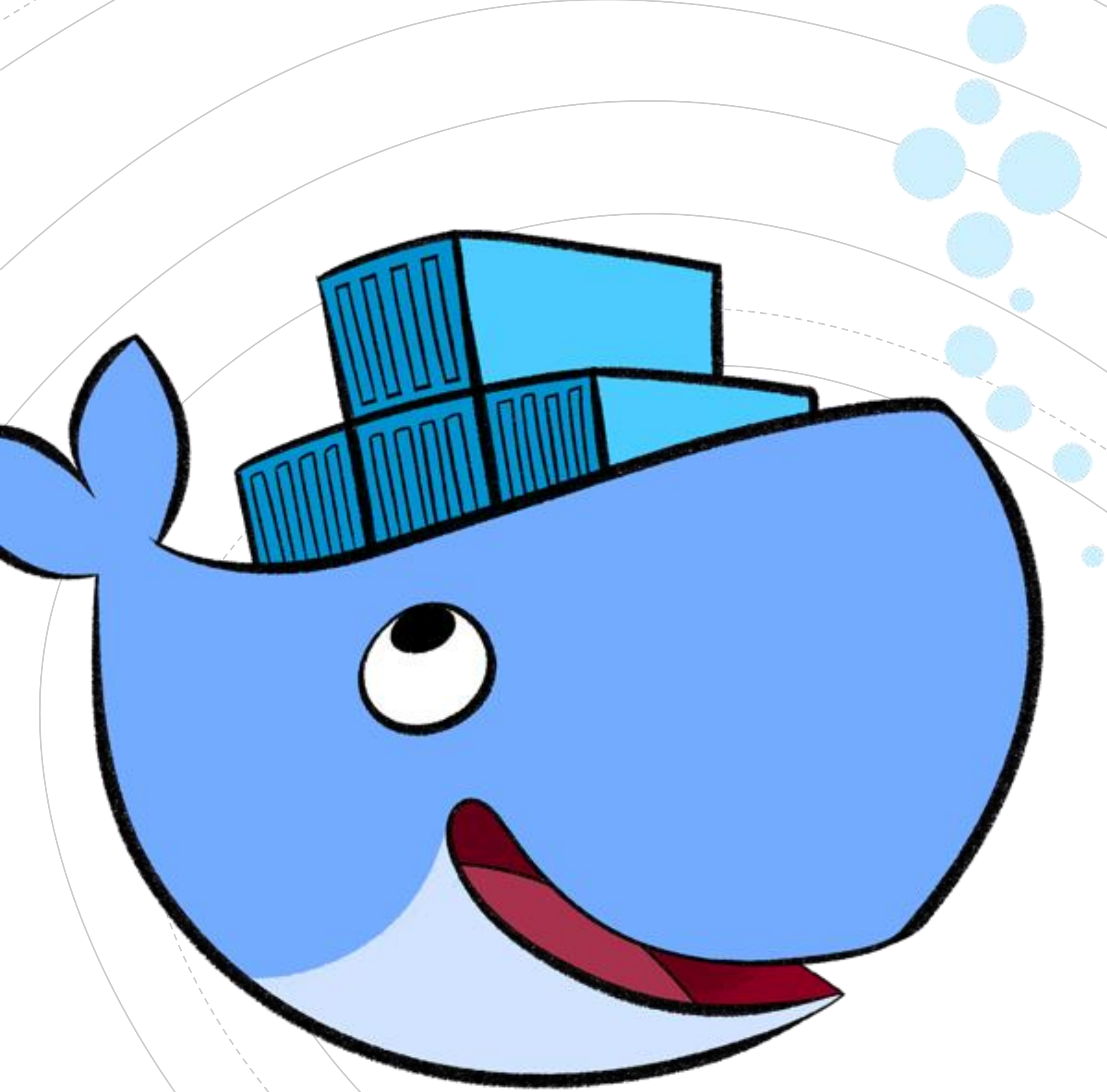
