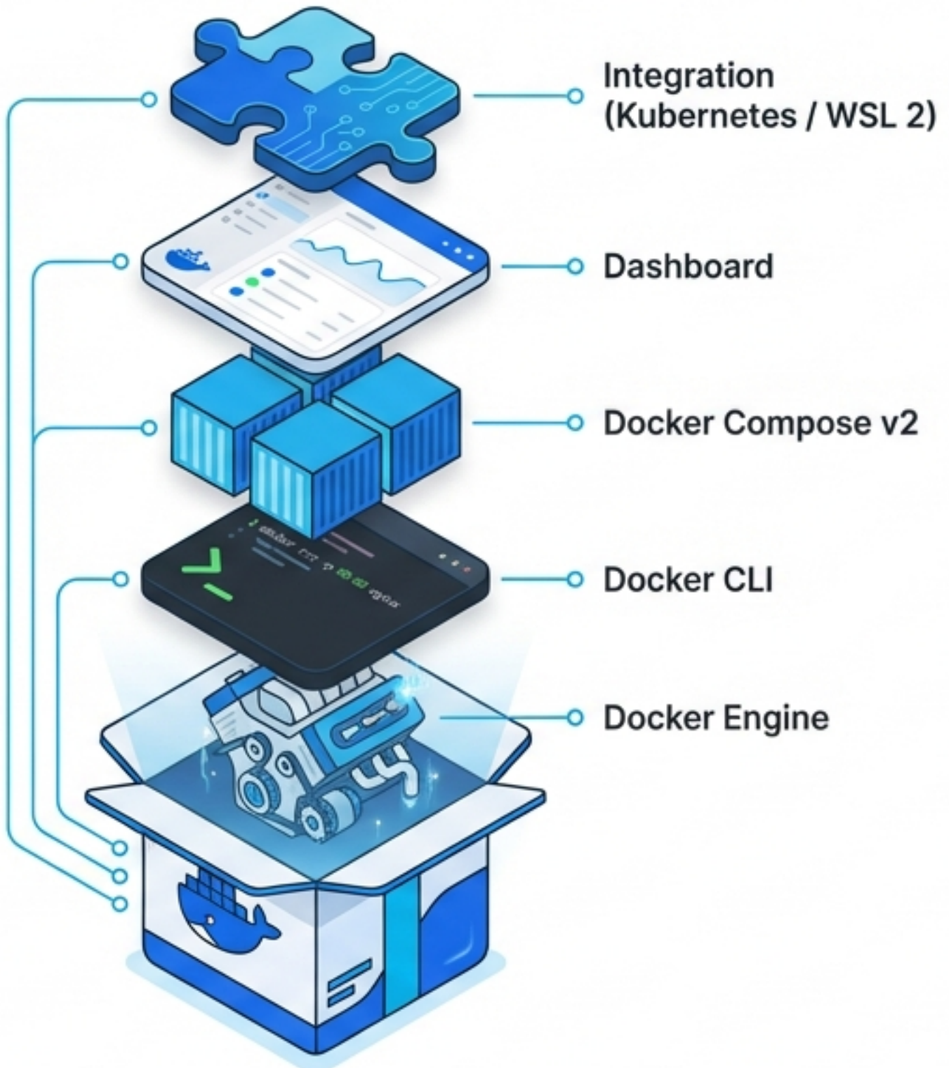
Ready, Set, Docker

A Zero-to-Hero Guide

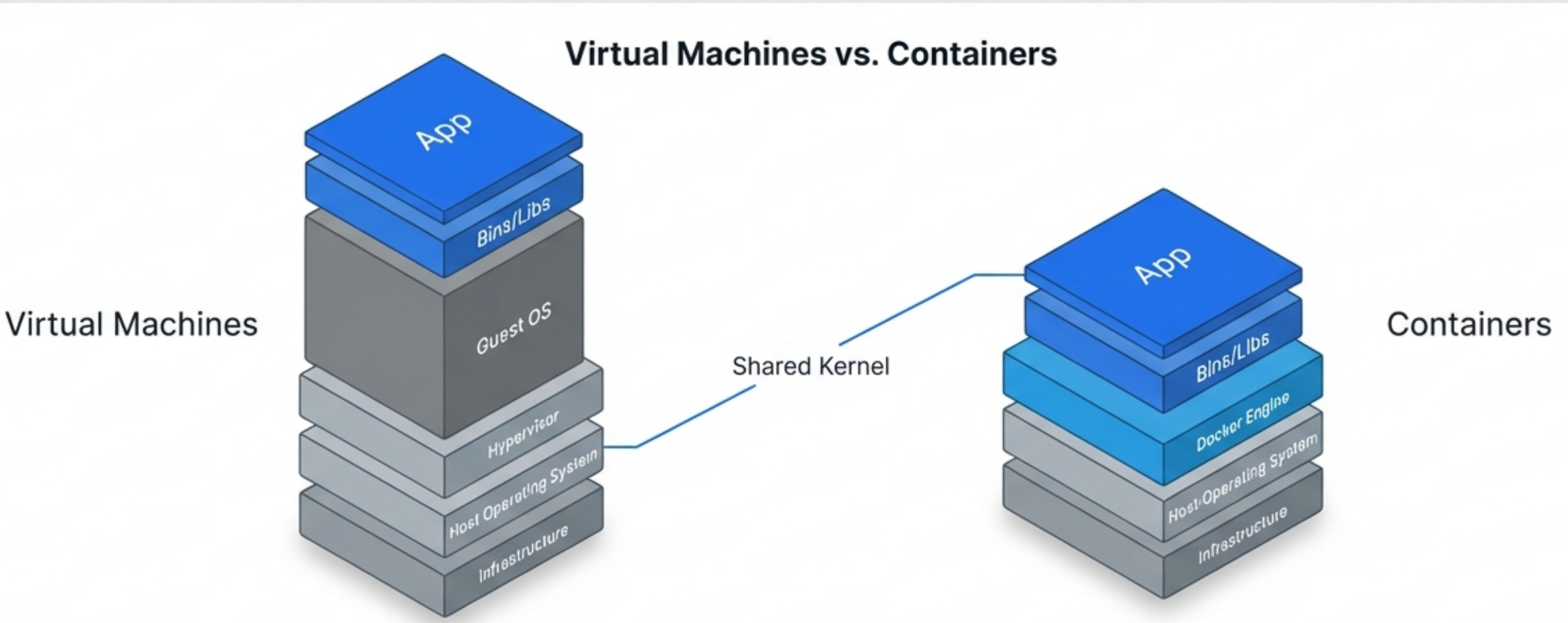
Installation, configuration, and core concepts for macOS and Windows.

Everything you need in one package

Docker Desktop is the easiest way to run containers locally. It bundles the entire development ecosystem into a single point-and-click installation.



Maximum efficiency, minimum overhead



Shared Kernel

Uses Linux namespaces to share the host OS kernel. No heavy Guest OS.

Isolation

Each container runs in a private ‘box’, separating files and networks.

