Intro to Docker Containers

March, 2018

Mike Raab
Senior Principal Product Manager
Oracle Container Development
Houston, Texas

@mikeraab

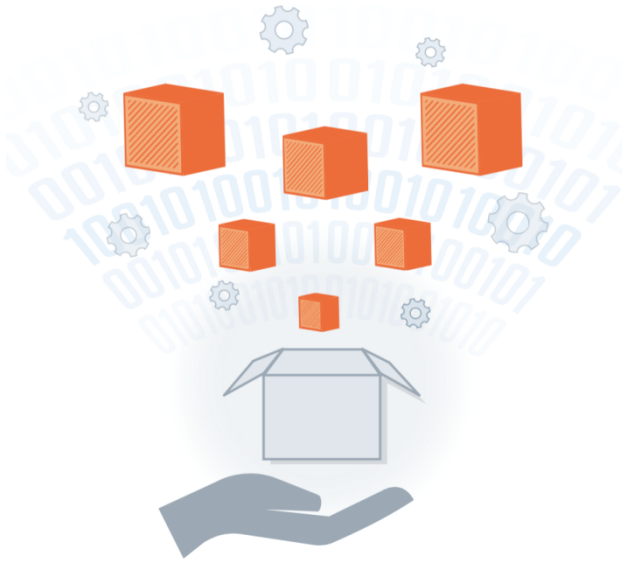
Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

ORACLE

Program Agenda

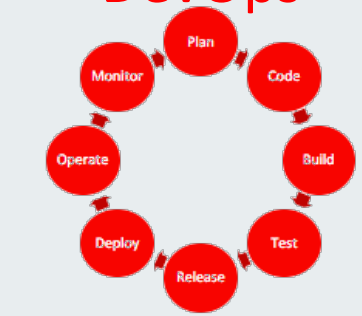
1. History of Containers
2. Excellent Use Cases for Containers
3. Basic Architecture and Nomenclature
4. Why Docker is Hot
5. Additional Resources
6. Q&A



History of Containers

History and Multi-Dimensional Evolution of Computing

Development Process	Application Architecture	Deployment and Packaging	Application Infrastructure
Waterfall	Monolithic	Physical Server	Datacenter
Agile	N-Tier	Virtual Servers	Hosted
DevOps	Microservices	Containers	Cloud



Historic Timeline of Unix Containers

Docker is both a Company and Technology

While Docker has been playing a key role in adoption of the Linux container technology, they did not invent the concept of containers

However, they have made the technology consumable by mere humans

Excellent Use Cases for Containers

Ready to Run Application Stacks

- Excellent for Dev/Test setups
- Deployment in Seconds, not Hours/Days
- Start Up, Tear Down Quickly

