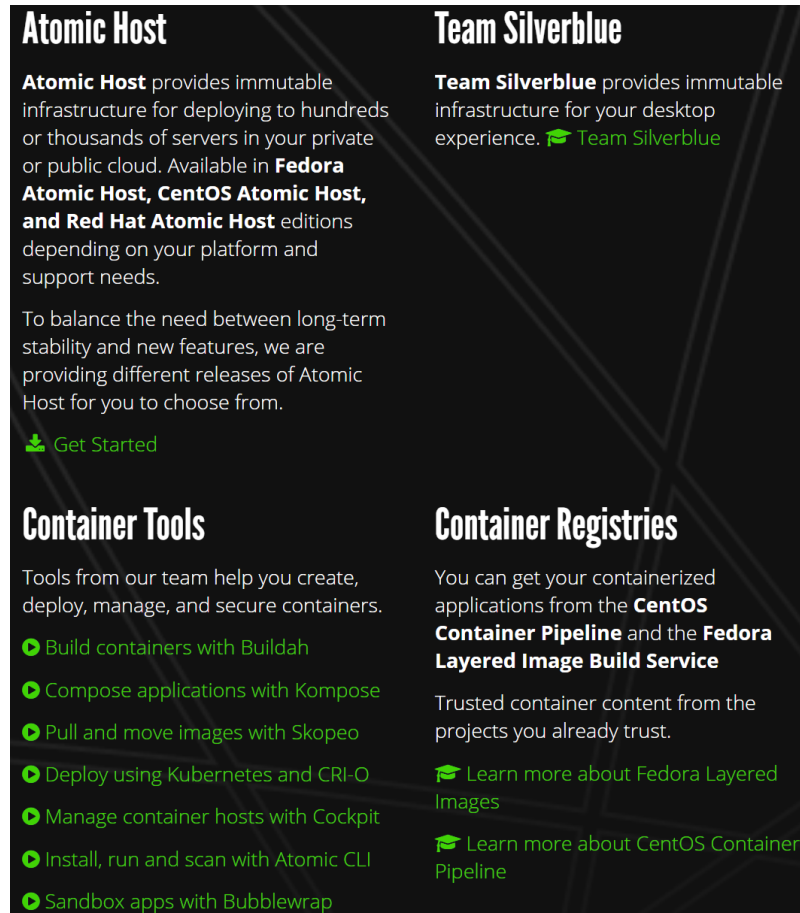


# Weitere Plattformen und Abschluss

# Zeitlicher Ablauf

- ★ Weitere Plattformen
- ★ Bücher
- ★ Beispiele
- ★ Abschluss

# Atomic Host (RHEL/CentOS)



**Atomic Host**

**Atomic Host** provides immutable infrastructure for deploying to hundreds or thousands of servers in your private or public cloud. Available in **Fedora Atomic Host, CentOS Atomic Host, and Red Hat Atomic Host** editions depending on your platform and support needs.

To balance the need between long-term stability and new features, we are providing different releases of Atomic Host for you to choose from.

[Get Started](#)

**Container Tools**

Tools from our team help you create, deploy, manage, and secure containers.

- Build containers with Buildah
- Compose applications with Kompose
- Pull and move images with Skopeo
- Deploy using Kubernetes and CRI-O
- Manage container hosts with Cockpit
- Install, run and scan with Atomic CLI
- Sandbox apps with Bubblewrap

**Team Silverblue**

**Team Silverblue** provides immutable infrastructure for your desktop experience. [Team Silverblue](#)

**Container Registries**

You can get your containerized applications from the **CentOS Container Pipeline** and the **Fedora Layered Image Build Service**

Trusted container content from the projects you already trust.

- [Learn more about Fedora Layered Images](#)
- [Learn more about CentOS Container Pipeline](#)

- ★ Atomic fungiert als dedizierte, extrem abgespeckte RHEL/CentOS/Fedora-basierte Container-Plattform.
- ★ Der Fokus von Atomic liegt klar auf dem Einsatz als Arbeitspferd im Rahmen von Kubernetes Clustern.

