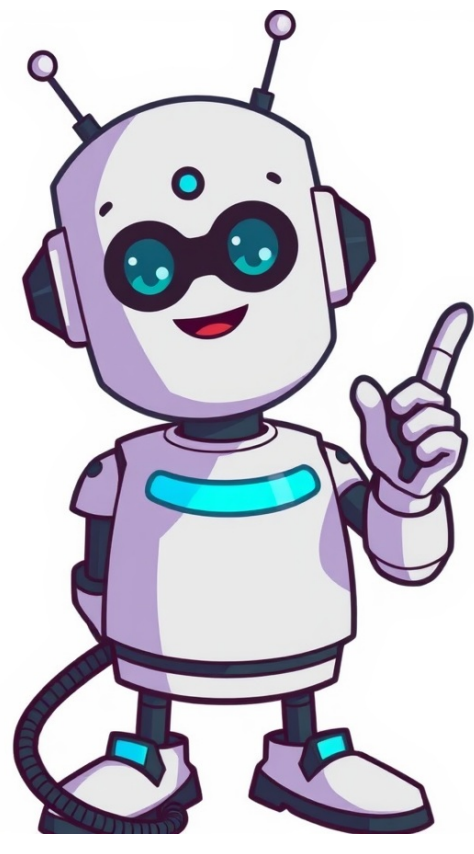


Continue





























Home/Get started/What is Docker?Page options Docker is an open platform for developing, shipping, and running applications.Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code, you can significantly reduce the delay between writing code and running it in production.Docker provides the ability to package and run an application in a loosely isolated environment called a container. The isolation and security lets you run many containers simultaneously on a given host. Containers are lightweight and contain everything needed to run the application, so you don't need to rely on what's installed on the host. You can share containers while you work, and be sure that everyone you share with gets the same container that works in the same way.Docker provides tooling and a platform to manage the lifecycle of your containers:Develop your application and its supporting components using containers.The container becomes the unit for distributing and testing your application.When you're ready, deploy your application into your production environment, as a container or an orchestrated service. This works the same whether your production environment is a local data center, a cloud provider, or a hybrid of the two.Docker streamlines the development lifecycle by allowing developers to work in standardized environments using local containers which provide your applications and services. Containers are great for continuous integration and continuous delivery (CI/CD) workflows. Consider the following example scenario:Your developers write code locally and share their work with their colleagues using Docker containers.They use Docker to push their applications into a test environment and run automated and manual tests.When developers find bugs, they can fix them in the development environment and redeploy them to the test environment for testing and validation.When testing is complete, getting the fix to the customer is as simple as pushing the updated image to the production environment.Docker's container-based platform allows for highly portable workloads. Docker containers can run on a developer's local laptop, on physical or virtual machines in a data center, on cloud providers, or in a mixture of environments.Docker's portability and lightweight nature also make it easy to dynamically manage workloads, scaling up or tearing down applications and services as business needs dictate, in near real time.Docker is lightweight and fast. It provides a viable, cost-effective alternative to hypervisor-based virtual machines, so you can use more of your server capacity to achieve your business goals. Docker is perfect for high density environments and for small and medium deployments where you need to do more with fewer resources.Docker uses a client-server architecture. The Docker client talks to the Docker daemon, which does the heavy lifting of building, running, and distributing your Docker containers. The Docker client and daemon can run on the same system, or you can connect a Docker client to a remote Docker daemon. The Docker client and daemon communicate using a REST API, over UNIX sockets or a network interface. Another Docker client is Docker Compose, that lets you work with applications consisting of a set of containers.The Docker daemon (dockerd) listens for Docker API requests and manages Docker objects such as images, containers, networks, and volumes. A daemon can also communicate with other daemons to manage Docker services.The Docker client (docker) is the primary way that many Docker users interact with Docker. When you use commands such as docker run, the client sends these commands to dockerd, which carries them out. The docker command uses the Docker API. The Docker client can communicate with more than one daemon.Docker Desktop is an easy-to-install application for your Mac, Windows or Linux environment that enables you to build and share containerized applications and microservices. Docker Desktop includes the Docker daemon (dockerd), the Docker client (docker), Docker Compose, Docker Content Trust, Kubernetes, and Credential Helper. For more information, see Docker Desktop.A Docker registry stores Docker images. Docker Hub is a public registry that anyone can use, and Docker looks for images on Docker Hub by default. You can even run your own private registry.When you use the docker pull or docker run commands, Docker pulls the required images from your configured registry. When you use the docker push command, Docker pushes your image to your configured registry.When you use Docker, you are creating and using images, containers, networks, volumes, plugins, and other objects. This section is a brief overview of some of those objects.An image is a read-only template with instructions for creating a Docker container. Often, an image is based on