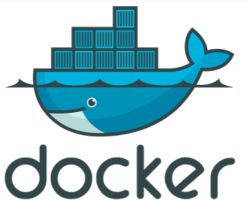
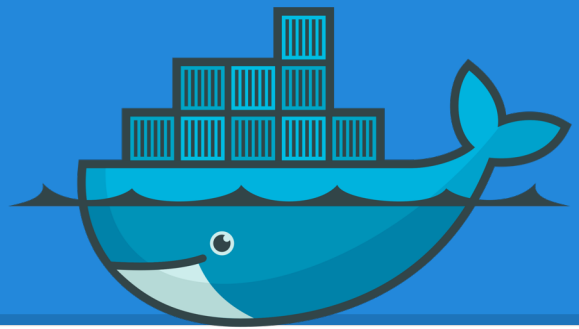
Docker

Workshop



Docker Overview



docker

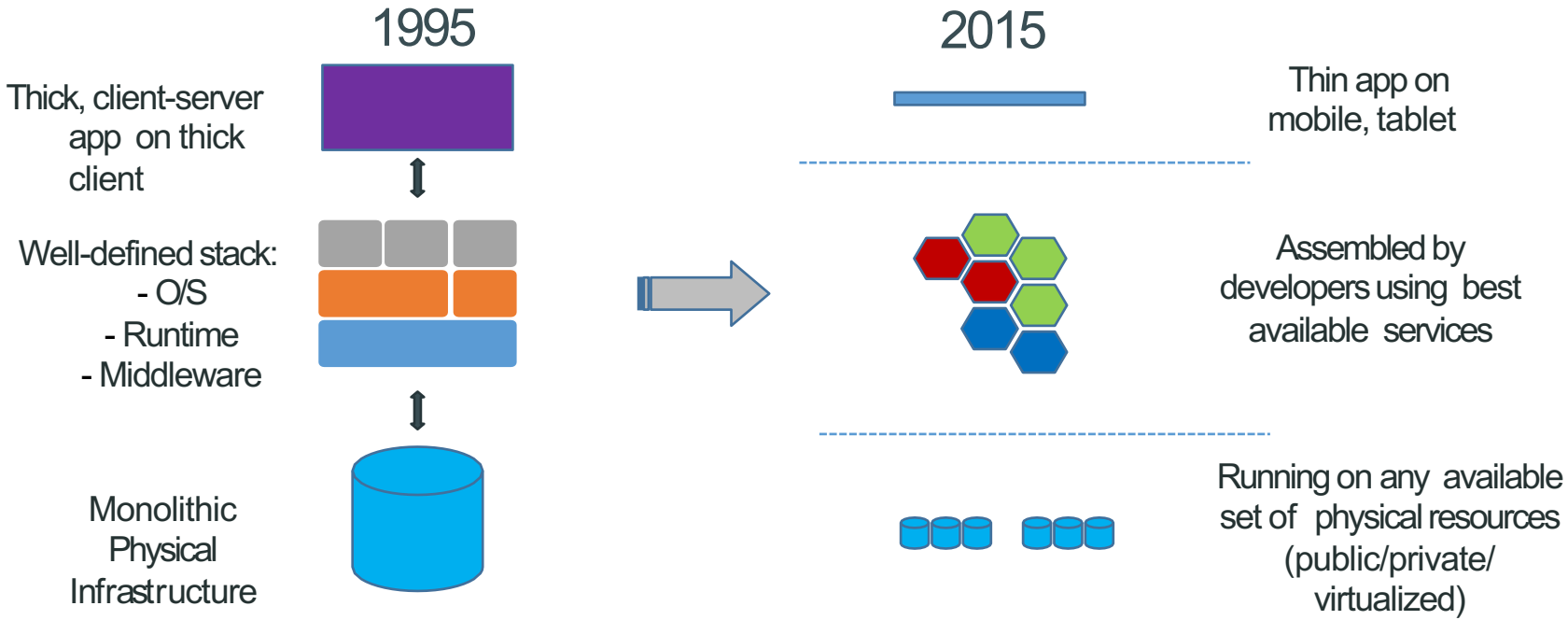
An Quick Introduction

What is Docker?

Docker is an open-source project that automates the deployment of applications inside software containers, by providing an additional layer of abstraction and automation of operating system–level virtualization on Linux.

[Source: en.wikipedia.org]

Market View: Evolution of IT

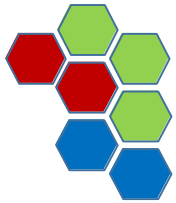


Challenges

Thin app on mobile, tablet

2015

Assembled by developers using best available services



How to ensure services interact consistently, avoid dependency hell

How to avoid n Xn different configs

Running on any available set of physical resources (public/private/ virtualized)

How to migrate & scale quickly, ensure compatibility

The Challenge

Multiplicity of Stacks

Static website

nginx 1.5 + modsecurity + openssl + bootstrap 2

User DB

postgresql + pgv8 + v8

Queue

Redis + redis-sentinel

Analytics DB

hadoop + hive + thrift + OpenJDK

Background workers

Python 3.0 + celery + pyredis + libcurl + ffmpeg + libopencv + nodejs + phantomjs

Web frontend

Ruby + Rails + sass + Unicorn

API endpoint

Python 2.7 + Flask + pyredis + celery + psycopg + postgresql-client

Do services and apps interact appropriately?

