Stock Watcher
Build a custom app to monitor your portfolio

SPECIAL docker

LINUX MAGAZINE

ISSUE 277 – DECEMBER 2023

Low-Code Tools
Save money by keeping it simple

Defending WordPress: Stay ahead of intruders

GitHub Codespaces: Coding in the cloud

Create a Smart Home Info Center

Collage Tools: Showcase your pics with Fotowall and PhotoCollage

DNS over TLS: Keep your lookups under wraps

10 TERRIFIC FOSS FINDS!

LINUX NEW MEDIA
The Pulse of Open Source

Open source container tools

Total Package

Docker provides the open source tools and resources for compiling, building, and testing containerized applications. *By Amy Pettle*

By all accounts, Docker's developer tools have been an important player in the recent history of enterprise IT. Although containers were not new in 2013, the release of the open source Docker platform made containers more accessible to everyday admins by simplifying development and deployment. Docker also helped shape the container landscape by joining with other container industry leaders to establish the Open Container Initiative (OCI), a Linux Foundation project that maintains open industry standards around container formats and runtimes. Docker even contributed runc, the original OCI container runtime, to the foundation in 2015.

In recent years, container technology has proven to be reliable and ubiquitous, resulting in attention shifting to orchestration tools such as Kubernetes.

Docker the company actually sold its Enterprise division to Mirantis in 2019. Included in that sale was the Docker Swarm orchestration platform, which Mirantis has continued to market under the Docker Swarm brand, causing some confusion over what is Docker and what isn't. For instance, some viewers (incorrectly) visualize Docker as somehow competing with Kubernetes, when in fact, Mirantis competes with Kubernetes, and the original Docker container platform is fully integrated into the Kubernetes environment.

Docker's developers are still hard at work, focused on development tools. Their philosophy is that, although it is possible to build a single container on the fly without the need for an enhanced toolset, if you build containers for a living or are concerned with security and consistency in your container creations, you'll need a versatile set of development tools.

Docker remains heavily invested in open source, and several Docker tools are available under open source licenses. This article takes a tour of the container-building tools in the Docker toolset and offers a glimpse at where the company has been putting its energies.

Many of these open source tools have found their way into the Moby Project, an upstream, community-governed project for container components, which Docker founded in 2017. Moby offers a toolkit of components, a framework for assembling these components, and a community for testing and sharing ideas.

Docker Engine

Docker Engine [1], the open source container engine for building containerized applications, forms the core of Docker's developer tool suite. Developed upstream in the Moby Project, Docker

Engine uses a client-server architecture (Figure 1). Docker Engine consists of the daemon (`dockerd`) and APIs that specify which programs can talk to and instruct the daemon.

Docker's open source CLI client (`docker`) interacts with Docker Engine, letting you manage your containers from the command line. It talks to the daemon, which does the work of building, running, and distributing containers. Written in Go, the Docker CLI manages single containers, as well as container images, networks, and volumes. For managing multiple containers, you will need a separate tool, Docker Compose (see below).

For its container runtime, Docker Engine relies on containerd, which manages the container life cycle and handles creating, stopping, and starting your containers (see the containerd section for more information).

Docker Engine also integrates the open source BuildKit component. BuildKit replaced and improved the legacy builder in the release of Docker Engine 23.0. BuildKit offers improvements in performance, storage management, and extensibility. Unlike the legacy builder, which performs builds serially from Dockerfiles (the text file containing all the commands called at the command line to assemble an image), BuildKit allows parallel build processing and introduces support for handling more complex scenarios, such as the ability to detect and skip execution of unused build stages.

Docker Engine binaries are available as DEB or RPM packages for CentOS, Debian, Fedora, Ubuntu, Red Hat Enterprise Linux (RHEL), SUSE Linux Enterprise Server (SLES), and Raspberry Pi OS. Docker also offers a static binary for non-supported Linux distributions, but it is not recommended for production environments.

Docker Compose

If you are looking to define and run multi-container Docker apps, you will need Docker Compose [2]. Available as a plugin for Docker Engine, Docker Compose lets you run a project with multiple containers from a single source.

Docker Compose uses a YAML file to configure your application's services. Once configured, you can use a single command to create and start all of your configuration's services.

Docker Compose provides commands for an application's entire life cycle. You can use Docker Compose in all environments: development, testing, staging, continuous integration (CI), and even production. Because Docker Compose is an abstraction that aligns with developers' mental model of their applications, it particularly supports the inner loop of application development.

