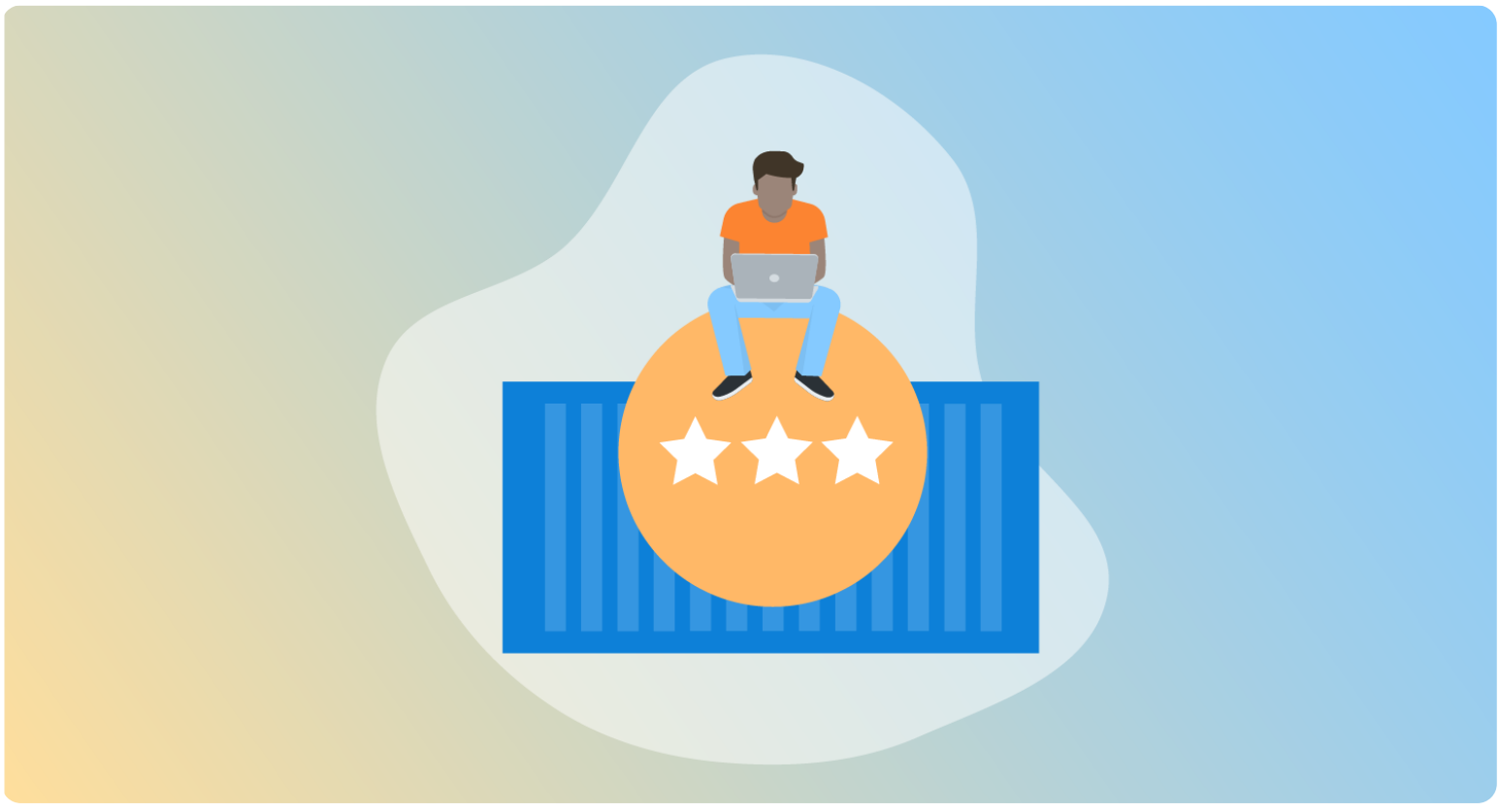


The benefits of containerization

Terence Wong

July 13, 2022 • 5 mins

As a recent technology, containers have emerged as a tool that can help your business become more agile in your software development lifecycle. Containers have many benefits that can give you a competitive advantage compared with more traditional software delivery methods.

In this post, I explain what containers are, share the key benefits of containers for software development, and discuss why you might consider adding them to your DevOps processes.

What is containerization?

A container is a lightweight, portable computing environment that includes all the necessary files to run independently.

Containerization is the process of making an application runnable as a container. Once the application can run as a container, it runs the same regardless of the infrastructure used to execute the container. Containers are loaded with container images that run a specific application inside the container. If you want to build a modern application, from setting up a database, to loading different operating systems, to accessing a deep learning platform, you're going to need containerization.

