

# Demystifying Docker

---

RAVI GARG – LET'S DO IT



How many of you have faced the issue “It works on my machine but not somewhere else” ?

---



# What were the issues?

---

Different operating system

Different servers

Conflicting software/dependencies/version requirements

Different toolsets

Basically different environment

# What we will learn?

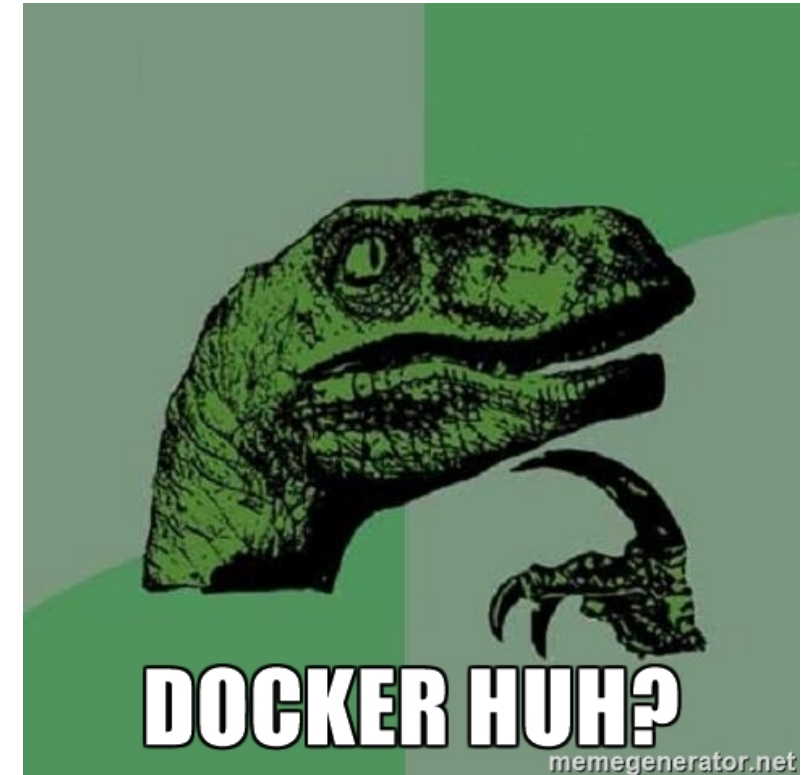
---

What is Docker? What it does? Why Docker?

How is it different from virtual machines?

