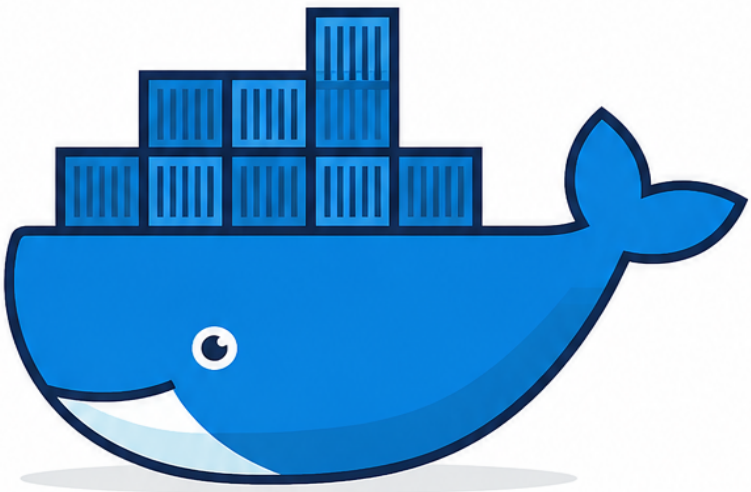
DOCKER CONTAINERS

FROM BEGINNER TO MASTERY

A COMPLETE GUIDE TO BUILDING, RUNNING, AND MANAGING CONTAINERIZED APPLICATIONS



BUILD IMAGES

NETWORKING & STORAGE

SECURITY BEST PRACTICES

ORCHESTRATION WITH DOCKER & KUBERNETES

CI/CD INTEGRATION

REAL-WORLD WORKFLOWS

MICHAEL TURN

Docker Containers: From Beginner to Mastery

A Complete Guide to Building, Running, and Managing Containerized Applications

Steve T. Publications

This book is available at
https://leanpub.com/dockercontainersfrombeginnertomastery

This version was published on 2026-07-17

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

Contents

A Complete Guide to Building, Running, and Managing Containerized Applications

About this book: Docker containers have fundamentally transformed how software is built, shipped, and run. This comprehensive guide takes you from zero container knowledge through advanced production patterns. You will learn Docker architecture, image building, networking, storage, security, orchestration, CI/CD integration, and real-world workflows. Every chapter includes practical examples, annotated command-line snippets, hands-on exercises, best practices, and common pitfalls. Whether you are a developer new to containers or an experienced DevOps engineer looking for advanced techniques, this book provides the complete knowledge path from first concepts to production-grade mastery.

Introduction: The Container Revolution

In March 2013, at the PyCon conference in Santa Clara, California, a French startup called dotCloud demonstrated a tool that would quietly reshape the entire software industry. Solomon Hykes, the company's founder, showed an audience how to package an application with all its dependencies into a single, portable unit that could run anywhere. He called it Docker. Nobody in that room could have predicted that within a decade, containers would become the foundational building block of modern cloud computing, powering everything from microservices at Netflix to the Kubernetes clusters running billions of requests per day.

This book is about understanding and mastering that technology.

What This Book Covers

Docker has grown from a simple container runtime into an entire ecosystem of tools, standards, and practices. This book covers the full spectrum:

Foundation. You will start with the fundamentals: what containers are, how they differ from virtual machines, and the Linux kernel features that make them possible. You will install Docker on your machine and run your first container.

Core Mechanics. The heart of the book dives into Docker's core concepts: images, containers, layers, registries, networking, storage, and the Docker Engine architecture. You will learn to write optimized Dockerfiles, build multi-stage images, manage volumes, and configure complex networks.

Advanced Operations. Once you understand the basics, you will move into advanced territory: security hardening, performance tuning, logging and monitoring, debugging and troubleshooting, secrets management, and CI/CD integration.