New App Dev & Microservices

- Refactor all or part of legacy app
- Containers are great for Microservices

One-Time Run Jobs and Analytics

- Run the Job / Analysis and quit

Front-End App Servers

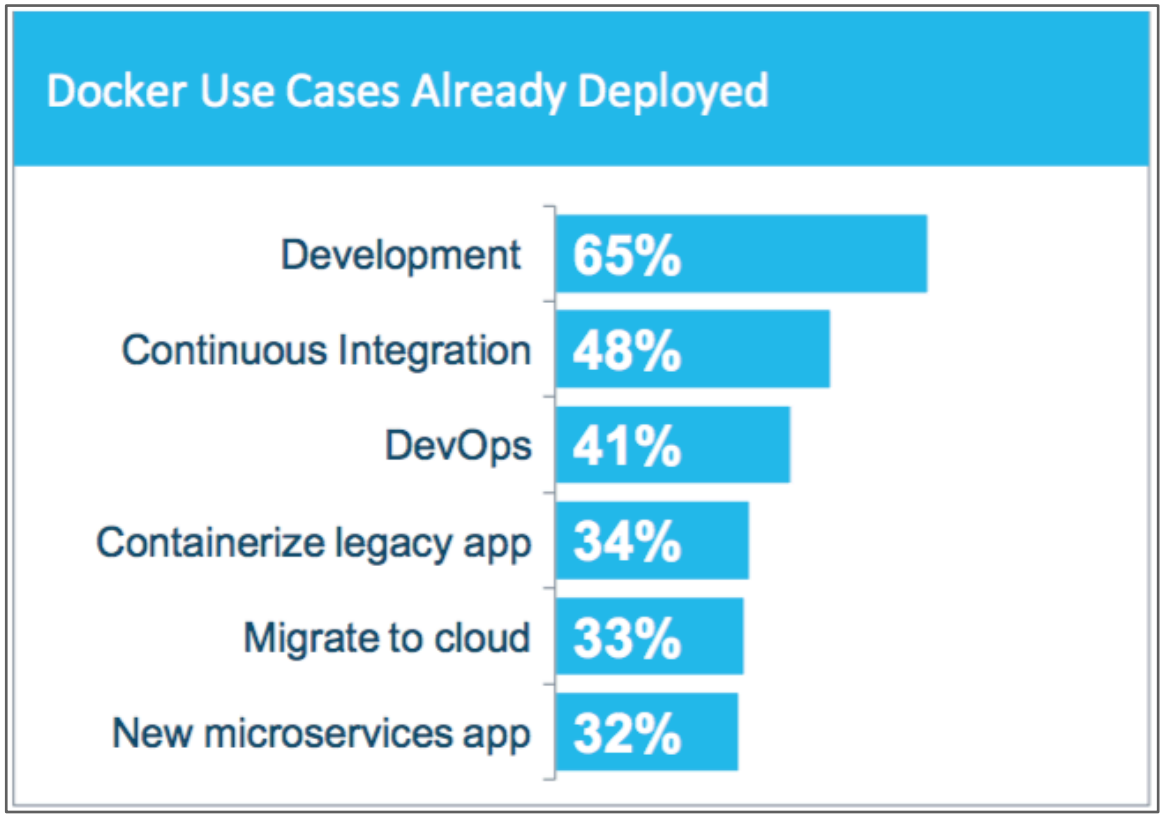
- Highly horizontally scalable
- Cattle Not Pets
- Fast A/B, Rolling Deployments
- Optimize CX
- Traditional Technologies - MW/Backend

Server Density

- Containers can use dynamic ports
- Run many of the same app on a server
  - instead of one per VM

ORACLE

How Containers are Being Used – Survey Says:



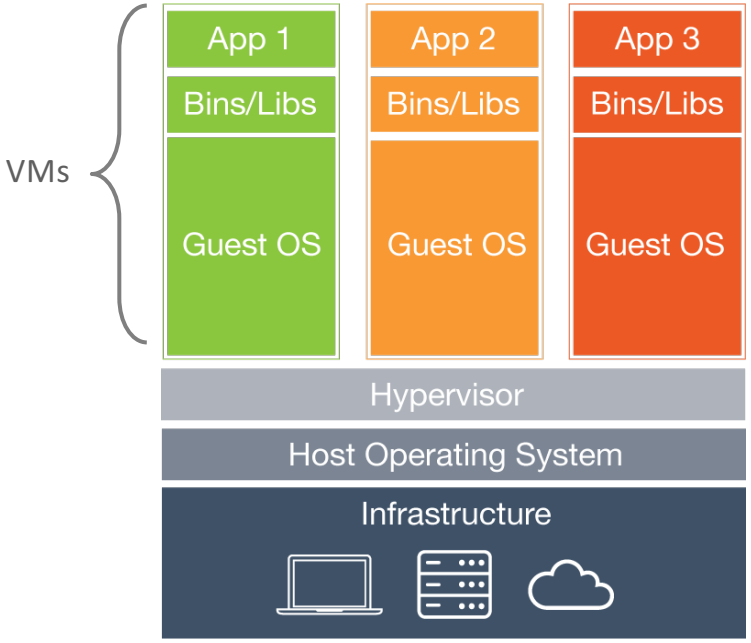
SOURCE: THE EVOLUTION OF THE MODERN SOFTWARE SUPPLY CHAIN, DOCKER SURVEY 2016

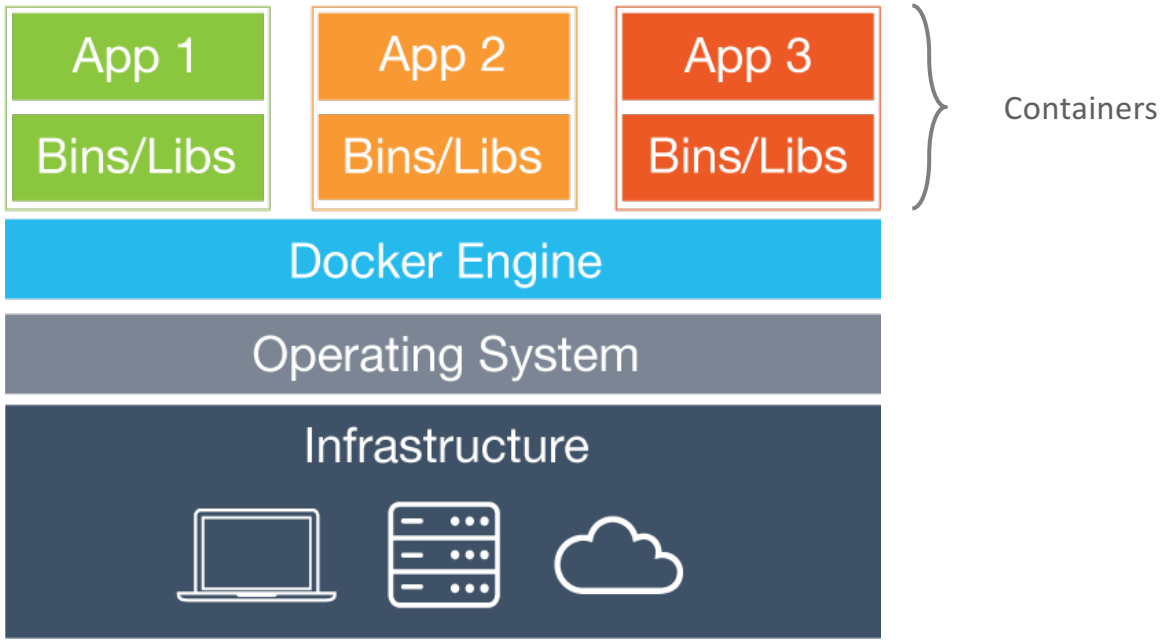
- Developer productivity a top use case today
- Building out CI/CD pipelines
  - Consistent container image moves through pipeline
  - Preventing “it worked in dev” syndrome
- Application modernization and portability are also key adoption drivers (Prem <-> cloud)

