

*Antony Kervazo-Canut*

# Helm for Teenagers

**TEMPLATES DE  
CONFIGURATION KUBERNETES**



# SOMMAIRE

---

Introduction	3
Installation de Helm	4
Configuration de Helm	5
Les Charts Helm	6
Installation d'un Chart	7
Releases	8
Gestion des Releases	9
Création d'un Chart	10
Dépendances dans un Chart	14

# Introduction



Helm est un gestionnaire de paquets pour Kubernetes. Il permet aux développeurs et aux administrateurs systèmes de facilement déployer, configurer et gérer des applications sur des clusters Kubernetes. En utilisant Helm, vous pouvez empaqueter vos applications ainsi que toutes leurs dépendances dans un format standardisé appelé "chart".

- **Simplification du Déploiement:** Helm réduit la complexité des déploiements Kubernetes en gérant les paquets d'applications sous forme de charts.
- **Gestion des Configurations:** Les charts Helm permettent une configuration facile et répétable des applications Kubernetes, rendant possible le déploiement de différentes instances de l'application avec des configurations variées.
- **Réutilisabilité:** Les charts Helm sont conçus pour être partagés et réutilisés, facilitant ainsi la collaboration et la standardisation des déploiements au sein d'une organisation ou de la communauté.
- **Gestion des Dépendances:** Helm peut automatiquement télécharger et installer les dépendances nécessaires pour un chart, simplifiant la gestion des applications complexes.

En résumé, Helm est un outil essentiel pour tout utilisateur de Kubernetes, offrant une méthode standardisée et efficace pour le déploiement d'applications. Il encapsule les meilleures pratiques de déploiement dans Kubernetes, permettant ainsi aux équipes de se concentrer sur le développement d'applications plutôt que sur la gestion de leur déploiement.

# Installation de Helm



```
● ● ●

### LINUX ###

# Télécharger le script d'installation de Helm
# wget [URL] - commande pour télécharger des fichiers depuis
Internet
wget
https://raw.githubusercontent.com/helm/helm/main/scripts/get-
helm-3

# Rendre le script exécutable
# chmod +x [fichier] - change les permissions pour rendre un
fichier exécutable
chmod +x get-helm-3

# Exécuter le script d'installation
# ./[script] - exécute un script dans le shell
./get-helm-3

### WINDOWS ###

# Installation via Chocolatey, un gestionnaire de paquets
pour Windows
# choco install [paquet] - commande pour installer des
paquets avec Chocolatey
choco install kubernetes-helm

### MACOS ###

# Installation via Homebrew, un gestionnaire de paquets pour
macOS
# brew install [paquet] - commande pour installer des paquets
avec Homebrew
brew install helm
```

