



[Contents](#)



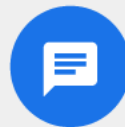
[What are containers?](#)



[Misconceptions](#)



[Where to start](#)

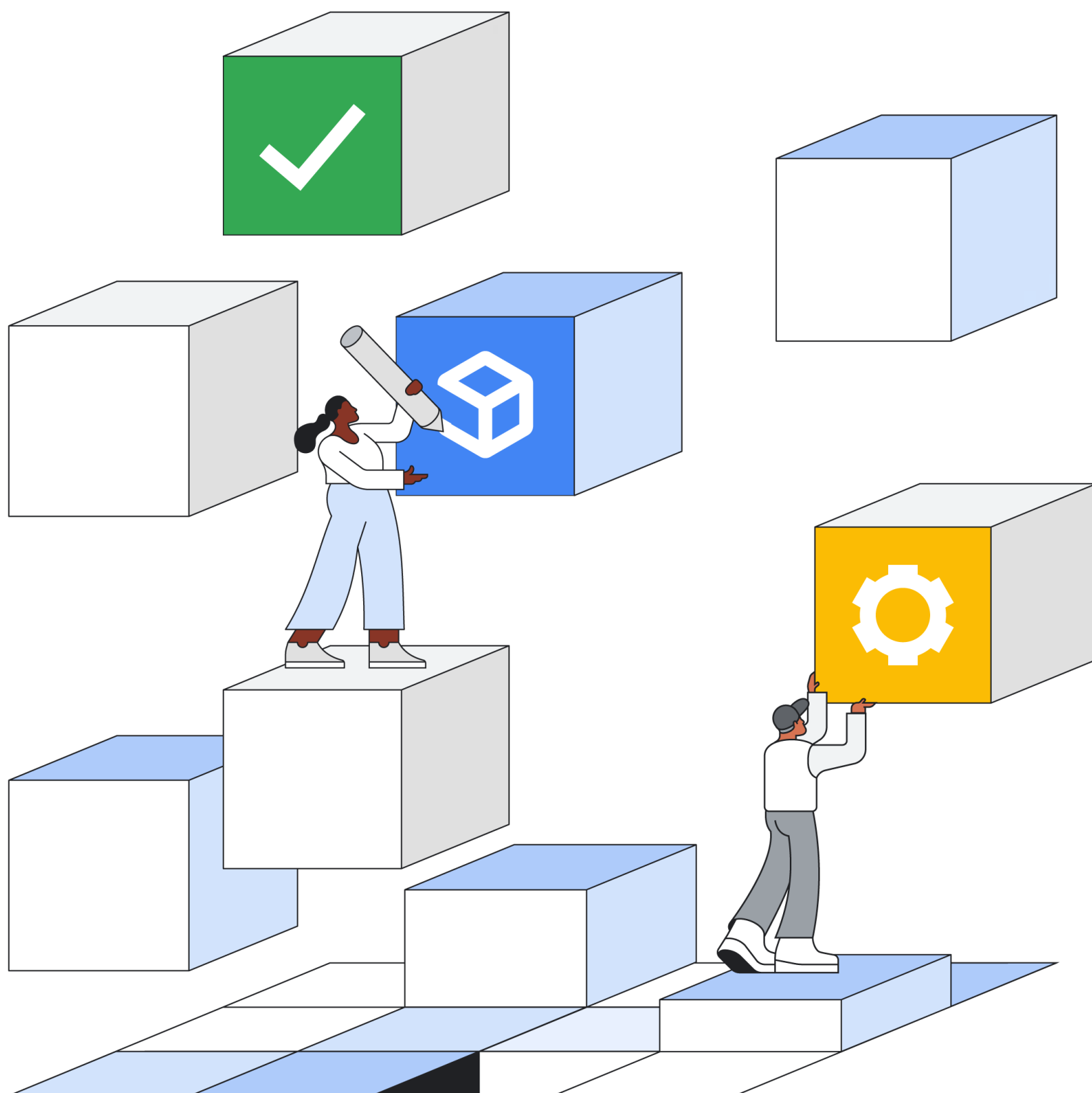


[Talk to us](#)



Why containers are the total package

Drive change with a simpler way to package application software





In this guide

Containers are the foundation of modern software development, revolutionizing the way we develop and deploy applications. They're crucial to adopt in order to stay agile and efficient in today's world. Discover how IT leaders like you can package software simply and unlock innovation at scale.