Images, Containers, Docker-file, Docker-hub and Workflow

Installation and Demo



# Intuition

---

# Why and When Docker?

---

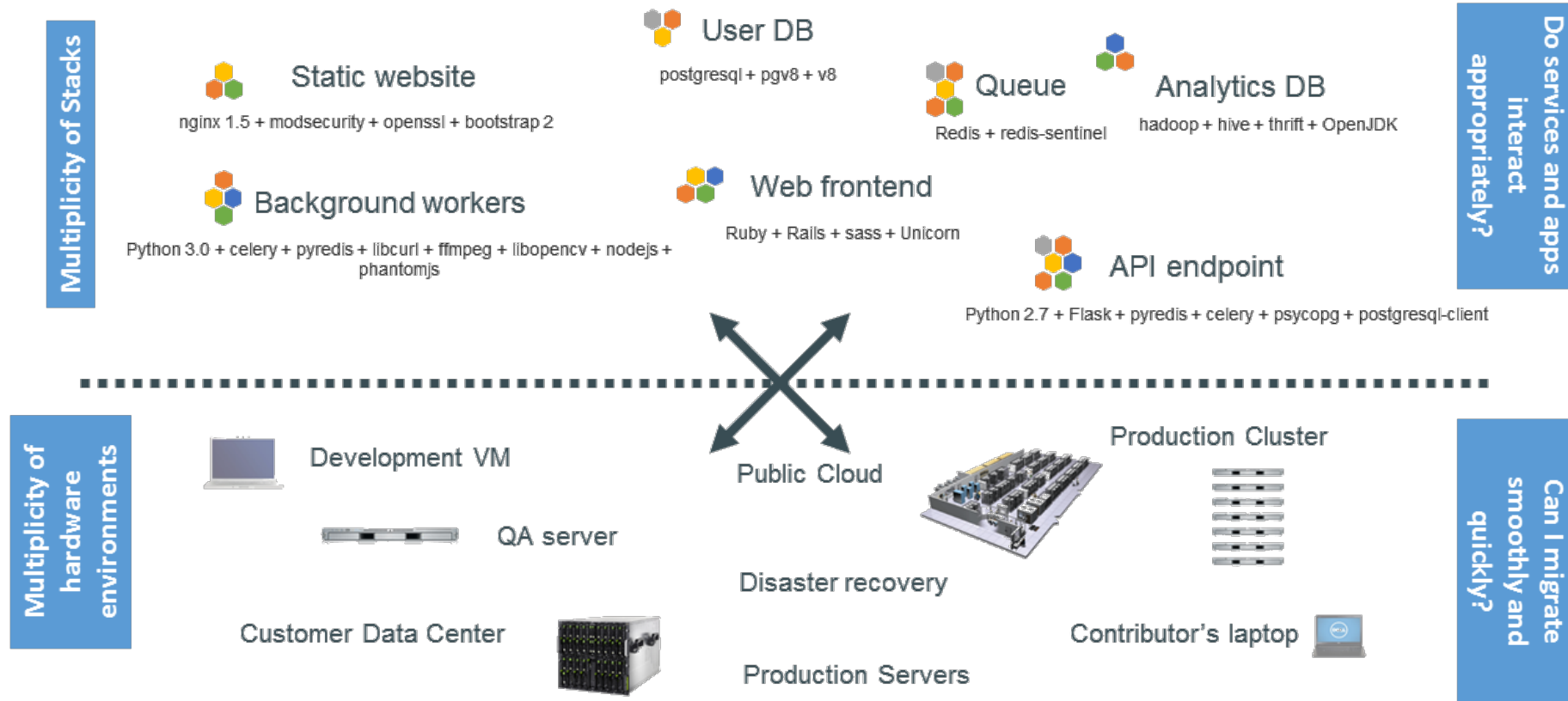
To solve the issue **“It runs on my machine but not anywhere else.”**