# Configuration de Helm



```
# Ajouter un référentiel de charts public
# helm repo add [nom] [URL] - ajoute un nouveau référentiel
# de charts
helm repo add stable https://charts.helm.sh/stable

# Ajouter un référentiel de charts privé avec
# authentification
# helm repo add [nom] [URL] --username [utilisateur] --
# password [mot_de_passe]
# Remplacez [utilisateur] et [mot_de_passe] par vos
# identifiants
helm repo add private-repo https://myprivaterrepo.com --
# username myuser --password mypassword

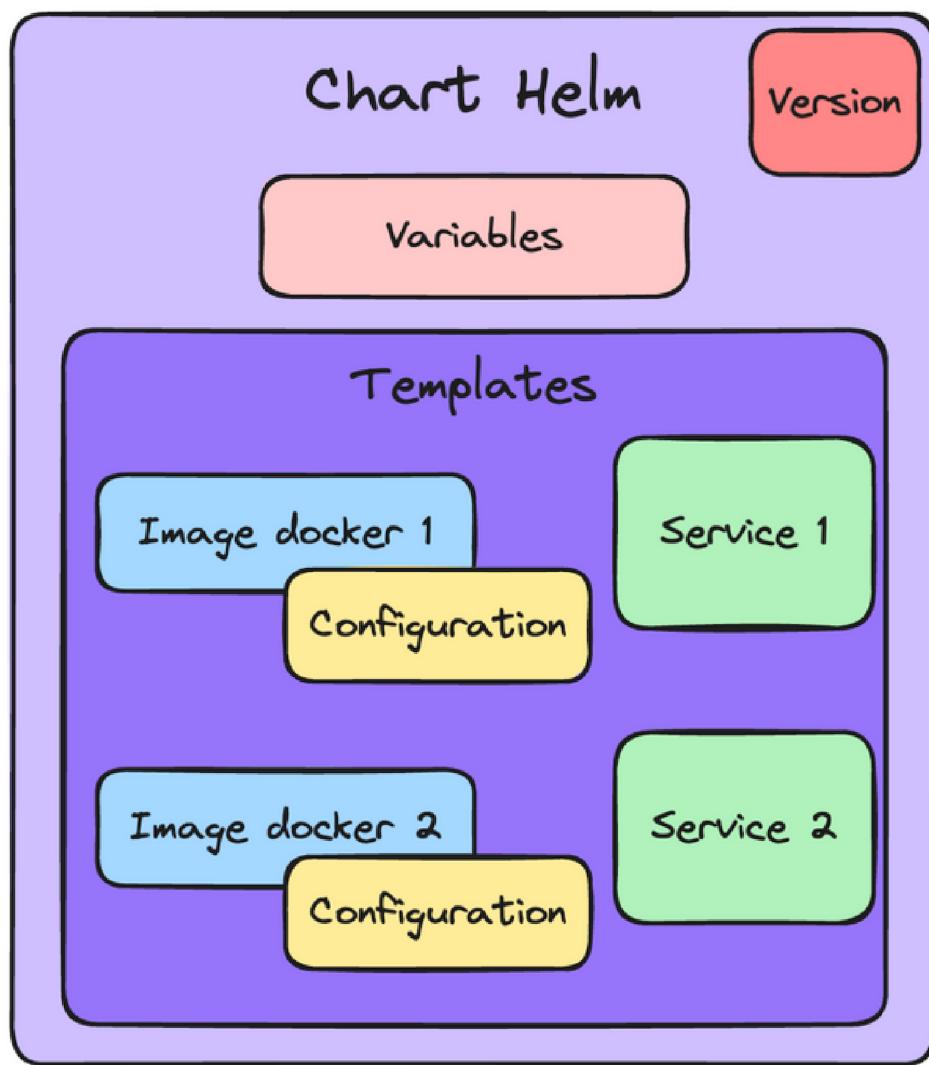
# Mettre à jour les informations du référentiel de charts
# helm repo update - met à jour la liste des charts
# disponibles dans les référentiels ajoutés
helm repo update

# Lister les référentiels de charts
# helm repo list - affiche tous les référentiels de charts
# configurés
helm repo list
```

# Les Charts Helm



Un chart Helm est un package qui simplifie le déploiement et la gestion d'applications sur Kubernetes. Pensez-y comme à une recette contenant des instructions et des fichiers nécessaires pour installer une application sur Kubernetes. Chaque chart contient des fichiers de configuration et des templates qui peuvent être personnalisés pour adapter l'application à différents environnements ou besoins spécifiques.



