



KubeCon



CloudNativeCon

North America 2025

Introducing Helm 4

#KubeCon #CloudNativeCon



Releasing Helm 4



KubeCon



CloudNativeCon

North America 2025

Who we are



KubeCon



CloudNativeCon

North America 2025

Matt Farina

- Open Source Contributor and Maintainer
- Distinguished Engineer
- Committed Vegan



Robert Sirchia



- Open Source Contributor and Maintainer
- Director Technical & Community Marketing
- Avid Hunter

Some History



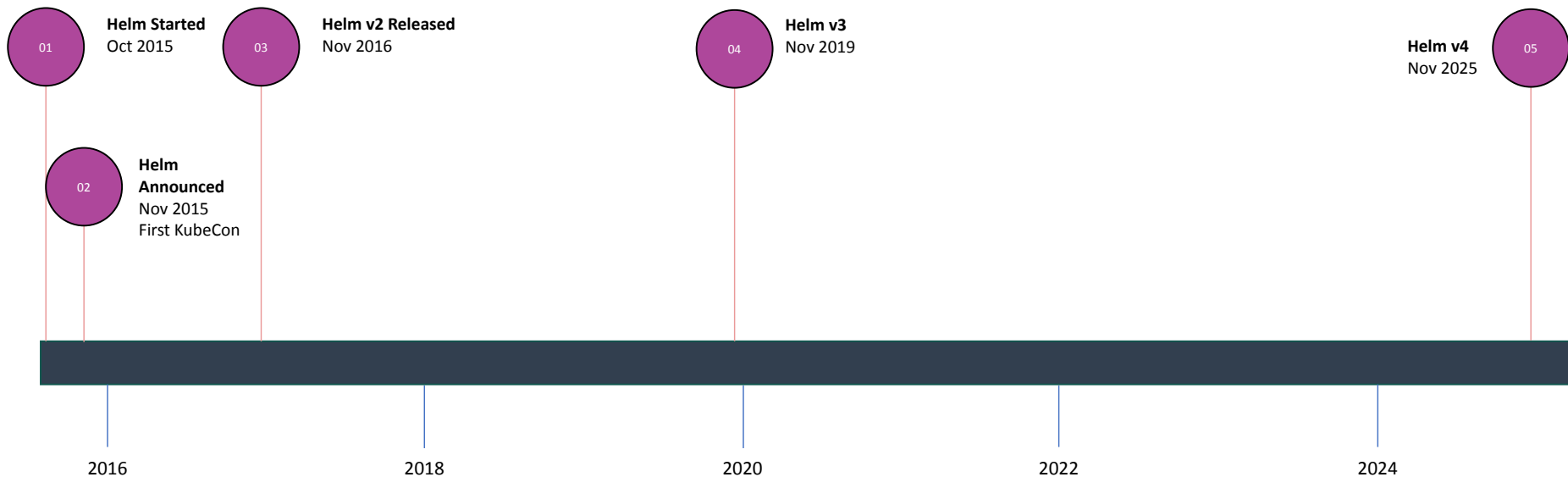
KubeCon

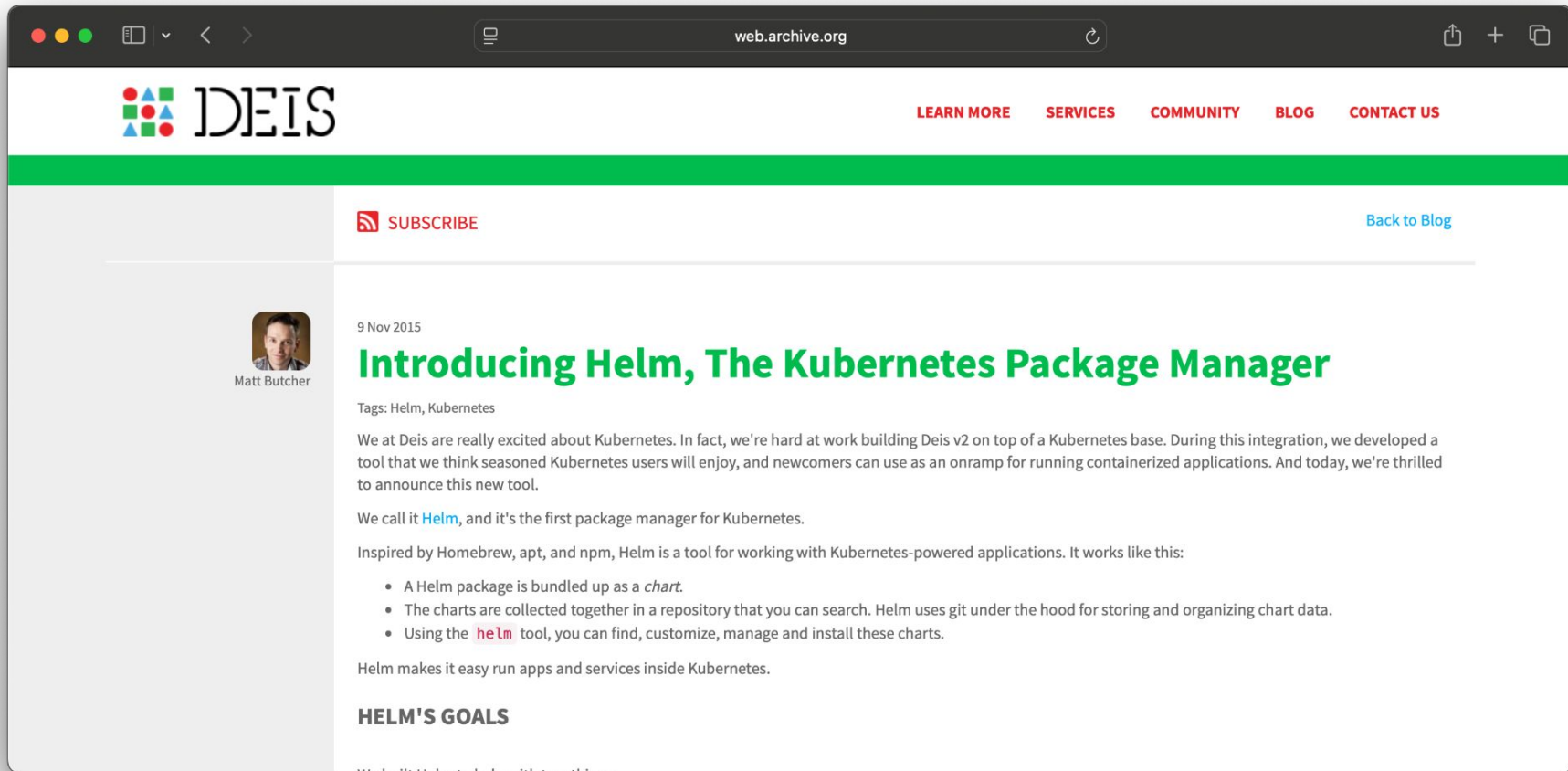


CloudNativeCon

North America 2025

Helm At 10 Years Old





Why Helm 4?