Inside the container revolution »

Dig into the definition of containers and the critical role they play in today's world.



The truth about container misconceptions »

What's holding IT leaders back from implementing containers? Uncover the truth behind four common misconceptions.



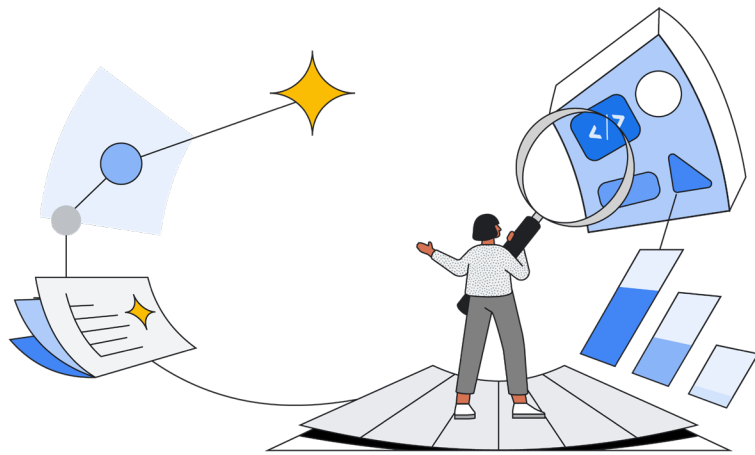
Kickstart your container journey »

Check out three strategies for adopting containers and decide which approach best fits your business.



■ Inside the container revolution

In today's rapidly changing business world, organizations need to be able to adapt quickly to new opportunities and challenges and accelerate their **digital transformation**.



Digital transformation is the process of using technologies to fundamentally change the way an organization operates—and containers are key to digital transformation. According to the [2022 State of DevOps report](#), containers have been the number one deployment target for primary services and applications for the past two years, and adoption continues to grow rapidly.

↑ 86%

of IT leaders are prioritizing containers to use in more applications.¹

↑ 90%

of global organizations will be running containerized applications in production by 2026—up from 40% in 2021.²

¹[The Forrester Report, Container Adoption in the Enterprise, June 2020.](#)

²[Gartner, The Innovation Leader's Guide to Navigating the Cloud-Native Container Ecosystem, August 18, 2021.](#)



Containers are the heart of modern applications and offer a number of benefits that help organizations accelerate their digital transformation:

Increased agility: Containers make it possible to quickly and easily deploy new applications and services. This efficiency allows organizations to respond rapidly to market changes and meet customer needs.

Reduced costs: Containers save organizations money on IT costs by reducing the need for physical servers and simplifying infrastructure management.

Improved security: Containers isolate applications from each other and from the underlying infrastructure, making it harder for attackers to exploit vulnerabilities.

To better understand the benefits, let's take a look inside containers.

Containers 101

A container is a software package with everything needed to run an application, including the application code, libraries, and operating system. Containers are lightweight and portable, making them easy to deploy and manage.