# Installation d'un Chart



```
# Rechercher des charts dans les référentiels
# helm search repo [mot-clé] - recherche des charts par mot-
clé
helm search repo nginx

# Installer un chart
# helm install [nom_release] [nom_chart] - installe un chart
dans le cluster
# Remplacez [nom_release] par le nom souhaité pour votre
déploiement
# Remplacez [nom_chart] par le nom du chart à installer
helm repo update # être sur d'être à jour
helm install my-nginx bitnami/nginx

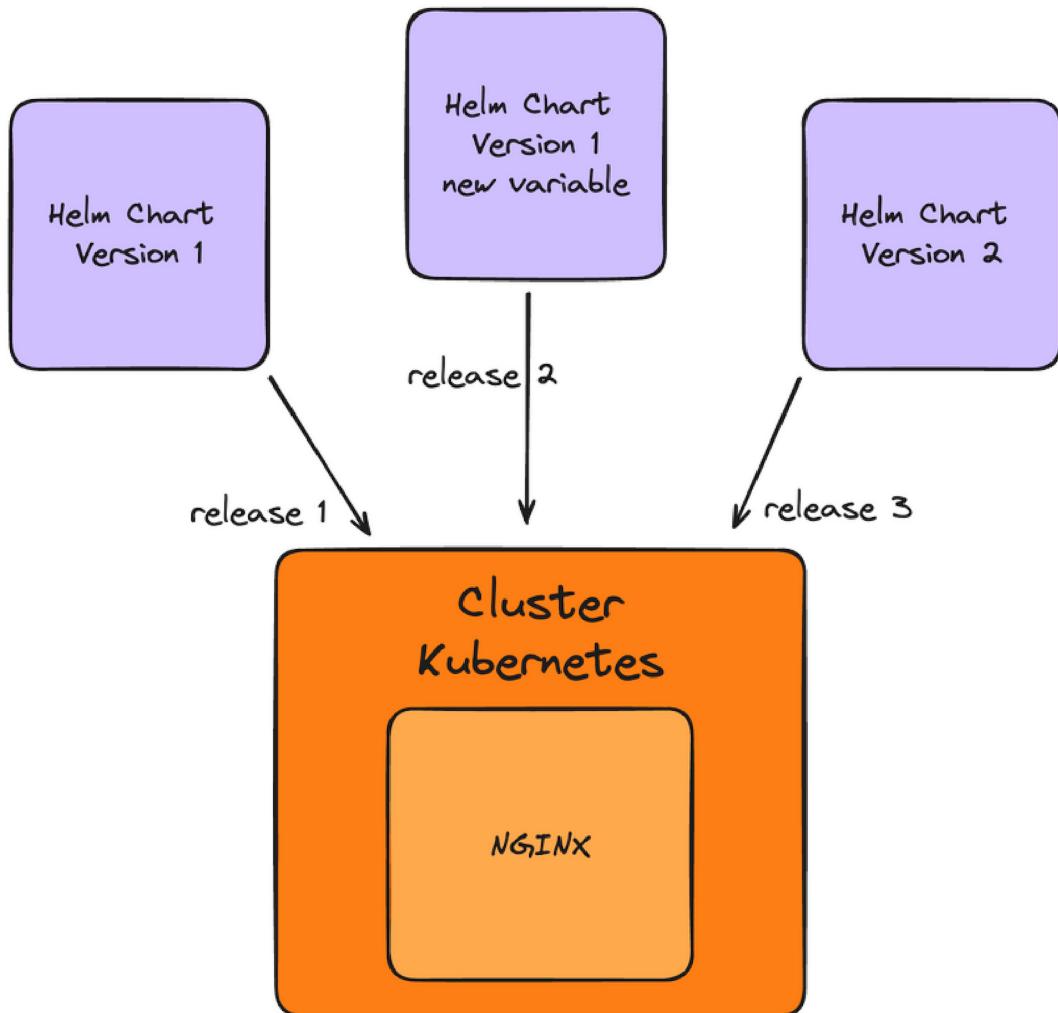
# Installer un chart avec une configuration personnalisée
# helm install [nom_release] [nom_chart] --set [paramètre]=
[valeur]
# Remplacez [nom_release] et [nom_chart] par les noms
appropriés
# Remplacez [paramètre] par le nom de la variable à
configurer et [valeur] par la valeur désirée
helm install my-nginx bitnami/nginx --set
service.type=NodePort

# Lister les releases Helm
# helm list - affiche toutes les releases Helm dans le
cluster
helm list
```

# Releases



Once a chart is deployed, Helm creates a "release". A release is an instance of a chart running in a Kubernetes cluster. Helm allows for efficient management of these releases.



