

Helm

Antonio Espín Herranz

Contenidos

- Despliegue automatizado con **Helm**:
 - Uso de Helm Charts para la automatización de despliegues y actualizaciones.
 - Gestión de configuraciones de microservicios en entornos de producción

Helm

- Es un **gestor de paquetes de Kubernetes**.
- Nos permite instalar, actualizar y gestionar aplicaciones dentro del clúster de **K8S**.
- Se puede instalar con chocolatey y si no, podemos descargar el release de **github**:
 - <https://github.com/helm/helm/releases>
 - Descargar: **Windows amd64**
 - Añadir al **PATH**
 - Prueba → **helm version**

Helm - charts

- Los charts son paquetes de aplicaciones listas para instalar.
- El siguiente paso es añadir un repositorio:
 - **helm repo add bitnami https://charts.bitnami.com/bitnami**
 - **helm repo update**
- Los repositorios se almacenan en la carpeta de usuario.
- Se pueden listar con:
 - **helm repo list**

```
C:\Users\Anton>helm repo add bitnami https://charts.bitnami.com/bitnami
"bitnami" has been added to your repositories

C:\Users\Anton>helm repo list
NAME      URL
bitnami   https://charts.bitnami.com/bitnami
```

Tipos de aplicaciones en helm

- **CMS:** WordPress, Drupal
- **Bases de datos:** MySQL, PostgreSQL, MongoDB
- **DevOps:** Jenkins, GitLab, Harbor
- **Monitoreo:** Prometheus, Grafana, ELK Stack
- **Seguridad:** Vault, Keycloak
- **Mensajería:** RabbitMQ, Kafka

Otros repositorios

Bitnami	Charts para apps populares como WordPress, MySQL, NGINX, etc.	bitnami
Artifact Hub	El buscador oficial de Helm charts, operadores, y más.	artifacthub.io
JBfrog ChartCenter	Repositorio centralizado de charts de múltiples fuentes.	chartcenter.io
Awesome Helm (GitHub)	Lista colaborativa con charts útiles y recursos adicionales.	awesome-helm

Instalar aplicaciones

- Instalar:
 - **helm install mi-wordpress bitnami/wordpress**
- Verificar:
 - **helm list**
 - **kubectl get all**
- Deinstalar:
 - **helm uninstall mi-wordpress**

Comprobar las instalaciones

helm list

kubectl get all

Para ver la URL (pero puede dar error) minikube service mi-wordpress --url

```
C:\Users\Anton>helm list
NAME      NAMESPACE    REVISION    UPDATED                               STATUS    CHART          APP VERSION
mi-wordpress  default      1           2025-10-05 11:59:25.590017 +0200 CEST  deployed  wordpress-26.0.0  6.8.2

C:\Users\Anton>kubectl get all
NAME                                     READY   STATUS              RESTARTS   AGE
pod/mi-wordpress-7dbdd64d47-5lfzp       0/1     Init:ErrImagePull    0           3m27s
pod/mi-wordpress-mariadb-0              0/1     Init:ImagePullBackOff 0           3m27s

NAME                                     TYPE          CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
service/kubernetes                      ClusterIP      10.96.0.1        <none>            443/TCP          106m
service/mi-wordpress                    LoadBalancer  10.107.64.195    <pending>         80:32750/TCP,443:30287/TCP  3m29s
service/mi-wordpress-mariadb            ClusterIP      10.108.139.146   <none>            3306/TCP          3m29s
service/mi-wordpress-mariadb-headless    ClusterIP      None             <none>            3306/TCP          3m29s

NAME                                     READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/mi-wordpress            0/1     1             0           3m28s

NAME                                     DESIRED   CURRENT   READY   AGE
replicaset.apps/mi-wordpress-7dbdd64d47 1         1         0       3m27s

NAME                                     READY   AGE
statefulset.apps/mi-wordpress-mariadb    0/1     3m28s
```


Crear chart

- Para crear un chart personalizado:
 - **helm create mi-microservicio**
- Crea una estructura básica que se puede modificar para empaquetar nuestra propia aplicación.

```
mi-microservicio/  
├── charts/           # Dependencias (otros charts)  
├── templates/       # Manifiestos YAML con variables  
│   ├── deployment.yaml  
│   ├── service.yaml  
│   └── ...  
├── values.yaml      # Configuración por defecto  
├── Chart.yaml       # Metadatos del chart  
└── README.md
```

Crear chart

- Obtener una **imagen** Docker del microservicio en C++ y publicarla en el **Docker Hub**
- Ejemplo: **values.yaml**
image:
 repository: piri12345/cpp-microservice
 tag: latest
 pullPolicy: IfNotPresent

Crear chart

- Configurar el **deployment: deployment.yaml**
 - Este archivo usa la **imagen** y crea el **pod**
- Configurar el **servicio: service.yaml**
 - Definir como se va a exponer el servicio: ClusterIP o NodePort
- **Instalar el chart en el cluster:**
 - `helm install mi-microservicio ./mi-microservicio`
- Actualizar o desinstalar:
 - `helm upgrade mi-microservicio ./mi-microservicio`
 - `helm uninstall mi-microservicio`

Tipo de servicio: ClusterIP / NodePort

- **ClusterIP (por defecto)**

- **Acceso interno** dentro del clúster.
- No se puede acceder desde fuera (por ejemplo, desde tu navegador local).
- Ideal para comunicación entre microservicios.
- Ejemplo: Un microservicio de autenticación que solo necesita ser accedido por otros servicios internos.

- **NodePort**

- Expone el servicio **fuera del clúster** a través de un puerto en cada nodo.
- Puedes acceder desde tu máquina usando la IP del nodo + el puerto asignado.
- Útil para pruebas locales o acceso básico externo.
- Ejemplo: Acceder a tu app desde `http://<IP-del-nodo>:<NodePort>`

Acceso externo / Comunicación Interna

- **1. Acceso externo → NodePort o LoadBalancer**

- Si el microservicio necesita ser accesible **desde fuera del clúster** (por ejemplo, desde tu navegador, una app móvil, o una API externa), entonces se tiene que exponer con:
 - **NodePort**: útil en entornos **locales** como **Minikube**.
 - Expone el servicio en un puerto del nodo.
 - **LoadBalancer**: **ideal** en la **nube** (AWS, GCP, Azure).
 - Asigna una IP pública automáticamente.
- En el caso de estar con Minikube, lo más práctico es usar NodePort.

- **2. Comunicación interna entre microservicios → ClusterIP**

- Los microservicios suelen necesitar hablar entre ellos (autenticación, base de datos, etc.). Para eso:
 - Se usa el tipo ClusterIP, que permite acceso **solo dentro del clúster**.
 - Kubernetes gestiona el DNS interno, se puede llamar a otro servicio por su nombre:
 - `http://nombre-del-servicio:puerto`