K8S 49: Kubernetes. Helm. Шаблонизатор

Описание:

Как уже было сказано в предыдущем задании, Helm - это пакетный менеджер для чартов (Charts). Но что же такое чарты?
В простейшем описании чарт представляет из себя структуру файлов, которые описывают требуемое состояние ресурсов. Эти файлы можно условно разделить на обязательные и необязательные. Давайте посмотрим подробнее:

```
/...
/templates/                - можно сказать, сердце чарта, директория, которая содержит шаблоны манифестов. ресурсов Kubernetes
/templates/NOTES.txt       - необязательный файл с примечанием, которое будет выводиться пользователю при работе с чартом.
/charts/                   - каталог с вложенными чартами (необязателен).
/Chart.yaml                - файл с общей информацией о чарте.
/LICENSE                   - файл с лицензией чарта (необязателен).
/README.md                 - файл с описанием/документацией чарта (необязателен).
/requirements.yaml         - файл со списком зависимостей (иных чартов, требуемых для работы текущего, необязателен).
/values.yaml               - файл со значениями переменных для шаблонов.
```

В целом, в нашем случае мы можем отбросить все необязательные элементы для начала работы с Helm. При создании чарта после описания ряда информационных файлов вашей задачей, по сути, будет верное оформление темплейтов (шаблонов манифестов). В качестве шаблонизатора Helm использует дополненный Golang-шаблонизатор.
{{ }}
Фигурные скобки являются указателем на то, что внутри содержится шаблонизированное значение. Все, что не ставится в фигурные скобки, при рендере чарта остается неизменным
.
Точка — это переменная контекста, можно сказать, что это указатель для подстановки значений, содержащихся в структуре данных чарта.
Даже в общих чертах ознакомление со структурой шаблонизатора — достаточно обширная тема, и в рамках саммари проще рассмотреть практические примеры:

```
{{- if .Capabilities.APIVersions.Has "apps/v1beta2" }}
```

```
apiVersion: apps/v1beta2
{{- else }}
apiVersion: extensions/v1beta1
{{- end }}
metadata:
  labels:
    chart: "{{ .Chart.Name }}"
    release: "{{ .Release.Name }}"
containers:
- name: nginx
  image: "nginx:{{ .Values.ImageTag }}"
...
  data:
    nginx.conf: |
{{ .Files.Get "configs/nginx.conf" | indent 6 }}
```

Итак, .Chart — это структура данных, содержит информацию о чарте и описывается в файле Chart.yaml. Release содержит данные о релизе, инсталляции и обновлении, такие как имя релиза, неймспейс, в который производится выкладка и т.д. По сути, это мета-информация о нашем релизе. И наконец, .Values — это параметры, определенные в values.yaml и определенные опциями --set — именно тут параметризуется «пользовательская» информация. Capabilites — это доступная информация о кластере, в который производится выкатка приложения. Files — структура, содержащая информацию о файлах, которые хранятся в директории чарта.

Полезные ссылки:

- Helm Docs

Задание:

1. Напишите структуру чарта для деплоя приложения gocalc.
    - pod должен состоять из двух контейнеров — с приложением gocalc и nginx, который будет обрабатывать входящие запросы и отправлять их на приложение.
    - chart должен содержать deployment, service, ingress, configMap с конфигом nginx. Поле image в контейнере с приложением должно быть шаблонизировано, для того чтобы при деплое мы могли задавать конкретную версию кода.
2. Разместите ваш чарт в git-репозитории и предоставьте ссылку в качестве ответа.