Published in FAUN Publication

Md Shamim Follow
Nov 4, 2022 · 6 min read · Listen

Save

Helm —Template Actions, Functions, and Pipelines

Overview of helm template actions, functions, and pipelines

Photo by Daniel Bernard on Unsplash

This article will cover template actions, functions, and pipelines; that will help us get started with writing custom helm templates.

Before starting the main part, Let's create a helm chart named **webserver.** That will help us when we move on to the further parts of this article.

```
$ helm create webserver
```

After creating the chart (by default — nginx chart). To get a shallow idea about helm templates, let's observe a template resides under the templates folder.

```
$ cat ~/webserver/templates/service.yaml
```

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


```

service-template.yaml hosted with ❤ by **GitHub** **view raw**

Helm actions {{ }}

In the **helm** template files, we often see this `{{- something }}` syntax, it's called template actions. Within `{{ }}` template actions, we can use several other elements from the helm templating syntax, such as defining variables, conditional logic (if/else), loops, functions, and so on.

In the template files, anything outside of the action elements will be printed as it is in the output. And the elements inside the `{{ }}` actions will help us to retrieve data from other sources.

Handling whitespaces :

To avoid leading or trailing whitespace, we can use a hyphen inside helm actions. If a hyphen is added before the statement, `{{- something }}` then the leading whitespace will be ignored. And similarly, if a hyphen is added after the statement `{{something -}}` , then trailing whitespace will be ignored. And obviously, this is `{{- something -}}` also allowed to eliminate leading and trailing whitespaces.

Dynamic Templating

For creating a dynamic template, we can inject data into the template files from several built-in/custom objects. Some of the frequently used objects are **Release, Values, Chart, and Template.** One object can contain other objects or functions.

For instance — `{{- .Release.Name -}}`

Here, **(.)** dot is the root object. Other objects reside underneath the root object.

`Release` **:** This object describes the release itself. It has several objects inside of it:

```
{{ .Release.Name }}
{{ .Release.Service }}
{{ .Release.Namespace }}
{{ .Release.Time }}
{{ .Release.IsInstall }}
{{ .Release.IsUpgrade }}
```

`Values` **:** Values passed into the template from the `values.yaml` file and from user-supplied files.

`Chart` **:** The contents of the `Chart.yaml` file can be passed into the template files.

```
{{ .Chart.Name }}
{{ .Chart.Version }}
{{ .Chart.AppVersion }}
{{ .Chart.Annotations }}
```

`Template` **:** Contains information about the current template that is being executed.

```
{{ .Template.Name }}
{{ .Template.BasePath }}
```

Create a template

In this phase, we will create a new template by leveraging what we have learned so far.

Currently, our chart directory looks like this:

```
>> tree webserver/

webserver/
├── charts
├── Chart.yaml
├── templates
│   ├── deployment.yaml
│   ├── _helpers.tpl
│   ├── hpa.yaml
│   ├── ingress.yaml
│   ├── NOTES.txt
│   ├── serviceaccount.yaml
│   ├── service.yaml
│   └── tests
│       └── test-connection.yaml
└── values.yaml
```

Let’s start from the very beginning; Delete all the **template files** and **values.yaml** file from the chart directory.

```
>>  rm -rf webserver/templates/*
>>  rm -rf webserver/values.yaml

>>  tree webserver/

 webserver/
 ├── charts
 ├── Chart.yaml
 └── templates
```

Now, we will create a configmap template under the **templates** directory. And then, we will inject some data from the **release** object and **values** object into the configmap template. For that, we have to create new **values.yaml** file. After creating **values.yaml** file, we will populate it with some entries.

Create **values.yaml** file and populate it with the following entries :

```
#values.yaml

configMap:
  data:
    app_state: stateless
    app_mode: dev

```

values.yaml hosted with ❤ by **GitHub** **view raw**

Pass data from **values.yaml** to a configmap template that resides under the templates directory :

```
#configmap.yaml

apiVersion: v1
kind: ConfigMap
metadata:
  name: {{ .Release.Name }}-configmap
data:
  state: {{ .Values.configMap.data.app_state }}
  mode: {{ .Values.configMap.data.app_mode }}


```

configmap.yaml hosted with ❤ by **GitHub** **view raw**

Finally, verify if the template can render data without any issues :

```
>> helm template ~/webserver

---
# Source: webserver/templates/configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: release-name-configmap
data:
  state: stateless
  mode: dev
```

We have created a configmap template, which will retrieve data from the **Release** object and **Values** object.

Functions and Pipelines

In the template file, inside the template actions, we can use various template functions. There is a huge list of functions available for helm templating, view it from here.

In this phase, we will see some of the template functions and also **pipelines**, which will help us to chain multiple template commands, expressions, or function calls.

toYaml :

We can use the **toYaml** function inside the helm actions to convert an object into YAML.

Modify the configmap template file created earlier.