Many potential use cases mainly inclined for Software developers














But has value for Scientists and Researchers as well



# Conventional Software development

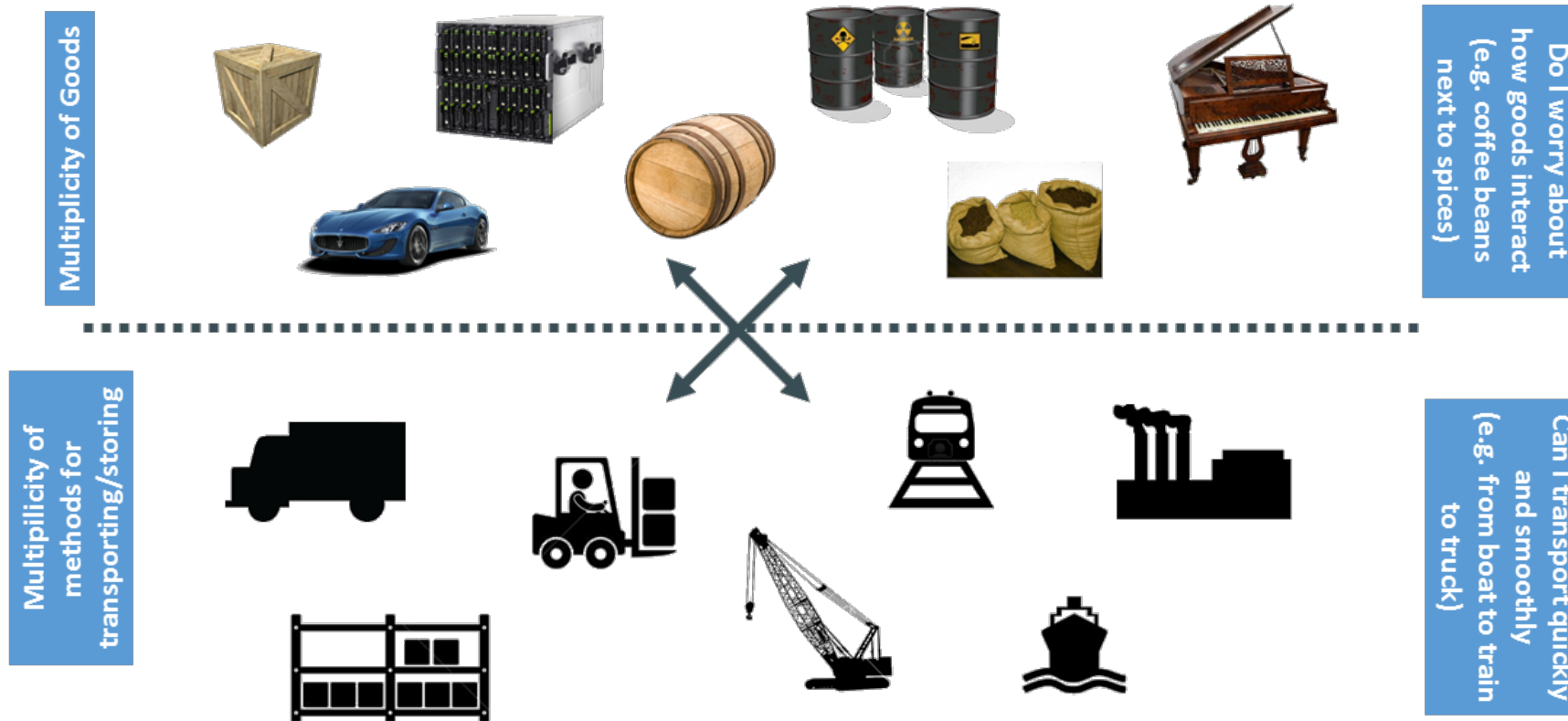


# Matrix from hell














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



# Analogy – Shipping Industry



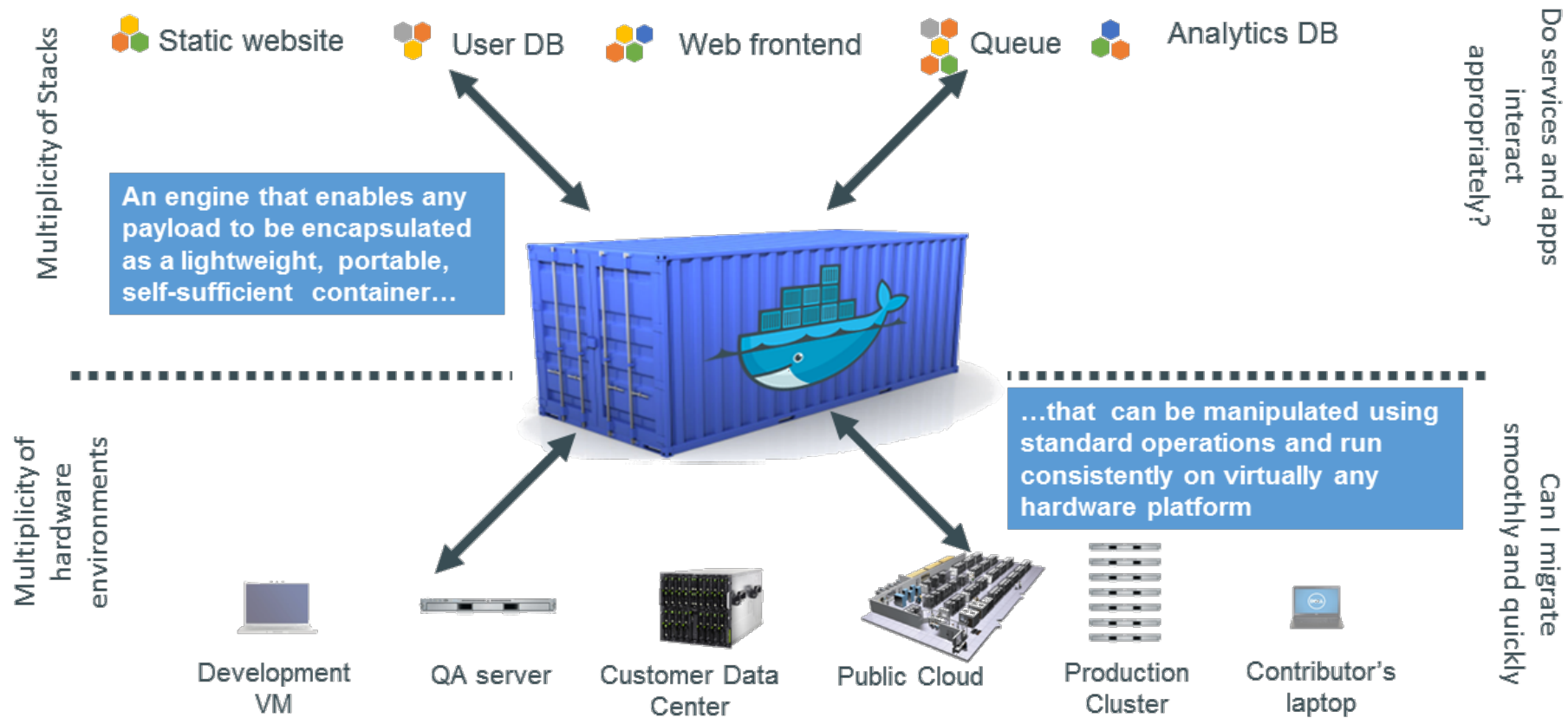
# Another matrix from hell

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

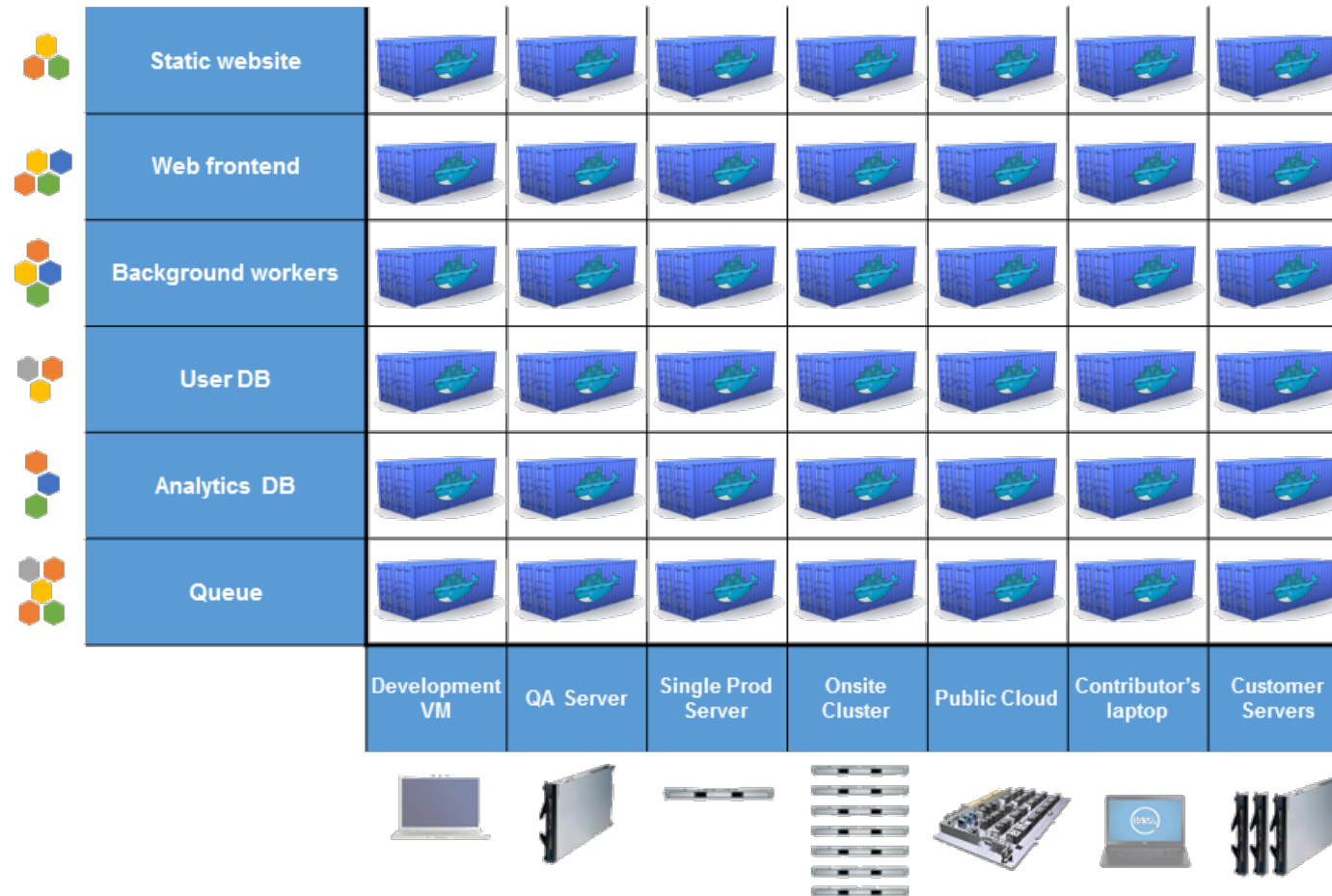
