

docker

Introduction to Docker

Feb, 2019

Agenda

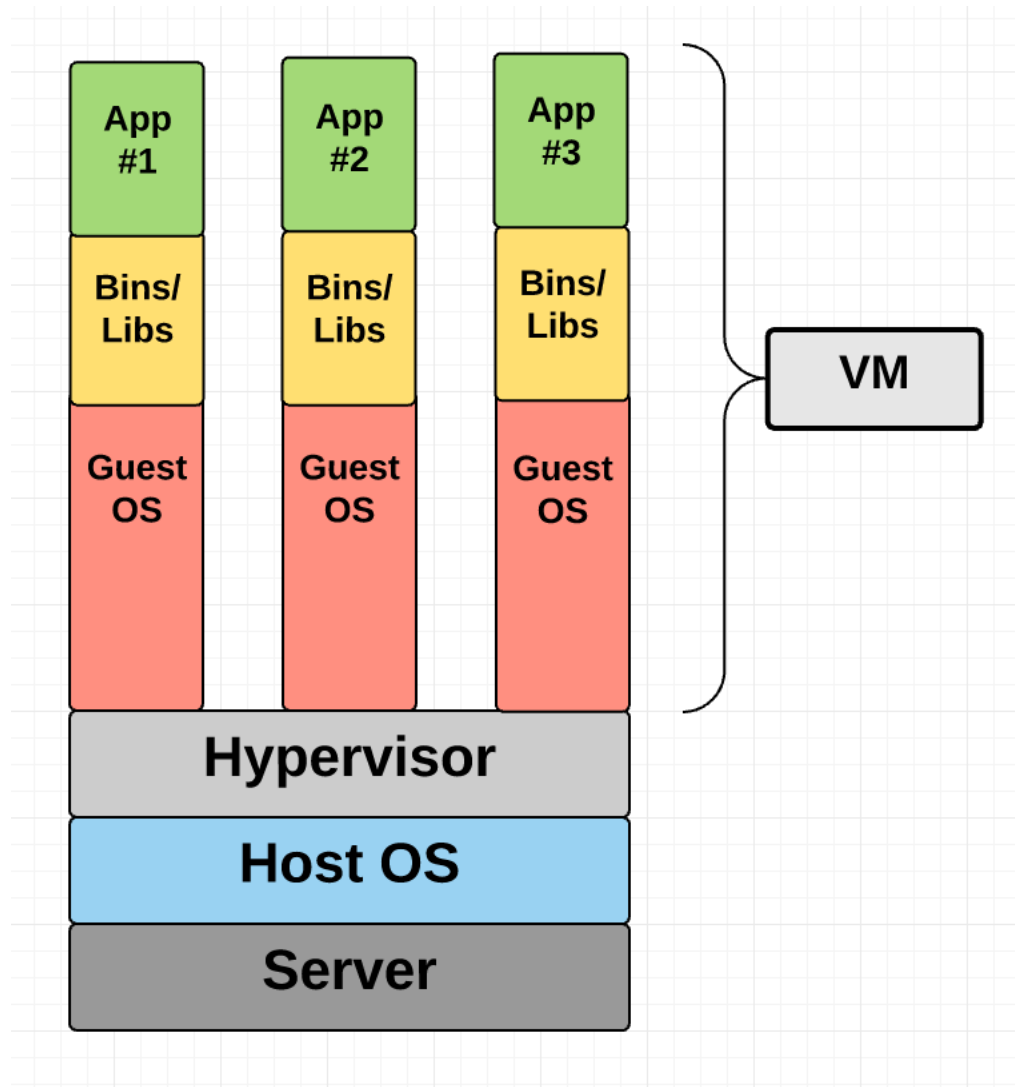
- **What is Docker?**
- **Containers vs. VMs**
- **How Docker works**
- **Why Dockers ? Docker Benefits**
- **Why Developers Care ?**
- **Docker Architecture**
- **What are Docker Images**
- **What are Docker Containers**
- **How to run Tomcat on Docker Container**
- **What is Docker File**
- **What is Docker Compose**
- **What is Docker Volumes**
- **Docker Swarm**

