

# Inhalt

Vorwort .....	9
---------------	---

## TEIL I Einführung

---

<b>1 Hello World .....</b>	<b>13</b>
1.1 Docker-Schnellinstallation .....	13
1.2 Apache mit PHP7 .....	14
1.3 Node.js .....	17
1.4 Python .....	20
1.5 Fazit .....	21
<b>2 Installation und Grundlagen .....</b>	<b>23</b>
2.1 Installation .....	23
2.2 Grundlagen und Nomenklatur .....	28
2.3 Docker kennenlernen .....	33
2.4 Docker administrieren .....	50
2.5 Ein Blick hinter die Kulissen .....	52
<b>3 Eigene Docker-Images (Dockerfiles) .....</b>	<b>61</b>
3.1 Dockerfiles .....	61
3.2 Images in den Docker Hub hochladen .....	71
3.3 Pandoc- und LaTeX-Umgebung als Image einrichten .....	72
<b>4 docker-Kommandoreferenz .....</b>	<b>77</b>
<b>5 docker-compose .....</b>	<b>97</b>
5.1 Installation von docker-compose unter Linux .....	98
5.2 YAML-Syntax .....	99
5.3 Hello World! .....	101
5.4 Die docker-compose.yml-Datei .....	106
5.5 Das docker-compose-Kommando .....	114

**TEIL II Werkzeugkasten**

---

<b>6</b>	<b>Alpine Linux</b> .....	121
6.1	Merkmale .....	122
6.2	Paketverwaltung mit apk .....	125
<b>7</b>	<b>Webserver und Co</b> .....	129
7.1	Apache HTTP Server .....	129
7.2	Nginx .....	133
7.3	Nginx als Reverse-Proxy mit SSL-Zertifikaten von Let's Encrypt .....	136
7.4	Node.js mit Express .....	144
7.5	HAProxy .....	150
<b>8</b>	<b>Datenbanksysteme</b> .....	155
8.1	MySQL und MariaDB .....	155
8.2	PostgreSQL .....	161
8.3	MongoDB .....	166
8.4	Redis .....	171
<b>9</b>	<b>Programmiersprachen</b> .....	175
9.1	JavaScript (Node.js) .....	175
9.2	Java .....	179
9.3	PHP .....	182
9.4	Ruby .....	189
9.5	Python .....	190
9.6	Swift .....	197
<b>10</b>	<b>Webapplikationen und CMS</b> .....	203
10.1	WordPress .....	203
10.2	Nextcloud .....	211
10.3	Joomla .....	214

## TEIL III Praxis

---

<b>11</b>	<b>Eine moderne Webapplikation</b> .....	219
11.1	Die Anwendung .....	220
11.2	Das Frontend – Vue.js .....	222
11.3	Der API-Server – Node.js Express .....	233
11.4	Die Mongo-Datenbank .....	243
11.5	Der Session-Speicher – Redis .....	247
<b>12</b>	<b>Grafana</b> .....	249
12.1	Grafana-Docker-Setup .....	250
12.2	Provisioning .....	259
12.3	Ein angepasstes Telegraf-Image .....	261
<b>13</b>	<b>Modernisierung einer traditionellen Applikation</b> .....	267
13.1	Die bestehende Applikation .....	268
13.2	Planung und Vorbereitung .....	270
13.3	Die Entwicklungsumgebung .....	284
13.4	Produktivumgebung und Migration .....	285
13.5	Updates .....	287
13.6	Tipps zur Umstellung .....	289
13.7	Fazit .....	289
<b>14</b>	<b>GitLab</b> .....	291
14.1	GitLab-Schnellstart .....	293
14.2	GitLab-Webinstallation .....	294
14.3	HTTPS über ein Reverse-Proxy-Setup .....	296
14.4	E-Mail-Versand .....	298
14.5	SSH-Zugriff .....	301
14.6	Volumes und Backup .....	302
14.7	Eigene Docker-Registry für GitLab .....	304
14.8	Die vollständige docker-compose-Datei .....	306
14.9	GitLab verwenden .....	307
14.10	GitLab Runner .....	312
14.11	Mattermost .....	318

<b>15</b>	<b>Continuous Integration und Continuous Delivery</b> .....	325
15.1	Die dockerbuch.info-Website mit gohugo.io .....	326
15.2	Docker-Images für die CI/CD-Pipeline .....	331
15.3	Die CI/CD-Pipeline .....	334
<b>16</b>	<b>Sicherheit</b> .....	347
16.1	Software-Installation .....	347
16.2	Herkunft der Docker-Images .....	349
16.3	»root« in Docker-Images .....	351
16.4	Der Docker-Dämon .....	353
16.5	User-Namespaces .....	354
16.6	cgroups .....	356
16.7	Secure Computing Mode .....	358
16.8	AppArmor-Sicherheitsprofile .....	358
<b>17</b>	<b>Swarm und Amazon ECS</b> .....	361
17.1	Docker Swarm .....	363
17.2	Docker Swarm in der Hetzner Cloud .....	367
17.3	Amazon Elastic Container Service .....	378
<b>18</b>	<b>Kubernetes</b> .....	389
18.1	Minikube .....	390
18.2	Amazon EKS (Elastic Container Service for Kubernetes) .....	401
18.3	Microsoft AKS (Azure Kubernetes Service) .....	409
18.4	Google Kubernetes Engine .....	416
	Index .....	427

# Vorwort

Zu Beginn der 2000er-Jahre stellte Virtualisierungssoftware den Alltag vieler Entwickler auf den Kopf: Plötzlich war es möglich, auf einem Rechner Linux *und* Windows auszuführen, Programme unkompliziert in verschiedenen Umgebungen bzw. Web-Apps in alten Versionen von Webbrowsern auszuprobieren, verschiedene Software-Stacks in virtuellen Maschinen parallel zu installieren und zu testen und vieles mehr.