# Rocket Science? – CoreOS/Container Linux

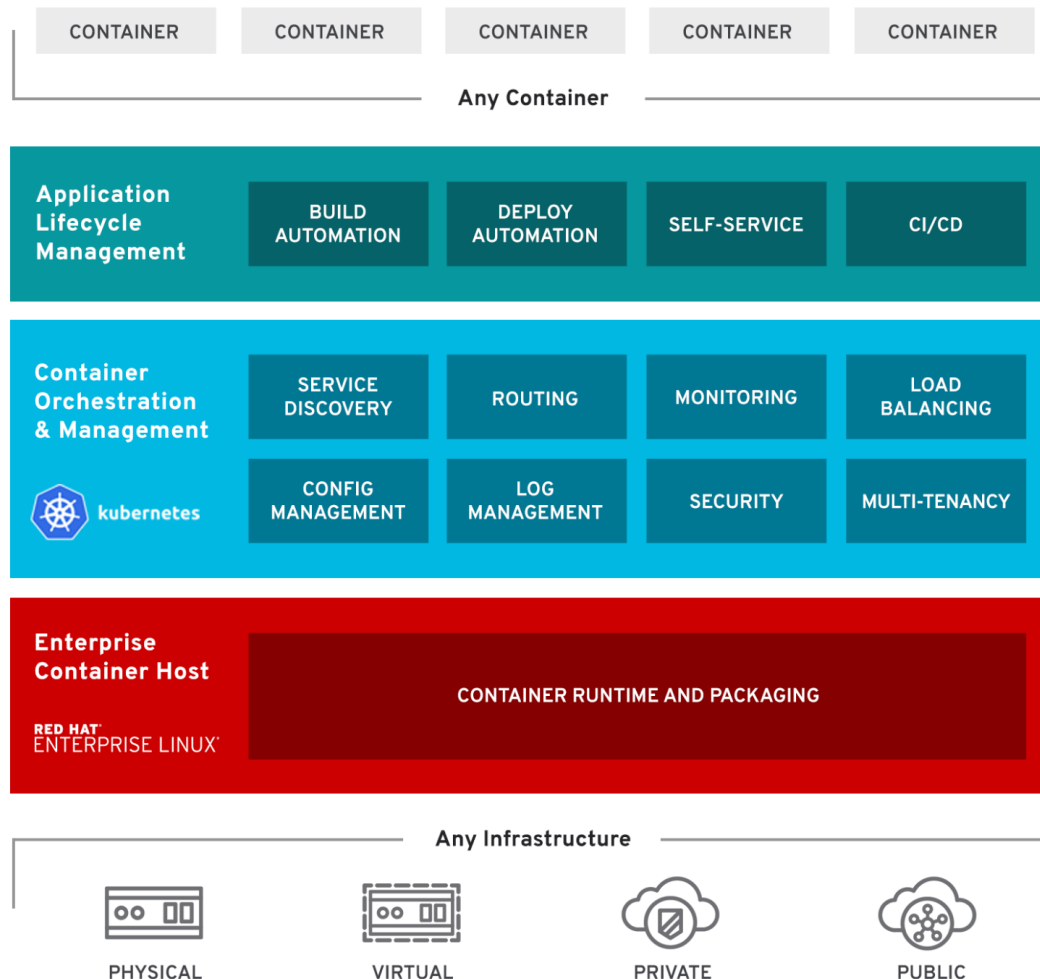
## CoreOS has joined the Red Hat family

The CoreOS team is thrilled to have joined Red Hat®.

Together we are working to further extend the value of Kubernetes for all of our customers.

- ★ Die zunächst euphorische Marktakzeptanz für CoreOS/Rocket schwindet nach wie vor.
- ★ **Im Single Node-Einsatz bringt CoreOS / Container Linux keinen wirklichen Benefit gegenüber Docker, im Gegenteil – es ist bisweilen eher sperriger zu handhaben.**
- ★ Unterschiede zwischen Docker- und Rocket-Container
  - **actool**: das Rocket-spezifische Container-Build-Tool
  - **rkt**: der eigentliche Rocket-Kernbefehl: Verwaltung und Administration von Images, Pods (*Containern*):

# OpenShift (Red Hat)



Quelle: Buch Skalierbare Container-Infrastrukturen, ab Kapitel 17

Quelle: <https://www.openshift.com/learn/what-is-openshift/>

## The Kubernetes platform for big ideas

Focus on writing code and let OpenShift build, run, and scale your apps in the cloud

- ★ Traditionell und Cloud-nativ. On-Premise und in der Cloud. Ganz gleich, was Sie bauen, Red Hat® OpenShift® ist die führende **Kubernetes**-Plattform, um Ihren Kunden das Aussergewöhnliche schneller zu liefern, als Sie sich vorstellen können.
- ★ Oder einfach ausgedrückt:
  - **Der übergeordnete Orchestrierer für den K8s-Orchestrierer, mit intelligenten Management und Build-Funktionalitäten, integrierter Registry und Software-Katalogen und einem komplettem Lifecycle-Management.**
- ★ <https://cloudowski.com/articles/10-differences-between-openshift-and-kubernetes/>

# OpenShift (Red Hat) - Varianten

## ★ 17.2.1 OpenShift Origin/OKD

- OpenShift Origin ist das ohne Lizenzkosten verfügbare Upstream-OpenSource-Community-Projekt, das als Grundlage für OpenShift Online, OpenShift Dedicated und OpenShift Container Platform verwendet wird.