ORACLE

Basic Architecture and Nomenclature

Virtual Machines vs. Containers





Virtual Machines

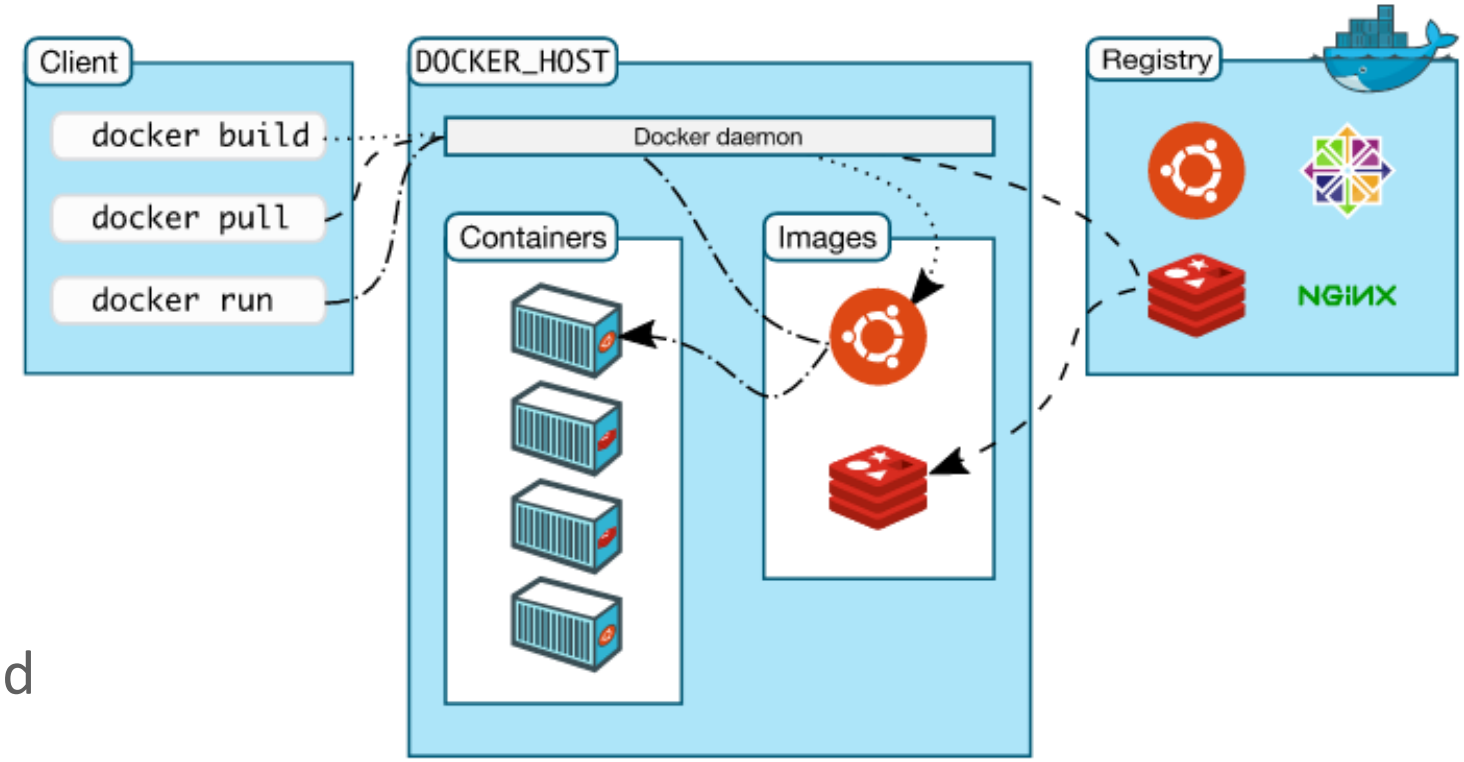
- Each virtual machine (VM) includes the app, the necessary binaries and libraries and an **entire guest operating system**

Containers

- Containers include the app & all of its dependencies, but **share the kernel** with other containers.
- Run as an isolated process in userspace on the host OS
- Not tied to any specific infrastructure – containers run on any computer, infrastructure and cloud.

Docker Architecture

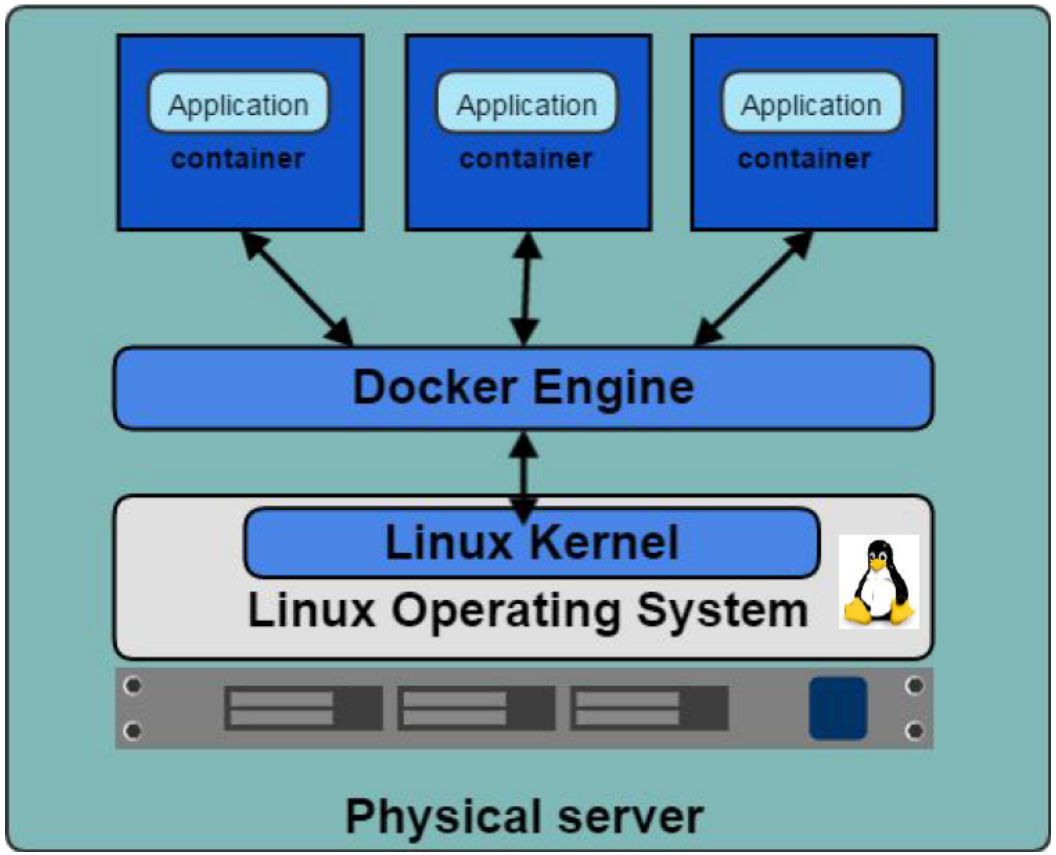
- Docker client – Command Line Interface (CLI) for interfacing with the Docker
- Dockerfile – Text file of Docker instructions used to assemble a Docker Image
- Image – Hierarchies of files built from a Dockerfile, the file used as input to the docker build command
- Container – Running instance of an Image using the docker run command
- Registry – Image repository



