

Desafío # 9

Fecha de entrega: 22/01/2025

Objetivo:

El objetivo de este desafío es poner en práctica lo visto sobre helm y desarrollar nuestro propio helm chart, tomando como entrada lo visto en los desafíos 5 y 8.

Escenario:

Nuestro equipo identificó que Kubernetes agrega mucho valor a la hora de mantener los deployments de nuestra aplicación y ha estado analizando la manera en que gestionamos el código de estos deployments.

Luego de varias reuniones, identificaron que los manifiestos duplican mucho código y que esto se puede resolver utilizando algún sistema de templates. Durante este sprint, se nos asignó la tarea de desarrollar un Helm chart para gestionar el deployment de la aplicación. Este chart debe desplegar la aplicación y el servicio de base de datos MongoDB para almacenar los datos de nuestra aplicación.

La aplicación que va a ser manejada por este proceso se encuentra en el siguiente enlace:

<https://github.com/yosoyfunes/app-template-nestjs>

Requisitos:

1. Elaborar el Chart para realizar el deployment.

2. Redactar la documentación necesaria.

Entregables:

Los entregables establecidos para este proyecto con:

1. Código fuente de todo lo producido.
2. Documentación.
3. Evidencia de las pruebas con resultado exitoso.

Evaluación:

- Entrega en fecha.
- Redactar documentación legible y que sea comprendida por terceros..
- Añade material de soporte adicional.
 - Ejemplo: Diagrama de alto nivel.
- Cumple con las consignas solicitadas.
- El entregable es funcional.
 - Ejemplo: el script bash al ejecutarse funciona sin errores y realiza lo solicitado.

Documentos de referencia:

- [Documentación NestJS.](#)
- [Manifiesto Kubernetes](#)
- [Helm - Chart Development Tips](#)

Solución:

Setup:

Voy a instalar helm en mi server Ubuntu siguiendo las instrucciones de aqui:

<https://helm.sh/docs/intro/install/#from-script>

```
(ubuntuuser) 172.20.0.209 — Konsole
New Tab Split View
Copy Paste Find...
ubuntuuser@ubuntu:~/Desf9$ curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3
ubuntuuser@ubuntu:~/Desf9$ chmod 700 get_helm.sh
ubuntuuser@ubuntu:~/Desf9$ ./get_helm.sh
Downloading https://get.helm.sh/helm-v3.17.0-linux-amd64.tar.gz
Verifying checksum... Done.
Preparing to install helm into /usr/local/bin
[sudo] password for ubuntuuser:
helm installed into /usr/local/bin/helm
ubuntuuser@ubuntu:~/Desf9$ helm version
version.BuildInfo{Version:"v3.17.0", GitCommit:"301108edc7ac2a8ba79e4ebf5701b0b6ce6a31e4", GitTreeState:"clean", GoVersion:"go1.23.4"}
ubuntuuser@ubuntu:~/Desf9$
```

Implementación:

Voy a crear el directorio del chart con `helm create devops2025-chart`

```
ubuntuuser@ubuntu:~/Desf9$ helm create devops2025-chart
Creating devops2025-chart
ubuntuuser@ubuntu:~/Desf9$ tree
.
├── devops2025-chart
│   ├── charts
│   ├── Chart.yaml
│   ├── templates
│   │   ├── deployment.yaml
│   │   ├── _helpers.tpl
│   │   ├── hpa.yaml
│   │   ├── ingress.yaml
│   │   ├── NOTES.txt
│   │   ├── serviceaccount.yaml
│   │   ├── service.yaml
│   │   └── tests
│   │       └── test-connection.yaml
│   └── values.yaml
└──
5 directories, 10 files
ubuntuuser@ubuntu:~/Desf9$
```