Docker: Containerization for Software

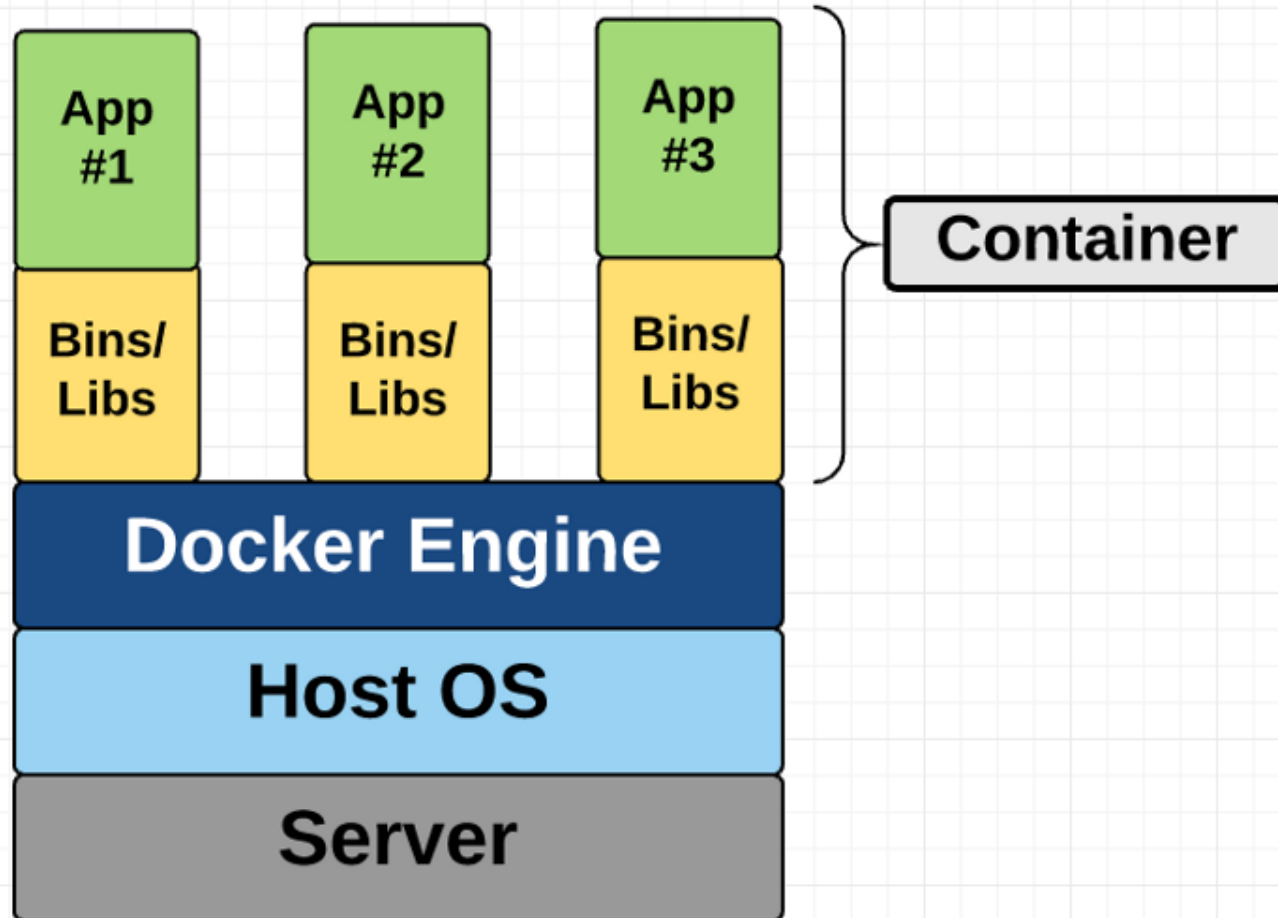


“Docker is an open platform for developers and sysadmins to build, ship, and run distributed applications”