Multiplicity of hardware environmen

Development VM

QA server

Customer Data Center

Public Cloud

Production Cluster

Disaster recovery

Production Servers

Contributor's laptop

Can I migrate smoothly and quickly?

Results in N X N compatibility nightmare

	Development VM	QA Server	Single Prod Server	Onsite Cluster	Public Cloud	Contributor's laptop	Customer Servers
Static website	?	?	?	?	?	?	?
Web frontend	?	?	?	?	?	?	?
Background workers	?	?	?	?	?	?	?
User DB	?	?	?	?	?	?	?
Analytics DB	?	?	?	?	?	?	?
Queue	?	?	?	?	?	?	?

The Analogy

Multiplicity of Goods





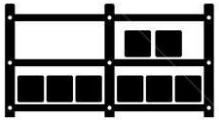


Do I worry about how goods interact (e.g. coffee beans next to spices)

Multiplicity of methods for transporting/storing

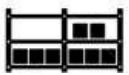




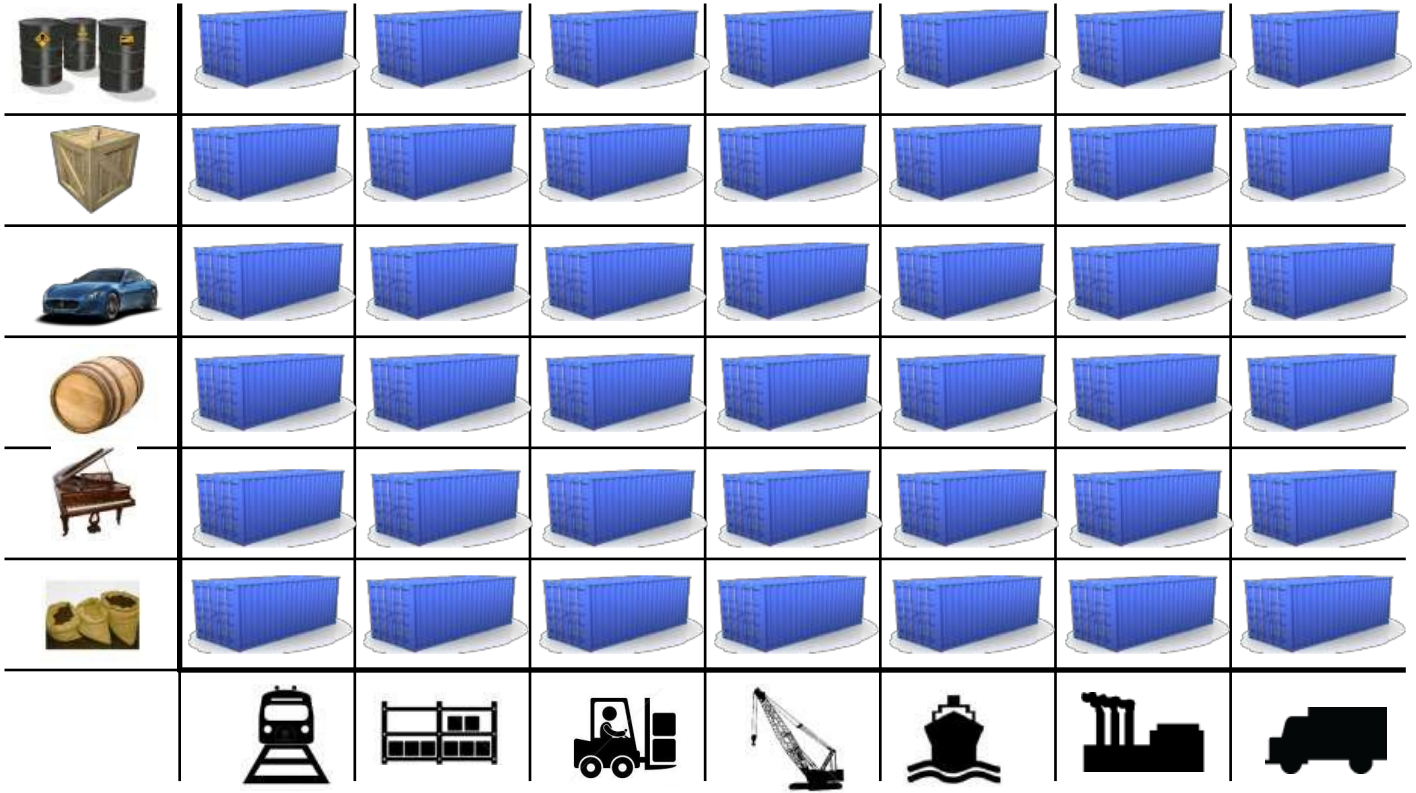

Can I transport quickly and smoothly (e.g. from boat to train to truck)

Also an NxN Matrix

	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
	?	?	?	?	?	?	?
							