# Gestion des Releases



```
# Mettre à jour une release
# helm upgrade [nom_release] [nom_chart] - met à jour la
release spécifiée
# Vous pouvez également passer des paramètres supplémentaires
avec --set ou en utilisant un fichier de valeurs
helm upgrade my-nginx stable/nginx --set
service.type=LoadBalancer

# Revenir à une version précédente d'une release
# helm rollback [nom_release] [revision] - revient à une
version antérieure de la release
# Remplacez [revision] par le numéro de révision auquel vous
souhaitez revenir
helm rollback my-nginx 1

# Vérifier l'état d'une release
# helm status [nom_release] - affiche l'état actuel de la
release
helm status my-nginx

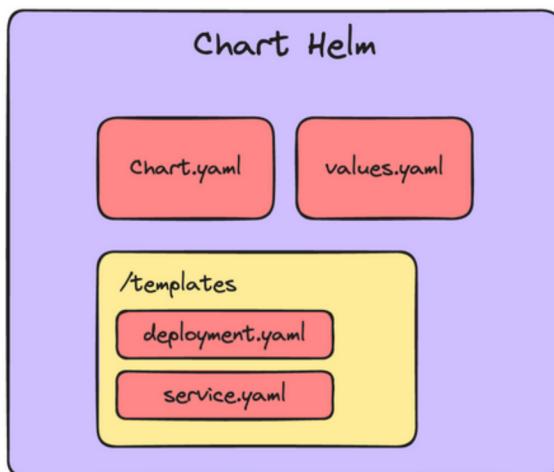
# Supprimer une release
# helm uninstall [nom_release] - supprime la release du
cluster
helm uninstall my-nginx
```

# Création d'un Chart



Un chart Helm est composé de plusieurs fichiers et répertoires. La structure typique d'un chart est la suivante :

- **Chart.yaml:** Le fichier de métadonnées du chart, contenant des informations comme le nom, la version, et une description.
- **values.yaml:** Le fichier de configuration par défaut, où vous définissez les valeurs utilisées par les templates du chart.
- **templates/:** Un répertoire contenant les templates YAML pour les ressources Kubernetes, qui seront personnalisées à l'installation en utilisant du Go Template et les variables du fichier values.
- **charts/:** Un répertoire optionnel contenant les dépendances sous forme de charts.



```
# Créez un nouveau chart Helm
# helm create [nom_chart] - crée une structure de base pour
# un nouveau chart
helm create mychart
```

# Création d'un Chart



```
● ● ●

apiVersion: v2
name: mychart
description: Un exemple simple de chart Helm
version: 0.1.0
```

Chart.yaml

```
● ● ●

replicaCount: 1
image:
  repository: nginx
  tag: "stable"
  pullPolicy: IfNotPresent
service:
  type: ClusterIP
  port: 80
```

values.yaml

# Création d'un Chart



```
● ● ●

apiVersion: apps/v1
kind: Deployment
metadata:
  name: {{ include "mychart.fullname" . }}
  labels:
    {{- include "mychart.labels" . | nindent 4 }}
spec:
  replicas: {{ .Values.replicaCount }}
  selector:
    matchLabels:
      {{- include "mychart.selectorLabels" . | nindent 6 }}
  template:
    metadata:
      labels:
        {{- include "mychart.selectorLabels" . | nindent 8 }}
    spec:
      containers:
        - name: {{ .Chart.Name }}
          image: "{{ .Values.image.repository }}:{{ .Values.image.tag }}"
          imagePullPolicy: {{ .Values.image.pullPolicy }}
          ports:
            - containerPort: {{ .Values.service.port }}
```

deployment.yaml

# Création d'un Chart



```
● ● ●

apiVersion: v1
kind: Service
metadata:
  name: {{ include "mychart.fullname" . }}
  labels:
    {{- include "mychart.labels" . | nindent 4 }}
spec:
  type: {{ .Values.service.type }}
  ports:
    - port: {{ .Values.service.port }}
      targetPort: {{ .Values.service.port }}
  selector:
    {{- include "mychart.selectorLabels" . | nindent 4 }}
```

service.yaml

# Dépendances dans un Chart



Les charts Helm peuvent dépendre d'autres charts, ce qui est utile pour les applications complexes nécessitant plusieurs composants.

Lorsque vous travaillez avec des charts complexes ayant plusieurs dépendances, il est important de vérifier la compatibilité, assurez-vous que les versions des charts dépendants sont compatibles entre elles.

```
● ● ●  
  
dependencies:  
- name: nginx  
  version: "1.16.0"  
  repository: "https://charts.helm.sh/stable"
```

```
● ● ●  
  
# Mettre à jour les dépendances du chart  
# helm dependency update [chemin_vers_chart]  
helm dependency update ./mychart
```

# Dans la même collection

## Orchestration et Gestion de Conteneurs



## Infrastructure as Code



## Sécurité & Gestion des secrets



## Développement & CI/CD



↓ FOLLOW ME ↓



[ANTONYCANUT](#)



[ANTONY KERVAZO-CANUT](#)