Voy a eliminar todos los .yaml en el directorio templates y reemplazarlos por los yaml del desafío anterior.

```
(ubuntuuser) 172.20.0.209 — Konsole
New Tab Split View
Copy Paste Find...
ubuntuuser@ubuntu1ab:~/Desf9$ helm create devops2025-chart
Creating devops2025-chart
ubuntuuser@ubuntu1ab:~/Desf9$ tree
.
├── devops2025-chart
│   ├── charts
│   ├── Chart.yaml
│   ├── templates
│   │   ├── deployment.yaml
│   │   ├── _helpers.tpl
│   │   ├── hpa.yaml
│   │   ├── ingress.yaml
│   │   ├── NOTES.txt
│   │   ├── serviceaccount.yaml
│   │   ├── service.yaml
│   │   └── tests
│   │       └── test-connection.yaml
│   └── values.yaml
5 directories, 10 files
ubuntuuser@ubuntu1ab:~/Desf9$ rm devops2025-chart/templates/*.yaml
ubuntuuser@ubuntu1ab:~/Desf9$ cp ../Desf8/*.yaml devops2025-chart/templates/
ubuntuuser@ubuntu1ab:~/Desf9$ tree
.
├── devops2025-chart
│   ├── charts
│   ├── Chart.yaml
│   ├── templates
│   │   ├── app-deployment.yaml
│   │   ├── app-service.yaml
│   │   ├── _helpers.tpl
│   │   ├── mongodb-deployment.yaml
│   │   ├── mongodb-service.yaml
│   │   ├── NOTES.txt
│   │   └── tests
│   │       └── test-connection.yaml
│   └── values.yaml
5 directories, 9 files
ubuntuuser@ubuntu1ab:~/Desf9$
```

Validación:

En este punto ya se puede instalar el chart con: `helm install devops2025-chart ./devops2025-chart`

Verificando con `kubectl get all` se verifica que el estado del cluster coincide con el esperado (del desafío anterior)

```
(ubuntuser) 172.20.0.209 — Konsole
New Tab Split View
Copy Paste Find...
ubuntuser@ubuntu:~$ helm install devops2025-chart ./devops2025-chart
NAME: devops2025-chart
LAST DEPLOYED: Thu Jan 30 20:09:24 2025
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
1. Get the application URL by running these commands:
  export POD_NAME=$(kubectl get pods --namespace default -l "app.kubernetes.io/name=devops2025-chart,app.kubernetes.io/instance=devops2025-chart" -o jsonpath="{.items[0].metadata.name}")
  export CONTAINER_PORT=$(kubectl get pod --namespace default $POD_NAME -o jsonpath="{.spec.containers[0].ports[0].containerPort}")
  echo "Visit http://127.0.0.1:8080 to use your application"
  kubectl --namespace default port-forward $POD_NAME 8080:$CONTAINER_PORT
ubuntuser@ubuntu:~$ kubectl get all
NAME                                READY    STATUS    RESTARTS   AGE
pod/app-546ccc4dd6-zh8xl            1/1      Running   0           24s
pod/mongodb-567b7775d4-wgnwl        1/1      Running   0           24s

NAME                                TYPE                CLUSTER-IP      EXTERNAL-IP    PORT(S)          AGE
service/app                        NodePort            10.97.239.93    <none>          3000:30000/TCP   25s
service/kubernetes                 ClusterIP           10.96.0.1       <none>          443/TCP          5h59m
service/mongodb                    ClusterIP           10.102.181.184  <none>          27017/TCP        25s

NAME                                READY    UP-TO-DATE    AVAILABLE    AGE
deployment.apps/app                 1/1      1              1            25s
deployment.apps/mongodb             1/1      1              1            25s

NAME                                DESIRED    CURRENT    READY    AGE
replicaset.apps/app-546ccc4dd6      1          1          1        24s
replicaset.apps/mongodb-567b7775d4  1          1          1        24s
ubuntuser@ubuntu:~$
```

Implementación de variables:

voy a pasar algunas variables al archivo de valores.yaml. voy a modificar los manifiestos. Para la app voy a definir:

- nombre de la imagen
- número de réplicas
- puerto nodeport

```
(ubuntuser) 172.20.0.209 — Konsole
New Tab Split View
GNU nano 7.2 values.yaml *
appReplicaCount: 1

appImage:
  repository: jcavaiuolo/app
  tag: latest

serviceNodePort: 30000
```