# Solution –Shipping Container



# Solution - Containers



# Eliminates matrix from hell

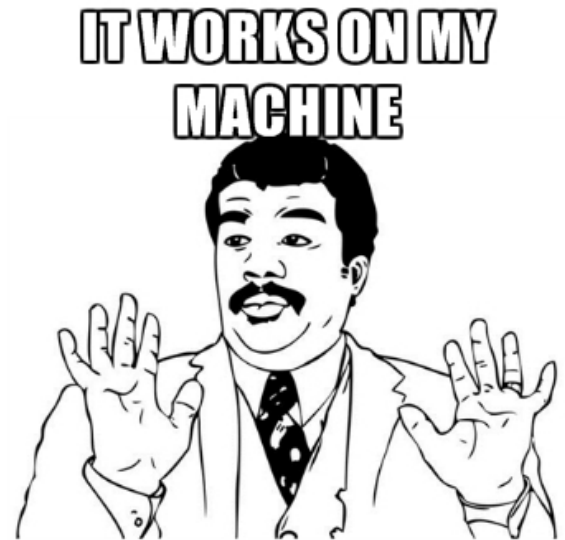


# Seriously Why?

---

## BUILD ONCE AND RUN ANYWHERE

- A clean, safe, hygienic, portable runtime environment for your application
- No worries about missing dependencies, packages and other pain points during deployment
- 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, 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.



# So finally what is Docker?

*“Docker is software containerization platform which wraps a piece of software in a complete file system (called container) that contains everything needed to run: code, runtime, systemtools, system libraries, which thereby guarantees that the software will always run the same regardless of its environment”*

In simple words every thing from source code, dependencies, binaries i.e. all the bare minimum get packed into Linux Containers

- Units of software delivery
- Run everywhere
  - Regardless of kernel version
  - Regardless of host distro
- Run anything
  - If it can run on host, it can run in the container