Orchestration and Production. The final chapters cover Docker Swarm, Kubernetes integration, real-world production workflows, and emerging features in the Docker ecosystem. You will see how all the pieces fit together in a complete end-to-end project.

Who This Book Is For

This book is written for three audiences:

- **Developers new to containers** who want a clear, practical introduction to Docker without getting lost in theory.
- **Experienced developers and DevOps engineers** who use Docker daily but want a deeper understanding of architecture, security, and performance.
- **IT professionals and system administrators** transitioning into container-based infrastructure who need both conceptual clarity and operational competence.

No prior container experience is required. Familiarity with the command line, basic Linux concepts, and general software development practices will help, but every concept is explained from first principles.

How to Use This Book

The chapters are designed to be read sequentially. Each chapter builds on concepts introduced earlier, so jumping around may leave gaps in your understanding. That said, once you have completed the first six chapters (the foundation and core mechanics), you can treat later chapters as reference material for specific topics.

Each chapter follows a consistent structure:

- **Conceptual explanation** of what the topic is and why it matters
- **Practical examples** with annotated command-line snippets
- **Best practices** drawn from production experience
- **Common pitfalls** to avoid
- **Hands-on exercises** to reinforce learning
- **Chapter summary** with key takeaways

Prerequisites and Setup

To follow along with the practical examples in this book, you will need:

- A computer running Linux, macOS, or Windows 10/11
- At least 4 GB of RAM (8 GB recommended for running multi-container applications)
- Basic comfort with the command line (bash or PowerShell)
- A text editor for writing Dockerfiles and Compose files

Chapter 2 walks you through installing Docker on each major operating system. By the end of that chapter, you will have a working Docker installation and will have run your first container.

The Journey Ahead

Over the next twenty chapters, you will go from running `docker run hello-world` to designing production-grade container architectures with automated CI/CD pipelines, security scanning, multi-node orchestration, and observability. Along the way, you will understand not just how to use Docker, but why it works the way it does. That deeper understanding is what separates practitioners who can follow tutorials from engineers who can design, debug, and optimize containerized systems under pressure.

Let us begin.

Chapter 1: The Container Revolution – Why Docker Changed Everything

The Pre-Container Era: Virtual Machines and Their Limits

Before containers, the dominant approach to application isolation was the virtual machine. A VM uses a hypervisor to emulate an entire physical computer: CPU, memory, disk, network interface. Each VM runs its own full operating system kernel. This provides strong isolation but at a steep cost.

Consider a typical web application stack in 2012: an Nginx reverse proxy, a Node.js application server, a PostgreSQL database, and a Redis cache. To isolate each component, you would spin up four separate virtual machines. Each VM needed its own guest OS (typically several gigabytes), plus the application itself. On a server with 32 GB of RAM, you might fit two or three such stacks before running out of memory – not because the applications needed it, but because each guest OS consumed 1-2 GB just sitting idle.

VMs were also slow to start. Booting a full operating system takes seconds to minutes, making rapid scaling impractical. Deploying updates meant restarting entire VMs with all their overhead.

The fundamental problem was granularity. Virtual machines virtualized hardware, but most applications did not need separate hardware – they needed isolated processes that shared the same kernel.

What Is a Container? Linux Primitives Explained

A container is an isolated user space on top of a shared operating system kernel. Unlike a VM, which emulates hardware and runs its own kernel, a container shares the host's kernel while using two key Linux features to create isolation:

Namespaces provide isolation. When a process runs inside a namespace, it sees only the resources allocated to that namespace. Linux provides several namespace types:

- **PID namespace:** Isolates the process table. A process in a PID namespace sees only its own children, making it appear as PID 1 (init) inside the container.
- **Network namespace:** Gives each container its own network stack – interfaces, routes, firewall rules. Containers get their own IP addresses and ports.
- **Mount namespace:** Isolates the filesystem view. Each container sees its own directory tree, independent of the host and other containers.
- **UTS namespace:** Allows containers to have their own hostname and domain name.
- **IPC namespace:** Isolates inter-process communication mechanisms like shared memory and message queues.
- **User namespace:** Maps user IDs inside the container to different IDs on the host, allowing a process to be root inside the container while running as an unprivileged user on the host.

