Introduction to Docker

Krishantha Dinesh Msc, MIEEE, MBCS
Software Architect

www.krishantha.com

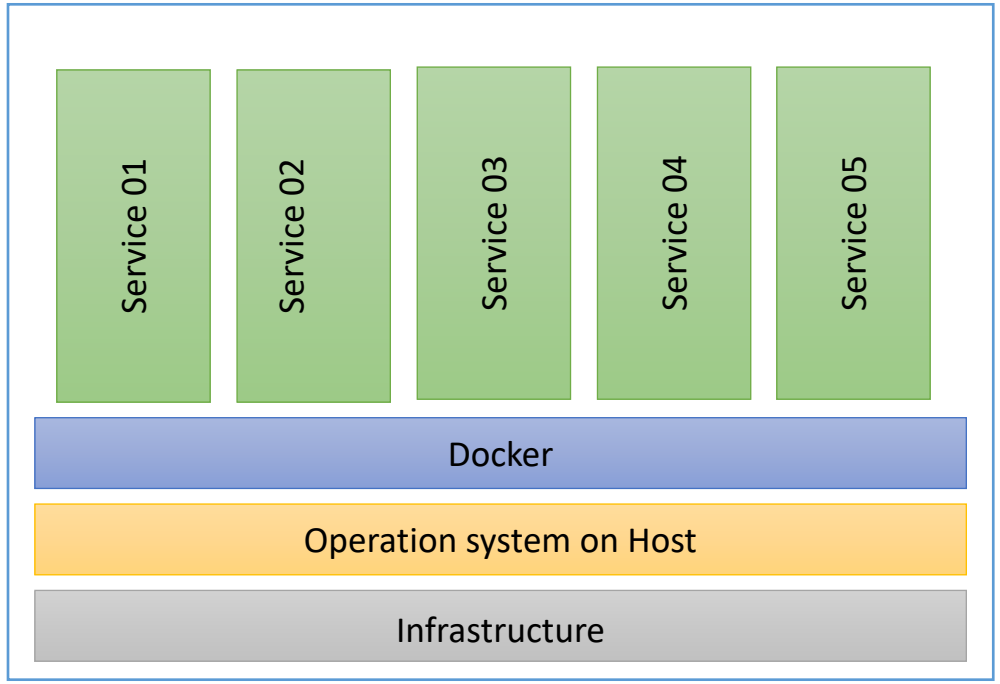
www.youtube.com/krish

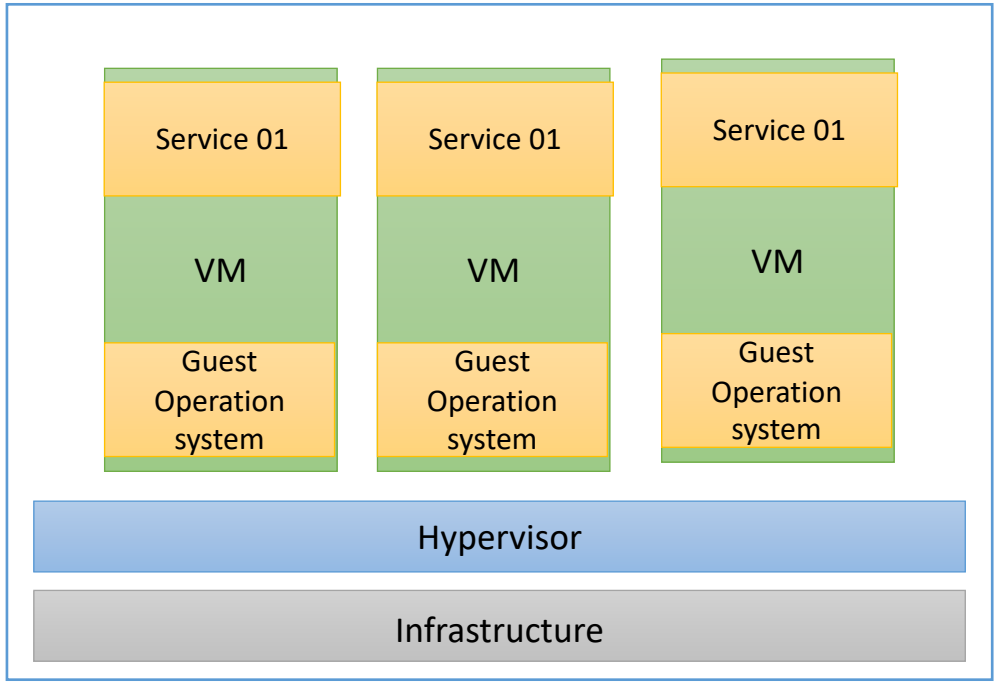
@krishantha

Why Containers

- One service per host concept
- Don’t need to maintain multiple configurations
- No configuration or security or any other conflicts
- Faster born time
- Easy of maintain
- No need to allocate memories for unwanted services or features

Containers vs VM





* http://www.krishantha.com * https://www.youtube.com/krish * https://www.linkedin.com/in/krish-din/

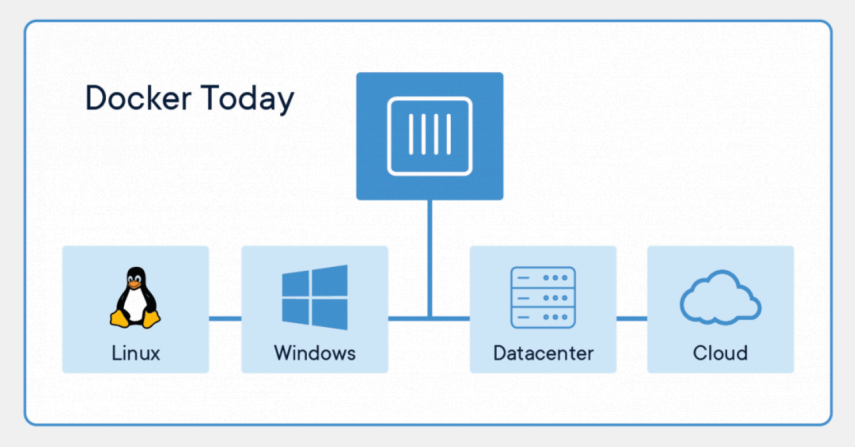
Docker

- It's a Name… not a technology
- Server vs VM vs docker
- Once docker per processor / core ? Myth
- Docker swarm and scalability (after 1.12)
- Docker + socketplane = 1.12+ docker swarm networking

* http://www.krishantha.com * https://www.youtube.com/krish * https://www.linkedin.com/in/krish-din/

Docker today

- Docker container technology was launched in 2013 as an open source Docker Engine.
- Docker's technology is unique because it focuses on the requirements of developers and systems operators to separate application dependencies from infrastructure.
- Success in the Linux world drove a partnership with Microsoft that brought Docker containers and its functionality to Windows Server.



* http://www.krishantha.com * https://www.youtube.com/krish * https://www.linkedin.com/in/krish-din/

What is Docker

- Docker is a tool
- It is designed to make it easier to create, deploy, and run applications by using containers.
- Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies, and ship it all out as one package.
- It can assured that the application will run on any other environment regardless of any customized settings that machine might have that could differ from the machine used for writing and testing the code.