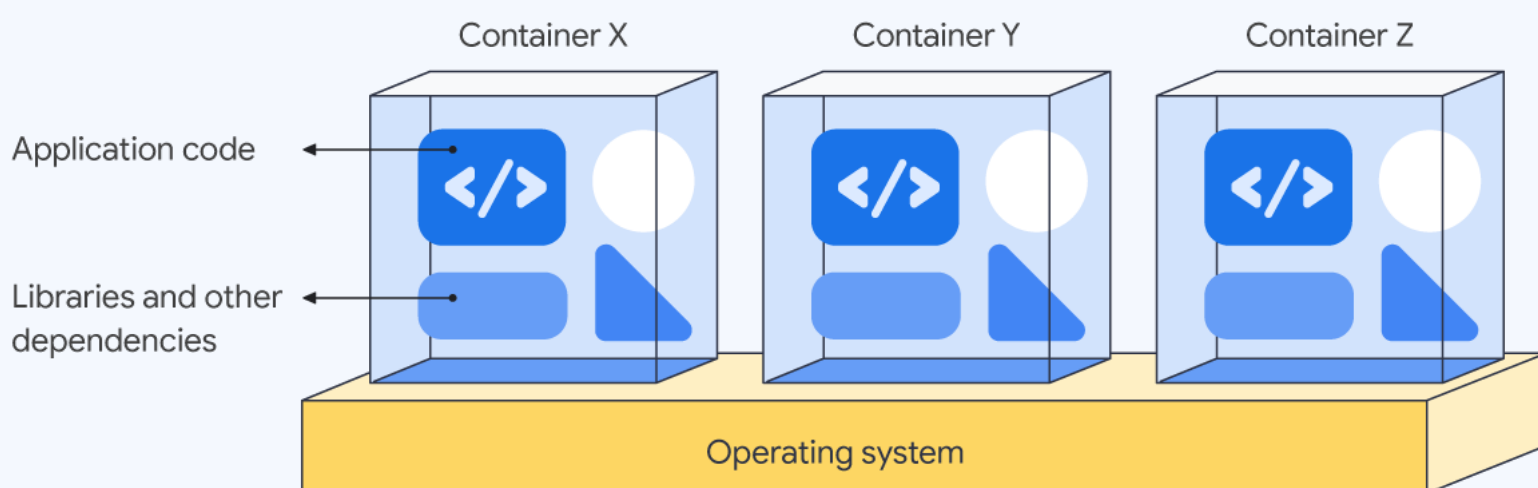


Figure 1: A container packages an application code and its dependencies.

Running code in a container is simple. You don't need to worry about installing any libraries, frameworks, or other supporting software for this application—they're already in the container.

■ Breaking down containers

To understand how containers work, it's important to first understand two different aspects:

1. **A container image contains the compiled code.** This typically includes the application code along with code for libraries or other dependencies the container includes. Container images can be stored on disk and transmitted across a network.
2. **A container instance is a running occurrence of a container image.** It exists in the memory of a physical or virtual machine, and it's created by loading and executing a container image.

Figure 2 illustrates how these two ideas relate to one another.