Source: Docker docs and https://docs.docker.com/glossary/

Docker Engine

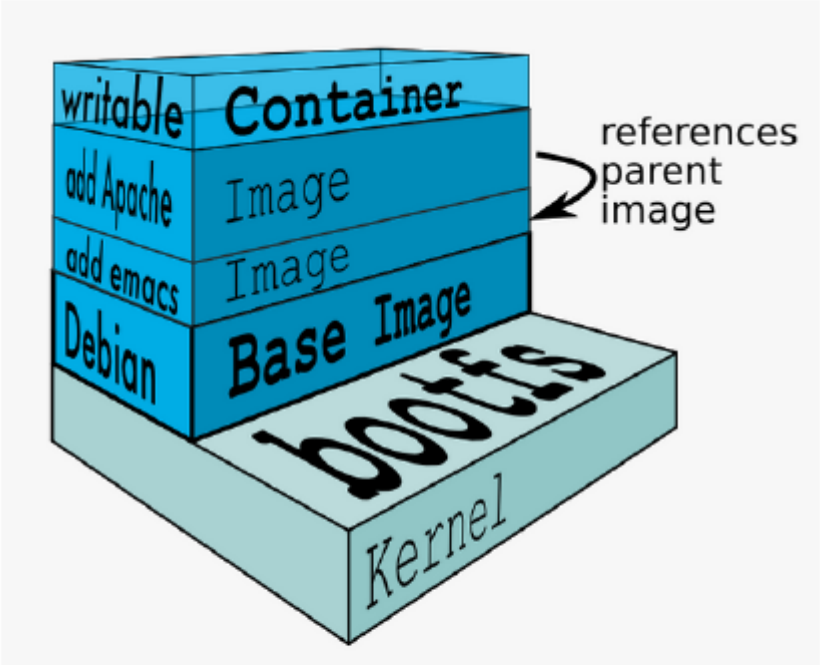
- Container execution and admin
- Uses Linux Kernel namespaces and control groups
- Namespaces provide for isolated workspace



Source: Docker docs and https://docs.docker.com/glossary/

Docker Images

- An image is a collection of files and some meta data
- Images are comprised of multiple layers, multiple layers referencing/based on another image (Union File System)
- Each image contains software you want to run
- Every image contains a base layer
- Layers are read only



Source: Docker docs and https://docs.docker.com/glossary/

Dockerfile – Text file (recipe) used to create Docker images

Example Hello World Dockerfile

```
FROM nginx:1.10.1-alpine
Add index.html /usr/share/nginx/html/index.html
# Override the nginx start from the base container
COPY start.sh /start.sh
RUN chmod +x /start.sh
ENTRYPOINT ["/start.sh"]
```