y ajusto los manifiestos de app-deployment y app-service:

```
! app-deployment.yaml.bak -- app-deployment.yaml X ! app-deployment.yaml -- app-deployment.yaml ~/DevOps2025-chart/templates app-deployment.yaml.bak ! app-service.yaml
Desf9 > devops2025-chart > templates > ! app-deployment.yaml
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   annotations:
5     kompose.cmd: kompose convert
6     kompose.version: 1.35.0 (9532ceef3)
7   labels:
8     io.kompose.service: app
9   name: app
10 spec:
11   replicas: 1
12   selector:
13     matchLabels:
14       io.kompose.service: app
15   template:
16     metadata:
17       annotations:
18         kompose.cmd: kompose convert
19         kompose.version: 1.35.0 (9532ceef3)
20       labels:
21         io.kompose.service: app
22     spec:
23       containers:
24       - image: jcavaiuolo/app:latest
25         name: app
26         ports:
27         - containerPort: 3000
28         protocol: TCP
29       restartPolicy: Always

1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   annotations:
5     kompose.cmd: kompose convert
6     kompose.version: 1.35.0 (9532ceef3)
7   labels:
8     io.kompose.service: app
9   name: app
10 spec:
11   replicas: {{ .Values.appReplicaCount }}
12   selector:
13     matchLabels:
14       io.kompose.service: app
15   template:
16     metadata:
17       annotations:
18         kompose.cmd: kompose convert
19         kompose.version: 1.35.0 (9532ceef3)
20       labels:
21         io.kompose.service: app
22     spec:
23       containers:
24       - image: "{{ .Values.appImage.repository }}:{{ .Values.appImage.tag }}"
25         name: app
26         ports:
27         - containerPort: 3000
28         protocol: TCP
29       restartPolicy: Always

! app-deployment.yaml.bak -- app-deployment.yaml ! app-service.yaml ! app-service.yaml.bak -- app-service.yaml
Desf9 > devops2025-chart > templates > ! app-service.yaml
1 apiVersion: v1
2 kind: Service
3 metadata:
4   annotations:
5     kompose.cmd: kompose convert
6     kompose.version: 1.35.0 (9532ceef3)
7   labels:
8     io.kompose.service: app
9   name: app
10 spec:
11   type: NodePort # Cambiar el tipo a NodePort
12   ports:
13     - name: "3000"
14       port: 3000 # El puerto dentro del clúster
15       targetPort: 3000 # El puerto que escucha el contenedor de la app
16       nodePort: 30000 # El puerto asignado en la máquina local
17   selector:
18     io.kompose.service: app

1 apiVersion: v1
2 kind: Service
3 metadata:
4   annotations:
5     kompose.cmd: kompose convert
6     kompose.version: 1.35.0 (9532ceef3)
7   labels:
8     io.kompose.service: app
9   name: app
10 spec:
11   type: NodePort # Cambiar el tipo a NodePort
12   ports:
13     - name: "3000"
14       port: 3000 # El puerto dentro del clúster
15       targetPort: 3000 # El puerto que escucha el contenedor de la app
16       nodePort: {{ .Values.serviceNodePort }} # El puerto asignado en la
17   selector:
18     io.kompose.service: app
```

Y, dado que en el ejercicio anterior me dio problemas la versión de la imagen de mongodb voy a usar la versión como variable en el deployment para el pod de mongo:

agregando mongoImageTag: 4.4 al values.yaml queda así:

```
(ubuntuser) 172.20.0.209 — Konsole
GNU nano 7.2 values.yaml *
appReplicaCount: 1
appImage:
  repository: jcavaiuolo/app
  tag: latest
serviceNodePort: 30000
mongoVersion: 4.4
```

y actualizo el manifiesto del deployment para que use esta versión:

```
Desf9 > devops2025-chart > templates > / mongodb-deployment.yaml
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   annotations:
5     kompose.cmd: kompose convert
6     kompose.version: 1.35.0 (9532ceef3)
7   labels:
8     io.kompose.service: mongodb
9   name: mongodb
10 spec:
11   replicas: 1
12   selector:
13     matchLabels:
14       io.kompose.service: mongodb
15   template:
16     metadata:
17       annotations:
18         kompose.cmd: kompose convert
19         kompose.version: 1.35.0 (9532ceef3)
20       labels:
21         io.kompose.service: mongodb
22     spec:
23       containers:
24         - image: mongo:4.4
25           name: mongodb
26           ports:
27             - containerPort: 27017
28             protocol: TCP
29           restartPolicy: Always
30
```

```
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   annotations:
5     kompose.cmd: kompose convert
6     kompose.version: 1.35.0 (9532ceef3)
7   labels:
8     io.kompose.service: mongodb
9   name: mongodb
10 spec:
11   replicas: 1
12   selector:
13     matchLabels:
14       io.kompose.service: mongodb
15   template:
16     metadata:
17       annotations:
18         kompose.cmd: kompose convert
19         kompose.version: 1.35.0 (9532ceef3)
20       labels:
21         io.kompose.service: mongodb
22     spec:
23       containers:
24         - image: "mongo:{{ .Values.mongoVersion }}"
25           name: mongodb
26           ports:
27             - containerPort: 27017
28             protocol: TCP
29           restartPolicy: Always
30
```