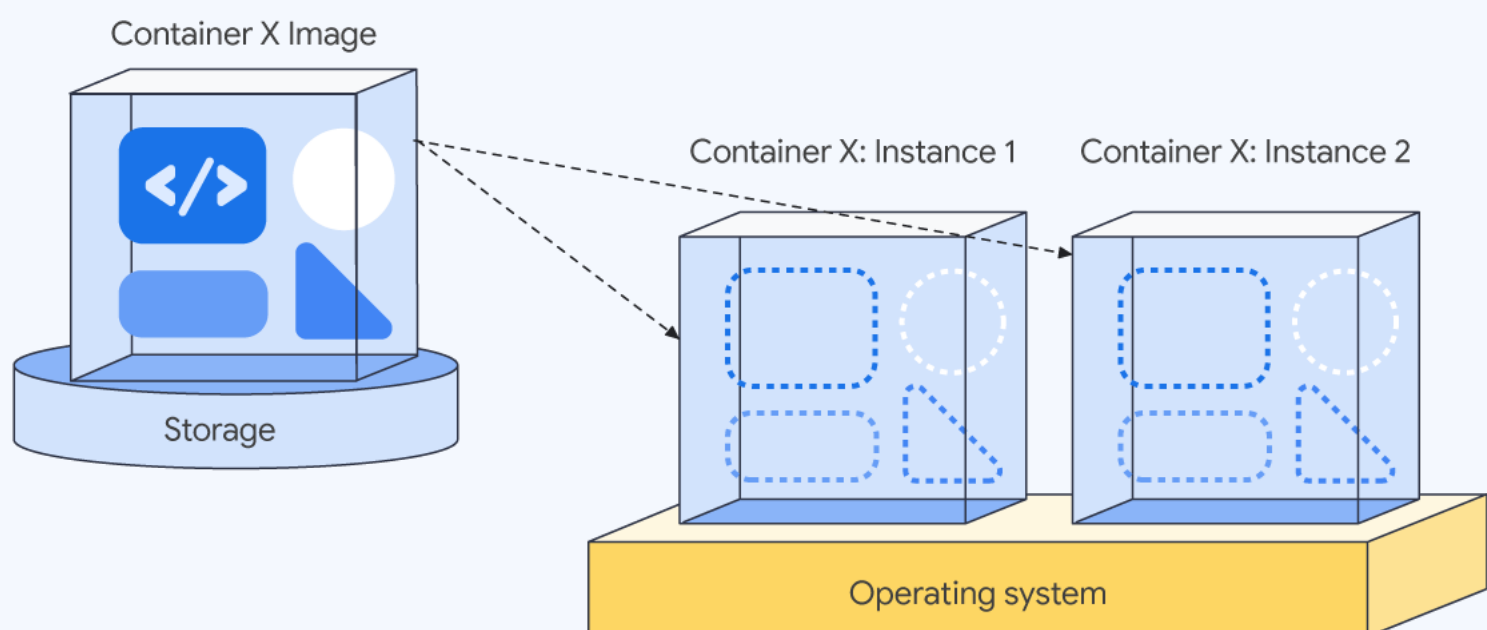


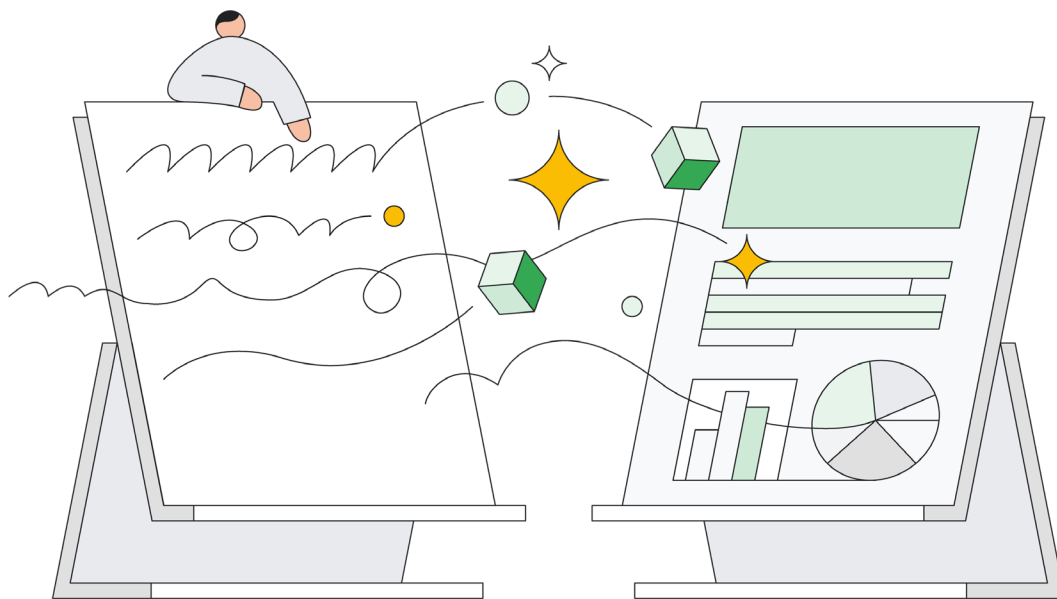
Figure 2: The same container image can be used to create many running container instances.

The application code in a container might comprise a complete application, or a single application's code can be divided across two or more containers. And the application can be either a new greenfield solution or an existing app that you've chosen to package into a container.



■ The truth about container misconceptions.

There are many misconceptions that hold IT leaders back from adopting containers. Let's clear up the most common ones and set the record straight.



The truth:
It's riskier to miss out on this useful technology.

Misconception: “Containers are too risky for my organization; they’re only for cutting-edge organizations.”

Kubernetes, an open sourced system for containerized apps, was first announced by Google in 2014, and the Google Cloud Google Kubernetes Engine service was made available that same year. These are not new technologies—they’ve been used by thousands of organizations around the world for years—and they’re no longer for just early adopters. A May 2021 IDC study predicts a significant shift to containers and anticipates that 80% of workloads will be moved to or created with containers and microservices, reducing per-app infrastructure needs by 60% while driving a 70% improvement in digital service resiliency.¹

The truth is that containers have become so mainstream that avoiding this technology is almost certainly riskier than adopting it. Containers are commonplace today in application development and deployment.

¹[CIO, Why Enterprise IT Organizations Will Benefit from Application Containerization, November 2021.](#)



The truth:

Using
containers
makes your
world simpler.