Solution: Intermodal Shipping Container

This eliminated the NXN problem…



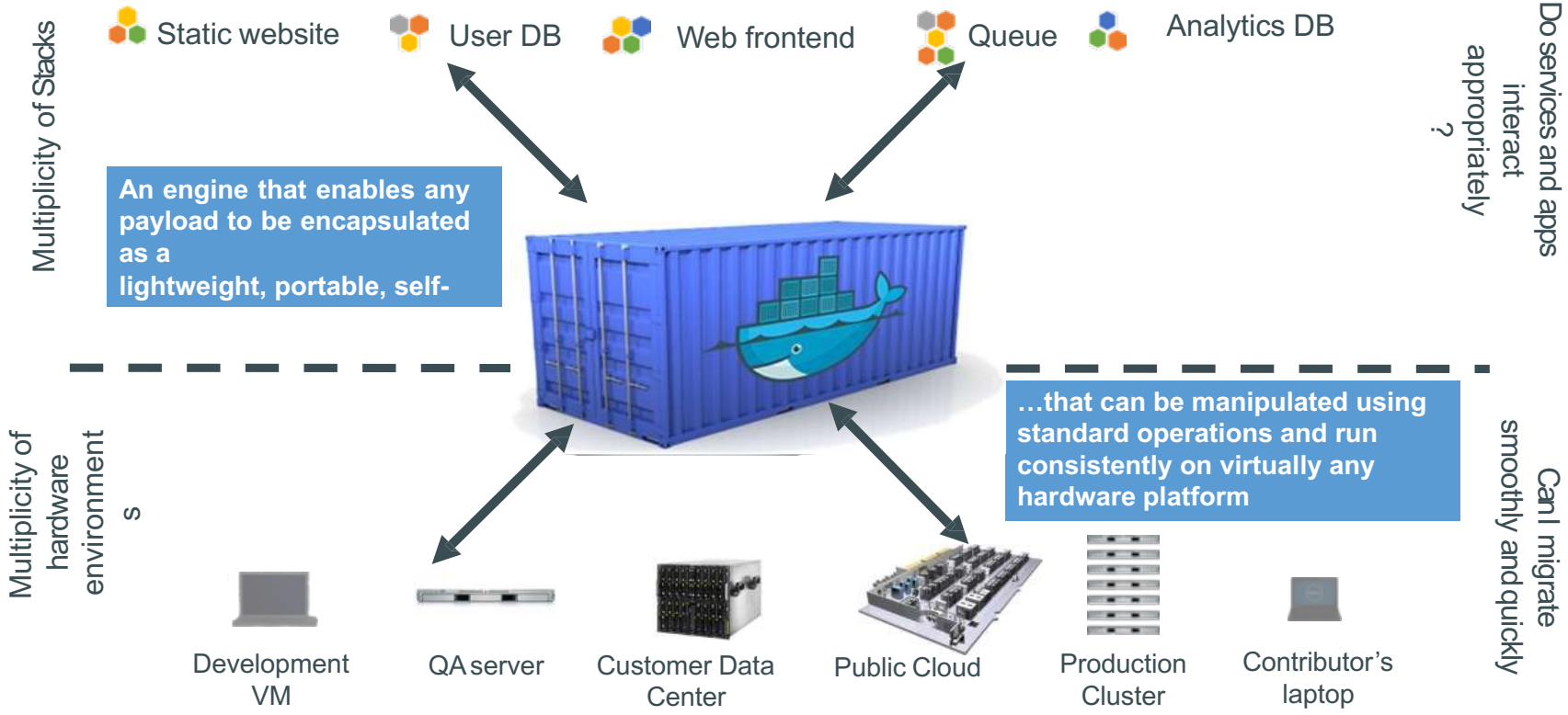
Spawned an Intermodal Shipping Container Ecosystem

- 90% of all cargo now shipped in a standard container
- Order of magnitude reduction in cost and time to load and unload ships
- Massive reduction in losses due to theft or damage
- Huge reduction in freight cost as percent of final goods (from >25% to <3%)