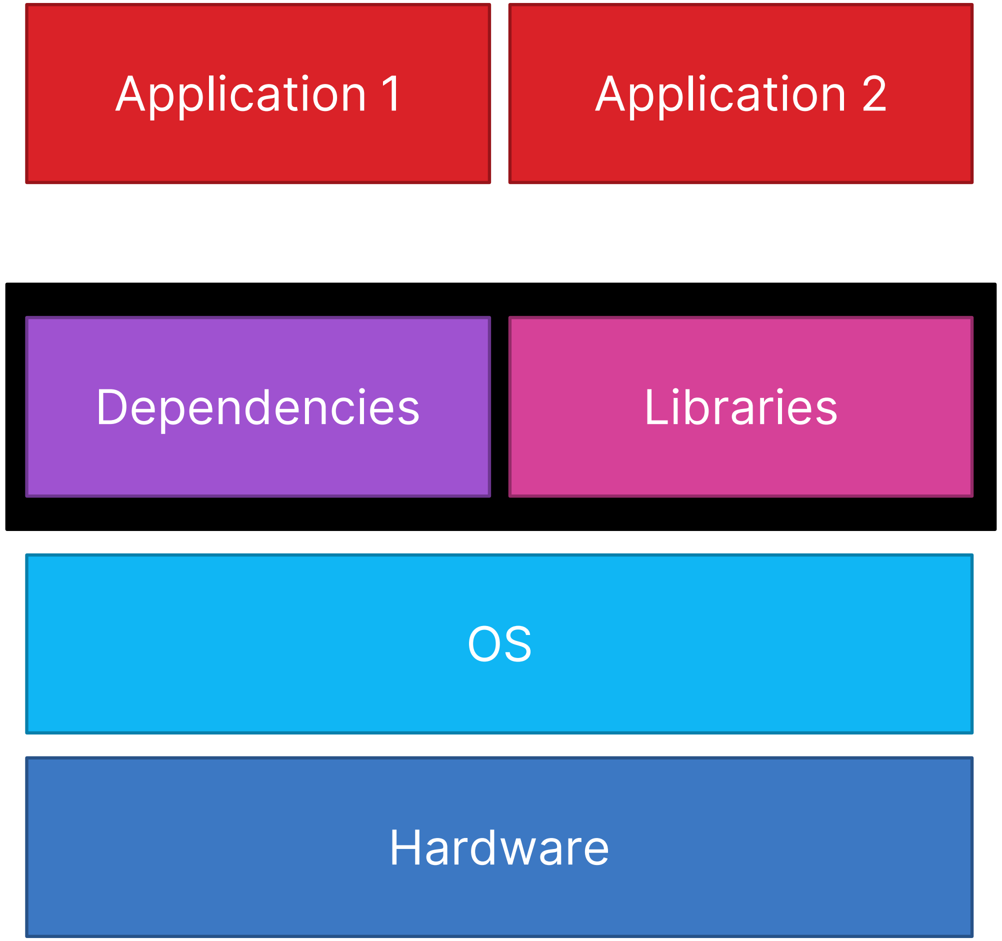


