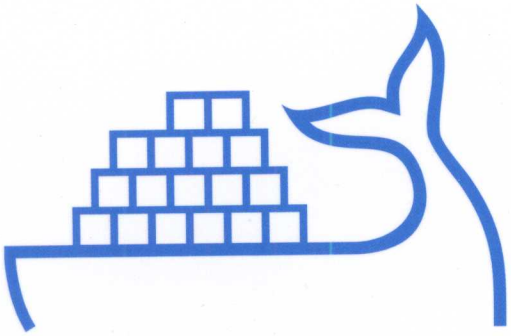
HZ BOOKS
华章IT

容器技术系列



Docker
技术入门与实战

第2版

DOCKER PRIMER

杨保华 戴王剑 曹亚仑 编著

学习Docker的第一本入门书，畅销书升级
基于Docker 1.12及以上版本

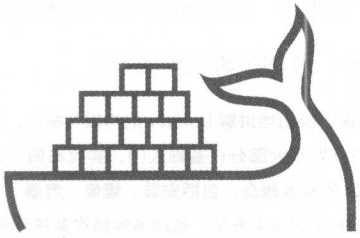
机械工业出版社
China Machine Press

作者简介

杨保华 博士，毕业于清华大学，现为 IBM 资深研究员。主要负责核心系统方案的架构设计和研发，包括云计算、大数据、金融科技等领域。他热爱开源文化，是容器、软件定义网络、区块链等开源技术的早期推广者，曾为 OpenStack、HyperLedger 等开源项目作出过贡献。个人主页为 https://yeasy.github.com。

戴王剑 资深架构师，多年来一直从事系统平台、计算机网络、服务器架构设计，负责过多个省级项目的架构设计。热衷于开源事业，积极推动开源技术在生产实践中的应用。

曹亚仑 阿里云高级系统工程师（花名法喜），擅长云产品运维与云平台技术保障，对PaaS、IaaS层架构设计与实践有较丰富的实战经验，同时也是DevOps实践者与全栈开发者。微信 allengaller，个人主页为 allengaller.github.io。



Docker
技术入门与实战

第2版

DOCKER PRIMER

杨保华 戴王剑 曹亚仑 编著

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

Docker 技术入门与实战 / 杨保华，戴王剑，曹亚仑编著．—2 版．—北京：机械工业出版社，2017.1（2017.7 重印）
（容器技术系列）

ISBN 978-7-111-55582-7

I. D… II. ①杨… ②戴… ③曹… III. Linux 操作系统－程序设计 IV. TP316.85

中国版本图书馆 CIP 数据核字（2016）第 308604 号

本书从 Docker 基本原理开始，深入浅出地讲解 Docker 的构建与操作，内容系统全面，可帮助开发人员、运维人员快速部署 Docker 应用。本书分为四大部分：基础入门、实战案例、进阶技能和开源项目。第一部分（第 1 ~ 8 章）介绍 Docker 与虚拟化技术的基本概念，包括安装、镜像、容器、仓库、数据卷，端口映射等；第二部分（第 9 ~ 16 章）通过案例介绍 Docker 的应用方法，包括各种操作系统平台、SSH 服务的镜像、Web 服务器与应用、数据库的应用、各类编程语言的接口、容器云等，还介绍了作者在容器实战中的思考与经验总结；第三部分（第 17 ~ 21 章）介绍一些进阶技能，如 Docker 核心技术实现原理、安全、高级网络配置、libernetwork 插件化网络功能等；第四部分（第 22 ~ 28 章）介绍与容器开发相关的开源项目，包括 Etcd、Docker Machine、Docker Compose、Docker Swarm、Mesos 和 Kubernetes 等。

第 2 版参照 Docker 技术的最新进展对全书内容进行了修订，并增加了第四部分专门介绍与容器相关的知名开源项目，利用好这些优秀的开源平台，可以更好地在生产实践中受益。