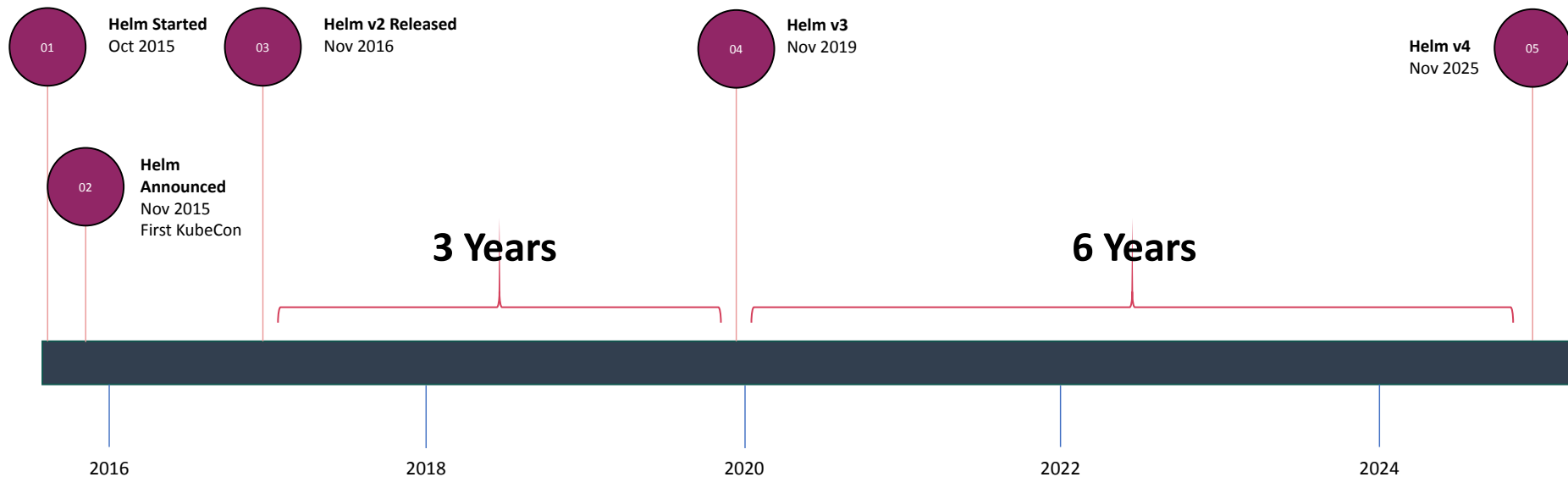
KubeCon



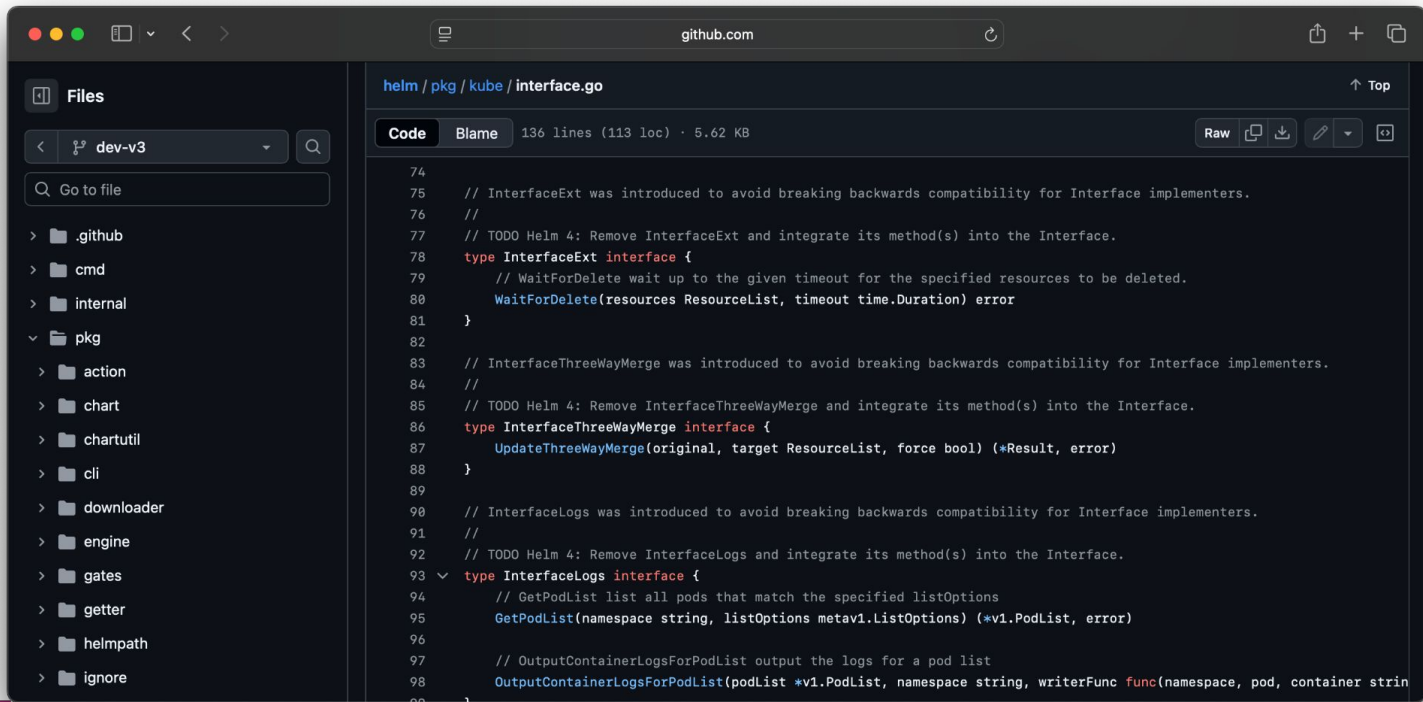
CloudNativeCon

North America 2025

Helm At 10 Years Old

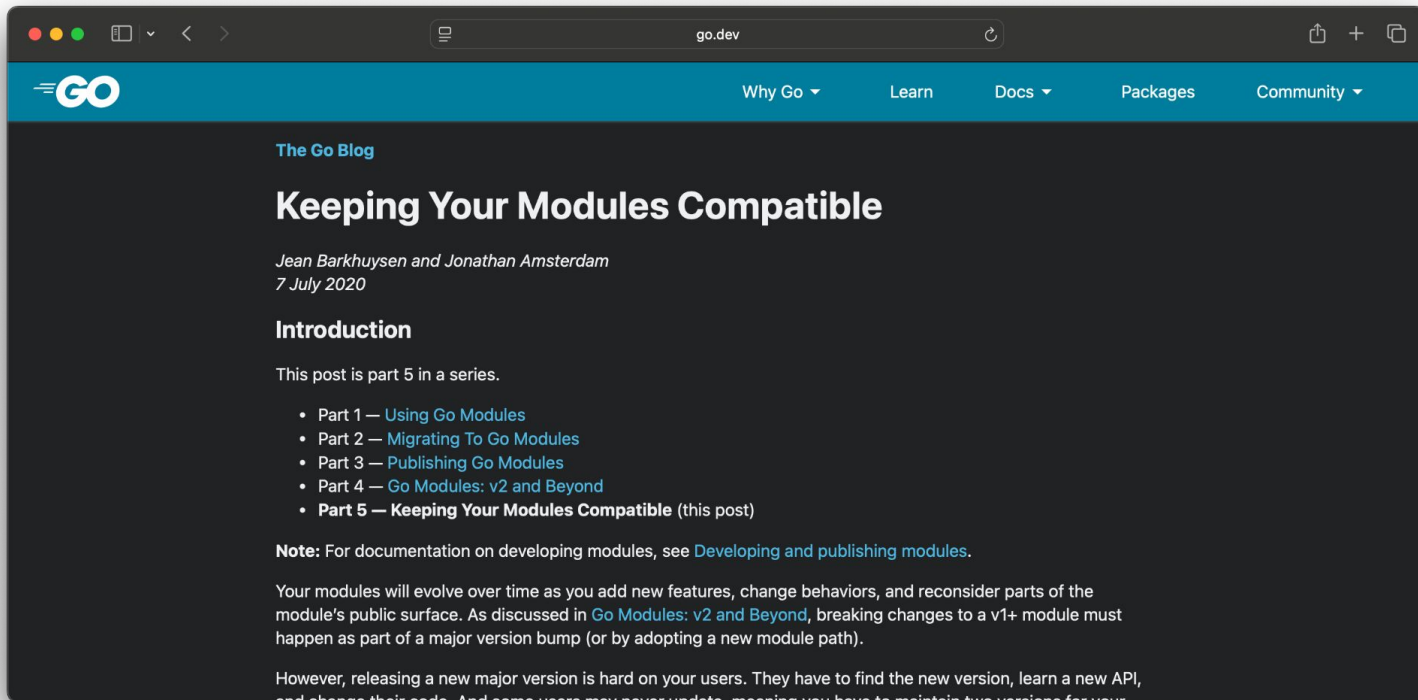


Ugly & Separated Interfaces



```
74
75 // InterfaceExt was introduced to avoid breaking backwards compatibility for Interface implementers.
76 //
77 // TODO Helm 4: Remove InterfaceExt and integrate its method(s) into the Interface.
78 type InterfaceExt interface {
79     // WaitForDelete wait up to the given timeout for the specified resources to be deleted.
80     WaitForDelete(resources.ResourceList, timeout time.Duration) error
81 }
82
83 // InterfaceThreeWayMerge was introduced to avoid breaking backwards compatibility for Interface implementers.
84 //
85 // TODO Helm 4: Remove InterfaceThreeWayMerge and integrate its method(s) into the Interface.
86 type InterfaceThreeWayMerge interface {
87     UpdateThreeWayMerge(original, target ResourceList, force bool) (*Result, error)
88 }
89
90 // InterfaceLogs was introduced to avoid breaking backwards compatibility for Interface implementers.
91 //
92 // TODO Helm 4: Remove InterfaceLogs and integrate its method(s) into the Interface.
93 type InterfaceLogs interface {
94     // GetPodList list all pods that match the specified listOptions
95     GetPodList(namespace string, listOptions metav1.ListOptions) (*v1.PodList, error)
96
97     // OutputContainerLogsForPodList output the logs for a pod list
98     OutputContainerLogsForPodList(podList *v1.PodList, namespace string, writerFunc func(namespace, pod, container string) error)
```