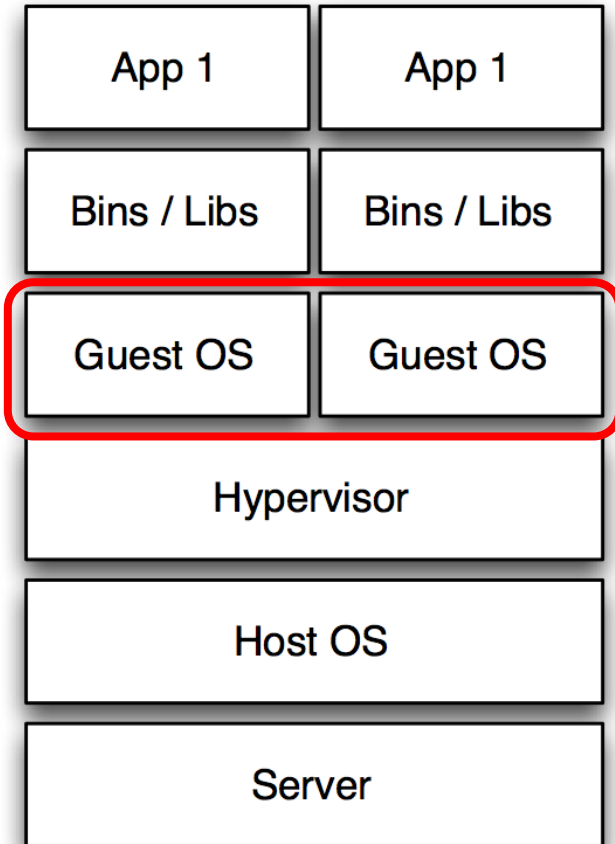
Virtual Machine



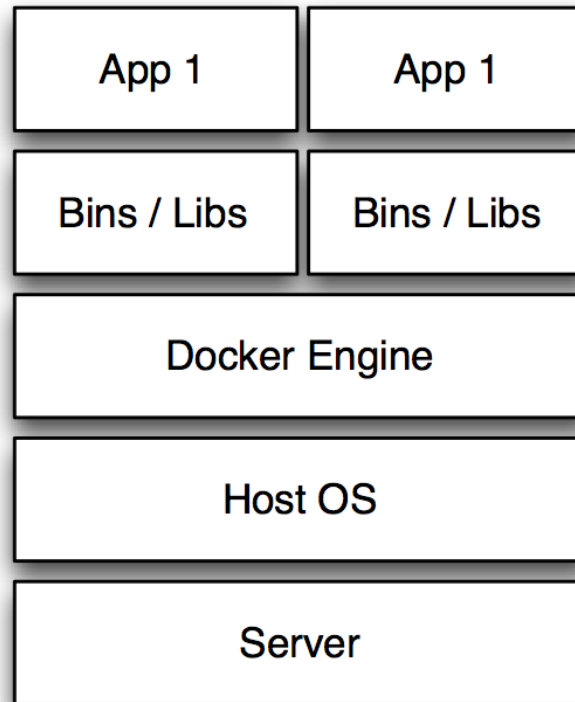
Container



VM vs. Docker (Containers)