Images vs. Containers

Images are read-only templates. Containers are writable instances.

Check your system before you start

General Requirements: 4 GB RAM minimum • Virtualization enabled in BIOS

macOS Prerequisites

- **OS:** Current release or previous two versions.
- **Architecture:** Supports Intel & Apple Silicon (M-series).
- **Access:** Admin privileges required.

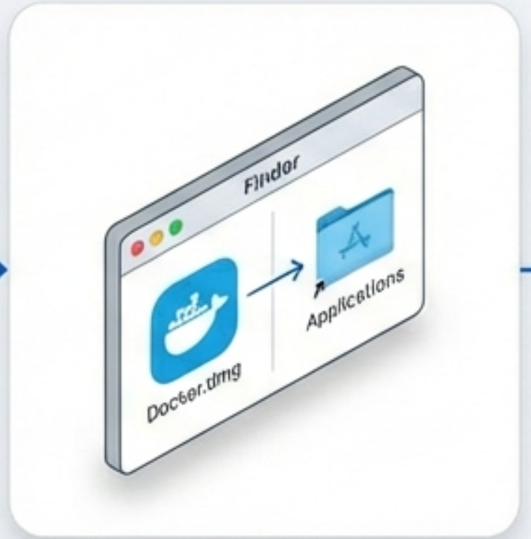
Windows Prerequisites

- **OS:** Windows 10/11 (64-bit).
- **Backend:** WSL 2 (Recommended) or Hyper-V.
- **Access:** Admin privileges required.