Misconception: “Containers are too complex.”

This misunderstanding is especially damaging because the truth is just the opposite. As with adopting any technology, your team will need to learn some new things, but once you’re over this initial hump, you can expect lower costs, faster development, and better applications.

You can also use containers for most enterprise use cases, including wrapping legacy apps and creating new solutions. While leading-edge organizations sometimes use containers to create more advanced applications using microservices, this is in no way required; while containers can make this kind of architecture easier to create, adopting containers doesn’t necessarily mean using microservices.

Also, don’t confuse using containers with using Kubernetes. Kubernetes can bring some complexity, but there are solutions that can handle most of the management for you. And even simpler, running containers with a serverless option requires no infrastructure management at all.

Serverless computing: what it is and how it works

“Serverless” means the ability to run a function on demand, then pay only for the compute resources this function uses. A technology is serverless if it meets three criteria:

- It doesn’t expose an infrastructure that requires user management.
- Applications scale automatically.
- It charges only for the resources applications actually used.

Developers sometimes treat serverless computing and containers as competing alternatives, but any technology that meets the three requirements described above qualifies as serverless.

Serverless functions are commonly single purpose—they do only one thing—while containers are usually multipurpose, even when they use a serverless technology.



The truth:

Adopting containers will make top developers want to work with you.

Misconception: “My organization needs rare and expensive development talent to use containers.”

When you use a container-based environment, quality developers want to work with you. Few developers want to join an organization that uses solely legacy code instead of the mainstream platforms. And because containers rely on broadly supported technologies such as Docker and Kubernetes, your organization can use standard skills that every container developer has learned.

In fact, *not* adopting containers will likely make it harder for you to hire and retain developer talent going forward. As top talent keeps up with the times, staying locked into a legacy world limits your future hiring to the ever-shrinking set of people with only legacy skills, people whose increasing scarcity will make them more and more expensive.



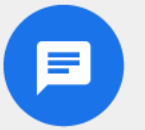
The truth:

Modern container environments can run all kinds of workloads.

Misconception: “We can only run stateless workloads with containers.”

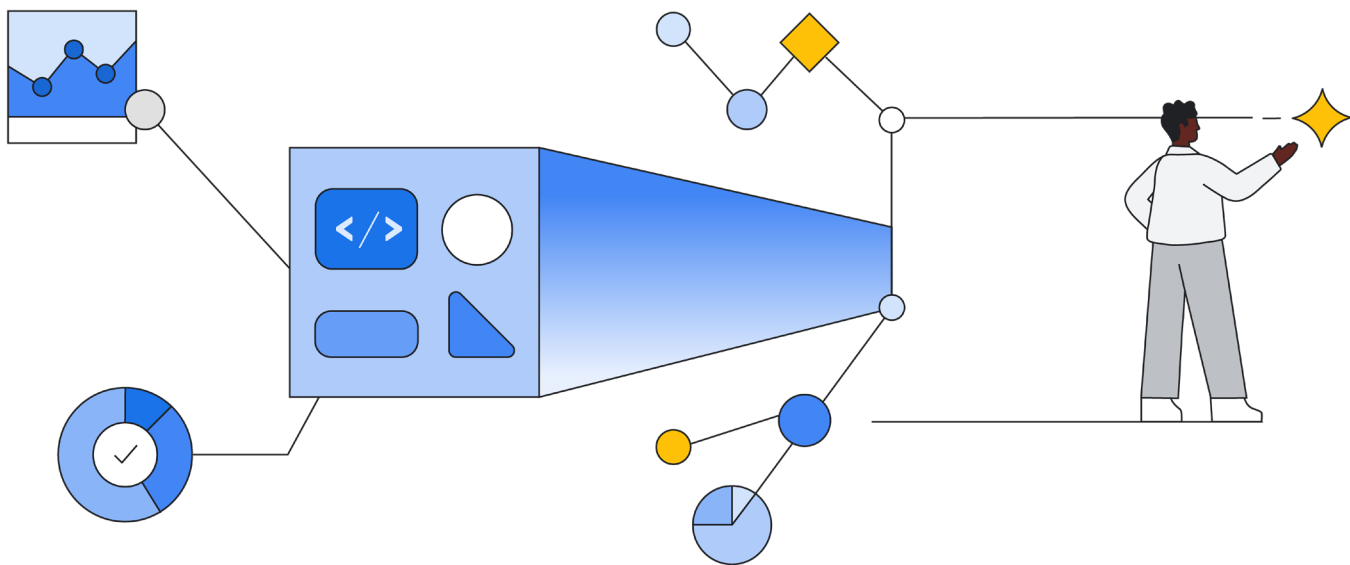
Stateless applications, which don’t maintain any state on their local machine, are quite common today. Unlike a stateful application, which stores data on the specific server it runs on, a stateless app is an application program that does not save client data generated in one session for use in the next session with that client. Each session is carried out as if it was the first instance and responses are not dependent upon data from a previous session.