An Intro to Docker

The Problem


Application 1

Application 2

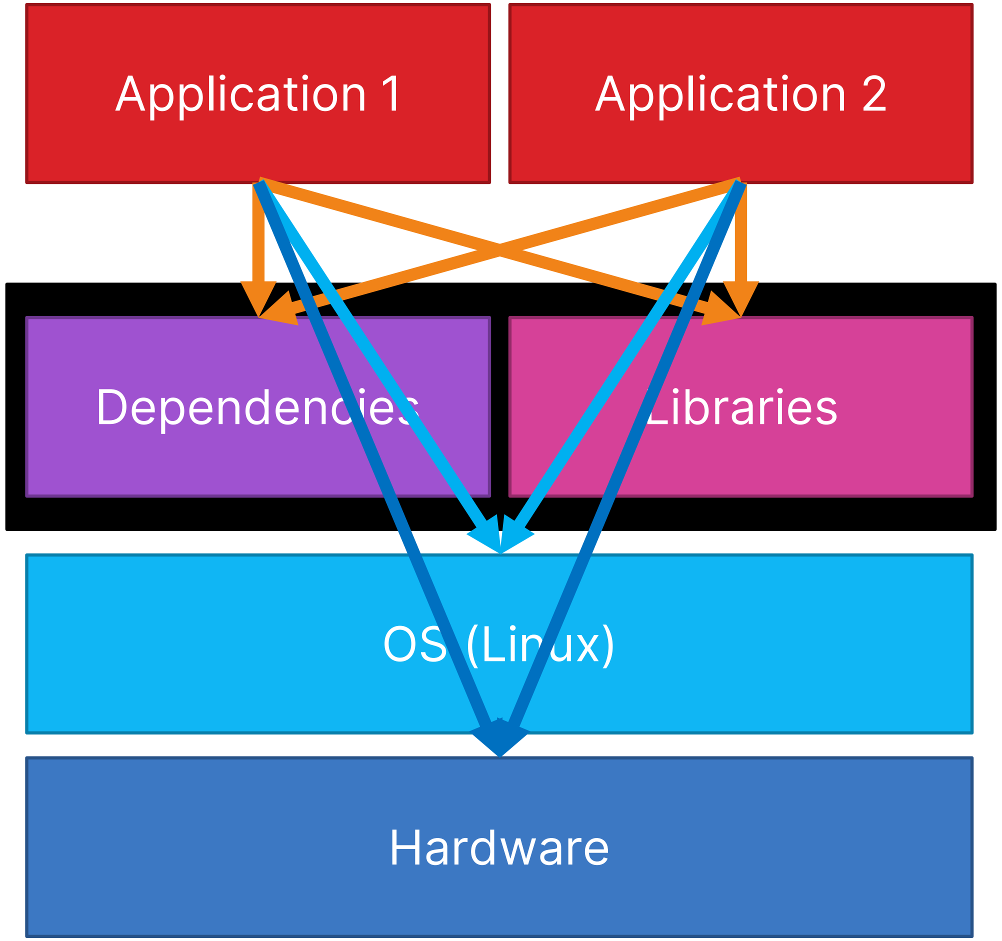
Dependencies

Libraries