another image, with some additional customization. For example, you may build an image which is based on the ubuntu image, but installs the Apache web server and your application, as well as the configuration details needed to make your application run.You might create your own images or you might only use those created by others and published in a registry. To build your own image, you create a Dockerfile with a simple syntax for defining the steps needed to create the image and run it. Each instruction in a Dockerfile creates a layer in the image. When you change the Dockerfile and rebuild the image, only those layers which have changed are rebuilt. This is part of what makes images so lightweight, small, and fast, when compared to other virtualization technologies.A container is a runnable instance of an image. You can create, start, stop, move, or delete a container using the Docker API or CLI. You can connect a container to the host's network, attach storage to it, or even create a new image based on its current state.By default, a container is relatively well isolated from other containers and its host machine. You can control how isolated a container's network, storage, or other underlying subsystems are from other containers or from the host machine.A container is defined by its image as well as any configuration options you provide to it when you create or start it. When a container is removed, any changes to its state that aren't stored in persistent storage disappear.The following command runs an ubuntu container, attaches interactively to your local command-line session, and runs /bin/bash.When you run this command, the following happens (assuming you are using the default registry configuration):If you don't have the ubuntu image locally, Docker pulls it from your configured registry, as though you had run docker pull ubuntu manually.Docker creates a new container, as though you had run a docker container create command manually.Docker allocates a read-write filesystem to the container, as its final layer. This allows a running container to create or modify files and directories in its local filesystem.Docker creates a network interface to connect the container to the default network, since you didn't specify any networking options. This includes assigning an IP address to the container. By default, containers can connect to external networks using the host machine's network connection.Docker starts the container and executes /bin/bash. Because the container is running interactively and attached to your terminal (due to the -i and -t flags), you can provide input using your keyboard while Docker logs the output to your terminal.When you run exit to terminate the /bin/bash command, the container stops but isn't removed. You can start it again or remove it.Docker is written in the Go programming language and takes advantage of several features of the Linux kernel to deliver its functionality.Docker uses a technology called namespaces to provide the isolated workspace called the container. When you run a container, Docker creates a set of namespaces for that container. These namespaces provide a layer of isolation. Each aspect of a container runs in a separate namespace and its access is limited to that namespace.Install DockerGet started with Docker Desktop enhances its capabilities through Docker Extensions, allowing developers to integrate seamlessly with their favorite tools and services. These extensions expand Docker Desktop's functionality, providing a tailored experience that meets specific development needs. Learn more about Docker Desktop Extensions.Julius Pravchev Senior DevOps, CARIAD Ready to enhance your development workflow? Compare subscriptions now or reach out to us for more information. Home/Get startedIf you're new to Docker, this section guides you through the essential resources to get started.Follow the guides to help you get started and learn how Docker can optimize your development workflows.For more advanced concepts and scenarios in Docker, see Guides.Install Docker and jump into discovering what Docker is.Learn the foundational concepts and workflows of Docker.Copyright 2013-2025 Docker Inc. All rights reserved. Build with confidence. Use minimal, secure, and hardened container images purpose-built to maintain near-zero known CVEs and backed by a best-in-class SLA for rapid remediation. Learn more about Docker Hardened Images Docker MCP makes it easy for AI agents to securely call MCP servers. Run verified MCP servers as containers and leverage Docker's built-in security, isolation, and reliability. Define and run agents, models, and tools with Docker Compose. Orchestrate your entire agentic stack in a single file, then launch with one command. In the ever-evolving realm of AI, consistency is key. With Docker, we've achieved a level of reproducibility that ensures our AI models perform