Modern container environments, including both Google Kubernetes Engine and Cloud Run, can run all kinds of workloads, ranging from stateless e-commerce apps to complex stateful workloads. Whether a workload is stateful or stateless is no longer a factor in deciding whether it should be containerized. Whichever protocol is right for your organization, it can run on containers.



■ Kickstart your container journey.

Once you've decided to adopt containers, you need a place to start. As with most new technologies, you probably don't want to begin with a mission-critical application; that way, if there are challenges, it won't seriously damage your organization.



■ Here are three ways to start on your container journey:

- 01 | **Wrap an existing application and its dependencies in a container.** This is relatively simple to do, and it will make infrastructure management easier. It will also help your organization familiarize itself with the container tools and services you've chosen.
- 02 | **If an existing, stateful custom application needs to be more scalable and/or reliable, modify the app to make it stateless, then wrap the result in one or more containers.** This gives you easier infrastructure management, along with the ability to let whatever service runs the app handle scalability and reliability for you.
- 03 | **If you're creating a new greenfield application, use containers throughout development and deployment.** This approach gives you all the benefits of container-based development, like a modern workflow with fewer bugs, along with a more secure development environment. You also get the benefits of container-based deployment, such as simpler infrastructure management.



Pro tip

Try to use containers first in a scenario with demonstrable business value. For example, a container-based application can often be updated more quickly than a conventional solution—a clear win for the business. Just remember to accurately measure update frequency before and after the move.

As with any new technology, you want to make sure your first efforts are successful. Once you've demonstrated both the feasibility and the value of using containers, expect others in your organization to join you on the container bandwagon.

Containers on Google Cloud: a powerful combination

It's important to understand the basics of containers, but it's also helpful to look at a real example of a modern container-based environment to get a sense of the technology's true value. And while you don't need to move to the cloud to adopt containers, Google Cloud is a good example of how you can leverage container-based technologies alongside cloud platforms.

Google Cloud provides a complete set of services for creating and running container-based applications. These include:

- [Cloud Run](#), a serverless approach to running containers. (See pg 7: “Serverless computing: what it is and how it works.”) This service provides built-in scalability, automatically creating and destroying container images as needed to handle changing loads on the application. It also provides built-in load balancing, spreading requests across container instances.
- [Google Kubernetes Engine \(GKE\)](#), a scalable Kubernetes platform. Originally created by Google, Kubernetes is a container orchestration technology that's now widely supported across the industry. Like Cloud Run, it provides built-in scalability and load balancing for container-based applications.

Cloud Run and GKE both use the same container image format, so you can choose the approach that best meets your needs. Cloud Run is simpler to use and manage than GKE, but it also offers less detailed control of your running containers. The most common approach is to start with Cloud Run, then move to GKE only if you need more control than Cloud Run offers. And you're always free to change your mind since your containers will run unchanged across both services.



Kick off your transformation.

Leading organizations have had great success using containers. This total-package solution helped them lower their infrastructure costs, create better applications in less time, and run those applications more effectively.

If you're interested in learning more about containers or getting started with Google Cloud, please [contact a Google Cloud specialist](#) today. We're here to help you make the most of containers and accelerate your innovation.

Explore three key resources to help you dive deeper into the world of containers:

- [What is Cloud Run](#) Learn how to use Cloud Run to run containers in Google's existing infrastructure.
- [GKE overview](#) Learn when and how to use Google Kubernetes Engine (GKE), a managed Kubernetes service to deploy and operate your containerized applications.
- [What are containers?](#) Get the essential primer on containers and how to best use them for your business.