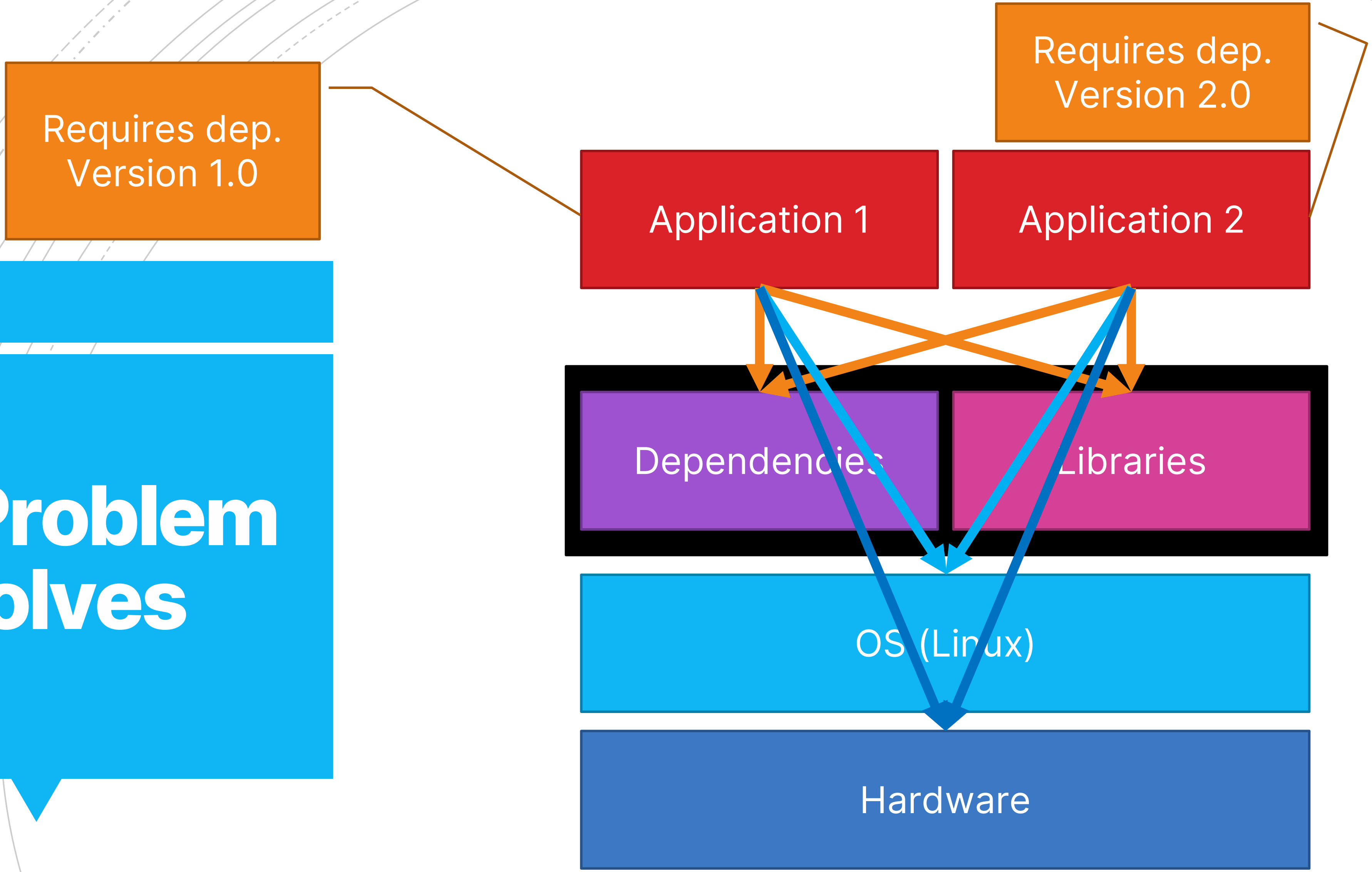
OS

Hardware

The Problem Evolves



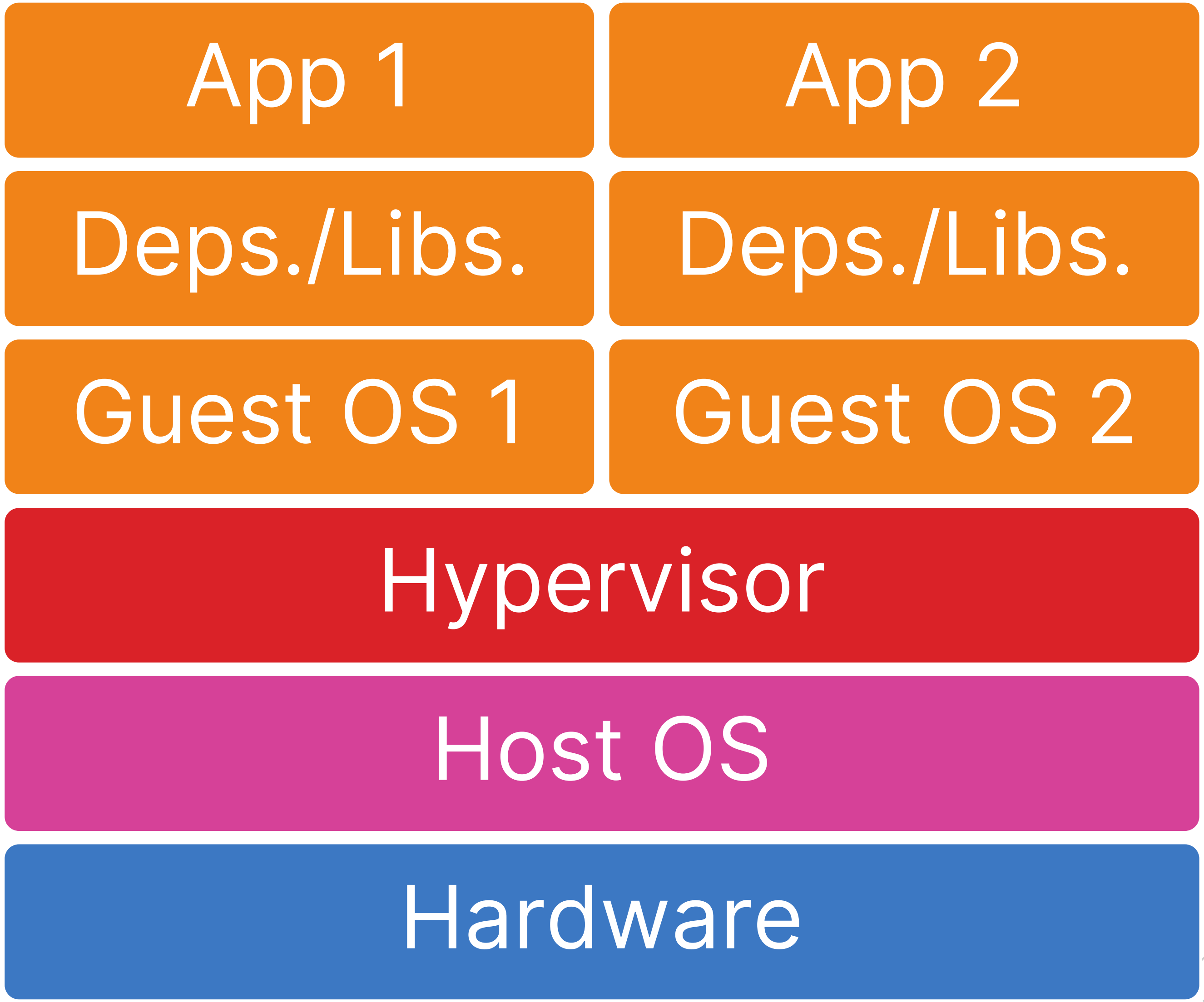
The Problem Evolves



THIS IS FINE.

The Problem ****ing Evolves

Virtual Machines



App 1

App 2

Deps./Libs.

Deps./Libs.