## ★ 17.2.2 OpenShift Online (NextGen OpenShift)

- OpenShift Online ist Red Hats kommerzieller Public Cloud Application-Entwicklungs- und Hosting-Service, der seit Mai 2017 verfügbar ist. OpenShift Online ist die online angebotene und von Red Hat gehostete Version des Origin-Projekt-Quellcodes.

## ★ 17.2.3 OpenShift Dedicated (Cloud Services)

- OpenShift Dedicated ist der von Red Hat gehostete OpenShift Private Cloud Service für private Unternehmens-Cluster (Single-Tenant) in Public Clouds.

## ★ Weitere Informationen: <https://www.openshift.com/products/pricing/>

# OpenShift (Red Hat) - Unterschiede

- ★ **Projects** (K8s Namespaces), welches folgendes beinhalten:
  - Objects – Pods, ReplicaSets, Deployments, Services usw.
  - Policies (ACLs) – vergleiche Kubernetes → RBAC
  - Constraints – im Sinne von OpenShift sind damit Quotas gemeint
  - Service Accounts – bestimmen den Zugriff auf Objekte im Project.
- ★ **OAuth-Server** (Authentication und IDM)
- ★ **Service Broker** und **Catalog** (Applikationsverzeichnis und Bereitstellung)
- ★ **Networking**, Standard K8s und Isolierung auf Projektebene für Pods und Services
- ★ **Router**, vereinfachte Version des Ingress Dienstes
- ★ **Steuerung eines kompletten Container-Lifecycles**: Build plus Rollout per OpenShift-GUI

# Ubuntu

## Install Kubernetes on Ubuntu

Kubernetes on Ubuntu is free to use and always current - you get the latest innovations from the Kubernetes community within a week of upstream release. It works on any cloud (public, private, and bare-metal). Kubernetes on Ubuntu is the productive, open source way to manage containers and microservices, automating the time-consuming tasks of installing, patching, upgrading, and carrying out cluster health checks.

### Single node with MicroK8s

Install MicroK8s, the Linux snap that downloads in seconds. Microk8s is lightweight and deploys all Kubernetes services natively on Ubuntu. Ideal for:

- ✓ **Laptops:** develop microservices locally
- ✓ **Workstations:** develop and train machine learning models locally
- ✓ **CI pipelines:** create ephemeral Kubernetes quickly for testing microservices
- ✓ **IoT devices:** embed upgradeable Kubernetes in your IoT devices for easy evolution
- ✓ **Small edge clouds:** create simple Kubernetes clusters for your edge clouds

### Multi node with Charmed Kubernetes

Install Charmed Kubernetes, Ubuntu's highly available, multi node Kubernetes cluster on your infrastructure of choice:

- ✓ **Bare metal:** deploying Kubernetes on bare metal is easy using Charmed Kubernetes and MAAS (Metal-as-a-Service).
- ✓ **Private clouds:** take advantage of your on-premises clouds to deploy one or more Kubernetes clusters (VMware vSphere, OpenStack, LXD)
- ✓ **Public clouds:** deploy Charmed Kubernetes to AWS, GCP, Azure, IBM, and Oracle





# SLE MicroOS (SUSE CaaS-Plattform)

- ★ Die SUSE CaaS-Plattform ist eine Container-Verwaltungslösung für Geschäftsanwendungen, mit der IT- und DevOps-Experten containerbasierte Anwendungen und Services leichter bereitstellen, verwalten und skalieren können. Sie enthält **Kubernetes** zur Automatisierung der Verwaltung des Geräte-Lebenszyklus sowie entsprechende Technologien, die Kubernetes erweitern und eine einfache Bedienung der Plattform ermöglichen. Infolgedessen können Unternehmen, die auf die SUSE CaaS-Plattform setzen, Anwendungsbereitstellungszyklen verkürzen und die geschäftliche Agilität verbessern.
- ★ Weiter Informationen: <https://www.suse.com/de-de/products/caas-platform/>

