

Helm charts

Conteneurisation & Orchestration

Axel Simonet



© 2023

Sommaire

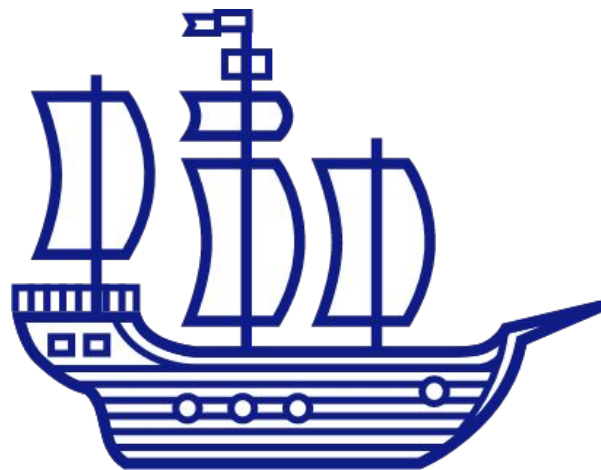


- Helm ?
- Utilité
- Charts
 - TP 1
 - TP 2
- Repo Helm
 - TP 3



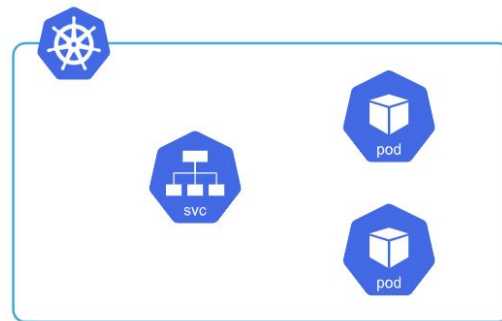
Helm ?

- Package manager
- Maintenu par la CNCF
- Helm charts



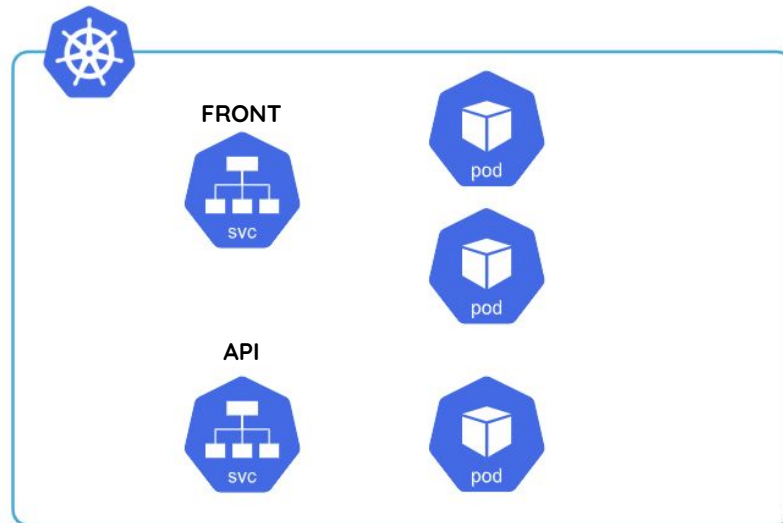
Utilité

- Peu de fichiers = pas utile



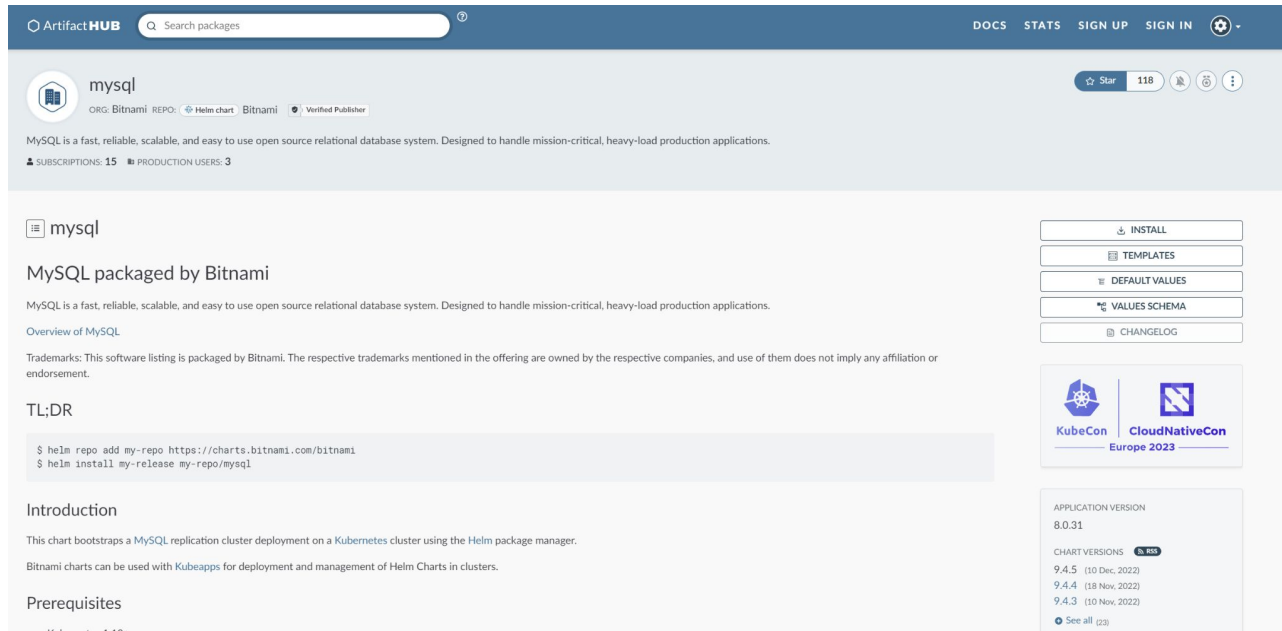
Utilité

- Nombreux fichiers
- Gestion d'environnements
production / staging / dev /
- Update / Rollback



