# GitOps

## Grundlagen und Best Practices



# DAS VORWORT

# Vorwort

Wir Autoren haben in Summe mehrere Jahrzehnte Erfahrung in den unterschiedlichsten Softwareprojekten gesammelt, teilweise als Softwareentwickler, oftmals als Platform Engineers. Uns sind Firmen und Problemstellungen mit unterschiedlichsten Größenordnungen, Branchen, Organisationsstrukturen und Deploymentwegen begegnet, und in alldem hat sich wenig so universal bewährt wie GitOps.

GitOps ist kein Tool, auch wenn es ohne gute Tools schwer umzusetzen ist. Statt eines Tools begegnen uns in GitOps eine Methodik, ein Satz an Grundprinzipien und eine Sammlung an etablierten Praktiken, mit deren Hilfe wir punktuell ausgeführte Deployments aus guten Gründen hinter uns lassen können. Und weil der Umstieg auf diese Methodik weit mehr als das Installieren eines Operators bedeutet, widmen wir diesem Thema ein umfassendes Buch.

Unsere Vision ist es, Entwicklungsteams im ganzen deutschsprachigen Raum zu helfen, in GitOps hineinzuwachsen. Wir wünschen uns, dass gerade kleine Teams ohne dedizierte DevOps-Kapazitäten in den Genuss derjenigen Vorteile von GitOps kommen, die wir mit traditionellem CIOps bisher nicht erreichen konnten:

- weitreichendere Automatisierung von Deployments
- höhere Stabilität unserer Softwaresysteme
- geringere Komplexität im Betrieb
- eine in Summe bessere Developer Experience
- erleichterter Zugang zu Kubernetes und dadurch leichteres Erlernen der Plattform

Wir erwarten, dass GitOps in wenigen Jahren der etablierte Standardweg sein wird, Kubernetes als Anwendungsentwickelnde zu nutzen. Unsere Hoffnung ist es, mit diesem Buch einen Beitrag zur Verbreitung von GitOps zu leisten.

## Hilfreiche Vorkenntnisse

GitOps ist aus den Erfahrungen von DevOps, Continuous Integration (CI), Continuous Deployment (CD) und Infrastructure as Code (IaC) hervorgegangen. Außerdem finden wir momentan nur im Bereich von Kubernetes wirklich gutes Tooling, um GitOps umzusetzen.

Deswegen ist es für das Verständnis dieses Buches sehr hilfreich, wenn du oder ihr als Team folgende Vorkenntnisse mitbringt:

- Ihr habt Erfahrung mit der Nutzung von CI/CD-Pipelines. (Die konkrete Plattform dafür spielt keine Rolle, und das Schreiben von Pipelines muss nicht eure Expertise sein, aber ihr solltet solche Pipelines aktuell nutzen.)
- Ihr habt ein grundlegendes Verständnis von Kubernetes und wie man darauf Container deployt. (Kenntnisse im Administrieren eines Clusters sind nicht notwendig, aber ihr solltet aus der Sicht eines Kubernetes-Nutzers wissen, was ein Deployment ist.)

## Aufbau des Buches

Unser Buch lässt sich grundsätzlich gut in der vorgegebenen Reihenfolge lesen. In Teil I beschäftigen wir uns mit den grundlegenden Fragen: Was ist GitOps, welchen Unterschied macht es im Entwicklungsalltag und wie fange ich praktisch mit GitOps an?

Teil II dreht sich dann um die Themen, die uns in der Praxis unmittelbar begegnen. Dabei gibt es einige Themen wie Secrets Management, Repo-Strukturen, Asynchronität und Alerting, bei denen ein allgemeines Verständnis zuerst wichtiger ist als ganz konkrete technische Implementierungen. Dennoch ist eine der sehr grundlegenden Entscheidungen, die wir beim Umstieg auf GitOps direkt am Anfang treffen müssen, die Wahl des GitOps-Operators. Mit Argo CD und Flux gibt es hier bereits zwei stark etablierte Tools.