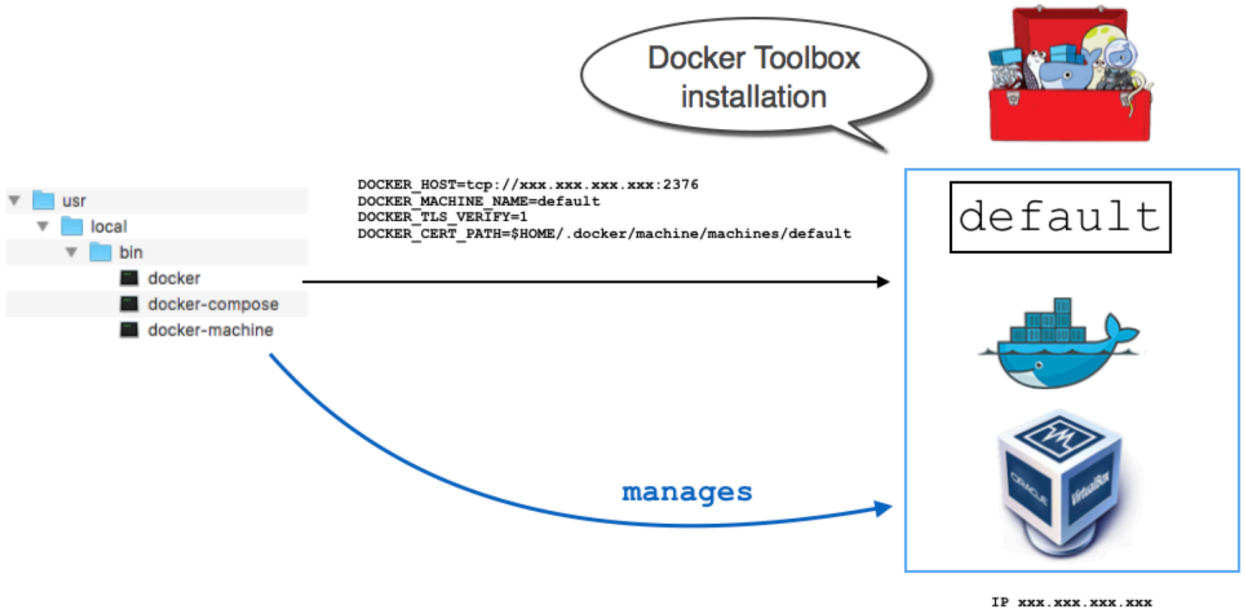
Docker to whom

- It is designed to benefit both developers and system administrators.
- It is now a part of many DevOps (developers + operations).
- For developers, it means that they can focus on writing code without worrying about the system that it will ultimately be running on. It also allows them to get a head start by using one of thousands of programs already designed to run in a Docker container as a part of their application.
- For operations staff, Docker gives flexibility and potentially reduces the number of systems needed because of its small footprint and lower overhead.

Installtion

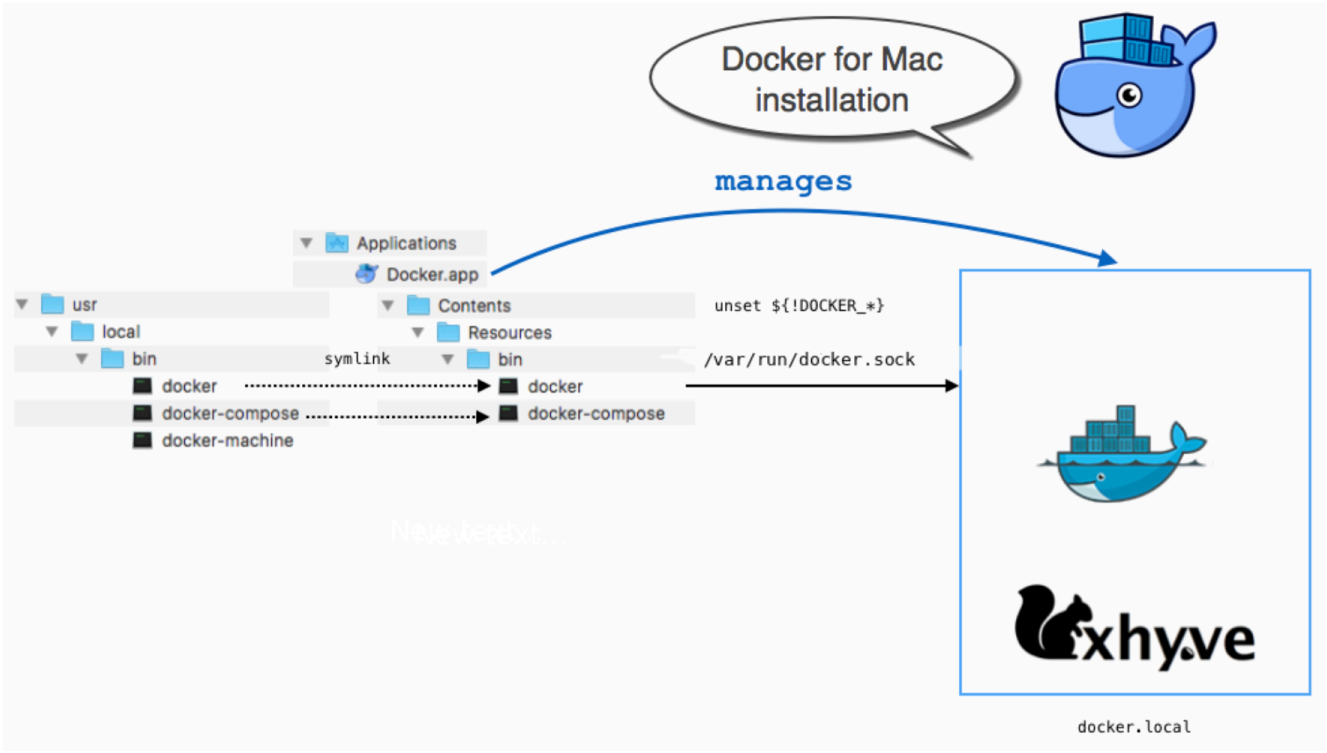
- Installation method change time to time. Especially for windows and Mac.
- For example docker for windows -> docker toolbox -> docker desktop
- So please refer current official documentation relevant to your operating system.

