Kubernetes CKAD - Helm

- starts at 9:05pm

Agenda

1. Helm Basics
    1. Understanding Helm
    2. Installing Helm
    3. Key Concepts of Helm (CLI, Charts, Repositories, Release)
    4. Benefits of Using Helm
2. Helm Charts
    1. Structure of a Chart
    2. Wrapper chart
3. Helm Templating Language
4. Demo
5. Some tips and personal experience with helm

→ Helm Basics

1. **Download the game files, there are so many files in a game**
2. **Place them in the correct directories**
3. **Install required dependencies**
4. **Adjust configuration settings**

Install

- The installer **places files** in the correct locations.

- It **installs dependencies** (like DirectX or .NET Framework).

- It **configures settings** based on your system.

- It provides an **easy way to update** the game when a new patch is released.

Deployments
Svc
CM
Secrets
PV
PVC
Storage Classes
Cronjob.

helm install

helm upgrade. → patching game.

helm uninstall.

Helm

✅ Automates installation

✅ Ensure all dependencies are met

✅ Simplify configuration

✅ Provide easy upgrades

✅ Allow clean uninstallation

Helm is the package manager for Kubernetes.

→ Install Helm

curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash

→ Key Concepts in Helm

① Helm CLI.

kubectl → kubeconfig.

helm → kubeconfig.

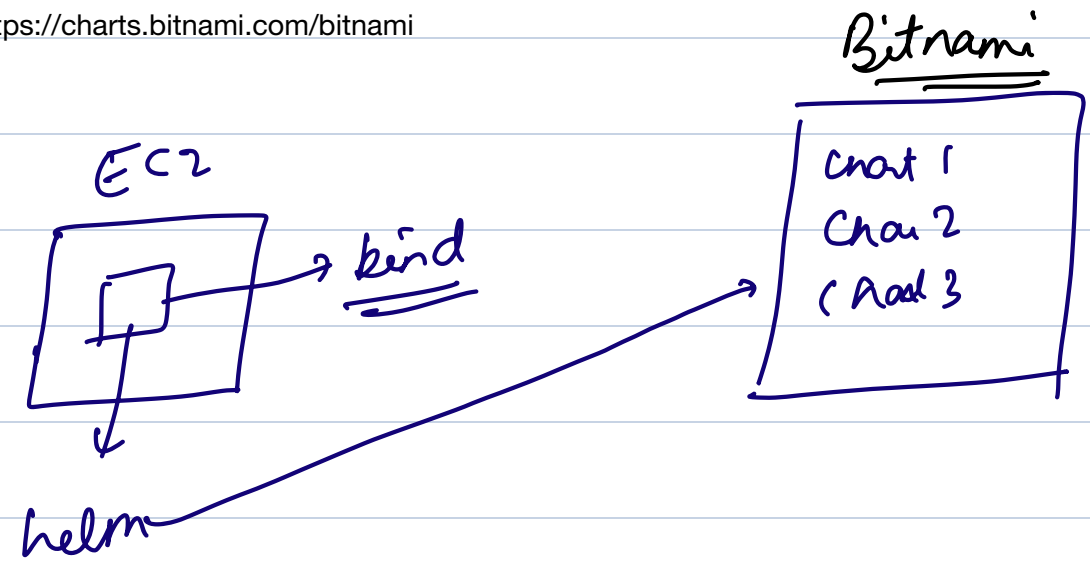
② Chart

A **Helm chart** is a **packaged Kubernetes application** that contains everything needed to deploy an app.

A Helm chart is a collection of files that describe a related set of Kubernetes resources.

③ Repositories

helm repo add bitnami https://charts.bitnami.com/bitnami



→ pull chart.

→ install.

→ Release

helm install.

→ install
→ upgrad
→ rollback
→ delete.

A **release** is a **deployed instance** of a Helm chart.

Every time you install a chart using `helm install`, Helm creates a new **release**.

List Helm Releases

helm list

Upgrade a Release

helm upgrade my-nginx bitnami/nginx --set replicaCount=2

helm list -> revision 2

kubectl get pods -> 2 pods

Rollback to a Previous Version

helm rollback my-nginx

kubectl get pods -> 1 pod

To rollback to a specific revision (e.g., revision 2), run:

helm rollback myrelease 2

Delete a Release

helm uninstall my-nginx

History of releases

helm history my-nginx

Benefits of Helm.

1. Simplified Deployment.
2. Version Control and Rollback
3. Reusable Templates.

→ Helm Chart

→ Structure of the chart

→ Chart.yaml
(metadata)

→ values.yaml. & template directory.