Following Go Language Guidance



Helm Support *OLD* Kubernetes

removing old apis #30585

Merged robertsirc merged 1 commit into helm:main from robertsirc:remove-deprecated-apis 3 weeks ago

Conversation 1 Commits 1 Checks 5 Files changed 2 +8 -13

Changes from all commits File filter Conversations

Filter changed files

pkg/kube

- ready.go
- ready_test.go

```
@@ -21,11 +21,8 @@ import (  
    "fmt"  
    appsv1 "k8s.io/api/apps/v1"  
    appsv1beta1 "k8s.io/api/apps/v1beta1"  
    appsv1beta2 "k8s.io/api/apps/v1beta2"  
    batchv1 "k8s.io/api/batch/v1"  
    corev1 "k8s.io/api/core/v1"  
    extensionsv1beta1 "k8s.io/api/extensions/v1beta1"  
    apiextv1 "k8s.io/apiextensions-apiserver/pkg/apis/apiextensions/v1"  
    apiextv1beta1 "k8s.io/apiextensions-apiserver/pkg/apis/apiextensions/v1beta1"  
    metav1 "k8s.io/apimachinery/pkg/apis/meta/v1"
```

Logging

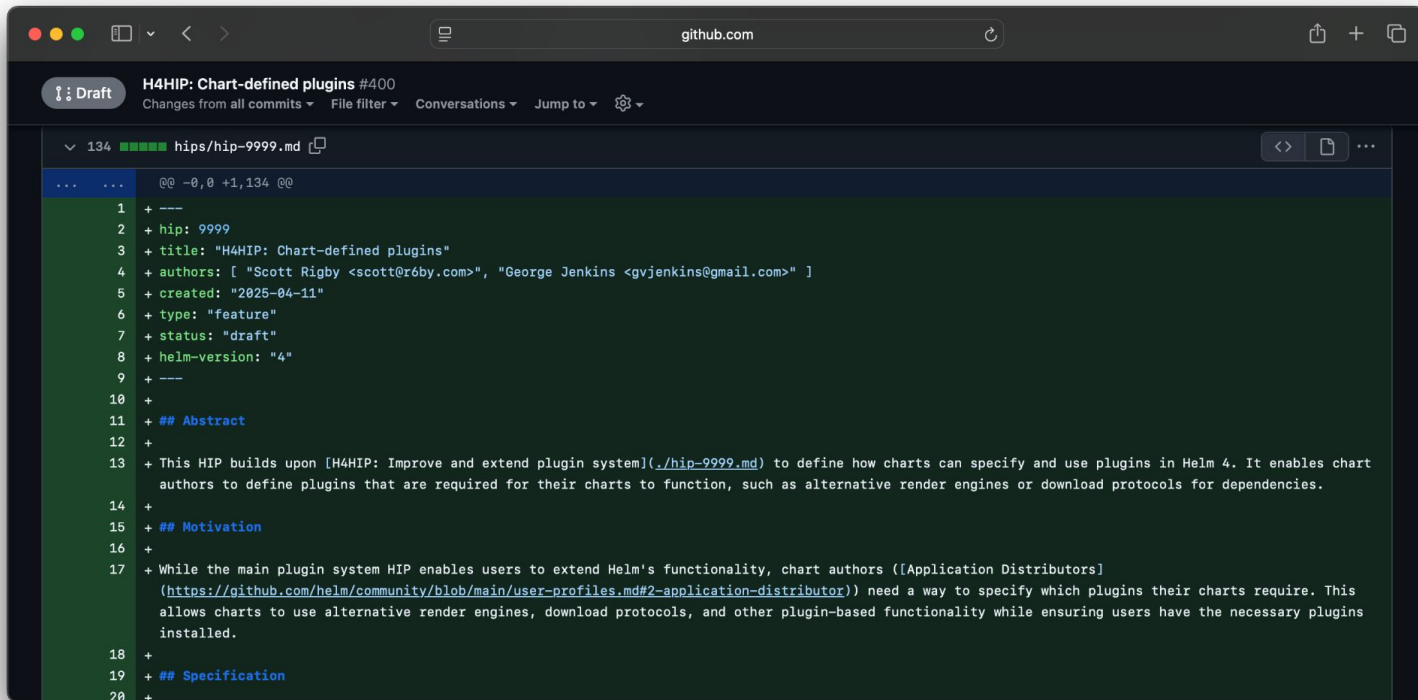
Helm v3 has simple logging that predated the popularity of logrus and other loggers.

```
func Debug(format string, v ...interface{}) {
    if settings.Debug {
        log.Output(2, fmt.Sprintf("[debug] "+format+"\n", v...))
    }
}

func Warning(format string, v ...interface{}) {
    fmt.Fprintf(os.Stderr, "WARNING: "+format+"\n", v...)
}
```

```
if errOutputting := cfg.outputLogsByPolicy(h, rl.Namespace, release.HookOutputOnFailed); errOutputting != nil {
    // We log the error here as we want to propagate the hook failure upwards to the release object.
    log.Printf("error outputting logs for hook failure: %v", errOutputting)
}
// If a hook is failed, check the annotation of the hook to determine whether the hook should be deleted
// under failed condition. If so, then clear the corresponding resource object in the hook
if errDeleting := cfg.deleteHookByPolicy(h, release.HookFailed, timeout); errDeleting != nil {
    // We log the error here as we want to propagate the hook failure upwards to the release object.
    log.Printf("error deleting the hook resource on hook failure: %v", errDeleting)
}
```