Docker toolbox (old)



* http://www.krishantha.com * https://www.youtube.com/krish * https://www.linkedin.com/in/krish-din/

Docker desktop



* http://www.krishantha.com
* https://www.youtube.com/krish
* https://www.linkedin.com/in/krish-din/

version

- If your installation worked fine then following or similar output should come.

```
docker version
```

```
Client: Docker Engine - Community
 Version:           18.09.0
 API version:       1.39
 Go version:        go1.10.4
 Git commit:        4d60db4
 Built:             Wed Nov  7 00:47:43 2018
 OS/Arch:           darwin/amd64
 Experimental:      false

Server: Docker Engine - Community
 Engine:
  Version:          18.09.0
  API version:      1.39 (minimum version 1.12)
  Go version:       go1.10.4
  Git commit:       4d60db4
  Built:            Wed Nov  7 00:55:00 2018
  OS/Arch:          linux/amd64
  Experimental:     true
```

info

- Below command will return everything about environment.

`Docker info`

```
Krish$: docker info
Containers: 1
 Running: 1
 Paused: 0
 Stopped: 0
Images: 7
Server Version: 18.09.0
Storage Driver: overlay2
 Backing Filesystem: extfs
 Supports d_type: true
 Native Overlay Diff: true
Logging Driver: json-file
Cgroup Driver: cgroupfs
Plugins:
 Volume: local
 Network: bridge host ipvlan macvlan null overlay
 Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
Swarm: inactive
Runtimes: runc
Default Runtime: runc
Init Binary: docker-init
containerd version: 468a545b9edcd5932818eb9de8e72413e616e86e
runc version: 69663f0bd4b60df09991c08812a60108003fa340
init version: fec3683
Security Options:
 seccomp
  Profile: default
Kernel Version: 4.9.125-linuxkit
Operating System: Docker for Mac
OSType: linux
Architecture: x86_64
CPUs: 4
Total Memory: 1.952GiB
Name: linuxkit-025000000001
```

Test docker environment

- We can use usual hello-world scenario to test docker

```
Krish$: docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:6540fc08ee6e6b7b63468dc3317e3303aae178cb8a45ed3123180328bcc1d20f
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

List running containers

```
docker ps
```

To show all containers (default shows just running)

```
docker ps –a
```

Name, shorthand	Default	Description
`--all , -a`		Show all containers (default shows just running)
`--filter , -f`		Filter output based on conditions provided
`--format`		Pretty-print containers using a Go template
`--last , -n`	`-1`	Show n last created containers (includes all states)
`--latest , -l`		Show the latest created container (includes all states)
`--no-trunc`		Don’t truncate output
`--quiet , -q`		Only display numeric IDs
`--size , -s`		Display total file sizes

* http://www.krishantha.com * https://www.youtube.com/krish * https://www.linkedin.com/in/krish-din/

Docker images vs containers

- Docker image is stopped container and docker container is running image ☺

`docker images`