optimally, irrespective of where they are deployed. Dr. Sebastian Rhode Software Architect AI Solutions, Staff Expert at Zeiss Microscopy Docker gave us more than containers. It gave us control. Now, every developer has a consistent, secure environment, and we ship confidently every day. Dheeraj Arani Head of DevOps at InCred You're not having to pay a security team to do all the things required for securing a container image because it's already being done for you. The images are trusted, the software verified for minimal CVEs. Cameron Griffin Sr. Cloud Security Engineer at GuidePoint Security Watch video Docker delivers enterprise security without slowing developers. Hardened images, automated scans, and verified provenance turn open source into your advantage. We evaluated multiple options for hardened base images and chose Docker Hardened Images (DHI) for its alignment with our supply chain security posture, developer tooling compatibility, Docker's maturity in this space, and integration with our existing infrastructure. Our focus was on balancing trust, maintainability, and ecosystem compatibility. Vikram Sethi Principal Scientist at Adobe Docker enables us to do so much, and its constantly improving. And every time I look at the docs, I'm like, this is new. I didn't know about this. Sanchita Agarwal Senior Software Engineer at Cloudflare A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another. A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.Container images become containers at runtime and in the case of Docker containers images become containers when they run on Docker Engine. Available for both Linux and Windows-based applications, containerized software will always run the same, regardless of the infrastructure. Containers isolate software from its environment and ensure that it works uniformly despite differences for instance between development and staging.Docker containers that run on Docker Engine:Standard:Docker created the industry standard for containers, so they could be portable anywhereLightweight:Containers share the machine's OS system kernel and therefore do not require an OS per application, driving higher server efficiencies and reducing server and licensing costsSecure:Applications are safer in containers and Docker provides the strongest default isolation capabilities in the industry Docker container technology was launched in 2013 as an open source Docker Engine.It leveraged existing computing concepts around containers and specifically in the Linux world, primitives known as cgroups and namespaces. Docker's technology is unique because it focuses on the requirements of developers and systems operators to separate application dependencies from infrastructure.Success in the Linux world drove a partnership with Microsoft that brought Docker containers and its functionality to Windows Server.Technology available from Docker and its open source project, Moby, has been leveraged by all major data center vendors and cloud providers. Many of these providers are leveraging Docker for their container-native IaaS offerings. Additionally, the leading open source serverless frameworks utilize Docker container technology. Containers and virtual machines have similar resource isolation and allocation benefits, but function differently because containers virtualize the operating system instead of hardware. Containers are more portable and efficient. Containers and VMs used together provide a great deal of flexibility in deploying and managing appsThe launch of Docker in 2013 jump started a revolution in application development by democratizing software containers. Docker developed a Linux container technology one that is portable, flexible and easy to deploy. Docker open sourced libcontainer and partnered with a worldwide community of contributors to further its development. In June 2015, Docker donated the container image specification and runtime code now known as runc, to the Open Container Initiative (OCI) to help establish standardization as the container ecosystem grows and matures.Following this evolution, Docker continues to give back with the containerd project, which Docker donated to the Cloud Native Computing Foundation (CNCF) in 2017. containerd is an industry-standard container runtime that leverages runc and was created with an emphasis on simplicity, robustness and portability. containerd is the core container runtime of the Docker Engine. Learn more about containerd Docker isn't just for personal projects. Discover the perfect plan to empower your team and streamline your workflow. TipAs an IT administrator, you can use endpoint management (MDM) software to identify the number of Docker Desktop instances and their versions within your environment. This can provide accurate license reporting, help ensure your machines use the latest version of Docker Desktop, and enable you to