Installation path for macOS users

Download

Select 'Mac with Apple Chip' or 'Mac with Intel Chip' on the Docker product page.



Install

Open Docker.dmg and drag the icon to Applications.

Authorize

Launch Docker. Grant system privileges and enter password.

Wait for the whale.

Installation path for Windows users

Download

Get ‘Docker Desktop Installer.exe’.

Install & Config

Run installer. Ensure ‘Use WSL 2 instead of Hyper-V’ is CHECKED.

Restart

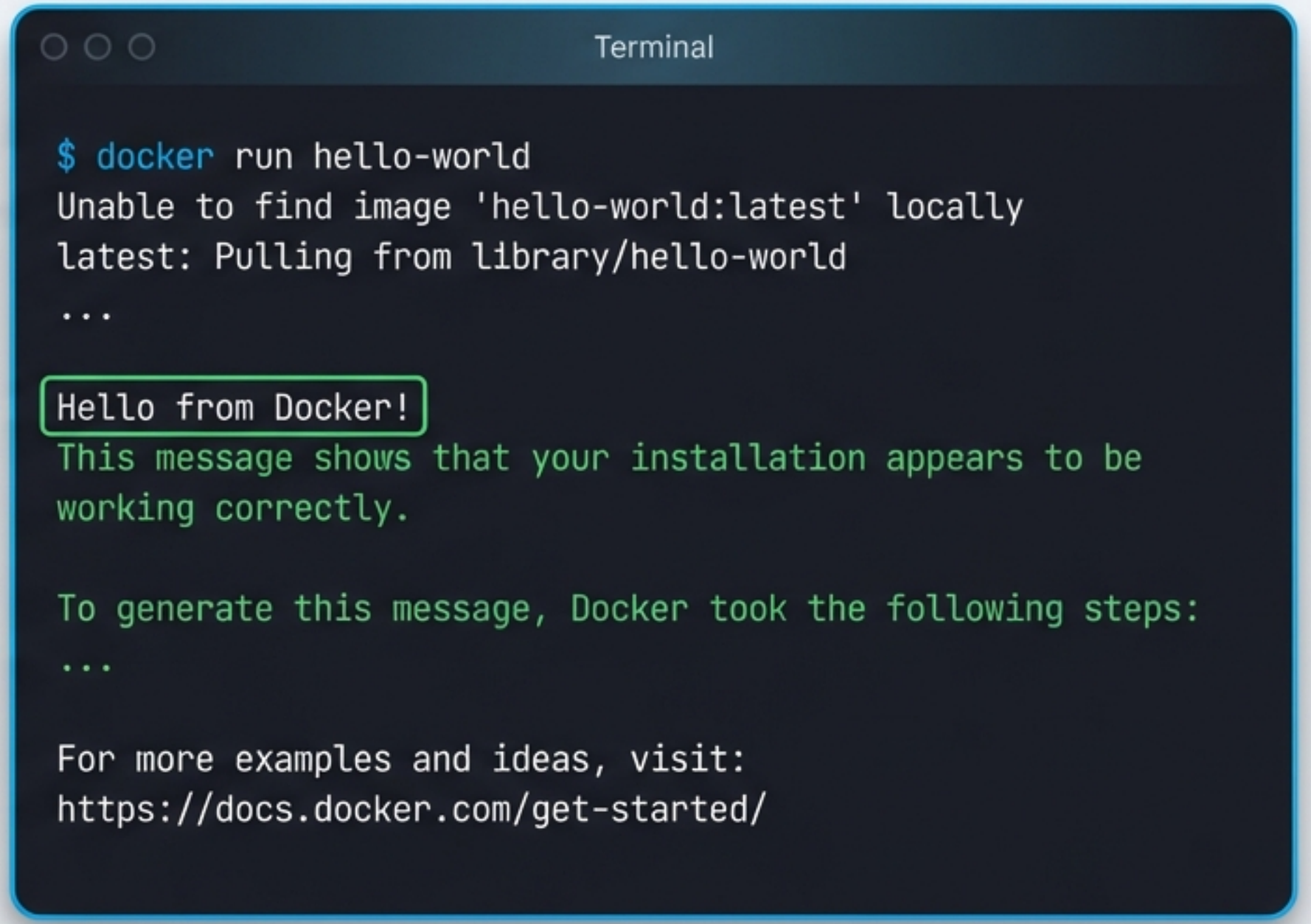
Reboot Windows to apply kernel/virtualization changes.