# RancherOS

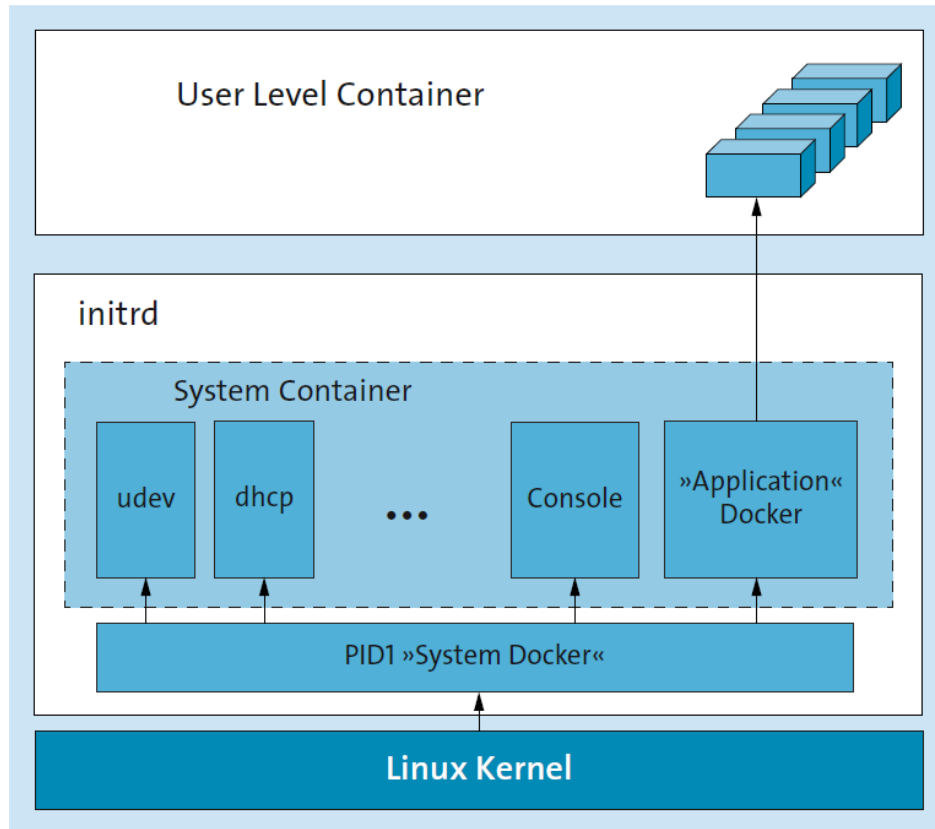


Abbildung 7.4 Aufbau des containerisierten RancherOS

- ★ RancherOS, aus dem Haus der Rancher Labs, setzt als Einzige der bisher betrachteten Plattformen echte Containerisierung um.
- ★ Konsequenter ist es im Vergleich zu seinen Kontrahenten allemal. Stark vereinfacht startet der Kernel eine speziell angepasste initrd, welche wiederum Docker mit PID 1 als Init-Prozess startet. Dieser startet dann wiederum alle Systemdienste wie udev, DHCP, die Shells usw. als Container und natürlich auch noch eine weitere Docker-Instanz. Diese verwaltet dann die eigentlichen Nutz-Container.

