Create your own Helm Cart

* Create a Helm Chart

***helm create mychart***

|  |
| --- |
| [postgres-chart](https://github.com/deekshahegde86/postgres-helm-operator/tree/master/postgres-chart)  |-- Chart.yaml  |-- charts  |-- templates  | |-- NOTES.txt  | |-- \_helpers.tpl  | |-- deployment.yaml  | |-- ingress.yaml  | `-- service.yaml  `-- values.yaml |

This command scaffolds the above set of directories and files.

* Templates DIR will contain all the deployment yaml files you need to perform required application deployment and its required resources (ex: PVC ,Secrets,Service,Configmap etc).
  + Fill in deployment.yaml file to do the deployments like creation of pods,deployments,stateful sets etc.
  + Write additional yaml files for additional resources.
  + If you don’t need any of the precreated yaml files (example : ingress.yaml) just leave it as it.
  + Refer following link to see yaml definitions used for creating simple postgresql deployment with a service and pvc using helm chart : <https://github.com/deekshahegde86/postgres-helm-operator/tree/master/postgres-chart/templates>
* Fill in the values.yaml file with required spec that your templates would use.
* Check syntax issues.

***helm lint ./mychart***

* Perform a Dry Run to see if all the values from Values.yaml is incorporated in your templates/\*yaml files when helm install is performed.

***helm install --dry-run --debug ./mychart***

* When the review of dry-run passes your review, you are ready to perform the actual install.

***helm install example ./mychart***

* Create a package of your chart if desired. This will create a .tgz file.

***helm package ./mychart***

* If you wish to install helm chart from a .tgz package use below command:

***helm install example3 mychart-0.1.0.tgz***

Create a Helm Operator

* Run the Operator-sdk command to scaffold the operator

***operator-sdk new postgres-helm-operator --api-version=postgres.example.com/v1alpha1 --kind=Postgres --type=helm --helm-chart=<full-path to your helm chart>***

*Note: Kind should begin with upper case lettor*

*Group/apiversion: Should be a dot notation string and version should be v1alpha1.*

|  |
| --- |
| ***postgres-helm-operator***  ***├── build***  ***│   └── Dockerfilel***  ***├── deploy***  ***│   ├── crds***  ***│   │   ├── postgres.example.com\_postgres\_crd.yaml***  ***│   │   └── postgres.example.com\_v1alpha1\_postgres\_cr.yaml***  ***│   ├── operator.yaml***  ***│   ├── role\_binding.yaml***  ***│   ├── role.yaml***  ***│   └── service\_account.yaml***  ***├── helm-charts***  ***│   └── postgres-chart***  ***│   ├── Chart.yaml***  ***│   ├── templates***  ***│   │   ├── deployment.yaml***  ***│   │   ├── \_helpers.tpl***  ***│   │   ├── ingress.yaml***  ***│   │   ├── NOTES.txt***  ***│   │   ├── primary-pvc.yaml***  ***│   │   ├── serviceaccount.yaml***  ***│   │   ├── service.yaml***  ***│   │   └── tests***  ***│   │   └── test-connection.yaml***  ***│   └── values.yaml***  ***└── watches.yaml*** |

The operator-sdk scaffolds the above set of files/DIR.

* + Build : Containd Docker file that is used to build your Operator Image.
  + Deploy : Contains
    - CRD,CR for your custom resources with Group,Kind,Version
    - Operator.yaml file : Used to create your operator.
    - Role.yaml : Defines roles an permission for the resources your role can access.
    - Serviceaccount.yaml : Service account for your operator deployment.
    - Role\_binding.yaml: binds your role with serviceaccount.
  + Helm-chart: Your helm chart that will be triggered for respective CR created for the defined CRD.
  + Watches.yaml: file that watched CR of a particular Group,Kind,Version and states which chart to invoke when an object of mentioned group,king,version is created.
* Note that the CR spec automatically is populated with key-value from your charts values.yaml.
* Build the operator image and push the operator image to your quay repo.

***operator-sdk build quay.io/deekshah86/postgres-helm-operator:v0.0.1***

***docker push quay.io/deekshah86/postgres-helm-operator:v0.0.1***

* Edit and mention the right path to your operator image in deploy/Operator.yaml file. (your quay repo)

**REPLACE\_IMAGE** with **quay.io/deekshah86/postgres-helm-operator in deploy/operator.yaml**

* **Create your new project:**

***oc new-project <project-name>***

* Ensure you are inside the operator DIR while you run the below command. Example In above tree structure the desired DIR to be in while running the command is: postgres-helm-operator
* Create resources:

***oc create -f deploy/crds/postgres.example.com\_postgres\_crd.yaml***

***oc create -f deploy/service\_account.yaml***

***oc create -f deploy/role.yaml***

***oc create -f deploy/role\_binding.yaml***

* Now Create an instance of your operator.

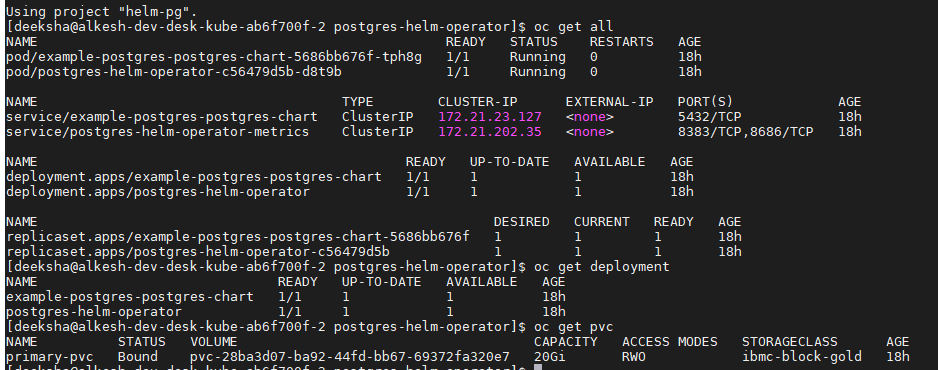
***oc create -f deploy/operator.yaml***

* Create an instance of your CR:

***oc create -f deploy/crds/postgres.example.com\_v1alpha1\_postgres\_cr.yaml***

* ***Check if the deployment is created and desired resources are created:***

|  |  |
| --- | --- |
| ***oc get deployment*** | ***oc get pods*** |
| ***oc get pvc*** | ***oc get all*** |

******