```
Krish$: docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
employee-service        latest              d9a6b3df9ff0        7 weeks ago         681MB
ubuntu                  latest              7698f282e524        2 months ago        69.9MB
hello-world             latest              fce289e99eb9        6 months ago        1.84kB
vulnerables/web-dvwa    latest              ab0d83586b6e        9 months ago        712MB
java                    8                   d23bdf5b1b1b        2 years ago         643MB
Krish$:
```

How it works

```
Docker run hello-world
```

1. Check on local store

2. Go to docker hub if results of 1 is empty

How it works cont

```
Docker run ubuntu
```

1. Check on local store - NO

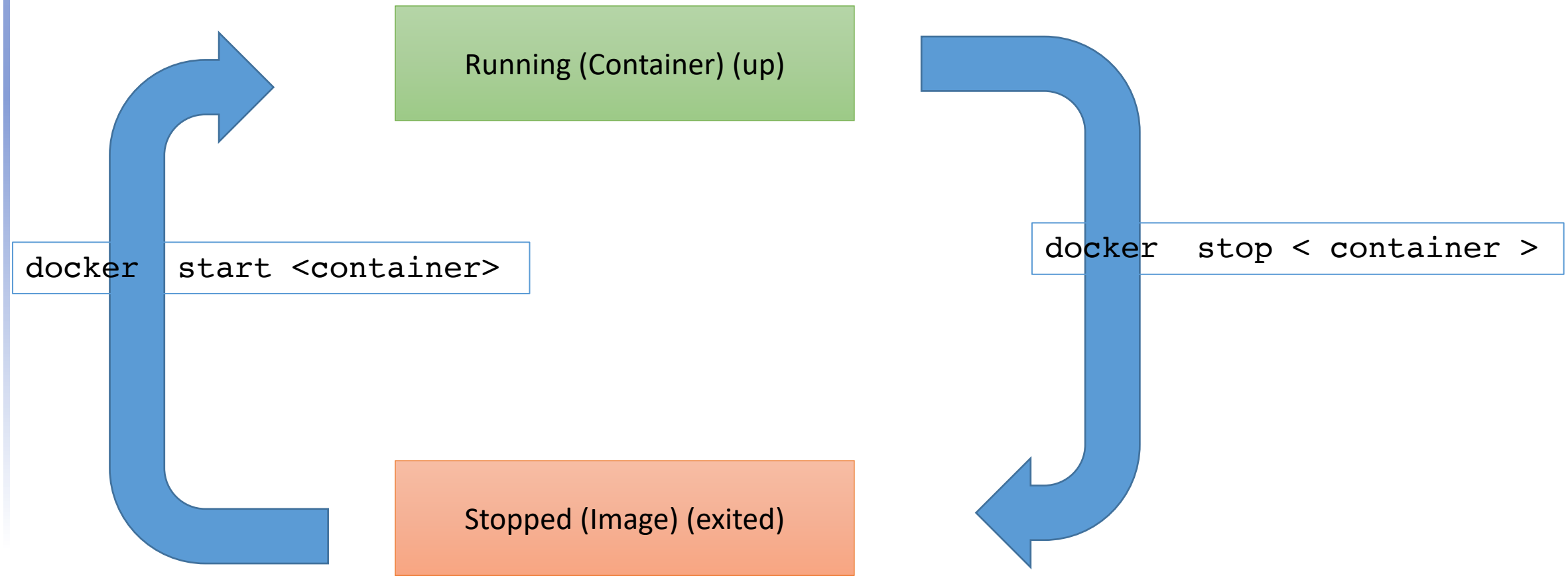
2. Go to docker hub
Download and store local

How it works cont