Virtual Machines

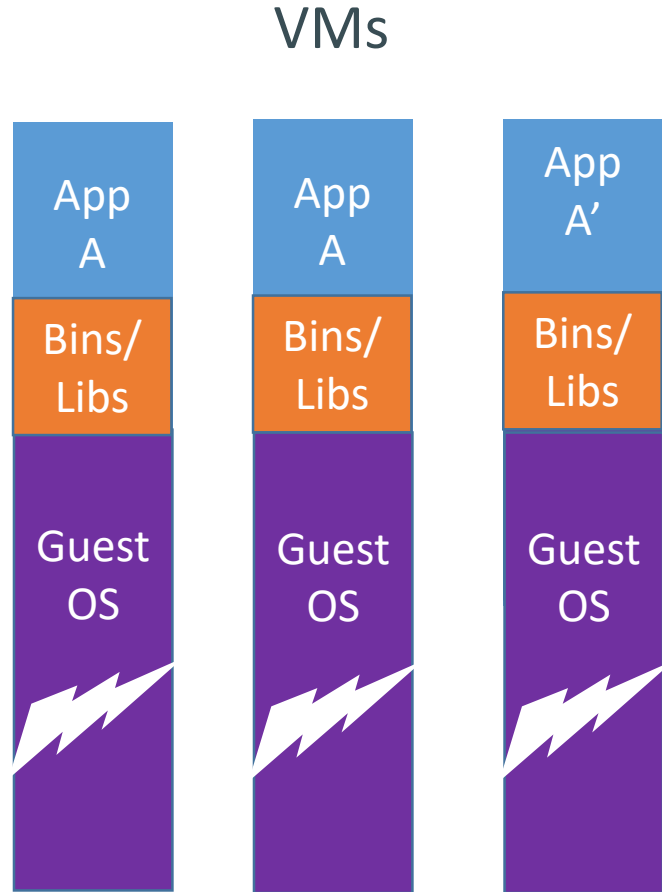


Docker

Docker Engine

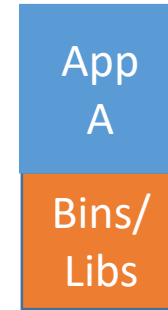
Docker engine is the layer on which Docker runs. It's a lightweight runtime and tooling that manages containers, builds, and more.

Why are Docker containers lightweight?

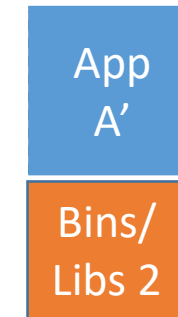


VMs

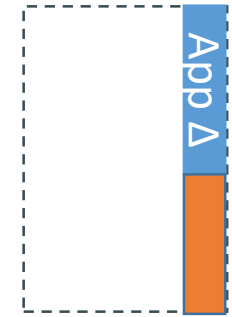
Every app, every copy of an app, and every slight modification of the app requires a new virtual server



Original App
(No OS to take up space, resources, or require restart)



Copy of App
No OS. Can Share bins/libs



Modified App
Copy on write capabilities allow us to only save the diffs Between container A and container A'

So why Docker?

- **Ease of use.** It allows anyone to package an application on their laptop, which in turn can run unmodified anywhere
 - The mantra is: “build once, run anywhere.”
- **Speed.** Docker containers are very lightweight and fast. Since containers are just sandboxed environments running on the kernel, they take up fewer resources. You can create and run a Docker container in seconds, compared to VMs which might take longer because they have to boot up a full virtual operating system every time.
- **Docker Hub.** Docker users also benefit from the increasingly rich ecosystem of Docker Hub, which you can think of as an “app store for Docker images.” Docker Hub has tens of thousands of public images created by the community that are readily available for use.
- **Modularity and Scalability.** Docker makes it easy to break out your application’s functionality into individual containers. With Docker, it’s become easier to link containers together to create your application, making it easy to scale or update components independently in the future.