Natürlich spielen virtuelle Maschinen für Entwickler weiter eine große Rolle; außerdem ist die Cloud in ihrer jetzigen Form ohne Virtualisierung gar nicht denkbar. Dennoch hat vor einigen Jahren ein Umbruch weg von virtuellen Maschinen hin zu Containern begonnen – und dieser Umbruch scheint sich mehr und mehr zu beschleunigen.

Container ermöglichen es, bestimmte Software-Komponenten (Webserver, Programmiersprachen, Datenbanken) ohne den Overhead einer virtuellen Maschine auszuführen. Warum ein ganzes Betriebssystem (meist Linux) in eine virtuelle Maschine installieren, wenn es doch nur um *eine* ganz spezifische Funktion geht?

Selten trifft das Paradigma »Weniger ist mehr« so gut zu wie auf die Container-Technologie. Das *Weniger* drückt sich in unzähligen Vorteilen aus: Container sind viel schneller aufgesetzt als virtuelle Maschinen, lassen sich leichter auf verschiedenen Entwicklungssystemen replizieren, beanspruchen weniger Ressourcen und bieten wesentlich bessere Möglichkeiten zur Skalierung und Lastverteilung. Container sind insofern nicht nur ein Segen für Entwicklerteams, sondern bieten auch vollkommen neue Möglichkeiten im Deployment, also im produktiven Betrieb der entwickelten Lösung.

## Docker

Docker ist *die* Container-Software schlechthin. Zwar gibt es mittlerweile durchaus interessante Alternativen, aber Docker hat den Container-Markt als solchen geschaffen und gibt mit seiner enorm schnellen Weiterentwicklung immer noch das Tempo vor.

Docker steht für alle gängigen Plattformen in einer *Community Edition* kostenlos als Open-Source-Software zur Verfügung. Parallel dazu bietet die Firma *Docker Inc.* Zusatzfunktionen und Support für kommerzielle Anwender an.

## Wozu dieses Buch?

In diesem Buch geben wir eine Einführung in den Umgang mit Docker und einen Überblick über die wichtigsten Bausteine (Images), aus denen Sie eigene Container-Welten zusammensetzen können. Wir zeigen anhand mehrerer großer Beispiele, wie Docker in der Praxis eingesetzt wird, und gehen ausführlich auf das Deployment in der Cloud (Docker Swarm und Kubernetes) ein.

Wir haben das Buch in drei Teile gegliedert:

- ▶ **Teil I** stellt Docker vor. Sie lernen anhand vieler Beispiele, wie Sie die Kommandos `docker` und `docker-compose` sinnvoll einsetzen und wie die Syntax der Dateien `Dockerfile` und `docker-compose.yml` aussieht.
- ▶ **Teil II** präsentiert wichtige Images, die als Basis für eigene Projekte dienen können. Dazu zählen unter anderem:
  - Alpine Linux
  - die Webserver Apache und Nginx (inklusive Proxy-Setup und Let's-Encrypt-Konfiguration)
  - die Datenbankserver MySQL/MariaDB, MongoDB, PostgreSQL und Redis
  - die Programmiersprachen JavaScript (Node.js), Java, PHP, Ruby, Python und Swift
  - die Webapplikationen WordPress, Joomla und Nextcloud
- ▶ **Teil III** zeigt den Einsatz von Docker in der Praxis. Wir zeigen Ihnen sowohl, wie Sie moderne Webapplikationen mit Docker besonders effizient entwickeln, als auch, wie Sie vorhandene Projekte mit all ihren Altlasten in besser wartbare Docker-Projekte umwandeln.

Zwei Kapitel zur Nutzung von GitLab mit Docker und zu *Continuous Integration* und *Continuous Delivery* (CI und CD) demonstrieren Ihnen neue Paradigmen und Hilfsmittel, um Software im Team zu entwickeln.

Auch das Deployment kommt nicht zu kurz: Mit Docker Swarm und Kubernetes bringen Sie Ihre Docker-Projekte in die Cloud und profitieren von den dort gegebenen Möglichkeiten zur Skalierung. Eine Sammlung von Tipps stellt sicher, dass dabei die Sicherheit nicht zu kurz kommt.

Wer Docker einmal ausprobiert und kennengelernt hat, will nie wieder auf seine Funktionen verzichten. Lassen Sie sich von uns in eine neue Welt führen!

Bernd Öggl (<https://komplett.cc>)  
Michael Kofler (<https://kofler.info>)