Pro Tip

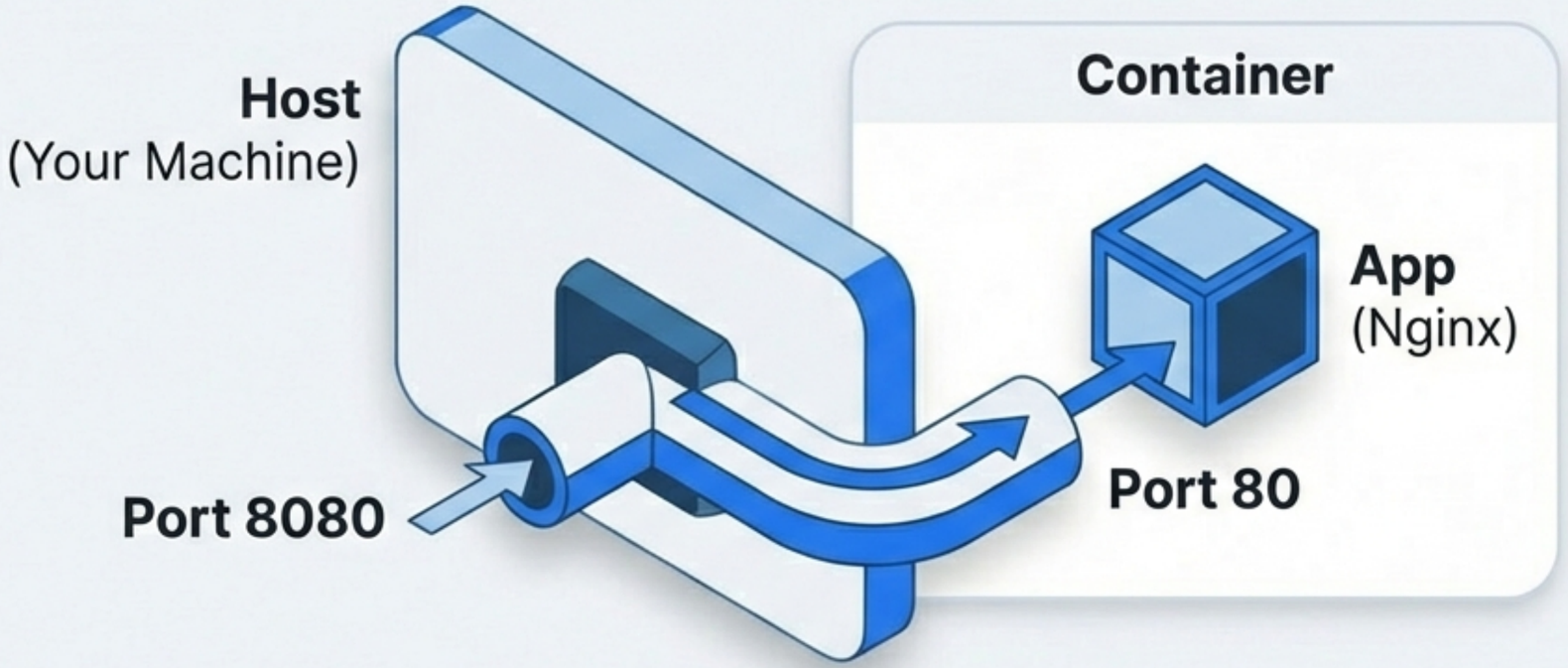
Permissions: Add your user account to the ‘docker-users’ group to manage Docker without admin rights.