Hard To Implement Chart Features



The screenshot shows a GitHub pull request interface. At the top, the title is "H4HIP: Chart-defined plugins #400". Below the title, there are tabs for "Changes from all commits", "File filter", "Conversations", and "Jump to". The main content area shows a diff for the file "hips/hip-9999.md". The diff includes a header section with metadata like title, authors, created date, type, status, and helm-version. It also includes an abstract, motivation, and specification section. The abstract states that this HIP builds upon [H4HIP: Improve and extend plugin system] to define how charts can specify and use plugins in Helm 4. The motivation section explains that the main plugin system HIP enables users to extend Helm's functionality, and chart authors need a way to specify which plugins their charts require. The specification section is partially visible.

```
... @@ -0,0 +1,134 @@
1 + ---
2 + hip: 9999
3 + title: "H4HIP: Chart-defined plugins"
4 + authors: [ "Scott Rigby <scott@r6by.com>", "George Jenkins <gvjenkins@gmail.com>" ]
5 + created: "2025-04-11"
6 + type: "feature"
7 + status: "draft"
8 + helm-version: "4"
9 + ---
10 +
11 + ## Abstract
12 +
13 + This HIP builds upon [H4HIP: Improve and extend plugin system](./hip-9999.md) to define how charts can specify and use plugins in Helm 4. It enables chart authors to define plugins that are required for their charts to function, such as alternative render engines or download protocols for dependencies.
14 +
15 + ## Motivation
16 +
17 + While the main plugin system HIP enables users to extend Helm's functionality, chart authors ([Application Distributors]
18 + (https://github.com/helm/community/blob/main/user-profiles.md#2-application-distributor)) need a way to specify which plugins their charts require. This allows charts to use alternative render engines, download protocols, and other plugin-based functionality while ensuring users have the necessary plugins installed.
19 +
20 + ## Specification
```