Docker 技术入门与实战（第 2 版）

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号　邮政编码：100037）
责任编辑：吴　怡　　责任校对：董纪丽
印　　刷：北京市荣盛彩色印刷有限公司　　版　　次：2017 年 7 月第 2 版第 3 次印刷
开　　本：186mm × 240mm　1/16　　印　　张：25.5
书　　号：ISBN 978-7-111-55582-7　　定　　价：69.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88379426　88361066　　投稿热线：（010）88379604
购书热线：（010）68326294　88379649　68995259　　读者信箱：hzit@hzbook.com

Preface 第2版前言

自云计算步入市场算起，新一代计算技术刚好走过了第一个十年。

在过去十年里，围绕计算、存储、网络三大基础服务，围绕敏捷服务和规模处理两大核心诉求，新的概念、模式和工具争相涌现。这些创新的开源技术成果，提高了整个信息产业的生产效率，降低了应用信息技术的门槛，让“互联网+”成为可能。

如果说软件定义网络（SDN）和网络功能虚拟化（NFV）让互联网络的虚拟化进入了崭新的阶段，那么容器技术的出现，毫无疑问称得上计算虚拟化技术的又一大创新。从 Linux Container 到 Docker，看似是计算技术发展的一小步，却是极为重要的历史性突破。容器带来的不仅仅是技术体验上的改进，更多的是新的开发模式、新的应用场景、新的业务可能……

容器技术自身在快速演进的同时，笔者也很欣喜地看到，围绕着容器的开源生态系统越发繁盛。Docker 三剑客 Machine、Compose、Swarm 相辅相成，集团作战；搜索巨人则推出 Kubernetes，领航新一代容器化应用集群平台；还有 Mesos、CoreOS，以及其他众多的开源工具。这些工具的出现，弥补了现有容器技术栈的不足，极大地丰富了容器技术的应用场景，增强了容器技术在更多领域的竞争力。

在第2版中，笔者参照容器技术最新进展对全书内容进行了修订完善，并增加了第四部分专门介绍与容器相关的知名开源项目，利用好这些优秀的开源平台，可以更好地在生产实践中受益。成书之际，Docker 发布了 1.13 版本，带来了更稳定的性能和更多有趣的特性。

再次感谢容器技术，感谢开源文化，希望开源技术能得到更多的支持和贡献！

最后，IBM 中国研究院的刘天成、李玉博等帮忙审阅了部分内容，在此表达最深厚的感谢！

杨保华

2016 年 12 月于北京

第 1 版前言 *Preface*

在一台服务器上同时运行一百个虚拟机，肯定会被认为是痴人说梦。而在一台服务器上同时运行一千个 Docker 容器，这已经成为现实。在计算机技术高速发展的今天，昔日的天方夜谭正在一个个变成现实。

多年的研发和运维（DevOps）经历中，笔者时常会碰到这样一个困境：用户的需求越来越多样，系统的规模越来越庞大，运行的软件越来越复杂，环境配置问题所造成的麻烦层出不穷……为了解决这些问题，开源社区推出过不少优秀的工具。这些方案虽然在某些程度上确能解决部分“燃眉之急”，但是始终没有一种方案能带来“一劳永逸”的效果。

让作为企业最核心资源的工程师们花费大量的时间，去解决各种环境和配置引发的 Bug，这真的正常吗？

回顾计算机的发展历程，最初，程序设计人员需要直接操作各种枯燥的机器指令，编程效率之低可想而知。高级语言的诞生，将机器指令的具体实现成功抽象出来，从此揭开了计算机编程效率突飞猛进的大时代。那么，为什么不能把类似的理念（抽象与分层）也引入到现代的研发和运维领域呢？

Docker 无疑在这一方向上迈出了具有革新意义的一步。笔者在刚接触 Docker 时，就为它所能带来的敏捷工作流程而深深吸引，也为它能充分挖掘云计算资源的效能而兴奋不已。我们深信，Docker 的出现，必将给 DevOps 技术，甚至整个信息技术产业的发展带来深远的影响。

笔者曾尝试编写了介绍 Docker 技术的中文开源文档。短短一个月的时间，竟收到了来自全球各个地区超过 20 万次的阅读量和全五星的好评。这让我们看到国内技术界对于新兴开源技术的敏锐嗅觉和迫切需求，同时也倍感压力，生怕其中有不妥之处，影响了大家学习和推广 Docker 技术的热情。在开源文档撰写过程中，我们一直在不断思考，在生产实践中到底怎么用 Docker 才是合理的？在“华章图书”的帮助下，终于有了现在读者手中的这本书。