Verifying success with Hello-World

1. Client contacts Daemon.
2. Daemon pulls ‘hello-world’ image from Hub.
3. Daemon creates new container.
4. Container runs and streams output.

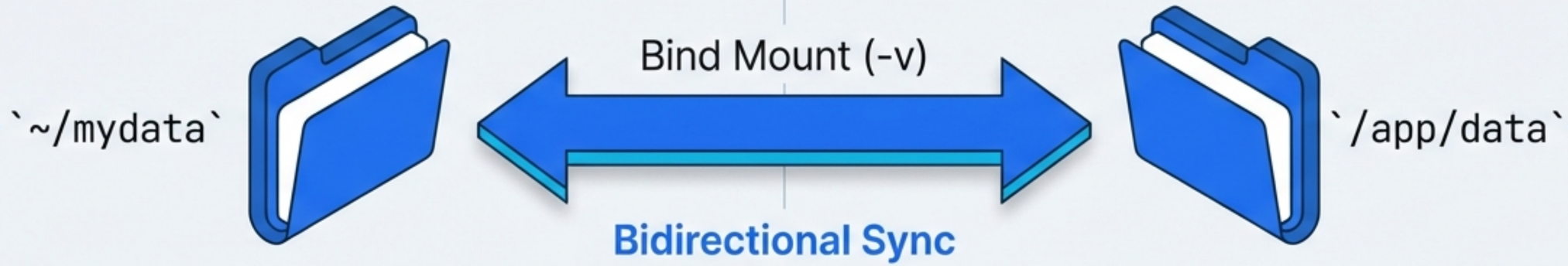


Bridging the gap with Port Mapping



Command: `docker run -p 8080:80 nginx`
Result: Accessing `localhost:8080 ` in your browser hits port 80 inside the container.

Persisting data with Volumes



The Problem: Containers are ephemeral. Data dies when they stop.

The Solution: Volumes map a host folder to a container folder, keeping data safe.

```
docker run -v ~/mydata:/app/data myimage
```

Three ways to run a container

Interactive

Runs in the foreground like a local program.

```
docker run -it [image]
```

Best for:
Exploration, quick scripts.

Detached

Runs in background as a service.

```
docker run -d [image]
```

Best for:
Databases, backend services.