Source: https://github.com/scottsbaldwin/docker-hello-world/blob/master/Dockerfile

Docker build image CLI example

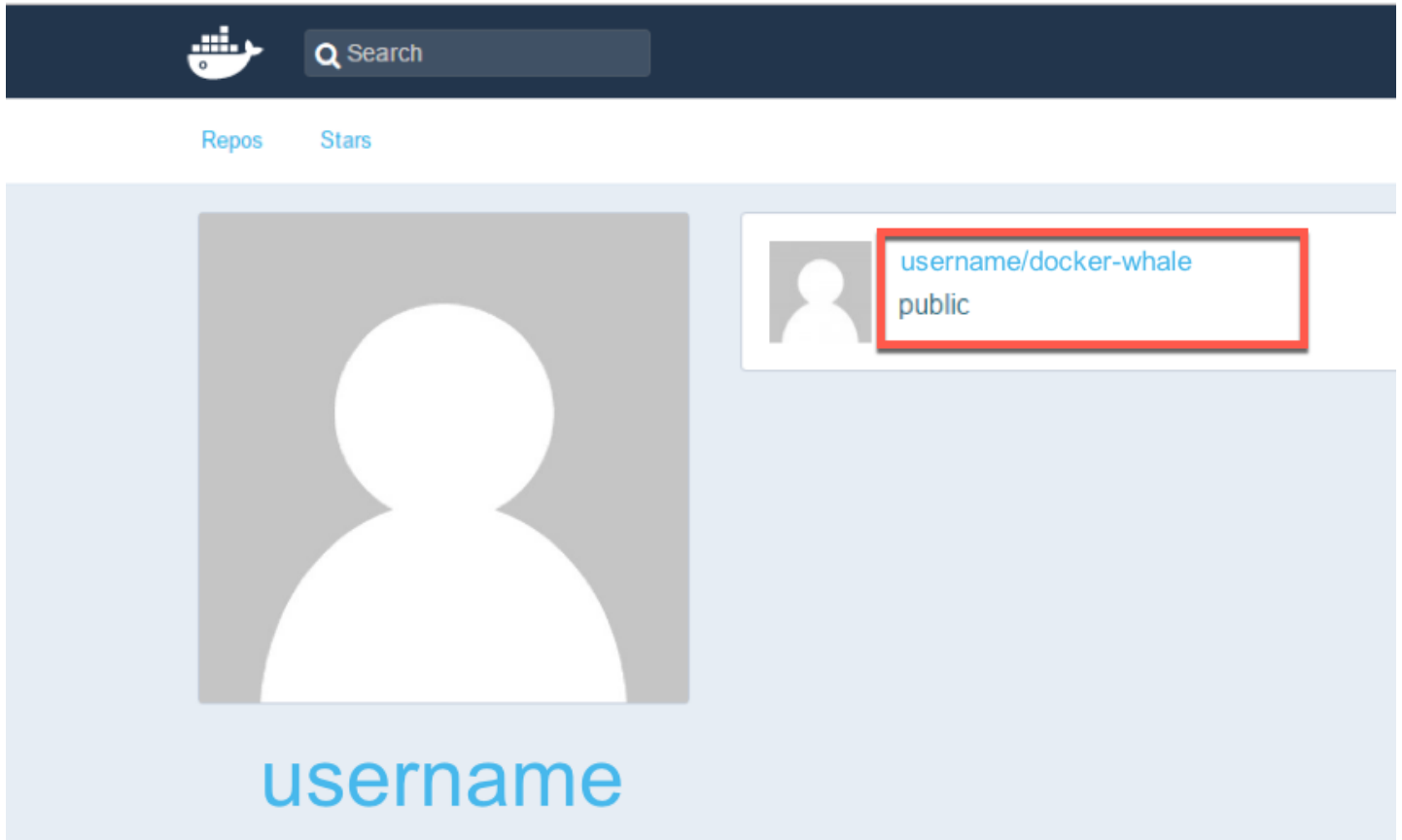
```
$ docker build -t helloworld:1.0 .
```