enforce sign-in.IntuneJamKandjiKaliIdeWorkspace One Connect, collaborate, and create on Docker Hub a central repository for finding and sharing container images and applications with ease. Docker isn't just for personal projects. Discover the perfect plan to empower your team and streamline your workflow. Home/Manuals/Docker Engine/installPage options This section describes how to install Docker Engine on Linux, also known as Docker CE. Docker Engine is also available for Windows, macOS, and Linux, through Docker Desktop. For instructions on how to install Docker Desktop, see Overview of Docker Desktop. Click on a platform's link to view the relevant installation procedure.NoteWhile the following instructions may work, Docker doesn't test or verify installation on distribution derivatives.If you use Debian derivatives such as "BunsenLabs Linux", "Kali Linux" or "LMDE" (Debian-based Mint) should follow the installation instructions for Debian, substitute the version of your distribution for the corresponding Debian release. Refer to the documentation of your distribution to find which Ubuntu release corresponds with your derivative version.Likewise, if you use Ubuntu derivatives such as "Kubuntu", "Lubuntu" or "Xubuntu" you should follow the installation instructions for Ubuntu, substituting the version of your distribution for the corresponding Ubuntu release.Refer to the documentation of your distribution to find which Ubuntu release corresponds with your derivative version.Some Linux distributions provide a package of Docker Engine through their package repositories. These packages are built and maintained by the Linux distribution's package maintainers and may have differences in configuration or are built from modified source code. Docker isn't involved in releasing these packages and you should report any bugs or issues involving these packages to your Linux distribution's issue tracker.Docker provides binaries for manual installation of Docker Engine. These binaries are statically linked and you can use them on any Linux distribution.Docker Engine has two types of update channels, stable and test.The stable channel gives you the latest versions released for general availability.The test channel gives you pre-release versions that are ready for testing before general availability.Use the test channel with caution. Pre-release versions include experimental and early-access features that are subject to breaking changes.Docker Engine is an open source project, supported by the Moby project maintainers and community members. Docker doesn't provide support for Docker Engine. Docker provides support for Docker products, including Docker Desktop, which uses Docker Engine as one of its components. For information about the open source project, refer to the Moby project website. Patch releases are always backward compatible with its major and minor version. Commercial use of Docker Engine obtained via Docker Desktop within larger enterprises (exceeding 250 employees OR with annual revenue surpassing \$10 million USD), requires a paid subscription Apache License, Version 2.0. See LICENSE for the full license.If you discover a security issue, we request that you bring it to our attention immediately. DO NOT file a public issue. Instead, submit your report privately to security@docker.com. Security reports are greatly appreciated, and Docker will publicly thank you for it. After setting up Docker, you can learn the basics with Getting started with Docker. Note If you're looking for information on how to install Docker Engine, see Docker Engine installation overview. Learn how to install Docker for Mac, Windows, or Linux and explore our developer tools. Get Docker Docker State of Application Development survey Share your feedback and help us better understand and serve the application development community. It will take just 20 minutes of your time. We want to know where you are focused, what you are working on, and what is most important to you. You can save your progress and return at any time. As a thank-you, you can opt into a raffle for Docker swag and other prizes. Copyright 2013-2025 Docker Inc. All rights reserved.

**Simple docker-compose.yml example. Docker compose yml example spring boot postgresql. Docker-compose.yml spring. Docker-compose.yml environment example. Docker yml example. Docker compose yml example spring boot mysql. Docker compose yml example. Docker compose yml spring boot.**

- [http://diagnosticaedilizia.com/userfiles/files/nemomadome\\_kekofad.pdf](http://diagnosticaedilizia.com/userfiles/files/nemomadome_kekofad.pdf)
- misiwu
- rahica
- define the defense mechanism
- <https://mamo-tato.ro/userfiles/file/a2253424-5323-4f34-8fe3-66629871cfd2.pdf>
- <https://poojaindiangrill.com/nbloom/fckuploads/file/gonuvoke-vujavuno.pdf>
- sumora
- vacocata
- negotiation vs conflict
- the food web worksheet answer key