```
#configmap.yaml

apiVersion: v1
kind: ConfigMap
metadata:
  name: {{ .Release.Name }}-configmap
data:
{{- toYaml .Values.configMap.data }}

```

configmap.yaml hosted with ❤ by **GitHub** **view raw**

Unlike the earlier created configmap template.In the above demonstration, we tried to fetch the entire object at once rather than fetching data one by one.

Now, generate the template using the **helm template** command:

```
>> helm template ~/webserver

---
# Source: webserver/templates/configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: release-name-configmap
```

```
data:app_mode: dev
app_state: stateless
```

Notice that there is an issue with the indentation. In a YAML file indentation is a very important thing. To resolve the indentation issue we can use another function called **indent** and **nindent.** To chain the two functions we will use the pipelines.

indent :

`indent` indents a string by the specified length.

Example :

```
...
data:
  state: {{ .Values.configMap.data.app_state }}
  mode: {{ .Values.configMap.data.app_mode | indent 4 }}

---

data:
  state: stateless
  mode:     dev
```

nindent

`nindent` indents a string by the specified length. But a new line appears before indentation.

```
...
data:
  state: {{ .Values.configMap.data.app_state }}
  mode: {{ .Values.configMap.data.app_mode | nindent 4 }}

---

data:
  state: stateless
  mode:
    dev
```

Now, it's time to make the indentation right for the configmap template file:

```
#configmap.yaml

apiVersion: v1
kind: ConfigMap
metadata:
  name: {{ .Release.Name }}-configmap
data:
  {{- toYaml .Values.configMap.data | nindent 2 }}

```

configmap.yaml hosted with ❤ by **GitHub** **view raw**

```
>> helm template ~/webserver
---
# Source: webserver/templates/configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: release-name-configmap
data:
  app_mode: dev
  app_state: stateless
```

upper

`upper` converts the string to uppercase.

```
...
data:
  {{- toYaml .Values.configMap.data | nindent 2 | upper }}

---

data:
  APP_MODE: DEV
  APP_STATE: STATELESS
```

Similarly, to convert the string to lowercase we can use `lower` function.

quote

`quote` enclose the string in double quotes (” ”).

```
data:
  state: {{ .Values.configMap.data.app_state }}
```

```
    mode: {{ .Values.configMap.data.app_mode | quote }}

  ---

  data:
    state: stateless
    mode: "dev"
```

To enclose the string in single quotes ('), we can use `squote` function.

trunc

`trunc` truncates a string to the specified length.

```
  ...
  data:
    state: {{ .Values.configMap.data.app_state | trunc 3 }}
    mode: {{ .Values.configMap.data.app_mode }}

  ---

  data:
    state: sta
    mode: dev
```

default

If we refer to an undefined value in the helm template, then during the installation there might be some unexpected errors or issues. To avoid any incidental errors or issues we can set a default value by using the **default** function.

Suppose, the **values.yaml** file contains the following entries:

```yaml
#values.yaml

configMap:
  data:
    app_state: stateless
    app_mode: dev

```

values.yaml hosted with ❤ by **GitHub** **view raw**

But In our template file, we have referred to an undefined value. This might cause errors during the installation. To avoid that we can use the **default** function in the template file.

```
...
data:
  state: {{ .Values.configMap.data.app_state | default "stateful" }}
  mode: {{ .Values.configMap.data.app_mode | default "test" }}
  region: {{ .Values.configMap.data.region | default "tokyo" }}

---

data:
  state: stateless
  mode: dev
  region: tokyo
```

Next —

Helm — Flow Control

Helm "flow control" using if/else, with and range

levelup.gitconnected.com

> If you found this article helpful, please **don't forget** to hit the **Clap** and **Follow** buttons to help me write more articles like this.
> Thank You 🖤

All Articles on Helm Chart -

Helm — In Action

Helm — Advanced Commands

Helm — Create Your First Chart

Helm — Template Actions, Functions, and Pipelines

Helm — Flow Control
Helm — Variables
Helm — Named Template
Helm — Dependencies

References

Template Functions and Pipelines

So far, we've seen how to place information into a template. But that information is placed into the template...

helm.sh

If you find this helpful, please click the clap 👏 button below a few times to show your support for the author 👇

🚀**Join FAUN & get similar stories in your inbox each week**

Helm Helm Chart Kubernetes Kubernetes Cluster Package Manager

208 | 1 | •••

Enjoy the read? Reward the writer. Beta

Your tip will go to Md Shamim through a third-party platform of their choice, letting them know you appreciate their story.