A recent addition to Docker Compose is Docker Compose Watch [3]. The Watch feature lets you define a list of rules that will cause an automatic service update when a file is modified. Watch monitors the files in the local directory, rebuilding the application container when necessary so the application stays up to date.

containerd

Docker donated containerd [4], an industry-standard container runtime, to the Cloud Native Computing Foundation (CNCF) in 2017. With an emphasis on simplicity and portability, containerd manages the complete container life cycle on a host – from image transfer and storage to container execution and supervision, to low-level storage and network attachments.

Designed to be embedded in a larger system such as Docker Engine, containerd functions as an internal runtime with minimal runtime requirements. The containerd daemon is available for both Linux and Windows, with most of its interactions with these operating systems' container feature sets being handled by runc or operating system-specific libraries.

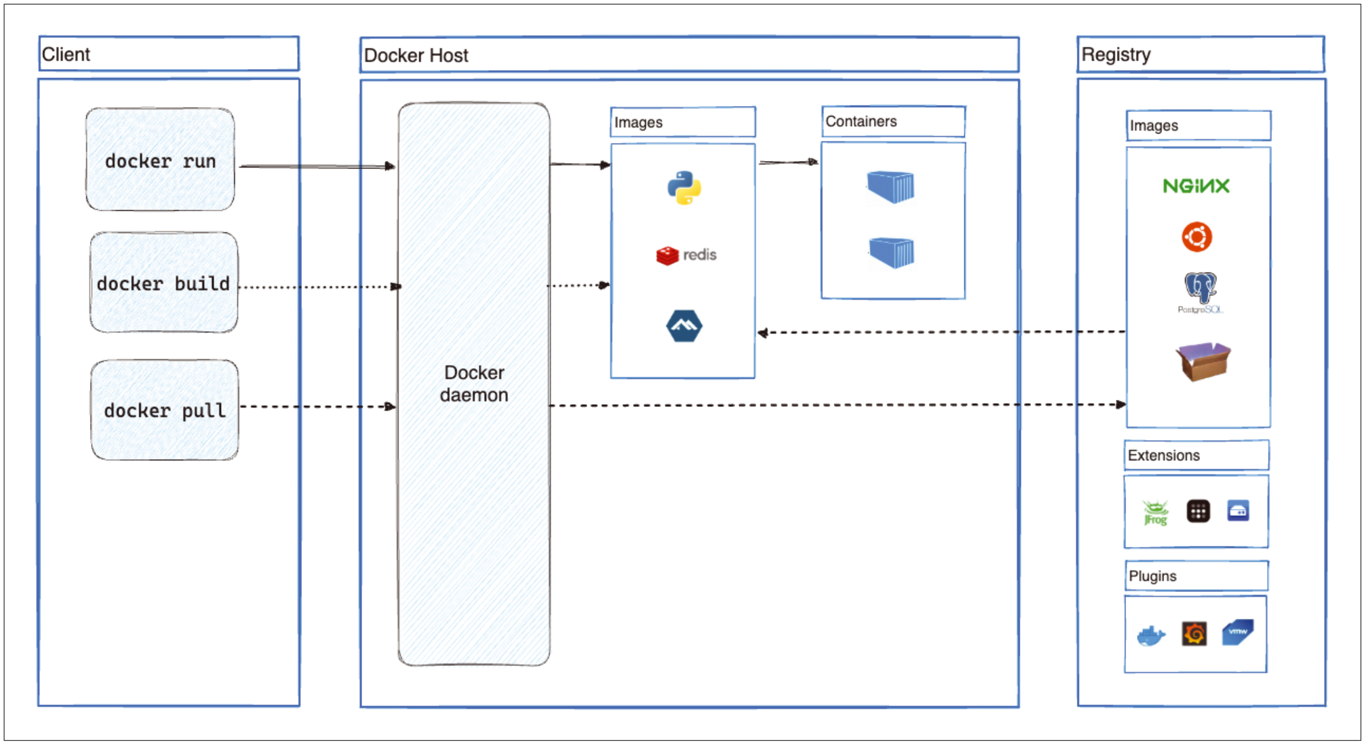


Figure 1: **Docker's client-server architecture.**

Figure 2: A badge appears alongside images that are published by DSOS projects.

By design, containerd works with Docker and Kubernetes, as well as any container platform that wants to abstract syscalls or operating system-specific functionality. Whereas containerd implements the Kubernetes Container Runtime Interface (CRI) for those wanting to run a Kubernetes cluster, you can just as easily use containerd without Kubernetes. However, if you do plan to use Kubernetes, you will want to use containerd v1.5.11 or v1.6.4 to address the recent removal of dockershim from Kubernetes.