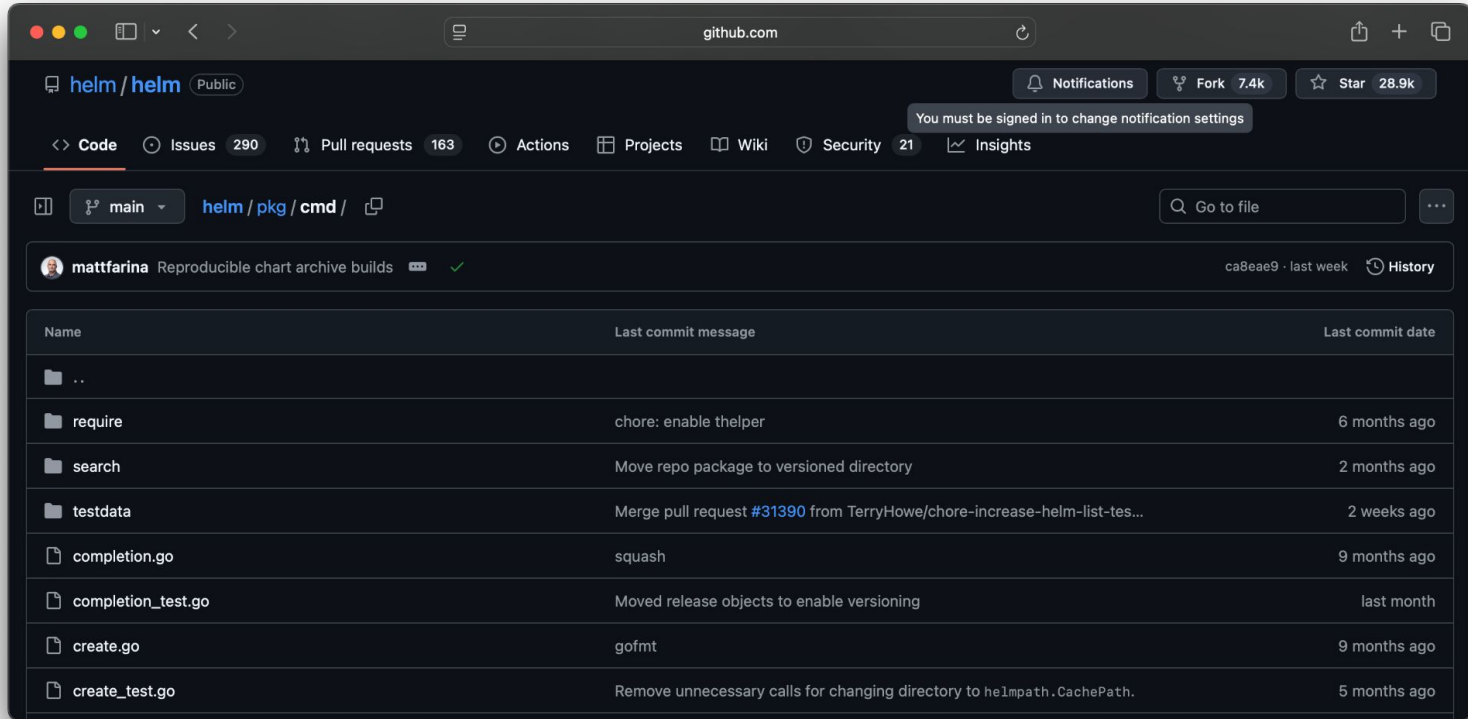
What Changed In Helm v4



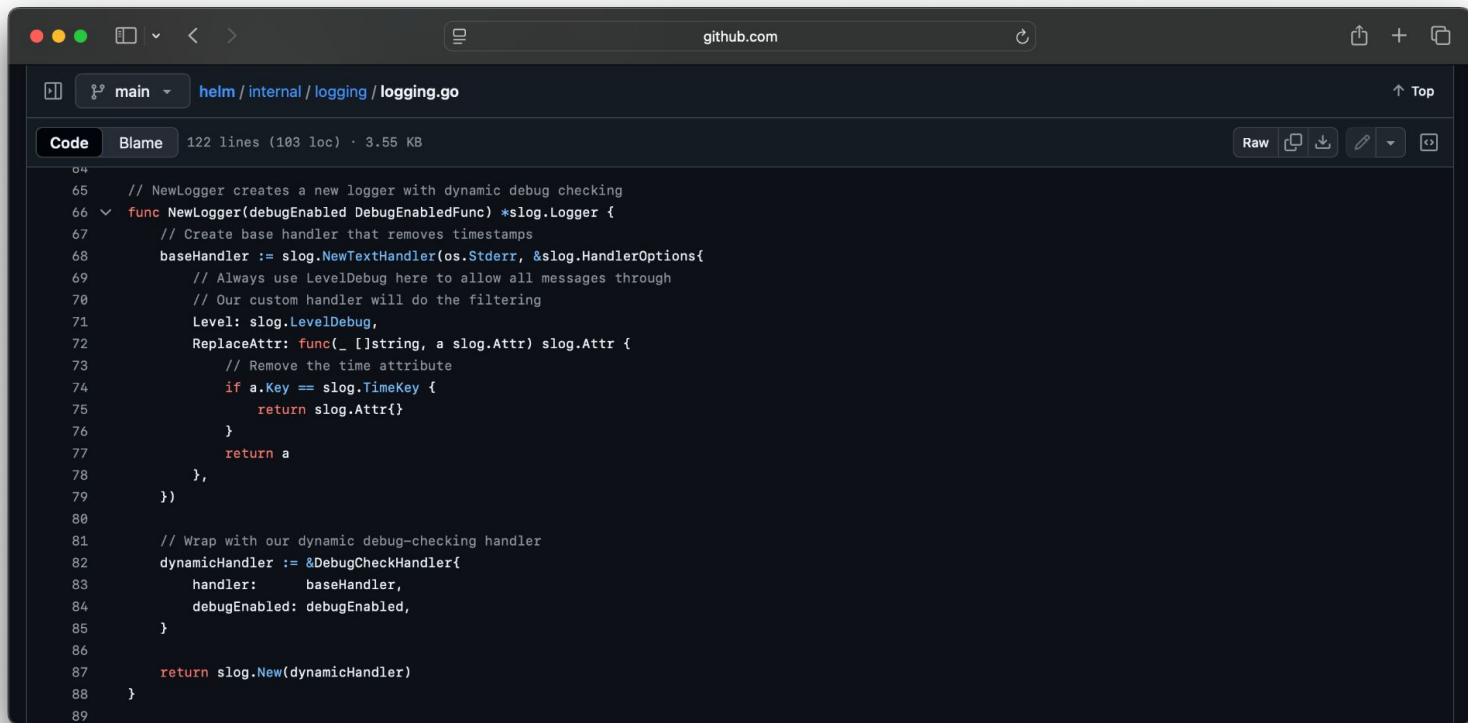
We Have Color

```
~  
> helm ls  
NAME      NAMESPACE    REVISION    UPDATED               STATUS      CHART          APP VERSION  
mdb       default       1           2025-11-04 15:57:16.478931 -0500 EST    deployed   mariadb-0.1.2   11.8.3  
  
~  
> |
```

Commands Can Be Embedded In Other Apps



slog Based Logging

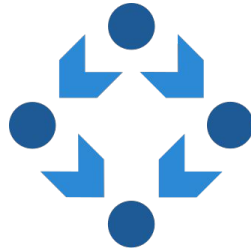


```
65 // NewLogger creates a new logger with dynamic debug checking
66 func NewLogger(debugEnabled DebugEnabledFunc) *slog.Logger {
67     // Create base handler that removes timestamps
68     baseHandler := slog.NewTextHandler(os.Stderr, &slog.HandlerOptions{
69         // Always use LevelDebug here to allow all messages through
70         // Our custom handler will do the filtering
71         Level: slog.LevelDebug,
72         ReplaceAttr: func(_ []string, a slog.Attr) slog.Attr {
73             // Remove the time attribute
74             if a.Key == slog.TimeKey {
75                 return slog.Attr{}
76             }
77             return a
78         },
79     })
80
81     // Wrap with our dynamic debug-checking handler
82     dynamicHandler := &DebugCheckHandler{
83         handler:    baseHandler,
84         debugEnabled: debugEnabled,
85     }
86
87     return slog.New(dynamicHandler)
88 }
89
```