```
Docker run ubuntu
```

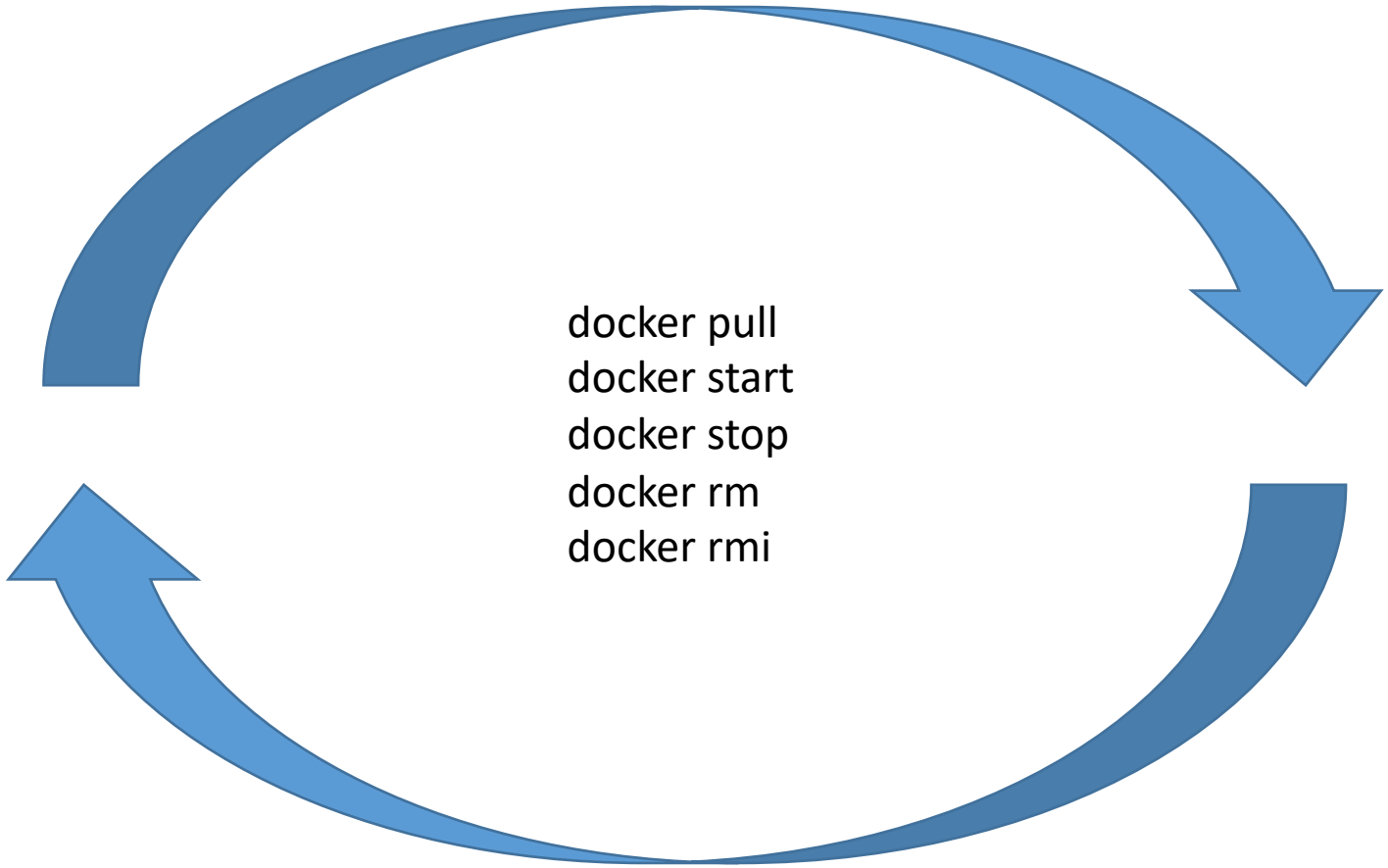
```
Krish$: docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
employee-service        latest              d9a6b3df9ff0        7 weeks ago         681MB
hello-world             latest              fce289e99eb9        6 months ago        1.84kB
vulnerables/web-dvwa    latest              ab0d83586b6e        9 months ago        712MB
java                    8                   d23bdf5b1b1b        2 years ago         643MB
Krish$:
Krish$:
Krish$: docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
5b7339215d1d: Pull complete
14ca88e9f672: Pull complete
a31c3b1caad4: Pull complete
b054a26005b7: Pull complete
Digest: sha256:9b1702dcfe32c873a770a32cfd306dd7fc1c4fd134adfb783db68defc8894b3c
Status: Downloaded newer image for ubuntu:latest
Krish$:
Krish$: docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
ubuntu                  latest              4c108a37151f        4 weeks ago         64.2MB
employee-service        latest              d9a6b3df9ff0        7 weeks ago         681MB
hello-world             latest              fce289e99eb9        6 months ago        1.84kB
vulnerables/web-dvwa    latest              ab0d83586b6e        9 months ago        712MB
java                    8                   d23bdf5b1b1b        2 years ago         643MB
Krish$:
```

Container life cycle



* http://www.krishantha.com
* https://www.youtube.com/krish
* https://www.linkedin.com/in/krish-din/

Full life cycle



* http://www.krishantha.com * https://www.youtube.com/krish * https://www.linkedin.com/in/krish-din/

Dockerfile

- Docker file use to write the instruction to build new docker image

Docker can build images automatically by reading the instructions from a Dockerfile. A Dockerfile 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.

Create own docker image

- Use case
  - Get Apache webserver docker
  - Add own custom HTML file
  - Create new image
  - Run newly created image

Create HTML page