You can find containerd as a binary package for the 64-bit architectures Intel AMD, PowerPC Little Endian, and RISC-V, as well as for the S390x architecture.

Other Open Source Tools

As an active member of the container open source community, Docker collaborates on other projects and tools. Some of these projects include:

- Distribution [5] (formerly Docker Distribution): This toolkit, which Docker donated to the CNCF, lets you pack, ship, and deliver content. It contains the Open Source Registry implementation for storing and distributing container images using the OCI Distribution Specification, which defines an API protocol to facilitate and standardize content distribution. Docker Hub, GitHub Container Registry, and GitLab Container Registry all use this open source code as the basis for their container registries.
- DataKit [6]: Developed upstream in the Moby Project, this tool orchestrates apps using a Git-like data flow. It is used as the coordination layer for HyperKit, another Moby tool that functions as the hypervisor component of Docker for macOS and Windows, as well as for the DataKitCLI continuous integration system.
- Notary [7]: Donated to the CNCF by Docker, this client and server runs and interacts with trusted collections. It allows users and publishers to easily verify content, making the Internet more secure. Notary is used in Docker Content Trust [8], which relies on digital signatures for sending and receiving data from remote Docker registries.
- runc [9]: This CLI tool spawns and runs tools on Linux in accordance with the OCI specification. This lightweight portable container runtime can be used in production.

Docker-Sponsored Open Source Program

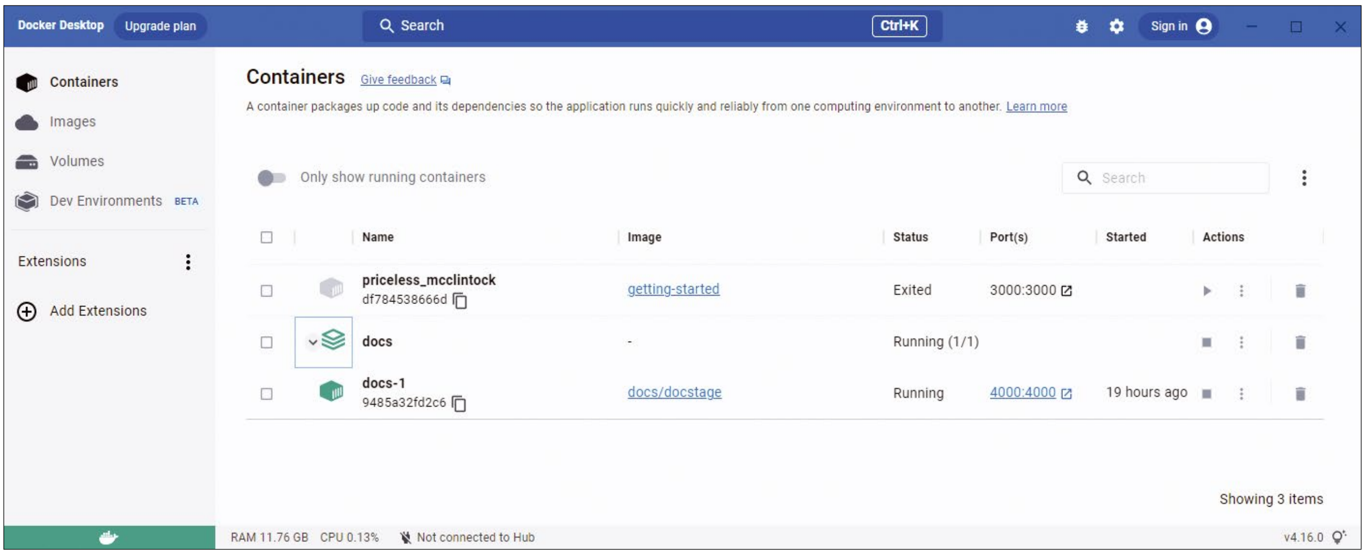
In addition to contributing open source tools, Docker offers a special program, Docker-Sponsored Open Source (DSOS) [10] for developers working on open source projects that don't have a path to commercialization. Started in 2020, DSOS is the successor to the Free Team subscription offered prior to 2021. Over 900 projects are currently part of the DSOS program.

