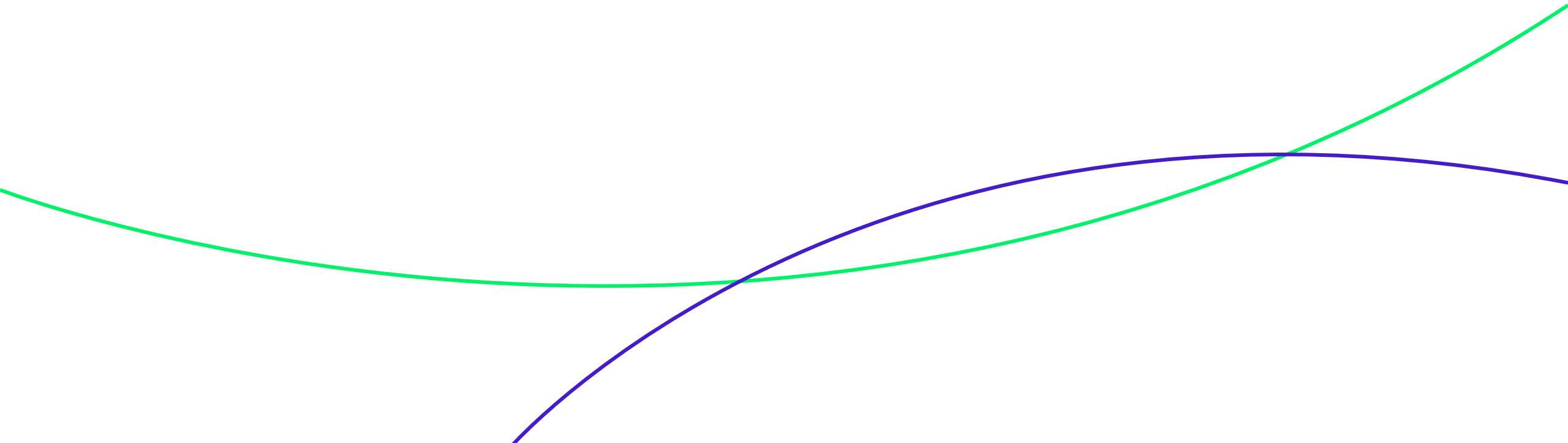
- Everything in code/config



# Automate the Deployment with Terraform / OpenTofu

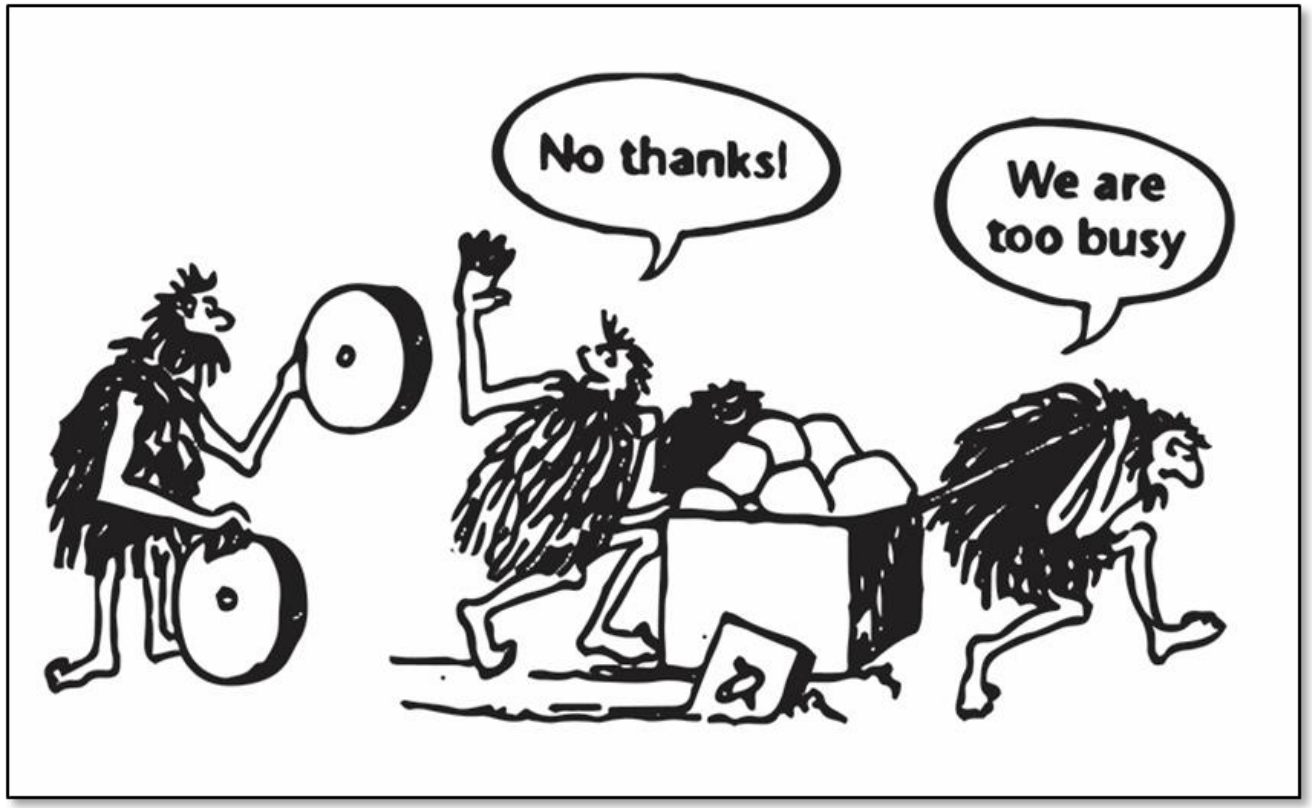
```
provider "helm" {
  kubernetes {
    config_path = "~/.kube/config"
  }
}

resource "helm_release" "hello-world" {
  name  = "hello-world"
  chart = "oci://registry-1.docker.io/bitnamicharts/nginx"

  set {
    name  = "replicaCount"
    value = "2"
  }
}
```

# Conclusion



## Helm & Helmfile simplify your life

1. Helm simplifies your life
2. Helmfile simplifies your live even more ;)




https://github.com/Lazzaretti/helm-demo

Themen-Tage

Deine Online-IT-Konferenz

14:00 - 14:45

Dr. Annegret Junker & Fabrizio Lazzaretti

# Domain-driven API Design

# #WeAreWavestone

Your most trusted partner for strategic transformations.