Templating Tools and Package Management

Templating Tools. Getting Started with Helm

Creating Simple Charts

SoftUni Team

Technical Trainers

Software University

SoftUni

Software University

https://softuni.bg

Have a Question?

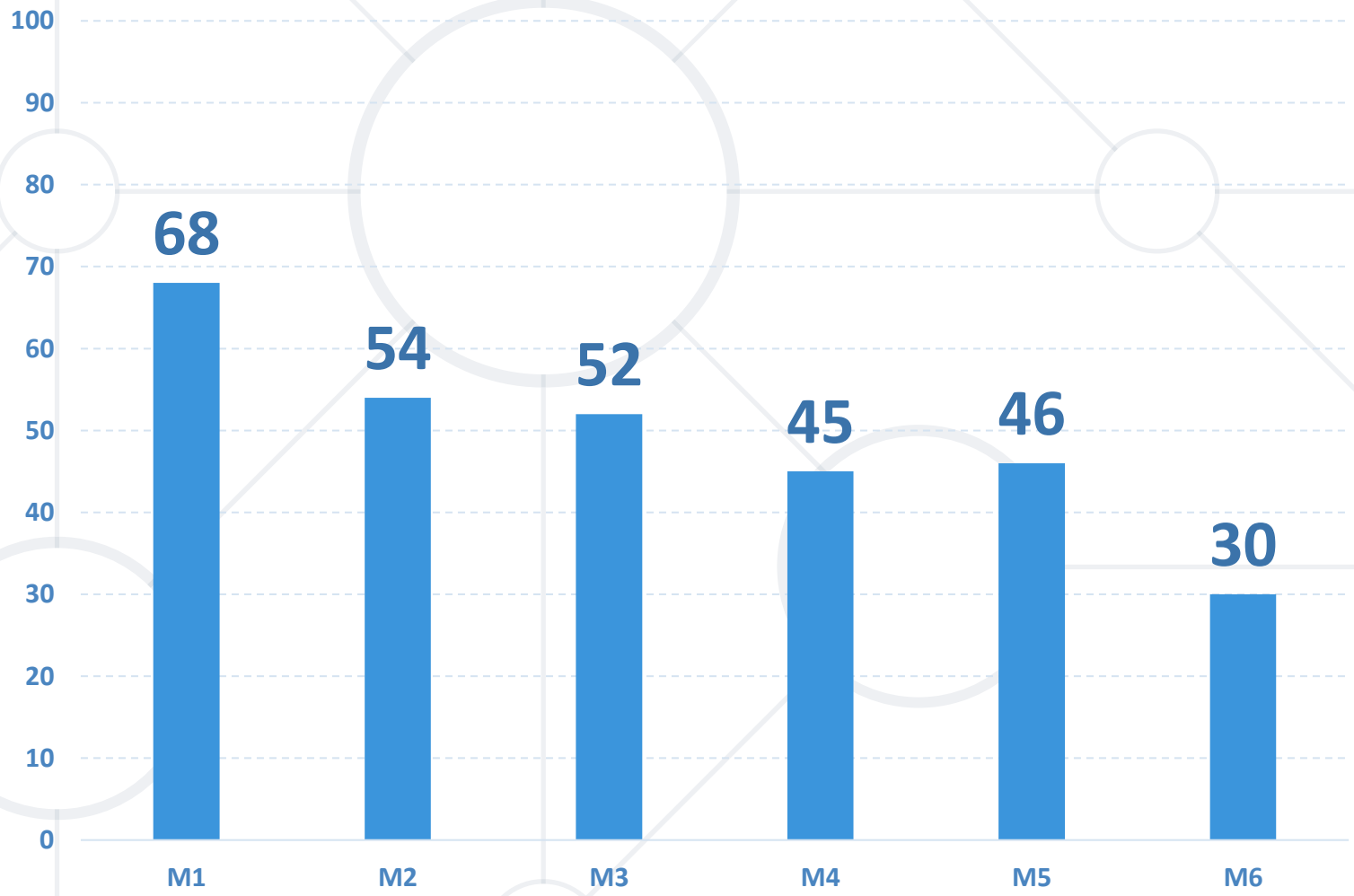
Software University

sli.do

#Kubernetes

facebook.com
/groups/kubernetesnovember2025

Homework Progress



Submit M6
until 23:59:59
on 16.12.2025

Submit M7
until 23:59:59
on 23.12.2025

The End is Near ☺

Software University

THIS MODULE

AND ONE MORE TO GO

Scoring

For example, if you score

Practice: **56**

Test: **37**

Total: **93**

Homework: **10**

New total: **103**

Final: **100**

Practice (Remote)