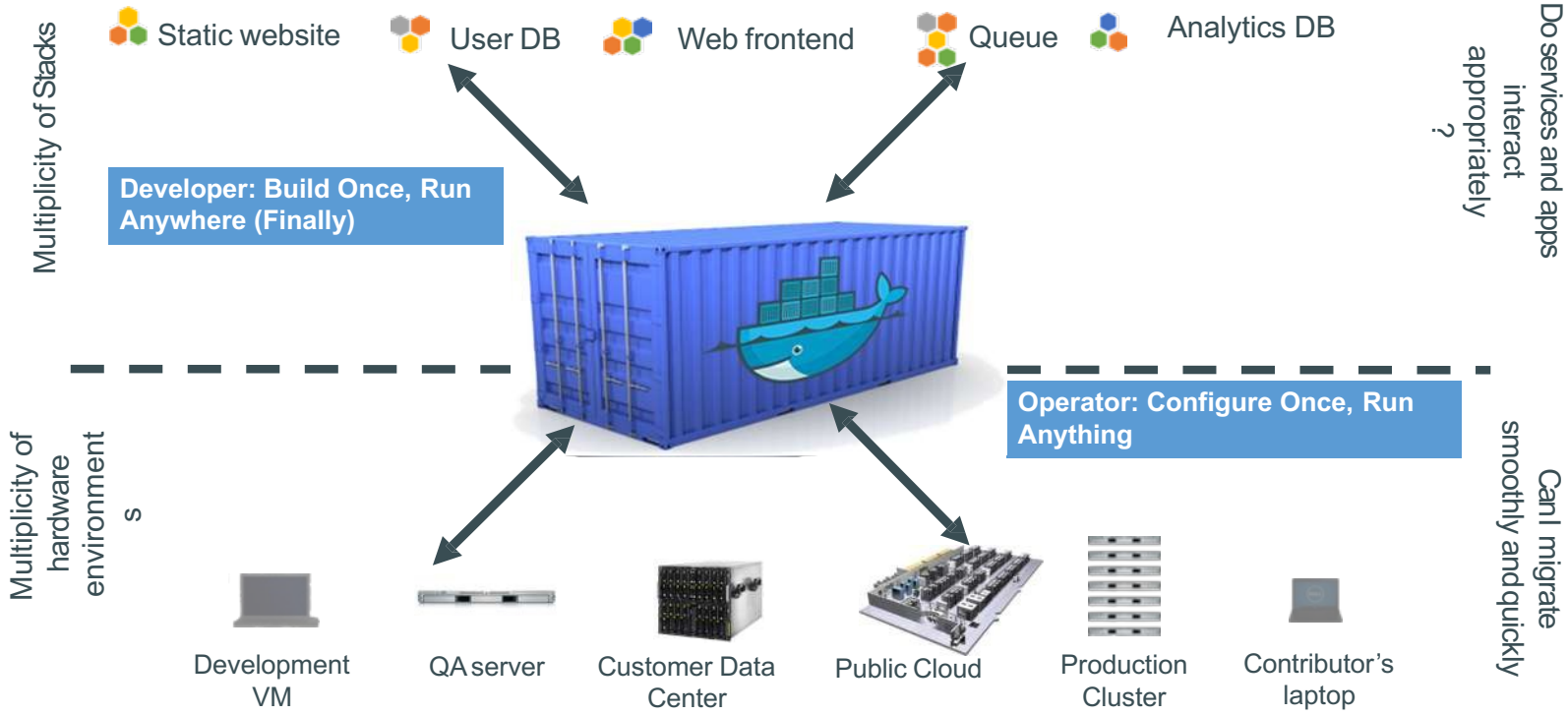
→ massive globalizations

- 5000 ships deliver 200M containers per year

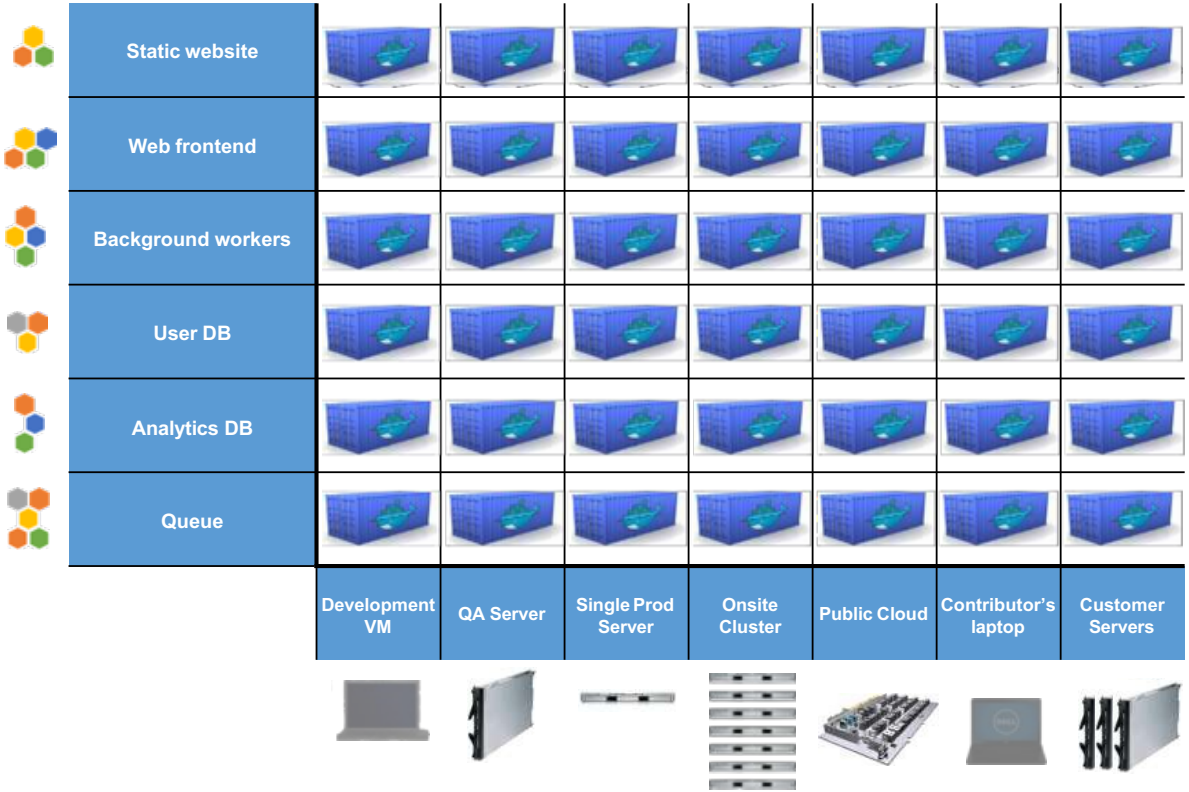
Docker is a shipping container system for code



Or…put more simply

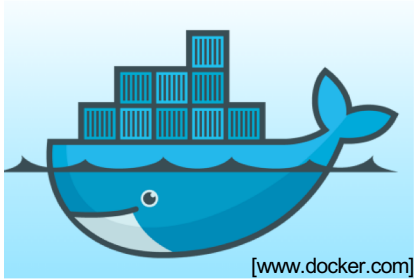


Docker solves the NXN problem



What is Docker?