# Docker unter Windows

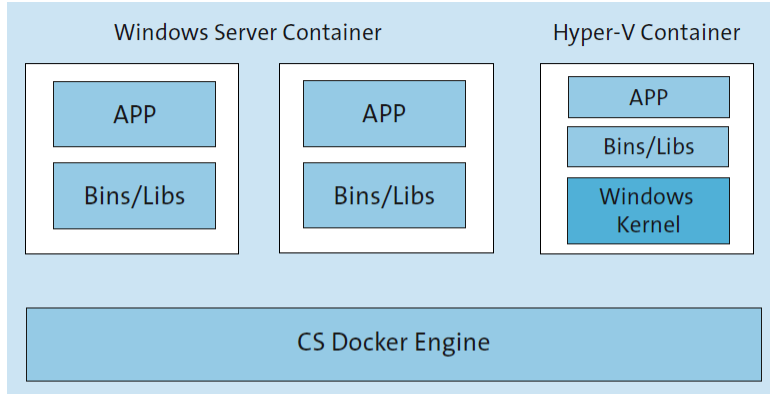


Abbildung 7.1 Docker unter

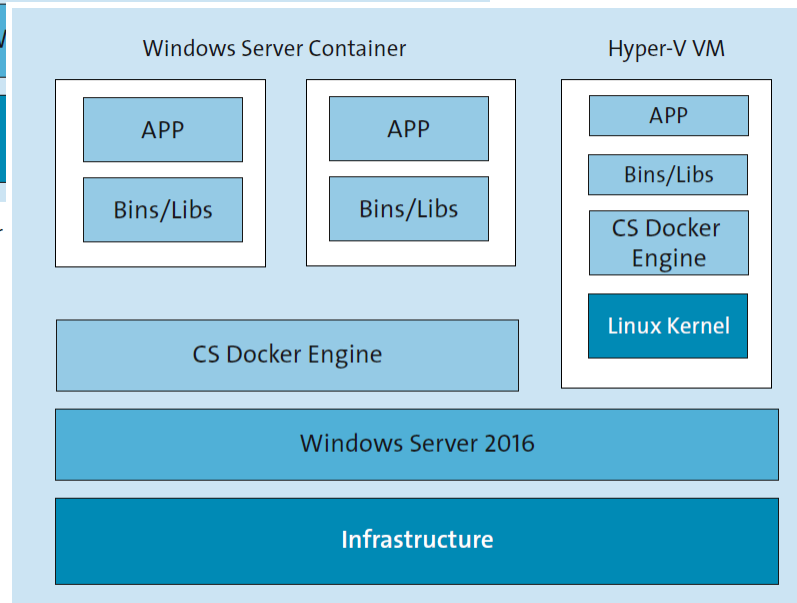


Abbildung 7.2 Normale und Hyper-V-Container unter Windows

- ★ Docker läuft seit Ende 2016 auch nativ unter Windows 10 oder Server 2016, d. h. als echter Windows Service und nicht mehr wie zuvor in einer Linux VM unter Windows.
- ★ Der Betrieb von Docker-Containern unter Windows, ist in etwa, grob vergleichbar mit den bereits vorgestellten Verfahren und Konzepten.
- ★ Weitere Informationen:
  - <https://docs.docker.com/docker-for-windows/>
  - <https://docs.microsoft.com/en-us/virtualization/windowscontainers/about/>
- ★ **Empfehlung:**
  - Windows Container gehören auf Windows, Linux Container auf Linux.