Containerization has been widely adopted in recent years, partly due to the availability of cloud technologies. Cloud technologies let you scale and replicate containers, and they lower the barrier to entry.

If you work in DevOps, you've probably worked with containers before. If you're trying to get started, check out the DockerHub library of images to see what images you can use or our post on getting started with containers.

What are the benefits of containerization?

Containerization can be a useful tool for you to enhance the software development lifecycle.

The benefits include:

- Containers complement your DevOps process
- Containers are scalable and allocate resources efficiently

- Containers are portable so you can build once, run anywhere

Containers complement your DevOps process

In our **introduction to DevOps post**, we discussed how DevOps as a concept is about removing barriers that get in the way of software delivery.

DevOps refines every process between the developer and the customer, and encourages faster feedback loops, experimentation, and learning. DevOps is a practice that focuses on agility and automation.

Containerization complements DevOps because software can be deployed and tested faster, improving feedback loops. Containerization is also a major factor in the popularity of microservices, a software architecture that improves flexibility and agility. You can use containerization to speed up the time it takes to develop new features and get feedback. Improving the feedback loop for your product leads to a better product and happier customers.

Containers are scalable and allocate resources efficiently

Platform as a Service (PaaS) solutions and container orchestration tools like Kubernetes let developers operate containers at scale. Container orchestrators can scale individual components in software applications up and down depending on demand and load. This leads to cost savings as components only run for as long as they're needed. Scaling also improves reliability as container orchestrators can allocate sufficient resources to high-demand parts of the application.

Scaling and cost savings are important factors when deciding to migrate to containerization. Many cloud providers have a cost calculator for cloud resources that you can use if you want your department to make the switch to containers.

Containers are portable: build once, run anywhere

Because containers are portable, they can run anywhere on any infrastructure, such as in the cloud, on a VM, or bare metal.

The Open Container Initiative (OCI) designs open standards for containers, ensuring any OCI compliant containers run the same way on any infrastructure.

To run applications, containers are loaded with container images. A container image is a static file that contains executable code to run a process on IT infrastructure. There are container images for different use cases such as databases, web servers, operating systems, and more. Container image repositories are public access points for container images, which makes them available to developers who can load a container with these images.

If you want to use a container for your application, you can be sure that any OCI image you use will work on your infrastructure, even if your infrastructure changes.

What are the top container images?

Docker Hub provides a list of popular container images. Some of the top container images are:

- Ubuntu: a Debian-based Linux operating system.
- NGINX: an open-source web server, load balancer, and reverse proxy used in several applications.
- Postgres: an open-source relational database system that uses the SQL language.
- Redis: an open-source in-memory data structure store used as a database, cache, and message broker.
- Alpine: a Linux distribution built around musl libc and BusyBox.

Popular container images are often open-source and address a fundamental need in software applications, such as databases, web servers, or caches. These use cases are common to most software projects and tools have already been built to address them.

If you're starting a software project, you don't want to reinvent the wheel and figure out how to build a relational database or webserver by yourself - and containerization means you don't have to! The power of containerization helps developers build on existing solutions to solve new problems.

What are the primary tools for container technologies?

Cloud PaaS solutions like Microsoft Azure, Amazon Web Services, and Google Cloud Platform provide the infrastructure to run technologies like Docker and Kubernetes. The open-source Docker container technology was launched in 2013. Since then, it's gained widespread adoption as the leading container technology. Kubernetes is the most popular container orchestration technology used alongside Docker to manage and scale container solutions. Kubernetes is a management layer that organises and provisions infrastructure to host containers and execute workloads.

The containerization landscape is fluid and ever changing, so you should monitor major updates to check whether your software stack is affected by any changes. For instance, while Docker has been the most common container technology run on Kubernetes, the **v1.24 Kubernetes update** deprecated Dockershim - an underlying module that provided compatibility between Docker and Kubernetes. The update was mainly due to Docker's compatibility with the Container Runtime Interface. Docker has developed a replacement for Dockershim called cri-dockerd that addresses compatibility issues for users that still want to use Docker Engine to run containers in Kubernetes. The cri-dockerd adapter lets you use Docker Engine through the Container Runtime Interface.

A **report by Datadog** in 2021 indicated a 6% increase in container adoption with a correlated dip in Docker usage. The increase in container adoption rate may continue as Kubernetes moves away from full Docker support. The containerization and container orchestration landscape is rapidly evolving. The technological tools and popularity may change, but the containerization and container orchestration concepts are here to stay.

Containerization support in Octopus Deploy

A deployment process can use some form of containers or container orchestration to deploy an application. Octopus is a deployment management tool that supports containerization. Octopus Deploy works with container registries, PaaS providers, Docker, and Kubernetes to provide a best-in-class deployment management tool. Regardless of which container technologies are most popular moving forward, Octopus Deploy can work with all of them to provide happier deployments.