- Manage and troubleshoot a set of clusters
- Deploy objects either standalone or as part of applications
- Troubleshoot applications as a whole or their components
- Additional tasks as per the exam requirements

The **practice exam** will be held **remotely** in a **controlled environment**

All you need is just a PC with **SSH client** and **Internet connectivity**

You will have **4 hours** to complete the tasks

Test (Remote)

40

minutes

40

single-choice questions

40

points in total

Test Your Knowledge

Software University

Practice (exam-like) questions:

https://zahariev.pro/q/k8s

Previous Module (M6)

Quick overview

Table of Contents

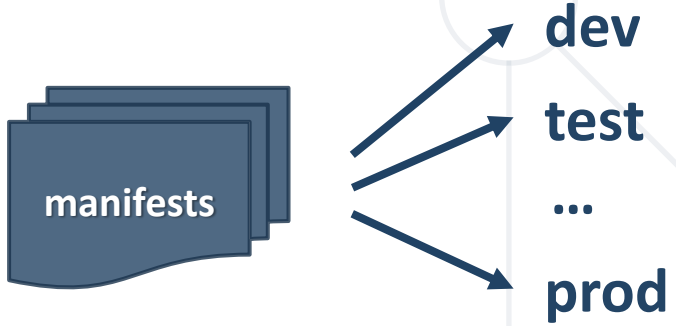
This Module (M7)

Table of Contents

Templating Tools

What and Why?

- In the typical situation, we want to be able to deploy an application to multiple environments or in multiple modes
- These can be development, test, production, etc.
- They may differ in terms of image name, image tag, service type, service port, etc.
- We can either have multiple manifest sets
- Or use some kind of tooling



Manual Approach

- Perhaps this is the easiest, especially when dealing with either simpler or fewer manifests
- Pros: familiar and easy to use tool – **sed**
- Cons: manifests should be **specially prepared** with **placeholders**, and it is difficult to set none or some of them – always we should provide values for all

manifest.yaml (original)

```
apiVersion: v1
kind: Pod
metadata:
  name: pod
spec:
  containers:
  - image: nginx:latest
    name: main
```

manifest.yaml (template)

```
apiVersion: v1
kind: Pod
metadata:
  name: pod
spec:
  containers:
  - image: %image%:%tag%
    name: main
```

**sed -e 's/%image%/alpine' **

-e 's/%tag%/3.14' manifest.yaml

manifest.yaml (result)

```
apiVersion: v1
kind: Pod
metadata:
  name: pod
spec:
  containers:
  - image: alpine:3.14
    name: main
```

https://www.gnu.org/software/sed/manual/sed.html

Kustomize

- Native Kubernetes configuration management
- Template-free way to customize application configuration
- Simplifies the use of off-the-shelf applications
- Built into **kubectl** as **apply -k**

kustomization.yaml

```yaml
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization
namePrefix: dev-
commonLabels:
  variant: dev
patches:
- path: patch.yaml
resources:
- ../../base
```

base/main.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: appa
spec:
  replicas: 1
  selector:
    matchLabels:
      app: appa
  template:
    metadata:
      labels:
        app: appa
    spec:
      containers:
      - name: main
```

overlays/dev/patch.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: appa
spec:
  replicas: 5
```