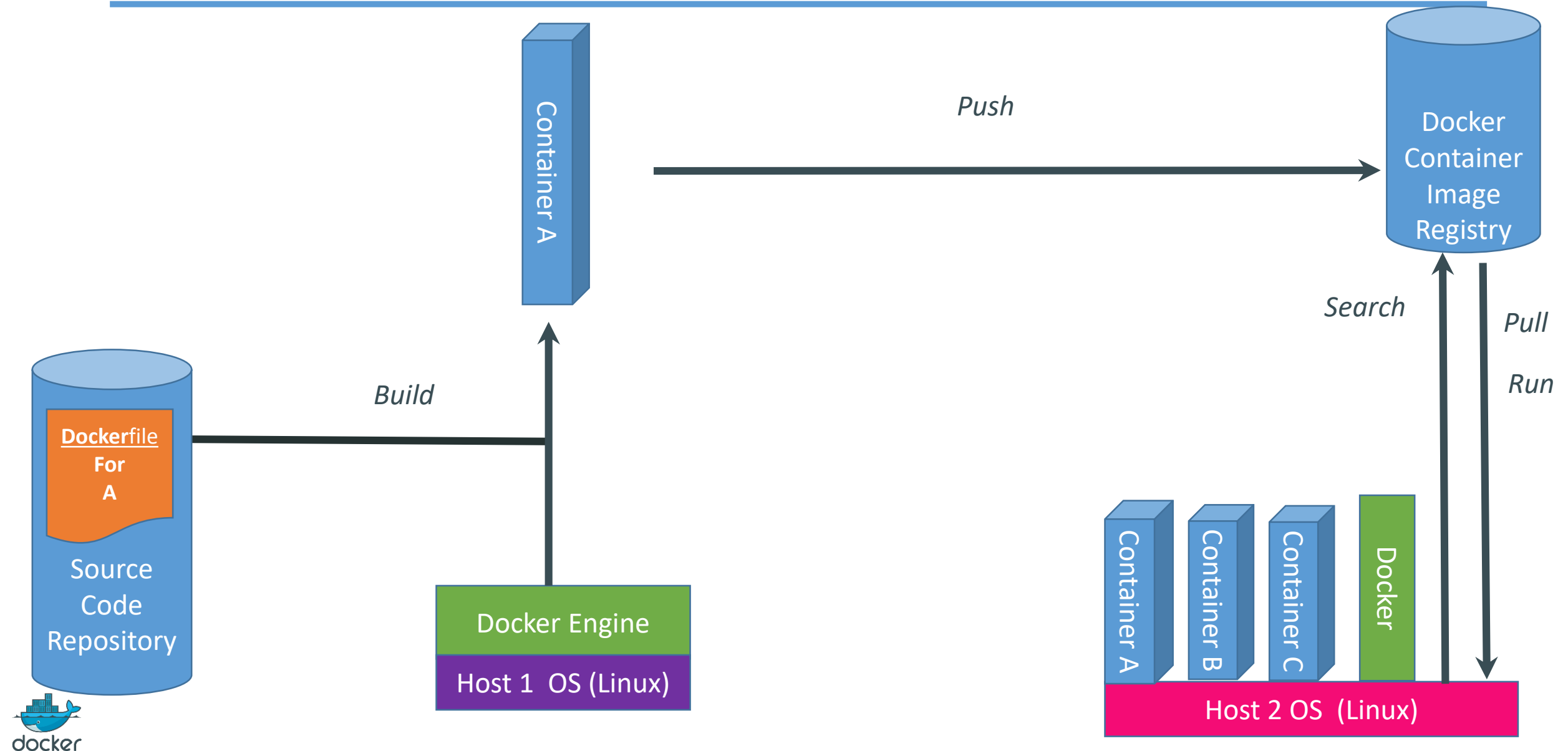
Docker Benefits

- 1. Local development environments can be set up that are exact replicas of a live environment/server.**
- 2. It simplifies collaboration by allowing anyone to work on the same project with the same settings, irrespective of the local host environment.**
- 3. Multiple development environments can be run from the same host each one having different configurations, operating systems, and software.**
- 4. It gives you instant application portability. Build, ship, and run any application as a portable container that can run almost anywhere.**