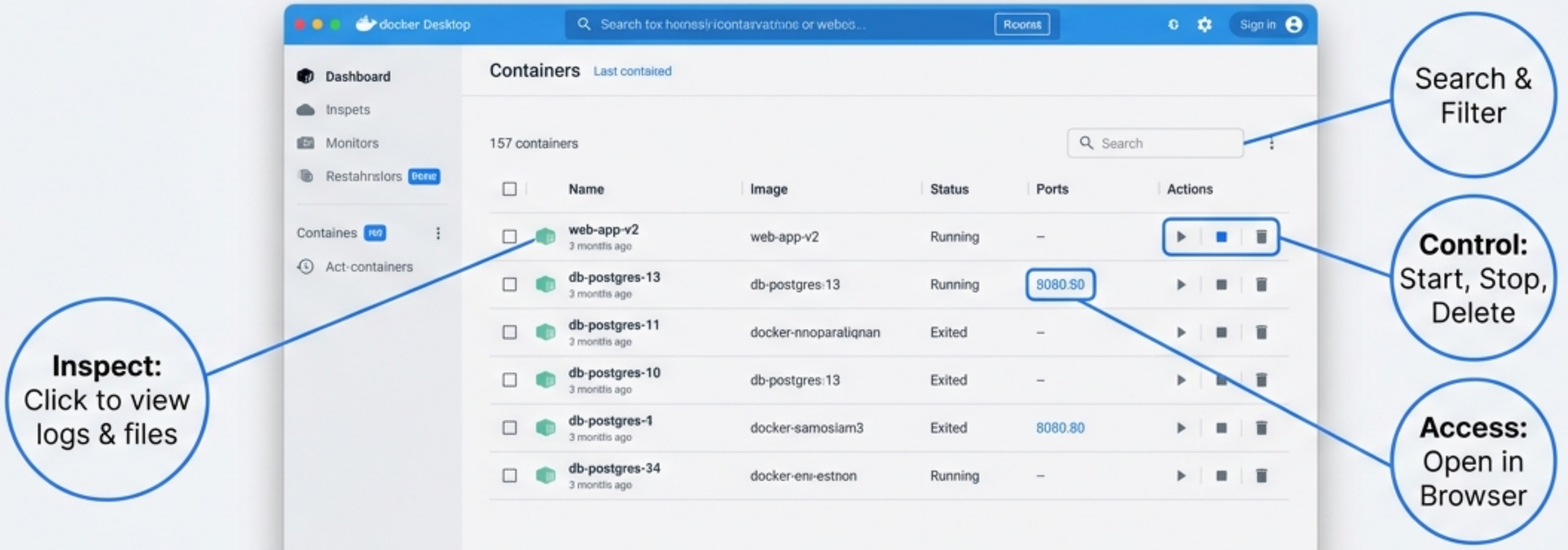
Web Server

Background service accessed via browser.

```
Map ports (-p) and visit localhost.
```

Best for:
Web apps (Nginx, Apache).

Managing coing containers without code



The Dashboard provides a full visual management interface for when you don’t want to use the CLI.

The essential CLI toolkit

```
docker pull [image]
```

Downloads an image from a registry (like Docker Hub) to your machine.

```
docker run [options] [image]
```

Creates and starts a container. The most frequently used command.