# Docker Community (CE) / Enterprise (EE)

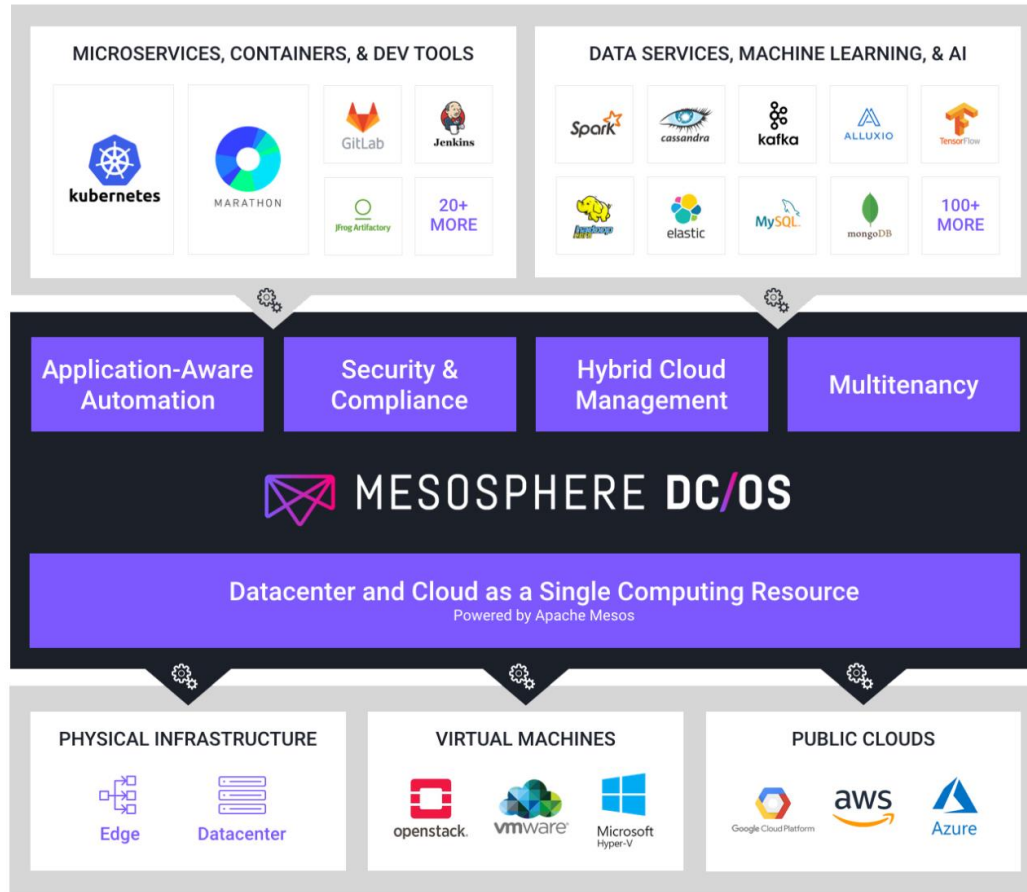
## ★ Docker CE

- Stellt dar, was ohnehin längst dringend benötigt wurde, um zumindest einen grundlegenden, längerfristigen und stabilen Container-Betrieb zu gewährleisten.
- Der Swarm Mode hat grosse Defizite im Bereich Constraints, was ihn für **grosse Cluster-Umgebungen disqualifiziert**.

## ★ Docker EE

- Die EE (Enterprise Edition), die nichts anderes als ein neuer Name für das DDC mit dem inkludierten und bereits vorgestellten CVE-Scanner ist, resultiert wohl am ehesten aus dem Bestreben, zumindest namentlich einen Anschluss an Mesospheres DC/OS EE zu erhaschen.

# (Mesosphere) DC/OS



- ★ Wenn K8s eine ziemlich dicke Luxus-Hochseejacht ist, dann ist **DC/OS** der **Flugzeugträger**.
- ★ **Gross, mächtig, beeindruckend.** Und genau wie der Träger hat DC/OS derzeit etliche Stellschrauben, an denen gedreht werden muss, bevor es losgeht.
- ★ Im Kern muss Mesosphere insbesondere daran arbeiten, den **Setup-Vorgang** von Bare Metal Clustern deutlich zu vereinfachen???. Hier ist Nacharbeit angesagt, ebenso bei der Dokumentation.

# Bücher und Links

- ★ Skalierbare Container-Infrastrukturen, ISBN 978-3-8362-6386-3, [2. Auflage](#)
- ★ Kubernetes, ISBN: 978-3-86490-542-1 (free eBook liegt bei)
- ★ Cloud Native Infrastructure, ISBN: 978-1-49198-425-3 (free eBook liegt bei)
- ★ [Kubernetes Learning Resources](#) List
- ★ [50 Best Kubernetes Architecture](#) Tutorials
- ★ [70 Best Kubernetes](#) Tutorials
  
- ★ <https://12factor.net/>
- ★ <https://www.owasp.org>

# KubeWeekly

- ★ [Kubernetes Security Audit](#)
- ★ [Tools and Methods for Auditing Kubernetes RBAC Policies](#)
- ★ [K8s scheduling](#) — deep dive
- ★ [KUBERNETES WEB UIS](#) IN 2019
- ★ [Multitenancy](#) on kubernetes with Istio, External Authentication Server and OpenID Connect
- ★ What Is [GitOps](#) Really?
- ★ [12 Kubernetes configuration](#) best practices

# Verzeichnis der Beispiele

## ★ Codebeispiele zum Buch

- [https://s3-eu-west-1.amazonaws.com/gxmedia.galileo-press.de/supplements/4252/skalierbare container infrastrukturen 4366.zip](https://s3-eu-west-1.amazonaws.com/gxmedia.galileo-press.de/supplements/4252/skalierbare_container_infrastrukturen_4366.zip)

## ★ DevOps (Beispiele zu Docker, Kubernetes)

- <https://github.com/mc-b/duk>

## ★ Beispiele zum Kurs Microservices-Grundlagen («MISEGR»)

- Hauptprojekt - <https://github.com/mc-b/misegr>
- Frontend Integration - <https://github.com/mc-b/SCS-ESI>
- Asynchrone Microservices - <https://github.com/mc-b/microservice-kafka>
- Synchrone Microservices - <https://github.com/mc-b/microservice-kubernetes>
- BPMN - <https://github.com/mc-b/bpmn-tutorial>



# Abschluss

★ Viel Erfolg mit Docker und Kubernetes!

★ Marcel Bernet - <https://github.com/mc-b>