Troubleshooting:

Como borré el archivo values.yaml, algunas de las funciones preconstruidas en el helm chart dejaron de funcionar. Así que tuve que hacer algunos ajustes a las variables. La versión final del archivo values quedó así:

```
(ubuntuser) 172.20.0.209 — Konsole
New Tab Split View
ubuntuser@ubuntu:~/Desf9$ helm lint ./devops2025-chart/
==> Linting ./devops2025-chart/
[INFO] Chart.yaml: icon is recommended

1 chart(s) linted, 0 chart(s) failed
ubuntuser@ubuntu:~/Desf9$ nano devops2025-chart/values.yaml
ubuntuser@ubuntu:~/Desf9$ cat devops2025-chart/values.yaml

appReplicaCount: 2

appImage:
  repository: jcavaiuolo/app
  tag: latest

serviceNodePort: 30002 # esta variable quedaria mejor en la seccion "service"

mongoVersion: 4.4

ingress:
  enabled: false

service:
  type: NodePort
ubuntuser@ubuntu:~/Desf9$
```

Nota: cambie el número de réplicas a 2 y el nodeport a 30002 solo para poder ver el cambio en el estado del cluster en el paso de verificación.

Verificación:

Luego de hacer `helm install devops2025-chart ./devops2025-chart` y verificar el STATUS: deployed. Se puede ejecutar `kubectl get all` y se verifica el despliegue. Particularmente los 2 pods de app, y el nodeport seteado a 30002 en el paso anterior.

```
(ubuntuser) 172.20.0.209 — Konsole
New Tab Split View
Copy Paste Find...
ubuntuser@ubuntu1ab:~/Desf9$ kubectl get all
NAME                TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
service/kubernetes  ClusterIP     10.96.0.1     <none>         443/TCP          7h9m
ubuntuser@ubuntu1ab:~/Desf9$ helm install devops2025-chart ./devops2025-chart
NAME: devops2025-chart
LAST DEPLOYED: Thu Jan 30 21:20:34 2025
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
1. Get the application URL by running these commands:
  export NODE_PORT=$(kubectl get --namespace default -o jsonpath="{.spec.ports[0].nodePort}" services devops2025-chart)
  export NODE_IP=$(kubectl get nodes --namespace default -o jsonpath="{.items[0].status.addresses[0].address}")
  echo http://$NODE_IP:$NODE_PORT
ubuntuser@ubuntu1ab:~/Desf9$ kubectl get all
NAME                READY   STATUS    RESTARTS   AGE
pod/app-546ccc4dd6-8dlmd    1/1     Running   0          56s
pod/app-546ccc4dd6-w2chv    1/1     Running   0          56s
pod/mongodb-567b7775d4-fktgm 1/1     Running   0          56s

NAME                TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
service/app         NodePort      10.103.97.90   <none>         3000:30002/TCP   57s
service/kubernetes  ClusterIP     10.96.0.1     <none>         443/TCP          7h10m
service/mongodb     ClusterIP     10.110.151.166 <none>         27017/TCP        57s

NAME                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/app  2/2     2            2          57s
deployment.apps/mongodb 1/1     1            1          57s

NAME                DESIRED   CURRENT   READY   AGE
replicaset.apps/app-546ccc4dd6 2         2         2       57s
replicaset.apps/mongodb-567b7775d4 1         1         1       57s
ubuntuser@ubuntu1ab:~/Desf9$
```

al obtener la ip de minikube con `minikube ip`. se puede hacer un curl al puerto expuesto en el cluster y se valida la comunicación con la aplicación:

```
ubuntuser@ubuntu1ab:~/Desf9$ minikube ip
192.168.67.2
ubuntuser@ubuntu1ab:~/Desf9$ curl 192.168.67.2:30002
Hello World!ubuntuser@ubuntu1ab:~/Desf9$
```