```
Krish$: cd ~/sliit/docker/
Krish$: vim index.html
Krish$:
```

```
<html>
<body>
<h1> hello from CodeLabs.. this is run from Docker </h1>

</body>

</html>
~
```

Create Dockerfile

```
Krish$: vim Dockerfile
Krish$:
```

```
From httpd:2.4
COPY index.html /usr/local/apache2/htdocs
```

`From httpd:2.4`. → take base image of httpd version 2.4

`COPY index.html /usr/local/apache2/htdocs` → copy `index.html` from local file system to `/usr/local/apache2/htdocs` of docker

Build new Docker

```
Krish$: docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
ubuntu                  latest              4c108a37151f        4 weeks ago         64.2MB
employee-service        latest              d9a6b3df9ff0        7 weeks ago         681MB
vulnerables/web-dvwa    latest              ab0d83586b6e        9 months ago        712MB
java                    8                   d23bdf5b1b1b        2 years ago         643MB
Krish$:
Krish$:
Krish$: docker build -t codelabs-httpd .
Sending build context to Docker daemon  3.072kB
Step 1/2 : From httpd:2.4
2.4: Pulling from library/httpd
f5d23c7fed46: Pull complete
b083c5fd185b: Pull complete
bf5100a89e78: Pull complete
98f47fcaa52f: Pull complete
622a9dd8cfed: Pull complete
Digest: sha256:dc4c86bc90593c6e4c5b06872a7a363fc7d4eec99c5d6bfac881f7371adcb2c4
Status: Downloaded newer image for httpd:2.4
 ---> ee39f68eb241
Step 2/2 : COPY index.html /usr/local/apache2/htdocs
 ---> 5fb495919180
Successfully built 5fb495919180
Successfully tagged codelabs-httpd:latest
Krish$:
Krish$:
Krish$: docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
codelabs-httpd          latest              5fb495919180        13 seconds ago      154MB
httpd                   2.4                 ee39f68eb241        8 days ago          154MB
ubuntu                  latest              4c108a37151f        4 weeks ago         64.2MB
employee-service        latest              d9a6b3df9ff0        7 weeks ago         681MB
vulnerables/web-dvwa    latest              ab0d83586b6e        9 months ago        712MB
java                    8                   d23bdf5b1b1b        2 years ago         643MB
Krish$:
```

pull httpd from dockerhub

build new image

* http://www.krishantha.com
* https://www.youtube.com/krish
* https://www.linkedin.com/in/krish-din/

Run new docker