Conclusion

Containers are standalone computing environments, and containerization converts an application into a runnable container. Containerization gives the development process flexibility and agility, which helps DevOps processes. Containers are highly portable, and OCI compliant containers can be built once and run anywhere. With PaaS solutions and container orchestration tools like Kubernetes, containers are scalable to allocate resources efficiently.

Containerization is an ever-changing field of research. The popularity of specific tools may shift and change, but Octopus Deploy is container and cloud-agnostic. It works with a range of container registries, PaaS providers, Docker, and Kubernetes to help make your complex deployments easier.

Happy deployments!

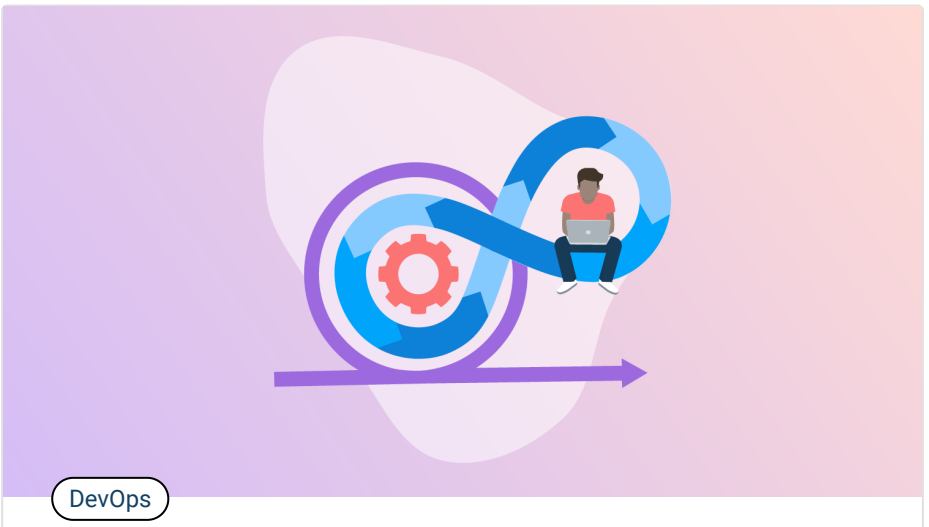
Tagged with: DevOps Containers Cloud Orchestration

Related posts

DevOps

Common mistakes in DevOps metrics

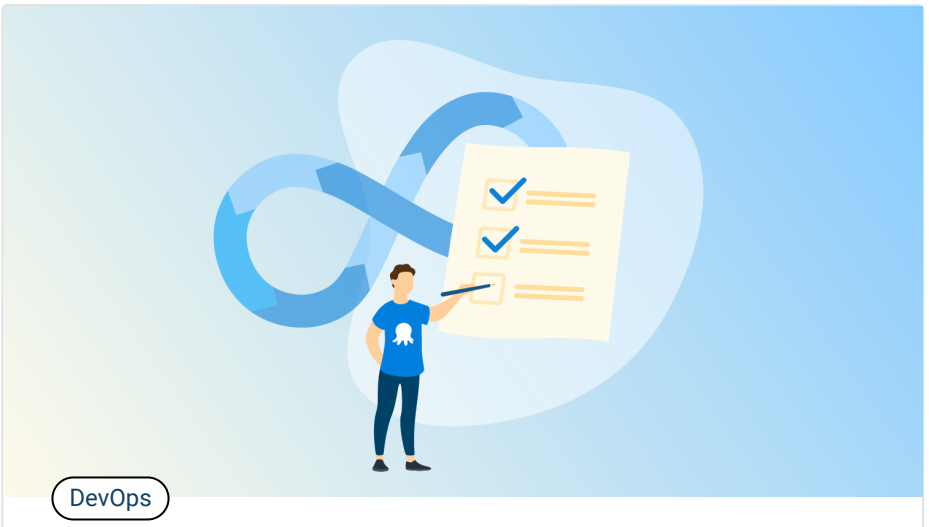
Steve Fenton
December 7, 2022 • 6 mins



DevOps

Comparing Lean, Agile, and Continuous Delivery

Steve Fenton
December 5, 2022 • 6 mins



DevOps

Best practices for CI/CD

Terence Wong
November 30, 2022 • 5 mins

Newsletter

Logged in as Terence Wong (terence.wong@octopus.com)

Join ~48,000 DevOps professionals and sign up for the latest Octopus news, events, and opinions. No spam. Unsubscribe at any time.

Subscribe

Your privacy is important to us. Read more in our Privacy Policy.