Guest OS 1

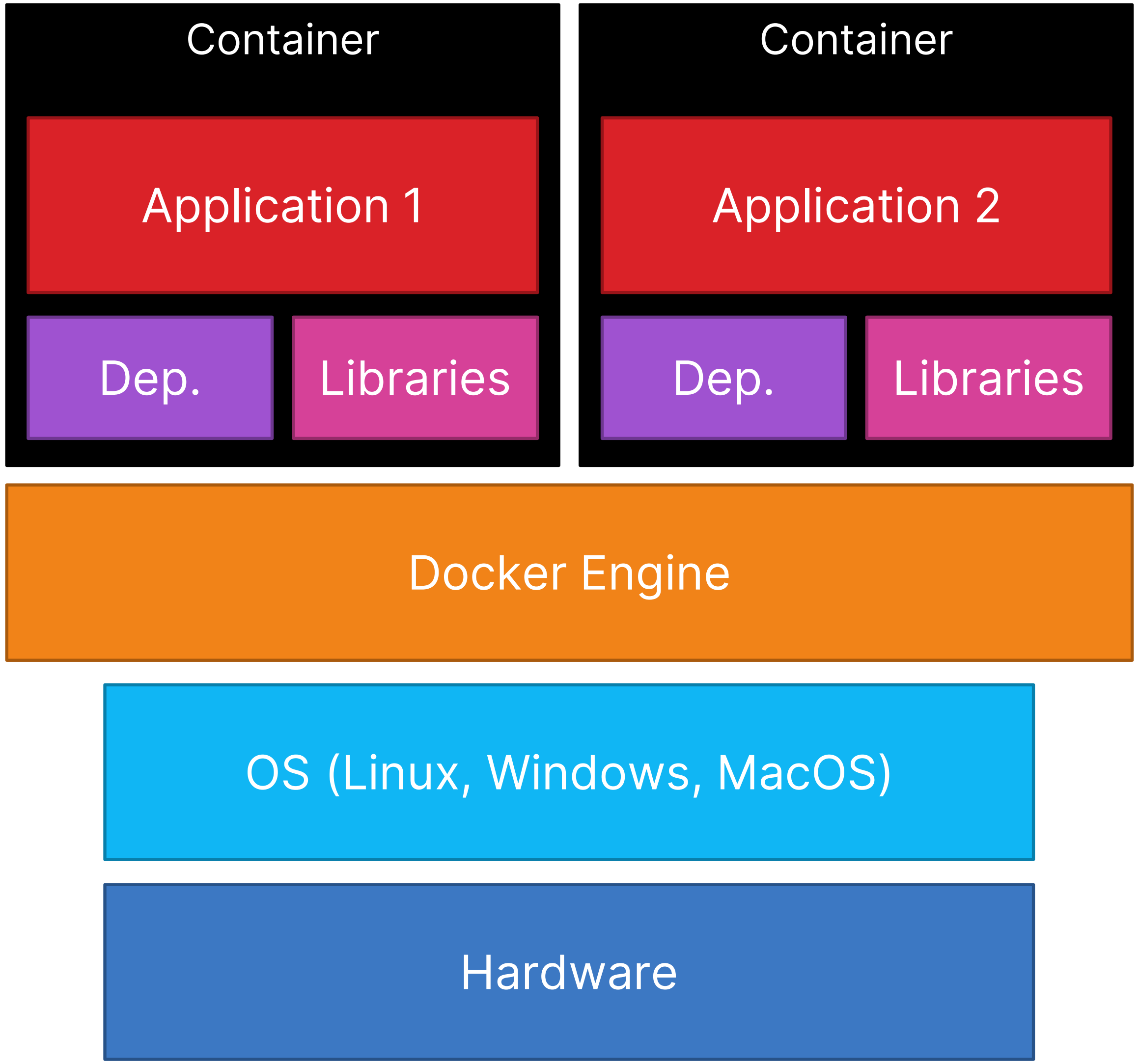
Guest OS 2

Hypervisor

Host OS

Hardware

The Proper Solution



Container

Application 1

Dep.

Libraries

Container

Application 2

Dep.

Libraries

Docker Engine

OS (Linux, Windows, MacOS)

Hardware

~~Docker == Virtualization~~

When To Use

- For **local development/testing**
- For **CI/CD**
- For **production** (Docker Swarm)
- For **multiple sub-projects**
- For **scaling**
- When you **depend on third-party products** (databases, etc.)
- When you target **many different OS**
- When you want to **decouple from dependencies**
- When you want to **save time**
- **Many more…**

Terminology

- Registry
- Image
  - e.g. nginx, node, mongo, etc.
- Dockerfile
- Container
- Volumes
- Networks

Hands on
Custom Image

- Download and install Docker Desktop on your machines
  - Linux users, go to get.docker.com
  - Make sure you enabled virtualization before
- Navigate to https://github.com/ginomessmer/intro-to-docker
- Refer to README and start hacking

Cheers

Further Links & Steps

- Docker Introduction — What You Need To Know To Start Creating Containers: https://medium.com/@kelvin_sp/8ffaf064930a
- A Beginner-Friendly Introduction to Containers, VMs and Docker: https://www.freecodecamp.org/news/a-beginner-friendly-introduction-to-containers-vms-and-docker-79a9e3e119b/
- What is Docker? https://docs.microsoft.com/en-us/dotnet/architecture/microservices/container-docker-introduction/docker-defined
- *Add Docker to your CI/CD environment*
- *Docker Compose*
- *Docker Stack*