与很多技术类书籍不同，本书中避免一上来就讲述冗长的故事，而是试图深入浅出、直奔主题，在最短时间内让读者理解和掌握最关键的技术点，并且配合实际操作案例和精炼的点评，给读者提供真正可以上手的实战指南。

本书在结构上分为三大部分。第一部分是 Docker 技术的基础知识介绍，这部分将让读者对 Docker 技术能做什么有个全局的认识；第二部分将具体讲解各种典型场景的应用案例，供读者体会 Docker 在实际应用中的高效秘诀；第三部分将讨论一些偏技术环节的高级话题，试图让读者理解 Docker 在设计上的工程美学。最后的附录归纳了应用 Docker 的常见问题和一些常用的参考资料。读者可根据自身需求选择阅读重点。全书主要由杨保华和戴王剑主笔，曹亚仑写作了编程开发和实践之道章节。

本书在写作过程中参考了官方网站上的部分文档，并得到了 DockerPool 技术社区网友们的积极反馈和支持，在此一并感谢！

成稿之际，Docker 已经发布了增强安全特性的 1.3.2 版本。衷心祝愿 Docker 及相关技术能够快速成长和成熟，让众多 IT 从业人员的工作和生活都更加健康、更加美好！

作者于 2014 年 11 月

目　录 *Contents*

第一部分 Part 1

基础入门

- 第 1 章　初识容器与 Docker
- 第 2 章　核心概念与安装配置
- 第 3 章　使用 Docker 镜像
- 第 4 章　操作 Docker 容器
- 第 5 章　访问 Docker 仓库
- 第 6 章　Docker 数据管理
- 第 7 章　端口映射与容器互联
- 第 8 章　使用 Dockerfile 创建镜像

本部分共有 8 章内容，笔者将介绍 Docker 和容器的相关基础知识。

第 1 章介绍 Docker 的前世与今生，以及它与现有的虚拟化技术，特别是 Linux 容器技术的关系。

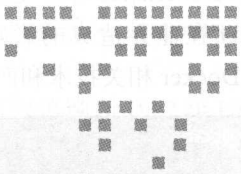
第 2 章介绍 Docker 的三大核心概念，以及如何在常见的操作系统环境中安装 Docker。

第 3 章到第 5 章通过具体的示例，讲解使用 Docker 的常见操作，包括镜像、容器和仓库。

第 6 章剖析如何在 Docker 中使用数据卷来保存持久化数据。

第 7 章介绍如何使用端口映射和容器互联来方便外部对容器服务的访问。

第 8 章介绍如何编写 Dockerfile 配置文件，以及使用 Dockerfile 来创建镜像的具体方法和注意事项。



第 1 章 Chapter 1

初识容器与 Docker

如果说主机时代大家比拼的是单个服务器物理性能（如 CPU 主频和内存），那么在云时代，最为看重的则是凭借虚拟化技术所构建的集群处理能力。

伴随着信息技术的飞速发展，虚拟化技术早已经广泛应用到各种关键场景中。从 20 世纪 60 年代 IBM 推出的大型主机虚拟化，到后来以 Xen、KVM 为代表的虚拟机虚拟化，再到现在以 Docker 为代表的容器技术，虚拟化技术自身也在不断进行创新和突破。

传统来看，虚拟化既可以通过硬件模拟来实现，也可以通过操作系统软件来实现。而容器技术则更为优雅，它充分利用了操作系统本身已有的机制和特性，可以实现远超传统虚拟机的轻量级虚拟化。因此，有人甚至把它称为“新一代的虚拟化”技术，并将基于容器打造的云平台亲切地称为“容器云”。

Docker 毫无疑问正是众多容器技术中的佼佼者，是容器技术发展过程中耀眼的一抹亮色。那么，什么是 Docker？它会带来哪些好处？它跟现有虚拟化技术又有何关系？