Control groups (cgroups) provide resource limits. While namespaces hide resources from containers, cgroups restrict how much of those resources a container can consume. Cgroups control CPU time, memory usage, block I/O bandwidth, and network bandwidth. Without cgroups, a single container could starve the entire host by consuming all available CPU or RAM.

Together, namespaces and cgroups create the illusion that an application has its own private machine, when in reality it is just a set of processes running on the shared kernel with carefully controlled visibility and resource allocation.

Docker vs. Virtual Machines: A Fundamental Comparison

The difference between containers and virtual machines is architectural, not merely quantitative. Understanding this distinction is essential for choosing the right tool for each job.

Aspect	Virtual Machine	Docker Container
Isolation level	Hardware-level (hypervisor)	OS-level (kernel namespaces)
Guest OS	Full operating system per VM	Shares host kernel
Image size	Gigabytes (full OS + app)	Megabytes (app + dependencies)
Startup time	Seconds to minutes	Milliseconds to seconds
Density	2-10 VMs per physical server	10-100+ containers per server
Performance overhead	Significant (hardware emulation)	Minimal (near-native)
Cross-OS support	Run Windows on Linux, etc.	Same OS family required
Security isolation	Stronger (separate kernels)	Good, but shared kernel is a risk
Use case	Legacy apps, different OSes, strong isolation needs	Microservices, CI/CD, rapid scaling

The key insight: containers are not lighter VMs. They are something fundamentally different. A VM gives you a complete computer. A container gives you an isolated process environment. This makes containers ideal for running applications but unsuitable when you need different operating systems (e.g., running Windows software on a Linux host) or maximum security isolation between workloads.

In practice, the two technologies complement each other. Many production environments run Docker containers inside virtual machines, getting the best of both worlds: VM-level isolation between tenants and container-level density for applications within each tenant.

The Birth of Docker: From dotCloud to Open Source

The story of Docker is a textbook example of how solving your own problems can create an industry-defining product.

dotCloud was founded in 2010 by Solomon Hykes in Paris. It was a Platform-as-a-Service (PaaS) provider – essentially a competitor to Heroku. The company offered multi-language web hosting, allowing developers to deploy applications without managing infrastructure.

Internally, dotCloud faced the same problems every PaaS faces: how to isolate customer applications, manage resources efficiently, and deploy updates quickly. The team experimented with various Linux virtualization technologies, including Virtuozzo containers and OpenVZ. Eventually, they settled on Linux Containers (LXC), a tool created by IBM engineers around 2008 that provided a user-friendly interface to the kernel's namespace and cgroup features.

But LXC was not developer-friendly. It required deep knowledge of Linux internals, configuration files were complex, and there was no simple way to package an application with its dependencies for distribution.

So dotCloud built their own abstraction layer on top of LXC. They created a Python CLI tool called `dc` that made container management simpler. Over time, this evolved into a more sophisticated system with image packaging, dependency management, and a concept they called "Dockers" – self-contained application bundles.

In March 2013, after testing internally for months, dotCloud released Docker as open source at PyCon. The response was explosive. Developers immediately recognized what LXC had failed to deliver: a container technology that felt natural to software developers, not system administrators.

By September 2013, dotCloud rebranded itself as Docker, Inc. The company raised $400 million in funding and became one of the most valuable startups in Silicon Valley. More importantly, the container concept took root in the broader technology community, eventually leading to the Open Container Initiative (OCI) in 2015, Kubernetes in 2014, and the cloud-native revolution that followed.

Why Containers Won: Speed, Density, and Portability

Docker did not invent containers. Solaris Zones existed since 2004. FreeBSD jails predated them. LXC had been around since 2008. What Docker did was make containers accessible to software developers.