Charts

- Ensemble de fichiers
- Go Template
- Hébergé sur des repos



The screenshot shows the Artifact Hub interface for the Bitnami MySQL Helm chart. The top navigation bar includes 'ArtifactHUB', a search bar, and links for 'DOCS', 'STATS', 'SIGN UP', and 'SIGN IN'. The main header for the 'mysql' chart shows it is from the 'Bitnami' organization, has 118 stars, and is a 'Verified Publisher'. A description states: 'MySQL is a fast, reliable, scalable, and easy to use open source relational database system. Designed to handle mission-critical, heavy-load production applications.' It also shows '15 SUBSCRIPTIONS' and '3 PRODUCTION USERS'.

The main content area is titled 'mysql' and 'MySQL packaged by Bitnami'. It includes a description, an 'Overview of MySQL', and a 'Trademarks' section. Below this is the 'TL;DR' section with the following commands:

```
$ helm repo add my-repo https://charts.bitnami.com/bitnami
$ helm install my-release my-repo/mysql
```

The 'Introduction' section states: 'This chart bootstraps a MySQL replication cluster deployment on a Kubernetes cluster using the Helm package manager.' and 'Bitnami charts can be used with Kubeapps for deployment and management of Helm Charts in clusters.'

The 'Prerequisites' section lists 'Kubernetes 1.19+'. On the right side, there are buttons for 'INSTALL', 'TEMPLATES', 'DEFAULT VALUES', 'VALUES SCHEMA', and 'CHANGELOG'. Below these are logos for 'KubeCon' and 'CloudNativeCon Europe 2023'. At the bottom right, there is a section for 'APPLICATION VERSION' (8.0.31) and 'CHART VERSIONS' (9.4.5, 9.4.4, 9.4.3) with a 'See all (23)' link.

<https://artifacthub.io/packages/helm/bitnami/mysql>



Command line

- `helm repo add <url>`
- `helm install <nom> <repo>/<app>`
- `helm upgrade`
- `helm rollback`
- `helm status`

-f <fichier>.yaml

https://helm.sh/docs/intro/using_helm/



TP 1 - Utilisation de Helm

- Installer Helm <https://helm.sh/docs/intro/install/>
- Déployer un nextcloud en utilisant la chart helm officielle
- Créer un fichier values.yaml pour augmenter le nombre de replicas.
Déployer la mise à jour.



Syntaxe

- Go Template
- `{{ }}`
https://helm.sh/docs/chart_template_guide/getting_started/
- https://helm.sh/docs/chart_template_guide/control_structures/

values.yaml

```
favorite:  
  drink: coffee  
  food: pizza
```



```
apiVersion: v1  
kind: ConfigMap  
metadata:  
  name: {{ .Release.Name }}-configmap  
data:  
  myvalue: "Hello World"  
  drink: {{ .Values.favorite.drink }}  
  food: {{ .Values.favorite.food }}
```

```
{{ if PIPELINE }}  
  # Do something  
{{ else if OTHER PIPELINE }}  
  # Do something else  
{{ else }}  
  # Default case  
{{ end }}
```



Example

Structure

```
mychart/  
  Chart.yaml  
  values.yaml  
  charts/  
  templates/
```

main ▾ examples / charts / hello-world / templates /

adamreese Add "hello-world.chart" function again

..

NOTES.txt	fix
_helpers.tpl	Ad
deployment.yaml	Sir
service.yaml	Le
serviceaccount.yaml	Le

<https://github.com/helm/examples/tree/main/charts/hello-world/templates>

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

```
{{/*  
Selector labels  
*/}}  
{{- define "hello-world.selectorLabels" -}}  
app.kubernetes.io/name: {{ include "hello-world.name" . }}  
app.kubernetes.io/instance: {{ .Release.Name }}  
{{- end }}
```



TP 2 - Créer sa première chart

- La chart devra déployer un serveur nginx.
- Les déploiements, services, ingress, ... seront donc créés par cette chart.
L'image, tag, ports, type de service doivent être modifiables à l'aide d'un fichier values.yaml.
- **2 fichiers** values-(prod / dev).yaml devront être créés.
Le fichier prod déploiera un serveur nginx en version **1.22** avec comme domaine **srv-prod.test** dans le namespace **production**.
Le fichier dev déploiera un serveur nginx en version **1.23** avec comme domaine **srv-dev.test** dans le namespace **development**.

Il est interdit de réutiliser une chart existante.