→ values

```
replicaCount: 2

image:
  repository: nginx
  tag: latest

service:
  type: ClusterIP
  port: 80

ingress:
  enabled: true
  host: example.com
```

Stores user-configurable values that templates use.

replicas: {{ .Values.replicaCount }}

template/

File	Purpose
NOTES.txt	Post-installation instructions for users.
_helpers.tpl	Stores reusable template functions.
deployment.yaml	Defines a Deployment for the application.
hpa.yaml	Defines a HorizontalPodAutoscaler (HPA).
ingress.yaml	Defines an Ingress resource for the app.
service.yaml	Defines a Service to expose the app.
serviceaccount.yaml	Defines a ServiceAccount.
tests/test-connection.yaml	Used for testing the Helm chart deployment.

_helpers.tpl

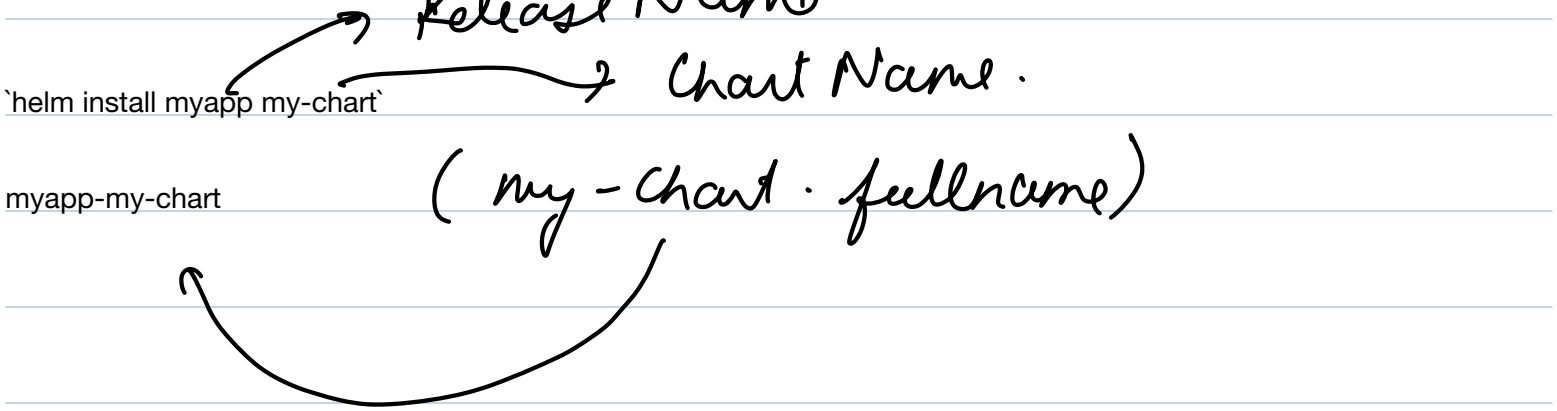
used to define **template functions and reusable variables**. It helps **avoid duplication** in YAML templates and

makes Helm charts **more maintainable**.

{{- define "my-chart.fullname" -}} → Name of function.

{{ .Release.Name }}-{{ .Chart.Name }}

{{- end -}}




```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: {{ include "my-chart.fullname" . }}
  labels:
    app: {{ include "my-chart.fullname" . }}
spec:
  replicas: 1
  selector:
    matchLabels:
      app: {{ include "my-chart.fullname" . }}
  template:
    metadata:
      labels:
        app: {{ include "my-chart.fullname" . }}
    spec:
```

name: myapp-my-chart

```
    containers:
      - name: my-app
        image: nginx
```

```
{{- define "my-chart.labels" -}}
app.kubernetes.io/name: {{ .Chart.Name }}
app.kubernetes.io/instance: {{ .Release.Name }}
app.kubernetes.io/version: {{ .Chart.Version }}
{{- end -}}
```

```
apiVersion: v2
name: my-chart
description: A Helm chart for Kubernetes
version: 1.0.0
```

Chart.yaml.

```
helm install myapp my-chart
```

Release → myapp
Chart → my-chart

```
metadata:
  labels:
    app.kubernetes.io/name: my-chart
    app.kubernetes.io/instance: myapp
    app.kubernetes.io/version: 1.0.0
```

test / test-connection.yaml.

(3) Charts Directory & Wrapper Charts.

subcharts → dependencies

A **wrapper chart** is a **higher-level Helm chart** that groups multiple subcharts (dependencies) under one umbrella.

Combo Meal → Wrapper Chart.

Why use wrapper chart

1. **Centralized Deployment**: Deploy and manage multiple applications with one chart.

2. **Consistent Configurations**: Apply common settings to all sub-charts.

3. **Simplifies Helm Operations**: Instead of handling many releases, just manage one.

4. **Overrides Default Values**: Customize sub-charts without modifying their source.

charts /

↓

→ Storing dependent charts are stored here.