NOTE: The "." references Dockerfile in local directory

Docker Hub

- Docker Inc.
  - Repository
  - public and private images

- Enables images to be shared and moved off the laptop



- Example usage:
  - $ docker tag docker-whale:latest username/docker-whale:latest
  - $ docker push username/docker-whale:latest
  - $ docker pull username/docker-whale:latest

Docker CLI – Common / useful commands

- docker build : build docker image from Dockerfile
- docker run : run docker image
- docker logs : show log data for a running or stopped container
- docker ps : list running docker containers (analogous to ps)
- docker ps –a : list all containers including not running
- docker images : list all images on the local volume
- docker rm : remove/delete a container | docker rmi : remove/delete an image
- docker tag : name a docker image
- docker login : login to registry
- docker push/pull : push or pull volumes to/from Docker Registries
- docker inspect : return container run time configuration parameter metadata

See the docs here: https://docs.docker.com/edge/engine/reference/commandline/docker/

ORACLE

Docker Run

Pulls the image and runs it as a container

- Examples:
  - Simple:

```
$ docker run hello-world
```

  - Complex:

```
$ docker run -d  --restart=always  -p=443:5000/tcp
-e="REGISTRY_HTTP_TLS_CERTIFICATE=/certs/domain.crt"
-e="REGISTRY_HTTP_TLS_KEY=/certs/registry.example.com.key"
-e="REGISTRY_AUTH=htpasswd"
-e="REGISTRY_AUTH_HTPASSWD_PATH=/auth/htpasswd"
-e="REGISTRY_AUTH_HTPASSWD_REALM=Our Test Registry"
-v=/home/opc/certs:/certs   -v=/home/opc/auth:/auth
-v=/home/opc/registry:/var/lib/registry  "registry:2"
```

Docker Compose

- Docker Compose
  - Docker Tool for defining and running multi-container Docker applications
  - Reference file defined in YAML
    - docker-compose.yml

- $ docker-compose up -d

```yaml
version: '2'

services:
   db:
     image: mysql:5.7
     volumes:
       - db_data:/var/lib/mysql
     restart: always
     environment:
       MYSQL_ROOT_PASSWORD: wordpress
       MYSQL_DATABASE: wordpress
       MYSQL_USER: wordpress
       MYSQL_PASSWORD: wordpress

   wordpress:
     depends_on:
       - db
     image: wordpress:latest
     ports:
       - "80:80"
     restart: always
     volumes:
       - /var/www/html:/var/www/html:rw
     environment:
       WORDPRESS_DB_HOST: db:3306
       WORDPRESS_DB_PASSWORD: wordpress
volumes:
     db_data:
```

Why Docker is Hot

Why Docker is Hot – Its simple, Devs love it

- Dev/Test of Legacy Apps
- New App Dev (including parts of legacy apps)
- Code Agility, CI/CD Pipeline, DevOps
- Adoption of Open Source
- Microservices & Cloud Native Apps

ORACLE

Why Containers?

Developers care because:

- Quickly create ready-to-run packaged applications, low cost deployment and replay
- Automate testing, integration, packaging
- Reduce / eliminate platform compatibility issues (“It works in dev!”)
- Support next gen applications (microservices)

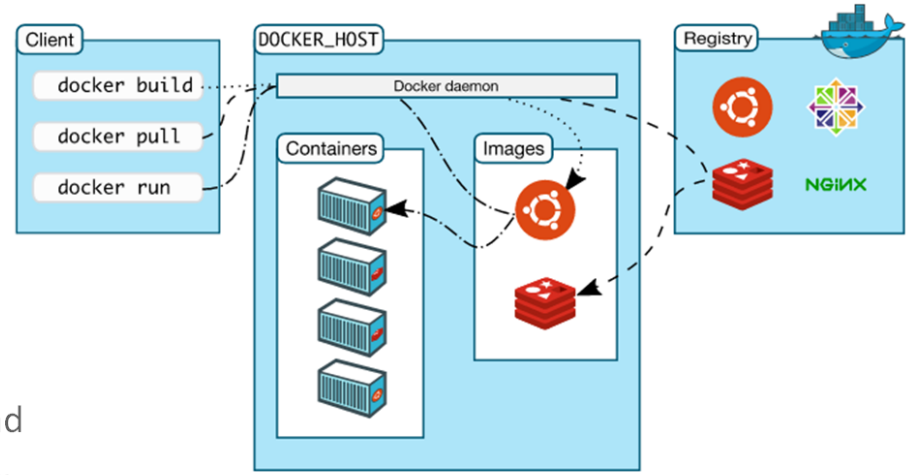
IT cares because:

- Improve **speed** and frequency of releases, reliability of deployments
- Makes app lifecycle efficient, consistent and repeatable – configure once, run many times
- Eliminate environment inconsistencies between development, test, production
- Improve production application resiliency and scale out / in on demand

Containers are Portable, but How about Advanced Functions

Core Docker Architecture

- Docker client – Command Line Interface (CLI) for interfacing with the Docker
- Dockerfile – Text file of Docker instructions used to assemble a Docker Image
- Image – Hierarchies of files built from a Dockerfile, the file used as input to the docker build command
- Container – Running instance of an Image using the docker run command
- Registry – Image repository



Advanced Functions

- Orchestration, Monitoring, Operations, Service Discovery
- Docker Environment Provisioning

Fragmented Market Solutions

- Kubernetes
- Swarm, Docker Data Center, Docker Cloud
- Consul, ETCD, Docker Networking
- etc

Oracle Cloud Infrastructure and Docker

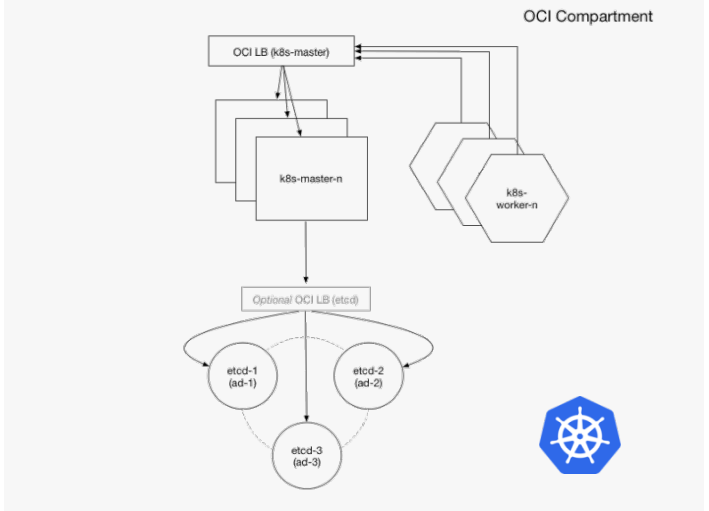
Roll Your Own, Pre-Built Installer, Container Service Classic and Managed Kubernetes Service

OCI

DIY Container Management

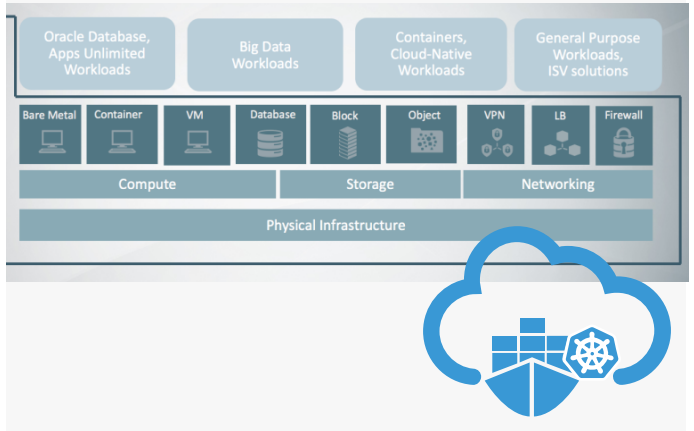
Quickstart Experience

OSS Terraform Installer on GitHub



Self Managed Kubernetes Service

OCI Container Engine for Kubernetes



Enterprise Class Managed Kubernetes Service

IaaS

CaaS

Additional Resources

Resource	Location
Entry Level Hands-on Lab	https://github.com/oracle/cloud-native-devops-workshop/tree/master/containers/docker001
Oracle Container Cloud Service	https://cloud.oracle.com/en_US/container
Official Image Registries	Oracle Images on the Docker Store Oracle Container Registry

Questions?