本章首先会介绍 Docker 项目的起源和发展过程，之后会为大家剖析 Docker 和相关容器技术，以及它在 DevOps 等场景带来的巨大便利。最后，还将阐述 Docker 在整个虚拟化领域中的技术定位。

1.1 什么是 Docker

1. Docker 开源项目背景

Docker 是基于 Go 语言实现的开源容器项目，诞生于 2013 年年初，最初发起者是 dotCloud 公司。Docker 自开源后受到广泛的关注和讨论，目前已有多个相关项目（包括

Docker 三剑客、Kubernetes 等），逐渐形成了围绕 Docker 容器的生态体系。

由于 Docker 在业界造成的影响力实在太大，dotCloud 公司后来也直接改名为 Docker Inc，并专注于 Docker 相关技术和产品的开发。

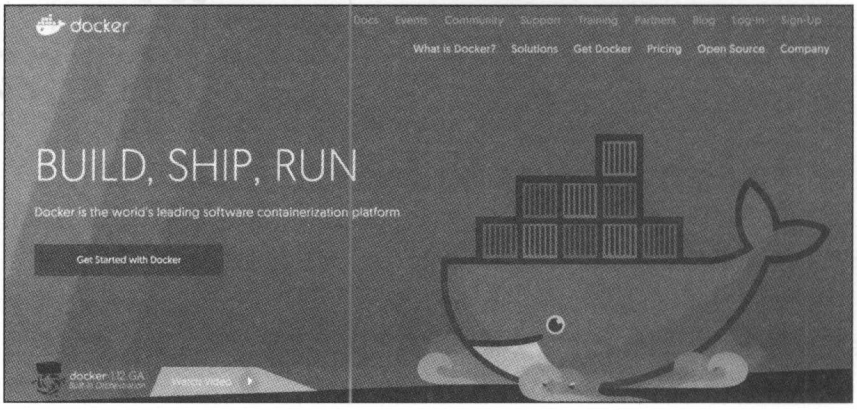


图 1-1 Docker 官方网站

Docker 项目已加入了 Linux 基金会，并遵循 Apache2.0 协议，全部开源代码均在 https://github.com/docker/docker 上进行维护。在 Linux 基金会最近一次关于“最受欢迎的云计算开源项目”的调查中，Docker 仅次于 2010 年发起的 OpenStack 项目，并仍处于上升趋势。

现在主流的 Linux 操作系统都已经支持 Docker。例如，红帽公司的 RHEL 6.5/CentOS 6.5 往上的操作系统、Ubuntu 14.04 往上的操作系统，都已经在软件源中默认带有 Docker 软件包。Google 公司宣称在其 PaaS（Platform as a Service）平台及服务产品中广泛应用了 Docker 容器。IBM 公司跟 Docker 公司达成了战略合作伙伴关系。微软公司在其云平台 Azure 上加强了对 Docker 的支持。公有云提供商亚马逊也推出了 AWS EC2 Container 服务，提供对 Docker 和容器业务的支持。

Docker 的构想是要实现“ Build, Ship and Run Any App, Anywhere”，即通过对应用的封装（Packaging）、分发（Distribution）、部署（Deployment）、运行（Runtime）生命周期进行管理，达到应用组件“一次封装，到处运行”的目的。这里的应用组件，既可以是一个 Web 应用、一个编译环境，也可以是一套数据库平台服务，甚至是一个操作系统或集群。

基于 Linux 平台上的多项开源技术，Docker 提供了高效、敏捷和轻量级的容器方案，并支持部署到本地环境和多种主流云平台。可以说，Docker 首次为应用的开发、运行和部署提供了“一站式”的实用解决方案。

2. Linux 容器技术——巨人的肩膀

跟大部分新兴技术的诞生一样，Docker 也并非“从石头缝里蹦出来的”，而是站在前人的肩膀上，其中最重要的就是 Linux 容器（Linux Containers，LXC）技术。

IBM DeveloperWorks 网站关于容器技术的描述十分准确：“容器有效地将由单个操作系统管理的资源划分到孤立的组中，以更好地在孤立的组之间平衡有冲突的资源使用需求。与