Three factors explain why containers won:

Speed. Containers start in milliseconds because they are just processes, not booting operating systems. This enables rapid scaling – spinning up dozens of instances in seconds to handle traffic spikes. It also makes CI/CD pipelines dramatically faster. A test suite that needed a fresh VM for each run (taking 30+ seconds per VM) can instead spin up containers in under a second.

Density. Because containers share the host kernel and do not carry the overhead of guest operating systems, you can pack many more containers onto a single physical machine than virtual machines. A server that runs four VMs might comfortably run fifty containers. This density translates directly to cost savings at scale.

Portability. The Docker image format packages an application with all its dependencies – libraries, configuration files, runtime – into a single unit. That image runs identically on any machine with Docker installed: a developer's laptop, a staging server, a production cluster in AWS, Azure, or Google Cloud. The famous phrase "it works on my machine" becomes irrelevant because the machine is packaged inside the image.

These three qualities – speed, density, and portability – aligned perfectly with the industry's move toward microservices architectures and cloud-native development. Containers became the natural packaging format for small, independently deployable services that need to scale rapidly and run consistently across heterogeneous infrastructure.

Chapter Summary and Key Takeaways

- **Containers are not lightweight VMs.** They use OS-level virtualization (namespaces and cgroups) rather than hardware emulation. This makes them faster to start, denser to pack, and more portable, but they share the host kernel.

- **Namespaces provide isolation** (PID, network, mount, UTS, IPC, user) while **cgroups enforce resource limits** (CPU, memory, I/O).
- **Docker succeeded by making containers developer-friendly.** LXC existed first but was too complex for most developers. Docker's simple CLI and image format democratized containerization.
- **The three pillars of container adoption are speed, density, and portability.** These qualities make containers ideal for microservices, CI/CD, and cloud-native architectures.
- **Containers and VMs complement each other.** Many production environments run containers inside VMs for layered isolation.

Hands-on exercise: Before moving to the next chapter, think about your current development or deployment workflow. Where do you encounter the "it works on my machine" problem? How long does it take to spin up a new environment for testing? Write down three pain points that containers might solve in your context.

Chapter 2: Installing Docker Across Platforms

Docker Desktop: The Easiest Path for Developers

For most developers working on macOS or Windows, Docker Desktop is the simplest way to get started. It bundles everything you need: the Docker Engine, Docker CLI, Docker Compose, Kubernetes integration (optional), and a graphical dashboard.

Docker Desktop system requirements:

- **macOS:** Apple Silicon (M1/M2/M3) or Intel-based Mac running macOS 12 Monterey through macOS 15 Sequoia. At least 4 GB of RAM (8 GB recommended).
- **Windows:** Windows 10 64-bit (version 2004 or higher) or Windows 11. Requires WSL 2 backend on Windows 10/11 Home, or Hyper-V + WSL 2 on Pro/Enterprise editions. At least 4 GB of RAM.
- **Linux:** Various distributions supported including Ubuntu, Debian, Fedora, SUSE, and CentOS.

Installing Docker Desktop on macOS:

Download the `.dmg` file from the Docker website, drag the Docker icon to your Applications folder, and launch it. Docker Desktop runs as a menu bar application and manages a lightweight Linux VM in the background (using Virtualization.framework on Apple Silicon or HyperKit on Intel).

Installing Docker Desktop on Windows:

Download the `.exe` installer and run it. During installation, you will be prompted to install the WSL 2 backend and a Ubuntu distribution if they are not already present. After installation, restart your computer. Docker Desktop uses WSL 2 to run Linux containers natively, providing near-native performance without the overhead of a full virtual machine.

Licensing note: Docker Desktop is free for personal use, small businesses (fewer than 250 employees AND less than $10 million annual revenue), and educational use. Organizations exceeding these thresholds need a paid subscription. If you need a free alternative on Linux, install Docker Engine directly (covered below).