*Kapitel 4 streift bereits einige nachfolgende Themen von Teil II.*

Weil die Wahl des Operators weitreichende Konsequenzen haben wird, stellen wir den Vergleich von Argo CD und Flux mit Kapitel 4 an den Anfang von Teil II. Wir streifen dabei auch einige der späteren Themen von Teil II. Wenn die Details von Kapitel 4 für den Anfang zu viel sind, dann kann es durchaus Sinn ergeben, dieses Kapitel anfangs zu überspringen und sich erst beispielsweise nach Kapitel 9 mit diesem Kapitel intensiver zu beschäftigen.

Teil III befasst sich mit weiterführenden Themen, die nicht immer unmittelbar mit dem Entwicklungsalltag zu tun haben, darunter die Verwaltung von Infrastruktur mittels GitOps und wie GitOps außerhalb von Kubernetes eingesetzt werden kann.

## Zielgruppen

*Softwareentwickelnde*

Wir schreiben dieses Buch in der Hoffnung, dass Softwareentwickelnde in Stream-Aligned Teams, wie sie in »Team Topologies«[1] beschrieben werden, zu schnelleren und stabileren Deployments befähigt werden. Damit sind Entwicklungsteams unsere primäre Zielgruppe.

Die Anforderungen, die heutzutage an Softwareentwicklung gestellt werden, reichen weit über reine Feature-Entwicklung hinaus und berühren auch Security und Finanzen. Mit »Shifting Left« sollen crossfunktionale Entwicklungsteams neben Feature-Entwicklung vermehrt auch viele dieser zusätzlichen Verantwortungen übernehmen.

*Platform Engineers*

Diese Komplexität ist oftmals zu viel für Entwicklungsteams, und deshalb bilden sich in vielen Organisationen sogenannte Plattformteams, die zentralisierte Services betreiben, um Entwicklungsteams mit geebneten Wegen viel kognitive Last abzunehmen. Unsere sekundäre Zielgruppe sind genau solche Platform Engineers.

Eine Kernkompetenz von Platform Engineering ist das Befähigen von Entwickelnden. In diesem Sinne sprechen wir zwar mit Teil III des Buches deutlich mehr Platform Engineers an, während Teil I und II aus Sicht von Platform Engineers besonders für das Enablement von Entwickelnden hilfreich sind, weil dort komplizierte GitOps-Konzepte gut zugänglich gemacht werden.

## Hervorhebungen

Wir nutzen im Buch folgende Hervorhebungen:

- *Kursiven Text* nutzen wir bei der ersten Erwähnung eines wichtigen Konzepts und auch zur generellen Hervorhebung im Fließtext.
- `Monospace` nutzen wir für Folgendes:
  1. CLI-Befehle im Fließtext (zum Beispiel `kubectl apply`)
  2. Namen von Custom Resource Definitions in Kubernetes (zum Beispiel `Kustomization`)
  3. Dateipfade (zum Beispiel `ch05/bootstrap/argo-cd.yaml`)
  4. Namen von Entitäten aus Code-Listings (zum Beispiel `podinfo`)

[1] Skelton, M., Pais, M., & Malan, R. Team Topologies: Organisation von Business- und IT-Teams für einen schnellen Arbeitsfluss, 2023.

Abgesehen davon gibt es einige Blockformatierungen, die wir folgendermaßen nutzen:

Eine Nebenbemerkung, die oftmals externe Verweise enthält, zum Beispiel auf Beispiel-Repositories.

**Exkurs**

Eine Ausführung, die weiterführende Infos zum Thema gibt, aber zum Verständnis des Fließtexts nicht zwingend notwendig ist.

## Website

Wir wünschen viel Spaß beim Lesen unseres Buches und viel Gewinn davon für dein Team! Auf der Website zum Buch[2] werden wir Neuigkeiten und (sofern nötig) Errata veröffentlichen.

Wir freuen uns sehr, wenn du uns Feedback zum Buch geben möchtest! Nutz dafür gerne die Website zum Buch oder kontaktiere uns auf LinkedIn.

[2] *https://gitops-book.dev*