- Provide a uniformed wrapper around a software package: *«Build, Ship and Run Any App, Anywhere»*
- Similar to shipping containers: The container is always the same, regardless of the contents and thus fits on all trucks, cranes, ships, ...



What is Docker?

Docker is an open-source project that automates the deployment of applications inside software containers, by providing an additional layer of abstraction and automation of operating system–level virtualization on Linux.

[Source: en.wikipedia.org]

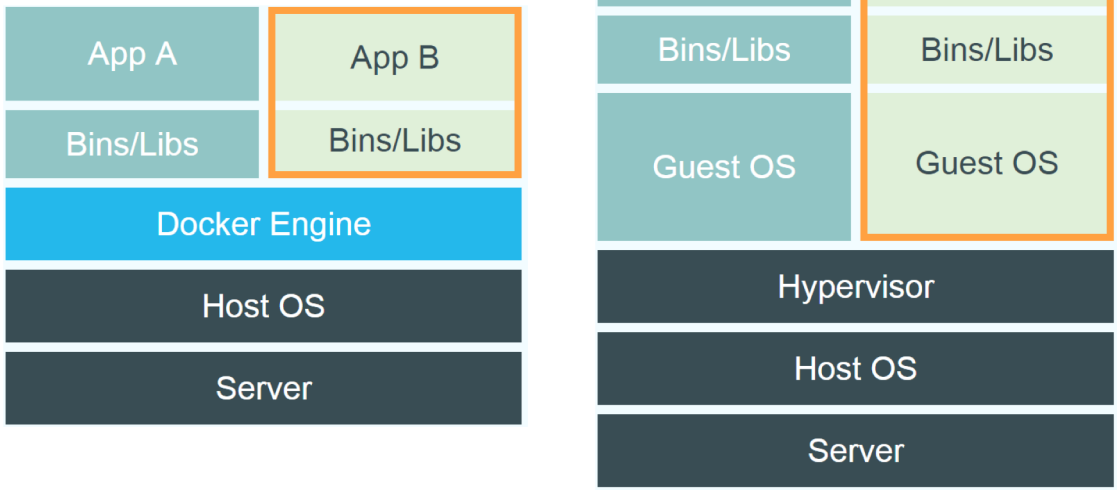
Why Developers Care

- Build once…run anywhere
  - A clean, safe, hygienic and portable runtime environment for your app.
  - No worries about missing dependencies, packages and other pain points during subsequent deployments.
  - Run each app in its own isolated container, so you can run various versions of libraries and other dependencies for each app without worrying
  - Automate testing, integration, packaging…anything you can script
  - Reduce/eliminate concerns about compatibility on different platforms, either your own or your customers.
  - Cheap, zero-penalty containers to deploy services? A VM without the overhead of a VM?
  - Instant replay and reset of image snapshots? That's the power of Docker

Why Devop's Cares?

- Configure once…run anything
  - Make the entire lifecycle more efficient, consistent, and repeatable
  - Increase the quality of code produced by developers.
  - Eliminate inconsistencies between development, test, production, and customer environments
  - Support segregation of duties
  - Significantly improves the speed and reliability of continuous deployment and continuous integration systems
  - Because the containers are so lightweight, address significant performance, costs, deployment, and portability issues normally associated with VMs

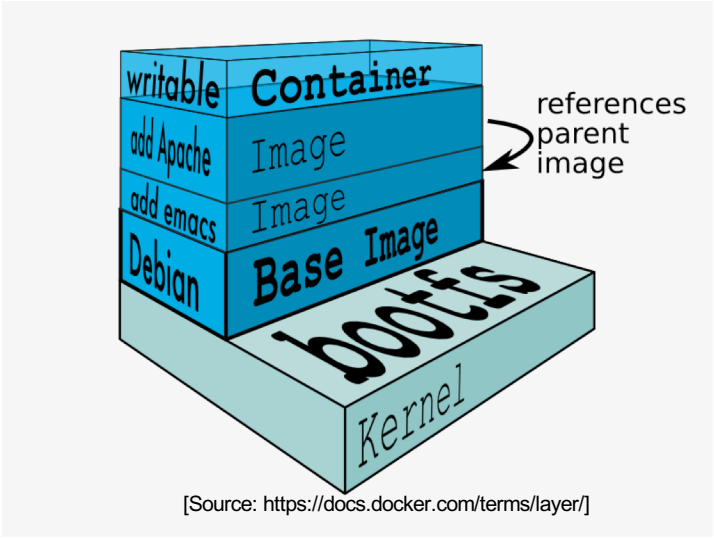
Docker Vs VM



Source: https://www.docker.com/whatisdocker/

Containers

- libvirt: Platform Virtualization
- LXC (LinuX Containers): Multiple isolated Linux systems (containers) on a single host
- Layered File System