```
Krish$: docker run -dit -p 8191:80  --name web codelabs-httpd
3daa9eeba79a782dc14aafa607be8aae4eb399e572e7be10cd4b44b897047cd6
Krish$:
```

-d → detached

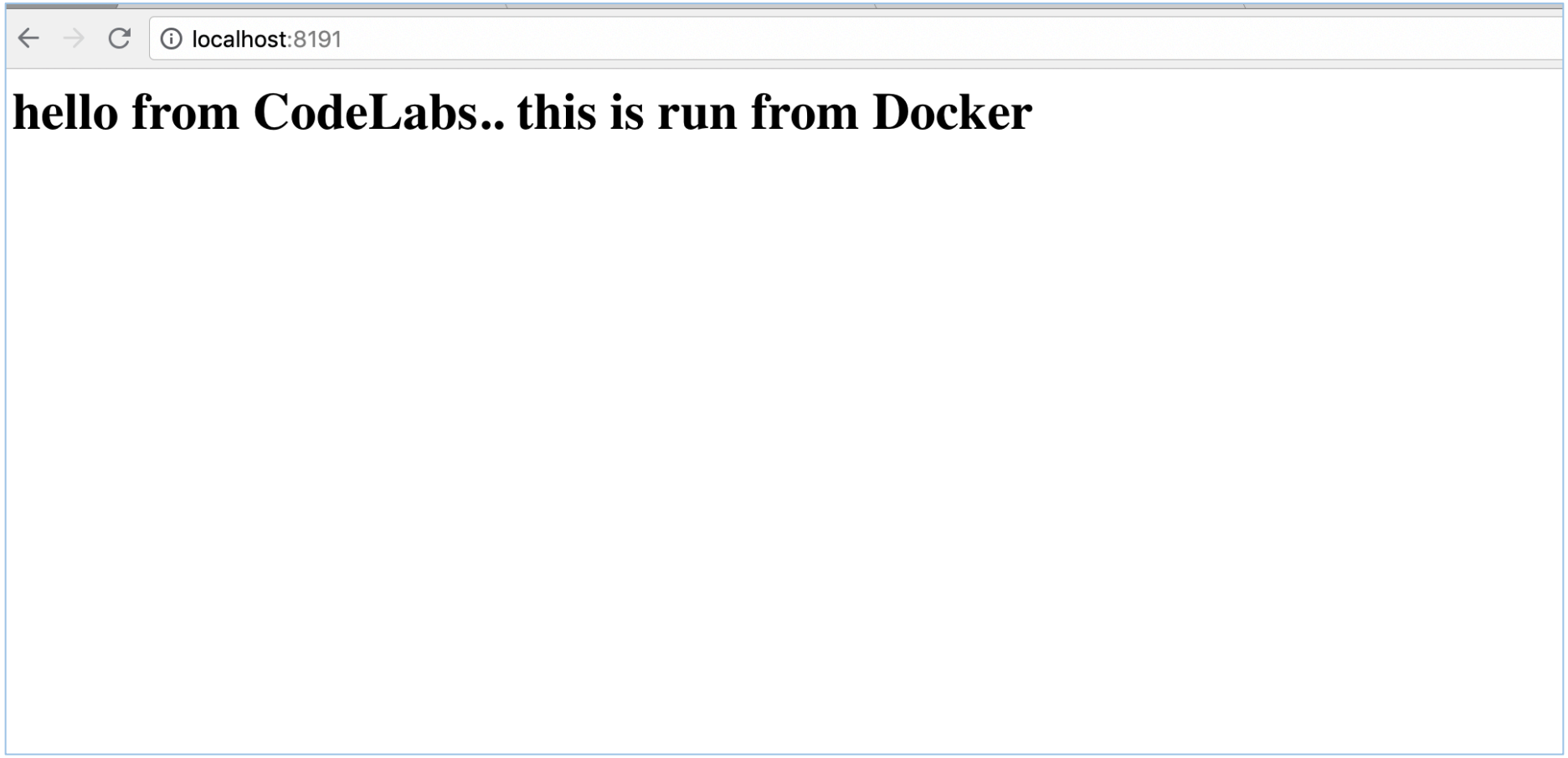
-I → Keep STDIN open even if not attached

-t → For interactive processes (like a shell), you must use -i -t together in order to allocate a tty for the container process. -i -t is often written -it.

--name → assing name for the running container

-p 8191:80 → all traffic comes to host on port 8191 request to forward to port 80 of docker

Browse new docker



* http://www.krishantha.com
* https://www.youtube.com/krish
* https://www.linkedin.com/in/krish-din/