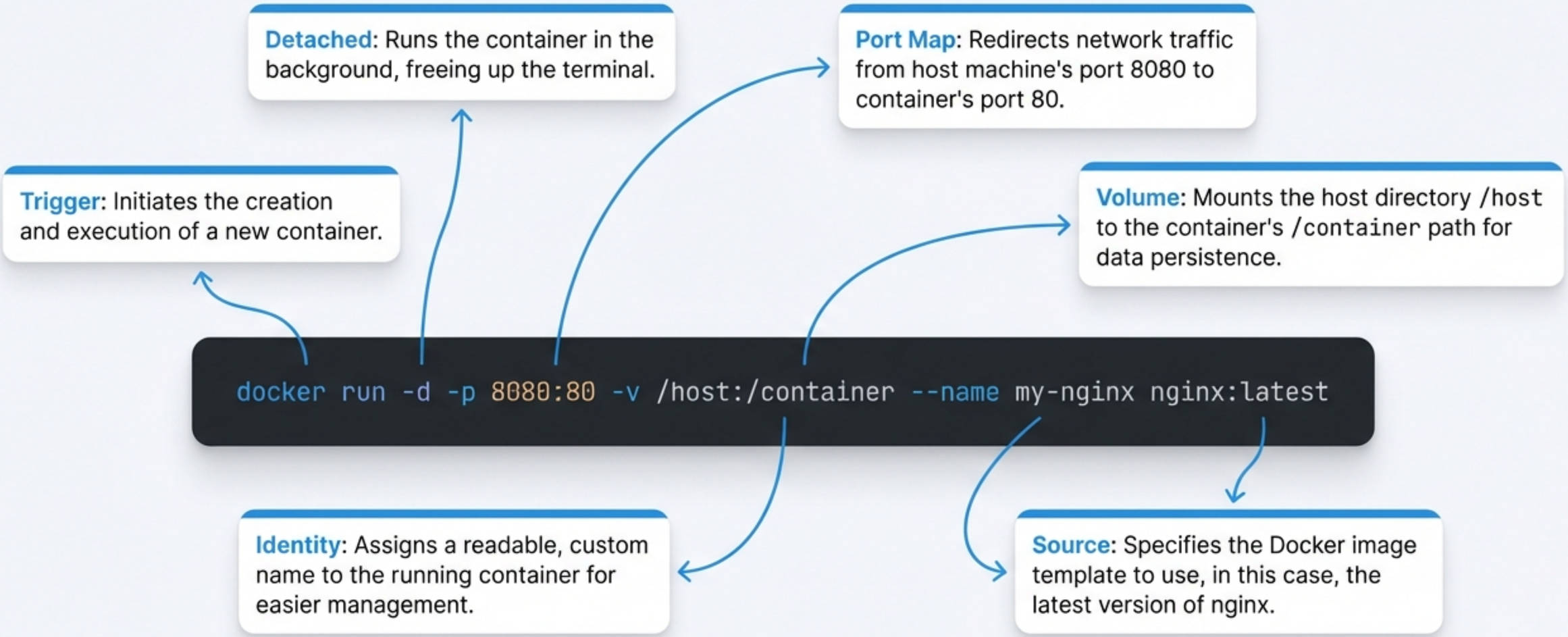
```
docker ps
```

Lists all currently running containers. Add `-a` to see stopped ones.

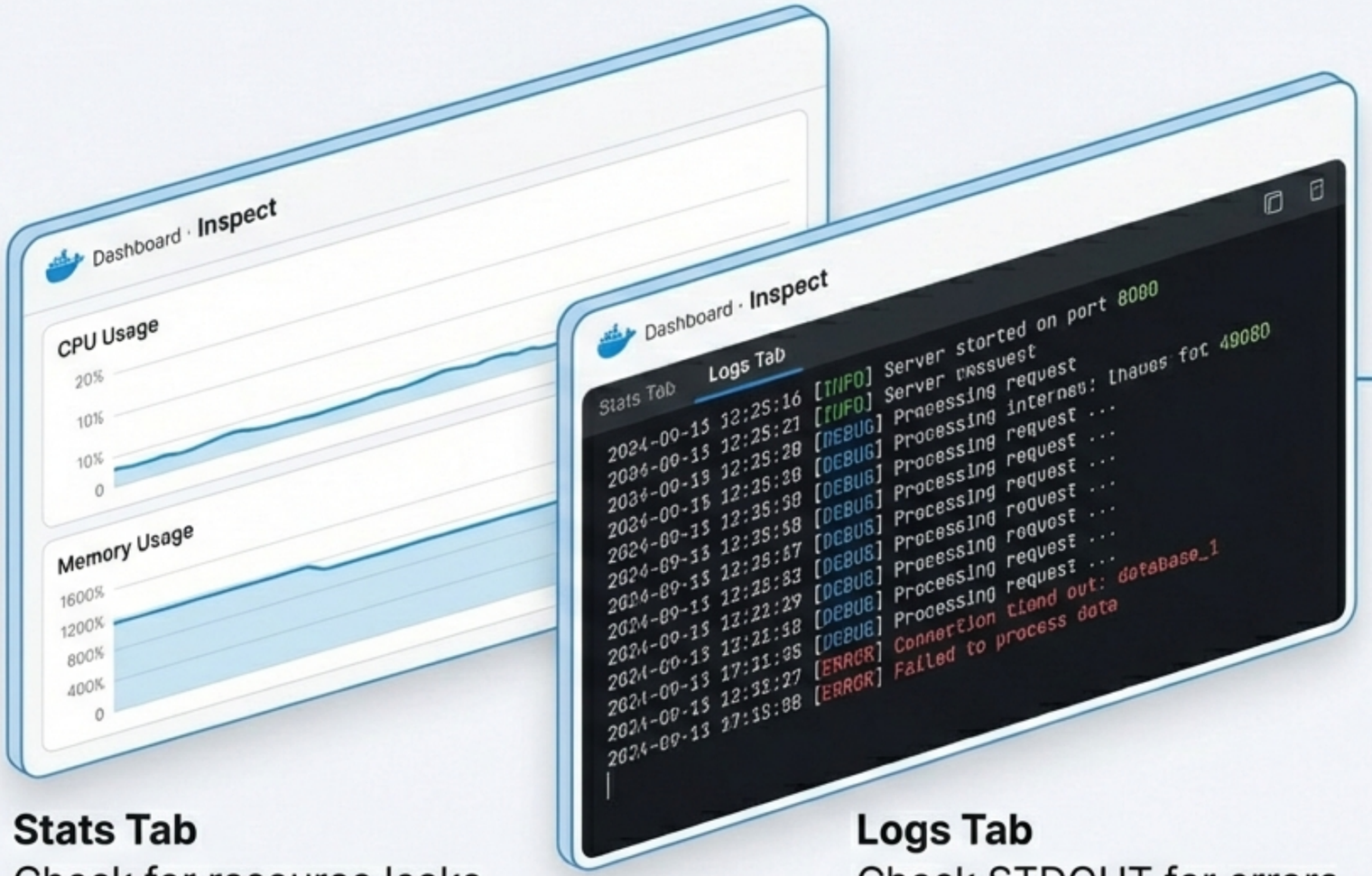
```
docker stop [id] / docker rm [id]
```