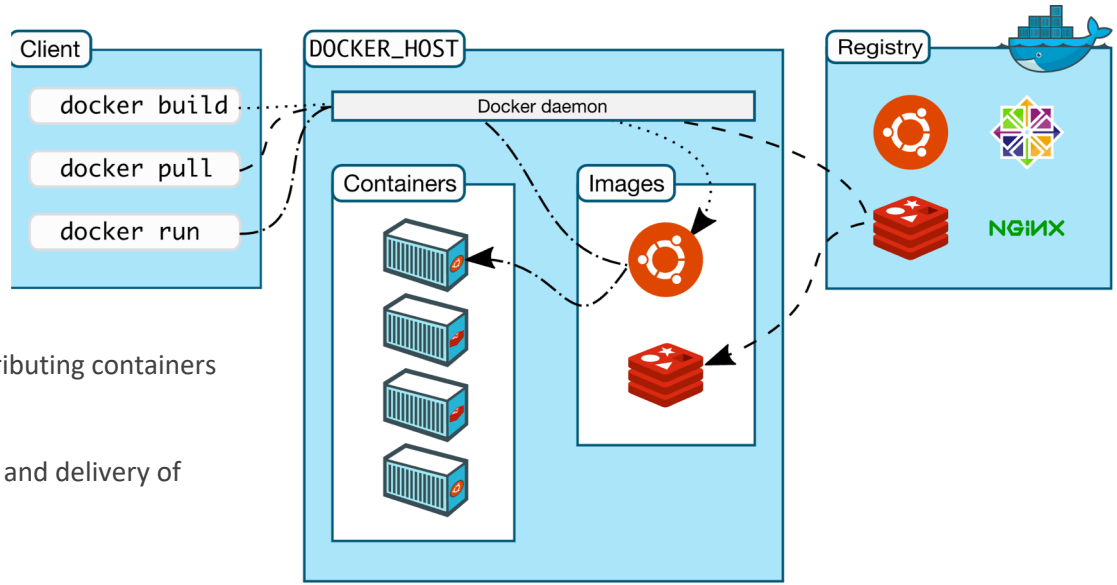
[Source: https://docs.docker.com/terms/layer/]

Containers

- Containers are operating system level virtualization
  - Allows for multiple isolated user space instances called containers
  - They share a single kernel
  - Can be added or removed at any time
- Containers consist of a self contained Linux file system
  - Can be from any Linux distribution which is compatible with the host kernel
  - Usually contain a single application such as a server
- Operating system level virtualization is lightweight
  - Is often used in Cloud Computing

Architecture

- Docker uses a client-server architecture
- Client
  - a REST API Client
  - Over HTTP
  - Over local Unix socket
- Server
  - Is the Docker daemon
  - Responsible for building, running, and distributing containers
- Registry
  - Responsible for the storage, management, and delivery of Docker Images
    - Docker Hub
    - Private
    - Other vendors



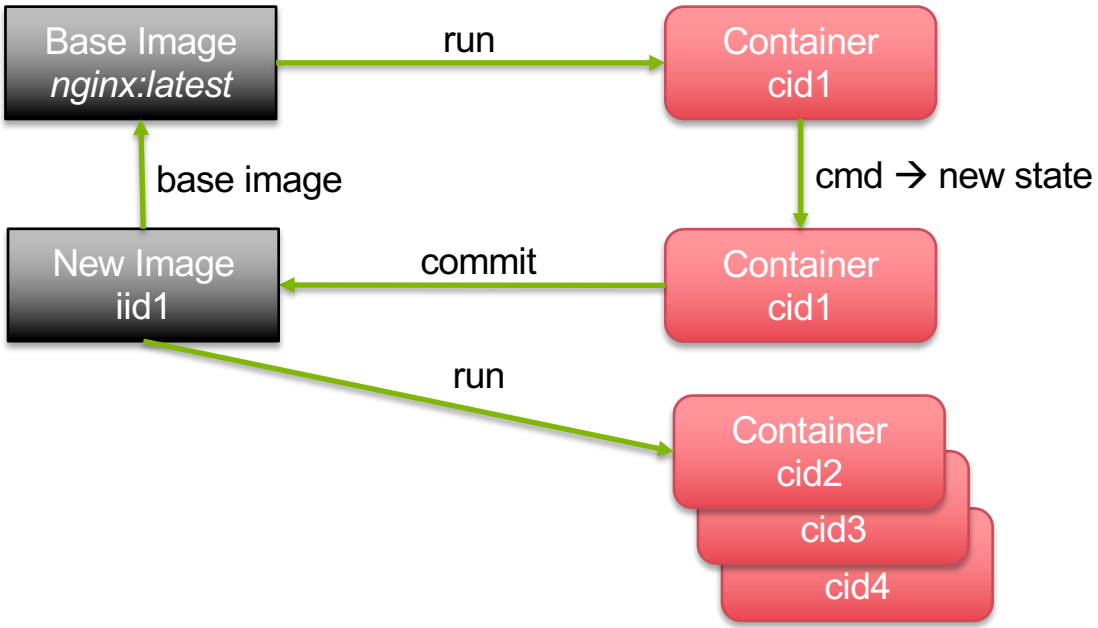
Docker Image

- Persisted snapshot that can be run
  - *images:* List all local images
  - *run*: Create a container from an image and execute a command in it
  - *tag*: Tag an image
  - *pull*: Download image from repository
  - *rmi*: Delete a local image
    - This will also remove intermediate images if no longer used