Reproducible Chart Builds



+



**Reproducible
Builds**



KubeCon



CloudNativeCon

North America 2025

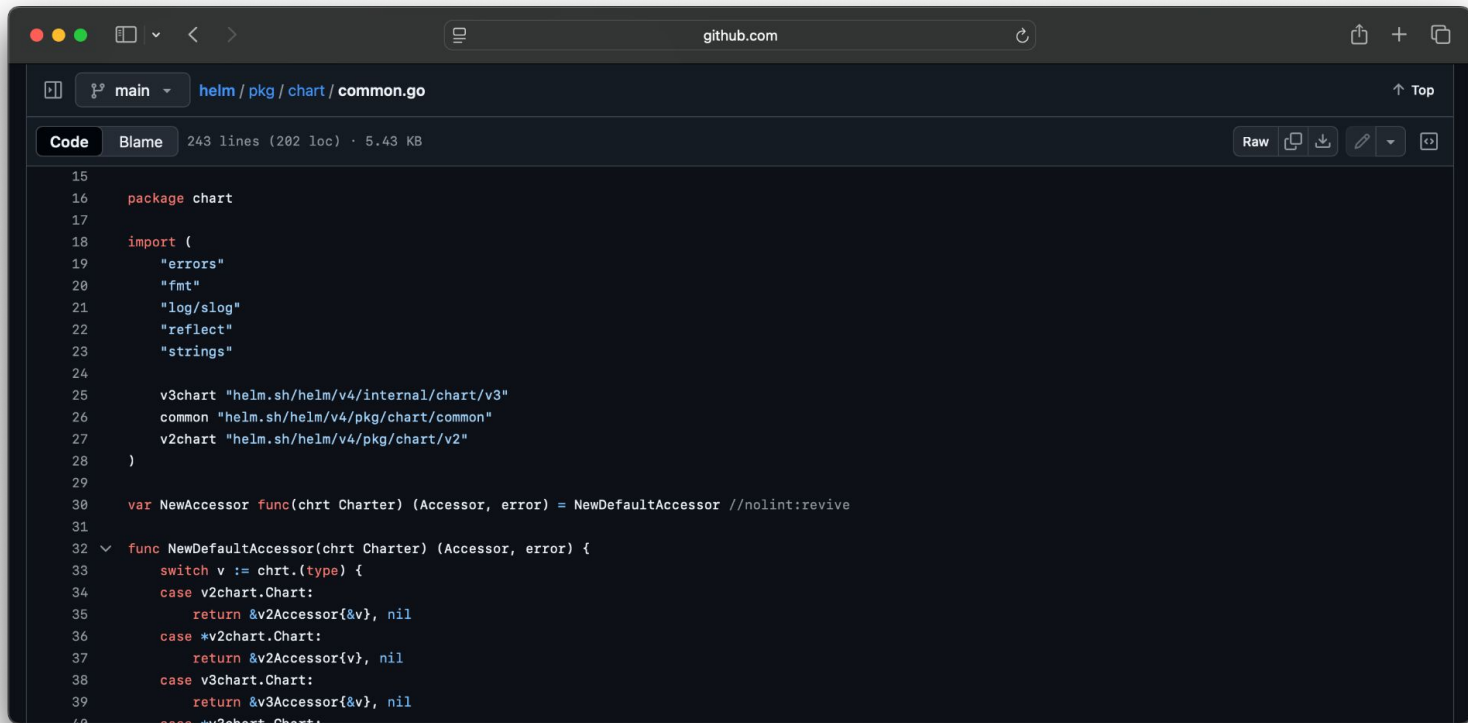
Helm Plugins Built With Web Assembly



+



Groundwork For New Chart apiVersions

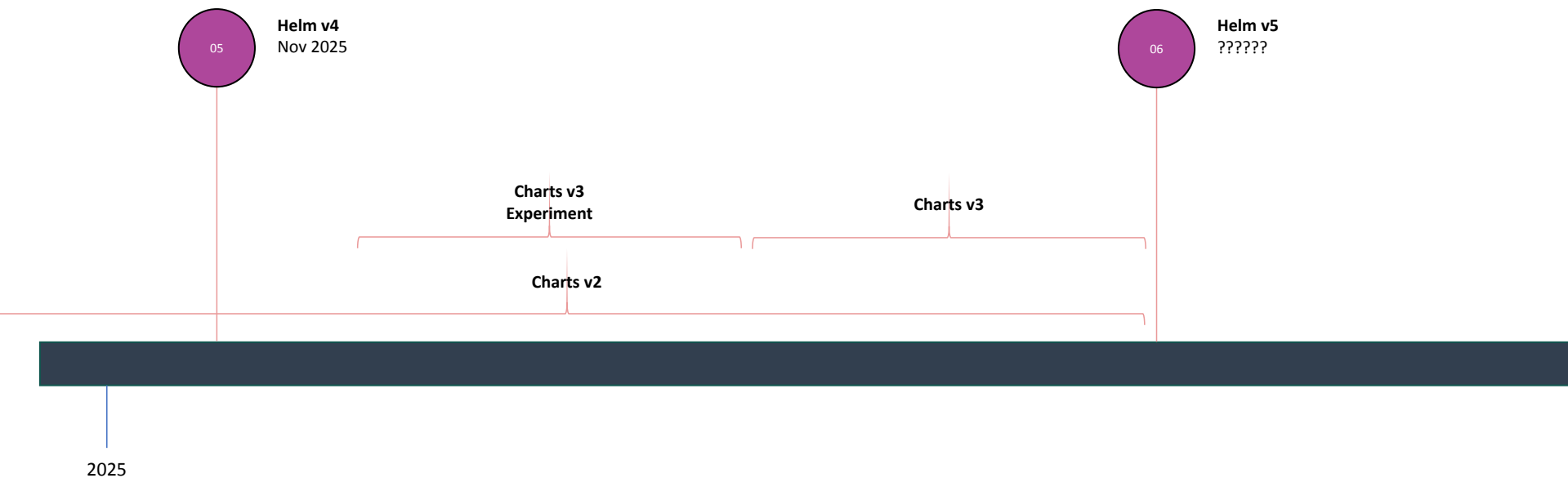


```
15
16 package chart
17
18 import (
19     "errors"
20     "fmt"
21     "log/slog"
22     "reflect"
23     "strings"
24
25     v3chart "helm.sh/helm/v4/internal/chart/v3"
26     common "helm.sh/helm/v4/pkg/chart/common"
27     v2chart "helm.sh/helm/v4/pkg/chart/v2"
28 )
29
30 var NewAccessor func(chrt Charter) (Accessor, error) = NewDefaultAccessor //nolint:revive
31
32 func NewDefaultAccessor(chrt Charter) (Accessor, error) {
33     switch v := chrt.(type) {
34     case v2chart.Chart:
35         return &v2Accessor{&v}, nil
36     case *v2chart.Chart:
37         return &v2Accessor{v}, nil
38     case v3chart.Chart:
39         return &v3Accessor{&v}, nil
40     case *v3chart.Chart:
```

What Is Still To Come In Helm v4?



Charts v3 Experiment Timeline



SDK Improvements



HELM SDK



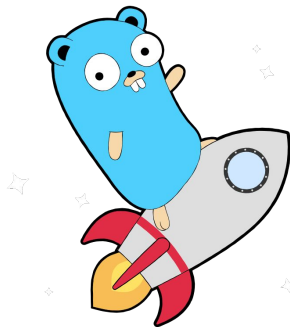
KubeCon



CloudNativeCon

North America 2025

goreleaser + cosign



+



sigstore
cosign



KubeCon



CloudNativeCon

North America 2025

How Long Will v3 Be Supported?