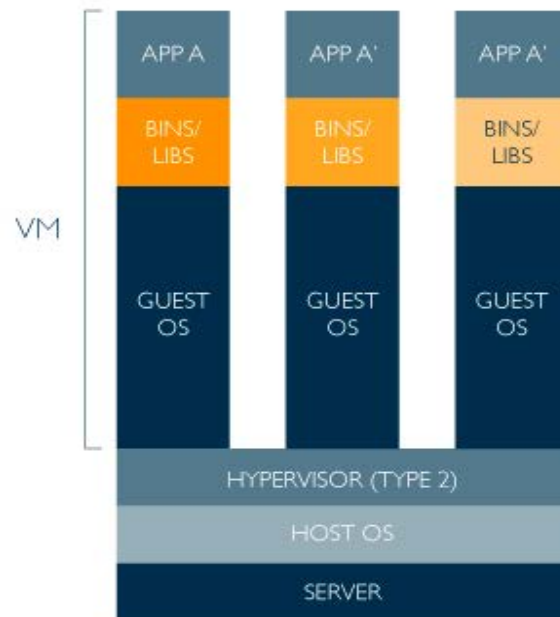
# Difference with VMs

---

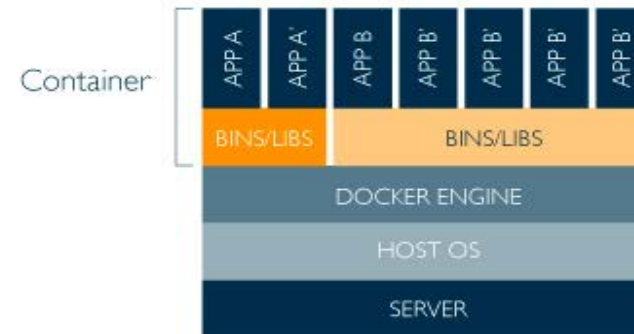


# Containers vs VMs

## CONTAINERS VS VMs



Containers are isolated, but share OS and, where appropriate, bins/libraries



Side note : A **hypervisor** or virtual machine monitor (VMM) is a piece of computer software, firmware or hardware that creates and runs virtual machines. A computer on which a hypervisor runs one or more virtual machines is called a host machine, and each virtual machine is called a guest machine.

# Internals

---

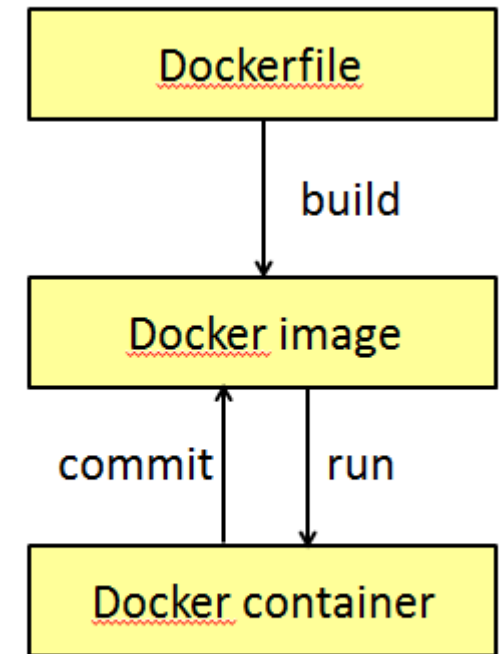
# Let's get technical: Docker Image, Container, Docker-file.

DOCKER IMAGE is a inert, immutable, file that's essentially a snapshot of a container. It doesn't have state and never changes. It is a set of layers. You start with base layer, make changes which are saved in layers forming another image.

DOCKER CONTAINERS is a running instance of the image. They're lightweight and portable encapsulations of an environment in which to run applications.

DOCKER FILE is a text document that contains all the commands a user could call on the command line to assemble an image.



“IF AN IMAGE IS A CLASS, THEN FILE IS THE DEFINITION AND CONTAINER IS AN INSTANCE OF A CLASS—A RUNTIME OBJECT.”



