HELM

1. **What is HELM**

Helm is a package manager for kubernetes,

In kubernetes, we will create multiple yaml files right for each kubernetes object like –

1 yaml file for Deployment object

1 yaml file for Pod

1 yaml file for Service-and 1 for each sub type right,

1 yaml file for Ingress

In kubernetes we will execute this file using command “kubectl –f apply.yaml”.

Whereas in HELM,if u run 1 command all those file will be executed and objects will be created in kubernetes cluster

1. Helm architecture

Generally we will place all the yaml files like deployment.yaml .. everything under templates folder

But values.yaml file alone will be under root folder-not inside templates folder

As mentioned above we have many yaml files, like deployment.yaml, pod.yaml, secret.yaml, configmap.yaml will be packaged and executed as one.

HELM as a package manager will package all the yaml files and create single helm chart

3)Helm advantages

All the yaml files (deployment.yaml,pod,yaml,service.yaml,hpa.yaml which are present in the templates folder) will be deployed in single shot and everything is versioned means it will maintain versions

4)Reference links

If u want to see sample deployment.yaml file,where u can fetch data from values.yaml file then refer below

<https://github.com/bitnami/charts/blob/main/bitnami/nats/templates/deployment.yaml>

Kubernetes commands

1) Kub cluster commands

#### Get cluster info

kubectl cluster-info

To get nodes

kubectl get nodes

1. Helm Basic commands

If u are giving any input values to in the command, then u should pass double hyphen

kubectl config set-context --current --namespace=default

this is not only here, this is in all languages

1. Create a helm project

Chart contains kubernetes yaml file

helm create <desired-helm-project folder name>

helm create helloworld

if u execute it, it will create a folder called “helloworld” where it contains charts..

1. Values.yaml

If u want to fetch anything from values.yaml being in any other file ,we have to use

“{{ .Values.hostAliases }}” whereas Values. Refers to values.yaml file

Sample values.YAML



1. Install-uninstall all the yaml files

In simple words-helm will execute all those yaml files and creates kubernetes objects as mentioned as per

Yaml files in the namespace.

Install means, helm will package & combine all yaml together & makes a single executable file and it will execute that file

Single yaml file

helm install <release-name> <foldername where all files are present>

Multiple yaml files

when multiple yaml files are present 1 per environment basis, then we should install as below

in case of multiple yaml files, we have to tell which yaml file and where is that yaml file

helm install <release-name> <folder-name> ---values ./<foldername>/<values-\*.yaml>

helm install v1 proj1 --values proj1/values-dev.yaml

or by mentioning the current directory

**helm install r1 proj1** -**-values ./proj1/values-dev.yaml**

Uninstall the installed,

If u uninstall all the deployed objects present in that namespace will be uninstalled or deleted

helm uninstall <release-name>

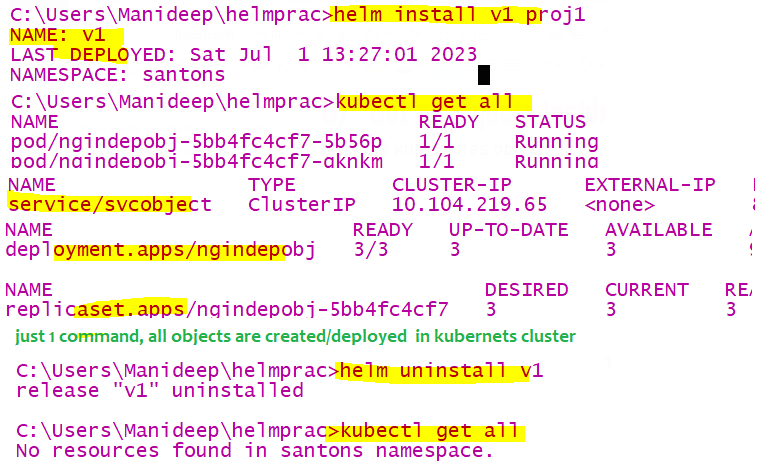
When u install many kubernetes objects like pod,deployment,service objects would have got created,

If u want to uninstall all those objects at once then use this command

Some releases will be another namespace, so u have to go to that namespace and delete else, u can use –n flag

**helm uninstall v5 -n manins**

**helm uninstall v4 -n manins**



1. Upgrade the image name

example project:- for simplicity purpose, keep only deployment,hpa,service yaml files and remove everything

when u made any changes like updating the image name in deployment.yaml or if u made any changes in any yaml

if u want to uninstall all objects and create all objects again **,then prefer upgrade**

Let’s say if u modified the project and then no need to helm install, u should do

helm upgrade <existing-release-name> <folder-name>

helm upgrade <release-name> <folder-name> ---values ./<foldername>/<values-\*.yaml>

helm upgrade r1 proj1 --values ./proj1/values-dev.yaml

lets say u upgrade it and u want to see what is the revision made ,then type “helm history r1”

1. List out all helm release

|  |  |
| --- | --- |
| To list out all helm releases | helm list |
| if u want to get all releases (some release build number) from all namespaces | helm list -A |
|  |  |
|  |  |

1. Get version/revision history

helm history <release-name>

helm history v1

1. Rollback to older release

helm rollback <release-name>

helm rollback r1