Break → 10:34 pm

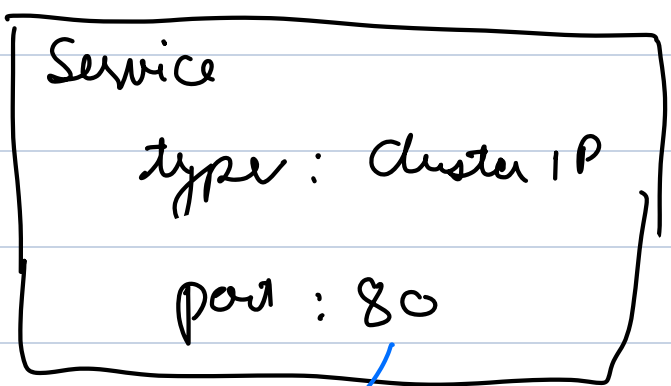
→ Helm Templating language

① Variables. {{ }}

② Values. values ← values.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: {{ .Release.Name }}-service
spec:
  type: {{ .Values.service.type }}
  ports:
    - port: {{ .Values.service.port }}
      targetPort: 80
  selector:
    app: myapp
```

→ values.yaml.



(3) Functions

```
apiVersion: v1
kind: Service
metadata:
  name: {{ .Release.Name }}-{{ .Values.service.name | lower }}
spec:
  type: {{ .Values.service.type }}
  ports:
    - port: {{ .Values.service.port }}
      targetPort: 80
  selector:
    app: myapp
```

Cluster IP

cluster ip.

(4) Control structures.

→ if

```
{{- if .Values.ingress.enabled }}
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
```

values.yaml

ingress
enabled: false

```
  name: {{ .Values.appName }}
spec:
  rules:
  - host: {{ .Values.ingress.host }}
{{- end }}
```

Loops → for iterating over lists

↓

→ range

```
ports:
{{- range .Values.service.ports }}
  - name: {{ .name }}
    port: {{ .port }}
    targetPort: {{ .targetPort }}
{{- end }}
```

deployment template

values.yaml

```
service:
  ports:
    - name: http
      port: 80
      targetPort: 8080
```

Value. yaml

- name: https

  port: 443

  targetPort: 8443

→ Flags -f --values.

helm install myrelease myapp -f custom-values.yaml --namespace mynamespace

helm install myrelease myapp -f values1.yaml -f values2.yaml --namespace mynamespace

values1.yaml values2.yaml

Service.type : LoadBalancer

Service.type : ClusterIP

→ Demo

helm plugin install https://github.com/databus23/helm-diff

helm diff revision 1 2

→ created helm chart

helm create nginx-website

→ update values.yaml

```
replicaCount: 2

image:
  repository: nginx
  tag: latest
  pullPolicy: IfNotPresent

service:
  type: ClusterIP
  port: 80

config:
  indexHtml: |-
    <!DOCTYPE html>
    <html>
    <head>
        <title>Welcome to My Helm-Deployed Website</title>
    </head>
    <body>
        <h1>Successfully deployed using Helm!</h1>
```

```
    </body>
    </html>
```

configmap.yaml

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: {{ .Release.Name }}-config
data:
  index.html: |
    {{- if .Values.config }}
    {{- .Values.config.indexHtml | nindent 4 }}
    {{- else }}
    Default index.html content
    {{- end }}
```

deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
```

```
  name: {{ .Release.Name }}
spec:
  replicas: {{ .Values.replicaCount }}
  selector:
    matchLabels:
      app: {{ .Release.Name }}
  template:
    metadata:
      labels:
        app: {{ .Release.Name }}
    spec:
      containers:
      - name: nginx
        image: "{{ .Values.image.repository }}:{{ .Values.image.tag }}"
        imagePullPolicy: {{ .Values.image.pullPolicy }}
        volumeMounts:
        - name: html-config
          mountPath: /usr/share/nginx/html
      volumes:
      - name: html-config
        configMap:
          name: {{ .Release.Name }}-config
```

Service.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: {{ .Release.Name }}
spec:
  type: {{ .Values.service.type }}
  selector:
    app: {{ .Release.Name }}
  ports:
  - protocol: TCP
    port: {{ .Values.service.port }}
    targetPort: 80
```

Install the Helm Chart

```
helm install my-nginx ./nginx-website
```

Port forward

kubectl port-forward svc/my-nginx 8080:80

→ Helm Package

helm package ./nginx-website → .tar.gz

helm registry login registry-1.docker.io --username vedant120

helm push nginx-website-0.1.0.tgz oci://registry-1.docker.io/vedant120

helm show chart oci://registry-1.docker.io/vedant120/nginx-website

get manifest for what is currently running

helm get manifest my-release

Manifest for what is going to be applied

helm template my-release ~/my-chart