Installing Docker on Linux (Ubuntu, Debian, CentOS, Fedora)

On Linux, you have two main options: Docker Desktop (convenient, with GUI) or Docker Engine (lightweight, no overhead). For production servers and most Linux workstations, Docker Engine is the preferred choice.

Installing Docker Engine on Ubuntu 22.04/24.04:

The recommended approach is to use Docker's official APT repository rather than the distribution's built-in packages, which tend to lag behind:

```
# Step 1: Remove any conflicting packages
sudo apt-get remove docker docker-engine docker.io containerd runc

# Step 2: Install prerequisites
sudo apt-get update
sudo apt-get install -y ca-certificates curl gnupg

# Step 3: Add Docker's official GPG key
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
↪  /etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg

# Step 4: Add the Docker repository
echo \
  "deb [arch=$(dpkg --print-architecture)
  ↪  signed-by=/etc/apt/keyrings/docker.gpg]
  ↪  https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

# Step 5: Install Docker Engine
sudo apt-get update
sudo apt-get install -y docker-ce docker-ce-cli containerd.io
↪  docker-buildx-plugin docker-compose-plugin

```

```
# Step 6: Verify installation
docker run hello-world
```

The `hello-world` image is Docker's equivalent of "Hello, World!" in programming. When you run it, Docker pulls the image from Docker Hub (if not cached locally), creates a container, runs the program inside it, and prints a confirmation message.

Installing on Debian:

The process is nearly identical to Ubuntu. Replace `ubuntu` with `debian` in the GPG key URL and repository path, and use `$(lsb_release -cs)` instead of `$VERSION_CODENAME`.

Installing on CentOS/RHEL 8+:

```
# Add Docker repository
sudo dnf install -y dnf-plugins-core
sudo dnf config-manager --add-repo
↪  https://download.docker.com/linux/centos/docker-ce.repo

# Install Docker Engine
sudo dnf install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin
↪  docker-compose-plugin

# Start and enable Docker
sudo systemctl start docker
sudo systemctl enable docker
```

Installing on Fedora:

```
sudo dnf install -y dnf-plugins-core
sudo dnf config-manager --add-repo
↪  https://download.docker.com/linux/fedora/docker-ce.repo
sudo dnf install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin
↪  docker-compose-plugin
sudo systemctl start docker
sudo systemctl enable docker
```

Post-installation: Running Docker without sudo

By default, the Docker daemon binds to a Unix socket owned by the root user. To run Docker commands without `sudo`, add your user to the `docker` group:

```
sudo groupadd docker          # Usually already exists after installation
sudo usermod -aG docker $USER
newgrp docker                 # Apply group membership in current session
docker ps                     # Test -- no sudo needed
```

Security note: Being in the `docker` group is effectively equivalent to having root access, because Docker containers can mount the host filesystem. Only add trusted users to this group.

Docker on Windows: WSL2 and Hyper-V Backends

Docker Desktop on Windows supports two backend architectures for running Linux containers:

WSL 2 Backend (recommended):

WSL 2 runs a real Linux kernel inside a lightweight utility VM managed by the Windows hypervisor. Docker integrates with WSL 2 to run containers directly inside your Ubuntu distribution, providing excellent file I/O performance and native Linux tooling.

To set up WSL 2:

```
# Run in PowerShell as Administrator
wsl --install
# Restart your computer when prompted
# Set WSL 2 as the default version
wsl --set-default-version 2
```

After installing Docker Desktop, go to Settings > General and ensure "Use the WSL 2 based engine" is checked.

Hyper-V Backend (legacy):

The older Hyper-V backend creates a full virtual machine running Linux. It works but has significant file I/O overhead when accessing files from the Windows filesystem. Microsoft and Docker both recommend WSL 2 for new installations.

Docker Engine vs. Docker Desktop: Understanding the Difference

Understanding what you are installing matters, especially when troubleshooting or making architectural decisions.