# Docker-Hub

<https://hub.docker.com>

Explore Official Repositories

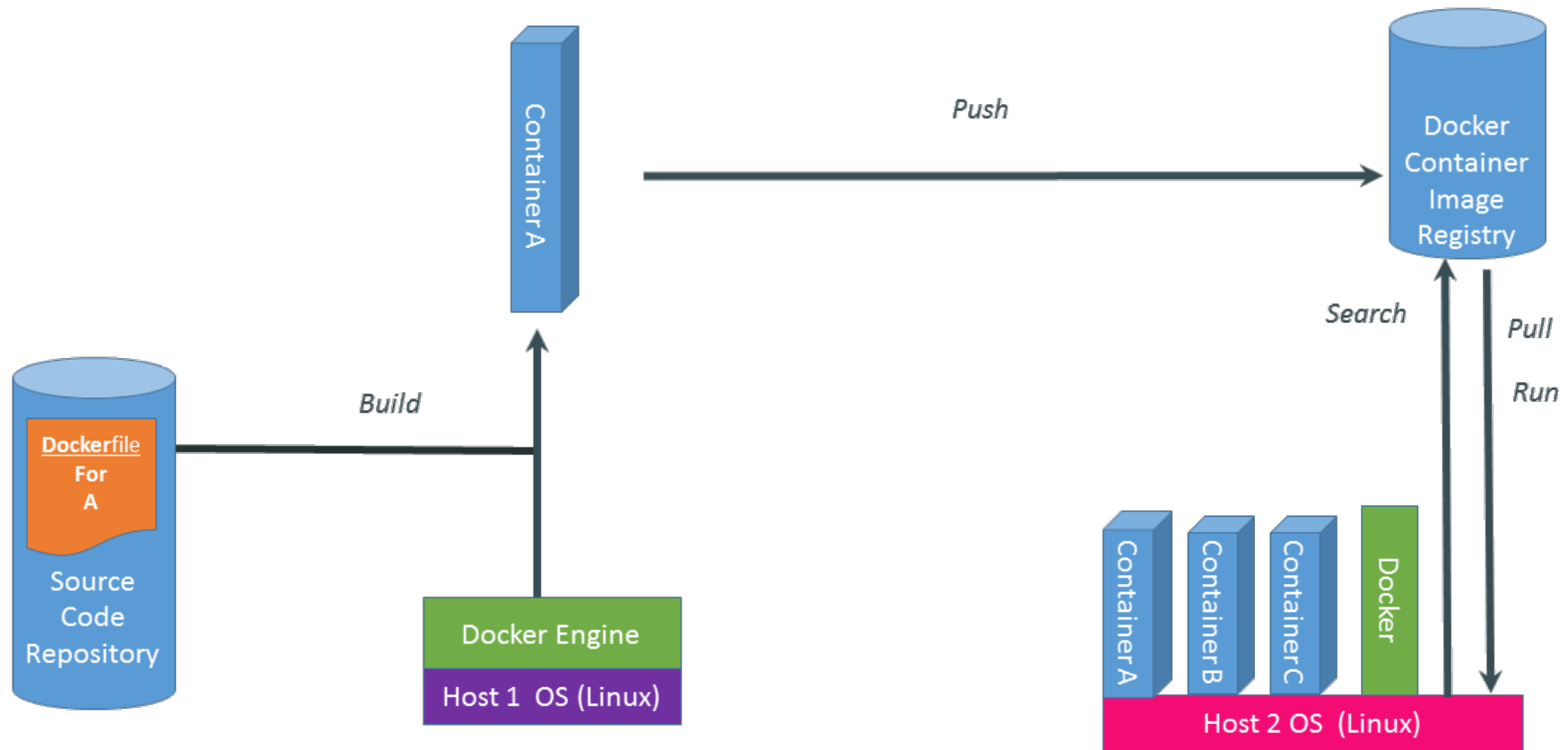
 <b>nginx</b> official	4.6K STARS	10M+ PULLS	<a href="#">➤ DETAILS</a>
 <b>redis</b> official	3.0K STARS	10M+ PULLS	<a href="#">➤ DETAILS</a>
 <b>busybox</b> official	860 STARS	10M+ PULLS	<a href="#">➤ DETAILS</a>
 <b>ubuntu</b> official	5.1K STARS	10M+ PULLS	<a href="#">➤ DETAILS</a>
 <b>registry</b> official	1.2K STARS	10M+ PULLS	<a href="#">➤ DETAILS</a>