Halts execution and removes the container instance to clean up.

Deconstructing a complex command



Monitoring performance and troubleshooting



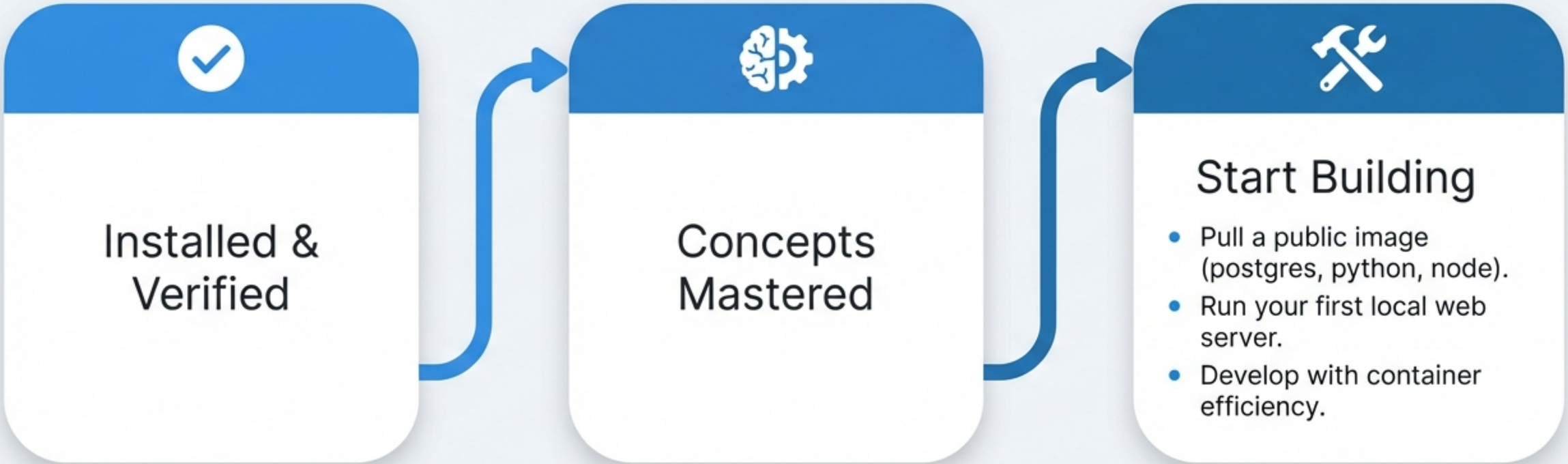
Stats Tab
Check for resource leaks.

Logs Tab
Check STDOUT for errors.

```
$ docker logs my-container
```

View the same logs from the terminal.

Your local environment is ready



Go build something amazing.