KubeCon



CloudNativeCon

North America 2025

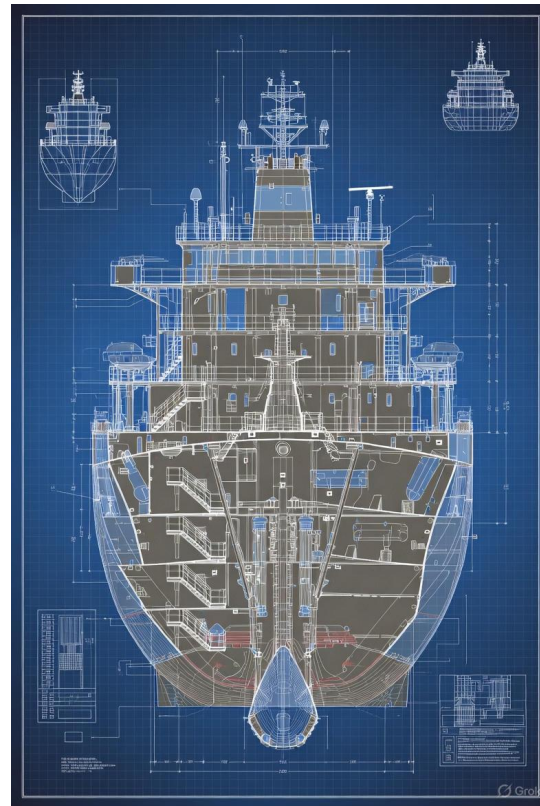
Helm v3 Support Cycle

- Bug fixes until July 8th 2026.
- Security fixes until November 11th 2026.
- No feature backports



Helm v5

- When: Sometime in the future (years away)
- Why: Breaking changes are needed to enable the future of Helm
- Ideas: ???



Getting Involved



Collaborate on Slack

Engage with other members of the Helm community



Code Contribution

Development of core capabilities, documentation and other assets



Join the Helm Community Meetings

Participate with other members of the Helm community to discuss efforts related to Helm

Getting Involved

CHARTS



Charts Tooling

Testing, CI actions, and more



helm.sh

Documentation

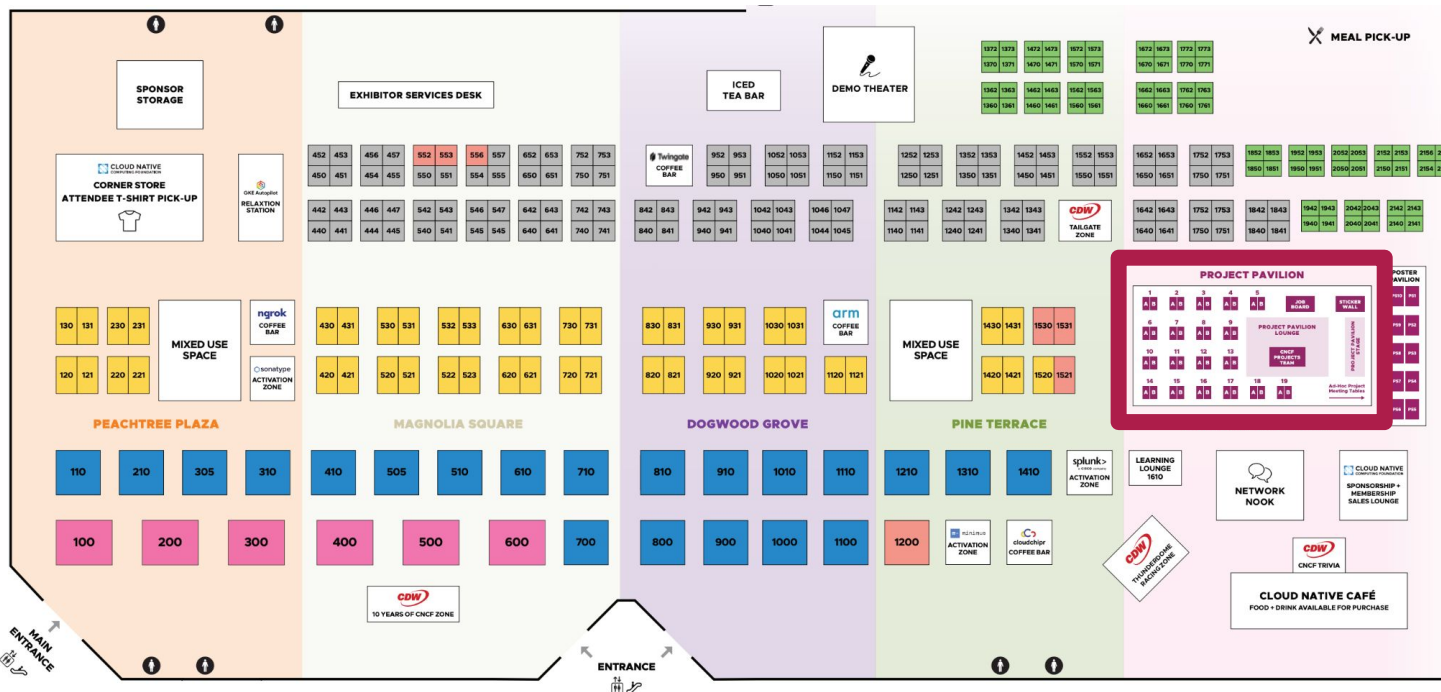
The website, docs, and translations



Helm CLI & SDK

Work on Helm itself. The Client and the SDK used by other applications such as Flux

KubeCon CloudNativeCon
North America 2025



Q&A



KubeCon



CloudNativeCon

— **North America 2025** —