# Git Analogy

---

<b>Docker</b>	<b>Git</b>	<b>Description</b>
Image	Repository	Collection of commits
Container	Clone	Used for local execution
Docker-hub	Github	Remote server

# Docker workflow - 1



# Docker workflow in words

---

Find an Image on Docker hub

Pull the Image from Docker hub

Run the Image on Docker host (Container)

Stop the instance

Commit the changes made to the instance (optional)

Remove the instance

Remove the image

# Docker workflow in commands

---

- `docker search ubuntu`
- `docker pull ubuntu`
- `docker images`
- `docker run -it ubuntu bash`
- `apt-get install this and that`
- `git clone git://....mycode`
- `pip install -r requirements.txt`
- `docker ps -a (-l)`
- `docker commit containerName imageName`
- `docker rm containerName`
- `docker rmi imageName`
- `docker push`

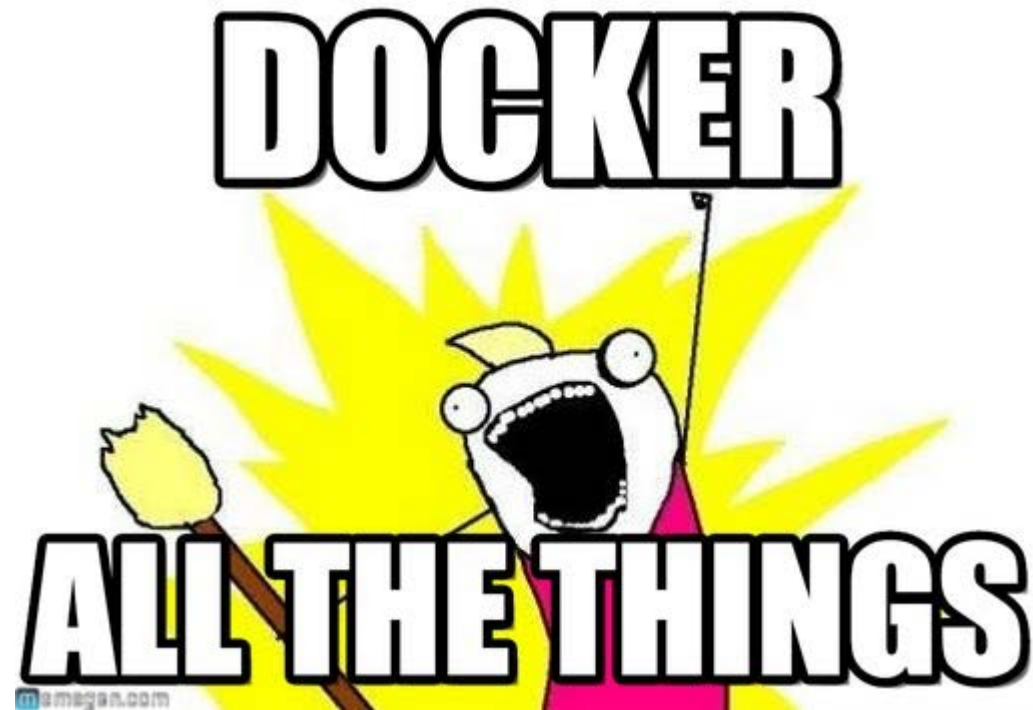


# Installation and Demo

---

Pavan's PyGlmnet

Eva's Project (optional)



# Docker commands

---

docker search

docker pull

docker images

docker run

docker ps

docker commit

docker rm

docker rmi

docker-machine ls

# Questions

---