Component	Docker Engine	Docker Desktop
Docker daemon (dockerd)	Yes	Yes
Docker CLI	Yes	Yes
Containerd	Yes	Yes
runc	Yes	Yes
Docker Compose	Plugin (separate install)	Bundled
Buildx	Plugin (separate install)	Bundled
Kubernetes cluster	No	Optional, built-in
GUI Dashboard	No	Yes
Resource management CLI	Manual (systemd)	Sliders in settings
Linux VM on Mac/Windows	N/A (runs natively on Linux)	Required (Virtualization.framework / WSL2)
Cost	Free (Apache 2.0)	Free for personal/small business; paid for enterprise

Docker Engine is the open-source container runtime. It includes the daemon, CLI, and core components. Docker Desktop wraps Docker Engine with additional features: a graphical dashboard, automatic updates, Kubernetes integration, and on macOS/Windows, a Linux VM to host the containers.

On Linux, Docker Engine runs natively on the host kernel. On macOS and Windows, Docker Desktop must run a Linux VM because containers require a Linux kernel. This is why Docker Desktop has higher resource requirements on those platforms – it is running an entire Linux virtual machine in addition to your containers.

Verifying Your Installation and Running Your First Container

After installation, verify everything is working:

```
# Check Docker version and components
docker version

# Expected output shows Client (CLI) and Server (daemon) versions
# Both should report the same major version number

# Get detailed system information
docker info

# This shows:
# - Storage driver in use (typically overlay2 on Linux)
# - Number of containers, images
# - Operating system details
# - Cgroup driver and version

# Run your first real container
docker run --name my-nginx -d -p 8080:80 nginx:latest

# Visit http://localhost:8080 in your browser
# You should see the Nginx welcome page

# Check that the container is running
docker ps

# See container logs
docker logs my-nginx

# Stop and remove the container
docker stop my-nginx
docker rm my-nginx
```

Let us unpack that `docker run` command, because it contains several concepts you will use throughout this book:

- `--name my-nginx`: Assigns a human-readable name to the container
- `-d`: Runs the container in detached mode (in the background)
- `-p 8080:80`: Maps port 80 inside the container to port 8080 on your host
- `nginx:latest`: Specifies the image to use (image name and tag)

Common Installation Pitfalls and Fixes

Problem: "Cannot connect to the Docker daemon"

This usually means the Docker daemon is not running.

```
# On Linux, start the daemon
sudo systemctl start docker
sudo systemctl status docker

# On macOS/Windows, check that Docker Desktop is running
# Look for the whale icon in your menu bar / system tray
```

Problem: "Got permission denied while trying to connect to the Docker daemon socket"

Your user is not in the `docker` group.

```
sudo usermod -aG docker $USER
# Log out and log back in, or run: newgrp docker
```

Problem: Port already in use

```
# Check what is using port 8080
sudo lsof -i :8080
# Or on Windows:
netstat -ano | findstr :8080
```

Problem: Docker Desktop on Mac consuming too much RAM

Go to Docker Desktop Settings > Resources and reduce the memory allocation. The default is often generous (4 GB or more). For most development workloads, 2 GB is sufficient.

Problem: Slow file I/O on macOS/Windows

This is a known limitation of sharing files between the host OS and the Linux VM. To mitigate:

- Store project files inside the WSL 2 filesystem (on Windows) rather than in `/mnt/c/`
- Use Docker volumes instead of bind mounts for large directories
- Exclude unnecessary directories from Docker's file sharing using `.dockerignore` or Docker Desktop settings

Chapter Summary and Key Takeaways

- **Docker Desktop** is the easiest path for macOS and Windows developers. It bundles everything needed and provides a GUI dashboard.
- **Docker Engine** on Linux runs natively without a VM overhead. Install from Docker's official APT/YUM repository, not the distribution's packages.
- **WSL 2** is the recommended backend for Docker on Windows, providing near-native performance.
- **Adding your user to the `docker` group** eliminates the need for `sudo`, but remember this grants effective root access.
- **The `docker run hello-world` command** is your first verification step, and `docker version` / `docker info` provide detailed system diagnostics.
- **File I/O performance** on macOS and Windows depends on where your files are stored. On Windows, use the WSL 2 filesystem for best results.