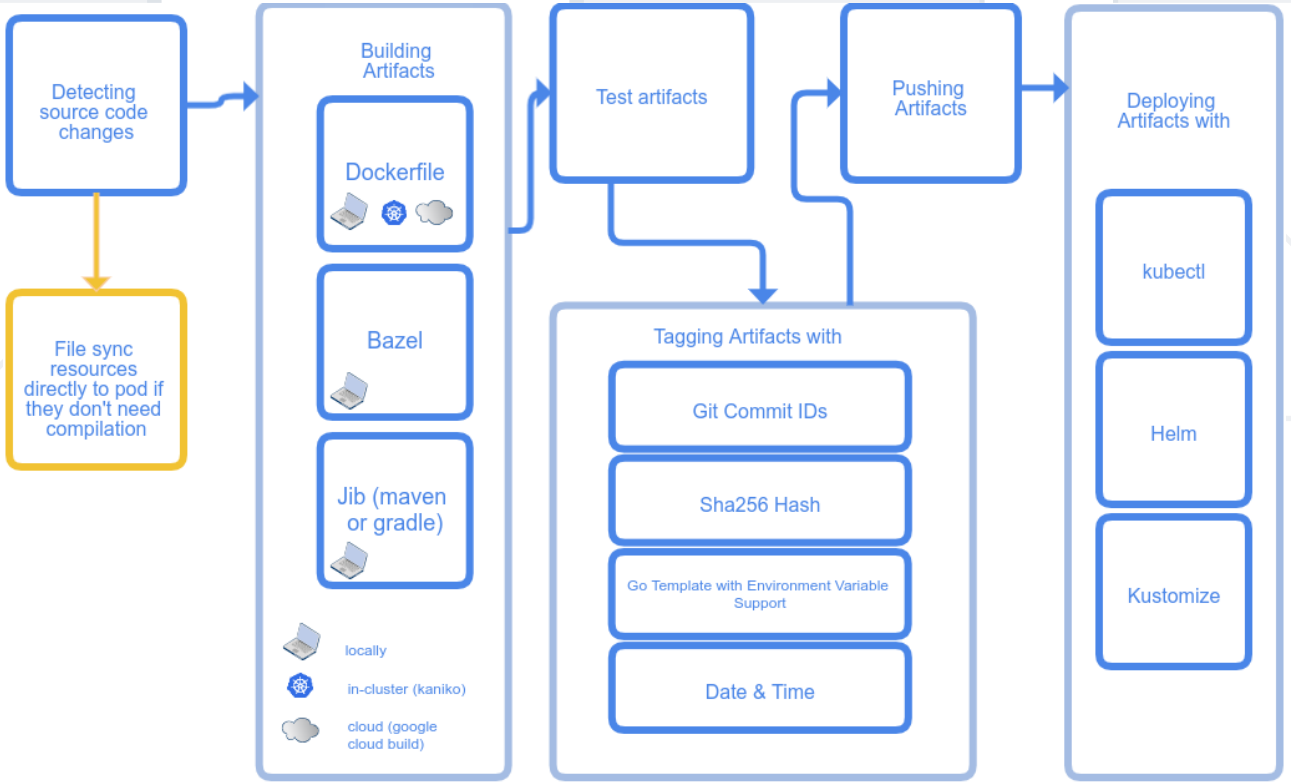
Resulting manifest

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    variant: dev
  name: dev-appa
spec:
  replicas: 5
  selector:
    matchLabels:
      app: appa
      variant: dev
  template:
    metadata:
      labels:
        app: appa
        variant: dev
    spec:
      containers:
      - name: main
```

https://kustomize.io/

Skaffold

- Focused more on the pipeline
- Handles the workflow for building, pushing and deploying your application



https://skaffold.dev/

Other (Templating and Workflow) Tools *

- **Kubes**
  - https://kubes.guru/
- **YQ**
  - https://mikefarah.gitbook.io/yq/
- **YTT**
  - https://carvel.dev/ytt/
- **Write Your Own** (using client libraries)
  - https://kubernetes.io/docs/reference/using-api/client-libraries/
- **Helm**
  - https://helm.sh/

Practice

Live Exercise in Class (Lab)

Getting Started with Helm

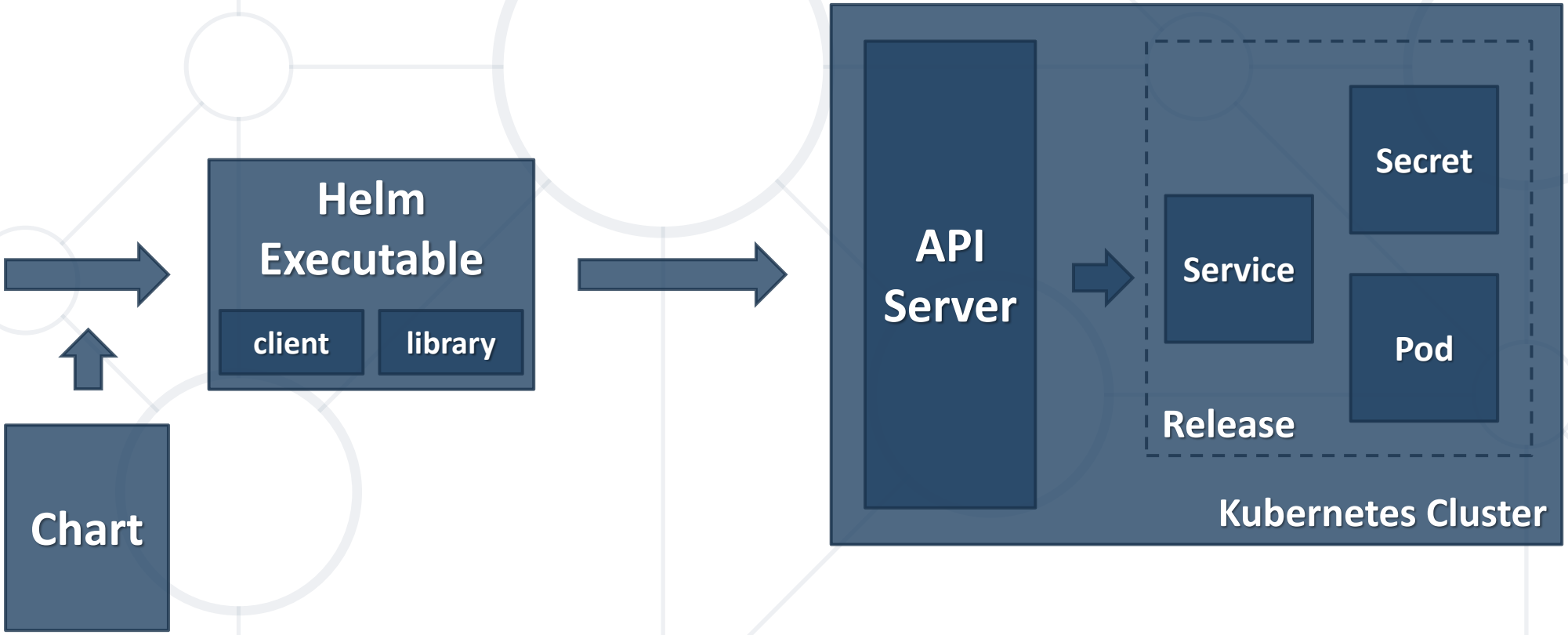
What is Helm?

- It is a **package manager** for Kubernetes
- Its packages are called **charts**
- Charts help us define, install, and upgrade complex applications
- They are easy to create, version, share, and publish
- Charts are **organized in repositories**
- Repositories may be accessed either **directly** or via a **hub**
- One such hub is the **ArtifactHUB** (*https://artifacthub.io/*)

https://helm.sh/

Architecture

- Two parts in one executable – client and library



https://helm.sh/docs/topics/architecture/

Charts

- Collection of files that describe a related set of Kubernetes resources
- A single chart might be used to deploy something simple like a single pod with nginx, redis, etc.
- Or a set of different resources. For example, a full web app stack with HTTP servers, databases, caches, and so on
- They are created as a set of files with particular names and structure and then packaged into versioned archives

https://helm.sh/docs/topics/charts/

Charts

- Chart is organized as a collection of files inside of a directory