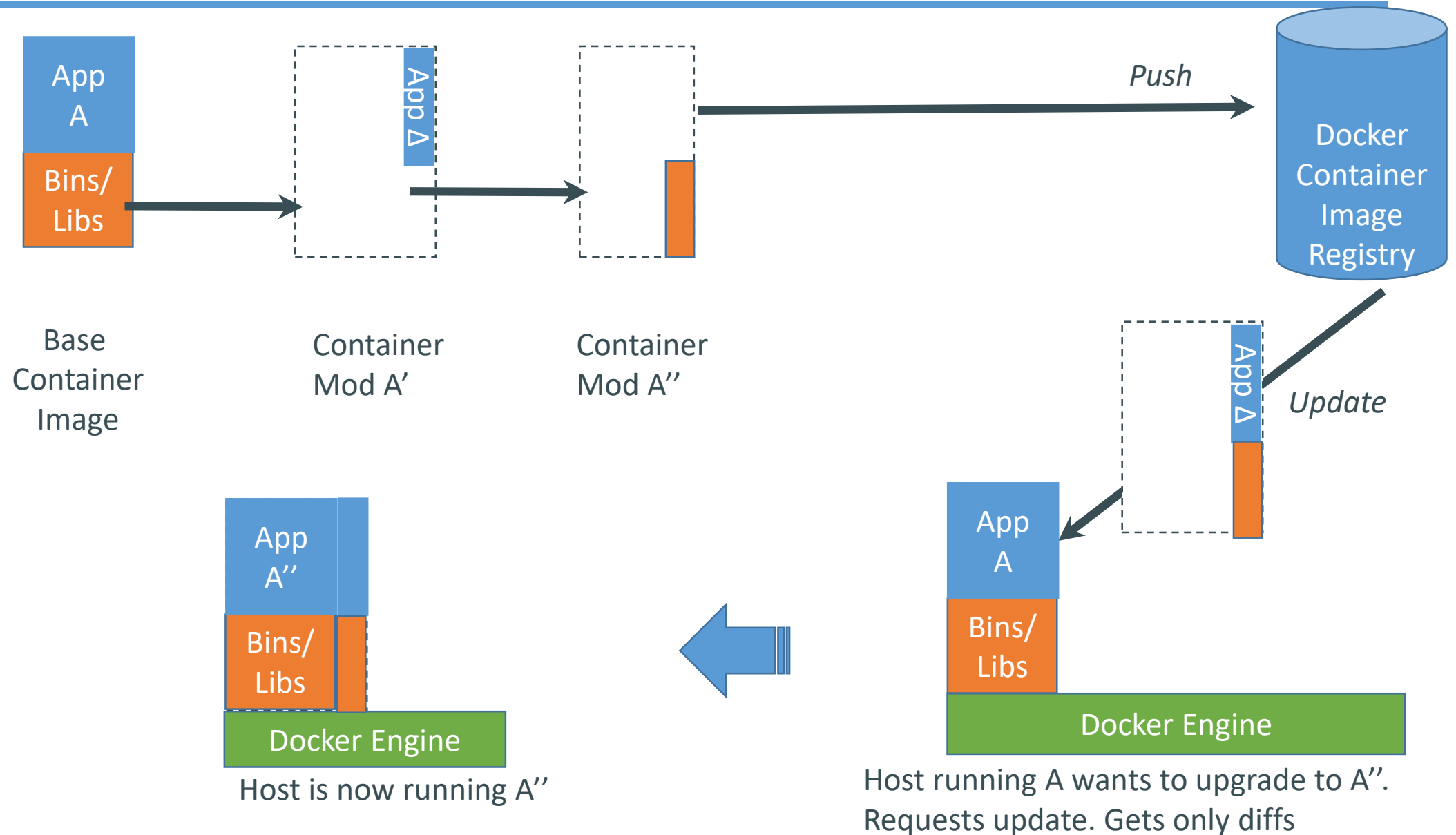
Why Developers Care

- **Build once...(finally) run anywhere***
 - **A clean, safe 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**
 - **Reduce/eliminate concerns about compatibility on different platforms**
 - **Cheap, zero-penalty containers to deploy services. A VM without the overhead of a VM.**

What are the basics of the Docker system?



Changes and Updates



What are Docker Images

- **Docker Images are templates used to create Docker containers.**
- **Where are Images Stored**

Registries (e.g. docker hub)

Can be stored locally or remote

COMMANDS :

:docker images --help

:docker pull image

:docker images

:docker images -q

:docker run imageName

:docker rmi imageName

:docker rmi -f imageName

:docker inspect

:docker history imagename

What are Docker Containers

- Containers are running instances of Docker Images.
- A container image is a lightweight, stand-alone, executable package of a piece of software that includes everything needed to run it: code, runtime, system tools, system libraries, settings.
- Containers run on the same machine sharing the same Operating system Kernel, this makes it faster
- Features of Containers:
 - Are lightweight
 - Fewer resources are used
 - Booting of containers is very fast
 - Can start, stop, kill, remove containers easily and quickly
 - Operating System resources can be shared within Docker

Example Commands :

:docker run ImageName

:docker start/stop ContainerName/ID

:docker pause/unpause ContainerName/ID

:docker stats/attach ContainerName/ID

:docker kill/rm ContainerName/ID

What is Docker File

- Docker can build images automatically by reading the instructions from a Docker file.
- A Docker file is a text document that contains all the commands a user could call on the command line to assemble an image.
- Using docker build users can create an automated build that executes several command-line instructions in succession.
- Each instruction creates one layer:

FROM creates a layer from the ubuntu:15.04 Docker image.