Docker Container

- Runnable instance of an image
  - *ps:* List all running containers
  - *ps –a*: List all containers (incl. stopped)
  - *top*: Display processes of a container
  - *start*: Start a stopped container
  - *stop*: Stop a running container
  - *pause*: Pause all processes within a container
  - *rm*: Delete a container
  - *commit*: Create an image from a container

Lifecycle - Image vs. Container



Dockerfile

- Create images automatically using a build script: «Dockerfile»
- Can be versioned in a version control system like Git or SVN, along with all dependencies
- Docker Hub can automatically build images based on dockerfiles on Github

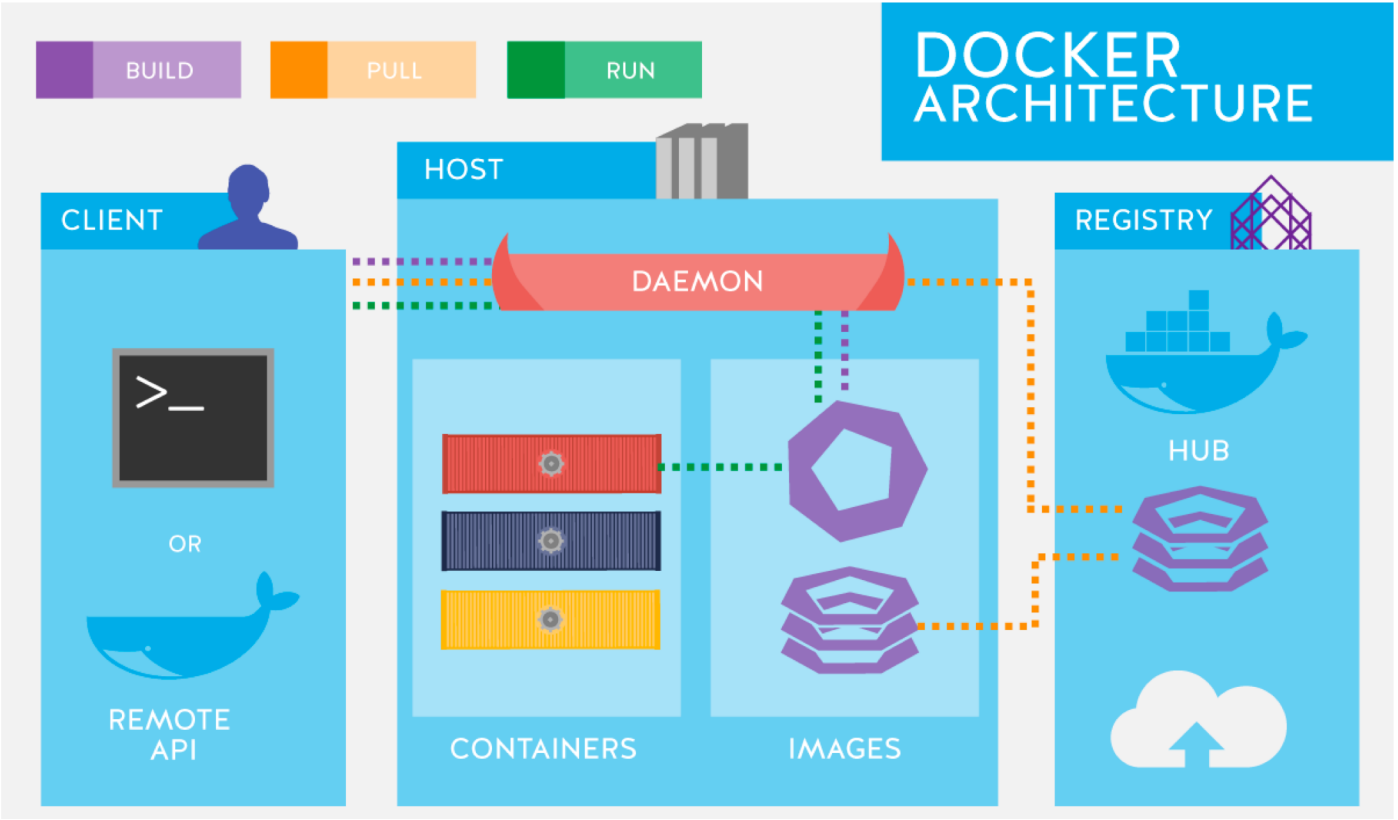
Example

- Dockerfile:
  FROM busybox:latest
  CMD ["date"]
- docker build [DockerFileDir]
- docker inspect [imageId]