- The directory name is the name of the chart without versioning information

```
wordpress/
  Chart.yaml          # A YAML file containing information about the chart
  LICENSE             # OPTIONAL: A plain text file containing the license for the chart
  README.md           # OPTIONAL: A human-readable README file
  values.yaml         # The default configuration values for this chart
  values.schema.json  # OPTIONAL: A JSON Schema for imposing a structure on the values.yaml file
  charts/             # A directory containing any charts upon which this chart depends.
  crds/               # Custom Resource Definitions
  templates/          # A directory of templates that, when combined with values,
                      # will generate valid Kubernetes manifest files.
  templates/NOTES.txt # OPTIONAL: A plain text file containing short usage notes
```

https://helm.sh/docs/topics/charts/

Practice

Live Exercise in Class (Lab)

Creating Simple Charts

Chart Creation

- We can start either from a skeleton chart or from an existing manifest (or set of manifests)
- In any case, we should have at least the following
  - Folder named after our chart to store its files
  - **Chart.yaml** file to describe it
  - **values.yaml** file to contain any default values
  - **templates/** sub-directory for the actual chart template files

https://helm.sh/docs/chart_template_guide/getting_started/

Practice

Live Exercise in Class (Lab)

Questions?

SoftUni

Software University | SoftUni AI | SoftUni Digital | SoftUni Creative | SoftUni Foundation | СофтУни БУДИТЕЛ | Finance Academy

SoftUni Diamond Partners

Software University

Coca-Cola HBC Bulgaria

Postbank

top EMPLOYER България Bulgaria 2025

FOR A BETTER WORLD OF WORK

SUPER HOSTING .BG

encorp.ai

createX

DRAFT KINGS

THE CROWN IS YOURS

INDEAVR

Serving the high achievers

VIVACOM

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, about.softuni.bg
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://about.softuni.bg/
- © Software University – https://softuni.bg