Being a DSOS member means your projects receive a special badge (Figure 2) in Docker Hub, Docker's container image registry that lets open source contributors find, share, and use container images. The badge signifies that your project has been verified and vetted by Docker and is part of Docker's Trusted Content [11]. In addition, DSOS projects receive free automatic builds on Docker Hub. Program members and users who pull images from your project namespace also will get unlimited pulls and egress, and DSOS members also receive a freeDocker Team subscription, which includes Docker Desktop. By the end of 2023, DSOS projects will also receive Docker Scout Team as part of their participation in the program.

To find out if your project qualifies for DSOS, see the "DSOS Requirements" box.

Docker Commercial Tools

If you are looking for an easy way to get started with Docker, you might be interested in Docker Desktop [14], Docker's out-of-the-box containerization software. Docker Desktop's simple interface doesn't require you to run Docker from the command line. In addition, it handles container setup and automatically



Containers

A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another.

Name	Image	Status	Port(s)	Started
priceless_mcclintock df784538666d	getting-started	Exited	3000:3000	
docs	-	Running (1/1)		
docs-1 9485a32fd2c6	docs/docstage	Running	4000:4000	19 hours ago

RAM 11.76 GB CPU 0.13% Not connected to Hub v4.16.0

Figure 3: You can easily build, run, and share containers from the Docker Desktop dashboard.

applies kernel updates and security patches. Docker Desktop combines Docker's open source components into an easy-to-use GUI (Figure 3) that lets you access important development options with one click. In addition to Docker's open source components, Docker Desktop includes useful tools such as Docker Extensions, which lets you connect the Docker environment to tools you are already using, plus access to Docker Hub for container images and templates, as well the ability to run a single-node Kubernetes cluster.

Although Docker Desktop is a subscription-based offering, Docker does offer a free Docker Personal subscription [15], which is best suited to individual developers, students and educators, noncommercial open source projects, and small businesses with fewer than 250 employees and less than $10 million in revenue.

Additionally, Docker Scout [16] is a software supply chain product that provides context-aware recommendations to developers. The goal of these recommendations is to help the developer build applications that are reliable and secure from the start. You can use Docker Scout in Docker Desktop, Docker Hub, and the Docker Scout Dashboard. In the Docker CLI, you can use the Docker Scout CLI plugin, which is available as a script or can be installed manually as a binary. As of early October 2023, Docker Scout is in general availability

DSOS Program Requirements

To qualify for the DSOS program, your project must meet the following requirements:

- Shared in Docker Hub's public repositories with the source code publicly accessible
- Compliant with the Open Source Initiative definition of open source software [12]
- Active on Docker Hub with updates pushed regularly in the past six months or dependencies updated regularly, even if your code is stable
- No pathway to commercialization, which means you cannot profit through services or charge for higher tiers, although you can accept donations
- Your Docker Hub repositories contain documentation that meets the recommended community standards

If you are interested in the DSOS program, submit an application online [13].

Conclusion

Docker plays an important role in the open source container ecosystem. Many core Docker components are open source, and initiatives such as the DSOS program help independent software developers make their projects available to the community. Docker developer tools also support projects such as Moby, CNCF, and OCI that encourage a free, standards-based approach to container development. ■■■

This article was made possible by support from Docker through Linux New Media's Topic Subsidy Program (https://www.linuxnewmedia.com/Topic_Subsidy).

Author

Amy Pettle is an editor for *ADMIN* and *Linux Magazine*. She started out in tech publishing with *C/C++ Users Journal* over 20 years ago and has worked on various Linux New Media publications.

Info

[1] Docker Engine: *https://docs.docker.com/engine/*

[2] Docker Compose: *https://github.com/docker/compose*

[3] Docker Compose Watch: *https://docs.docker.com/compose/file-watch/*

[4] containerd: *https://github.com/containerd/containerd*

[5] Distribution: *https://github.com/distribution/distribution*

[6] DataKit: *https://github.com/moby/datakit*

[7] Notary: *https://github.com/notaryproject/notary*

[8] Docker Content Trust: *https://docs.docker.com/engine/security/trust/*

[9] runc: *https://github.com/opencontainers/runc*

[10] DSOS: *https://www.docker.com/community/open-source/application/*

[11] Trusted Content: *https://docs.docker.com/trusted-content/*

[12] Open Source Initiative definition: *https://opensource.org/osd/*

[13] DSOS application: *https://www.docker.com/community/open-source/application/*

[14] Docker Desktop: *https://www.docker.com/products/docker-desktop/*

[15] Docker Personal: *https://www.docker.com/pricing/*

[16] Docker Scout: *https://docs.docker.com/scout/*