Hands-on exercise: Install Docker on your machine following the steps above. Run `docker run hello-world`, then run an Nginx container with port mapping. Visit the Nginx page in your browser. Finally, stop and remove the container. Take note of any errors or unexpected behavior – you will encounter and resolve these patterns throughout the book.

Chapter 3: Docker Architecture Under the Hood

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

The Client-Server Architecture: Docker CLI and Docker Daemon

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Containerd and runc: The Runtime Stack

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

How cgroups Isolate Resources

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

How Namespaces Provide Isolation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Union File Systems and Layered Storage Drivers

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

The Docker API: RESTful Communication with the Daemon

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 4: Core Concepts – Images, Containers, and Layers

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Docker Images: Immutable, Layered Blueprints

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

How Image Layers Work: The Union File System in Practice

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Running Containers: From Image to Process

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

The Container Lifecycle: Create, Start, Stop, Remove

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Inspecting Images and Containers with the CLI

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Image Tags, Digests, and Versioning Strategies

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 5: Docker Registries – Publishing and Distributing Images

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Docker Hub: The Default Public Registry

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Pulling, Tagging, and Pushing Images

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Understanding Image Tags and the Latest Tag Problem

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Private Registries: Docker Trusted Registry and Alternatives

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Registry Security: Scanning, Signing, and Notary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Managing Registry Storage and Cleanup Policies

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 6: Writing Dockerfiles – Building Custom Images

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Dockerfile Anatomy: Instructions and Syntax

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

FROM, RUN, COPY, and ADD: The Core Instructions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

CMD vs ENTRYPOINT: Understanding Execution Models

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

ENV, ARG, and EXPOSE: Configuration and Documentation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

WORKDIR, USER, HEALTHCHECK, and LABEL

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Best Practices for Dockerfile Authoring

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Common Dockerfile Anti-Patterns and How to Avoid Them

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 7: Multi-Stage Builds and Image Optimization

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

The Problem: Bloated Docker Images

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Multi-Stage Builds: Concept and Syntax

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Real-World Multi-Stage Build Examples (Node.js, Go, Python)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Distroless and Minimal Base Images

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Image Size Optimization Techniques

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Scanning and Analyzing Image Contents

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 8: Container Networking Deep Dive

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Docker Network Drivers: Bridge, Host, None, and Overlay

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

The Default Bridge Network and Its Limitations

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Creating Custom Bridge Networks for Service Discovery

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Port Mapping: Published vs Exposed Ports

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Docker DNS and Container-to-Container Communication

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Overlay Networks for Multi-Host Communication

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Macvlan and IPvlan: Advanced Networking Drivers

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 9: Persistent Storage – Volumes, Bind Mounts, and tmpfs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

The Ephemeral Container Filesystem Problem

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Docker Volumes: Managed, Persistent Storage

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Bind Mounts: Bridging Host and Container Filesystems

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

tmpfs Mounts: In-Memory, Ephemeral Storage

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Volume Drivers and Third-Party Storage Solutions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Backup, Restore, and Migration Strategies for Container Data

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 10: Environment Variables, Secrets, and Configuration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Environment Variables: -e, —env-file, and Dockerfile ENV

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

The Twelve-Factor App Configuration Philosophy

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Docker Secrets: Secure Credential Management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Config Files in Swarm Mode

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

.env Files and Docker Compose Variable Substitution

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Security Implications of Storing Sensitive Data in Images

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 11: Docker Compose – Multi-Container Applications