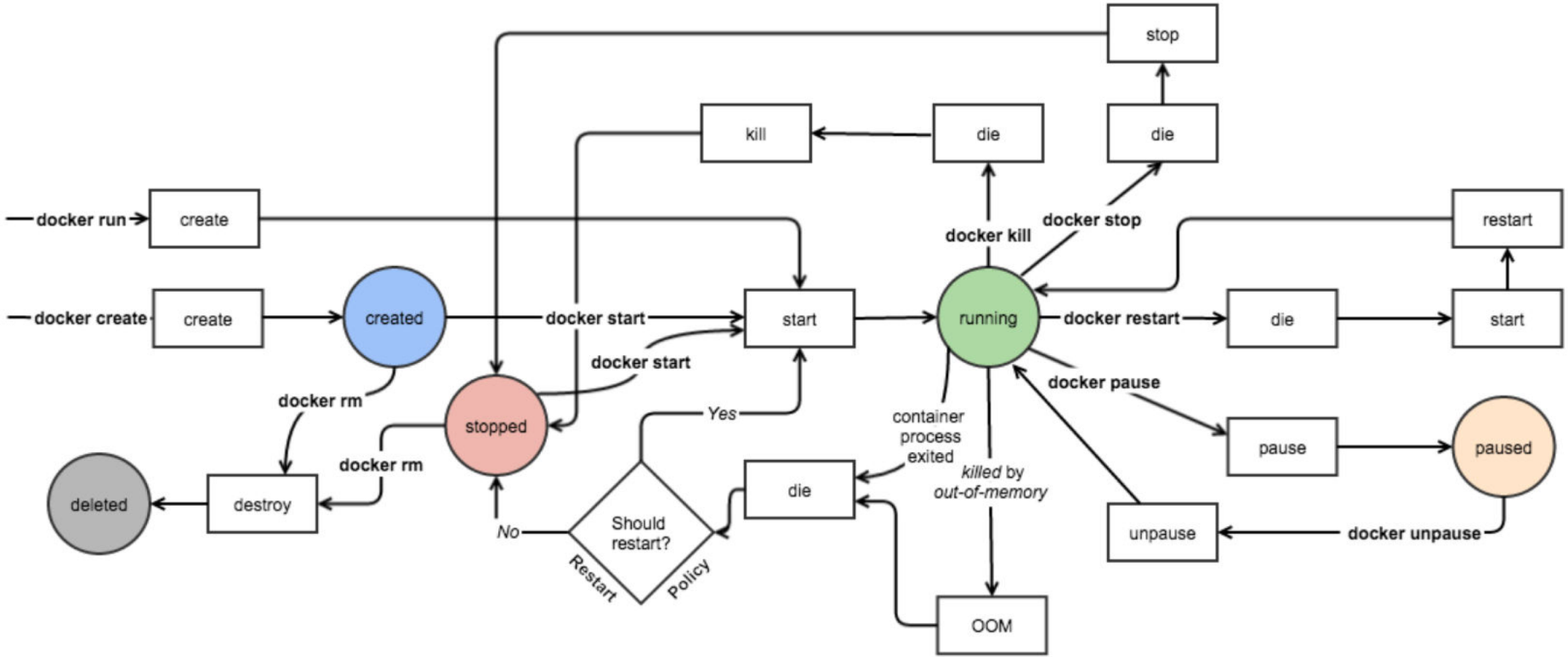
Volumes

- docker run –ti **–v /hostLog:/log** ubuntu
- Run second container: Volume can be shared
  - docker run –ti --volumes-from firstContainerName ubuntu

Summary

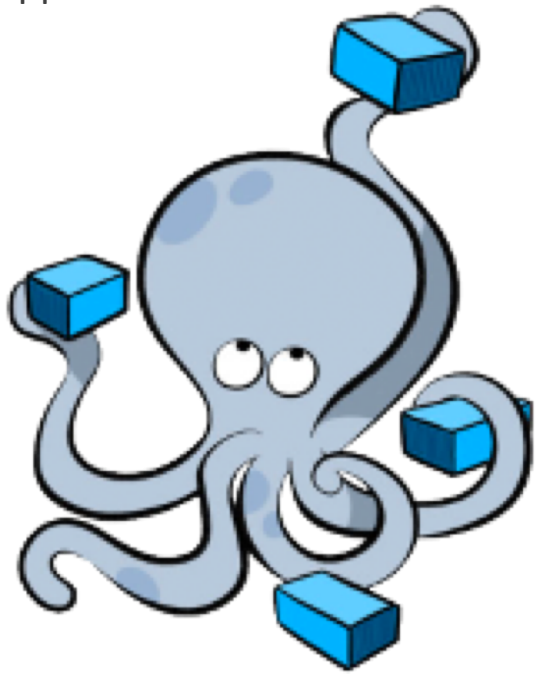

BUILD
PULL
RUN
DOCKER ARCHITECTURE
HOST
CLIENT
DAEMON
REGISTRY
HUB
OR
REMOTE API
CONTAINERS
IMAGES

Docker Mind Map



Docker Compose

- Docker Compose is a tool for running multi-container Docker applications
- A configuration file is used to define the services
- All of the services can be run using a single command
- Compose can manage the lifecycle of an application
  - Start, stop, and rebuild services
  - View the status of running services
  - Get the log output of running services
  - Run a command on a service



Questions?