COPY adds files from your Docker client's current directory.

RUN builds your application with make.

CMD specifies what command to run within the container.

COMMANDS :

:docker build

:docker build -t ImageName:Tag directoryOfDockerfile

:docker run image

What is Docker Compose

- **Tool for defining & running multi-container docker applications.**
- **Use yaml files to configure application services (docker-compose.yml)**
- **Can start all services with a single command : docker compose up**
- **Can stop all services with a single command : docker compose down**
- **Can scale up selected services when required**

What is Docker Volumes

- **Volumes are stored in a part of the host filesystem which is managed by Docker.**
- **Non-Docker processes should not modify this part of the filesystem**
- **Bind mounts may be stored anywhere on the host system**
- **A given volume can be mounted into multiple containers simultaneously.**
- **Use of Volumes**

Decoupling container from storage

Share volume (storage/data) among different containers

Attach volume to container

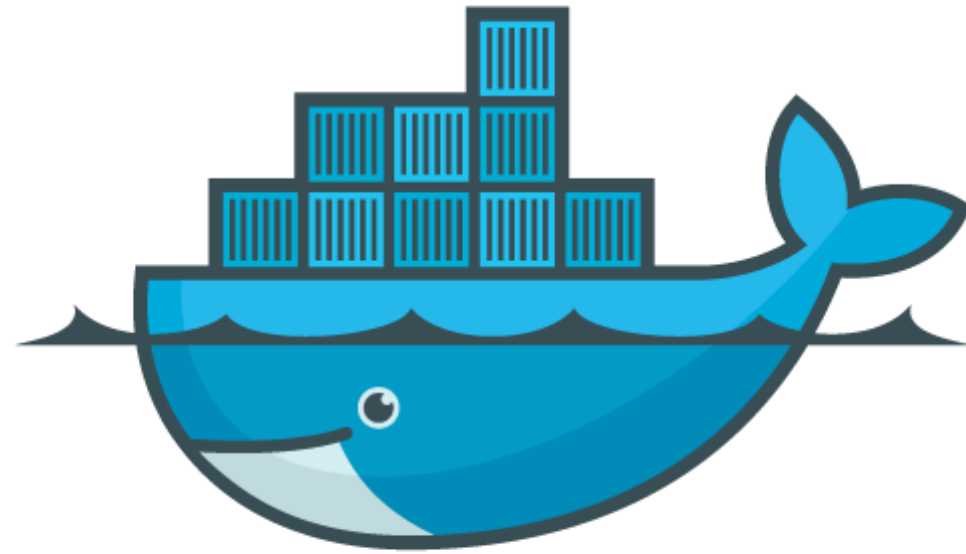
On deleting container, volume is not deleted

Docker Swarm

- **A swarm is a group of machines that are running Docker and joined into a cluster.**
- **Docker Swarm is a tool for Container Orchestration.**
- **Let's take an example You have 100 containers**
- **You need to do**
 - Health check on every container**
 - Ensure all containers are up on every system**
 - Scaling the containers up or down depending on the load**
 - Adding updates/changes to all the containers**
- **Orchestration - managing and controlling multiple docker containers as a single service**
- **Tools available - Docker Swarm, Kubernetes, Apache Mesos**

Want to learn more?

- **www.docker.io:**
 - **Documentation**
 - **Getting started: interactive tutorial, installation instructions, getting started guide,**
 - **About: Introductory whitepaper: <http://www.docker.io/the-whole-story/>**
- **Github: `dotcloud/docker`**
- **IRC: `freenode/#docker`**
- **Google groups: `groups.google.com/forum/#!forum/docker-user`**
- **Twitter: follow `@docker`**



docker

www.docker.io