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

What Is Docker Compose and When to Use It

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

The docker-compose.yml File: Structure and Syntax

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Services, Networks, and Volumes in Compose

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Compose Commands: Up, Down, Logs, Exec, and More

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Compose Profiles and Selective Service Management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Multi-File Compose and Environment Overrides

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Real-World Compose Project: A Full-Stack Application

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 12: Logging, Monitoring, and Observability

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Docker Logging Drivers: json-file, syslog, journald, and More

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Collecting Container Logs with docker logs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Log Rotation and Storage Management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Container Health Checks: HEALTHCHECK Instruction

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Docker Stats and Resource Monitoring

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Integrating with External Monitoring (Prometheus, Datadog, ELK)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 13: Docker Security Best Practices

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

The Container Security Threat Model

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Image Security: Scanning for Vulnerabilities

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Running Containers as Non-Root Users

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Read-Only Filesystems and Minimal Capabilities

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Seccomp Profiles and AppArmor/SELinux Integration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Securing the Docker Daemon and API

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Supply Chain Security: SBOM, Signing, and Provenance

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 14: Performance Tuning and Resource Management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

CPU Limits and Shares: Controlling Compute Resources

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Memory Limits, Swapping, and OOM Handling

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

I/O Bandwidth and Block Device Weighting

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Pids Limit and Kernel Resource Constraints

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Container Density: Packing More Containers per Host

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Performance Profiling Tools and Techniques

Chapter Summary and Key Takeaways

Chapter 15: Debugging and Troubleshooting Docker

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Common Docker Error Messages and Their Meanings

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Inspecting Container Health with docker inspect

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Executing into Running Containers for Live Debugging

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Analyzing Container Logs and Daemon Logs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Network Troubleshooting: Connectivity Between Containers

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Storage Troubleshooting: Disk Space and Volume Issues

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Docker Events and Real-Time Diagnostics

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 16: CI/CD Integration with Docker

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Building Docker Images in CI Pipelines

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Caching Strategies for Faster Builds

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Automated Testing with Docker Compose

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Image Scanning as a Pipeline Gate

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Pushing to Registries from CI/CD

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Real-World CI/CD Pipeline Examples (GitHub Actions, GitLab CI, Jenkins)

Chapter Summary and Key Takeaways

Chapter 17: Container Orchestration Fundamentals

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Why Single-Host Docker Is Not Enough for Production

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Docker Swarm: Built-In Orchestration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Swarm Mode: Managers, Workers, and Services

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Deploying Stacks with Compose Files in Swarm

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Service Discovery, Load Balancing, and Rolling Updates

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Swarm vs Kubernetes: Choosing the Right Tool

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 18: Docker and Kubernetes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Kubernetes Architecture at a Glance

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Pods, Containers, and the Relationship to Docker

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Deploying Docker Images on Kubernetes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Kubernetes Services, ConfigMaps, and Secrets

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Helm Charts and Application Packaging

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Running Minikube and Kind for Local Development

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

From Docker Compose to Kubernetes Manifests

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 19: Real-World Production Workflows

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

The Modern Containerized Application Architecture

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Development Workflow: Local Docker Compose Setup

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Staging and Testing Environments

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Production Deployment Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Disaster Recovery and Backup Strategies

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Case Study: Containerizing a Full-Stack Web Application

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter 20: Emerging Features and the Docker Ecosystem

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Docker BuildKit: The Next-Generation Build Engine

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Docker Compose V2: Plugin Architecture and Improvements

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Docker Desktop Extensions and Marketplace

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Container Image Formats: OCI, Docker Manifest, and Distribution Specs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

eBPF and Container Observability

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

The Future of Containers: Wasm, Serverless, and Edge Computing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Chapter Summary and Key Takeaways

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Conclusion: The Container Journey Ahead

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Glossary of Docker Terms

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

Interview Questions for Docker Roles

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.

References

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dockercontainersfrombeginnertomastery.